User Manual Linux AVB

ID: RK-YH-YF-369

Release Version: V2.2.5

Release Date: 2020-06-18

Security Level: □Top-Secret □Secret □Internal ■Public

DISCLAIMER

THIS DOCUMENT IS PROVIDED “AS IS”. ROCKCHIP ELECTRONICS CO., LTD.(“ROCKCHIP”)DOES NOT PROVIDE ANY WARRANTY OF ANY KIND, EXPRESSED, IMPLIED OR OTHERWISE, WITH RESPECT TO THE ACCURACY, RELIABILITY, COMPLETENESS,MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE OR NON-INFRINGEMENT OF ANY REPRESENTATION, INFORMATION AND CONTENT IN THIS DOCUMENT. THIS DOCUMENT IS FOR REFERENCE ONLY. THIS DOCUMENT MAY BE UPDATED OR CHANGED WITHOUT ANY NOTICE AT ANY TIME DUE TO THE UPGRADES OF THE PRODUCT OR ANY OTHER REASONS.

Trademark Statement

"Rockchip", "瑞芯微", "瑞芯" shall be Rockchip’s registered trademarks and owned by Rockchip. All the other trademarks or registered trademarks mentioned in this document shall be owned by their respective owners.

Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian,PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

AVB strives to ensure all executed code next to U-Boot comes from a trusted source, rather than from an attacker or corruption. It establishes a full chain of trust, starting from a hardware-protected root of trust to the bootloader, to the partitions next to U-Boot (such as boot partition and recovery partition).

This document applies to RK3308/RK3326/PX30/RK3399/RK3328/RK3288/RK1808.

Product Version

Chip	Security Storage	Kernel (branch)	U-Boot(commit)	RKBin(commit)
RK3308	OTP	ALL	d3a731e	e00dab6
RK3326	OTP	4.4	d3a731e	20af526
PX30	OTP	4.4	d3a731e	20af526
RK3328	OTP	4.4	d3a731e	158ccc6
RK3399	eFuse	4.4	d3a731e	158ccc6
RK3288	eFuse	4.4	d3a731e	c021945
RK1808	eFuse	4.4	d3a731e	890556f

Intended Audience

This document (this guide) is mainly intended for:

Technical support engineers

Software development engineers

Revision History

Version	Author	Date	Change Description
V2.0.0	Zain Wang	2019-01-28	Support lastest U-Boot Add rk3399 support
V2.1.0	Zain Wang	2019-05-22	Add new command to write AVB keys to eFuse/OTP Add rk3328 support
V2.1.1	Zain Wang	2019-06-03	Fix error description
V2.2.0	Zain Wang	2019-07-13	Add chapter "Notice" Add parameter.txt description Support lastest U-Boot Add chapter "Verified Platform"
V2.2.1	Zain Wang	2019-08-20	Add description for U-Boot CONFIG_RK_AVB_LIBAVB_ENABLE_ATH_UNLOCK
V2.2.2	Zain Wang	2019-09-02	Add platform defconfig description
V2.2.3	Zain Wang	2019-09-10	Add En Version Fixed some descriptions
V2.2.4	Zain Wang	2020-02-06	Fix error description
v2.2.5	Zain Wang	2020-06-18	Transfer PDF

Contents

Notice before reading

1. When the device is in **unlock state**, AVB will still verify the boot.img. AVB will show the error if boot.img is invalid, but the device **boot normally**. When the device is in lock state, AVB will **stop booting** if boot.img is invalid and show the error as well. Therefore, setting the device to unlock state is convenient for debugging.
2. AVB does not support compressing kernel images.
3. Chips used eFuse must enable Base SecureBoot (Rockchip_Developer_Guide_Linux4.4_SecureBoot.pdf **chapter 2 Base SecureBoot**)

AVB Configuration

Trust

Make sure the trust.img has enable secure option.

Take rk3308 as an example.

Enter `rkbin/RKTRUST`, find **RK3308TRUST.ini**(This file selected by make.sh in U-Boot) and change:

```
diff --git a/RKTRUST/RK3308TRUST.ini b/RKTRUST/RK3308TRUST.ini
index 0b2839d..51ec627 100644
--- a/RKTRUST/RK3308TRUST.ini
+++ b/RKTRUST/RK3308TRUST.ini
@@ -8,7 +8,7 @@ SEC=1
 PATH=bin/rk33/rk3308_bl31_v2.21.elf
 ADDR=0x00010000
 [BL32_OPTION]
-SEC=0
+SEC=1
 PATH=bin/rk33/rk3308_bl32_v1.12.bin
 ADDR=0x00200000
 [BL33_OPTION]
```

Secure option is enabled as default for TOS format, like RK3288TOS.ini.

