# KEIm-CVSoC での AWS IoT Greengrass の開始方法

## 1. はじめに

このドキュメントでは、KEIm-CVSoC 開発キットで Ubuntu 18.04 を動作させ AWS Greengrass を実行する方法について説明します。
開発キットの接続や設定、起動の方法などについては、KEIm-CVSoC開発キットスタートアップガイドをご参照ください。

## 2. 開発環境の準備

下記の機材を準備しておく必要があります:

**KEIm-CVSoC 開発キット**

- KEIm-CVSoC 開発キットのパッケージに同梱されているもの以外に、下記のものが必要になります。
  - **16GB Micro SD カード**
  - **Ethernet ケーブル**

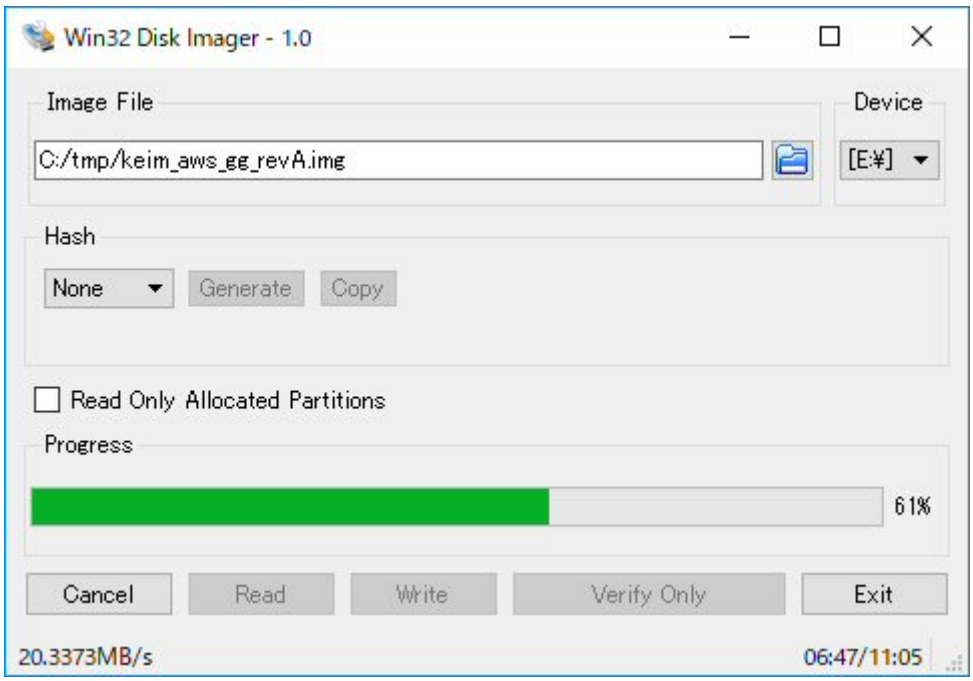
**ホストPC**

- Windows 10 がインストールされた PC を用意します。
  - **7-Zip 解凍ツール**
  - **Win32DiskImager** などの Micro SD カードにブートイメージを書き込めるツール
  - **TeraTerm**

