AWSへのマイグレーションストラテジー

アマゾンデータサービスジャパン
エンタープライズソリューション部 ソリューションアーキテクト　布目 拓也
プロフェッショナルサービス コンサルタント　千葉 悠貴

amazon web services

自己紹介

布目 拓也
エンタープライズソリューション
ソリューションアーキテクト

千葉 悠貴
プロフェッショナルサービス
コンサルタント

本セッションの目的

- AWS移行を検討する際のポイントを理解する
- 移行に利用できるツール、移行方式について理解する

移行検討ステップ

amazon web services

移行検討ステップ

- 何を移行するのか
- どのように移行するのか
- 移行後どのように運用するのか

移行検討ステップ

- 何を移行するのか
- どのように移行するのか
- 移行後どのように運用するのか

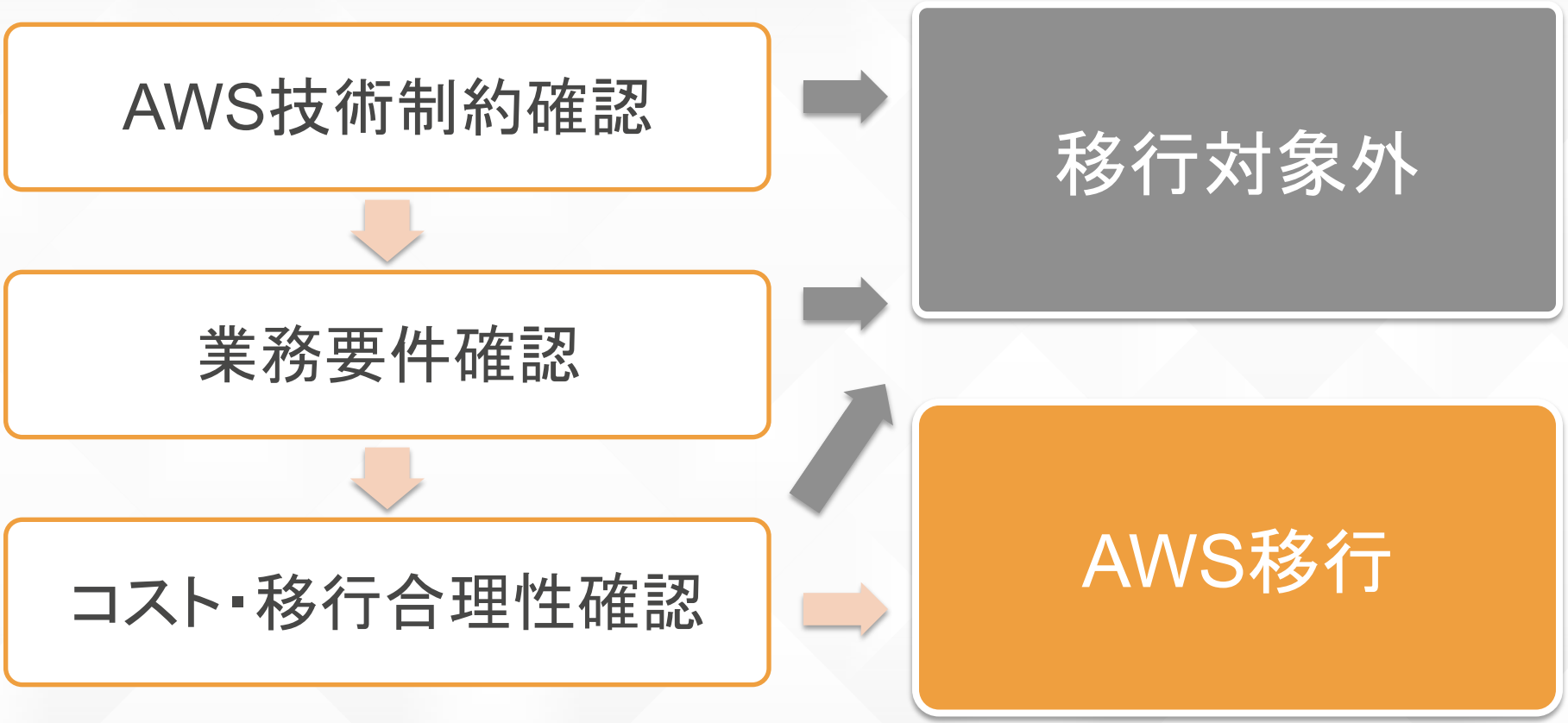
陥りがちなワナ

一部システムがAWSに向かないため移行検討が進まない

- ✓ All or Nothingではない
- ✓ 大多数のエンタープライズはハイブリッド型
- ✓ 移行対象の選定が必要

amazon web services

移行対象選定フロー



AWS技術制約確認

AWSの技術制約に対象システム要件が該当しないか

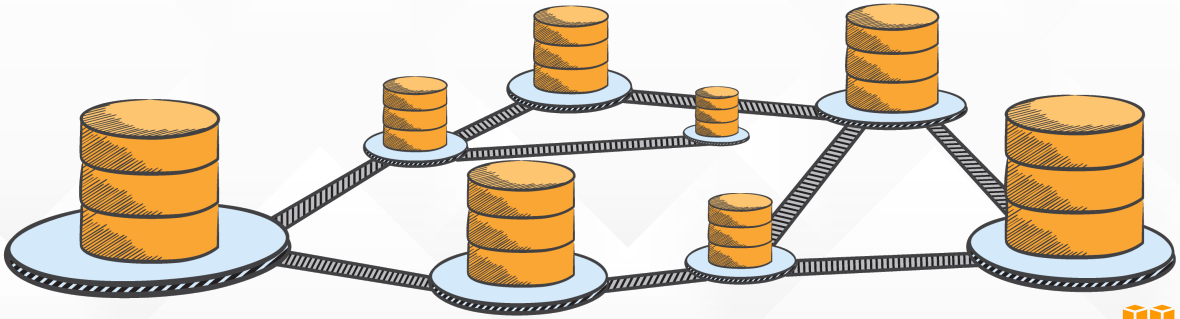
カテゴリ	制約対象
物理機器	USBドングル、プリンター、テープ装置、物理ストレージ装置
共有ディスク	共有ディスク型クラスターソフトウェア
ネットワーク	物理ネットワーク機器持ち込み
未対応OS	RHEL4以前、Windows2000以前、 Windows系デスクトップOS、AIXなど
ミドルウェア・ パッケージ	ベンダーがAWS上の動作保証をしていないミドルウェア・パッケージ
性能	CPU36コア・メモリ244GBを上回る性能が要求されるサーバー

