[AWS Black Belt Online Seminar]
AWS IoT Greengrass

サービスカットシリーズ

Prototyping Solutions Architect 市川 純

2020/12/15

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

過去資料
https://amzn.to/JPArchive

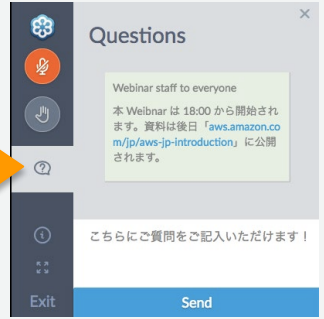
AWS Black Belt Online Seminar とは

「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12月15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

名前：

市川 純（いちかわ じゅん）

所属：

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

プロトタイピング ソリューション アーキテクト

担当：

IoTに関係するプロトタイピングやお客様の支援

前置き

※ 本セッションでは、AWS IoT CoreやMQTTプロトコルについての基本的な話については取り扱いません。

AWS IoT Coreのサービス、MQTTプロトコルについて詳しく知りたい場合は、2020/10にBlackbeltで取り上げていますので、そちらも合わせてご覧ください。

動画：

https://youtu.be/vs9i_yOhYCA

資料：

https://www.slideshare.net/AmazonWebServicesJapan/20201027-aws-black-belt-online-seminar-aws-iot-core

本日のアジェンダ

- AWS IoT Greengrassとは
- AWS IoT Greengrassの始め方
- AWS IoT Greengrassのユースケース
- まとめ

本日のアジェンダ

- AWS IoT Greengrassとは
- AWS IoT Greengrassの始め方
- AWS IoT Greengrassのユースケース
- まとめ

