

# SRPC-MBL for AWS IoT

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



## Table of contents

初期設定.....	3
使用方法.....	26
工場出荷時.....	30
フォーマット.....	30
変更履歴.....	31

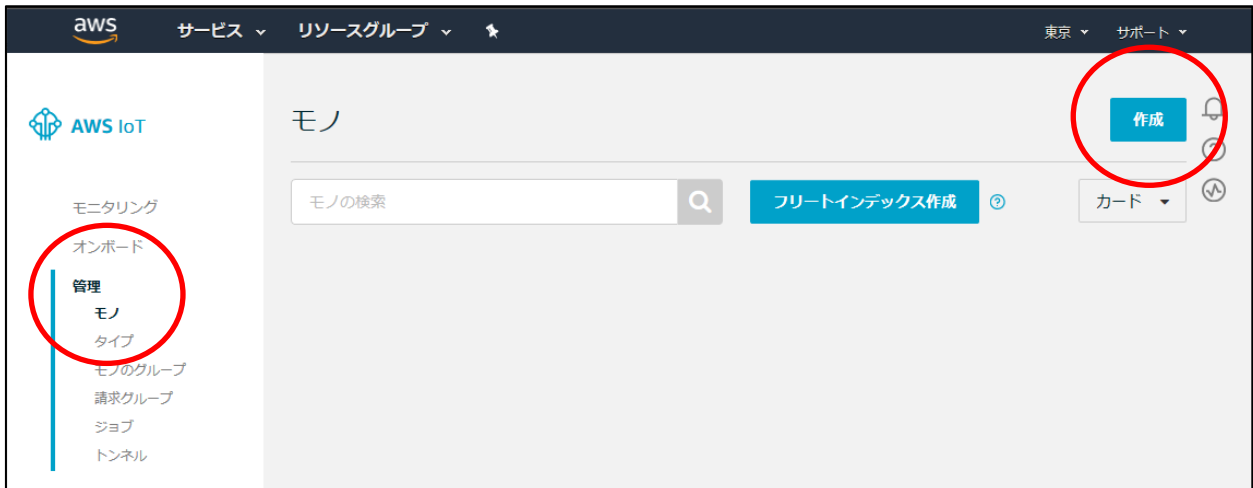
製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

## 初期設定

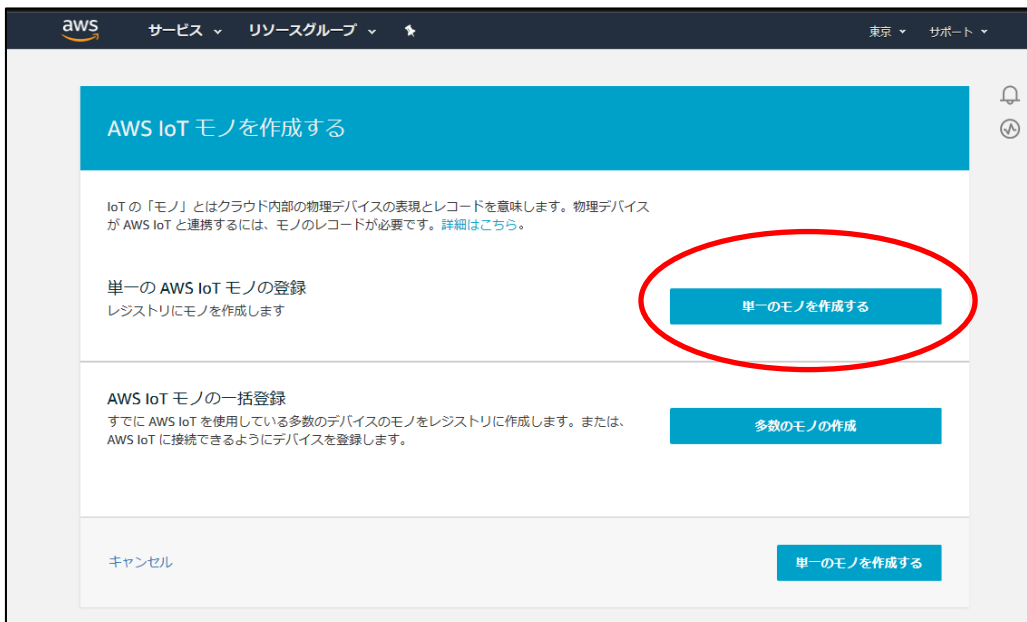
### 手順1

AWS Console (<https://aws.amazon.com/jp/console/>) を開いて、ログインしてください。この後の手順で AWS IoT に”モノ”を登録するので、ログインユーザーにはその操作が可能な権限（ポリシー）が与えられている必要があります。

“IoT Core”サービスを開いてください。見つからない場合には、サービスの検索をしてください。



“管理”→”モノ”を選択し、右上の“作成”ボタンを押してください。



“単一のモノを作成する”ボタンを押してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

モノの作成  
Thing Registry にデバイスを追加  
ステップ 1/3

このステップは、デバイスの Thing Registry と Thing Shadow にエントリーを作成します。

名前  
MONO\_TEST X

“名前”を入力して、一番下にある”次へ”ボタンを押してください。

モノの作成  
モノに証明書を追加  
ステップ 2/3

証明書は、AWS IoT へのデバイスの接続を認証するために使用されます。

1-Click 証明書作成 (推奨)  
AWS IoT の認証局を使用して証明書、パブリックキー、プライベートキーを作成します。  
証明書の作成

CSR による作成  
所有しているプライベートキーに基づいて固有の証明書署名リクエスト (CSR) をアップロードします。  
CSR による作成

お持ちの証明書を使用する  
CA 証明書を登録し、1 つ以上のデバイスに独自の証明書を使用します。  
開始方法

証明書をスキップしてモノを作成  
デバイスを AWS IoT に接続できるようにする前に、後で証明書をモノに追加する必要があります。  
証明書なしでモノを作成

“証明書なしでモノを作成” ボタンを押して、“モノ” の登録を完了します。

捕捉)

“証明書の作成”ボタンを押すと、AWS がデバイス用（SRPC-MBL 用）の証明書を作成してくれるのですが、SRPC-MBL はその証明書のシリアル番号の桁数をサポートできないため使用できません。自前で証明書を作成する必要があります。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

