[AWS Black Belt Online Seminar]

AWS IoT Core

サービスカットシリーズ

Solutions Architect 嶺 行伸
2020/10/27

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive

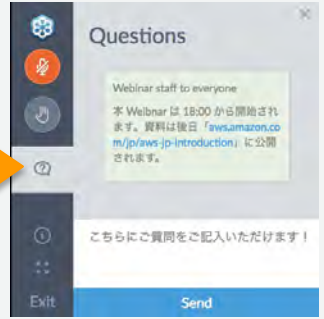
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾン ウェブ サービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問は
  お答えできませんのでご了承下さい

① 吹き出しをクリック
② 質問を入力
③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2020年10月27日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、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。

- 価格は税抜表記となっています。日本居住者のお客様には別途消費税をご請求させていただきます。

- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at http://aws.amazon.com/agreement/. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

自己紹介

嶺　行伸 (みね　ゆきのぶ)

アマゾン ウェブ サービス ジャパン株式会社
デジタルトランスフォーメーション本部
IoT ソリューションアーキテクト

製造業のお客様を中心に、IoT導入をサポートしています

好きなサービス

- AWS Lambda
- AWS IoT Core

本日のアジェンダ

- IoT のユースケースと要件
- AWS IoT Core の概要
- AWS IoT Core によるデータ収集
- AWS IoT Core による遠隔制御
- デバイスの運用・管理
- まとめ

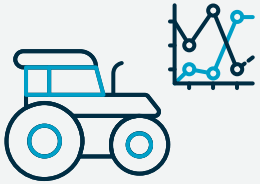
IoT のユースケース

生産性と
プロセスの最適化

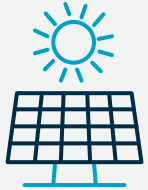
リモートで患者の健康と
状態をモニター

在庫の可視化と
倉庫業務の最適化

家庭、建物、都市のスマート化
及びより良い体験の構築



効率的に良品の
作物を育てる



エネルギー資源を
効率的に管理

コネクテッドカー、自動運転
で輸送の変革

家庭、オフィス、工場
の安全性向上

aws

IoT の代表的な要件

✓モニタリング

位置情報管理・状態監視・
実績把握・導線把握

✓データ連携

保守作業・スマートデバイス・
企業連携・オープンデータ

✓分析・予測

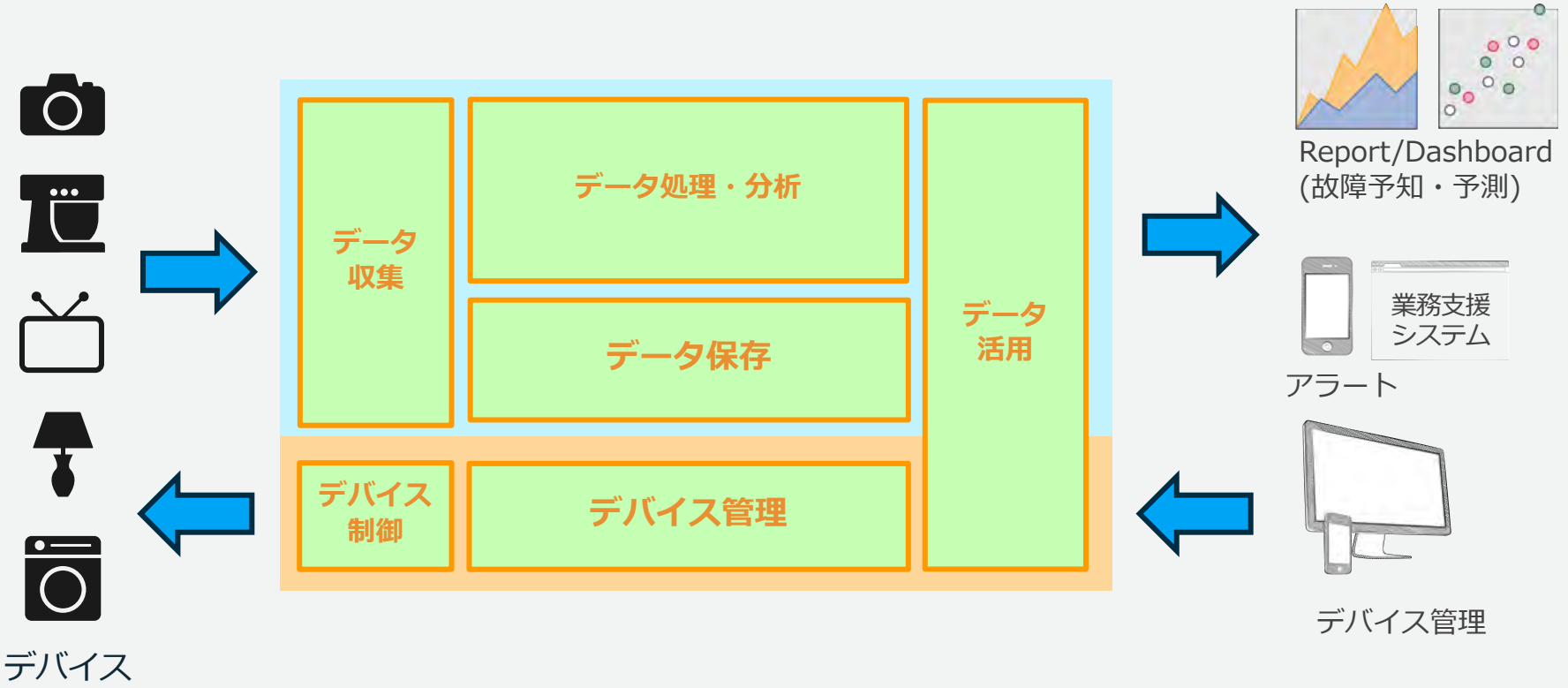
異常検知・故障予測

✓遠隔制御

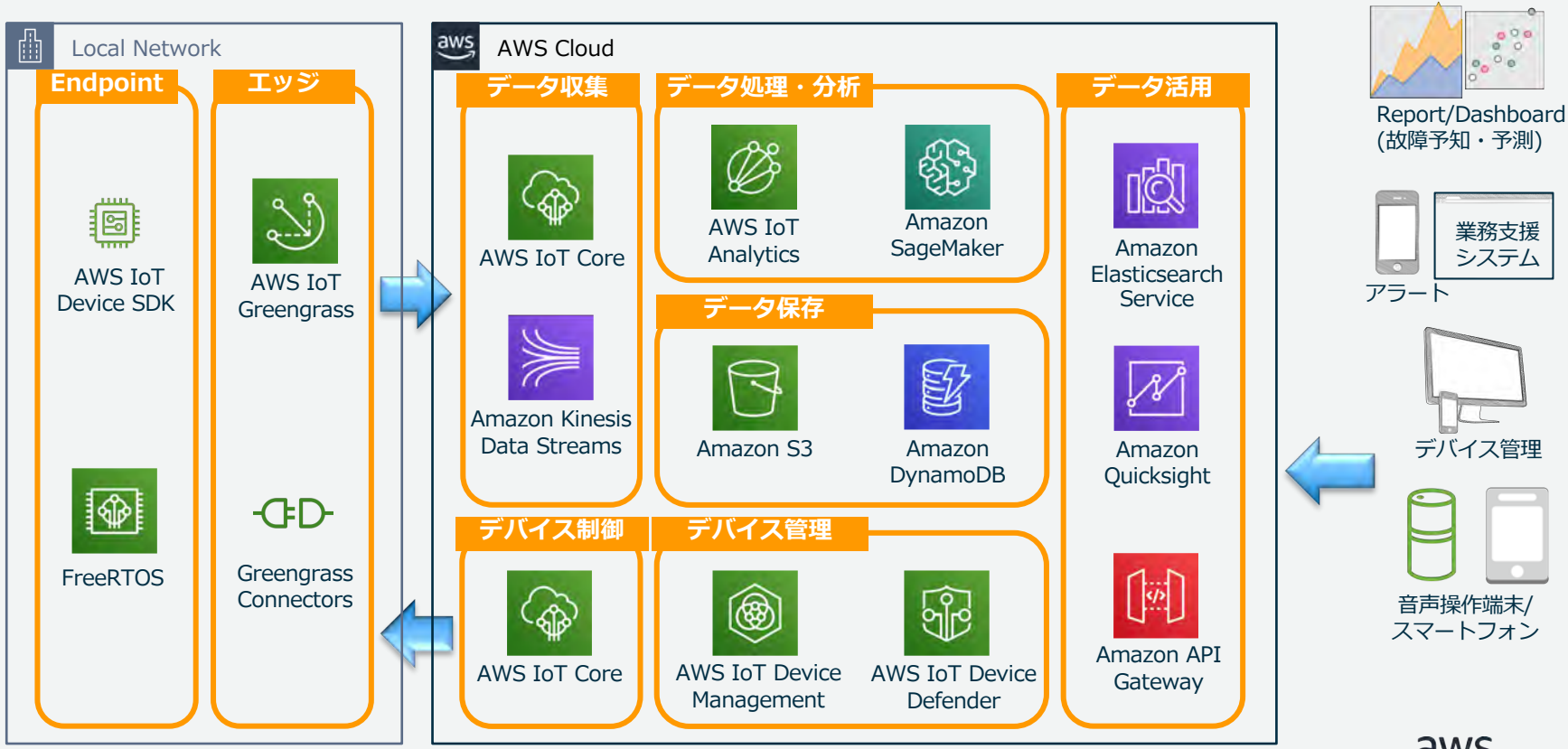
機器運用・アップデート・作業
自動化

aws

IoT におけるクラウドの役割



各レイヤーで AWS が提供するクラウドサービス



本日のアジェンダ

- IoT のユースケースと要件
- AWS IoT Core の概要
- AWS IoT Core によるデータ収集
- AWS IoT Core による遠隔制御
- デバイスの運用・管理
- まとめ

AWS IoT アーキテクチャ

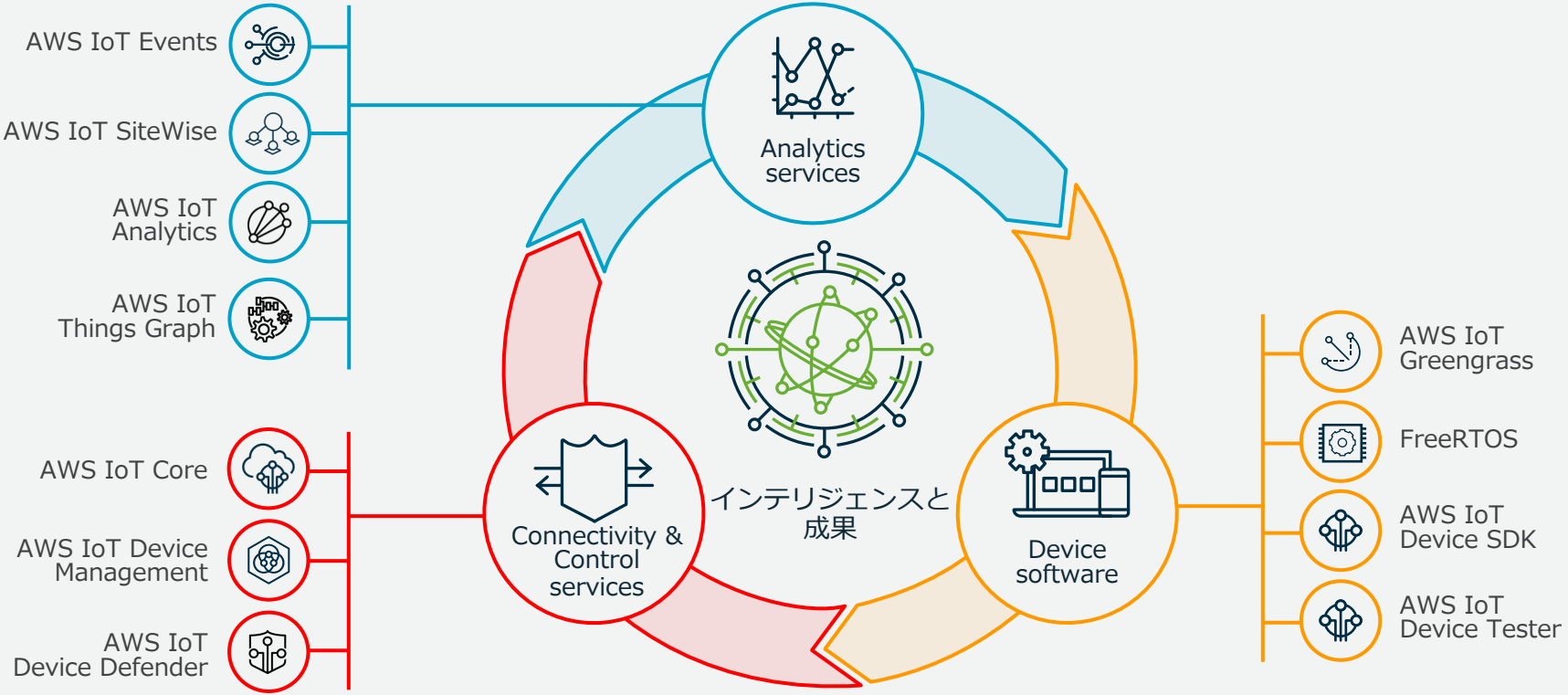
どのようにしてIoTデータから価値を抽出するのか?

デバイスと通信・管理・保護する方法は?

