OPTIGA™ TPM Application Note

Remote Attestation

26 Nov, 2020

Revision 1.0

www.infineon.com

About this document

Scope and purpose

This document explains how an OPTIGA™ TPM SLx 9670 TPM2.0 can be used on a Raspberry Pi® to perform TPM-based remote attestation.

Remote attestation is a mechanism to enable a remote system (server) to determine the integrity of a platform of another system (Raspberry Pi®). In a Linux-based system, a security feature known as the Integrity Measurement Architecture (IMA) can be used to capture platform measurements. Together with TPM a hardware-based security and its set of attestation features, it can be used to perform authentication and to protect the IMA measurement.

The OPTIGA™ TPM SLx 9670 TPM2.0 uses a SPI interface to communicate with the Raspberry Pi®. The OPTIGA™ TPM SLx 9670 TPM2.0 product family with SPI interface consists of 3 different products:

- OPTIGA™ TPM SLB 9670 TPM2.0 standard security applications
- OPTIGA™ TPM SLI 9670 TPM2.0 automotive security applications
- OPTIGA™ TPM SLM 9670 TPM2.0 industrial security applications

OPTIGA™ TPM SLx 9670 TPM2.0 products are fully TCG compliant TPM products with CC (EAL4+) and FIPS certification. The OPTIGA™ TPM SLx 9670 TPM2.0 products standard, automotive, and industrial differ with regards to supported temperature range, lifetime, quality grades, test environment, qualification, and reliability to fit the target applications requirements. An overview of all Infineon OPTIGA™ TPM products can be found on Infineon's website [2][3]. More information on TPM specification can be found on Trusted Computing Group (TCG) in reference [4].

Intended audience

This document is intended for customers who want to increase the security level of their embedded platforms using a TPM 2.0 and like to evaluate the implementation of TPM-based remote attestation for their target applications.

Table of Contents

1 Prepare Raspberry Pi®

This section describes all necessary steps needed to build a Raspberry Pi® bootable SD card image.

1.1 Prerequisites

- Raspberry Pi®
    - (Recommended) Raspberry Pi® 4 with 4GB RAM or above
    - Raspberry Pi® 3 Model B V1.2
- Micro SD card (≥8GB) flashed with Raspberry Pi® OS. Download the official image (raspbian-2020-02-14) from [1]
- SD card reader
- Host machine running Ubuntu 18.04 LTS
- OPTIGA™ TPM (TPM2.0)
    - SLB 9670
    - SLI 9670
    - SLM 9670