※AWS Summit Tokyo 2015 開催時の情報

業務要件確認

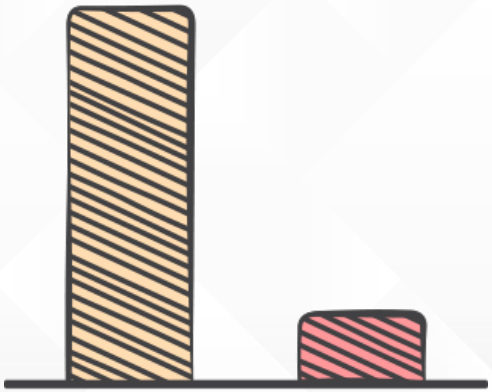
業務要件上オンプレミスで稼働させる必要性がないか

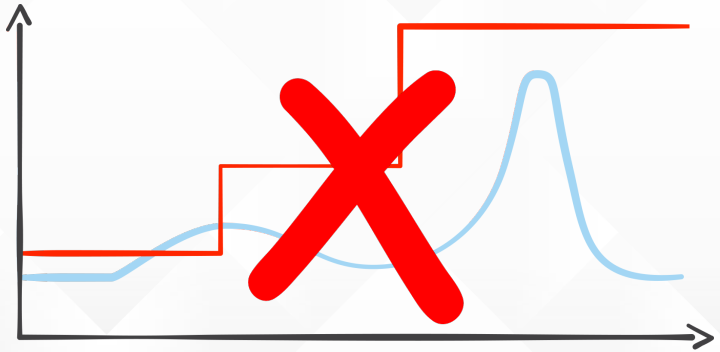
カテゴリ	制約対象
セキュリティポリシー	社内セキュリティポリシー上、扱う情報の社外持ち出しが許可されないシステム
オンプレミスシステムとの結合度	オンプレミスに残るシステムとの結合度が高いシステム（WSUSサーバー、認証サーバー、統合監視サーバー、アンチウィルスサーバーなど）



コスト・移行合理性確認

- ランニングコスト、移行コストを算出し、ROIを試算
- コスト比較だけでなく、AWSを使うメリットを再確認
  - ✓ システムの柔軟性、拡張性を手に入れる
  - ✓ 今までできなかったビジネスを新たに始める





移行検討ステップ

何を移行するのか

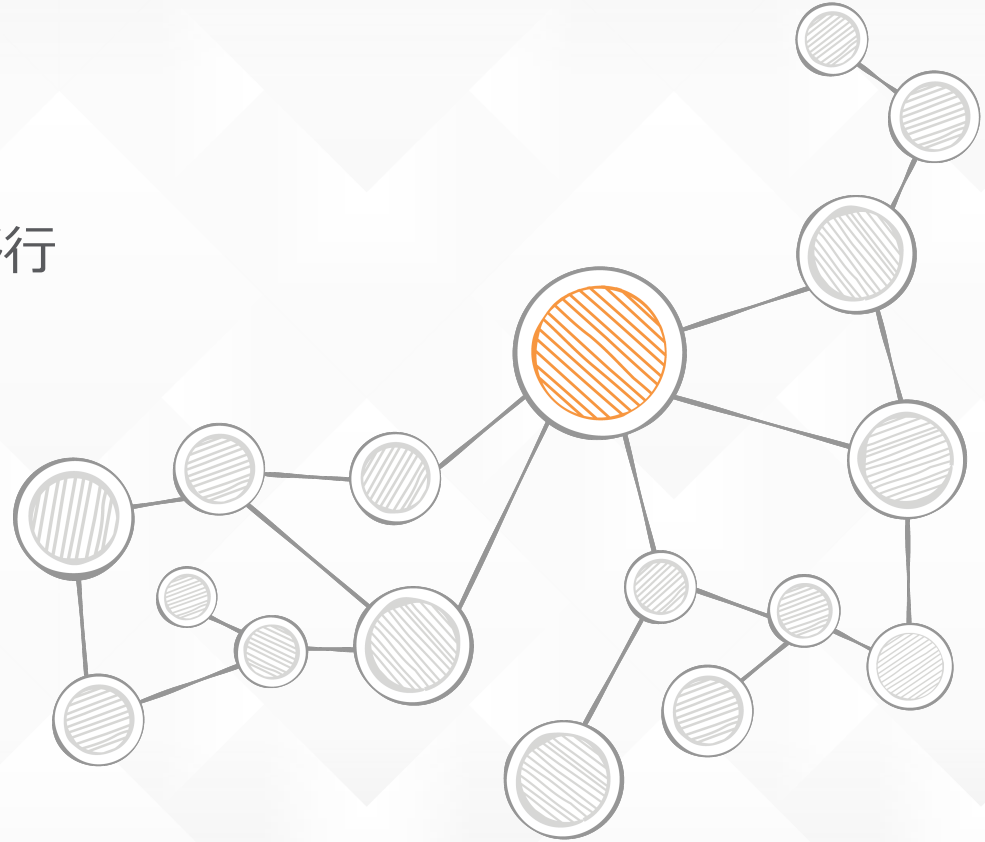
どのように移行するのか

移行後どのように運用するのか

移行設計の検討事項

- 移行後のアーキテクチャ
  - 移行後のサービスレベル
  - 単純移行 or AWS最適化移行

- 移行計画
  - 移行ツール、切替方法
  - 移行時期、停止時間
  - 利用者への周知方法



ケーススタディ - サービスレベルの見直し -

- 課題
  - 目標サービスレベルは現行を踏襲し99.999%
  - 目標を達成し得るアーキテクチャはコストが非常に高い

- 検討事項
  - RTO(目標復旧時間)、RPO(目標復旧地点)は？
  - 現行環境の構成は？
  - 現行環境は目標サービスレベルを満たせているか？
  - 99.999%のサービスレベルが本当に必要か？

クラウド移行時のアーキテクチャパターン

Gartner Identifies Five Ways to Migrate Applications to the Cloud (http://www.gartner.com/newsroom/id/1684114)

