TPM LDevID Enrollment with GlobalSign IoT Edge Enroll & Infineon OPTIGA™ TPM

Version 1.4

July 28, 2020

Table of Contents

1. Concept

One of the most difficult hurdles in an IoT ecosystem is handling identity, especially on first boot. In the case of certificate-based identities, problems arise when pre-provisioned credentials can be exported from the device to enable identity spoofing and loss of trust. Some of these concerns can be mitigated by offloading secret storage to a specialized device, in this case a Trusted Platform Module (TPM). The TPM allows for a system in which certain security assurances can be demonstrated cryptographically to a remote party, but the standard identity baked into the TPM device is alone insufficient for the task.

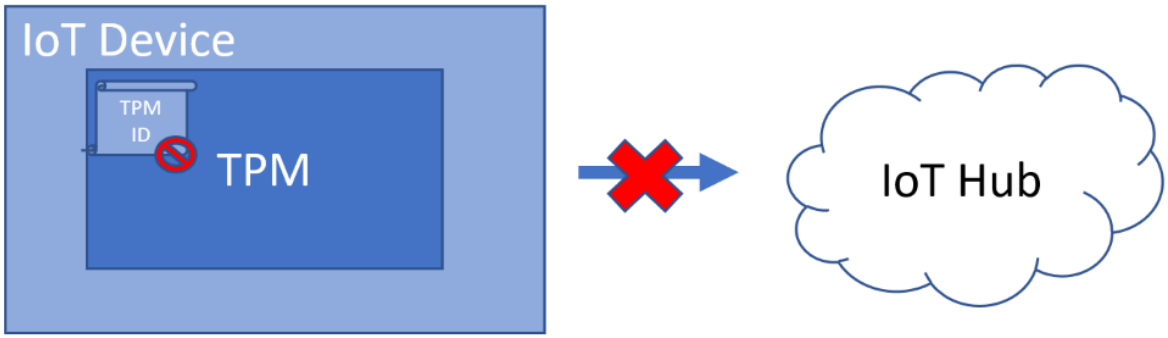

Figure 1. TPM EK Limitations.

Each TPM is assigned a unique and secret Endorsement Key (EK) at the time of manufacture. An accompanying EK certificate contains information attesting to the capabilities and genuineness of the TPM containing that key. However, usage of the EK is restricted by the TPM to decryption, and an EK cannot sign data to demonstrate proof-of-possession of the EK to an authentication system. Due to this privacy-inspired limitation, TPMs may enroll for a secondary certificate (an LDevID) that attests to the presence of a TPM and allows for more general use in authentication.

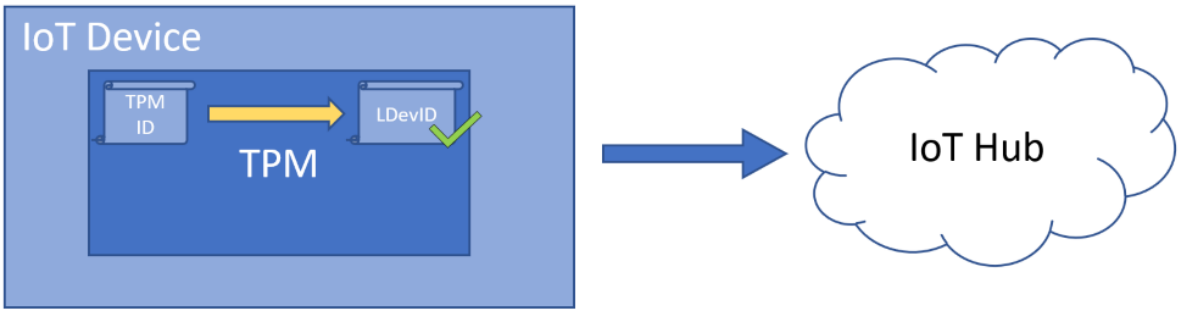

Figure 2. Moving from EK Cert to LDevID.

