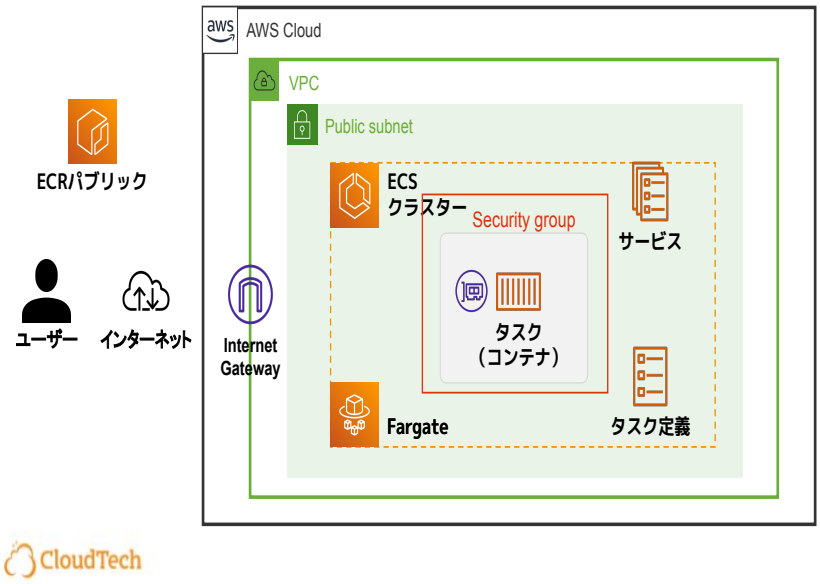
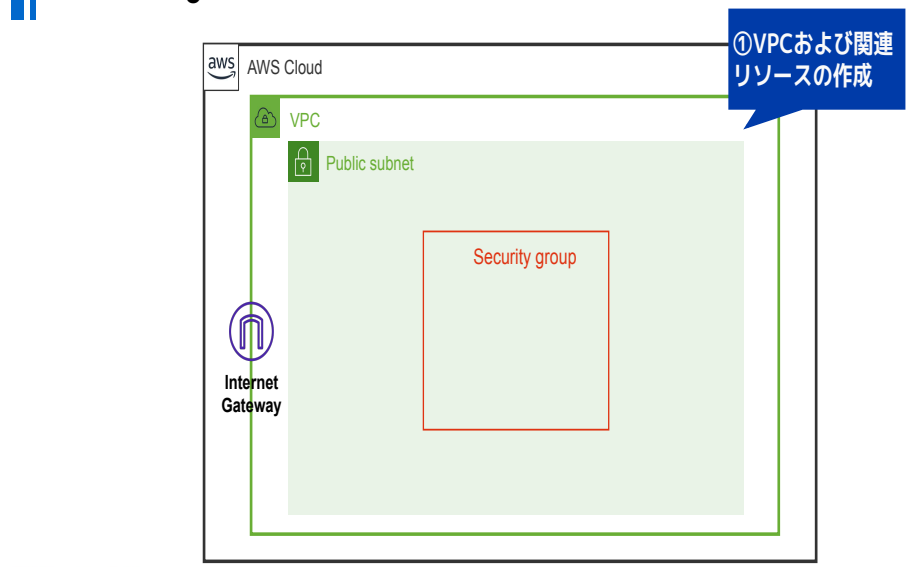
AWS Fargateハンズオン



