[AWS Hands-on for Beginners]
AWS環境のコード管理
AWS CloudFormationでWebシステムを構築する

アマゾン ウェブ サービス ジャパン株式会社
Solutions Architect
木村 友則/ Tomonori Kimura
（収録日: 2020/06/25)

自己紹介

- 名前
  - 木村 友則 (Tomonori Kimura)
- ロール
  - テクニカル ソリューション アーキテクト
- 経歴
  - インターネットメディアのインフラエンジニア
  - サーバー・ストレージベンダーのプリセール・開発等
- 好きなAWSサービス
  - AWS CLI、AWS SDK

AWS Hands-on for Beginners とは

実際に手を動かしながら AWS の各サービスを学んでいただきます

初めてそのサービスをご利用される方がメインターゲットです

お好きな時間、お好きな場所でご受講いただけるオンデマンド形式です

テーマごとに合計1~2時間の内容 & 細かい動画に分けて公開
スキマ時間の学習や、興味のある部分だけの聴講も可能

内容についての注意点

- 資料では2020年6月25日収録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。最新の情報はAWS公式ウェブサイト(http://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、資料とAWS公式ウェブサイトとで記載内容に相違があった場合、AWS公式ウェブサイトの記載を優先させていただきます。
- マネージメントコンソールについても、収録時点のものとなります。
  差異がある場合がございますので、ご注意ください。

- ハンズオンでは AWS の各種サービスの利用、リソースの作成を行います。
  無料枠を超えるハンズオンもございますが、その場合はご利用料金が発生することをあらかじめご認識ください。
- 学習後のリソースの削除についても、お客様の責任でご実施いただくようお願いいたします。

本シリーズのゴール

- クラウドにおける構成管理の考え方を理解する

- AWS CloudFormationを利用して、WebシステムのAWSインフラを構築する手法を理解していただく

本シリーズの前提条件・知識

- AWS アカウントをお持ちであること
  - ハンズオンの作業が同一AWSアカウントの他のリソースに影響が出る場合があります
  - ハンズオン用にAWSアカウントを取得していただくことをオススメします
  - AdministratorAccess ポリシーのついた IAM ユーザーで作業を進めています

- 事前に AWS Hands-on for Beginners ～ スケーラブルウェブサイト構築編 ～ (※1)をご受講いただいていると、より理解を深めやすいと思います

※1 AWS Hands-on for Beginners ～ スケーラブルウェブサイト構築編 ～
https://pages.awscloud.com/event_JAPAN_Hands-on-for-Beginners-Scalable_LP.html

本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

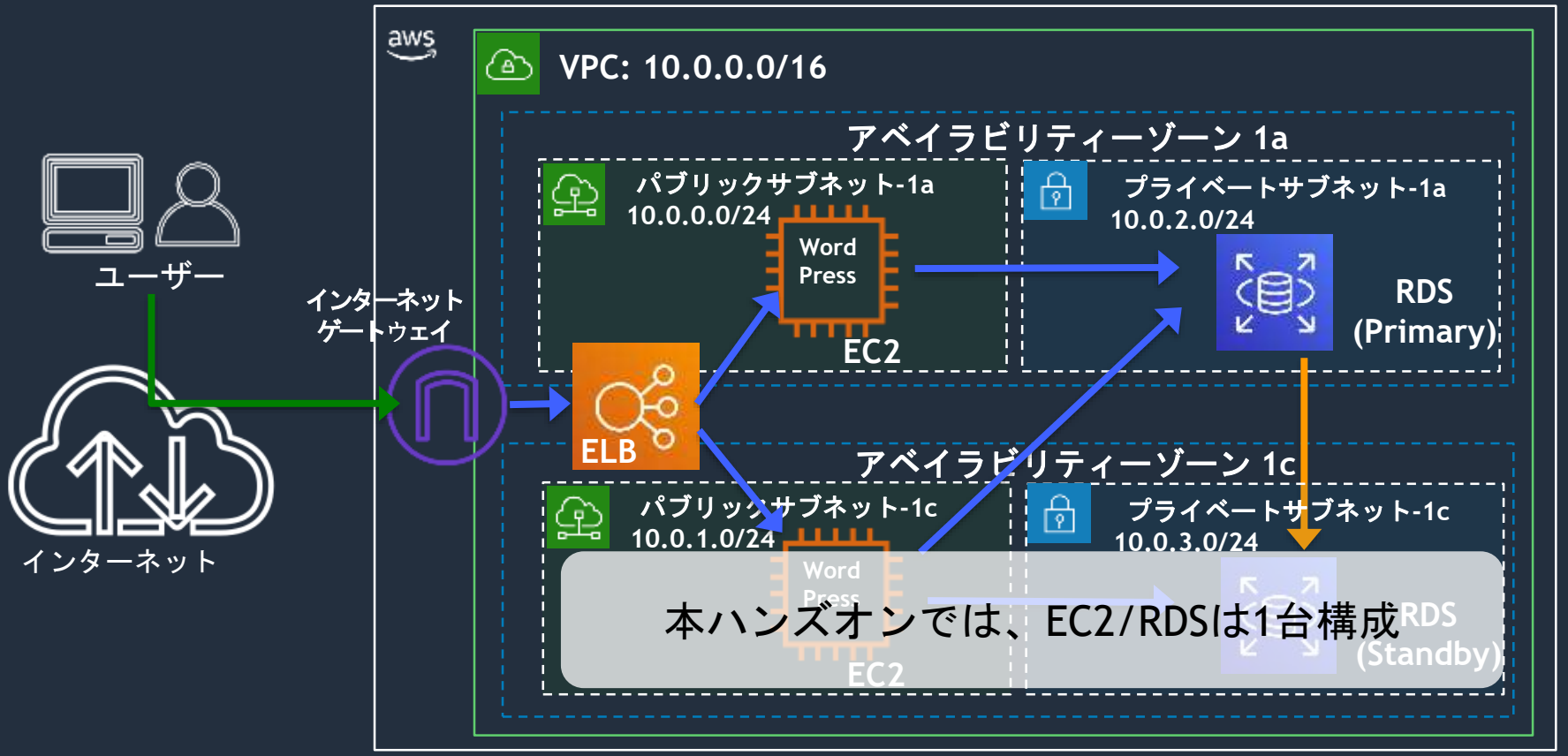
aws

本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

aws

ハンズオンで構築する構成



なぜコードで管理するのか

- 構成がコード管理されていないと…
  - 時間がかかり、再現性に乏しく、複製が困難
  - アプリケーションを含むワークロード全体の品質担保に課題

- イベントへの対応がコード管理されていないと…
  - 人為的ミス発生や対応内容の一貫性への懸念

Well-Archtected フレームワーク

運用上の優秀性の柱

1. 運用をコードとして実行する
2. ドキュメントに注釈を付ける
3. 定期的に、小規模な、元に戻すことができる変更を適用する
4. 運用手順を定期的に改善する
5. 障害を予想する
6. 運用上のすべての障害から学ぶ

※ AWS Well Architected フレームワーク
https://aws.amazon.com/jp/architecture/well-architected/
https://wa.aws.amazon.com/wat.pillar.operationalExcellence.ja.html

