密级状态： 绝密(　　) 秘密(　　) 内部资料(　　) 公开(√)
Security Class：Most Confidential() Confidential() Internal() Public(√)

Rockchip_Introduction_Android9.0_AVB_Howto_CN&EN

（Technical Department, R & D Dept. I）

文件状态： File Status ［√］草稿 Draft ［ ］正在修改 Modifying ［ ］正式发布 Released	**文件标识：** File Identification:	RK-SM-YF-506
	当前版本： Current Version:	1.0.0
	作　　者： Author:	TZB
	完成日期： Completion Date:	2018-12-01
	审　　核： Checked By:	HJC,ZXZ
	审核日期： Audit Date:	2019-12-03

福州瑞芯微电子股份有限公司
Fuzhou Rockchip Electronics Co.,Ltd.

免责声明 Warranty Disclaimer

本文档按“现状”提供，福州瑞芯微电子股份有限公司（“本公司”，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

This document is provided according to "current situation" and Fuzhou Rockchip Electronics Co., Ltd. (“the company”, the same below) is not responsible for providing any express or implied statement or warranty of accuracy, reliability, completeness, marketability, specific purpose or non-infringement of any statement, information and content of this document. This document is intended as a guide only.

Due to product version upgrades or other reasons, this document may be updated or modified from time to time without notice.

商标声明 Brand Statement

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

Rockchip, “瑞芯微”, “瑞芯”, and other Rockchip trademarks are trademarks of Fuzhou Rockchip electronics Co., Ltd., and are owned by Fuzhou Rockchip electronics Co., Ltd.

All other trademarks or registered trademarks mentioned in this document are owned by their respective owners

福州瑞芯微电子股份有限公司

Fuzhou Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园 A 区 18 号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

Fuzhou Rockchip Electronics Co., Ltd.

Address: 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

前言 Preface

概述 Overview

本文档主要介绍怎么在 Rockchip Android Pie 平台开启 AVB 功能，和替换 AVB 的密钥。

This document mainly describes how to enable AVB function and replace AVB secret key on Rockchip Android9.0 Platform.

产品版本 Product Version

芯片名称 Chip Name	内核版本 Kernel Version	Uboot 版本 Uboot Version
RK3128H	4.4	Next-dev
RK3229	4.4	Next-dev
RK3328	4.4	Next-dev

读者对象 Readers

本文档（本指南）主要适用于以下工程师：

This document(or guide) is mainly applicable to below engineers:

- 技术支持工程师 Technical Support Engineers
- 软件开发工程师 Software Engineers

修订记录 Modification Records

日期 Date	版本 Version	作者 Author	修改说明 Remarks
2018-12	1.0	TZB	初始版本 Initial Release

目录

1 AVB

1.1 概述 Overview

AVB 会验证 boot recovery system vendor dtbo 完整性，完整性相关的元数据放在 vbmeta 分区，只有验证了 vbmeta 的签名，AVB 才是有效的。vbmeta 签名验证在 bootloader(U-Boot)中，bootloader 会验证 vbmeta 中的公钥，并用公钥验证 vbmeta 签名。bootloader 的完整性是由 Rockchip 安全启动验证，具体参考 Rockchip secure boot 指南。在安全启动开启后，Boot ROM 作只读代码 BL1，是可信的，BL1 会验签 loader，loader 会验签 uboot 和 trust，采用逐级验签的方式验证加载的每个字节是否篡改。

AVB will verify the integrity of boot, recovery, system, vendor and dtbo, integrity related meta data is located in the partition of vbmeta. Only vbmeta signature has been verified, then AVB is effective. Vbmeta signature verification is in bootloader(U-Boot), bootloader verifies the public key of vbmeta, and then uses the public key to verify vbmeta signature. Bootloader's integrity is verified by Rockchip security boot, for detailed information, please refer to Rockchip secure boot guide. After the security boot is enabled, Boot ROM is used as read only code BLI, it's believable, BLI would verify loader, loader would verify uboot and trust, it takes step by step verification way to verify whether each loaded byte is tampered.