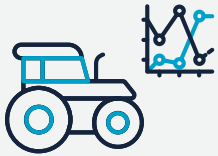
IoTのユースケース

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

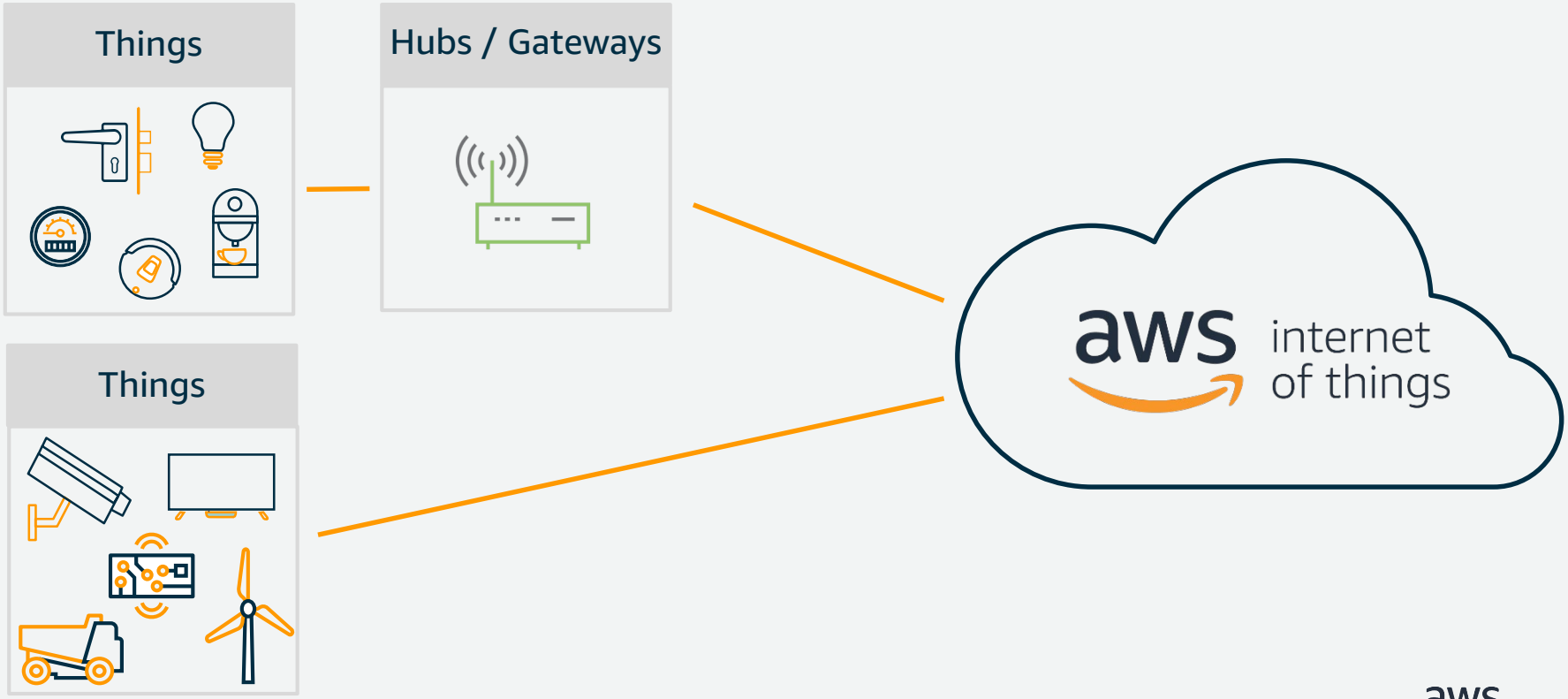
エネルギー資源を
効率的に管理

コネクテッドカー、
自動運転で輸送の変革

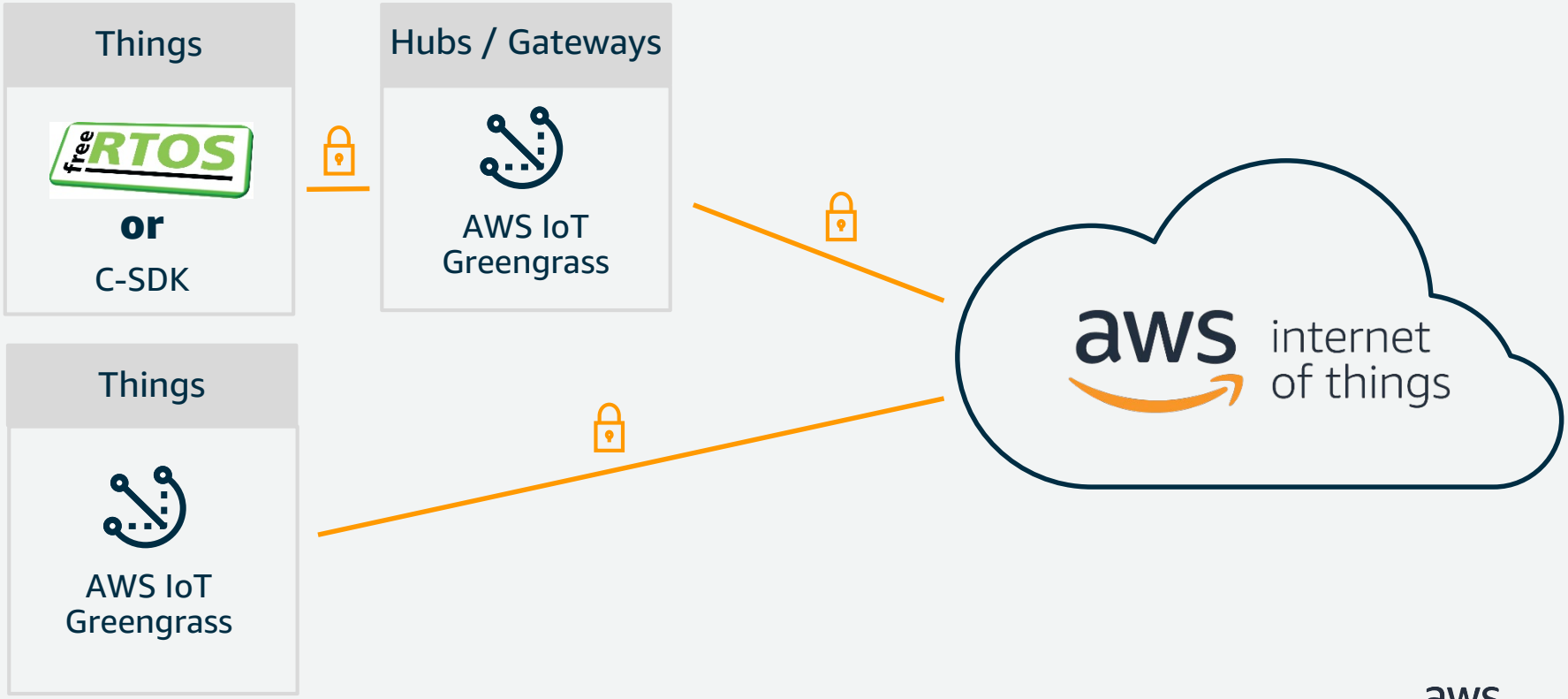
家庭、オフィス、工場
の安全性向上

aws

すでに多くのIoTデバイスがクラウドに繋がっている

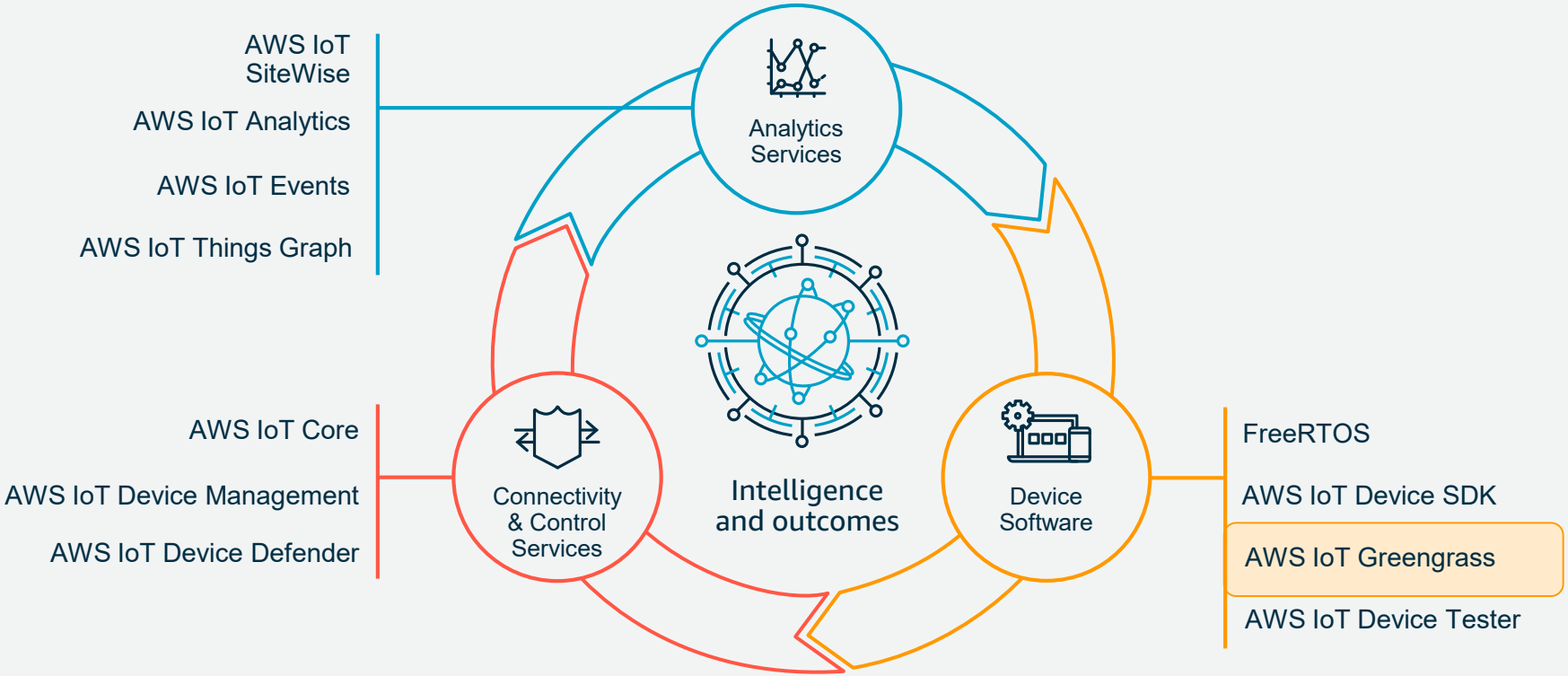


すでに多くのIoTデバイスがクラウドに繋がっている

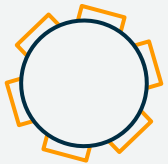


エッジに機能を持っていきたい理由

AWS IoTのサービス



AWS IoT Greengrassの基本コンポーネント

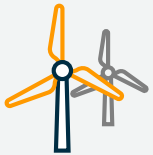


ローカルメッセージとトリガー

クラウドとの接続を必要としないメッセージング

ローカルアクション

AWS Lambdaを使いアプリケーションの開発を効率化



データと状態の同期

オフライン状態で発生した出来事を、オンラインになったら同期

セキュリティー

クラウドとデバイス間の相互認証と承認

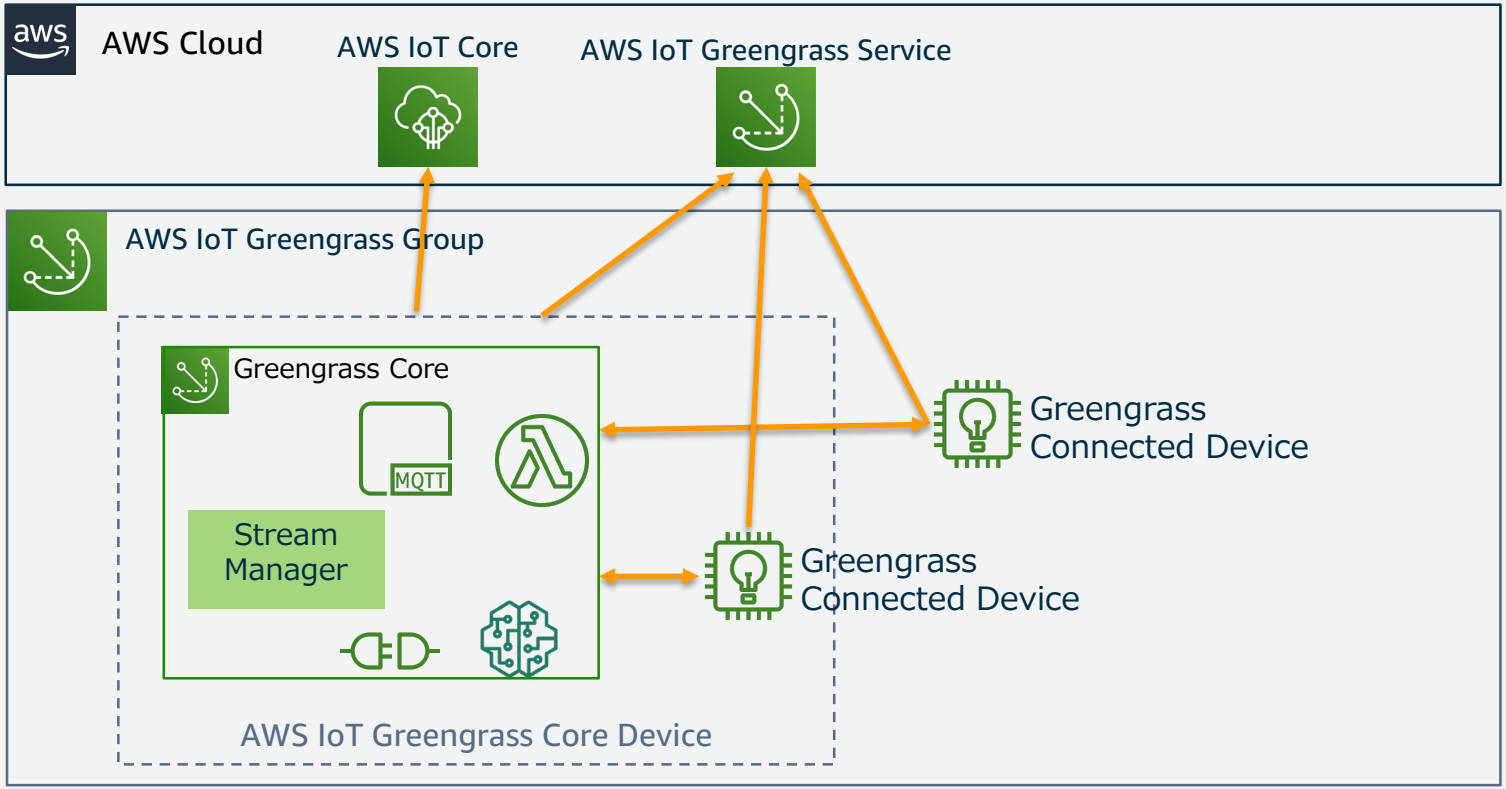
ローカルリソースアクセス

AWS Lambdaからデバイス上のリソースに対してアクセスが可能

Over the Air アップデート

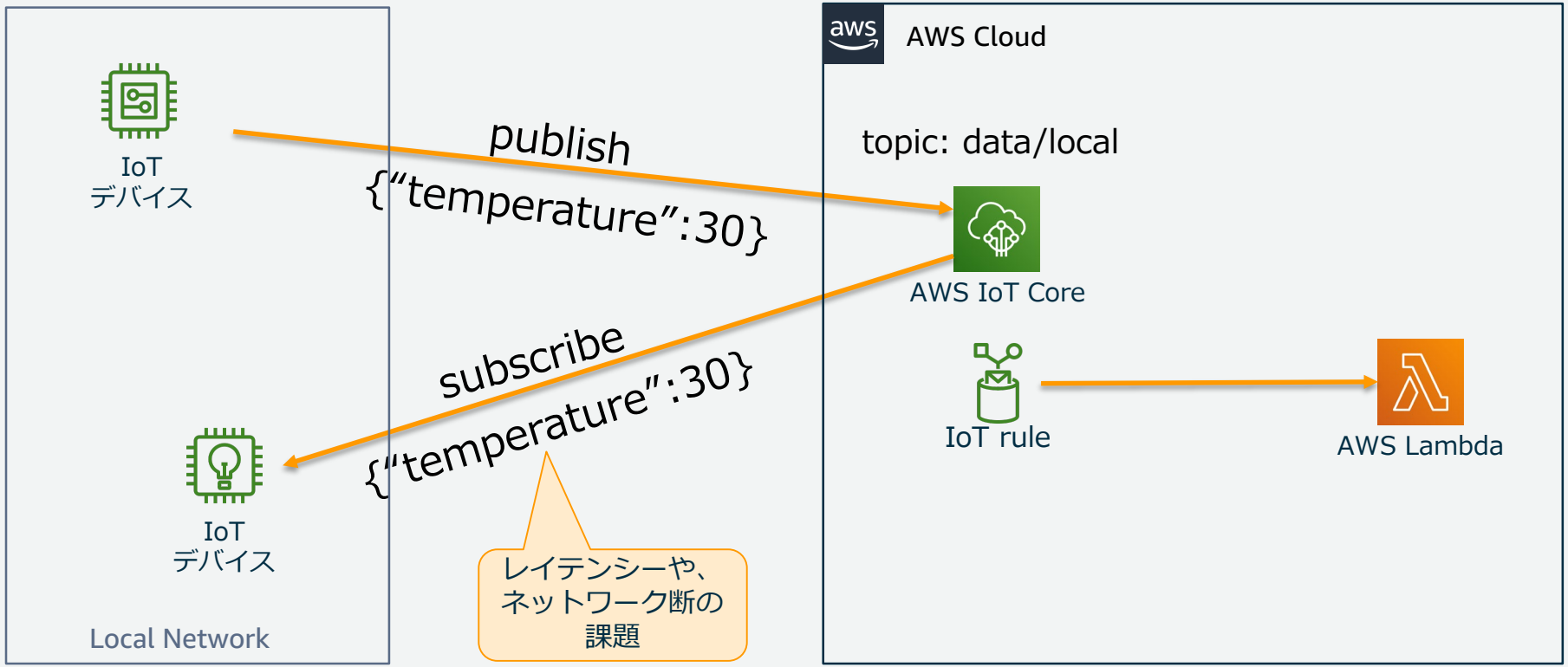
AWS IoT Greengrass Coreを簡単にアップデート

AWS IoT Greengrassサービスとgroupとcoreの関係



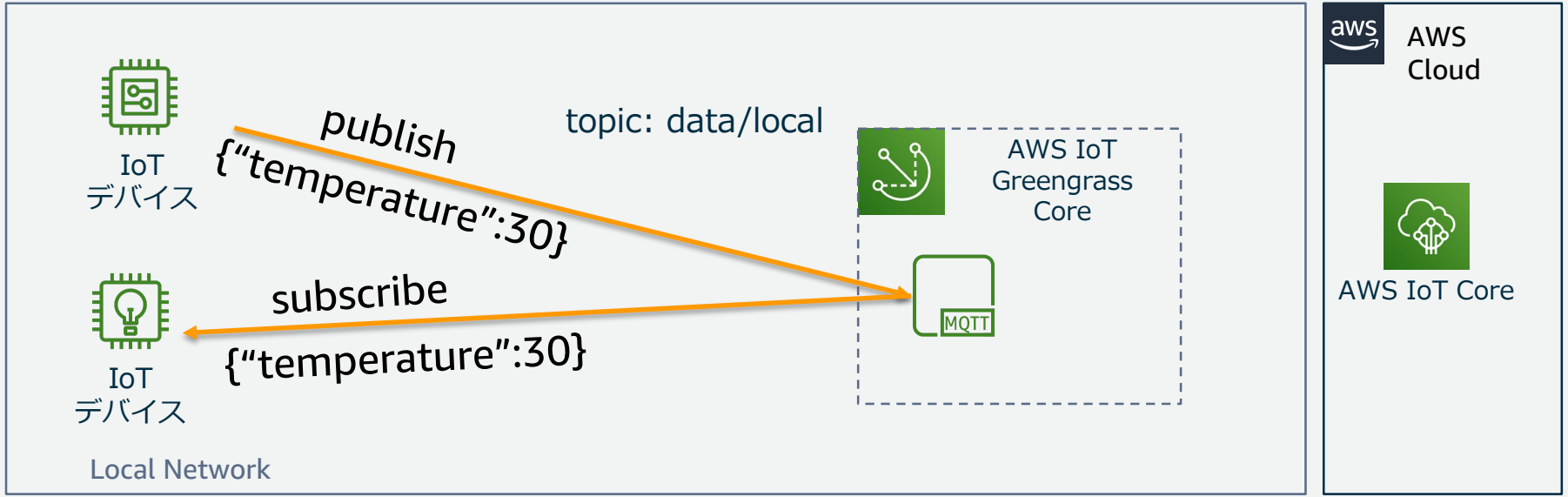
ローカルメッセージとトリガー

AWS IoT Coreを経由する場合



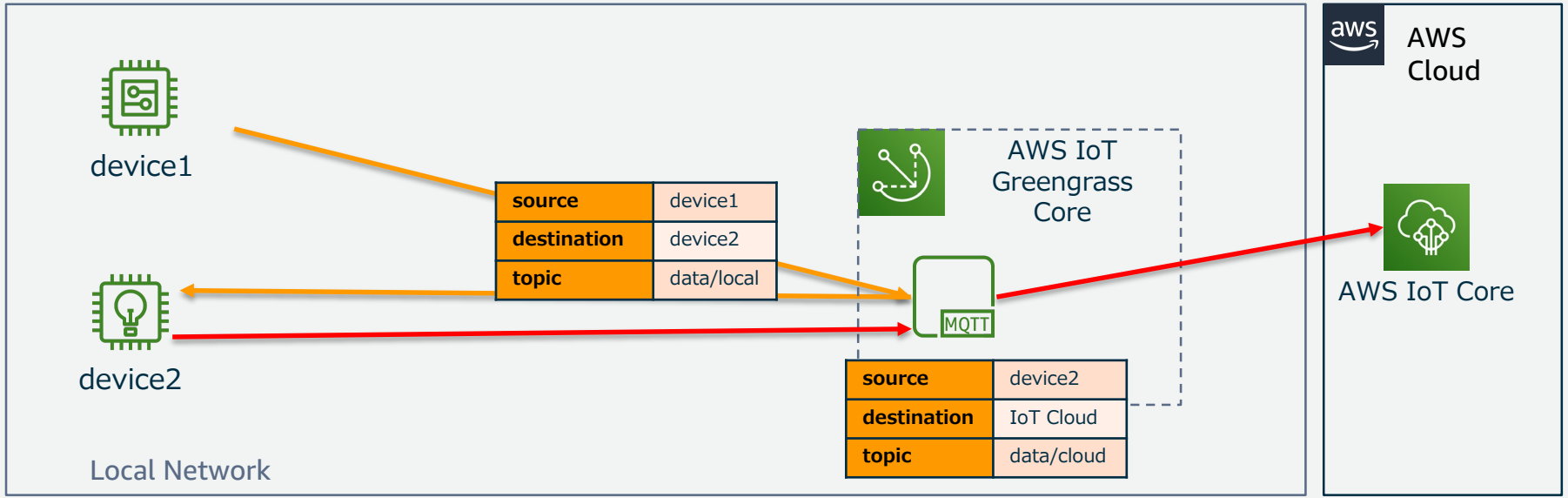
ローカルメッセージとトリガー

クラウドとの接続なしに通信可能



AWS IoT Greengrass Coreが提供するMQTTブローカーを介して、デバイス間の通信が可能になり、クラウドとの接続が切れるような環境でも動作可能

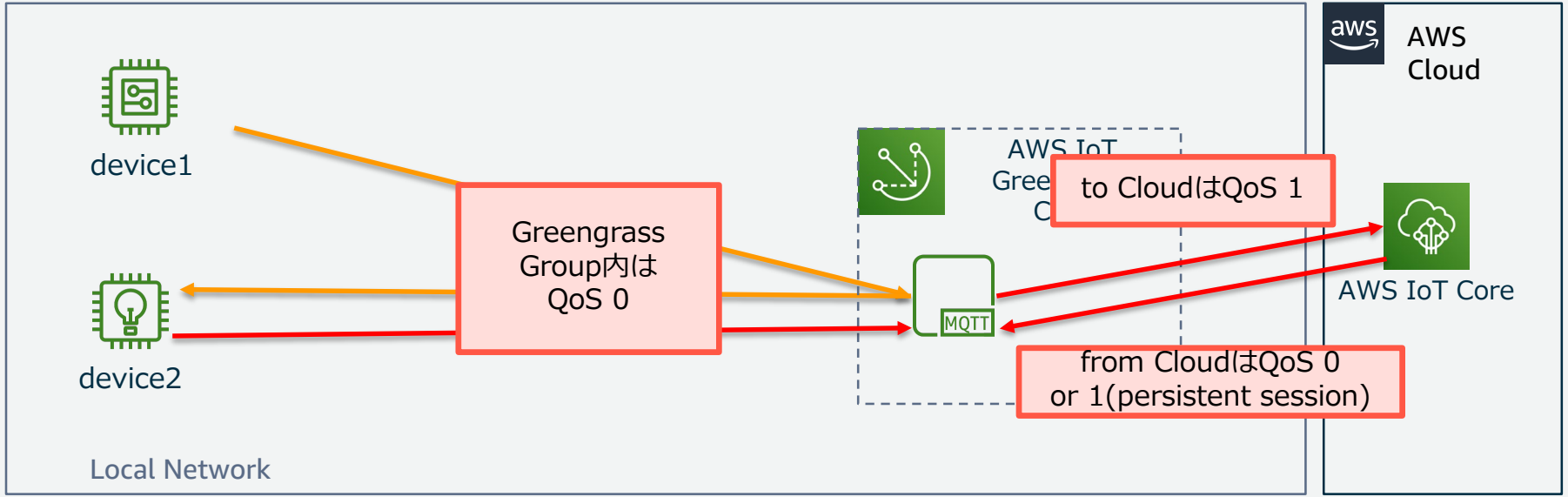
ローカルメッセージとトリガー
サブスクリプション



サブスクリプションの設定によって、どのMQTT Topicに届いたメッセージを、誰に送るかを設定する

ローカルメッセージとトリガー

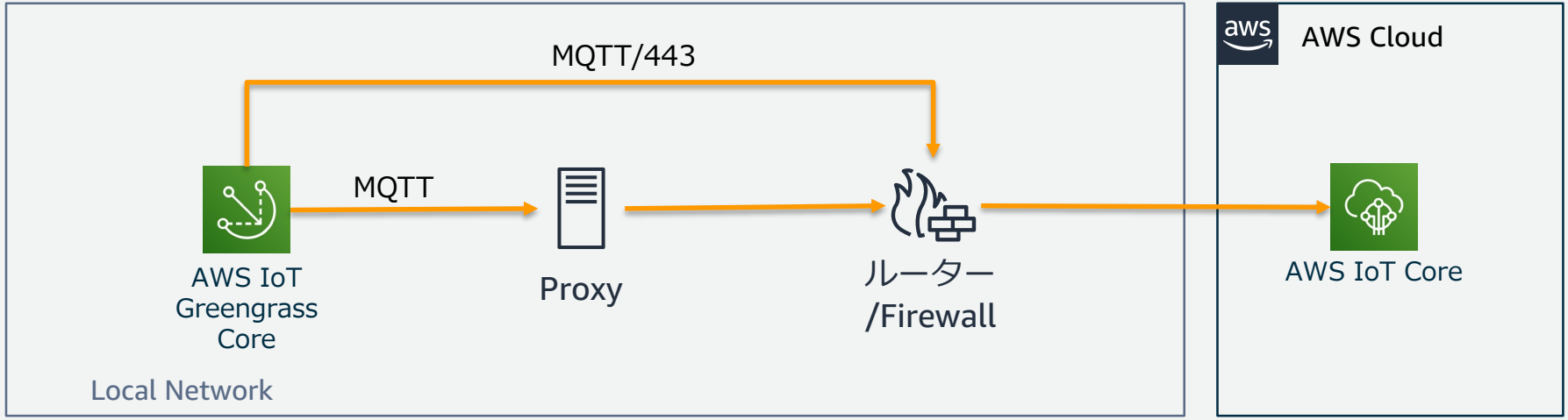
QoSの扱い



QoS 0: At most once
QoS 1: At least once
https://docs.aws.amazon.com/greengrass/latest/developerguide/gg-core.html#message-quality-of-service

ローカルメッセージとトリガー

443ポートやネットワークプロキシの対応



- ALPNにより、443ポートを利用したMQTT通信が可能
- HTTP/HTTPS Proxyに対応(basic認証も対応)
  - ただし、OTAはProxy非対応

https://docs.aws.amazon.com/greengrass/latest/developerguide/gg-core.html#alpn-network-proxy

クラウド転送時のオフラインキャッシュ

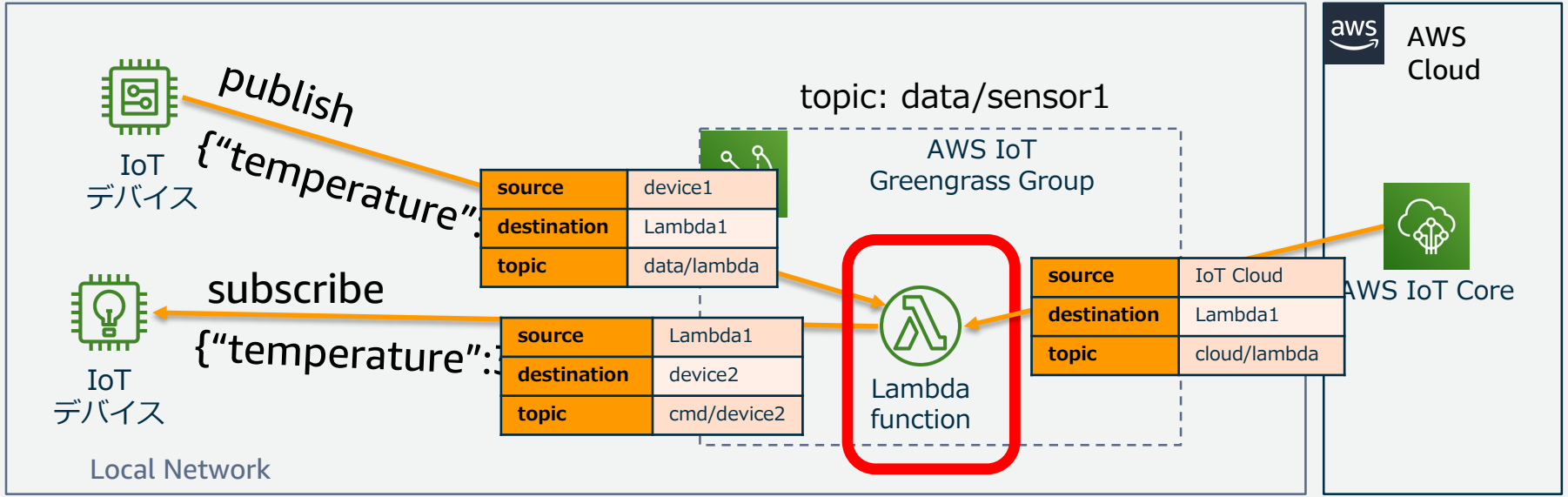
- AWS IoT Greengrassでは、デバイスがオフライン状態になった時に、MQTTメッセージの宛先がクラウドだった場合、メッセージキューにキャッシュします
- デバイスがオンラインになったら、メッセージキューからデータを送信します
- キャッシュはFileまたはMemoryを指定可能
- メッセージキューはFIFO