コードを使った運用のポイント from AWS Well-Architected

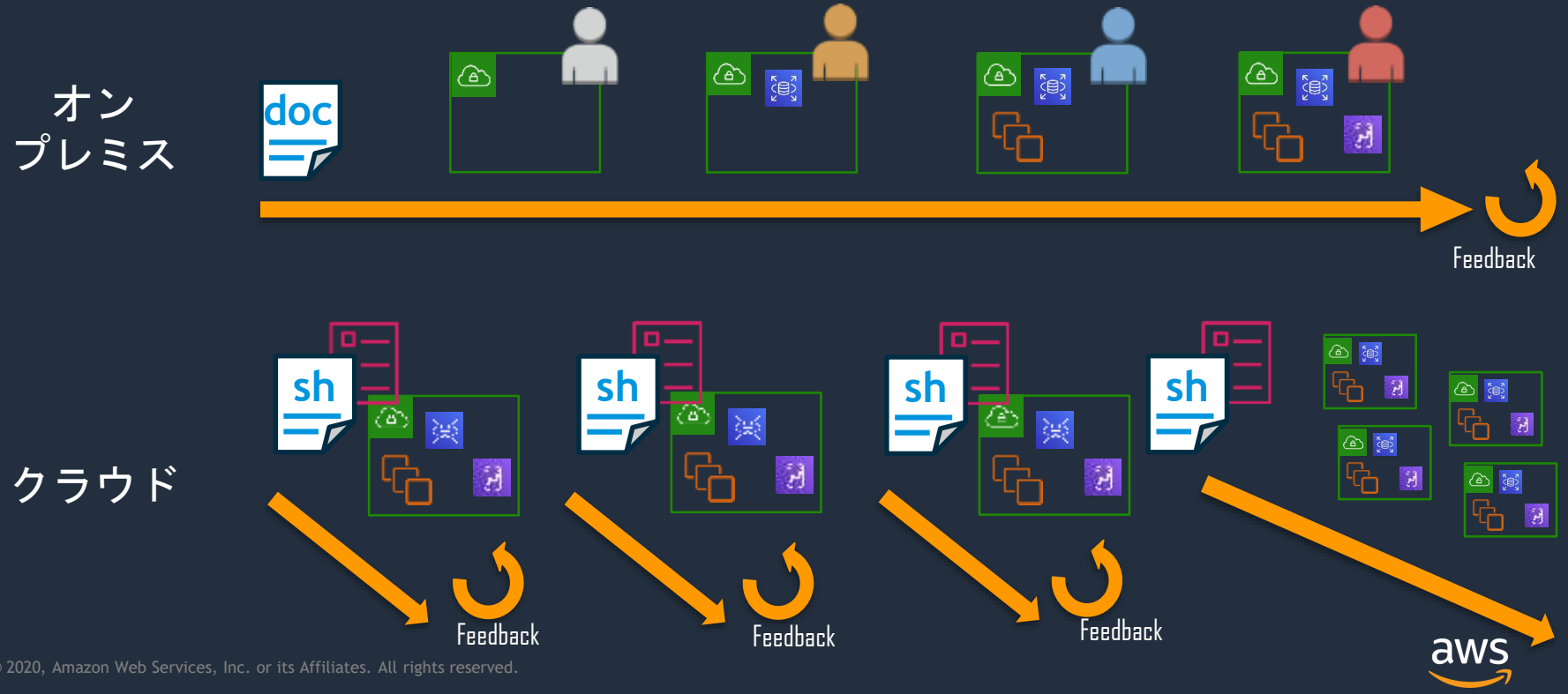
1. コードで全ての構成を定義

2. イベントに対してスクリプトで対処

3. アプリケーションと同じ手法でコードを開発

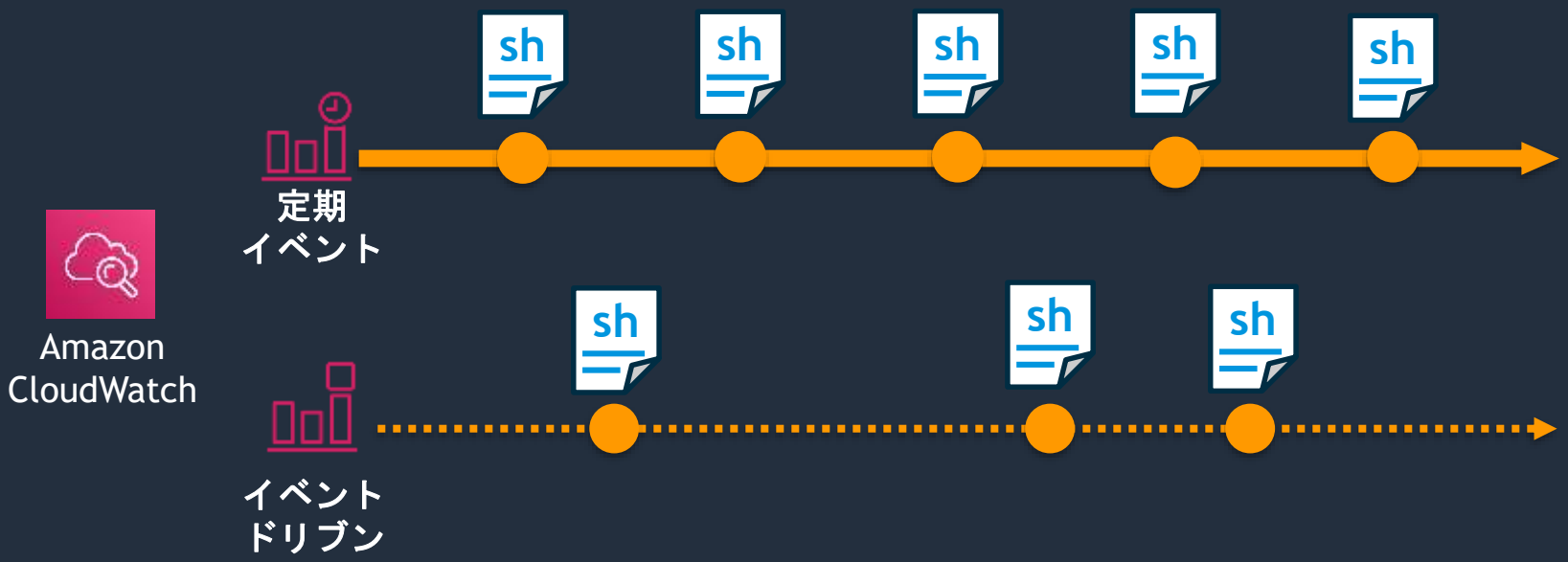
1. コードで全ての構成を定義

同じ環境を、迅速に、繰り返し作成可能



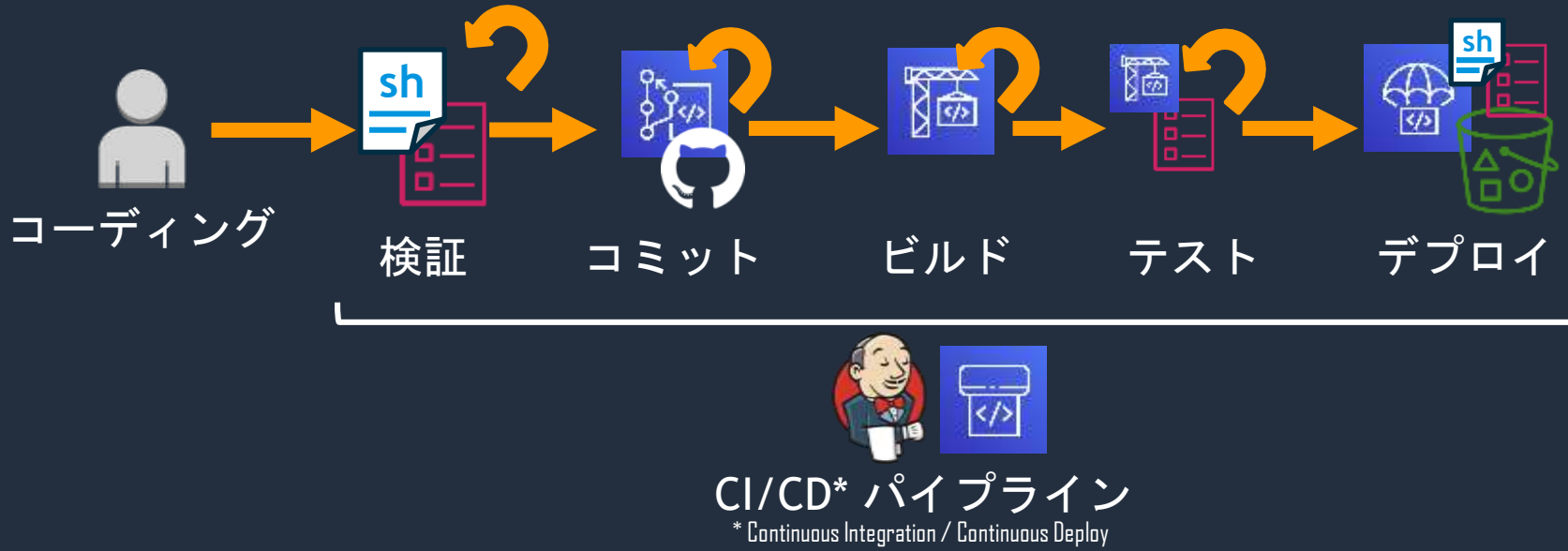
2. イベントに対してスクリプトで対処

自動的に、同じ処理を、繰り返し実施可能



3. アプリケーションと同じ手法でコードを開発

コードと 作成した環境の品質を担保



コードを使った運用のポイント from AWS Well-Architected

再掲

1. コードで全ての構成を定義

2. イベントに対してスクリプトで対処

3. アプリケーションと同じ手法でコードを開発

aws

AWS環境をコード管理するためのサービス・ツール

AWS CloudFormation

AWS Cloud Development Kit
(AWS CDK)