U-Boot

AVB needs these U-Boot configs:

```
# OPTEE support
CONFIG_OPTEE_CLIENT=y
CONFIG_OPTEE_V1=y           # For rk3288/rk3328/rk3399 and NOT work with V2
CONFIG_OPTEE_V2=y           # For rk3308/rk3326/px30/rk1808 and NOT work with V1

# CRYPTO support
CONFIG_DM_CRYPTO=y          # Only For efuse
CONFIG_ROCKCHIP_CRYPTO_V1=y # For efuse chips, like rk3399/rk3288
CONFIG_ROCKCHIP_CRYPTO_V2=y # For efuse chips, like rk1808

# AVB support
CONFIG_AVB_LIBAVB=y
CONFIG_AVB_LIBAVB_AB=y
CONFIG_AVB_LIBAVB_ATX=y
```

```
CONFIG_AVB_LIBAVB_USER=y
CONFIG_RK_AVB_LIBAVB_USER=y
CONFIG_AVB_VBMETA_PUBLIC_KEY_VALIDATE=y
CONFIG_ANDROID_AVB=y
CONFIG_ANDROID_AB=y                             # Not necessary
CONFIG_OPTEE_ALWAYS_USE_SECURITY_PARTITION=y    # Enable it if no RPMB
CONFIG_ROCKCHIP_PRELOADER_PUB_KEY=y             # Only for efuse
CONFIG_RK_AVB_LIBAVB_ENABLE_ATH_UNLOCK=y        # See below

#fastboot support
CONFIG_FASTBOOT=y
CONFIG_FASTBOOT_BUF_ADDR=0x2000000              # Not fixed, change it if necessary
CONFIG_FASTBOOT_BUF_SIZE=0x08000000             # Not fixed, change it if necessary
CONFIG_FASTBOOT_FLASH=y
CONFIG_FASTBOOT_FLASH_MMC_DEV=0
```

`CONFIG_RK_AVB_LIBAVB_ENABLE_ATH_UNLOCK` is newer config for U-Boot, if no this config in U-Boot, try to fix it manually.
It's important to run `rk_auth_unlock()` to ensure unlock operation in valid.
Note: changes base on U-Boot commit 46a8a26905fc68e6683b93c97adae0dd9a4e37ba

```
diff --git a/drivers/usb/gadget/f_fastboot.c b/drivers/usb/gadget/f_fastboot.c
index 99d62a6..fec465e 100755
--- a/drivers/usb/gadget/f_fastboot.c
+++ b/drivers/usb/gadget/f_fastboot.c
@@ -1370,7 +1370,7 @@ static void cb_oem(struct usb_ep *ep, struct usb_request *req)
                        fastboot_tx_write_str("FAILThe vboot is disable!");
                } else {
                        lock_state = 1;
-#ifdef CONFIG_RK_AVB_LIBAVB_ENABLE_ATH_UNLOCK
+#if 1
                        if (rk_auth_unlock((void *)CONFIG_FASTBOOT_BUF_ADDR,
                                           &out_is_trusted)) {
                                printf("rk_auth_unlock ops error!\n");
```

If Ramdisk is required when used A/B Partition, U-Boot has to be modified.
(This change unable to merge to master since conflict with android)

```
diff --git a/common/android_bootloader.c b/common/android_bootloader.c
index 230bf0e..e5c81e6 100644
--- a/common/android_bootloader.c
+++ b/common/android_bootloader.c
@@ -1061,9 +1061,6 @@ int android_bootloader_boot_flow(struct blk_desc *dev_desc,
                 * "skip_initramfs" to the cmdline to make it ignore the
                 * recovery initramfs in the boot partition.
                 */
-#ifdef CONFIG_ANDROID_AB
-               mode_cmdline = "skip_initramfs";
-#endif
                break;
        case ANDROID_BOOT_MODE_RECOVERY:
                /* In recovery mode we still boot the kernel from "boot" but
```

Now, run ./make.sh xxx to generate uboot.img, trust.img and loader.bin

Parameter

Partition `vbmeta` and `system` are required, `security` is optional:

- `vbmeta` use to store the signature information, size 1M, no matter where `vbmeta` is.
- `system` has different name in some unique platform, like `Buildroot` calls `system` to `rootfs`. Rename `rootfs` to `system`, and if used `uuid`, changes `uuid` partition name as well.
- `security` partition is required if storage medium is not eMMC (no RPMB partition), it used to store AVB information instead of RPMB. AVB content is encrypted, the size is 4M and the location is optional.

Here are examples of AVB parameter：

```
# AVB parameter:
0x00002000@0x00004000(uboot),0x00002000@0x00006000(trust),0x00002000@0x00008000(misc),0x00010000@0x0000a000(boot),0x00010000@0x0001a000(recovery),0x00010000@0x0002a000(backup),0x00020000@0x0003a000(oem),0x00300000@0x0005a000(system),0x00000800@0x0035a000(vbmeta),0x00002000@0x0035a800(security),-@0x0035c800(userdata:grow)
uuid:system=614e0000-0000-4b53-8000-1d28000054a9

# AVB and A/B parameter:
0x00002000@0x00004000(uboot),0x00002000@0x00006000(trust),0x00004000@0x00008000(misc),0x00010000@0x0000c000(boot_a),0x00010000@0x0001c000(boot_b),0x00010000@0x0002c000(backup),0x00020000@0x0003c000(oem),0x00300000@0x0005c000(system_a),0x00300000@0x0035c000(system_b),0x00000800@0x0065c000(vbmeta_a),0x00000800@0x0065c800(vbmeta_b),0x00002000@0x0065d000(security),-@0x0065f00(userdata:grow)
```

When downloading, the partition names on the RKDevTool (windows PC）should be modified synchronously. After modification, reload parameter.

AVB Keys

AVB contains the following four Keys:

- Product RootKey (PRK): root Key of AVB, in eFuse devices, related information is verified by Base SecureBoot Key. In OTP devices, PRK-Hash information pre-stored in OTP is directly read and verified;
- ProductIntermediate Key (PIK): intermediate Key;
- ProductSigning Key (PSK): used to sign a firmware;
- ProductUnlock Key (PUK): used to unlock a device.

There is already a set of test certificates and keys in avb_keys directory.

If you need new Keys and Certificates, you can generate them by the following (This operation will remove avb_keys first, so please backup your previous keys properly if necessary.)

```
./avb_user_tool.sh -n <Product ID> #The size of Product ID is 16 bytes.
```

For eFuse devices, you need to generate additional permanent_attributes_cer.bin (OTP devices can skip this step)

```
./avb_user_tool.sh -f < /Path/to/PrivateKey.pem >
```

The `PrivateKey.pem` which is the key of Base SecureBoot is required. This operation will generate .setting configuration file, and set device type to eFuse. If you want to reconfigurate device type, **DO NOT FORGET TO CHANGE THE TYPE**. You can set `efuse` or `otp` in .setting.

Please keep the generated files properly, otherwise you will not be able to unlock after locking, and the machine will not be able to upgrade anymore.

Sign the firmware

AVB can verify the boot.img and recovery.img, sign them with this command:

```
./avb_user_tool.sh -s -b < /path/to/boot.img > -r < /path/to/recovery.img > #
Remove -r option if no recovery.img
```

The signed firmware and vbmeta.img generated in `out` directory.

Now, we get all files of AVB, include:

- AVB uboot.img/trust.img/loader.bin generated from **chapter 2 AVB Configuration**
- Key files in `avb_keys`
- signed firmware and vbmeta in `out`

Pack them together, we can get the `update.img` with AVB. In order to use Base SecureBoot, update.img should be signed with rk_signed_tool (or other signed tools), do not forget to enable **`exclude_boot_sign = True`** in `setting.ini`.

Download firmware

Download firmware with windows tool (RKDevTool), try to add `vbmeta` partition (and `security` partition if necessary) to the tool and blank the address. Then reload parameter.txt, the tool will update new partition address. `security` partition do not need to download any images, it would initialized by U-Boot.

After downloading firmware, the device is unlock state, it can not stop booting invalid image.

AVB lock and unlock

AVB support Lock and Unlock states:

- Lock: verify the image next stage to U-Boot like boot.img and recovery.img, and **stop booting** invalid image, report error if image is invalid
- verify the image next stage to U-Boot like boot.img and recovery.img, report error if image is invalid, **but still boot**.

Therefore, setting the device to unlock state is convenient for debugging.

All of AVB user operations are used fastboot. There are some ways to enter device fastboot mode:

- Press "fastboot" key at bootimg if there is "fastboot" button in board.
- Run `reboot fastboot` in system
- Run `fastboot usb 0` in U-Boot console. (set CONFIG_BOOTDELAY in U-Boot, it support specific delay to wait Ctrl-C to enter U-Boot console)

Anyway, device must communicate to PC with fastboot.

Store user password for fastboot which run with supper user, it helps us run fastboot without input user password manually.

```
./avb_user_tool.sh --su_pswd < /user/password >
```

Download AVB root information (must be done before "Lock" and "Unlock"):

```
./avb_user_tool.sh -d
```

Lock the device:

```
./avb_user_tool.sh -l # reboot device after finishing lock.
```

Unlock the device:

```
./avb_user_tool.sh -u # reboot device after finishing unlock.
```

Verity

If everything goes well, the log below will be shown if the device is LOCKED.

```
ANDROID: reboot reason: "(none)"
Could not find security partition
read_is_device_unlocked() ops returned that device is LOCKED
```