クラウド提案ブートキャンプ - **課題**

目次

1. 背景

弊社は日本全国に支店を持つ中規模の銀行です。現在、弊社では顧客体験の向上を目的とし、お客様が自身の資産状況を可視化できる Web アプリケーションを開発しています。

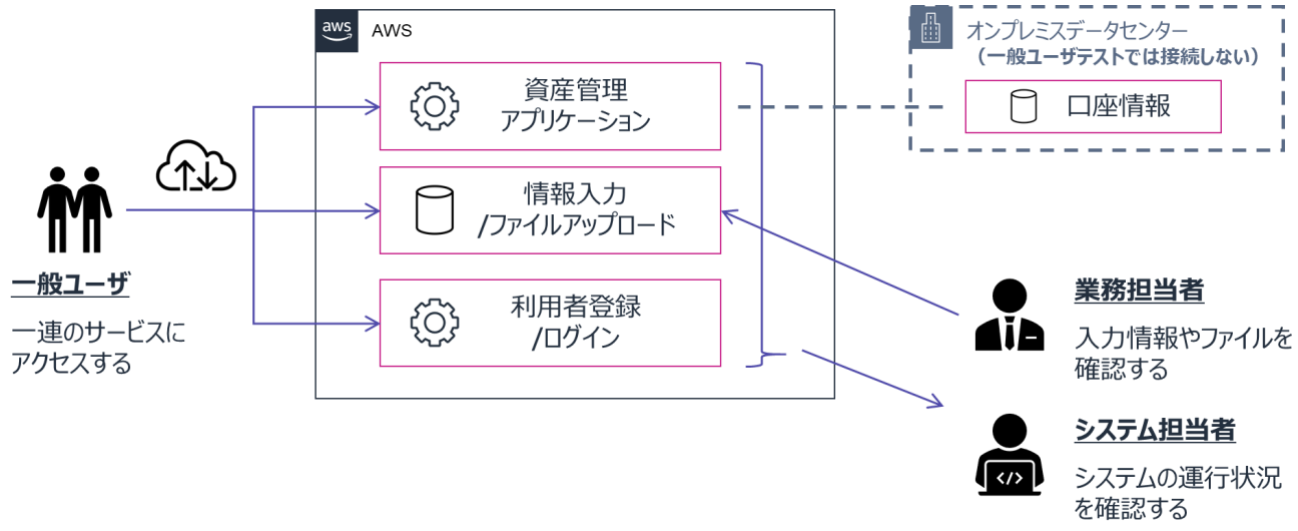
このアプリケーションは弊社内の開発環境で、開発および検証が進められてきました。当初、利用するクラウドプラットフォームが確定していなかったことから、どのような環境でも動作するよう、アプリケーションはコンテナ上で動作するように作られています。運用・監視や認証といった周辺機能を除くアプリケーションの基本機能の開発が完了し、社内開発環境で一定のシステム品質が確保できたことから一般ユーザテストを実施することになりました。

一般ユーザテストでは、弊社の顧客から選出された約１,０００人のテストユーザがシステムを試用して使い勝手を評価します。そのため、アクセス方法やパフォーマンスについては本番相当の構成でテストを実施することが求められています。

弊社のＣＩＯはこの度、拡張性や開発速度を考慮して、一般ユーザテストをＡＷＳクラウド上で行うことを選択しました。以下の要件を読み、ＡＷＳクラウド上で一般ユーザテストを行うための環境構成を提案してください。

（補足）本アプリケーションは、将来的にオンプレミスデータセンター内に配置されている勘定系システムと接続し、希望する顧客に対して口座情報等と連動した追加サービスを提供する予定です。しかし、今回の一般ユーザテストでは、追加サービスの申し込み機能が有効に動作するところまでを検証スコープとし、勘定系システムとの接続や連携はスコープ外とします。

図１：システムの全体像



2. ステークホルダーおよび基本的な要求事項

1. 「**利用者**」：弊社の既存顧客から選出された約１０００名のテストユーザです。日本全国からのアクセスが想定され、アクセス時間帯もまちまちです。事前アンケートではインターネット経由で個人情報を登録、表示するセキュリティ面の心配の声が多く聞かれました
2. 「**システム管理者**」：弊社のシステム全体を管轄する部署で、一般ユーザテスト期間中のトラブル対応や運行状況の監視を担当します。一般ユーザテスト用の環境は可能な限り運用作業を自動化し、また障害が発生しにくい構成で実現されることを期待しています
3. 「**業務担当者**」：本サービスの企画および運営を担当します。一般ユーザテストでは定期的な利用状況の確認や、追加サービスの申請処理を本番同様に実施します。IT のスペシャリストではないため AWS 環境の操作等には抵抗感があります。