- キャッシュのサイズ(デフォルト2.5MB)は変更可能で、キャッシュが一杯になったら、古いものから破棄される

https://docs.aws.amazon.com/greengrass/latest/developerguide/gg-core.html#mqtt-message-queue

aws

ローカルアクション

ローカルのLambdaで組み込みソフトウエアの開発を簡素化



Lambda Functionでアプリケーションを作成し、MQTTのメッセージをトリガーに実行することが可能。クラウド側からのメッセージでも起動が可能。

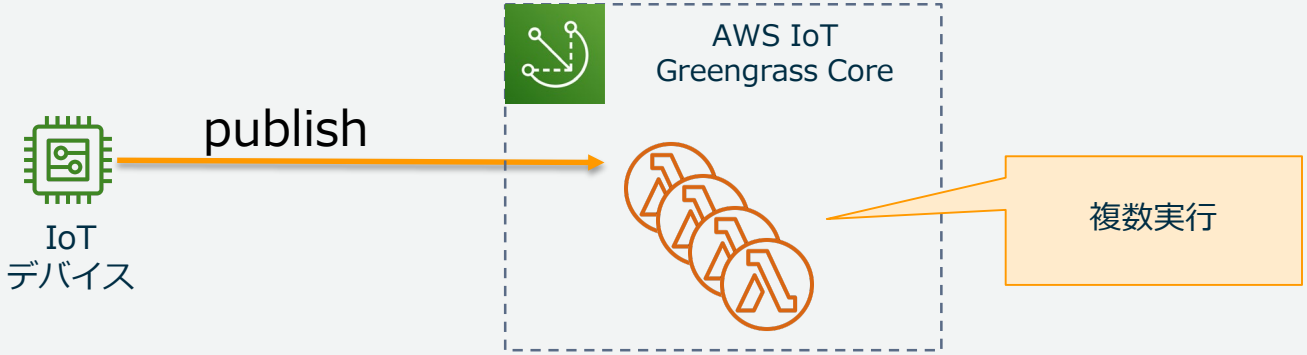
Greengrass Lambda の特徴

	On Cloud	On AWS IoT Greengrass
実行環境	AWS上	AWS IoT Greengrass Core(GGC) がインストールされているデバイス上
プログラミング言語	Node.js Java Python .NET Core and more…	Node.js: v12.x (*3 v6.10, v8.10) Python: 2.7、3.7、3.8 Java: 8 C, C++
イベントソース	S3, DynamoDB, Kinesis, …(*1)	MQTT メッセージ、他のLambdaから実行
タイムアウト設定	最大 900s	デフォルト3秒、または上限なしで常駐させることも可
メモリ設定	最小 128MB 最大 10GB	デフォルト16M、上限はデバイスに依存
課金	実行回数と実行時間、割り当てたメモリ量に応じて課金	無料(*2)

*1: https://docs.aws.amazon.com/ja_jp/lambda/latest/dg/invoking-lambda-function.html
*2: AWS IoT Greengrass 自体の料金は別途発生します
*3: https://docs.aws.amazon.com/greengrass/latest/developerguide/lambda-functions.html

Greengrass Lambdaの特徴

オンデマンドLambda



- メッセージによってLambdaが起動し、ハンドラにメッセージが渡る
  - 古いコンテナを再利用する可能性あり
- メッセージの処理が間に合わないと必要なLambdaを起動
  - 設定値(maxWorkItemCount)を超えるもしくは、デバイスのリソースが足りなくなった場合は起動できません
- 処理可能なメッセージ量を正しく設計する必要あり

https://docs.aws.amazon.com/greengrass/latest/developerguide/gg-core.html

aws

Greengrass Lambdaの特徴

Long Lived Lambda

- Greengrass Core起動時にLambdaも起動
  - 起動時にループ処理を呼び出すことで定期的な処理も可能
- メッセージが届くとハンドラにメッセージを渡す
  - ループ処理している場合は受け取れない
- ハンドラの処理が終わっていないと、新しいメッセージはキューに溜まる。キューが溢れるとエラーになる
- メッセージが溜まらないように処理できるメッセージ量や処理時間を設計する必要あり

https://docs.aws.amazon.com/greengrass/latest/developerguide/lambda-functions.html#lambda-lifecycle

aws

Greengrass Lambda 用のSDK

AWS IoT Greengrassでは、Java, Python, Node.js, Cの各言語向けにGreengrass SDKを提供しています。

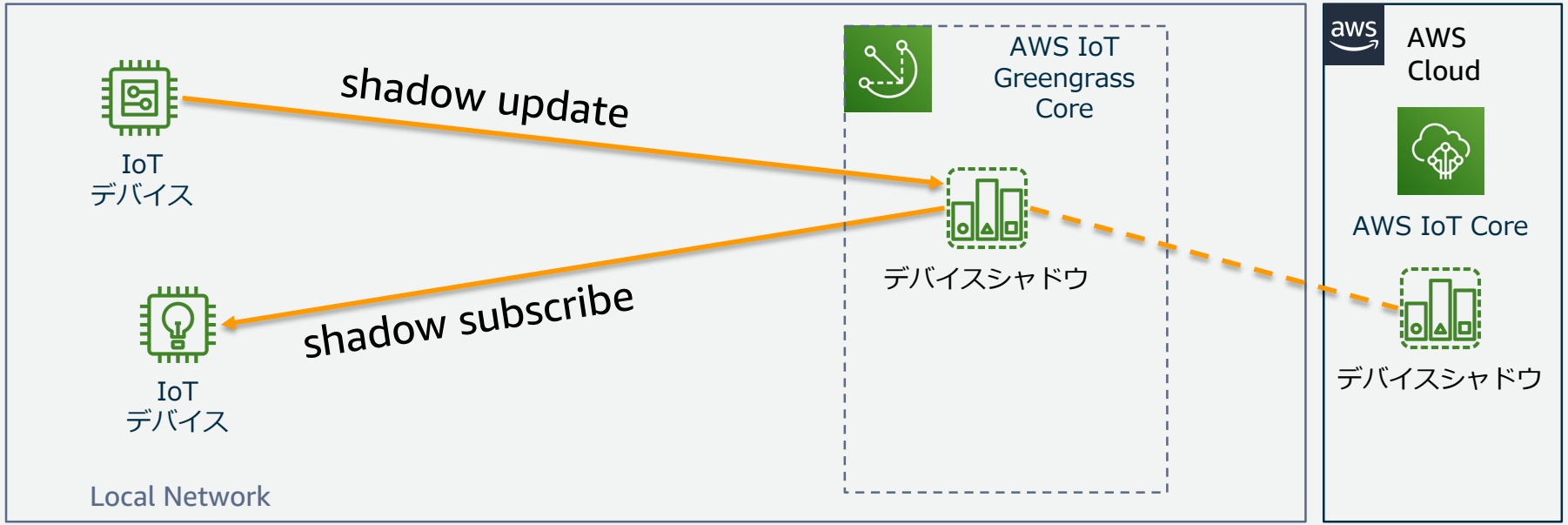
- AWS IoT Greengrass Core SDK for JavaScript
- AWS IoT Greengrass Core Python SDK
- AWS IoT Greengrass Core SDK for Java
- AWS IoT Greengrass Core C SDK