## 手順2

AWS のルート証明書 (Amazon Root CA1) をダウンロードします。SRPC-MBL から AWS に送信するデータを暗号化するために使用されます。

レポジトリ (<https://www.amazontrust.com/repository/>) を開いて下さい。

### Root CAs

Distinguished Name	SHA-256 Hash of Subject Public Key Information	Self-Signed Certificate	Test URLs
CN=Amazon Root CA 1,O=Amazon,C=US	fbe3018031f9586bcbf41727e417b7d1c45c2f47f93be372a17b96b50757d5a2	DER PEM	Valid Revoked Expired
CN=Amazon Root CA 2,O=Amazon,C=US	7f4296fc5b6a4e3b35d3c369623e364ab1af381d8fa7121533c9d6c633ea2461	DER PEM	Valid Revoked Expired
CN=Amazon Root CA 3,O=Amazon,C=US	36abc32656acfc645c61b71613c4bf21c787f5cabbee48348d58597803d7abc9	DER PEM	Valid Revoked Expired

Amazon Root CA 1 の PEM を右クリックして、ファイルに保存してください。  
 ファイル名は、“AmazonRootCA1.pem” とします。

## 手順3

AWS CLI をインストールします。

インストール方法は、[https://docs.aws.amazon.com/ja\\_jp/cli/latest/userguide/install-windows.html](https://docs.aws.amazon.com/ja_jp/cli/latest/userguide/install-windows.html) を参照してください。

インストールが完了後、Windows の「スタートメニュー」を開いて、“cmd”を検索して実行してください。コマンドプロンプトが起動されます。

コマンドプロンプト上で、aws --version と入力し、バージョン番号が表示されればインストールは成功しています。(ハイホンは2つ)

```

c:\> コマンドプロンプト

C:\Users\%> aws --version
aws-cli/1.16.192 Python/3.6.0 Windows/10 botocore/1.12.182

C:\Users\%>
    
```

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

#### 手順4

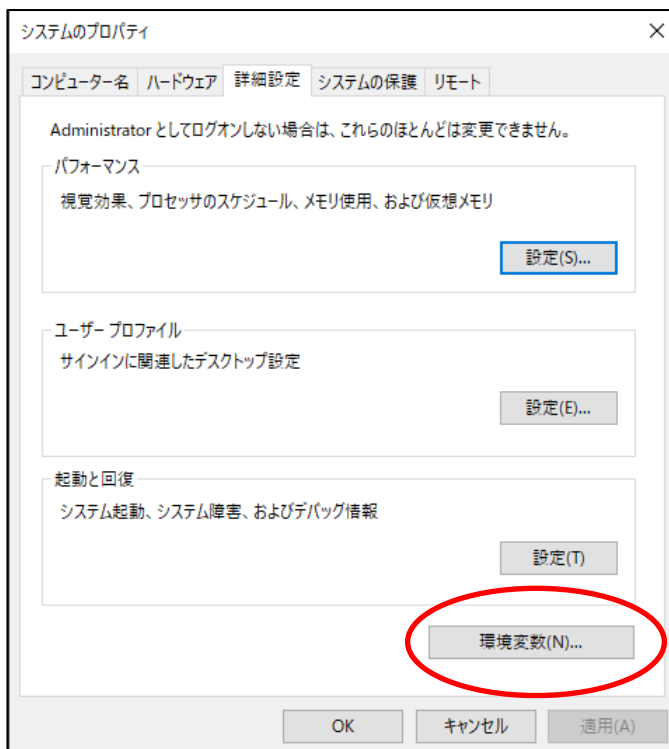
OpenSSL をインストールします。

ダウンロード画面 (<https://slproweb.com/products/Win32OpenSSL.html>) を開いてください。

File	Type	Description
Win64 OpenSSL v1.1.1d Light EXE   MSI (experimental)	3MB Installer	Installs the most commonly used essentials of Win64 OpenSSL v1.1.1d (Recommended for users by the creators of <a href="#">OpenSSL</a> ). Only installs on 64-bit versions of Windows. Note that this is a default build of OpenSSL and is subject to local and state laws. More information can be found in the legal agreement of the installation.
Win64 OpenSSL v1.1.1d EXE   MSI (experimental)	43MB Installer	Installs Win64 OpenSSL v1.1.1d (Recommended for software developers by the creators of <a href="#">OpenSSL</a> ). Only installs on 64-bit versions of Windows. Note that this is a default build of OpenSSL and is subject to local and state laws. More information can be found in the legal agreement of the installation.
Win32 OpenSSL v1.1.1d Light EXE   MSI (experimental)	3MB Installer	Installs the most commonly used essentials of Win32 OpenSSL v1.1.1d (Only install this if you need 32-bit OpenSSL for Windows. Note that this is a default build of OpenSSL and is subject to local and state laws. More information can be found in the legal agreement of the installation).
Win32 OpenSSL v1.1.1d EXE   MSI (experimental)	30MB Installer	Installs Win32 OpenSSL v1.1.1d (Only install this if you need 32-bit OpenSSL for Windows. Note that this is a default build of OpenSSL and is subject to local and state laws. More information can be found in the legal agreement of the installation).
Win64 OpenSSL v1.1.0L Light EXE   MSI (experimental)	3MB Installer	Installs the most commonly used essentials of Win64 OpenSSL v1.1.0L (Recommended for users by the creators of <a href="#">OpenSSL</a> ). Only installs on 64-bit versions of Windows. Note that this is a default build of OpenSSL and is subject to local and state laws. More information can be found in the legal agreement of the installation.
Win64 OpenSSL v1.1.0L EXE   MSI (experimental)	37MB Installer	Installs Win64 OpenSSL v1.1.0L (Recommended for software developers by the creators of <a href="#">OpenSSL</a> ). Only installs on 64-bit versions of Windows. Note that this is a default build of OpenSSL and is subject to local and state laws. More information can be found in the legal agreement of the installation.