それではデモの全体概要を説明します。
今回はECSコンソールを使用してECSを作成し、Apacheコンテナの最小環境を構築してみましょう。

全体は**このような構成図**となります。
1つ1つどのリソースから順番に作成していくか流れを解説します。

AWS Fargateハンズオン

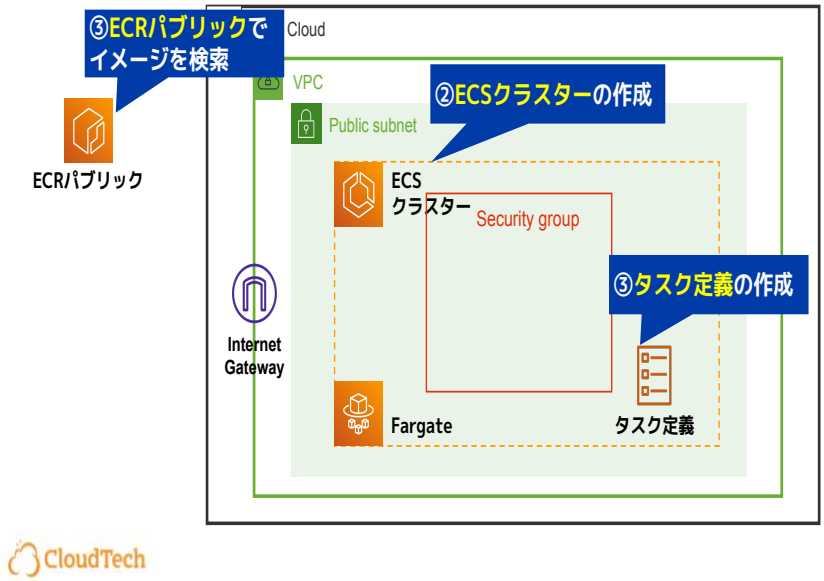


CloudTech

まずは**VPCおよび関連リソース**を作成します。
VPCを1つ作成し、そこにPublicSubnetを1つ作成します。
今回はECRパブリックからコンテナイメージを取得するため、
PrivateSublicではなく、PublicSubnetlにコンテナを構築します。

またコンテナ起動時に必要となる **セキュリティグループ**も1つ作成します。

AWS Fargateハンズオン

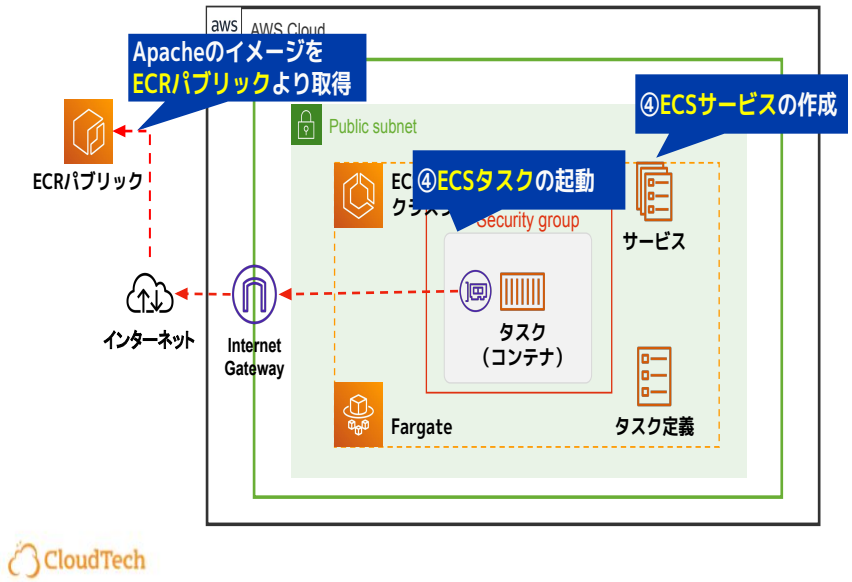


次にECSを実行する基盤となる**ECSクラスター**を作成します。
ECSクラスターを作成する際に起動タイプとして、今回の講座で学んだ**Fargateを選択**します。

そして、次にECSタスクを起動するテンプレートとなる **タスク定義** を作成します。
タスク定義ではApacheコンテナを作成するために、Apacheの Dockerイメージを定義する必要があります。

