AWS DevOps 祭り 2018

AWS Management Tools サービスアップデートのご紹介

大 村幸敬
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

大村 幸敬（おおむら ゆきたか）

ソリューションアーキテクト

- お客様のパートナーとして
  AWSに限らず最適なアーキテクチャ検討をサポート
- エンタープライズ企業を担当
- Management Tools & DevOps 系サービス担当

Background : SI / Infra / Financial / DevOps

アジェンダ

1. DevOps と AWS Management Tools
2. AWS Management Tools の概要
3. 各サービスの使い所と直近のアップデート
4. マルチアカウント管理

Notice

- 直近3ヶ月からre:Invent 2018 までのアップデートを中心に取り扱います

DevOps と AWS Management Tools

AWS re:Invent

aws

What is DevOps?

デリバリのパイプライン

開発者

build → test → release

plan ← monitor

顧客

フィードバックループ

DevOps = 無駄やボトルネックを取り除くことで、
ライフサイクルを効率化し、高速化すること

Metrics for DevOps

作り始めてからデプロイされるまでのストリームが重要
ストリームのベロシティ（速度）をどう改善するか

ベロシティ向上の阻害要因

心配

責任

ベロシティ向上のために自動化を

- 不安を取り除くためにシステムを使う
  - 正しくOperationが行えるスクリプト
  - 問題があったときに止められるシステム
  - バグがあったときに再発防止できるテストコード

全てを自動化する

自動化は最適化を可能にする

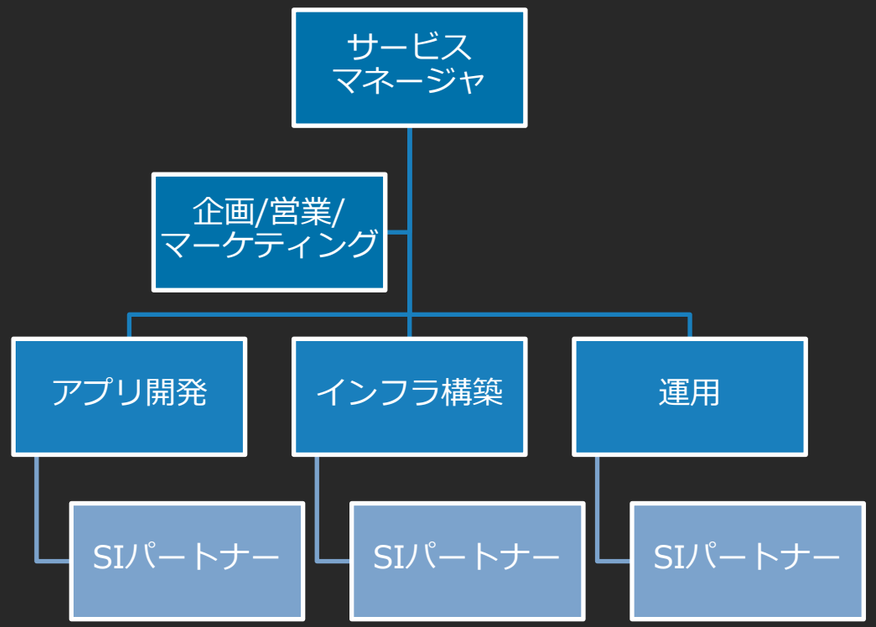
- “どのような手順も明確にされ、自動化することは可能”
  - Werner Vogels

インフラ周りの新しいマインドセット

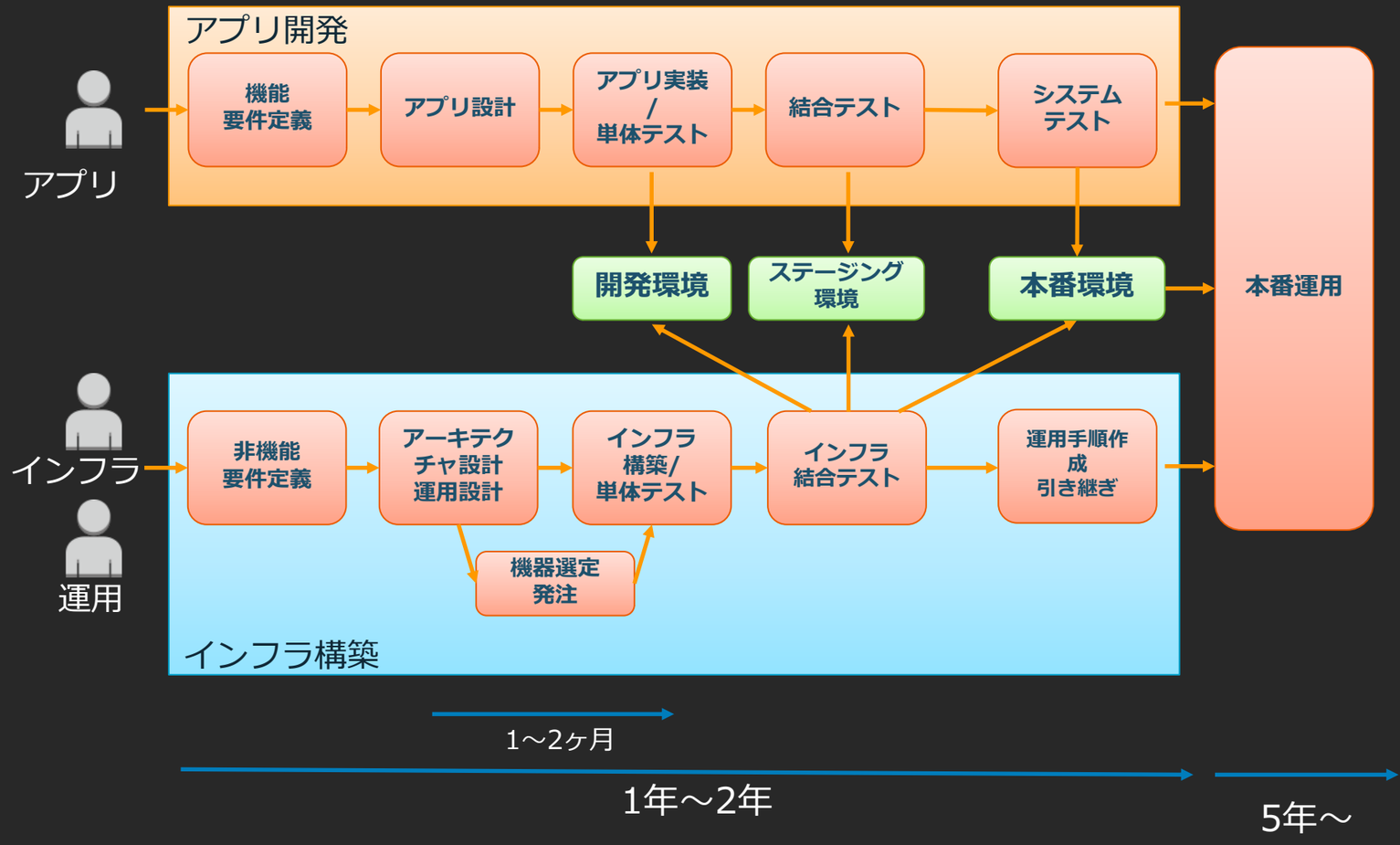
- 環境に関するすべてのことはコードで説明される
- 自動化はサービスによってもたらされる
- 自動化はツールとフレームワークの要件を導出する
- 頻繁なリリースは、頻繁なイノベーションを起こす

オンプレミスでの開発の一般的な体制とスケジュール

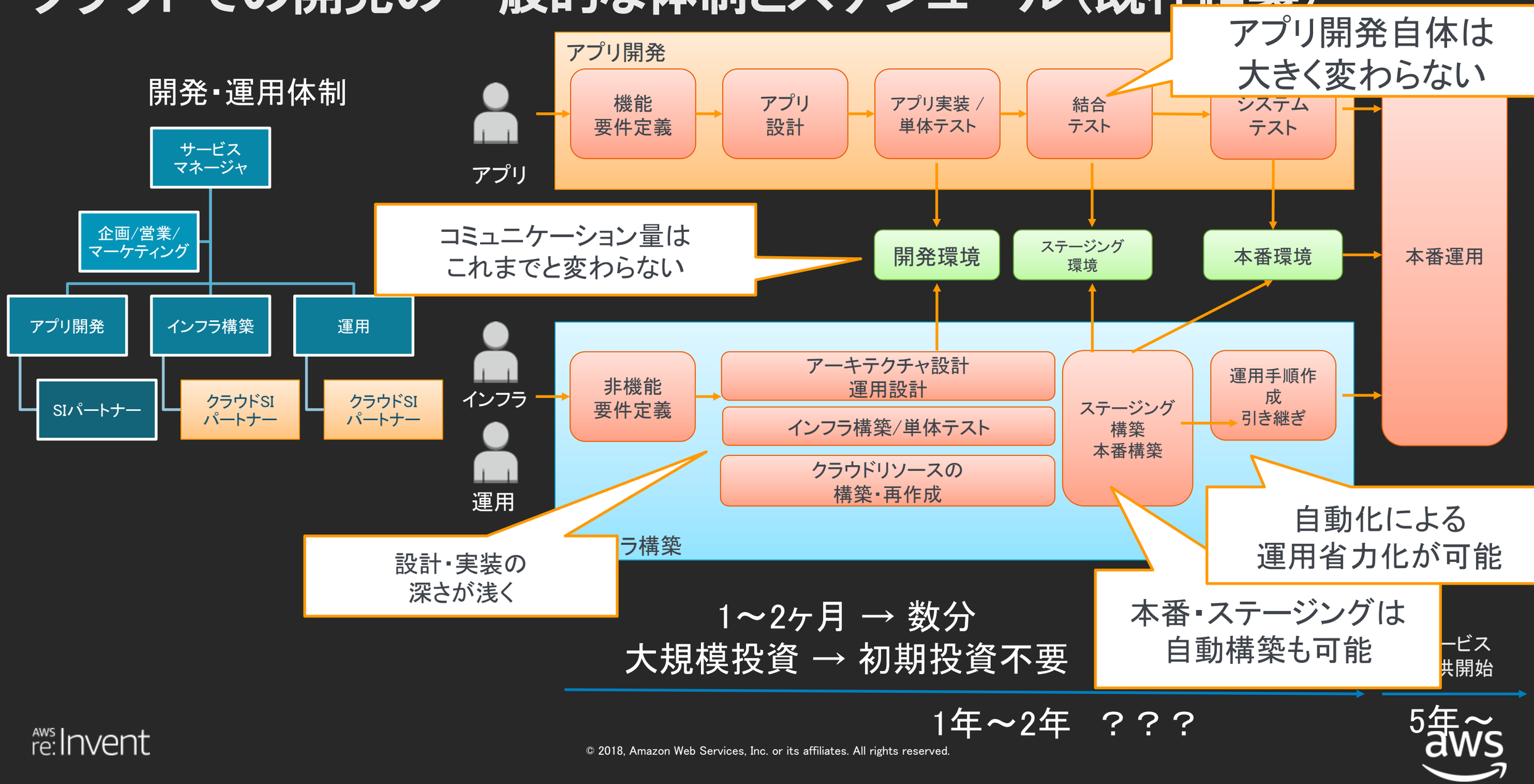
開発・運用体制



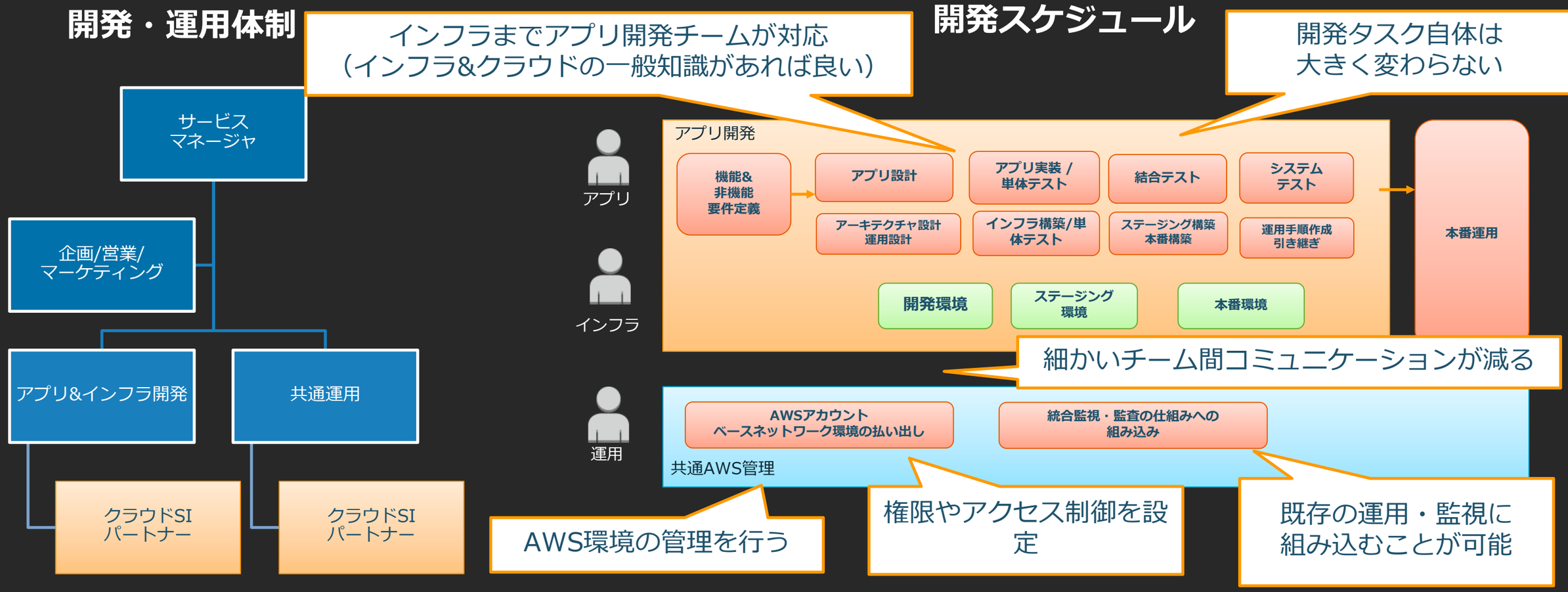
開発スケジュール



クラウドでの開発の一般的な体制とスケジュール（既存踏襲）

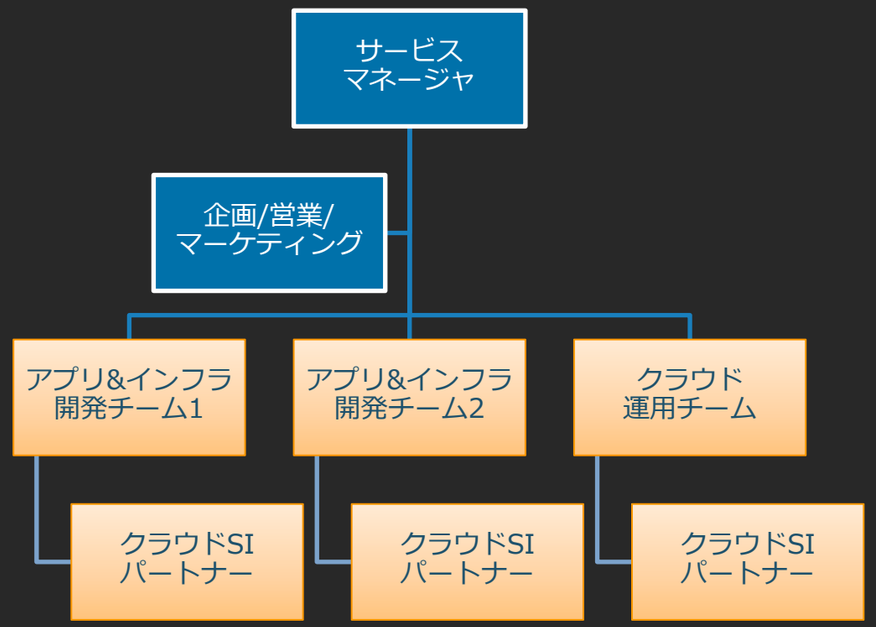


クラウドを活用した開発体制



クラウド上のアジャイル(DevOps)開発体制とスケジュール

開発・運用体制



開発スケジュール

クラウドによりインフラもアプリも
ユーザからのフィードバックを得つつ
開発することが可能に

運用における2つの選択肢

分散化とガードレール

- セルフサービス、実験、革新
- アジリティの向上
- DevOpsの実現
- 危険な操作の定義と対応
- 新メンバの操作の監査

ロックダウンと事前承認

- 完全な制御と実験余地の剥奪
- アジリティの低下
- 承認者数の増加
- 開発者が関与しない

クラウドを活用したDevOpsのポイント

- ビジネス（アプリ開発）チームがアプリ/インフラ/運用まで担当
  - 利害関係の異なるチーム間でのコミュニケーションを削減
  - 自動化によりトイル（手作業の運用などスピードを落とす業務）を減らす

