



SUMMIT  
Tokyo

B1-01, H3-04

# EC2スポットインスタンスのすべて

滝口 開資 (Haruyoshi Takiguchi)  
ソリューションアーキテクト  
アマゾンウェブサービスジャパン株式会社

# 自己紹介

## 滝口 開資 (はるよし)

ソリューションアーキテクト - EC2スポットインスタンススペシャリスト  
日本市場でのEC2スポットインスタンス技術担当

### 普段の業務

スポットインスタンスを活用するお客様を技術面からサポート

### 好きなAWSサービス

- Amazon EC2 Auto Scaling
- AWS Support




# 本セッションの目的

- EC2スポットインスタンスが分かるようになる
- EC2スポットインスタンスがこわくなる
- 次のポイントをカバーしてお伝えします
  - EC2スポットインスタンスのつかいどころ、押さえておきたい基本事項
  - 活用する上で必ず役に立つ実践情報
  - 使いやすくなった機能改善

# はじめに

# 今年2月、ある記事が話題になりました

 **ITmedia**  
**エンタープライズ**

[DX×ビジネス](#) [DX×組織](#) [セキュリティ](#) [Transborder](#) [働き方改革](#) [俺たちの情シス](#) [記事一覧](#)

[ホワイトペーパー](#) [オルタナティブ・ブログ](#) [用語辞典](#)

週末エンプラこぼれ話：

## 囲碁AIブームに乗って、若手棋士の間で「AWS」が大流行 その理由とは？ (1/4)

人間の能力をAIが完全に上回りつつある「囲碁」の世界。最近では、AIを活用した研究を行う棋士も増えているそうだが、その裏側でAWSが若手棋士の中で大流行しているという。一体何が起きているのだろうか。

🕒 2019年02月22日 08時00分 公開 [池田憲弘, ITmedia]

ITmedia エンタープライズ「囲碁AIブームに乗って、若手棋士の間で「AWS」が大流行 その理由とは？」(2019年02月22日)  
<https://www.itmedia.co.jp/enterprise/articles/1902/22/news006.html>



## 「予想よりも“30年”早く、AIが人間を超えてしまった」

この“AWSブーム”を仕掛けたのは大橋拓文六段。コンピュータ囲碁を活用した研究に熱心なことで知られ、囲碁AIの解説本『[よくわかる囲碁AI大全](#)』も出版している。大橋さんが囲碁AIに興味を持ったのは2010年ごろで、当時はAIがプロ棋士に勝つことなど想像できない状態だったという。

「もともと、囲碁AIは老後の趣味にしようと思って研究を始めたんです。自分が引退するくらいの頃に人間を上回るだろうと。そしたら、想像よりも30年くらい早く人間を超えてしまいました……。その頃の囲碁AIは『[モンテカルロ法](#)』の適用によってブレークスルーが起きており、アマチュアの高段者くらいのレベルに達していました」



大橋拓文六段

こうして大橋さんは2018年7月にAWSを導入。分からないことだらけだったが、東京・目黒にあるAWS開発者向けのスペース「AWS Loft Tokyo」に足しげく通い、ASKコーナーで質問を繰り返した。さまざまな開発者と話し合い、Amazon囲碁部ともつながり、スポットインスタンス（※）担当だった部長のアドバイスを得て、利用額も大幅に抑えられた。現在は「月に数千円から1万円程度かかっている」（大橋さん）という。

※Amazon EC2の機能。AWSサーバ上で使われていない（余っている）EC2インスタンスに対し、入札制で一時利用を行う。自らの提示額よりも高い価格で入札されると、すぐに利用を止められるリスクもあるが、一般的な「オンデマンド」制に比べて平均で7～9割引きになっている

次ページ

若手棋士が、次々とAWSのインスタンスを立てるように

前のページへ

1

2

3

4

次のページへ



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. ストレージの追加 5. タグの追加 6. セキュリティグループの設定 7. 確認

### ステップ 3: インスタンスの詳細の設定

要件に合わせてインスタンスを設定します。同じ AMI からの複製インスタンス作成や、より低料金を実現するためのスポットインスタンスのリクエスト、インスタンスへのアクセス管理ロール割り当てなどを行うことができます。

インスタンス数 ①  Auto Scaling グループに作成する ①

購入のオプション ① ☐ スポットインスタンスのリクエスト

ネットワーク ①

サブネット ①

自動割り当てパブリックIP ☐ ネット

配置グループ ① ☐ インスタ

Capacity Reservation ①

IAM ロール ①

CPU options ① ☐ Specify C

シャットダウン動作 ①


疎性保護の有効化 ① ☐ 適切な

モニタリング ① ☐ CloudWatch

EBS 最適化インスタンス ① ☐ EBS 最適化イン

次の手順: ストレージの追加

キャンセル 戻る **確認と作成**



インスタンスタイプを選んだら  
詳細設定に入ります。

ここで**重要ポイント!**  
スポットインスタンス  
のリクエスト  
を押し忘れないように**注意!**

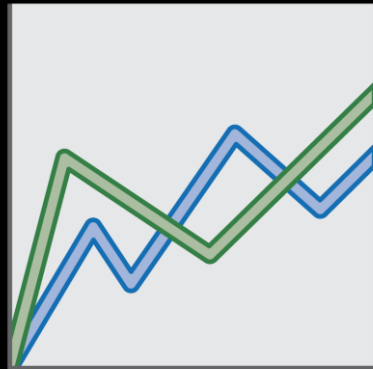
そして画面右端の  
次の手順: ストレージの追加  
の白いボタンを押す。

女流棋聖の上野愛咲美さん（現在17歳）が作ったAWSの使い方マニュアルの一部。こうした資料も研究会のメンバーに共有されている

# Amazon EC2の購入オプション

## オンデマンドインスタンス

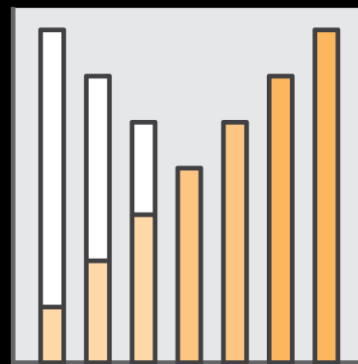
長期コミット無し、使用分への支払い(**秒単位/時間単位**)。  
Amazon EC2の定価



スパイクするような  
ワークロード

## リザーブドインスタンス

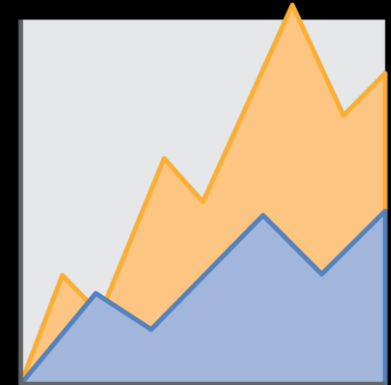
1年/3年の長期コミットをする  
代わりに**大幅なディスカウント**  
価格



一定の負荷の見通しがある  
ワークロード

## スポットインスタンス

Amazon EC2の空きキャパシティを活用。**最大90%値引き**



中断に強く、かつ様々な  
インスタンスタイプを  
活用できるワークロード

EC2インスタンスとしての性能に違いはない

# スポットインスタンス 基礎編

# スポットインスタンス 基礎編

- スポットインスタンスの起動方法
- スポットインスタンスのライフサイクル
- スポットインスタンスのしくみと価格

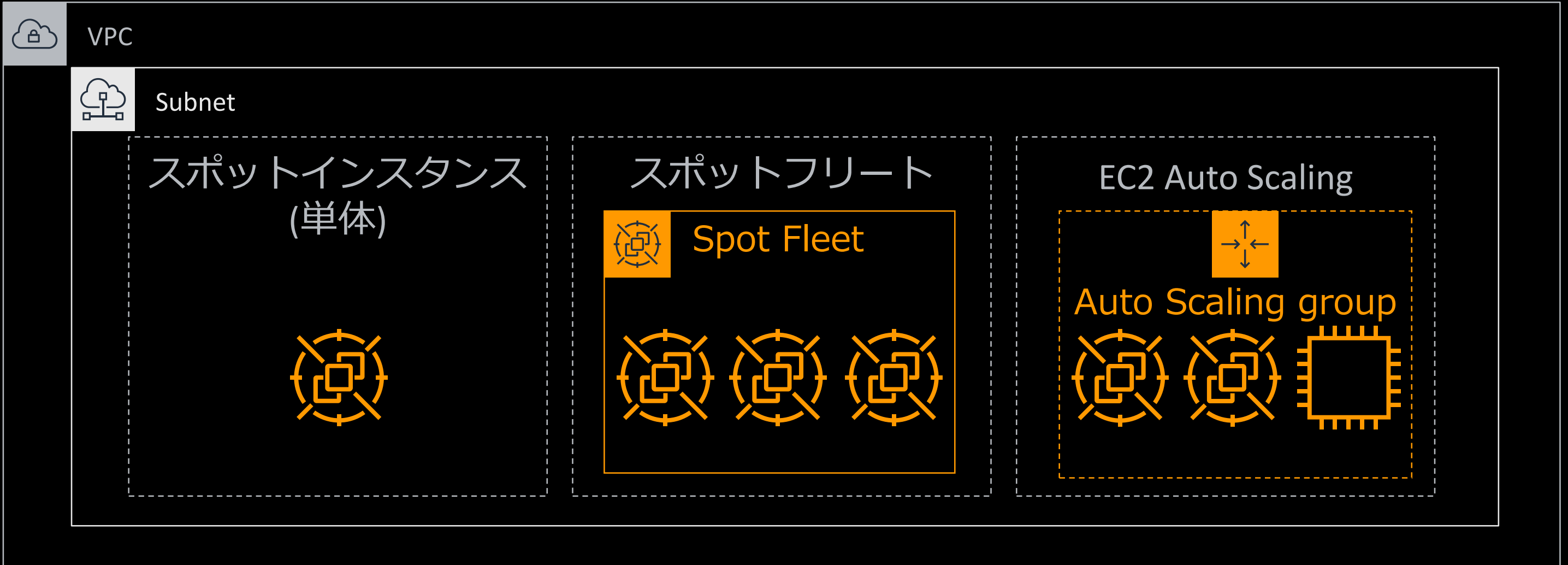
# スポットインスタンスの起動方法



# スポットインスタンスを起動する4つの方法

- スポットインスタンス(単体)
  - クイックに起動するときにオススメ
- スポットフリート
  - スポットインスタンスをまとめて起動・管理するときにオススメ
  - 中断のテストが可能
- EC2 Auto Scaling
  - スポットインスタンスをまとめて起動・管理するときにオススメ
  - Auto Scalingの各種機能を使いたいときはこちら
- EC2フリート
  - 同期処理や最小起動台数保証など、上の3つよりもきめ細やかなコントロールが可能

# 本でご紹介する起動方法



# スポットインスタンス単体の起動



# スポットインスタンス(単体)の起動方法(1)



- 「インスタンス」→「インスタンスの作成」

# スポットインスタンス(単体)の起動方法(2)

1. AMI の選択

2. インスタンスタイプの選択

3. インスタンスの設定

4. ストレージの追加

5. タグの追加

6. セキュリティグループの設定

7. 確認

手順 1: Amazon マシンイメージ (AMI)

キャンセルして終了

AMI は、インスタンスの作成に必要なソフトウェア構成 (OS、アプリケーションサーバー、アプリケーション) を含むテンプレートです。AMI は、AWS が提供するもの、ユーザーコミュニティが提供するもの、または AWS Marketplace に掲載されているものを選択できます。独自の AMI のいずれかを選択することもできます。

検索用語を入力して AMI を検索します (「Windows」 など)。

クイックスタート


38 AMI 中の 1 ~ 38

マイ AMI

AWS Marketplace

コミュニティ AMI

☐ 無料利用枠のみ ⓘ




Amazon Linux

無料利用枠の対象

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-015954d5e5548d13b

Amazon Linux 2 には 5 年間のサポートが含まれます。Amazon EC2、systemd 219、GCC 7.3、Glibc 2.26、Binutils 2.29.1 で最適なパフォーマンスを発揮できるように調整された Linux カーネル 4.14、および、追加の最新のソフトウェアパッケージを提供します。

ルートデバイスタイプ: ebs    仮想化タイプ: hvm    ENA 有効: はい



Amazon Linux

無料利用枠の対象

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-063dd30adbb186909

Amazon Linux AMI は、AWS がサポートする EBS-backed イメージです。デフォルトのイメージには、AWS コマンドラインツール、Python、Ruby、Perl、および Java が含まれま

選択

64 ビット (x86)

選択

64 ビット (x86)

- 手順 1: Amazon マシンイメージ (AMI)を選択



# スポットインスタンス(単体)の起動方法(3)

1. AMI の選択

2. インスタンスタイプの選択

3. インスタンスの設定

4. ストレージの追加

5. タグの追加

6. セキュリティグループの設定

7. 確認

## 手順 2: インスタンスタイプの選択

Amazon EC2 では、異なるユースケースに合わせて最適化されたさまざまなインスタンスタイプが用意されています。インスタンスは、アプリケーションを実行できる仮想サーバーです。インスタンスタイプはさまざまな CPU、メモリ、ストレージ、ネットワークキャパシティの組み合わせによって構成されているため、使用するアプリケーションに合わせて適切なリソースの組み合わせを柔軟に選択できます。インスタンスタイプおよびそれをコンピューティングのニーズに適用する方法に関する [詳細はこちら](#)。

フィルター条件:

すべてのインスタンスタイプ

現行世代

列の表示/非表示

現在選択中: t2.micro (可変 ECU, 1 vCPU, 2.5 GHz, Intel Xeon Family, 1 GiB メモリ, EBS のみ)

	ファミリー	タイプ	vCPU	メモリ (GiB)	インスタンスストレージ (GB)	EBS 最適化利用	ネットワークパフォーマンス	IPv6 サポート
<input type="checkbox"/>	汎用	t2.nano	1	0.5	EBS のみ	-	低から中	はい
<input checked="" type="checkbox"/>	汎用	t2.micro 無料利用枠の対象	1	1	EBS のみ	-	低から中	はい
<input type="checkbox"/>	汎用	t2.small	1	2	EBS のみ	-	低から中	はい

- 手順 2: インスタンスタイプの選択

# スポットインスタンス(単体)の起動方法(4)

1. AMI の選択

2. インスタンスタイプの選択

3. インスタンスの設定

4. ストレージの追加

5. タグの追加

6. セキュリティグループの設定

7. 確認

## 手順 3: インスタンスの詳細の設定

要件に合わせてインスタンスを設定します。同じ AMI からの複数インスタンス作成や、より低料金を実現するためのスポットインスタンスのリクエスト、インスタンスへのアクセス管理ロール割り当てなどを行うことができます。

インスタンス数



1

Auto Scaling グループに作成する



購入のオプション



☐ スポットインスタンスのリクエスト

ネットワーク



vpc-09ea3873 (デフォルト)



[新しい VPC の作成](#)

サブネット



優先順位なし (アベイラビリティゾーンのデフォルト)



[新しいサブネットの作成](#)

自動割り当てパブリック IP



サブネット設定を使用 (有効)



配置グループ



☐ インスタンスをプレイスメントグループに追加します。

キャパシティの予約



開く



[新しいキャパシティ予約の作成](#)

- 手順 3: インスタンスの詳細の設定

# スポットインスタンス(単体)の起動方法(4)

1. AMI の選択

2. インスタンスタイプの選択

3. インスタンスの設定

4. ストレージの追加

5. タグの追加

6. セキュリティグループの設定

7. 確認

## 手順 3: インスタンスの詳細の設定

要件に合わせてインスタンスを設定します。同じ AMI からの複数インスタンス作成や、より低料金を実現するためのスポットインスタンスのリクエスト、インスタンスへのアクセス管理ロール割り当てなどを行うことができます。

インスタンス数



1

Auto Scaling グループに作成する



購入のオプション



☐ スポットインスタンスのリクエスト

ネットワーク



vpc-09ea3873 (デフォルト)



[新しい VPC の作成](#)

サブネット



優先順位なし (アベイラビリティゾーンのデフォルト)



[新しいサブネットの作成](#)

自動割り当てパブリック IP



サブネット設定を使用 (有効)



配置グループ



☐ インスタンスをプレイスメントグループに追加します。

キャパシティの予約



開く



[新しいキャパシティ予約の作成](#)

- 手順 3: インスタンスの詳細の設定

# スポットインスタンス(単体)の起動方法(4)

1. AMI の選択

2. インスタンスタイプの選択

3. インスタンスの設定

4. ストレージの追加

5. タグの追加

6. セキュリティグループの設定

7. 確認

## 手順 3: インスタンスの詳細の設定

要件に合わせてインスタンスを設定します。同じ AMI からの複数インスタンス作成や、より低料金を実現するためのスポットインスタンスのリクエスト、インスタンスへのアクセス管理ロール割り当てなどを行うことができます。

インスタンス数



1

Auto Scaling グループに作成する



購入のオプション



☒ スポットインスタンスのリクエスト

現在の価格



アベイラビリティゾーン	現在の価格
us-east-1a	\$0.0035
us-east-1b	\$0.0035
us-east-1c	\$0.0035
us-east-1d	\$0.0035
us-east-1e	\$0.0035
us-east-1f	\$0.0035

オプションが  
追加表示される

- 手順 3: 「スポットインスタンスのリクエスト」にチェック



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. ストレージの追加 5. タグの追加 6. セキュリティグループの設定 7. 確認

### ステップ 3: インスタンスの詳細の設定

要件に合わせてインスタンスを設定します。同じ AMI からの複製インスタンス作成や、より低料金を実現するためのスポットインスタンスのリクエスト、インスタンスへのアクセス管理ロール割り当てなどを行うことができます。

インスタンス数 ①  Auto Scaling グループに作成する ①

購入のオプション ① ☐ スポットインスタンスのリクエスト

ネットワーク ①

サブネット ①

自動割り当てパブリックIP ☐ ネット

配置グループ ① ☐ インスタ

Capacity Reservation ①

IAM ロール ①

CPU options ① ☐ Specify C

シャットダウン動作 ①

疎性保護の有効化 ① ☐ 適切な

モニタリング ① ☐ Amazon

EBS 最適化インスタンス ① ☐ EBS 最適化

追加料金が適用

そして画面右端の  
次の手順: ストレージの追加  
の白いボタンを押す。

インスタンスタイプを選んだら  
詳細設定に入ります。

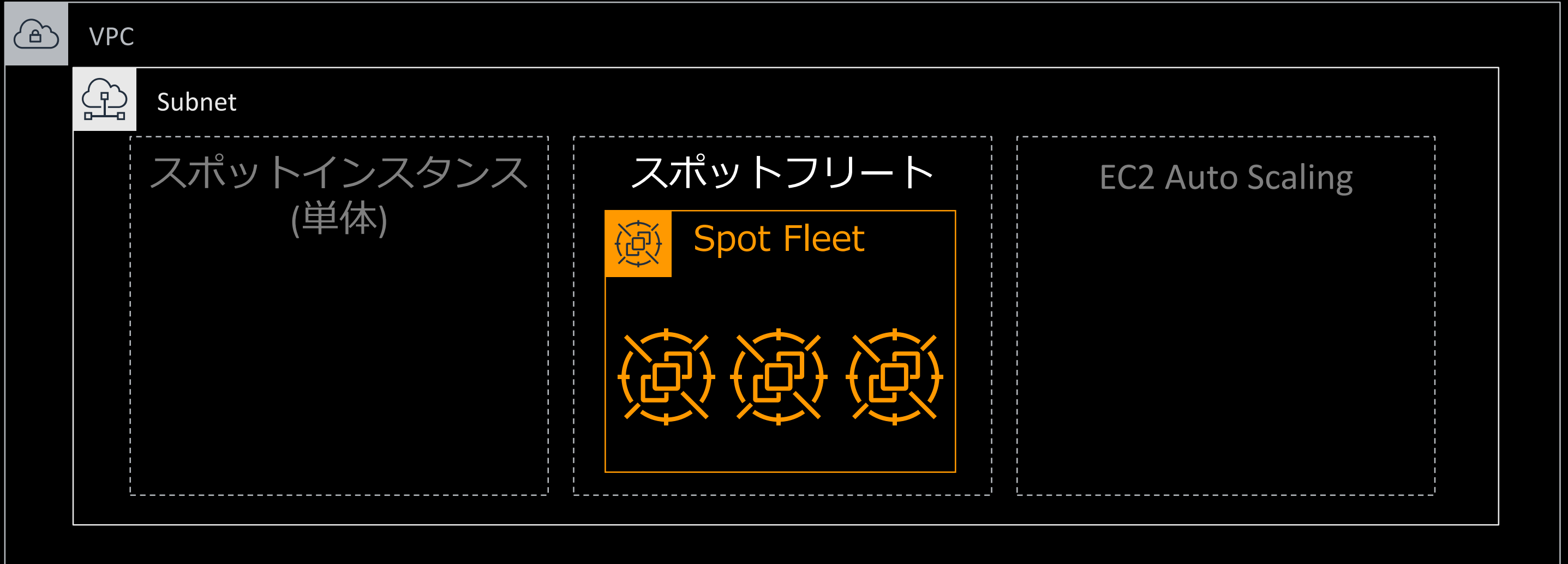
ここで**重要ポイント!**  
スポットインスタンス  
のリクエスト  
を押し忘れないように**注意!**

キャンセル 戻る **確認と作成** 次の手順: ストレージの追加

女流棋聖の上野愛咲美さん（現在17歳）が作ったAWSの使い方マニュアルの一部。こうした資料も研究会のメンバーに共有されている



# スポットフリートの起動



# スポットフリートの起動方法(1)

EC2 ダッシュボード

イベント

タグ

レポート

制限

インスタンス

インスタンス

起動テンプレート

**スポットリクエスト**

リザーブドインスタンス

専用ホスト

スケジュール済みインスタンス

スポットインスタンスのリクエスト

アクション

価格設定履歴

Savings Summary

リクエストタイプ: all

状態: all

キーワードによる検索

リクエスト ID

リクエストタ...

インスタンスタ...

状態

容量

ステータス

現在、このリージョンにスポットリクエストはありません。

EC2 スポットインスタンスを初めて使用する場合は、「[開始方法](#)」ページにアクセスしてください。

スポットインスタンスを起動するには、スポットインスタンスのリクエスト ボタンをクリックします。

スポットインスタンスのリクエスト

詳細を表示するには、上記から 1 つのスポットリクエストを選択します。

- 「スポットリクエスト」→「スポットインスタンスのリクエスト」

# スポットフリートの起動方法(2)

EC2 > スポットリクエスト > スポットインスタンスのリクエスト

## スポットインスタンスのリクエスト

アプリケーションまたはタスクのニーズについてお知らせください

ジョブに最も適切なコンピューティング性能を AWS が確認できるように、アプリケーションまたはタスクのニーズに最も一致するものを選択します。

☒ **Load balancing**

**workloads**

Launch instances of the same size, in any Availability Zone. Good for running web services.

☐ **Flexible workloads**

Launch instances of any size, in any Availability Zone. Good for running batch and CI/CD jobs.

☐ **Big data workloads**

Launch instances of any size, in a single Availability Zone. Good for MapReduce jobs.

☐ **Defined duration**

**workloads**

Launch instances into a Spot block for 1 to 6 hours.

1 時間

ワークロードに応じたインスタンスタイプの種類を提案してくれる

- Load balancing: 近しいサイズのインスタンスタイプから選択
- Flexible workloads: サイズの異なるインスタンスタイプから選択

# スポットフリートの起動方法(2)

EC2 > スポットリクエスト > スポットインスタンスのリクエスト

## スポットインスタンスのリクエスト

アプリケーションまたはタスクのニーズについてお知らせください

ジョブに最も適切なコンピューティング性能を AWS が確認できるように、アプリケーションまたはタスクのニーズに最も一致するものを選択します。

☒ **Load balancing**  
**workloads**  
Launch instances of the same size, in any Availability Zone. Good for running web services.

☐ **Flexible workloads**  
Launch instances of any size, in any Availability Zone. Good for running batch and CI/CD jobs.

☐ **Big data workloads**  
Launch instances of any size, in a single Availability Zone. Good for MapReduce jobs.

☐ **Defined duration**  
**workloads**  
Launch instances into a Spot block for 1 to 6 hours.  
1 時間

ワークロードに応じたインスタンスタイプの種類を提案してくれる

- Load balancing: 近しいサイズのインスタンスタイプから選択
  - Webワークロード: バックエンドインスタンス間の性能差が出ないように
  - 性能差がある場合、性能の高いインスタンスに負荷が偏ることがある

# スポットフリートの起動方法(2)

EC2 > スポットリクエスト > スポットインスタンスのリクエスト

## スポットインスタンスのリクエスト

アプリケーションまたはタスクのニーズについてお知らせください

ジョブに最も適切なコンピューティング性能を AWS が確認できるように、アプリケーションまたはタスクのニーズに最も一致するものを選択します。

☒ **Load balancing workloads**  
Launch instances of the same size, in any Availability Zone. Good for running web services.

☐ **Flexible workloads**  
Launch instances of any size, in any Availability Zone. Good for running batch and CI/CD jobs.

☐ **Big data workloads**  
Launch instances of any size, in a single Availability Zone. Good for MapReduce jobs.