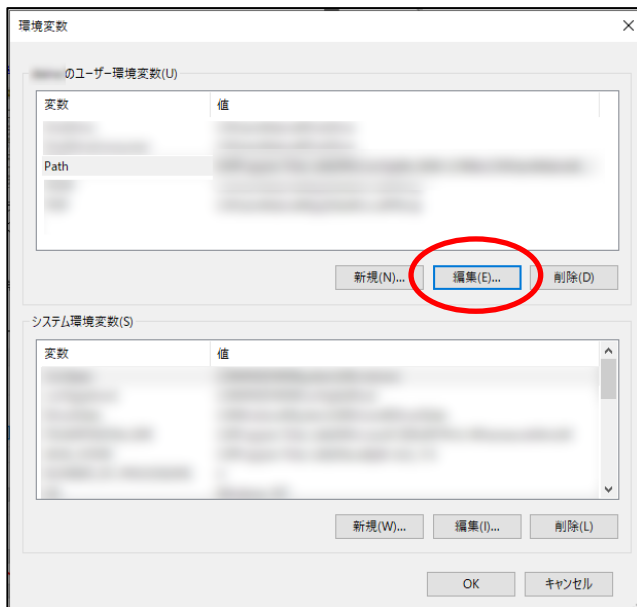
Win64 OpenSSL の“MSI”を選択し、インストーラーをダウンロードしてください。ダウンロード後、インストーラーを起動して、インストールを行ってください。この際、インストール先を覚えておいてください、後で使用します。

Windows の「スタートメニュー」を開いて、「環境変数」と検索し、実行してください。システムのプロパティ画面が表示されます。

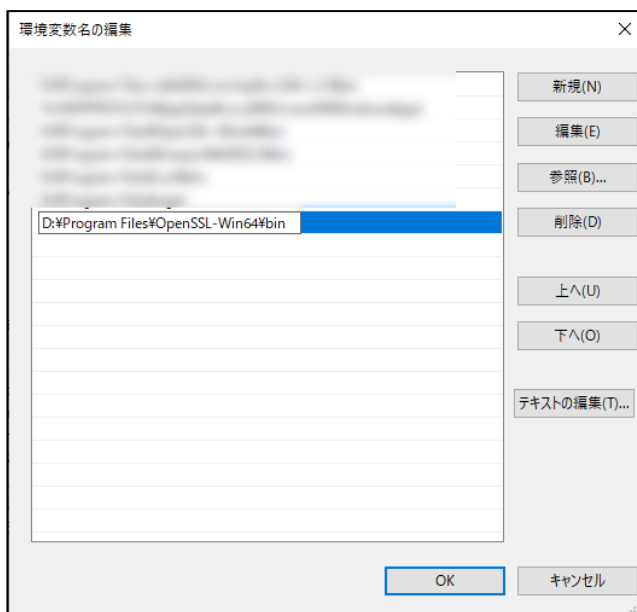


“環境変数” ボタンを押してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



“Path”を選択し、“編集” ボタンを押して下さい。



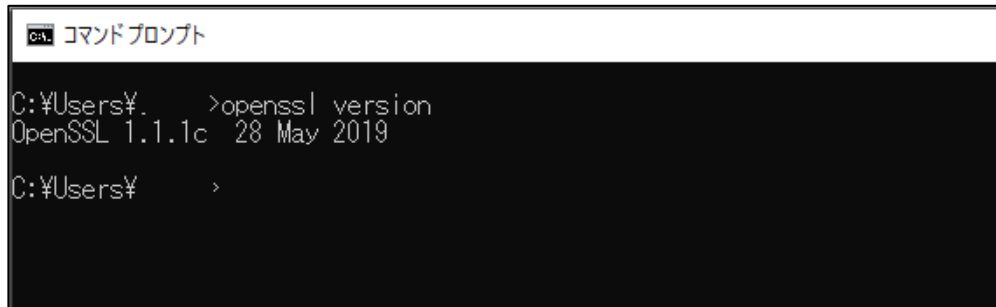
先ほどのインストール先を追加して“OK” ボタンを押してください。最後の“bin”まで指定してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



環境変数の設定が完了したら、Windows の「スタートメニュー」を開いて、「cmd」を検索して実行してください。コマンドプロンプトが起動されます。

コマンドプロンプト上で、openssl version と入力し、バージョン番号が表示されれば正しく設定されています。



```
cmd コマンドプロンプト
C:\Users¥. >openssl version
OpenSSL 1.1.1c 28 May 2019
C:\Users¥ >
```

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

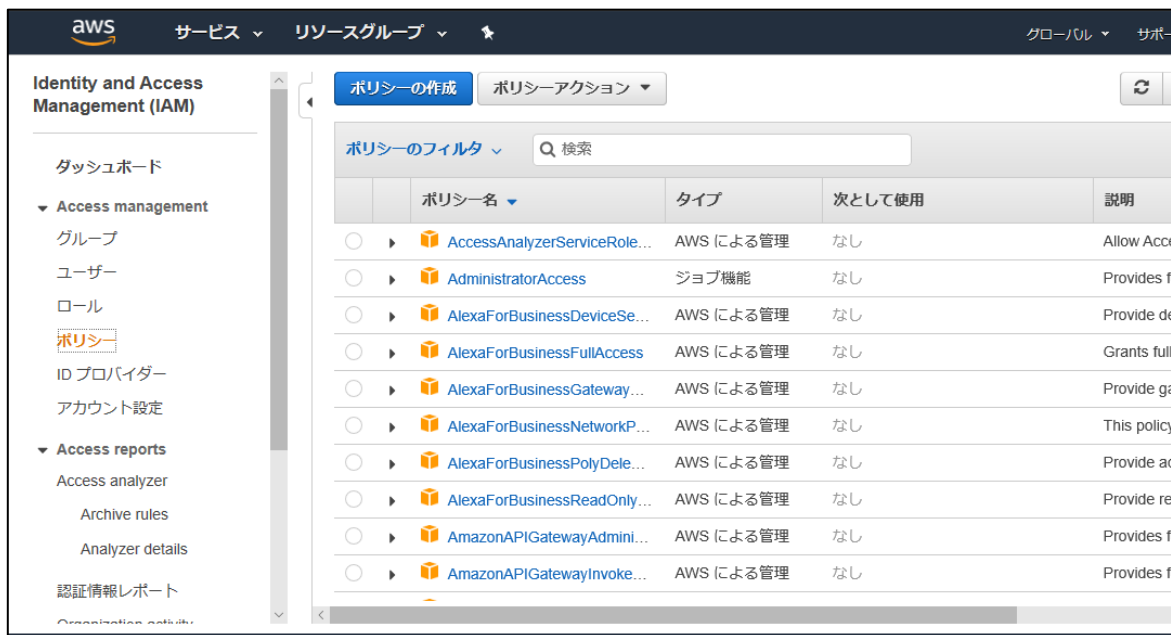


#### 手順4

AWS CLI で使用する AWS アカウントを登録します。AWS のルートアカウントは使用されないことを推奨します。

AWS Console (<https://aws.amazon.com/jp/console/>) を開いて、ログインしてください。この後の手順でユーザーとポリシーを登録するので、ログインユーザーにはその操作が可能な権限（ポリシー）が与えられている必要があります。