- 共通基盤・運用チームはパターン化した環境の提供と
  ガバナンスの確保をメインタスクとする
  - 開発チームのスキルレベルに合わせてセキュリティ設定を提供
  - 払い出したアカウント中で自由に操作可能とする（マルチアカウント）

- システムの一部をクラウド側に任せ、自動化することで
  迅速化、高機能化、低リスク、低コストを実現

AWS Management Tools

AWS Management Tools

Provisioning & entitlement

AWS CloudFormation
AWS Service Catalog

Configuration management

AWS OpsWorks

Monitoring

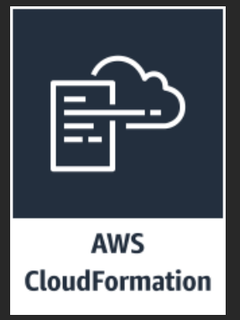
Amazon CloudWatch
AWS X-Ray

Operations and compliance management

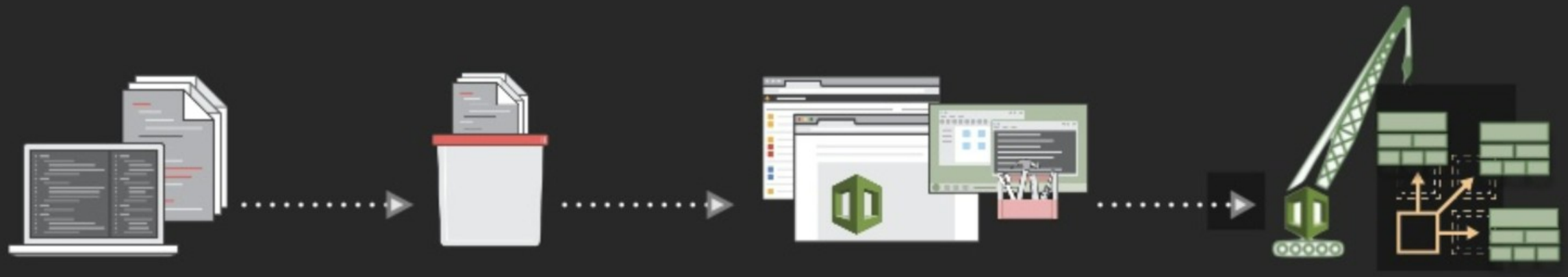
AWS CloudTrail
AWS Config
AWS Systems Manager

Integrated & interoperable

AWS CloudFormation



- 250以上のAWSリソースの作成を自動化
- リソースのアップデートを安全かつロールバック可能な形で実施
- サーバ、コンテナ、サーバレス、いずれのアプリ形態でも対応可能



Code in YAML or JSON directly or use sample templates

Upload local files or from an S3 bucket

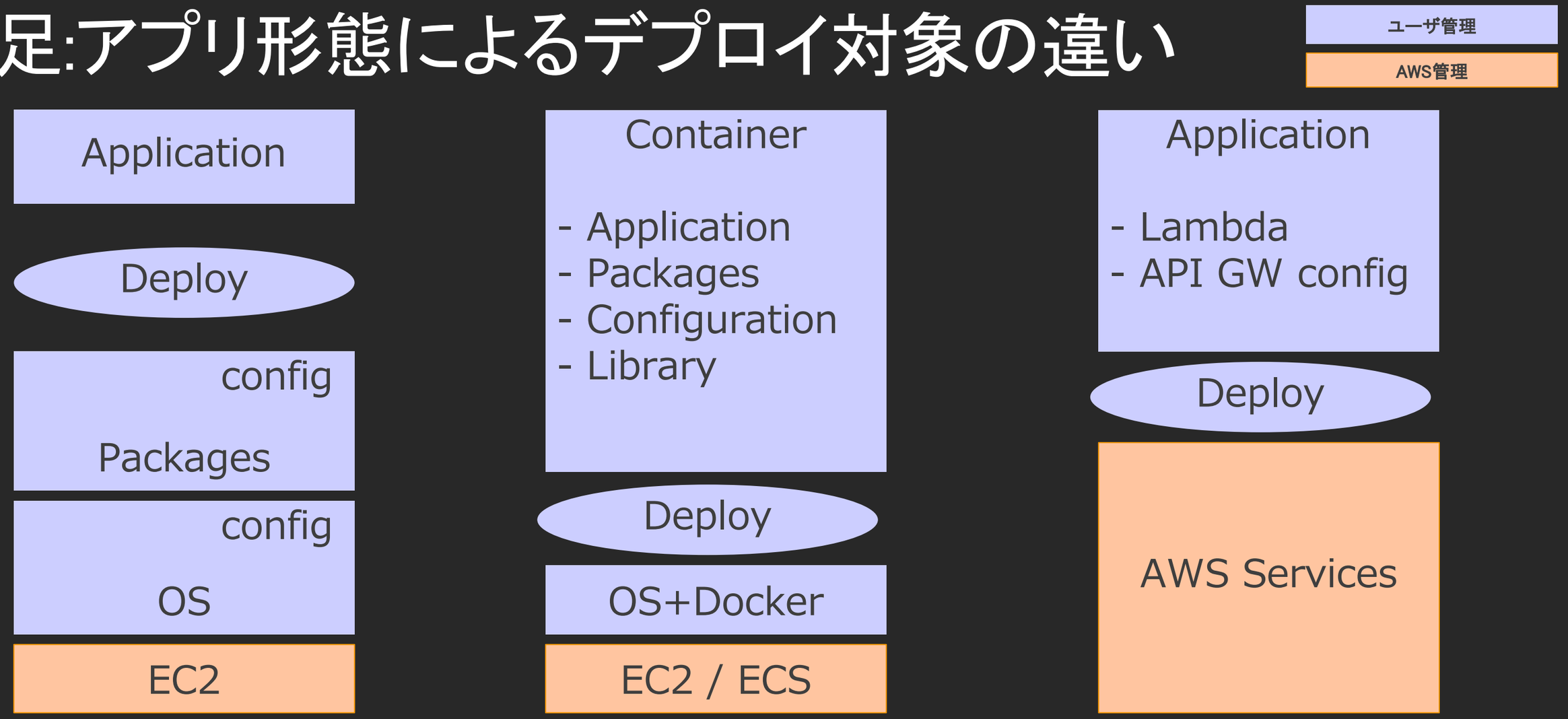
Create stack using console, API or CLI

Stacks and resources are provisioned

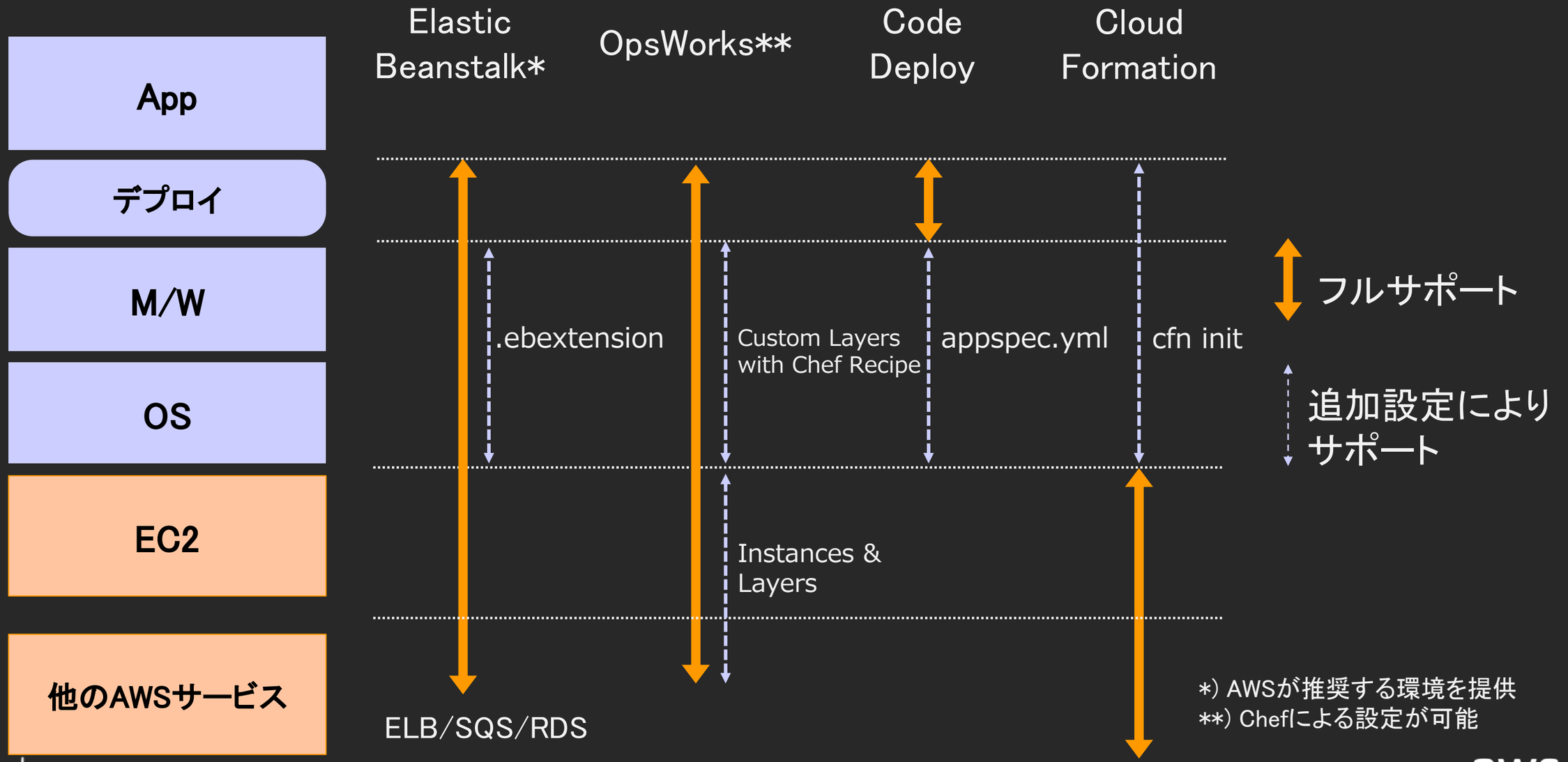
AWS re:Invent

aws

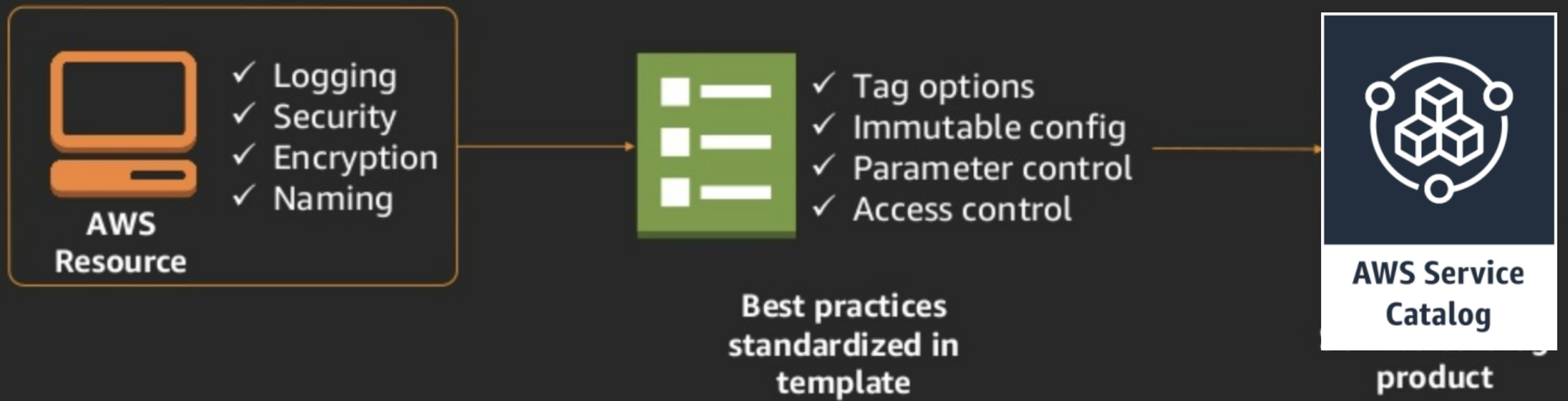
補足:アプリ形態によるデプロイ対象の違い



補足:AWS プロビジョニングサービスのカバー範囲



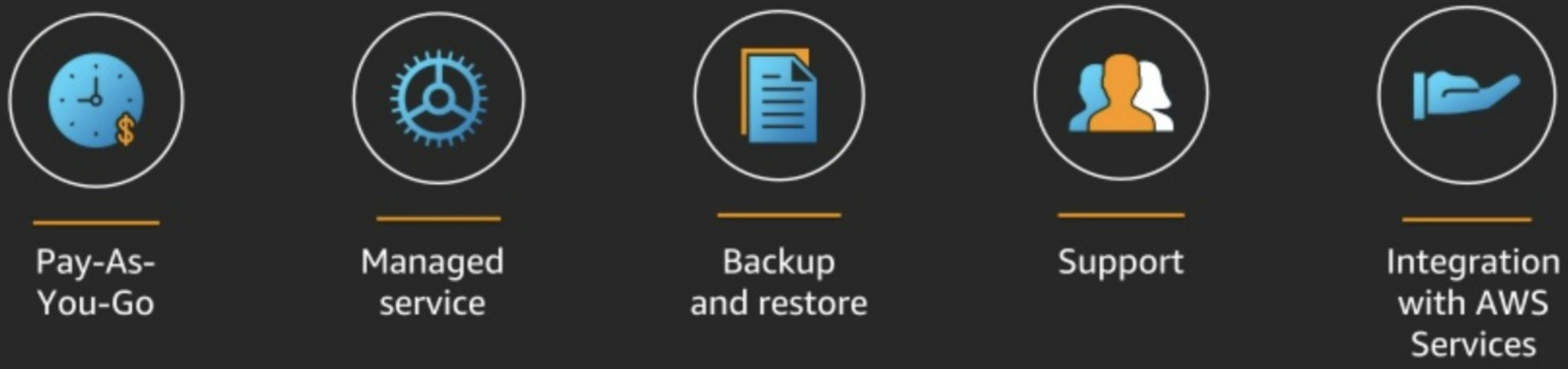
AWS Service Catalog



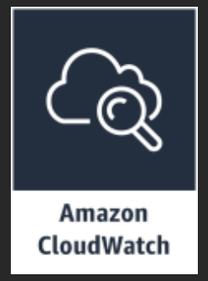
- 固定のベストプラクティステンプレートを作成 & 共有
- AWS サービスの限定されたアクセス制御（SSMで一部操作のみ可能）
- エンドユーザは即時に使えるセルフサービス環境を利用可能