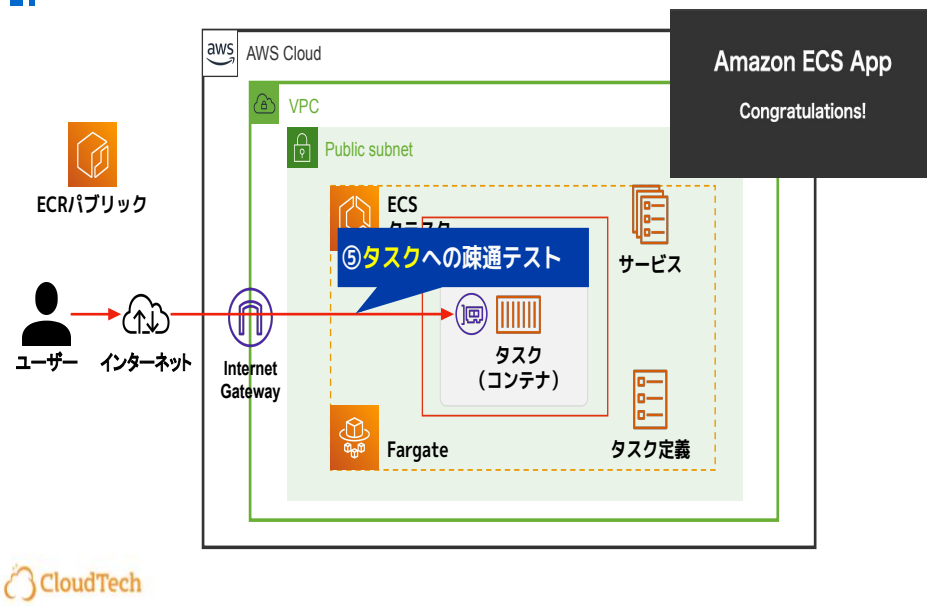
今回は**ECRパブリックよりイメージ**を取得します。

AWS Fargateハンズオン



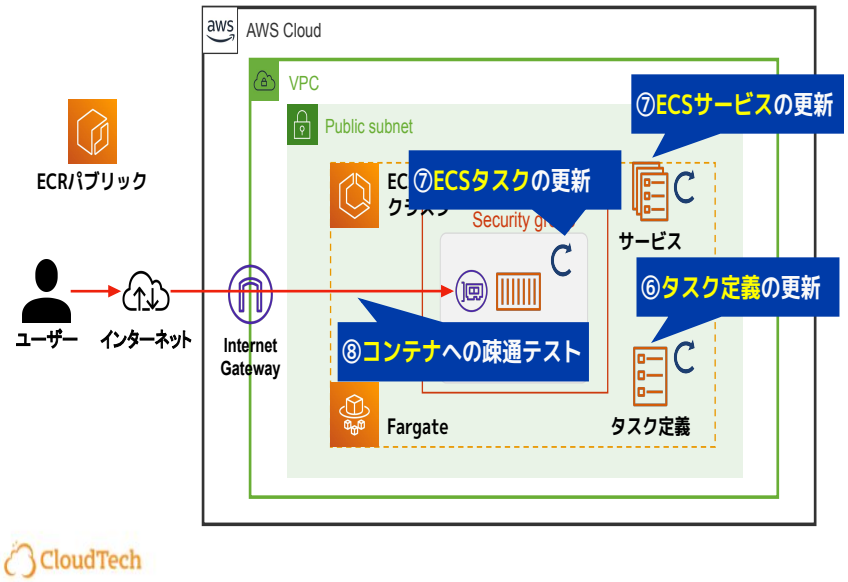
次に実行中のタスクを管理する**ECSサービス**を作成します。
ECSサービスを作成すると、作成したタスク定義に基づいて **ECSタスク**が同時に起動されます。

このタスク起動時にインターネット経由で **Apacheのイメージを ECRパブリック**より取得します。



そして、起動したECSタスクにアクセスし、実際に**Apacheコンテナが起動**されていることを確認します。
うまく起動すると、このようにブラウザに**「Amazon ECS App Congratulations!」**と表示されるようになります。

