AWS Black Belt Online Seminar

Karpenter Basic

多田 慎也

Solutions Architect
2023/12

自己紹介

名前：多田 慎也

所属：アマゾン ウェブ サービス ジャパン
技術統括本部 西日本ソリューショングループ
ソリューションアーキテクト

経歴：インフラエンジニア@SIer

好きなAWSサービス： Amazon ECS , AWS Fargate

本セミナーの対象者とゴール

対象者

- AWS 上で Kubernetes を利用している方、もしくは利用を検討している方
- Kubernetes 上の ノードグループの管理が複雑と感じている方
- Karpenter について基本を学びたい方

ゴール

- Karpenter が解決する課題や機能概要を知っていただく
- ご自身が管理するワークロードに対して Karpenter が有用なものとなるか参考にしていただく

アジェンダ

1. Why Karpenter
2. Karpenter 概要
3. Karpenter 使い方
4. まとめ

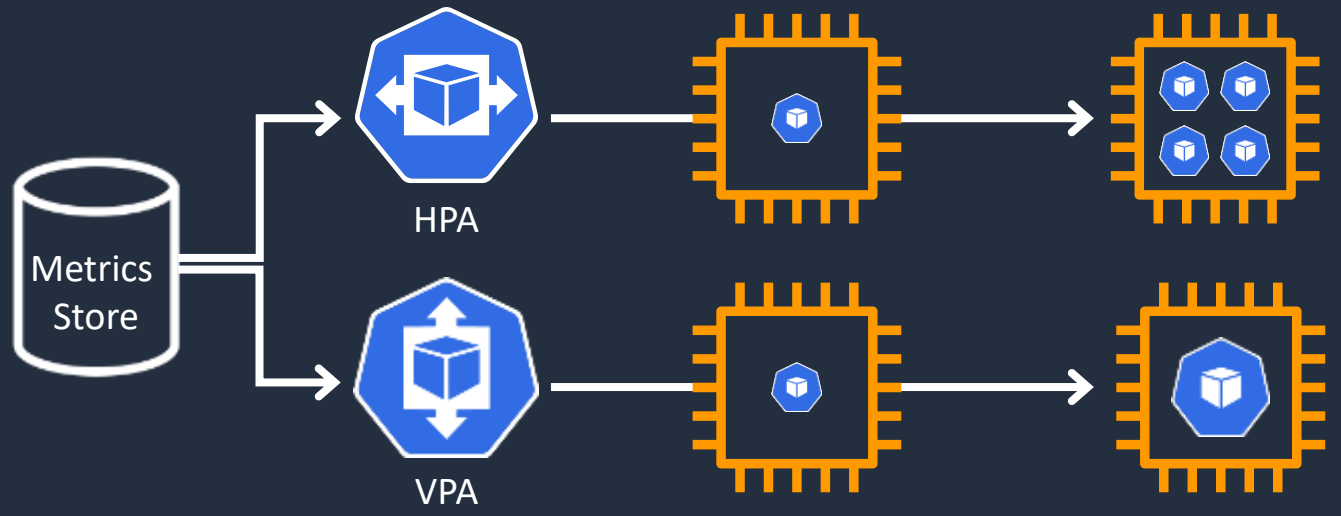
Why Karpenter

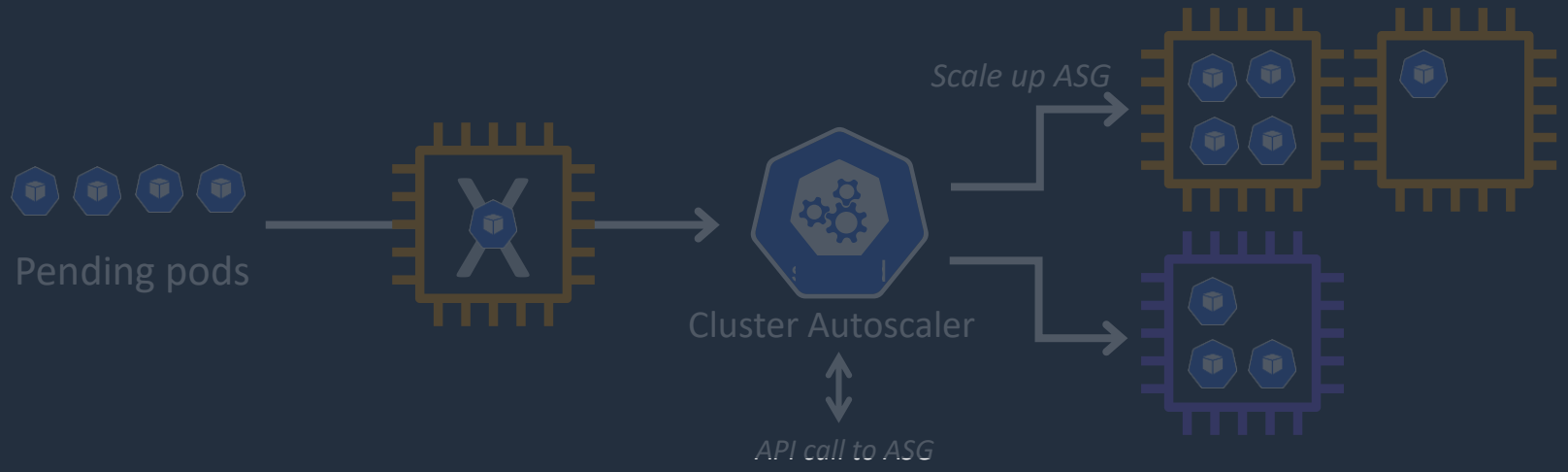
Karpenter とは

- 柔軟で高パフォーマンスのオープンソース Kubernetes クラスターオートスケーラー
  - 適切なコンピューティングリソースを動的に決定
  - 必要に応じてコンピューティングリソースを追加削除

Kubernetes のオートスケーリング

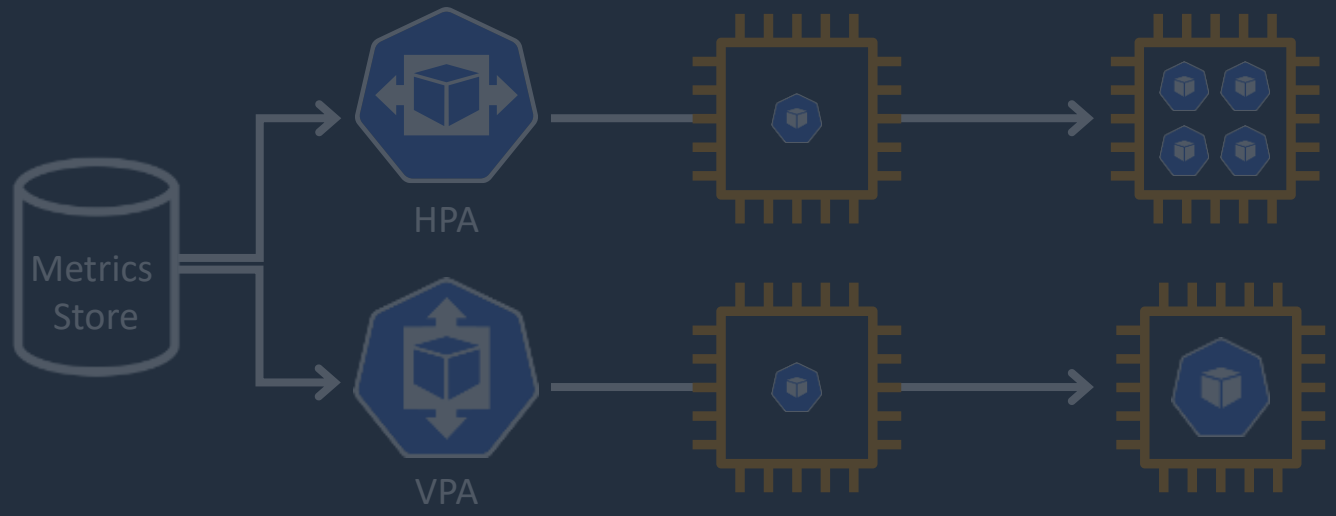
1. Horizontal Pod Autoscaling (HPA)
2. Vertical Pod Autoscaling (VPA)
3. Cluster Autoscaler