AWS OpsWorks

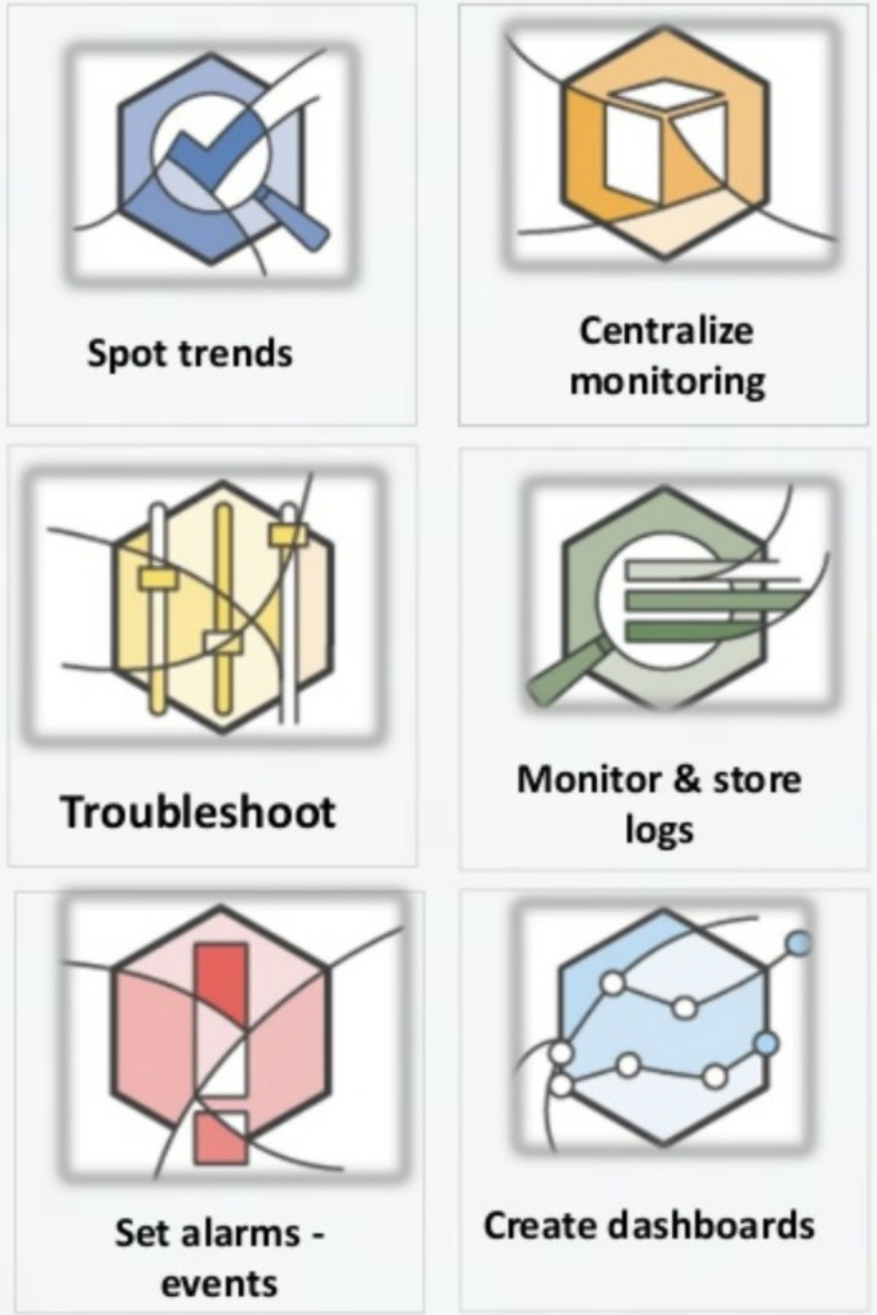
- マネージドなコンフィグレーションサーバを提供
- Chef Automate と Puppet Enterprise に対応
- DSLを使用してコンフィグレーションを管理



Amazon CloudWatch

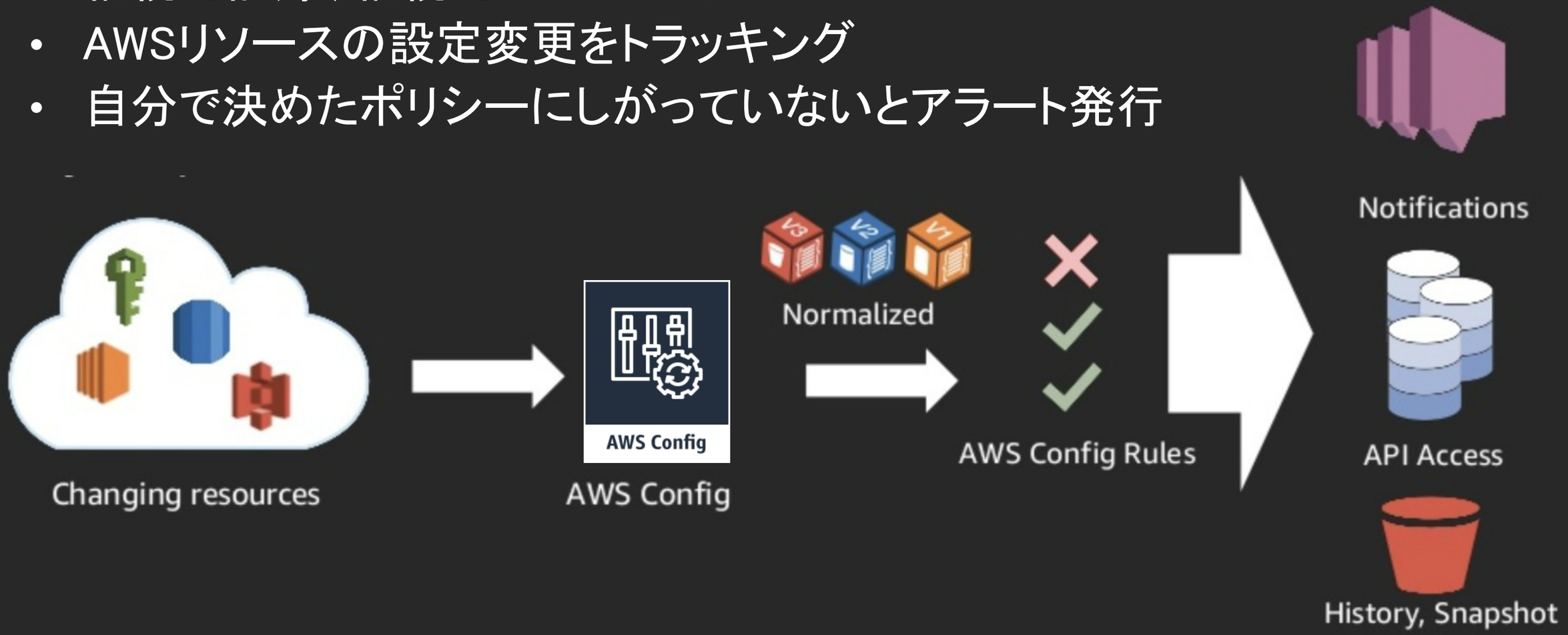


- AWS上のリソースのモニタリングと
- オンプレミス環境のモニタリングを実施
- メトリクスおよびログを管理

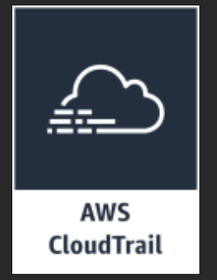


AWS Config & AWS Config Rules

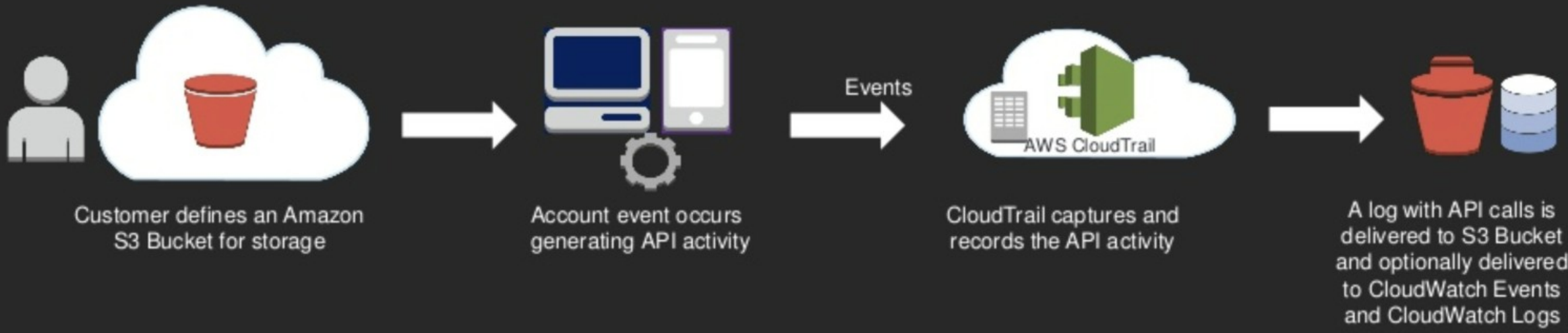
- 継続的記録、継続的アセスメントサービス
- AWSリソースの設定変更をトラッキング
- 自分で決めたポリシーにしがっていないとアラート発行



AWS CloudTrail



- アカウントの活動（操作ログ）を自動的かつ集中して管理
- API usage event でセキュリティ監査と運用上のトラブルシュートを実施



AWS Systems Manager

Resource groups

Patch manager

State manager

Run command

Automation

Maintenance window

Inventory

Parameter store

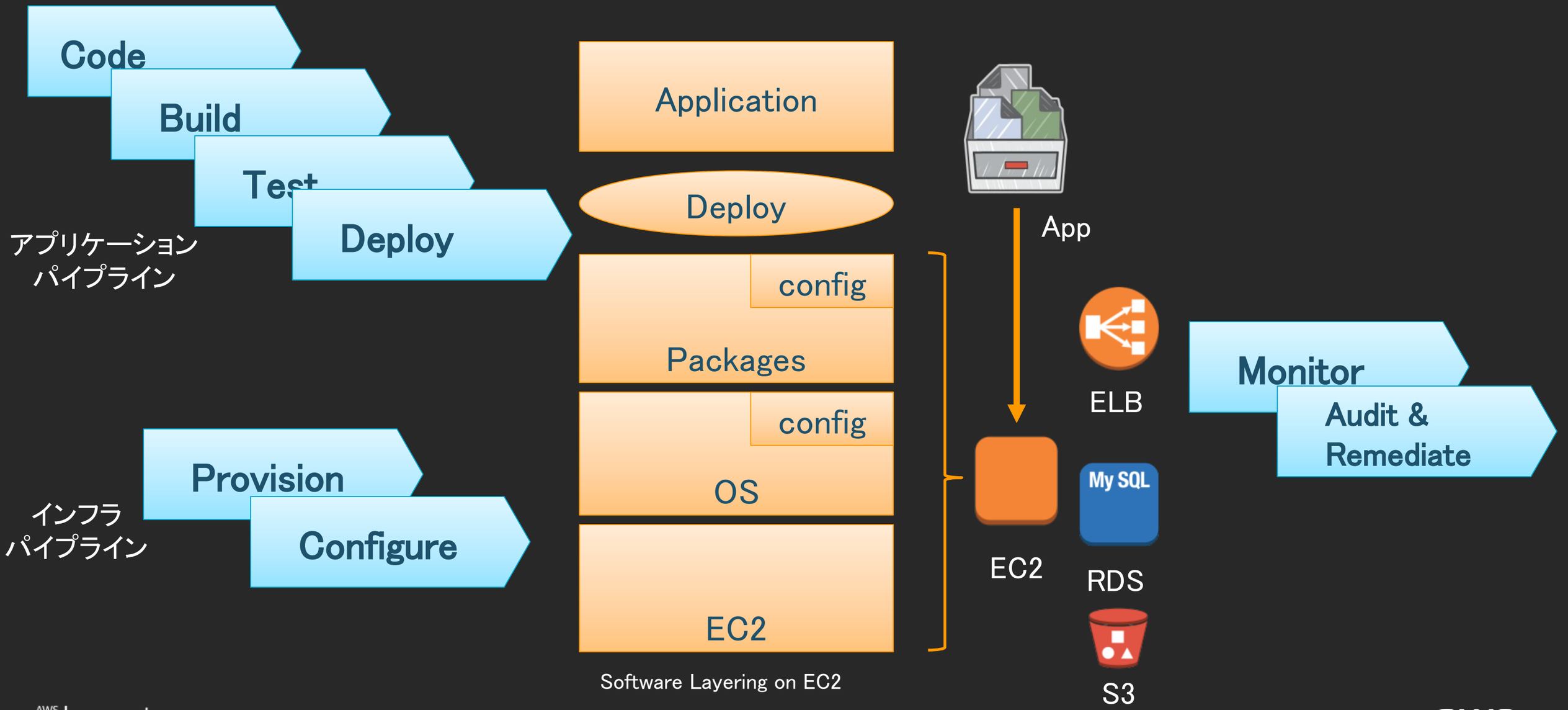
Session Manager

Distributor

AWS re:Invent

aws

Application & Infrastructure Pipelines



Application & Infrastructure Pipelines

Code

Build

Test

Deploy

アプリケーション
パイプライン

AWS CloudFormation
AWS Service Catalog

Provision

Configure

インフラ
パイプライン

AWS OpsWorks
AWS Systems Manager

Software Layering on EC2

App

EC2

ELB

RDS

S3

Amazon CloudWatch

Monitor

Audit &
Remediate

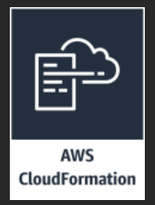
AWS CloudTrail
AWS Config

各サービスの使い所とアップデート

AWS re:Invent

aws

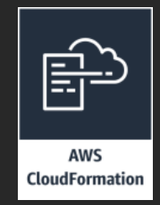
Provisioning – CloudFormation (CFn)



1. テンプレート開発　　各種ツールとプラグイン紹介
2. Cloud Development Kit　　一般の言語によるテンプレート開発
3. Drift Detection　　テンプレートとの差分検出
4. Macros　　テンプレート記述の拡充

2018/11/28 実施のBlackBelt オンラインセミナーもご覧ください

CFn – テンプレート開発のサポート

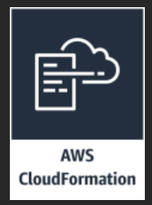


- **cfn-lint**
  - Validate AWS CloudFormation yaml/json templates against the AWS CloudFormation spec and additional checks
- **cfn_nag**
  - Look for patterns in AWS CloudFormation templates that may indicate insecure infrastructure.
- **Taskcat**
  - Catch problems that aren't obvious in a single template/stack
- **Cloud Development Kit**
  - Define cloud infrastructure in code and provision it through AWS CloudFormation
- **DSLs**
  - Troposphere/SparkleFormation/GoFormation