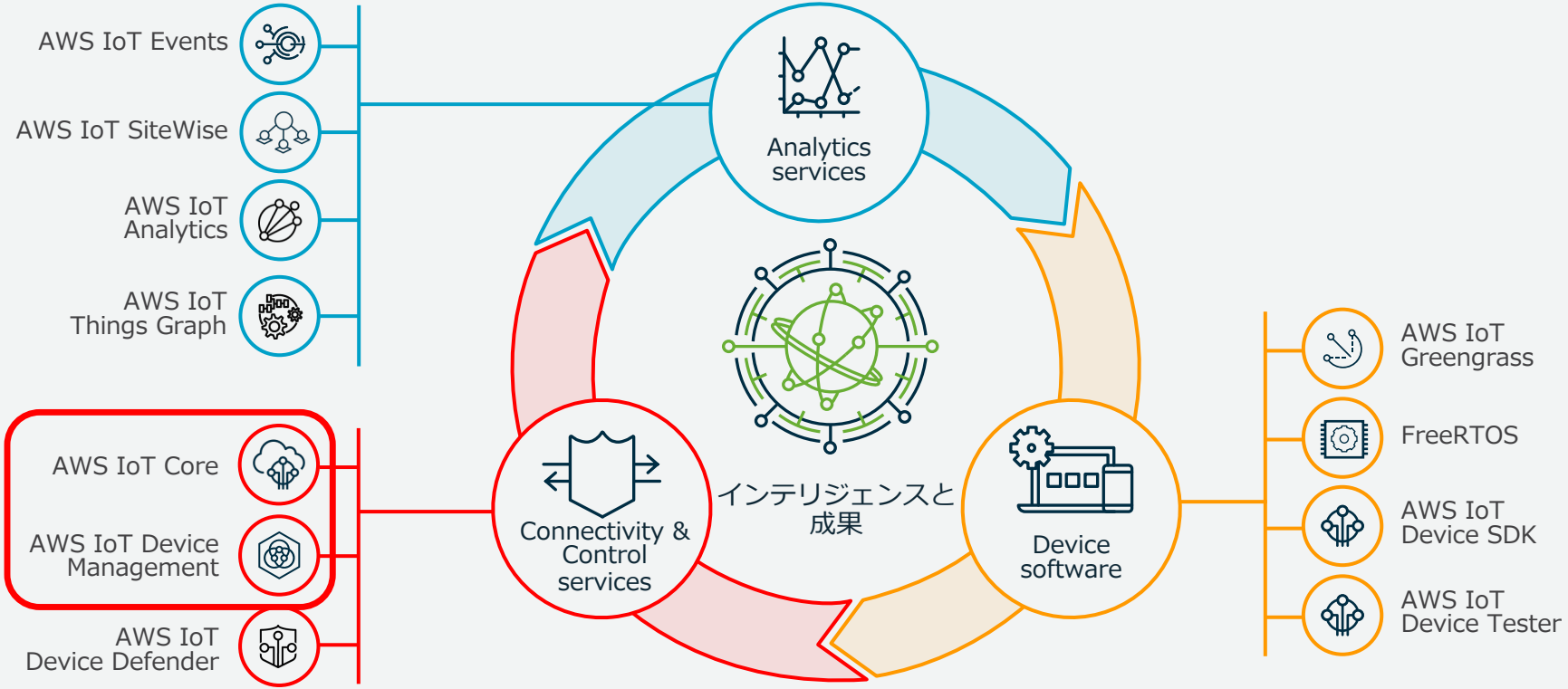
エッジデバイスをどのように接続・操作するのか？

aws

AWS IoT サービス一覧



AWS IoT サービス一覧

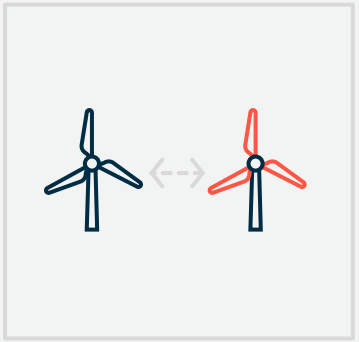


AWS IoT Core

AWS IoT Core は、IoT デバイスを簡単、かつ安全にクラウドアプリケーションや他のデバイスと通信できるようにするマネージドサービスです

大量のデバイスをAWSクラウドや他のデバイスに安全に接続する

接続されたデバイスからのデータに基づいて、ルーティング、処理、実行を行う

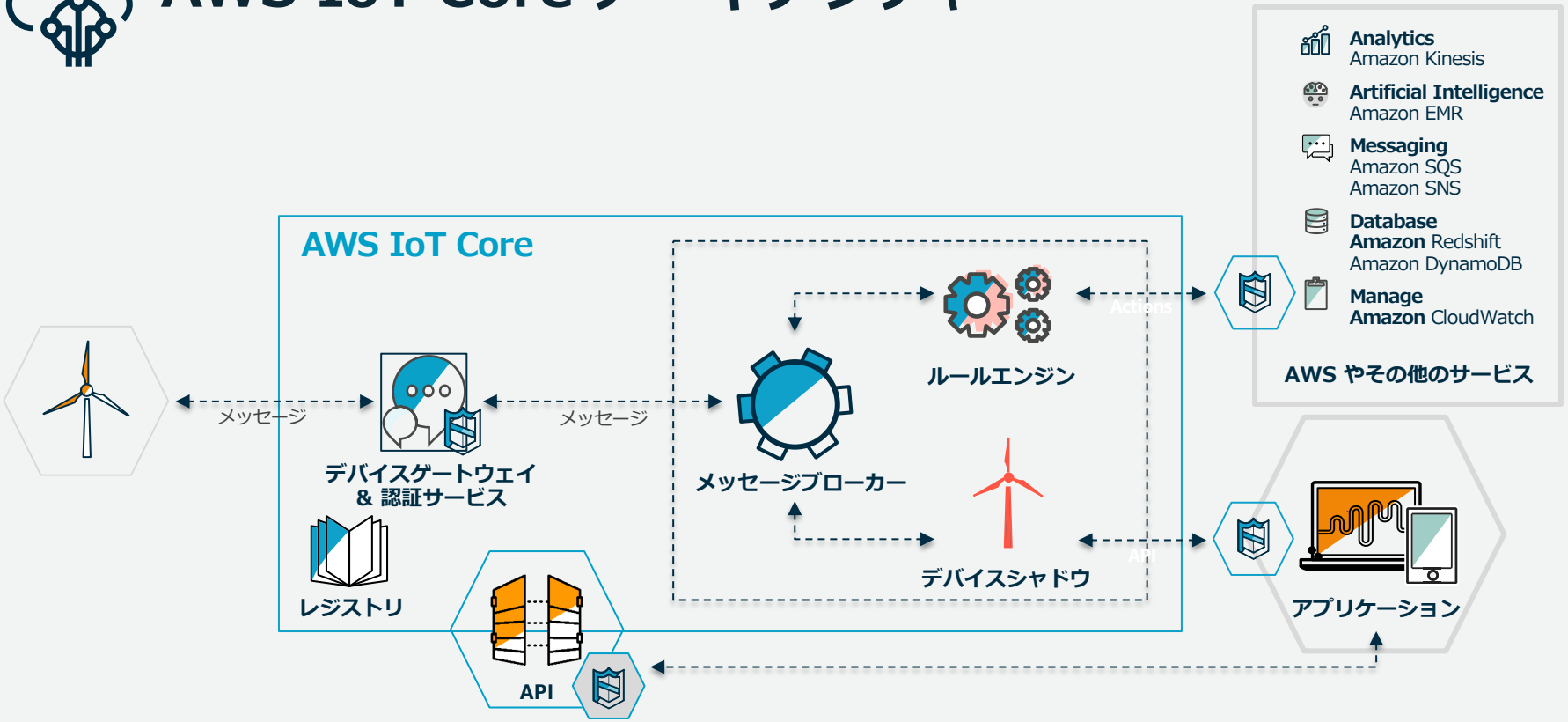


デバイスがオフラインであっても、アプリケーションがデバイスと対話できるようにする

データを基に推論を行うためのAWSの各種サービス(Analytics、データベース、AIなど)との完全な統合

Connectivity & Control Services

AWS IoT Core アーキテクチャ



デバイスゲートウェイ

IoT ワークロードに最適化されたフルマネージドな接続管理

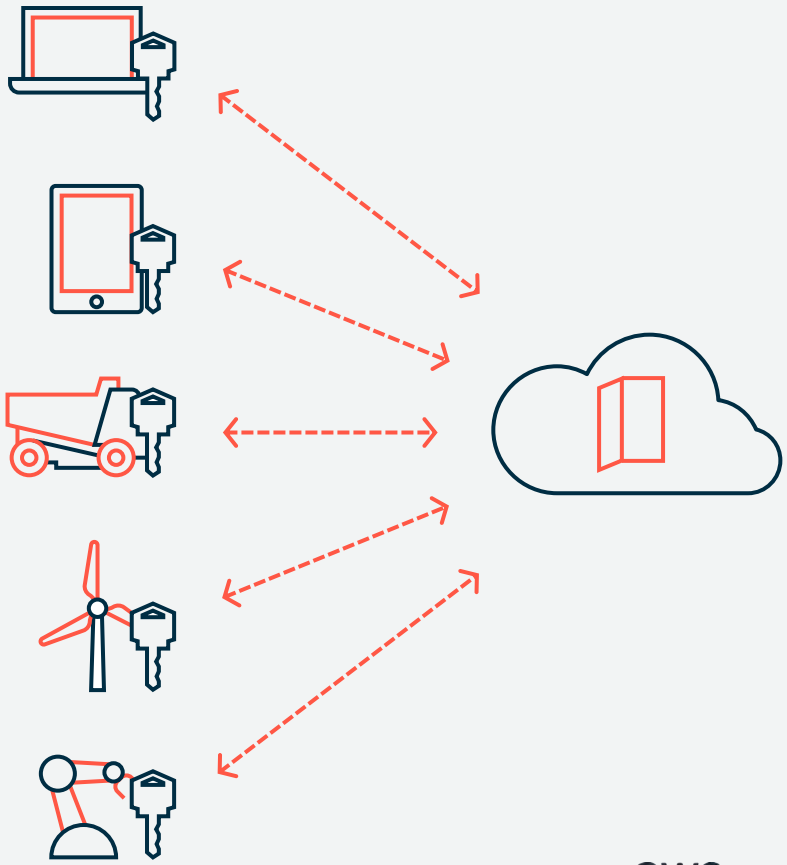
MQTT、WebSocket、HTTP のプロトコルをサポート

TLS 1.2 を使用した安全な通信

制約のあるデバイスへの最適化

ECC鍵交換と証明書

最大フラグメント長のネゴシエーション



メッセージブローカー

IoT デバイス間で信頼性の高い高速通信

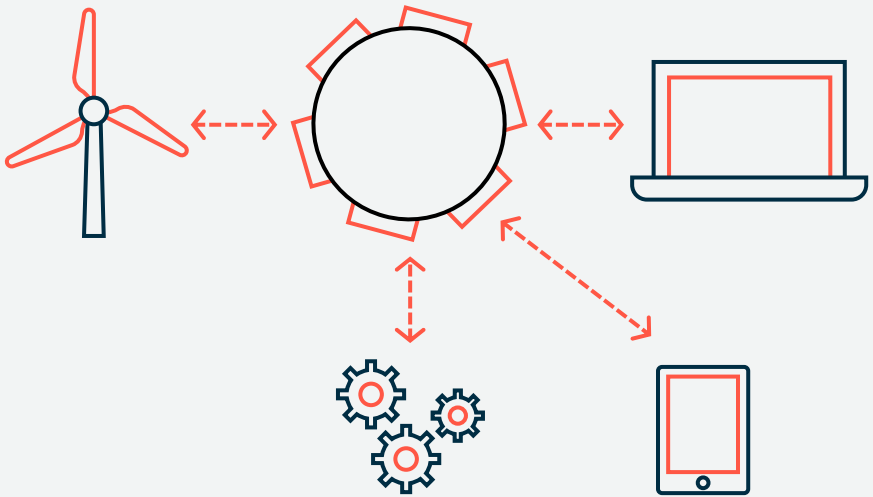
デバイスとアプリケーション間の
双方向メッセージ・ストリーミング

オフラインデバイス向け
メッセージキュー

デバイスとアプリケーションを
疎結合にするための
パブリッシュ＆サブスクライブ

QoS0 および QoS1 メッセージングの
サポート

ワイルドカード・トピックフィルタをサポートするカスタマイズ可能なトピック空間



aws

MQTT プロトコル

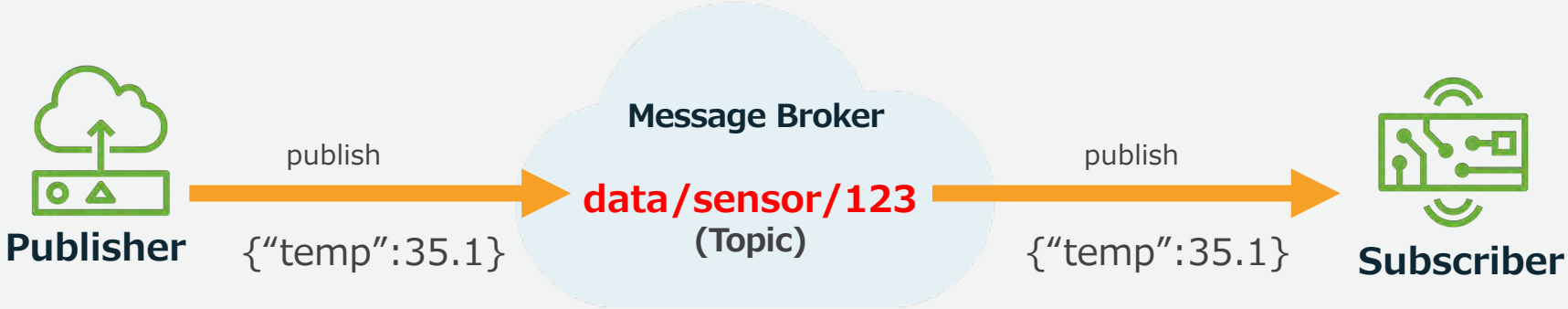
Message Queuing Telemetry Transport

- IoT / M2M で利用される通信プロトコル
  - OASIS 標準プロトコル
  - AWS IoT Core は v3.1.1 に準拠
- リソースや回線帯域が限られているデバイスに最適化
- Publish / Subscribe メッセージ交換モデル
  - ブローカーがメッセージを仲介

aws

Publish / Subscribe モデル

- スケーラブルな非同期型メッセージ交換モデル
- 最初に Publish もしくは Subscribe されたタイミングで Topic が生成される
- Publisher は Topic に対してメッセージを Publish(送信) する
- Message Broker はマッチする Topic に Subscribe(受信) している Subscriber へメッセージを Publish する



Topic とは

- メッセージ(データ) を送る先を示す文字列
- 最低一つの Topic Level(階層) を持つ
- “/” (スラッシュ) で複数(最大8階層) の Topic Level を連結
  (先頭に “/” は不要)

例)

foo

foo/bar

foo/bar/buz

トピックレベル

Topic の注意点

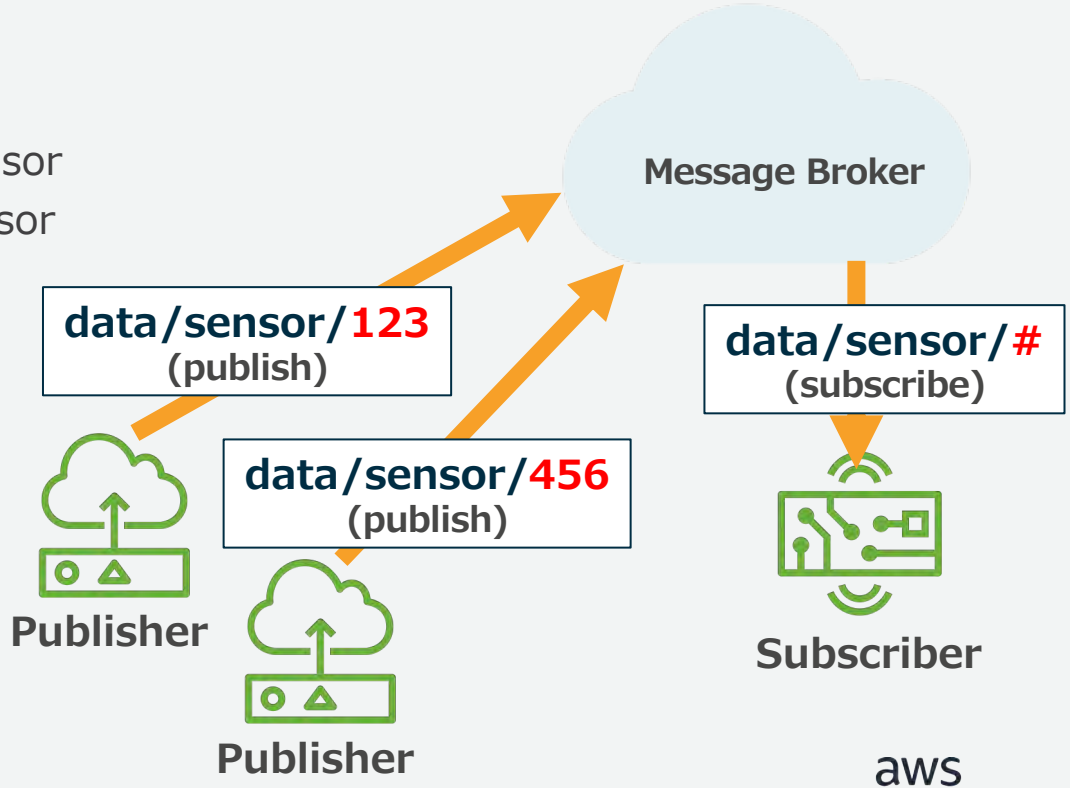
- “$” で始まる Topic は予約されているので使用しない
  - 例: $aws/event/～～
- 大文字と小文字は区別される
  - 可能な限り小文字と数字と ”-” だけを使う
- Topic の長さは UTF-8 で 256bytes まで
  - 可読性を損なわない範囲で短くする

Topic 設計の詳細については white paper をご参照下さい
https://d1.awsstatic.com/whitepapers/Designing_MQTT_Topics_for_AWS_IoT_Core.pdf

ワイルドカード

- Subscribe する Topic の指定でワイルドカードを利用できる

- 1階層にマッチ: +
  - meguro-building/1f/+/sensor
  - meguro-building/+/+/sensor
- 末尾の複数階層にマッチ: #
  - meguro-building/#
  - meguro-building/1f/#



メッセージについて

- 128KB までの任意のデータを送信可能
- 特に制約がなければ JSON 形式を推奨
    - メリット
        - ルールエンジン(後述)で JSON 内の属性を参照できる
        - 他の AWS サービスとの連携もしやすい
    - デメリット
        - メッセージサイズは大きくなる
        - デバイスが JSON を Parse / Serialize できるリソースを必要とする