AWS Fargateハンズオン



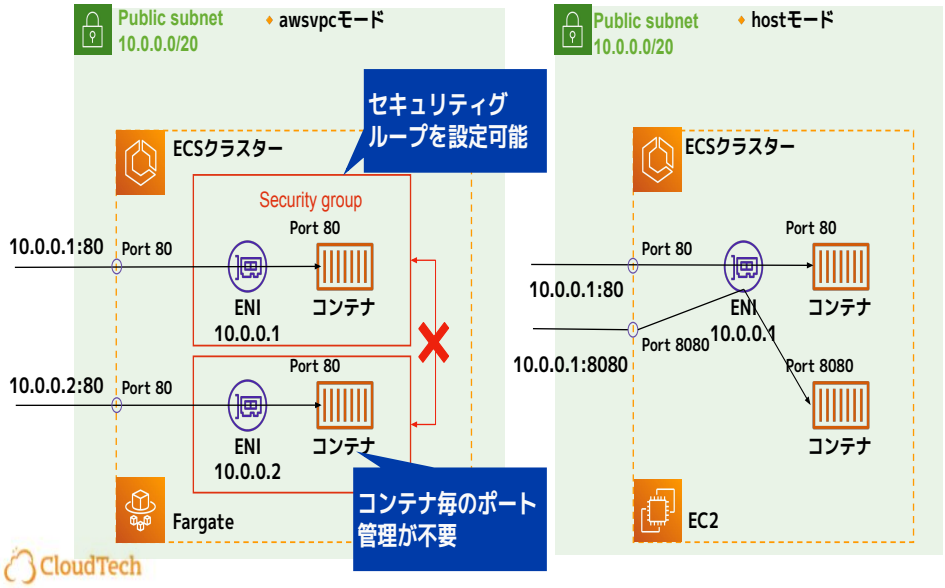
ECSタスクは一度作成後、何度も更新していくことが一般的となります。

そのため、1つ目のECSタスクの起動が確認できたら、次に **タスク定義の更新**をしてみましょう。
ブラウザに表示される文言が変更になるよう定義します。

そして、新しいタスク定義のバージョンを更新するように **ECSサービスも更新**しましょう。
すると、**新しいタスクが起動し、古いタスクが削除されるよう更新**されます。

最後に新しく起動した**Apacheコンテナにアクセス**し、更新したタスク定義に合わせてブラウザに表示される文言が変更されていることを確認してみましょう。
1つ1つの手順は簡単なので、ポイントを抑えて Fargateでコンテナを起動してみましょう！

ネットワーク設定：awsvpcについて



それでは先ほどのタスク定義でデフォルトで指定されていたネットワーク設定「awsvpc」について簡単に解説します。
まず**このような構成の場合**、awsvpcでは起動されるコンテナ毎に**ENIが作成され、独自のプライベートIP**が割り当てられます。

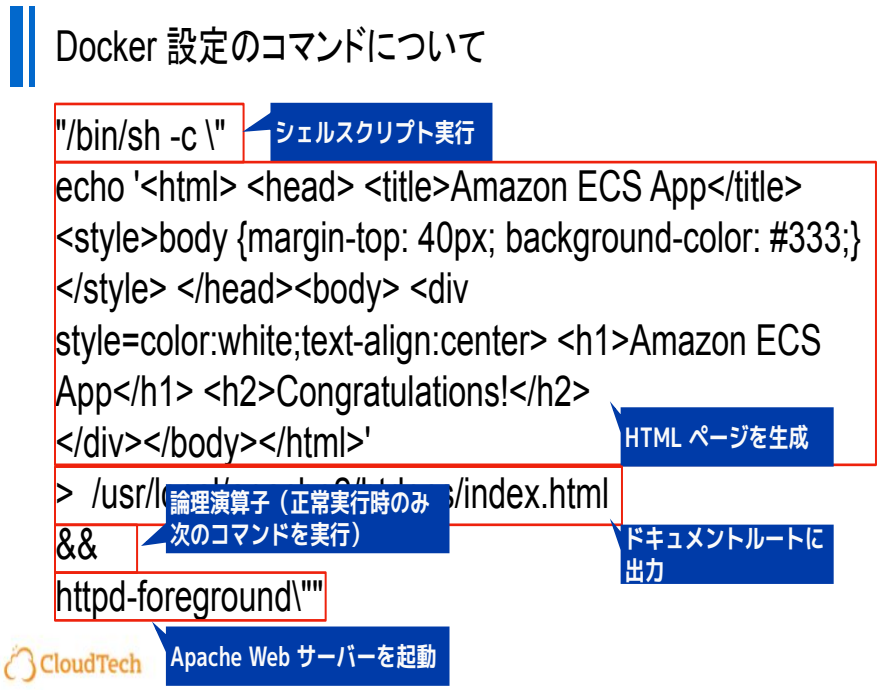
なおFargateの場合はネットワークモードにawsvpcしか選ぶことができなかったと思います。
これはEC2の管理が不要になった分、コンテナを起動した際にVPC内で専用のENIを持つ必要があるためです。
そのためFargateを選ぶと自動的にawsvpcのネットワークモードが選ばれるということですね。