☐ **Defined duration workloads**  
Launch instances into a Spot block for 1 to 6 hours.  
1 時間

ワークロードに応じたインスタンスタイプの種類を提案してくれる

- Flexible workloads: サイズの異なるインスタンスタイプから選択
  - CI/CDワークロード: 必ずしもインスタンスの性能が揃っている必要はない
  - 計算が早く終われば終わるほど良い→高性能のインスタンスタイプを提案



# スポットフリートの起動方法(3)

インスタンスの設定

起動テンプレートを使用して迅速にインスタンス起動パラメータを設定します (一部のパラメータは変更できます)。複数のアベイラビリティゾーンを含めることで、スポット容量へのアクセスを強化します。

起動テンプレート

なし (デフォルトバージョン) | 起動テンプレートの作成

AMI

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type カスタム AMI を使う

最小コンピュータユニット ☒ 仕様として ☐ インスタンスタイプとして

vCPU 2 メモリ (GiB) 7

ネットワーク

起動するスポットインスタンスが最低限満たす仕様を指定する

- 「仕様として」：vCPUとメモリを指定
- 「インスタンスタイプとして」：起動したいインスタンスタイプの一例を指定

# スポットフリートの起動方法(4)

必要な容量をお知らせください

起動するターゲット容量 (インスタンス数または vCPU 数) を設定します。起動テンプレートを指定した場合、ターゲット容量の一部をオンデマンドとして割り当てることができます。オンデマンドインスタンスの数は常に保持されますが、スポットインスタンスはスケールできます。

**合計ターゲット容量 ⓘ**

1 インスタンス▼

**オプションのオンデマンド部分 詳細はこちら**  
ら ↗

0 インスタンス

起動テンプレートを指定するリクエストのみがオンデマンドの対象です

☒ ターゲット容量を維持する

中断動作 ⓘ

終了 ▼

- スポットフリートに起動するスポットインスタンスの容量を、台数もしくはvCPUで指定する

# スポットフリートの起動方法(4)

必要な容量をお知らせください

起動するターゲット容量 (インスタンス数または vCPU 数) を設定します。起動テンプレートを指定した場合、ターゲット容量の一部をオンデマンドとして割り当てることができます。オンデマンドインスタンスの数は常に保持されますが、スポットインスタンスはスケールできます。

合計ターゲット容量 ⓘ

1 インスタンス

オプションのオンデマンド部分 [詳細はこちら](#)

0 インスタンス

起動テンプレートを指定するリクエストのみがオンデマンドの対象です

☒ ターゲット容量を維持する

中断動作 ⓘ

終了

- スポットフリートに起動するスポットインスタンスの容量を、台数もしくは vCPU で指定する
- スポットフリートの一部をオンデマンドインスタンスで構成できる

# スポットフリートの起動方法(4)

必要な容量をお知らせください

起動するターゲット容量 (インスタンス数または vCPU 数) を設定します。起動テンプレートを指定した場合、ターゲット容量の一部をオンデマンドとして割り当てることができます。オンデマンドインスタンスの数は常に保持されますが、スポットインスタンスはスケールできます。

合計ターゲット容量 ⓘ

1 インスタンス

オプションのオンデマンド部分 [詳細はこちら](#)

0 インスタンス

起動テンプレートを指定するリクエストのみがオンデマンドの対象です

☒ ターゲット容量を維持する

中断動作 ⓘ

終了

- スポットフリートに起動するスポットインスタンスの容量を、台数もしくは vCPU で指定する
- スポットフリートの一部をオンデマンドインスタンスで構成できる
- 「ターゲット容量を維持する」にチェックを入れると、中断などで指定容量を下回った場合にスポットフリートが自動で新しいインスタンスを起動する

# スポットフリートの起動方法(5)

フリートリクエストの設定

☒ 推奨事項の適用

フリートリクエスト

Amazon EC2 はこれらのインスタンスタイプからターゲット容量をリクエストします。指定したインスタンスタイプが多いほど、ターゲット容量を満たす可能性が高まります。

インスタンスタイプ	vCPUs	Memory (GiB)	スポット価格	オンデマンド料金の削減
t2.large	2 vCPUs	8GiB	\$0.0278/hr	70 %
m3.large	2 vCPUs	7.5GiB	\$0.0307/hr	77 %
m4.large	2 vCPUs	8GiB	\$0.0322/hr	68 %
r4.large	2 vCPUs	15.25GiB	\$0.0337/hr	75 %
m5.large	2 vCPUs	8GiB	\$0.0338/hr	65 %

✔ フリートの強度: 強度が十分

フリートには、ターゲット容量リクエストを満たす十分なインスタンスプールが含まれています。  
5 instance types x 4 Availability Zones = 20 instance pools

- 指定した値をもとにEC2スポットサービスがおすすめのインスタンスタイプを選定した結果が表示される


# スポットフリートの起動方法(5)

フリートリクエストの設定 推奨事項の適用

### フリートリクエスト

Amazon EC2 はこれらのインスタンスタイプからターゲット容量をリクエストします。指定したインスタンスタイプが多いほど、ターゲット容量を満たす可能性が高まります。

インスタンスタイプ	vCPUs	Memory (GiB)	スポット価格	オンデマンド料金の削減
t2.large	2 vCPUs	8GiB	\$0.0278/hr	70 %
m3.large	2 vCPUs	7.5GiB	\$0.0307/hr	77 %
m4.large	2 vCPUs	8GiB	\$0.0322/hr	68 %
r4.large	2 vCPUs	15.25GiB	\$0.0337/hr	75 %
m5.large	2 vCPUs	8GiB	\$0.0338/hr	65 %

 **フリートの強度: 強度が十分**  
フリートには、ターゲット容量リクエストを満たす十分なインスタンスプールが含まれています。  
5 instance types x 4 Availability Zones = 20 instance pools

- 指定した値をもとにEC2スポットサービスがおすすめのインスタンスタイプを選定した結果が表示される
- この結果をカスタマイズしたい場合は「推奨事項の適用」チェックを外す



# スポットフリートの起動方法(6)

フリートリクエストの概要

合計ターゲット容量 インスタンス (maintain capacity)	インスタンスの設定 <b>mylaunchtemplate, v.1</b> 2 vCPU, 7 GiB (min)   4 Availability Zones	フリートの強度 <b>強度が十分</b> 20 instance pools	推定料金 ~\$0/hr ターゲット容量で <b>71% 削減額</b> オンデマンドと比較
--	--	--	--

JSON 設定      キャンセル      作成

- フリートの強度が「強度が十分」であることを確認する  
（「強度」については後述）

# スポットフリートの起動方法(6)

フリートリクエストの概要

合計ターゲット容量 <b>インスタンス</b> (maintain capacity)	インスタンスの設定 <b>mylaunchtemplate, v.1</b> 2 vCPU, 7 GiB (min)   4 Availability Zones	フリートの強度 <b>強度が十分</b> 20 instance pools	推定料金 <b>~\$0/hr</b> ターゲット容量で <b>71% 削減額</b> オンデマンドと比較
---	--	--	---

JSON 設定      キャンセル      作成

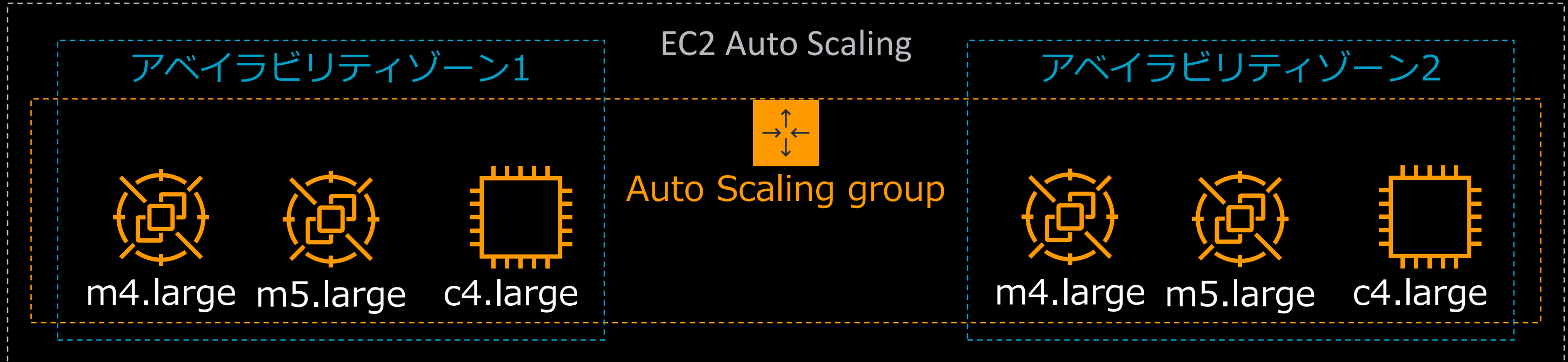
- フリートの強度が「強度が十分」であることを確認する
- フリート全体の削減額の見通しが表示される
- 問題なければ「作成」

# EC2 Auto Scalingの起動

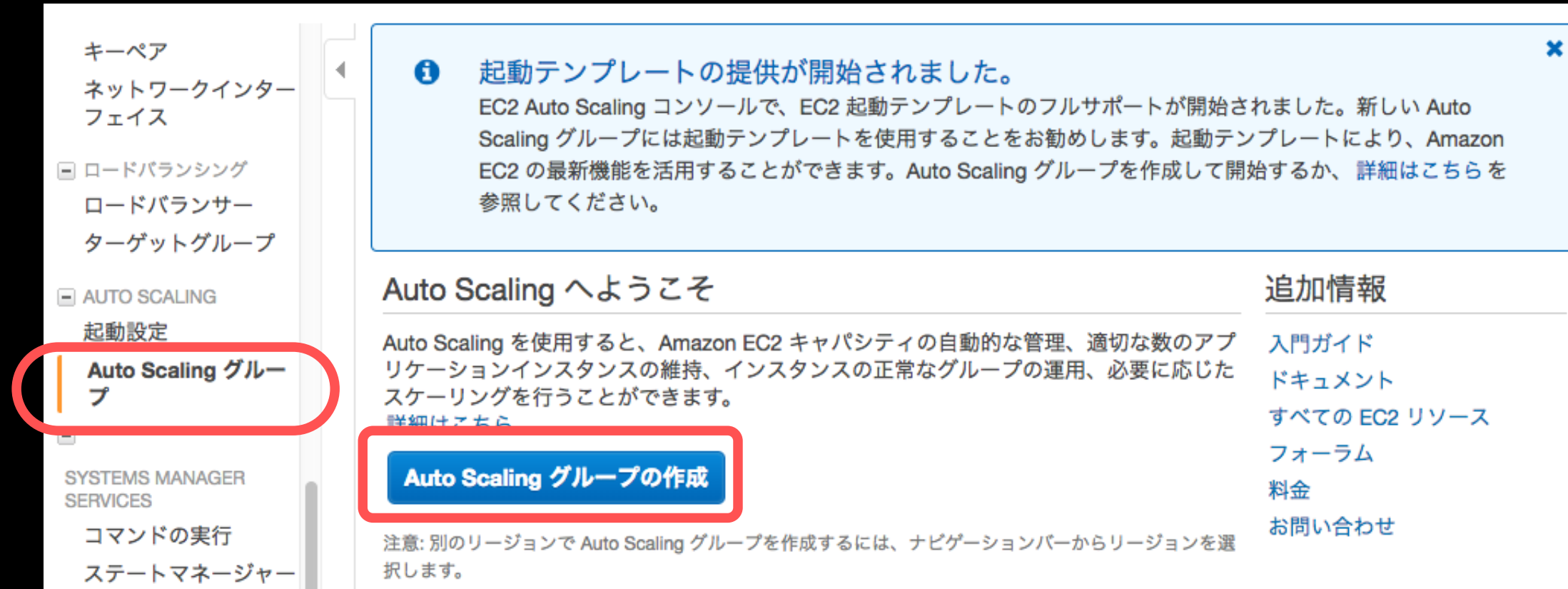


# EC2 Auto Scalingの新機能 – ミックスインスタンスグループ

- オンデマンドとスポットをひとつのASGに
  - (オンデマンド:スポット) = (9:1)といった指定ができる。スモールスタートに最適
- インスタンスタイプを複数指定可能
  - インスタンスタイプを分散できる(後述)



# EC2 Auto Scaling Groupの作成(1)



- EC2フリートと統合した新しいEC2 Auto Scalingの機能を使うには「起動テンプレート」を用いる必要がある

# EC2 Auto Scaling Groupの作成(2)

## Auto Scaling グループの作成

[キャンセルして終了](#)

このウィザードを終了して Auto Scaling グループを作成します。最初に、起動設定または起動テンプレートを選択して、インスタンスの起動に Auto Scaling グループが使用するパラメータを指定します。

☐ 起動設定

必要な Amazon EC2 の機能をサポートしている場合は、引き続き起動設定を使用できます。 [詳細はこちら](#)

☒ 起動テンプレート **新規**

起動テンプレートにより、1つの種類のインスタンスを起動するか、複数のインスタンスタイプと購入オプションの組み合わせを起動するかのオプションを利用できます。起動テンプレートには Amazon EC2 の最新機能が含まれていて、更新とバージョニングができます。 [詳細はこちら](#)  
[新しい起動テンプレートの作成](#)

- 「起動テンプレート」を選択する(事前に作成しておく)



# EC2 Auto Scaling Groupの作成(3)

1. Auto Scaling グループの詳細設定

2. スケーリングポリシーの設定

3. 通知の設定

4. タグを設定

5. 確認

## Auto Scaling グループの作成

グループ名 ⓘ

起動テンプレート ⓘ

lt-0757b443c586fc262

起動テンプレートのバージョン ⓘ

1 (デフォルト)



新しい起動テンプレートの作成

起動テンプレートの説明 ⓘ

-

フリートの構築

☒ 起動テンプレートに従う

起動テンプレートにより、インスタンスタイプと購入オプション (オンデマンドまたはスポット) が決まります。

☐ 購入オプションとインスタンスを組み合わせる

オンデマンドインスタンスとスポットインスタンスの組み合わせ、および複数のインスタンスタイプを選択します。スポットインスタンスは、利用できる最も安い料金で自動的に起動されます。

グループサイズ ⓘ

開始時  インスタンス

- 「購入オプションとインスタンスを組み合わせる」を選択

# EC2 Auto Scaling Groupの作成(3)

1. Auto Scaling グループの詳細設定

2. スケーリングポリシーの設定

3. 通知の設定

4. タグを設定

5. 確認

## Auto Scaling グループの作成

グループ名 ⓘ

起動テンプレート ⓘ

lt-0757b443c586fc262

起動テンプレートのバージョン ⓘ

1 (デフォルト)



新しい起動テンプレートの作成

起動テンプレートの説明 ⓘ

-

フリートの構築

☐ 起動テンプレートに従う

起動テンプレートにより、インスタンスタイプと購入オプション (オンデマンドまたはスポット) が決まります。

☒ 購入オプションとインスタンスを組み合わせる

オンデマンドインスタンスとスポットインスタンスの組み合わせ、および複数のインスタンスタイプを選択します。スポットインスタンスは、利用できる最も安い料金で自動的に起動されます。

インスタンスタイプ

許容できるインスタンスタイプをフリートに追加します。順序を変更し、オンデマンドインスタンスの起動の優先度を設定します。この順序によるスポットインスタンスへの影響はありません。

インスタンスタイプの選択



最低2つのインスタンスタイプを追加してください

インスタンスタイプの追加

インスタンスの分散 ⓘ

☒ 次のデフォルト設定を使用し、すぐに開始します。

# EC2 Auto Scaling Groupの作成(3)

1. Auto Scaling グループの詳細設定

2. スケーリングポリシーの設定

3. 通知の設定

4. タグを設定

5. 確認

## Auto Scaling グループの作成

グループ名 ⓘ

起動テンプレート ⓘ

lt-0757b443c586fc262

起動テンプレートのバージョン ⓘ

1 (デフォルト)



新しい起動テンプレートの作成

起動テンプレートの説明 ⓘ

-

フリートの構築

☐ 起動テンプレートに従う

起動テンプレートにより、インスタンスタイプと購入オプション (オンデマンドまたはスポット) が決まります。

☒ 購入オプションとインスタンスを組み合わせる

オンデマンドインスタンスとスポットインスタンスの組み合わせ、および複数のインスタンスタイプを選択します。スポットインスタンスは、利用できる最も安い料金で自動的に起動されます。

インスタンスタイプ

許容できるインスタンスタイプをフリートに追加します。順序を変更し、オンデマンドインスタンスの起動の優先度を設定します。この順序によるスポットインスタンスへの影響はありません。

インスタンスタイプの選択



最低2つのインスタンスタイプを追加してください

インスタンスタイプの追加

インスタンスの分散 ⓘ

☒ 次のデフォルト設定を使用し、すぐに開始します。

# EC2 Auto Scaling Groupの作成(3)

## フリートの構築

☐ 起動テンプレートに従う

起動テンプレートにより、インスタンスタイプと購入オプション (オンデマンドまたはスポット) が決まります。

☒ 購入オプションとインスタンスを組み合わせる

オンデマンドインスタンスとスポットインスタンスの組み合わせ、および複数のインスタンスタイプを選択します。スポットインスタンスは、利用できる最も安い料金で自動的に起動されます。

## インスタンスタイプ

許容できるインスタンスタイプをフリートに追加します。順序を変更し、オンデマンドインスタンスの起動の優先度を設定します。この順序によるスポットインスタンスへの影響はありません。

m4.large (2vCPU、8GiB)



c4.large (2vCPU、3.75GiB)



インスタンスタイプの追加

- 「購入オプションとインスタンスを組み合わせる」を選択し、要件に合うインスタンスタイプを複数選択する
- 起動テンプレートに指定しておくことも可能

# EC2 Auto Scaling Groupの作成(4)

## インスタンスの分散 ⓘ

☒ 次のデフォルト設定を使用し、すぐに開始します。

- 上記の優先度に基づき、オンデマンドインスタンスを起動します。
- アベイラビリティゾーンごとに 2 つの最低価格インスタンスタイプ間でスポットインスタンスを多様化します。
- 各インスタンスタイプの最大スポット料金を、オンデマンド料金と同じに設定します。
- 70% のオンデマンドインスタンスと 30% のスポットインスタンスを組み合わせで維持します。

## グループサイズ ⓘ

開始時  インスタンス

- 「インスタンスの分散」のチェックを外す

# EC2 Auto Scaling Groupの作成(4)

インスタンスの分散 ⓘ

☐ 次のデフォルト設定を使用し、すぐに開始します。

オンデマンドの割り当て戦略 ⓘ

優先順位付け

最大スポット料金 ⓘ

☒ デフォルトを使用 (推奨)

デフォルトでは現在のスポット料金を使用されますが、オンデマンド価格に上限が設定されます。

☐ 上限価格を設定 (1 インスタンス/時間あたり)

スポットの配分戦略 ⓘ

スポットインスタンスを  アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する

オプションのオンデマンドベース ⓘ

最初のインスタンスを  オンデマンドとして指定します

ベースを超えるオンデマンド割合 ⓘ

% オンデマンドおよび 30% スポット

グループサイズ ⓘ

開始時  インスタンス

- 追加オプションが表示される
- 「ベースを超えるオンデマンド割合」にオンデマンド：スポット比率を指定できる



# EC2 Auto Scaling Groupの作成(4)

インスタンスの分散 ⓘ	<input type="checkbox"/> 次のデフォルト設定を使用し、すぐに開始します。
オンデマンドの割り当て戦略 ⓘ	優先順位付け
最大スポット料金 ⓘ	<input checked="" type="radio"/> デフォルトを使用 (推奨) デフォルトでは現在のスポット料金が使用されますが、オンデマンド価格に上限が設定されます。 <input type="radio"/> 上限価格を設定 (1 インスタンス/時間あたり)
スポットの配分戦略 ⓘ	スポットインスタンスを <input type="text" value="2"/> アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する
オプションのオンデマンドベース ⓘ	最初のインスタンスを <input type="text" value="0"/> オンデマンドとして指定します
ベースを超えるオンデマンド割合 ⓘ	<input type="text" value="70"/> % オンデマンドおよび 30% スポット
グループサイズ ⓘ	開始時 <input type="text" value="1"/> インスタンス

- 台数の考え方
  - 1. 全体数は「グループサイズ」
  - 2. そこから「オプションのオンデマンドベース」の台数をオンデマンドで起動
  - 3. 残りを「ベースを超えるオンデマンド割合」にしたがって分配
- 台数の考え方の例
  - 「グループサイズ」 : 12
  - 「オプションのオンデマンドベース」 : 2
  - 「ベースを超えるオンデマンド割合」 : 30:70
- 結果
  - オンデマンド2台+3台
  - スポット7台

# EC2 Auto Scaling Groupの作成(4)

インスタンスの分散 ⓘ	<input type="checkbox"/> 次のデフォルト設定を使用し、すぐに開始します。
オンデマンドの割り当て戦略 ⓘ	優先順位付け
最大スポット料金 ⓘ	<input checked="" type="radio"/> デフォルトを使用 (推奨) デフォルトでは現在のスポット料金が使用されますが、オンデマンド価格に上限が設定されます。 <input type="radio"/> 上限価格を設定 (1 インスタンス/時間あたり)
スポットの配分戦略 ⓘ	スポットインスタンスを <input type="text" value="2"/> アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する
オプションのオンデマンドベース ⓘ	最初のインスタンスを <input type="text" value="0"/> オンデマンドとして指定します
ベースを超えるオンデマンド割合 ⓘ	<input type="text" value="70"/> % オンデマンドおよび 30% スポット
グループサイズ ⓘ	開始時 <input type="text" value="1"/> インスタンス

- 台数の考え方
  - 1. 全体数は「グループサイズ」

グループサイズ

# EC2 Auto Scaling Groupの作成(4)

インスタンスの分散 ⓘ	<input type="checkbox"/> 次のデフォルト設定を使用し、すぐに開始します。
オンデマンドの割り当て戦略 ⓘ	優先順位付け
最大スポット料金 ⓘ	<input checked="" type="radio"/> デフォルトを使用 (推奨) デフォルトでは現在のスポット料金が使用されますが、オンデマンド価格に上限が設定されます。 <input type="radio"/> 上限価格を設定 (1 インスタンス/時間あたり)
スポットの配分戦略 ⓘ	スポットインスタンスを <input type="text" value="2"/> アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する
オプションのオンデマンドベース ⓘ	最初のインスタンスを <input type="text" value="0"/> オンデマンドとして指定します
ベースを超えるオンデマンド割合 ⓘ	<input type="text" value="70"/> % オンデマンドおよび 30% スポット
グループサイズ ⓘ	開始時 <input type="text" value="1"/> インスタンス

- 台数の考え方
  - 1. 全体数は「グループサイズ」
  - 2. そこから「オプションのオンデマンドベース」の台数をオンデマンドで起動

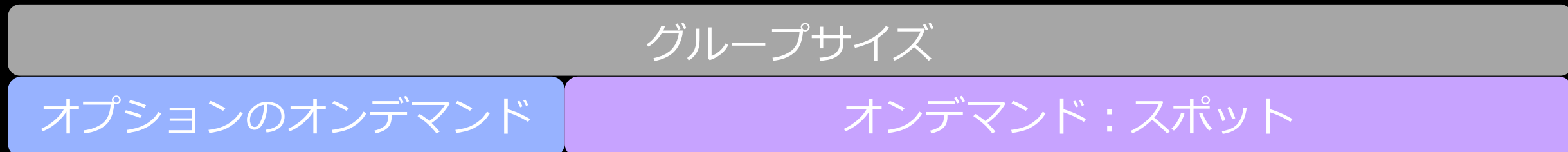
グループサイズ

オプションのオンデマンド

# EC2 Auto Scaling Groupの作成(4)

インスタンスの分散 ⓘ	<input type="checkbox"/> 次のデフォルト設定を使用し、すぐに開始します。
オンデマンドの割り当て戦略 ⓘ	優先順位付け
最大スポット料金 ⓘ	<input checked="" type="radio"/> デフォルトを使用 (推奨) デフォルトでは現在のスポット料金が使用されますが、オンデマンド価格に上限が設定されます。 <input type="radio"/> 上限価格を設定 (1 インスタンス/時間あたり)
スポットの配分戦略 ⓘ	スポットインスタンスを <input type="text" value="2"/> アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する
オプションのオンデマンドベース ⓘ	最初のインスタンスを <input type="text" value="0"/> オンデマンドとして指定します
ベースを超えるオンデマンド割合 ⓘ	<input type="text" value="70"/> % オンデマンドおよび 30% スポット
グループサイズ ⓘ	開始時 <input type="text" value="1"/> インスタンス

- 台数の考え方
  - 1. 全体数は「グループサイズ」
  - 2. そこから「オプションのオンデマンドベース」の台数をオンデマンドで起動
  - 3. 残りを「ベースを超えるオンデマンド割合」にしたがって分配



# EC2 Auto Scaling Groupの作成(4)

インスタンスの分散 ⓘ	<input type="checkbox"/> 次のデフォルト設定を使用し、すぐに開始します。
オンデマンドの割り当て戦略 ⓘ	優先順位付け
最大スポット料金 ⓘ	<input checked="" type="radio"/> デフォルトを使用 (推奨) デフォルトでは現在のスポット料金が使用されますが、オンデマンド価格に上限が設定されます。 <input type="radio"/> 上限価格を設定 (1 インスタンス/時間あたり)
スポットの配分戦略 ⓘ	スポットインスタンスを <input type="text" value="2"/> アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する
オプションのオンデマンドベース ⓘ	最初のインスタンスを <input type="text" value="0"/> オンデマンドとして指定します
ベースを超えるオンデマンド割合 ⓘ	<input type="text" value="70"/> % オンデマンドおよび 30% スポット
グループサイズ ⓘ	開始時 <input type="text" value="1"/> インスタンス

- 台数の考え方の例
  - 「グループサイズ」 : 12
  - 「オプションのオンデマンドベース」 : 2

グループサイズ = 12

オンデマンド=2

# EC2 Auto Scaling Groupの作成(4)

インスタンスの分散 ⓘ	<input type="checkbox"/> 次のデフォルト設定を使用し、すぐに開始します。
オンデマンドの割り当て戦略 ⓘ	優先順位付け
最大スポット料金 ⓘ	<input checked="" type="radio"/> デフォルトを使用 (推奨) デフォルトでは現在のスポット料金が使用されますが、オンデマンド価格に上限が設定されます。 <input type="radio"/> 上限価格を設定 (1 インスタンス/時間あたり)
スポットの配分戦略 ⓘ	スポットインスタンスを <input type="text" value="2"/> アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する
オプションのオンデマンドベース ⓘ	最初のインスタンスを <input type="text" value="0"/> オンデマンドとして指定します
ベースを超えるオンデマンド割合 ⓘ	<input type="text" value="70"/> % オンデマンドおよび 30% スポット
グループサイズ ⓘ	開始時 <input type="text" value="1"/> インスタンス

- 台数の考え方の例
  - 「グループサイズ」 : 12
  - 「オプションのオンデマンドベース」 : 2
  - 「ベースを超えるオンデマンド割合」 : 30:70

グループサイズ = 12

オンデマンド=2

オンデマンド=3

スポット=7



# EC2 Auto Scaling Groupの作成(4)

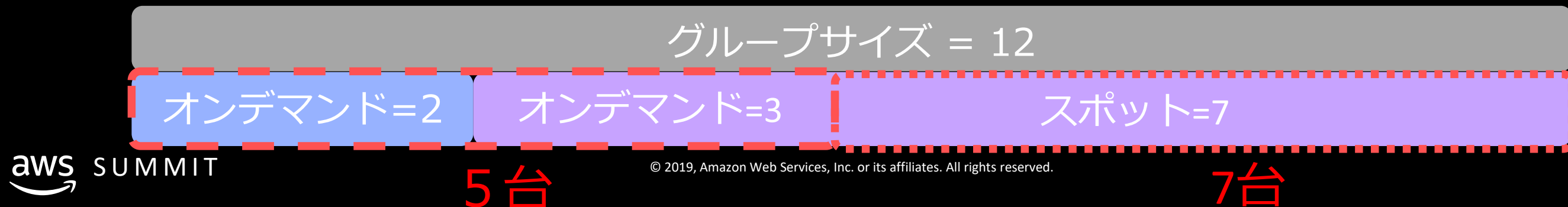
インスタンスの分散 ⓘ	<input type="checkbox"/> 次のデフォルト設定を使用し、すぐに開始します。
オンデマンドの割り当て戦略 ⓘ	優先順位付け
最大スポット料金 ⓘ	<input checked="" type="radio"/> デフォルトを使用 (推奨) デフォルトでは現在のスポット料金が使用されますが、オンデマンド価格に上限が設定されます。 <input type="radio"/> 上限価格を設定 (1 インスタンス/時間あたり)
スポットの配分戦略 ⓘ	スポットインスタンスを <input type="text" value="2"/> アベイラビリティゾーンごとに最も価格の安いインスタンスタイプ間で多様化する
オプションのオンデマンドベース ⓘ	最初のインスタンスを <input type="text" value="0"/> オンデマンドとして指定します
ベースを超えるオンデマンド割合 ⓘ	<input type="text" value="70"/> % オンデマンドおよび 30% スポット
グループサイズ ⓘ	開始時 <input type="text" value="1"/> インスタンス

- 台数の考え方の例

- 「グループサイズ」 : 12
- 「オプションのオンデマンドベース」 : 2
- 「ベースを超えるオンデマンド割合」 : 30:70

- 結果

- オンデマンド5台
- スポット7台



# スポットリクエストの確認方法

# スポットインスタンスの確認方法

# スポットリクエストの確認方法(1)

EC2 ダッシュボード  
イベント  
タグ  
レポート  
制限

インスタンス  
インスタンス  
起動テンプレート  
**スポットリクエスト**  
リザーブドインスタンス  
専有ホスト  
スケジュール済みインスタンス

スポットインスタンスのリクエスト    アクション ▾    価格設定履歴    Savings Summary    ↺    ⚙

リクエストタイプ: all ▾    状態: all ▾    キーワードによる検索    « < 4 リクエスト中 1 から 4 を表示 > »

<input type="checkbox"/>	リクエスト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス	永続性
<input type="checkbox"/>	sir-ig5i7vzj	instance	m4.large	● active	i-0d0492c682ed...	fulfilled	one-ti
<input type="checkbox"/>	sir-63rr4ykj	instance	m4.large	● active	i-0c25fe1b485b...	fulfilled	one-ti
<input type="checkbox"/>	sir-eq8r49hj	instance	t2.micro	● active	i-01e817995825...	fulfilled	one-ti
<input type="checkbox"/>	▶ sfr-8b58cc41-7ce9...	fleet	c3.large,c4.large	● active	2 of 2	fulfilled	mainti