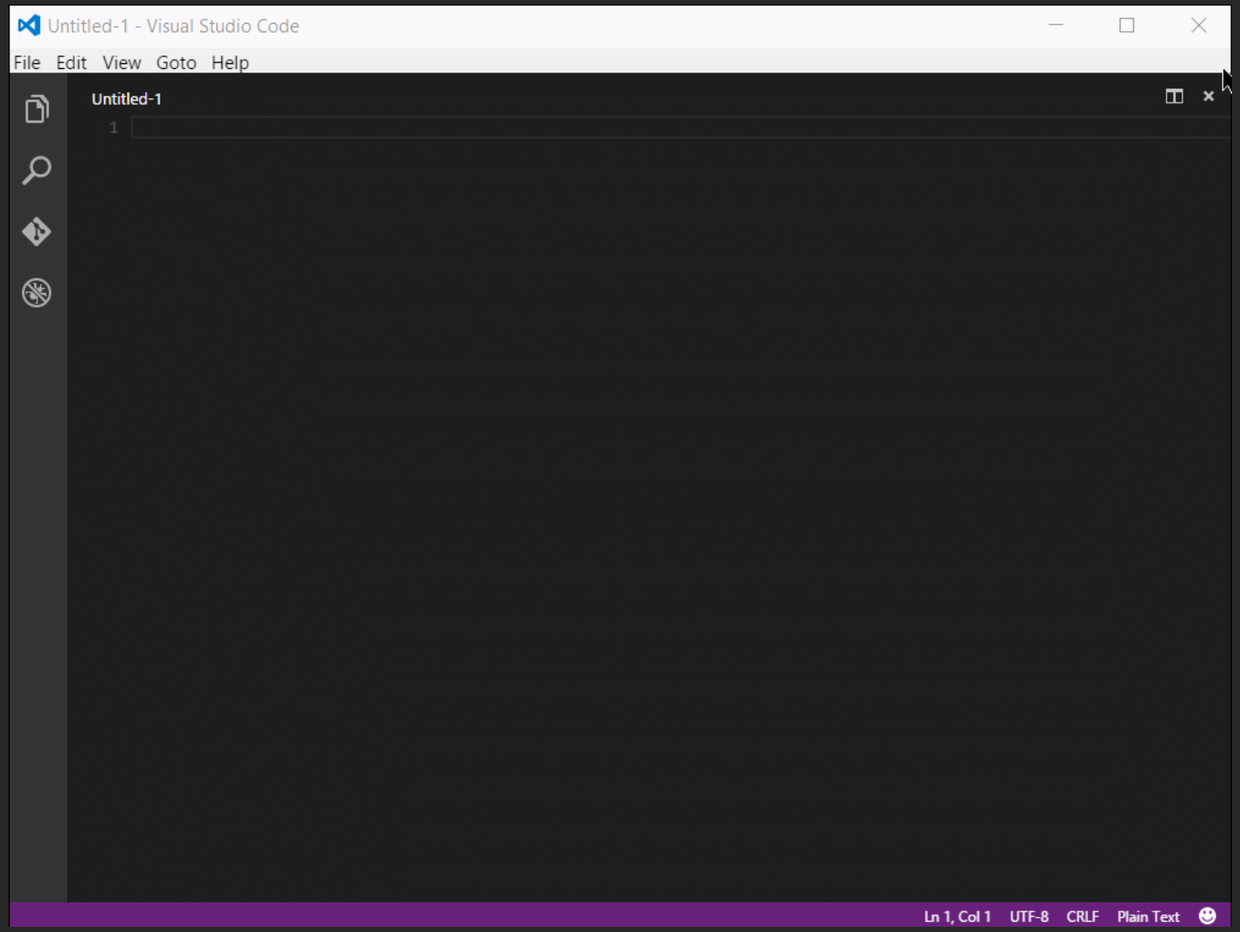
CFn – テンプレート開発のサポート

https://github.com/aws-samples/aws-cloudformation-advanced-reinvent-2018

CloudFormation support for Visual Studio Code

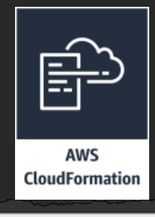


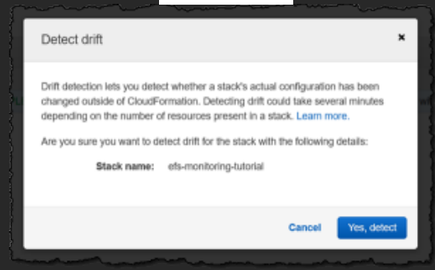
https://github.com/aws-scripting-guy/cform-VSCode

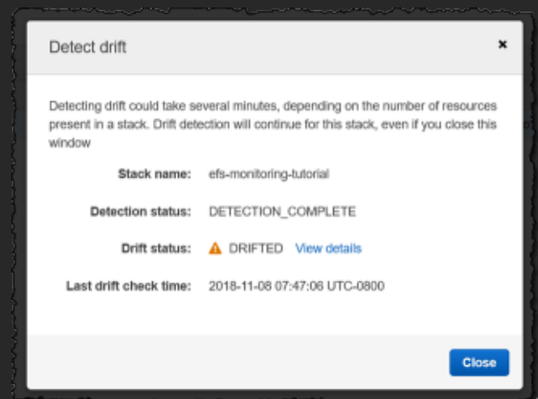


CloudFormationでスタック差分の検出機能をリリース

UPDATE





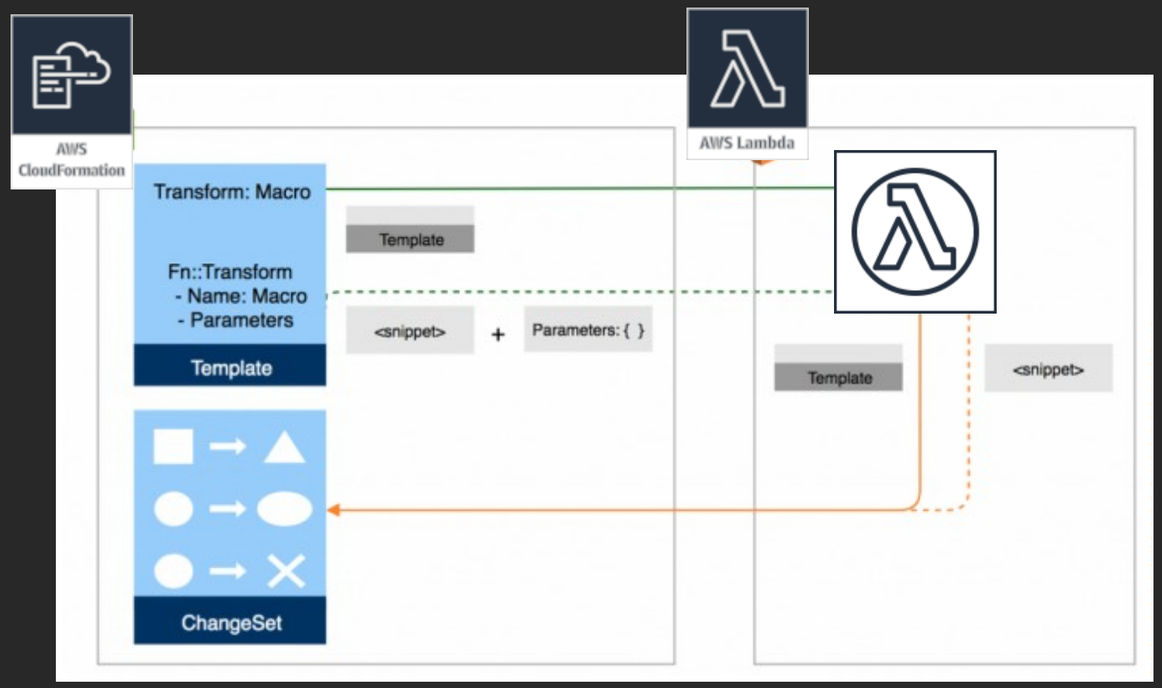


- CloudFormationで作成したリソース状態と、現状との差分(Drift)を検出できるようになった
- スタックをデプロイした後に手動で実施した変更作業によって発生した差をチェックできる
- 現時点ではEC2, Auto Scaling, ECS, ELB, Lambda, RDSなどでサポート。詳細は下記
  https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-stack-drift-resource-list.html
- AWS Configのマネージドルールもリリース。差分が発生したらすぐに検知できる

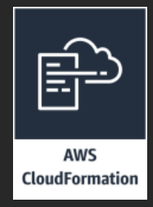
Macros

UPDATE

- テンプレートの標準的な機能では実現できない処理を、Lambda関数を呼び出す事で実現
- 検索や置換などの単純な操作からテンプレート全体の変換まで独自処理が可能
- Lambdaでリソースの作成や削除を行わない事を推奨



Macros



UPDATE

- Macros作成

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  Macro:
    Type: "AWS::CloudFormation::Macro"
    Properties:
      FunctionName: arn:aws:lambda:us-east-1:1234567:function:EchoFunction
      Name: EchoMacro
```

Macrosリソースタイプ

Lambda関数（arn）を指定

Macrosの名前

- Macros呼び出し

呼び出すMacros

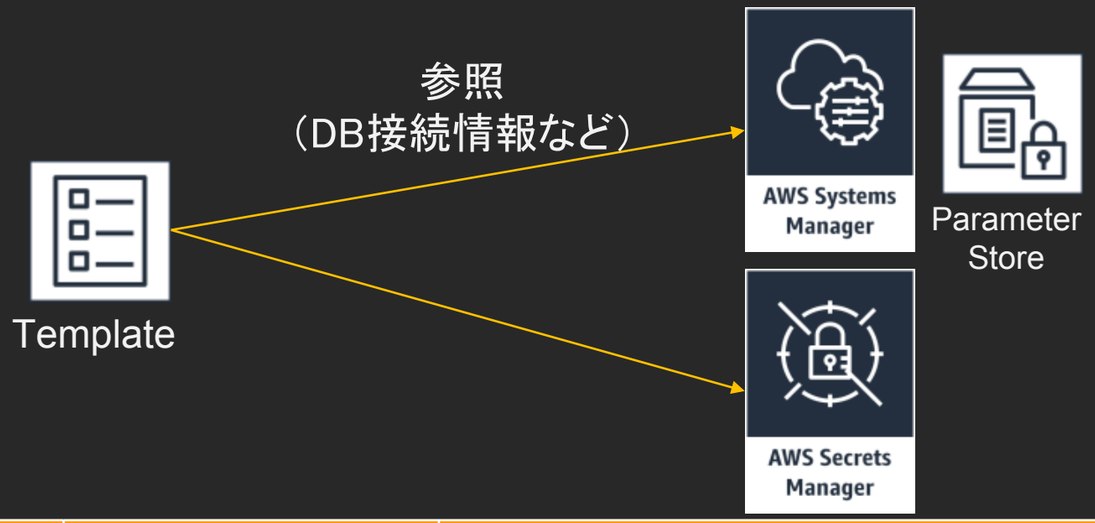
```
AWSTemplateFormatVersion: '2010-09-09'
Transform: [EchoMacro, 'AWS::Serverless-2016-10-31']
Resources:
  FancyTable:
    Type: AWS::Serverless::SimpleTable
```


39

Dynamic References

UPDATE

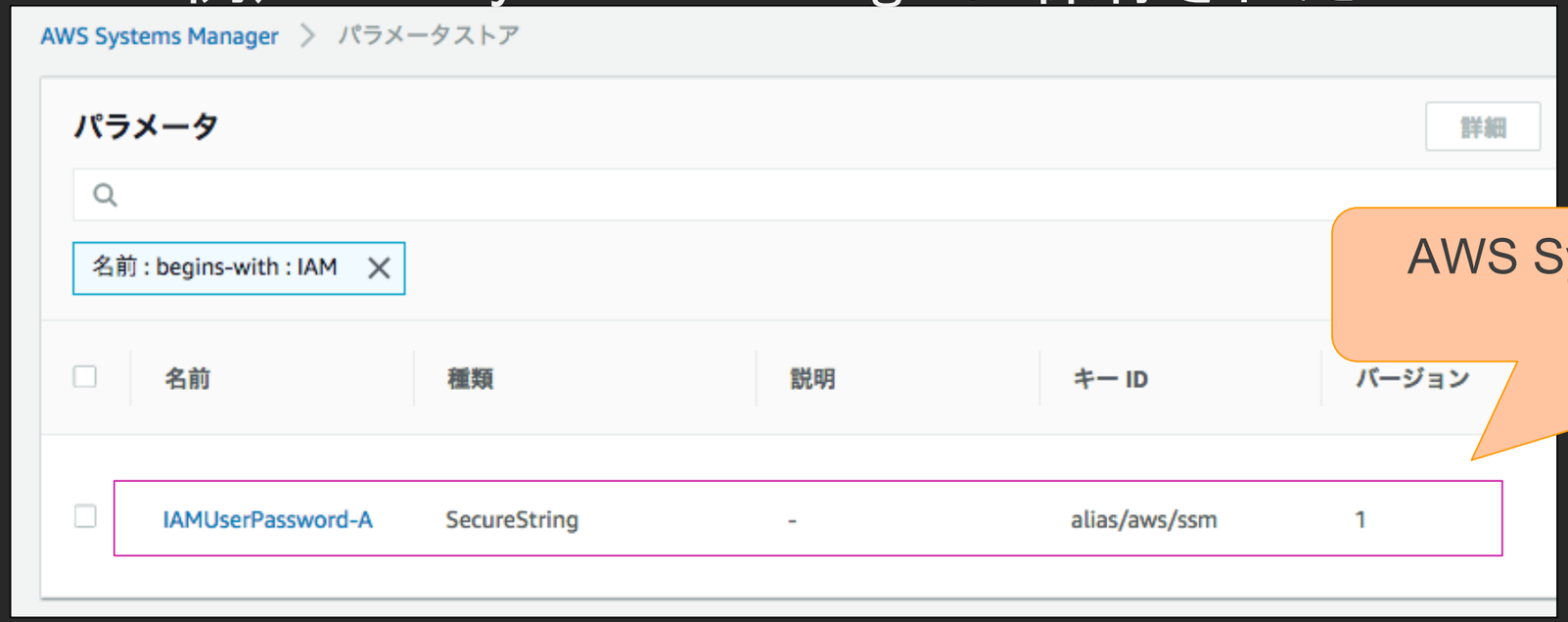
- 他のサービスに格納されたデータを動的に参照



サービス	テンプレート内の service-name	参照可能なデータ
AWS Systems Manager	ssm	パラメータストアに格納されているString/StringList（平文で保存されているデータ）
	ssm-secure	パラメータストアに格納されているSecureString（暗号化されて保存されているデータ）
AWS Secrets Manager	secretsmanager	保存されているすべてのシークレットまたは特定のシークレット

Dynamic References

- 例）AWS Systems Managerに保存されたSecureStringを参照



AWS Systems Managerパラメータストアに
IAMUserPassword-Aを保存

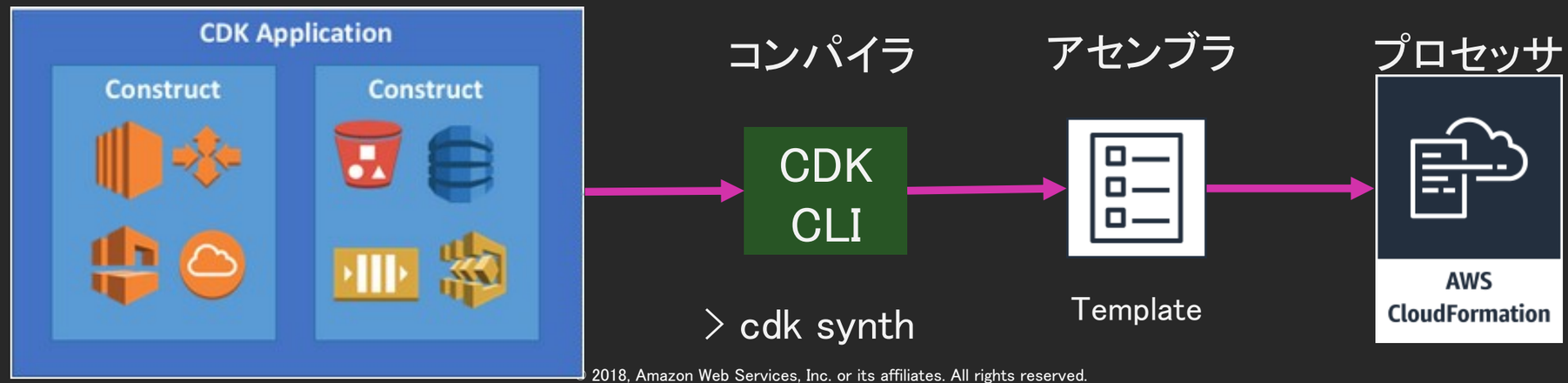
```
MyIAMUser:
    Type: AWS::IAM::User
    Properties:
      UserName: 'MyUserName'
      LoginProfile:
        Password: '{{resolve:ssm-secure:IAMUserPassword-A:1}}'
```

テンプレートから参照

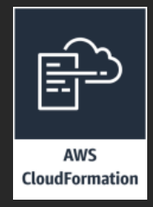
AWS CDK（AWS Cloud Development Kit）

Preview

- 開発者プレビュー
- ソフトウェア開発のフレームワーク
- インフラをJava等のコードで定義してAWS CloudFormationでプロビジョニング
- オブジェクト指向的に記述&ループ等が使用可能
- 推奨デフォルト値が設定済みのためコード量が少なく済む
- 現在、Java,JavaScript,TypeScript,.NET に対応



AWS CDK（AWS Cloud Development Kit）



Preview

- 例）S3バケット作成