これらのSDKではAWS IoT Greengrassで提供している、MQTT Publish、Stream Manager、Secret Manager等の機能を利用するためのクライアントを提供しています。

(言語によって提供されない機能はありますので、詳細は各SDKのページを参照してください)

データの状態の同期(Device Shadow)

Device Shadowをローカル、クラウドと同期

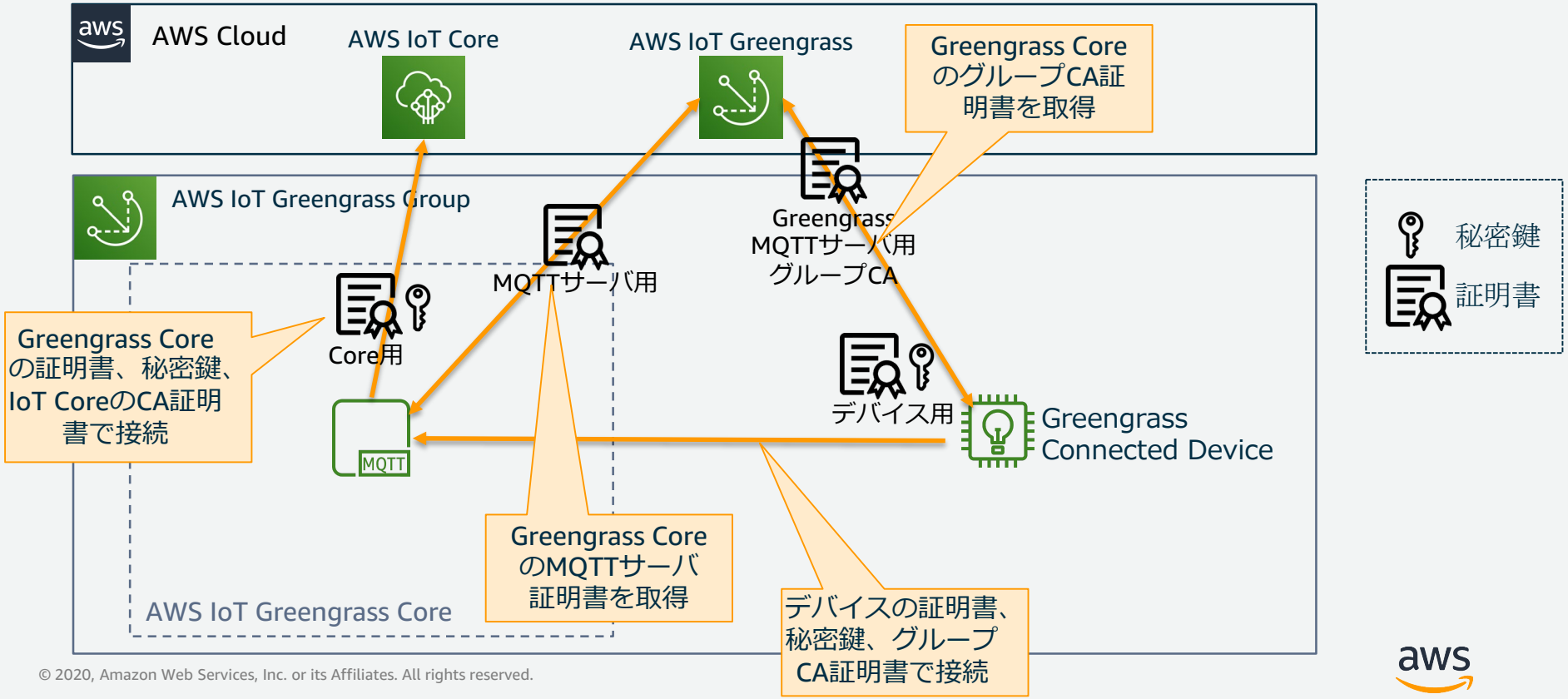


Greengrass Coreと同期するだけのLocal Device Shadowと、AWS IoT Coreと同期させるDevice Shadowの2つの使い方が可能

https://docs.aws.amazon.com/greengrass/latest/developerguide/config-dev-subs.html

セキュリティー

AWS IoT Core <-> Greengrass Core <-> Greengrass Connected Device

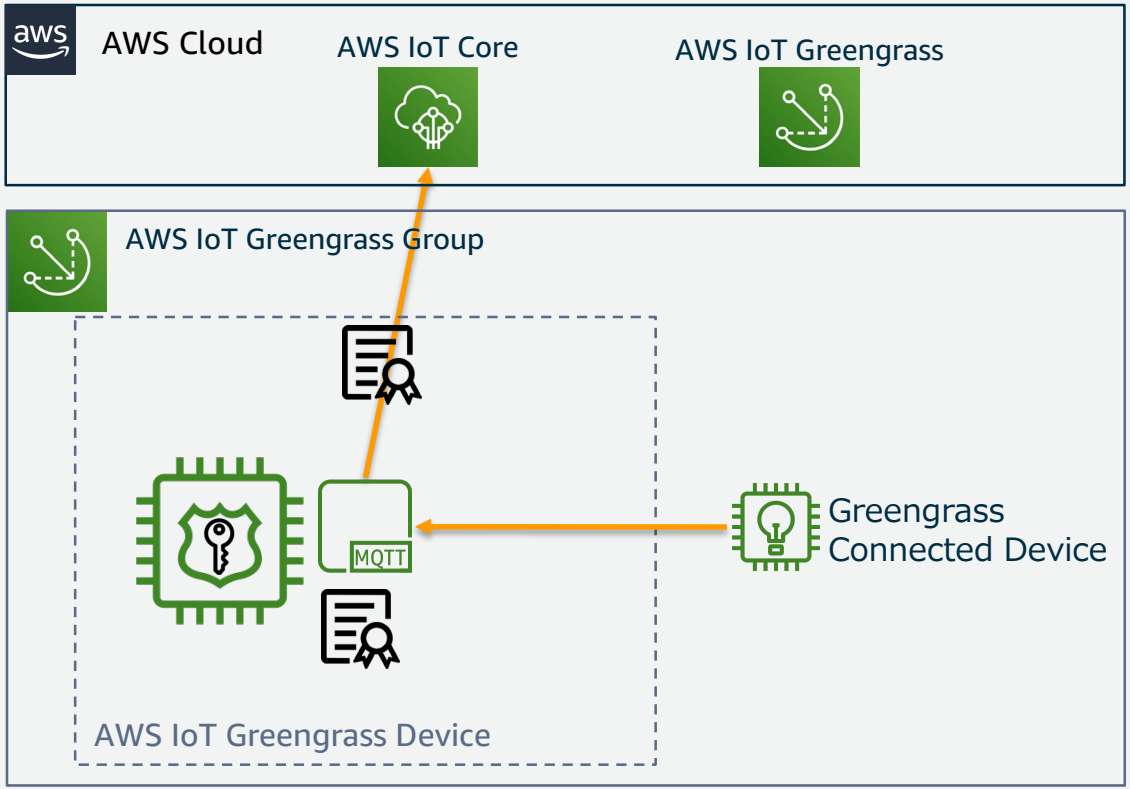


Greengrass上のMQTTブローカー用証明書の注意点

- サーバ証明書の有効期限はデフォルト7日で、最大30日です(さらに延長可能ですが、この場合はサポートにもお問い合わせください)
- 有効期限が切れる前にサーバ証明書が自動更新される
    - Greengrassがクラウドとの接続が切れた状態で、証明書の有効期限が切れると、認証ができないためデバイスからの新たな接続ができなくなります
- サーバ証明書の入れ替えが行われると、MQTTブローカーが再起動するため、接続しているデバイスは一度切断されます
    - デバイスのアプリケーションには必ず再接続の処理を実装しておくこと
- サーバのCA証明書を強制的に更新した場合は、Greengrass Coreに接続するデバイスのグループCA証明書の再取得が必須

aws

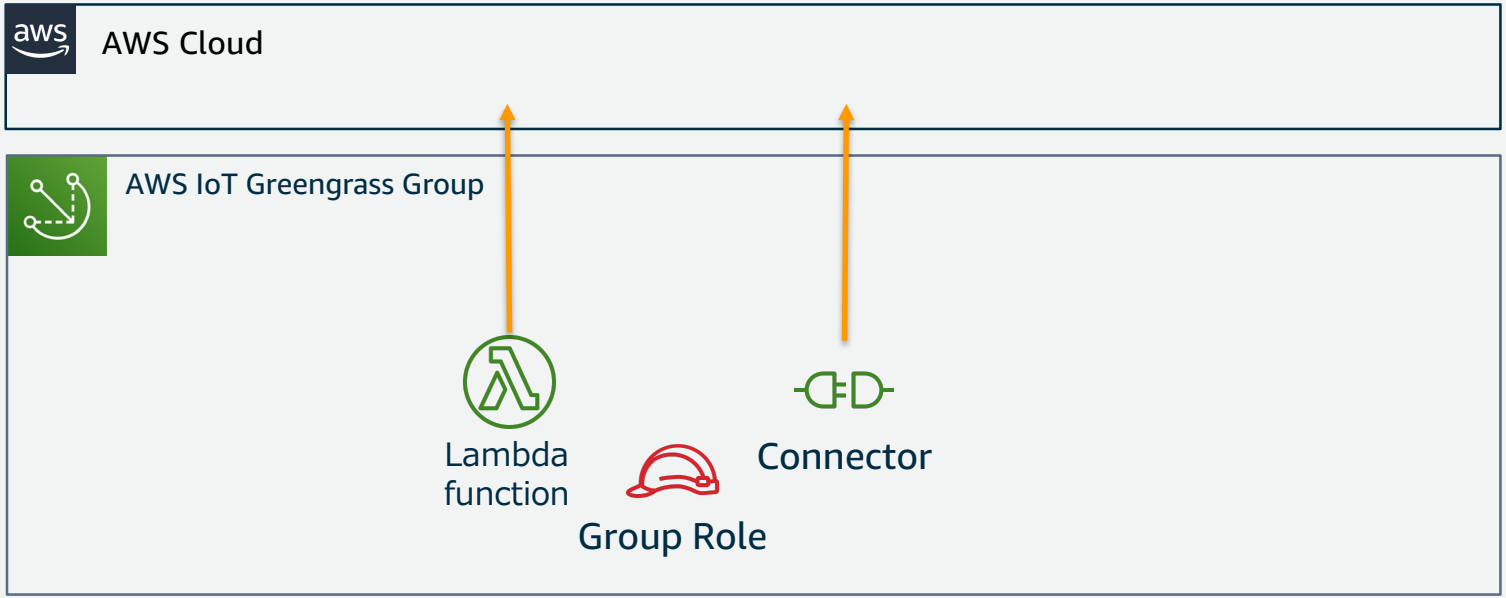
セキュリティー
ハードウエアセキュリティーモジュール(HSM)の利用



Greengrass Coreで利用する秘密鍵は、PKCS#11 インターフェイスに対応したセキュアモジュールに対応しています。
対応するデバイスは、AWS Partner Device Catalogで検索できます

セキュリティー

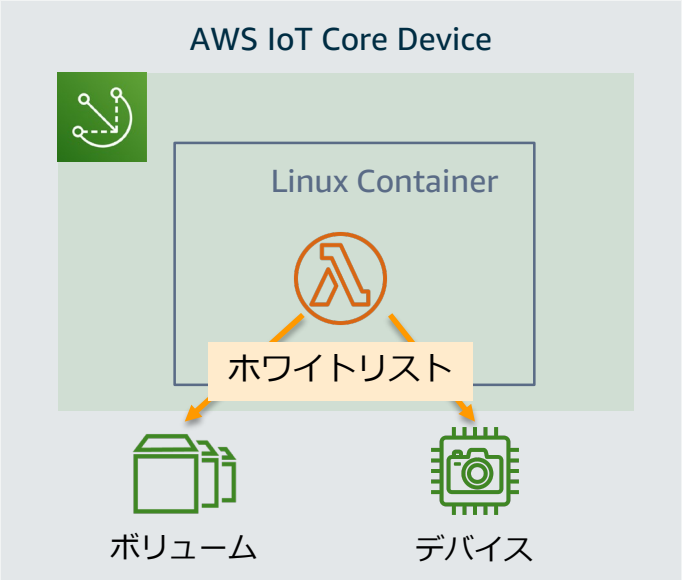
Greengrass Core <-> AWSサービス



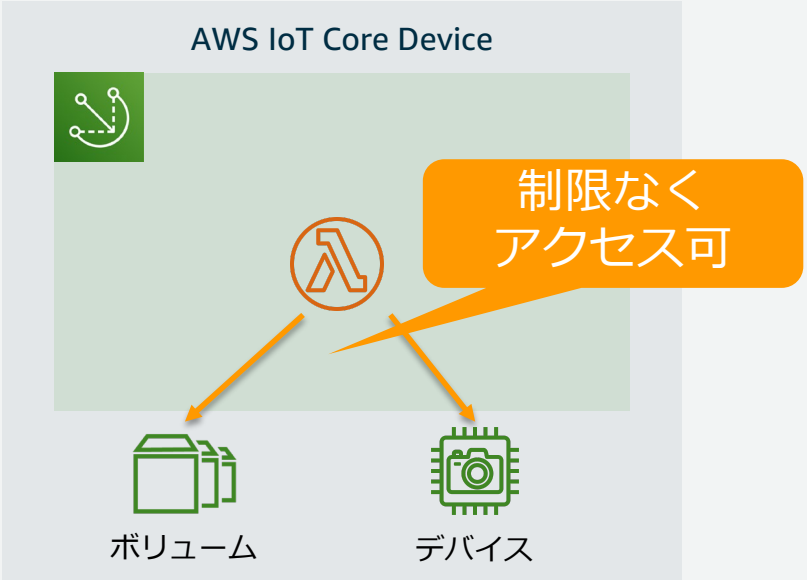
Greengrass GroupのRoleに必要な権限を持つPolicyを設定しておくことで、LambdaやConnectorから、直接AWSのサービスに対してアクセスすることが可能になります。