## 3. 動作環境の構築

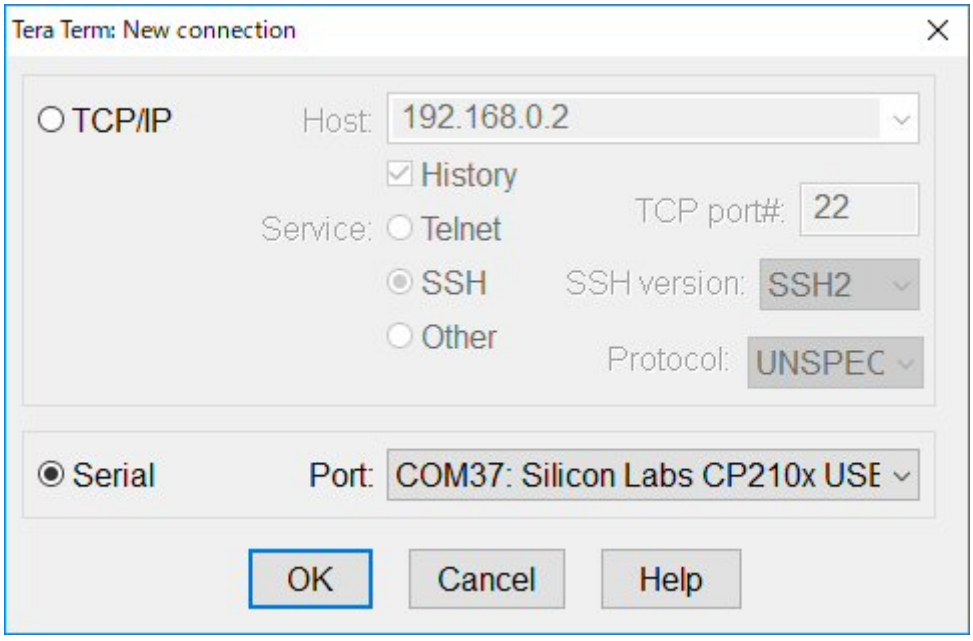
### 3-1. SD カードの準備

まず、 SD カードに起動可能なイメージを書き込む必要があります。
起動可能なイメージは、こちら からダウンロードできます。
イメージには ubuntu 18.04 がインストールされており、SD カードを KEIm-CVSoC に挿入し、電源を投入するだけで Linux が起動されます。
起動可能なイメージは 7-Zip で圧縮されていますので、解凍してから SD カードに書き込んでください。