“IAM”サービスを開いてください。見つからない場合には、サービスの検索をしてください。



“ポリシー” を選択し、“ポリシーの作成” ボタンを押してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

ポリシーの作成

ポリシーにより、ユーザー、グループ、またはロールに割り当てることができる AWS アクセス権限が定義されます。ビジュアルエディタで JSON を使用してポリシーを作成または編集できます。詳細はこちら

ビジュアルエディタ JSON 管理ポリシーのインポート

すべて展開 | すべて折りたたむ

IoT (1つのアクション) クローン 削除

サービス IoT

アクション 許可されるアクションを IoT で指定 ? アクセス権限の拒否に切り替え ⓘ

閉じる

GetRegi

GetRegistrationCode ?

リソース 選択したアクションはすべてのリソースをサポートします。

“サービス” の項目には、“IoT” を選んでください。

“アクション” の検索を利用して、“GetRegistrationCode”を表示させ、✓を入れてください。

IoT (1つのアクション) ⚠️ 1つの警告 クローン 削除

サービス IoT

アクション 許可されるアクションを IoT で指定 ? アクセス権限の拒否に切り替え ⓘ

閉じる

UpdateC

UpdateCACertificate ?

UpdateCertificate ?

リソース Specify **cacert** resource ARN for the **UpdateCACertificate** アクション.  
Specify **cert** resource ARN for the **UpdateCertificate** アクション.

リクエスト条件 リクエスト条件の指定 (オプション)

再び、“アクション” の検索を利用して、“UpdateCACertificate”と”UpdateCertificate”を表示させ、✓を入れてください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



IoT (1つのアクション) クローン 削除

サービス IoT

アクション 読み込み  
GetRegistrationCode  
書き込み  
UpdateCACertificate  
UpdateCertificate

リソース  指定  
 閉じる  すべてのリソース

リクエスト条件 リクエスト条件の指定 (オプション)

さらにアクセス許可を追加する

キャンセル **ポリシーの確認**

“リソース”で“すべてのリソース”に✓を入れて、“ポリシーの確認”ボタンを押してください。

ポリシーの作成 1 2

ポリシーの確認

名前\* CLI\_USER\_POLICY  
英数字と「+、@、\_」を使用します。最大 128 文字。

説明  
最大 1000 文字。英数字と「+、@、\_」を使用します。

概要  
Q フィルター

サービス	アクセスレベル	リソース	リクエスト条件
許可 (214 サービス中 1) 残りの 213 を表示			
IoT	制限: 読み込み、書き込み	すべてのリソース	なし

\* 必須

キャンセル 戻る **ポリシーの作成**

“名前”の項目を入力して、“ポリシーの作成”ボタンを押して登録を完了させてください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



“ユーザー” を選択し、“ユーザーを追加” ボタンを押してください。



“ユーザー名” を入力し、“アクセスの種類” の“プログラムによるアクセス” に✓を入れてください。  
 “次のステップ” ボタンを押してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



ユーザーを追加

1 2 3 4 5

▼ アクセス許可の設定

ユーザーをグループに追加    アクセス権限を既存のユーザーからコピー    既存のポリシーを直接アタッチ

ポリシーの作成

ポリシーのフィルタ: CLI\_USER    1件の結果を表示中

ポリシー名	タイプ	次として使用
<input checked="" type="checkbox"/> CLI_USER_POLICY	ユーザーによる管理	なし

キャンセル    戻る    次のステップ: タグ

“既存のポリシーを直接アタッチ” を選択してください。

ポリシーのフィルタを利用して、先ほど登録したポリシーを表示させ、✓を入れてください。

“次のステップ” ボタンを押してください。

ユーザーを追加

1 2 3 4 5

タグの追加 (オプション)

IAM タグは、ユーザー に追加できるキーと値のペアです。タグには、E メールアドレスなどのユーザー情報を含めるか、役職などの説明文とすることができます。タグを使用して、このユーザー のアクセスを整理、追跡、制御できます。 [詳細はこちら](#)

キー	値 (オプション)	削除
<input type="text" value="新しいキーを追加"/>	<input type="text"/>	<input type="button" value="削除"/>