QoS とは

QoS=0
1回のみ送信

ベストエフォート型。
メッセージの到達を保証していない。
オーバーヘッドは少ない。

QoS=1
最低1回の到達を保証

保証型。
メーッセージの到達は保証。
オーバーヘッドはQoS=0に比べて
大きい。

認証サービス

デバイス認証を管理し、ユニークなアイデンティティを大規模に提供

AWS IoT Core が発行した証明書か、
お客様独自の CA が発行した証明書を使用可能

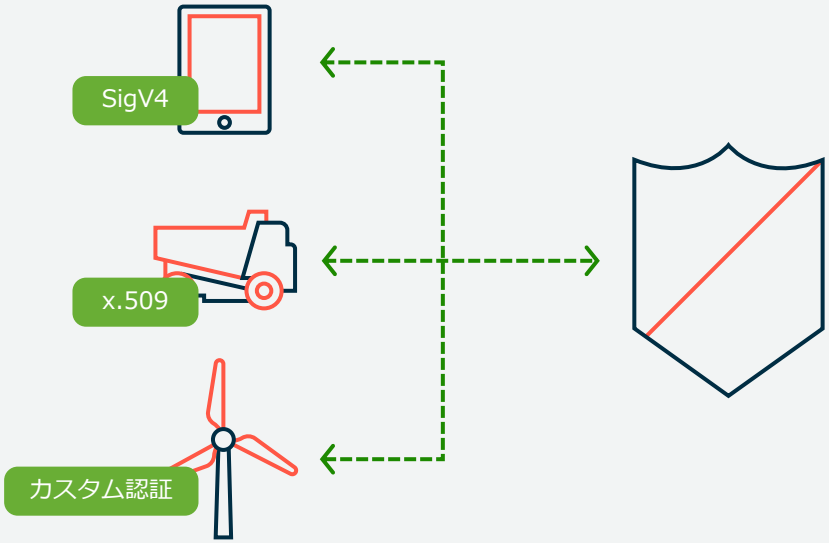
ジャストインタイム登録を使用した
自動デバイス・プロビジョニング

x.509 デバイス証明書による TLS 相互認証、
SigV4、およびカスタム認証をサポート

IoT ポリシーによる
柔軟できめ細かいアクセス制御

ポリシーはIDまたはレジストリ項目に関連付けることができます

MQTT トピックレベルまでアクセスを制御できます



AWS IoT Core で利用可能な認証方法

1. **x.509クライアント証明書**
   - TLS 相互認証
   - 登録数に上限なし、証明書の持ち込みも可能

2. **Amazon Cognito**
   - スマートフォンなど、入力装置があるデバイスで利用可能
   - 人による認証情報の入力ができない場合はマッチしない

3. **AWS IAM**
   - SigV4 セキュリティトークンでの認証
   - 作成数に上限があるため、大量のデバイスの運用には適さない

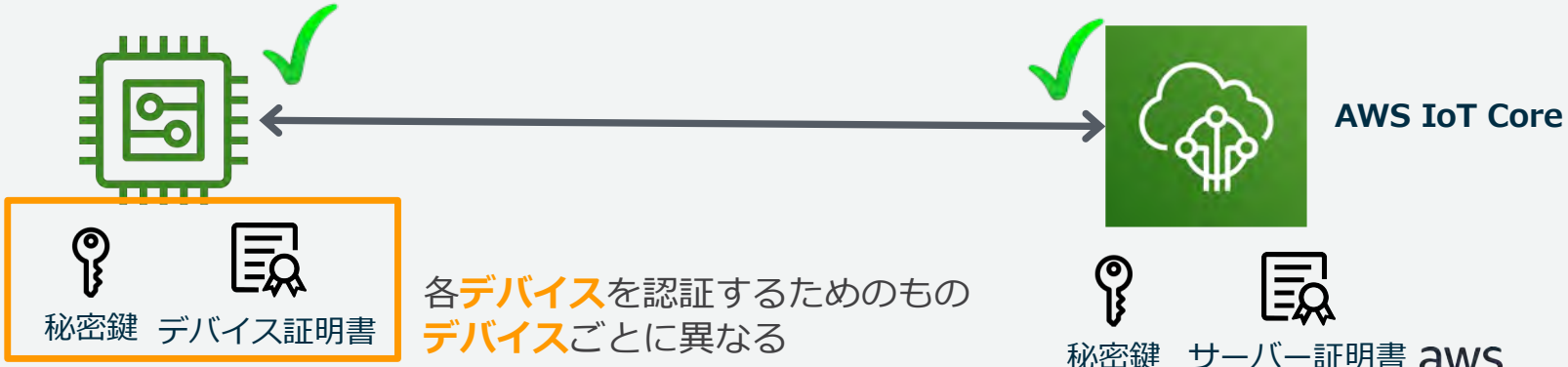
4. **カスタム認証**
   - AWS Lambda で独自の認証方法を実装
   - HTTP ヘッダー、MQTT connect 時の ID/Password 等

TLS 相互認証によるデバイス認証

通常の Web サーバーにおけるユーザー認証

各**ユーザ**を認証するためのもの
ユーザごとに異なる

TLS 相互認証によるデバイス認証

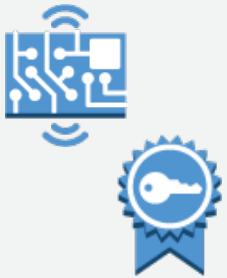


各**デバイス**を認証するためのもの
デバイスごとに異なる

IoT ポリシーによる認可

各デバイスに許可する動作を IoT ポリシーとして定義する
- connect, publish, subscribe, receive 等

定義した IoT ポリシーはデバイスの証明書またはグループにアタッチできる



証明書にアタッチ

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action":["iot:Publish"]
  "Resource": ["arn:aws:iot:ap-northeast-1-1:123456789012:topic/foo/bar"]
 }, {
  "Effect": "Allow",
  "Action": ["iot:Connect"],
  "Resource": ["*"]
 }]
}
```

AWS IoT Coreで利用できるプロトコルと認証認可方法

	MQTT	MQTT over Websocket	HTTPS	
Pub/Sub	Pub/Sub	Pub/Sub	Pub	Pub
通信方向	双方向	双方向	デバイス → クラウド	デバイス → クラウド
ポート	8883, 443	443	8443, 443	443
認証方法	x.509 証明書 カスタム認証	Cognito SigV4 カスタム認証	x.509 証明書 カスタム認証	Cognito SigV4 カスタム認証
権限管理	IoT ポリシー	IoT ポリシー IAM ロール	IoT ポリシー	IoT ポリシー IAM ロール

aws

AWS IoT Core における証明書発行とデバイス登録方法一覧

1. AWS IoTによる秘密鍵・証明書発行＆事前登録
   (デバイスキッティング時登録)
2. AWS IoTによる証明書発行＆事前登録
   (デバイスキッティング時登録)
3. Fleet Provisioning登録 [new!]
4. 独自CAによる証明書発行＆AWS IoTへの事前登録
5. 独自CAによる証明書発行＆JITRによる登録
6. 独自CAによる証明書発行＆JITPによる登録
7. CA登録無しの証明書登録（マルチアカウント登録）[new!]

詳細については AWS IoT におけるデバイスへの認証情報のプロビジョニングで解説しています

レジストリ

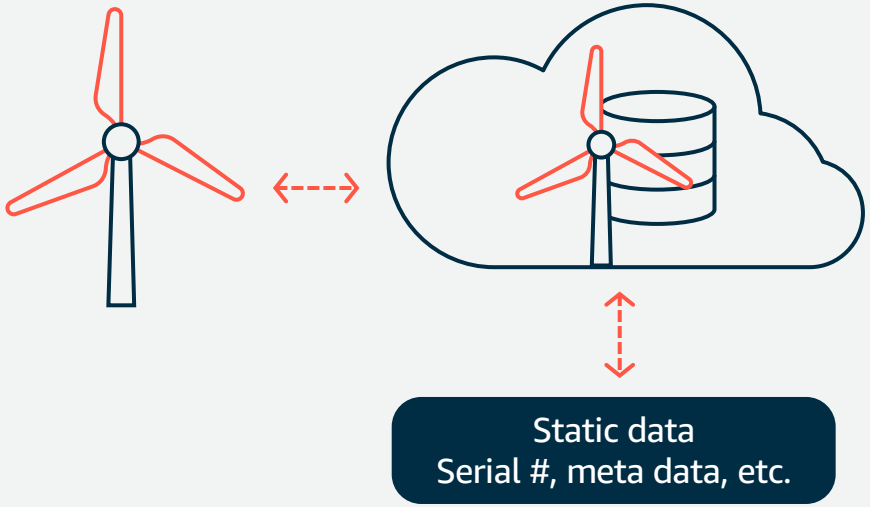
AWS サービスで簡単に使えるようにデバイスを定義してカタログ化する

デバイスにカスタム属性を定義できます

属性値でデバイスを検索できます
(例: 2010年に製造されたデバイス)

Thing Type の定義により、デバイス間で属性とポリシーの標準化を可能にします

Thing Group の定義により、デバイスの一括管理を可能にします
（ジョブの実行、ポリシーの設定など）



AWS IoT Core におけるデバイス情報

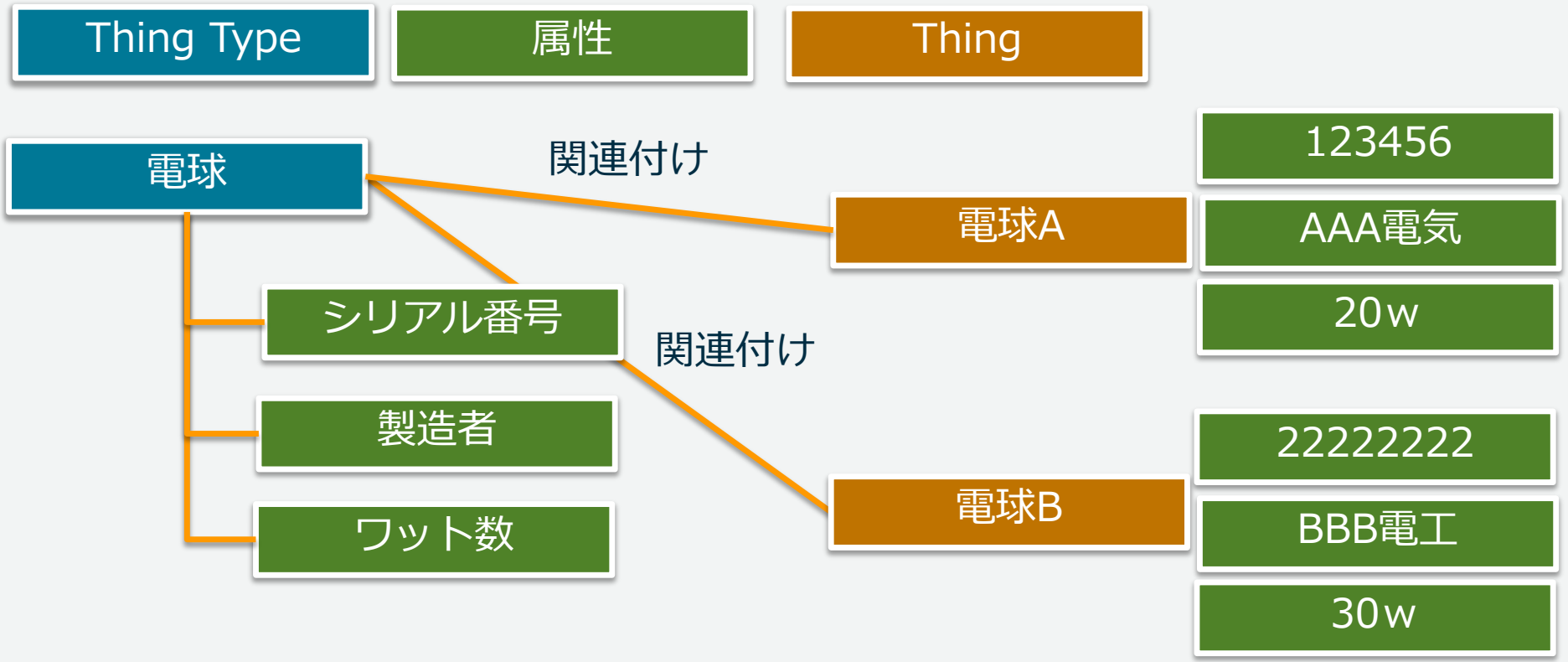
デバイス毎に「静的な属性」と「動的な状態」を管理

静的な属性: レジストリに保存　　動的な状態: デバイスシャドウ(後述)に保存

```
{
        "version": 3,
        "thingName": ”9005689918",
        "defaultClientId": "9005689918",
        "thingTypeName": "LightBulb",
        "attributes": {
                "model": "123",
                "wattage": "75”
         }
}
```