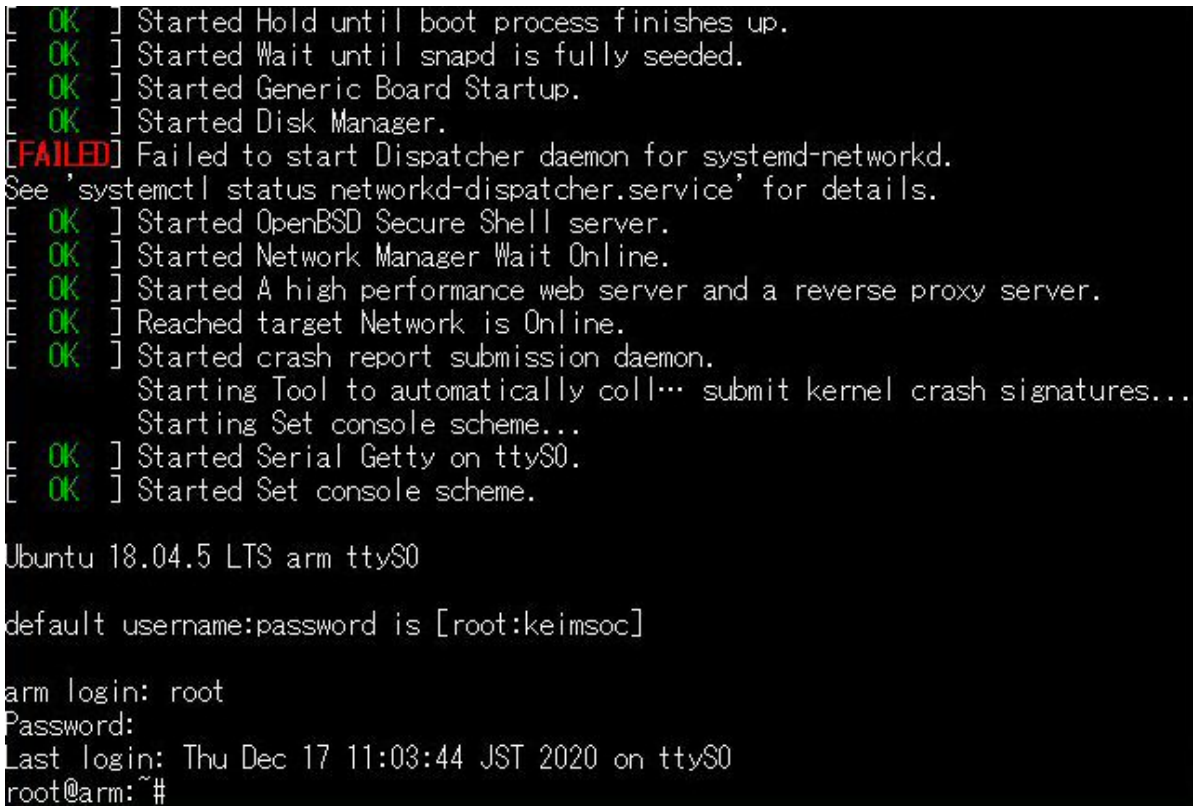



## 3-2. KEIm-CVSoC の起動

KEIm-CVSoC開発キットスタートアップガイド を参考に、Ethernet ケーブル や USB ケーブルを接続します。
ホスト PC の TeraTerm を起動して、KEIm-CVSoC の UART にシリアル接続します。




Linux が起動したら、ユーザー : **root** 、パスワード : **keimsoc** でログインしてください。




ログインが完了したら、外部ネットワークに接続するための環境設定などを行ってください。

# 4. AWS Greengrass Core のインストール

## 4-1. ユーザーとグループを設定

TeraTerm から下記のコマンドで新しいユーザとグループを作成します。

```
sudo adduser --system ggc_user
sudo addgroup --system ggc_group
```

## 4-2. Greegrass 依存性チェッカーの実行

greengrass-dependency-checker を使用して Greengrass が動作する環境かどうかのチェックを行います。

```
mkdir -p /root/Downloads/greengrass-dependency-checker-GGCv1.11.x
cd /root/Downloads/greengrass-dependency-checker-GGCv1.11.x
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
modprobe configs
sudo ./check_ggc_dependencies | more
```

シェルスクリプトの出力をチェックして、Greengrass が動作可能な環境である事を確認します。
なお、 NodeJS 12.x は必要に応じてインストールすれば問題ありません。

```
==========Checking script dependencies==========
The device has all commands required for the script to run.

==========Dependency check report for GGC v1.11.x==========
System configuration:
Kernel architecture: armv7l
Init process: /lib/systemd/systemd
Kernel version: 5.4.54
C library: Ubuntu GLIBC 2.27-3ubuntu1.2
C library version: 2.27
Directory /var/run: Present
/dev/stdin: Found
/dev/stdout: Found
/dev/stderr: Found
----------Commands and software packages----------
Python 2.7 version: 2.7.17
Python 3.7 version: 3.7.5
NodeJS 12.x: Not found
Java version: 1.8.0_265
wget: Present
--More--
realpath: Present
tar: Present
readlink: Present
basename: Present
dirname: Present
pidof: Present
df: Present
grep: Present
umount: Present
mv: Present
gzip: Present
mkdir: Present
rm: Present
ln: Present
cat: Present
cut: Present
/bin/bash: Present

----------Platform security----------
Hardlinks_protection: Enabled
Symlinks protection: Enabled

--More--
----------User and group----------