- 「スポットリクエスト」で一覧表示

# スポットリクエストの確認方法(1)

EC2 ダッシュボード

イベント

タグ

レポート

制限

インスタンス

インスタンス

起動テンプレート

**スポットリクエスト**

リザーブドインスタンス

専有ホスト

スケジュール済みインスタンス

スポットインスタンスのリクエスト

アクション ▾

価格設定履歴

Savings Summary

リクエストタイプ: all ▾

状態: all ▾

キーワードによる検索

《 < 4 リクエスト中 1 から 4 を表示 > 》

<input type="checkbox"/>	リクエスト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス	永続性
<input type="checkbox"/>	sir-ig5i7vzj	instance	m4.large	● active	i-0d0492c682ed...	fulfilled	one-ti
<input type="checkbox"/>	sir-63rr4ykj	instance	m4.large	● active	i-0c25fe1b485b...	fulfilled	one-ti
<input type="checkbox"/>	sir-eq8r49hj	instance	t2.micro	● active	i-01e817995825...	fulfilled	one-ti
<input type="checkbox"/>	▶ sfr-8b58cc41-7ce9...	fleet	c3.large,c4.large	● active	2 of 2	fulfilled	mainti

- 「スポットリクエスト」で一覧表示
- sir-xxx: スポットインスタンスリクエスト
- sfr-xxx: スポットフリートリクエスト

# スポットリクエストの確認方法(1)

EC2 ダッシュボード

イベント

タグ

レポート

制限

インスタンス

インスタンス

起動テンプレート

スポットリクエスト

リザーブドインスタンス

専有ホスト

スケジュール済みインスタンス

スポットインスタンスのリクエスト

アクション ▾

価格設定履歴

Savings Summary

リフレッシュ

設定

リクエストタイプ: all ▾

状態: all ▾

キーワードによる検索

4 リクエスト中 1 から 4 を表示

	リクエスト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス	永続性
<input type="checkbox"/>	sir-ig5i7vzj	instance	m4.large	● active	i-0d0492c682ed...	fulfilled	one-ti
<input type="checkbox"/>	sir-63rr4ykj	instance	m4.large	● active	i-0c25fe1b485b...	fulfilled	one-ti
<input type="checkbox"/>	sir-eq8r49hj	instance	t2.micro	● active	i-01e817995825...	fulfilled	one-ti
<input type="checkbox"/>	▶ sfr-8b58cc41-7ce9...	fleet	c3.large,c4.large	● active	2 of 2	fulfilled	maint:

- 「スポットリクエスト」で一覧表示
  - sir-xxx: スポットインスタンスリクエスト
  - sfr-xxx: スポットフリートリクエスト

# スポットリクエストの確認方法(1)

EC2 ダッシュボード  
イベント  
タグ  
レポート  
制限

インスタンス  
インスタンス  
起動テンプレート  
**スポットリクエスト**  
リザーブドインスタンス  
専用ホスト  
スケジュール済みインスタンス

スポットインスタンスのリクエスト    アクション ▾    価格設定履歴    Savings Summary    [refresh] [settings]

リクエストタイプ: all ▾    状態: all ▾    キーワードによる検索    << < 4 リクエスト中 1 から 4 を表示 > >>

<input type="checkbox"/>	リクエスト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス	永続性
<input type="checkbox"/>	sir-ig5i7vzj	instance	m4.large	● active	i-0d0492c682ed...	fulfilled	one-ti
<input type="checkbox"/>	sir-63rr4ykj	instance	m4.large	● active	i-0c25fe1b485b...	fulfilled	one-ti
<input type="checkbox"/>	sir-eq8r49hj	instance	t2.micro	● active	i-01e817995825...	fulfilled	one-ti
<input type="checkbox"/>	▶ sfr-3b58cc41-7ce9...	fleet	c3.large,c4.large	● active	2 of 2	fulfilled	mainti

- スポットフリートリクエスト(sfr-xxx)には複数のスポットインスタンスリクエスト(sir-xxx)が関連づいている



# スポットリクエストの確認方法(2)

EC2 ダッシュボード  
イベント  
タグ  
レポート  
制限

インスタンス  
インスタンス  
起動テンプレート  
**スポットリクエスト**  
リザーブドインスタンス  
専有ホスト  
スケジュール済みインスタンス

スポットインスタンスのリクエスト    アクション    価格設定履歴    Savings Summary    [refresh] [settings]

リクエストタイプ: all    状態: all    キーワードによる検索    << < 4 リクエスト中 1 から 4 を表示 > >>

<input type="checkbox"/>	リクエスト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス	永続性
<input type="checkbox"/>	sir-ig5i7vzj	instance	m4.large	● active	i-0d0492c682ed...	fulfilled	one-ti
<input type="checkbox"/>	sir-63rr4ykj	instance	m4.large	● active	i-0c25fe1b485b...	fulfilled	one-ti
<input type="checkbox"/>	sir-eq8r49hj	instance	t2.micro	● active	i-01e817995825...	fulfilled	one-ti
<input checked="" type="checkbox"/>	▼ sfr-8b58cc41-7ce9...	fleet	c3.large,c4.large	● active	2 of 2	fulfilled	mainti
<input type="checkbox"/>	sir-22e85sbh	instance	c4.large	● active	i-096c8ec87443...	fulfilled	persis
<input type="checkbox"/>	sir-g7bi4jyj	instance	c3.large	● active	i-05a0978dfea5...	fulfilled	persis

- スポットフリートリクエスト(sfr-xxx)には複数のスポットインスタンスリクエスト(sir-xxx)が関連づいている

# スポットインスタンスの確認方法(1)

The screenshot shows the AWS Management Console interface. On the left sidebar, the 'Instances' tab is highlighted with a red circle. The main content area displays a table of EC2 instances. The table has columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, and Status Checks. All instances shown are in the 'running' state. The top navigation bar includes buttons for 'Instancesの作成', '接続', and 'アクション'. A search bar is located above the table, and pagination controls show '7 中の 1 ~ 7'.

<input type="checkbox"/>	Name	インスタンス ID	インスタンスタイプ	アベイラビリティゾーン	インスタンスの状態	ステータスチェック	アクション
<input type="checkbox"/>		i-01e817995825cc0ce	t2.micro	us-east-1a	● running	✓ 2/2 のチェック...	な
<input type="checkbox"/>		i-044dea2c05605c895	m4.large	us-east-1b	● running	✓ 2/2 のチェック...	な
<input type="checkbox"/>		i-05a0978dfea599a3e	c3.large	us-east-1b	● running	✓ 2/2 のチェック...	な
<input type="checkbox"/>		i-096c8ec87443a1871	c4.large	us-east-1b	● running	✓ 2/2 のチェック...	な
<input type="checkbox"/>		i-0c25fe1b485b52de7	m4.large	us-east-1c	● running	✓ 2/2 のチェック...	な
<input type="checkbox"/>		i-0d0492c682edddb33	m4.large	us-east-1d	● running	✓ 2/2 のチェック...	な
<input type="checkbox"/>		i-0e007facc04d3762d	m4.large	us-east-1f	● running	✓ 2/2 のチェック...	な

- 「インスタンス」で一覧表示

# スポットインスタンスの確認方法(1)



EC2 ダッシュボード

イベント

タグ

レポート

制限

インスタンス

**インスタンス**

起動テンプレート

スポットリクエスト

リザーブドインスタンス

専有ホスト

スケジュールされたインスタンス

キャパシティの予約

インスタンスの作成

接続

アクション

タグや属性によるフィルター、またはキーワードによる検索

7 中の 1 ~ 7

<input type="checkbox"/>	Name	インスタンス ID	インスタンスタイプ	アベイラビリティー	インスタンスの状態	ステータスチェック	アクション
<input type="checkbox"/>		i-01e817995825cc0ce	t2.micro	us-east-1a	● running	✓ 2/2 のチェック	な
<input type="checkbox"/>		i-044dea2c05605c895	m4.large	us-east-1b	● running	✓ 2/2 のチェック	な
<input type="checkbox"/>		i-05a0978dfea599a3e	c3.large	us-east-1b	● running	✓ 2/2 のチェック	な
<input type="checkbox"/>		i-096c8ec87443a1871	c4.large	us-east-1b	● running	✓ 2/2 のチェック	な
<input type="checkbox"/>		i-0c25fe1b485b52de7	m4.large	us-east-1c	● running	✓ 2/2 のチェック	な
<input type="checkbox"/>		i-0d0492c682edddb33	m4.large	us-east-1d	● running	✓ 2/2 のチェック	な
<input type="checkbox"/>		i-0e007facc04d3762d	m4.large	us-east-1f	● running	✓ 2/2 のチェック	な

- 右上の歯車アイコンをクリックし、表示列をカスタマイズする

# スポットインスタンスの確認方法

列の表示/非表示

タグキー

☒ Name

☒ aws:autoscaling:groupName

☐ aws:ec2:fleet-id

☐ aws:ec2:launchtemplate:id

☐ aws:ec2:launchtemplate:version

☒ aws:ec2spot:fleet-request-id

☐ loadBalancerTargetGroup

インスタンス の属性

☐ フォンド、ドメイン、ドレハ

☐ 仮想化タイプ

☐ アーキテクチャ

☒ ライフサイクル

☐ 所有者

☐ EBS 最適化

☐ 状態遷移の理由

☐ 状態遷移メッセージ

☐ ルートデバイス名

☐ ブロックデバイス

☐ プレイメントグループ

☐ Partition Number

☐ テナンシー

☐ ホスト ID

☐ アフィニティ

閉じる

# スポットインスタンスの確認方法

EC2 ダッシュボード

イベント  
タグ  
レポート  
制限

インスタンス  
**インスタンス**  
起動テンプレート  
スポットリクエスト  
リザーブドインスタンス  
専有ホスト  
スケジュールされたインスタンス

インスタンスの作成 ▼ 接続 アクション ▼

タグや属性によるフィルター、またはキーワードによる検索

<input type="checkbox"/>	Name ▼	aws:autosca ▲	aws:ec2spot: ▼	インスタンス ID ▼	インスタンスタイプ ▼	ライフサイクル ▼
<input type="checkbox"/>		myasg		i-044dea2c05605c895	m4.large	normal
<input type="checkbox"/>		myasg		i-0c25fe1b485b52de7	m4.large	spot
<input type="checkbox"/>		myasg		i-0d0492c682edddb33	m4.large	spot
<input type="checkbox"/>		myasg		i-0e007facc04d3762d	m4.large	normal
<input type="checkbox"/>				i-01e817995825cc0ce	t2.micro	spot
<input type="checkbox"/>			sfr-8b58cc41...	i-05a0978dfea599a3e	c3.large	spot
<input type="checkbox"/>			sfr-8b58cc41...	i-096c8ec87443a1871	c4.large	spot

- Auto Scalingグループ名を表示できる

# スポットインスタンスの確認方法

EC2 ダッシュボード

イベント  
タグ  
レポート  
制限

インスタンス  
**インスタンス**  
起動テンプレート  
スポットリクエスト  
リザーブドインスタンス  
専有ホスト  
スケジュールされたインスタンス

インスタンスの作成 ▼ 接続 アクション ▼

タグや属性によるフィルター、またはキーワードによる検索

<input type="checkbox"/>	Name ▼	aws:autosca ▲	aws:ec2spot:▼	インスタンス ID ▼	インスタンスタ▼	ライフサイクル ▼
<input type="checkbox"/>		myasg		i-044dea2c05605c895	m4.large	normal
<input type="checkbox"/>		myasg		i-0c25fe1b485b52de7	m4.large	spot
<input type="checkbox"/>		myasg		i-0d0492c682edddb33	m4.large	spot
<input type="checkbox"/>		myasg		i-0e007facc04d3762d	m4.large	normal
<input type="checkbox"/>				i-01e817995825cc0ce	t2.micro	spot
<input type="checkbox"/>			sfr-8b58cc41...	i-05a0978dfea599a3e	c3.large	spot
<input type="checkbox"/>			sfr-8b58cc41...	i-096c8ec87443a1871	c4.large	spot

- スポットフリートリクエストIDを表示できる

# スポットインスタンスの確認方法

EC2 ダッシュボード

イベント  
タグ  
レポート  
制限

インスタンス  
**インスタンス**  
起動テンプレート  
スポットリクエスト  
リザーブドインスタンス  
専有ホスト  
スケジュールされたインスタンス

インスタンスの作成 接続 アクション

タグや属性によるフィルター、またはキーワードによる検索

7 中の 1 ~ 7

	Name	aws:autosca	aws:ec2spot:	インスタンス ID	インスタンスタイプ	ライフサイクル
<input type="checkbox"/>	myasg			i-044dea2c05605c895	m4.large	normal
<input type="checkbox"/>	myasg			i-0c25fe1b485b52de7	m4.large	spot
<input type="checkbox"/>	myasg			i-0d0492c682edddb33	m4.large	spot
<input type="checkbox"/>	myasg			i-0e007facc04d3762d	m4.large	normal
<input type="checkbox"/>				i-01e817995825cc0ce	t2.micro	spot
<input type="checkbox"/>		sfr-8b58cc41...		i-05a0978dfea599a3e	c3.large	spot
<input type="checkbox"/>		sfr-8b58cc41...		i-096c8ec87443a1871	c4.large	spot

- 購入オプションを表示できる

- normal: オンデマンド・リザーブド
- spot: スポット



# <参考> スポットインスタンスの起動方法大全

機能名	起動用AWS CLI (aws ec2 ...)	リクエスト 単位	インスタンスタイプ の指定	容量 変更	対応するマネジメン トコンソール (EC2)
スポット インスタンス	run-instances および request- spot-instances	インスタンス台数	単一のインスタンス タイプ	×	「インスタンス」
スポット フリート	request-spot-fleet	インスタンス台数 / vCPU数	複数のインスタンス タイプ	○	「スポット リクエスト」
Auto Scaling Group	create-auto-scaling- group (aws autoscaling)	インスタンス台数	複数のインスタンス タイプ	○	「Auto Scaling グルー プ」
EC2 フリート	create-fleet	インスタンス台数 / vCPU数	複数のインスタンス タイプ	○	なし (API/CLIのみ)

# <参考> スポットインスタンスの起動方法大全

機能名	起動用AWS CLI (aws ec2 ...)	リクエスト 単位	インスタンスタイプ の指定	容量 変更	対応するマネジメン トコンソール (EC2)
スポット インスタンス	run-instances および request- spot-instances	インスタンス台数	単一のインスタンス タイプ	×	「インスタンス」
スポット フリート	request-spot-fleet	インスタンス台数 / vCPU数	複数のインスタンス タイプ	○	「スポット リクエスト」
Auto Scaling Group	create-auto-scaling- group (aws autoscaling)	インスタンス台数	複数のインスタンス タイプ	○	「Auto Scaling グルー プ」
EC2 フリート	create-fleet	インスタンス台数 / vCPU数	複数のインスタンス タイプ	○	なし (API/CLIのみ)

# <参考> EC2 Auto Scalingとスポットフリートの比較

機能	EC2 Auto Scaling	Spot Fleet
希望容量の維持	○	○
自動スケーリング	○	○
複数AZの指定	○	○
複数インスタンスタイプの指定	○	○
オンデマンドインスタンスの混在	○ [1] 1. 全体数はTotal target capacity 2. そこから"Optional On-demand portion"の台数をオンデマンドで起動 3. 残りをスポットで起動	○ [2] 1. 全体数はDesired Capacity 2. そこから"On-Demand base"の台数をオンデマンドで起動 3. 残りをOD:Spot Ratioにしたがって分配
ELBからのトラフィック受信	○	○
Launch Template \$latest, \$defaultでの参照	○	×

[1] [https://docs.aws.amazon.com/ja\\_jp/AWSEC2/latest/UserGuide/spot-fleet.html#on-demand-in-spot](https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/spot-fleet.html#on-demand-in-spot)

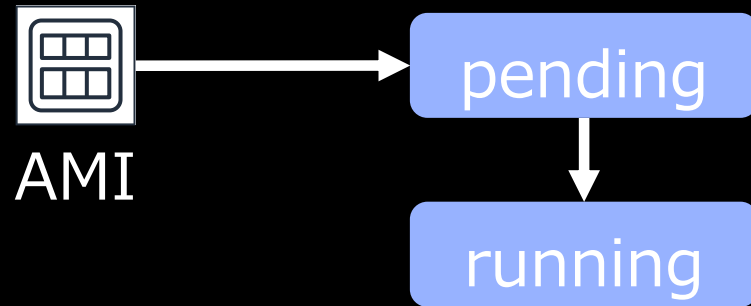
[2] [https://docs.aws.amazon.com/ja\\_jp/autoscaling/ec2/userguide/AutoScalingGroup.html#asg-purchase-options](https://docs.aws.amazon.com/ja_jp/autoscaling/ec2/userguide/AutoScalingGroup.html#asg-purchase-options)

## <参考> EC2 Auto Scalingとスポットフリーストの比較 (2) - 独自機能

- EC2 Auto Scaling
  - 終了ポリシー
  - インスタンスの保護
  - ライフサイクルフック
  - 一時的なインスタンスの削除
  - スケーリングプロセスの中断
  - アクティビティの再分散
- スポットフリースト
  - スポットコンソールの機能
    - 複数インスタンスタイプの提示
    - 推定料金(リクエスト時)および削減額(表示時)の提示
  - 中断通知の発行