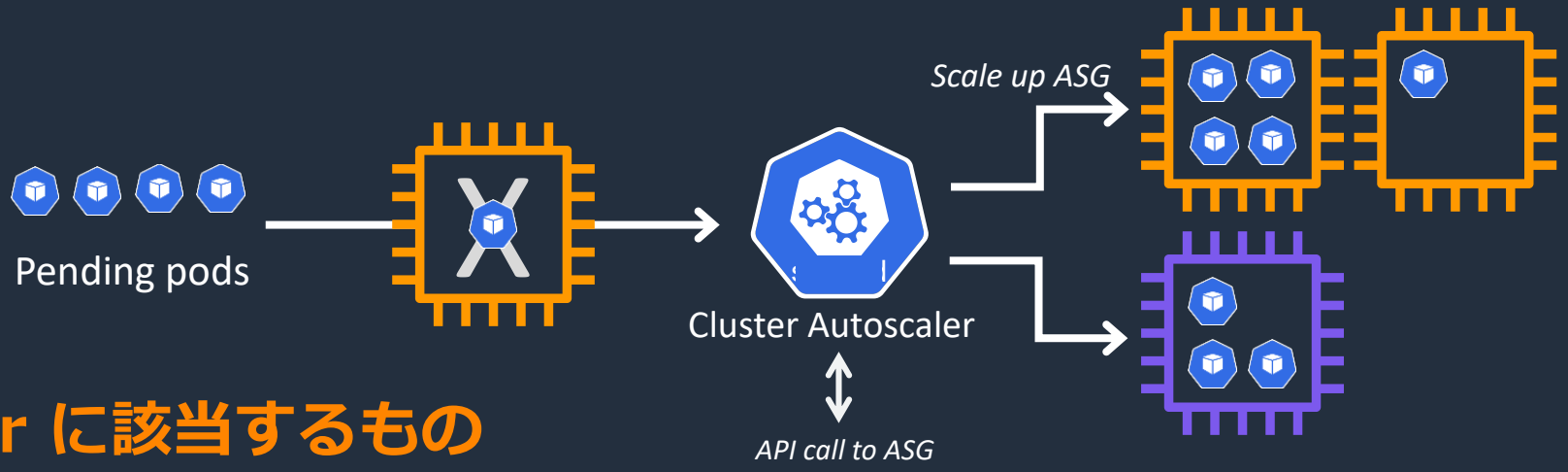


Kubernetes のオートスケーリング

1. Horizontal Pod Autoscaling (HPA)
2. Vertical Pod Autoscaling (VPA)



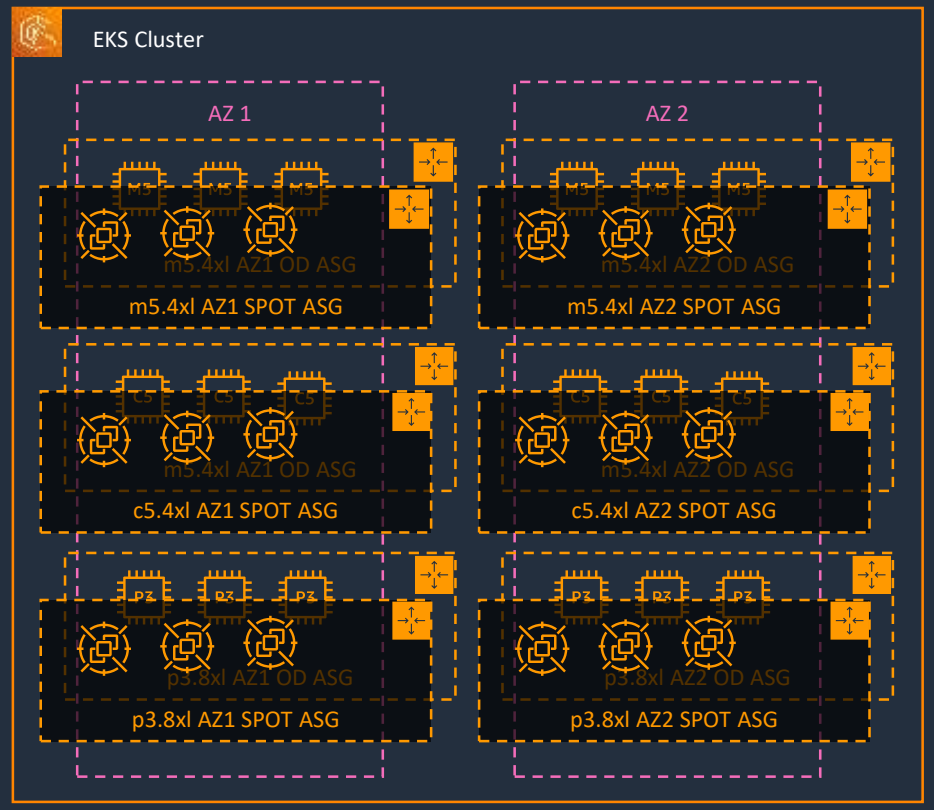
3. Cluster Autoscaler



Karpenter は
Cluster Autoscaler に該当するもの