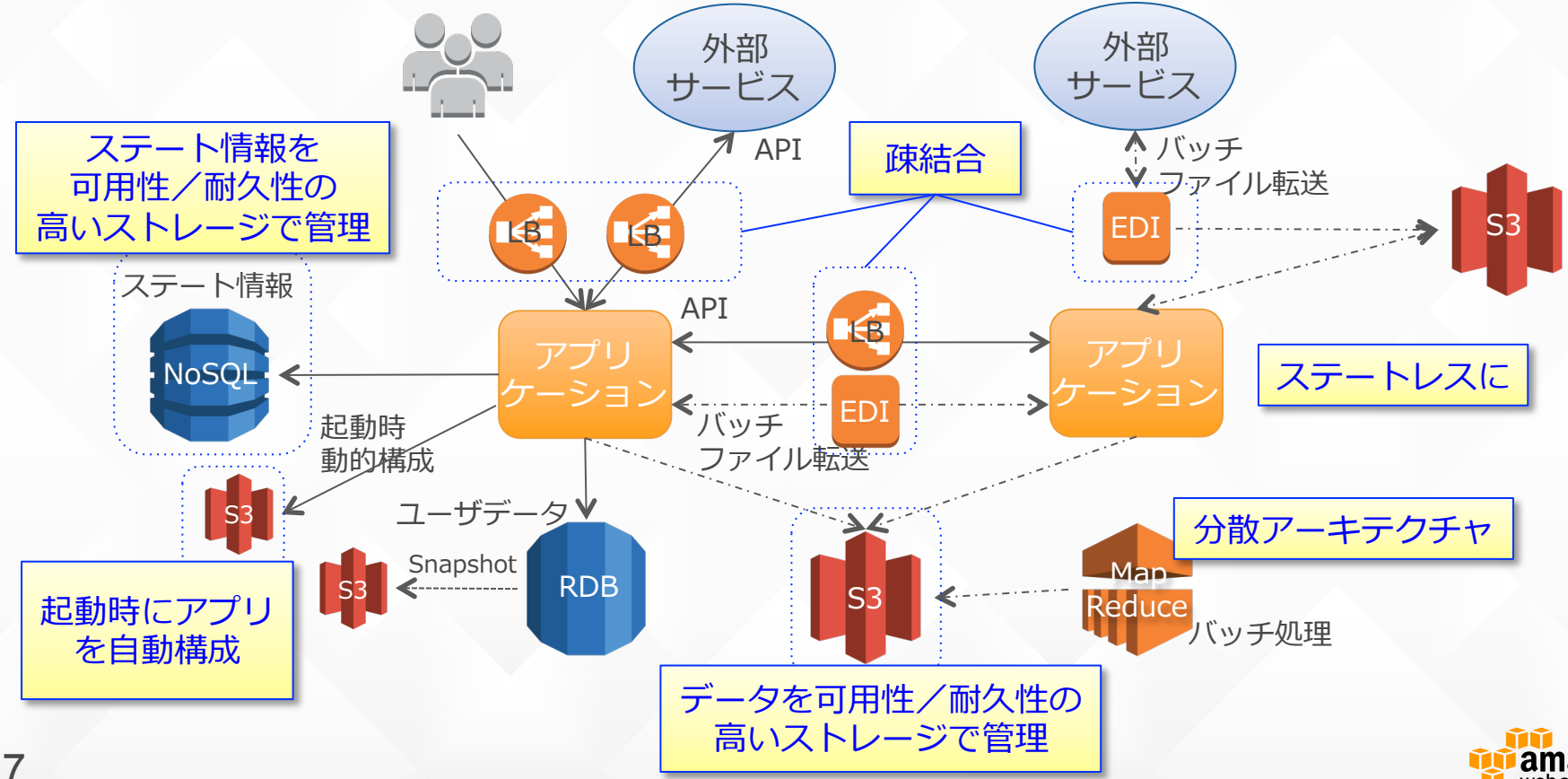
- **Rehost**
  - アプリに改修を加えずそのまま移行
- **Refactor**
  - アプリ仕様は維持し、一部マネージドサービスを利用
- **Revise**
  - 既存のアプリ仕様をベースに機能追加／改修
- **Rebuild**
  - アプリを書き換え、マネージドサービスを活用
- **Replace**
  - 既存のアプリをマネージドアプリサービスに置き換え

AWSサービスへの移行例

方式	概要	移行例
Rehost	アプリに改修を加えずそのまま移行	既存の仮想マシンをVMImportを利用してEC2に移行
Refactor	アプリ仕様は維持し、一部マネージドサービスを利用	Web/APサーバーはEC2に移行し、RDB部分にRDSを利用
Revise	既存のアプリ仕様をベースに機能追加／改修	Webアプリをステートレス化し、AutoScalingを利用
Rebuild	アプリを書き換え、マネージドサービスを活用	DynamoDB, Lambda, Cognitoなどを活用し、モバイルアプリを2Tierアーキテクチャー化
Replace	既存のアプリをマネージドアプリサービスに置き換え	既存のVDIシステムをWorkSpaces, WorksDocsに置き換え

移行スケジュール・移行難易度を考慮

マネージドサービスを利用したクラウド最適化例



マネージドサービス移行時の注意点

全ての機能をマネージドサービスで実現できるとは限らない

- RDS
  - データサイズがRDSの最大容量を超える
  - DBパラメーターを細かくチューニングしたい
- ELB
  - 物理負荷分散装置で利用していた機能を使いたい

✓ 一部機能はアプリケーション側に実装する

✓ 現行機能のソフトウェアをEC2上に導入する

移行検討ステップ

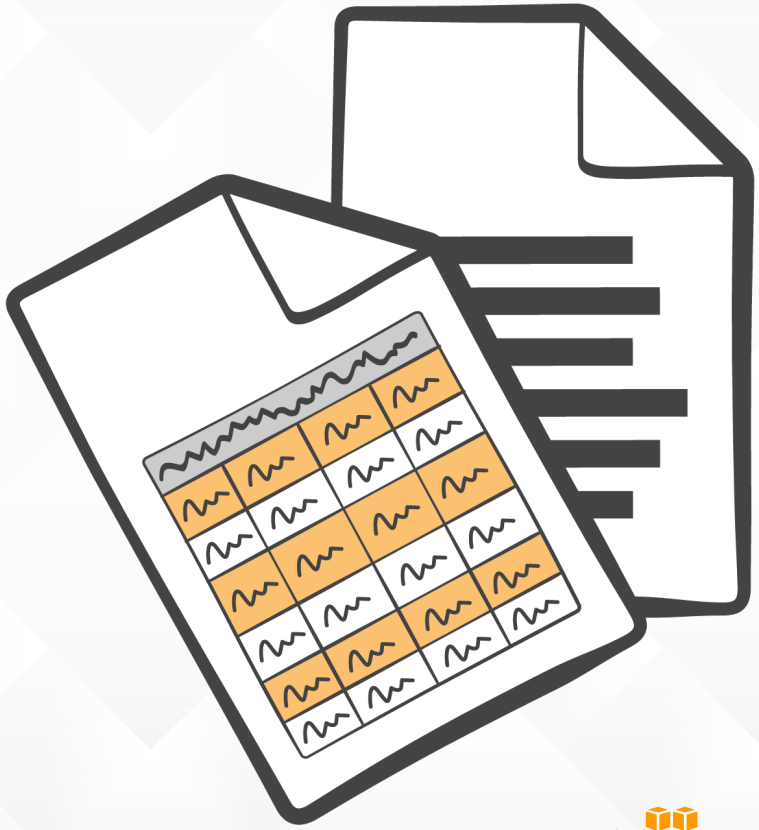
何を移行するのか

どのように移行するのか

移行後どのように運用するのか

運用設計時の検討事項

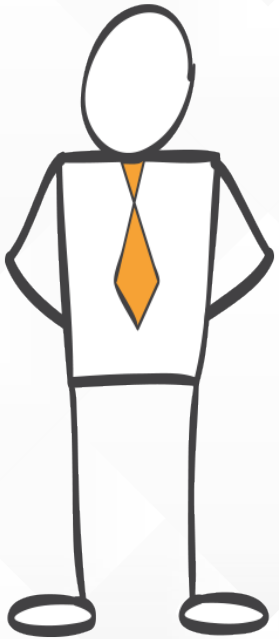
- 運用変更項目洗い出し
  - 既存運用で必要なくなる作業
  - 追加で必要になる作業
  - 手順が変更になる作業
- 役割分担整理
  - どのチームが何の作業を担当するか
  - AWSの権限管理をどのようにするか
- ガバナンス方針
  - コスト管理方法
  - セキュリティチェック方法



ケーススタディ - 社内標準ガイドライン作成 -

- 課題
  - 複数システムを移行したい
  - インフラ設計、運用設計を効率化したい
  - ガラパゴスなシステムができないよう、ガバナンスを効かせたい

- 解決策
  - サービスレベル／インフラ／運用の標準を定義
  - 標準をガイドラインドキュメントを作成
  - 各システム担当者が標準に沿ったシステム構成を設計