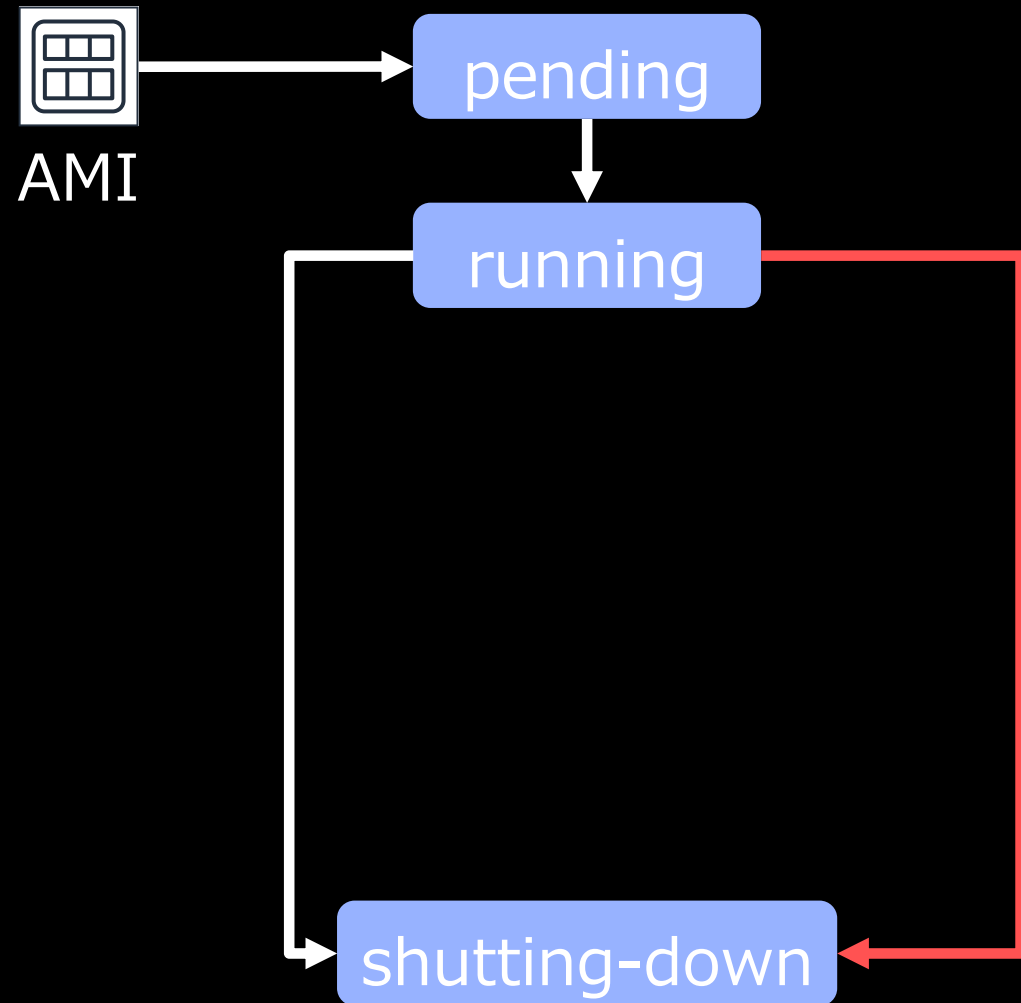
# スポットインスタンスの ライフサイクル

# スポットインスタンスのライフサイクル



- 起動までの流れはオンデマンドインスタンスと同一

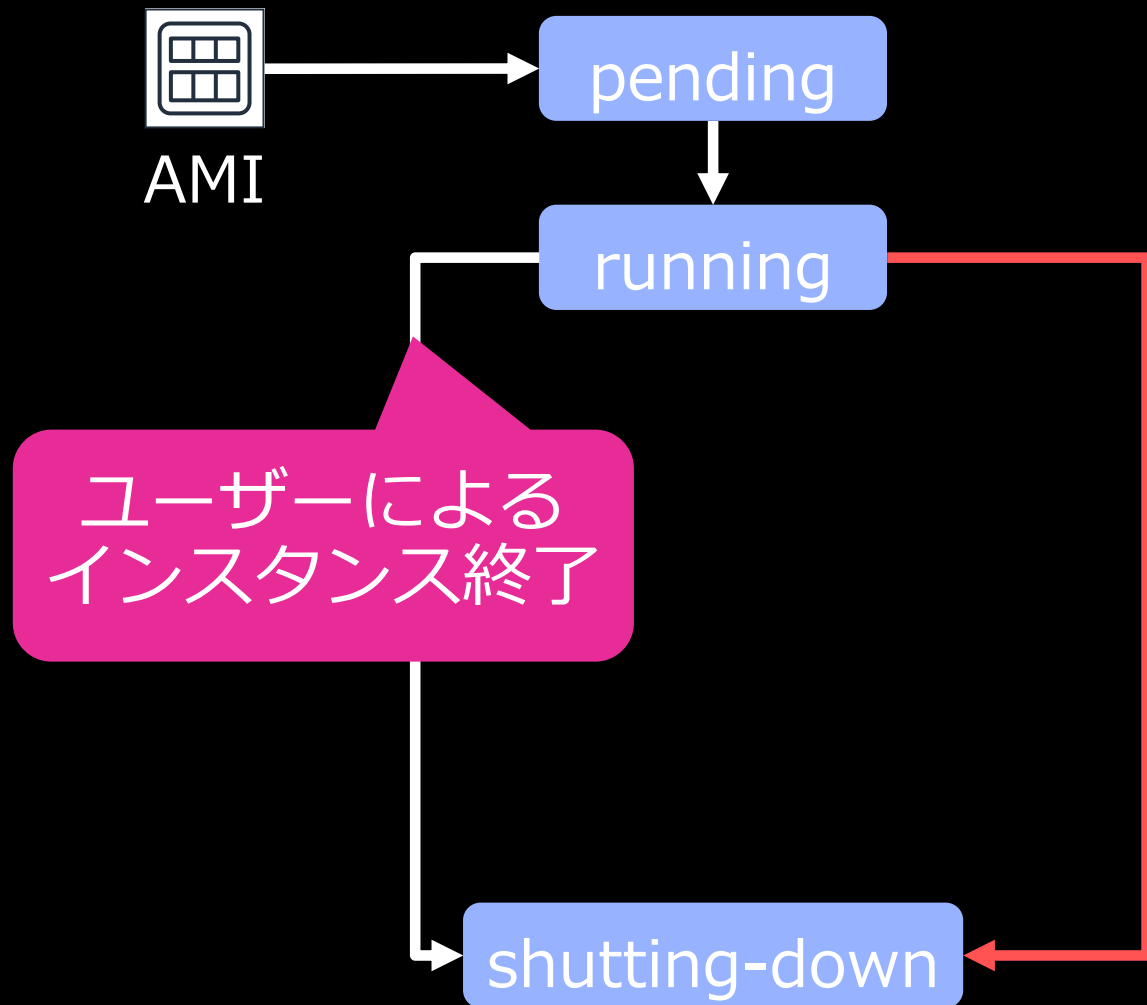
# スポットインスタンスのライフサイクル



- 起動までの流れはオンデマンドインスタンスと同一
- 終了に至る道筋は2通り

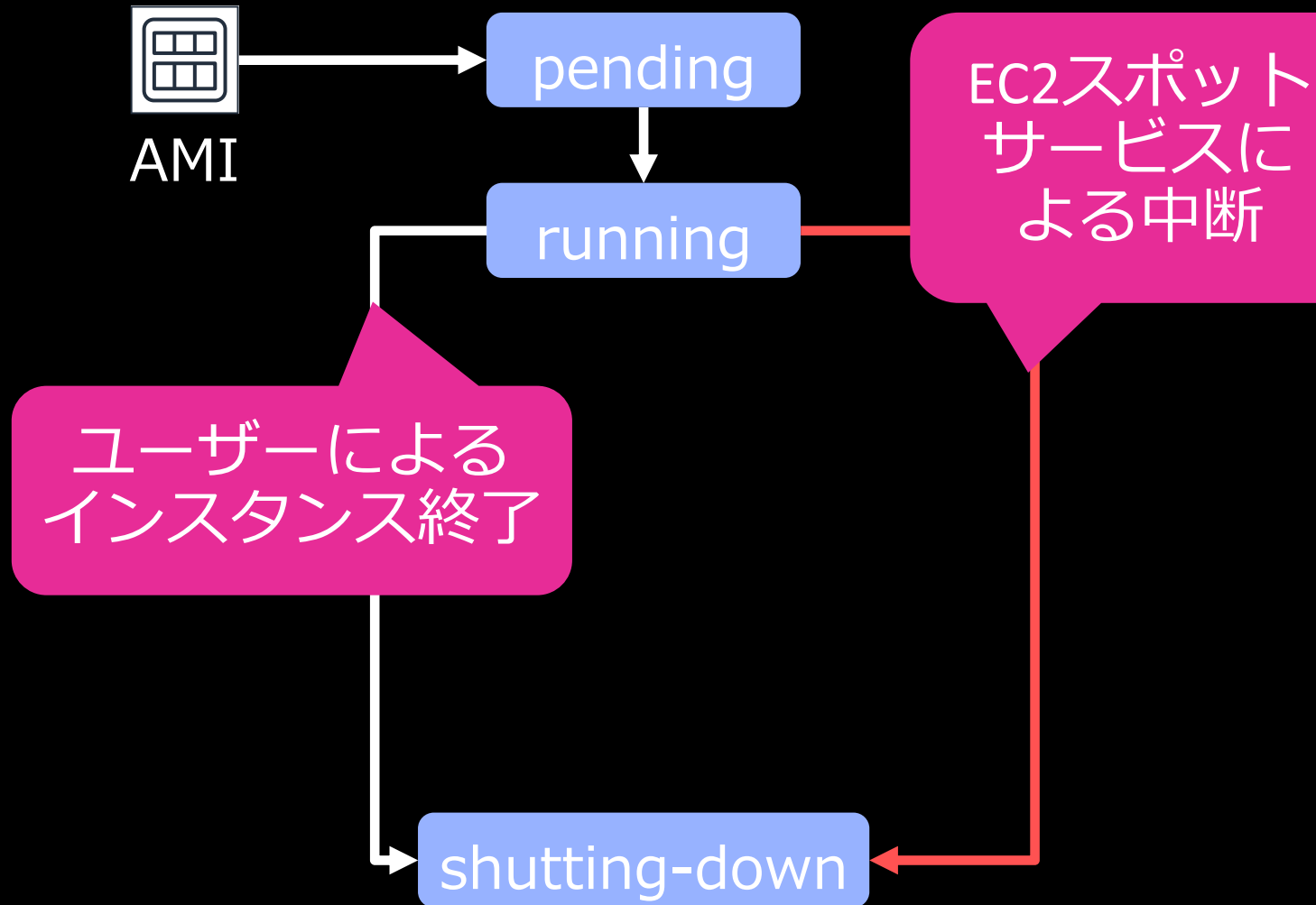


# スポットインスタンスのライフサイクル



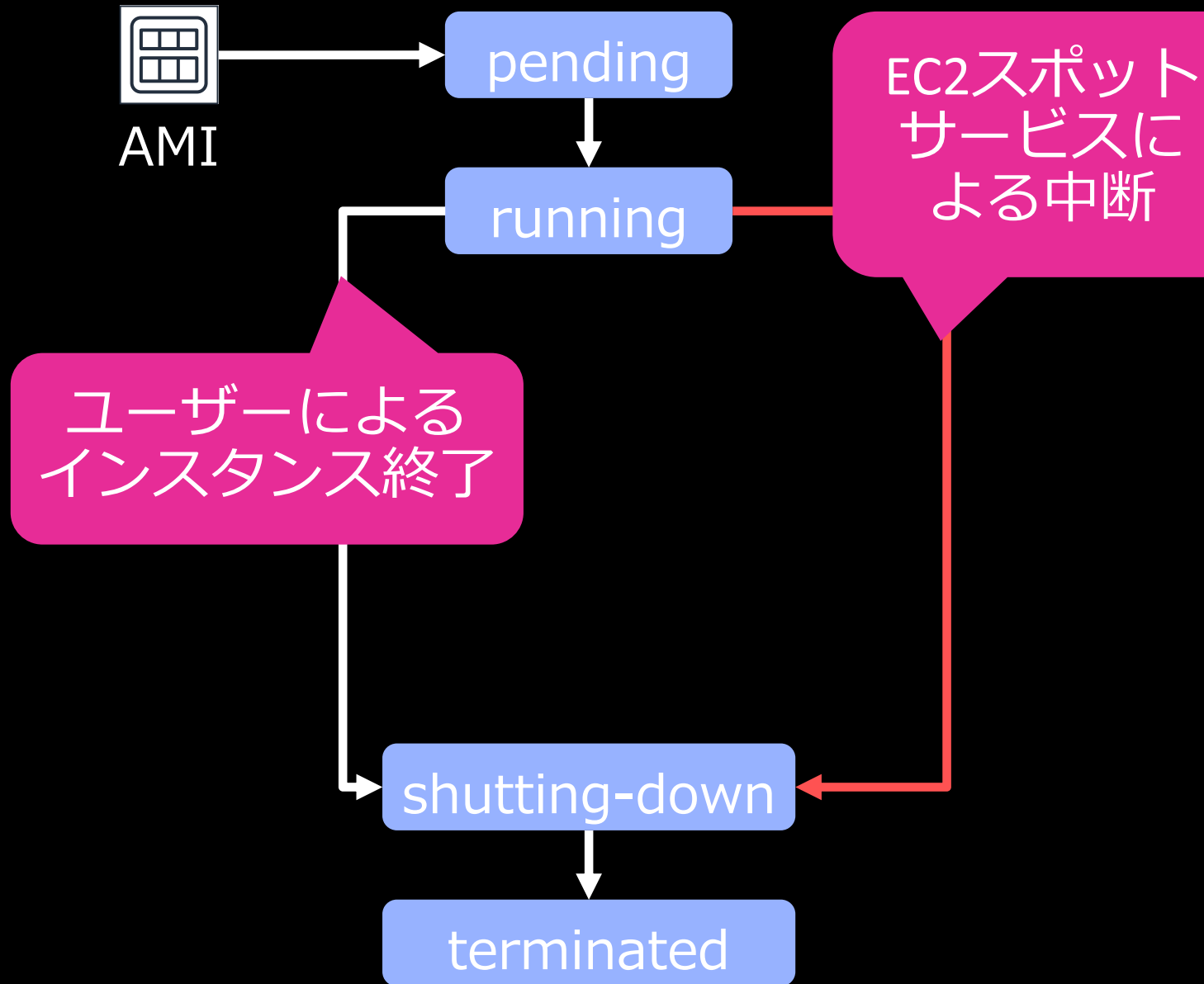
- 起動までの流れはオンデマンドインスタンスと同一
- 終了に至る道筋は2通り
  - ユーザーによるインスタンス終了

# スポットインスタンスのライフサイクル



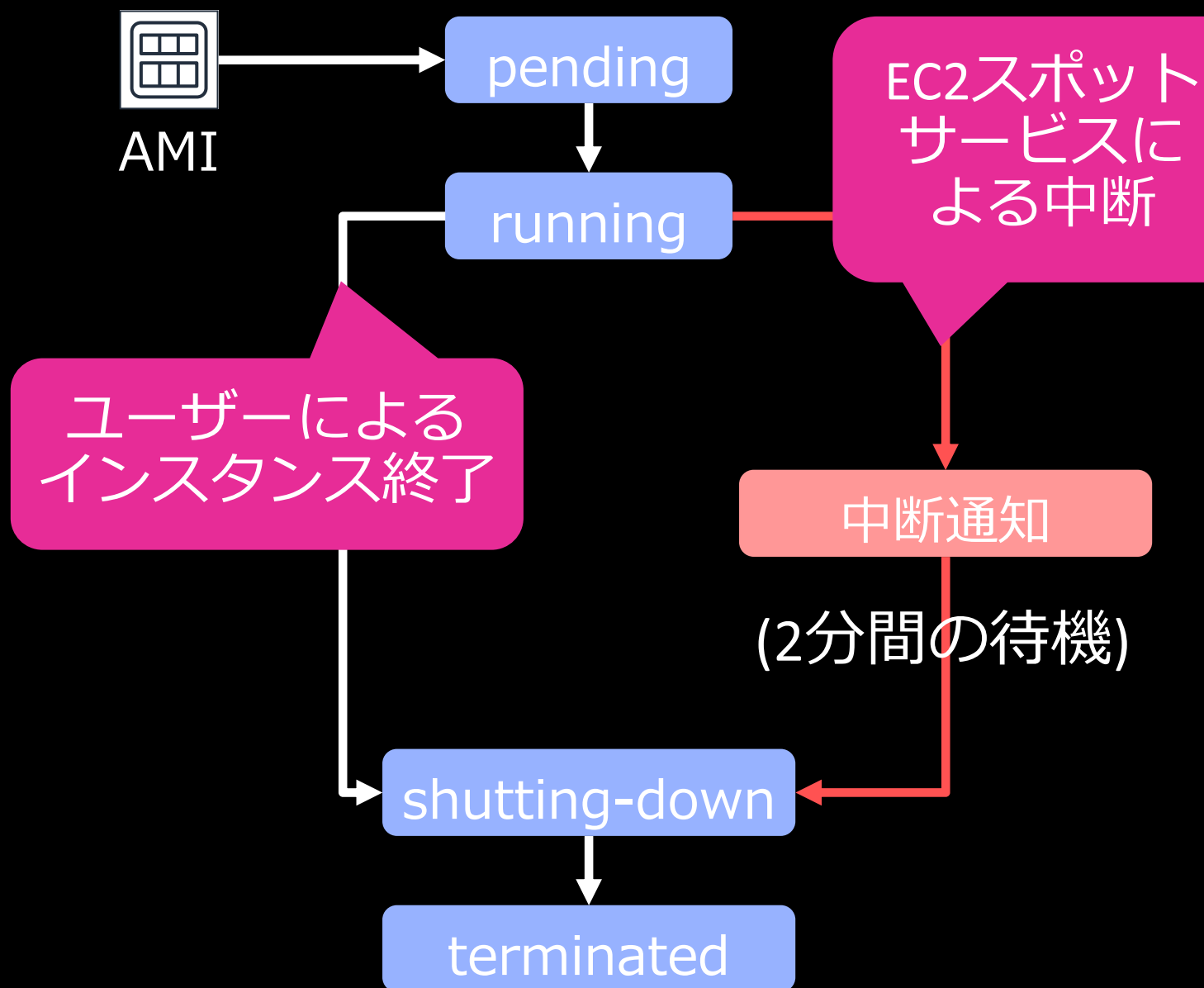
- 起動までの流れはオンデマンドインスタンスと同一
- 終了に至る道筋は2通り
  - ユーザーによるインスタンス終了
  - EC2スポットサービスによる中断

# スポットインスタンスのライフサイクル



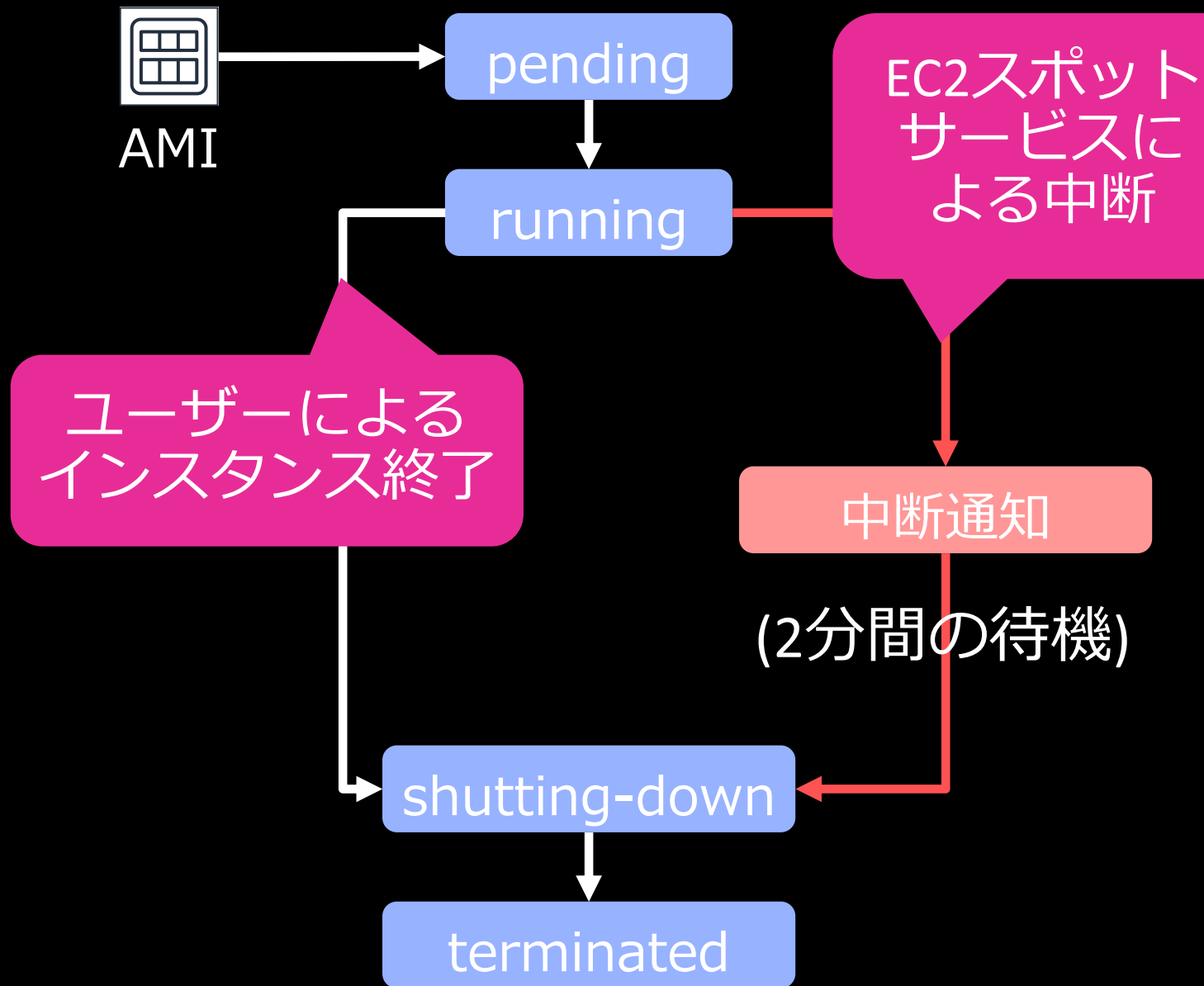
- 起動までの流れはオンデマンドインスタンスと同一
- 終了に至る道筋は2通り
  - ユーザーによるインスタンス終了
  - EC2スポットサービスによる中断
- インスタンスは削除される(デフォルト動作)

# スポットインスタンスのライフサイクル



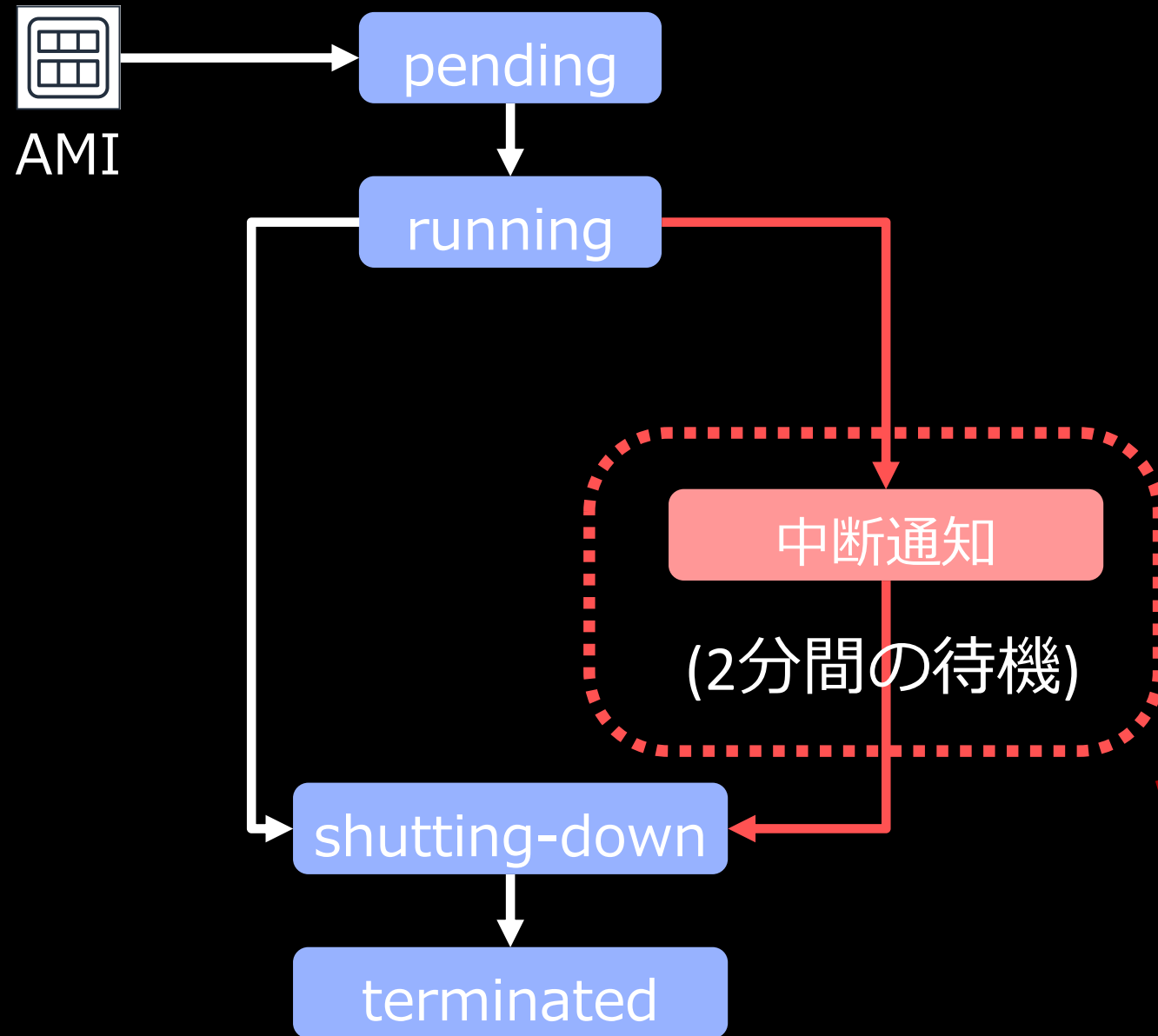
- 起動までの流れはオンデマンドインスタンスと同一
- 終了に至る道筋は2通り
  - ユーザーによるインスタンス終了
  - EC2スポットサービスによる中断
- インスタンスは削除される(デフォルト動作)

# スポットインスタンスのライフサイクル



- 起動までの流れはオンデマンドインスタンスと同一
- 終了に至る道筋は2通り
  - ユーザーによるインスタンス終了
  - EC2スポットサービスによる中断
- インスタンスは削除される(デフォルト動作)
- 92%のスポットインスタンスがユーザーにより終了されている

# 中断通知→すみやかに後処理を！！



- 仕掛けり中の処理を退避する
- 途中の計算結果をS3にアップロード
- 新規処理の受付を停止する
  - ELBから登録解除
  - コンテナステータスをDRAININGに
  - などなど

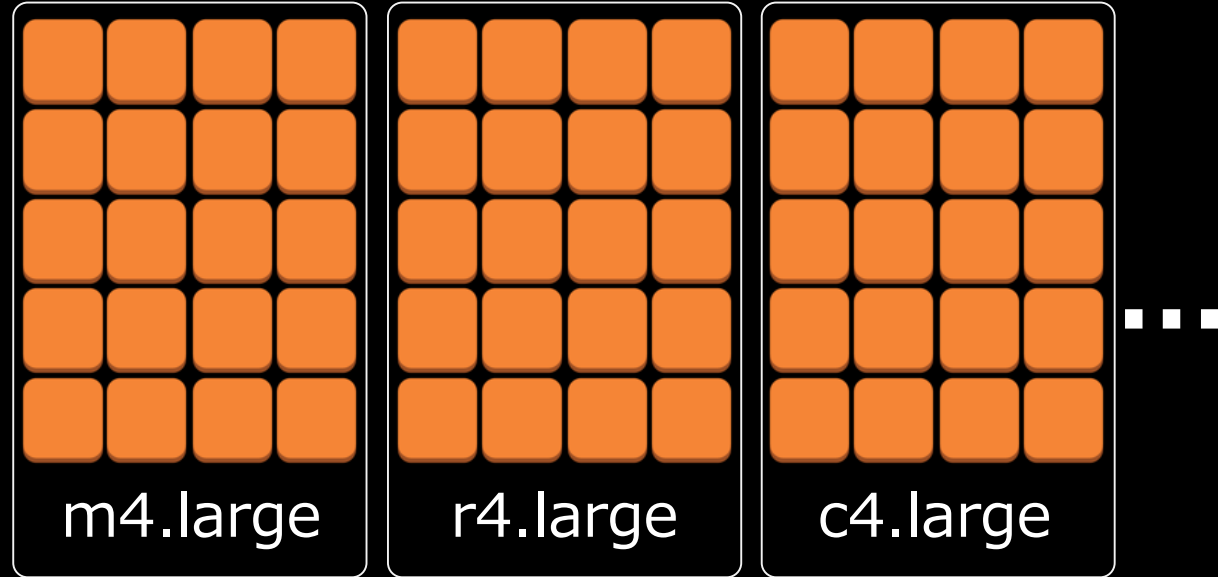
2分間の活用が  
使いこなしのポイント

# スポットインスタンスのしくみと 価格

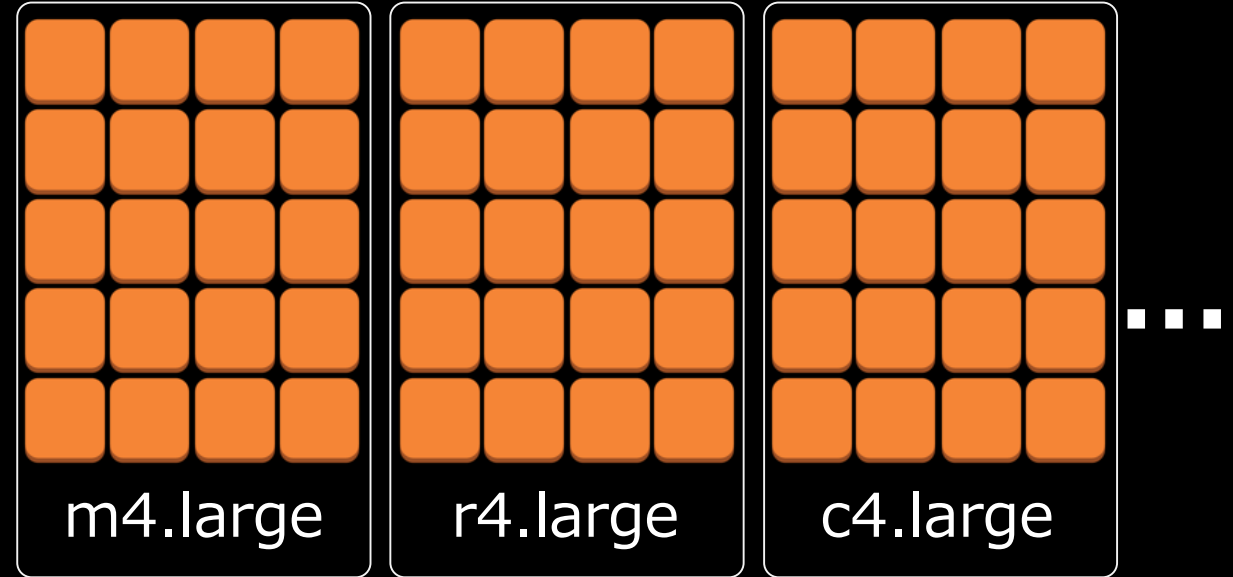


# 空きキャパシティとスポットプール(1)

AWS



`ap-northeast-1a`

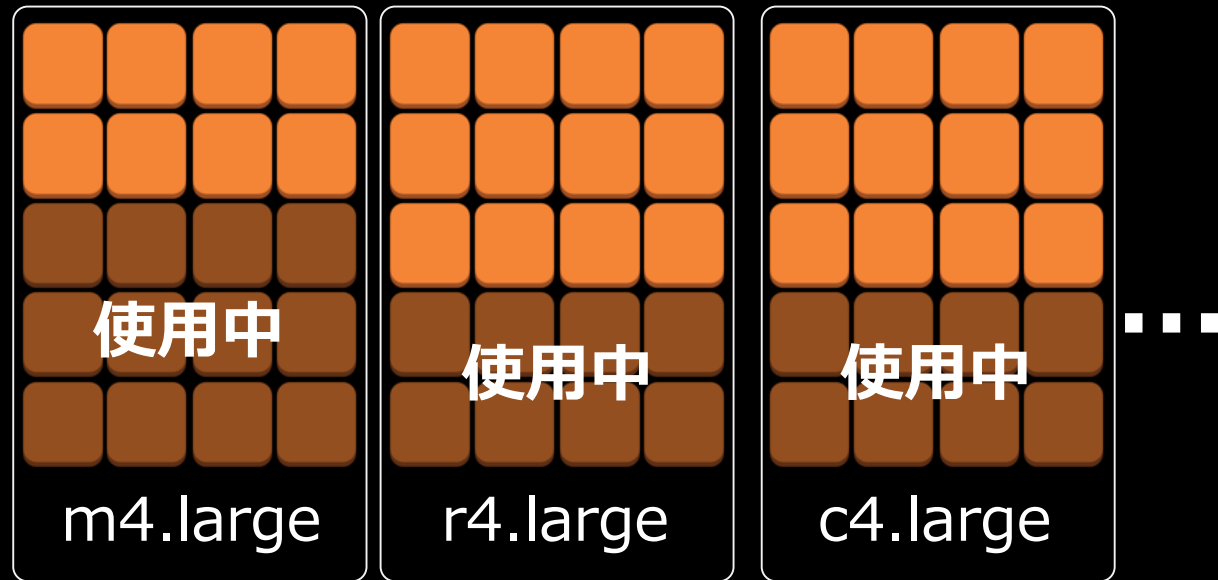


`ap-northeast-1c`

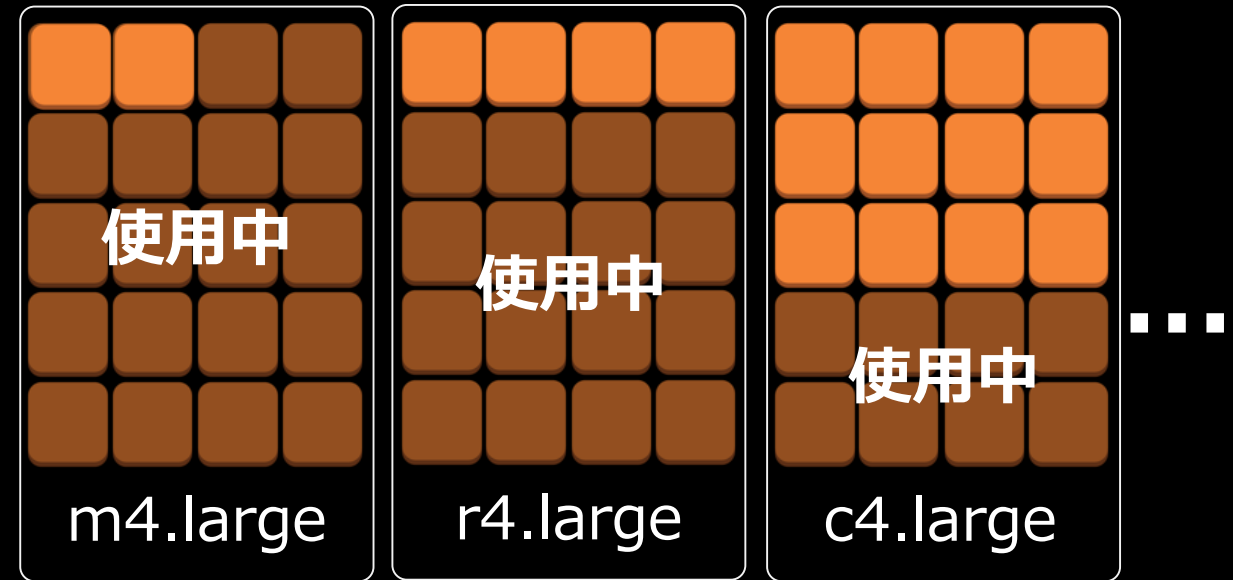
(Tokyo Region)