AWS Command Line Interface
(AWS CLI)

AWS Tools and SDKs

※ SDKを利用したサード
パーティーのツールもある

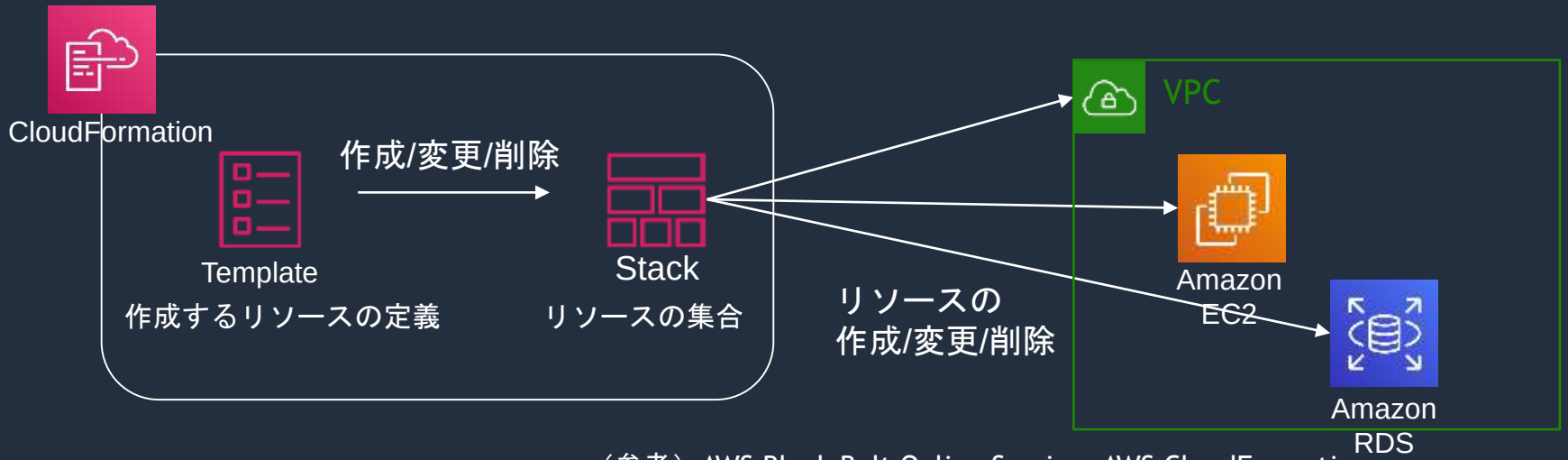
本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

aws

AWS CloudFormation

- AWS環境のコードによる管理を実現
- テンプレートで定義した環境を 作成 / 変更 / 削除



（参考）AWS Black Belt Online Seminar AWS CloudFormation
https://www.slideshare.net/AmazonWebServicesJapan/aws-black-belt-online-seminar-aws-cloudformation

CloudFormation 基本機能

- 作成
  - テンプレートに定義された構成でスタックを自動作成
  - 並列でリソースを作成し依存関係がある場合は自動的に解決
- 変更
  - スタックに前回のテンプレートとの差分を適用（冪等性）
  - リソース変更時は 無停止変更 / 再起動 / 再作成 のいずれかが発生
  - Change Set を作ることで差分の内容を事前に確認可能
- 削除
  - 依存関係を解決しつつリソースを全て削除
  - データストアはスナップショット取得 / 保持が可能