さらに 50 個のタグを追加できます。

キャンセル    戻る    次のステップ: 確認

“次のステップ” ボタンを押してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



ユーザーを追加

1 2 3 4 5

確認

選択内容を確認します。ユーザーを作成した後で、自動生成パスワードとアクセスキーを確認してダウンロードできます。

ユーザー詳細

ユーザー名 CLI\_USER

AWS アクセスの種類 プログラムによるアクセス - アクセスキーを使用

アクセス権限の境界 アクセス権限の境界が設定されていません

アクセス権限の概要

次のポリシー例は、上記のユーザーにアタッチされます。

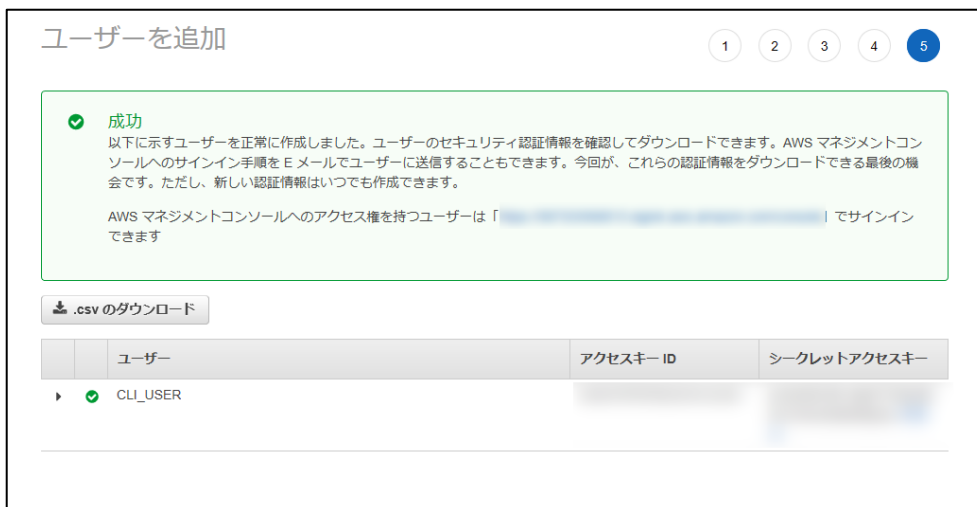
タイプ	名前
管理ポリシー	CLI_USER_POLICY

タグ

追加されたタグはありません。

キャンセル 戻る ユーザーの作成

“ユーザーの作成” ボタンを押してください。



ユーザーを追加

1 2 3 4 5

成功

以下に示すユーザーを正常に作成しました。ユーザーのセキュリティ認証情報を確認してダウンロードできます。AWS マネジメントコンソールへのサインイン手順を E メールでユーザーに送信することもできます。今回が、これらの認証情報をダウンロードできる最後の機会です。ただし、新しい認証情報はいつでも作成できます。

AWS マネジメントコンソールへのアクセス権を持つユーザーは「 [redacted] 」でサインインできます

.csv のダウンロード

ユーザー	アクセスキー ID	シークレットアクセスキー
CLI_USER	[redacted]	[redacted]

ここに表示されている“アクセスキーID (AccessKey)”とシークレットアクセスキー (SecretKey) を後ほど使用しますので、記録もしくは“.csv のダウンロード” ボタンでダウンロードしておいてください。

AccessKey と SecretKey は、他に流出しないように注意してください。流出してしまった場合には、先ほどの登録したユーザーを削除すれば問題ありません。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



## 手順5

AWS のドキュメントに詳細な説明があります

([https://docs.aws.amazon.com/ja\\_jp/iot/latest/developerguide/device-certs-your-own.html](https://docs.aws.amazon.com/ja_jp/iot/latest/developerguide/device-certs-your-own.html))

一部のコマンドが違うので、注意してください。

以降のコマンドは、全てコマンドプロンプトで実行されていきます。コマンドプロンプトは、Windows の「スタートメニュー」を開き、「cmd」を検索して実行すると起動します。

### コマンド) **aws configure**

手順3で入手した AccessKey と SecretKey を入力します

リージョンは、「ap-northeast-1」、出力フォーマットは、「json」を選択してください。

クライアント側のルート証明書を作成します。AWS IoT は、登録されたルート証明書を使用できるので認証局に登録する必要はありません。作成者は紛失しないよう管理してください。

### コマンド) **openssl genrsa -out rootCA.key 2048**

“rootCA.key”というファイルが出力されます。このファイルは、ルート証明書の秘密鍵なので、流出することが無いように管理してください。

### コマンド) **openssl req -x509 -new -nodes -key rootCA.key -sha256 -days 1024 -out rootCA.pem**

“rootCA.pem”というファイルが出力されます。このファイルは、ルート証明書の公開鍵なので、他に渡しても問題ありません。AWS には、このファイルがアップロードされます。

```
Country Name (2 letter code) [AU]:JP
State or Province Name (full name) [Some-State]:Kanagawa
Locality Name (eg, city) []:Yamato-Shi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Nomura Engineering Co.,Ltd
Organizational Unit Name (eg, section) []:Tech
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

“Country Name”には、「JP」と入力してください。それ以外は、自由に入力することができます。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

コマンド) **aws iot get-registration-code**

AWS IoT の登録コードが出力されます (AWS のコンソール画面でも確認できます)。

```
C:\Users% >aws iot get-registration-code
{
  "registrationCode": "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
}
```

赤線の部分をコピーして、保管してください。後ほど使用します。

コマンド) **openssl genrsa -out verificationCert.key 2048**

“verificationCert.key”というファイルが出力されます。AWS IoT にルート証明書を登録するには、先ほどの登録コードを AWS IoT に渡す必要があります。AWS IoT は、証明書の情報の一部として渡すことで実現しています。このコマンドでは、その証明書 (検証証明書) 用の鍵を作成しています。

コマンド) **openssl req -new -key verificationCert.key -out verificationCert.csr**

“verificationCert.csr”というファイルが出力されます。

```
Country Name (2 letter code) [AU]:JP
State or Province Name (full name) [Some-State]:Kanagawa
Locality Name (eg, city) []:Yamato-Shi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Nomura Engineering Co.,Ltd
Organizational Unit Name (eg, section) []:Tech
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

“Country Name”には、“JP”と入力し、“Common Name”には、先ほどの AWS IoT の登録コードを貼り付けてください。それ以外は、自由に入力することができます。

コマンド) **openssl x509 -req -in verificationCert.csr -CA rootCA.pem -CAkey rootCA.key (続く)  
 -CAcreateserial -out verificationCert.pem -days 500 -sha256**

“verificationCert.pem”というファイルが出力されます。AWS IoT には、このファイルがアップロードされます。

コマンド) **aws iot register-ca-certificate --ca-certificate file://rootCA.pem (続く)  
 --verification-cert file://verificationCert.pem**

AWS IoT にクライアント側のルート証明書 (rootCA.pem) と検証証明書 (verificationCert.pem) をアップロードします。成功すると、AWS IoT から証明書 ID (certificateId) が返ってきます。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



## 手順6

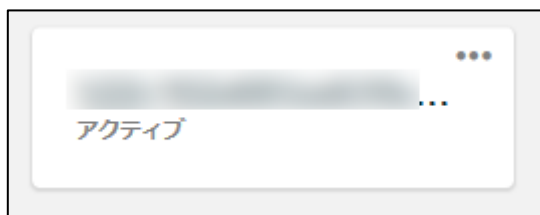
“IoT Core”サービスを開いてください。



“安全性” → “CA” を選択してください。先ほど登録したクライアント側のルート証明書が“無効”の状態が表示されています。表示されている番号は、証明書 ID になります。