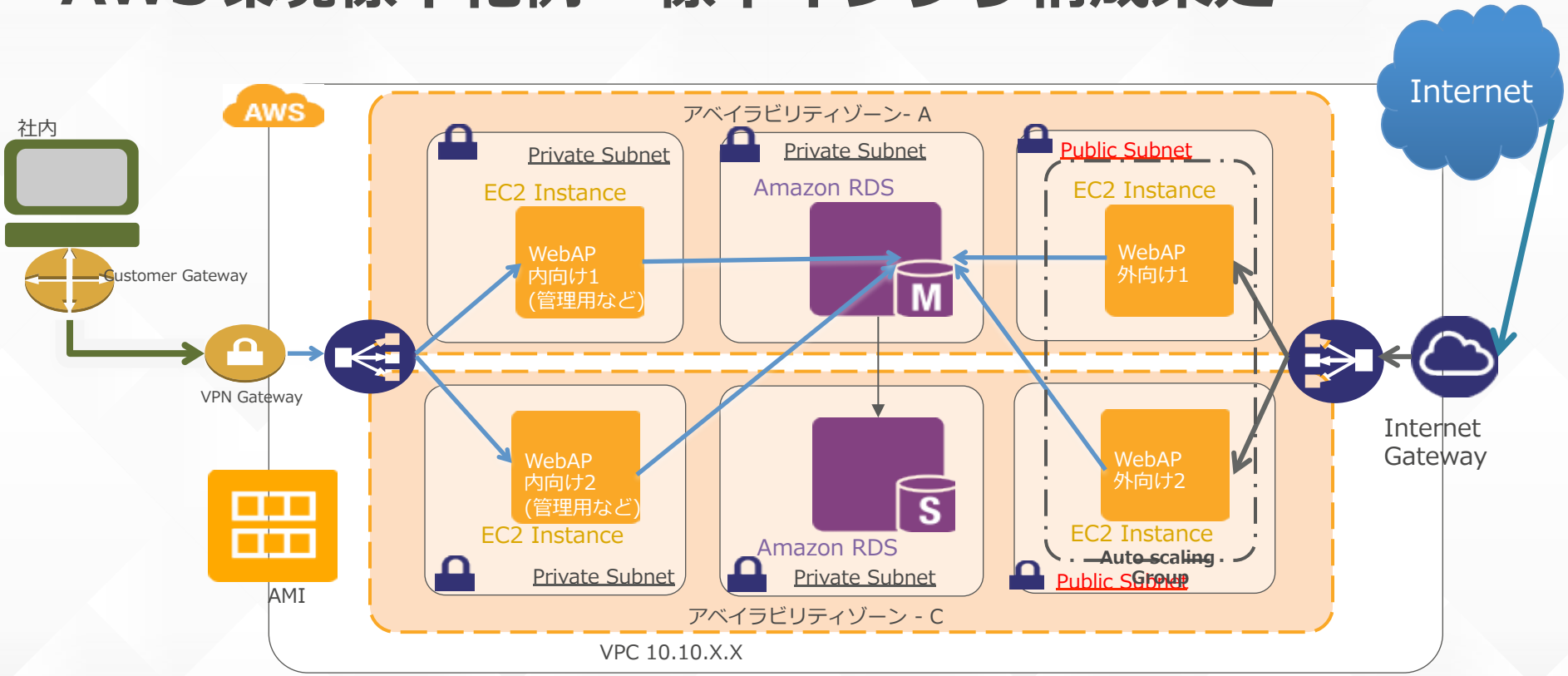


AWS環境標準化例 - 標準サービスレベル策定 -

項目		本番：サービスレベル高	本番：サービスレベル低	検証開発
サービス時間	サービス提供時間	24時間365日	月-土 9:00-22:00	月-金9:00-22:00
	計画停止	1週間前通知で夜間6時間	数日前通知で数時間	随時
可用性	障害時停止許容時間	1-2時間	24時間	24時間
	障害時リカバリポイント	直前	1日前	1日前
	障害時性能	100%	50%以上	なし
バックアップ	リストア時間	6時間	1営業日	数営業日
	リストア時リカバリポイント	直前（5分前）	1営業日	1週間
	バックアップ保管世代	7世代	3世代	1世代
災害対策	災害対策RTO	1日以内	なし	なし
	災害対策RPO	1日前	なし	なし
拡張性	システム追加	n/a	n/a	1時間以内
	リソース増強	1営業日以内	1営業日以内	数分

