5/31/2021

TCG RIM Tool Users Guide v1.0

HOST INTEGRITY
AT RUNTIME
& STARTUP

Laboratory for Advanced Cybersecurity Research

Table of Contents

Table of Contents

1 Introduction

1.1 Purpose

The purpose of this document is to support and define a command line application called the tcg_rim_tool. The tcg_rim_tool can be used to create NISTIR 8060 compatible SWID tags that adhere to the TCG PC Client RIM specification. It supports the ability to digitally sign a Base RIM file that then may be uploaded to the HIRS ACA if a valid signature is evident.

1.2 Background

1.2.1 Reference Integrity Manifests

The Reference Integrity Information Model[1] (RIM) defines structures that a Verifier (i.e. a system that analyzes evidence from a platform or platform component to determine it's state) uses to validate expected values (Assertions) against actual values (Evidence).

The PC Client RIM is an OEM produced artifact that can be used by the ACA when the Firmware Validation Policy option is enabled. Firmware Validation compliments the Platform Certificate for Supply Chain acceptance testing by providing an automated means to verify the firmware and boot software for the platform before an Attestation Certificate will be issued.

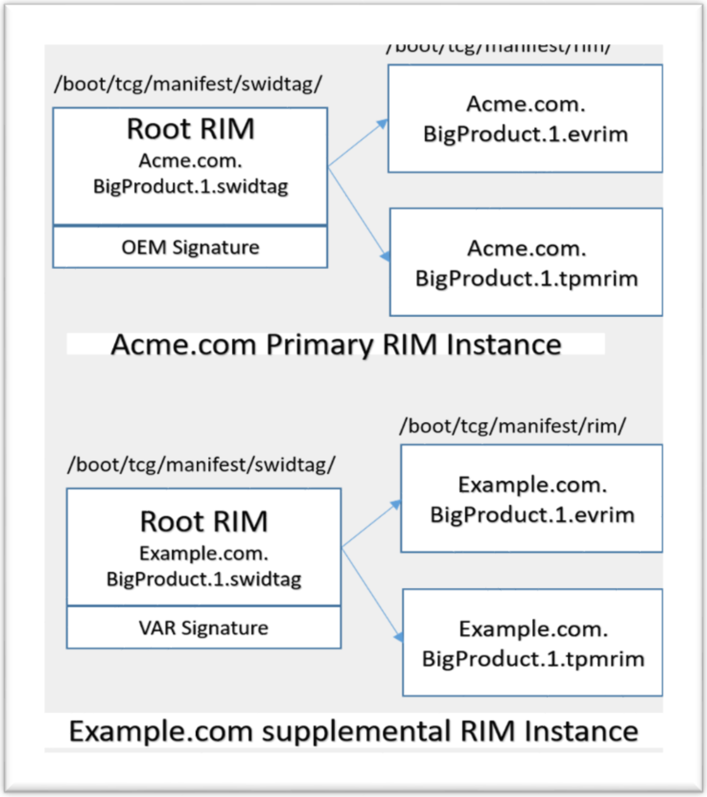
For the PC Client,[2] there are two different types of RIM files: the Base RIM and the Support RIMs. This is designated by the TCG as the "RIM Bundle".

[1] https://trustedcomputinggroup.org/wp-content/uploads/TCG_RIM_Model_v1p01_r0p16_pub.pdf
[2] https://trustedcomputinggroup.org/wp-content/uploads/TCG_PC_Client_RIM_r0p15_15june2020.pdf

The PC Client RIM defines the Base Rim as a ISO 197770-2 Software Identity (SWID) standard compatible file. The Base RIM provides a verifiable identity of the RIM creator and also integrity information of Support RIMs. The Base RIM contains:

1. Cryptographically verifiable identification of the Creator of the RIM and Support RIMs.
2. A unique identifier (tagId) for a set of RIM Bundles.
3. A reference to the binding specification that defines the Support RIMs.
4. Cryptographic hashes (digests) of all Payload references including Support RIMs.
5. A digital signature of the RIM signed by the RIM's Creator.

For PC Clients, the Support RIM utilizes the TPM Log created during the boot process. The TPM log defined by the TCG PC Client Platform Firmware Profile captures[3] all events that extend any of the TPMs Platform Configuration Register (PCR) contents. The OEM that creates the RIM captures the event log at the end of the production process and inserts a hash of the log into the Base RIM before the Base RIM is signed. It then stores the RIM onto the device or optionally provides a Uniform Resource Identifier (URI).



As depicted in the image above, the RIM files can be optionally placed in the boot partition. The provisioner will send these files to the ACA and they will be processed and stored in the database.

[3] https://trustedcomputinggroup.org/wp-content/uploads/TCG_PC_Client-FIM_v1r24_3feb20.pdf

1.2.2 TPM Event Log

The TPM Event log is defined in the TCG PC Client Platform Firmware Profile[4] which is referred to as the "PFP". The Event log contains all the hashes that get extended into the TPM PCRs values during the boot cycle, so one can recreate the resultant PCRs by extending the values within this file, therefore TPM PCR List may not be needed. This file will be needed to show what each PCR covers and to provide details should TPM Quote verification fail.

For provisioning, the TCG Event Log is one of the Support RIM file options for PC Client systems. This means that the Base RIM (SWID tag) file will have a hash of it in its payload for verification purposes.

1. The digest values found within the logs can be used to calculate the expected values in the TPM Quote.
2. The events in the RIM can be used to compare against the log provided by the client to detail which event caused the mis-compare.

[4] https://trustedcomputinggroup.org/wp-content/uploads/TCG_PCClientSpecPlat_TPM_2p0_1p04_pub.pdf

1.2.3 UEFI Boot Process

UEFI can record hashes of firmware components to the Trusted Platform Module (TPM) in the TPM Event Log. The TPM must be both activated and enabled for hashes to be written. Hashes normally capture firmware images, firmware configuration, expansion component firmware images, expansion component firmware configurations, and the bootloader. TPM-aware bootloaders can continue logging hashes to describe the kernel, initial file system, and any modules. Kernels, applications, and drivers can log runtime hashes to the TPM too.

Hashes are stored in the TPM's Platform Configuration Registers (PCR) in accordance with Figure 1. Most TPMs have 24 PCRs per supported hash algorithm. TPM 1.2 supports SHA-1 (24 PCRs). TPM 2.0 supports SHA-1 and SHA-256 at the minimum (48 PCRs minimum). PCR values are computed via a series of one-way hashes where each measurement hash is appended to the current PCR value, then the combination is hashed and becomes the new PCR value. Measurement hashes are recorded in an audit log for verification later.