Cluster Autoscaling の課題

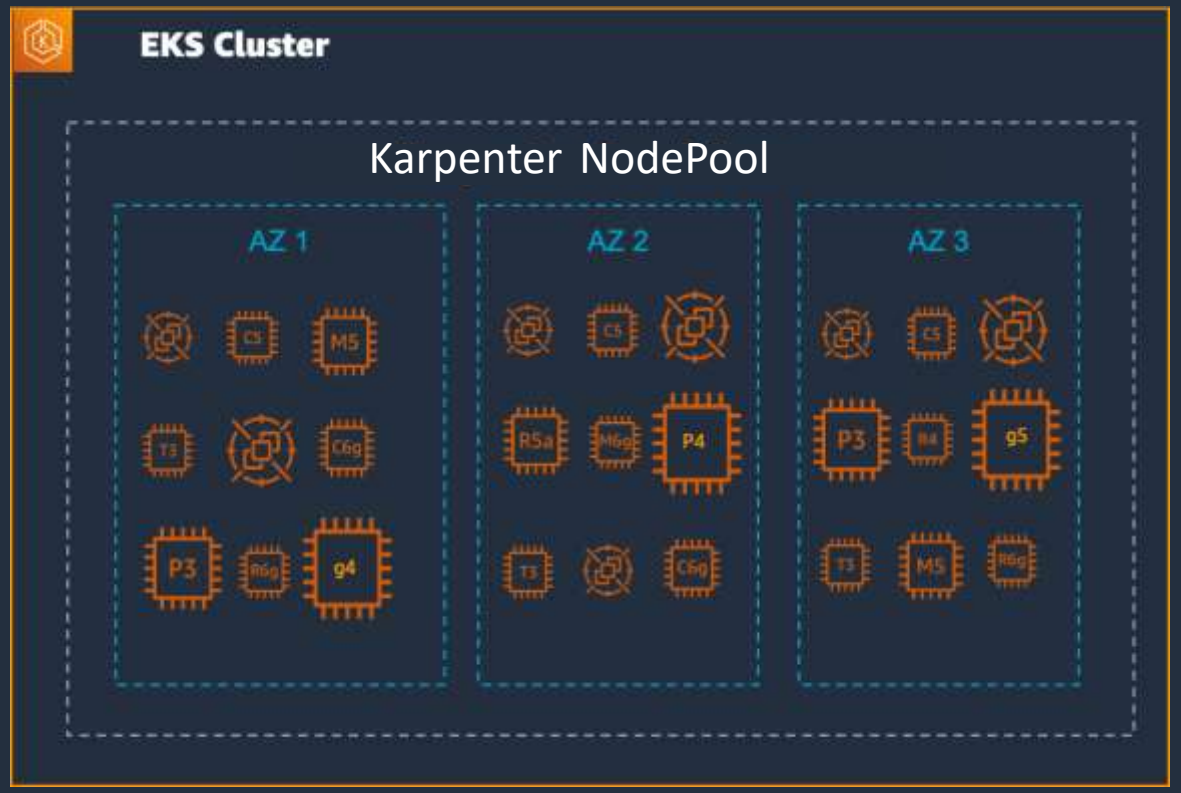
可用性、コストを考慮した最適な設定の難易度が高い



- 多様な Pod に対応するためには、
  多数のノードグループを構成する必要があり、
  お客様からつらいとのお声を聞いている
- インスタンスサイズ別にノードグループの分割の必要
- 購入オプション別にノードグループの分割の必要
- PV (EBS) 利用時には AZ 別にノードグループの分割の必要