求められる要件に応じた標準サービスレベルを定義

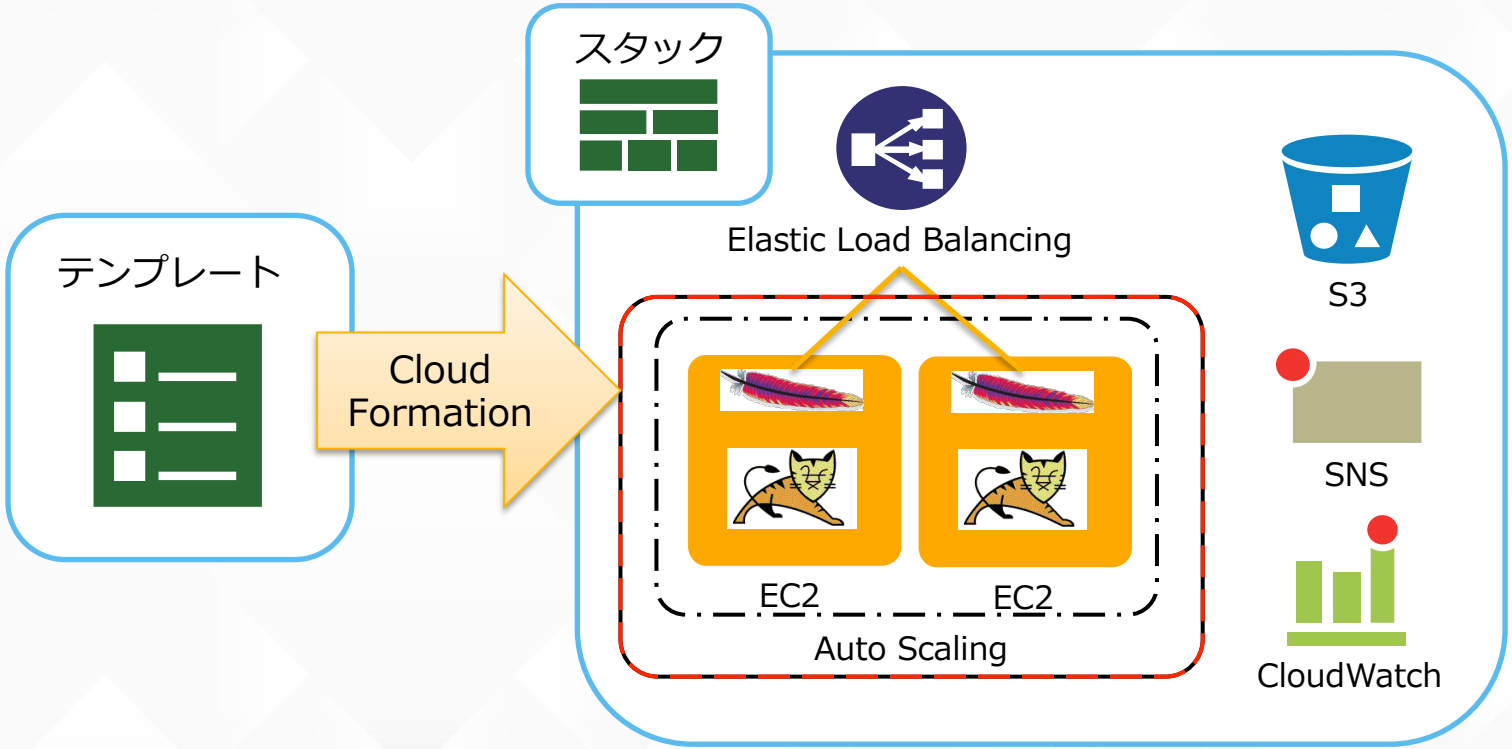
AWS環境標準化例 - 標準インフラ構成策定 -



標準サービスレベルごとにインフラ構成を定義

amazon web services

AWS環境標準化例 - 環境構築自動化 -



標準インフラ構成ごとにCloudFormationテンプレートを作成

amazon web services

移行検討ステップのまとめ

- AWS技術制約・業務要件・移行合理性を確認する
- 単純移行・AWS最適化など移行パターンを整理する
- 運用変更点を整理し、最適化をおこなう

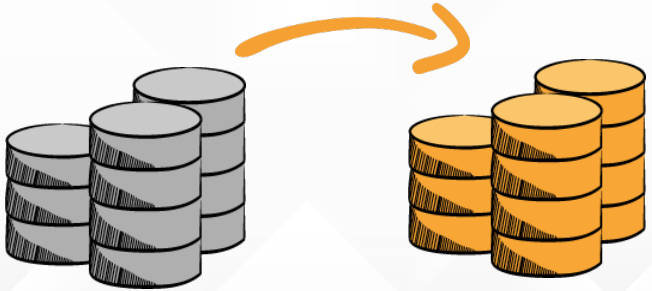
移行方法・ツール

amazon web services

AWSへの移行方式

- サーバの移行
  - サーバ再構築＋既存アプリ導入
  - 仮想マシンのインポート

- データの移行
  - ファイル単位での移行
  - ブロックデバイス単位での移行
  - ミドルウェアの機能を利用した移行

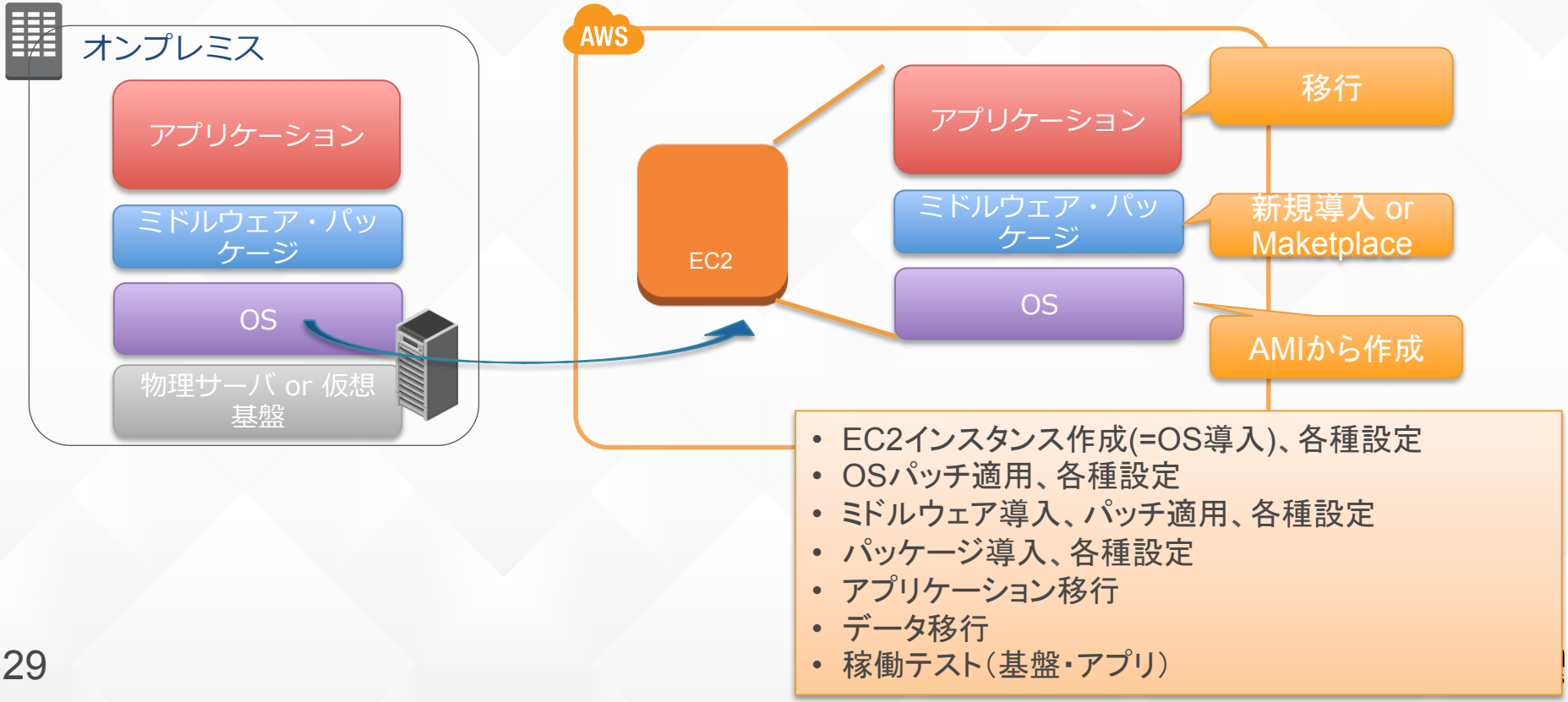


サーバの移行

amazon web services

サーバ再構築+アプリ導入

OSを新規構築する場合の作業イメージ



仮想マシンのインポート

仮想環境であればVM ImportでOSイメージを丸ごと移行可能

事前作業

- クローン作成
- 事前作業
- イメージ抽出

Import後作業

- 各種設定(ネットワーク設定等)
- データ移行
- 稼働テスト(基盤・アプリ)

VM ImportによるOSの変更点

- OS起動、ネットワーク接続等、AWS上で稼働するための最低限の変更が加えられる
- CentOS6.6をVM Importで移行した例
  - ブートローダー関連
    - /boot/grub/grub.conf　⇒ **ttyS0追加**
    - /etc/securtty ⇒ **ttyS0追加**
  - ネットワーク関連
    - /etc/resolv.conf ⇒**DHCPにより内部DNS参照へ変更。**
    - /etc/sysconfig/network-scripts/ifcfg-eth0
      ⇒**DHCP用として新たに作成**
  - ファイルシステム関連
    - /etc/fstab ⇒**UUID指定に変更**
  - その他起動スクリプト類
    - /etc/rc.d/rc.local
    - /etc/rc.local　⇒**具体的な変更なし**