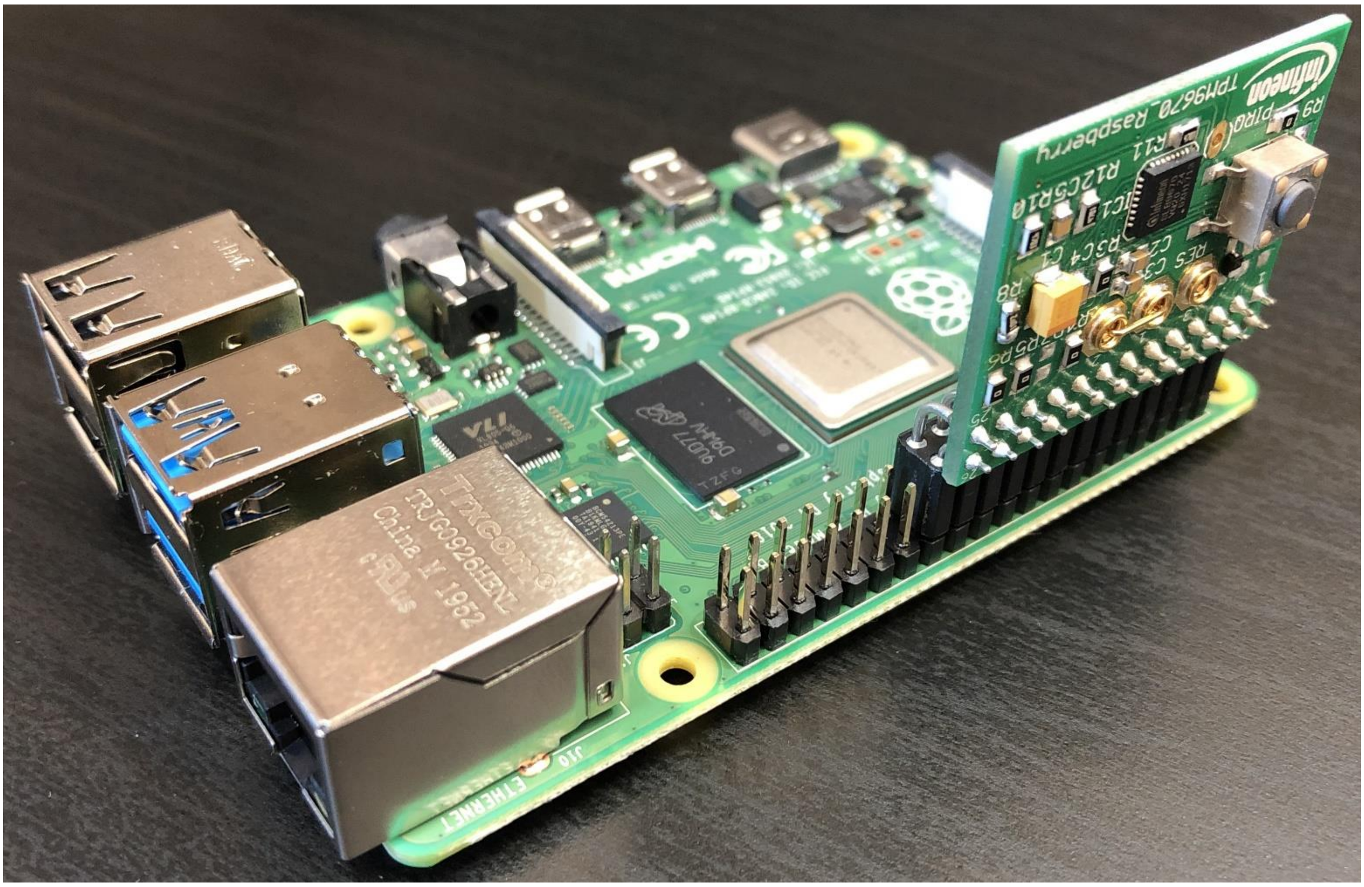

Figure 1: Infineon Iridium SLx 9670 TPM2.0 SPI Board [2] on Raspberry Pi® 4

1.2 Kernel Build Guide

This guide is for cross-compilation only. Optionally, native-compilation guide can be found at [5].

Install required dependencies on host machine:

```
$ sudo apt install git bc bison flex libssl-dev make libc6-dev libncurses5-dev
libncurses5-dev
```

Install toolchain and set environment variable:

```
$ git clone https://github.com/raspberrypi/tools ~/tools
$ export PATH=$PATH:~/tools/arm-bcm2708/arm-linux-gnueabihf/bin
```

Download Linux kernel source (Approx. 3.5GB):

```
$ git clone -b rpi-4.19.y https://github.com/raspberrypi/linux
$ cd linux
```

At the time of testing:

```
$ git checkout 06606627043f72d22881563d485268fec2acd56d
```

Build for Raspberry Pi® 3:

```
# Prepare
$ KERNEL=kernel7
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- bcm2709_defconfig

# Configure (optional)
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig

# Build
$ make -j$(nproc) ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- zImage modules dtbs
```

Build for Raspberry Pi® 4:

```
# Prepare
$ KERNEL=kernel7l
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- bcm2711_defconfig

# Configure (optional)
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig

# Build
$ make -j$(nproc) ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- zImage modules dtbs
```