ローカルリソースアクセス

コンテナLambda(デフォルト)



ノンコンテナLambda

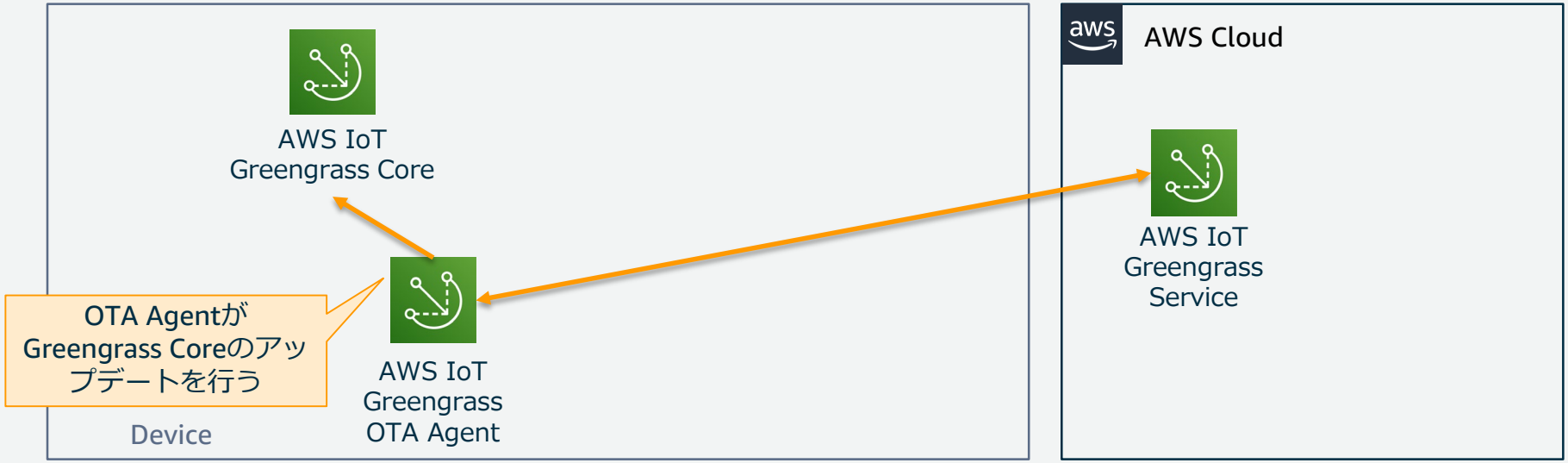


コンテナ上でLambdaを実行する場合は、ローカルリソースへのアクセスはホワイトリストで指定(*)するため、不必要にデバイスのリソースやメモリを使わず、セキュアに実行することが可能です。

*) 特定の場所のみ可能https://docs.aws.amazon.com/greengrass/latest/developerguide/access-local-resources.html

Over the Air (OTA)アップデート

Greengrass CoreとOTA Agentを安全にアップデート



Greengrass CoreやOTA Agentの新しいバージョンがでた場合は、OTA Agentを利用してアップデートを行えます。ただし、 aptなどパッケージ管理ツールでGreengrassをインストールした場合は、aptを利用して更新します。

AWS IoT Greengrassの拡張機能

機械学習の推論

機械学習の推論をデバイス上で行う

コネクター

コネクターを利用し、様々な機能やサービスと連携

シークレットマネージャー

秘密情報を安全にデバイスへ展開

コンテナのサポート

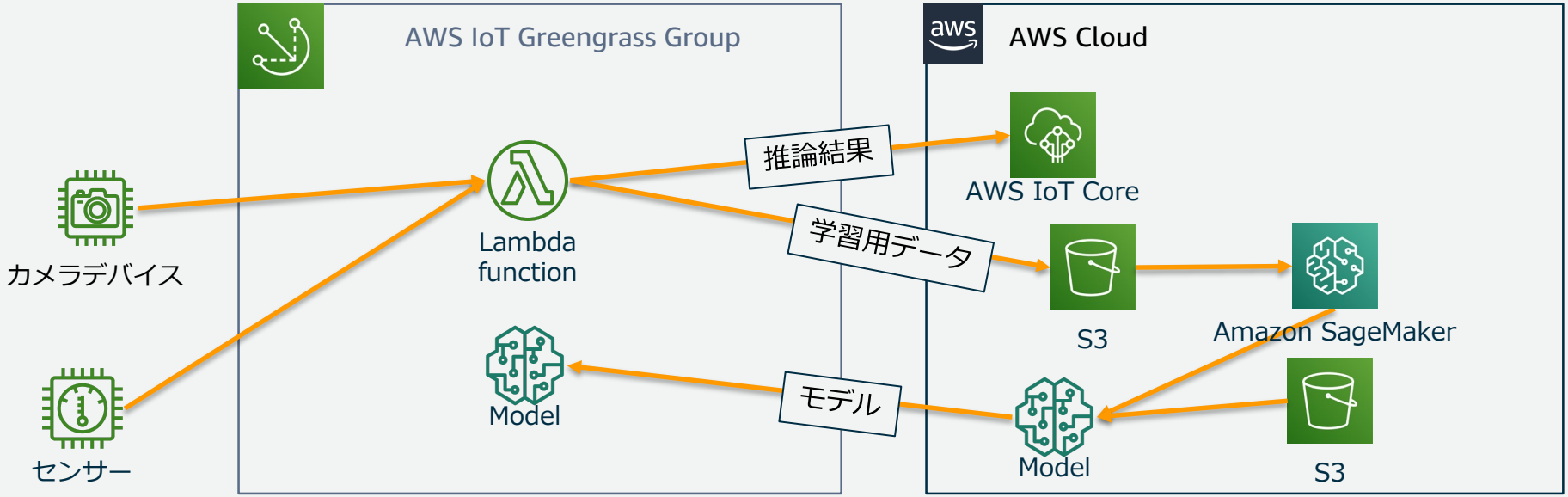
AWS LambdaやDockerのコンテナを利用可能

ストリームマネージャー

ストリームデータをエッジデバイスで収集、処理

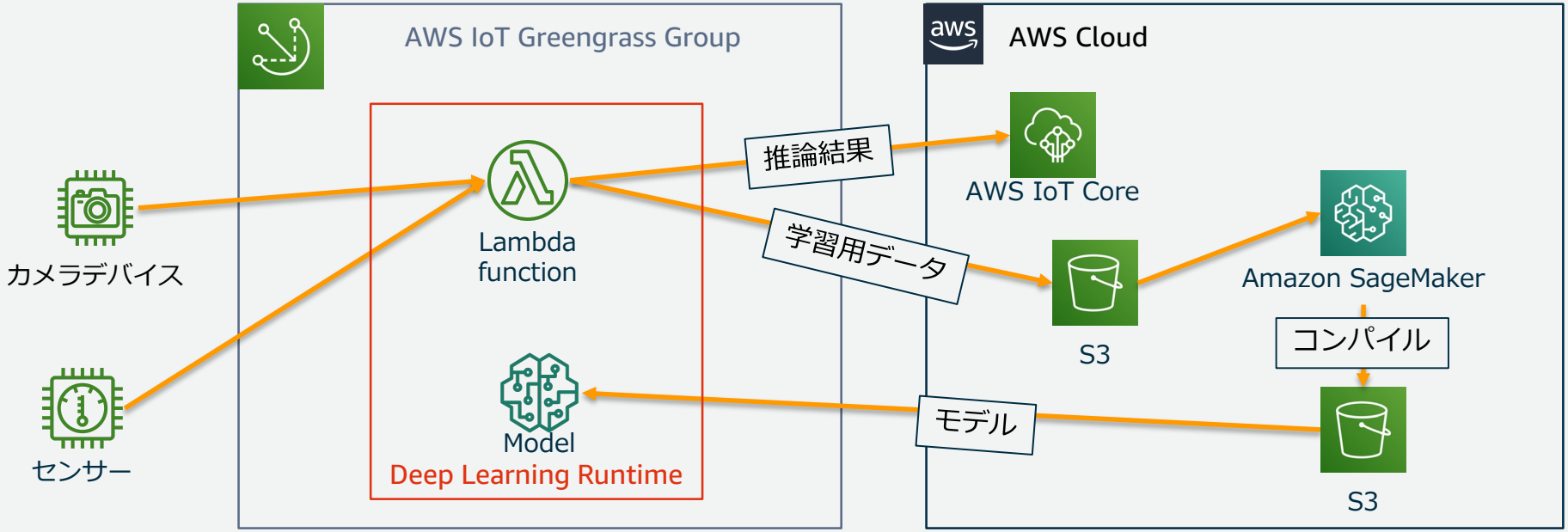
機械学習の推論

クラウドに集めたデータで学習したモデルを利用しエッジで推論



推論モデルの学習のような重い処理はクラウド上で行い、デバイス上で推論モデルを使って推論を行うことで、効率的に機械学習を利用したデバイスを作成可能

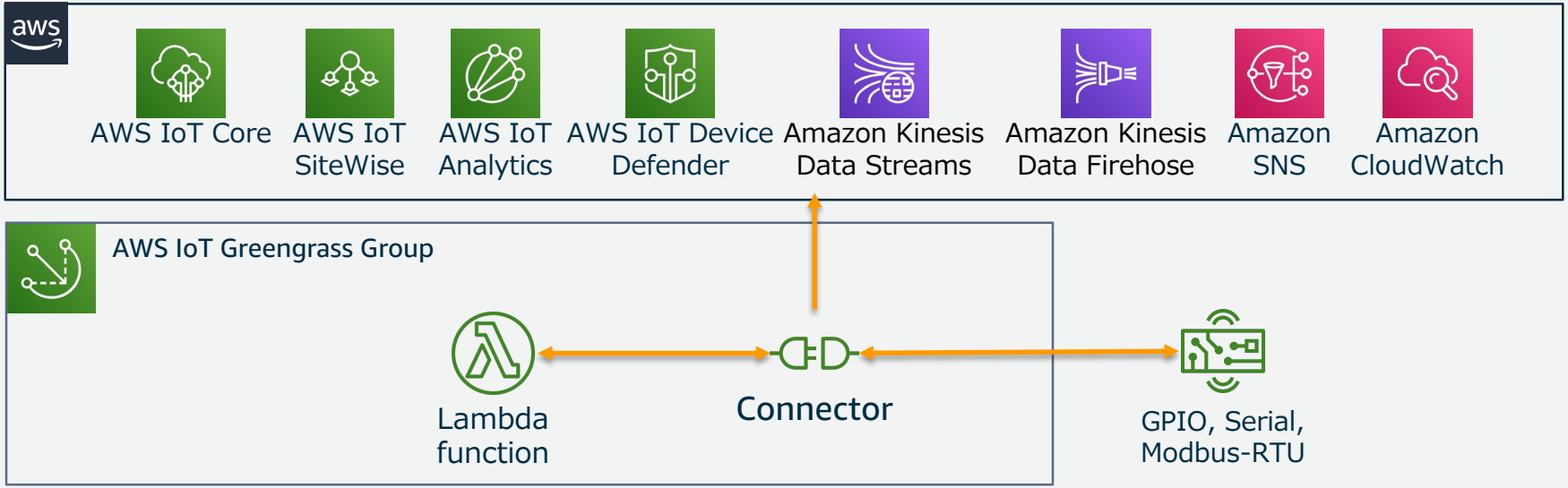
機械学習の推論
SageMaker Neoの利用



通常、利用するフレームワーク毎にデバイスへ必要なライブラリのインストールが必要だが、SageMaker Neoでコンパイルされたモデルは、各アーキテクチャ向けに用意されたDeep Learning Runtimeを使うことで、どのフレームワークで作ったモデルでも、低リソースで実行することが可能