ggc_user: Present
ggc_group: Present

----------(Optional) Greengrass container dependency check----------

----------Kernel configuration----------
Kernel config file: /proc/config.gz

Namespace configs:
CONFIG_IPC_NS: Enabled
CONFIG_UTS_NS: Enabled
```

```
CONFIG_USER_NS: Enabled
CONFIG_PID_NS: Enabled

Cgroup configs:
CONFIG_CGROUP_DEVICE: Enabled
CONFIG_CGROUPS: Enabled
CONFIG_MEMCG: Enabled

--More--
Other required configs:
CONFIG_POSIX_MQUEUE: Enabled
CONFIG_OVERLAY_FS: Enabled
CONFIG_HAVE_ARCH_SECCOMP_FILTER: Enabled
CONFIG_SECCOMP_FILTER: Enabled
CONFIG_KEYS: Enabled
CONFIG_SECCOMP: Enabled
CONFIG_SHMEM: Enabled

-----------------------------------Cgroups check-------------------------
---------
Cgroups mount directory: /sys/fs/cgroup

Devices cgroup: Enabled and Mounted
Memory cgroup: Enabled and Mounted


-----------------------------------Results-------------------------------
---------
Note:
1. It looks like the kernel uses 'systemd' as the init process. Be sure to set t
he
'useSystemd' field in the file 'config.json' to 'yes' when configuring Greengras
--More--
'useSystemd' field in the file 'config.json' to 'yes' when configuring Greengras
s core.

Missing optional dependencies:
1. Could not find the binary 'nodejs12.x'.

If NodeJS 12.x or later is installed on the device, name the binary 'nodejs12.x'
 and
add its parent directory to the PATH environment variable. NodeJS 12.x or later
is
required to execute NodeJS lambdas on Greengrass core.

Supported lambda isolation modes:
No Container: Supported
Greengrass Container: Supported

---------------------------------Exit status---------------------------
---------
You can now proceed to installing the Greengrass core 1.11.x software on t
he device.
Please reach out to the AWS Greengrass support if issues arise.

root@arm:~/Downloads/greengrass-dependency-checker-GGCv1.11.x/greengrass-depende
ncy-checker-GGCv1.11.x#
```