3. 利用者から見たシステムの動作

1. アプリケーションの基本動作
   (ア) Ｗｅｂブラウザを使用しインターネットを経由してアプリケーションにアクセスします
   (イ) アプリケーションには資産情報を入力・保存することができます
   (ウ) 自分が入力したデータはいつでも参照できるものとします
2. ユーザ登録およびログイン
   (ア) 初回アクセス時は、氏名やメールアドレス、生年月日などの情報と、ログインパスワードをシステムに登録します
   (イ) ２回目以降のアクセスでは、ログイン画面でメールアドレスとログインパスワードを入力し、認証されるとアプリケーションにアクセスすることができます
3. 追加サービスへの登録
   (ア) 弊社の銀行口座を持つ利用者は、追加サービスの適用を申請することができます。システムに口座番号を登録し、身分証明書画像をアップロードすることで申請が可能です
   (イ) 追加サービスの申請完了時および承認完了時、登録されたメールアドレスに対して通知メールが発行されます
   (ウ) 追加サービスが承認されると、システムに口座情報や取引情報が反映され、より高度な資産管理サービスを利用することができます

4. システム管理者から見たシステムの動作

1. システム運行状況の把握

   (ア) システム管理者はダッシュボード[i]などを通じてシステムの稼働状況を確認します

   (イ) システムに障害[ii]が発生した場合、システム管理者に通知されます

   (ウ) システムにセキュリティインシデント[iii]が発生した場合、システム管理者に通知されます

   (エ) システム管理者は障害発生時等に、収集・保管されたログを参照します。その他、利用者やシステム管理者、業務担当者がシステムにアクセスした形跡を保管します。

5. 業務担当者から見たシステムの動作

1. システム利用状況の確認

   (ア) 適宜、ユーザ登録者数や最終ログイン時刻を確認します

   (イ) クラウド利用料金が一定値を超えた場合、業務担当者に通知されます

2. 追加サービスの処理

   (ア) 利用者が追加サービス申請を行うと、業務担当者に通知されます

   (イ) 追加サービスの承認にあたり、利用者が登録した情報やアップロードしたファイルにアクセスし内容を確認します

   (ウ) 審査が正常に完了すると、当該利用者を追加サービス有効化済ユーザとして登録します

6. その他の要件

1. 利用者が登録したデータは毎日自動でバックアップしてください

   ただし、利用者の認証情報(ID/PASSWORD)に対するバックアップの考慮は不要です

2. 個人情報を扱うため、転送中および保管時の両面でデータの安全性を保つ対策を講じてください
3. ファイアウォールなど、ネットワークを保護する対策を講じてください
4. パッチ適用やソフトウェアの脆弱性診断など、サーバーやアプリケーションを保護する対策を講じてください
5. システム管理者、業務担当者向けのシステムについては、社内拠点用のグローバル IP アドレス以外からのアクセスを遮断する対策を講じてください
6. 特定要素の単一障害により一般ユーザテストが中断しないよう、構成要素を冗長化してください
7. ＡＷＳクラウド内の通信については、可能な限りインターネットを経由しない（パブリックＩＰアド

レスを使用しない）通信経路を検討してください

8. ＡＷＳクラウド内からインターネットへの通信については、URL フィルタリングによるアクセス制御をする対策を講じてください
9. 要件を実現できる方式が複数ある場合は、より簡素・安価な方式を提案してください
10. 一般ユーザテストでは潤沢な性能を必要としませんが、本番化を想定しアクセス数の増加に応じたスケールアップ／スケールアウト[iv]が可能な構成を提案してください
11. ユーザがシステムにアクセスする際の URL はカスタムドメイン(例：xxxassetstatus.com)を利用する方式を検討してください

7. 追加検討課題

余力があれば、以下についても検討、提案してください。

1. 将来的にはオンプレミスデータセンターとの接続を予定しています。これを実現するためのアーキテクチャを提案してください
   これに伴い、システム管理者、業務担当者向けのシステムに対して社内拠点以外からのアクセスを遮断する対策を講じてください
2. 大規模災害に備えて、顧客データのバックアップを遠隔地域に保管したいと考えています。マルチリージョン構成でデータの遠隔保管を実現するアーキテクチャを提案してください
3. このアプリケーションはお客様フィードバックに応じ迅速にアップデートすることを予定しています。テスト済みのアップデートをテスト環境、ステージング環境、および本番環境へ迅速に反映するアーキテクチャを提案してください
4. 個人情報や口座情報を扱うため、高度なセキュリティ監視が必要です。アプリケーションに含まれる脆弱性や脅威をモニタリングする方法を提案してください
5. DDoS 攻撃（Distributed Denial of Service Attack）に備えて、CDN の利用、同一 IP アドレスからのリクエストのレート制限、オリジンサーバーの保護等の対策をしたいと考えています。これらを AWS 上で実現するアーキテクチャを提案してください
   なお、CDN を経由せずにオリジンサーバに対して直接アクセスすることを防止する仕組みを検討してください
6. 要件のうち、ここを変えれば／無くせば、より良い実装が可能である、というポイントがあれば、該当する要件と想定される要件見直し後の実装について提案してください