The major advantage of this system is that the resulting LDevID key is cryptographically proven (via the "Privacy CA") to be stored on a valid TPM so that an authenticating party is assured of the following:

1. The device presenting the LDevID has access to a genuine TPM from a manufacturer the Privacy CA trusts.

2. The LDevID key is stored on the TPM, improving key security considerably over typical filesystem storage.

2. Prerequisites

A Raspberry Pi 3+ and an Infineon Iridium TPM board will be used to demonstrate the LDevID enrollment functionality available as part of GlobalSign's IoT Edge Enroll service. To follow along, an OPTIGA™ SLM 967 (or feature equivalent) TPM will be needed on a Linux-based platform.

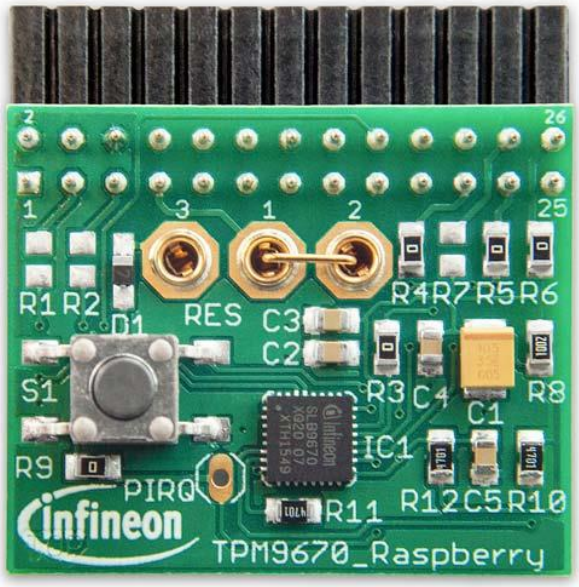

Figure 3. Infineon Iridium SLB 9670

The TPM should be available to the OS via /dev/tpmrm0. The user account should have RW access to the TPM without sudo, see appendix B for TPM setup.

The Go v1.13 (or later) runtime should be available with appropriate PATH set. See appendix C for setup.

3. Setup

1. Download and install any preferred TPM tool. These instructions will use the following one:

```
go get github.com/paulgriffiths/tpmtool
```

2. Download and install the GlobalSign EST Client:

```
go get github.com/globalsign/est/cmd/estclient
```

3. Retrieve the TPM EK certificate from NV Flash, this location may vary by TPM. In this case, the Infineon TPM RSA certificate is located at NV Index 0x1c00002

```
tpmtool nvread -handle 0x1c00002 > tpmcert.crt
```

4. Download the root and any intermediate CA certificates from the TPM manufacturer's website. Typically, the EK certificate will contain a URI pointing to the issuing CA cert within the Authority Information Access (AIA) field:

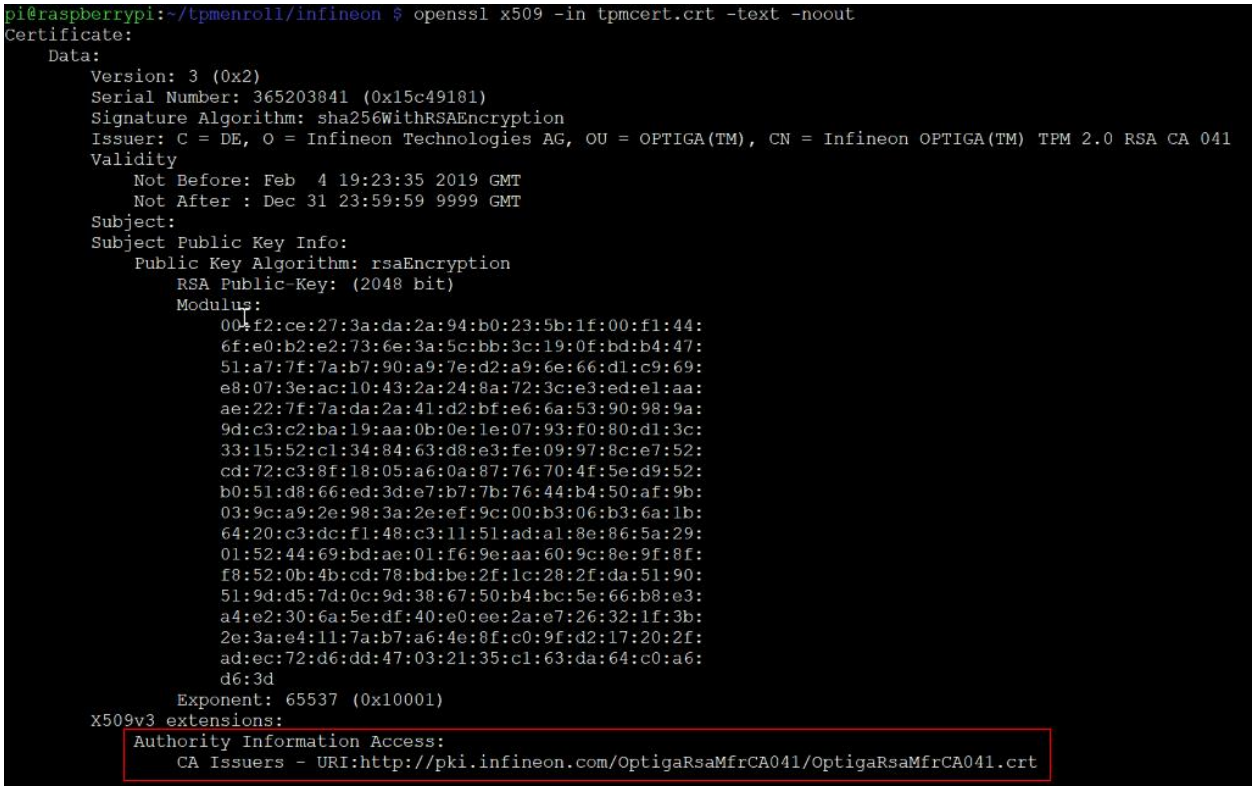

Figure 4. Finding the CA URIs.

- Note: If using an Infineon TPM with GlobalSign cross-signed EK (like the Limpet board), an alternative GlobalSign chain could be used in place of the Infineon chain

5. Combine the TPM certificate with the root and any intermediate CA certs into one file. Each certificate should be PEM encoded, usually requiring conversion from the DER encoding pulled from the TPM or from the downloaded CA certificates.

```
openssl x509 -in tpmcert.crt -inform der -out tpmcert.pem
cat tpmcert.pem inter.pem root.pem > combined.pem
```

The result should be a file starting with the EK cert and ending with the root cert.

4. Template and Key Creation

- *Note: RSA keys will be used, for ECC templates see appendix D*

1. Create the EK template. Due to the way TPMs are architected, the EK is generally not kept in persistent storage when leaving the factory but can be deterministically regenerated from the Endorsement Seed each time it is needed. The following template file will be used:

```
{
    "type": "TPM2_ALG_RSA",
    "name_alg": "TPM2_ALG_SHA256",
    "attributes": [
        "TPMA_OBJECT_RESTRICTED",
        "TPMA_OBJECT_ADMINWITHPOLICY",
        "TPMA_OBJECT_DECRYPT",
        "TPMA_OBJECT_FIXEDTPM",
        "TPMA_OBJECT_FIXEDPARENT",
        "TPMA_OBJECT_SENSITIVEDATAORIGIN"
    ],
    "auth_policy": "g3GXZ0SEs/gakMyNRqXXJP1S124GUgtk8qHaGzMUaao=",
    "rsa": {
        "symmetric": {
            "algorithm": "TPM2_ALG_AES",
            "key_bits": 128,
            "mode": "TPM2_ALG_CFB"
        },
        "scheme": {
            "algorithm": "TPM2_ALG_NULL"
        },
        "key_bits": 2048,
        "exponent": 0,
        "modulus": 0
    }
}
```

2. Generate the EK using the template file:

```
tpmtool createprimary -endorsement -template ek_template.json -persistent
0x81010001
```

