

AWS Black Belt Online Seminar

Amazon ECS 入門

吉田 英史

Solutions Architect

2024/08



自己紹介

吉田 英史

アマゾンウェブサービスジャパン
ソリューションアーキテクト

小売・消費財のお客様を中心にご支援しています。
生活に欠かせない様々なビジネスをクラウドで加速
するお手伝いができることを、何より嬉しく感じ
ています。

好きな AWS サービス
AWS Fargate



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



本セミナーの主な対象者

- これから AWS を利用される予定の、アプリケーションおよびインフラ担当者
- Amazon Elastic Container Service (Amazon ECS) の概要や始め方を知りたい方
- クラウド上の既存ワークロードの、コンテナ化を検討している方
- オンプレミスの既存コンテナワークロードの、クラウド移行を検討している方

アジェンダ

1. Amazon Elastic Container Service (Amazon ECS) とは
2. Amazon ECS の構成
3. Amazon ECS の始め方
4. まとめ

Amazon Elastic Container Service (Amazon ECS) とは

AWS のコンテナサービス

オーケストレーション

コンテナのデプロイ、スケジューリング、スケーリング



Amazon ECS



Amazon EKS

イメージレジストリ

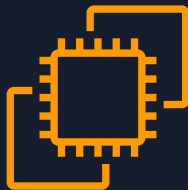
コンテナイメージの格納



Amazon ECR

ホスティング

コンテナ実行環境



Amazon EC2



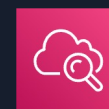
AWS Fargate

その他の関連サービス



AWS App Runner

AWS
Cloud Map



**Amazon CloudWatch
Container Insights**

Amazon ECS: フルマネージドコンテナサービス



- ミドルウェアへのパッチ、アップグレード、セキュリティ対応不要
- 他の AWS サービスとのネイティブな統合
- ログ、メトリクス、イベントを最初からサポート
- グローバルに利用可能で、パフォーマンスが高く、スケーラブル
- 追加料金不要！

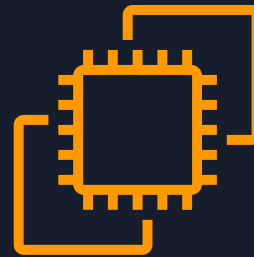
コンテナ実行環境の選択肢

コンテナ
オーケストレーター
(コントロールプレーン)

コンテナの実行環境
(データプレーン)



Amazon ECS



Amazon EC2

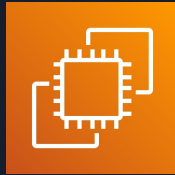


AWS Fargate

Amazon ECS におけるコンテナ実行環境

ECS on EC2

(コンテナを仮想サーバー上で動作)



アプリケーションコンテナ

ホストのスケールリング

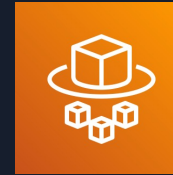
コンテナエージェント設定

ホスト OS / ライブラリ設定

お客様が管理するレイヤー

ECS on Fargate

(コンテナをサーバーレスで動作)



アプリケーションコンテナ

ホストのスケールリング

コンテナエージェント設定

ホスト OS / ライブラリ設定

AWSが提供するレイヤー

Fargateを活用するとアプリケーションコンテナの開発に集中できる

Amazon ECS の動作イメージ (on EC2)



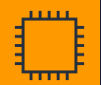
Amazon ECS

スケジューリングとオーケストレーション

クラスター管理

配置エンジン

クラスター



EC2インスタンス

タスク



タスク



ecs-agent

Docker Engine

OS



EC2インスタンス

タスク



タスク



ecs-agent

Docker Engine

OS

タスク



Docker
コンテナ



Docker
コンテナ



D
コ

Amazon ECS (on EC2) の特徴と課題

特徴

- コンテナホストを自由に選択、設定できる
- CPU、メモリ、ディスク、OS、バージョンなど
- ネットワークモードの選択 (外部接続しない、ホストのネットワークを利用、ENIへのアタッチなど)
- 柔軟なデータボリューム利用

課題

- コンテナホストの管理が必要
 - OSやエージェント類へのパッチ当てや更新
- EC2インスタンス数のスケーリング

- ✓ Amazon ECS on Fargate で解決
- ✓ コンテナホストがマネージドになるため、トレードオフがある

Amazon ECS の動作イメージ (on Fargate)