注意： 手動で行った変更はCloudFormation管理外

テンプレートとスタックの関係

テンプレート

JSON/YAML形式のテキスト

リソースの定義
パラメータの定義

CloudFormation

フレームワーク

スタックの作成/変更/削除
エラー検知とロールバック

スタック

AWS リソースの集合

テンプレートとスタックの関係

テンプレート

JSON/YAML形式のテキスト

リソースの定義
パラメータの定義

CloudFormation

フレームワーク

スタックの作成/変更/削除
エラー検知とロールバック

スタック

AWS リソースの集合

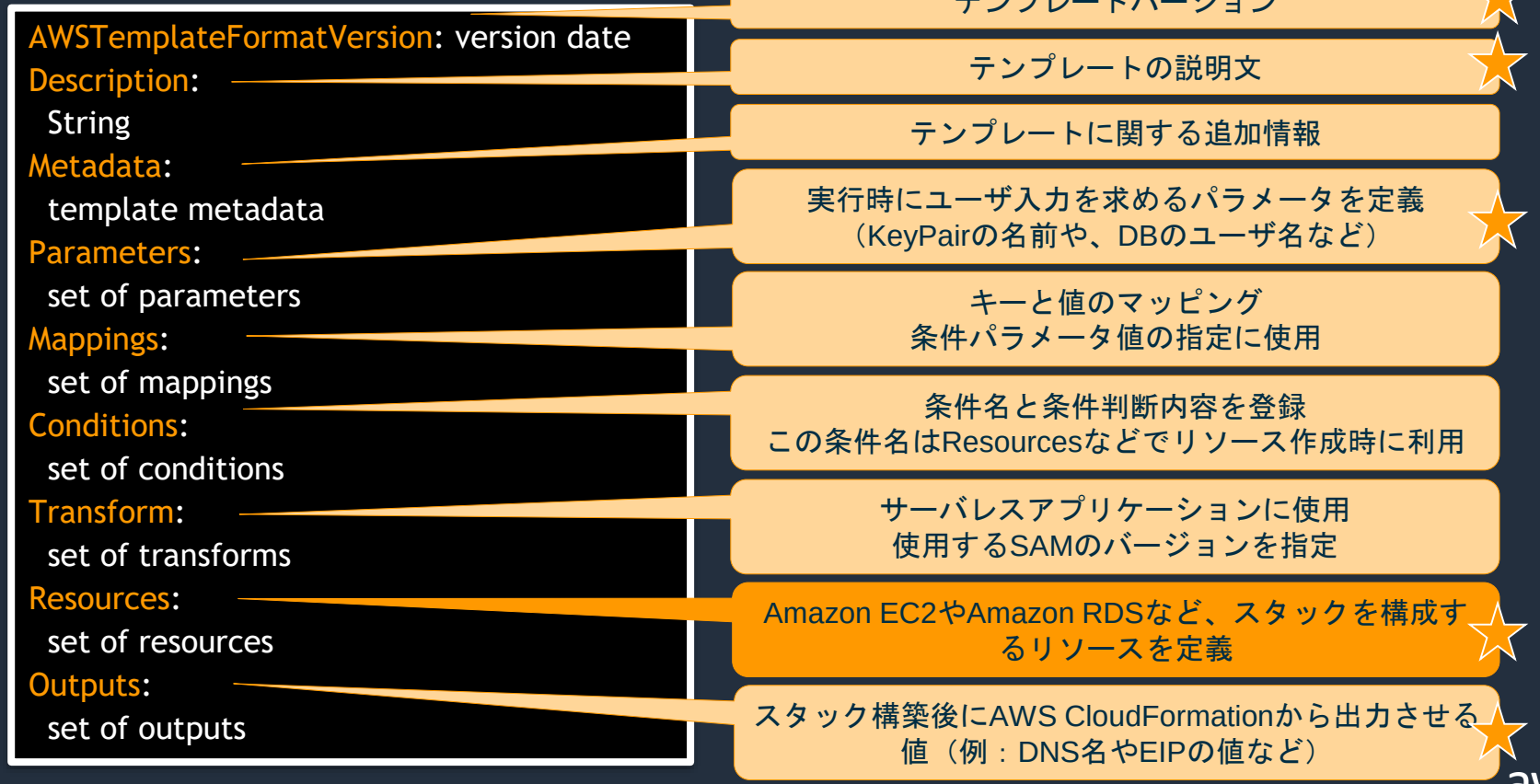
テンプレート

- CloudFormationの心臓部

- スタックの設計図
  - どのリソースをどう起動するかを記述
  - リソースの依存関係は CloudFormationが自動判別

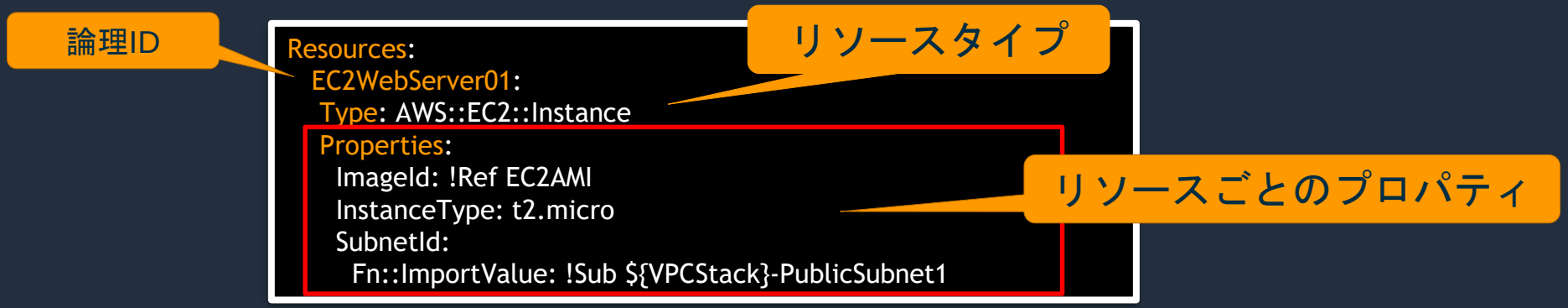
```
AWSTemplateFormatVersion: 2010-09-09
Description: Hands-on template for EC2

Parameters:
  VPCStack:
    Type: String
    Default: handson-cfn
  EC2AMI:
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-amazon-linux-latest/amzn2-ar

Resources:
  EC2WebServer01:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref EC2AMI
      InstanceType: t2.micro

Outputs:
  EC2WebServer01:
    Value: !Ref EC2WebServer01
    Export:
      Name: !Sub ${AWS::StackName}-EC2WebServer01
```

テンプレートの要素（セクション）



テンプレートの要素 – Resources (必須)

- EC2やELB、 RDSなど、起動するリソースを指定
- リソースごとに決められたプロパティを設定
- 利用可能なリソースタイプは 483種　※最新リストはリファレンスを参照



（参考）公式ドキュメントのテンプレートリファレンスとリリース履歴
https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html
https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/ReleaseHistory.html

テンプレートの要素 – Format version and Description

- Format version
  - 現状「2010-09-09」が唯一有効な値

- Description
  - テンプレートに関する説明文

```
AWSTemplateFormatVersion: 2010-09-09
Description:  some details about the template.
```

テンプレートの要素 – Parameters

- スタック構築時にユーザに指定させる値を定義
- データ型、デフォルト値、最大最小値などのプロパティを設定可能

パラメータごとのプロパティ

```
Parameters:
  VPCStack:
    Type: String
    Default: handson-cfn
  DBUser:
    Type: String
    Default: dbmaster
  DBPassword:
    Type: String
    Default: H&ppyHands0n
    MinLength: 8
    NoEcho: true
```

テンプレートの要素 – Outputs

- スタック構築後に取得・表示したい情報の定義
- アクセスURLや、DBの通信先情報、作成したIAMユーザー名など、あとで使用したい情報がある場合に便利

```
Resources:
  EC2WebServer01:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref EC2AMI
      InstanceType: t2.micro
Outputs:
  EC2WebServer01:
    Value: !Ref EC2WebServer01
    Export:
      Name: !Sub ${AWS::StackName}-EC2WebServer01
```

出力データの名称

出力する値

組み込み関数を使って文字列を加工

キー	値	説明	エクスポート名
EC2WebServer01	i-0bad9055e937004e5	-	handson-cfn-ec2-EC2WebServer01

テンプレートの要素 – Function、疑似パラメータ

組み込み関数

```
Fn::Base64
Fn::Cidr
Fn::FindInMap
Fn::GetAtt
Fn::GetAZs
Fn::ImportValue
Fn::Join
Fn::Select
Fn::Split
Fn::Sub
Fn::Transform
Ref
```

```
Fn::And
Fn::Equals
Fn::If
Fn::Not
Fn::Or
```

疑似パラメータ

```
AWS::AccountId
AWS::NotificationARNs
AWS::NoValue
AWS::Partition
AWS::Region
AWS::StackId
AWS::StackName
AWS::URLSuffix
```


aws

テンプレートのまとめ

- JSONもしくはYAML形式でスタックの情報を記載
- 定義された構成を並列で作成し、リソースに依存関係がある場合はCloudFormationが自動的に解決する
- Parametersを利用すれば、パラメータのユーザ入力を受け取れる
- マネジメントコンソール等に値を出力したい場合はOutputsに記述する

テンプレートとスタックの関係

テンプレート

JSON/YAML形式のテキスト

リソースの定義
パラメータの定義

CloudFormation

フレームワーク

スタックの作成/変更/削除
エラー検知とロールバック

スタック

AWS リソースの集合

スタック

- 単一のユニットとして管理できる AWS リソースのコレクション
- スタック単位でリソースの管理が可能
  スタックの削除を実行すると、スタックにひもづくリソースが削除される
- 使用するリソースおよびリソースの構築順は、テンプレートの依存関係から CloudFormationが自動的に決定

参考資料

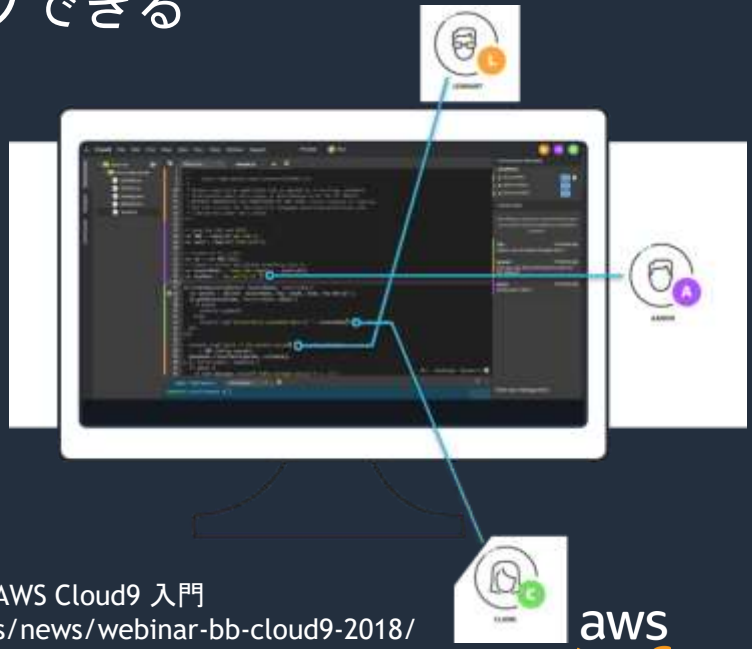
- ユーザーガイド
  - ベストプラクティス、テンプレートリファレンス等
  - https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/Welcome.html
- サンプル
  - https://github.com/awslabs/aws-cloudformation-templates
  - https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/cfn-sample-templates.html

本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

AWS Cloud9

- ブラウザのみでコードを記述、実行、デバッグ可能、クラウドベースの統合開発環境（IDE）
- 複数人でリアルタイムに共同コーディングできる
- AWSサービスに直接ターミナルアクセス
- AWS CLIがプリインストール



（参考）AWS Black Belt Seminar: AWS Cloud9 入門
https://aws.amazon.com/jp/blogs/news/webinar-bb-cloud9-2018/

ここで行うこと

- 開発環境としてAWS Cloud9の環境を構築
  - マネジメントコンソールからインスタンスを作成
  - Cloud9の環境確認と設定
    - バージョン情報の確認
    - AWS CLIの補完設定
    - エディタ設定