8. 補足事項

1. 上記要件にない事項については、適宜前提をおいて提案してください。ただし、提案の中でどのような前提を置いたのか説明するようにしてください
2. システムの実装に必要なアプリケーションの設計・開発は、貴社からの提案を踏まえて弊社が別途用意する開発メンバーが実施します。必要なアプリケーション機能があれば前提として定義いただく

とともに、提案におけるロジック等の詳細な検討は不要とします
例：「XXX」という機能を持ったアプリケーションを EC2 に配置する

3. 精緻なコストは一般ユーザテスト実施後に改めて見積もる予定であるため、現段階においては、システム全体で発生するコストの算出は不要です
   ただし、アーキテクチャ選定においては各案のコストを考慮した上で検討を進めるようにしてください
4. ファイアウォール設定、権限設定ほか、パラメータレベルの検討は不要とします。ただし可能な範囲でどのような意図、方針をもって、アーキテクチャを設計したか説明するようにしてください

9. 検討のためのヒント

3－1：アプリケーションの基本動作

① 利用者から見たシステム構成を整理しましょう。利用者はインターネットから接続するようです。まず A）インターネットアクセスを受け付ける入口を設置し、ここにファイアウォールや認証機能などの関門を設置します。B）関門を通過した利用者はアプリケーションが設置されているコンピューティングリソースにアクセスし、情報の閲覧や登録を行うでしょう。C）実際に情報やデータが配置されているストレージは利用者に晒さず、アプリケーションを通じてアクセスするよう層を分けると良いでしょう。このように、必要な層を設定していきましょう。

② A）の部分です。ユーザはインターネットから接続するようです。AWS 環境をインターネットに接続するために必要なコンポーネントは何でしょうか。また、サーバー（コンテナや EC2）が直接インターネットにさらされる構成は望ましくありません。それらの前面に設置できる AWS サービスは何でしょうか。

③ B）の部分です。アプリケーションはすでに開発済のようです。このアプリケーションをどのようなコンピューティングリソースに搭載するか考えてみましょう。コンテナを利用する場合、コンテナを実行するリソースと、コンテナを管理するためのサービスを配置する必要があります。なお、コンテナテクノロジーの理解について心配がある方は仮想マシン（EC2）に置き換えても構いません。

追加のヒント１０-１：コンテナワークロードの選択

④ C）の部分です。ユーザ登録時に利用者が入力した情報をシステムで保持するようです。この情報は氏名、メールアドレス、生年月日などのテキスト情報（＝構造化データ）であることがわかります。このようなデータを保管するのに適した AWS サービスは何でしょうか。

3－2：ユーザ登録及びログイン

① Web アプリケーションにおけるユーザ登録や認証を簡単に実現するための AWS サービスはなんでしょうか。思いつかない方は仮想マシン（EC2）上にログインアプリケーションを実装する構成に置き換えても構いません。

追加のヒント１０-２：認証の仕組み

② 利用者がシステムへログインする際の流れを確認しましょう。まず利用者は未ログインの状態でシステムにアクセスしてきます。システムは利用者が未ログイン状態であると判定すると、認証サービスに通信をリダイレクトします。さて、この判定を行う機能はシステムのどこで実装されるでしょうか。

3 − 3：追加サービスへの登録

① 追加サービスの受付、入力の画面はアプリケーションにて実装される前提としてよいでしょう。追加サービスの申請には利用者が身分証明書をアップロードする必要があるようです。①のアプリケーションを経由してアップロード先にアクセスするとして、画像ファイルを配置するのに適した AWS サービスは何でしょうか。

② 追加サービスの申請が行われると、利用者と業務担当者にメールを発送します。メールは利用者がアプリケーションの提出ボタンを押下したことを契機に発送される仕組みが想定されますが、メール送付に適した AWS サービスはなんでしょうか。

4 − 1：システムの動作状況の把握

① システムの動作状況の把握には、以下のような目的があります。すべての事象を記録、把握しようとすると開発者や運用者に過度な負担やコストを強いることになるため、システムに求められる要件や特性に応じて、代表的な監視箇所や監視項目を検討することが重要です
- システムが正常に動作していることを確認する（エラーや障害の発生に気づける）
- システムにエラーや障害が発生した原因を調査する
- システムを効率化、改善するための示唆を得る

② エラーや障害に気づく方法を考えましょう。そのためにはエラーや障害が発生しうる箇所を特定することが大切です。検討したシステム構成のうち、「ここが止まるとシステム運行に支障をきたす」場所はどこでしょうか。洗い出してみましょう。