Transfer kernel modules, kernel image, and device tree blobs to a SD card (remember to set */dev/sdbX* and */dev/sdbY* accordingly):

```
$ mkdir mnt
$ mkdir mnt/fat32
```

```
$ mkdir mnt/ext4
$ sudo umount /dev/sdbX
$ sudo umount /dev/sdbY
$ sudo mount /dev/sdbX mnt/fat32
$ sudo mount /dev/sdbY mnt/ext4
$ sudo env PATH=$PATH make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf-
INSTALL_MOD_PATH=mnt/ext4 modules_install
$ sudo cp mnt/fat32/$KERNEL.img mnt/fat32/$KERNEL-backup.img
$ sudo cp arch/arm/boot/zImage mnt/fat32/$KERNEL.img
$ sudo cp arch/arm/boot/dts/*.dtb mnt/fat32/
$ sudo cp arch/arm/boot/dts/overlays/*.dtb* mnt/fat32/overlays/
$ sudo cp arch/arm/boot/dts/overlays/README mnt/fat32/overlays/
$ sudo umount mnt/fat32
$ sudo umount mnt/ext4
```

1.3 Kernel Modification

Configure Linux source to opt-in Integrity Measurement Architecture (IMA) security module:

```
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig
```

Enable IMA module:

```
Security options  --->
[*] Enable different security models
[*] Integrity subsystem
[*]   Integrity Measurement Architecture(IMA)
        Default template (ima-sig)  --->
        Default integrity hash algorithm (SHA256)  --->
[*]     Enable multiple writes to the IMA policy
[*]     Enable reading back the current IMA policy
```

Make following changes to enable early initialization of SPI and TPM before IMA activation.

Set TPM as **built-in** module:

```
Device Drivers  --->
Character devices  --->
-*- TPM Hardware Support  --->
<*>   TPM Interface Specification 1.3 Interface / TPM 2.0 FIFO Interface - (SPI)
```

Set SPI as **built-in** module:

```
Device Drivers  --->
[*] SPI support  --->
      <*>   BCM2835 SPI controller
```

To enable early loading of SPI subsystem, remove the following line from *drivers/clk/bcm/clk-bcm2835.c*:

~~postcore_initcall(__bcm2835_clk_driver_init);~~

Replace it with:

```
subsys_initcall(__bcm2835_clk_driver_init);
```

Modify line 122 of file *security/integrity/ima/ima_policy.c*. The modification simplifies the policy for ease of understanding. The new policy restricts IMA measurement to files that are executed by root. Now re-build kernel image following section 1.2.

```
static struct ima_rule_entry default_measurement_rules[] __ro_after_init = {
      {.action = MEASURE, .func = FILE_CHECK, .mask = MAY_EXEC,
       .uid = GLOBAL_ROOT_UID, .uid_op = &uid_eq,
       .flags = IMA_FUNC | IMA_INMASK | IMA_UID},
};
```

2 Software Setup

This section describes how to install and enable all necessary software on a Raspberry Pi®.

2.1	Enable TPM and IMA.
2.2	Install TPM software stack.
2.3	Download and build server source.
2.4	Download and build device source.

Table 1: Software setup

2.1 Enable TPM & IMA

Insert the flashed SD card and boot the Raspberry Pi®.

Config.txt

Open the file config.txt in an editor:

```
$ sudo nano /boot/config.txt
```

Insert the following lines to enable SPI and TPM:

```
dtparam=spi=on
dtoverlay=tpm-slb9670
```

Save the file and exit the editor.

Cmdline.txt

Open the file cmdline.txt in an editor:

```
$ sudo nano /boot/cmdline.txt
```

Append the following to the existing line:

```
ima_policy=tcb
```

Save the file then exit the editor.