“...” を選択すると、サブメニューが表示されます。その中の“有効化”を選択してください。



“無効” から “アクティブ” に変化したことを確認してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

## 手順7

“モノ”に関連付けるデバイス証明書を作成します。SRPC\_MBL の台数分、この手順を繰り返してください。

以降のコマンドは、全てコマンドプロンプトで実行されていきます。コマンドプロンプトは、Windows の「スタートメニュー」を開き、“cmd”を検索して実行すると起動します。

コマンド) **openssl genrsa -out deviceCert.key 2048**

“deviceCert.key”というファイルが出力されます。このファイルはデバイス証明書の秘密鍵なので、流出することが無いようにしてください。

コマンド) **openssl req -new -key deviceCert.key -out deviceCert.csr**

“deviceCert.csr”というファイルが出力されます。デバイス証明書の情報を入力します。

```
Country Name (2 letter code) [AU]:JP
State or Province Name (full name) [Some-State]:Kanagawa
Locality Name (eg, city) []:Yamato-Shi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Nomura Engineering Co.,Ltd
Organizational Unit Name (eg, section) []:Tech
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

AWS IoT の登録コードはもう使用しないので、“Common Name”に貼り付ける必要はありません。

コマンド) **openssl x509 -req -in deviceCert.csr -CA rootCA.pem -CAkey rootCA.key (続く)**  
**-set\_serial 1 -out deviceCert.pem -days 500 -sha256**

“deviceCert.pem”というファイルが出力されます。このファイルはデバイス証明書の公開鍵なので、他に渡しても問題ありません。AWS IoT にはこのファイルをアップロードします。

AWS IoT のドキュメントには、-CAcreateserial と記述されていますが、SRPC-MBL ではこのオプションで自動で作成されたシリアル番号の桁数をサポートできないので、-set\_serial X でシリアル番号を手動で設定してください。

コマンド) **aws iot register-certificate --certificate-pem file://deviceCert.pem (続く)**  
**--ca-certificate-pem [file://rootCA.pem](#)**

AWS IoT にクライアント側のルート証明書(rootCA.pem)と一緒に、デバイス証明書(deviceCert.pem)をアップロードします。成功すると、AWS IoT から証明書 ID (certificateId) が返ってきます。

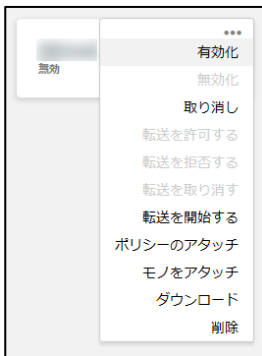
製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

### 手順8

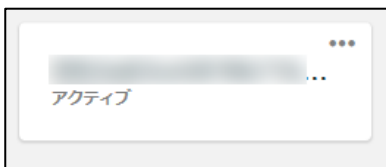
“IoT Core”サービスを開いてください。



“安全性” → “証明書” を選択してください。先ほど登録したデバイス証明書が“無効”の状態が表示されています。表示されている番号は、証明書 ID になります。

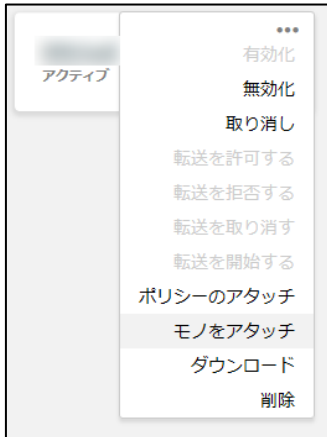


“...” を選択するとサブメニューが表示されます。  
“有効化” を選択してください。

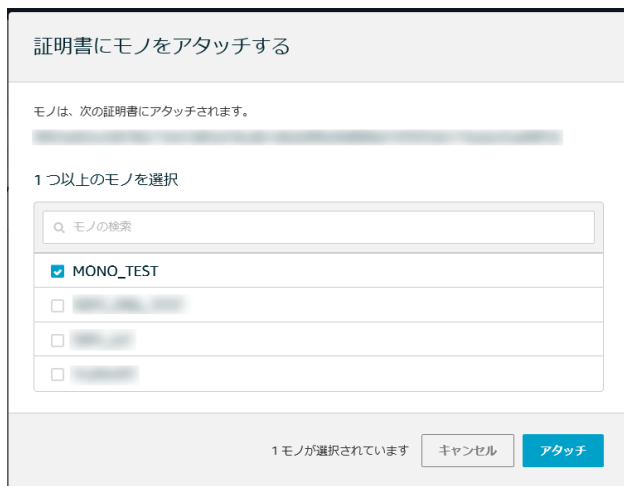


“無効” から “アクティブ” に変化したことを確認してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



“...” を選択するとサブメニューが表示されます。  
“モノをアタッチ” を選択してください。



登録した“モノ”を選択して、“アタッチ”ボタンを押してください。

以上で AWS 側の初期設定は完了になります。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

## 手順9

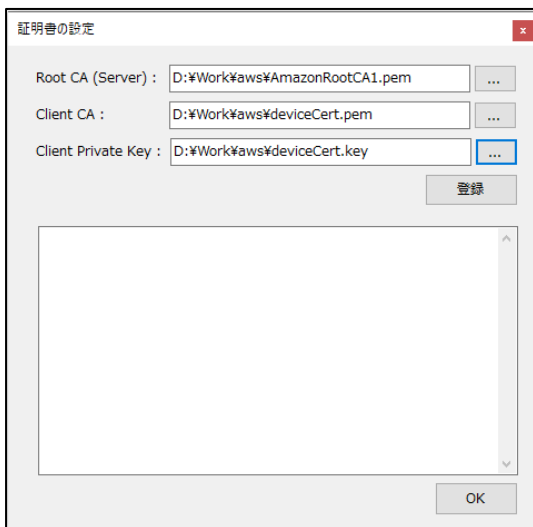