Amazon ECS

スケジューリングとオーケストレーション

クラスター管理

配置エンジン

クラスター



AWS Fargate

タスク



タスク



タスク



AWS Fargate

タスク



タスク



タスク

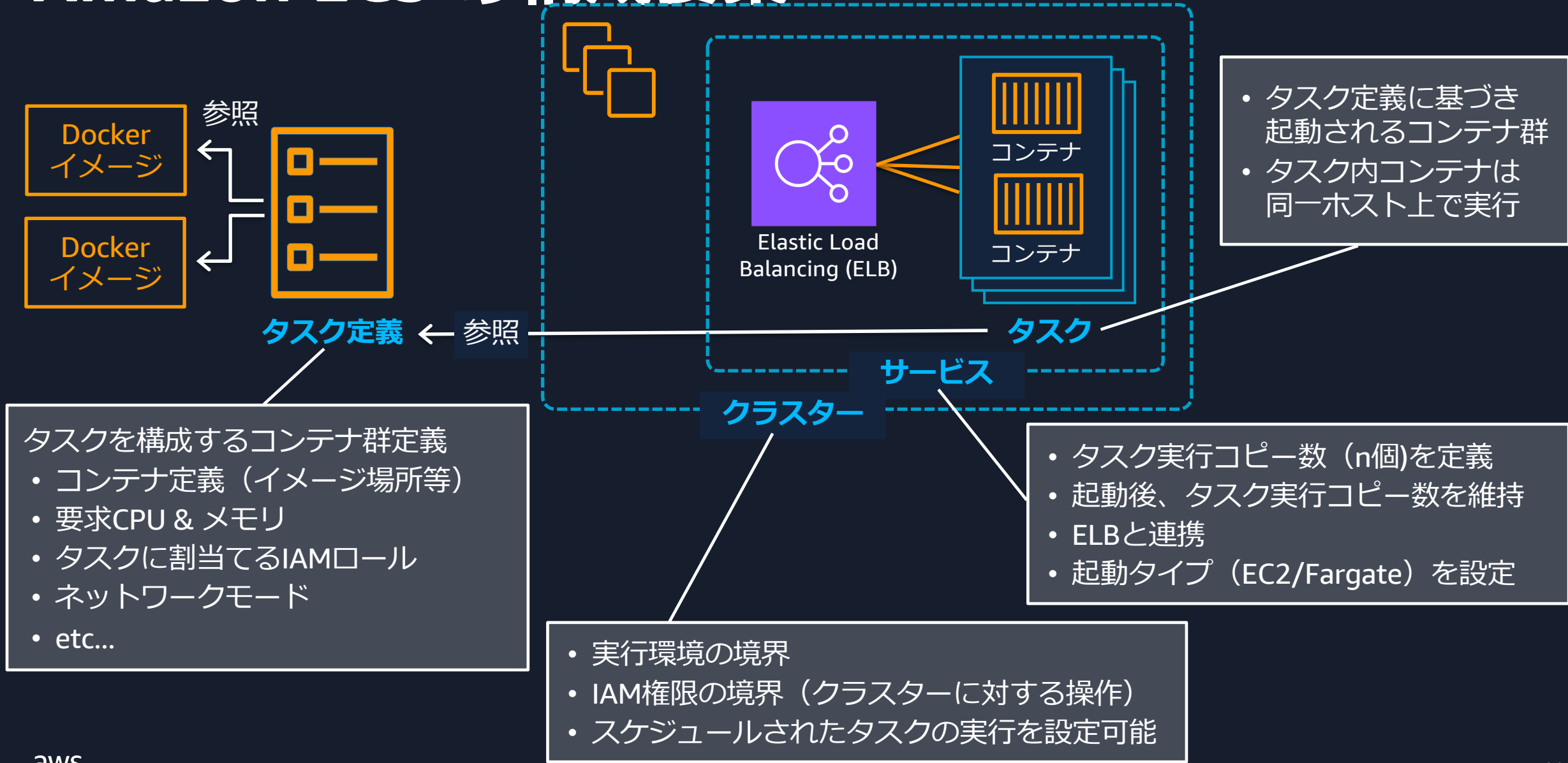


AWS Fargate

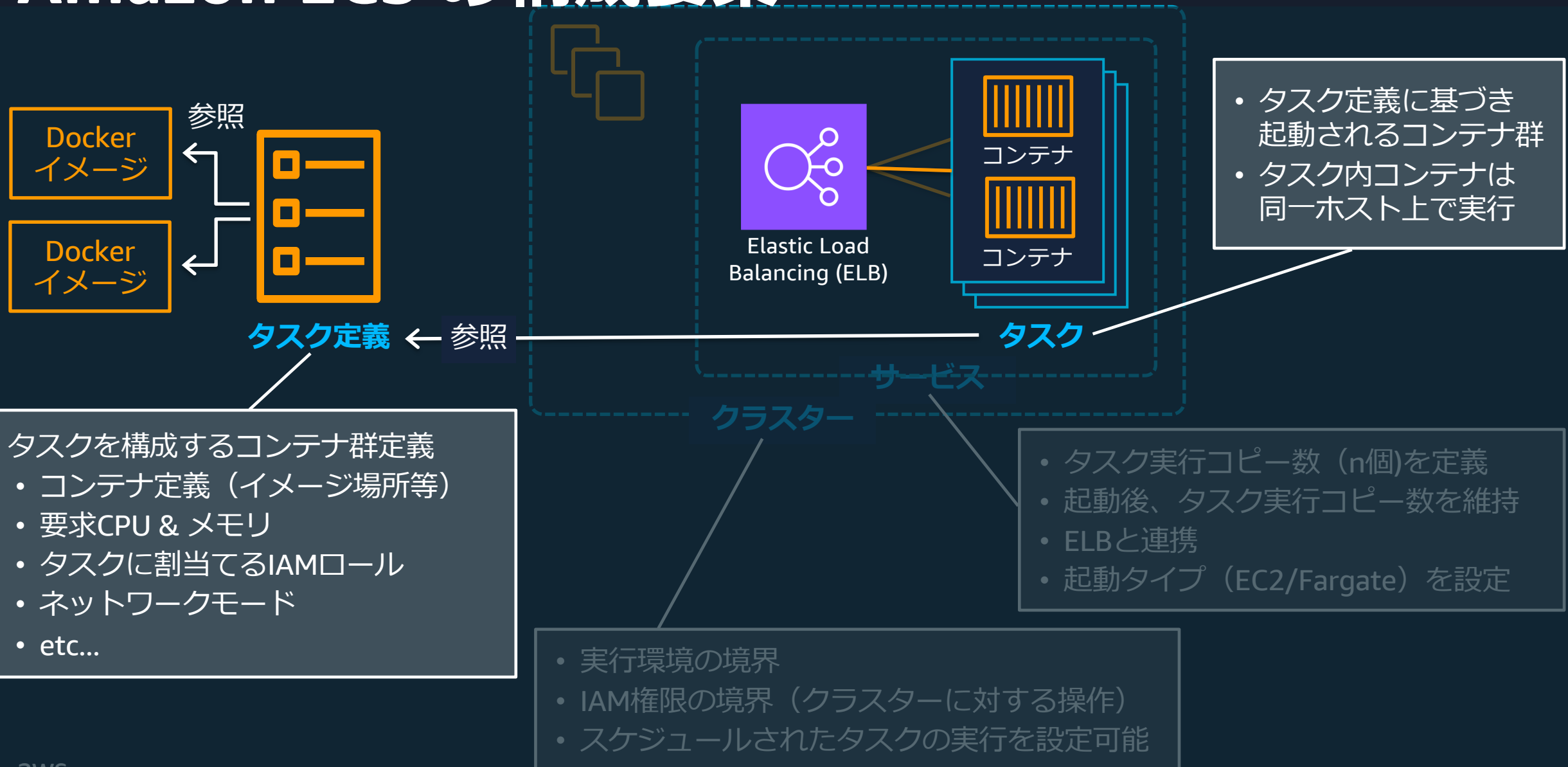


Amazon ECS の構成

Amazon ECS の構成要素



Amazon ECS の構成要素



タスク定義 = コンテナ(群)の実行単位定義

タスクとしてのまとまりを
所定の JSON 書式で定義したもの

- 例：Front-end サービス、Back-end サービス、...
- family と revision (1から始まる数)で特定

スタンドアロンタスク or サービスによって、
タスク定義から起動される

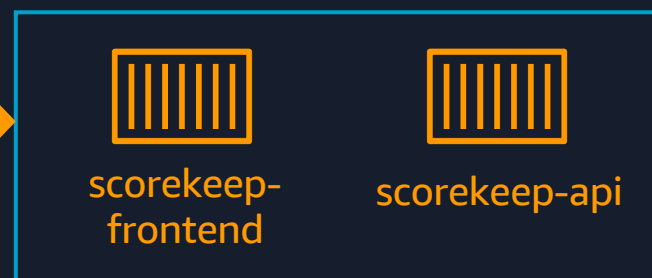
- 1つ以上のコンテナを実行するリソース
- タスク内のコンテナ群は必ず同じホスト上で実行
- 1つのタスク定義は最大 10 コンテナ指定可能

タスク定義

```
{
  "family": "scorekeep",
  "containerDefinitions": [
    {
      "name": "scorekeep-frontend",
      "image": "xxx.dkr.ecr.us-east-1.amazonaws.com/fe"
    },
    {
      "name": "scorekeep-api",
      "image": "xxx.dkr.ecr.us-east-1.amazonaws.com/api"
    }
  ]
}
```

タスク

実行



タスク定義 | 代表的なパラメータ

必須

1. ファミリー (family)
2. コンテナ定義 (containerDefinitions)

オプション

3. タスクサイズ (cpu / memory)
4. タスクロール (taskRoleArn)
5. タスク実行ロール (executionRoleArn)
6. ネットワークモード (networkMode)
7. ボリューム (volumes)

```
{  
  "family": "",  
  "taskRoleArn": "",  
  "executionRoleArn": "",  
  "networkMode": "none",  
  "containerDefinitions": [...],  
  "volumes": [...],  
  "placementConstraints": [...],  
  "requiresCompatibilities": [...],  
  "cpu": "",  
  "memory": "",  