3. Create a template to be used for the SRK:

```
{
    "type": "TPM2_ALG_RSA",
    "name_alg": "TPM2_ALG_SHA256",
    "attributes": [
        "TPMA_OBJECT_RESTRICTED",
        "TPMA_OBJECT_USERWITHAUTH",
        "TPMA_OBJECT_DECRYPT",
        "TPMA_OBJECT_FIXEDTPM",
        "TPMA_OBJECT_FIXEDPARENT",
        "TPMA_OBJECT_SENSITIVEDATAORIGIN"
    ],
    "rsa": {
```

```
        "symmetric": {
            "algorithm": "TPM2_ALG_AES",
            "key_bits": 128,
            "mode": "TPM2_ALG_CFB"
        },
        "key_bits": 2048,
        "exponent": 65537
    }
}
```

4. Like the EK, the Storage Root Key (SRK) is typically generated by the end user. This is the top key of the storage hierarchy and will be the parent to the LDevID private key:

```
tpmtool createprimary -template srk_template.json -persistent 2164260865
```

5. Create the LDevID private key template. This is the private key that will be linked to the certificate issued by IoT Edge Enroll:

```
{
    "type": "TPM2_ALG_RSA",
    "name_alg": "TPM2_ALG_SHA256",
    "attributes": [
        "TPMA_OBJECT_USERWITHAUTH",
        "TPMA_OBJECT_SIGN_ENCRYPT",
        "TPMA_OBJECT_FIXEDTPM",
        "TPMA_OBJECT_FIXEDPARENT",
        "TPMA_OBJECT_SENSITIVEDATAORIGIN"
    ],
    "rsa": {
        "key_bits": 2048,
        "exponent": 0
    }
}
```

6. Finally, generate the key to be certified:

```
tpmtool create -template rsa_template.json -persistent 2164391936 -parent
2164260865
```

7. With the keys created the EST client config can be set using the handles pointing to them. Create the EST config file, replacing the server value if needed:

```
{
    "server": "opentpm.est.edge.dev.globalsign.com:443",
    "private_key": {
        "tpm": {
            "device": "/dev/tpmrm0",
            "persistent_handle": 2164391936,
            "storage_handle": 2164260865,
            "ek_handle": 2164326401,
            "ek_certs": "combined.pem"
        }
    }
}
```

5. Enroll

1. Enroll for the certificate using the EST client, specifying the CN as desired:

```
estclient tpmenroll -config client.cfg -cn tpm.test.com
```

The requested certificate should be returned, linked to a verified TPM-backed private key. For details on how this process works, see Appendix A.

6. Cleanup

1. To restore the TPM to the pre-enrollment state after testing is complete, evict the generated keys:

```
tpmtool evict -handle 2164260865
tpmtool evict -handle 2164391936
tpmtool evict -handle 2164326401
```

Appendix A: Credential Enrollment Detail

The following flowchart describes the process happening behind the scenes during enrollment:

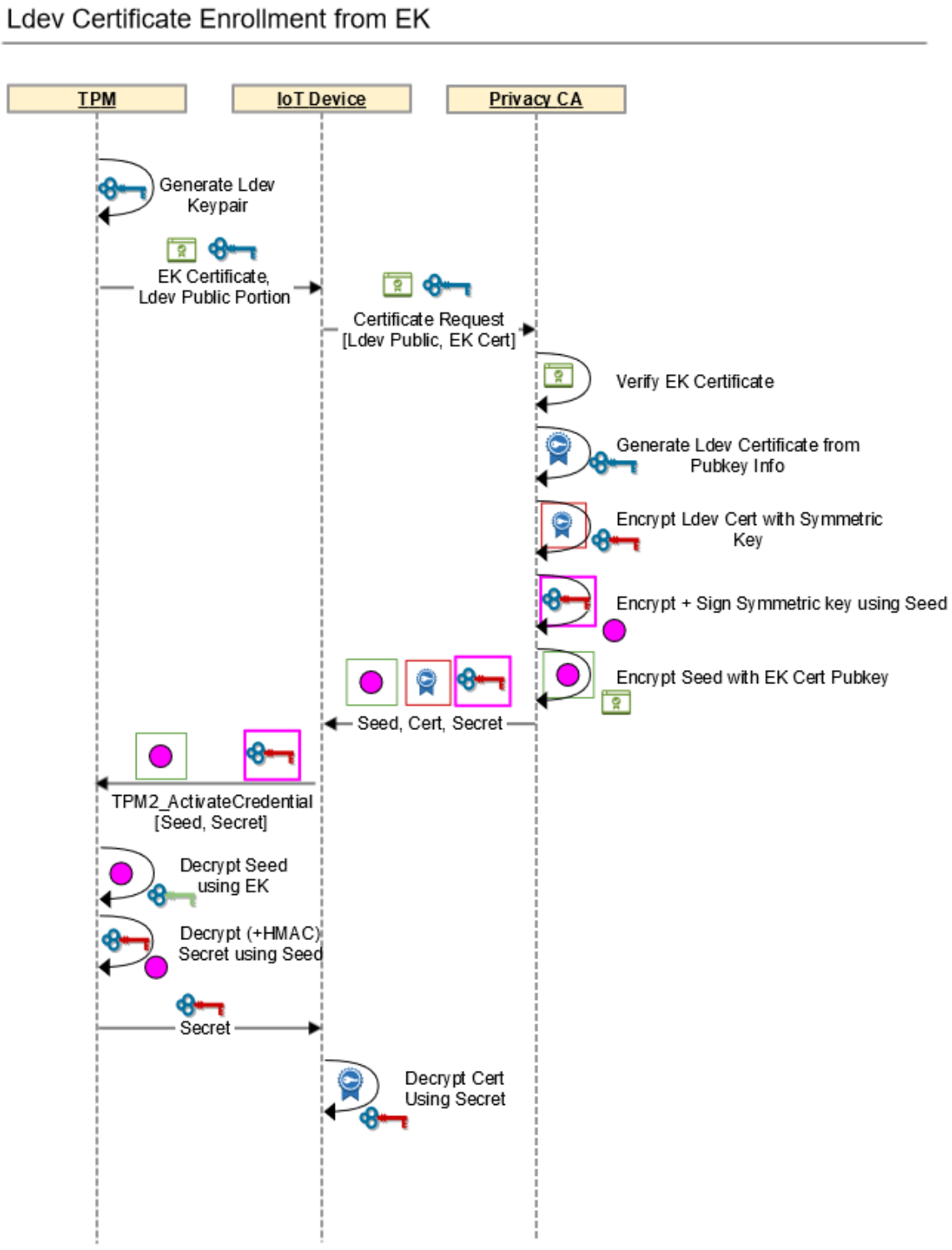

Figure 5. TPM Credential Enrollment and Activation Flowchart

Appendix B: TPM Setup

For a complete overview of Infineon's Optiga™ TPM product portfolio and available evaluation boards, visit https://www.infineon.com/tpm.

Infineon Limpet v1.0.2 Setup

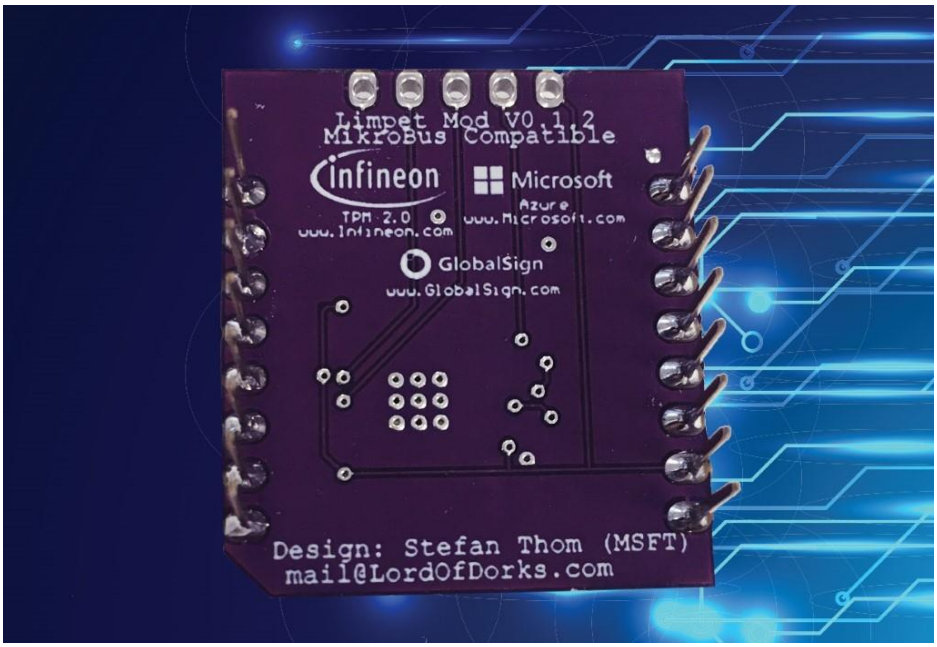

Figure 6. Infineon Limpet TPM board.