Karpenter による課題解決

煩雑なノードグループの管理が不要

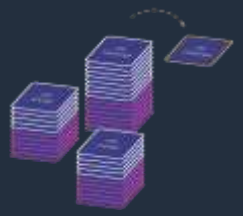


アプリケーションファースト

ノード起動は *Pod* の要求に合わせて行われます

シンプルな設定

オンデマンド・スポットの購入オプションの組み合わせ、また複数のインスタンスタイプの指定を簡単に

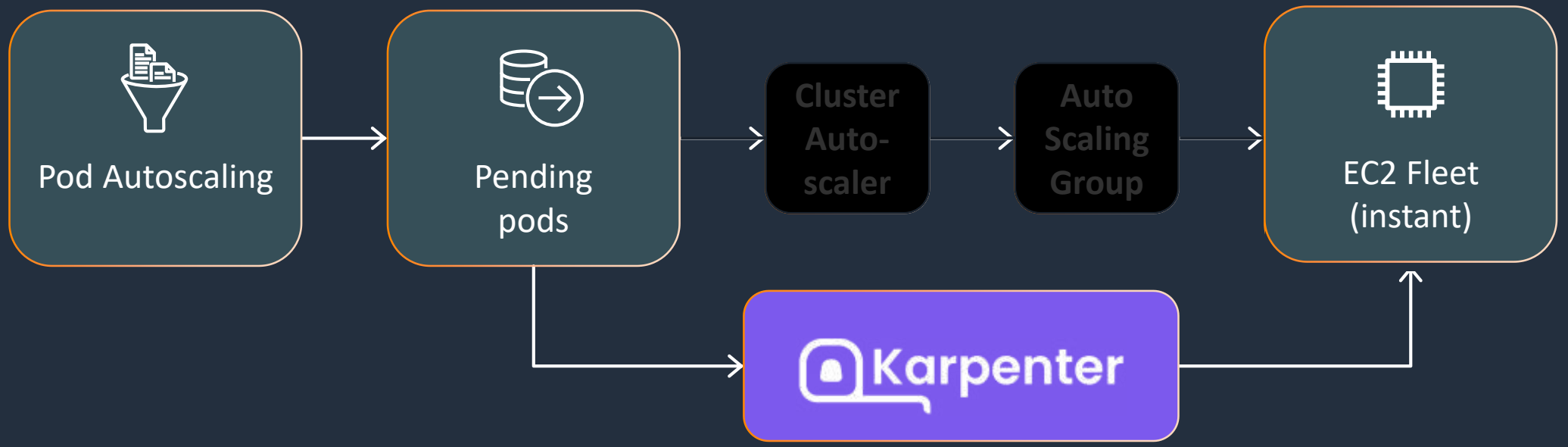