```
const cdk = require('@aws-cdk/cdk');
const s3 = require('@aws-cdk/aws-s3');

class MyStack extends cdk.Stack {
    constructor(parent, id, props) {
        super(parent, id, props);

        new s3.Bucket(this, 'MyFirstBucket', {
            versioned: true
        });
    }
}
```

Remediation – Systems Manager (SSM)

New

1. Session Manager　　SSH/RDPやポート開放なくシェルアクセス
2. Distributer　　独自パッケージの配布

Updates

1. ResourceGroup リニューアルして77のリソースに対応、連携強化
2. Automation　　AWS API呼び出し,条件分岐,マルチアカウント対応
3. RunCommand　Windows DSC対応
4. Maintenance Window　スケジュール管理画面の強化
5. Inventory　　マルチアカウント対応とAthenaと連携した検索
6. Patch Manager　Custom Patch Approval

AWS Systems Manager によるサーバ運用

マネージドインスタンス

AWS cloud

tag: yyy

tag: xxx

data center

tag: xxx

スケジューリング

定期運用

メンテナンスウィンドウ

非定期・障害時運用

マネジメントコンソール

AWS CLI

Session Manager

オペレーション

Run Command

Automation

Distributer

Lambdaファンクション

Step Functions

操作記録・通知

SNS

CloudWatchLogs

S3

IAM Role

SSM Agent

Inspector Agent

CloudWach Agent

サーバ運用

Inspector
セキュリティ評価

CloudWatch
メトリクス&ログ

構成検証・監査

ステートマネージャー

インベントリ

Patch Manager

構成情報・操作手順

ドキュメント

パラメータストア

記録・通知・対応・分析

Athena&Quicksight

S3

AWS Config

CloudWatch Event

パラメータストアの参照

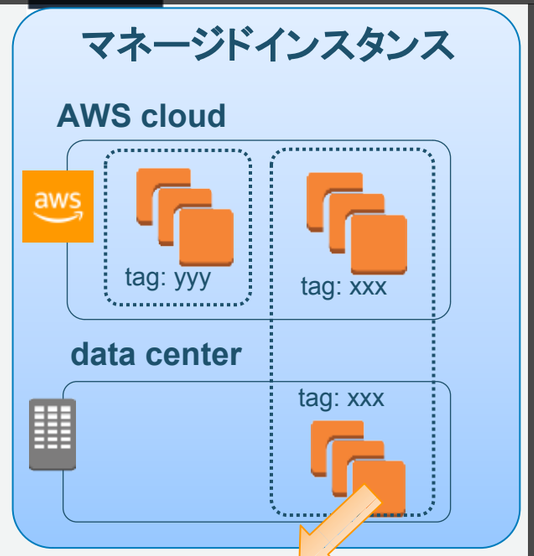
CFn

CLI

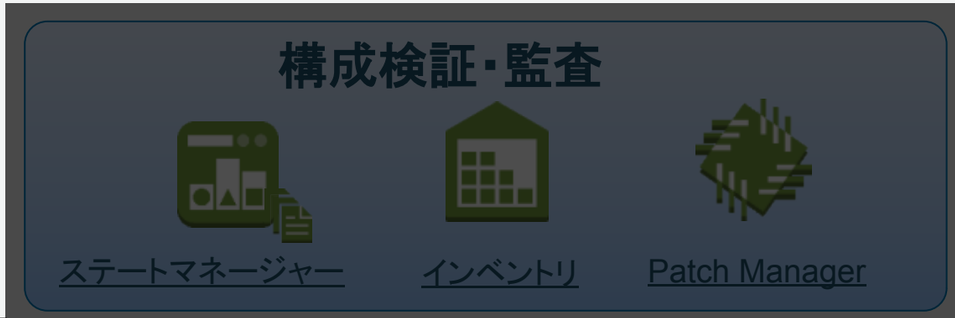
Lambda

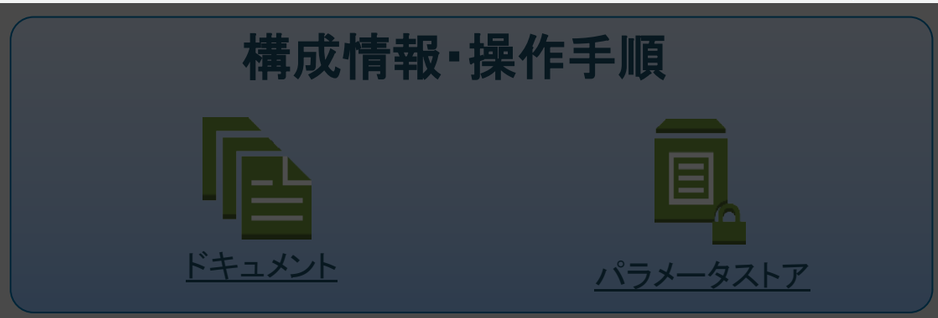
ECS

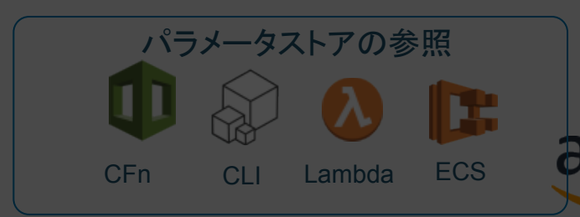
AWS Systems Manager によるサーバ運用



サーバ運用







AWS Systems Manager によるサーバ運用

Run Command	OS上の管理作業を事前定義済みのドキュメントを使用して安全に実施する SSHやRDPを使用せず多数のサーバを一括操作
Automation	AWS上の管理作業を事前定義済みのドキュメントを使用して安全に実施する パッチを適用してAMIを作成する、インスタンススペックを変更するなど
Distributer	独自のソフトウェアパッケージを定義し配布する
Session Manager	通信ポートを解放せずにサーバへのシェルアクセスを行う インタラクティブあるいは非定型の操作に使用

AWS Systems Manager によるサーバ運用

マネージドインスタンス

AWS cloud

スケジューリング

定期運用

オペレーション

Run Command

操作記録・通知

ステートマネージャー	あるべきOSの設定値を定義し維持する ポリシーに準拠したファイアウォール設定やアンチマルウェアツールの設定
インベントリ	オペレーティングシステム (OS)、アプリケーション、インスタンスのメタデータを収集する
Patch Manager	OSパッチの選択と適用を自動的に実施する

IAM Role

SSM Agent

Inspector Agent

CloudWach Agent

サーバ運用

Inspector
セキュリティ評価

CloudWatch
メトリクス&ログ

構成検証・監査

ステートマネージャー

インベントリ

Patch Manager

構成情報・操作手順

ドキュメント

パラメータストア

記録・通知・対応・分析

Athena&Quicksight

S3

AWS Config

CloudWatch Event

パラメータストアの参照

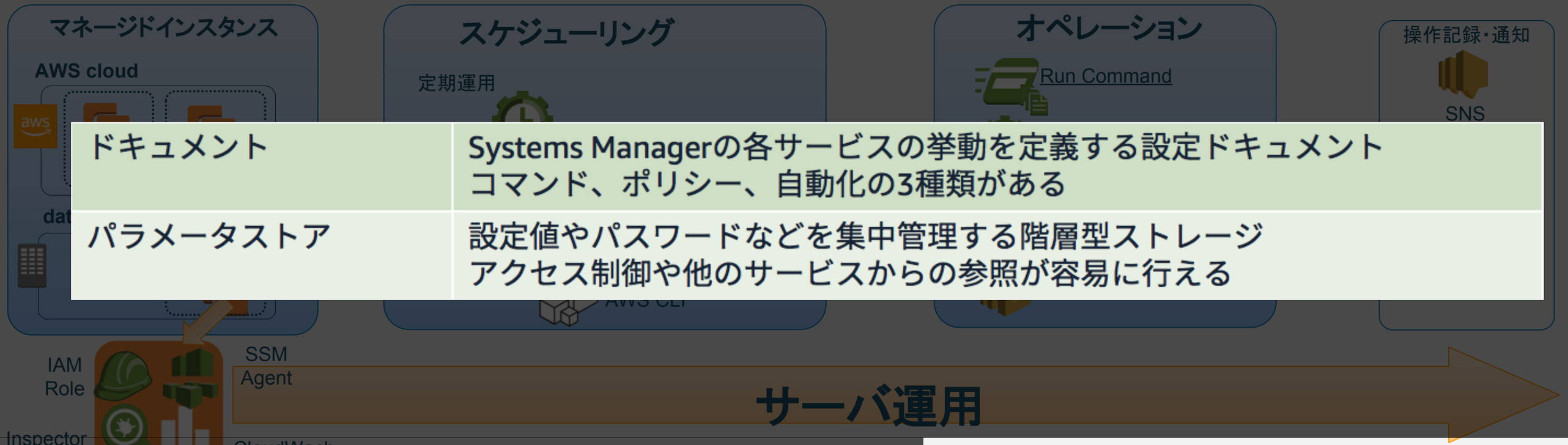
CFn

CLI

Lambda

ECS

AWS Systems Manager によるサーバ運用



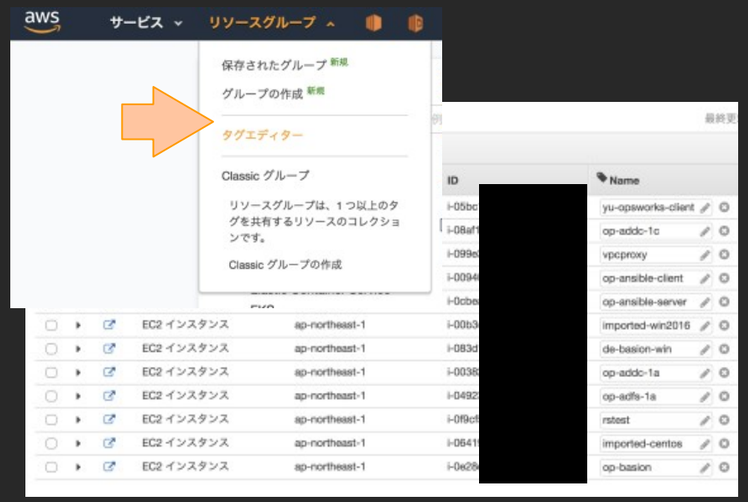
ドキュメント	Systems Managerの各サービスの挙動を定義する設定ドキュメント コマンド、ポリシー、自動化の3種類がある
パラメータストア	設定値やパスワードなどを集中管理する階層型ストレージ アクセス制御や他のサービスからの参照が容易に行える

AWS Systems Manager サービス群

サービス名	概要
メンテナンスウィンドウ	定期的に行う管理作業やサービス停止の実施時間帯を定義する
Run Command	OS上の管理作業を事前定義済みのドキュメントを使用して安全に実施する SSHやRDPを使用せず多数のサーバを一括操作
Automation	AWS上の管理作業を事前定義済みのドキュメントを使用して安全に実施する パッチを適用してAMIを作成する、インスタンススペックを変更するなど
Distributer	独自のソフトウェアパッケージを定義し配布する
Session Manager	通信ポートを解放せずにサーバへのシェルアクセスを行う インタラクティブあるいは非定型の操作に使用
ステートマネージャー	あるべきOSの設定値を定義し維持する ポリシーに準拠したファイアウォール設定やアンチマルウェアツールの設定
インベントリ	OS、アプリケーション、インスタンスのメタデータを収集する
Patch Manager	OSパッチの選択と適用を自動的に実施する
ドキュメント	Systems Managerの各サービスの挙動を定義する設定ドキュメント コマンド、ポリシー、自動化の3種類がある
パラメータストア	設定値やパスワードなどを集中管理する階層型ストレージ アクセス制御や他のサービスからの参照が容易に行える

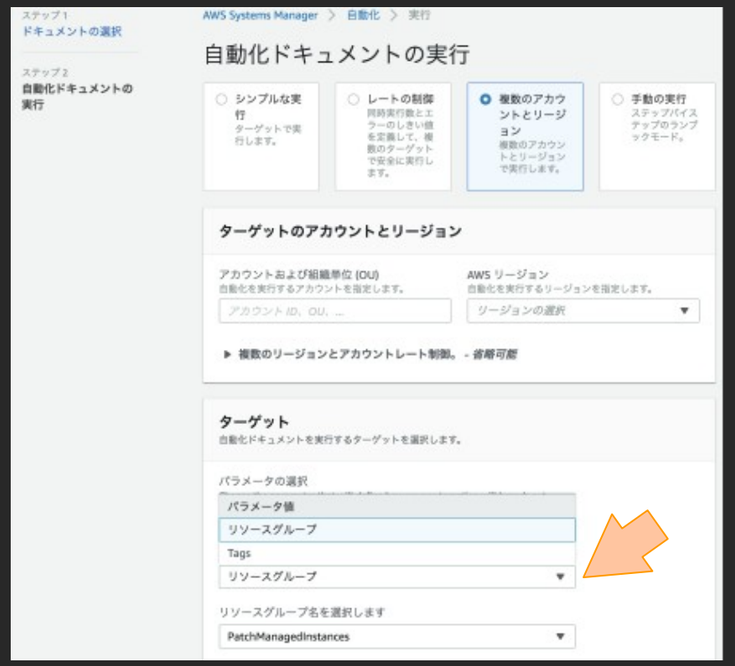
Resource Group

- AWS上のリソースをグループ化して一括管理
  - 77のリソースに対応
  - タグエディタによる一括タグ付け
- Management Tools と統合
  - Systems Manager
    - Automation, RunCommand, etc..
  - CloudWatch Dashboard

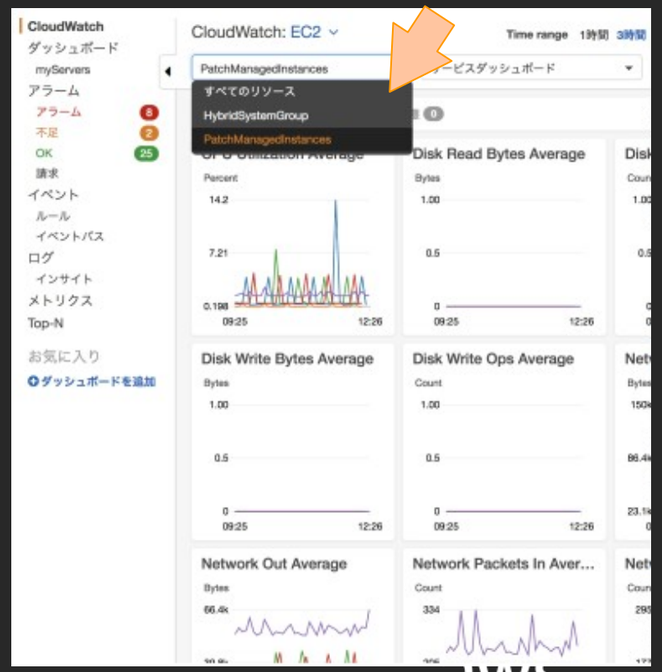


タグエディタ

Resource Group 作成



SSM Automationなど



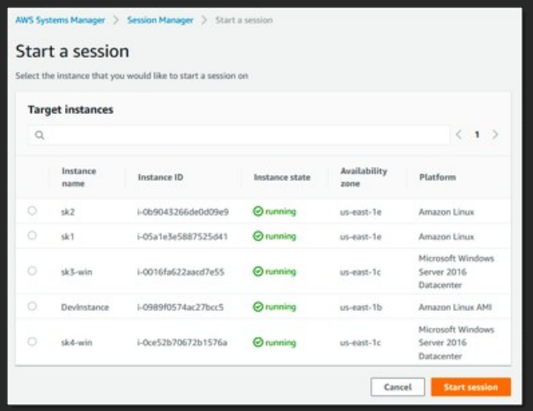
CloudWatch

AWS Systems Manager の Session Manager でマネジメントコンソールからOSへシェルアクセス

UPDATE

AWS Systems Manager

- 通信ポートを開放せずにサーバへのシェルアクセスが可能
  - マネジメントコンソールおよびCLIからアクセス
  - Linux は bash、Windows は PowerShellが利用可能
  - サーバのログイン情報（キーペアおよびID・パスワード）が不要



- その他の機能
  - IAMユーザがアクセス可能なインスタンスをIAM Policyで指定可能
  - 操作ログを CloudWatch Logs や S3 に保存。暗号化も可能。

- 前提：SSM Agentの導入とエンドポイントへの通信が必要
  - 対象サーバに最新の Systems Manager Agentが導入されていること
  - 適切な権限(AmazonEC2RoleforSSMなど)を持つ IAM Role が EC2 に付与されていること
  - インターネットへのアウトバウンドアクセスまたは、
    VPC内に作成したSystems Manager PrivateLink エンドポイントへの通信が行えること

AWS re:Invent

aws

Automation で AWSの「操作手順」をYAML/JSONで記述

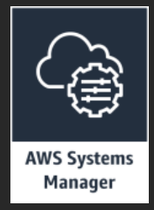
- 全てのAWS APIを呼び出せる executeAwsApi アクションが追加
  - 従来は事前定義済みのEC2に関する操作のみが指定可能
- Automation Documents (YAML or JSON)でAWS環境を操作するための「手続き」を記述できる
  - シェルやLambdaに比べ自由度は下がるが一定の書式やルールに従うことが可能
  - 簡単な処理フローも記述可能
  - AWS環境とサーバ上の操作を組み合わせたリカバリ処理などを定義できる

