Getting Started Guide for AWS IoT Greengrass

Secure Edge Node (SEN)
Secure Compute Module (SCM) Dev Kit
Secure Compute Module (SCM)

Table of Contents

Rev 1.2 7/12/2023

1 Document information

1.1 Document revision history

DATE	VERSION	COMMENTS
7/7/2023	1.0	Initial release
7/11/2023	1.1	Added ref to dev kit
7/12/2023	1.2	Incorporated AWS input from Zymkey doc

1.2 Applicable operating systems for this guide

This guide will show you how to setup AWS IoT Greengrass v2 HSI features secured by a Zymbit Secure Compute Module (SCM) or Secure Edge Node (SEN), running Raspberry PI OS Bullseye 64bit. The instructions include how to run an example script accessing the SCM/SEN Python API. Also included are the setup steps to test with AWS IoT Device Tester for AWS IoT Greengrass V2.

2 Overview

The Zymbit Secure Compute Module (SCM) provides private keys for X.509 certificate maintained in hardware through a PKCS#11 key store. The Zymbit Secure Edge Node (SEN) is a fully enclosed, edge node with an integrated Secure Compute Module (SCM). The functionality of the SCM and the SEN with an integrated SCM are identical. The SCM is also available as a developer's kit which includes no case, just the SCM and an IO carrier board.

AWS IoT Greengrass Core software uses a private key and X.509 certificate to authenticate connections to the AWS IoT and AWS IoT Greengrass services. The SCM stores all private keys in hardware. The SCM/SEN can be configured to use either of the supported AWS IoT Greengrass signature schemes, RSA-2048 or EC keys.

3 Hardware description

3.1 Datasheet

Secure Compute Platform Overview

3.2 Standard kit contents

The Secure Compute Module (SCM) is available individually, as part of a Development Kit, or integrated into the Secure Edge Node (SEN).

Secure Compute Module
Secure Compute Module - Developer Kit 2
Secure Edge Node D35

For more details, please visit: Secure Compute Module - Overview

3.3 User provided items

Power supply (can be purchased separately from Zymbit), ethernet. Optionally HDMI display, keyboard, and mouse.

3.4 3rd party purchasable items

The following Power Supply option is available from Zymbit:
12VDC, 2.5A Power Supply, International

4 Set up your development environment

4.1 Tools installation (IDEs, Toolchains, SDKs)

The SCM/SEN comes pre-installed with the Raspberry Pi OS (bullseye 64bit lite) and the Zymbit software. No other software tools are necessary for the development environment of the Zymbit software. All software necessary for AWS IoT Greengrass is referenced below.

4.2 Additional software references

See docs.zymbit.com for additional information and examples.

5 Set up device hardware

As shipped, all software has been installed for you. Access to the SCM/SEN is only available via SSH.

1. Connect an ethernet cable to the ethernet connector
2. Connect the 12V power supply to the Power 12V power barrel
3. Power up.

The boot cycle can take 1-2 minutes. Monitor the Blue LED. It will flash a series of codes, and once it successfully boots, it will flash once every three seconds. You can now login via ssh. The default hostname is zymbit-dev and the login is zymbit with a password of zymbit. Please change the default hostname and login/password.

Details of the default configuration and start up procedure can be found in the Getting Started guide.

6 About AWS IoT Greengrass

To learn more about AWS IoT Greengrass, see How AWS IoT Greengrass works and What's new in AWS IoT Greengrass Version 2.

7 Greengrass Hardware Security Integration

The AWS IoT Greengrass Core software uses a private key and X.509 certificate to authenticate connections to the AWS IoT and AWS IoT Greengrass services. AWS IoT Greengrass Core software can use a hardware security module (HSM) such as the Zymbit SCM/SEN through the PKCS#11 interface to protect that private key. The private key used to generate the device certificate is kept on the Zymbit module and never exposed.

PKCS#11 support for the Zymbit products is available through the Zymbit package called *zkpkcs11*. The package is preinstalled with the rest of the Zymbit software. The *zkpkcs11* package is based on the SoftHSM2 source code. Zymbit added two important extra features:

1. Zymbit private keys can be used for signing by specifying --use-zkslot when creating a new key object with zk_pkcs11-util. This only applies to ECC NIST-P256 (secp256r1).
2. Even though SoftHSM2 does key wrapping to protect its key objects, Zymbit goes a step further and protects all key material in its private object store with its data lock/unlock feature, even for slots that Zymbit products do not support, such as RSA. For example, if you wanted to setup a zkpkcs11 slot that was RSA, you could do that as well and, even though all actions would be done by OpenSSL in software on the host computer rather than the Zymbit module, the Zymbit module would still use its lock/unlock feature to protect the generated RSA private key.

For this Getting Started example, the first method will be used to sign with the Zymbit module based ECC 256 keys. We've made available scripts to help bootstrap the process.