③ ②で洗い出された各要素はどのように監視することができますか。以下のような機能が必要になるでしょう。このシステムにおける監視方法を議論してみましょう。
- サーバーが正しく稼働している（ダウンしていない）ことの確認
- CPU 使用率やストレージ使用率が溢れていないことの確認
- 上記の情報を取りまとめ、ダッシュボード化する機能
- AWS サービス全体が正しく動作していることの確認

④ システムに異常が発生した場合、システムがシステム管理者に通知し、その後システム管理者はダッシュボード等を利用してシステムの状況を確認します。通知機能は 3 - 3 -②で検討した仕組みが流用できるでしょう。さて、システム管理者はダッシュボードへどのようにアクセスするでしょうか。通信経路を確認してみましょう。

⑤ 誰もがアクセス、操作できるクラウド環境ではログの確保が欠かせません。システムのログはどのように取得するでしょうか。「アプリケーションや通信のログ」と「クラウド環境の操作や変更結果のログ」の 2 つの観点で、確保すべきログの種類と取得方法を検討しましょう。また取得したログをどこに保管すべきか合わせて考えてみましょう。

5 － 1：システム利用状況の確認

① システム利用状況の確認を行う業務担当者は IT のスペシャリストではないため、システム開発には不慣れで、AWS サービスをマジメントコンソールや CLI 等から直接操作するのはハードルが高い可能性があります。そういった場合は業務担当者向けのアプリケーションを構築すると良いでしょう[v]。この検討の中では業務担当者向けアプリケーションの具体的な機能について議論する必要はありませんが、以下のような内容について考えてみましょう。

- 業務担当者向けアプリケーションが動くコンピューティングリソース（サーバー）は？
- サーバーを設置するネットワークセグメントは？
- 業務担当者がサーバーにアクセスするための経路は？
- サーバーがデータベースやストレージにアクセスする経路や権限は？

② AWS の利用料金の予測とレポートを行うサービスは何でしょうか。そのサービスは通知機能を備えており、今回のシナリオではそのまま利用できるでしょうか。

5 － 2：追加サービスの処理

① 申請書提出時の通知は 3 - 3 -②で検討した仕組みが流用できるでしょう。

② 5-1-①で紹介したとおり、業務担当者が証明書データやユーザープールに直接アクセスする構成は推奨されません。5-1-①とおなじ形で（もしくは 5-1-①のサーバー機能の中で）追加サービスの処理ができるとよいでしょう。

6 － 1：バックアップ

① まず必要なバックアップ対象を特定しましょう。バックアップには次の 2 つの目的があります。それぞれの目的を達成するための AWS サービスは何でしょうか。

- 格納先機器の破損等からデータを守る
- オペレーションミスやプログラムバグ等による想定しない更新・削除からデータを守る

6 － 2：暗号化

① 利用者はインターネットを経由して個人情報等を登録します。通信データの傍受に備え、暗号化通信（SSL/TLS）を利用します。まずは暗号化通信が必要な経路を特定しましょう。暗号化通信を行うためにはサーバー証明書が必要です。サーバー証明書を発行する AWS サービスは何だったでしょうか。また発行されたサーバー証明書をどこに配置して暗号化通信を行いますか。

② 保管されているデータは暗号鍵を使用して暗号化します。まずは暗号化が必要なデータがどこに保管されているか特定しましょう。暗号鍵の適用および管理機能を提供する AWS サービスは何だったでしょうか。

6－3：ファイアウォール

① DDoS 攻撃や SQL インジェクション攻撃など、インターネットからの悪意ある攻撃を防ぐにはウェブアプリケーションファイアウォールが有効です。ウェブアプリケーションファイアウォールを提供する AWS サービスは何でしょうか。また、そのサービスを使ってウェブアプリケーションファイアウォールをどこに設置しますか。

② AWS には、インスタンス単位やサブネット単位で設定するファイアウォール機能があります。すべての設定を整理する必要はありませんが、どういった方針でセグメント間[vi]の通信を制御するか、方針について議論してみましょう。

6－4：脆弱性対策

① コンテナを利用する場合、イメージスキャンによりコンテナ内のソフトウェアの脆弱性を診断することができます。コンテナイメージスキャンを行える AWS サービスは何でしょうか。まだイメージスキャンはどのタイミングで実施しますか。

② EC2 を利用する場合、定期的な OS アップデートが必要です。パッチ適用のためには、各 EC2 がリポジトリと呼ばれるパッチ置き場にアクセスできる必要があります。リポジトリはどこで、EC2 はどんな経路でリポジトリにアクセスしますか。

③ 追加の対策として、EC2 の脆弱性チェックが行える AWS サービスの利用を検討しましょう。スキャン方法が何種類かあり、またそれぞれの利用条件が決まっています。AWS ドキュメントを確認してみましょう。