```yaml
description: automate instance deployment
schemaVersion: '0.3'
mainSteps:
- name: getGoldenImageId
  action: aws:executeAwsApi
  inputs:
    Service: ssm
    Api: GetParameter
    Name: GoldenImageId
  outputs:
  - Name: Value
    Selector: "$.Parameter.Value"
    Type: String

- name: launch_ec2_instance
  action: aws:executeAwsApi
  inputs:
    Service: ec2
```

AWS Systems Manager の Automation でパラメータによる条件分岐が可能に

UPDATE



- Automation ドキュメント内で
  aws:branch を使うことで
  パラメータに応じた条件分岐が可能

  従来は前のステップの成功/失敗に
  応じた分岐のみが可能

- Systems Managerが利用可能な全てのリージ

EC2のOSタイプを取得して条件分岐する例

```
mainSteps:
- Name: GetInstance
  action: aws:executeAwsApi
  inputs:
    Service: ssm
    Api: DescribeInstanceInformation
  outputs:
  - Name: myInstance
    Selector: "$.InstanceInformationList[0].InstanceId"
    Type: String
  - Name: platform
    Selector: "$.InstanceInformationList[0].PlatformType"
    Type: String
- name: ChooseOSforCommand
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
    Default:
      Sleep
```

https://aws.amazon.com/jp/about-aws/whats-new/2018/09/Conditional-Branching-Now-Supported-in-AWS-Systems-Manager-Automation/
https://docs.aws.amazon.com/systems-manager/latest/userguide/automation-branchdocs.html

様々なサービスをプログラミングレスでつなぎ合わせる AWS Step Functions API Connectorsを発表

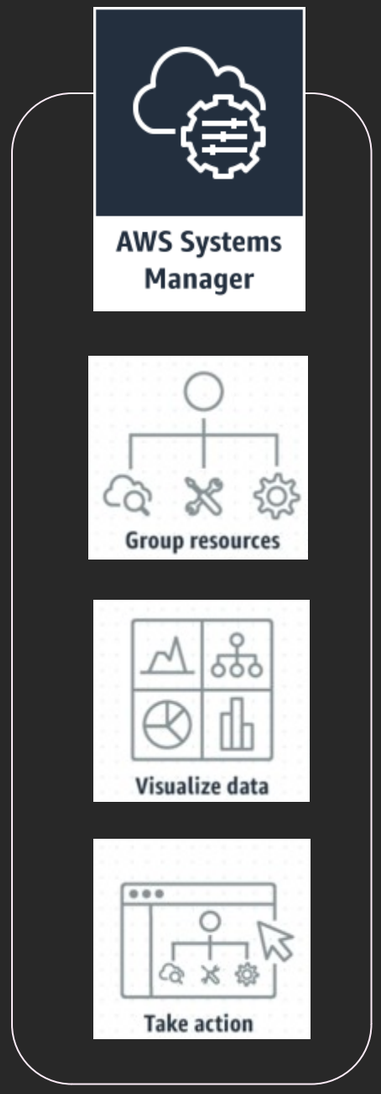
UPDATE

- Step Functionsのステートマシンから他のAWSサービス群に対して直接操作できるようになり、Lambdaを介在させる必要がなくなった
  - ✓ DynamoDB: 既存のテーブルからitemの取り出し、新規itemの追加
  - ✓ AWS Batch: バッチジョブの開始と完了待機
  - ✓ Amazon ECS/Fargate: ECSまたはFargateのタスクを実行する
  - ✓ Amazon SNS: SNSトピックにメッセージをパブリッシュする
  - ✓ Amazon SQS: キューにメッセージをプッシュする
  - ✓ AWS Glue: ジョブを開始する
  - ✓ Amazon SageMaker: 学習ジョブ、変換ジョブを開始する

AWS Systems Manager パッケージ管理機能 (Distributer) の追加

UPDATE

- 独自のソフトウェアパッケージを作成してサーバへ配布
  - ✓ ソフトウェアと install / uninstall スクリプトを zip 化してパッケージ登録
  - ✓ 配布対象のサーバ群を定義しアクセスコントロール
  - ✓ RunCommand (1回) や StateManager (定期) を使って配布



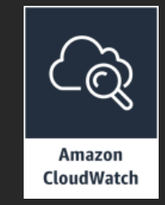
AWS Systems Manager
Inventory および Automation のマルチアカウント対応

UPDATE

AWS Systems Manager

- Inventory
  - OS上のソフトウェア構成を収集しAthenaでアカウントやリージョンをまたいで検索
- Automation
  - キュメントに定義した一連のAPI操作をアカウントやリージョンをまたいだAWSリソースに対して実行

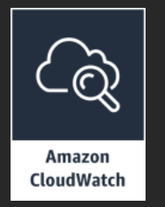
Monitoring – CloudWatch

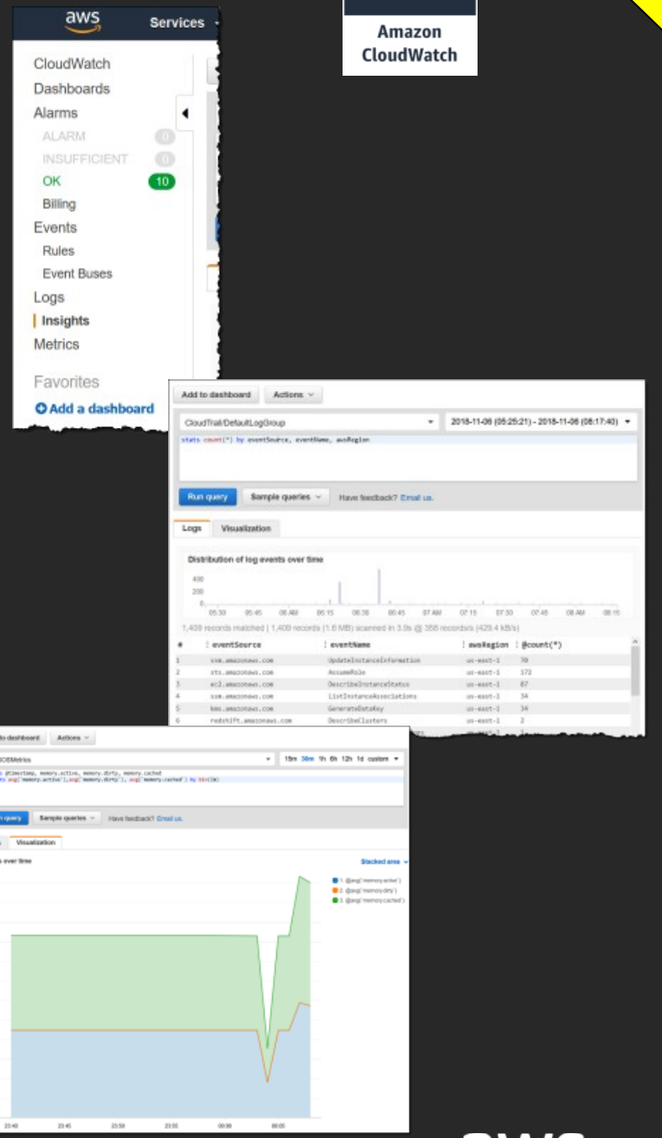


1. CloudWatch Logs Insight　　収集ログをクエリでリアルタイム分析
2. Automated Dashboard　　集約ダッシュボードの自動生成
3. Metric Math Alarm　　計算値に基づくアラームの発行
4. CloudWatch Snapshot Graphs　　グラフをマネコン以外で利用

インタラクティブなログ分析を実現する Amazon CloudWatch Logs Insightsを発表

NEW

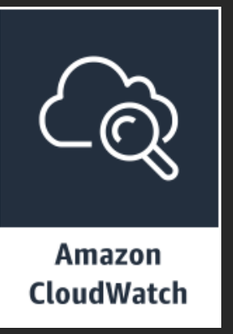




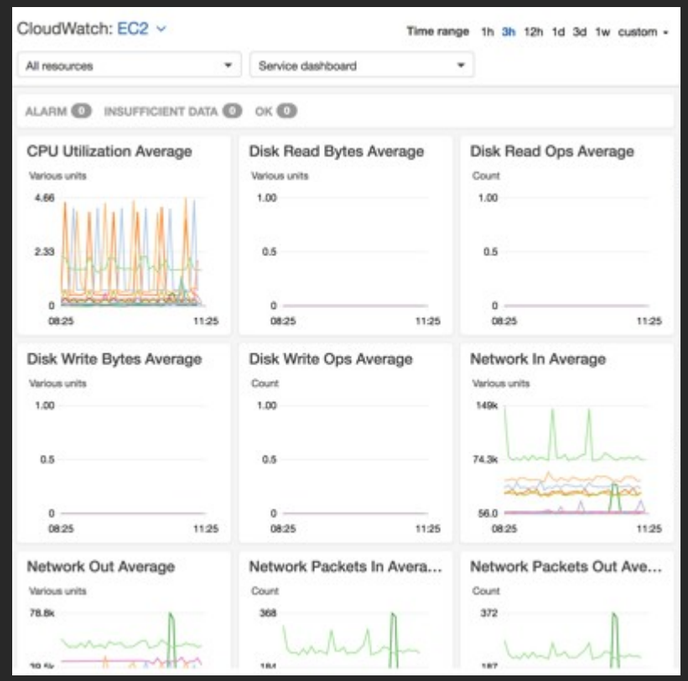
- AWSが生成する各種ログや、様々なサーバアプリケーションのログの分析や可視化を実現するフルマネージドサービス
- CloudWatch Logsで収集したログに対して、クエリ言語を利用してソースや時間でのフィルタリング・ビジュアライズを実行できる