VM Importのこれまでの課題

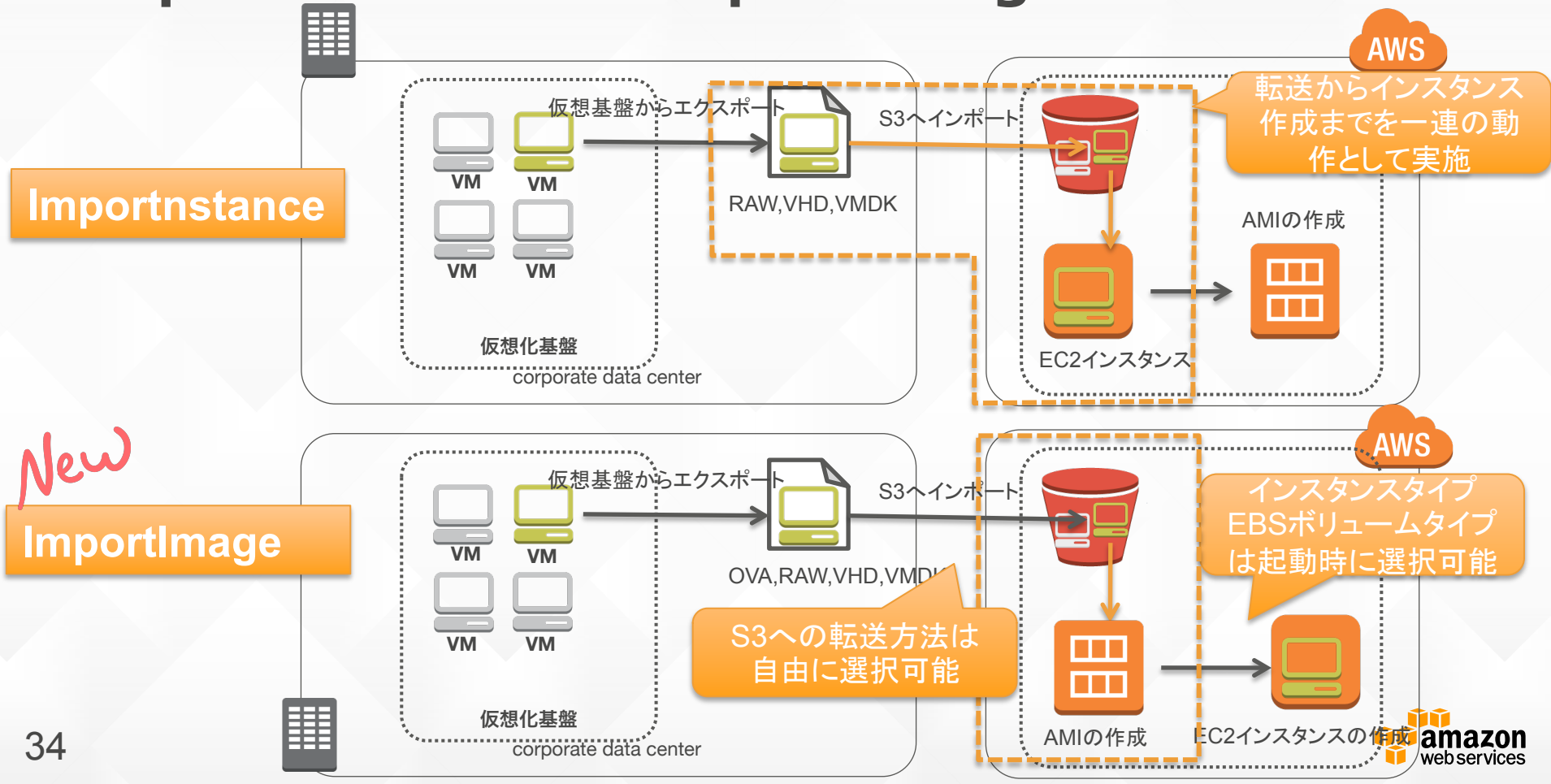
- 転送に時間が掛かる
- 失敗した場合ファイル転送からやり直し
- 複数ボリュームの仮想マシンが移行できない
- 同時にImportできる数が少ない
- EC2 CLIのセットアップが必要
- オプション指定が複雑
- EBSのタイプが選択できない

新しいVM Import API New

- ImportImage/ImportSnapshot
  - イメージ転送とImport処理を分離
  - S3にあるイメージを起点としてAMI/EBSスナップショット作成を実行
  - OVA形式、複数Volumeで構成されるVMにも対応

	従来のAPI(ImportInstance)	新しいAPI(ImportImage)
データソース	オンプレミスからのアップロード	S3上のイメージまたはEBSスナップショット
ターゲット	EC2インスタンス(停止)	AMI
VM構成	単一Volumeのみ	複数Volumeも可
同時Import数	5	20
CLI	EC2 CLI	AWS CLI
フォーマット	VMDK,VHD,RAW	VMDK,VHD,RAW,**OVA**

ImportInstanceとImportImage



amazon web services

OVA形式でのImport例

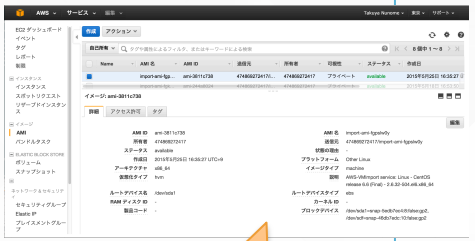
```
$ aws ec2 import-image --cli-input-json
"{  \"Description\": \"CentOS OVA\",
 \"DiskContainers\": [ { \"Description\": \"CentOS6.6
OVA file\", \"UserBucket\": { \"S3Bucket\":
\"nunomet-data\", \"S3Key\" : \"nunomet-centos.ova
\" } }  ]}"
{
    "Status": "ac
    "Description"
    "Progress": "
    "SnapshotDeta
        {
            "UserBucket": {
                "S3Bucket": "nunomet-data",
                "S3Key": "nunomet-centos.ova"
            },
            "DiskImageSize": 0.0
    ],
    "StatusMessage": "pending",
    "ImportTaskId": "import-ami-fgpslw0y"
```

OVAファイルのS3バケットとキーを指定

OVA内の複数Volumeを一括Import

StatusMessageが“pending”→“converting”→”updateing”→”booting”→”preparing ami”と変化し、最終的にStatusがcompleteに

```
$ aws ec2 describe-import-image-tasks
{
    "ImportImageTasks": [
        {
            "Status": "active",
            "LicenseType": "BYOL",
            "Description": "CentOS OVA",
            "Platform": "Linux",
            "Architecture": "x86_64",
            "Progress": "59",
            "SnapshotDetails": [
                {
                    "UserBucket": {
                        "S3Bucket": "nunomet-data",
                        "S3Key": "nunomet-centos.ova"
                    },
                    "DiskImageSize": 818199552.0,
                    "DeviceName": "/dev/sda1",
                    "Format": "VMDK"
                },
                {
                    "UserBucket": {
                        "S3Bucket": "nunomet-data",
                        "S3Key": "nunomet-centos.ova"
                    },
                    "DiskImageSize": 818699264.0,
                    "DeviceName": "/dev/sdf",
                    "Format": "VMDK"
                }
            ],
            "StatusMessage": "booting",
            "ImportTaskId": "import-ami-fgpslw0y"
```