## 4-3. Greengrass Core セキュリティリソースを入手する

こちら の手順に従い、セキュリティリソース（hash-setup.tar.gz）をダウンロードしてください。

## 4-4. Greengrass Core ソフトウェアのダウンロード

Greengrass Core ソフトウェアはこちら からダウンロードしてください。
本ドキュメントでは、v1.11 を使用しています。

| アーキテクチャ | 配信 | OS | リンク |
|---|---|---|---|
| Armv8 (AArch64) | Arch Linux | Linux | ダウンロード |
| Armv8 (AArch64) | OpenWrt | Linux | ダウンロード |
| Armv7l | Raspbian | Linux | ダウンロード |
| Armv7l | OpenWrt | Linux | ダウンロード |
| Armv6l | Raspbian | Linux | ダウンロード |
| x86_64 | Linux | Linux | ダウンロード |

ARMv7hf Ubuntu 18.04 用の Greengrass Core が無いため、ARMv7l Raspbian 用の Core を使用します。
Raspbian と Ubuntu は Debian Linux から派生しているため、代わりに Raspbian 用の　Core を使用する事ができます。

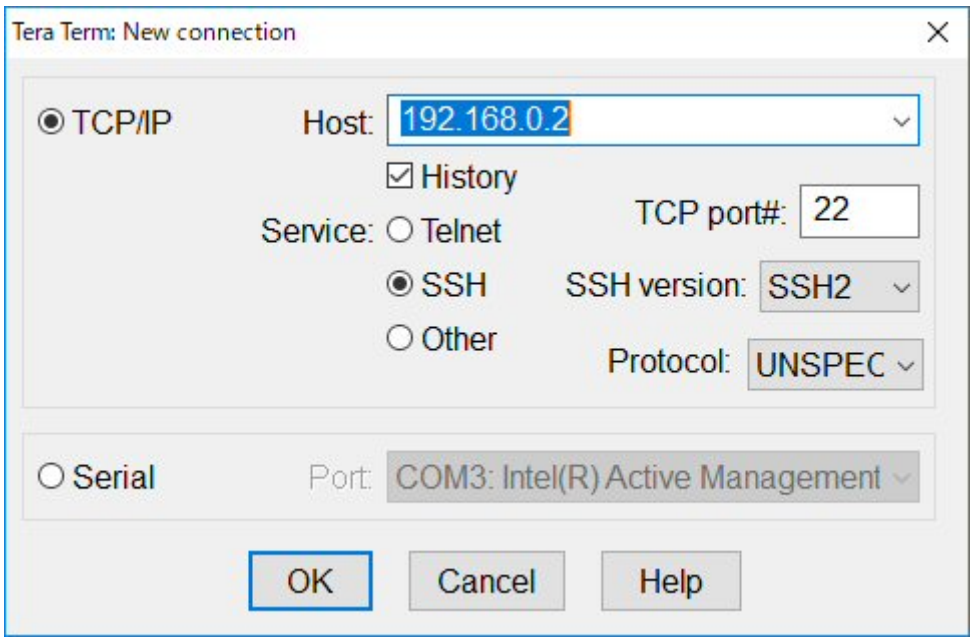
## 4-5. Greengrass Core ソフトウェアと Greengrass セキュリティリソースのコピー