- クエリの結果をDashboardに追加することも
- 東京を含む各リージョンで利用可能。クエリによりスキャンされたログデータ量に応じた課金があり、$0.0076/GB(東京)となる

UPDATE

CloudWatch が Automatic Dashboard と 計算値に基づくアラームに対応

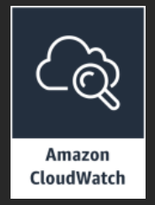


- AWSが推奨するベストプラクティスに基づいた集約ダッシュボードを自動生成
  - ✓ 各主要サービスごとあるいはサービスをまたいだダッシュボード
  - ✓ リソースグループを使用してシステムごとにフィルタリング
- 複数のメトリックの計算結果に基づきアラーム発行
  - ✓ 計算値メトリックのグラフはすでに利用可能だったがアラーム発行も計算値に基づいて可能に
  - ✓ +, -, /, * や Sum, Average, Min, Max などが利用可能

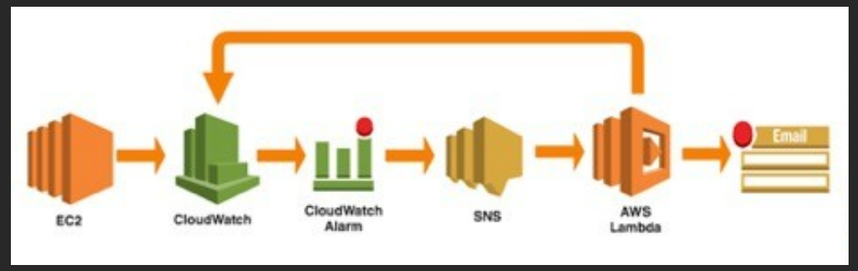


CloudWatch のグラフを外部から取得可能に

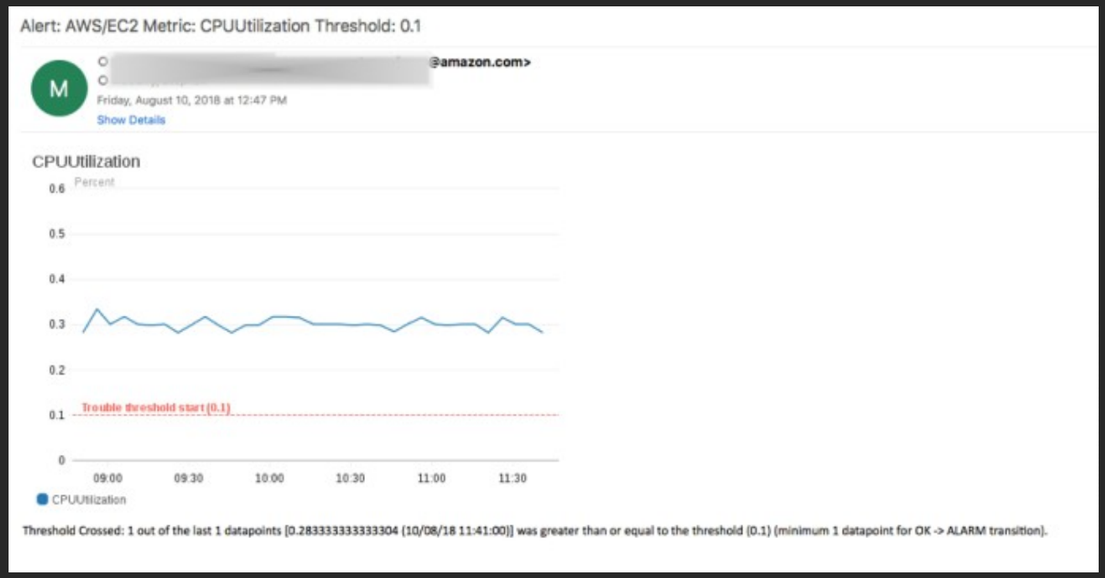
UPDATE



- CloudWatchで作成したグラフの画像イメージをAPIで取得可能



- マネジメントコンソール外でグラフを参照可能
  - CloudWatch Alermと組み合わせてSlackやメールに異常時のグラフを添付する
  - プロジェクトのポータルサイトにCloudWatchのグラフを表示する



Audit – CloudTrail & AWS Config & Config Rules

1. Config Rules – Detect Drift for CFn CFnテンプレートとの差異を検出
2. AWS Config Multi-Account Aggregation　マルチアカウント情報集約

マルチアカウント関連アップデート

AWS re:Invent

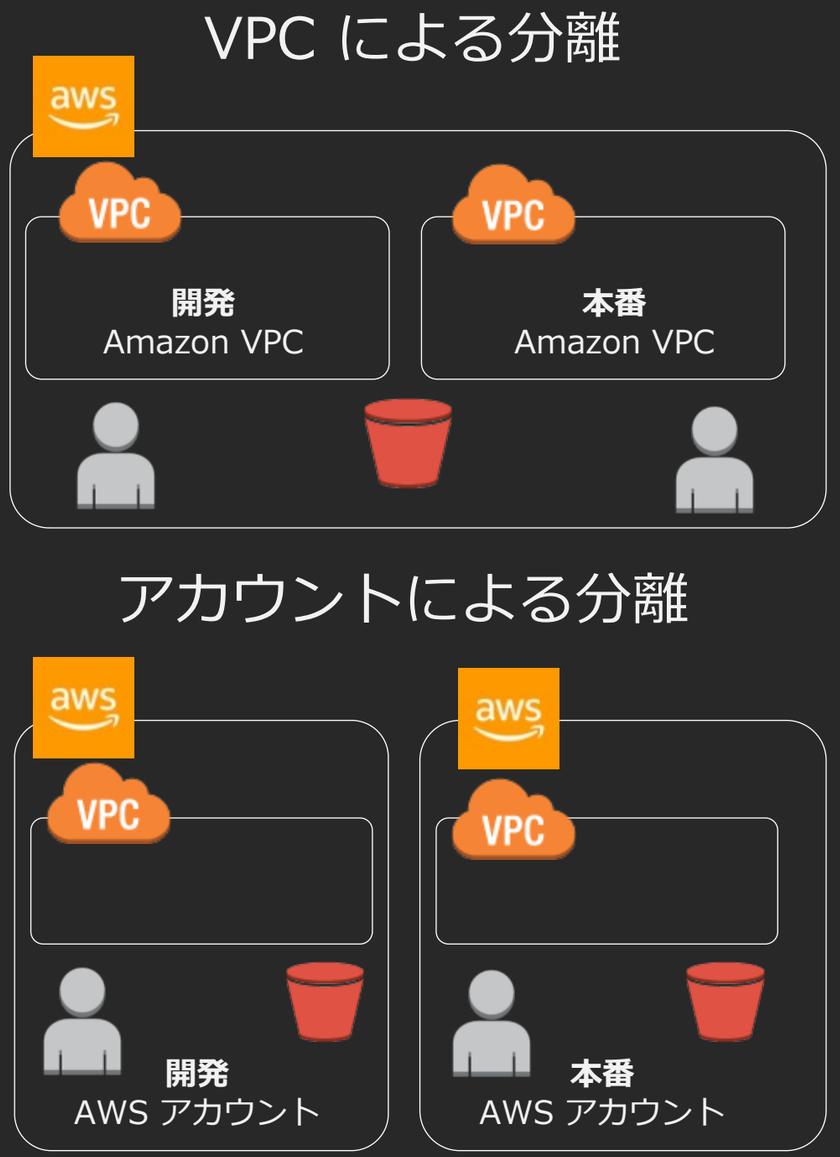
aws

VPCとアカウントの分離レベルの違い

VPCによる分離
- VPC間通信はデフォルトNG*1
- APIアクセスは共有(IAMで制御)

アカウントによる分離
- VPC間通信はデフォルトNG*1
- API相互アクセスはデフォルトNG*2

*1 - VPC Peeringを結ぶことで相互に通信可能
*2 - IAMのクロスアカウントアクセスで相互にアクセス可能



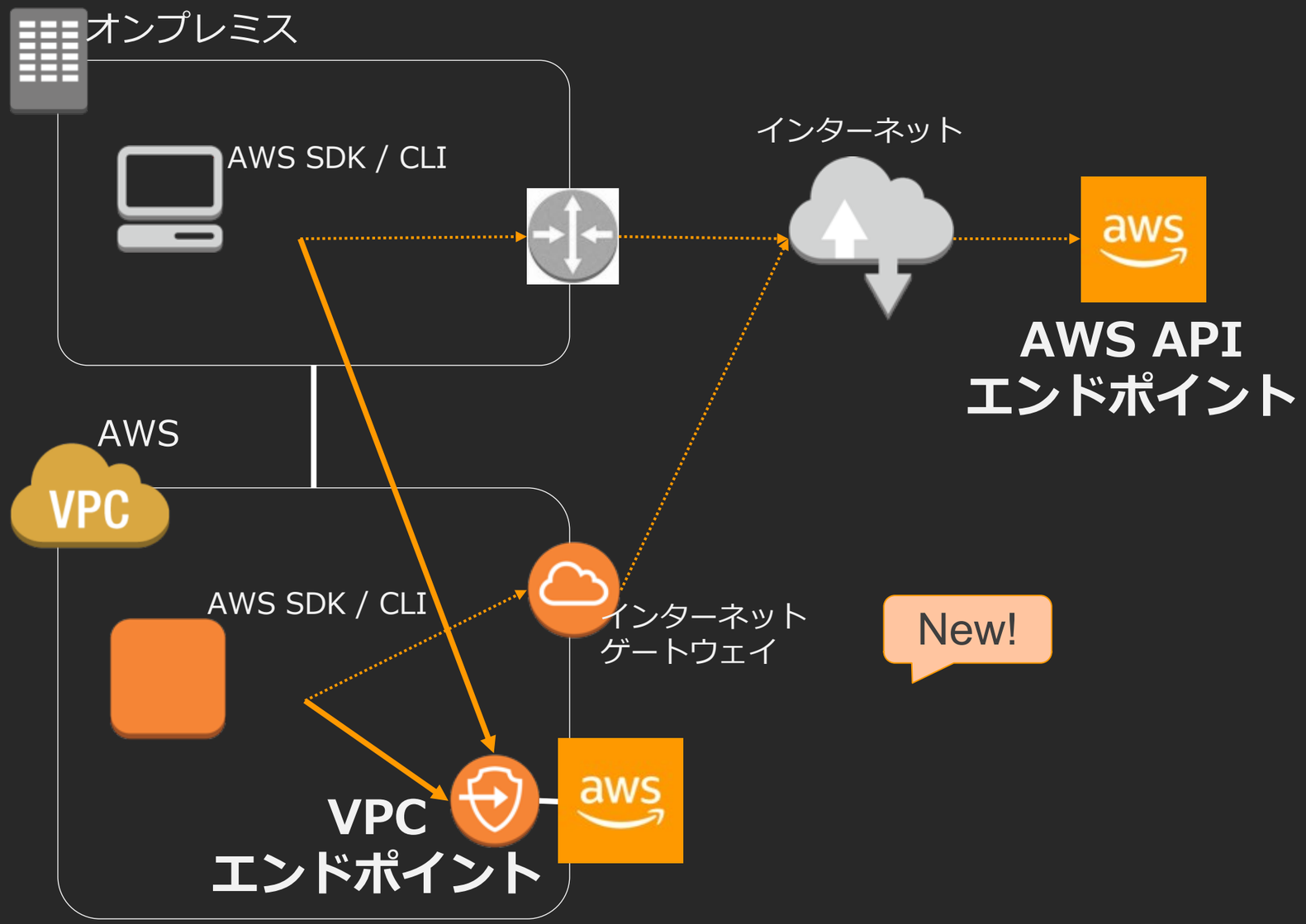
Private Link によるプライベートネットワーク経由APIアクセス

通常のAPIアクセスは
インターネット経由

VPCエンドポイントで
APIに対してプライベート
アクセスが可能

- 専用線 (Direct Connect)
- VPN

New!



マルチアカウント関連アップデート

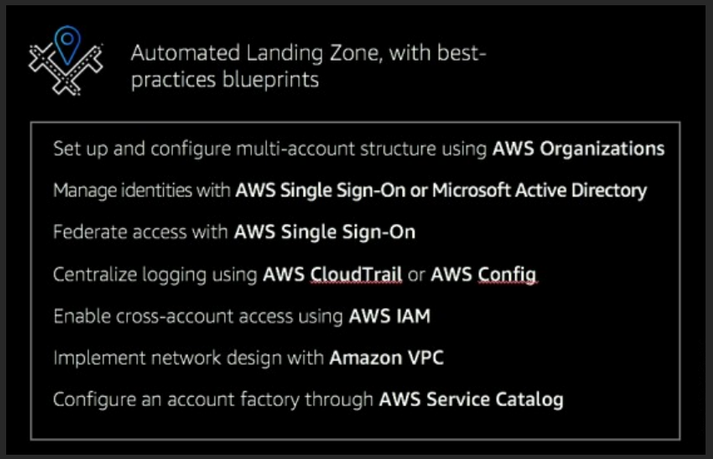
アカウントレベルの払い出し管理がより容易に

1. **アカウント作成　　– Control Tower**
2. **リソース共有　　– Resource Access Manager, Shared VPC**
3. **アクセス可能 – Transit GW, EFS**
4. Provisioning　　– CFn StackSets
5. Monitoring　　– CloudWatch Events Bus
6. Audit　　– Security Hub, AWS Config, SSM Inventory
7. Remediation – SSM Automation

マルチアカウント環境におけるガバナンスを強化する AWS Control Towerを発表

Preview

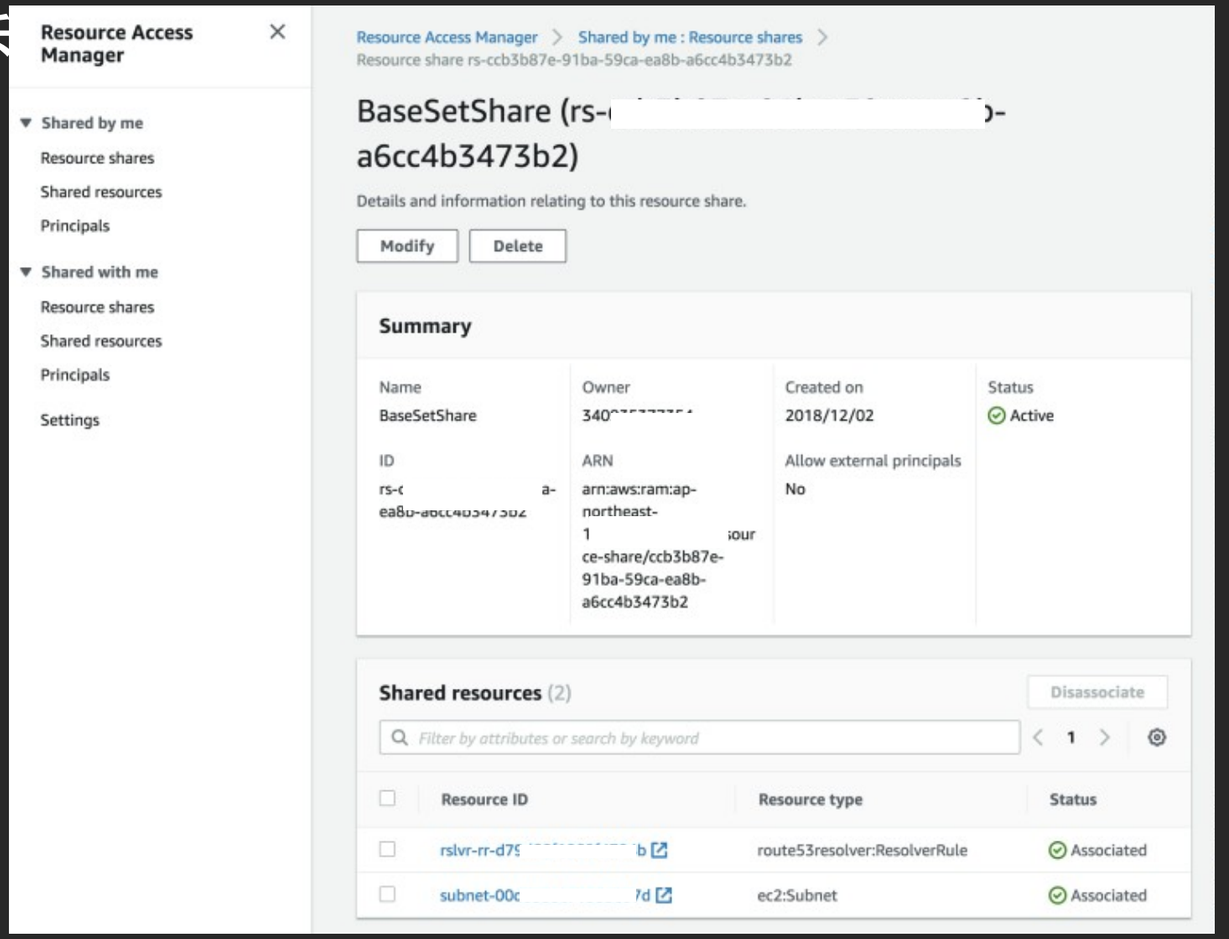
- 複数のアカウントに組織として必要とするセキュリティやコンプライアンスを確保し、ベストプラクティスに沿ったシステムを展開することを容易に
- システムがルールに沿っているかをハイレベルな視点でチェックし、必須または推奨のレベルで通知する機能を備えており、準拠状況をモニタできる
- 現状を把握するためのダッシュボードが備わっており、準拠すべきルールにどの程度従えているかを視覚的に把握する機能も
- Control Towerは無料で利用できる



クロスアカウントでのリソース共有が可能に AWS Resource Access Manager を発表

NEW

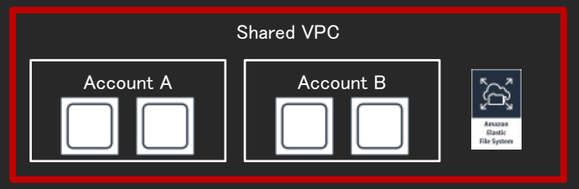
- アカウント間で同じリソースを共有して利用することが可能に
- 現時点の対象リソース
  - VPC サブネット
  - R53 リゾルバルール
  - License Manager - License Configuration
- 自分のリソースを Organization、Organization Unit（OU）、AWSアカウントで共有（再共有はNG）
- すべての商用リージョンで利用可能

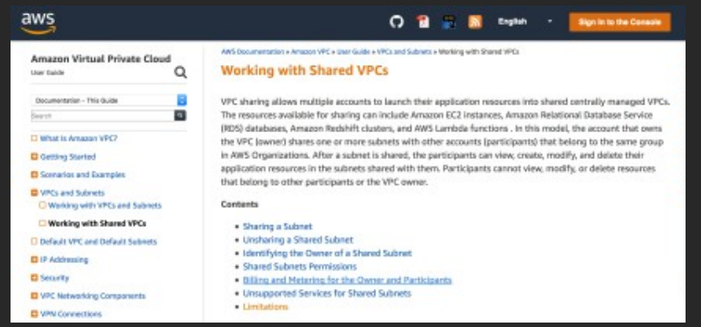


複数のAWSアカウントで1つのVPCを共有できる Shared VPCを発表

NEW

- これまでは複数のVPCとピアリングによって構成する必要があったが、単一のVPCで実現可能になった
- 設計上1VPCに集約することが必要なケースでも、複数のチームに対して個別のAWSアカウントを割り当てて権限分割やコストの明確な分離を行える
- VPCのオーナーはVPCレベルの要素(RouteTableやSubnetなど)を管理。利用者はSecurity Groupや自分が配置したリソースに責任を持つ
- 共有するアカウントは同一のAWS Organizationグループにある必要がある



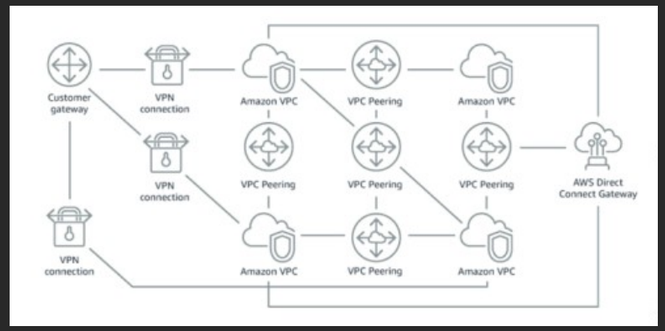