# 空きキャパシティとスポットプール(2)

AWS



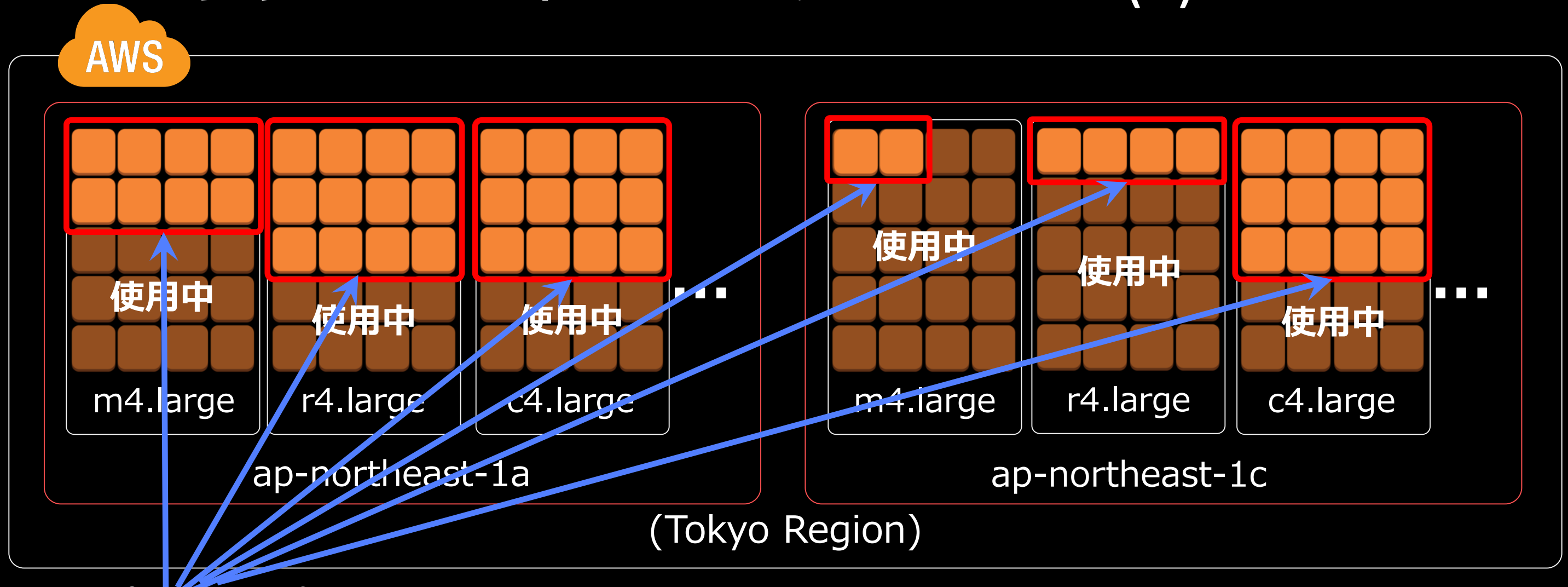
ap-northeast-1a



ap-northeast-1c

(Tokyo Region)

# 空きキャパシティとスポットプール(3)



**スポットプール**...リージョン、アベイラビリティゾーン(AZ), インスタンスタイプごとに独立した空きキャパシティ

# スポットインスタンスの価格設定履歴

- EC2マネジメントコンソール「スポットリクエスト」→「価格設定履歴」

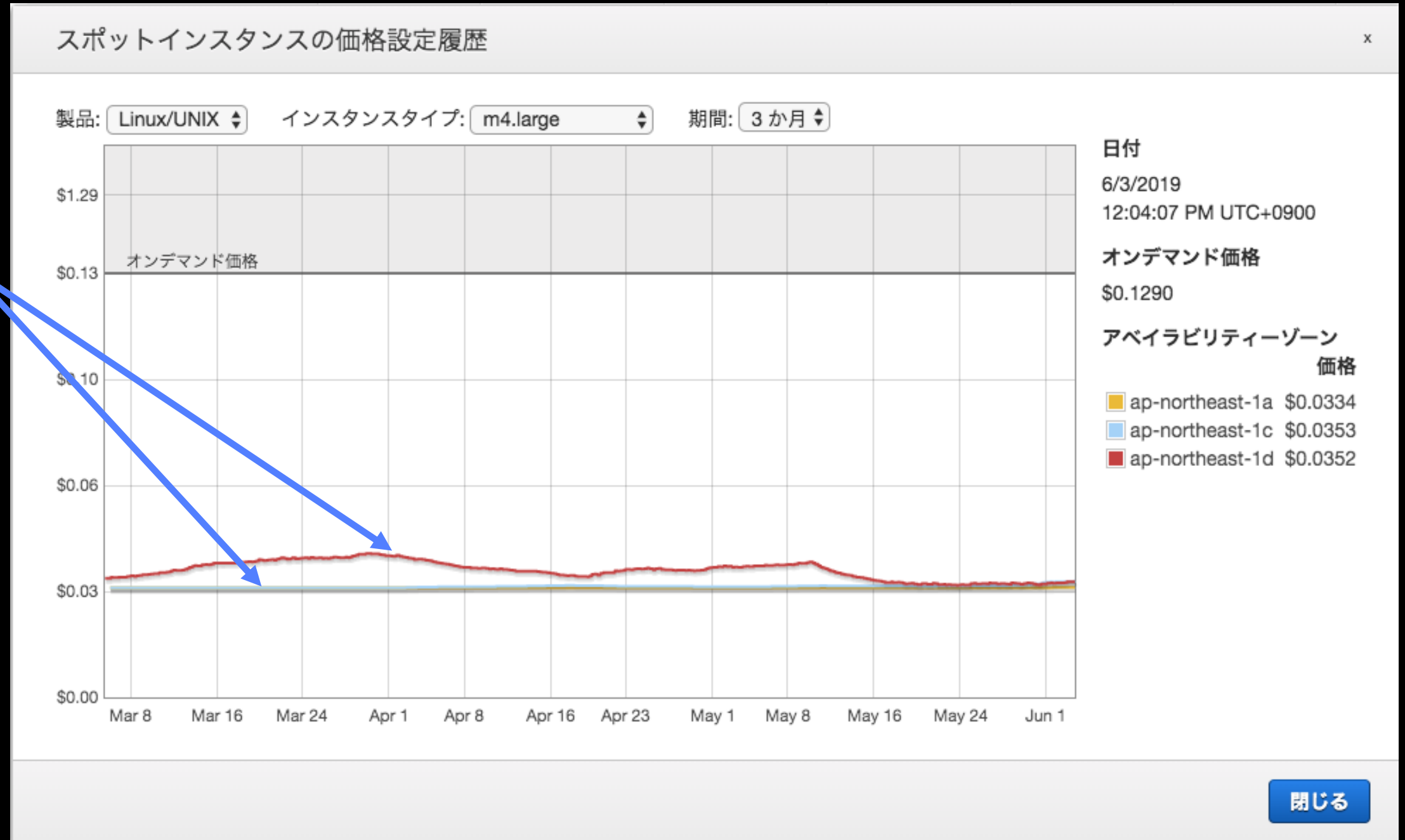


# スポットインスタンス価格の特徴



# スポットインスタンス価格の特徴

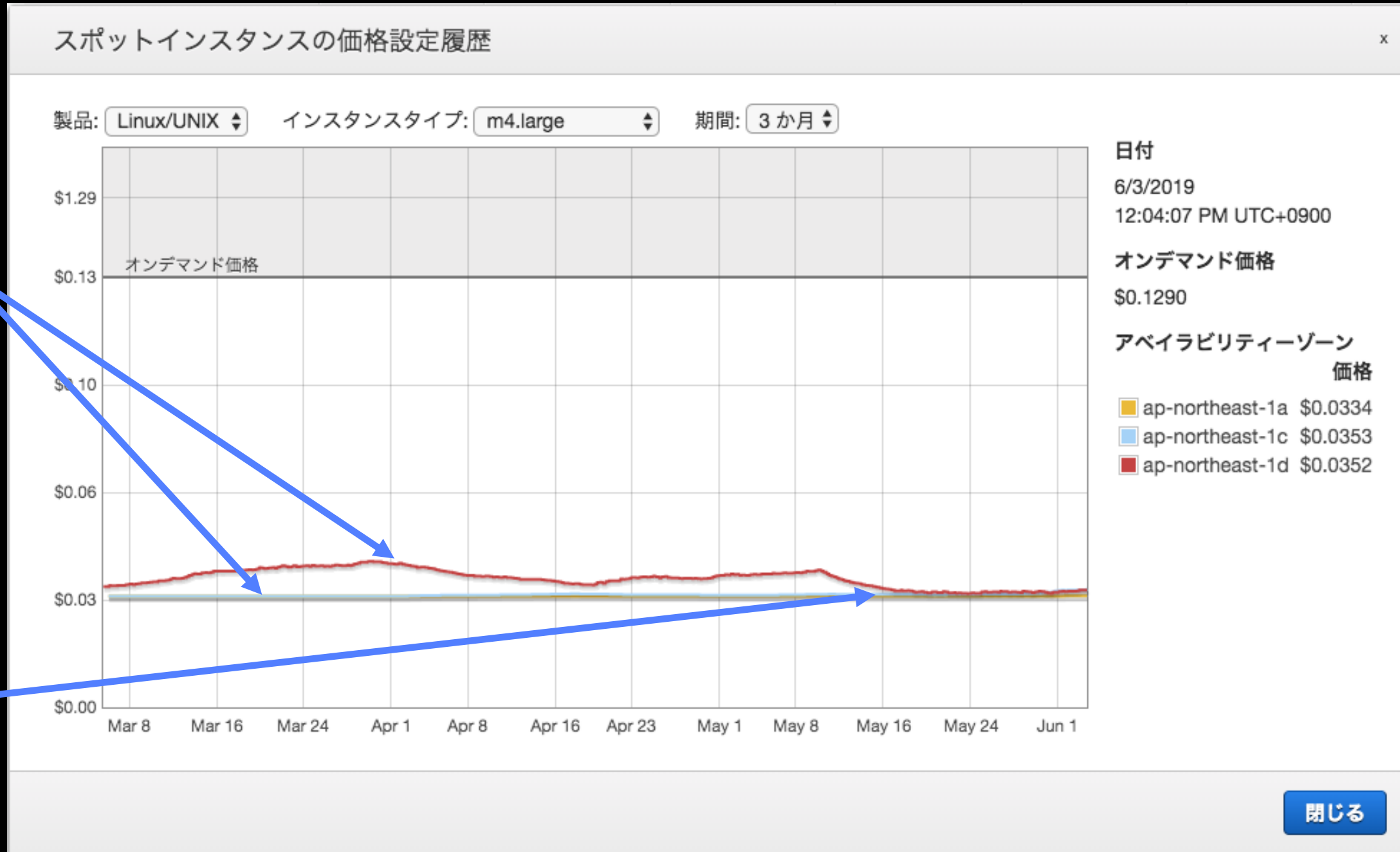
アベイラビリティゾーンごとに価格が異なる



# スポットインスタンス価格の特徴

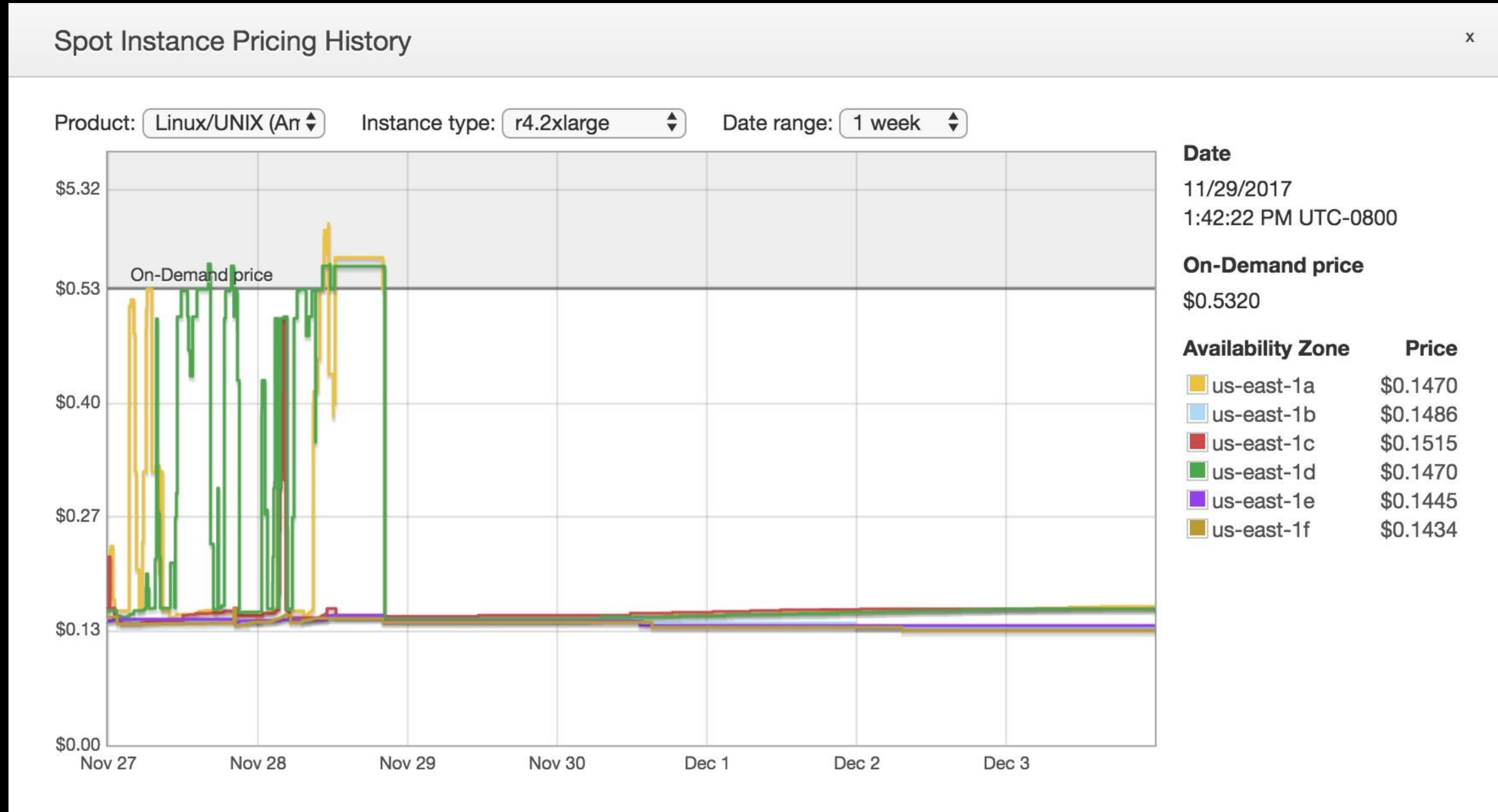
アベイラビリティゾーンごとに価格が異なる

緩やかに変化

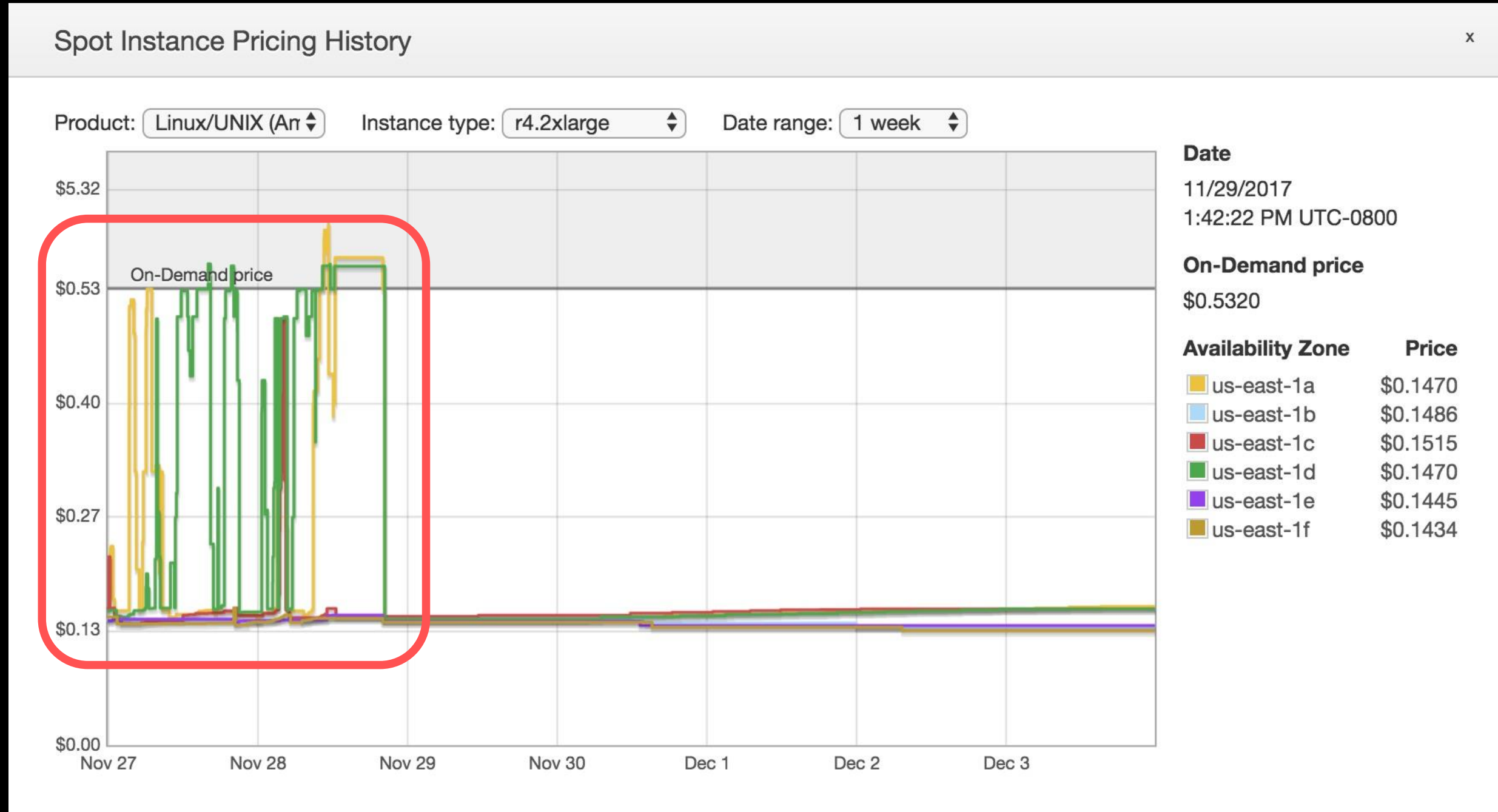




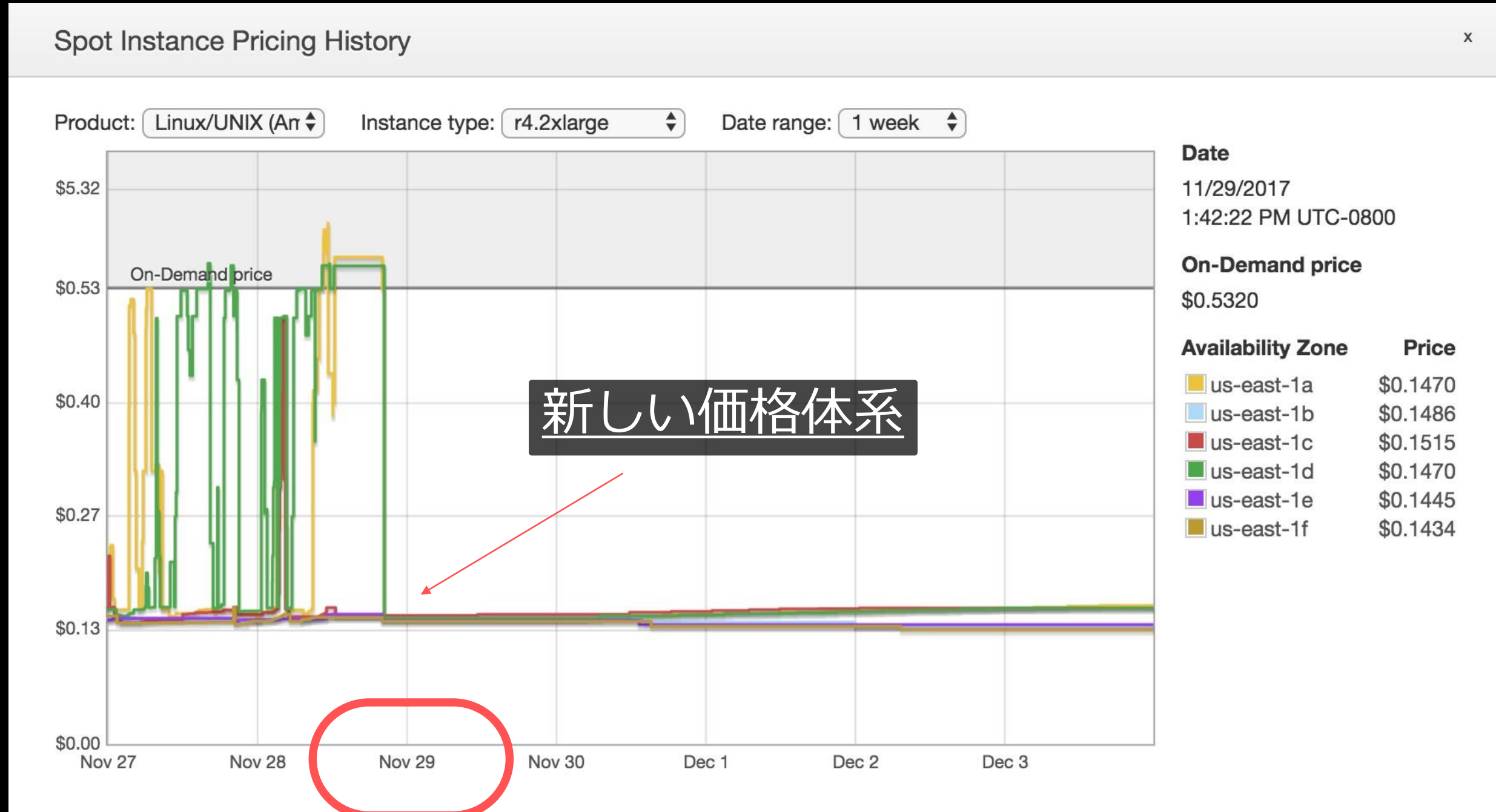
# スポット価格安定化 - 2017年11月



# スポット価格安定化 - 2017年11月

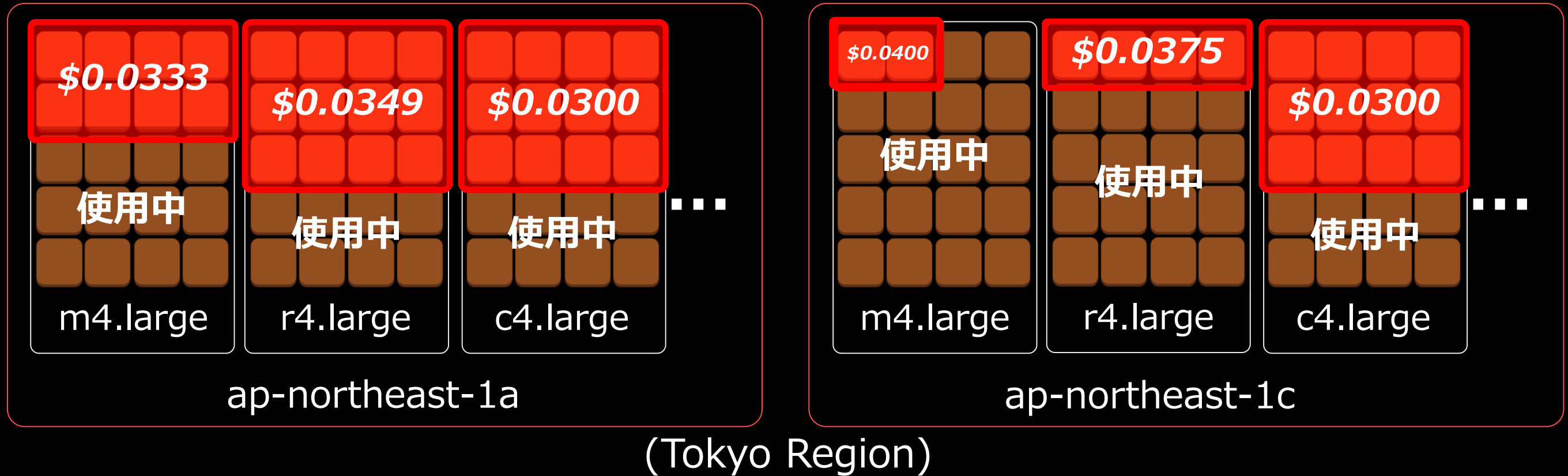


# スポット価格安定化 - 2017年11月



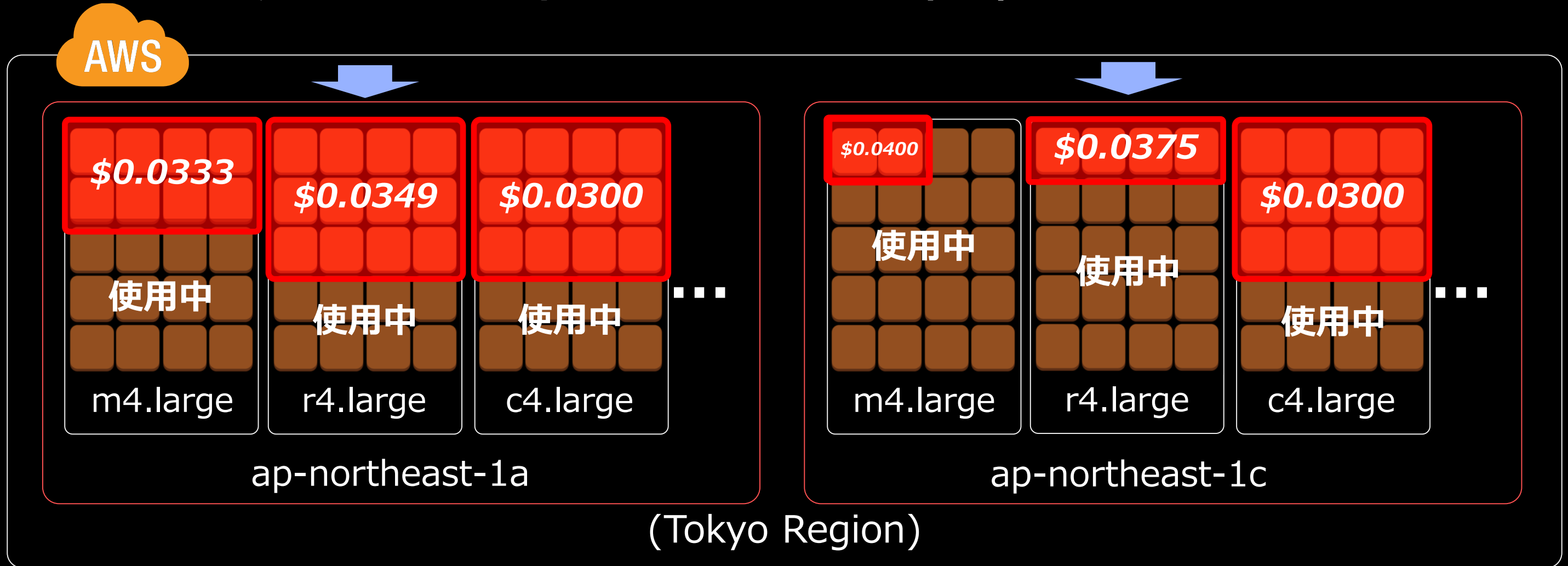
# 空きキャパシティとスポット価格

AWS



**スポット価格**...その時点のスポットインスタンス価格。スポットプールごとの需要と供給で決まる

# 空きキャパシティとスポット価格



**スポット価格**...その時点のスポットインスタンス価格。スポットプールごとの需要と供給で決まる

# スポットインスタンス価格の決まり方

スポットインスタンス価格は長期供給と需要に基づいて徐々に調整される



# 今のスポットインスタンスは入札不要

AWS

- 東京リージョン1aでc4.largeを起動したい

\$0.0300

使用中

使用中

使用中

m4.large

r4.large

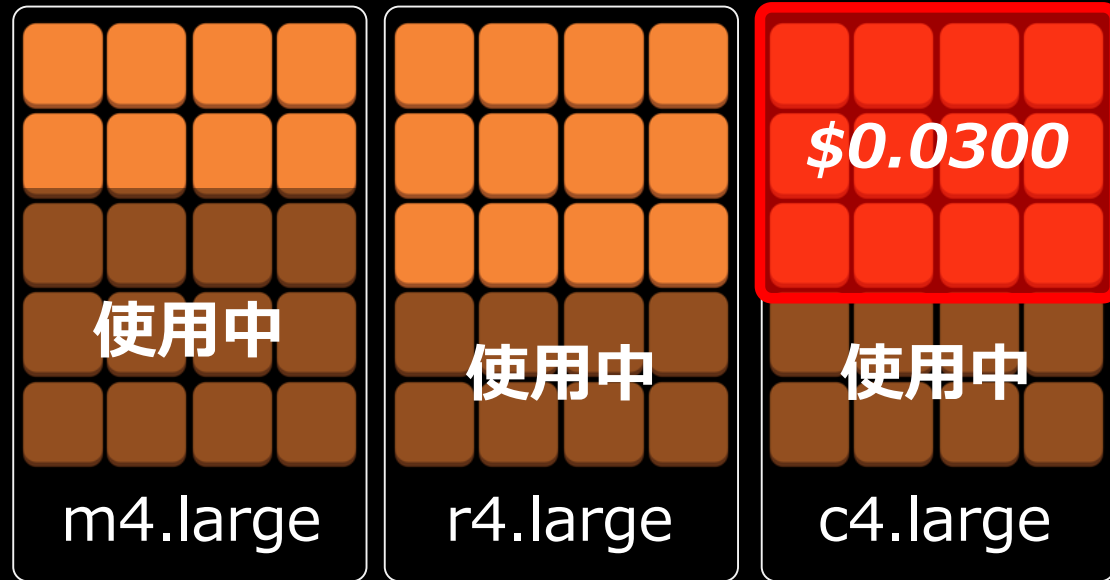
c4.large

ap-northeast-1a

(Tokyo Region)