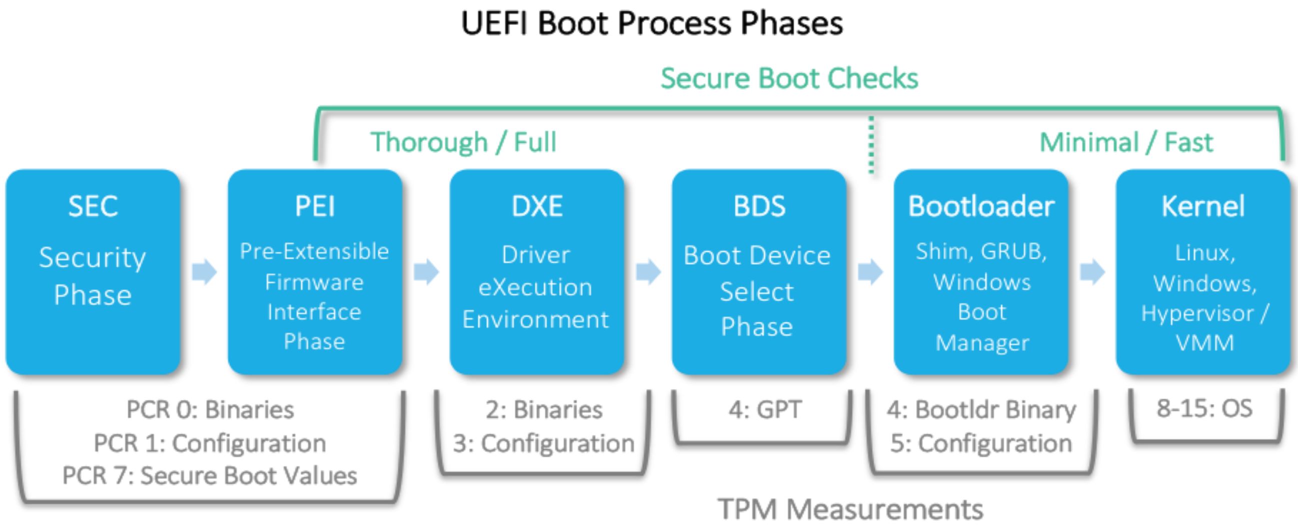


Fig. 1. *The interaction of Secure Boot and TPM with UEFI boot phases.*[5]

[5] https://media.defense.gov/2019/Jul/16/2002158058/-1/-1/0/CSI-BOOT-SECURITY-MODES-AND-RECOMMENDATIONS.PDF

2 Installing the RIM Tool

To install the RIM Tool, access the releases page on the HIRS GitHub repository. Make sure to download the RPM files which apply to the latest release. Currently installation packages are only available for Centos 7.

2.1 Centos 7 Installation

To install this tool, use the following command from the directory where the RIM Tool is placed:

```
sudo yum localinstall tcg_rim_tool-X.X.X-1.el7.x86_64.rpm
```

Where X.X.X is the latest version of the tcg_rim_tool package.

Note: Once installed, the tcg_rim_tool can be run from any directory in Linux. Sudo will be required when using the system log file as a support rim file when run on Linux.

2.2 Default Keys and Certs

After the tools are installed, a default key and associated self signed certificate are also created for testing purposes. When using the RIM Tool for creation or validation of a Base RIM, it should be noted that this default key and certificate will be used unless otherwise specified in the input command.

In reference to the examples used later in Sections 3.2 and 3.3, not using the -k option when inputting a command will use the default key instead. Similarly, if the -p option is not used, then the default certificate will be used instead.

If you would like to view the default certificate, Java Keytool can be invoked from the command line. The default key is stored within the .jks file which is associated with the keystore housing both of these.

- Refer to the key tool command page and Section 2.2.1 below for more information on the Java Keytool Keystore and it's many uses.

2.2.1 Using Java Keytool

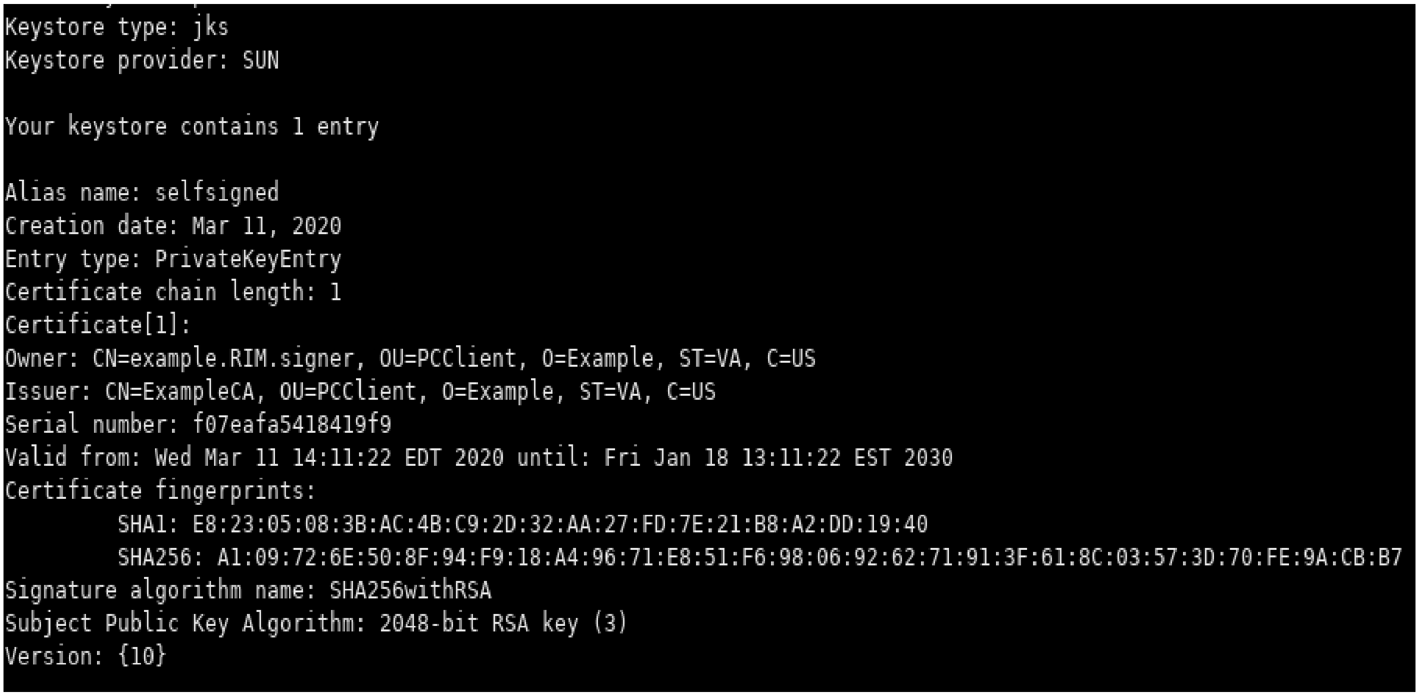
Java Keytool can be used to display the default keys and certificates that have been created after installation of the RIM Tool. To view them, use this command:

```
keytool -list -v -keystore /opt/hirs/rimtool/keystore.jks
```

After inputting this command, it will ask for a password to be able to view the keystore's contents'.

The password is: password

Once it has been typed in, the contents of the keystore will display like so:



```
Keystore type: jks
Keystore provider: SUN

Your keystore contains 1 entry

Alias name: selfsigned
Creation date: Mar 11, 2020
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=example.RIM.signer, OU=PCClient, O=Example, ST=VA, C=US
Issuer: CN=ExampleCA, OU=PCClient, O=Example, ST=VA, C=US
Serial number: f07eafa5418419f9
Valid from: Wed Mar 11 14:11:22 EDT 2020 until: Fri Jan 18 13:11:22 EST 2030
Certificate fingerprints:
         SHA1: E8:23:05:08:3B:AC:4B:C9:2D:32:AA:27:FD:7E:21:B8:A2:DD:19:40
         SHA256: A1:09:72:6E:50:8F:94:F9:18:A4:96:71:E8:51:F6:98:06:92:62:71:91:3F:61:8C:03:57:3D:70:FE:9A:CB:B7
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key (3)
Version: {10}
```

3 Using the RIM Tool – Linux

The tcg_rim_tool RPM will create a RIM command line shortcut. This can be invoked from a command line by using:

```
rim –h
```

Invoking this command will bring up a Help Page, which lists out the RIM Tool's many uses and functions.

3.1 Parameters

-c: --create "base"
The type of RIM to create. A base RIM will be created by default.

-a: -- attributes <path>
The configuration file holding attributes to populate the base RIM with. The Linux installation package will create a default configuration file located in /opt/hirs/rimtool. Refer to Appendix A for an example of a JSON configuration file.