```
{
   "state": {
      "reported": {
         "lights": {
            "color": "GREEN"
         },
         "engine": "ON"
      }
   },
   “metadata”: { ... },
   "version": 10,
   "timestamp": 123456789
}
```

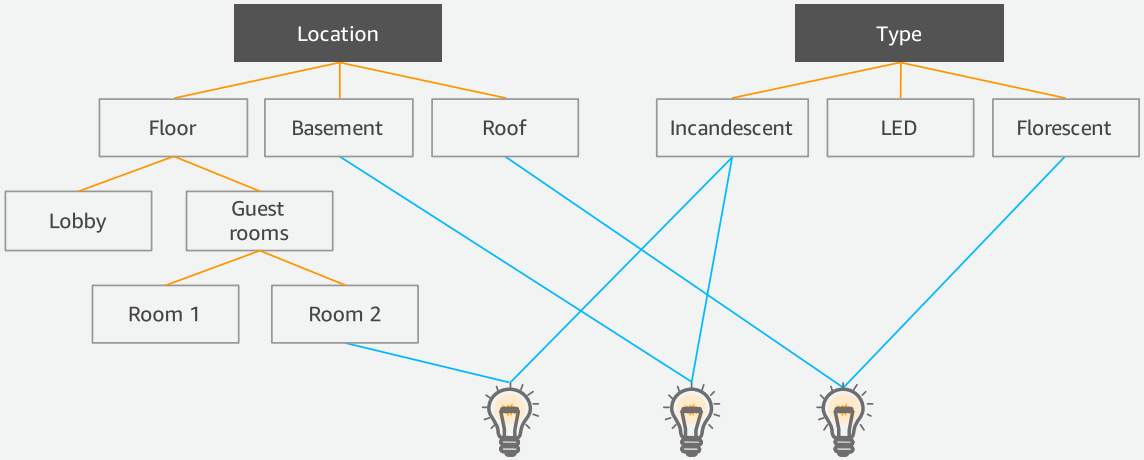
Thing Type



Thing Type の制約

- 1つのデバイスに指定できる Thing Type は 1つ
  - 一度指定した Thing Type は変更できない
- Thing Type につけた名前は変更できない
- デバイスに設定できるカスタム属性数の制限
  - Thing Type が指定されたデバイス: 最大 50 個
  - Thing Type が指定されていないデバイス: 最大 3 個

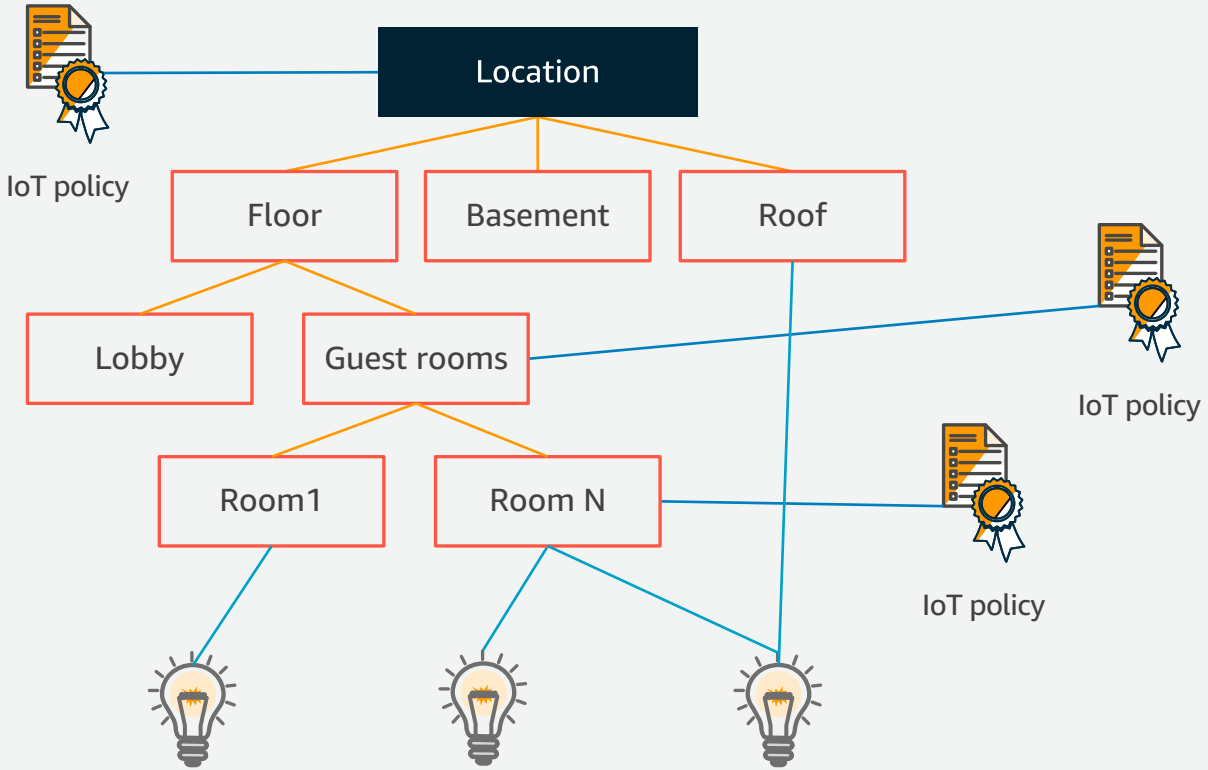
Thing Group



- デバイスをグループ化して管理
  - ビル → フロア → 部屋 といった階層設計など、自由に設定可能

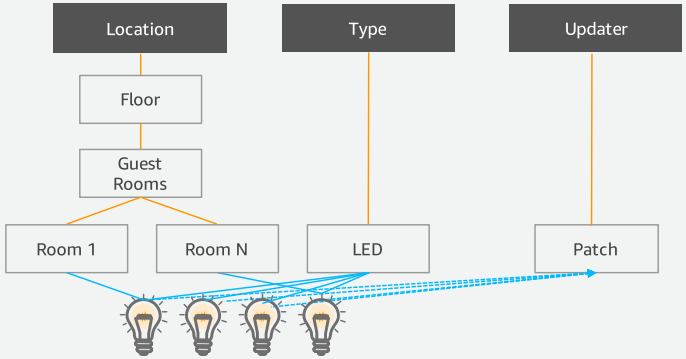
グループ単位での IoT ポリシー設定

テンプレートにより、グループ単位で IoT ポリシーを一括設定



Thing Group の制約

- Thing Group につけた名前は変更できない
  - û、é、ñ などの国際文字を含めることはできない
- 1デバイスに指定できるグループは最大10個
- Thing Group は最大1つの直接の親を持ち、一度設定した親は変更できない
  - 階層は 7 階層まで
  - 子グループの数は最大100個
  - グループ階層は入念に計画し、それに含む子グループを作成する前に親グループを作成する
- デバイスは同じ階層構造の中で1つの Thing Group にのみ所属できる
  - 右記の例だと、同時にRoom1, Room2に所属できない
- グループにアタッチできる IoT ポリシーは最大２つ



https://docs.aws.amazon.com/ja_jp/iot/latest/developerguide/thing-groups.html

AWS IoT Core の利用方法

AWS IoT Device SDK

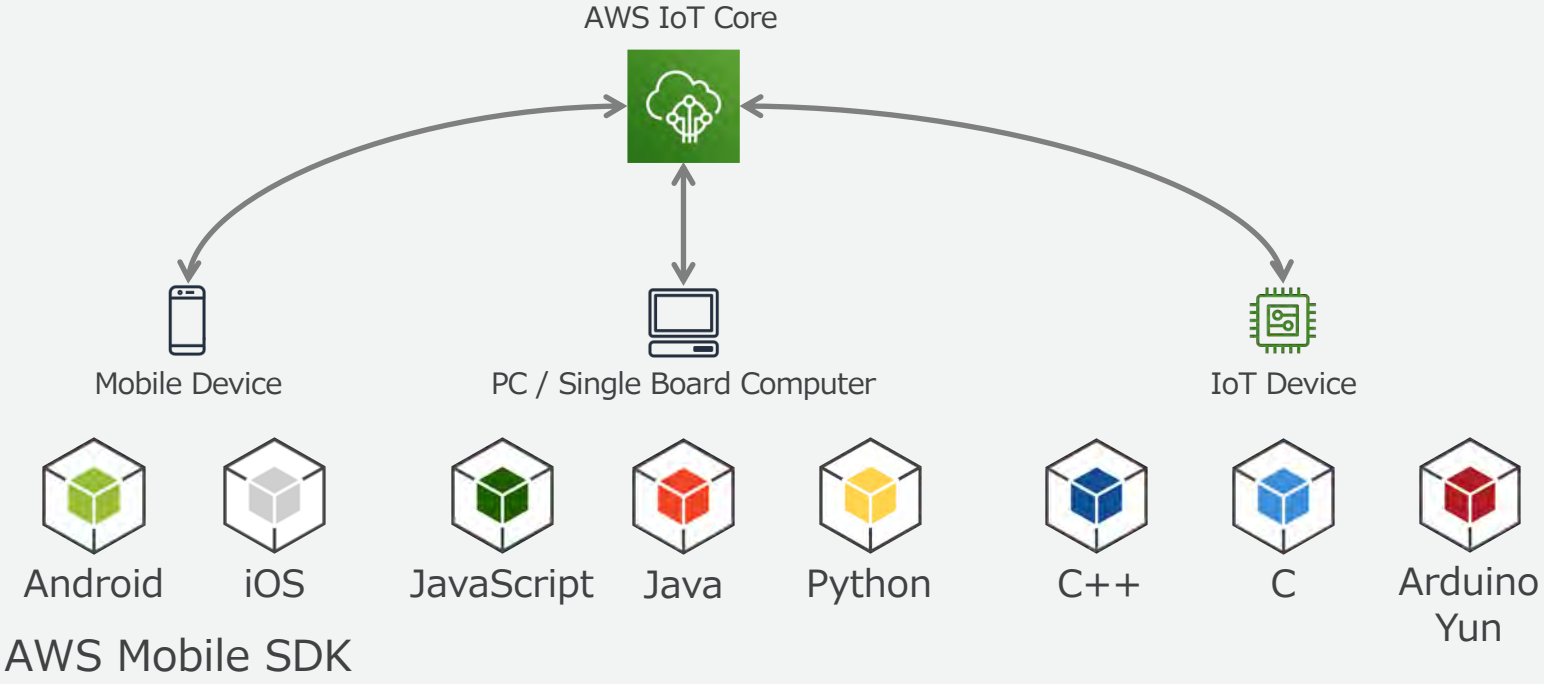
AWS SDK

FreeRTOS

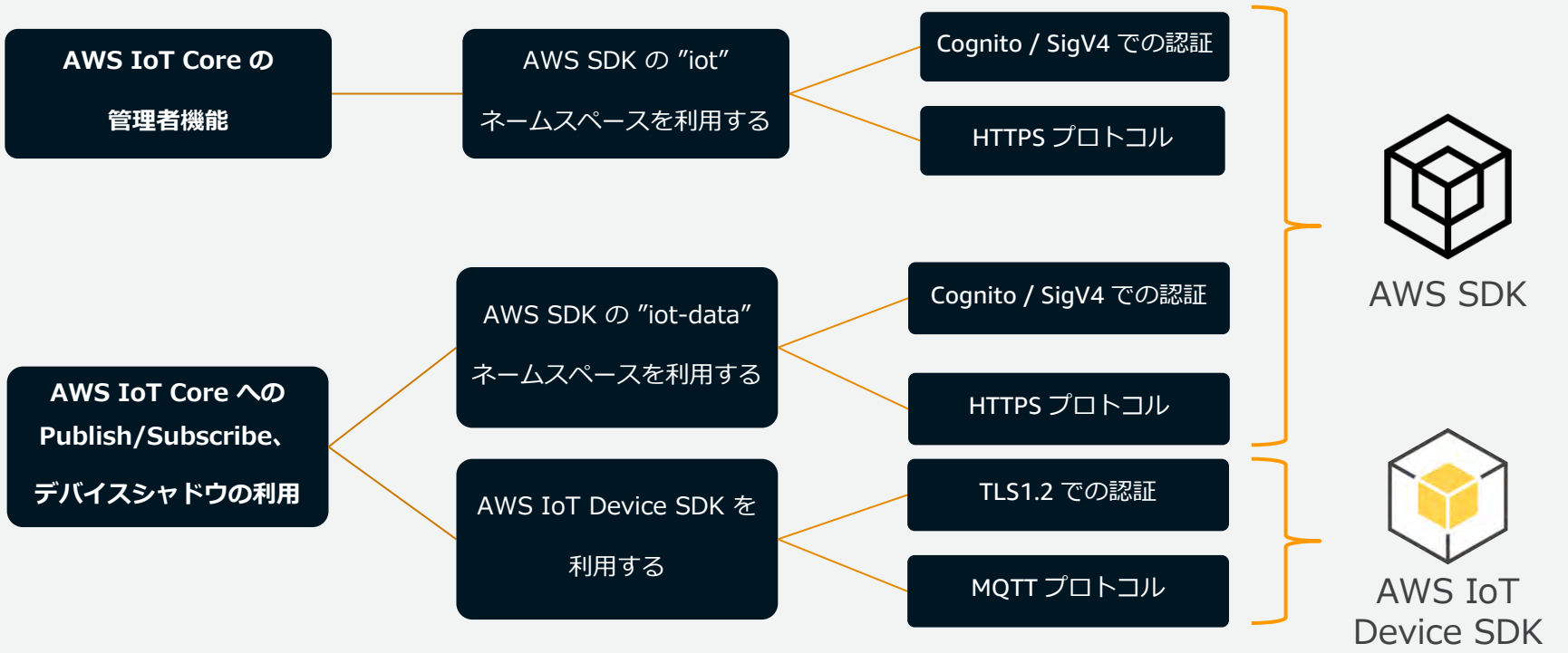
aws

AWS IoT Device SDK

様々なデバイスを AWS IoT Core に接続するための SDK



AWS SDK とAWS IoT Device SDK の違い



FreeRTOS

- 15年にわたり信頼され、広く配布された
- RISC-V やArm v8-Mを含む、40以上のアーキテクチャでサポート
- 広範囲なエコシステム
- フリーでオープンソース
- MITオープンライセンス
- 最も有名なリアルタイムOS

詳細については Black Belt Online Seminar FreeRTOS で解説しています

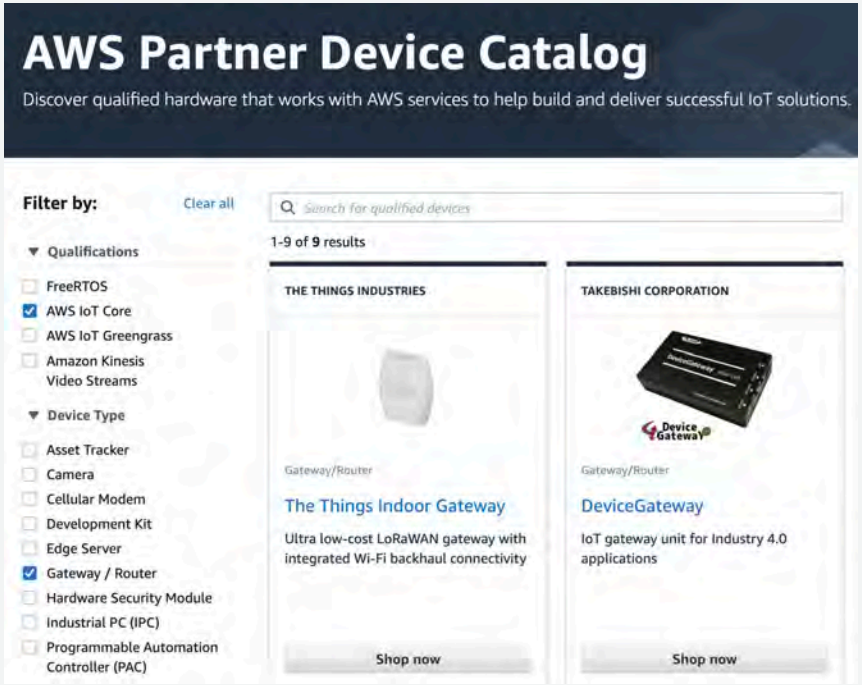
Device
software

aws

パートナーデバイスの利用

AWS IoT Core に対応したパートナーデバイスもご利用いただけます

AWS Partner Device Catalog



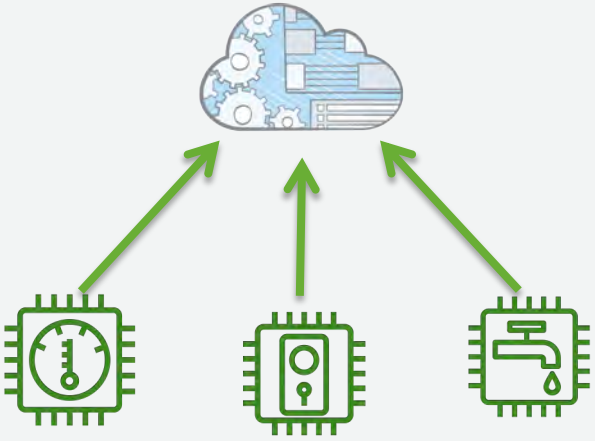
本日のアジェンダ

- IoT のユースケースと要件
- AWS IoT Core の概要
- AWS IoT Core によるデータ収集
- AWS IoT Core による遠隔制御
- デバイスの運用・管理
- まとめ

AWS IoT Core の 2 つの利用用途

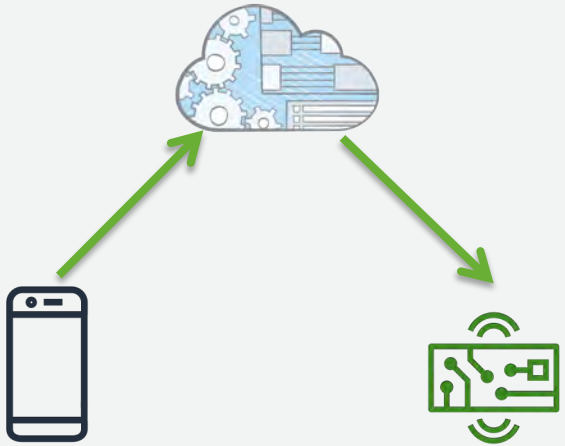
データ収集

数十万規模のデバイスからのデータ収集



遠隔制御

遠隔にあるデバイスをクラウドを介してコントロール

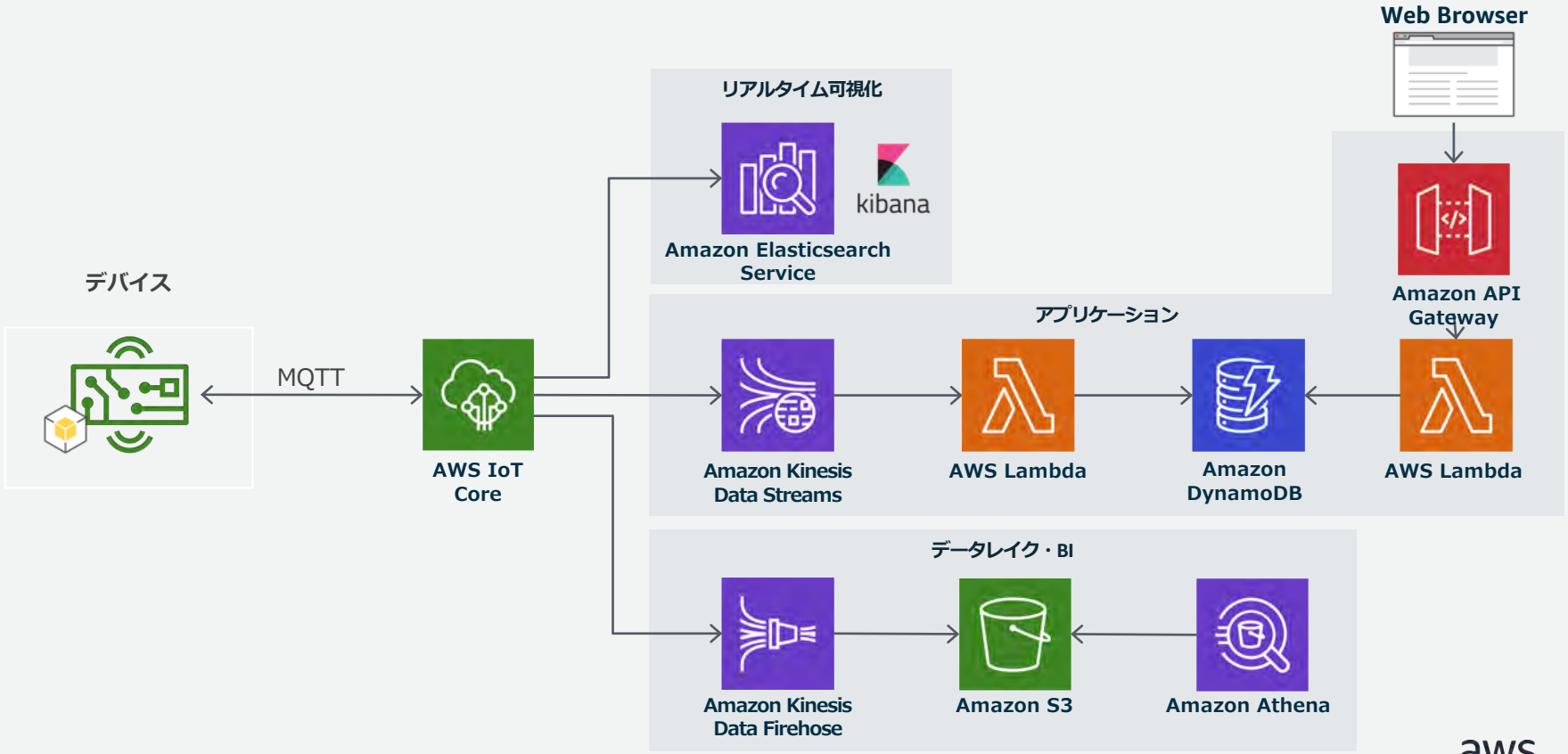


使用例

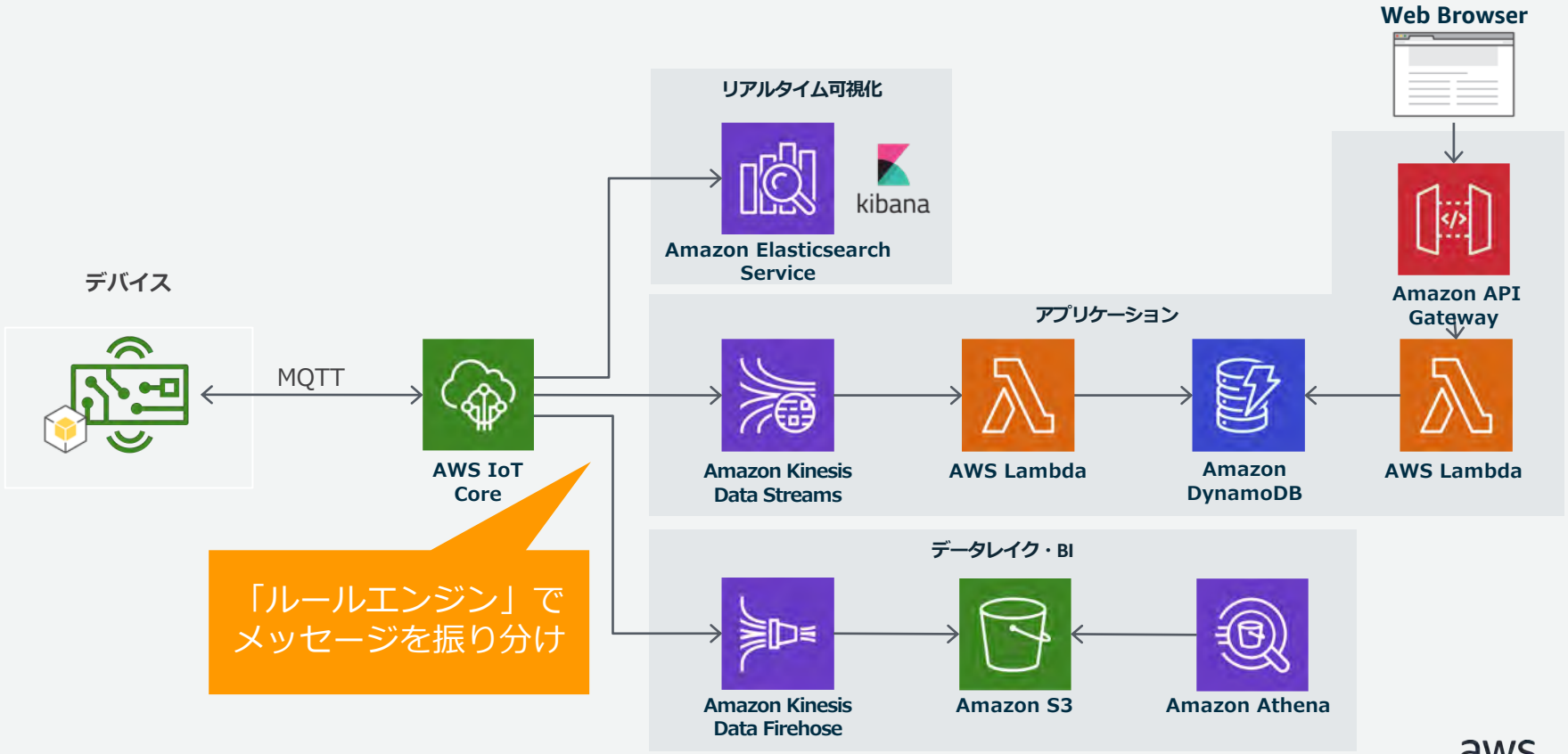
デバイスのデータの可視化
故障予測・異常検知

リモート機器制御
ファームウェア更新

AWS IoT Core によるデータ収集アーキテクチャ例



AWS IoT Core によるデータ収集アーキテクチャ例



ルールエンジン

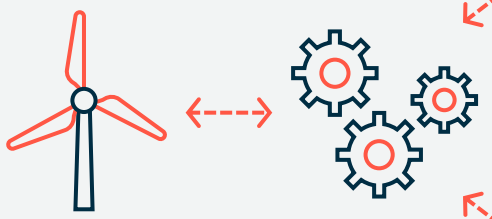
大量のデータを SQL ライクな文法で取り込み、分析や視覚化などのために10以上のサービスを利用可能

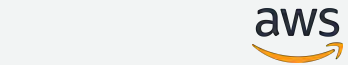
変換: 数値演算、文字列操作、日付などの組み込み関数

フィルター: WHERE 句を使用して必要なデータのみを取得

エンリッチ: Device Shadow と Amazon Machine Learning または インライン AWS Lambda 実行を介した外部ソースからのコンテキスト取り込み

ルーティング: 10以上の AWS サービスや Salesforce, HERE などのサードパーティサービスにデータを送信





Analytics
Kinesis
Elasticsearch Services
IoT Analytics

Database
DynamoDB

Manage
CloudWatch

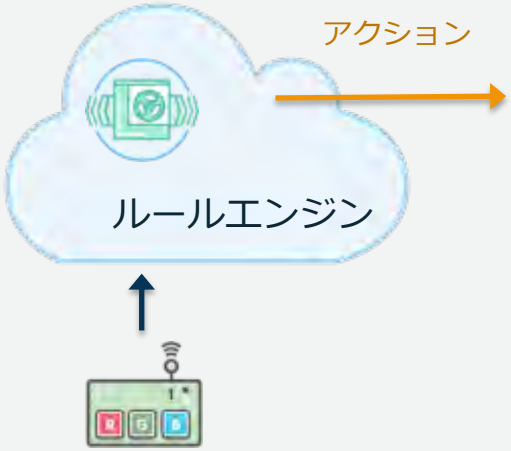
Compute
Lambda

Application Integration
SNS
SQS
Step Functions

aws

ルールエンジン - アクション

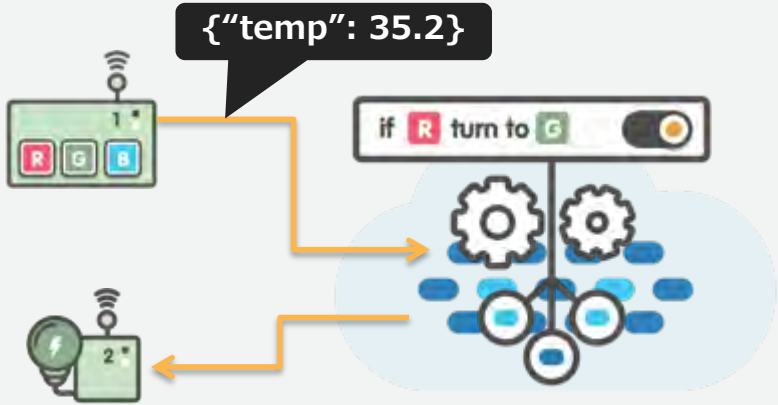
AWS IoT Core から
他の AWS サービスや
外部サービスと連携



- Amazon CloudWatch (Alarm, Metrics, Logs)
- Amazon Simple Notification Service
- Amazon Simple Queue Service
- Amazon Kinesis
- Amazon Kinesis Data Firehose
- Amazon Elasticsearch Service
- AWS Step Functions
- AWS Lambda
- Amazon DynamoDB
- Amazon Timestream [new!]
- AWS IoT Analytics
- AWS IoT Events
- AWS IoT SiteWise
- Amazon Simple Storage Service
- HTTP endpoint
- MQTT republish

ルールエンジン – メッセージのフィルタリング

- SQL 文を使ったトピックのフィルタ
- WHERE 句でフィルタリング条件を記述することが可能
- JSON 形式のメッセージをサポート



- トピックのフィルタの例

取得するデータ →
対象となるTopic →
条件 →