# 今のスポットインスタンスは入札不要

AWS



ap-northeast-1a

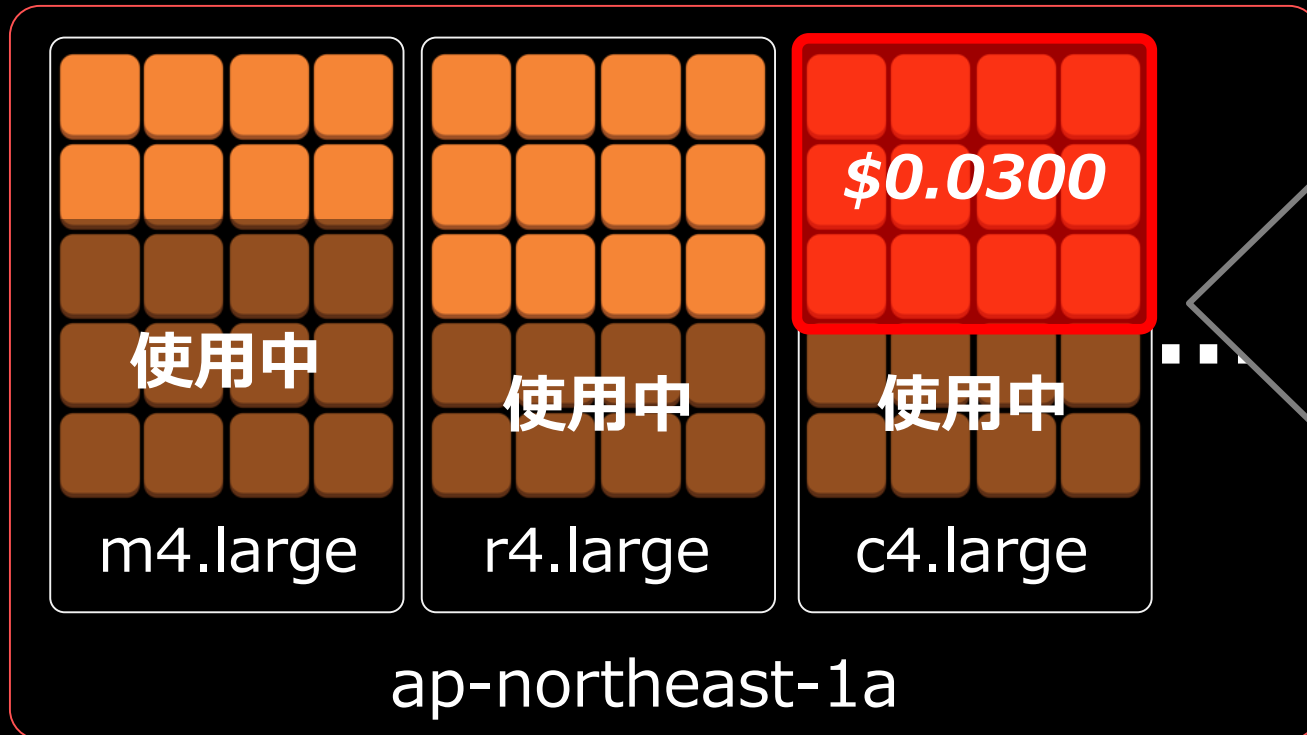
(Tokyo Region)

- 東京リージョン1aでc4.largeを起動したい
- リクエスト時に「上限価格」を指定できる



# 今のスポットインスタンスは入札不要

AWS



(Tokyo Region)

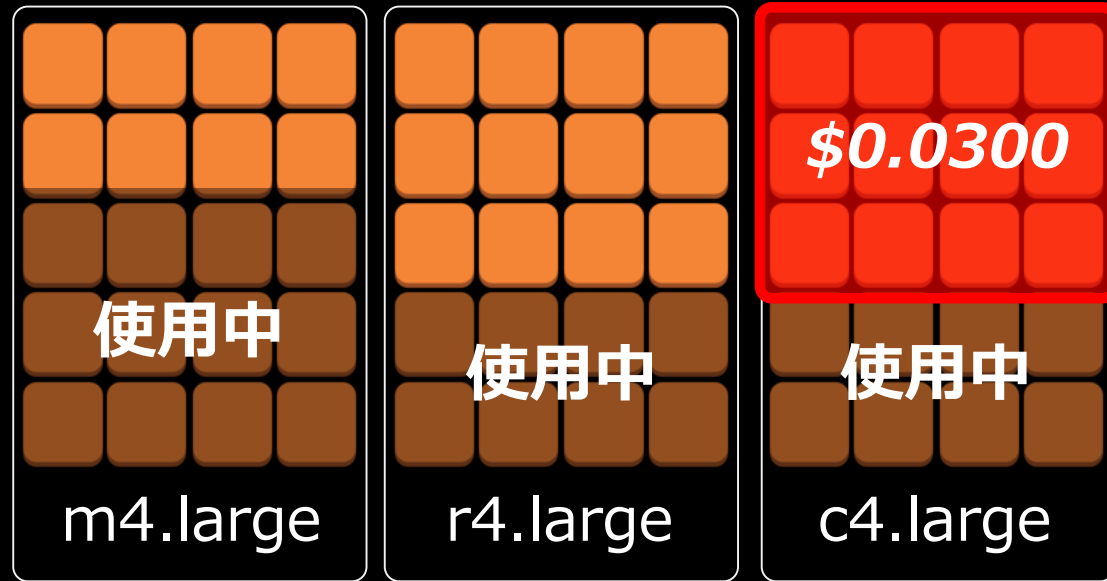
- 東京リージョン1aでc4.largeを起動したい
- リクエスト時に「上限価格」を指定できる
- デフォルトはオンデマンド価格

**上限価格**…スポットインスタンスに支払っても良いと思う最大料金。  
デフォルトはオンデマンドインスタンス価格

# 今のスポットインスタンスは入札不要

AWS

オンデマンド=\$0.1260



ap-northeast-1a

(Tokyo Region)

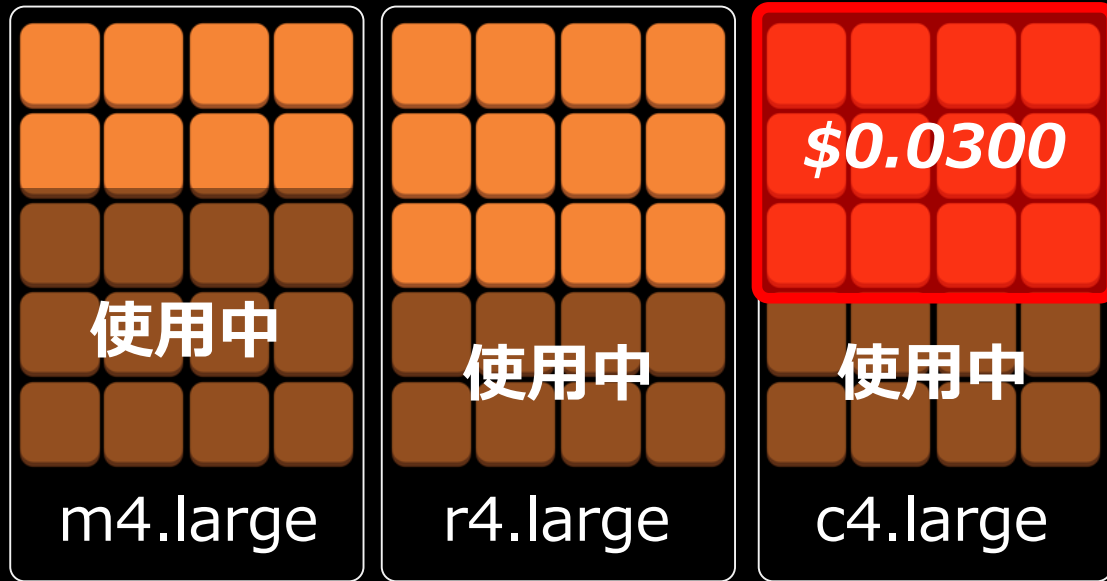
- 東京リージョン1aでc4.largeを起動したい
- リクエスト時に「上限価格」を指定できる
- デフォルトはオンデマンド価格

**上限価格**…スポットインスタンスに支払っても良いと思う最大料金。  
デフォルトはオンデマンドインスタンス価格

# 今のスポットインスタンスは入札不要

AWS

オンデマンド=\$0.1260



ap-northeast-1a

(Tokyo Region)

- 東京リージョン1aでc4.largeを起動したい
- リクエスト時に「上限価格」を指定できる
- デフォルトはオンデマンド価格

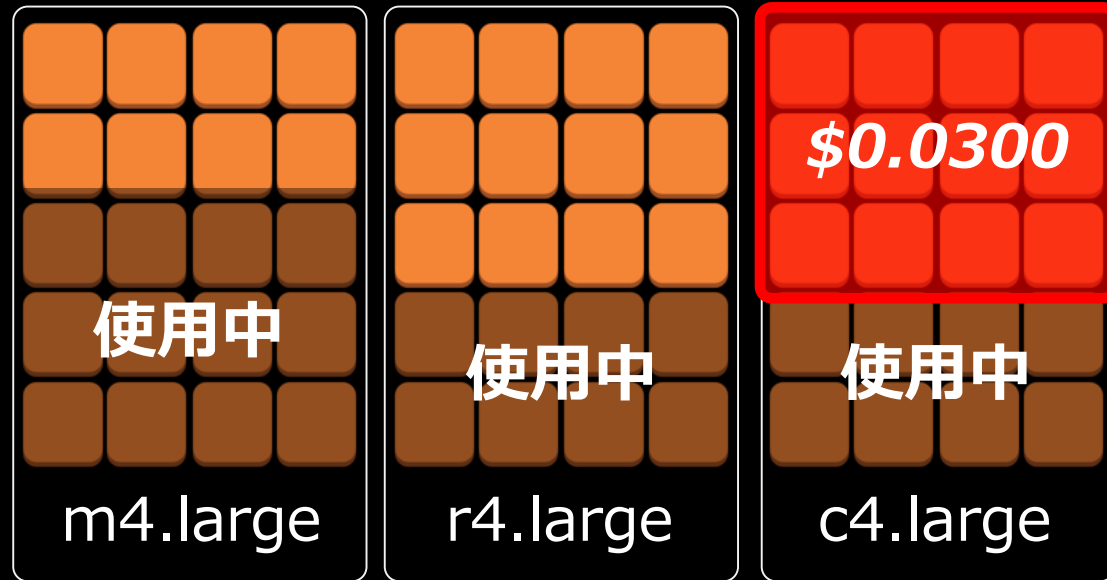
→入札不要モデルになりました

上限価格…スポットインスタンスに支払っても良いと思う最大料金。  
デフォルトはオンデマンドインスタンス価格

# スポットインスタンスの起動条件

AWS

オンデマンド=\$0.1260



ap-northeast-1a

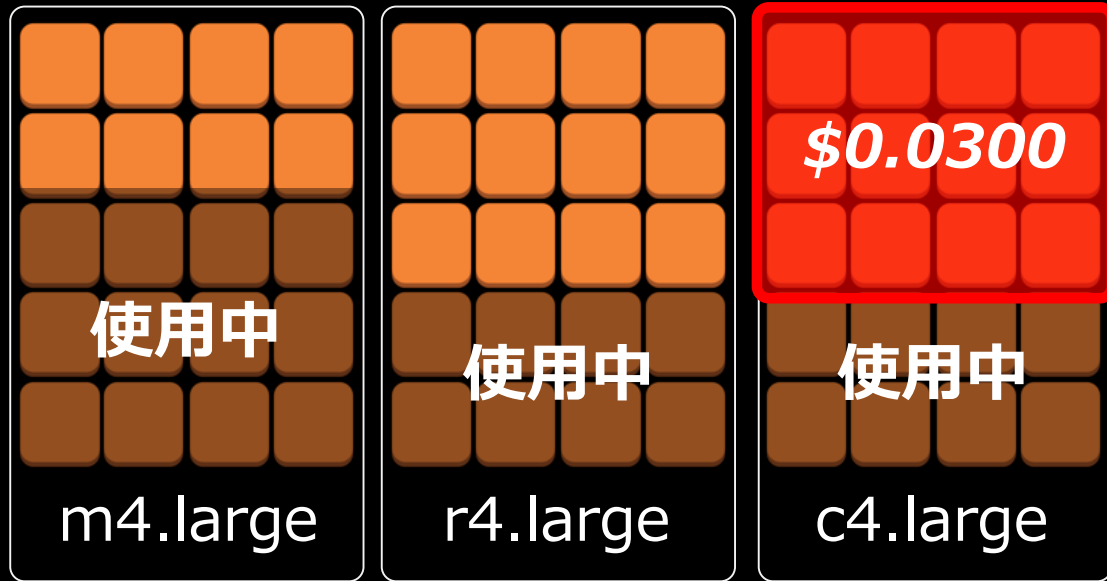
(Tokyo Region)

- 東京リージョン1aでc4.largeを起動したい

# スポットインスタンスの起動条件

AWS

オンデマンド=\$0.1260



ap-northeast-1a

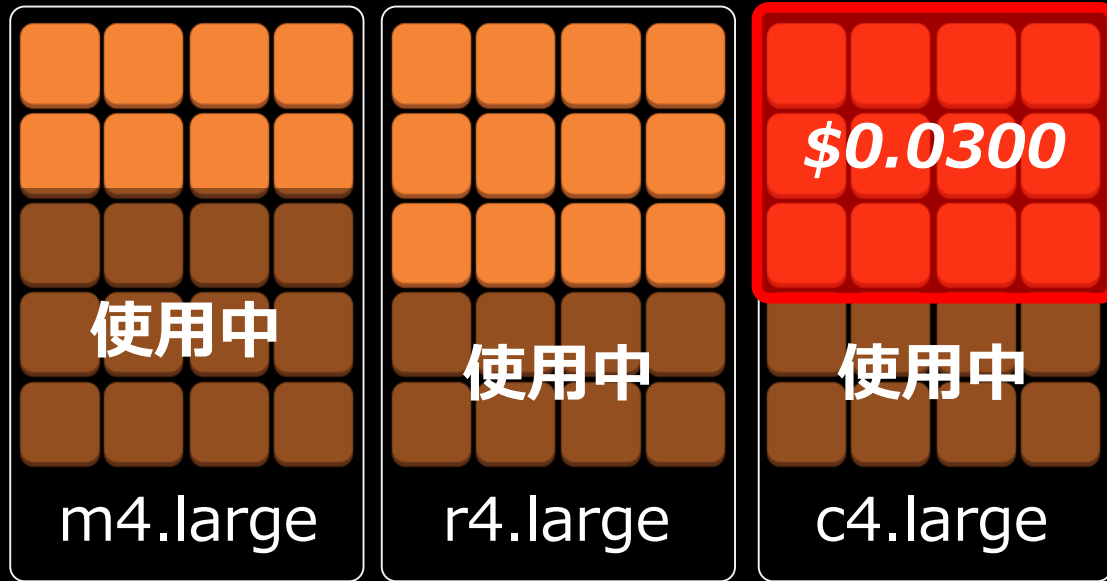
(Tokyo Region)

- 東京リージョン1aでc4.largeを起動したい
  - スポット価格(\$0.0300)が上限価格(\$0.1260)以内

# スポットインスタンスの起動条件

AWS

オンデマンド=\$0.1260



ap-northeast-1a

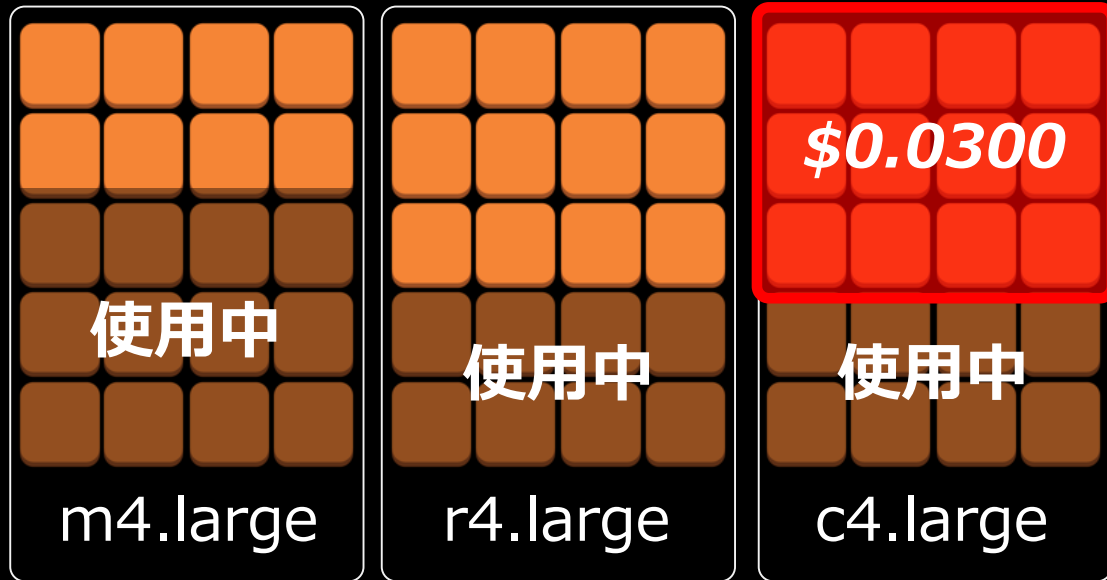
(Tokyo Region)

- 東京リージョン1aでc4.largeを起動したい
  - スポット価格(\$0.0300)が上限価格(\$0.1260)以内
  - スポットプールに空きあり

# スポットインスタンスの起動条件

AWS

オンデマンド=\$0.1260



ap-northeast-1a

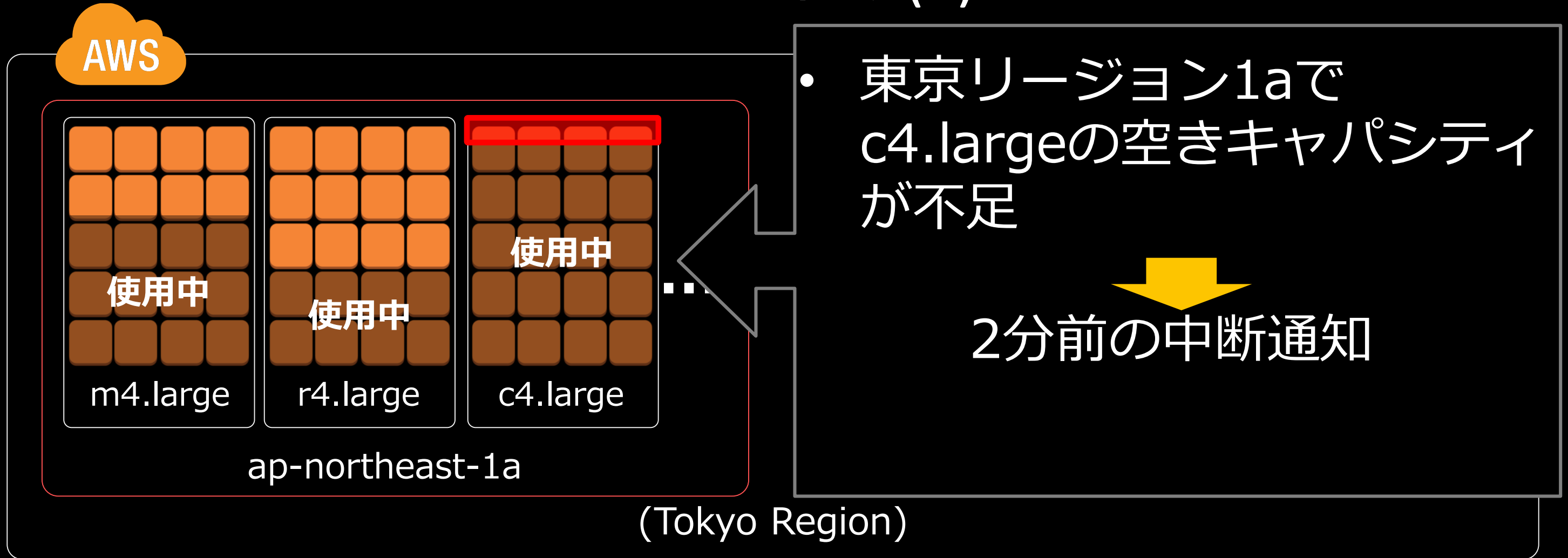
(Tokyo Region)

- 東京リージョン1aでc4.largeを起動したい
  - スポット価格(\$0.0300)が上限価格(\$0.1260)以内
  - スポットプールに空きあり

リクエストが即座に受理され  
起動できる

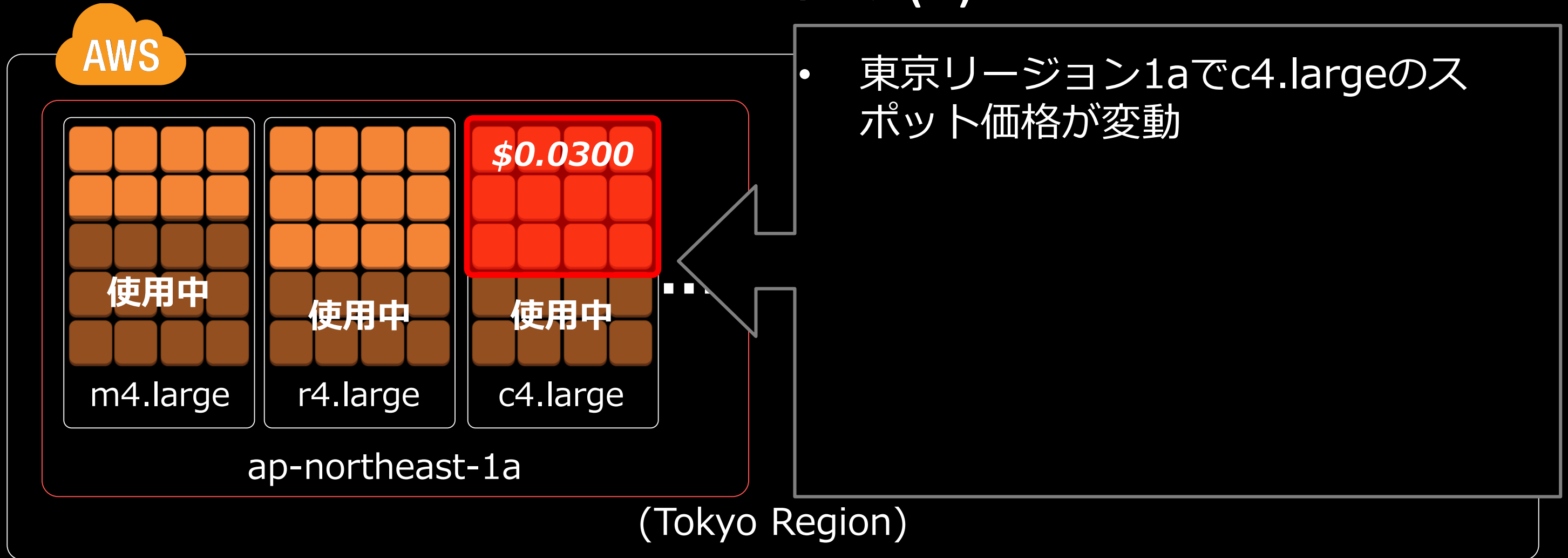


# スポットインスタンスの中断(1)

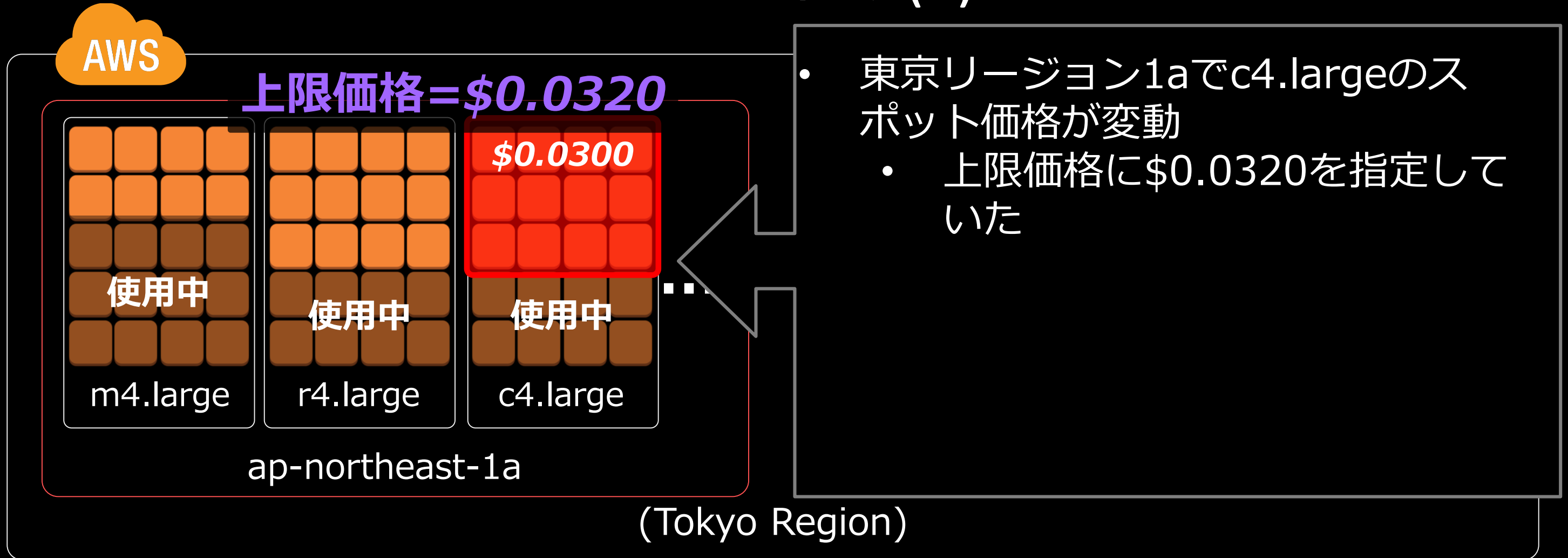


中断理由(1)・・・Amazon EC2のキャパシティ要件(オンデマンドインスタンスの需要増加など)