シリアルコンソールで ip アドレスを取得し、

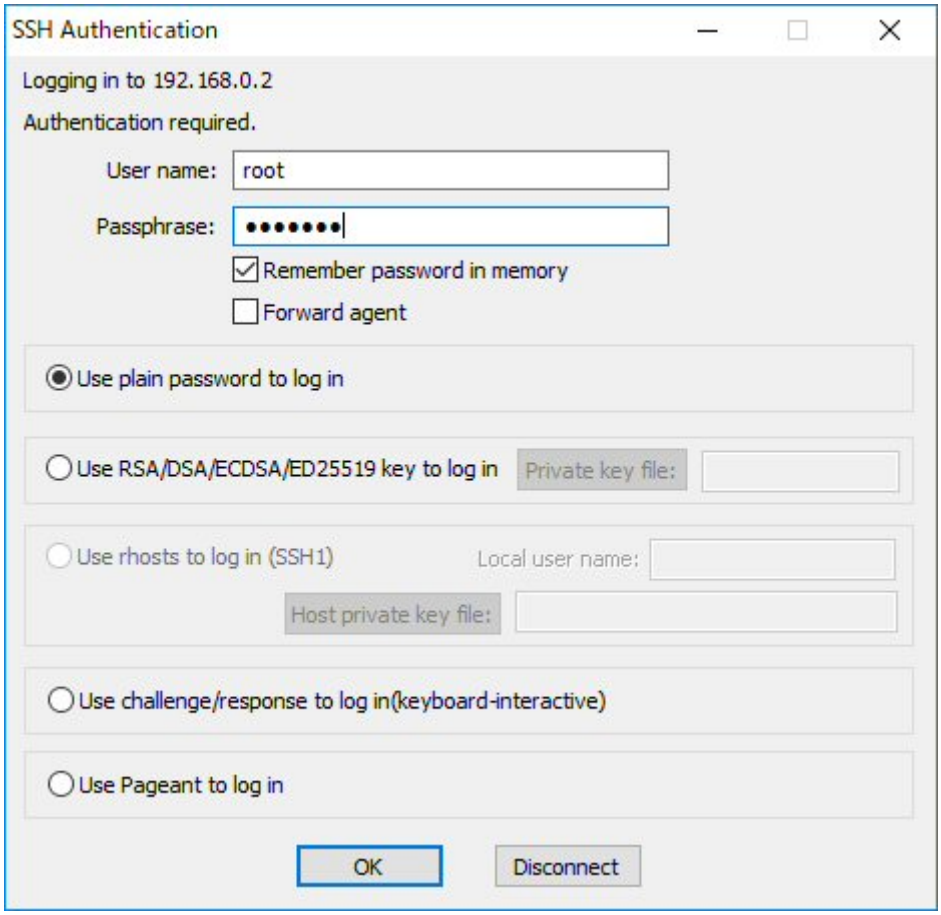
```
ip addr
```

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,DYNAMIC,UP,LOWER_UP> mtu 1500 qdisc mq state UP gr
oup default qlen 1000
    link/ether 22:d7:32:be:72:fd brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.2/24 brd 192.168.0.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 240f:7f:119:1:20d7:32ff:febe:72fd/64 scope global dynamic mngtmpaddr
       valid_lft 3068710468sec preferred_lft 3068710468sec
    inet6 fe80::20d7:32ff:febe:72fd/64 scope link
       valid_lft forever preferred_lft forever
3: sit0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group default qlen 1000
    link/sit 0.0.0.0 brd 0.0.0.0
```

TeraTerm で SSH 接続します。

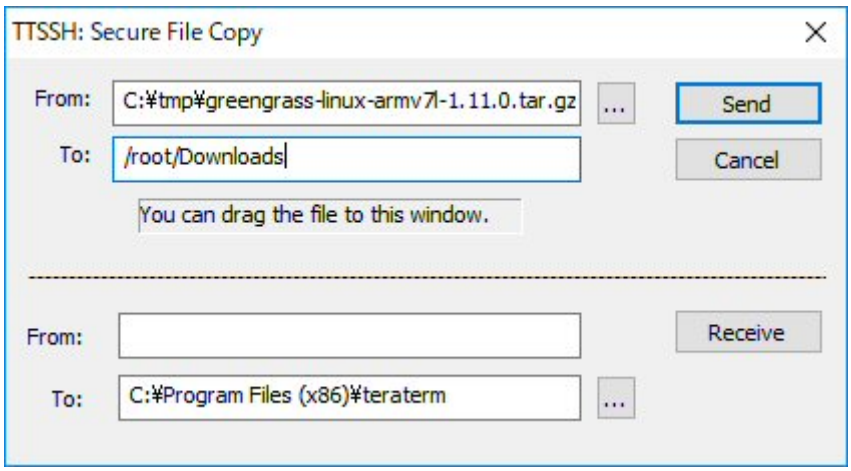


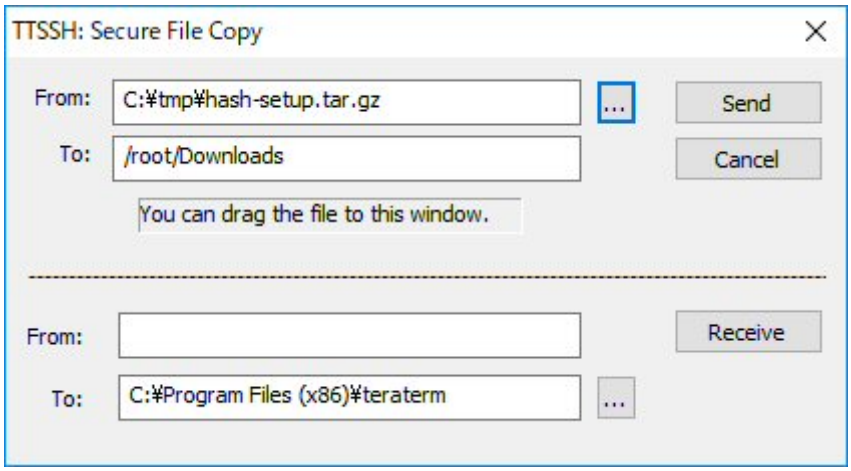

SSH Authentication 画面が表示されたら、*User name:* に **root**、*Passphrase:* に **keimsoc** を入力して *OK* をクリックします。




正常に接続されるとターミナルにコマンドプロンプトが出力されます。

続いて、メニューの *File* ⇒ *SSH CSP...* を開いて、 *Send* 側の *From:* に 4-3. と 4-4. でダウンロードしたファイルを、*To:* に転送先となる */root/Downloads* を入力して *Send* をクリックします。




転送が正常に行われた事を、シリアルコンソールから確認してください。

```
root@arm:~/Downloads# ls
greengrass-dependency-checker-GGCv1.11.x  hash-setup.tar.gz
greengrass-linux-armv7l-1.11.0.tar.gz
root@arm:~/Downloads#
```

## 4-6. Greengrass Core とセキュリティリソースの展開

転送した２つのファイルを展開します。

```
cd /root/Downloads
tar -xvzf greengrass-linux-armv7l-1.11.0.tar.gz -C /
tar -xvzf hash-setup.tar.gz -C /greengrass
```

```
de/__init__.py
greengrass/ggc/packages/1.11.0/runtime/python/greengrass_ipc_python_sdk/ipc_faca
de/client.py
greengrass/ggc/packages/1.11.0/runtime/python/greengrass_ipc_python_sdk/utils
greengrass/ggc/packages/1.11.0/runtime/python/greengrass_ipc_python_sdk/utils/__
init__.py
greengrass/ggc/packages/1.11.0/runtime/python/greengrass_ipc_python_sdk/utils/ex
ponential_backoff.py
greengrass/ggc/packages/1.11.0/runtime/python/lambda_runtime.py
greengrass/ota
greengrass/ota/ota_agent
greengrass/ota/ota_agent_v1.3.0
greengrass/ota/ota_agent_v1.3.0/LICENSE
greengrass/ota/ota_agent_v1.3.0/LICENSE/LICENSE
greengrass/ota/ota_agent_v1.3.0/LICENSE/THIRD-PARTY-LICENSES
greengrass/ota/ota_agent_v1.3.0/ggc-ota
greengrass/ota/ota_agent_v1.3.0/ggc-ota-1
greengrass/ota/ota_agent_v1.3.0/ggc-ota-1-1
root@arm:~/Downloads# tar -xvzf hash-setup.tar.gz -C /greengrass
certs/[redacted].cert.pem
certs/[redacted].private.key
certs/[redacted].public.key
config/config.json
root@arm:~/Downloads#
```

## 4-7. AWS からルート CA をコピー

```
cd /greengrass/certs/
sudo wget -O root.ca.pem
https://www.amazontrust.com/repository/AmazonRootCA1.pem
```

```
root@arm:~/Downloads# cd /greengrass/certs/
root@arm:/greengrass/certs# sudo wget -O root.ca.pem https://www.amazontrust.com
/repository/AmazonRootCA1.pem
--2020-12-17 14:42:53--  https://www.amazontrust.com/repository/AmazonRootCA1.pe
m
Resolving www.amazontrust.com (www.amazontrust.com)... 65.9.42.36, 65.9.42.95, 6
5.9.42.41, ...
Connecting to www.amazontrust.com (www.amazontrust.com)|65.9.42.36|:443... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 1188 (1.2K) [text/plain]
Saving to: ‘root.ca.pem’

root.ca.pem         100%[===================>]   1.16K  --.-KB/s    in 0s

2020-12-17 14:42:53 (47.1 MB/s) - ‘root.ca.pem’ saved [1188/1188]

root@arm:/greengrass/certs#
```

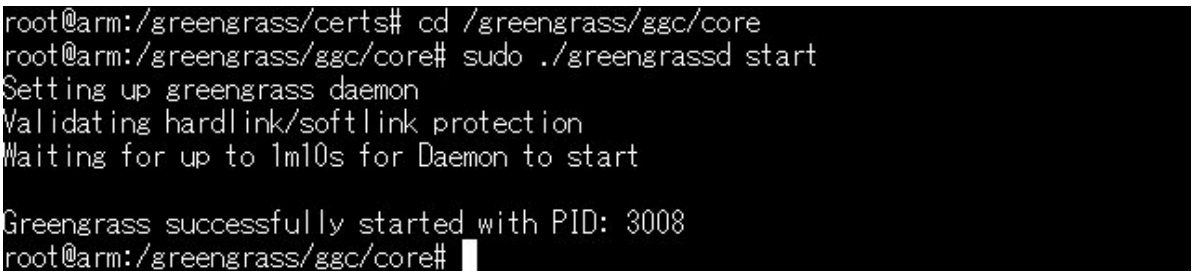
root.ca.pem　に内容があることを確認してください。

```
cat root.ca.pem
```

root.ca.pem　が空の場合にはネットワークに問題がある可能性がありますので、ネットワーク設定やネットワーク管理者に確認してください。

## 4-8. Greengrass Core を実行

```
cd /greengrass/ggc/core
sudo ./greengrassd start
```




greengrassd　を実行できない場合は、こちらの ドキュメントを参照してください。

## 以降、こちらの手順に従って、KEIm-CVSoC で Lambda 関数を実行できるようになります。