Reboot the Raspberry Pi® for both changes to take effect:

```
$ reboot
```

Check if TPM is activated by:

```
$ ls /dev | grep tpm
tpm0
tpmrm0
```

Check if IMA is activated. The return value must be greater than 1.

```
$ sudo cat /sys/kernel/security/ima/runtime_measurements_count
146
```

Check if IMA policy is configured correctly.

```
$ sudo cat /sys/kernel/security/ima/policy
dont_measure fsmagic=0x9fa0
dont_measure fsmagic=0x62656572
dont_measure fsmagic=0x64626720
dont_measure fsmagic=0x1021994
dont_measure fsmagic=0x1cd1
```

```
dont_measure fsmagic=0x42494e4d
dont_measure fsmagic=0x73636673
dont_measure fsmagic=0xf97cff8c
dont_measure fsmagic=0x43415d53
dont_measure fsmagic=0x27e0eb
dont_measure fsmagic=0x63677270
dont_measure fsmagic=0x6e736673
measure func=FILE_CHECK mask=^MAY_EXEC uid=0
```

Lastly, check if IMA template (ima-sig) and algorithm (SHA256) is set correctly by inspecting the file ascii_runtime_measurements.

```
$ sudo cat /sys/kernel/security/ima/ascii_runtime_measurements
10 <20 bytes of hash value> ima-sig sha1:<20 bytes of hash value> boot_aggregate
10 <20 bytes of hash value> ima-sig sha256:<32 bytes of hash value> <filename with path>
…
```

2.2 Install TPM Software

Boot the Raspberry Pi® and install the following software:

Software	Link	Version
tpm2-tss	https://github.com/tpm2-software/tpm2-tss	2.4.0
tpm2-tools	https://github.com/tpm2-software/tpm2-tools	4.2

Table 2: TPM 2.0 software

Install dependencies:

```
$ sudo apt update
$ sudo apt -y install autoconf-archive libcmocka0 libcmocka-dev procps iproute2
build-essential git pkg-config gcc libtool automake libssl-dev uthash-dev autoconf
doxygen libgcrypt-dev libjson-c-dev libcurl4-gnutls-dev uuid-dev pandoc
```

Download, build, and install TPM software stack:

```
$ git clone https://github.com/tpm2-software/tpm2-tss.git
$ cd tpm2-tss
$ git checkout 2.4.0
$ ./bootstrap
$ ./configure
$ make -j$(nproc)
$ sudo make install
$ sudo ldconfig
```

Download, build, and install TPM tools:

```
$ git clone https://github.com/tpm2-software/tpm2-tools.git
$ cd tpm2-tools
$ git checkout 4.2
$ ./bootstrap
$ ./configure
$ make -j$(nproc)
$ sudo make install
$ sudo ldconfig
```

2.3 Install Server Software

Information on software licenses used at frontend & backend of server:

Software	Link	License
Spring Framework	https://spring.io/	Apache License 2.0
Material Design for Bootstrap (Free version)	https://github.com/mdbootstrap/bootstrap-material-design	MIT License
TPM Software Stack from Microsoft Research	https://github.com/microsoft/TSS.MSR	MIT License
OpenJDK	https://openjdk.java.net/	OpenJDK Community TCK License Agreement
SockJS-client	https://github.com/sockjs/sockjs-client	MIT License
STOMP.js	https://github.com/stomp-js/stompjs	MIT License

Table 3: Server software licensing information

Install dependencies:

```
$ sudo apt install maven openjdk-9-jre
```

Download and build server source:

```
$ git clone https://github.com/infineon/remote-attestation-optiga-tpm -b server
$ cd remote-attestation-optiga-tpm
$ mvn install
```

2.4 Install Device Software

The device software is composed of application to communicate with server, and step-by-step scripts to perform remote attestation.

Install dependencies:

```
$ sudo apt update
$ sudo apt install libconfig-dev libjson-c-dev libcurl4-gnutls-dev
```

Download and build device software:

```
$ git clone https://github.com/infineon/remote-attestation-optiga-tpm -b device
$ cd remote-attestation-optiga-tpm
$ make
```

3 Operation Guide

This section describes all necessary steps to perform remote attestation in the following sequence.

3.1	Run Server.
3.2	Provision TPM.
3.3	Run Device Scripts.

Table 4: Operation guide

3.1 Run Server

For better user experience and quicker response time, it is possible to host the server on a separate machine or remote server. The guide for server hosting is not covered in this document.