スポットとオンデマンドを柔軟に

購入オプションの多様化を簡単に

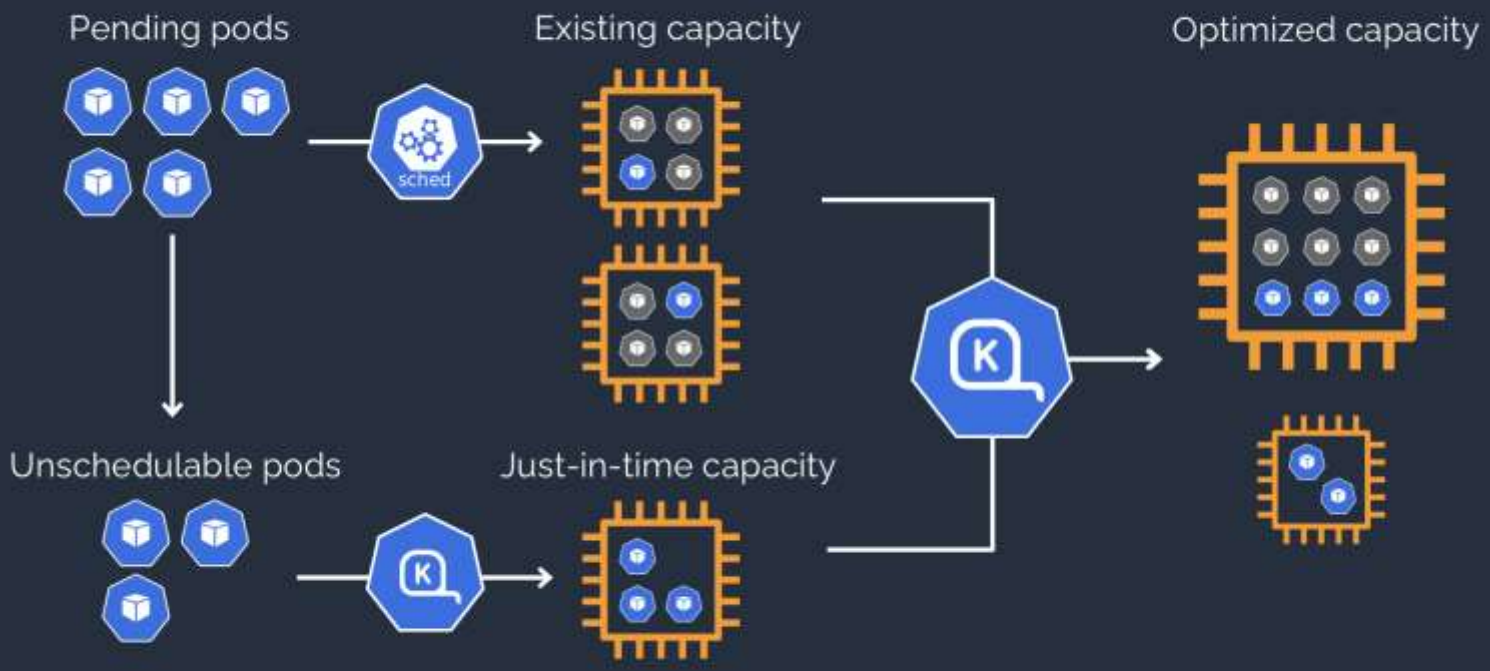
Karpenter 概要

Karpenter の仕組み



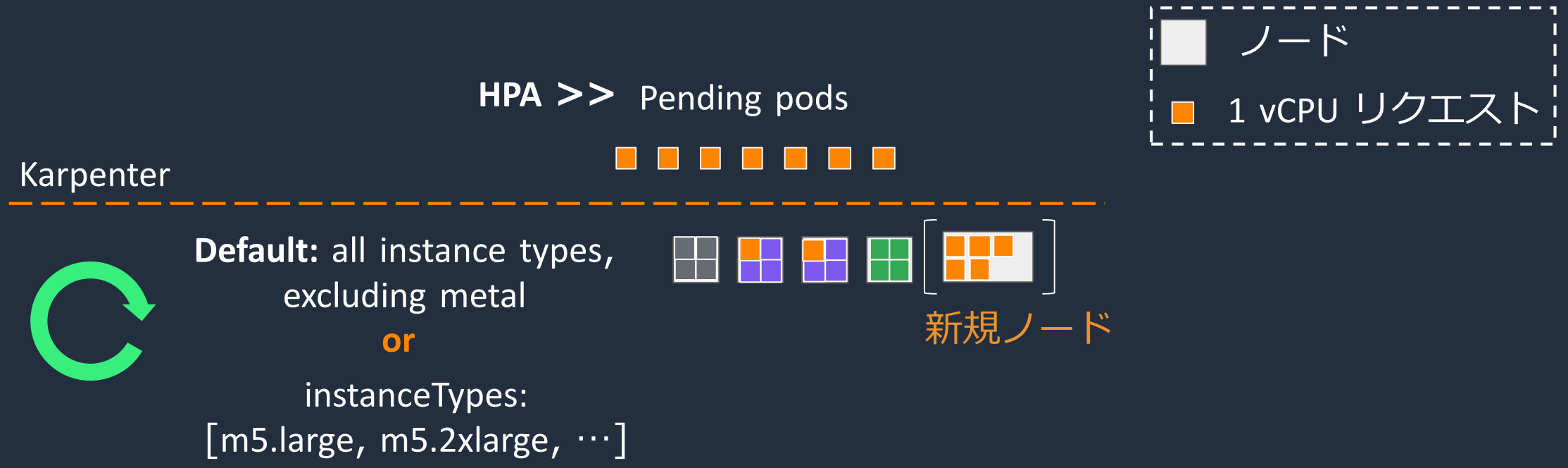
ノードのオーケストレーション責任を
単一システム (Kubernetes) 内に統合