6－5：多重化

① サーバーやネットワーク経路の多重化を考えましょう。一つのサーバー、ネットワークがダウンした場合に代わりの経路、サーバーで業務を継続する仕組みが望まれます。一方で必要以上に多重化を行うと利用料金が増大してしまいます。どのような障害ケースを想定するか、チームで議論の上、多重化するポイントを洗い出してみてください。
   - OS やサーバー機器の単一障害
   - AWS データセンターレベルの障害
   - 地域の大規模被災

② 多重化したネットワーク・サーバは通常時「Active-Ative」「Active-stanby」「Read-Write」のような形で動作しています。どの方式を選択するか、また障害時に待機系へ処理を切り替える方法について検討してみましょう。

6－5：プライベート IP アドレスを使った通信

① AWS においてサーバーが設置される VPC 内の通信はプライベート IP アドレス[vii]を用いて行われます。しかし AWS が提供するサービスの多くは VPC 外部に存在しており、VPC 内⇔VPC 外の通信方法の検討が必要です。簡単にはインターネットゲートウェイを配置して VPC 内⇔VPC 外の通信を実現することができますが、これでは「パブリックＩＰアドレスを使用しない」という条件を満たすことができません[viii]。プライベート IP アドレスを使って通信を行うためには、VPC エンドポイントと呼ばれる通信経路を設定する必要があります。今回利用するサービスのうち、どのサービスに対して VPC エンドポイントが必要か、議論してみましょう

6－6：コスト

① コスト削減はクラウドに限らずシステム開発において重要視される指標です。今回の取り組みでは精緻な見積もりは必要ありませんが、コスト観点で構成を見直してみましょう
- 一般にコストの大部分は EC2 や RDS が占めています。これらをマネージドサービスや EC2 を使わない実装に置き換えることはできないでしょうか
- 商用ライセンスを OSS に置き換えることはできないでしょうか
- 料金を抑えるための契約プランは適用できるでしょうか
- 常時起動しておかなくてもよいインスタンスは削減余地があるでしょうか

6－7：スケールアップ・スケールアウト

① 事前に見込まれたよりもアクセスが集中したり、普段はあまり使われないシステムがある瞬間のみ利用が増大したりするケースでは、あらかじめ多くの or 強力なサーバーを構えておくよりも、利用増に比例して動的に能力を追加する方法が効率的です。検討した構成のうち、利用増に伴ってサーバーの数や性能を上下させる必要がある個所はどこか特定してみましょう。

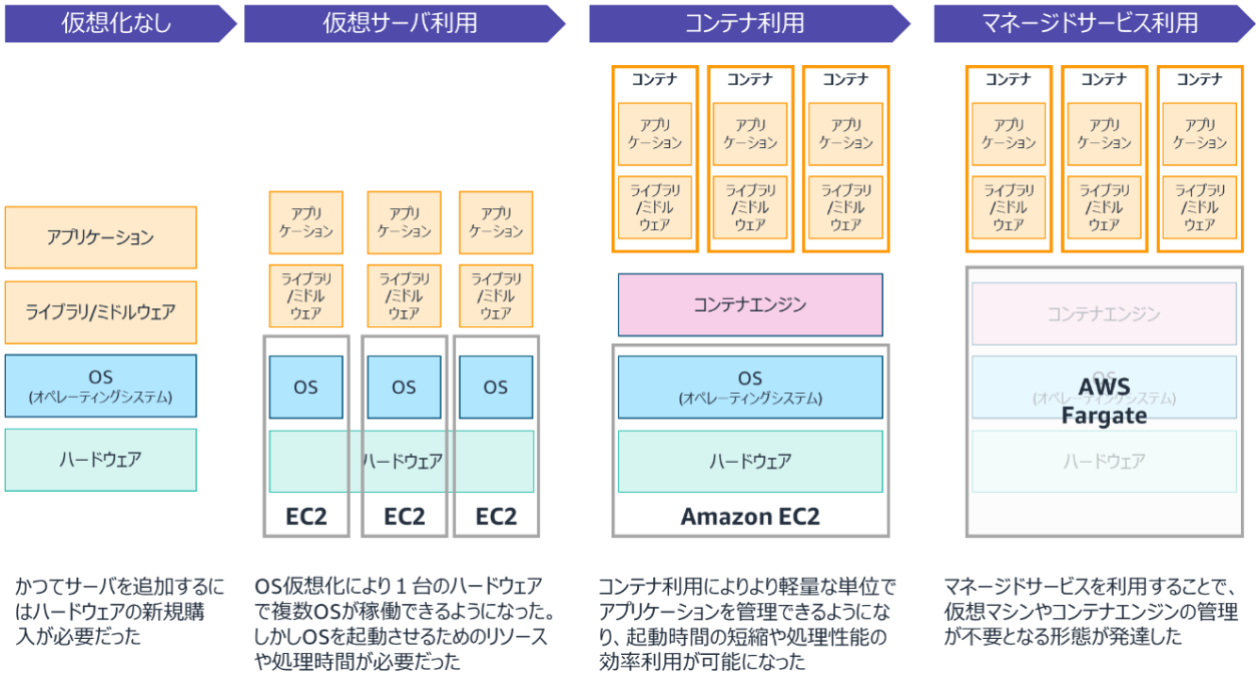
② サーバーをより強力なコンピュータに置き換えて対応するスケールアップと、サーバー台数を増やして性能を向上させるスケールアウト、どちらの戦略を採用すべきでしょうか。