Run server on Raspberry Pi®:

```
$ cd remote-attestation-optiga-tpm/server/target
$ sudo java -jar server-0.0.1-SNAPSHOT.jar
```

The server is ready for operation once you see the following message:

```
...
2020-06-10 22:37:51.856  INFO 12828 --- [           main]
o.s.m.s.b.SimpleBrokerMessageHandler     : Started.
2020-06-10 22:37:52.414  INFO 12828 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 443 (https)
80 (http) with context path ''
2020-06-10 22:37:52.418  INFO 12828 --- [           main]
com.ifx.server.ServerApplication         : Started ServerApplication in 91.269
seconds (JVM running for 98.966)
```

View the webpage (https://localhost) using Raspberry Pi® OS built-in web browser. A warning message may appear, it is expected since server is using a self-signed certificate. Bypass the warning and proceed as usual. Slower loading time is expected on Raspberry Pi® 3.

On the upper menu bar, click on "Start" to enter the sign in page (*Figure 2: Sign in*). Sign in using the following credential to enter a self-explanatory dashboard page.

Username	infineon
Password	password

Table 5: User account

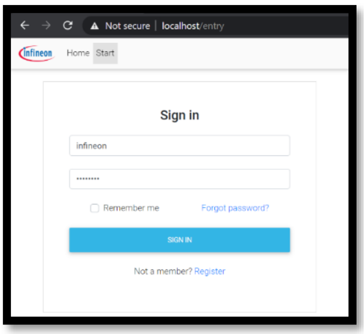

Figure 2: Sign in

3.2 Provision TPM

Following steps are executed on Raspberry Pi®.

Perform a TPM clear on the platform hierarchy:

```
$ sudo chmod a+rw /dev/tpm0
$ sudo chmod a+rw /dev/tpmrm0
$ tpm2_clear -c p
```

Generate TCG profile compliant endorsement key (EK) and store it as persistent key:

```
$ tpm2_createek -G rsa -u ek.pub -c ek.ctx
$ tpm2_evictcontrol -C o -c ek.ctx 0x81010001
```

Generate attestation key (AK) and store it as persistent key:

```
$ tpm2_createak -C 0x81010001 -c ak.ctx -G rsa -g sha256 -s rsassa -u ak.pub -n
ak.name
$ tpm2_evictcontrol -C o -c ak.ctx 0x81000002
```

Verify generated keys by reading TPM persistent handles:

```
$ tpm2_getcap handles-persistent
-     0x81000002
-     0x81010001
```

3.3 Run Device Scripts

Navigate to directory:

```
$ cd remote-attestation-optiga-tpm/
```

The *config.cfg* is a configuration file. View the file for more information.

Navigate to directory:

```
$ cd remote-attestation-optiga-tpm/bin
```

Execute following scripts sequentially.

0_prep.sh	Authorize non-privileged access to the TPM device node.
1_cleanup.sh	Erase non-essential files and restore *config.cfg*.
2_pcr.sh	Read TPM PCRs and the IMA log.
3_attune.sh	Register a good platform measurement to a server.
4_atelic.sh	Ask for a server encrypted challenge.
5_credential.sh	Decrypt the challenge using the tool *tpm2_activatecredential*.
6_quote.sh	Generate a quote and a signature using the tool *tpm2_quote*. Skip step **6_quote-bad.sh** if this script is executed.
6_quote-bad.sh	This is to trigger a failure using an invalid challenge. Skip **6_quote.sh** if this script is executed.
7_attest.sh	Send the quote, signature, and the latest IMA log to a server to perform attestation.

Table 6: Device scripts

4 Architecture

This section describes the architecture of the system.