Karpenter の仕組み



- Unschedulable な Pod を検知して適切なサイズのノードを起動
- Kubernetes の スケジューリングに関する constraints と連携
- キャパシティ最適化
  (使用されていないノードの削除、ノードの統合、安価なノードへ置換)

Karpenter のスケールアウト



Karpenter のスケールイン

Karpenter

Expiration：
クラスターにノードが存在できる有効期限
セキュリティやメモリリークなど長時間の起動問題軽減するためノードを定期的に置換

Consolidation：
クラスターのコスト効率を高めるためノード削除、統合や置換を実施
- 空ノードを削除
- 複数ノードを統合
- 安価なノードに置き換え

- 削除 / 統合対象のノードを決定後、対象ノードに新しく Pod がスケジューリングされないように設定
- 対象ノード上の Pod を退避し、安全にノードを削除

Karpenter 使い方

Karpenter の利用方法

1. Karpenter のインストール
   - Karpenter は Custom Resources として、Helm chart でインストールされる。
   - ノードのプロビジョニング権限に、AWS IAM Roles for Service Accounts (IRSA) を必要とする。

2. NodePool の作成
   - NodePool は Karpenter が作成可能なノード制約の設定
   - CRD NodePool と EC2NodeClass の設定が必要

Getting Started with Karpenter
https://karpenter.sh/docs/getting-started/getting-started-with-karpenter/

NodePoolとEC2NodeClassの設定例

```yaml
apiVersion: karpenter.sh/v1beta1
kind: NodePool
metadata:
  name: default
spec:
  template:
    spec:
      requirements:
        - key: kubernetes.io/arch
          operator: In
          values: ["amd64"]
        - key: kubernetes.io/os
          operator: In
          values: ["linux"]
        - key: karpenter.sh/capacity-type
          operator: In
          values: ["spot"]
        - key: karpenter.k8s.aws/instance-category
          operator: In
          values: ["c", "m", "r"]
        - key: karpenter.k8s.aws/instance-generation
          operator: Gt
          values: ["2"]
      nodeClassRef:
        name: default
  limits:
    cpu: 1000
  disruption:
    consolidationPolicy: WhenEmpty
    consolidateAfter: 60s
    expireAfter: 168h
---
apiVersion: karpenter.k8s.aws/v1beta1
kind: EC2NodeClass
metadata:
  name: default
spec:
  amiFamily: AL2 # Amazon Linux 2
  role: "KarpenterNodeRole-${CLUSTER_NAME}"
  subnetSelectorTerms:
    - tags:
        karpenter.sh/discovery: "${CLUSTER_NAME"
  securityGroupSelectorTerms:
    - tags:
        karpenter.sh/discovery: "${CLUSTER_NAME}"
```

NodePool 設定

ノード要件

クラスターの合計サイズの制約

ノード削除・統合の設定

EC2NodeClas 設定

ノードの AMI (Custom 指定可能)

ノードの IAM Role

ノードの Subnet

ノードの SG

・Nodepool設定(詳細)： https://karpenter.sh/docs/concepts/nodepools/
・NodeClasses設定(詳細) ： https://karpenter.sh/docs/concepts/nodeclasses/
・その他の設定例：https://github.com/aws/karpenter/tree/main/examples