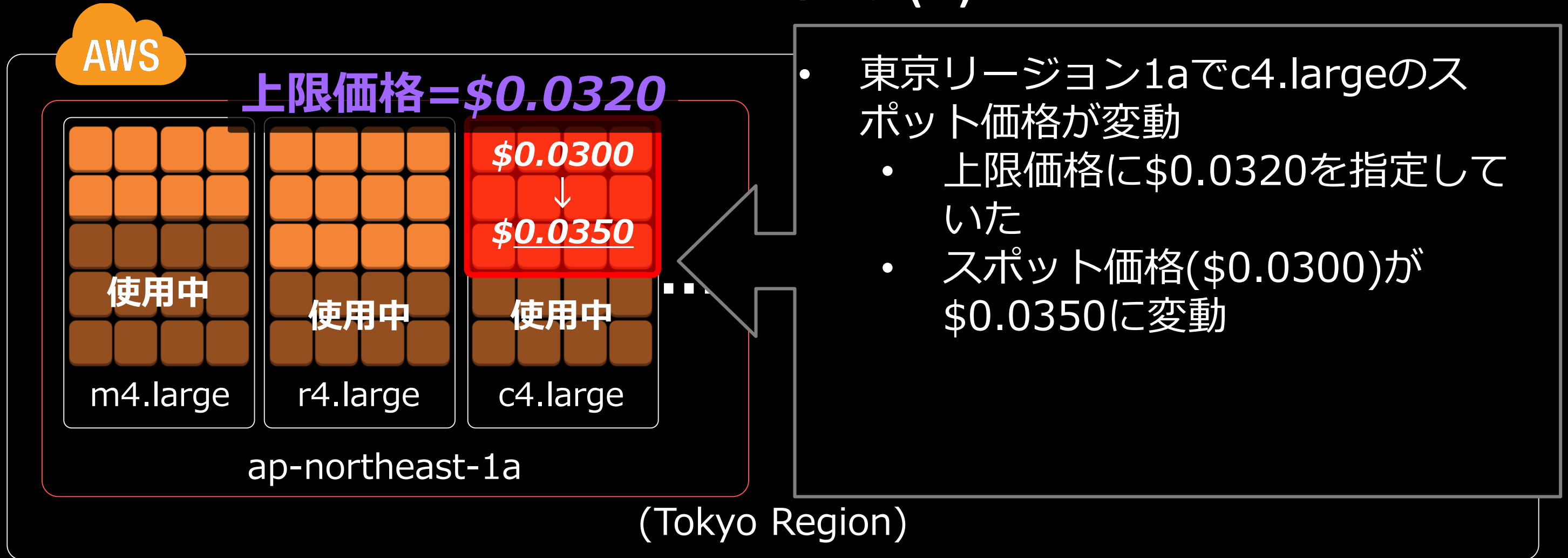
# スポットインスタンスの中断(2)



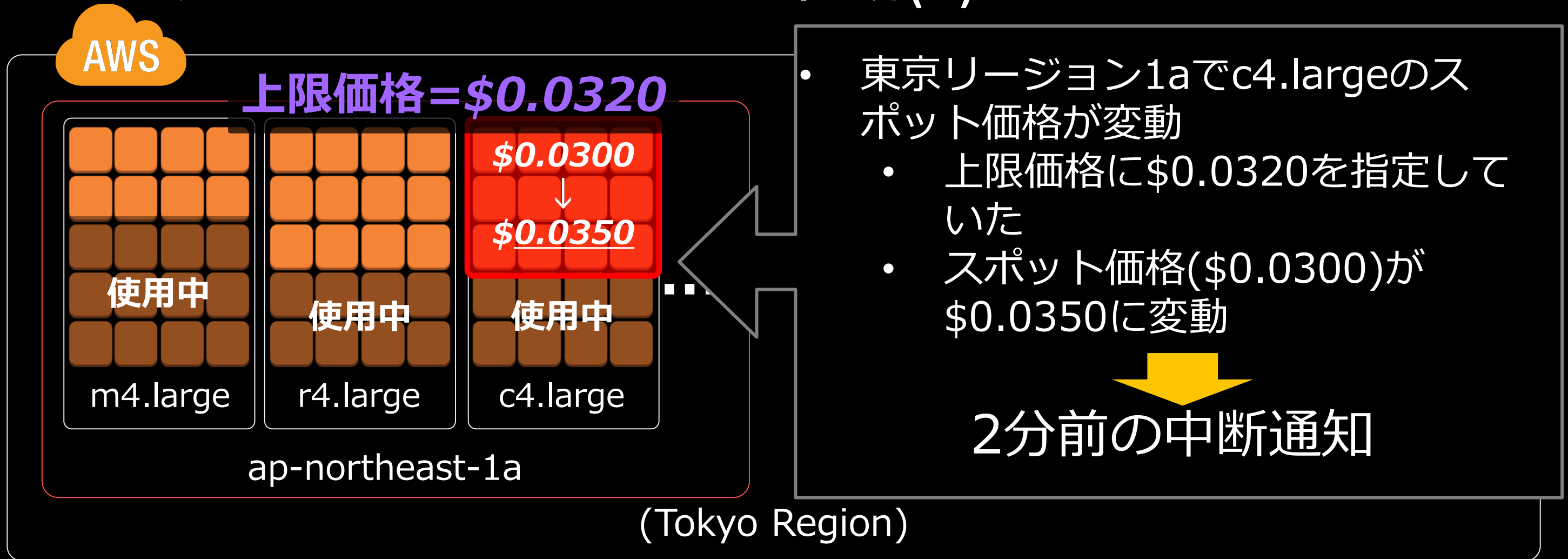
# スポットインスタンスの中断(2)



# スポットインスタンスの中断(2)



# スポットインスタンスの中断(2)



中断理由(2)・・・スポット価格が変動した結果、指定していた上限料金を上回った

# スポットインスタンス 実践編

# スポットインスタンス 実践編

- 中断のハンドリング
- スポットインスタンス活用の4原則
- スポットインスタンスの活用シーン



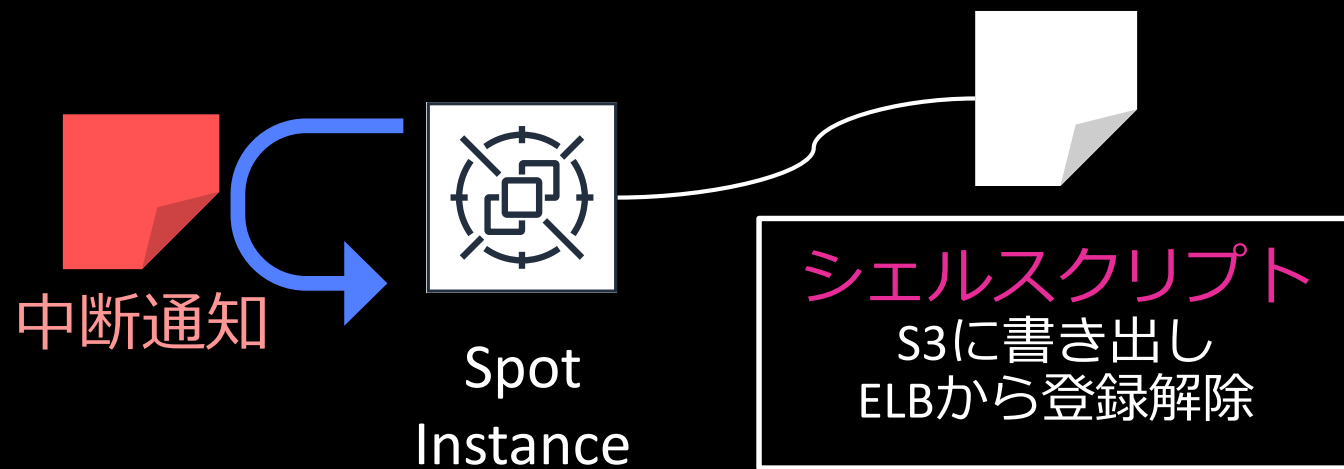
# 中断のハンドリング

# 中断通知の例

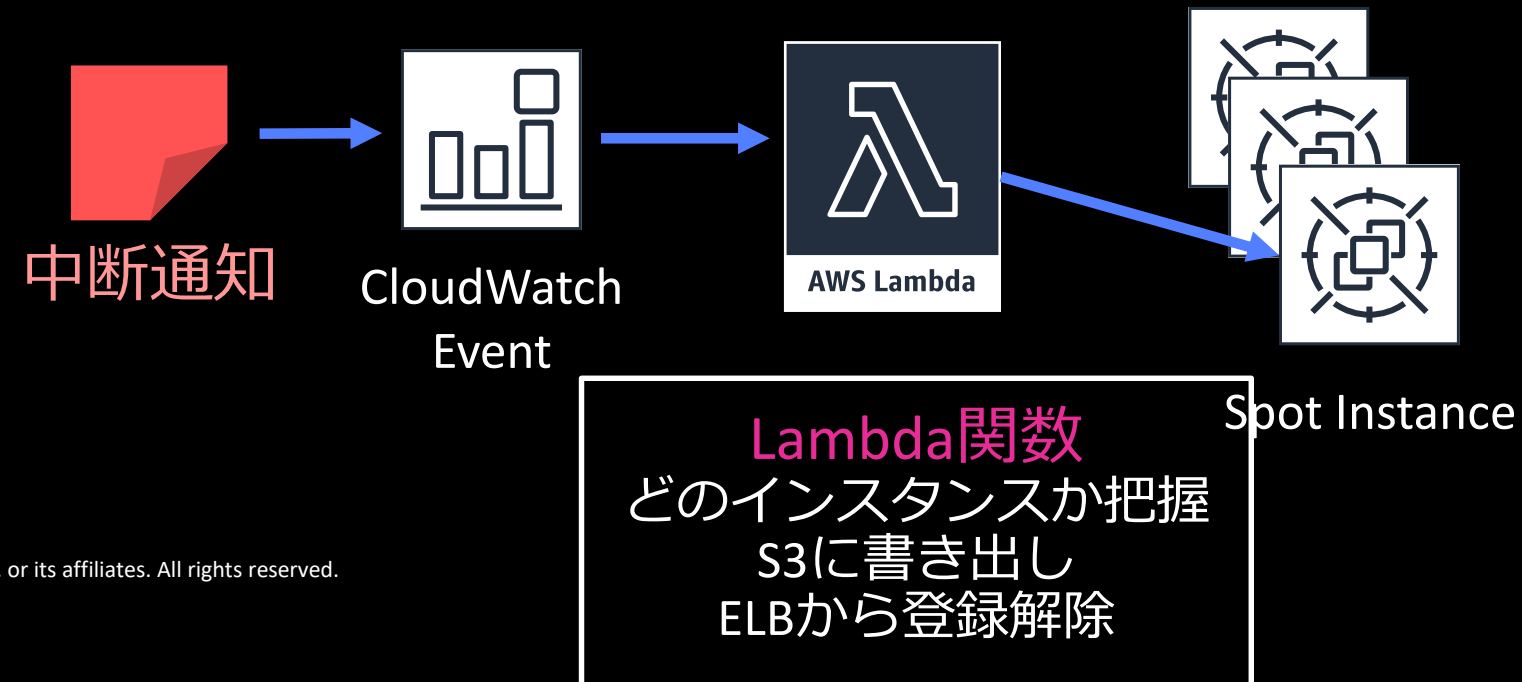
```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Spot Instance Interruption Warning",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2019-06-12T12:00:00Z",
  "region": "us-east-2",
  "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-1234567890abcdef0"],
  "detail": {
    "instance-id": "i-1234567890abcdef0",
    "instance-action": "action"
  }
}
```

# 中断通知 - 受信の2つの方法

- インスタンスメタデータ
  - インスタンス内部からアクセス
    - 5秒おきのチェック(pull型)
  - 後処理
    - シェルスクリプトを書く
    - 自インスタンスのことなので自分の処理を記述

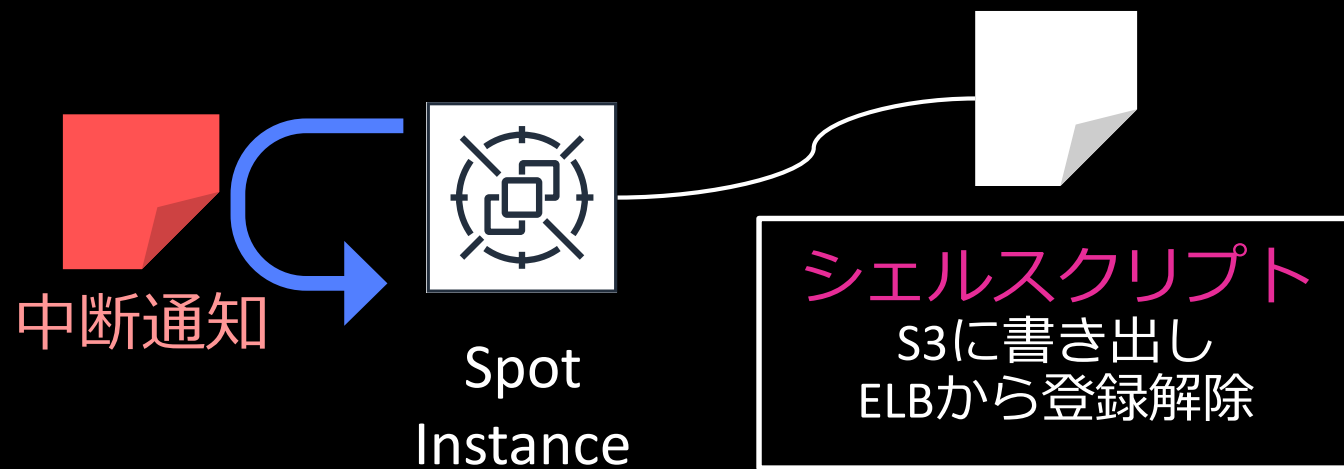


- CloudWatch Events
  - インスタンス外部からアクセス
    - 自動受信(push型)
  - 後処理
    - Lambda関数を書く
    - どのインスタンスが中断対象なのかをまず把握
    - 汎用的に記述

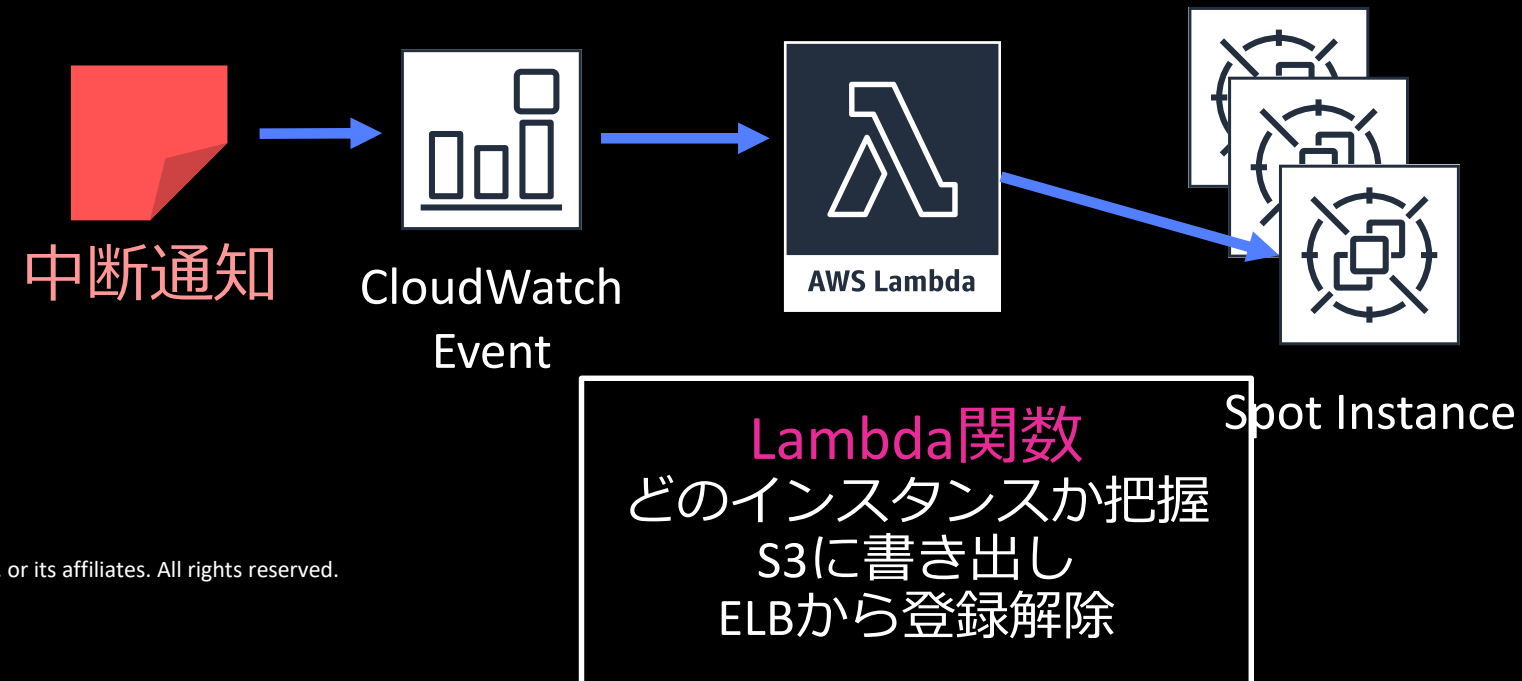


# 中断通知 - 受信の2つの方法

- インスタンスメタデータ
  - インスタンス内部からアクセス
    - 5秒おきのチェック(pull型)
  - 後処理
    - シェルスクリプトを書く
    - 自インスタンスのことなので自分の処理を記述

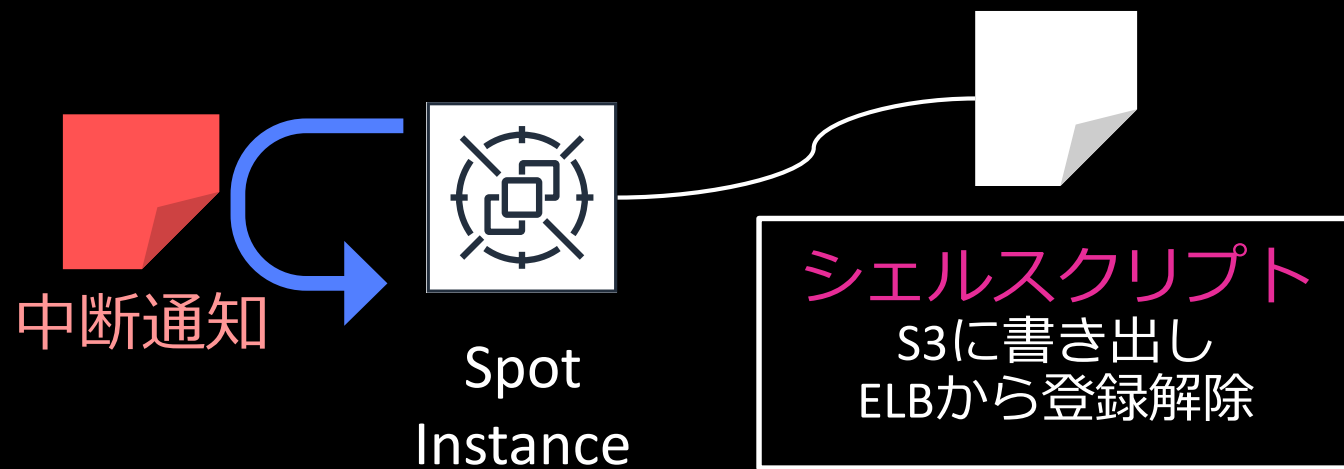


- CloudWatch Events
  - インスタンス外部からアクセス
    - 自動受信(push型)
  - 後処理
    - Lambda関数を書く
    - どのインスタンスが中断対象なのかをまず把握
    - 汎用的に記述

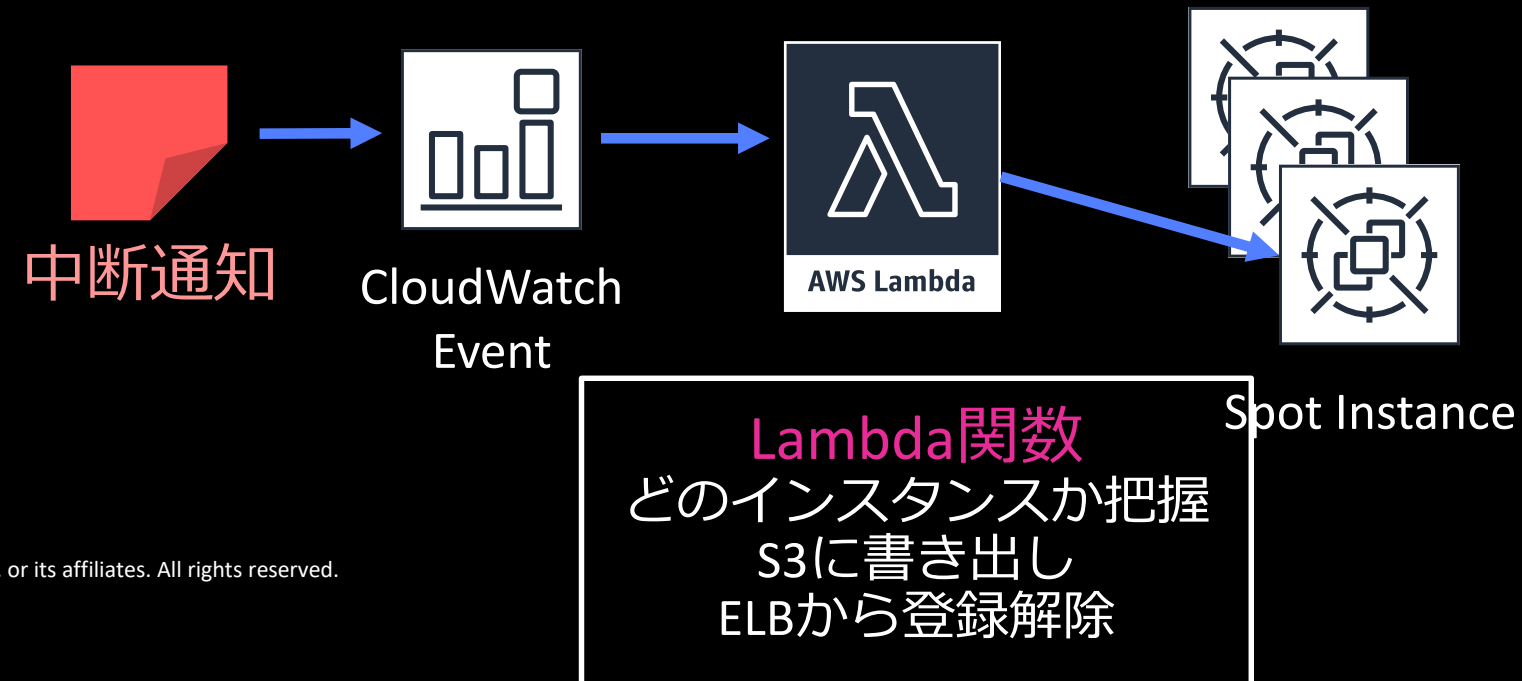


# 中断通知 - 受信の2つの方法

- インスタンスメタデータ
  - インスタンス内部からアクセス
  - 5秒おきのチェック(pull型)
- 後処理
  - シェルスクリプトを書く
  - 自インスタンスのことなので自分の処理を記述



- CloudWatch Events
  - インスタンス外部からアクセス
  - 自動受信(push型)
- 後処理
  - Lambda関数を書く
  - どのインスタンスが中断対象なのかをまず把握
  - 汎用的に記述



# <参考> 中断通知の受信 - インスタンスメタデータから

- インスタンスメタデータサービス
  - 実行中のインスタンスそのものに関する情報・属性を提供
  - <http://169.254.169.254/latest/meta-data/> 以下に格納される
  - [https://docs.aws.amazon.com/ja\\_jp/AWSEC2/latest/UserGuide/ec2-instance-metadata.html](https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/ec2-instance-metadata.html)
- 中断通知
  - <http://169.254.169.254/latest/meta-data/spot/instance-action> に格納される
  - 普段は空
  - cronやシェルスクリプトから5秒おきのチェックを推奨
  - [https://docs.aws.amazon.com/ja\\_jp/AWSEC2/latest/UserGuide/spot-interruptions.html](https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/spot-interruptions.html)
- 後処理
  - シェルスクリプトを記述する

# <参考> 中断通知の受信 – CloudWatch Eventsから

- Amazon CloudWatch Events
  - AWSリソースの変更をリアルタイムに把握できるストリーム(イベントの流れ)を提供
  - あるイベントに対してルールを記述し、一致したときにターゲットアクションを自動実行
  - [https://docs.aws.amazon.com/ja\\_jp/AmazonCloudWatch/latest/events/WhatIsCloudWatchEvents.html](https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/events/WhatIsCloudWatchEvents.html)
- 中断通知 – イベントソース
  - サービス名 : EC2
  - EC2 Spot Instance Interruption Warning
- 後処理 – ターゲットアクション
  - Lambda関数
  - どのインスタンスが中断するかをまず把握
  - SNSトピック

ステップ 1: ルールの作成

AWS 環境で発生するイベントに基づいてターゲットを呼び出すためのルールを作成します。

イベントソース **中断通知が来たら**

イベントパターンを構築またはカスタマイズするか、スケジュールを設定してターゲットを呼び出します。

☒ イベントパターン ⓘ ☐ スケジュール ⓘ

サービス別のイベントに一致するイベントパターンの構築

サービス名: EC2

イベントタイプ: EC2 Spot Instance Interruption Warning

▼ イベントパターンのプレビュー クリップボードにコピー 編集

```
{
  "source": [
    "aws.ec2"
  ],
  "detail-type": [
    "EC2 Spot Instance Interruption Warning"
  ]
}
```

ターゲット

イベントがイベントパターンに一致するか、スケジュールがトリガーされたときに呼び出すターゲットを選択します。

Lambda 関数

機能\* **関数を実行する**

バージョン/エイリアスの設定

入力の設定

SNS トピック

トピック\* **トピックに通知する**

入力の設定

ターゲットの追加\*

スポットフリートで  
中断通知のテストができます！



# 中断のテスト - スポットフリートから(1)

EC2 ダッシュボード

イベント

タグ

レポート

制限

インスタンス

インスタンス

起動テンプレート

**スポットリクエスト**

リザーブドインスタンス

専用ホスト

スケジュール済みインスタンス

**スポットインスタンスのリクエスト** アクション 価格設定履歴 Savings Summary

リクエストタイプ: all 状態: all キーワードによる検索

《 < リクエストなし > 》

リクエスト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス
----------	-----------	------------	----	----	-------

現在、このリージョンにスポットリクエストはありません。

EC2 スポットインスタンスを初めて使用する場合は、「[開始方法](#)」ページにアクセスしてください。

スポットインスタンスを起動するには、スポットインスタンスのリクエスト ボタンをクリックします。

**スポットインスタンスのリクエスト**

詳細を表示するには、上記から 1 つのスポットリクエストを選択します。

# 中断のテスト - スポットフリートから(2)

スポットインスタンスのリクエスト   アクション ▾   価格設定履歴   Savings Summary

リクエストタイプ: all ▾   状態: all ▾

ターゲット容量の変更  
スポットリクエストのキャンセル

	リクエスト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス
<input checked="" type="checkbox"/>	▶ sfr-3a87db52-063e...	fleet	t2.medium,m4.l...	● active	3 of 3	fulfilled

3台のスポットインスタンス

- 「ターゲット容量の変更」を選択

# 中断のテスト - スポットフリートから(3)

ト ID	リクエストタ...	インスタンスタ...	状態	容量	ステータス
b52-063e...	fleet	t2.medium,m4.l...	active	3 of 3	fulfilled

### ターゲット容量の変更

リクエスト ID: sfr-3a87db52-063e-40b4-8e47-677899706d2a

従来のターゲットキャパシティ: 3

新しいターゲットキャパシティ:

☒ インスタンスの削除

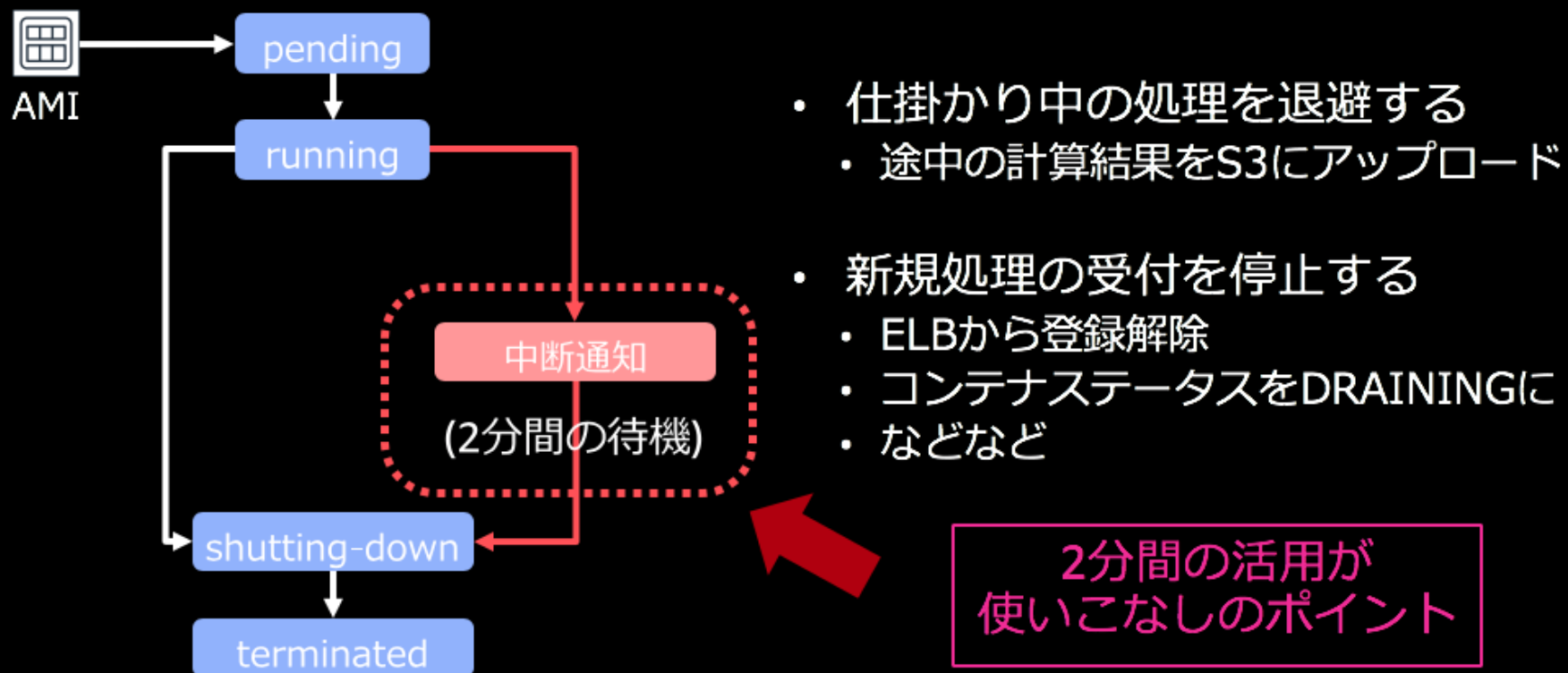
キャンセル

# スポットインスタンス活用の 4原則

# スポットインスタンス活用のポイント

- 中断通知を受け取ったら  
すみやかに後処理を実行

中断通知→すみやかに後処理を！！

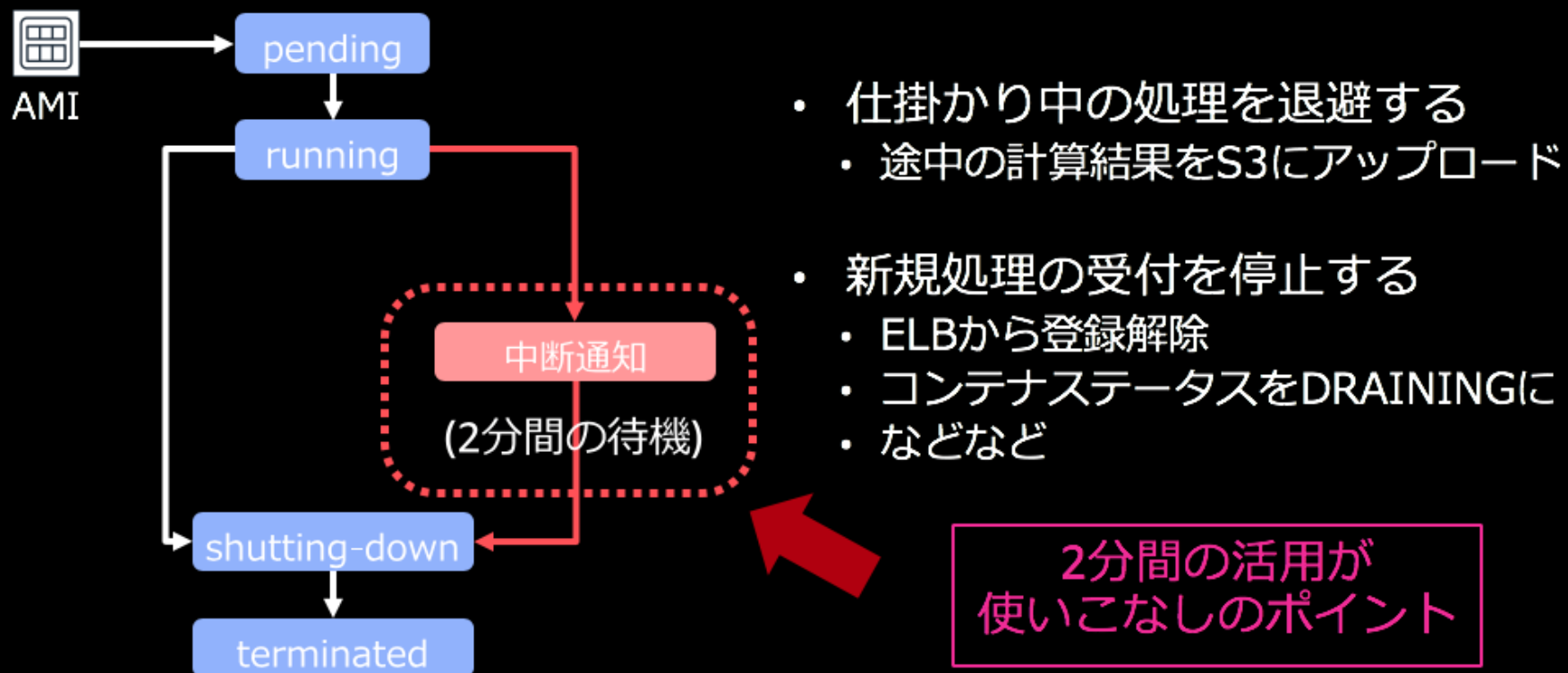


# スポットインスタンス活用のポイント

- 中断通知を受け取ったら  
すみやかに後処理を実行

- 耐障害性のある設計に
  - アプリケーション面
  - システム面

中断通知→すみやかに後処理を！！



# 耐障害性：使いこなすための4原則

アプリ



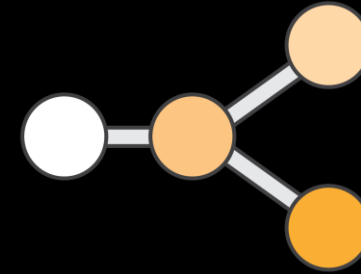
ステートレス：  
状態を持たせない

アプリ



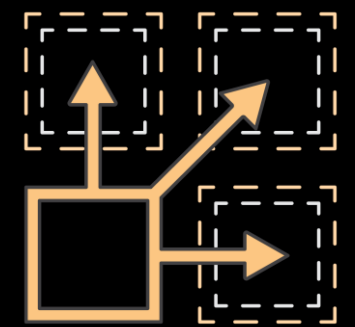
再開可能な  
ワークロード：  
安全な再開

システム



疎結合：  
周辺影響の極小化

システム



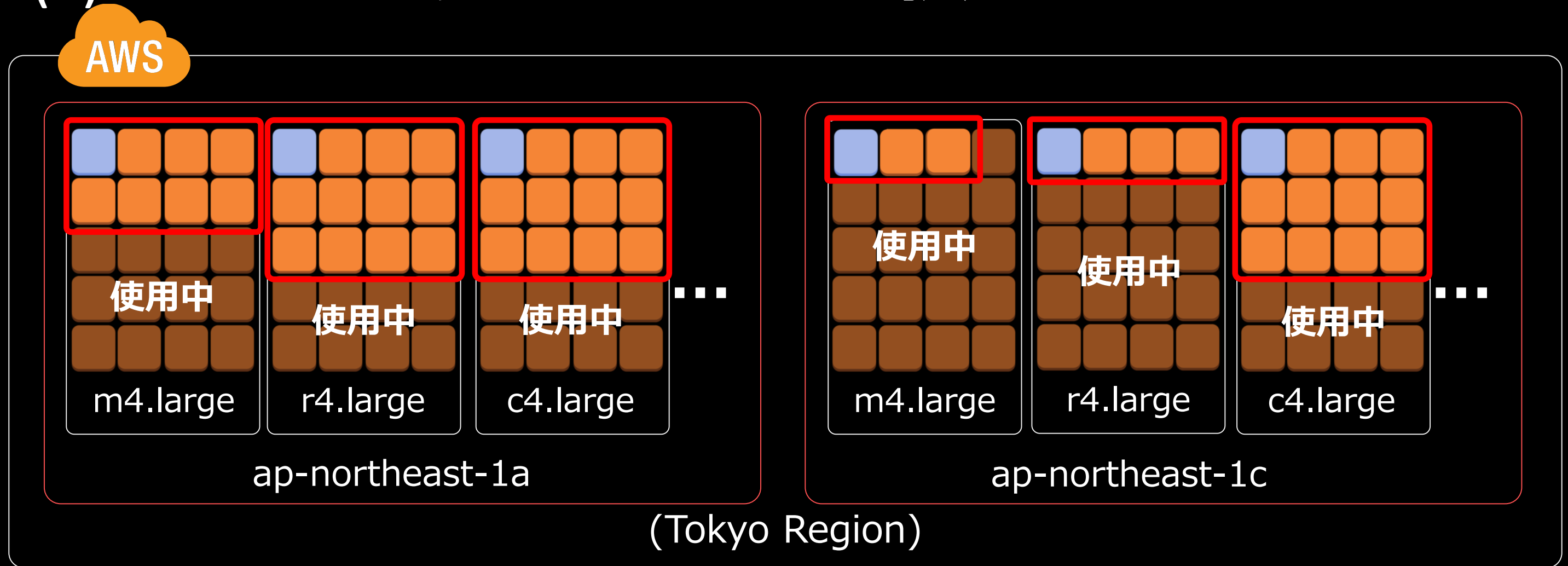
分散：  
複数アベイラビリ  
ティゾーンと複数  
インスタンスタイ  
プの活用

6台のスポットインスタンスを  
起動します。

どちらが中断に強いでしょうか？



# (a) 6つのスポットプールを選択

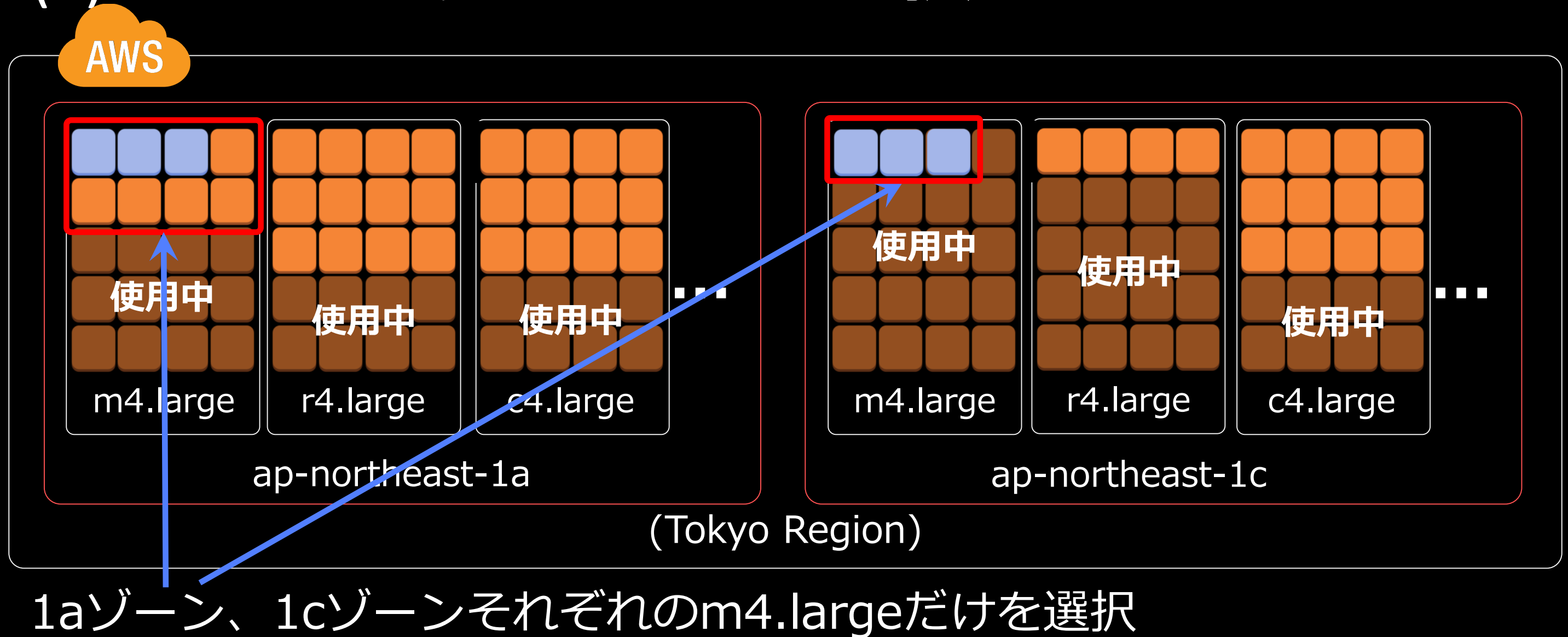


## 6つのスポットプールすべてを選択

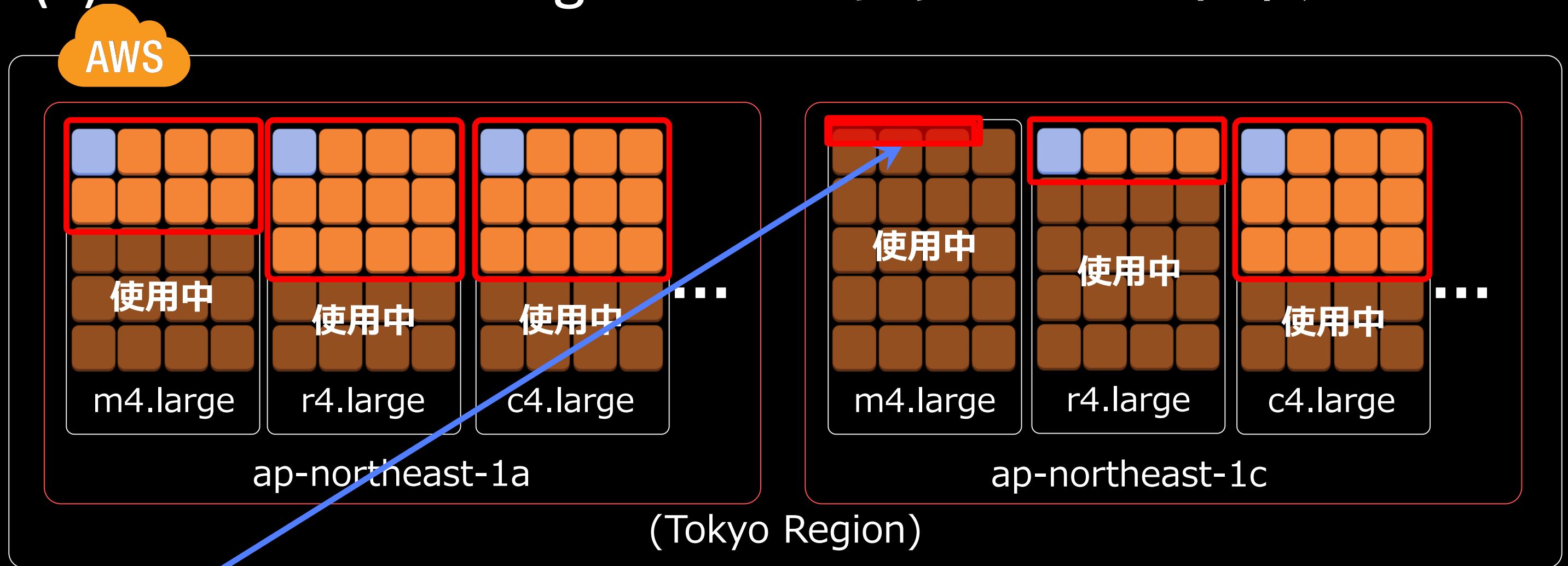
<参考> 選択したスポットプールすべてに均等にスポットインスタンスを分散する起動オプションを指定した  
(「配分戦略」 – diversified)

[https://docs.aws.amazon.com/ja\\_jp/AWSEC2/latest/UserGuide/spot-fleet.html#spot-fleet-allocation-strategy](https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/spot-fleet.html#spot-fleet-allocation-strategy)

## (b) 2つのスポットプールを選択

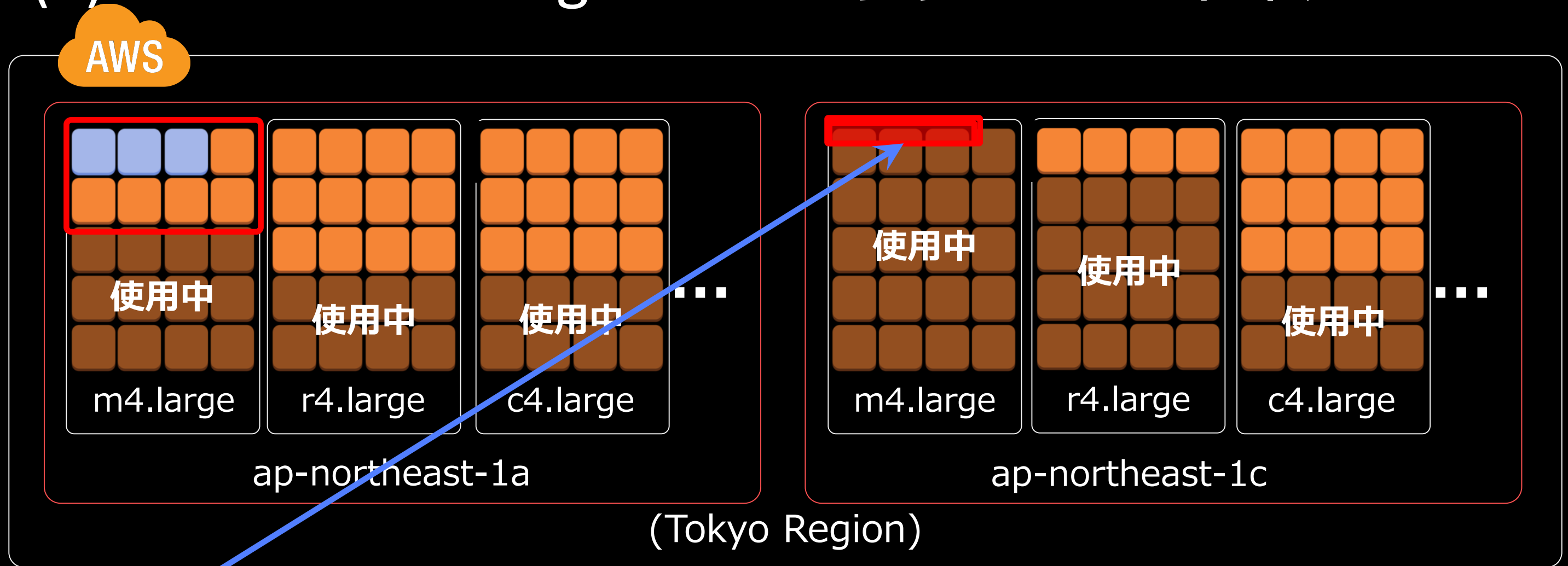


# (a) 1cゾーンm4.largeの空きキャパシティ不足



1台は中断するが、残り5台は業務継続できる

# (b) 1cゾーンm4.largeの空きキャパシティ不足



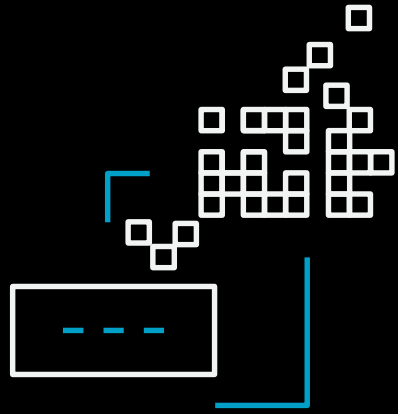
半数の3台が中断リスクを負うことになる

# スポットプールの分散

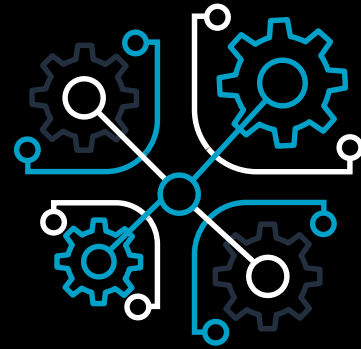
- スポットインスタンスを起動するとき、候補となるインスタンスタイプ・アベイラビリティゾーンを複数選択することで中断に対するリスクを分散できる
  - (a)パターンが理想的
  - (b)パターンはスポットプールの数が少なく、できる限り避けたい
- スポットフリート、EC2 Auto Scalingのミックスインスタンスグループを活用する

# スポットインスタンスの 活用シーン

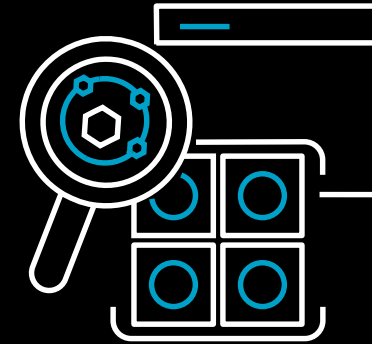
# スポットインスタンスの活用シーン



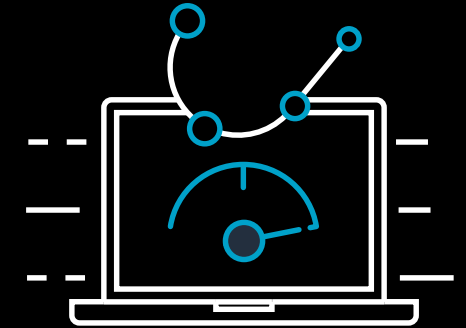
ビッグデータ分析



CI/CD



ステートレスな  
Webアプリ



アドホックな  
計算処理



さらに……  
コンテナ化されたワークロード

使いこなすための4原則

- ✓ ステートレス
- ✓ 再開可能
- ✓ 疎結合
- ✓ スポットプールの分散

# 株式会社トクバイー インフラコストを3割削減

- PC/スマートフォン向けのチラシ・買物情報サービス
  - 自社開発アプリの99%をAmazon ECSで稼働
- ECSクラスターに  
スポットインスタンスを活用
  - AWSの活用を進めた結果、想定  
の1.5～2倍の費用に
  - EC2費用を最適化するためスポッ  
トインスタンスを採用
  - EC2インスタンスの費用を約6.5割削減



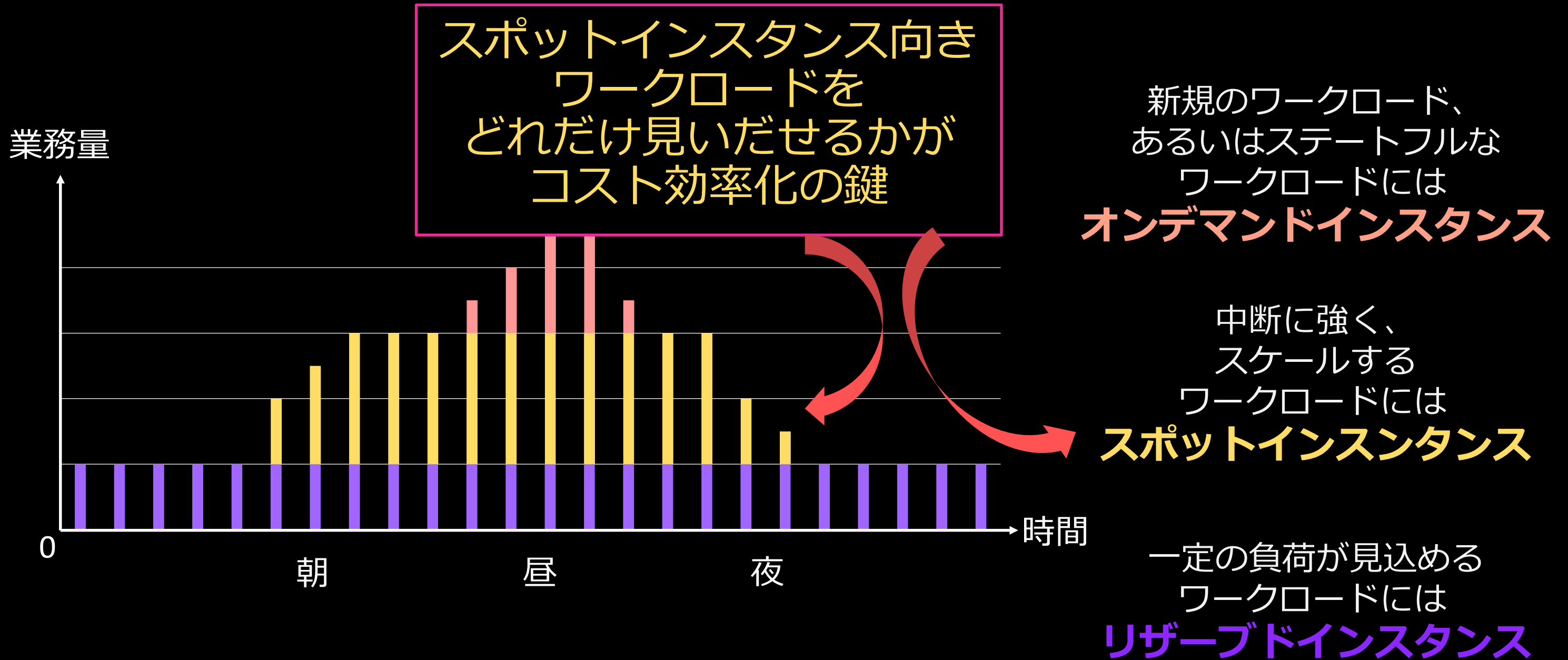


# おわりに

# 本日のまとめ

- スポットインスタンスはこわくない
- スポットインスタンスは安くて簡単
- スポットインスタンスを使いこなす
  - 中断のハンドリング
  - 活用のための4原則
  - さまざまな活用シーン

# EC2購入オプションをどう組み合わせるか？



# 関連セッション、関連情報

(I1-04) B2Cビジネスの本番環境で必要な継続性と高レスポンス性能を支えるコンテナアーキテクチャ

株式会社トクバイ 前田 卓俊 様

---

(H2-03) DeNA の QCT マネジメント IaaS 利用のベストプラクティス

株式会社ディー・エヌ・エー土屋 圭 様

---

EC2 Spot Instances Workshops (英語)

<https://ec2spotworkshops.com/>

# 本セッションのFeedbackをお願いします

お手元のサミットガイドブックの表紙、受講票にも記載している  
『QRコード』からご回答ください。  
もれなく**素敵なAWSオリジナルグッズ&アイス**をプレゼントします。

涼感マフラータオル（巾着入り）



プレゼントの引き換えは、EXPOエリア内アンケートコーナー・出口付近のいずれかにお越しください。