```
SELECT *
FROM 'data/meguro-building/1f/+/sensor'
WHERE temp > 35.0
```


aws

データ収集における Topic 設計のベストプラクティス

- Topic Levelは、抽象度が高い情報から、低い情報へ

  例)

  - ビルID/フロアID/エアコンID/センサーID

    抽象度高　>>>>>>>>>>>>　抽象度低

  - meguro-building/1f/aircon1/sensor
  - japan/tokyo/shinagawa/kamiosaki
  - factory1/building2/line1/stage3

aws

MQTT では扱えないデータの収集方法

プロトコル、アップロード頻度、データサイズ、データ種別、双方向性などにより選択

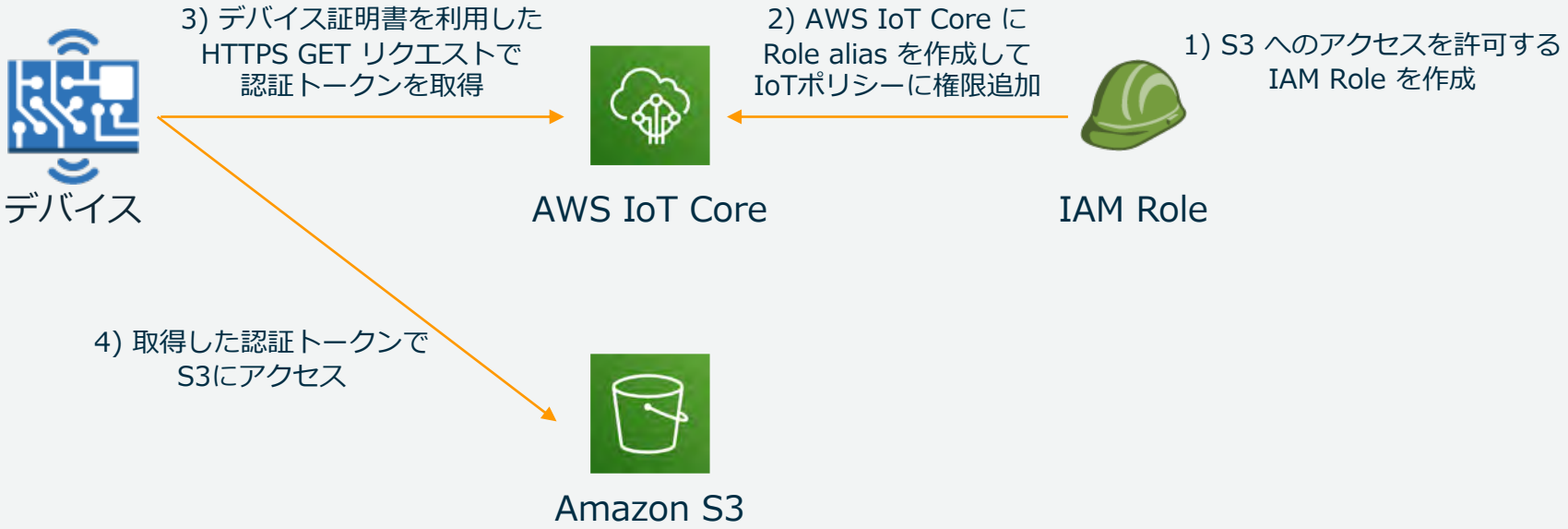



	AWS IoT Core	Amazon Kinesis Data Streams	Amazon Kinesis Video Streams	Amazon S3
ペイロード長	1メッセージあたり128KB	1レコードあたり1MB	MKV Fragment 50MB or 12.5MB/sec	1オブジェクトあたり5TB
プロトコル	MQTTS / MQTTS over WebSocket / HTTPS	HTTPS	HTTPS	HTTPS
料金 ※東京リージョン 2020年10月27日現在の料金です	5 kBを1メッセージとして $1.2/百万メッセージ	1シャード$0.0195/時 ペイロードユニットを25KBとして$0.0215/百万ペイロード(Streams)	インジェスト $0.01097/GB, コンシューム $0.01097/GB, データストア $0.025/GB/Month	PUT: $0.0047/千ファイル ストレージ $0.025/GB(最初の50TB, 月)
認証	デバイス証明書 / Cognito / SigV4 / カスタム認証	SigV4	SigV4	SigV4
使いどころ	ペイロードが小さい リアルタイム処理が必要 デバイスとクラウド間の双方向通信が必要 高いセキュリティを求められるデバイスが低スペック	ペイロードが大きい リアルタイム処理が必要 送信頻度が高い	ペイロードがビデオライクなデータであり、ストリームの形でアップロードする必要がある リアルタイム処理が必要	ペイロードがファイルやメディアなど非常に大きい 送信頻度が低い リアルタイム処理は不要

デバイス証明書を利用した認証トークンの取得

Kinesis や S3 といった SigV4 の認証トークンを必要とするサービスを利用するため、
AWS IoT Core ではデバイス証明書を使って認証トークンを取得する機能を提供

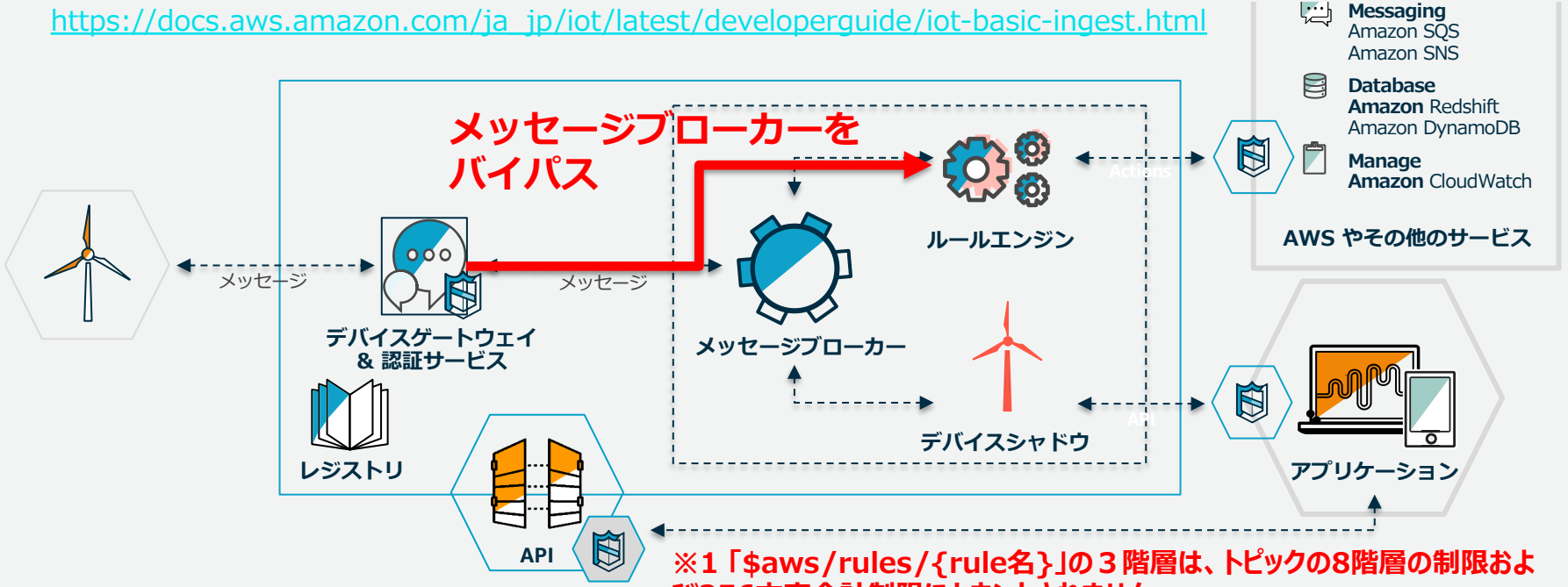
https://docs.aws.amazon.com/ja_jp/iot/latest/developerguide/authorizing-direct-aws.html



Basic Ingest によるメッセージングコストの削減

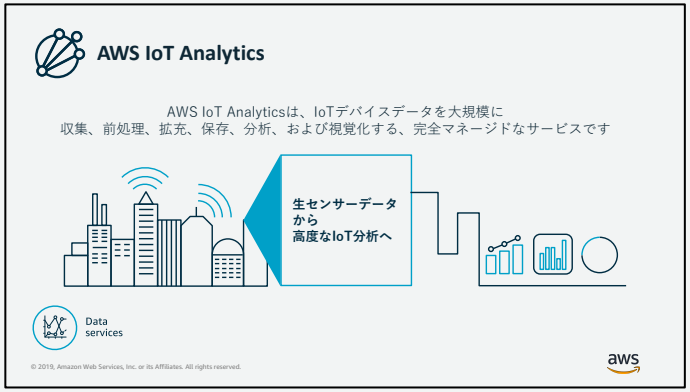
AWS IoT Core に publish するメッセージを扱うのがルールエンジンだけの場合、
Topic の先頭に「$aws/rules/{rule名} ※1 」を付けるとメッセージブローカーがバイパスされ、
メッセージングのコスト ($1.2/100万メッセージ ※2) がかからなくなります

https://docs.aws.amazon.com/ja_jp/iot/latest/developerguide/iot-basic-ingest.html



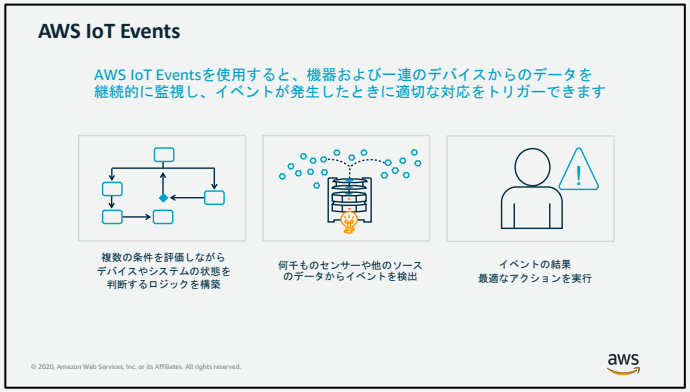
※1「$aws/rules/{rule名}」の３階層は、トピックの8階層の制限および256文字合計制限にカウントされません。
※2 東京リージョン 2020年10月27日現在の料金です

データ収集で活用いただけるその他の IoT サービス



AWS IoT Analytics

Black Belt Online Seminar はこちら
AWS IoT Analytics Dive Deep



AWS IoT Events

Black Belt Online Seminar はこちら
AWS IoT Events

本日のアジェンダ

- IoT のユースケースと要件
- AWS IoT Core の概要
- AWS IoT Core によるデータ収集
- AWS IoT Core による遠隔制御
- デバイスの運用・管理
- まとめ

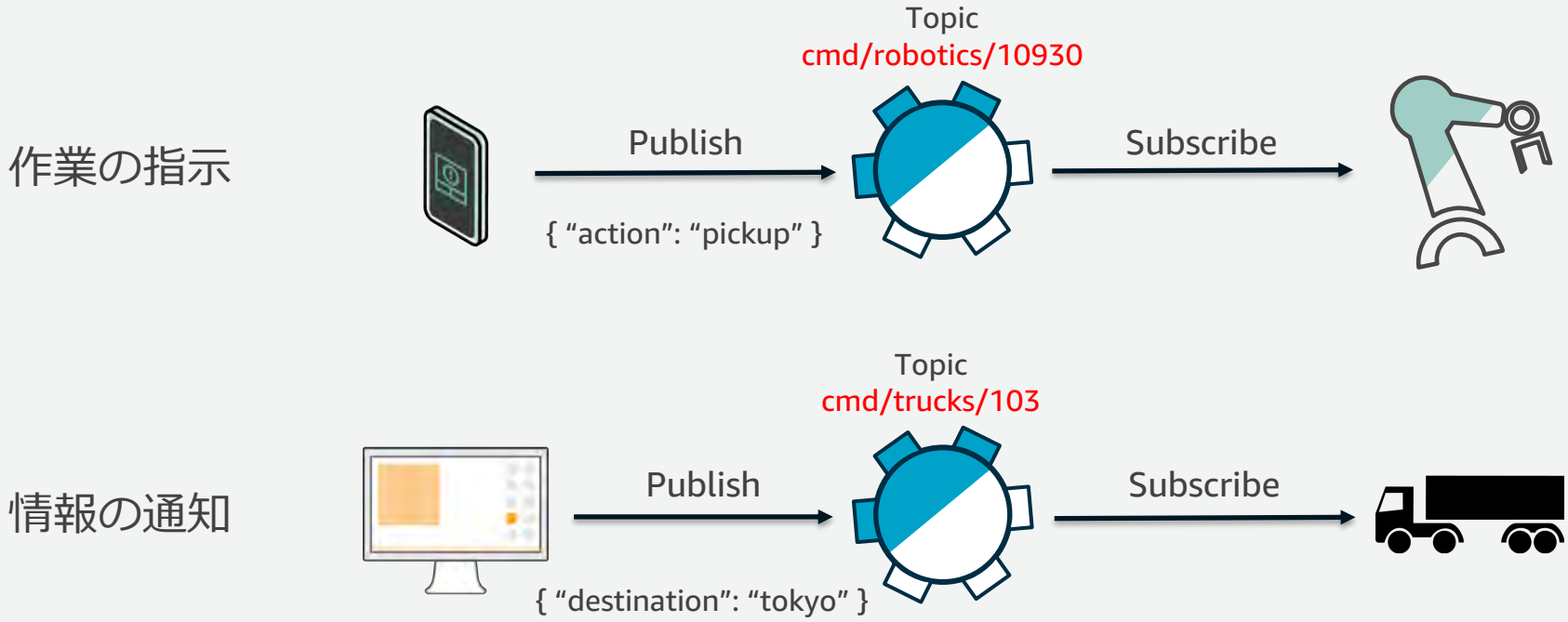
遠隔制御のパターン

① コマンドの送信
作業の指示
情報の通知など

② 状態の同期
照明のON/OFF
設定変更など

③ 一括制御
ソフトウェア更新
コンテンツ配信など

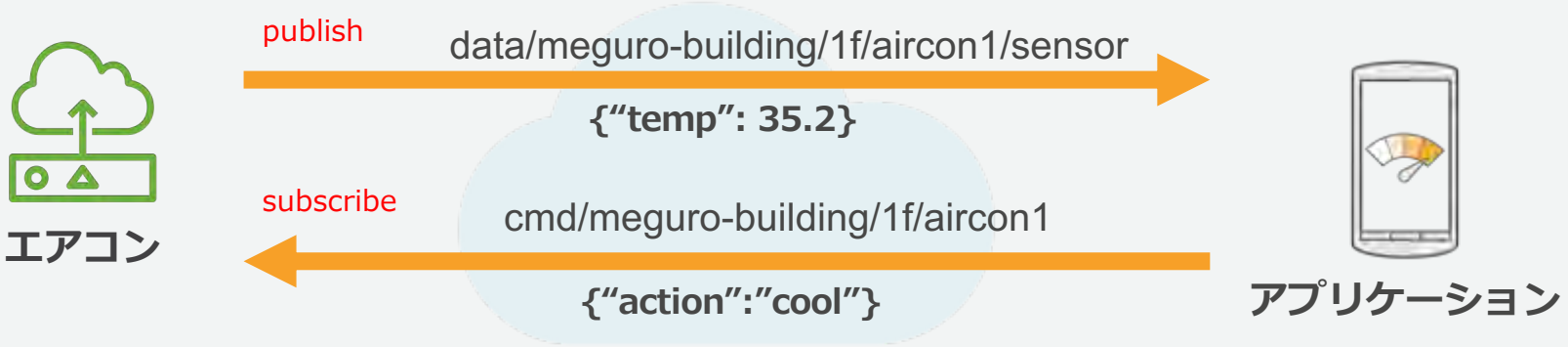
コマンドの送信



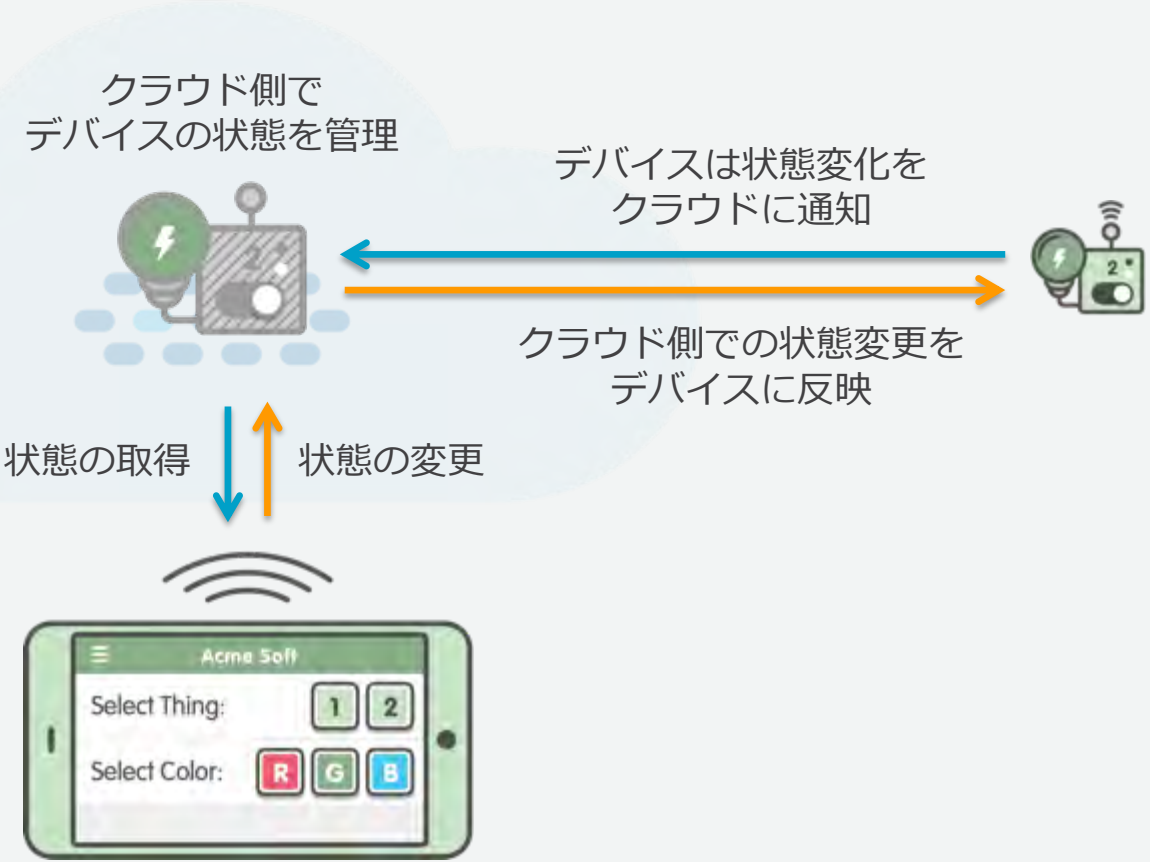
データ収集とコマンド送信を併用する場合の Topic 設計

データとコマンドの Topic を分ける

- デバイスからのデータ受信
  - 例: **data**/meguro-building/1f/aircon1/sensor
- デバイスへのコマンド送信
  - 例: **cmd**/meguro-building/1f/aircon1



状態の同期



デバイスシャドウ

いつでもあなたのデバイスの状態を把握し、管理する

デバイスの最後の既知の状態を報告します。
例えば「現在把握している電球の最後の色は赤です」など

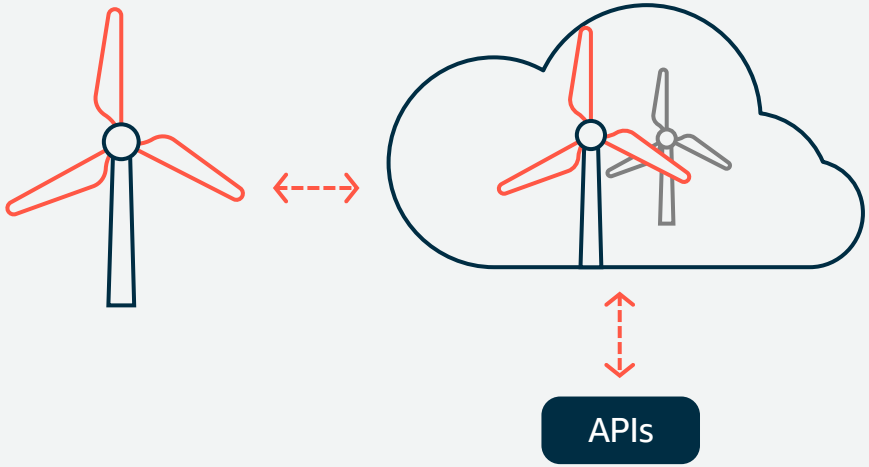
デバイスの状態を変更します。
例えば「電球の色を青に変える」など

MQTT を使用した状態変更のリアルタイム通知

オフラインデバイスとの非同期通信

デバイスへの簡単な実装のための AWS IoT Device SDK への統合

アプリケーションがデバイスと通信するための REST API



デバイスシャドウ

デバイス

現在の状態 (reported) をシャドウに通知
シャドウから状態変更内容 (delta) を取得

シャドウ

デバイスの現在の状態 (reported)、変更後の状態 (desired) 及び差分 (delta) をバージョン管理

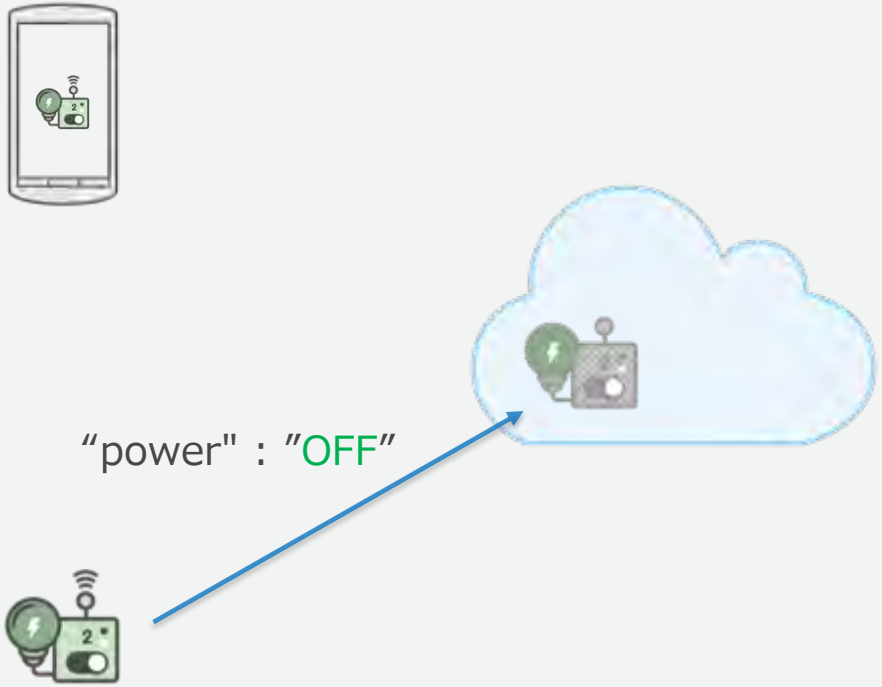
アプリ

シャドウから現在の状態 (reported) を取得
変更後の状態 (desired) をシャドウにセット