10. さらに追加のヒント

※紙面の都合上、AWS サービスの呼称に略称を用い、また「AWS」や「Amazon」といった表記を省略していいます。

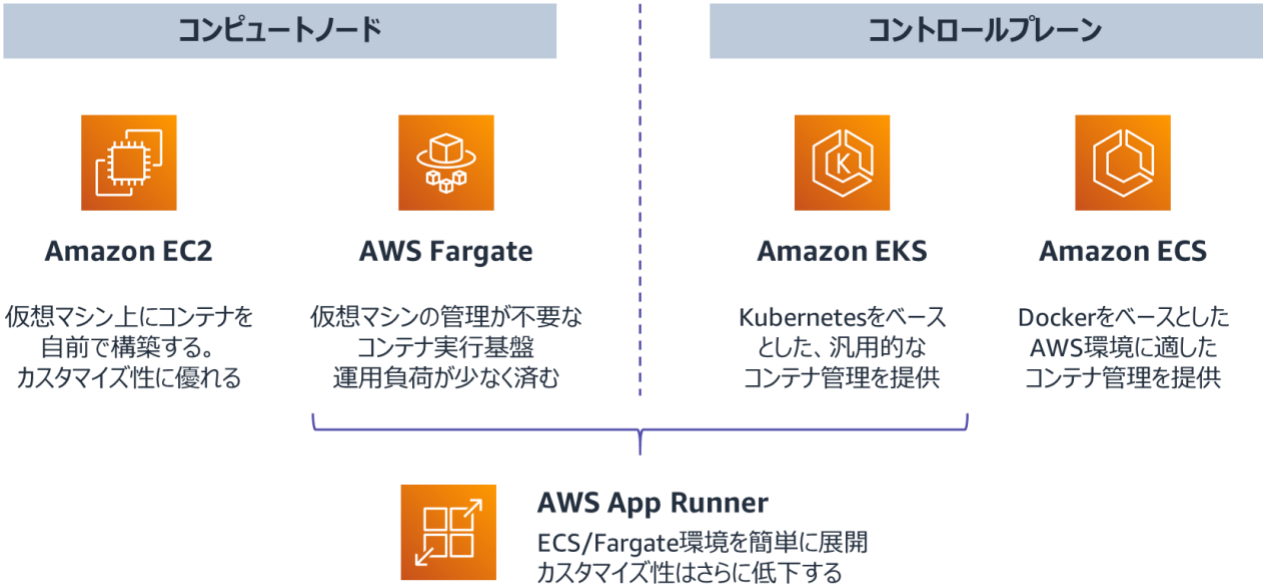
１０-１：コンテナワークロードの選択

- コンテナはコンテナエンジンを導入することで、ひとつの仮想マシンのなかに独立した環境を用意する技術のことです。AWS でコンテナを利用するには、コンテナを実行するノードと、ノードを管理するコントロールプレーンを選択します。一般に以下のようなメリットがあります
  - 必要最小限の機能単位で用意するため起動が早い（OS を起動する必要がないため）
  - コンテナエンジンがあればどこでも起動できるため、環境の移動が簡単にできる
  - ゲスト OS を必要としないため、少ないコンピューティングリソースで動作できる
- **ノード**：実際に CPU やメモリを持ち、アプリケーションが動作する環境のことです。AWS では Amazon EC2 または AWS Fargate を利用することが多い[ix]でしょう。
  - **EC2：**AWS が提供する仮想サーバーです。EC2 上にコンテナエンジンをインストールしてその上にコンテナを起動します。開発者自身が整備、運用する必要がある一方で、カスタマイズ性に優れる従来型の方式です
  - **Fargate：**AWS がコンテナエンジンごと提供するプラットフォームです。仮想サーバーの管理運用を AWS が実施するため、システム管理者が障害対応やパッチ適用等を意識する必要がなく運用コストを下げる効果が期待できます。一方で、EC2 に比べると利用料金がやや高いといったデメリットもあります