  "tags": [...],  
  "pidMode": "host",  
  "ipcMode": "host",  
  "proxyConfiguration": {...}  
}
```

タスク定義 | ファミリー

必須

1. ファミリー (family)
2. コンテナ定義 (containerDefinitions)

- タスク定義の名前のようなもの
- タスク定義を登録する際に必ず指定する
- ファミリーとリビジョン番号(最初は1)で1つのタスク定義が特定される

7. ボリューム (volumes)

```
{  
  "family": "",  
  "taskRoleArn": "",  
  "executionRoleArn": "",  
  "networkMode": "none",  
  "containerDefinitions": [...],  
  "volumes": [...],  
  "placementConstraints": [...],  
  "requiresCompatibilities": [...],  
  "cpu": "",  
  "memory": "",  
  "tags": [...],  
  "pidMode": "host",  
  "ipcMode": "host",  
  "proxyConfiguration": {...}  
}
```

タスク定義 | コンテナ定義

必須

1. ファミリー (family)
2. コンテナ定義 (containerDefinitions)

オプション

- タスク実行時、コンテナランタイムに渡されるコンテナ定義
- コンテナのイメージ/ポートマッピング/メモリ制限などを指定

```
{  
  "family": "",  
  "taskRoleArn": "",  
  "executionRoleArn": "",  
  "networkMode": "none",  
  "containerDefinitions": [...],  
  "volumes": [...],  
  "placementConstraints": [...],  
  "requiresCompatibilities": [...],  
  "cpu": "",  
  "memory": "",  
  "tags": [...],  
  "pidMode": "host",  
  "ipcMode": "host",  
  "proxyConfiguration": {...}  
}
```

コンテナ定義 | 代表的なパラメータ (1/3)

名前 (name)

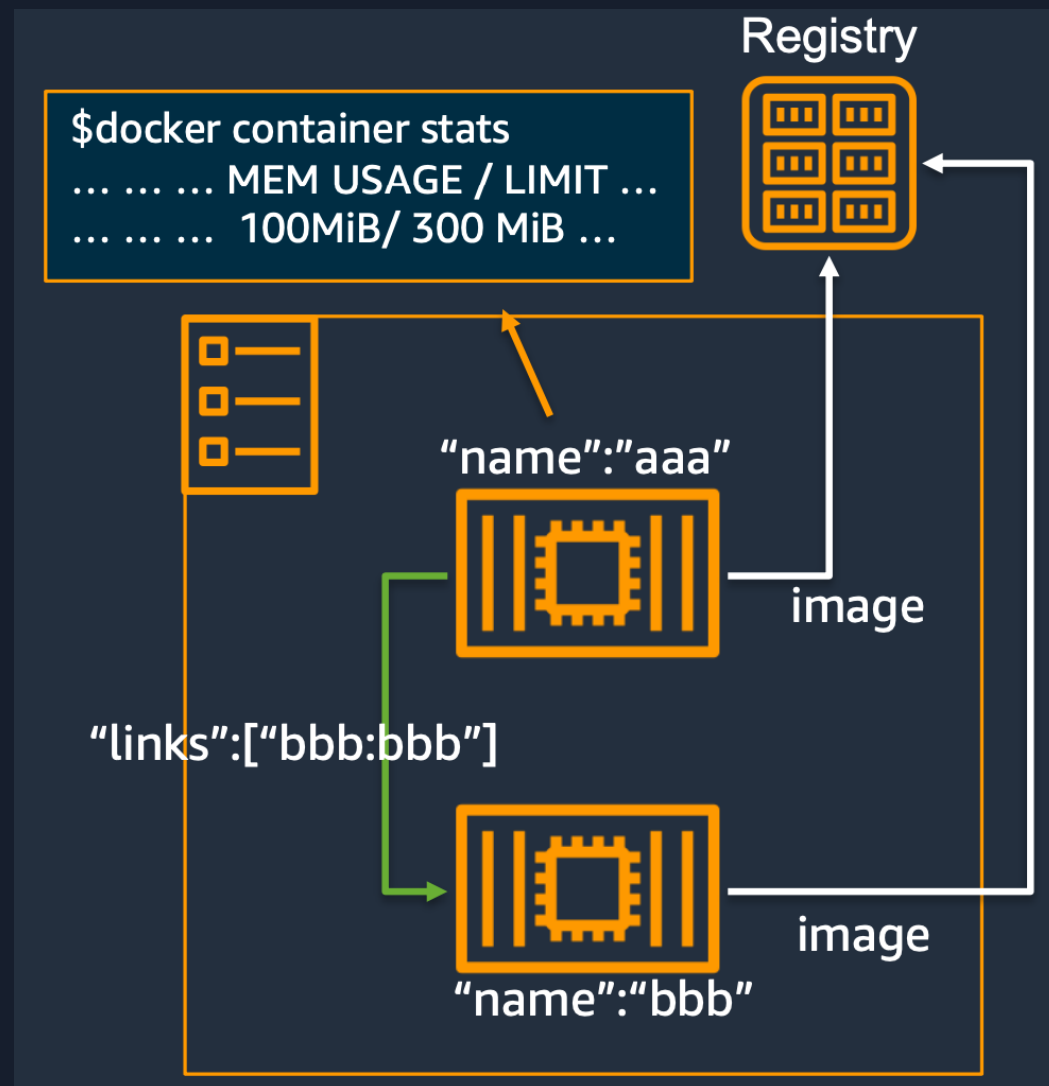
- コンテナの名前。
- ECS Exec を用いてコンテナ内で作業する際や、bridge モードのタスク内で複数のコンテナをリンクする際に、この名前を利用