SRPC-MBL に AWS のルート証明書 (AmazonRootCA1.pem)、デバイス証明書の公開鍵 (deviceCert.pem)、デバイス証明書の秘密鍵 (deviceCert.key) を登録します。

注意！！ 先に SIM カードの設定を行う必要があります。

SRPC-MBL のディップスイッチ#1 を ON にして、メンテナンスソフトに接続してください。



アイコンを右クリックすると、サブメニューが表示されます。“証明書の設定” を選択してください。

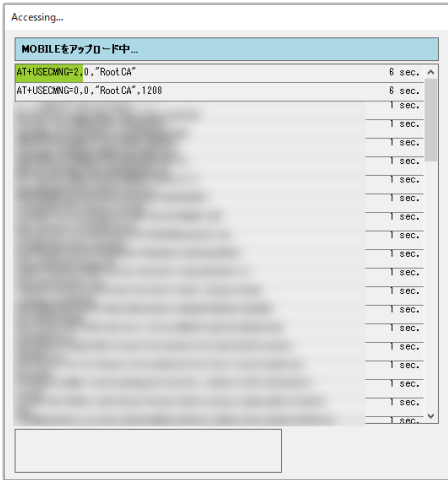


Root CA の欄には、AWS のルート証明書、Client CA の欄には、デバイス証明書の公開鍵、Client Private Key の欄には、デバイス証明書の秘密鍵のファイルパスを入力してください。

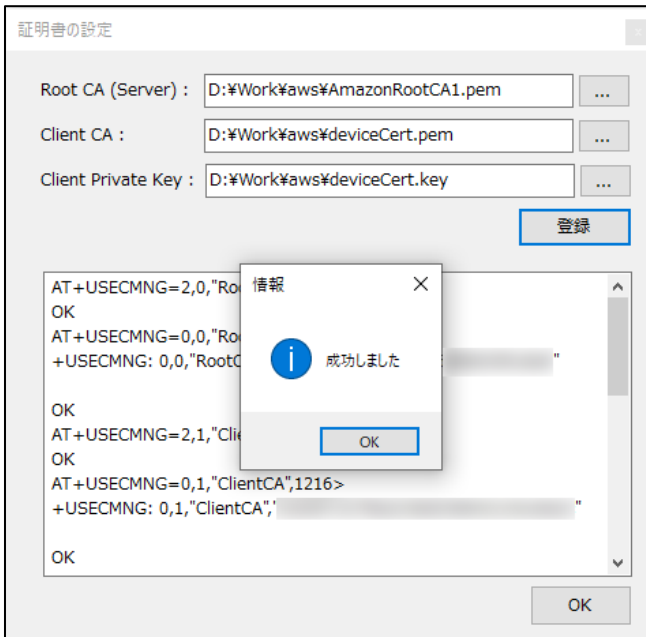
“...” ボタンを押すと、参照ダイアログが表示されません。

登録ボタンを押すと、空欄になっていない証明書がインストールされます。全て空欄の場合、証明書の登録の確認が行えます。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



インストール中は、上記の画面が表示されています。インストールが終了すると自動で閉じられます。



3種類の証明書が全て登録されていた場合、“成功しました”と表示されます。

SRPC-MBL に登録された証明書は抜き出すことができないようになっています。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



AWS の送信先を確認するために、“IoT Core”サービスを開いてください。



“設定”を選択すると、右画面に“エンドポイント”が表示されます。この“エンドポイント”を送信先に設定するので、記録しておいてください。



アイコンを右クリックすると、サブメニューが表示されます。“ネットワークの設定”を選択してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

“サーバー名”の欄に、先ほどの“エンドポイント”を貼り付けてください。

“ポート番号”の欄は、8883 固定になります。

“パス名”の欄は、トピック名を入力してください。

“プロトコル”の枠内の、“MQTT”を選択して、“Thing名”を入力してください。“Thing名”はAWS IoT サービスに登録されている必要があります。

メンテナンスソフトから切断して、SRPC-MBL のディップスイッチ#1 を OFF にすると、AWS IoT への送信が開始されます。

確認するために、“IoT Core”サービスを開いてください。

“テスト”を選択した後、“トピックのサブスクリプション”の欄に“#”を入力して、“トピックのサブスクリライブ”ボタンを押してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。





設定されたパス名（トピック名）でデータが送られてくることを確認してください。インターバル間隔の設定によっては、数分かかることもあるのでインターバル間隔は事前に把握しておいてください。

SRPC-MBL からは、“RAW” の項目だけが上がります。“RAW”の項目は、1バイトを2文字の16進数で表現したデータになっています。データを抜き出すために、Lambda関数等を使用することができます。

以上で、初期設定は完了です。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

## 使用方法

メンテナンスソフトを利用する場合の使用方法について

メンテナンスソフトが使用する AWS IoT の証明書を準備します。上記の手順を例にすると、deviceCert.key、deviceCert.pem、rootCA.pem の3つになります。

メンテナンスソフトが制御するために、上記の3つのファイルを1つのファイル（PKCS12）に纏めます。

以降のコマンドは、全てコマンドプロンプトで実行されていきます。コマンドプロンプトは、Windows の「スタートメニュー」を開き、「cmd」を検索して実行すると起動します。

コマンド) **openssl pkcs12 -export -out deviceCert.pfx -inkey deviceCert.key -in deviceCert.pem -certfile rootCA.pem**

上記のコマンドを実行すると、パスワードが求められます。入力したパスワードはメンテナンスソフトの設定で使用します。

“deviceCert.pfx”というファイルが作成されます。

メンテナンスソフトを起動してください。

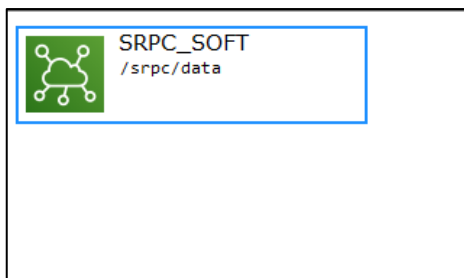


左ツリー内の“AWS IoT”を右クリックし、“インターネットに接続”を選択してください。または、上のツールバー内の“AWS IoT アイコン”を押してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