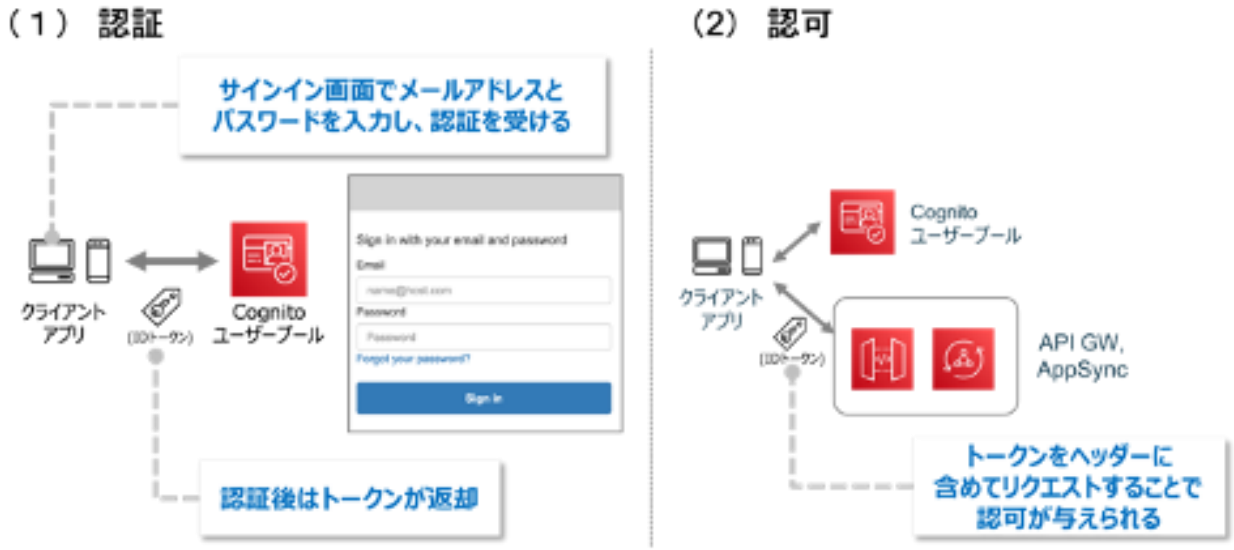
- **コントロールプレーン：**コンテナの処理内容、割り当て CPU、コンテナどうしのリンク方法といった設定などを通じて、ノードの管理を行います。AWS ではコンテナ管理に Amazon ECS と Amazon EKS の 2 種類が利用できます。
  - **ECS：**Docker をベースに AWS が開発したコンテナサービスです。AWS 環境上で運用、管理しやすいように設計、実装されています。今回はとくにこだわりがない限り ECS を利用するとよいでしょう。
  - **EKS：**コンテナ管理ツールのデファクトスタンダードである Kubernetes をベースにクラウド上で利用可能なマネージドサービスです。オンプレミスで Kubernetes を利用している場合、クラウド上でも同一の管理できができる利点があり ECS にない高度な機能も提供されています。ただし ECS に比べ割高、1 年ごとのアップデートが必須など、考慮事項を踏まえて利用を選択する必要があります。

- Amazon ECS＋AWS Fargate 構成を取る場合、アプリケーションをより簡単に展開することができる AWS AppRunner というサービスも利用できます。コンテナに関連した面倒な設定を自動で行うことができるサービスです。一方、Amazon ECS＋AWS Fargate で個別に構築するケースと比べてさらにカスタマイズ性は劣り、タグや環境変数が設定できない、セキュリティグループや WAF がアタッチできないといった制限があります。これらが問題にならない状況、または別の方法でデメリットを回避できる場合に利用すると良いでしょう。