コネクター

汎用的な処理はコネクターの利用で素早く開発

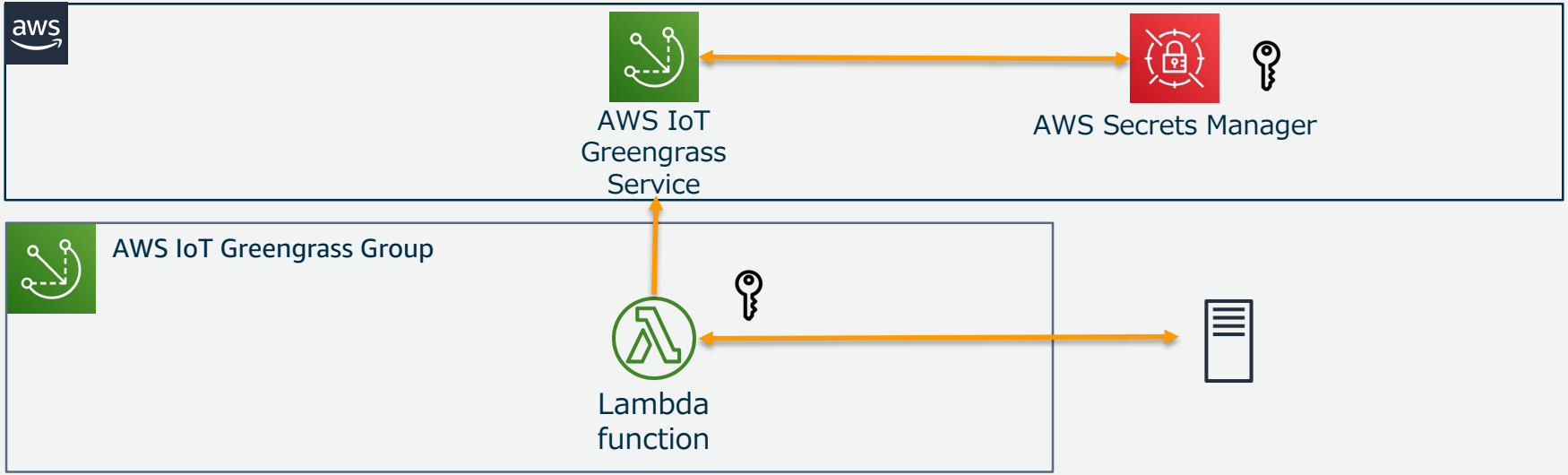


AWSのサービス連携や、デバイスへのアクセス、プロトコル変換向けにコネクターが用意されています。パラメータの設定や、MQTTトピックを介してデータを渡すだけで簡単に使えます。

https://docs.aws.amazon.com/greengrass/latest/developerguide/modbus-protocol-adapter-connector.html

シークレットマネージャー

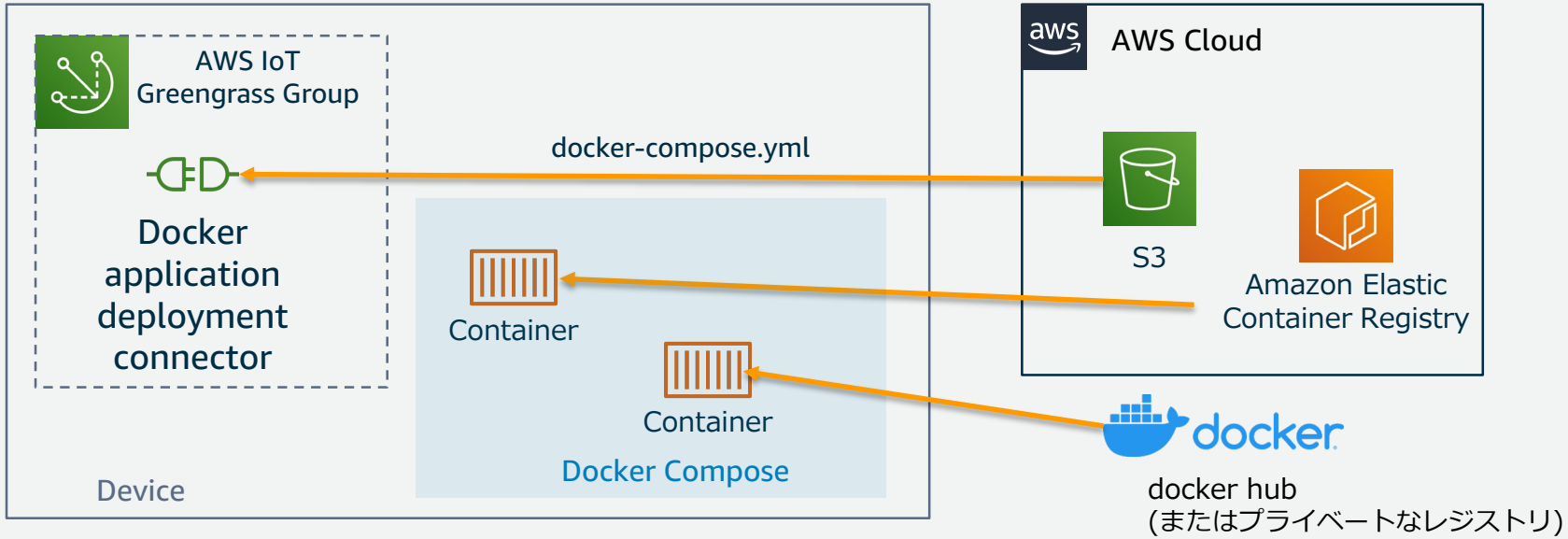
秘密情報をエッジデバイスに安全に展開



デバイス上に持たせたくないような、他のサービスへの認証情報やトークンなどを、AWS Secret Managerと連携させることで、必要な時に取得し、セキュアに他のデバイスやサービスにアクセスさせることが可能です。
取得した情報は、Greengrass Coreの秘密鍵を使って暗号化して保持

コンテナサポート

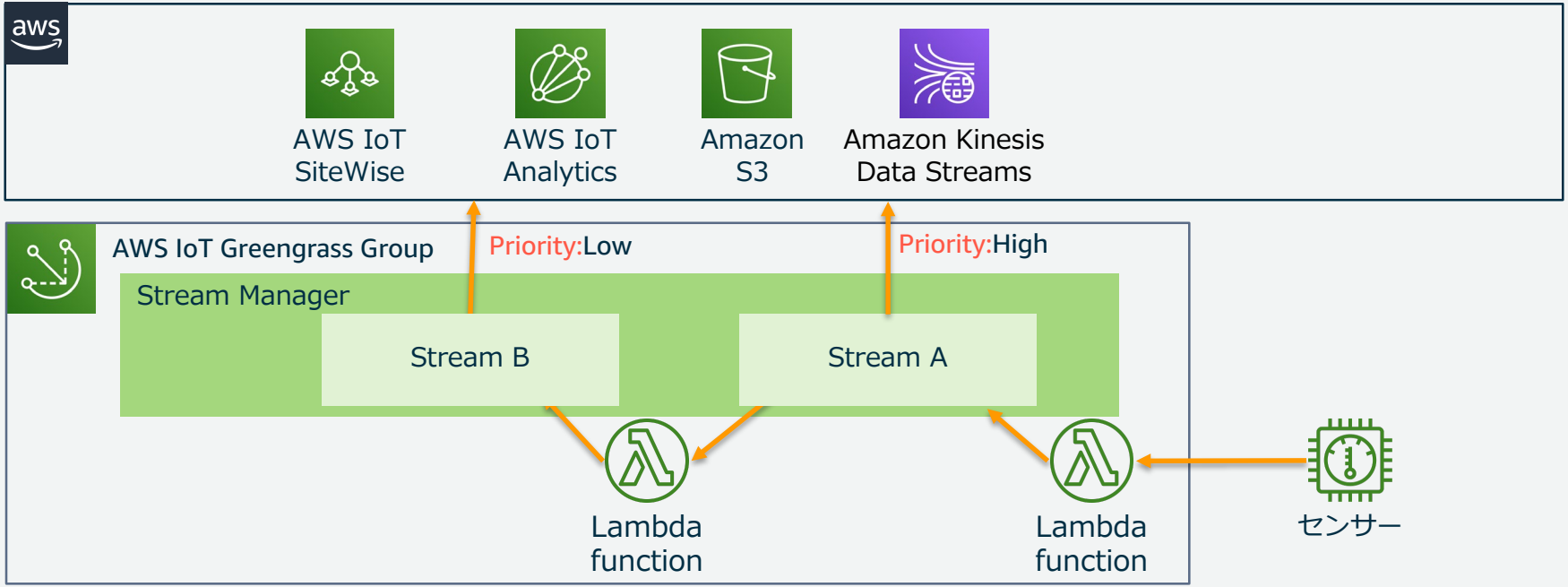
Dockerコンテナを利用して、エッジデバイスへアプリケーションの利用を拡張



既存のアプリケーションを含むDockerイメージを作成することで、Greengrassのデバイス上に簡単にデプロイし利用することが可能です。

ストリームマネージャー

大容量なストリームをエッジデバイスで収集、処理、エクスポート



ストリームのエクスポート先を指定することで、接続状態を気にせずに、アプリケーションからデータを書き込むことが可能。未接続の時はデータを溜め込み、接続されたら指定された優先度、帯域を元にデータをアップロードします。

StreamManagerの注意点

- StreamManagerを利用するにはJava8のインストールが必要になります
- S3 Exporterのデータはファイルで渡す必要があるため、ディスクI/Oによっては期待したスループットがでない場合があります
- TTLを指定しないとメッセージは指定されたストレージサイズを超えるまで、ストリームに残ります
- ストリームの保存先をメモリにした場合、Greengrassが再起動するとメッセージは失われます

https://docs.aws.amazon.com/greengrass/latest/developerguide/stream-manager.html

aws

本日のアジェンダ

- AWS IoT Greengrassとは
- AWS IoT Greengrassの始め方
- AWS IoT Greengrassのユースケース
- まとめ

システム要件(2020/12/15現在のv1.11向け)

- アーキテクチャ
  - Armv7l
  - Armv8(AArch64)
  - Armv6l
  - x86_64
- OS
  - Linux(kernel4.4以上、コンテナを使わない場合は、3.17以上)
    - Distribution: Raspberry Pi OS, OpenWrt, Amazon Linux, Ubuntu 18.04
  - Windows, Mac OS, Linux (with AWS IoT Greengrass in Docker)
- ハードウエア
  - Minimum 128MB Disk space(with OTA need 400MB Disk space)
  - Minimum 128 MB RAM(with Stream Manager minimum is 198MB)

注意) 上記はサポート対象の環境であり、要件が合えば動作させることも可能です
その他詳細な要件はこちら https://docs.aws.amazon.com/greengrass/latest/developerguide/what-is-gg.html

AWS IoT Greengrassの始め方

- Quick Startのスクリプトを利用
    - インストールとAWS IoT Greengrassの設定をデバイス上から一括で実行

- AWS IoTのマネージメントコンソールから設定
    - Greengrass Group、Coreの設定
    - 設定ファイルと証明書をダウンロード
    - Greengrass Coreソフトウエアのインストール
        - tarファイルを利用
        - aptを利用
        - Dockerイメージを利用

https://docs.aws.amazon.com/greengrass/latest/developerguide/gg-gs.html

Quick Start Scriptの利用

デバイス上でコマンドを実行してセットアップ。名前などはインタラクティブに指定可。

```
> export AWS_ACCESS_KEY_ID=YOUR-ACCESS-KEY
> export AWS_SECRET_ACCESS_KEY=YOUR-SECRET-KEY
> export AWS_SESSION_TOKEN=YOUR-SEEEION-TOKEN-IF-NEED

> wget -q -O ./gg-device-setup-latest.sh https://d1onfpft10uf5o.cloudfront.net/greengrass-device-setup/downloads/gg-device-setup-latest.sh && chmod +x ./gg-device-setup-latest.sh && sudo -E ./gg-device-setup-latest.sh bootstrap-greengrass-interactive

################ Greengrass Device Setup v1.0.3 ################
[GreengrassDeviceSetup] The Greengrass Device Setup bootstrap log is available at: /tmp/greengrass-device-setup-bootstrap-1606889614.log
[GreengrassDeviceSetup] Using package management tool: apt-get...
[GreengrassDeviceSetup] Using runtime: python3.7...
[GreengrassDeviceSetup] Installing a dedicated pip for Greengrass Device Setup...
[GreengrassDeviceSetup] Validating and installing required dependencies...
[GreengrassDeviceSetup] The Greengrass Device Setup configuration is complete. Starting the Greengrass environment setup...
[GreengrassDeviceSetup] Forwarding command-line parameters: bootstrap-greengrass-interactive
```

https://docs.aws.amazon.com/greengrass/latest/developerguide/quick-start.html

aws

AWS IoT Greengrassのデプロイについて

- マネージメントコンソールや、CLIから特定のGreengrass Groupに対してデプロイ

- バルクデプロイ
    - 特定のフォーマットに従って、デプロイ対象のGreengrass Group IDのJSONファイルをS3にアップロードし、CLIからデプロイを実行

コマンドの例

```
> aws greengrass start-bulk-deployment --cli-input-json "{
 "InputFileUri":"URI of file in S3 bucket",
 "ExecutionRoleArn":"ARN of execution role",
 "AmznClientToken":"your Amazon client token"
 }"
```

https://docs.aws.amazon.com/greengrass/latest/developerguide/bulk-deploy-cli.html

AWS IoT Greengrassのモニタリングについて
ログ

- 設定を変えてデプロイするだけで、Greengrass CoreとLambdaのログをCloudWatch Logsにアップロードできる
    - デバイスが遠隔地にあっても、簡単にログが確認できます

- ログレベルも同様にクラウド側で設定を変更してデプロイすることで変更が可能

AWS IoT Greengrassのモニタリングについて

Greengrass Core自体、Lambdaのテレメトリを取得

- Greengrass Coreの設定を変更することで、AWS IoT Greengrass CoreのCPU使用量、メモリ使用量、LambdaやStream Managerの情報を取得可能