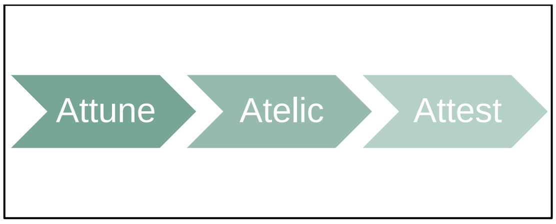

Figure 3: Operation flow

4.1 Attune

Attune is a process to register the following parameters to a server. These parameters are considered as a good reference and it will be used for verification purpose at the attestation stage.

sha1pcrs	PCR bank (SHA1) register indexes, e.g. [9,10].
sha2pcrs	PCR bank (SHA256) register indexes, e.g. [9,10].

ekCrt	EK certificate issued by TPM manufacturer:
	`$ tpm2_nvread 0x1c00002 -s 1184 --offset 0 -o ek.crt`
	Inspect the certificate:
	`$ openssl x509 -inform der -in ek.crt -text -noout`

akPub	AK public key: `$ tpm2_readpublic -c 0x81000002 -o ak.pub`
pcrs	TPM PCR values: `$ tpm2_pcrread -o pcr` PCRs that are not indicated by sha1pcrs/sha2pcrs are filtered.
imaTemplate	A log that records IMA measured files: /sys/kernel/security/ima/binary_runtime_measurements A human readable version at: /sys/kernel/security/ima/ascii_runtime_measurements