このネットワーク設定により、メリットが大きく2点挙げられます。
1点目は各コンテナ毎に**セキュリティグループを設定**できるという点です。
セキュリティグループが設定できることで同じホスト内のコンテナからの**通信を制限すること**ができるようになります。

セキュリティが保てるということですね。

2点目は**コンテナ毎のポート管理が不要になる** という点です。
ここで仮に複数のコンテナに **ENIが1つしか割り与えられていない場合** を想像してみましょう。
これはFagateでは選択ができない、**hostモード**というネットワーク設定になります。
この場合、2つのWEBサーバのコンテナにアクセスできるようにするためには、**別々のポート番号を割り当てる必要があります。**
これにより別々のコンテナに **アクセスできるようになります** が、同じポートは使用できないということなります。
しかしawsvpcでは、コンテナ毎に対応する ENI とプライベート IP アドレスが割り与えられています。
各ENIは別々であるため、トラフィックとコンテナのポートに **同じ80番のポート**を利用することができます。
よって別々のコンテナに同じポート番号で **アクセスできるようになります**。
コンテナ毎にどのポートを割り当てていたんだけ？という管理の手間が省けますね。

このようにawsvpcでは起動されるコンテナ毎にENIが作成され、独自のプライベートIPが割り当てられるモードということを覚えておきましょう。



次にDocker設定で定義したコマンドについて解説します。

このコマンドはコンテナ内で ApacheのWeb サーバーが起動され、生成された HTML ページが表示されるようになるコマンドです。
それぞれ意味のあるセクションを分けて解説します。

- `/bin/sh -c` は、シェルスクリプトを実行するためのコマンドです。
- `echo '<html> ... </html>'` は、HTMLコンテンツを生成するためのコマンドです。この HTML コードは、タイトル、スタイル、およびメッセージを含む単純な Web ページを生成しています。画面には背景色が暗い (#333) で、テキストは白色で中央揃えにされており、「 Amazon ECS App」というタイトルと「Congratulations!」というメッセージが含まれるように設定しています。
- `> /usr/local/apache2/htdocs/index.html` は、生成された

- HTML コンテンツを `/usr/local/apache2/htdocs/index.html` ファイルに出力することを示しています。ちなみにこれは、 Apache Web サーバーのデフォルトのドキュメントルートです。
- `&&` は論理演算子であり、前のコマンドが正常に実行された場合にのみ、次のコマンドを実行することを示しています。
- `httpd-foreground` は、Apache Web サーバーを起動し、フォアグラウンドモードで実行するコマンドです。 これにより、作成したウェブページがウェブサーバー上でホストされ、アクセス可能になります。

要約すると、このコマンドは、Amazon ECS コンテナ内で 単純な HTML ページを生成し、
生成された HTML ページを Apache Web サーバーのドキュメントルートに書き込み、 Apache Web サーバーを起動するというコマンドです。