イメージ (image)

- 使用するコンテナのイメージ URI を指定

メモリ (memory, memoryReservation)

- 使用するメモリ容量を指定



コンテナ定義 | 代表的なパラメータ (2/3)

環境変数 (environment)

- コンテナに**環境変数**として設定する
- 定義ファイルに平文で記述されるので**機密情報には向かない**

```
"environment": [  
  {"name": "hoge", "value": "fuga"}  
]
```

シークレット (secrets)

- **機密データ**を環境変数に設定する場合に利用
 - AWS Secrets Manager シークレット
 - AWS Systems Manager パラメータストアのパラメータを参照可能
- それぞれタスク実行ロールに適切なIAMアクセス許可の設定が必要

```
"secrets": [  
  {"name": "environment_variable_name",  
   "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-AbCdEf"  
  }  
]
```

コンテナ定義 | 代表的なパラメータ (3/3)

ログ設定 (logConfiguration)

- 標準出力 (stdout) と標準エラー出力 (stderr) に出だし、ストリームとして扱うのがベストプラクティス
- 出力されるログの形式や、保存・転送などを制御するログドライバーやオプションを指定

#	logDriver 名	説明	EC2	Fargate
1	awslogs	CloudWatch Logs へ転送	○	○
2	awsfirelens	Fluentd, CloudWatch Logs, Kinesis へ転送	○	○
3	gelf	Graylog Extended Log Format 形式ログ出力	○	-
4	json-file	JSON 形式ログを出力	○	-
5	journald	systemd のログ管理システムに転送	○	-
6	logentries	Logentries ログ管理システムへ転送	○	-
7	splunk	Splunk ログ管理システムへ転送	○	○
8	fluentd	EC2 インスタンスの fluentd へ転送	○	-
9	syslog	EC2 インスタンスの syslog へ転送	○	-

タスク定義 | タスクサイズ

必須

1. ファミリー (family)
2. コンテナ定義 (containerDefinitions)

オプション

3. タスクサイズ (cpu / memory)

4. タスクロール (taskRoleArn)
5. エグゼキューションロール (executionRoleArn)

- タスクが使用する CPU とメモリの合計値
- Fargate でホストされるタスクの場合は、必須のフィールド

```
{  
  "family": "",  
  "taskRoleArn": "",  
  "executionRoleArn": "",  
  "networkMode": "none",  
  "containerDefinitions": [...],  
  "volumes": [...],  
  "placementConstraints": [...],  
  "requiresCompatibilities": [...],  
  "cpu": "",  
  "memory": "",  
  "tags": [...],  
  "pidMode": "host",  
  "ipcMode": "host",  
  "proxyConfiguration": {...}  
}
```

タスク定義 | CPU とメモリ定義

タスクレベルリソース

✓ タスク単位の要求リソース量を定義

- cpu : cpu-unit 数を指定
 - 1 vCPU = 1,024 cpu-units
- memory : MiB 単位で指定
- Fargate では必須パラメータ

コンテナレベルリソース

- ### ✓ 各コンテナへのタスクリソースの分配を定義
- オプション指定

```
{
  "family": "scorekeep",
  "cpu": 1024,
  "memory": 2048,
  "containerDefinitions": [
    {
      "name": "scorekeep-frontend",
      "image": "xxx.dkr.ecr.us-east-1.amazonaws.com/fe",
      "cpu": 256,
      "memory": 768,
      "memoryReservation": 512
    },
    {
      "name": "scorekeep-api",
      "image": "xxx.dkr.ecr.us-east-1.amazonaws.com/api",
      "cpu": 768,
      "memoryReservation": 512
    }
  ]
}
```

タスクレベル
リソース指定

コンテナレベル
リソース指定

タスク定義 | IAM との連携

必須

1. ファミリー (family)

- **コンテナ**が利用できる IAM ロールを指定
- アプリケーションはこの IAM ロールで許可された AWS サービスのAPIを実行できる

5. タスクリソース (cpu / memory)

4. タスクロール (taskRoleArn)

5. タスク実行ロール (executionRoleArn)

6. ネットワークモード (networkMode)

- **ECS コンテナエージェント**が利用する IAM ロールを指定
- この権限を使ってコンテナのイメージを pull したり CloudWatch Logs に書き込みを行う

```
{  
  "family": "",  
  "taskRoleArn": "",  
  "executionRoleArn": "",  
  "networkMode": "none",  
  "containerDefinitions": [...],  
  "volumes": [...],  
  "placementConstraints": [...],  
  "requiresCompatibilities": [...],  
  "cpu": "",  
  "memory": "",  
  "tags": [...],  
  "pidMode": "host",  
  "ipcMode": "host",  
  "proxyConfiguration": {...}  
}
```