完了後AMIとして登録される

サーバ移行のポイント

	サーバー再構築＋既存アプリ導入	仮想マシンインポート
採用ケース	• 物理マシンからの移行の場合 • 移行対象OSがEC2のサポート対象外の場合	• 仮想マシンからの移行の場合 • VMImportの制約に合致しない場合
留意事項	• OSが変更になる場合、アプリ・ミドルのバージョンアップ／改修が必要な可能性あり	• OSやサーバー構成に制限あり • インスタンスタイプに制限あり

- 物理サーバ移行、バージョンアップも同時に行う移行の場合は再構築
  - 再構築でも移行負荷はオンプレミス同士と変わらない
- 仮想サーバの移行ではVM Importを積極的に活用することによって移行負荷の軽減が可能
  - 移行後にミドルウェアバージョンアップなど段階的な対応も可能

データの移行

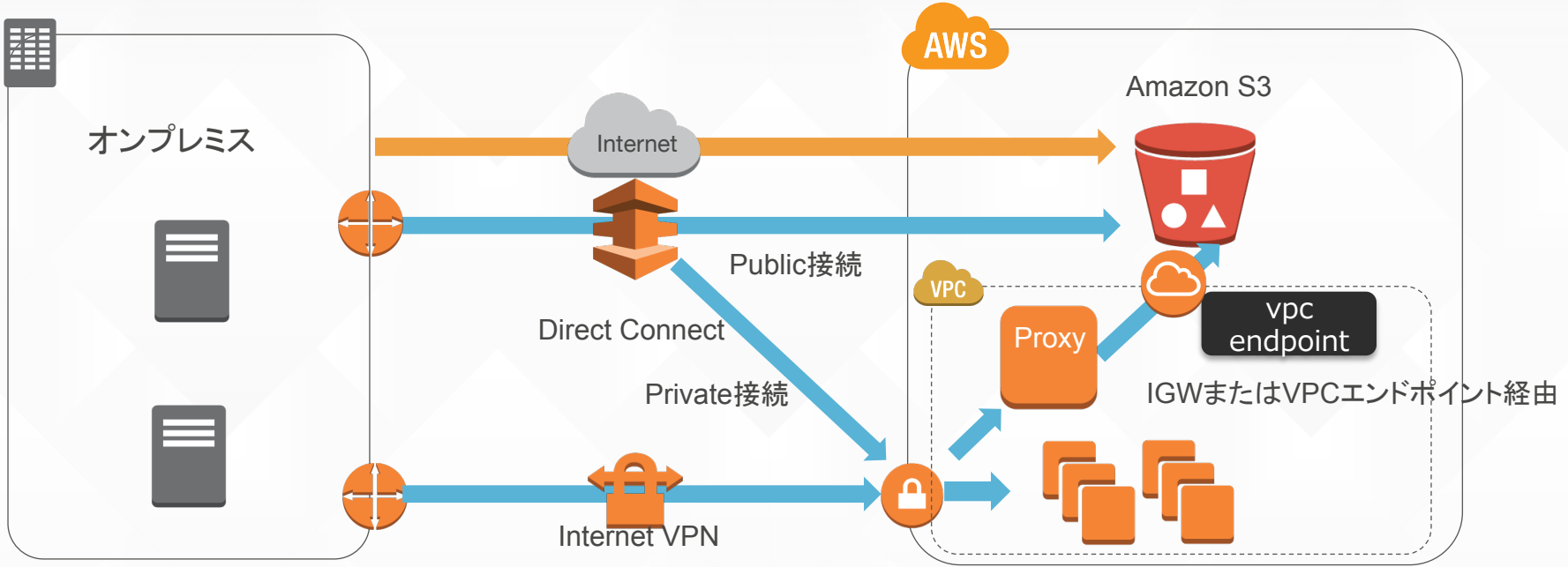
amazon web services

AWSへのデータ移行

ネットワーク経由での移行が基本

- ファイルでS3 or EC2インスタンスへ移行
  - S3を経由できれば利用範囲が広がる
- ブロックデバイスをそのまま移行
  - 移行対象のVolumeサイズに注意
- ミドルウェアレベルの機能を使って移行
  - バックアップ・リストア型 or レプリケーション型
  - 例：OracleのDataPump、GlolenGate

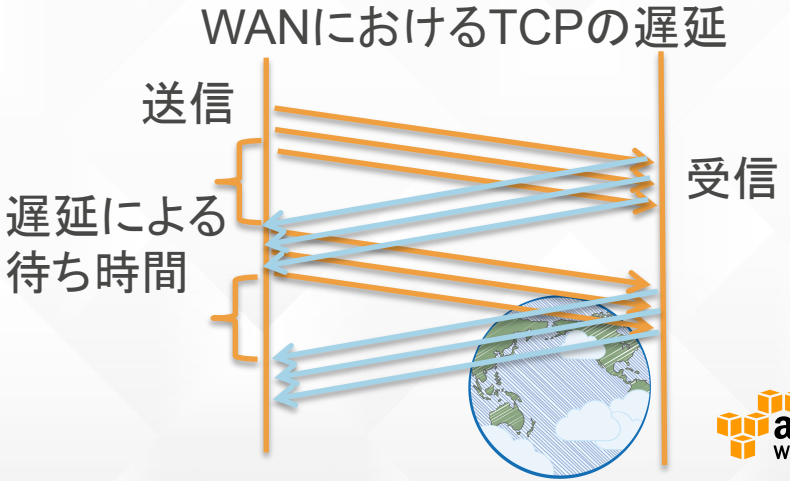
オンプレミスからAWSへの接続



	Internet	Internet VPN	DX(Public)	DX(Private)
ネットワーク帯域	△ オンプレミス依存	△ オンプレミス依存	○	○
S3へのアクセス	直接	要Proxy	直接	要Proxy