```
{
    "state" : {
      “desired" : {
          “power" : "ON"
      },
      "reported" : {
          “power" : ”OFF"
      },
      "delta" : {
          “power" : ”ON"
       }
    },
    "version" : 10
}
```

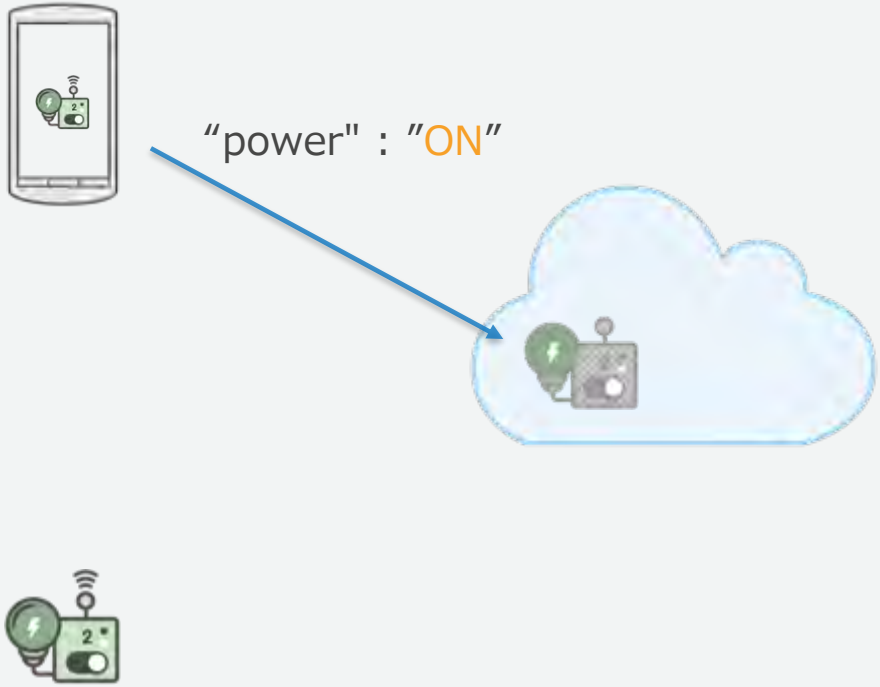

aws

デバイスが現在の状態を通知



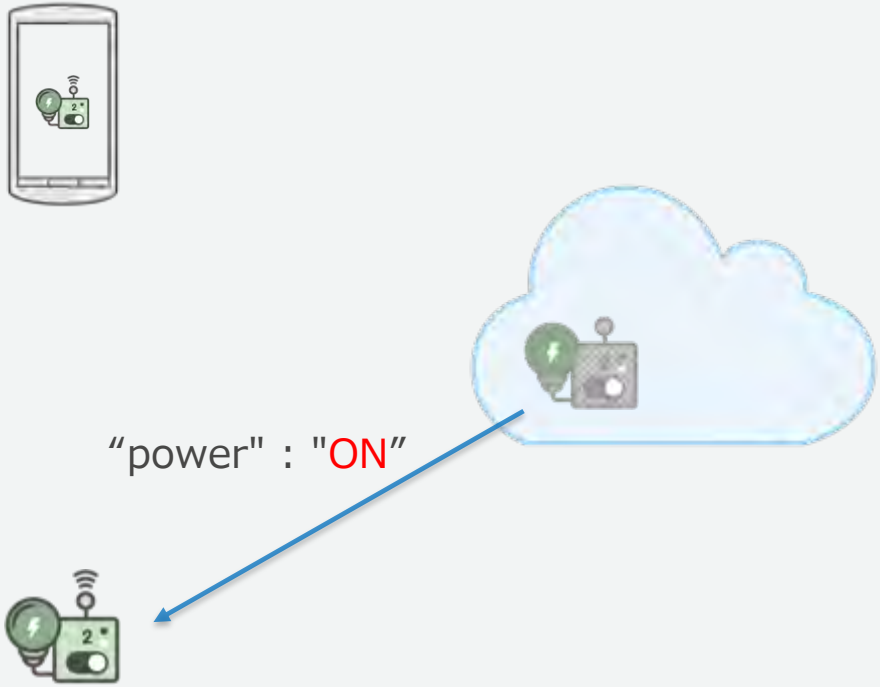