1.2 开发指引 Developer Guide

1.2.1 生成密钥–openssl Generate Secret Key-openssl

使用 openssl 命令行工具生成 pem 格式的 RSA 密钥。命令如下：

Use openssl command line tool to generate RSA secret key with pem format. The command is showed as below:

```
openssl genpkey -algorithm RSA -pkeyopt rsa_keygen_bits:4096 -outform PEM -out
avb_rsa4096.pem
```

1.2.2 抽取公钥–avbtool Extract Public Key-avbtool

avbtool 是 android 提供的工具，在 external/avb/avbtool。

Avbtool is the tool provided by Android, it's located in the directory of "external/avb/avbtool".

xxd 命令可以将一个文件以十六进制的形式显示出来。

xxd command is used to display a file in Hex format.

```
avbtool extract_public_key  --key avb_rsa4096.pem --output avb_root_pub.bin
xxd -i avb_root_pub.bin > avb_root_pub.h
```

1.2.3 替换公钥–U-Boot Replace Public Key-U-Boot

AVB 的公钥验证是在 U-Boot 中，抽取的公钥(avb_root_pub.h)替换 avb_root_pub 数组。

AVB public key verification is in U-Boot., replace avb_root_pub array with the extracted public key(avb_root_pub.h).

Android 默认使用的是 external/avb/test/data/testkey_rsa4096.pem。

Android uses external/avb/test/data/testkey_rsa4096.pem by default.

avb_root_pub 数组保存在 U-Boot: lib/avb/libavb_user/avb_ops_user.c，默认的公钥值是 external/avb/test/data/testkey_rsa4096.pem 抽取出来的公钥。

avb_root_pub array is saved in the file of U-Boot: lib/avb/libavb_user/avb_ops_user.c, the default public key is extracted from the file “external/avb/test/data/testkey_rsa4096.pem”

```
/**
 * Internal builds use testkey_rsa4096.pem
 * OEM should replace this Array with public key used to sign vbmeta.img
 *
 * openssl genpkey -algorithm RSA -pkeyopt rsa_keygen_bits:4096 \
 * -outform PEM -out avb_rsa4096.pem
 * avbtool extract_public_key  --key avb_rsa4096.pem --output avb_root_pub.bin
 * xxd -i avb_root_pub.bin > avb_root_pub.h
 */
static const char avb_root_pub [] = {
       0x00, 0x00, 0x10, 0x00, 0x55, 0xd9, 0x04, 0xad, 0xd8, 0x04, 0xaf, 0xe3,
       0xd3, 0x84, 0x6c, 0x7e, 0x0d, 0x89, 0x3d, 0xc2, 0x8c, 0xd3, 0x12, 0x55,
       0xe9, 0x62, 0xc9, 0xf1, 0x0f, 0x5e, 0xcc, 0x16, 0x72, 0xab, 0x44, 0x7c,
       0x2c, 0x65, 0x4a, 0x94, 0xb5, 0x16, 0x2b, 0x00, 0xbb, 0x06, 0xef, 0x13,
       0x07, 0x53, 0x4c, 0xf9, 0x64, 0xb9, 0x28, 0x7a, 0x1b, 0x84, 0x98, 0x88,
       0xd8, 0x67, 0xa4, 0x23, 0xf9, 0xa7, 0x4b, 0xdc, 0x4a, 0x0f, 0xf7, 0x3a,
       0x18, 0xae, 0x54, 0xa8, 0x15, 0xfe, 0xb0, 0xad, 0xac, 0x35, 0xda, 0x3b,
       0xad, 0x27, 0xbc, 0xaf, 0xe8, 0xd3, 0x2f, 0x37, 0x34, 0xd6, 0x51, 0x2b,
       …
…
…
```

1.2.4 指定密钥-Android Build 指定编译使用的密钥 Specify Secret Key-Android Build specifies the secret key used for compiling

配置 BOARD_AVB_KEY_PATH 和 BOARD_AVB_ALGORITHM 指定密钥和密钥算法。

Configure BOARD_AVB_KEY_PATH and BOARD_AVB_ALGORITHM to specify secret key and secret key algorithm.

```
ifeq ($(BOARD_AVB_ENABLE),true)

BUILT_VBMETAIMAGE_TARGET := $(PRODUCT_OUT)/vbmeta.img
AVB_CHAIN_KEY_DIR := $(TARGET_OUT_INTERMEDIATES)/avb_chain_keys

ifdef BOARD_AVB_KEY_PATH
$(if $(BOARD_AVB_ALGORITHM),,$(error BOARD_AVB_ALGORITHM is not defined))
else
# If key path isn't specified, use the 4096-bit test key.
BOARD_AVB_ALGORITHM := SHA256_RSA4096
BOARD_AVB_KEY_PATH := external/avb/test/data/testkey_rsa4096.pem
Endif
…
…
endif
```

1.2.5 锁定设备-fastboot oem at-lock-vboot Lock the Device-fastboot oem at-lock-vboot

机器通过 reboot fastboot 或者 reboot bootloader 命令可以进入 fastboot 模式。

The device could enter fastboot mode by reboot fastboot or reboot bootloader command.

锁定设备，U-Boot 会验证 AVB 的公钥。

Lock the devie, U-Boot would verify AVB’s public key.

1.2.6 解锁设备-fastboot oem at-unlock-vboot Unlock the Device- fastboot oem at-unlock-vboot

解锁设备，U-Boot 不会验证 AVB 的公钥，一般是调试阶段使用。第一次烧写机器是处于解锁状态。

Unlock the device, U-Boot would not verify AVB's public key, it's usually used during the debugging step. For the first time flashing, the device is under unlocked state.

1.2.7 禁止解锁 -fastboot oem at-disable-unlock-vboot Disable Unlock-fastboot oem at-disable-unlock-vboot

禁止解锁，该命令会锁定设备并且不允许通过解锁命令解锁设备。启用 rk secure boot 和 AVB 后需要在生产阶段禁止解锁设备，防止刷机。

disable unlock command will lock the device and forbid unlocking the device through unlock command. After rk secure boot and AVB is enabled, you need to disable unlocking the device during production phase, to avoid flashing.

1.2.8 回滚保护 Roll-back Protection

回滚保护通过保存在 vbmeta.img 里面的一个整数值 rollback_index，只有大于等于这个整数值的系统软件，才能升级。因为 vbmeta.img 开启 AVB 和锁定设备后，会进行 AVB 的公钥验证和签名验证，这个值受到密码系统保护。

roll-back protection mechanism judges whether a system software can upgrade by an integer value called rollback_index stored in vbmeta.img. Only the system software whose value is equal or larger than rollback_index can upgrade. After vbmeta.img enables AVB and lock the device, AVB public key verification and signature verification is in progress, then this value is protected by password system.

Android 通过定义 BOARD_AVB_ROLLBACK_INDEX 的值来配置 rollback_index，编译后 rollback_index 的值会保存在 vbmeta.img 中，没定义 BOARD_AVB_ROLLBACK_INDEX 默认值是 0。

Android system configures rollback_index by defining the value of BOARD_AVB_ROLLBACK_INDEX, after compiling, the value of rollback_index is saved in vbmeta.img. BOARD_AVB_ROLLBACK_INDEX's default value is 0 if it's not defined,

附录 A AVB 校验失败 AVB Verification Failure

附录 A-1 Vbmeta 校验失败 Vbmeta Verification Failure

公钥验证失败，进入 loader 模式，log 如下：

If the public key verification is failure, the system would enter loader mode, below is the log:

```
U-Boot 2017.09-02223-gebc4f4b-dirty (Dec 13 2018 - 16:38:24 +0800)

Model: Rockchip RK3328 EVB
DRAM:  2 GiB
Relocation Offset is: 7dc04000
Using default environment

rksdmmc@ff500000: 1, rksdmmc@ff520000: 0
Card did not respond to voltage select!
mmc_init: -95, time 9
switch to partitions #0, OK
mmc0(part 0) is current device
Load FDT from recovery part
DTB: rk-kernel.dtb
ANDROID: fdt overlay OK
Using kernel dtb
I2c speed: 100000Hz
PMIC:  RK8050 (on=0x80, off=0x00)
vdd_logic 1100000 uV
vdd_arm 1225000 uV
regulator(RK805_DCDC2) init 1225000 uV
…
ANDROID: reboot reason: "none"
get share memory, arg0=0x0 arg1=0x9e08000 arg2=0x3f8000 arg3=0x1
read_is_device_unlocked() ops returned that device is LOCKED
avb_slot_verify.c:720: ERROR: vbmeta: Public key used to sign data rejected.
Android boot failed, error -1.
AVB boot failed and enter rockusb or fastboot!
RKUSB: LUN 0, dev 0, hwpart 0, sector 0x0, count 0xe90000
```

附录 A-2 Boot Dtbo 校验失败 Boot Dtbo Verification Failure

boot 或者 dtbo 分区的 hash 不一致，校验失败进入 loader 模式，log 如下：

If boot or dtbo partition’s hash value is inconsistent, then verification is failure and the system enters loader mode, below is the log:

```
U-Boot 2017.09-02223-gebc4f4b (Dec 13 2018 - 16:16:43 +0800)

Model: Rockchip RK3328 EVB
DRAM:  2 GiB
Relocation Offset is: 7dc04000
Using default environment

rksdmmc@ff500000: 1, rksdmmc@ff520000: 0
Card did not respond to voltage select!
mmc_init: -95, time 9
switch to partitions #0, OK
mmc0(part 0) is current device
boot mode: None
Load FDT from boot part
DTB: rk-kernel.dtb
```

```
ANDROID: fdt overlay OK
Using kernel dtb
I2c speed: 100000Hz
PMIC:  RK8050 (on=0x80, off=0x00)
vdd_logic 1100000 uV
vdd_arm 1100000 uV
regulator(RK805_DCDC2) init 1225000 uV
…
ANDROID: reboot reason: "(none)"
get share memory, arg0=0x0 arg1=0x9e08000 arg2=0x3f8000 arg3=0x1
read_is_device_unlocked() ops returned that device is LOCKED
get share memory, arg0=0x0 arg1=0x9e08000 arg2=0x3f8000 arg3=0x1
avb_slot_verify.c:343: ERROR: boot: Hash of data does not match digest in
descriptor.
Android boot failed, error -1.
AVB boot failed and enter rockusb or fastboot!
RKUSB: LUN 0, dev 0, hwpart 0, sector 0x0, count 0xe90000
```

附录 A-3　Recovery Dtbo 校验失败 Recovery Dtbo Verification Failure

recovery 或者 dtbo 分区的 hash 不一致，校验失败进入 loader 模式，log 如下：

If recovery or dtbo partition’s hash is inconsistent, then verification is failure and the system enters loader mode, the log is shown as below:

```
U-Boot 2017.09-02223-gebc4f4b (Dec 13 2018 - 16:16:43 +0800)

Model: Rockchip RK3328 EVB
DRAM:  2 GiB
Relocation Offset is: 7dc04000
Using default environment

rksdmmc@ff500000: 1, rksdmmc@ff520000: 0
Card did not respond to voltage select!
mmc_init: -95, time 10
switch to partitions #0, OK
mmc0(part 0) is current device
Load FDT from recovery part
DTB: rk-kernel.dtb
ANDROID: fdt overlay OK
Using kernel dtb
I2c speed: 100000Hz
PMIC:  RK8050 (on=0x80, off=0x00)
vdd_logic 1100000 uV
vdd_arm 1100000 uV
regulator(RK805_DCDC2) init 1225000 uV
…
ANDROID: reboot reason: "recovery"
get share memory, arg0=0x0 arg1=0x9e08000 arg2=0x3f8000 arg3=0x1
read_is_device_unlocked() ops returned that device is LOCKED
get share memory, arg0=0x0 arg1=0x9e08000 arg2=0x3f8000 arg3=0x1
avb_slot_verify.c:343: ERROR: recovery: Hash of data does not match digest in
descriptor.
Android boot failed, error -1.
AVB boot failed and enter rockusb or fastboot!
RKUSB: LUN 0, dev 0, hwpart 0, sector 0x0, count 0xe90000
```

附录 A-4 System 校验失败 System Verification Failure

特别注意的是下划线红色字体部分，verity-avb 驱动做出错后的处理，通过 cmdline 配置 verity-avb 出错后的行为，目前配置的行为是无效 vbmeta 分区（androidboot.vbmeta.invalidate_on_error=yes），是由 u-boot 传递。verity-avb 发现校验失败后，会修改 vbmeta 分区的 header 的 magic，将导致重启 u-boot 校验 vbmeta 分区 magic 失败。

Pay attention to the below part with red font, verity-avb driver takes treating measures after mistakes happen. It configures verity-avb's action after mistakes through cmdline, current configured action is invalid vbmeta partition(androidboot.vbmeta.invalidate_on_error=yes), it's delivered by u-boot. After verify-avb finds that verification is failure, it will modify magic of header in vbmeta partition, which causes vbmeta partition's magic verification failure when u-boot restarts.

kernel cmdline：

```
Kernel command line: androidboot.storagemedia=emmc androidboot.mode=normal
androidboot.dtbo_idx=0 dm="1 vroot none ro 1,0 4127960 verity 1
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0 4096 4096 515995 515995 sha1
b999370688a3a4f794111ffb3790b10eb6888ae6
05d6831be3b39f8cdadf47bef0f4a9e2f2520af8 10 restart_on_corruption
ignore_zero_blocks use_fec_from_device
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0 fec_roots 2 fec_blocks 520060
fec_start 520060" root=/dev/dm-0
androidboot.vbmeta.device=PARTUUID=5b170000-0000-4075-8000-518d0000771b
androidboot.vbmeta.avb_version=1.1 androidboot.vbmeta.device_state=locked
androidboot.vbmeta.hash_alg=sha256 androidboot.vbmeta.size=3456
androidboot.vbmeta.digest=e61d132b9f39298632a30e203716d98f3e7506f6d7814134b696
ea0b2d04b6d7 androidboot.vbmeta.invalidate_on_error=yes
androidboot.veritymode=enforcing androidboot.verifiedbootstate=green
androidboot.slot_suffix= androidboot.serialno=123456789 console=ttyFIQ0
androidboot.baseband=N/A androidboot.selinux=enforcing
androidboot.wificountrycode=US androidboot.veritymode=enforcing
androidboot.hardware=rk30board androidboot.console=ttyFIQ0
firmware_class.path=/vendor/etc/firmware init=/init rootwait ro init=/init
loop.max_part=7 buildvariant=userdebug earlycon=uart8250,mmio32,0xff130000
swiotlb=1 kpti=0
```

挂载 system 时，发现数据 hash 异常。也有可能在运行时候发现异常，system 的校验是运行时校验，读到哪个块哪个块会校验。如果超级块数据 hash 不匹配，挂载的时候就会出错。

When mounting system, data hash exception happens. Or an exception happens during runtime. System's verification is conducted during runtime, the block which is read will be verified. If super block 's data hash is not matched, then mistake happens during mounting.

```
[    1.264641] device-mapper: init: attempting early device configuration.
[    1.265319] device-mapper: init: adding target '0 4127960 verity 1
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0 4096 4096 515995 515995 sha1
b999370688a3a4f794111ffb3790b10eb6888ae6
05d6831be3b39f8cdadf47bef0f4a9e2f2520af8 10 restart_on_corruption
ignore_zero_blocks use_fec_from_device
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0 fec_roots 2 fec_blocks 520060
fec_start 520060'
[    1.265361] device-mapper: table: 252:0: verity: Waiting for device
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0 ...
[    1.366651] device-mapper: table: 252:0: verity: Waiting for device
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0 ...
[    1.366693] devfreq ff300000.gpu: Couldn't update frequency transition
information.
[    1.390332] dwmmc_rockchip ff510000.dwmmc: Successfully tuned phase to 186
```

```
[    1.392975] mmc1: queuing unknown CIS tuple 0x91 (3 bytes)
[    1.393047] mmc1: new ultra high speed SDR104 SDIO card at address 0001
[    1.470011] device-mapper: table: 252:0: verity: Waiting for device
PARTUUID=af01642c-9b84-11e8-9b2a-234eb5e198a0 ...
[    1.550240] dwmmc_rockchip ff520000.dwmmc: Successfully tuned phase to 248
[    1.550375] mmc2: new HS200 MMC card at address 0001
[    1.554680] mmcblk2: mmc2:0001 8GME4R 7.28 GiB
[    1.558531] mmcblk2boot0: mmc2:0001 8GME4R partition 1 4.00 MiB
[    1.562375] mmcblk2boot1: mmc2:0001 8GME4R partition 2 4.00 MiB
[    1.562822] mmcblk2rpmb: mmc2:0001 8GME4R partition 3 512 KiB
[    1.567326]  mmcblk2: p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16
p17 p18 p19 p20 p21
[    1.581265] device-mapper: init: dm-0 is ready
[    1.587733] device-mapper: verity-fec: 179:15: FEC: recursion too deep
[    1.587786] device-mapper: verity: 179:15: metadata block 515995 is corrupted
[    1.587849] device-mapper: verity-avb: AVB error handler called for
PARTUUID=5b170000-0000-4075-8000-518d0000771b
[    1.587911] device-mapper: verity-avb: invalidate vbmeta: acting on device
179:10
[    1.588261] device-mapper: verity-avb: invalidate vbmeta: found vbmeta
partition
[    1.588523] device-mapper: verity-avb: invalidate vbmeta: completed.
[    1.588602] cpu0 limit freq=816000 min=816000 max=816000
[    1.598171] rk-vcodec ff360000.rkvdec: shutdown
[    1.598587] rk-vcodec vpu_combo: shutdown
[    1.599058] reboot: Restarting system with command 'dm-verity device corrupted
```

重启后 u-boot 发现 vbmeta magic 错误。启动失败，进入 loader。需要重新烧写 vbmeta.img 和对应的 system.img。

When u-boot restarts, it finds vbmeta magic error, then the system boots failure and enters loader mode. You need to reflash vbmeta.img and corresponding system.img.

```
U-Boot 2017.09-02223-gebc4f4b (Dec 13 2018 - 16:44:36 +0800)

Model: Rockchip RK3328 EVB
DRAM:  2 GiB
Relocation Offset is: 7dc04000
Using default environment

rksdmmc@ff500000: 1, rksdmmc@ff520000: 0
Card did not respond to voltage select!
mmc_init: -95, time 9
switch to partitions #0, OK
mmc0(part 0) is current device
boot mode: normal
Load FDT from boot part
DTB: rk-kernel.dtb
ANDROID: fdt overlay OK
Using kernel dtb
I2c speed: 100000Hz
PMIC:  RK8050 (on=0x80, off=0x01)
vdd_logic 1050000 uV
vdd_arm 1000000 uV
regulator(RK805_DCDC2) init 1225000 uV
…
ANDROID: reboot reason: "(none)"
get share memory, arg0=0x0 arg1=0x9e08000 arg2=0x3f8000 arg3=0x1
read_is_device_unlocked() ops returned that device is LOCKED
avb_vbmeta_image.c:59: ERROR: Magic is incorrect.
avb_slot_verify.c:648: ERROR: vbmeta: Error verifying vbmeta image: invalid vbmeta
header
Android boot failed, error -1.
```

```
AVB boot failed and enter rockusb or fastboot!
```

附录 A-5　Vendor 校验失败 Vendor Verification Failure

特别注意的是下划线红色字体部分，verity-avb 驱动做出错后的处理，通过 cmdline 配置 verity-avb 出错后的行为，目前配置的行为是无效 vbmeta 分区（androidboot.vbmeta.invalidate_on_error=yes），是由 u-boot 传递。verity-avb 发现校验失败后，会修改 vbmeta 分区的 header 的 magic，将导致重启 u-boot 校验 vbmeta 分区 magic 失败。

Pay attention to the below part with red font, verity-avb driver takes treating measures after mistakes happen. It configures verity-avb's action after mistakes through cmdline, current configured action is invalid vbmeta partition(androidboot.vbmeta.invalidate_on_error=yes), it's delivered by u-boot. After verify-avb finds that verification is failure, it will modify magic of header in vbmeta partition, which causes vbmeta partition's magic verification failure when u-boot restarts.

vendor 的挂载在 init 和 fs_mgr 中，解析 Android DT 后，通过系统调用配置 dm-verity 驱动。

Vendor's mounting is in init and fs_mgr, after Android DT is parsed, dm-verity driver is configured through system call.

挂载 vendor 时，发现数据 hash 异常。也有可能在运行时候发现异常，vendor 的校验是运行时校验，读到哪个块哪个块会校验。如果超级块数据 hash 不匹配，挂载的时候就会出错。

When mounting vendor, data hash exception happens. Or an exception happens during runtime. vendor's verification is conducted during runtime, the block which is read will be verified. If super block 's data hash is not matched, then mistake happens during mounting.

```
 [    1.655644] init: init first stage started!
[    1.657385] init: Using Android DT directory
/proc/device-tree/firmware/android/
[    1.703971] vendor storage:20160801 ret = 0
[    1.753616] init: [libfs_mgr]Returning avb_handle with status: 0
[    1.754699] init: [libfs_mgr]Loading verity table: '1 /dev/block/by-name/vendor
/dev/block/by-name/vendor 4096 4096 48356 48356 sha1
6406552bda00f29928661270fecad3d6af86d84f
aa51b0021a6bc0c624d7f3559a465efb0ac91722 10 use_fec_from_device
/dev/block/by-name/vendor fec_roots 2 fec_blocks 48738 fec_start 48738
restart_on_corruption ignore_zero_blocks'
[    1.760885] device-mapper: verity-fec: 179:17: FEC: recursion too deep
[    1.760914] device-mapper: verity: 179:17: metadata block 48356 is corrupted
[    1.760958] device-mapper: verity-avb: AVB error handler called for
PARTUUID=5b170000-0000-4075-8000-518d0000771b
[    1.760988] device-mapper: verity-avb: invalidate_vbmeta: acting on device
179:10
[    1.761262] device-mapper: verity-avb: invalidate_vbmeta: found vbmeta
partition
[    1.761415] device-mapper: verity-avb: invalidate_vbmeta: completed.
[    1.761453] cpu0 limit freq=816000 min=816000 max=816000
[    1.775846] rk-vcodec ff360000.rkvdec: shutdown
[    1.776276] rk-vcodec vpu_combo: shutdown
[    1.776724] reboot: Restarting system with command 'dm-verity device corrupted'
```

重启后 u-boot 发现 vbmeta magic 错误。启动失败，进入 loader。需要重新烧写 vbmeta.img 和对应的 vendor.img。

When u-boot restarts, it finds vbmeta magic error, then the system boots failure and enters loader mode. You need to reflash vbmeta.img and corresponding vendor.img.

```
U-Boot 2017.09-02223-gebc4f4b (Dec 13 2018 - 16:44:36 +0800)

Model: Rockchip RK3328 EVB
DRAM:  2 GiB
```

```
Relocation Offset is: 7dc04000
Using default environment

rksdmmc@ff500000: 1, rksdmmc@ff520000: 0
Card did not respond to voltage select!
mmc_init: -95, time 9
switch to partitions #0, OK
mmc0(part 0) is current device
boot mode: normal
Load FDT from boot part
DTB: rk-kernel.dtb
ANDROID: fdt overlay OK
Using kernel dtb
I2c speed: 100000Hz
PMIC:  RK8050 (on=0x80, off=0x01)
vdd_logic 1050000 uV
vdd_arm 1000000 uV
regulator(RK805_DCDC2) init 1225000 uV
…
ANDROID: reboot reason: "(none)"
get share memory, arg0=0x0 arg1=0x9e08000 arg2=0x3f8000 arg3=0x1
read_is_device_unlocked() ops returned that device is LOCKED
avb_vbmeta_image.c:59: ERROR: Magic is incorrect.
avb_slot_verify.c:648: ERROR: vbmeta: Error verifying vbmeta image: invalid vbmeta
header
Android boot failed, error -1.
AVB boot failed and enter rockusb or fastboot!
```

附录 B RK Secure Boot 校验失败 RK Secure Boot Verification Failure

附录 B-1 Uboot 校验失败 Uboot Verification Failure

uboot 校验失败后，无法启动，log 如下

if uboot verification is failure, then the system is unable to boot, the log is shown below:

```
DDR3
333MHz
Bus Width=32 Col=10 Bank=8 Row=15/15 CS=2 Die Bus-Width=16 Size=2048MB
ddrconfig:6
OUT
Boot1 Release Time: Sep  7 2018 15:49:55, version: 2.49
ChipType = 0x11, 259
mmc2:cmd19,100
SdmmcInit=2 0
BootCapSize=2000
UserCapSize=7456MB
FwPartOffset=2000 , 2000
SdmmcInit=0 NOT PRESENT
StorageInit ok = 18945
Raw SecureMode = 1
SecureInit read PBA: 0x4
SecureInit ret = 0, SecureMode = 1
GPT part:  0, name:          uboot, start:0x4000, size:0x2000
GPT part:  1, name:          trust, start:0x6000, size:0x2000
…
…
…
find partition:uboot OK. first_lba:0x4000.
find partition:trust OK. first_lba:0x6000.
LoadTrust Addr:0x6000
No find bl30.bin
Load uboot, ReadLba = 4000
Load OK, addr=0x200000, size=0xd76fc
SecureVerify
...SecureVerify error : rsahash[31] != datahash[0]!!!
Code check error -1
```

附录 B-2 Trust 校验失败 Trust Verification Failure

trust 校验失败后，系统无法启动，log 如下：

if trust verification is failure, then the system is unable to boot, the log is shown below:

```
DDR version 1.13 20180428
ID:0x805 Y
In
SRX
DDR3
333MHz
Bus Width=32 Col=10 Bank=8 Row=15/15 CS=2 Die Bus-Width=16 Size=2048MB
ddrconfig:6
OUT
Boot1 Release Time: Sep  7 2018 15:49:55, version: 2.49
ChipType = 0x11, 260
```

```
mmc2:cmd19,100
SdmmcInit=2 0
BootCapSize=2000
UserCapSize=7456MB
FwPartOffset=2000 , 2000
SdmmcInit=0 NOT PRESENT
StorageInit ok = 18985
Raw SecureMode = 1
SecureInit read PBA: 0x4
SecureInit ret = 0, SecureMode = 1
GPT part:  0, name:          uboot, start:0x4000, size:0x2000
GPT part:  1, name:          trust, start:0x6000, size:0x2000
…
…
…
find partition:uboot OK. first_lba:0x4000.
find partition:trust OK. first_lba:0x6000.
LoadTrust Addr:0x6000
Trust Verify error! flag:0x23
LoadTrust Addr:0x6400
LoadTrust Addr:0x6800
LoadTrust Addr:0x6c00
LoadTrust Addr:0x7000
Trust Verify error! flag:0x23
```

附录 B-3　Loader 校验失败 Loader Verification Failure

loader 的签名如果验证失败，在烧写 loader 阶段无法升级成功，没有任何 log。

if loader signature verification is failure, then the system is unable to upgrade successfully during loader flashing period, there is not any log.