-o: --out <path>
The file to write the RIM out to. The RIM will be written to stdout by default.

-v: --verify <path>
Specify a RIM file to verify. Will use the selfsigned cert in the keystore by default unless the –p parameter is used. It will check if the certificate used to check the <Signature> element is either in the default keystore.jks or corresponds to the certificate when provided with the –p option.

--keystore <path>
JKS keystore containing a private key to sign the base RIM created by the create function. The default keystore in /opt/hirs/rimtool/ is used by default. This parameter is not required if the –k or –p parameter(s) are specified.

-k: --privateKeyFile <path>
File containing the private key used to sign the base RIM created by the create function. The Private key in the default keystore will be used by default.

-p: --publicCertificate <path>
The public key certificate used to verify a RIM file or to embed in a signed RIM. A signed RIM generated by this tool by default will not show the signing certificate without this parameter present. The default for this option is the selfsigned cert in the keystore.

-l: --rimel <path>
The TCG eventlog file to use as a support RIM. By default the last system eventlog will be used.

-h: --help
Prints out the help text which provides the RIM's tools many uses and functions.

3.1.1 The JSON Configuration File

The tcg_rim _tool requires a json configuration file (specified by the –a option) to provide the attributes for the SWIDtag. Each tag within the json file corresponds to a file defined in the TCG Reference Integrity Manifest Information Model Table 1. Refer to Appendix A for an example JSON configuration file.

3.1.2 RIMLinkHash

The RIMLinkHash is a base64 encoded SHA256 digest of the RIM referenced by the Link element (Href rel=”supersedes” ,rel=”patches, or rel=”requires”). The Primary Base RIM does not require RIMLinkHash.

Note: The tcg_rim_tool will not perform encoding on RIMLinkHash and File.hash.

3.2 Creating a Base RIM

3.2.1 Creating a Base RIM without an Embedded Certificate

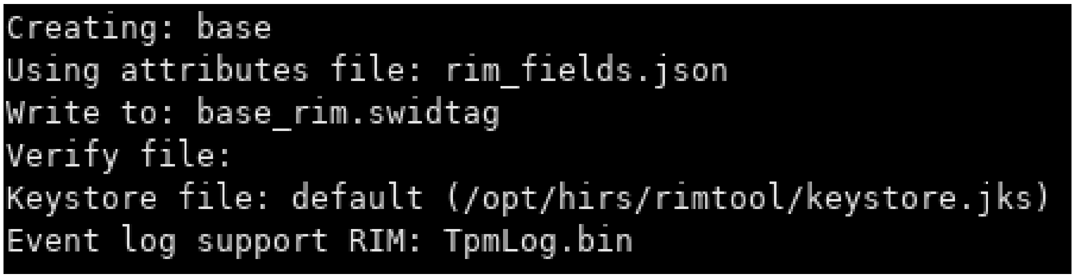
To create a Base RIM in this way, you need:

- A .json file which holds the attributes to populate the Base RIM
- A .bin file which holds a TCG Event Log file to use as a Support RIM
- A .pem file containing the private key used to sign the Base RIM

Once you have these files, you can input them into their specified places within this command:

```
rim -c base –a rim_fields.json –l TpmLog.bin –k privateRimKey.pem –o base_rim.swidtag
```

The tcg_rim_tool will create the Base RIM and it will be written out to the base_rim.swidtag file.



```
Creating: base
Using attributes file: rim_fields.json
Write to: base_rim.swidtag
Verify file:
Keystore file: default (/opt/hirs/rimtool/keystore.jks)
Event log support RIM: TpmLog.bin
```

This file will appear within the same directory the command was entered in, unless the path is specified using the –o option.

- ➢ See Appendix A for more information and test patterns.

3.2.3 Embedding a Certificate in the Base RIM

The XMLDSig specification allows for a certificate to be embedded into a signed object. The tcg_rim_tool provides a –p option to provide an embedded certificate.

Note: It is important to not include the entire certificate chain as it is prohibited by the PC Client RIM specification.

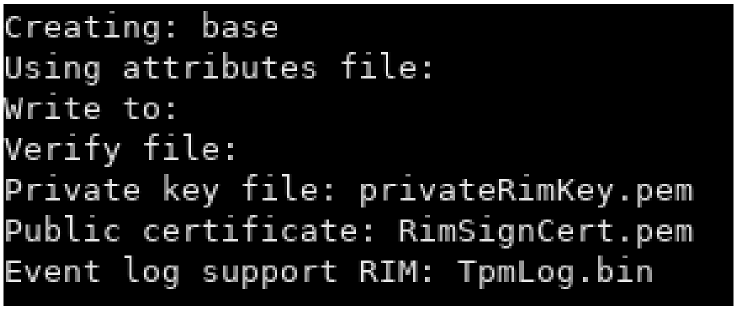
To create a Base RIM with an embedded cert, you need:

- A .bin file which holds a TCG Event Log file to use as a Support RIM
- A .pem file containing the private key used to sign the Base RIM
- A .pem file containing the public key certificate used to verify the RIM file

Once you have these files, you can input them into their specified places within this command:

```
rim -c base –l TpmLog.bin –k privateRimKey.pem –p RimSignCert.pem –o
base_rim_embed.swidtag
```

The tcg_rim_tool will create the Base RIM and embed the certificate into the signature block:



```
Creating: base
Using attributes file:
Write to:
Verify file:
Private key file: privateRimKey.pem
Public certificate: RimSignCert.pem
Event log support RIM: TpmLog.bin
```

This will also output the RimSignCert.pem file and it’s updated contents.

- See Appendix A for more information and test patterns.

3.3 Validating a Base RIM

Validating the base RIM checks for:

- Everything inside the <SoftwareIdentity> tags within the .swidtag is checked against the schema. This does not include the <Signature> block.
- The <Signature> block is validated by checking that the hash in the <DigestValue> element matches the value generated by hashing the <SoftwareIdentity> (without the <Signature>) by the <DigestMethod>.

To validate a Base RIM, you need:

- A .swidtag file which the Base RIM was written to
- A .bin file which holds an external Support RIM
- A .pem file containing the public key certificate(s) used to verify the RIM file

Note: This method will work with any .swidtag file which has an embedded certificate or one that doesn't have an embedded certificate. The difference lies within the .pem file holding the pub key certificate used to validate the Base RIM.

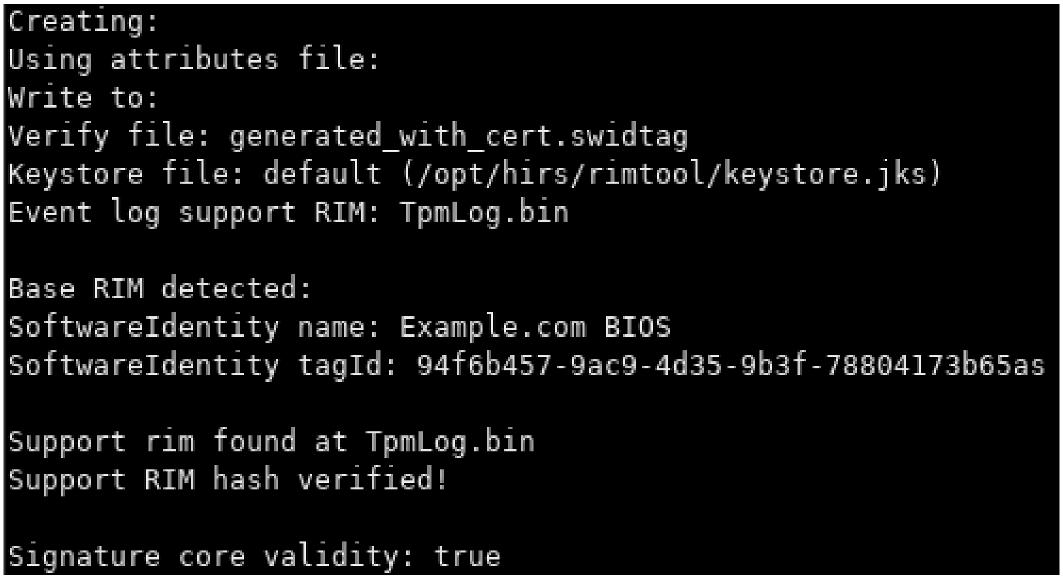
- ➢ See Appendix A for an example of a .pem file used to validate an example .swidtag file with an embedded cert and one without.