ネットワーク経由でのデータ移行における課題

- 移行時のダウンタイムを如何に短くするか？
  - データ同期までの時間がシステムのダウンタイムに直結
  - ダウンタイムに合わせたデータ移行方式を選択
  - 帯域を有効に使い切れる方式を取る
    - 距離の離れたリージョンを利用する場合はTCPの遅延により帯域を十分に使えない場合がある



amazon web services

ファイルの移行方式

S3へ

移行方式

- API、3rdパーティツール等を利用してデータをコピー

高速化

- 専用線接続を利用する
- 数十MBを超えるファイルはマルチパートアップロードする
- 並列で転送を行う
- 無駄なオペレーションは使用しない

EC2インスタンスへ

移行方式

- 任意のプロトコル、任意のアプリケーションを利用可能

高速化

- ソリューションの活用
  - Tsunami-UDP
  - Aspera
  - Skeed

ブロックデバイスの移行

S3を経由

- VM Importサービスの ImportSnapshot APIを利用
- AWS Storage Gatewayを利用

EC2インスタンスへ直接

データレプリケーションソフトウェアを活用

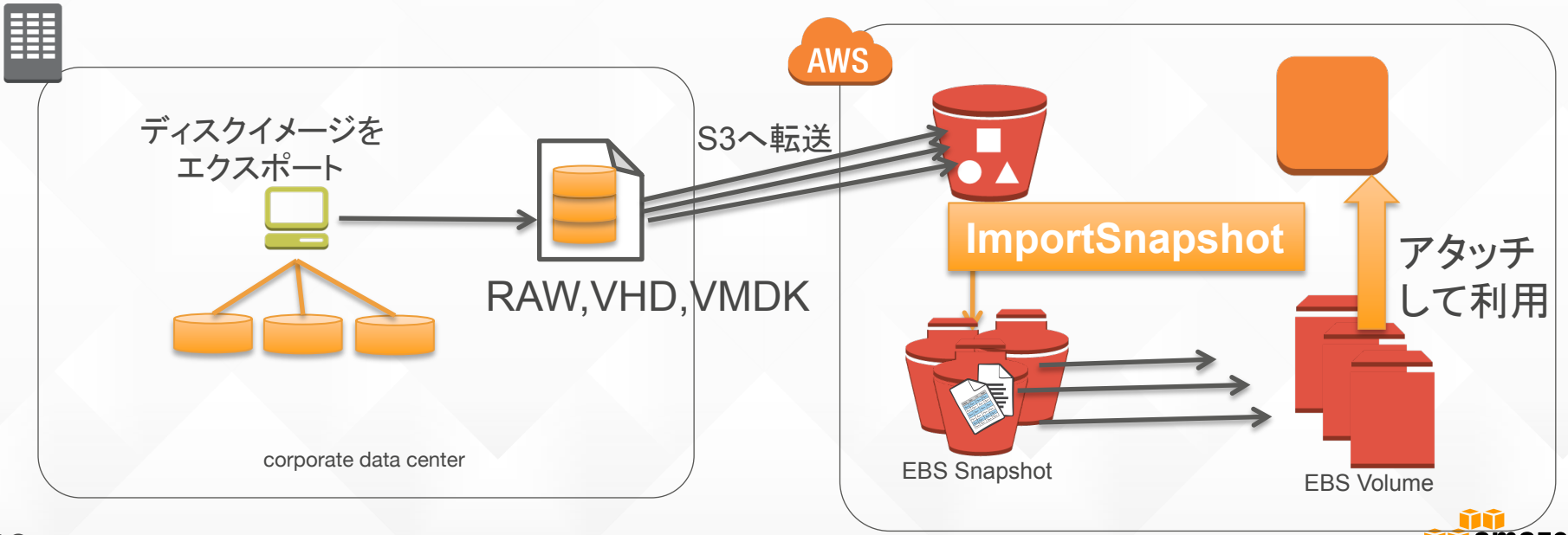
DRBD
DRBD Proxy

DataKeeper

amazon web services

Import-Snapshotによるボリューム移行

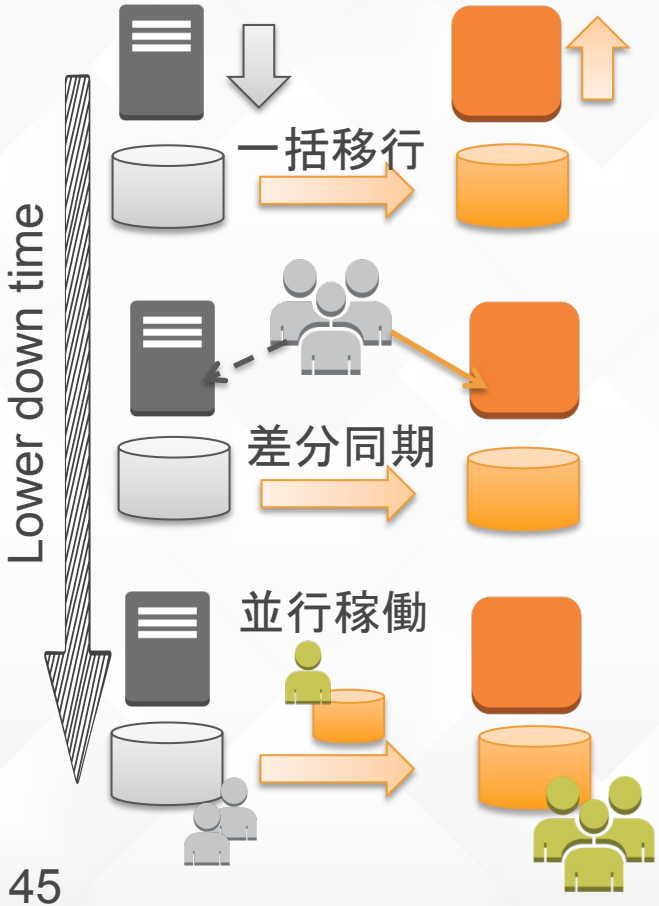
- データボリュームを丸ごと移行
- 移行元でRAWとして利用しているボリュームも移行可能
- 3rd パーティーのS3転送ツールによる差分コピー、WAN高速化ツールの併用も可能



ミドルウェアの機能を利用したデータ移行

- バックアップ・リストア
  - 現行環境には手を入れずに移行可能
  - 一般的には移行時間が長くなるため、十分なダウンタイムが取れる場合に利用
- 各ミドルウェアのレプリケーション機能を利用
  - 一旦現行の一部(スレーブ)として稼働し、移行時に切り離し、マスターへ昇格
  - Active Directoryなど一部のM/W
- レプリケーション専用ソフトウェアを利用
  - ライセンスが必要になるケースがある
  - 一時利用ライセンスが適用可能かミドルウェアベンダーに要確認
  - Oracle GoldenGateなど

データ移行時間を考慮したシステム切り替え方式の選択



- 一括移行
  - 既存環境に手をいれる必要が少ない
  - 十分なダウンタイムが必要
- 一括移行＋差分同期
  - バックアップを使ってある断面でのシステム移行を行い、切替時に差分を同期
  - ダウンタイムを比較的小さくできる
- 並行稼動による段階的移行
  - データを同期しながら利用ユーザーを徐々に移す
  - ファイルサーバ等では有効だが、アプリケーションによっては困難

ダウンタイムを考慮して決定した切り替え方式に合わせて適切なOS移行、データ移行の方法を選択する

データ移行のポイント

- 十分なネットワーク帯域を用意する
- 帯域を有効に使える移行方法を選択する
  - WAN高速化ソリューションの活用
  - 並列化、ブロック単位での転送
  - 3rdパーティツールを利用する場合は、現行環境への影響範囲、一時的な利用が可能かどうかも考慮
- ミドルウェアレベルでどの機能が使えるか調査する
- ダウンタイムを考慮して移行方式を選択する

まとめ

本セッションのまとめ

- AWS技術制約・業務要件・移行合理性を確認する
- 単純移行・AWS最適化など移行パターンを整理する
- 運用変更点を整理し、最適化をおこなう
- 作業負荷・ダウンタイムを考慮したツールの選択を行う

Thank You

amazon web services