- この設定を有効にすると、毎時に取得したデータを、24時間毎に受け取ることができます

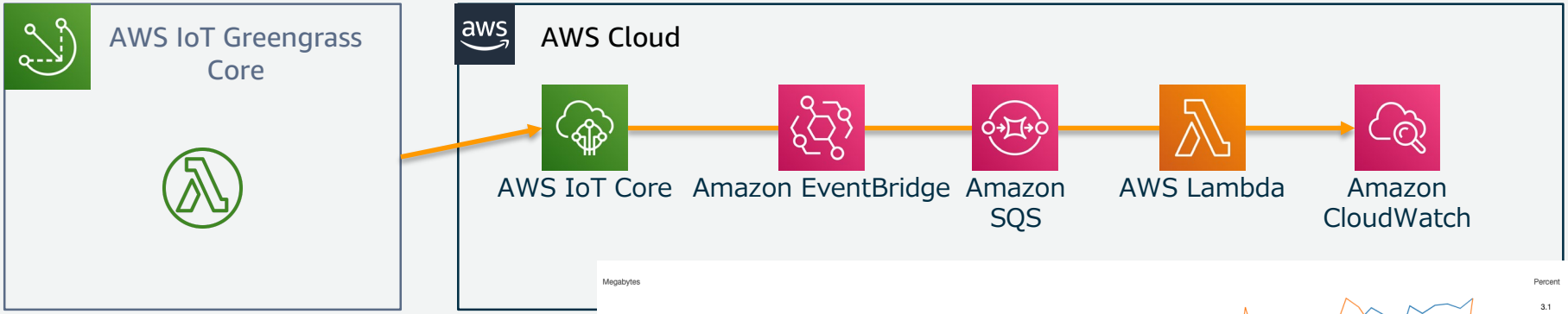
- 取得されたテレメトリデータは、MQTTでAWS IoT Core経由でAmazon EventBridgeに連携されます(このデータ送信分の費用は加算されません)

- Amazon EventBridgeのルールを設定することで、他のサービスに連携して監視する仕組みを作ることができます

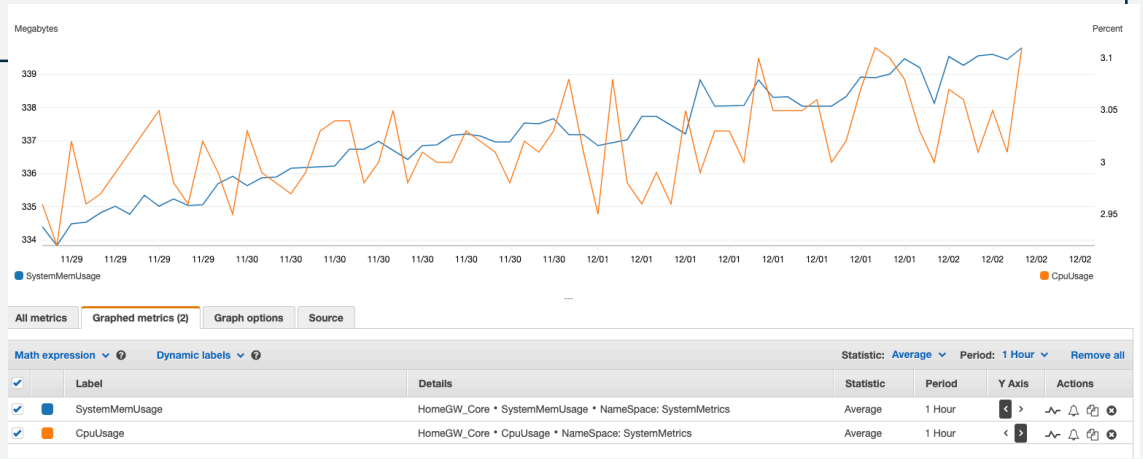
https://docs.aws.amazon.com/greengrass/latest/developerguide/telemetry.html

AWS IoT Greengrassのモニタリングについて

テレメトリの監視例



CloudWatchのメトリックスとして登録すれば、Alarmの設定等で監視することも可能



AWS IoT Greengrassのモニタリングについて

Local Health APIの利用

- 以下のURLにGETでデバイス上でアクセスすることで、Lambdaやシステムコンポーネントの状態が確認可能
    - http://localhost:8000/2016-11-01/health/workers
    - 特定のコンポーネントを指定する場合は、POSTで条件を指定

例

```
[
 {
  "FuncArn": "arn:aws:lambda:::function:GGShadowService:1",
  "WorkerId" : "65515053-2f70-43dc-7cc0-1712bEXAMPLE",
  "ProcessId": "1234",
  "WorkerState": "Waiting"
 },
 {
  . . .
]
```

https://docs.aws.amazon.com/greengrass/latest/developerguide/health-check.html

AWS IoT Greengrassの料金体系(2020/12/15現在)

- アクティブなAWS IoT Greengrassデバイス1台につき0.18ドル/月(東京リージョン)
    - 1ヶ月の間、1度でもAWSで認証されるとアクティブとカウントされます
    - 台数によって割引があります

- AWSの他のサービスとの連携、データの送受信の費用は、各サービスの料金がかかります。料金については利用する他のサービスの情報を参照してください

https://aws.amazon.com/greengrass/pricing/

独自のデバイスでAWS IoT Greengrassを動かしたい場合

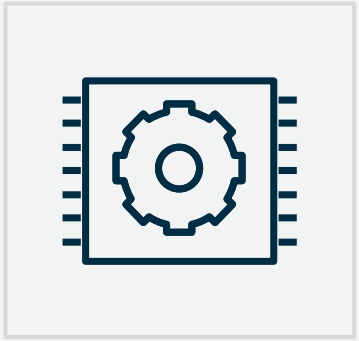
- GitHubに動作要件を満たすかをチェックするためのスクリプトがあります
  - https://github.com/aws-samples/aws-greengrass-samples
- このスクリプトを実行することで、足りている要件、不足している要件を確認できます。

実行例

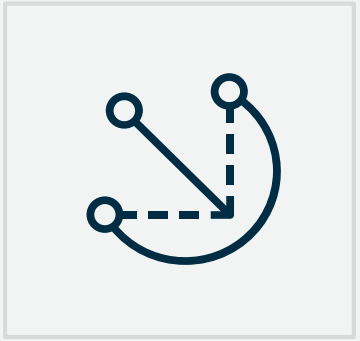
```
$ wget https://github.com/aws-samples/aws-greengrass-samples/raw/master/greengrass-dependency-checker-GGCv1.11.x.zip
$ unzip greengrass-dependency-checker-GGCv1.11.x.zip
$ cd greengrass-dependency-checker-GGCv1.11.x
$ sudo ./check_ggc_dependencies
```

AWS IoT Device Tester

AWS IoT Device Testerは利用したいデバイスがFreeRTOSやAWS IoT Greengrassが利用可能化を自動でテストするツールです



AWS IoT Device Tester for FreeRTOS
FreeRTOS が利用かどうか、AWS IoTのサービスと連携できるかをテストします



AWS IoT Device Tester for AWS IoT Greengrass
デバイスのCPUアーキテクチャ、カーネル設定、ドライバがAWS IoT Greengrassで利用可能化をテストします

AWS IoT Device Testerのダウンロードは
FreeRTOS と AWS IoT Greengrass 製品ページから

AWS Device Qualification Program

AWS Device Qualification Programはデバイスハードウェアが AWS IoT Greengrass および FreeRTOS が
動作することを検証することによって認定が受けられる、APNパートナーに公開されている
ハードウェア認定およびインセンティブプログラムです。認定されたデバイスは、
AWSで動作するIoTデバイスをお客様が発見できるようにするためにAWS Device Catalogへ掲載できます

*1 FreeRTOS および AWS IoT Greengrass のサポート
*2 APNはパートナーへハードウェアを要求する場合があります

https://devices.amazonaws.com/

本日のアジェンダ

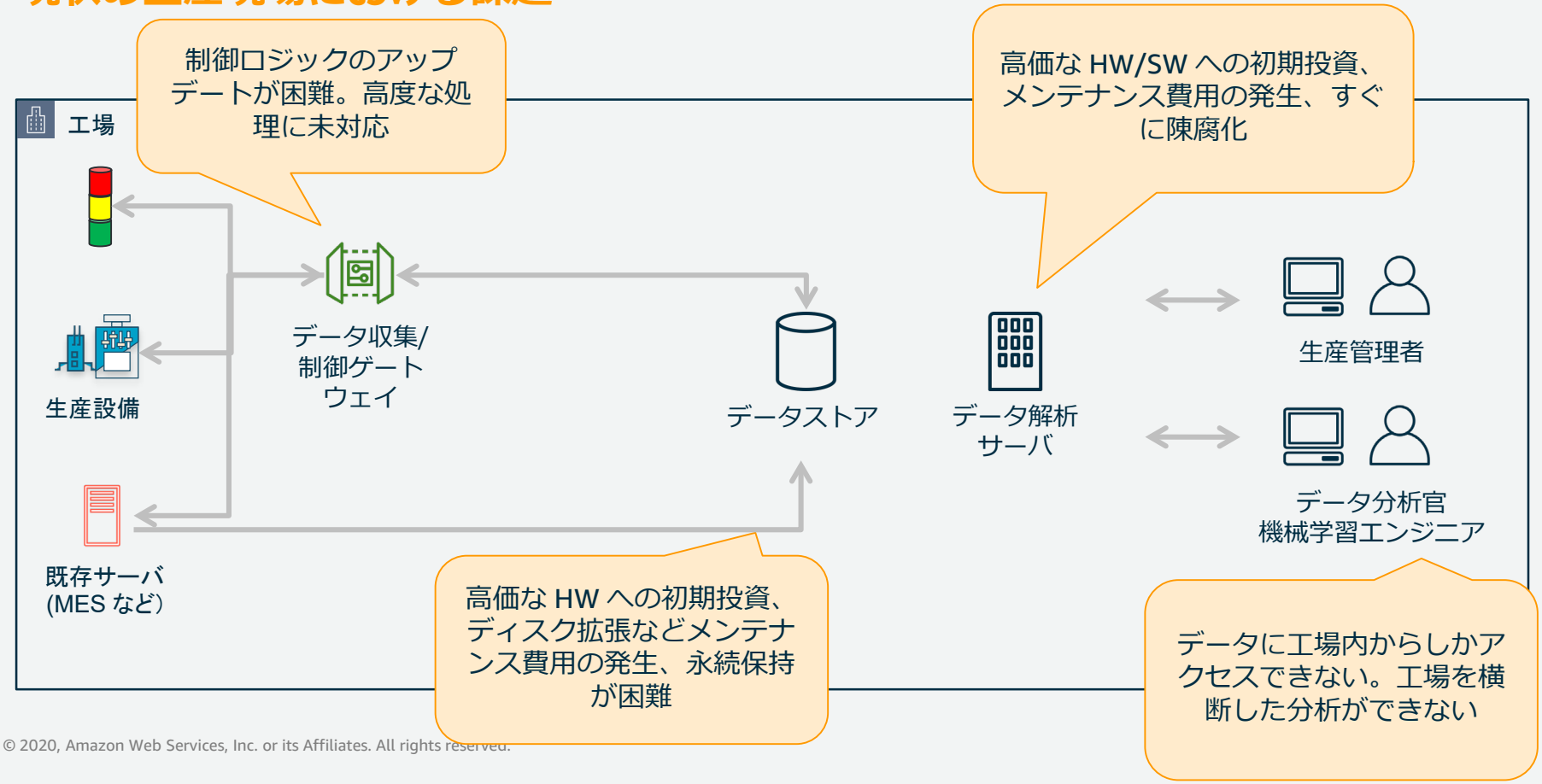
- AWS IoT Greengrassとは
- AWS IoT Greengrassの始め方
- AWS IoT Greengrassのユースケース
- まとめ