タスク定義 | ネットワークモード (1/2)

必須

1. ファミリー (family)

- タスクに属するコンテナが使用するネットワークモードを指定
- awsvpc, bridge, host, default, none のいずれかを指定

5. タスク実行ロール (executionRoleArn)

6. ネットワークモード (networkMode)

7. ボリューム (volumes)

```
{
  "family": "",
  "taskRoleArn": "",
  "executionRoleArn": "",
  "networkMode": "none",
  "containerDefinitions": [...],
  "volumes": [...],
  "placementConstraints": [...],
  "requiresCompatibilities": [...],
  "cpu": "",
  "memory": "",
  "tags": [...],
  "pidMode": "host",
  "ipcMode": "host",
  "proxyConfiguration": {...}
}
```

タスク定義 | ネットワークモード (2/2)

Fargate
実行環境

awsvpc モード

- ECS 管理下の ENI (仮想サーバーの仮想ネットワークカード) がタスクにアタッチされる

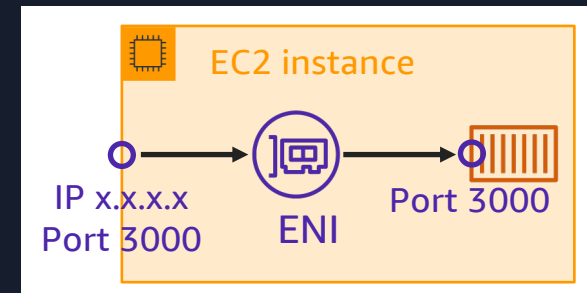
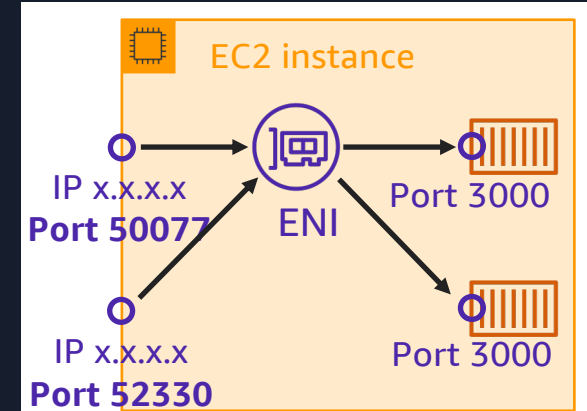
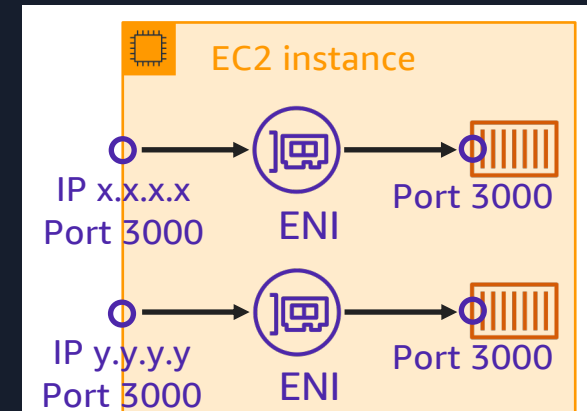
EC2
実行環境

bridge モード

- 仮想ネットワークブリッジを利用してホスト/コンテナポートをマッピングして通信

host モード

- コンテナをホストしている EC2 インスタンスの IP/ポートを介して通信



タスク定義 | ボリューム

必須

- データを共有・永続化するためのボリュームのリスト
- Docker ボリューム、バインドマウント、EBS ボリューム、EFS ボリューム、FSx for Windows File Server ボリュームの5種類が利用可能

6. ネットワークモード (networkMode)

7. ボリューム (volumes)

```
"family": "",
"taskRoleArn": "",
"executionRoleArn": "",
"networkMode": "none",
"containerDefinitions": [...],
"volumes": [...],
"placementConstraints": [...],
"requiresCompatibilities": [...],
"cpu": "",
"memory": "",
"tags": [...],
"pidMode": "host",
"ipcMode": "host",
"proxyConfiguration": {...}
}
```

タスクからのデータボリュームの使用

Fargate 実行環境

Fargate バインドマウント

- 揮発性のストレージをコンテナにマウントし、コンテナ間で共有
- 使用するすべてのコンテナが停止するとデータが削除される

Amazon EBS ボリューム

- 高スループットなトランザクション集約型アプリケーション向け
- タスクごとに1つの EBS ボリュームにアタッチ可能

Amazon EFS ボリューム

- ストレージ容量が伸縮自在で、自動的に拡大および縮小される

Docker ボリューム

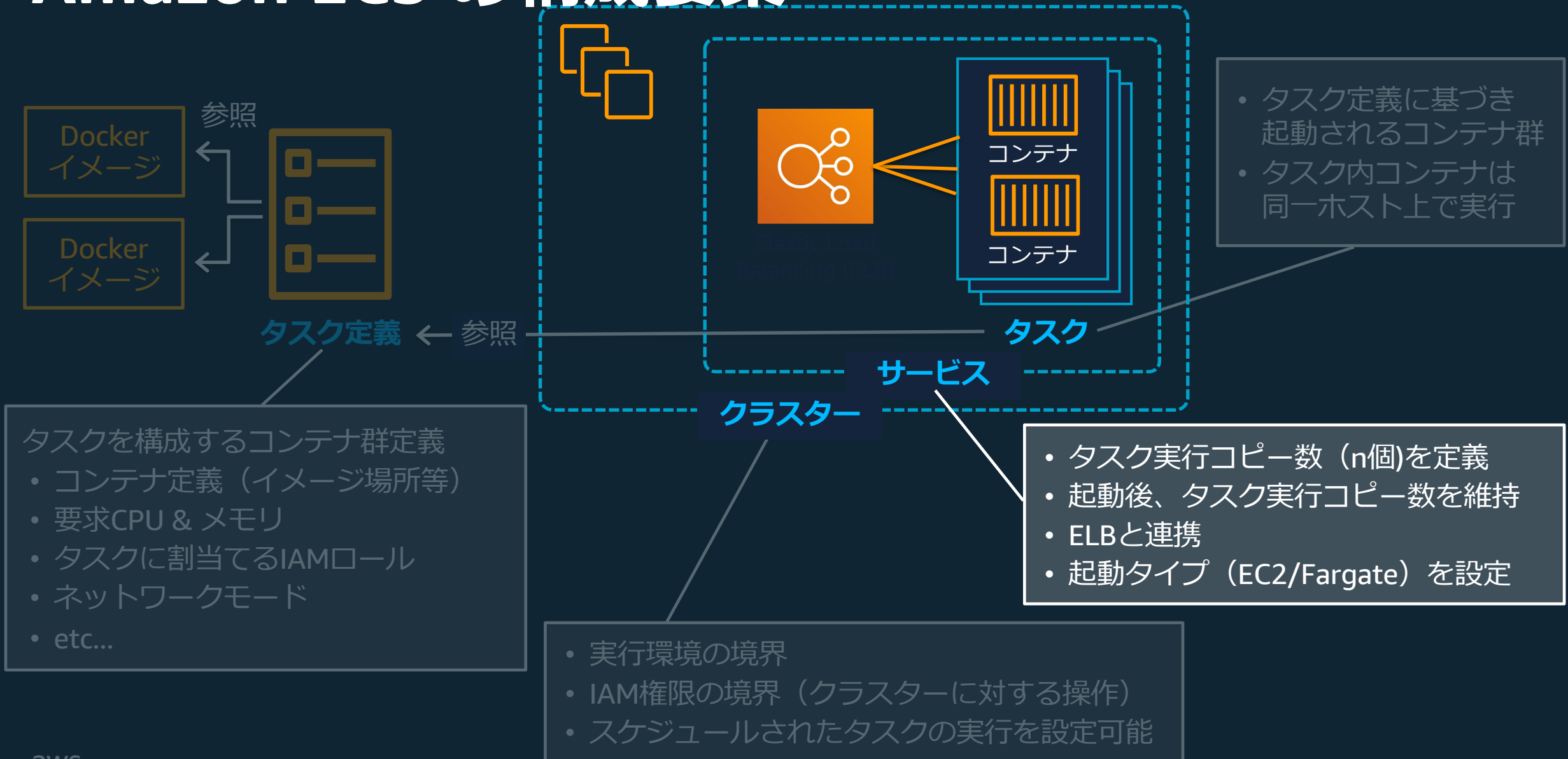
- タスク間での共有や明示的なライフサイクル管理
- 3rd Party のボリュームドライバーの利用ができる

EC2 バインドマウント

- ホストマシン上のファイルやディレクトリをコンテナにマウント
- タスク定義によってデータをホストのライフサイクルに関連付けが可能

EC2 実行環境

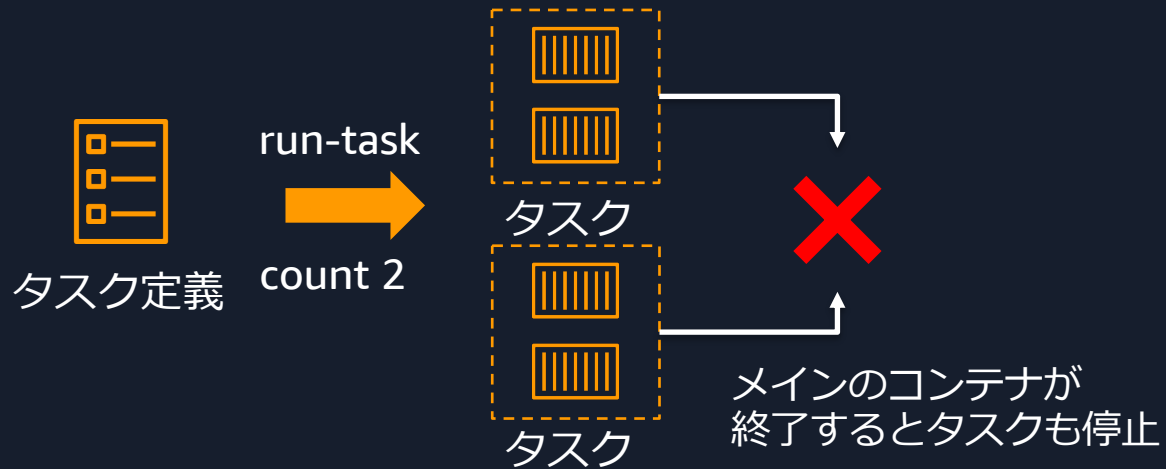
Amazon ECS の構成要素



コンテナの実行方法

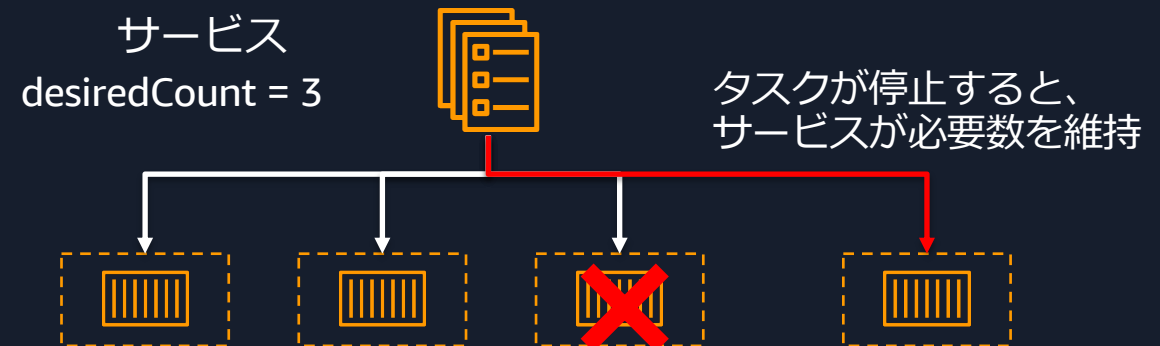
スタンドアロンタスク

- タスク定義に従って実行されるアプリケーションの実行単位
- タスク定義の一部のパラメータはタスク実行時に上書き可能
- 用途：バッチジョブなど処理が終わると停止するワークロードなど



サービス

- 指定した数のタスクを維持する
- タスクが失敗/停止した場合は新しいタスクを起動して置き換え
- 用途：Web アプリケーションなど長時間実行するワークロード
- ELB との連携 / Auto Scaling 機能



サービス | タスク数のAuto Scaling

1. ターゲット追跡スケーリングポリシー

- 指定したメトリクスがターゲットの値に近づくように自動的に調整

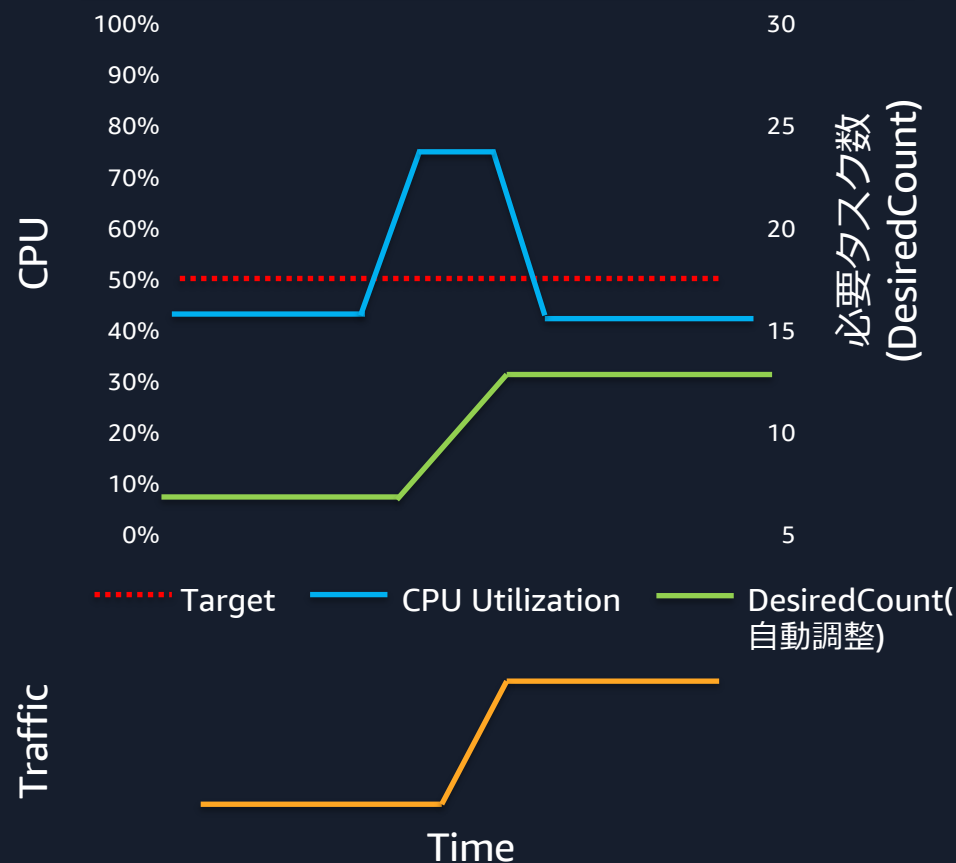
2. ステップスケーリングポリシー

- アラームをトリガーに調整値に基づいて増減
- ターゲット追跡スケーリングポリシーと組み合わせることでより高度なスケーリングも可能

3. スケジュールに基づくスケーリング

- 日付と時刻に基づいてタスク数を増減

ターゲット追跡スケーリングポリシーの動作イメージ



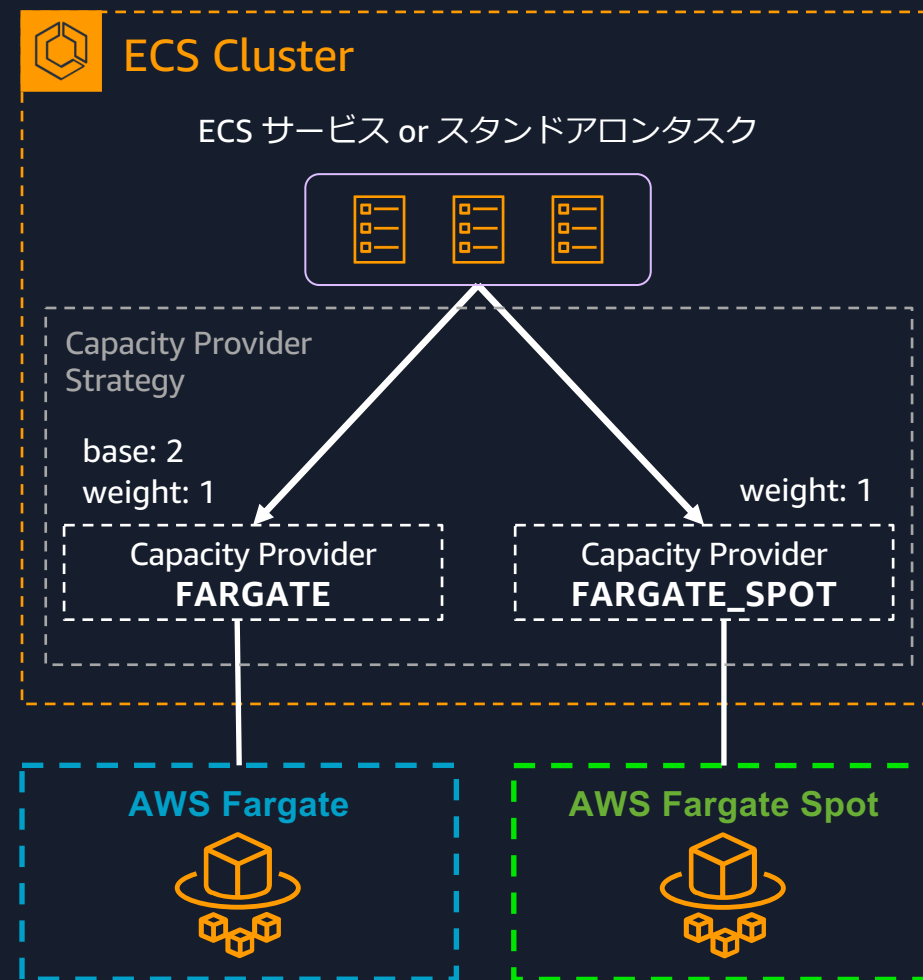
Capacity Providers の概要

Capacity Provider

- タスクが実行されるインフラストラクチャを設定
- Auto Scaling グループや Fargate を指定
- Fargate キャパシティプロバイダーは予約済み
 - FARGATE
 - FARGATE_SPOT

Capacity Provider Strategy

- タスクをどのキャパシティプロバイダーに配置するかを決定する「戦略」
- ベース値やウェイト値で配置を細かく制御可能



Amazon ECS の始め方

Amazon ECS の始め方

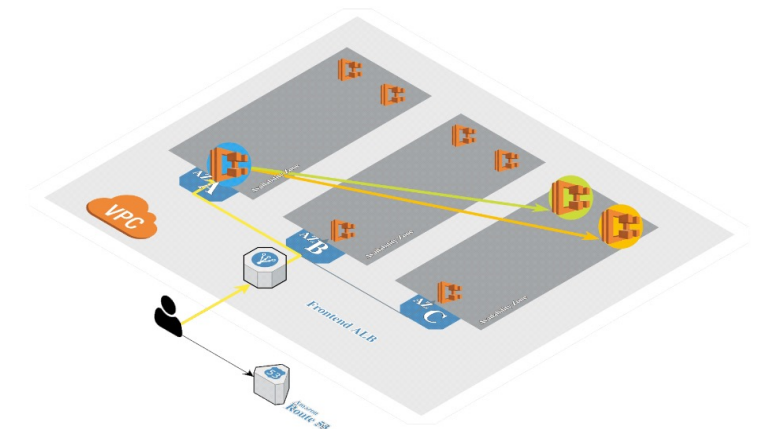
- Amazon Elastic Container Service の開始方法
 - <https://aws.amazon.com/jp/ecs/getting-started/>
- Amazon ECS を始めるにあたって、追加料金は不要
 - ワークロードが使用するリソースに対しての課金
 - <https://aws.amazon.com/jp/ecs/pricing/>
- Amazon ECS のお客様導入事例
 - <https://aws.amazon.com/jp/containers/customers/>
- Amazon ECS に関する AWS Blog 記事
 - <https://aws.amazon.com/jp/blogs/news/tag/amazon-ecs/>

The screenshot shows the AWS website's 'Amazon Elastic Container Service の開始方法' (Getting started with Amazon Elastic Container Service) page. The page is in Japanese and includes a navigation bar with links to '製品' (Products), 'ソリューション' (Solutions), '料金' (Pricing), 'ドキュメント' (Documentation), '学ぶ' (Learn), 'パートナーネットワーク' (Partner Network), 'AWS Marketplace', 'カスタマーサポート' (Customer Support), and 'イベント' (Events). The main heading is 'Amazon Elastic Container Service の開始方法'. Below this, there is a section for 'ログインとセットアップ' (Login and Setup) with a link to the 'コンソール' (Console) and a button for 'AWS Copilot のダウンロード' (Download AWS Copilot). Another section, 'Amazon ECS デジタルトレーニング' (Amazon ECS Digital Training), describes a free on-demand course. At the bottom, there are four cards for related services: 'Amazon Elastic Container Service 入門' (Getting started with Amazon ECS), 'AWS Fargate の紹介' (Introduction to AWS Fargate), 'Amazon Elastic Container Registry 入門' (Getting started with Amazon ECR), and 'トラブルシューティング: Amazon Elastic Container Service (Amazon ECS)' (Troubleshooting: Amazon ECS).

その他のリソース

- Amazon ECS の導入を加速させる Workshop およびツール
 - ECS Blueprints
 - <https://github.com/aws-ia/ecs-blueprints>
 - <https://aws.amazon.com/jp/blogs/news/accelerate-amazon-ecs-based-workloads-with-ecs-blueprints/>
 - AWS Copilot CLI
 - <https://aws.github.io/copilot-cli/ja/>
 - <https://aws.amazon.com/jp/blogs/news/introducing-aws-copilot/>
 - Amazon ECS Workshop
 - <https://ecsworkshop.com/>

Amazon ECS Workshop



In this workshop, we will launch a frontend and multiple backend services on Amazon Elastic Container Service, and explore how you might adopt this workflow into your environment.

まとめ

- Amazon ECS とは
 - フルマネージドのコンテナオーケストレーションサービス
- Amazon ECS の構成要素
 - タスク定義 / タスク / サービス / クラスター
 - タスク定義の詳細
- Amazon ECS の始め方



Amazon Elastic Container Service

AWS Black Belt Online Seminar とは

- 「サービス別」「ソリューション別」「業種別」などのテーマに分け、アマゾン ウェブ サービス ジャパン合同会社が提供するオンラインセミナーシリーズです
- AWS の技術担当者が、AWS の各サービスやソリューションについてテーマごとに動画を公開します
- 以下の URL より、過去のセミナー含めた資料などをダウンロードすることができます
 - <https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-service-cut/>
 - <https://www.youtube.com/playlist?list=PLzWGOASvSx6FlwIC2X1nObr1KcMCBBlqY>



ご感想は X (Twitter) へ！ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では資料作成時点のサービス内容および価格についてご説明しています。AWS のサービスは常にアップデートを続けているため、最新の情報は AWS 公式ウェブサイト (<https://aws.amazon.com/>) にてご確認ください
- 資料作成には十分注意しておりますが、資料内の価格と AWS 公式ウェブサイト記載の価格に相違があった場合、AWS 公式ウェブサイトの価格を優先とさせていただきます
- 価格は税抜表記となっています。日本居住者のお客様には別途消費税をご請求させていただきます
- 技術的な内容に関しましては、有料の [AWS サポート窓口](#) へお問い合わせください
- 料金面でのお問い合わせに関しましては、[カスタマーサポート窓口](#) へお問い合わせください (マネジメントコンソールへのログインが必要です)

Thank you!