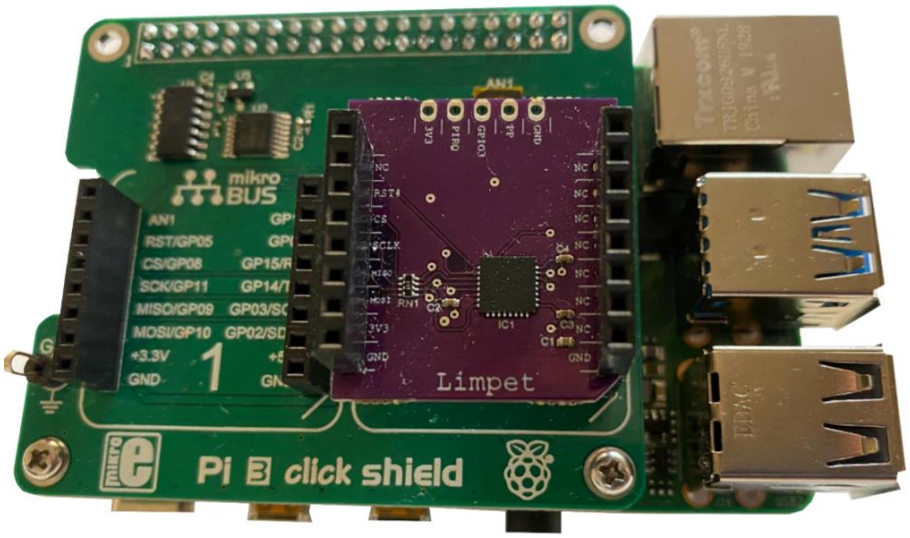

Figure 7. Raspberry Pi 3 with Infineon Limpet board attached.

1. Connect the Infineon Limpet board to the Pi 3 Click shield in slot 2. The Limpet will not work if inserted into slot 1.

2. Connect the Click shield to the Raspberry Pi, oriented to overlap the Pi board.

3. Power the Raspberry Pi.

4. Edit the /boot/config.txt file as root and add the following lines:

```
#SETING SLM9670 TPM2.0 RESET CONTROL IO AS OUTPUT AND HIGH.
gpio=12=op,dh
#TPM2.0
dtparam=spi=on
dtoverlay=tpm-slb9670
```

5. Reboot the Raspberry Pi for the changes to take effect. tpmrm0 and tpm0 devices should now show up in /dev/.

6. By default, TPM devices require root privileges for access. To perform the steps more easily in the solution guide, give TPM read-write access to the user:

```
sudo groupadd tpm
sudo chgrp tpm /dev/tpmrm0
sudo chmod g+rw /dev/tpmrm0
sudo usermod -a -G tpm [user]
```

Infineon Iridium SLB 9670 Setup

1. Connect the Iridium board directly to the Raspberry Pi GPIO headers (with power off), facing outwards and away from the USB ports, as in the below image.

Figure 8. Iridium board installed on Raspberry Pi 3

2. Power the Rapsberry Pi.
3. Edit the /boot/config.txt file as root and add the following lines:

```
#TPM2.0
dtparam=spi=on
dtoverlay=tpm-slb9670
```

4. Reboot the Raspberry Pi for the changes to take effect. tpmrm0 and tpm0 devices should now show up in /dev/.
5. By default, TPM devices require root privileges for access. To perform the steps more easily in the solution guide, give TPM read-write access to the user:

```
sudo groupadd tpm
sudo chgrp tpm /dev/tpmrm0
sudo chmod g+rw /dev/tpmrm0
sudo usermod -a -G tpm [user]
```

