Level 3 Security Solution for the HiSilicon Intelligent STB

User Guide

Issue	**00B01**
Date	**2015-07-06**

Trademarks and Permissions

Notice

The purchased products, services and features are stipulated by the contract made between HiSilicon and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

HiSilicon Technologies Co., Ltd.

Address:	Huawei Industrial Base Bantian, Longgang Shenzhen 518129 People's Republic of China
Website:	http://www.hisilicon.com
Email:	support@hisilicon.com

About This Document

Purpose

This document describes how to use the level-3 HiSilicon security solution.

Intended Audience

This document is intended for:

- Technical support engineers
- Software development engineers

Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
⚠ DANGER	Alerts you to a high risk hazard that could, if not avoided, result in serious injury or death.
⚠ WARNING	Alerts you to a medium or low risk hazard that could, if not avoided, result in moderate or minor injury.
⚠ CAUTION	Alerts you to a potentially hazardous situation that could, if not avoided, result in equipment damage, data loss, performance deterioration, or unanticipated results.
🔑 TIP	Provides a tip that may help you solve a problem or save time.
📖 NOTE	Provides additional information to emphasize or supplement important points in the main text.

Change History

Changes between document issues are cumulative. Therefore, the latest document issue contains all changes made in previous issues.

Issue 00B01 (2015-07-06)

This issue is the first draft release.

Contents

Figures

1 Introduction

1.1 Introduction to the Level-3 Security Solution

Compared with the level-2 security solution (see the *Level-2 Security Solution for the HiSilicon Intelligent STB User Guide*), STB vendors do not have the signed keys, so they cannot generate signed fastboot, bootargs, recovery, and kernel partitions during compilation. Vendors need to deliver the unsigned images to carriers for signatures, and then burn these signed images into the chip. The burning method is the same as that described in the level-2 security solution.

NOTE

This document takes Hi3798M V100 as an example. The solutions are similar for the other supported chips.

1.2 Roles in the Level-3 Security Solution

HiSilicon:

- Provides chips and development environments for the STB vendor.
- Provides the carrier with the CASignTool running on Windows in the directory **device/hisilicon/bigfish/sdk/tools/windows/advca/CASignTool**.

Carrier

- Generates two pairs of keys and signs the partition images for the STB vendor by using the signing tool. For details, see section 2.1 "Generating Keys" and 2.2 "Signing the fastboot Image."
- Provides the STB vendor with **root_rsa_pub.bin** and **root_rsa_pub_crc.bin**.

STB Vendor

- Compiles the unsigned advanced conditional access (CA) fastboot and other partition images.
- Provides the unsigned fastboot, bootargs, recovery, and kernel images to the carrier. The carrier signs these partition images and sends them back to the STB vendor.
- Compresses **update.zip** again after receiving the signed images.

2 Guidance for Carriers

2.1 Generating Keys

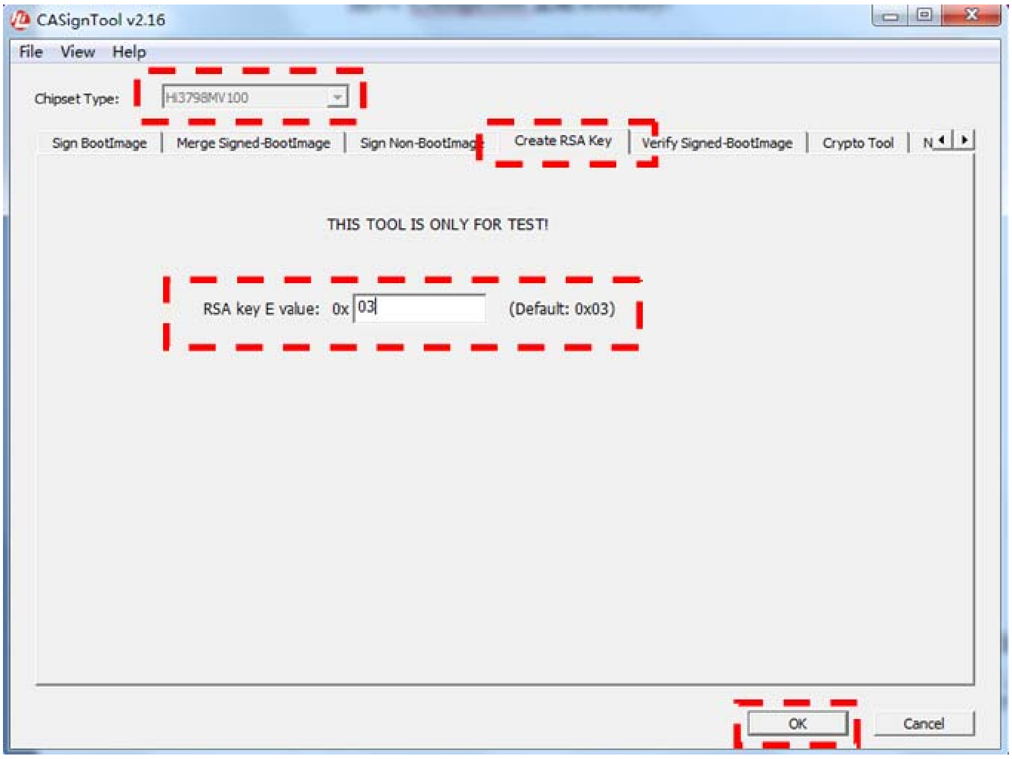
Images related to secure boot must be signed by using keys. There are two pairs of secure boot verification keys:

- One for verifying the fastboot signature
- One for verifying signatures of images except the fastboot image

To generate the two pairs of keys by using the CASignTool, perform the following steps:

Step 1 Start the CASignTool, click the **Create RSA Key** tab, and set **RSA key E value** to the default value **03**, as shown in Figure 2-1.

Figure 2-1 Generating RSA Key



Chipset Type: Hi3798MV100

THIS TOOL IS ONLY FOR TEST!

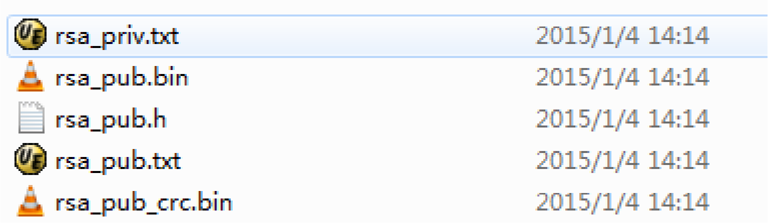
RSA key E value: 0x 03 (Default: 0x03)

CAUTION

The default value of **RSA key E value** is **0x03**, and it is used for generating the RSA key pairs. You can also enter a positive hexadecimal integer (maximum 0xffffffff). However, the default value **0x03** is recommended because the required calculation increases significantly if another value is used.

Step 2 Click **OK**. An **RSA_*XXXXXXX*** directory is generated in the CASignTool directory for storing the generated keys, as shown in Figure 2-2.

Figure 2-2 Generated keys



```
rsa_priv.txt      2015/1/4 14:14
rsa_pub.bin       2015/1/4 14:14
rsa_pub.h         2015/1/4 14:14
rsa_pub.txt       2015/1/4 14:14
rsa_pub_crc.bin   2015/1/4 14:14
```

The first pair of generated keys (**rsa_priv.txt** and **rsa_pub.bin**) and **rsa_pub_crc.bin** are used for verifying the fastboot signature.

- Rename **rsa_priv.txt root_rsa_priv.txt**.
- Rename **rsa_pub.bin root_rsa_pub.bin**.
- Rename **rsa_pub_crc.bin root_rsa_pub_crc.bin**.
- Delete other files that are not required (optional).

Step 3 Click **OK** again to generate another pair of keys for verifying the signatures of the bootargs, recovery, and kernel images.

- Rename **rsa_pub.txt extern_rsa_pub.txt**.
- Rename **rsa_priv.txt extern_rsa_priv.txt**.
- Delete other files that are not required (optional).

Step 4 Save **root_rsa_priv.txt**, **extern_rsa_pub.txt**, **extern_rsa_priv.txt**, **root_rsa_pub.bin**, and **root_rsa_pub_crc.bin** to a secure directory.

----End

CAUTION

- The carrier needs to keep the keys **root_rsa_priv.txt**, **extern_rsa_pub.txt**, and **extern_rsa_priv.txt** for signing the fastboot image.
- The carrier needs to send **root_rsa_pub.bin** and **root_rsa_pub_crc.bin** to the STB vendor.

2.2 Signing the fastboot Image

2.2.1 Environment Preparation

Prepare the following items:

- Key pairs
  - Private root key: **root_rsa_priv.txt**
  - External public key: **external_rsa_pub.txt**
  - External private key: **external_rsa_priv.txt**
- Board configuration file **cfg.bin** (provided by the STB vendor)
- A boot image (provided by the STB vendor)

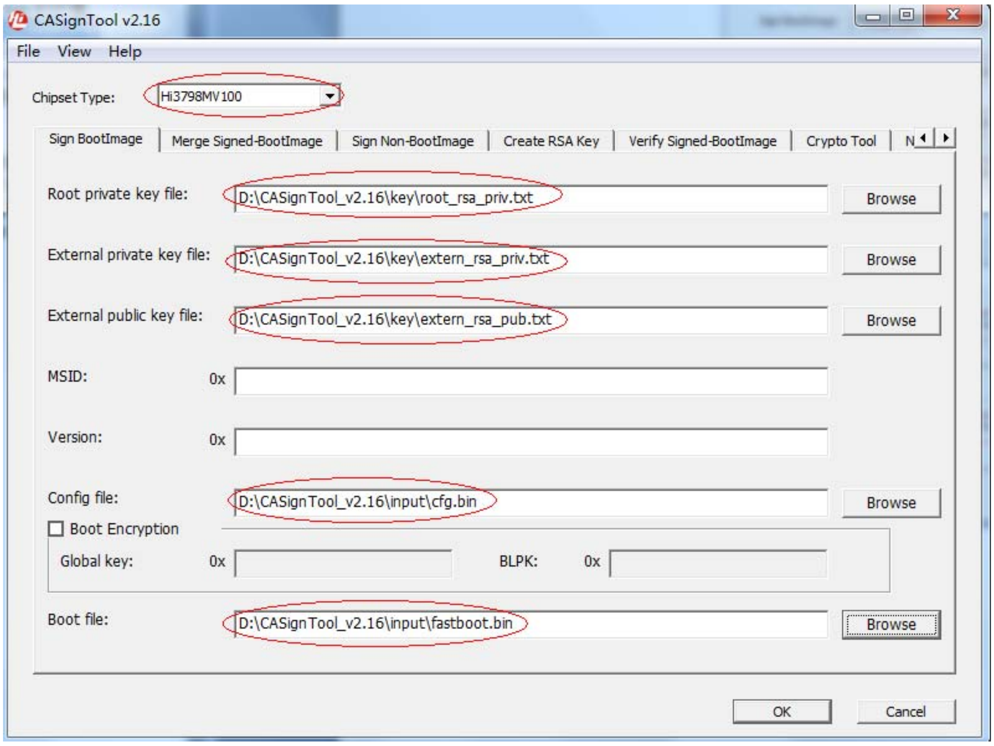
2.2.2 Procedures

To sign a boot image, perform the following steps:

Step 1 Start the CASignTool. The following steps use Hi3798M V100 as an example.

Step 2 Set **Chipset Type** to **Hi3798MV100**, click the **Sign BootImage** tab, and specify **Root private key file**, **External private key file**, **External public key file**, **Config file**, and **Boot file** on the tab page as shown in Figure 2-3.

Figure 2-3 Signing a fastboot image



Chipset Type: Hi3798MV100

Root private key file: D:\CASignTool_v2.16\key\root_rsa_priv.txt

External private key file: D:\CASignTool_v2.16\key\extern_rsa_priv.txt

External public key file: D:\CASignTool_v2.16\key\extern_rsa_pub.txt

MSID: 0x

Version: 0x

Config file: D:\CASignTool_v2.16\input\cfg.bin

☐ Boot Encryption

Global key: 0x

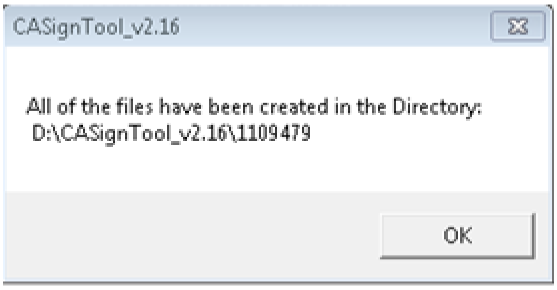
BLPK: 0x

Boot file: D:\CASignTool_v2.16\input\fastboot.bin

Step 3 Click **OK**.

The CASignTool creates a folder in the folder where the CASignTool locates for storing the generated files.

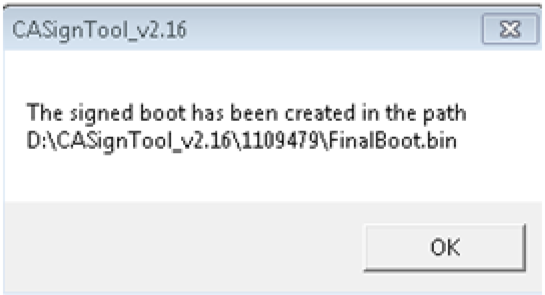
Figure 2-4 Generating files



All of the files have been created in the Directory:
D:\CASignTool_v2.16\1109479

The signed secure boot image **FinalBoot.bin** is generated.

Figure 2-5 Generating FinalBoot.bin



The signed boot has been created in the path
D:\CASignTool_v2.16\1109479\FinalBoot.bin

----End

2.3 Signing the bootargs Partition Image

2.3.1 Environment Preparation

Prepare the following items:

- Key pairs
  - External public key: **external_rsa_pub.txt**
  - External private key: **external_rsa_priv.txt**
- An unsigned **bootargs.bin** image (provided by the STB vendor)

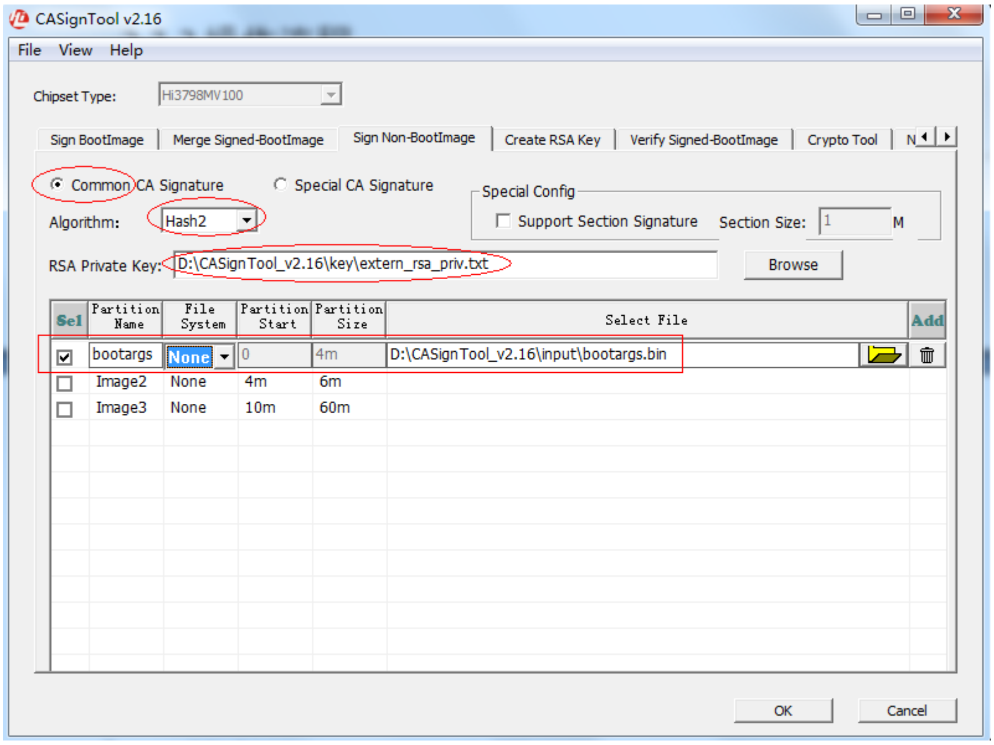
2.3.2 Procedures

To sign a bootargs image, perform the following steps:

Step 1 Start the CASignTool.

Step 2 Click the **Sign Non-BootImage** tab. Select **Common CA Signature**, and specify **Algorithm**, **RSA Private Key**, and the partition information on the tab page as shown in Figure 2-6.

Figure 2-6 Signing a bootargs image



Chipset Type: Hi3798MV100

● Common CA Signature ○ Special CA Signature

Algorithm: Hash2

□ Support Section Signature Section Size: 1 M

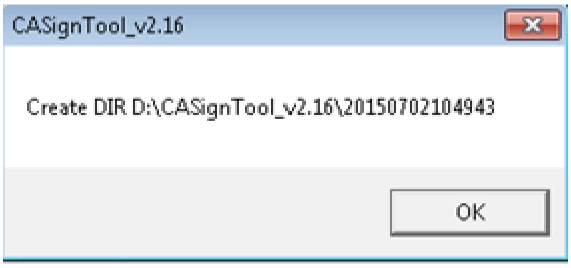
RSA Private Key: D:\CASignTool_v2.16\key\extern_rsa_priv.txt

Sel	Partition Name	File System	Partition Start	Partition Size	Select File	Add
☑	bootargs	None	0	4m	D:\CASignTool_v2.16\input\bootargs.bin	
□	Image2	None	4m	6m		
□	Image3	None	10m	60m		

Step 3 Click **OK**.

The CASignTool creates a folder in the folder where the CASignTool locates for storing the generated files.

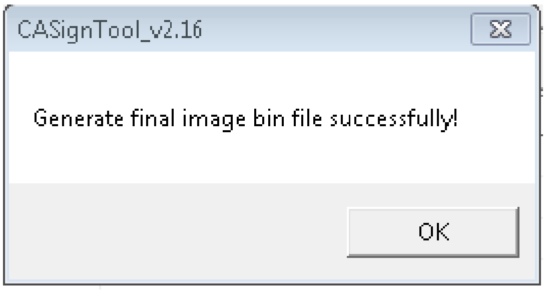
Figure 2-7 Generating the folder



Create DIR D:\CASignTool_v2.16\20150702104943

Step 4 The signed **bootargs_Sign.img** is generated.

Figure 2-8 Generating bootargs_Sign.img



Generate final image bin file successfully!

----End

2.4 Signing the recovery Partition Image

2.4.1 Environment Preparation

Prepare the following items:

- Key pairs
  - External public key: **external_rsa_pub.txt**
  - External private key: **external_rsa_priv.txt**
- An unsigned **recovery.img** (provided by the STB vendor)

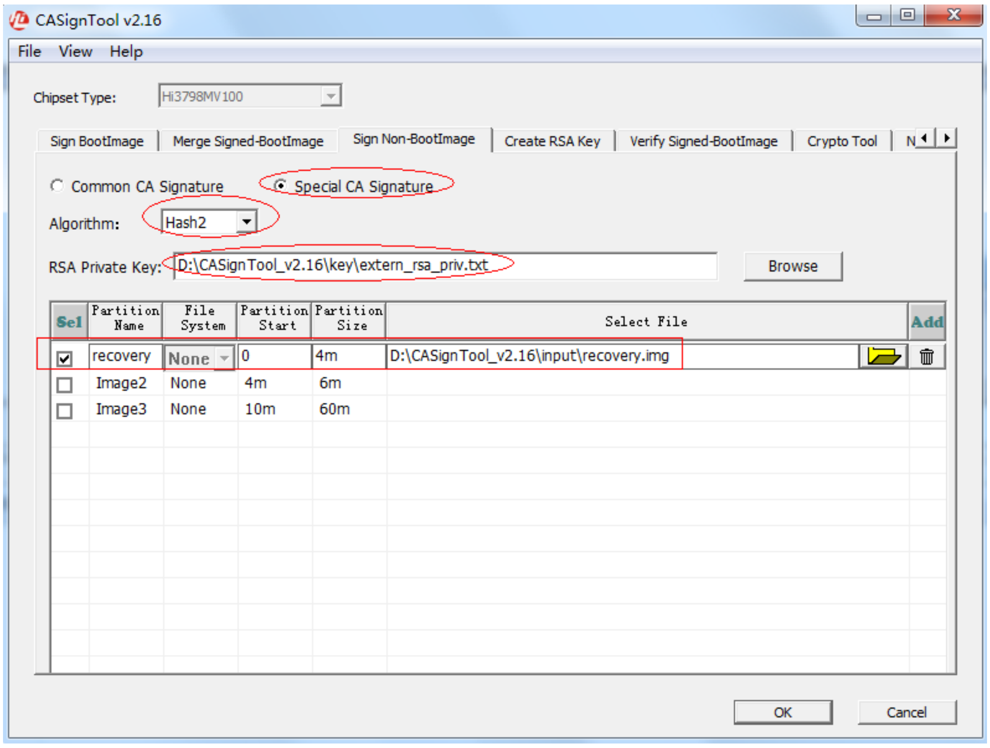
2.4.2 Procedures

To sign a recovery image, perform the following steps:

Step 1 Start the CASignTool.

Step 2 Click the **Sign Non-BootImage** tab. Select **Special CA Signature**, and specify **Algorithm**, **RSA Private Key**, and the partition information on the tab page as shown in Figure 2-9.

Figure 2-9 Singing a recovery image



Chipset Type: Hi3798MV100

○ Common CA Signature ● Special CA Signature

Algorithm: Hash2

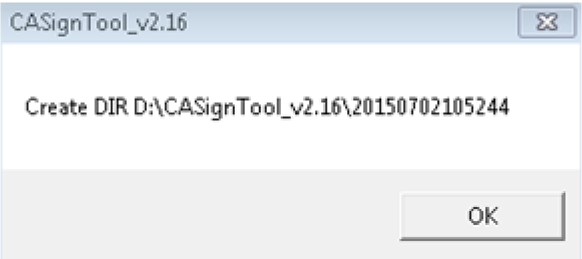
RSA Private Key: D:\CASignTool_v2.16\key\extern_rsa_priv.txt

Sel	Partition Name	File System	Partition Start	Partition Size	Select File	Add
☑	recovery	None	0	4m	D:\CASignTool_v2.16\input\recovery.img	
□	Image2	None	4m	6m		
□	Image3	None	10m	60m		

Step 3 Click **OK**.

The CASignTool creates a folder in the folder where the CASignTool locates for storing the generated files.

Figure 2-10 Generating the folder



CASignTool_v2.16

Create DIR D:\CASignTool_v2.16\20150702105244

The signed recovery image **FinalImage.bin** is generated. Rename **FinalImage.bin** **recovery_Sign.img**.

----End

2.5 Signing the kernel Partition Image

2.5.1 Environment Preparation

Prepare the following items:

- Key pairs
  - External public key: **external_rsa_pub.txt**
  - External private key: **external_rsa_priv.txt**
- An unsigned **kernel.img** (provided by the STB vendor)

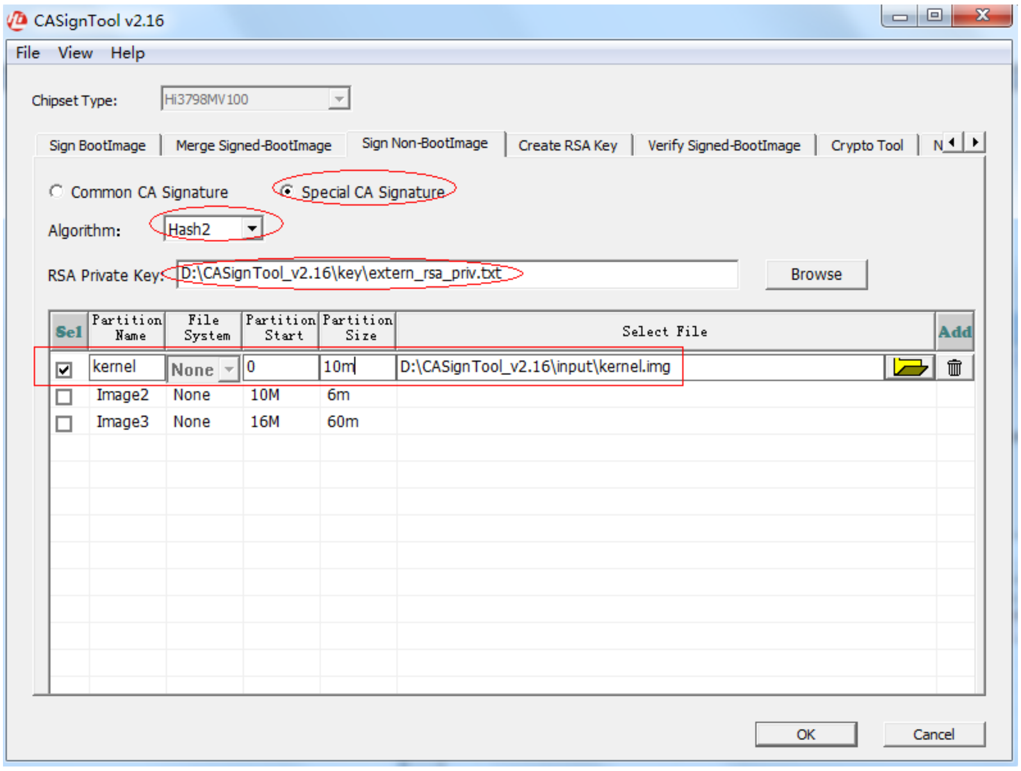
2.5.2 Procedures

To sign a kernel image, perform the following steps:

Step 1 Start the CASignTool.

Step 2 Click the **Sign Non-BootImage** tab. Select **Special CA Signature**, and specify **Algorithm**, **RSA Private Key**, and the partition information on the tab page as shown in Figure 2-11.

Figure 2-11 Signing a kernel image



Chipset Type: Hi3798MV100

○ Common CA Signature ● Special CA Signature

Algorithm: Hash2

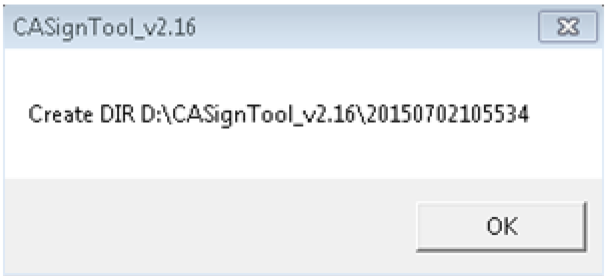
RSA Private Key: D:\CASignTool_v2.16\key\extern_rsa_priv.txt

Sel	Partition Name	File System	Partition Start	Partition Size	Select File
☑	kernel	None	0	10m	D:\CASignTool_v2.16\input\kernel.img
□	Image2	None	10M	6m	
□	Image3	None	16M	60m	

Step 3 Click **OK**.

The CASignTool creates a folder in the folder where the CASignTool locates for storing the generated files.

Figure 2-12 Generating the folder



Create DIR D:\CASignTool_v2.16\20150702105534

The signed kernel image **FinalImage.bin** is generated. Rename **FinalImage.bin kernel_Sign.img**.

----End

After all the images are signed, send the signed partition images to the STB vendor.

3 Guidance for STB Vendors

3.1 Compiling Images

Enable the level-3 security solution in **device/hisilicon/Hi3798MV100/customer.mk** by running the following command:

```
HISILICON_SECURITY_L3 := true
```

After the level-3 security solution is enabled, the unsigned **fastboot.bin** and the other unsigned partition images can be compiled in the following directory:

```
out/target/product/Hi3798MV100/Emmc
```

3.2 Signing Images

- After compiling the unsigned images, the STB vendor provides the following files and information to the carrier for signature:
  - Unsigned **fastboot.bin**, **bootargs.bin**, **recovery.img**, and **kernel.img**
  - Boot cfg file in the directory **device/hisilicon/bigfish/sdk/source/boot/sysreg**
  - Image signature mode:

    Common signature for the bootargs image

    Special signature for the recovery and kernel images
- The carrier signs these partition images and then sends the following files and information to the STB vendor:
  - Signed partition images: **Finalboot.bin**, **bootargs_Sign.img**, **recovery_Sign.img**, and **kernel_Sign.img**
  - One time programmable (OTP) root keys for secure boot: **root_rsa_pub.bin** and **root_rsa_pub_crc.bin**

3.3 Modifying and Generating the update.zip

Replace the unsigned partition images in **update.zip** with the signed **Finalboot.bin**, **bootargs_Sign.img**, **recovery_Sign.img**, and **kernel_Sign.img** to generate a new **update.zip** file by using the HiUpdateEdit in the directory **device/hisilicon/bigfish/sdk/tools/windows/advca/HiUpdateEdit**.

For details about how to use the HiUpdateEdit, see the *HiUpdateEdit User Guide*.

CAUTION

Note the followings when creating a new **update.zip** using the *HiUpdateEdit*:

- Copy all the keys in the source code package **device/hisilicon/Hi3798MV100/security** to the directory **HiUpdateEdit\Config\**.
- Rename **releasekey.pk8** and **releasekey.x509.pem testkey.pk8** and **testkey.x509.pem** respectively.

3.4 Burning Images by Using a USB Flash Drive

3.4.1 Burning Images to a Bare Chip

Prepare the following items before burning:

- Unsigned **fastboot.bin**, **bootargs.bin**, and **recovery.img** as well as the newly generated **update.zip**
- Key from the carrier: **root_rsa_pub_crc.bin**
- USB flash drive with the FAT32 file system

To burn images to a bare chip by using the USB flash drive, perform the following steps:

Step 1 Copy **fastboot.bin**, **bootargs.bin**, **recovery.img**, **update.zip**, and **root_rsa_pub_crc.bin** to the root directory of the USB flash drive.

Step 2 Insert the USB flash drive to the USB 2.0 port.

Step 3 Power on the board. The burning process automatically starts. The indicator blinks during the burning process and is steady on after the burning is complete.

----End

3.4.2 Burning Images to a Non-Bare Chip

Restart the board after images are burnt to a bare chip by using a USB flash drive. The chip is considered a CA chip. To reburn images to a CA chip, you need to adopt the non-bare chip burning solution using the USB flash drive.

Prepare the following items before burning:

- **FinalBoot.bin**, **bootargs_Sign.img**, and **recovery_Sign.img** signed by the carrier as well as the newly generated **update.zip**
- USB flash drive with the FAT32 file system

To burn images to a non-bare chip by using the USB flash drive, perform the following steps:

Step 1 Rename **FinalBoot.bin**, **bootargs_Sign.img**, and **recovery_Sign.img** as **fastboot.bin**, **bootargs.bin**, and **recovery.img** respectively. Copy the renamed files and **update.zip** to the root directory of the USB flash drive.

Step 2 Insert the USB flash drive to the USB 2.0 port.

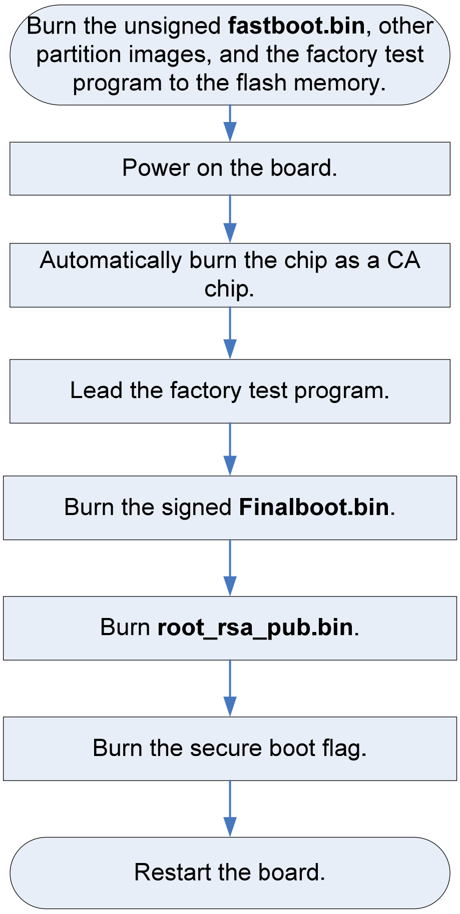
Step 3 Power on the board while holding down the USB burning button to enter the burning process. The indicator blinks during the burning process and is steady on after the burning is complete.

----End

3.5 Burning Images by Using a Burner

Figure 3-1 shows the process of burning images to a bare chip by using a burner.

Figure 3-1 Burning images to a bare chip


Burn the unsigned **fastboot.bin**, other partition images, and the factory test program to the flash memory.
Power on the board.
Automatically burn the chip as a CA chip.
Lead the factory test program.
Burn the signed **Finalboot.bin**.
Burn **root_rsa_pub.bin**.
Burn the secure boot flag.
Restart the board.

To burn images using the burner, perform the following steps:

Step 1 Burn the unsigned **fastboot.bin**, the signed **bootargs_Sign**, **recovery_Sign**, and **kernel_Sign**, and other partitions images using the burner.

Step 2 Power on the board. **fastboot.bin** runs automatically. The board is burnt as a CA chip and enters the factory test program.

Step 3 Burn the **Finalboot.bin** image signed by the carrier from the address with a 512-byte offset relative to the flash initial address using the factory test program.

Step 4 Burn the OTP root key and the secure boot flag by using the factory test program.

----End

3.6 Burning Images by Using the HiTool

To burn images to a bare chip by using the HiTool, perform the following steps:

Step 1 Burn the unsigned **fastboot.bin**.

1. Set the chipset type to **Hi3798MV100**.
2. Select the unsigned **fastboot.bin**.
3. Start burning the image. After the burning is complete, power on the board. The chip is burnt as a CA chip.

Step 2 Burn other images by using the HiTool.

1. Set the chipset type to **Hi3798MV100_CA**.
2. Select the **Finalboot.bin** image signed by the carrier as the programmer file.
3. Select the signed fastboot, bootargs, recovery, and kernel partition images.
4. Select other unsigned partition images.
5. Start burning the images.

Step 3 After the system is booted, burn **root_rsa_pub.bin** and the secure boot flag bit by running the following commands:

```
sample_ca_writeRSAkey /sdcard/root_rsa_pub.bin
sample_ca_opensecboot  emmc
```

----End

To burn images to a non-bare chip by using the HiTool, perform the proceeding steps staring from Step 2.