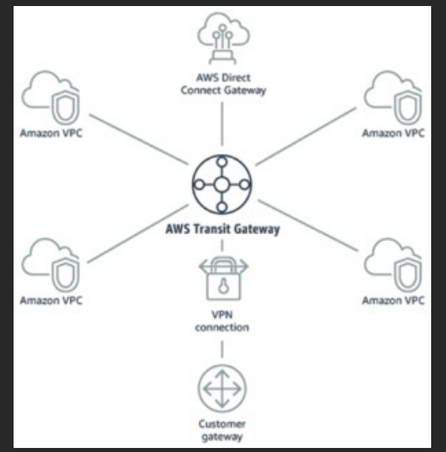
https://docs.aws.amazon.com/vpc/latest/userguide/vpc-sharing.html

様々な拠点や複数のVPC間のネットワークを柔軟に構築する AWS Transit Gatewayを発表

NEW

- 自分でソフトウェアルータを管理することなく、複数のVPCや拠点間を相互に接続・制御する
- CloudWatchによるメトリクス収集やVPC Flow Logs出力をサポート。SG/NACLによる通信制御も可能
- Transit Gatewayは冗長化がなされており、最大50Gbpsのバーストトラフィックまで処理できる
- 現時点ではDirect ConnectをTGWにアタッチすることはできない。現在鋭意開発中
- 東京リージョンは現在準備中。課金体系は時間課金と処理データ量課金の2つの軸となる

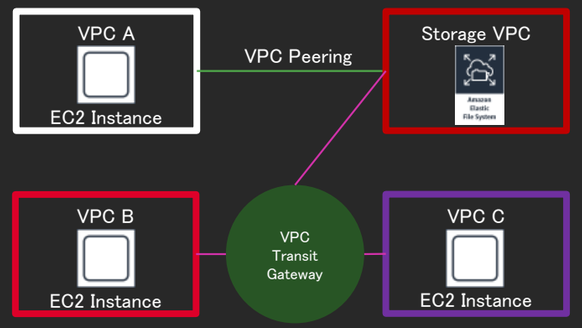


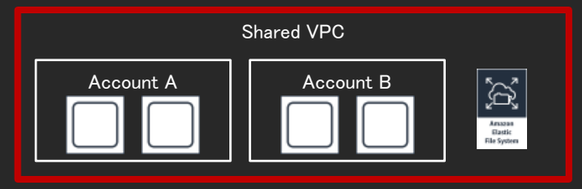


EFSが複数のアカウントやVPCからアクセス可能に

UPDATE

- EFSによるファイルシステムが、異なるアカウントや異なるVPCからアクセスできるようになり、1つのファイルシステムを共同利用しやすくなった
- VPC Peering/Transit Gatewayを経由してマウント可能。Shared VPCでも利用できる
- 細かな制約があるため、ドキュメントの確認を
  https://docs.aws.amazon.com/efs/latest/ug/manage-fs-access-vpc-peering.html
- 追加費用なしで利用可能。ピアリングやTGWの利用料・データ通信料は発生する





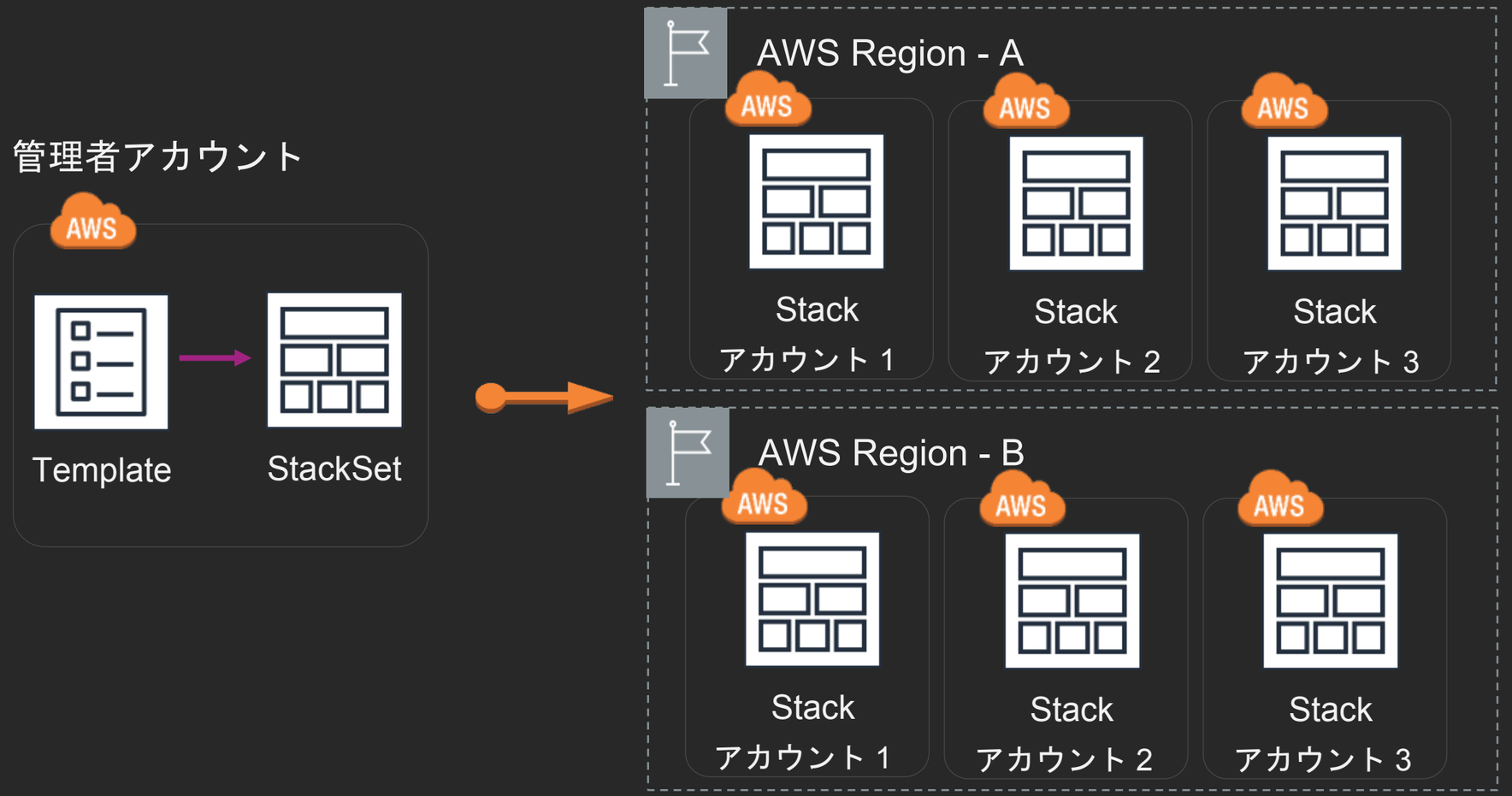
マルチアカウント関連アップデート

アカウントレベルの払い出し管理がより容易に

1. アカウント作成 – Control Tower
2. リソース共有 – Resource Access Manager, Shared VPC
3. アクセス可能 – Transit GW, EFS
4. **Provisioning – CFn StackSets**
5. **Monitoring – CloudWatch Events Bus**
6. **Audit – Security Hub, AWS Config, SSM Inventory**
7. **Remediation – SSM Automation**

CFn StackSets

- 一つのテンプレートを複数のAWSアカウント及び複数のリージョンに展開可能



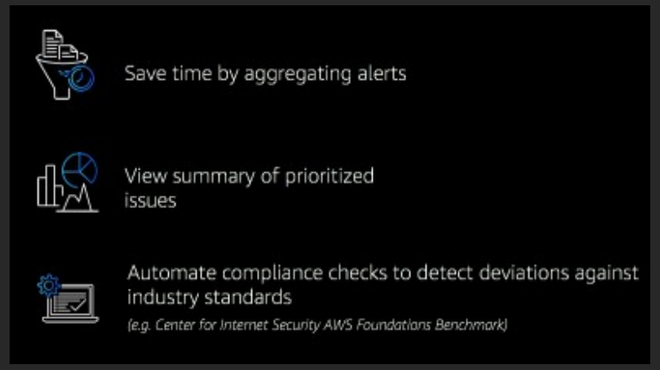
CloudWatch Event Bus

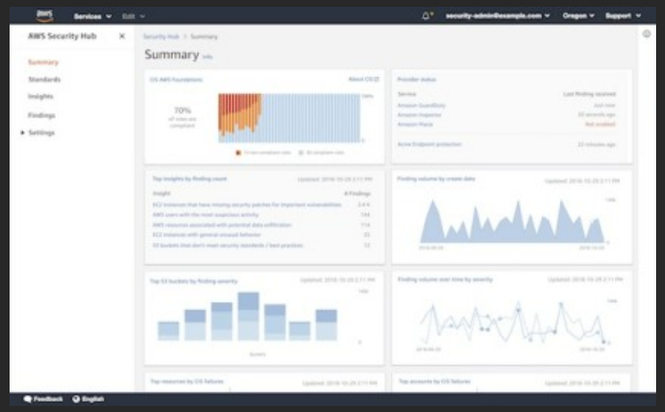
- 異なるアカウントに対して CloudWatch Event を発行
- アカウントごとに持つEvent Busを通じて配信
- 対象のEventについてCloudWatchルールを作成し、受け取るアカウントの Event Busをアタッチ

複数アカウントのセキュリティとコンプライアンスを管理 AWS Security Hubを発表

NEW

- アカウントの数が増えてくると、それぞれのアカウントでどういった事象が起きているかを管理することが難しくなってくる
- AWS Security Hubはそれぞれのアカウントでのセキュリティに関する留意事項を収集し、中央のアカウントで一括表示・対応ができるようにする
- アカウント毎に実施されるコンプライアンスチェックについても同様に対応できる
- 現在プレビュー期間につき追加コストは不要

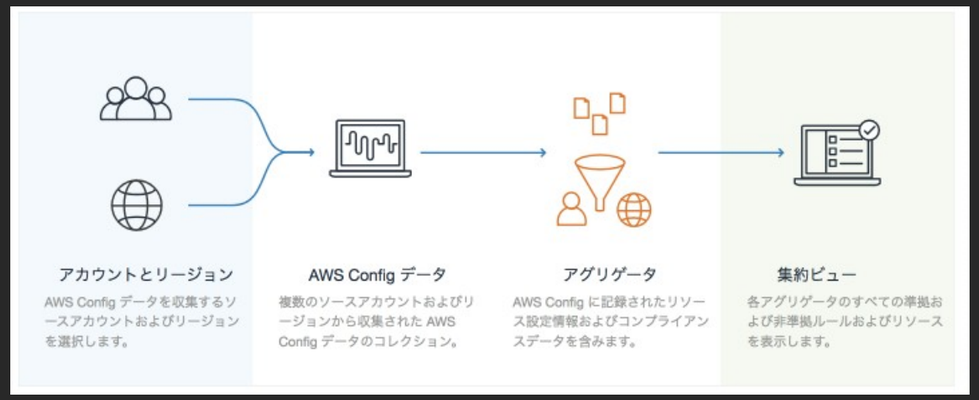


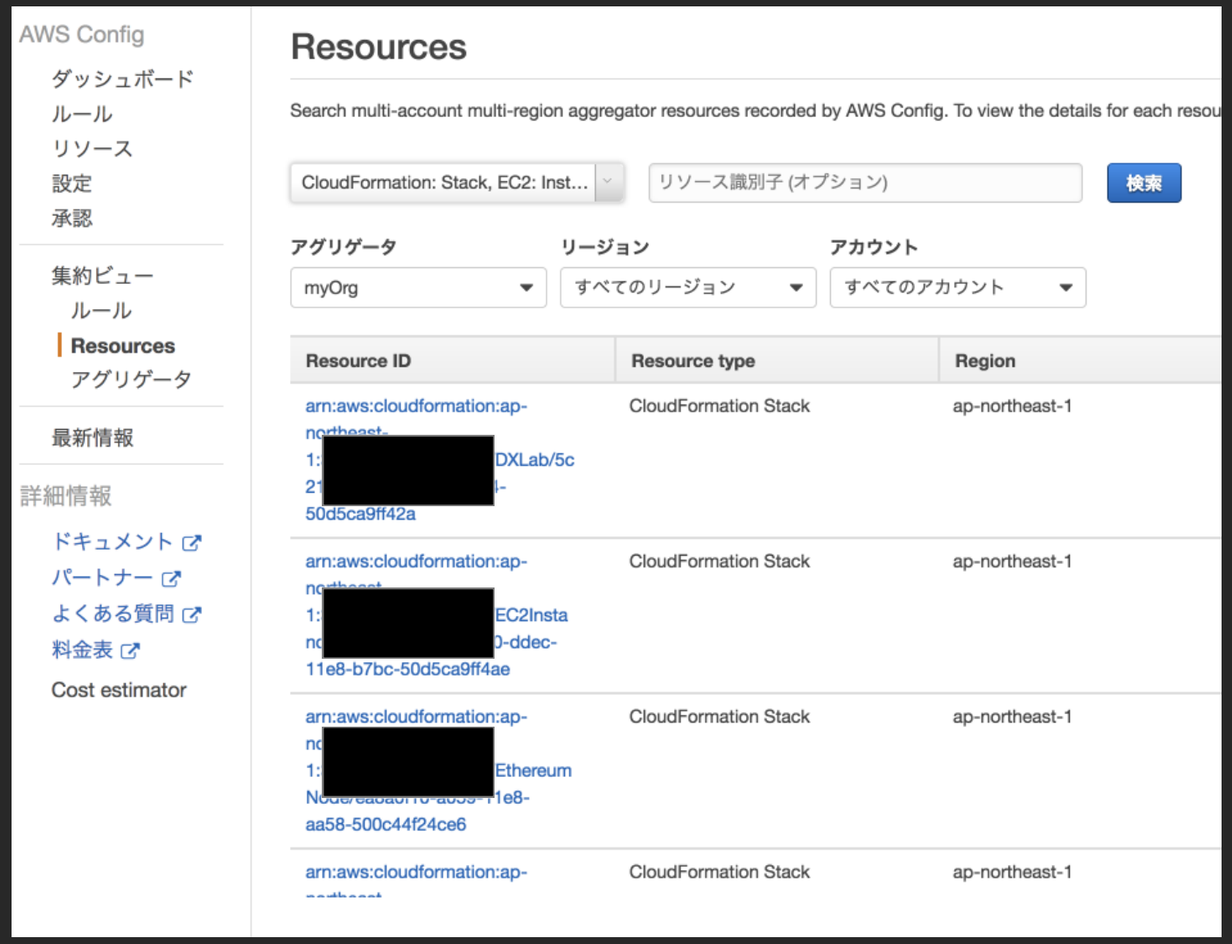


AWS Config Aggregator

UPDATE

- 複数アカウント、リージョンのConfig情報を集約





AWS re:Invent

aws

UPDATE

AWS Systems Manager
Inventory および Automation のマルチアカウント対応

- Inventory
  - OS上のソフトウェア構成を収集しAthenaでアカウントやリージョンをまたいで検索
- Automation
  - キュメントに定義した一連のAPI操作をアカウントやリージョンをまたいだAWSリソースに対して実行

マルチアカウント関連アップデート

アカウントレベルの払い出し管理がより容易に

1. アカウント作成 – Control Tower
2. リソース共有 – Resource Access Manager, Shared VPC
3. アクセス可能 – Transit GW, EFS
4. Provisioning – CFn StackSets
5. Monitoring – CloudWatch Events Bus
6. Audit – Security Hub, AWS Config, SSM Inventory
7. Remediation – SSM Automation

まとめ

まとめ

1. DevOps と AWS Management Tools
2. AWS Management Tools の概要
3. 各サービスの使い所と直近のアップデート
4. マルチアカウント管理

re:Invent 2018 をキャッチアップするために

1. CFn と Management Tools セッションガイド

AWS re:Invent 2018 ダイジェスト ～ AWS の最新動向を学ぶ～

https://amzn.to/2S4F1eN

https://amzn.to/2R23scQ

Thank you!

AWS re:Invent

aws