```
{
    "state" : {
       "reported" : {
           “power" : ”OFF"
       }
    },
    "version" : 10
}
```

アプリケーションから電源ON



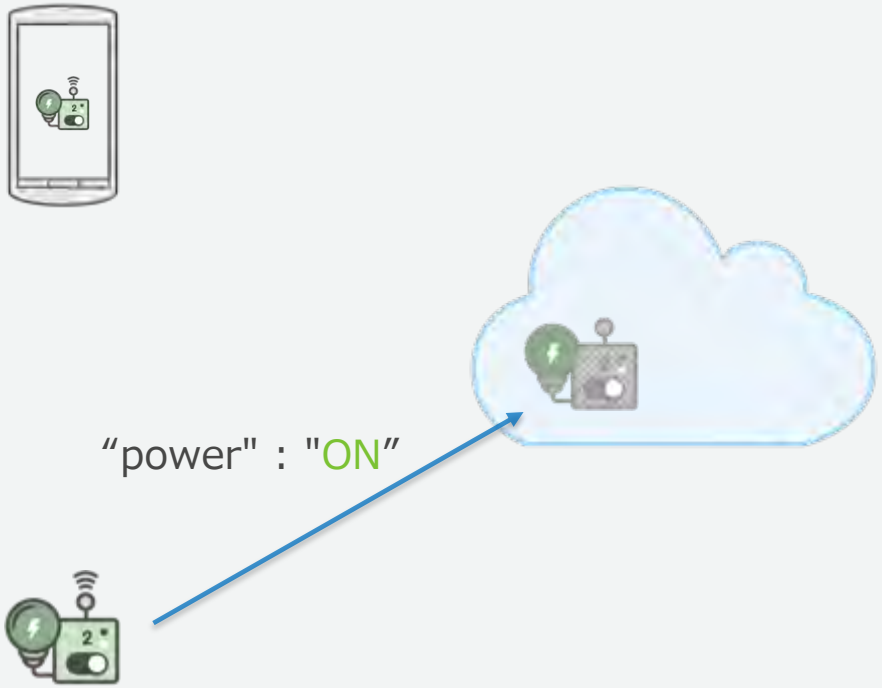

```
{
    "state" : {
      “desired" : {
          “power" : ”ON"
      },
      "reported" : {
          “power" : ”OFF"
      }
    },
    "version" : 11
}
```

デバイスに差分 (delta) が通知される




```
{
    "state" : {
       “desired" : {
            “power" : "ON"
       },
       "reported" : {
            “power" : ”OFF"
       },
       "delta" : {
            “power" : ”ON"
        }
    },
    "version" : 11
}
```

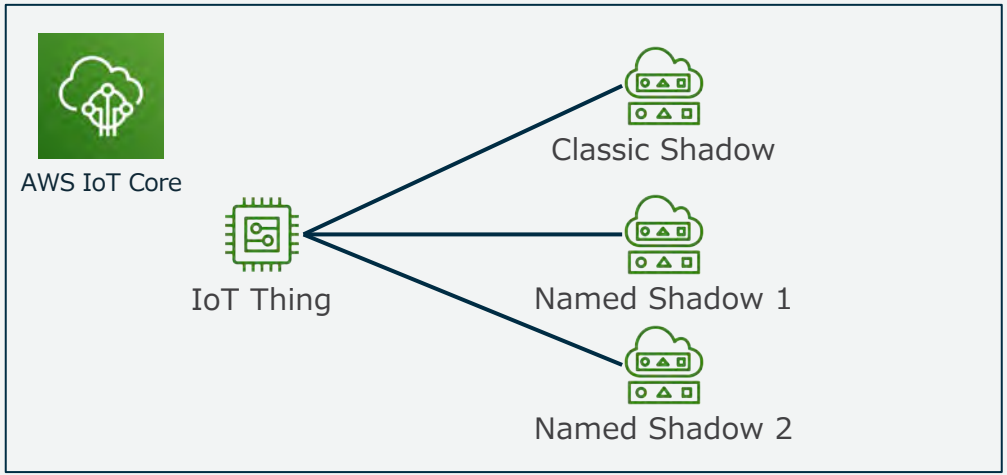
デバイスが変更後の状態を通知 (delta は消える)




```
{
    "state" : {
       “desired" : {
            “power" : "ON"
       },
       "reported" : {
            “power" : ”ON"
       },
       "delta" : {
            “power" : ”ON"
       }
    },
    "version" : 11
}
```

名前付きシャドウ [new!]

- 1つのデバイスシャドウに格納できるデータは最大8KB
- 用途に応じてシャドウを分割することで、各シャドウには必要な情報だけを記録し、シャドウのサイズを小さく保つことが可能



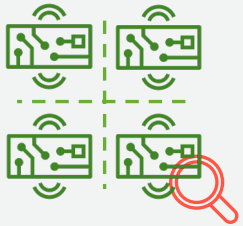
https://aws.amazon.com/jp/about-aws/whats-new/2020/07/aws-iot-core-now-supports-multiple-shadows-for-a-single-iot-device/

本日のアジェンダ

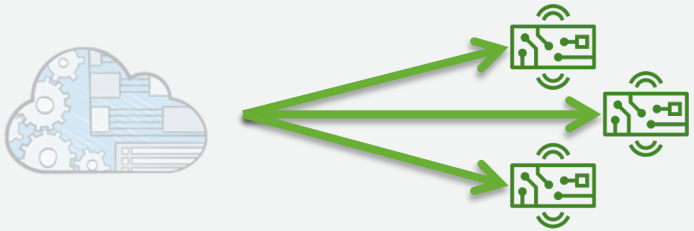
- IoT のユースケースと要件
- AWS IoT Core の概要
- AWS IoT Core によるデータ収集
- AWS IoT Core による遠隔制御
- デバイスの運用・管理
- まとめ

一括制御

例: OTA ソフトウェア更新



① 更新対象のデバイスを検索

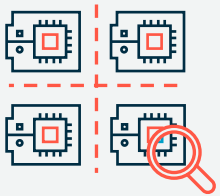


② 対象範囲を徐々に拡大しながら
更新指示を送信

AWS IoT Device Management

バッチ処理での プロビジョニング

数回のクリックで完了する迅速なデバイスの導入と設定



リアルタイムでの インデックス化と検索

全デバイスのステータスと状態を可視化

デバイスロギング とモニタリング

デバイスログを収集して問題をすばやく特定して修正

Jobの実行

グループに登録されているデバイスへジョブを実行し管理する

セキュアトンネリング

ファイアーウォール越しのデバイスにセキュアにアクセス

Connectivity & Control Services

aws

デバイスの検索とグルーピング

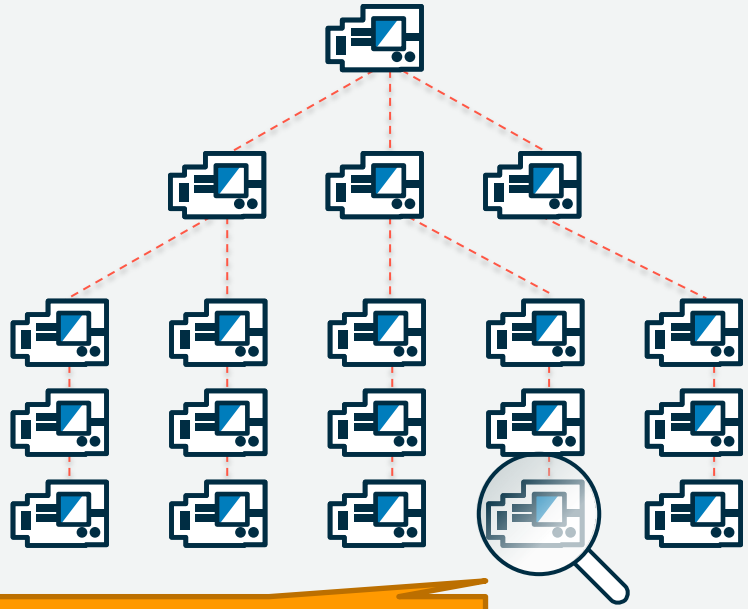
レジストリとシャドウに対する検索

- 静的な属性 (Attribute)
- 動的な状態 (Shadow)
- 接続の状態 (Connectivity, オンライン or オフライン)

の組み合わせでデバイスを検索

検索結果をグループ化して管理

- 検索結果のデバイスフリートをグループ化して管理できる
- グループに対し、Job を実行できる
- 検索条件を満たした/満たさなくなった際、自動的にグループに追加/削除することも可能 (Dynamic Group)



Jobs

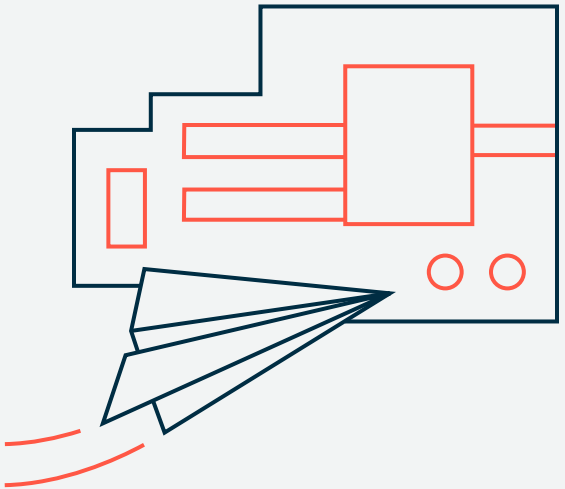
グループに登録されているデバイスでジョブを実行し管理する

実行したジョブの各デバイスのステータスを受信し、可視化します

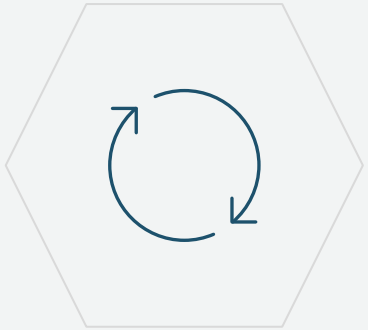
グループもしくは特定の１デバイスに対して実行可能

展開速度を制御し、障害基準を設定することによって、不具合の範囲を最小限に留めることも可能

ジョブに署名をすることで、不正なジョブの実行を防ぎます

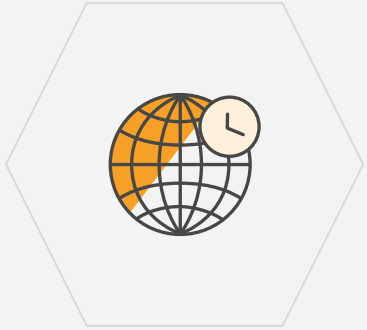


Job の種類



Continuous

- Job を作成した後も、**配信対象となるグループにデバイスが追加される**と Job 通知が行われる



Snapshot

- Job を作成した時に対象グループに所属していたデバイスが対象
- 一度作ると Job 通知の対象は変更されない

Job のコマンド例

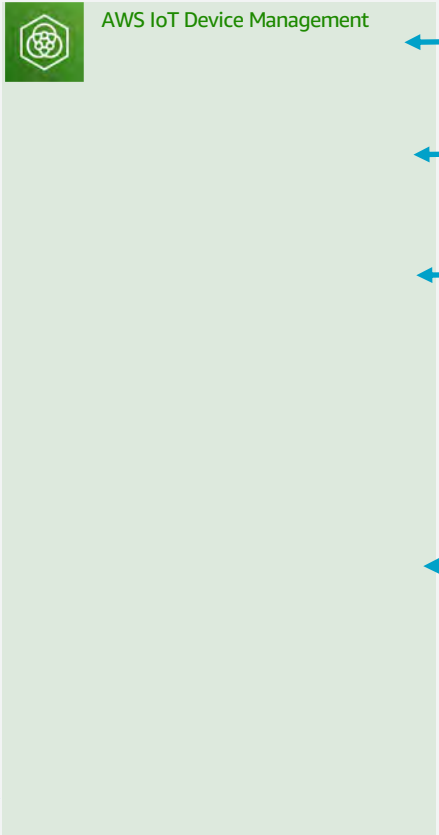
コマンド送信