8 Greengrass prerequisites

Refer to the online documentation detailing the prerequisites needed for AWS IoT Greengrass. Follow the instructions in the following sections:

Step 1: Set up an AWS account
Step 2: Set up your environment

9 Install AWS IoT Greengrass

First, install and configure the AWS CLI on your IoT device. Then, follow the online guide to *Install with manual provisioning*. Refer to the instructions in the following steps:

- *Retrieve AWS IoT Endpoints*
- *Create an AWS IoT Thing*
- *Create the thing certificate* (Create the thing certificate from a private key in an HSM)
    - Download the scripts from *this repository* onto your IoT device to automate this step.
    - Make sure your user has permission to execute them.
    - If you want a more verbose output, you can uncomment the **set -x** command at the beginning of both scripts
    - Make sure that the slot, pin, and id in bootstrap-zymbit.sh are what you want them to be based on your setup of PKCS11 on the HSM.
    - Run bootstrap-zymbit.sh to generate the CSR. If done correctly, the script will automatically invoke bootstrap-common.sh and create the certificate using the HSM for AWS.
    - Here is the example output on success:

```
shiv@raspberrypi:~ $ bash bootstrap-zymbit.sh
Hit:1 https://packages.microsoft.com/repos/code stable InRelease
Hit:2 https://deb.nodesource.com/node_19.x bullseye InRelease
Hit:3 http://archive.raspberrypi.org/debian bullseye InRelease
Hit:4 http://raspbian.raspberrypi.org/raspbian bullseye InRelease
Hit:5 https://zk-sw-repo.s3.amazonaws.com/apt-repo-bullseye-aarch64 bullseye InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  libfuse2 raspinfo
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
jq is already the newest version (1.6-2.1).
opensc is already the newest version (0.21.0-1).
python3-pip is already the newest version (20.3.4-4+rpt1+deb11u1).
The following packages were automatically installed and are no longer required:
  libfuse2 raspinfo
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Requirement already satisfied: awscli in /usr/lib/python3/dist-packages (1.19.1)
usermod: user 'pi' does not exist
engine "zymkey_ssl" set.
certificate.arn already exists, checking if cert still exists
Certificate still exists in AWS IoT, nothing to do
If you are using GGP your HSI options will be:

  --hsi P11Provider=/usr/lib/libzk_pkcs11.so,slotLabel=greengrass,slotUserPin=1234,OpenSSLEngine=/usr/lib/libzk_pkcs11.so,pkcs11EngineForCurl=zymkey_ssl
859c4

SUCCESS: arn:aws:iot:us-east-1:746864551739:cert/bf81614afd6c9ebeb8e44d00139bc8373d9ca65992167ccb2f9160b2942859c4
```

```
shiv@raspberrypi:~ $ sudo zk_pkcs11-util --show-slots
Available slots:
Slot 395221339
    Slot info:
        Description:      SoftHSM slot ID 0x178e995b
        Manufacturer ID:  SoftHSM project
        Hardware version: 2.5
        Firmware version: 2.5
        Token present:    yes
    Token info:
        Manufacturer ID:  SoftHSM project
        Model:            SoftHSM v2
        Hardware version: 2.5
        Firmware version: 2.5
        Serial number:    204af3ce978e995b
        Initialized:      yes
        User PIN init.:   yes
        Label:            greengrass
Slot 1
    Slot info:
        Description:      SoftHSM slot ID 0x1
        Manufacturer ID:  SoftHSM project
        Hardware version: 2.5
        Firmware version: 2.5
        Token present:    yes
    Token info:
        Manufacturer ID:  SoftHSM project
        Model:            SoftHSM v2
        Hardware version: 2.5
        Firmware version: 2.5
        Serial number:
        Initialized:      no
        User PIN init.:   no
        Label:
```

- Continue the *Install with manual provisioning* guide at *Configure the thing certificate* until the end.

10 Create a Greengrass component

10.1 Create the component on your edge device

Below is an example component that you can use as the HelloWorld component for this section. This component will showcase some of Zymbit's API functions running as a Greengrass component in addition to AWS's basic HelloWorld example.

```
import zymkey
import sys

message = "Hello, %s!" % sys.argv[1]

# Print the message to stdout, which Greengrass saves in a log file.
print(message)

# Get timestamp from the Zymkey's RTC
time = zymkey.client.get_time()
print("GMT Time from Zymkey's RTC: ", time)

# Get the CPU's Temperature
temp = zymkey.client.get_cpu_temp()
print("CPU Temperature: ", temp)

# Get Model Number
model_num = zymkey.client.get_model_number()
print("Zymkey Model Number: ", model_num)

# Get Firmware Version
firmware_version = zymkey.client.get_firmware_version()
print("Zymkey Firmware Version: ", firmware_version)
```

Follow the instructions online under the section Develop and test a component on your device to create a simple component on your device.

10.2 Upload the Greengrass component

Follow the instructions online at Create your component in the AWS IoT Greengrass service to upload your component to the cloud, where it can be deployed to other devices as needed.

10.3 Deploy your component

Follow the instructions online at Deploy your component to deploy and verify that your component is running.

11 Troubleshooting

A troubleshooting section for the SCM/SEN can be found on our doc server:

General Troubleshooting
Troubleshooting

The ZYMBIT Community pages can also be helpful.

If you need additional support, please contact support@zymbit.com
For more information, refer to the online documentation *Troubleshooting Greengrass v2*.