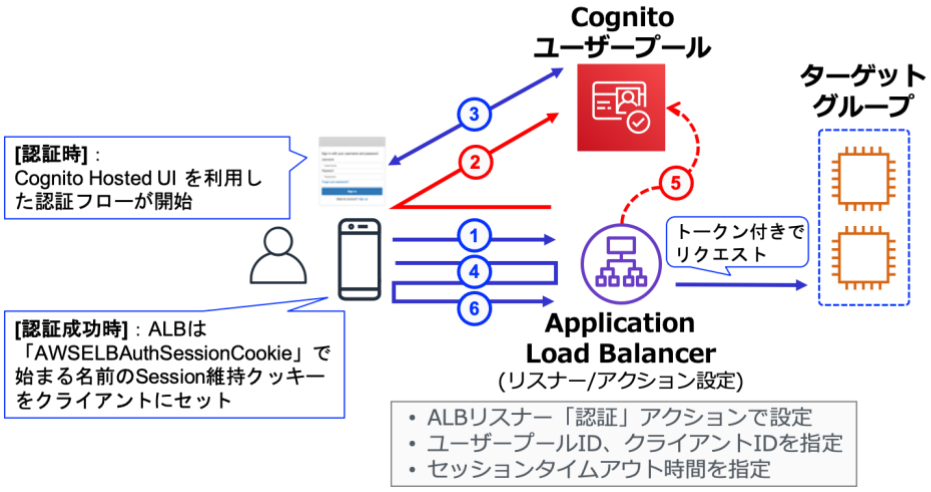


１０-２：Web アプリケーションにおけるユーザの認証・認可

- Web アプリケーションにおいて、システムに登録済みのユーザだけがアプリケーション上のリソースにアクセスできることを保証したい場合があります。具体的な実装例として、(1) サインインページを用意し、(2) ユーザにメールアドレスとパスワードの組み合わせを入力させ、(3) その組み合わせをアプリケーション内の情報と比較し、一致するものがあれば、登録済みのユーザであると判断し、(4) リソースへのアクセス権限を付与する方法があります。一般に、相手が誰であるかを確認することを「**認証**（Authentication)」と呼び、特定の誰かにリソースへのアクセス権限を与えることを「**認可**（Authorization)」と呼びます。前述の実装例では、(1) - (3) でユーザを「認証」し、(4) で「認可」を与えています。
- AWS 上に構築した Web アプリケーションに対して、なるべく簡単にユーザ認証・認可の機能を実現するためのマネージド型サービスとして、**Amazon Cognito**（コグニート）があります。Cognito が提供するサービスには、大きく分けて下記の 2 つがあります。
  - **ユーザープール**：ユーザの認証を担う機能です。Web アプリやモバイルアプリのユーザに対し、サインイン画面を含む認証機能を提供し、認証完了後に「トークン」を発行します。
  - **ID プール**：ユーザへ認可を与える機能です。ユーザープールで認証を完了したユーザーが特定の AWS サービス（Amazon S3 など）へアクセスできるよう、「トークン」と引き換えに一時的な AWS 認証情報を発行します。
- 最も簡単な実装方式として、Cognito によってホストされるサインアップページとサインインページを用いるやり方があります。Cognito でユーザープールを作成し、ドメインを設定すると、これらのサインイン/アップページが自動で作成されます。Web アプリケーションのユーザは、サインイン画面でメールアドレスとパスワードを入力し、認証を受けます。認証済みユーザには ID トークンが返却され、そのトークンを用いてリソースにアクセスすることで、認可が与えられます。



- Cognito は Application Load Balancer (ALB)、API Gateway、AppSync と合わせて使うことができます。特に ALB と連携する場合は、ALB 側で Cognito ユーザープールとの連携設定をするだけで、認証されていないユーザが ALB にアクセスした場合に、認証フローを自動的に実行するようにできます。Cognito と ALB の認証フローの概要は下記の図の通りです。ALB を使用したユーザ認証の詳細については、こちらのドキュメントを参照してください。



#	クライアントとALBの挙動
1	クライアントがALBにアクセス
2	セッションCookieで判定、未認証の場合はユーザープールの認証フロー開始
3	Hosted UIによる認証・同意と認可コード払い出し(ALBのURLへリダイレクト)
4	ID トークンをALBに送信
5	ALBが ID トークンを用いてアクセストークンを得てユーザー情報取得。Cookieをセットして元URLに対してリダイレクト
6	クライアントがCookie付きでALBにアクセス。認証OKとしてバックエンドにHTTPヘッダを付加してリクエストを送信

i 特定の目的に関連するグラフやチャートを一覧表示し、情報を視覚的に把握できる機能を指します。

ii 何らかの原因により、システムの停止や応答遅延が発生している状態を指します。

iii 外部からの悪意ある攻撃や情報漏洩など、情報セキュリティ上の脅威となる事象を指します。

iv システムの処理能力を向上させる目的で、サーバーの台数を増やすことを「スケールアウト」、サーバーのスペックを増強することを「スケールアップ」といいます。

v AWS Well-Architected フレームワーク - セキュリティの柱においても「データに人の手を入れない」とあるようにデータベースやストレージを手動で処理する必要を減らすためのツール導入が推奨されています。

vi ここでは VPC やサブネットなどのネットワークの単位を指しています。

vii LAN などの企業内ネットワークに存在する端末が、自由に使用できる IP アドレス

viii インターネットゲートウェイを通り、パブリック IP アドレスを使うからといって、即インターネットに出る、ということにはなりません。パブリック IP アドレスを使用する場合においても、AWS でホストされているインスタンスとサービス間のすべての通信は AWS のプライベートネットワークを使用するためです。

参考：https://aws.amazon.com/jp/vpc/faqs/ - Q:2 つのインスタンスがパブリック IP アドレスを使用して通信する場合、またはインスタンスがパブリックな AWS のサービスエンドポイントと通信する場合、トラフィックはインターネットを経由しますか?

ix AWS Outposts、ECS Anywhere、AWS Wavelenght などを、オンプレミスやエッジロケーションで利用することも可能です。