Once you have these files, you can input them into their specified places within this command:

```
rim –v base_rim.swidtag –l TpmLog.bin –p CaChain.pem
```

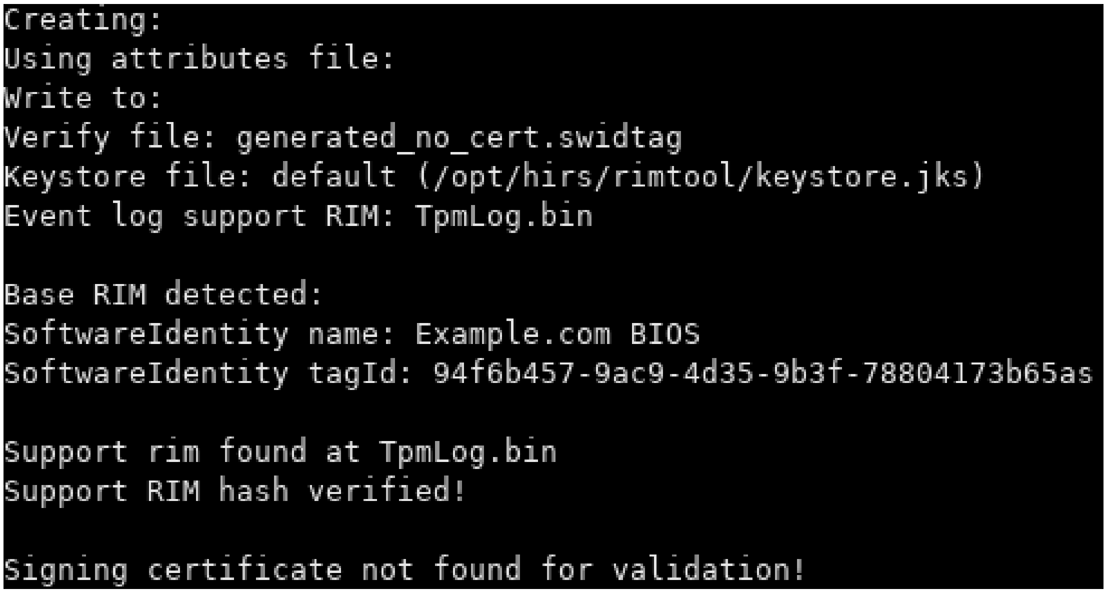
^ need cert chain

The tcg_rim_tool will verify if the signature on the certificate is valid or not.



```
Creating:
Using attributes file:
Write to:
Verify file: generated_with_cert.swidtag
Keystore file: default (/opt/hirs/rimtool/keystore.jks)
Event log support RIM: TpmLog.bin

Base RIM detected:
SoftwareIdentity name: Example.com BIOS
SoftwareIdentity tagId: 94f6b457-9ac9-4d35-9b3f-78804173b65as

Support rim found at TpmLog.bin
Support RIM hash verified!

Signature core validity: true
```

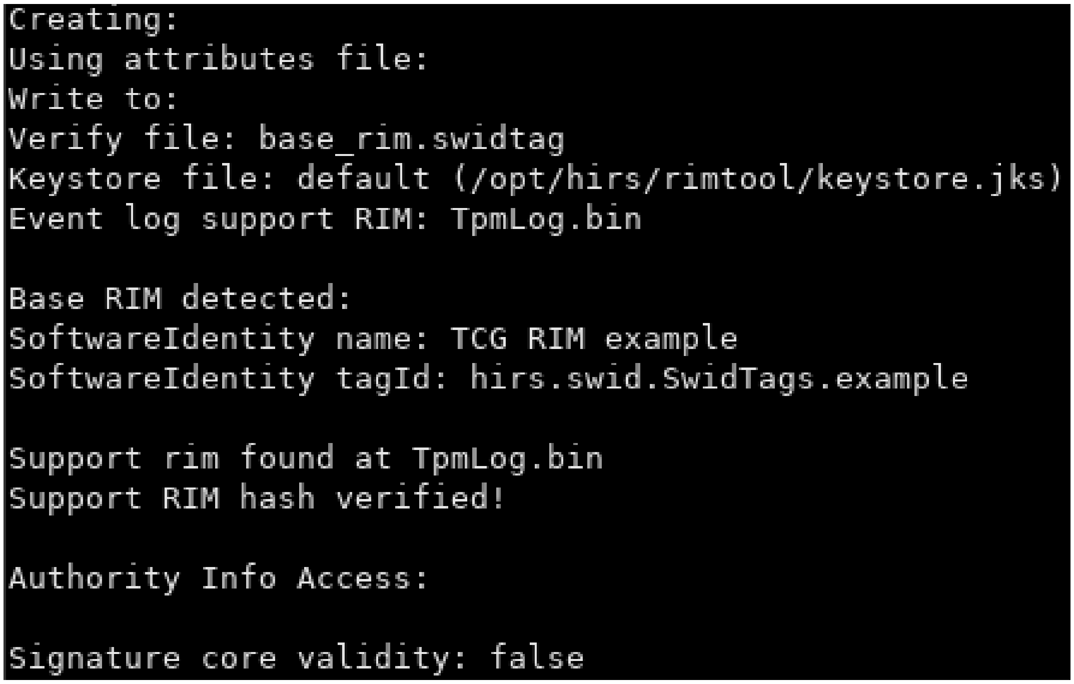
In this case, the signature on the certificate is valid. The signature on the Base RIM and the certificate match.



```
Creating:
Using attributes file:
Write to:
Verify file: generated_no_cert.swidtag
Keystore file: default (/opt/hirs/rimtool/keystore.jks)
Event log support RIM: TpmLog.bin

Base RIM detected:
SoftwareIdentity name: Example.com BIOS
SoftwareIdentity tagId: 94f6b457-9ac9-4d35-9b3f-78804173b65as

Support rim found at TpmLog.bin
Support RIM hash verified!

Signing certificate not found for validation!
```

There may also be a case where the validity of the Base RIM returned is false. This could mean that the signature on the certificate doesn’t match with the signature on the base RIM. You can see the output of this below:

```
rim –v base_rim.swidtag –l TpmLog.bin –p CaChain.pem
```



```
Creating:
Using attributes file:
Write to:
Verify file: base_rim.swidtag
Keystore file: default (/opt/hirs/rimtool/keystore.jks)
Event log support RIM: TpmLog.bin

Base RIM detected:
SoftwareIdentity name: TCG RIM example
SoftwareIdentity tagId: hirs.swid.SwidTags.example

Support rim found at TpmLog.bin
Support RIM hash verified!

Authority Info Access:

Signature core validity: false
```

- See Appendix A for more information and test patterns.

Appendix A: Test Patterns

JSON Config File

Config file (*rim_fields.json*)

```
{
"SoftwareIdentity": {
"name": "TCG RIM example",
"version": "0.1",
"tagId": "hirs.swid.SwidTags.example",
"tagVersion": "1",
"patch": false
},
"Entity": {
"name": "HIRS",
"role": "softwareCreator,tagCreator",
"regid": "www.example.com",
"thumbprint": ""
},
"Link": {
"href": "https://Example.com/support/ProductA/firmware/installfiles",
"rel": "installationmedia"
},
"Meta": {
"colloquialVersion": "",
"edition": "",
"product": "",
"revision": "",
"platformManufacturerStr": "Example.com",
"platformManufacturerId": "00201234",
"platformModel": "ProductA",
"bindingSpec": "IOT RIM",
"bindingSpecVersion": "1.2",
"rimLinkHash": "88f21d8e44d4271149297404df91caf207130bfa116582408abd04ede6db7f51"
},
"Payload": {
"Directory": {
"name": "iotBase",
"File": {
```

```
"name": "TpmLog.bin"
}
}
}
}
```

Keys

Private key file *(privateRimKey.pem)*:
An RSA 2048 bit key will be used as the key for signing the Base RIM. Any key or certificate will be shown in PEM format.

Private Key:

-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQCndZVhpEkbsQAG
NsB2oNhlcVJNXWcdj06X0Dn5by3nHAFcGvJkIZbEREavkCvIpY/F36rOOP5wWnHG
Qzo3XyMSFAjH8IRl48QmqmW4E7nDbBMQ57uGq0xq2qAMMx4NHFS4ik/wsY/TS+HX
DIMUV7D3R3129Pdtwu8WHdrdqn1tObpoGo+6nkJenOvAhTbBl+CQPK1hUGb7xseQ
xpSuCk3Iz7kECDbOX8WrDSqi2Noavs/Nsf66sK9oZsosw+haRUsiAATxDD3wdN+h
hIUcLtVhDJKvMiKdo4EVKBWJHvaQd6YfaVaMNB4eJjRUig+KT+NVUimNZ7rPN5nZ
LrpD8uaNAgMBAAECggEAcnG8npd9U0x7HMQMcsZoPaPdwHvF/gCzkLNA+8RM1bZh
A4ZzA5WlCQs0V8Wq9pyXjn7Wp8txsG1PdlT5k2AUgsVoXuR0R4IKyvYHQG9StEjH
GvWURmwJdLlnSg8hSYqEJ/52taNUDO6+MI8fgiaQDd8w0ryF4OCpLy9GJdnfkGYZ
Ayemb3USFUdj/S67NVqxnvAfFMM5FqkKGhkoy7wBRgO6eOeJvoTq8LMiPiponwwF
DW409ZStbrk1f1Oszst/UvFUWA9BdDfeoPmFR61y3eB5zlMQG8Mhr2v5hvkj9TPX
FU4Fm4EzZ1h/60cdWoP6XYCP7F2NqZ8N8u4UBQNAIQKBgQDcGIw5GJEvRF+FFTTR
hYatMRn80DGTVjdT32MgajdKx05OWxBmQsFob34fiSnr0wAXPJeDXG4ruMBE2bSk
EC8rCO08G8ihQoH8x0cvuERe1fpVWk3RWNucVGIiJSEXAIwWrlYZLTfYd5GqBkPE
OQxxo4MtOyqeHmVH1mOywk9ABQKBgQDCxt95luzqQZV9Xl78QQvOIbjOdHLjY23Z
yp8sGt9birL/WZ33TCRgmH1e61BdrSqO7Om/ail2Y59XM5UU6kLbDj0IgmOPTsrJ
JmIVf8r3bKltVUaLePgr4yex7dmtHRH8OkLXKnE0RCO0kCi9kJMB12yE3pWxk+Pu
zztQd3a66QKBgBNJd2g9deONe01fOVyu9clRhzR3ThDaOkj4R2h8xlGgO4V0R3Ce
ovIy6vt6epj2yYg/wAs720+rhfXCmijSXj/ILXnZ+W/gMyHimKNe42boG2LFYhJZ
Vg1R+7OAS3EHlD8ckeDs7Hrkp3gdymx0j1mZ+ZHKIIbwpPFxoRT2IBm9AoGBAI0Z
bIK0puP8psKvPrgWluq42xwUl7XKLaX8dtqIjQ3PqGP7E8g2TJP9Y7UDWrDB5Xas
gZi821R8Ts3o/DKukcgGxIgJjP4f4h9dwug4L1yWRxaBFB2tgHqqj/MBjxMtX/4M
Zqdgg6mNQyBm3lyVAynuWRrX9DE0JYa2cQ2VvVkhAoGBAMBv/oT813w00759PmkO
Uxv3LXTJuYBbq0Rmga25jN3ow8LrGQdSVg7F/af3I5KUF7mLiegDy1pkRfauyXH7
+WhEqnf86vDrzPpytDMxinWOQZusCqeWHb+nuVTuL3Fv+GxEdwVGYI/7lFJ7B//h
P5rU93ZoYY7sWcGVqaaEkMRU
-----END PRIVATE KEY-----

X.509 Certificates

Root CA certificate (*cacert.pem*) file:

```
-----BEGIN CERTIFICATE-----
MIIDITCCAgmgAwIBAgIGAWQoEa4DMA0GCSqGSIb3DQEBCwUAMEcxCzAJBgNVBAYTAlVTMQswCQYDVQQID
AJTVDEQMA4GA1UEBwwHRVhBTVBMRTEMMAoGA1UECgwDb3JnMQswCQYDVQQDDAJjYTAeFw0xODA2MjIx
NTE2NDRaFw0yODA2MjIxNTE2NDRaMEcxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJTVDEQMA4GA1UEBwwH
RVhBTVBMRTEMMAoGA1UECgwDb3JnMQswCQYDVQQDDAJjYTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAL/ix+VhoUknD897XTa/6E8+1G2lRdSUleSdLV3AXQGD2yMVhGTBby5YuxqUUtvLzellXszWlwaszF
oCLZ9EtqRr1cre/mu3W1MfTqo7eiNo2436sD2rq5Eab9P6hpsUxhBbA6KCiCrFuN/98ftF/LX6169ojDkoLYJ7D
whtPWQbJRqBHTwXWvw4D153IfCIeRR0jk1NR65DmFXYwlQM2jkpciIFh7lnDd6r0l/4F6bo1QA0mjEW65dmW6
f2hFUmXPGDxP+08wWE2TL7tJcSLRxkGBd9FAcvH0te3o0llYGiGAS23ys/UfzH5kN5U/dkl0DVaooxWUWRRZTe
gME0/EECAwEAAaMTMBEwDwYDVR0TAQH/BAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEAlcpXo4maGSl2
HfL8YxuCWOp5ZMMa6DNT0gHG5Nf9AYA0R+WNkxmULSZGEb5ARzvZe1gdgJxZaA530Al5gIcMQ/MHRiWpu
8uYzloax+HWJr0PJ6Cy4SOtpt8a2sZABTGcLEZjj+R8/qyJUnTJBYzzLIuXfCHg0TmrrNjUTe7md0G7rmCw8ixMd
Ta5ZHQ/EXl2xvYZmcNfmYIGTBaja4u2vQYBbLYqEycKmgxv/dp/RRFdGa73SZHRkGnK/S7cLOMwvW6f3O63+
ZZqx8+nkgOJ3Wjk/dbz8mKFH/7H2zpOSMZJQ6DfhhmWmTlPr+8h1kPkaOhrQnbRfg/db/xCCNx7NQ==
-----END CERTIFICATE-----
```

Certificate used to verify RIMs CSignature (*RimSignCert.pem*)

```
-----BEGIN CERTIFICATE-----
MIID2jCCAsKgAwIBAgIJAP0uwoNdwZDFMA0GCSqGSIb3DQEBCwUAMFMxCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJWQTEQMA4GA1UECgwHRXhhbXBsZTERMA8GA1UECwwI
UENDbGllbnQxEjAQBgNVBAMMCUV4YW1wbGVDQTAeFw0yMDA3MjEyMTQ1MDBaFw0z
MDA1MzAyMTQ1MDBaMFwxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJWQTEQMA4GA1UE
CgwHRXhhbXBsZTERMA8GA1UECwwIUENDbGllbnQxGzAZBgNVBAMMEmV4YW1wbGUu
UklNLnNpZ25lcjCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAKd1lWGk
SRuxAAY2wHag2GVxUk1dZx2PTpfQOflvLeccAVwa8mQhlsRERq+QK8ilj8Xfqs44
/nBaccZDOjdfIxIUCMfwhGXjxCaqZbgTucNsExDnu4arTGraoAwzHg0cVLiKT/Cx
j9NL4dcMgxRXsPdHfXb0923C7xYd2t2qfW05umgaj7qeQl6c68CFNsGX4JA8rWFQ
ZvvGx5DGlK4KTcjPuQQINs5fxasNKqLY2hq+z82x/rqwr2hmyizD6FpFSyIABPEM
PfB036GEhRwu1WEMkq8yIp2jgRUoFYke9pB3ph9pVow0Hh4mNFSKD4pP41VSKY1n
us83mdkuukPy5o0CAwEAAaOBpzCBpDAdBgNVHQ4EFgQUL96459AwoiCdqgGGGpZP
7ezyvMEwHwYDVR0jBBgwFoAURqG47dumcV/Q0ud6ijxdbprDljgwCQYDVR0TBAIw
ADALBgNVHQ8EBAMCBsAwEwYDVR0lBAwwCgYIKwYBBQUHAwMwNQYIKwYBBQUHAQEE
KTAnMCUGCCsGAQUFBzAChhlodHRwczovL2V4YW1wbGUuY29tL2NlcnRzMA0GCSqG
SIb3DQEBCwUAA4IBAQDpKx5oQlkS11cg7Qp58BmCvjCzFpof+qYePooJsD3i5SwK
fRTa2CkDMww9qrwBK7G60y7jhe5InKTdqIlVqaji5ZImR0QMKTtk7zt9AJ9EaEzK
xfDiE/qX34KxNe4ZmbvLH8N+BSujQXMMi56zGjW469Y/rbDMG8uU1dq3zqhO5b+d
Ur1ecdkYLgzxu6O+oWy5JpVibmcjvNezJsUtjc+km2FYm24vU3/fCNzZ2z0EHQES
cIEQ5OqfpdFrV3De238RhMH6J4xePSidnFpfBc6FrdyDI1A8eRFz36I4xfVL3ZnJ
P/+j+NE4q6yz5VGvm0npLO394ZihtsI1sRAR8ORJ
-----END CERTIFICATE-----
```

Cert Chain (*CaChain.pem*)

```
-----BEGIN CERTIFICATE-----
MIIDITCCAgmgAwIBAgIGAWQoEa4DMA0GCSqGSIb3DQEBCwUAMEcxCzAJBgNVBAYTAlVTMQswCQYDVQQID
AJTVDEQMA4GA1UEBwwHRVhBTVBMRTEMMAoGA1UECgwDb3JnMQswCQYDVQQDDAJjYTAeFw0xODA2MjIx
NTE2NDRaFw0yODA2MjIxNTE2NDRaMEcxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJTVDEQMA4GA1UEBwwH
RVhBTVBMRTEMMAoGA1UECgwDb3JnMQswCQYDVQQDDAJjYTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAL/ix+VhoUknD897XTa/6E8+1G2lRdSUleSdLV3AXQGD2yMVhGTBby5YuxqUUtvLzellXszWlwaszF
oCLZ9EtqRr1cre/mu3W1MfTqo7eiNo2436sD2rq5Eab9P6hpsUxhBbA6KCiCrFuN/98ftF/LX6169ojDkoLYJ7D
whtPWQbJRqBHTwXWvw4D153IfCIeRR0jk1NR65DmFXYwlQM2jkpciIFh7lnDd6r0l/4F6bo1QA0mjEW65dmW6
f2hFUmXPGDxP+08wWE2TL7tJcSLRxkGBd9FAcvH0te3o0llYGiGAS23ys/UfzH5kN5U/dkl0DVaooxWUWRRZTe
gME0/EECAwEAAaMTMBEwDwYDVR0TAQH/BAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEAlcpXo4maGSl2
HfL8YxuCWOp5ZMMa6DNT0gHG5Nf9AYA0R+WNkxmULSZGEb5ARzvZe1gdgJxZaA530Al5gIcMQ/MHRiWpu
8uYzloax+HWJr0PJ6Cy4SOtpt8a2sZABTGcLEZjj+R8/qyJUnTJBYzzLIuXfCHg0TmrrNjUTe7md0G7rmCw8ixMd
Ta5ZHQ/EXl2xvYZmcNfmYIGTBaja4u2vQYBbLYqEycKmgxv/dp/RRFdGa73SZHRkGnK/S7cLOMwvW6f3O63+
ZZqx8+nkgOJ3Wjk/dbz8mKFH/7H2zpOSMZJQ6DfhhmWmTlPr+8h1kPkaOhrQnbRfg/db/xCCNx7NQ==
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIID2jCCAsKgAwIBAgIJAP0uwoNdwZDFMA0GCSqGSIb3DQEBCwUAMFMxCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJWQTEQMA4GA1UECgwHRXhhbXBsZTERMA8GA1UECwwI
UENDbGllbnQxEjAQBgNVBAMMCUV4YW1wbGVDQTAeFw0yMDA3MjEyMTQ1MDBaFw0z
MDA1MzAyMTQ1MDBaMFwxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJWQTEQMA4GA1UE
CgwHRXhhbXBsZTERMA8GA1UECwwIUENDbGllbnQxGzAZBgNVBAMMEmV4YW1wbGUu
UklNLnNpZ25lcjCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAKd1lWGk
SRuxAAY2wHag2GVxUk1dZx2PTpfQOflvLeccAVwa8mQhlsRERq+QK8ilj8Xfqs44
/nBaccZDOjdfIxIUCMfwhGXjxCaqZbgTucNsExDnu4arTGraoAwzHg0cVLiKT/Cx
j9NL4dcMgxRXsPdHfXb0923C7xYd2t2qfW05umgaj7qeQl6c68CFNsGX4JA8rWFQ
ZvvGx5DGlK4KTcjPuQQINs5fxasNKqLY2hq+z82x/rqwr2hmyizD6FpFSyIABPEM
PfB036GEhRwu1WEMkq8yIp2jgRUoFYke9pB3ph9pVow0Hh4mNFSKD4pP41VSKY1n
us83mdkuukPy5o0CAwEAAaOBpzCBpDAdBgNVHQ4EFgQUL96459AwoiCdqgGGGpZP
7ezyvMEwHwYDVR0jBBgwFoAURqG47dumcV/Q0ud6ijxdbprDljgwCQYDVR0TBAIw
ADALBgNVHQ8EBAMCBsAwEwYDVR0lBAwwCgYIKwYBBQUHAwMwNQYIKwYBBQUHAQEE
KTAnMCUGCCsGAQUFBzAChhlodHRwczovL2V4YW1wbGUuY29tL2NlcnRzMA0GCSqG
SIb3DQEBCwUAA4IBAQDpKx5oQlkS11cg7Qp58BmCvjCzFpof+qYePooJsD3i5SwK
fRTa2CkDMww9qrwBK7G60y7jhe5InKTdqIlVqaji5ZImR0QMKTtk7zt9AJ9EaEzK
xfDiE/qX34KxNe4ZmbvLH8N+BSujQXMMi56zGjW469Y/rbDMG8uU1dq3zqhO5b+d
Ur1ecdkYLgzxu6O+oWy5JpVibmcjvNezJsUtjc+km2FYm24vU3/fCNzZ2z0EHQES
cIEQ5OqfpdFrV3De238RhMH6J4xePSidnFpfBc6FrdyDI1A8eRFz36I4xfVL3ZnJ
P/+j+NE4q6yz5VGvm0npLO394ZihtsI1sRAR8ORJ
-----END CERTIFICATE-----
```

Support RIM files

TCG TPM Event Log *(TpmLog.bin)* :

https://github.com/nsacyber/HIRS/blob/master/tools/tcg_rim_tool/src/test/resources/TpmLog.bin

Base RIM Files

Base RIM with embedded Signing Certificate (*base_rim_embedded.swidtag*):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<SoftwareIdentity xmlns="http://standards.iso.org/iso/19770/-2/2015/schema.xsd"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#" corpus="false" name="TCG RIM example"
patch="false" supplemental="false" tagId="hirs.swid.SwidTags.example" tagVersion="1" version="0.1"
versionScheme="multipartnumeric" xml:lang="en">
<Entity name="HIRS" regid="www.example.com" role="softwareCreator tagCreator"/>
<Link href="https://Example.com/support/ProductA/firmware/installfiles" rel="installationmedia"/>
<Meta xmlns:rim="https://trustedcomputinggroup.org/wp-content/uploads/TCG_RIM_Model"
rim:BindingSpec="IOT RIM" rim:BindingSpecVersion="1.2" rim:platformManufacturerId="00201234"
rim:platformManufacturerStr="Example.com" rim:platformModel="ProductA"
rim:rimLinkHash="88f21d8e44d4271149297404df91caf207130bfa116582408abd04ede6db7f51"/>
<Payload>
<Directory name="iotBase">
<File xmlns:SHA256="http://www.w3.org/2001/04/xmlenc#sha256"
SHA256:hash="4479ca722623f8c47b703996ced3cbd981b06b1ae8a897db70137e0b7c546848"
name="TpmLog.bin" size="7549"/>
</Directory>
</Payload>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#"><SignedInfo>
<CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
20010315"/><SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-
sha256"/><Reference URI=""><Transforms><Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/></Transforms>
<DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/><DigestValue>0smsf/30jpI5914zDv9QP5Qe
ZeplY11VGANEntnEYaU=</DigestValue></Reference></SignedInfo>
<SignatureValue>QM81vo7svJ0Z99Tw/n3UqXNdgbP7yEnovznd2QbOBGEzQdy1UpkEw7yJRT5oHyilWSBUdm
Uaf1Rz&#13;3OJOUj/6FN7IvOO99vbkSeieGYtHhLCurwXsmjj5eCiPUEHXjYD05VnnsqLuA4752WhiGu9l+g5q
&#13;kkKx5FSLnplYQ+2RRSN5tLKq2ocBQoUqN7V0iTKC+cein1GfkvrgTHQ5BNX6Ze6HdAL26KPgo0Ub&#13
;ZNxGewPNyn2nAN0mdl+gNWBvwe51W9v9mLo7Cz03lkbE3Bwwe7pEGhKzHnnLPrkQrLxOezW9W1Ns&#13;l
uBLvdhIqdJC0RiOtFKjC0DSAxBjh88dGA10nw==</SignatureValue>
<KeyInfo>
	<X509Data>
	<X509SubjectName>CN=example.RIM.signer,OU=PCClient,O=Example,ST=VA,C=US</X509Subj
	ectName>
	<X509Certificate>MIID2jCCAsKgAwIBAgIJAP0uwoNdwZDFMA0GCSqGSIb3DQEBCwUAMFMxCzAJBg
	NVBAYTAlVTMQsw&#13;CQYDVQQIDAJWQTEQMA4GA1UECgwHRXhhbXBsZTERMA8GA1UECwwIUE
	NDbGllbnQxEjAQBgNVBAMM&#13;CUV4YW1wbGVDQTAeFw0yMDA3MjEyMTQ1MDBaFw0zMDA1Mz
```

```
        AyMTQ1MDBaMFwxCzAJBgNVBAYTAlVT&#13;MQswCQYDVQQIDAJWQTEQMA4GA1UECgwHRXhhbX
        BsZTERMA8GA1UECwwIUENDbGllbnQxGzAZBgNV&#13;BAMMEmV4YW1wbGUuUklNLnNpZ25lcjCC
        ASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAKd1&#13;lWGkSRuxAAY2wHag2GVxUk1dZx2
        PTpfQOflvLeccAVwa8mQhlsRERq+QK8ilj8Xfqs44/nBaccZD&#13;OjdfIxIUCMfwhGXjxCaqZbgTucNs
        ExDnu4arTGraoAwzHg0cVLiKT/Cxj9NL4dcMgxRXsPdHfXb0&#13;923C7xYd2t2qfW05umgaj7qeQl
        6c68CFNsGX4JA8rWFQZvvGx5DGlK4KTcjPuQQINs5fxasNKqLY&#13;2hq+z82x/rqwr2hmyizD6FpFS
        yIABPEMPfB036GEhRwu1WEMkq8yIp2jgRUoFYke9pB3ph9pVow0&#13;Hh4mNFSKD4pP41VSKY1nus
        83mdkuukPy5o0CAwEAAaOBpzCBpDAdBgNVHQ4EFgQUL96459AwoiCd&#13;qgGGGpZP7ezyvMEw
        HwYDVR0jBBgwFoAURqG47dumcV/Q0ud6ijxdbprDljgwCQYDVR0TBAIwADAL&#13;BgNVHQ8EBAM
        CBsAwEwYDVR0lBAwwCgYIKwYBBQUHAwMwNQYIKwYBBQUHAQEEKTAnMCUGCCsGAQUF&#13;BzA
        ChhlodHRwczovL2V4YW1wbGUuY29tL2NlcnRzMA0GCSqGSIb3DQEBCwUAA4IBAQDpKx5oQlkS&#1
        3;11cg7Qp58BmCvjCzFpof+qYePooJsD3i5SwKfRTa2CkDMww9qrwBK7G60y7jhe5InKTdqIlVqaji&#1
        3;5ZImR0QMKTtk7zt9AJ9EaEzKxfDiE/qX34KxNe4ZmbvLH8N+BSujQXMMi56zGjW469Y/rbDMG8uU
        &#13;1dq3zqhO5b+dUr1ecdkYLgzxu6O+oWy5JpVibmcjvNezJsUtjc+km2FYm24vU3/fCNzZ2z0EH
        QES&#13;cIEQ5OqfpdFrV3De238RhMH6J4xePSidnFpfBc6FrdyDI1A8eRFz36I4xfVL3ZnJP/+j+NE4q6
        yz&#13;5VGvm0npLO394ZihtsI1sRAR8ORJ</X509Certificate>
        </X509Data>
<KeyValue>
        <RSAKeyValue><Modulus>p3WVYaRJG7EABjbAdqDYZXFSTV1nHY9Ol9A5+W8t5xwBXBryZCGWxER
        Gr5AryKWPxd+qzjj+cFpx&#13;xkM6N18jEhQIx/CEZePEJqpluBO5w2wTEOe7hqtMatqgDDMeDRxUu
        IpP8LGP00vh1wyDFFew90d9&#13;dvT3bcLvFh3a3ap9bTm6aBqPup5CXpzrwIU2wZfgkDytYVBm+8b
        HkMaUrgpNyM+5BAg2zl/Fqw0q&#13;otjaGr7PzbH+urCvaGbKLMPoWkVLIgAE8Qw98HTfoYSFHC7V
        YQySrzIinaOBFSgViR72kHemH2lW&#13;jDQeHiY0VIoPik/jVVIpjWe6zzeZ2S66Q/LmjQ==</Modulu
        s><Exponent>AQAB</Exponent></RSAKeyValue>
</KeyValue>
</KeyInfo>
</Signature>
</SoftwareIdentity>
```

Base RIM without an embedded certificate (*base_rim.swidtag)*:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<SoftwareIdentity xmlns="http://standards.iso.org/iso/19770/-2/2015/schema.xsd"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#" corpus="false" name="Example.com BIOS"
patch="false" supplemental="false" tagId="94f6b457-9ac9-4d35-9b3f-78804173b65as" tagVersion="0"
version="01" versionScheme="multipartnumeric" xml:lang="en">
<Entity name="Example Inc" regid="http://Example.com" role="softwareCreator tagCreator"/>
<Link href="https://Example.com/support/ProductA/firmware/installfiles" rel="installationmedia"/>
<Meta xmlns:n8060="http://csrc.nist.gov/ns/swid/2015-extensions/1.0"
xmlns:rim="https://trustedcomputinggroup.org/wp-content/uploads/TCG_RIM_Model"
n8060:colloquialVersion="Firmware_2019" n8060:edition="12" n8060:product="ProductA"
n8060:revision="r2" rim:BindingSpec="PC Client RIM" rim:BindingSpecVersion="1.2"
rim:PayloadType="direct" rim:firmwareManufacturerId="00213022"
rim:firmwareManufacturerStr="BIOSVendorA" rim:firmwareModel="A0" rim:firmwareVersion="12"
rim:pcURIGlobal="https://Example.com/support/ProductA/"
rim:pcURILocal="/boot/tcg/manifest/switag/" rim:platformManufacturerId="00201234"
rim:platformManufacturerStr="Example.com" rim:platformModel="ProductA" rim:platformVersion="01"/>
<Payload xmlns:rim="https://trustedcomputinggroup.org/wp-content/uploads/TCG_RIM_Model"
rim:supportRIMFormat="TCG_EventLog_Assertion"
rim:supportRIMURIGlobal="https://Example.com/support/ProductA/firmware/rims/">
<Directory name="rim">
<File xmlns:SHA256="http://www.w3.org/2001/04/xmlenc#sha256"
SHA256:hash="4479ca722623f8c47b703996ced3cbd981b06b1ae8a897db70137e0b7c546848"
name="Example.com.BIOS.01.rimel" size="7549"/>
</Directory>
</Payload>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<SignedInfo>
<CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
<SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<Reference URI="">
<Transforms>
<Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
</Transforms>
<DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<DigestValue>97uWB7zSsO5WaGbrcQrlKd1Bju0aDTjK1/ktUYBje8A=</DigestValue>
</Reference>
</SignedInfo>
<SignatureValue>N1YtTeo2Ryuj+CtlXIpICEay+ni7vt8+4J7tAsYpa3efnLwtea69PIqEylPWm9LdA8Eo8XDdpgx
V7h3hi2LTOU+Wxq3bLiLamo99T1EtIwl+ZPcOv8bsfEkmShHdMC0dlfcj6r7x4tc0XkNAhhJgfRNz
```

```
FsmPWKJb6FYcsHFbHO/Uw1hSokbAGcWWTshEOqvKHMa8UVkrFMUPnrnMtdyJqZlhDBrZHNi4rWth
8TjlUnQVSCF9s9I04FxJ1cUAdeVMHtXKM8Pvjv68PaJMJK73dW5Yd3SbcgoKLesf/HPWeeZL0rr4
TNjlqJ/wq61Ons45MFG9bIscVbnd+XxFHx8Skw==</SignatureValue>
<KeyInfo>
<KeyName>2fdeb8e7d030a2209daa01861a964fedecf2bcc1</KeyName>
<KeyValue>
<RSAKeyValue>
        <Modulus>p3WVYaRJG7EABjbAdqDYZXFSTV1nHY9Ol9A5+W8t5xwBXBryZCGWxERGr5AryKWPxd+
        qzjj+cFpxxkM6N18jEhQIx/CEZePEJqpluBO5w2wTEOe7hqtMatqgDDMeDRxUuIpP8LGP00vh1wyDFF
        ew90d9dvT3bcLvFh3a3ap9bTm6aBqPup5CXpzrwlU2wZfgkDytYVBm+8bHkMaUrgpNyM+5BAg2zl/
        Fqw0qotjaGr7PzbH+urCvaGbKLMPoWkVLIgAE8Qw98HTfoYSFHC7VYQySrzIinaOBFSgViR72kHemH2l
        WjDQeHiY0VIoPik/jVVIpjWe6zzeZ2S66Q/LmjQ==</Modulus>
        <Exponent>AQAB</Exponent>
</RSAKeyValue>
</KeyValue>
</KeyInfo>
</Signature>
</SoftwareIdentity>
```

Appendix B: Building the RIM Tool

B.1 Windows

1. Building the RIM Tool

Currently, only an installation file for Linux RPM is supported.

B.2 Linux

1. Building the RIM Tool

To build this tool, navigate to the tcg_rim_tool directory and use the following command:

```
.gradlew clean build
```

Appendix C: Packaging the RIM Tool

C.1 Windows

1. **Packaging the RIM Tool**

Currently, only an installation file for Linux RPM is supported.

C.2 Linux

1. **Packaging the RIM Tool**

To package the tcg_rim_tool use the package.sh script to produce an RPM file for Linux distributions that support the RPM package manager. The RPM file will be located in the rpmbuild/RPMS/x86_64/ directory if the package script was successful. Although packaging for other distributions is not currently available, the tool can be built and run on other systems that support Java and Gradle, such as Windows 10.

Appendix D: References

[1] Trusted Computing Group. (2020, November 12). *TCG Reference Integrity Manifest (RIM) Information Model.* Retrieved from Trusted Computing Group: https://trustedcomputinggroup.org/wp-content/uploads/TCG_RIM_Model_v1p01_r0p16_pub.pdf

[2] Trusted Computing Group. (2020, March 31). *TCG PC Client Reference Integrity Manifest Specification.* Retrieved from Trusted Computing Group: https://trustedcomputinggroup.org/wp-content/uploads/TCG_PC_Client_RIM_r0p15_15june2020.pdf

[3] Trusted Computing Group. (2019, June 3). *TCG PC Client Platform Firmware Profile Specification*. Retrieved from Trusted Computing Group: https://trustedcomputinggroup.org/wp-content/uploads/TCG_PCClientSpecPlat_TPM_2p0_1p04_pub.pdf

[4] Cybersecurity Requirements Center. (2019, June). *Boot Security Modes and Recommendations.* Retrieved from National Security Agency: https://media.defense.gov/2019/Jul/16/2002158058/-1/-1/0/CSI-BOOT-SECURITY-MODES-AND-RECOMMENDATIONS.PDF

[5] National Institute of Standards and Technology. (2016, April 29). *Guidelines for the Creation of Interoperable Software ID (SWID) Tags: NISTIR 8060.* Retrieved from National Institute of Standards and Technology: https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8060.pdf

[6] International Organization for Standardization/International Electrotechnical Commission. (2017, March). *ISO/IEC 19770-2:2015(en): Information technology — Software asset management — Part 2: Software identification tag.* Retrieved from International Organization for Standardization: https://www.iso.org/standard/65666.html

[7] The IETF & W3C. (2015, July 23). *XML Signature Syntax and Processing Version 2.0.* Retrieved from W3C Working Group: https://www.w3.org/TR/xmldsig-core2/