ノード要件の設定(詳細)①

```
spec:
    requirements:
      - key: kubernetes.io/arch
        operator: In
        values: ["amd64"]
      - key: kubernetes.io/os
        operator: In
        values: ["linux"]
      - key: karpenter.sh/capacity-type
        operator: In
        values: ["on-demand"]
      - key: karpenter.k8s.aws/instance-category
        operator: In
        values: ["c", "m", "r"]
      - key: karpenter.k8s.aws/instance-generation
        operator: Gt
        values: ["2"]
      - key: "karpenter.k8s.aws/instance-cpu"
        operator: Gt
        values: ["3"]
```

requirements: [] を設定すると
すべてのインスタンスタイプが対象

・ハードウェアアーキテクチャ

例	amd64を選択
備考	amd64, arm64を設定可能

・OS

例	linuxを選択
備考	linux, windowsを設定可能

・EC2購入オプション

例	on-demandを選択
備考	spot, on-demandを設定可能

Well-Known Labels　※設定可能なkey
https://karpenter.sh/docs/concepts/scheduling/#well-known-labels

ノード要件の設定(詳細)②

```yaml
spec:
  requirements:
    - key: kubernetes.io/arch
      operator: In
      values: ["amd64"]
    - key: kubernetes.io/os
      operator: In
      values: ["linux"]
    - key: karpenter.sh/capacity-type
      operator: In
      values: ["on-demand"]
    - key: karpenter.k8s.aws/instance-category
      operator: In
      values: ["c", "m", "r"]
    - key: karpenter.k8s.aws/instance-generation
      operator: Gt
      values: ["2"]
    - key: "karpenter.k8s.aws/instance-cpu"
      operator: Gt
      values: ["3"]
```

※1 Instance Types
https://karpenter.sh/docs/reference/instance-types/

- EC2インスタンスタイプカテゴリー

例	インスタンスタイプc,m,rから選択
備考	i,tなど他のタイプも指定可能(※1参照) 以下のkeyでより細かく設定も可能 karpenter.k8s.aws/instance-family node.kubernetes.io/instance-type

- EC2インスタンスタイプの世代

例	世代が2より上(3以上)から選択
備考	4,5,6なども可能(※1参照)

- EC2インスタンスのCPUサイズ

例	CPUが 3 より上 (4 以上) のインスタンスを選択
備考	以下のような設定も可能 operator: In values: [“4”, “8”, “16”, “32”]

ノード削除・統合の設定①

```
spec:
 disruption:
   consolidationPolicy: WhenEmpty
   consolidateAfter: 60s
   expireAfter: 168h
```

・コスト最適化のためのノード削除やノード統合設定

consolidationPolicy

例	**WhenEmpty** **podがないノードを削除(Deamon pod除く)**
備考	・以下のどちらかを指定 WhenEmpty、 WhenUnderutilized ・WhenUnderutilized (デフォルト)の挙動 利用率が低いノードを特定し、コスト削減可能な場合、ノード削除または置換を実施

consolidateAfter

例	**ノード削除や統合可能と判断してから60s後に** **アクション実施までの時間**
備考	・Never設定で、 consolidation を無効化可能 ・現在、 consolidationPolicy が WhenEmpty の場合のみ設定可能

ノード削除・統合の設定②

```
spec:
 disruption:
    consolidationPolicy: WhenEmpty
    consolidateAfter: 60s
    expireAfter: 168h
```

・ノードがクラスターから削除されるまでの有効期限

expireAfter

例	**168h(1週間)後にノード削除**
備考	・デフォルト値は720h(30日)後にノード削除 ・Never設定で、有効期限を無効化可能

まとめ

まとめ

- Karpenter は AWS が開発した Kubernetes 向けの新しいクラスターオートスケーラーです。
- Karpenter は既存の Cluster Autoscalerと比べて、複雑なノードグループの設定が不要です。
- Karpenter がプロビジョニングするノードは Nodepool の設定で、ノードの要件を細かく指定することができ、 Pod の要件に応じて動的にノードを選択します。
- Karpenter はコストを最適化するために、Pod をアクティブに移動させ、ノードを削除するか、より安価なノードに置き換えるコンソリデーション機能もサポートしています。
- Karpenter は、複雑なクラスター構成を意識することなく、シンプル且つコスト効率の高いオートスケーリングを実現できます。

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=PLzWGOASvSx6FIwIC2X1nObr1KcMCBBlqY

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!