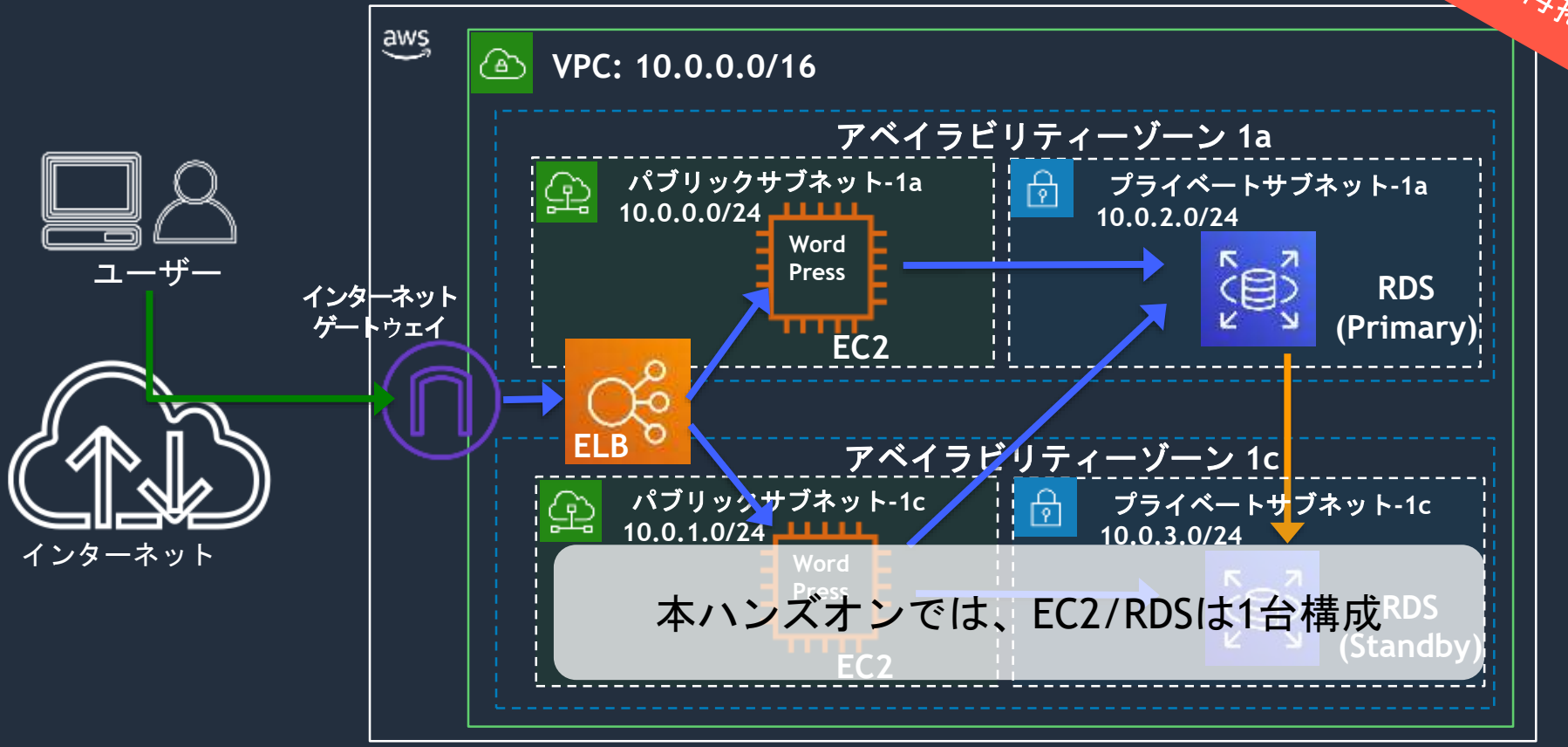
本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

aws

ハンズオンで構築する構成

再掲



スタックの操作

- AWS マネジメントコンソール
- AWS Command Line Tool (AWS CLI)
  - http://aws.amazon.com/cli/
- 各種SDK
  - https://aws.amazon.com/jp/getting-started/tools-sdks/
  - JavsScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, C++

ここで行うこと

- マネジメントコンソールからVPC用スタックを作る
- Cloud9でのテンプレートの編集
- AWS CLIを利用してスタックの更新

作成した環境

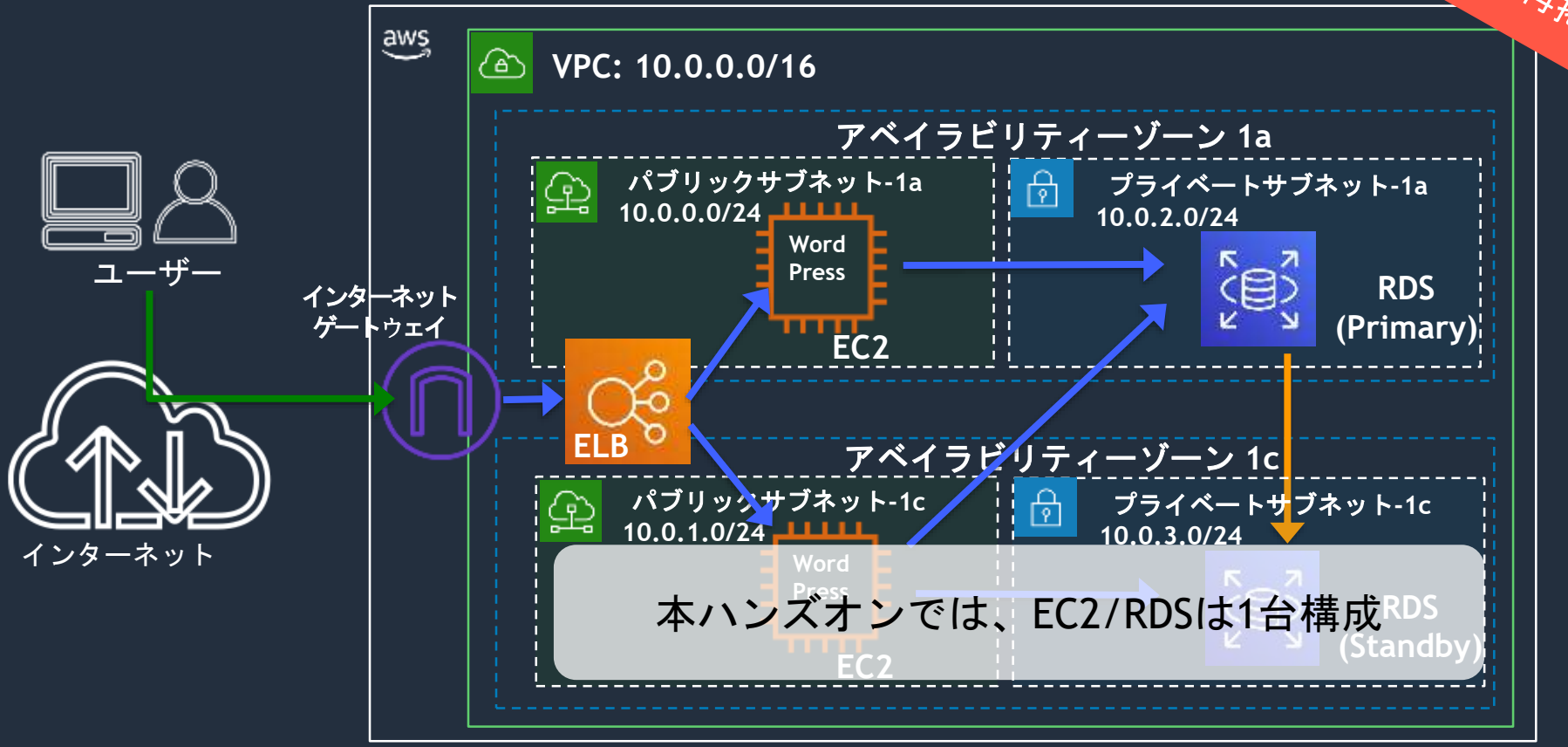
aws

本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

ハンズオンで構築する構成

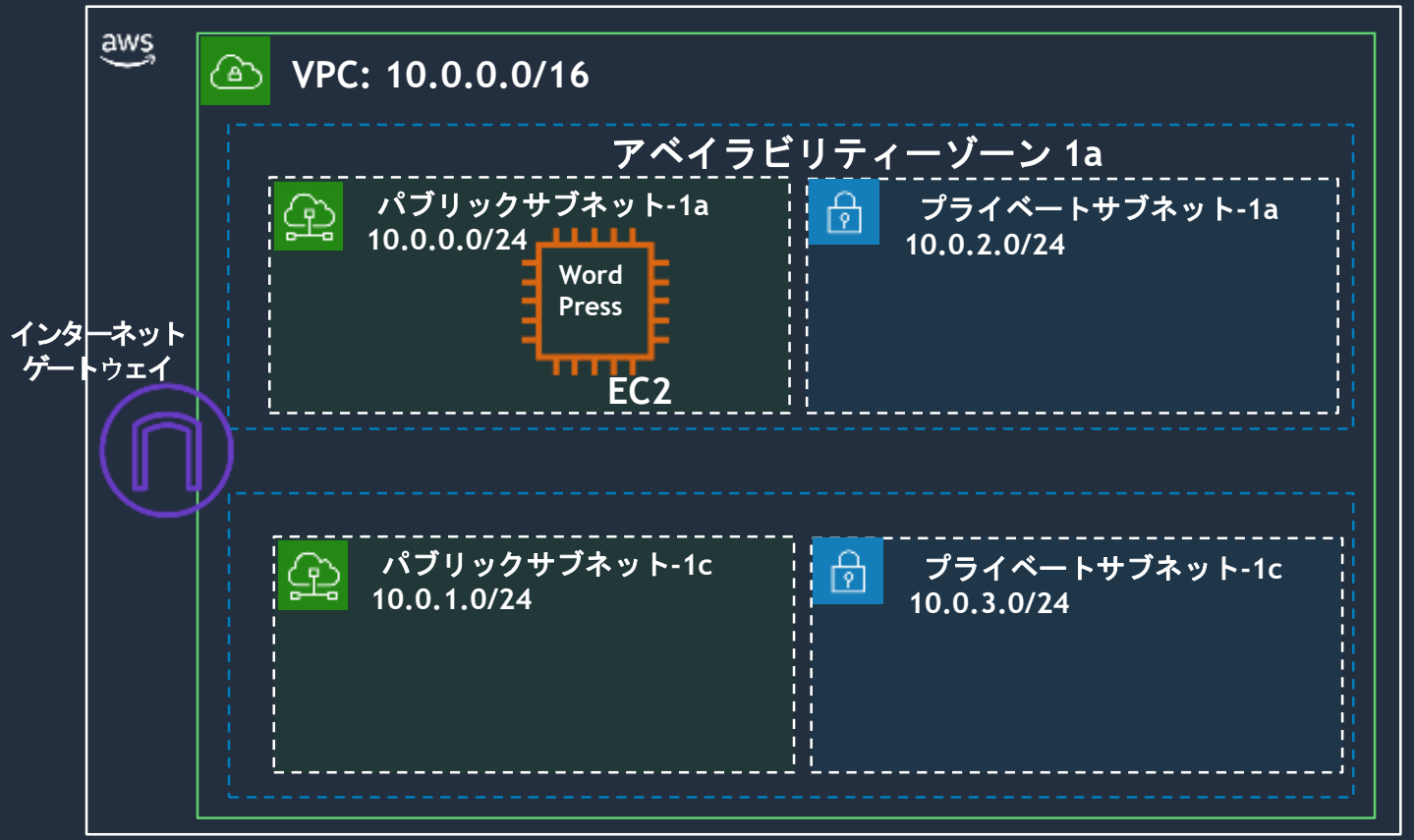
再掲



ここで行うこと

- EC2用スタックの作成
  - 配布したテンプレートでEC2を起動
  - 不足設定を書き加えて、テンプレートを完成させる

作成した環境



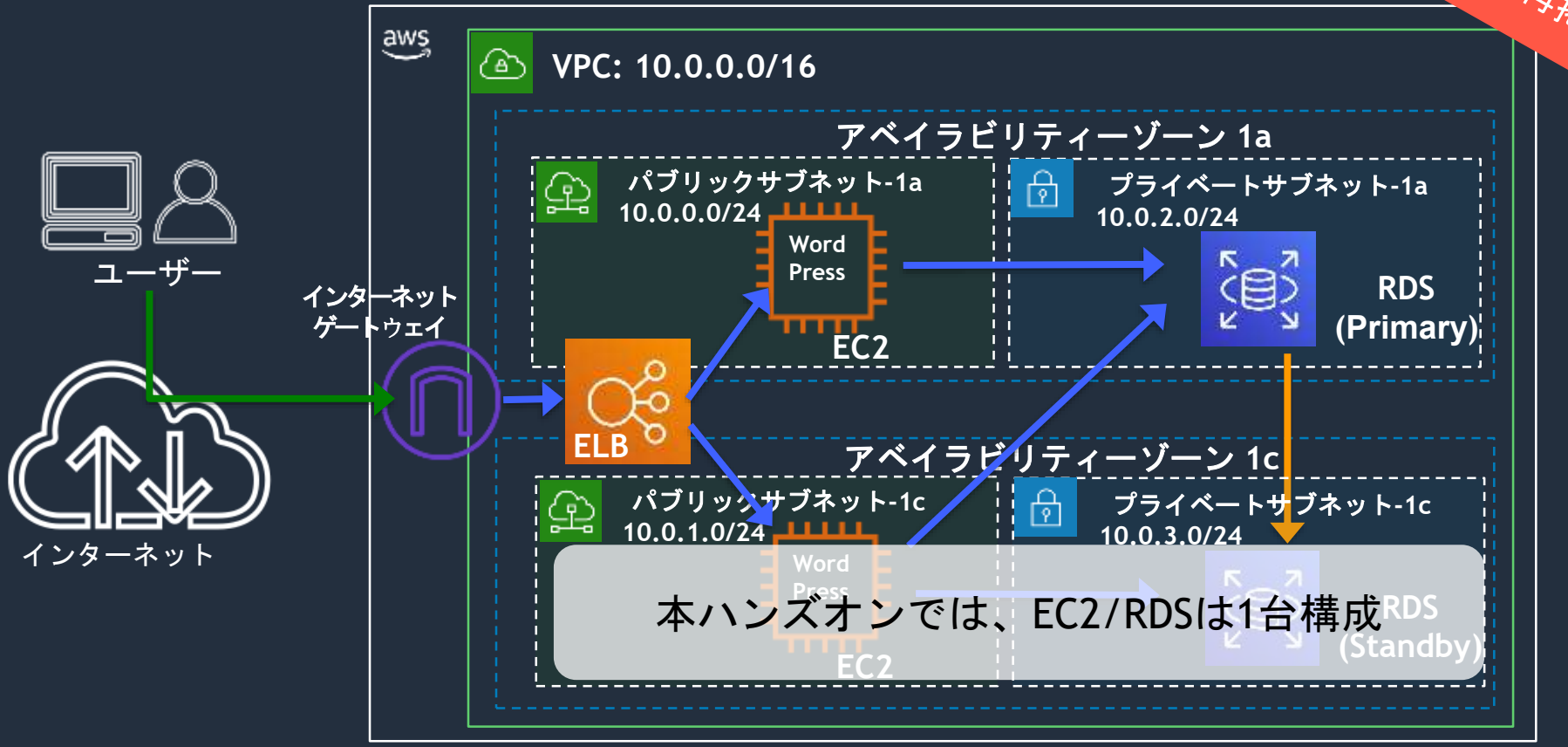
aws

本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

ハンズオンで構築する構成

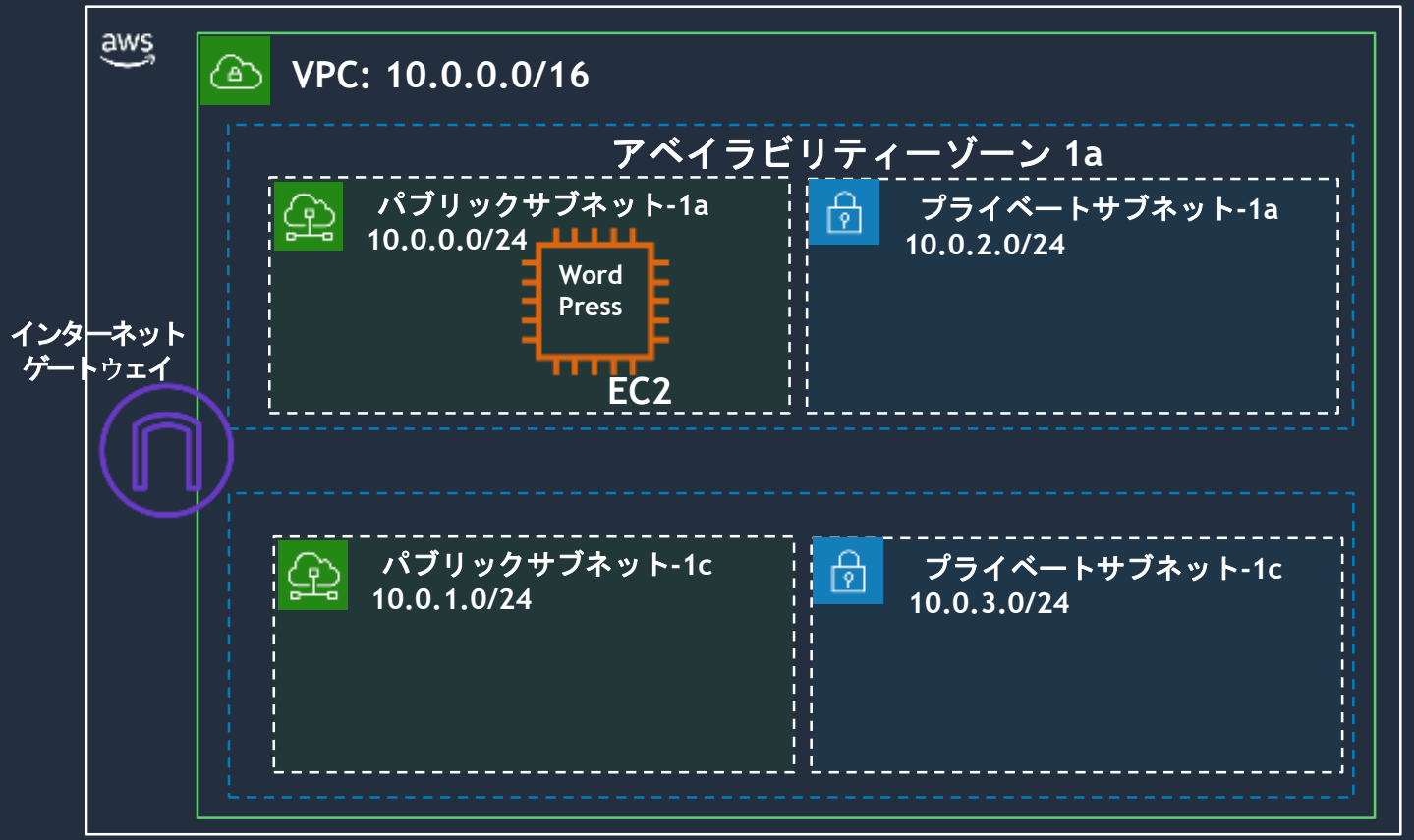
再掲



ここで行うこと

- EC2用スタックの作成
  - 不足設定を書き加えて、テンプレートを完成させる

作成した環境



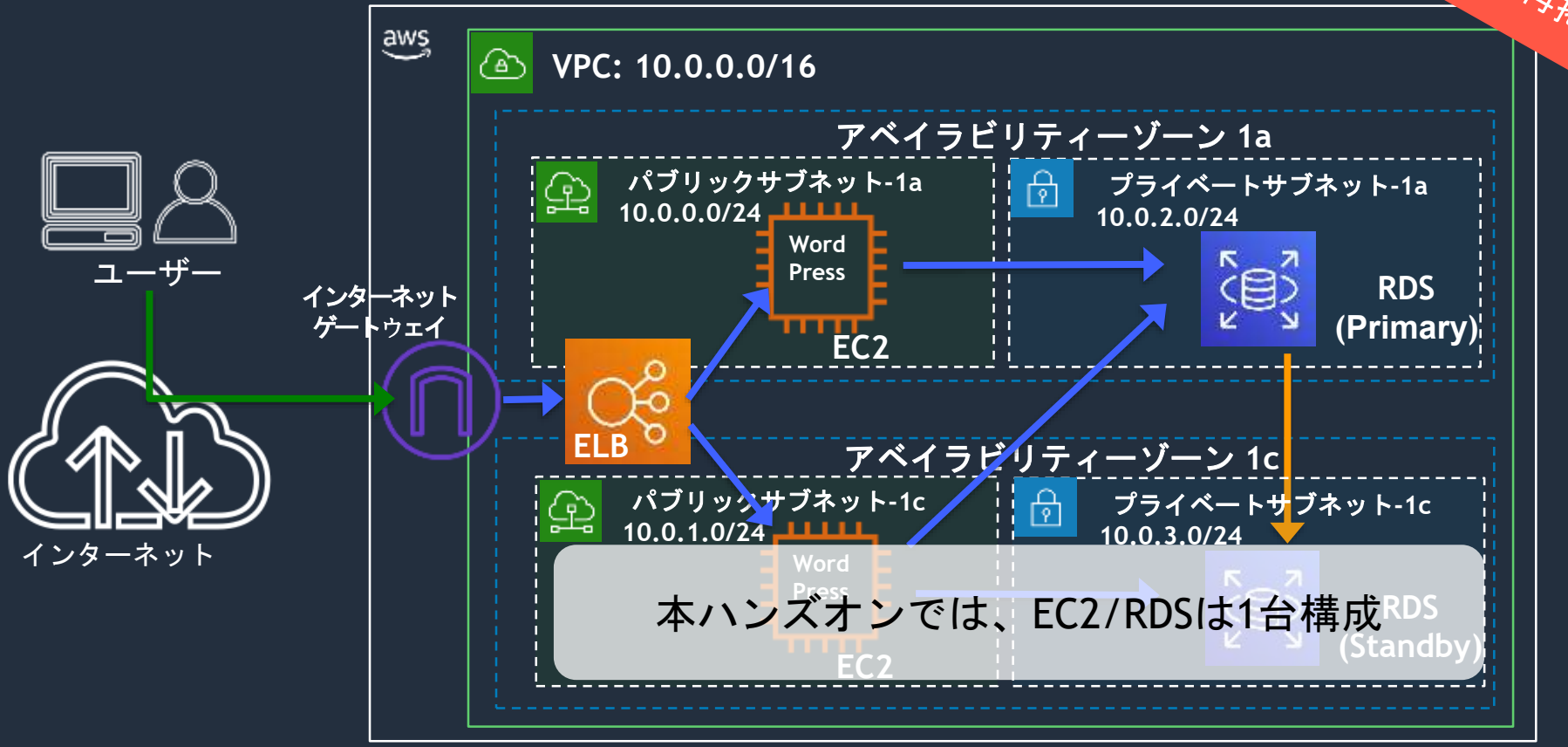
aws

本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

ハンズオンで構築する構成

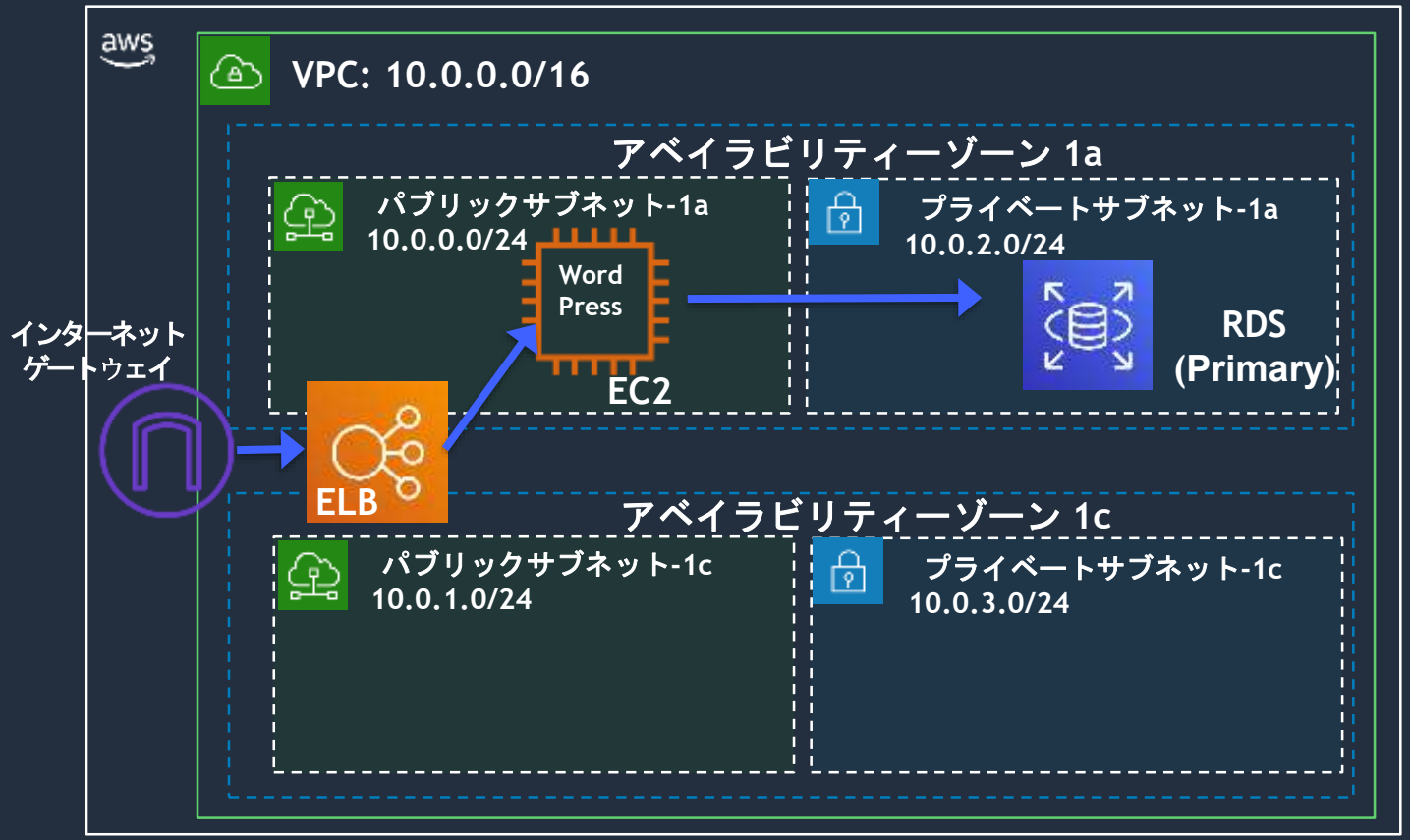
再掲



ここで行うこと

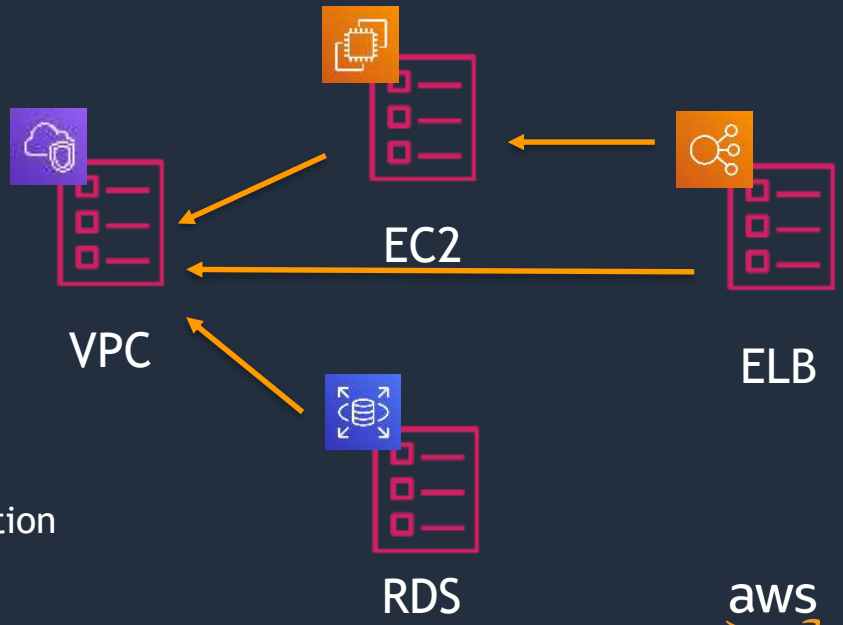
- RDS・ELB用スタックの作成
- 完成した構成の動作を確認

作成した環境



スタックのライフサイクル・分割について

- 個々のスタックは、ライフサイクルと所有者を基準に分割
- VPC用スタックをベースとして、他スタックからクロススタック参照によって連携



※ AWS CloudFormation のベストプラクティス
https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/best-practices.html#cross-stack

本シリーズのアジェンダ

1. ハンズオンで構築する構成とクラウドにおける構成管理について
2. CloudFormation概要
3. 開発環境の構築 ✋
4. テンプレートの実行方法、VPCの作成 ✋
5. EC2の作成 その1 ✋
6. EC2の作成 その2 ✋
7. RDS, ELBの作成、スタックのライフサイクル・分割 ✋
8. まとめ ✋ 、削除

aws

環境の削除

アンケートも
よろしくお願いします！

- 以下スタックを削除し、ハンズオンは終了です
  - handson-cfn-elb
  - handson-cfn-rds
  - handson-cfn-ec2
  - handson-cfn
  - aws-cloud9-handson～

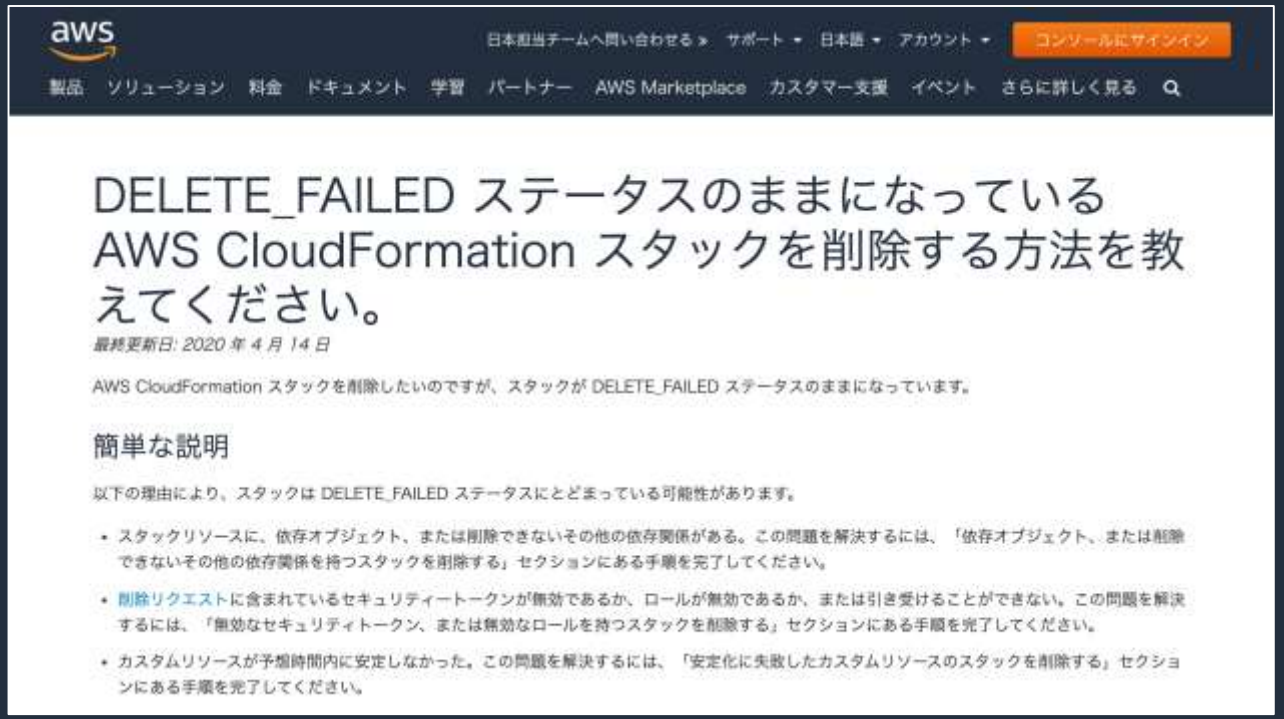
※ マネジメントコンソールから削除する場合

※ AWS CLIで削除する場合

aws cloudformation delete-stack --stack-name スタック名

スタックの削除に失敗したとき

「CloudFormation 削除 失敗」で検索

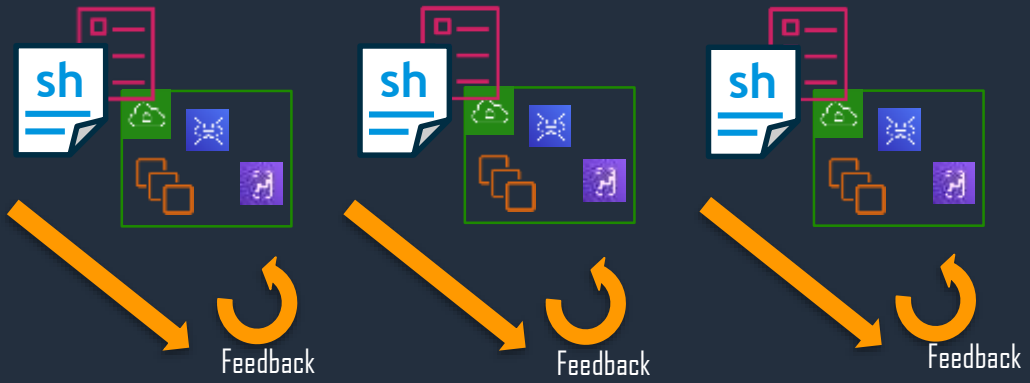


https://aws.amazon.com/jp/premiumsupport/knowledge-center/cloudformation-stack-delete-failed/

まとめ

アンケートも
よろしくお願いします！

- スタックの作成・更新を実施していただきました
- テンプレートを記述できるようになっていただきました
- フィードバックを繰り返し反映する開発フローを体験していただきました



今後のラーニングパス

アンケートも
よろしくお願いします！

- AWS CloudFormationのベストプラクティスを読み解く
  https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/best-practices.html

- インフラ構築に導入してみる

- CodePipelineを利用したデプロイ
  https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/continuous-delivery-codepipeline.html

- AWS Hands-on for Beginnersの他シリーズ
  https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-hands-on/

aws

アンケートも
よろしくお願いします！