AWS IoT Greengrassのユースケース

- スマートファクトリーでのユースケース
- コネクテッドカーでのユースケース

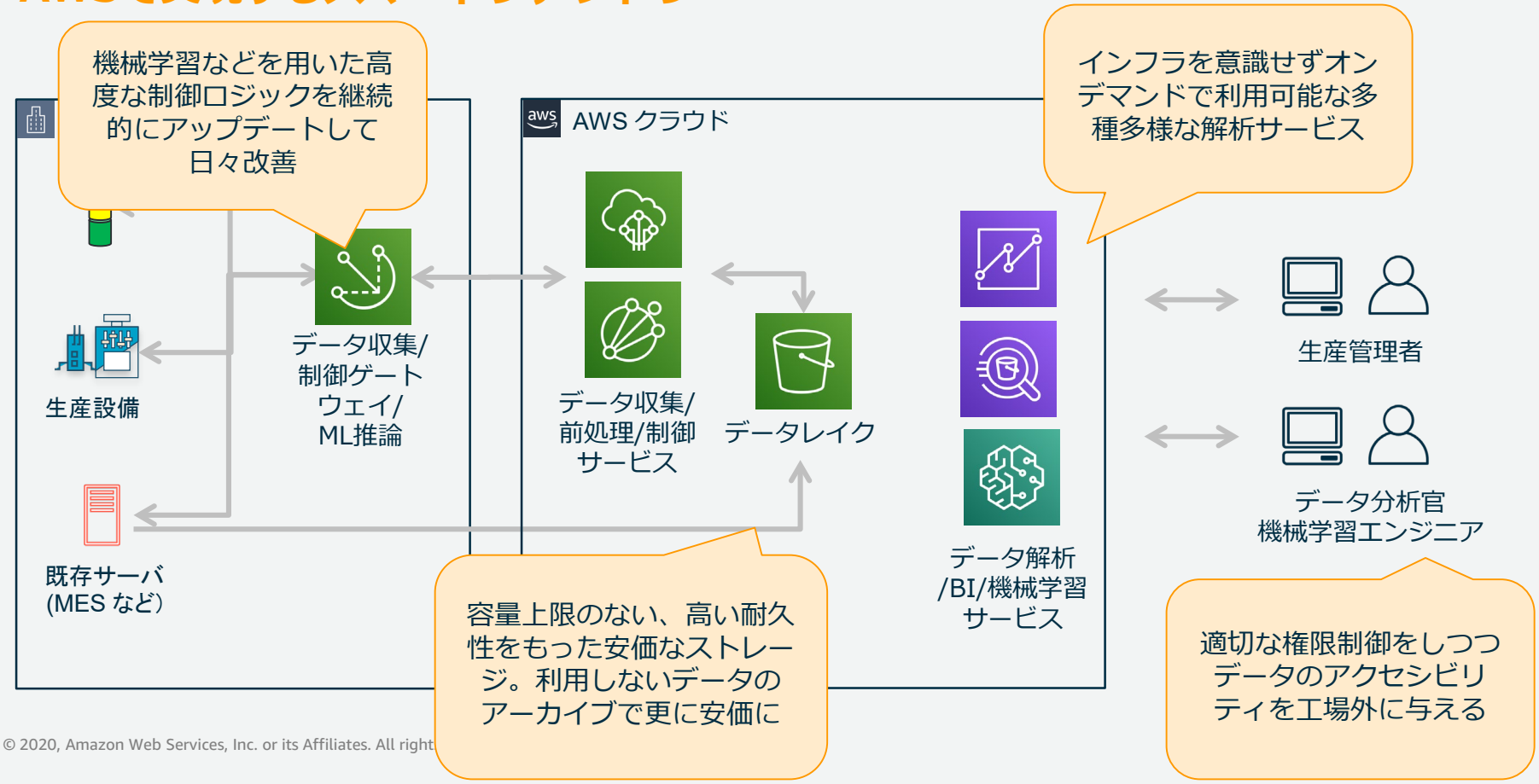
スマートファクトリーでのユースケース

現状の生産現場における課題



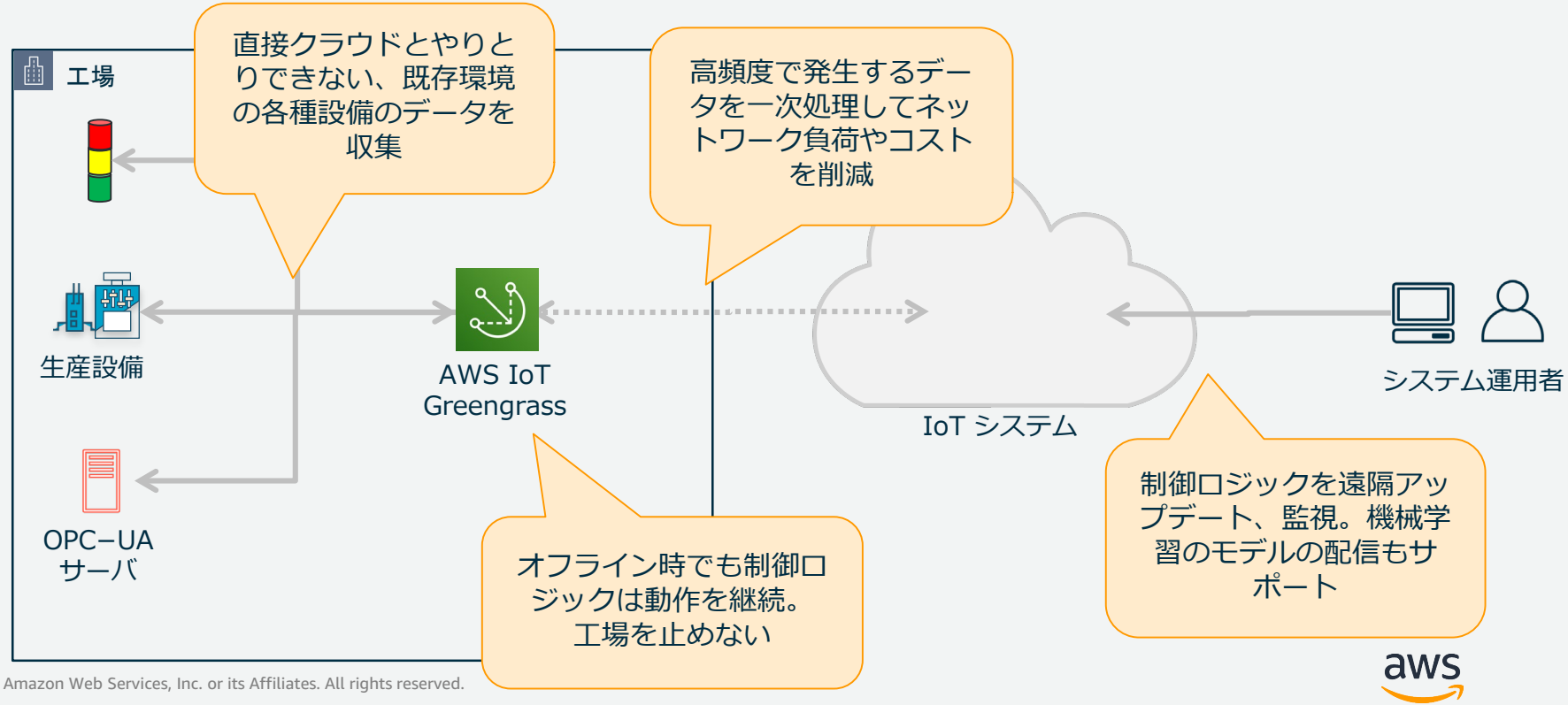
スマートファクトリーでのユースケース

AWSで実現するスマートファクトリー



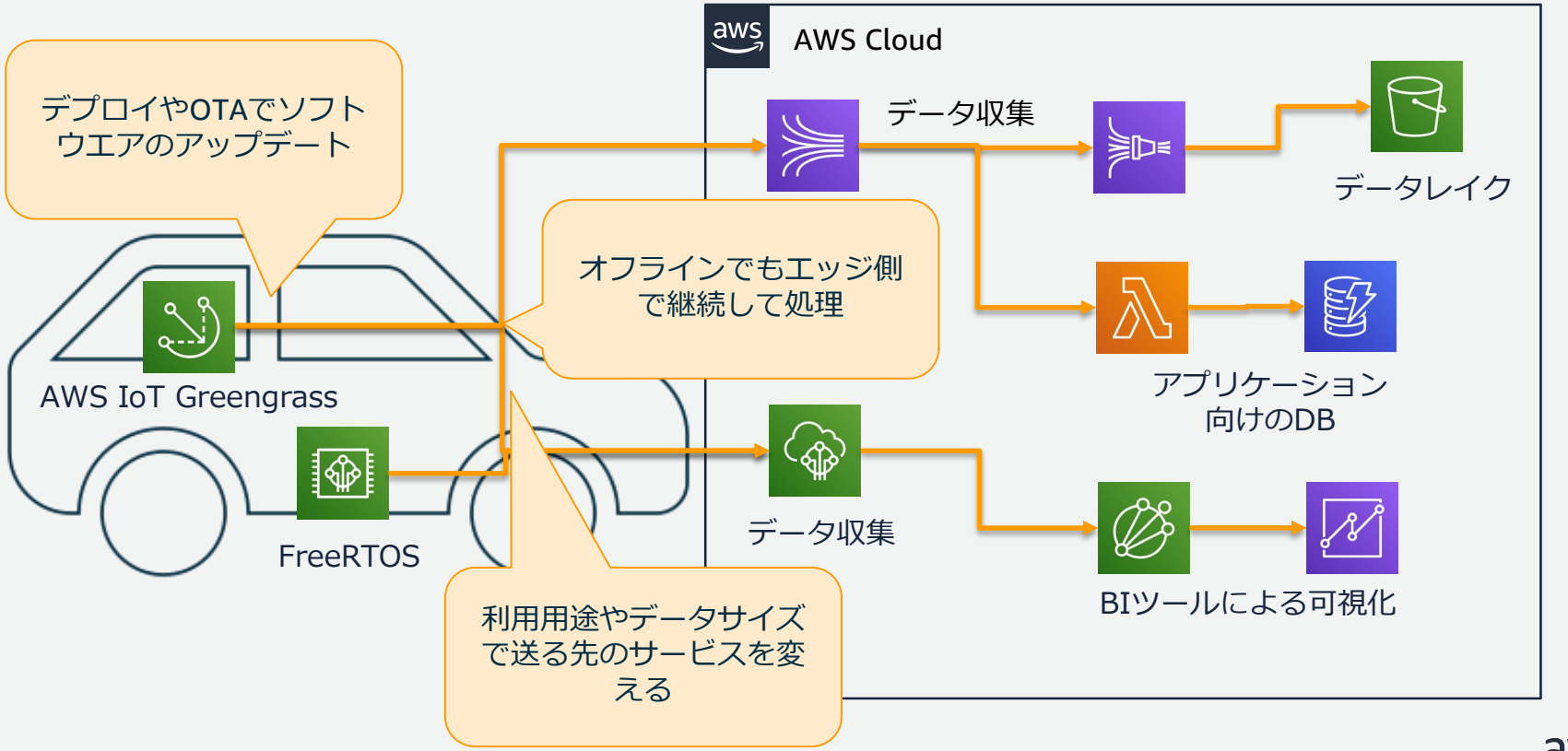
スマートファクトリーでのユースケース

AWS IoT Greengrassの役割



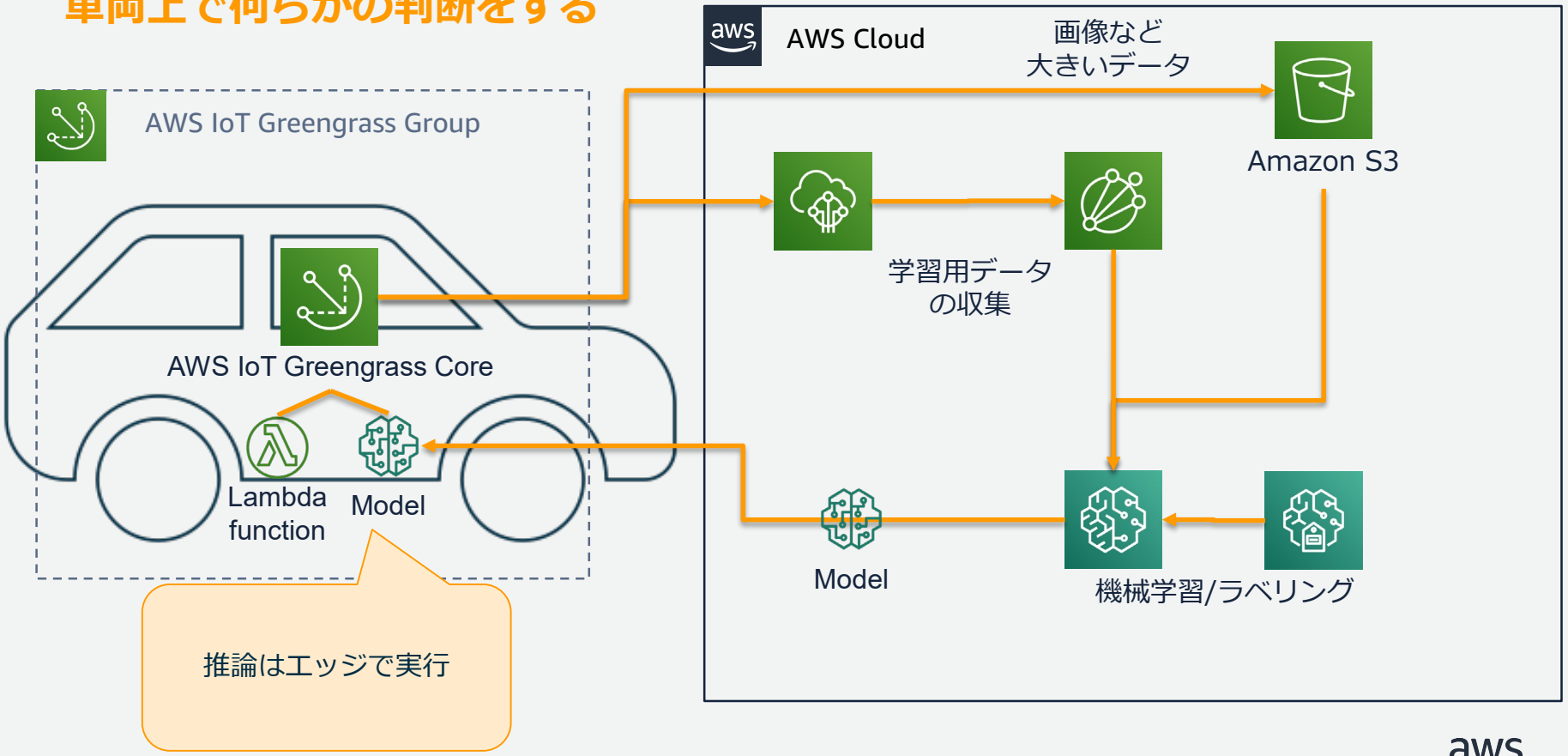
コネクテッドカーでのユースケース

車両のデータをクラウドに上げる



コネクテッドカーでのユースケース

車両上で何らかの判断をする



まとめ

まとめ

- データ量、ネットワーク、プライバシーの様な課題を解決する手段としてエッジに処理を持っていく必要がある場合、AWS IoT Greengrassを利用すると解決できます

- エッジ側にインテリジェンスな処理(機械学習や、ビジネスロジックなど)を持たせたい場合、AWS IoT Greengrassを利用することで、継続してアップデートする仕組みを簡単に実現することができます

- まずは触ってみようと思った方は、公開されているハンズオンを試してみてください

  https://aws-iot-greengrass-for-beginners.workshop.aws/

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/

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