- |                   |  |
|-------------------|--|
| AWS ルート証明書        | … アマゾンの証明書 (AmazonRootCA1.pem) へのファイルパス                      |
| クライアント証明書(pkcs12) | … 作成した PKCS ファイル (deviceCert.pfx) へのファイルパス                   |
| (パスワード)           | … PKCS ファイル (deviceCert.pfx) を作成したときに入力したパスワード               |
| エンドポイント           | … AWS IoT のエンドポイント   |
| ポート番号             | … AWS IoT のポート番号 (8883)                                      |
| クライアント ID         | … SRPC-MBL のモノ名以外であれば問題ありません                                 |
| トピック名             | … ここで設定されたトピックに対してサブスクライブするため、SRPC-MBL の設定で入力したパス名と同じにしてください |

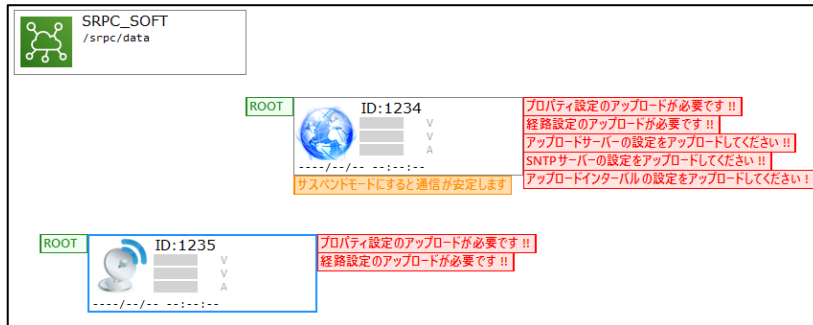
接続ボタンを押してください。



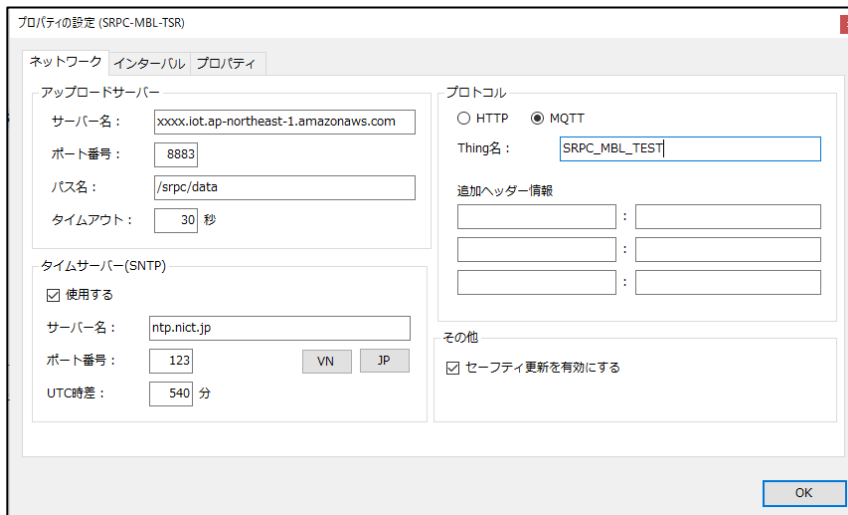
デザインビューに AWS IoT アイコンが表示されれば、接続は成功になります。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

SRPC-MBL と SRPC を追加します



SRPC-MBL を右クリックし、“プロパティの設定” を選択してください。

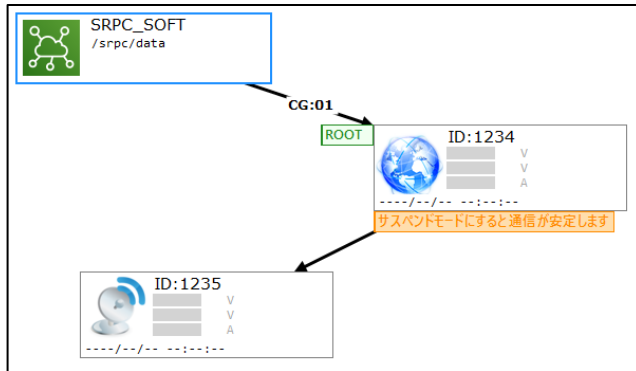


- サーバー名 … AWS IoT のエンドポイント
- ポート番号 … AWS IoT のポート番号 (8883)
- パス名 … SRPC-MBL がパブリッシュするトピック名
- MQTT … 選択してください
- Thing 名 … SRPC-MBL のモノ名

OK ボタンを押して画面を閉じてください。メンテナンスソフト上の設定情報なので、OK ボタンを押しても SRPC-MBL は更新されません。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

赤い警告は、“サービス”メニューの“全てアップロード済みにする”を選択すると表示されなくなります。アイコンを右クリックし、“親ノードの設定”や“起点 SRPC の設定”を行い、経路を設定してください。



“サービス”メニューの“開始”を実行してください。“開始”を実行しても、SRPC-MBL や SRPC の設定情報は変更されません。変更する場合には、アップロードを使用します。

メンテナンスソフトが起動している間に AWS IoT に送られたデータを受信して画面に表示していきま  
 す。メンテナンスソフトが起動していない間のデータは破棄されます。破棄されないようにするには、AWS  
 サービスを使用してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。

## 工場出荷時

弊社で初期設定を全て行ってから出荷できます。その場合には、御社で契約している AWS に初期設定で使用するユーザーを作成して頂いて、そのユーザーの AccessKey と SecretKey を弊社に送ってください。初期設定が終了したら、そのユーザーを無効にさせていただいて結構です。

弊社で証明書を作成して、AWS にアップロードしたらご連絡致します。その後、御社でアップロードされた証明書を有効化してください。通信テストが完了したら製品と一緒に証明書一式を全てお送りいたします。

セキュリティの関係上、弊社で行ってはいけない場合には、御社で初期設定を行ってください。弊社ではテスト用の証明書を使用して、弊社が契約している AWS IoT サービスに正常に通信できることを確認してから出荷致します。

## フォーマット

回収したデータは、“RAW”という項目で AWS IoT に上がります。

“RAW”データは、1byte を 2 文字の 16 進数で表現したデータになります。

詳しくは、別紙のデータシートを参照してください。

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。



## 変更履歴

2019.12.11	Rev0.1	新規作成
2020.01.30	Rev0.2	メンテナンスソフト部分追記

製品の故障や誤動作が直接人命に関わるような使い方は絶対にしないで下さい。