```
{
    "operation": "sys-info",
    "sys-info": "uptime",
    "topic": "sys/info"
}
```

コマンドとデータの送信 (署名済み Amazon S3 URL に置き換え)

```
{
    "operation": ”install",
    ”data": "${aws:iot:s3-presigned-url:https://s3.amazonaws.com/job-test-bucket/firmware}"
}
```

Job を利用する流れ



(1) 条件を指定して対象を検索
(shadow.reported.firmware <= 1.0.0)

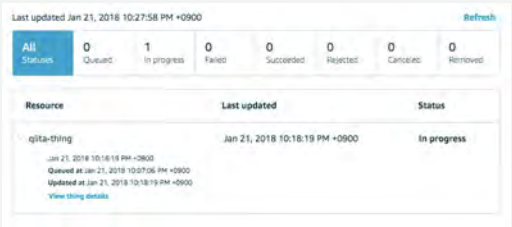
(2) 検索結果からGroup作成

(3) (2)で作成した
Group向けにJob生成

(4)Job 通知

(5)Job status通知

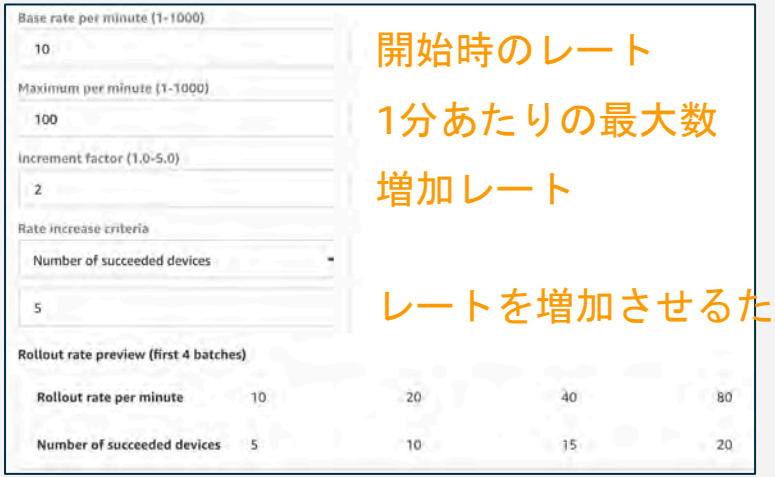
統計用API



(6)完了通知

Job のロールアウト

- 通常だと1～1000個/分 の速度でデバイスに対して Job が配信される
- Exponential rateを指定することで
  - 一定の増加量でデバイスに Job を配信できる
  - 一定の倍率で増加量を増やしデバイスに Job を配信できる
  - 増加量を増やす条件を指定できる



Job の中止設定

- 中止条件を設定することで、カナリアデプロイのようなことが出来る
  - 中止理由 (Failed, Rejected, Time Out, All)
    - Time Outは1分～7日の期間を指定でき、in_progressになってからその期間を経過するとTime Outとなる
  - 中止理由に該当するデバイスのパーセンテージ
  - 中止するまでに Job を実行するデバイスの最低数

aws

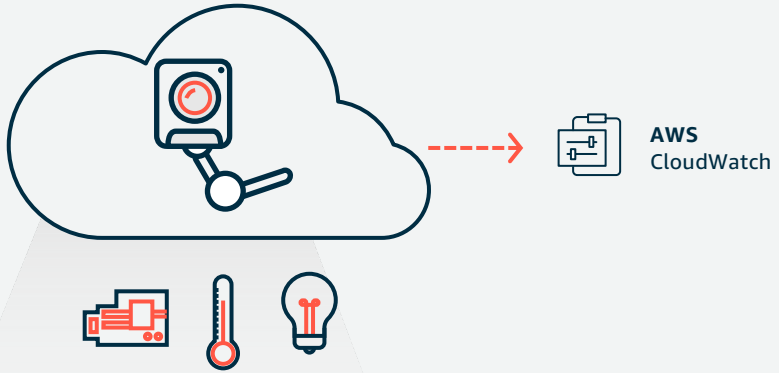
きめ細やかなデバイスのロギングとモニタリング

デバイスログを収集して問題を特定し修復する

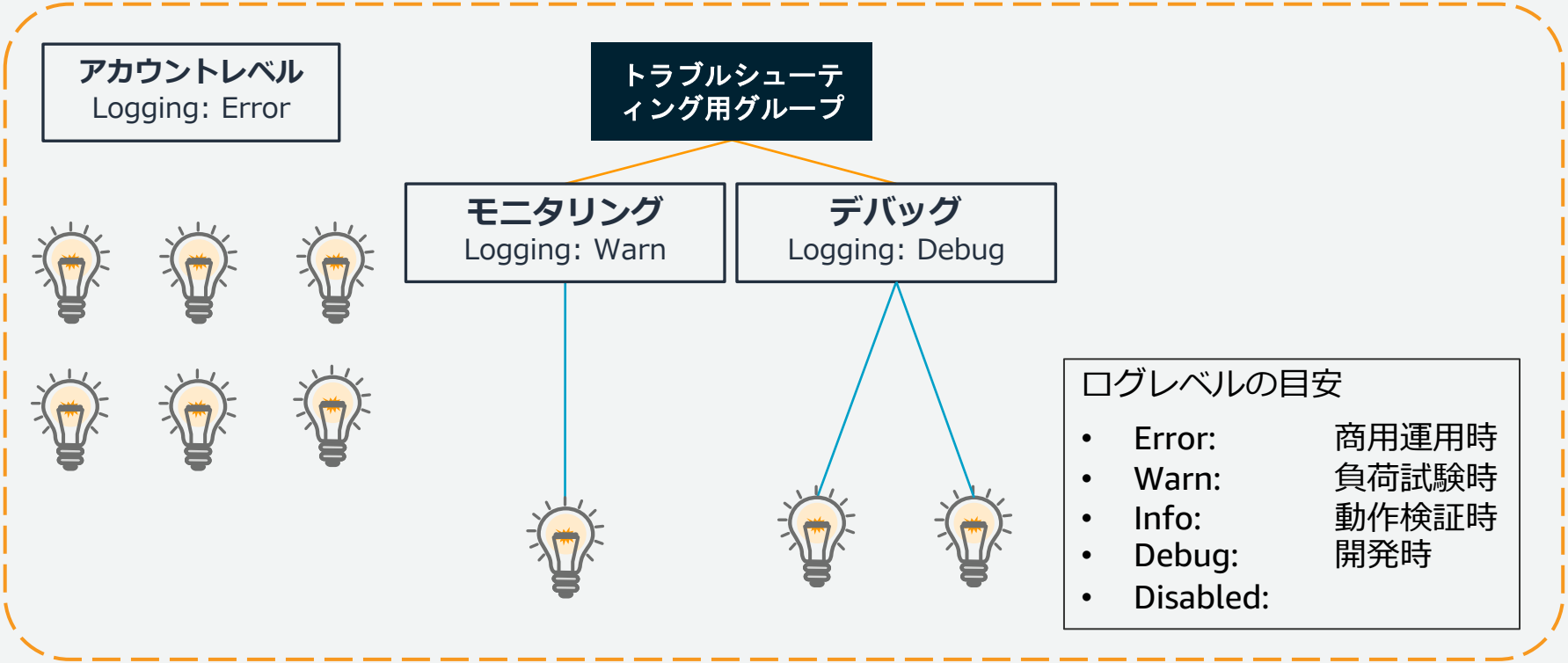
デバイスまたは Thing Group に対して CloudWatch Logs のログレベルを設定

トラブルシューティングのため、誤動作しているデバイスのログレベルを選択的に上げる事が可能

AWS CloudWatchを使用してアラームを設定し、ログを検索

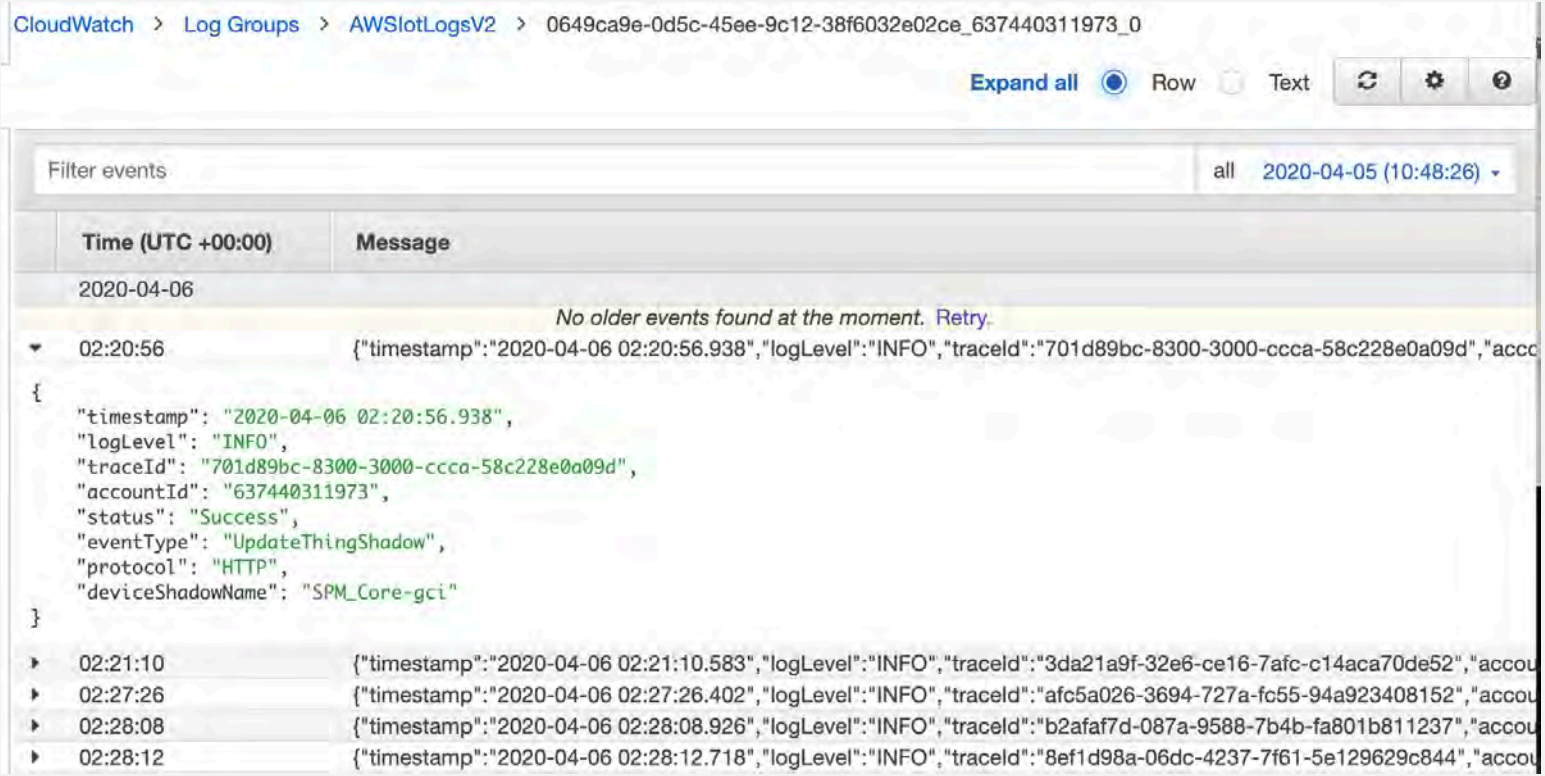


Thing Group に対するログレベルの設定



AWS IoT Core における CloudWatch Logs の有効化

CloudWatch Logs で AWS IoT Core のログを参照

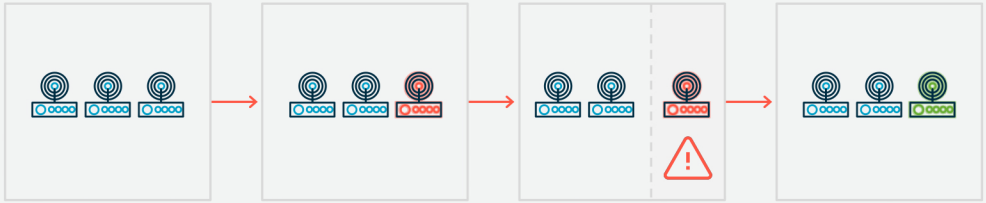


https://docs.aws.amazon.com/ja_jp/iot/latest/developerguide/cloud-watch-logs.html

デバイス管理で活用いただけるその他のサービス

AWS IoT Device Defender

AWS IoT Device Defenderは、フル・マネージドのIoTセキュリティ
サービスであり、接続されたデバイス群を継続的に保護することができます



Connectivity
& Control
Services

ハンズオンはこちら

AWS IoT Device Defender ハンズオン

本日のアジェンダ

- IoT のユースケースと要件
- AWS IoT Core の概要
- AWS IoT Core によるデータ収集
- AWS IoT Core による遠隔制御
- デバイスの運用・管理
- まとめ

まとめ

- AWS では、デバイスとの通信からデータの活用まで、IoT で必要となる要件を満たすサービスを提供している
- AWS IoT Core では、デバイスと安全に通信し、デバイスに与える権限をデバイスやグループ単位で管理できる
- AWS IoT Core は従量課金のフルマネージドサービスであるため、デバイス1台でのスモールスタートから、大量のデバイスを運用する商用サービスまでスケール可能

AWS IoT 関連リンク

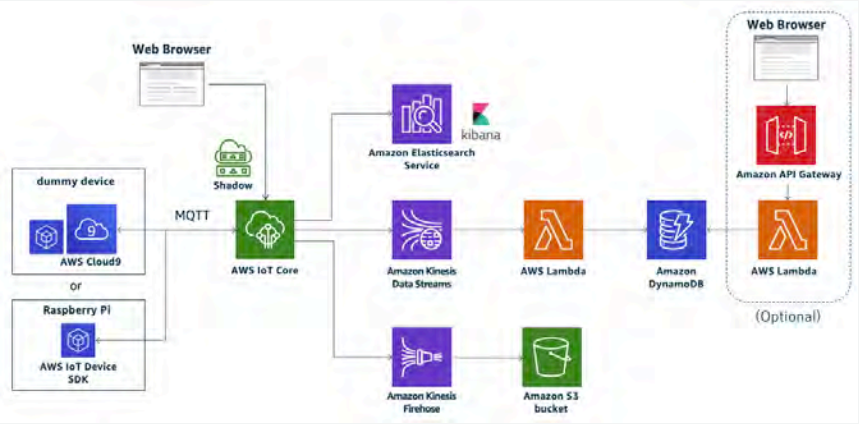
- AWS IoT 開発者ポータル
  - AWS の IoT に関する最新情報を発信しています
  - https://aws.amazon.com/jp/local/iot/

- AWS IoT サービスのオンラインハンズオン
  - AWS を使った IoT 開発を実際にお試し頂けます
  - https://aws.amazon.com/jp/blogs/news/tag/iot-workshop/

- IoT@Loft – IoT デベロッパー向けのオンラインイベント
  - IoT を使った開発事例やノウハウを共有するイベントです
  - https://aws.amazon.com/jp/start-ups/loft/tokyo/iot-loft/

ハンズオンコンテンツのご紹介

AWS IoT Core 初級ハンズオン
https://aws-iot-core-for-beginners.workshop.aws/



AWS IoT Device Management Workshop
https://iot-device-management.workshop.aws/

Q&A

お答えできなかったご質問については
AWS Japan Blog 「https://aws.amazon.com/jp/blogs/news/」にて
後日掲載します。

AWS の日本語資料の場所「AWS 資料」で検索

https://amzn.to/JPArchive

AWS Well-Architected 個別技術相談会

毎週”W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に対策などを相談することも可能
- 申込みはイベント告知サイトから

(https://aws.amazon.com/jp/about-aws/events/)

AWS イベント で[検索]

ご視聴ありがとうございました

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive