AMI ImageSigningTool 7.01 User Guide

AMI ImageSigningTool 7.01 User Guide

Document Revision 1.11
Jul 16, 2021

American Megatrends International LLC
3095 Satellite Boulevard
Building 800, Suite 425
Duluth, Georgia 30096, U.S.A

Legal

Disclaimer

This publication contains proprietary information which is protected by copyright. No part of this publication may be reproduced, transcribed, stored in a retrieval system, translated into any language or computer language, or transmitted in any form whatsoever without the prior written consent of the publisher, American Megatrends International LLC. American Megatrends International LLC retains the right to update, change, modify this publication at any time, without notice.

For Additional Information

Call American Megatrends International LLC. At 1-800-828-9264 for additional information.

Limitations of Liability

In no event shall American Megatrends be held liable for any loss, expenses, or damages of any kind whatsoever, whether direct, indirect, incidental, or consequential, arising from the design or use of this product or the support materials provided with the product.

Limited Warranty

No warranties are made, either expressed or implied, with regard to the contents of this work, its merchantability, or fitness for a particular use. American Megatrends assumes no responsibility for errors and omissions or for the uses made of the material contained herein or reader decisions based on such use.

Trademark and Copyright Acknowledgments

American Megatrends International LLC
3095 Satellite Boulevard
Building 800, Suite 425
Duluth, Georgia 30096, U.S.A

All product names used in this publication are for identification purposes only and are trademarks of their respective companies.

Table of Contents

Document Information

Purpose

This document is intended to provide information about ImageSigning tool.

Audience

MegaRAC SP-X™ Customers and Customer Support Teams.

Change History

Date	Revision	Description
2020-02-28	1.00	Initial release
2020-04-01	1.01	Updated for 7.00.0002 release
2020-04-19	1.02	SHA384 support added
2020-05-06	1.03	Updated example screenshots with new banner
2020-07-01	1.04	is_spx_image flag added
2020-09-25	1.05	Updated for 7.01 release & Updated Supported OS list
2020-10-30	1.06	add_pub_key_only flag added
2020-12-04	1.07	Updated Supported OS list
2020-12-08	1.08	Updated company address
2021-01-13	1.09	Added RSA-PSS Padding algorithm Support
2021-04-07	1.10	SPL_SUPPORT flag added
2021-07-16	1.11	Added configuration file for SPX 13
2021-07-23	1.12	Added hardware secure boot

ImageSigningTool

Overview

Image signing is the process of encrypting theimage (.ima)file sothat it can be accessedonly through security keys.

ImageSigningTool is used to embed the public keys for secure boot, signed hashed image, backup and restore configuration support and the corresponding encrypted hash to SPX firmware images.

The output of ImageSigningTool is encrypted image (.ima_enc) file

ImageSigningTool Features

- Signing the image with secure bootsupport
- Signing the image with signed hashedsupport
- Signing Tool with HSMsupport
- Signing Image with RSA_PKCS1_PSS_PADDING algorithm

Supported Operating Systems

- Ubuntu14.04
- Ubuntu16.04
- Ubuntu18.04
- Ubuntu20.04

Installation Guide

Installation

1. OpenSSL is pre-requisite forImageSigningTool
2. Download the ImageSigningTool from git.
3. Place and Extract the file in the requiredlocation.
4. This contains**, ImageSigningTool**, spx13_configuration.ini, **configuration.ini , extsign.sh and extspl.sh** files**.**
5. Open terminal and go topath/ImageSigningTool
6. Run ./ImageSigningTool

Configuration Description

```
[GLOBAL]
fwimagepath = rom.ima
output =  encrypted_rom.ima
overwrite_image = no
is_spx_image = yes
pss_padding = no

[SECURE_BOOT_SUPPORT]
publickey = pubkey.key
privatekey = privkey.key
#extended_uboot = yes //Should be enabled only when the firmware supports
backup uboot
#include_uboot = yes   //Should be enabled when UBOOT also need be includedfor
secure boot validation
sign_spl = yes //Should be enabled when ast2600 hardware secure boot
signature mode support
sign_script=extsign.sh


[SIGNED_HASHED_SUPPORT]
raskey1private = privkey.key
raskey1public = pubkey.key
spl_script = extspl.sh

[SIGNED_HASHED_SUPPORT]
privatekey = privkey.key
publickey = pubkey.key
hashtype    = SHA256
add_pub_key_only = no //Should be changed to yes only if public key is to be
added to the image

[BACKUP_AND_RESTORE_KEY]
aeskey = path_to_key
aesiv = path_to_iv
```

GLOBAL Section

fwimage

This section holds the location of the input firmware image.

output

Location of where signed images need to be generated.

overwrite_image

This field is for whether the output image need to be overwrite or not.

is_spx_image

This option specifies whether the input image is spx image or other image during signing. If the flag is not included in config file, then it will sign all the images as default. If the user wants to sign the other type of images rather than the spx images, then the flag is_spx_image should be included in config file and it should be set as "no".

pss_padding

This option will support RSA_PKCS1_PSS_PADDING algorithm when signing the image. To enable RSA-PSS padding algorithm, need to change this flag value from “no” to “yes”.

SECURED BOOT SUPPORT Section

This section contains the vital information for secured boot and extended secured boot support.

publickey

location of the public key in your build machine

privatekey

location of the private key in your build machine

extended_uboot

This option enables extended uboot hash generation. Separate hash for uboot and backup uboot will be calculated and stored in firmware info section. This can be used by the loader modules to validate the integrity of the uboot and backup uboot. This option is used only if the firmware has backup uboot.

include_uboot

This option is to include UBOOT also for secured boot validation. This option is used only when include uboot option is enabled in BMC firmware configuration(PRJ).

Note:

Secured boot support only works with 2048 bit keys.

sign_spl

This option is to include SPL also for secure boot validation.

SPL_SUPPORT Section

rsakey1public

location of the public key in your build machine

rsakey1private

location of the private key in your build machine

spl_script

This option indicates the name of the script file to be used for signing. The default is "extspl.sh".

SIGNED HASH SUPPORT Section

This section contains vital information for appending Signed hash to the input image.

publickey

location of the public key in your build machine

privatekey

location of the private key in your build machine

hash type

This option accepts following algorithms: SHA1, SHA224, SHA256, SHA512 and SHA384.

add_pub_key_only

This option is used if we want to add only public key to image file. If the flag is set to “no” or if the flag is not included in configuration file then signed hash support is carried out. If the flag is set to “yes” then only public key will be added to image. The default value will be “no”.

BACKUP AND RESTORE KEY Section

This section contains vital information for embedding AES KEY and AES IV to the input image.

aeskey

location of the AES key in your build machine

aesiv

location of the AES IV in your build machine

Note:

- If input image is not HPM based image, then “_enc” will be appended to the output file name. This image can be used by Web Interface and YAFUFlash.

- If input image is raw SPX image, then this tool generates additional file which will be ofthesame name mentioned in output parameter.

- This file willnot have encrypted hash appended to it. This image should beusedforfactory programming of BMC SPI.

- If input image is an HPM image, then all the components present in the HPM image willbe signed and stored in the outputfile.

Using ImageSigningTool

Usage

To perform image signing for SPX12

./ImageSigningTool -ini configuration.ini

To perform image signing for SPX13

./ImageSigningTool -ini spx13_configuration.ini

To display this help message

./ImageSigningTool –h

Signing the image (.ima) file

1. Open terminal and go topath/imagesigningtool-master/obj/Linux_x86_64
2. Create privatekey.pem and publickey.pem using openssl

Eg.

openssl genrsa -out privatekey.pem

openssl rsa -in privatekey.pem -pubout > publickey.pem

3. Modify the "configuration.ini" file as showninfigure

configuration.ini

```
[GLOBAL]
fwimagepath = rom.ima
output =  encrypted_rom.ima
overwrite_image = no
pss_padding = no

[SECURE_BOOT_SUPPORT]
publickey = pubkey.key
privatekey = privkey.key
#extended_uboot = yes //Should be enabled only when the firmware supports backup uboot
#include_uboot = yes  //Should be enabled when UBOOT also need be included for secure boot validation

[SIGNED_HASHED_SUPPORT]
privatekey = privkey.key
publickey = pubkey.key
hashtype   = SHA256

[BACKUP_AND_RESTORE_KEY]
aeskey = path_to_key
aesiv = path_to_iv
```

Figure 1. Example for modified configuration.ini file

4. Run the following command for getting signedimage(.ima_enc)
   ./ImageSigningTool -ini configuration.ini

Example Snapshots

./ImageSigningTool –h

```
prakash@prakash-desktop:~$ ./ImageSigningTool -h
Usage:
        ./ImageSigningTool -ini <configuration file>
        ./ImageSigningTool -h
                To display this help message
        ./ImageSigningTool -v
                To display the version number
prakash@prakash-desktop:~$
```

Figure 2. Help contents of ImageSigningTool

openssl genrsa -out privatekey.pem
openssl rsa -in privatekey.pem -pubout > publickey.pem

```
prakash@prakash-desktop:~/imagesigningtool-master/obj/Linux_x86_64$ openssl genrsa -out privatekey.pem
Generating RSA private key, 2048 bit long modulus (2 primes)
..+++++
...+++++
e is 65537 (0x010001)
prakash@prakash-desktop:~/imagesigningtool-master/obj/Linux_x86_64$ openssl rsa -in privatekey.pem -pubout > publickey.pem
writing RSA key
```

Figure 3. Creating privatekey and publickey (.pem) files

./ImageSigningTool -ini configuration.ini

```
prakash@prakash-desktop:~/imagesigningtool-master/obj/Linux_x86_64$ ./ImageSigningTool -ini configuration.ini
Secure Boot Signing successful
signed hashed image update successful
prakash@prakash-desktop:~/imagesigningtool-master/obj/Linux_x86_64$
```

Figure 4. Image Signing

./ImageSigningTool –V

```
prakash@prakash-desktop:~/BMC-UTIL/imagesigningtool/obj/Linux_x86_64$ ./ImageSigningTool -v
+-------------------------------------------------------+
|            Image Signing Tool v7.00.0004              |
| Copyright (c) 2020 American Megatrends International, LLC |
+-------------------------------------------------------+
prakash@prakash-desktop:~/BMC-UTIL/imagesigningtool/obj/Linux_x86_64$
```

Figure 5. ImageSigningTool Version

Hardware Secure Boot

Overview

AST2600 support hardware secure boot in SPX13, the verification key or encryption key will store in OTP of AST2600.Once firmware image signed or encrypt by imagesigningtool with same key. The firmware image will be measurement and decrypted when ast2600 power on by OTP key.Imagesigningtool will generate Rot header,RoT Signature and CoT Signature for rom.ima.

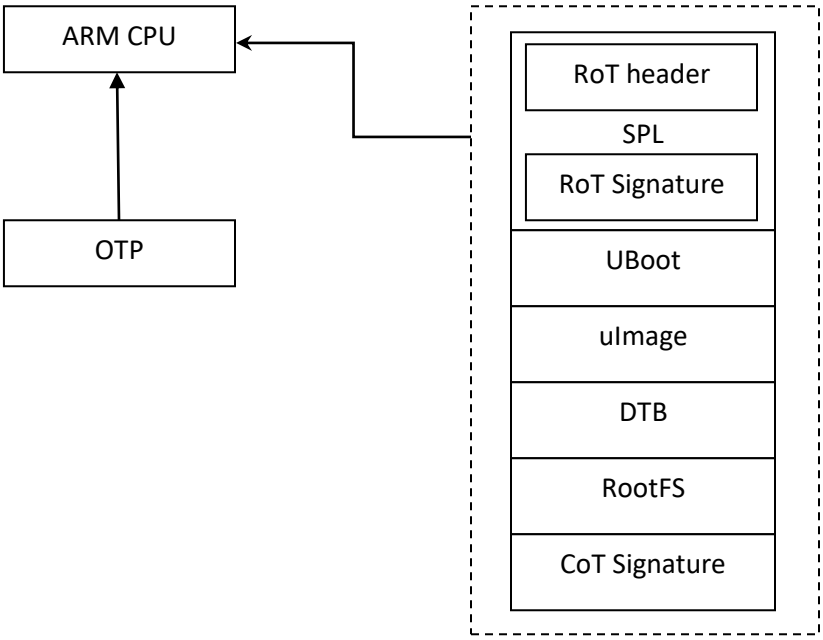

Figure 6. Boot procedure

Configure file example for signature mode

SIG_RSA_KEY1.pem and SIG_RSA_KEY2.pem could be generated by openssl.

And SIG_RSA_KEY1.pem should be same as OTP programed.

[GLOBAL]

fwimagepath = rom.ima

output = encrypted_rom.ima
#if PSS padding is enabled in firmware pss_padding=yes or no when disabled.

pss_padding = no

[SECURE_BOOT_SUPPORT]

publickey = ./key/SIG_RSA_KEY2_public.pem

privatekey = ./key/SIG_RSA_KEY2_private.pem

sign_spl = yes

sign_script=extsign.sh

[SPL_SUPPORT]

rsakey1private = ./key/SIG_RSA_KEY1.pem

rsakey1public = ./key/SIG_RSA_KEY1_pub.pem

spl_script = extspl.sh