Appendix C: Go Installation

1. Remove older versions of Go if present.
2. Download the latest Go archive appropriate for your CPU architecture. In the case of the Raspberry Pi 3, the armv6 version will work.
3. Extract and install the package, adjusting for the downloaded filename:

```
sudo tar -C /usr/local -xzf go1.14.4.linux-armv6l.tar.gz
```

4. Add the following PATH appendages to your $HOME/.profile file to find the Go binaries correctly. Adjust these if the defaults are not desired:

```
PATH=$PATH:/usr/local/go/bin
PATH=$PATH:$HOME/go/bin
```

5. Apply the new settings:

```
source $HOME/.profile
```

Appendix D: ECC TPM Templates

ecc_ek_template.json

```
{
    "type": "TPM2_ALG_ECC",
    "name_alg": "TPM2_ALG_SHA256",
    "attributes": [
        "TPMA_OBJECT_RESTRICTED",
        "TPMA_OBJECT_ADMINWITHPOLICY",
        "TPMA_OBJECT_DECRYPT",
        "TPMA_OBJECT_FIXEDTPM",
        "TPMA_OBJECT_FIXEDPARENT",
        "TPMA_OBJECT_SENSITIVEDATAORIGIN"
    ],
    "auth_policy": "g3GXZ0SEs/gakMyNRqXXJP1S124GUgtk8qHaGzMUaao=",
    "ecc": {
        "symmetric": {
            "algorithm": "TPM2_ALG_AES",
```

```
            "key_bits": 128,
            "mode": "TPM2_ALG_CFB"
        },
        "scheme": {
            "algorithm": "TPM2_ALG_NULL"
        },
        "elliptic_curve": "TPM2_ECC_NIST_P256",
        "kdf": {
            "algorithm": "TPM2_ALG_NULL"
        },
        "point": {
            "x": 0,
            "y": 0
        }
    }
}
```

ecc_srk_template.json

```
{
    "type": "TPM2_ALG_ECC",
    "name_alg": "TPM2_ALG_SHA256",
    "attributes": [
        "TPMA_OBJECT_RESTRICTED",
        "TPMA_OBJECT_USERWITHAUTH",
        "TPMA_OBJECT_DECRYPT",
        "TPMA_OBJECT_FIXEDTPM",
        "TPMA_OBJECT_FIXEDPARENT",
        "TPMA_OBJECT_SENSITIVEDATAORIGIN"
    ],
    "ecc": {
        "symmetric": {
            "algorithm": "TPM2_ALG_AES",
            "key_bits": 128,
            "mode": "TPM2_ALG_CFB"
        },
        "elliptic_curve": "TPM2_ECC_NIST_P256"
    }
}
```

ecc_template.json

```
{
    "type": "TPM2_ALG_ECC",
    "name_alg": "TPM2_ALG_SHA256",
    "attributes": [
        "TPMA_OBJECT_USERWITHAUTH",
        "TPMA_OBJECT_SIGN_ENCRYPT",
        "TPMA_OBJECT_FIXEDTPM",
        "TPMA_OBJECT_FIXEDPARENT",
        "TPMA_OBJECT_SENSITIVEDATAORIGIN"
    ],
    "ecc": {
        "scheme": {
            "algorithm": "TPM2_ALG_ECDSA",
            "hash": "TPM2_ALG_SHA256"
        },
        "elliptic_curve": "TPM2_ECC_NIST_P256",
        "point": {
```

```
            "x":
810496551864121446836233171094674311593599191317682303678108289046227863132
64,
            "y":
477790702464745772661134096742046103785661982224220519751171952881169324078
48
        }
    }
}
```


About GlobalSign

GlobalSign is the leading provider of trusted identity and security solutions enabling business, large enterprises, cloud-based service providers and IoT innovators around the world to conduct secure online communications, manage millions of verified digital identities and automate authentication and encryption. Its high-scale PKI and identity solutions support the billions of services, devices, people and things comprising the Internet of Everything. The company has offices in the Americas, Europe, and Asia.

Learn about GlobalSign's IoT Edge Enroll enrollment services:
https://www.globalsign.com/en/internet-of-things/iot-identity-platform/iot-edge-enroll