Table 7: Attune parameters

Figure 4: ascii_runtime_measurements is a sample log file generated by the IMA subsystem at runtime. It contains a list of measured files. Hash value of each file is extended to the TPM PCR-10 (IMA uses PCR index 10 by default) sequentially. Since Linux is not deterministic in the sequence it starts programs, the log changes after each power cycle, and this will result in a different PCR-10 value. Therefore, for server to correctly verify a PCR-10 value, the reference value must be computed from the *attune.imaTemplate* log based on the boot sequence.

```
pi@raspberrypi:~ $ sudo cat /sys/kernel/security/ima/ascii_runtime_measurements
10 d4db23d50c0e51c7eeb879ca78d6a650acb11648 ima-sig sha1:9797edf8d0eed36b1cf92547816051c8af4e45ee boot_aggregate
10 7ec8ebfc72f0e58ed0dfe426e9cc76d7724269a5 ima-sig sha256:f48531336b31a520d6ae1220f5227d4212853c15fef0a34114f2955f3911b9a2 /lib/systemd/systemd
10 c55ee62c6a9725a4b855534b45b22e238f68fad5 ima-sig sha256:b426895210b64173047feecd0c057230b39ba3dadac01bf7df57e8af5df9af19 /lib/arm-linux-gnueabihf/ld-2.28.so
10 574abd340a096327a36244ad4b61d890eda9fc40 ima-sig sha256:72010339b21d589433ed10f2cbec1aaee1227549abd29ce216e176192e439892 /lib/systemd/system-generators/systemd-bless-boot-generator
10 14a7694ec83a73ff823c43670536de7eed432323 ima-sig sha256:929bffa6603c921353eeb34ef18286c7780f84bbe2a467c5f1a4575d561f63d7 /lib/systemd/system-generators/systemd-debug-generator
10 1698d691ceb80b6c003665ed941738061e72efb9 ima-sig sha256:9a729c9edbe58b9138dc2105171e4db0be446b1255f04030f613610a35c711e2 /lib/systemd/system-generators/systemd-cryptsetup-generator
10 0586c321bed1a211498144 63ea4391495578b8e4 ima-sig sha256:34156752846acfe5f8fe038c991dea2411882c979fd191d30b2d61f8ded12c50 /lib/systemd/system-generators/systemd-fstab-generator
10 f48d3f3649c17373ca0a26ba0530915f4b13a355 ima-sig sha256:c0738fed6adceac60adcbb4dd7c8abc8a2264819d05b9af9bd3176d4a355a637 /lib/systemd/system-generators/systemd-getty-generator
10 e5a957ad5e07054fcd46843753bfaf43e1018a0c ima-sig sha256:ae7b72e84d11863fdfcbcb518ae07ce155f45322742e3465ed920f82998fe9d2 /lib/systemd/system-generators/systemd-gpt-auto-generator
10 ece01e99829be32ec87623d061670c92a99993e1 ima-sig sha256:433ac3acc1b8ff5ba5d1912fae3d08bb032fa397eafc5dacfbe3d5387279f8fc /lib/systemd/system-generators/systemd-hibernate-resume-generator
10 3ba4e0c2ffbeffd59c95eb72e7cad303253fb516 ima-sig sha256:df6a83dfacd2ec3e94370519452e71df06bec110efc4b97a88866dccc9572f4f /lib/systemd/system-generators/systemd-rc-local-generator
10 a58c6b6ef33bca5e107acf1dbbc7247bd9668abe ima-sig sha256:b5682ddf0642b51c8d5503bcaf662ac41033b9f8ee6067dbbdba103c188d05d8 /lib/systemd/system-generators/systemd-sysv-generator
10 064c900c578616f8793ce42f520cb48f732db719 ima-sig sha256:cc2865d346f251320c580b42a03a4c3eff5fbb3301be7f231558e9560d8e70b7 /lib/systemd/system-generators/systemd-veritysetup-generator
10 91548393708d96dfa7cc19e76c599e92a9f07485 ima-sig sha256:096cd803fac9d81aa72543ce00dd76555f1b56f3d4bf33b1f2161b5508824a2d /lib/systemd/system-generators/systemd-system-update-generator
10 587293a956f2acd3f1600e6a0433cf0314c9aae7 ima-sig sha256:62298a7da654b94023476ca2180ea268fda838e56e48ed6e6287608e7c64d8d0 /lib/systemd/system-generators/systemd-run-generator
10 08e79bbf98c27af8aa5c703e51331cb069442796 ima-sig sha256:ff4df159ae57f91c735f5c52d57bf59534556c19555e05417e40ad5f87a84035 /bin/kmod
10 bcc7167496d384e280dec06e60bf1c0814286faa ima-sig sha256:f40fcf2bd456d4fa704c3c540092799d0e50aa07fe1f2abd3bfc34ef1aa27355 /bin/mount
10 c14087bac819300520777bbb031139e62cc19699 ima-sig sha256:b7539bca0837fa5e1082384565119490d9a0b245fca148d6597bf45e8abfe551 /lib/systemd/systemd-modules-load
10 f1316f0c2c5bb8ff473c330f63cb6c539f335796 ima-sig sha256:ad11fd33353586c445c50f2f33c4401d6e32944636feb932d619e61bf82f9c6c /sbin/fake-hwclock
10 4567c7b3e8a59f55be0c761eb58b297dfd123a1c ima-sig sha256:f79265d2ad5b43ed44bbd6efa2b002f007cc97b369e45238b5385f35cdc00587 /lib/console-setup/keyboard-setup.sh
10 834af596f5816b4ca15d38b1e7b1c862de53bdad ima-sig sha256:645e88285fe1cd9c2e53c8c1a83dc322e1c4b1e6c2c5f8ad802192a55ce271c7 /bin/dash
10 523028663be5f0513aae3988d695a933d7ab7254 ima-sig sha256:874bf484f844ad1003255481c36b04bfba66051ed021b4c859c79beb3c232ea2 /etc/console-setup/cached_setup_keyboard.sh
10 f19b6be6e04dc6e4a5bf6b56ddfb99dadcc48e14 ima-sig sha256:c8fb15e6f0f91a06af084743ecda3e28ef664c2d38f818d9ab9caf0265dcefeb /bin/cat
10 9b73e4b7bf79bb20f4eb01e2529a8e85fe3f9aea ima-sig sha256:45182407854e8fab92d3260e192454fd8accbd4b4984b73e37a4687a59e47b50 /bin/kbd_mode
10 4bcaba5a8fc8a6bd80cb4cbee1c40564269de5a0 ima-sig sha256:181410a5546a78439b11dd9c30c33bf4024018132b9ad225fc8a7f2fdcde2972 /bin/udevadm
10 69801ab7bac5b302b2e3c8879dc966122b785521 ima-sig sha256:ab77c0996c002d302889a8fc87d4d463ef0272608d2bdbd294fdb09edb0e7a5e /bin/date
10 2178176175b01dd78356e452af87570125b4ceb3 ima-sig sha256:28dd0115f051ad8a89602062fa2bdefe49ab29c0c259fd4e970d8b8c865c37e5 /lib/systemd/systemd-journald
10 5b10c5758af969e3389ff0a9a168615e05f35c7d ima-sig sha256:35e108d5d3d50d02b873f5c20bd2d3b9b6fa08172a6fe29321f26ca3ab80e39d /lib/systemd/systemd-sysctl
```

Figure 4: ascii_runtime_measurements

4.2 Atelic

Atelic is a process to request for a challenge from a server. In response to atelic, the server generates a challenge then encrypts it using the TPM credential feature (*tpm2_makecredential*).

Parameters consumed by *tpm2_makecredential*:

EK public key	EK public key.
AK name	A name derived from AK public key blob.
Challenge	A random string.

Table 8: Make credential parameters

The requester can decrypt the credential blob and recover the challenge by performing the following:

```
$ tpm2_startauthsession --policy-session -S session.ctx
$ tpm2_policysecret -S session.ctx -c 0x4000000B
$ tpm2_activatecredential -c 0x81000002 -C 0x81010001 -i credential.blob -o
qualification -P"session:session.ctx"
$ tpm2_flushcontext session.ctx
$ rm session.ctx
```

A challenge is also known as a qualification value. This value will be used in the next section.

4.3 Attest

Attest is a process to request a server to perform remote attestation. The following parameters are attached to the request:

quote, sig	Quote and its signature can be produced by: `$ tpm2_quote -c 0x81000002 -q qualification -l sha1:9,10+sha256:9,10 -m quote -s sig`
imaTemplate	The latest IMA log, similar to *attune.imaTemplate.*

Table 9: Attest parameters

Attestation on a server is done by validating the content of a quote and its signature:

<table>
<tr><td rowspan="3">Quote</td><td>Detailed breakdown of a quote:</td></tr>
<tr><td><table>
<tr><td>PCR bank (SHA1) register indexes</td><td>Same value as attune.sha1pcrs.</td></tr>
<tr><td>PCR bank (SHA256) register indexes</td><td>Same value as attune.sha2pcrs.</td></tr>
<tr><td>PCRs digest</td><td>Same value as computed digest, see below.</td></tr>
<tr><td>Qualification</td><td>Same value as server challenge.</td></tr>
<tr><td>AK name</td><td>Not implemented.</td></tr>
<tr><td>Firmware version</td><td>Not implemented.</td></tr>
<tr><td>TPM clock</td><td>Not implemented.</td></tr>
</table></td></tr>
<tr><td>Computation of PCRs digest:

Set attest.imaTemplate as a sorting reference. Rearrange the order of attune.imaTemplate to match with the reference. Hash the reordered attune.imaTemplate to compute a new PCR-10 value.
Replace the PCR-10 value in attune.pcrs with the new value, hash the attune.pcrs to obtain the final digest.
The quote’s PCRs digest must be equal to the computed digest to pass the verification.
</td></tr>
<tr><td>Signature</td><td>Verify quote’s signature using AK public key.</td></tr>
</table>

Table 10: Attestation

References

[1] https://downloads.raspberrypi.org/raspbian/images/raspbian-2020-02-14/

[2] https://www.infineon.com/cms/en/product/evaluation-boards/iridium9670-tpm2.0-linux/

[3] http://www.infineon.com/tpm

[4] https://trustedcomputinggroup.org/resource/tpm-main-specification/

[5] https://www.raspberrypi.org/documentation/linux/kernel/building.md

[6] https://github.com/tpm2-software

[7] https://www.raspberrypi.org/

[8] https://www.raspberrypi.org/downloads/raspberry-pi-os/

[9] https://github.com/raspberrypi

[10] https://www.raspberrypi.org/documentation/installation/installing-images/README.md

[11] https://spring.io/

[12] https://github.com/microsoft/TSS.MSR

Revision history

Page or Reference	Description of change
Revision 1.0, 2020-11-26	
	Initial Release

Infineon Technologies AG

81726 Munich
Germany

Published by
Infineon Technologies AG

www.infineon.com