

Rockchip Android GMS用户配置指南

<div>文件状态: <input type="checkbox"/> 草稿 <input checked="" type="checkbox"/> 正式发布 <input type="checkbox"/> 正在修改</div>	文件标识:	RK-YH-YF-237
	当前版本:	V1.7
	作者:	卞金晨
	完成日期:	2022-04-01
	审核:	金华君
	审核日期:	2022-04-01

版本号	作者	修改日期	修改说明	备注
V1.0	卞金晨	2020-02-17	发布初稿	
V1.1	卞金晨	2020-06-30	添加user版本相关说明	
V1.2	卞金晨	2021-02-22	添加对Android 11的支持	
V1.3	卞金晨	2021-03-29	添加对测试时机器锁定状态的描述	
v1.4	蔡建清	2022-02-18	补充attestation key的相关说明	
v1.5	卞金晨	2022-03-28	添加对Android 12的支持	
v1.6	卞金晨	2022-03-30	添加对Android 12 incremental fs的说明 更换固件、Apex签名的说明 添加目录	
v1.7	卞金晨	2022-04-01	添加对adb/fastboot的版本说明 更正一些Android 12的细节	

文档问题反馈：kenjc.bian@rock-chips.com

目录

目录

1. 名词解释
2. 测前需知
3. GMS包的获取
4. GMS所有测试项
5. GMS SDK参数设置
 - 5.1 内核编译
 - 5.2 Widevine
 - 5.3 EEA
 - 5.4 GMS包使能
 - 5.5 Keymaster & Optee
 - 5.6 Hardware Features
 - 5.7 FRP (Factory Reset Protection)
 - 5.8 AVB (Android Verified Boot), A/B升级
 - 5.9 SELinux
 - 5.10 固定fingerprint (可选)
 - 5.11 安全补丁等级 Security Patch level
 - 5.12 Attestation key (Keybox)
 - 5.13 有关Android 12的内核版本使用说明
 - 5.14 有关incrementalfs的说明
 - 5.15 固件签名的生成、替换方法
 - 5.16 Apex签名的生成、替换方法
 - 5.17 检查脚本
 - 5.18 送测前需要检查的部分
6. 测前准备
7. 主机配置
8. 主机环境常见问题及注意事项
9. 多台机器协作测试
10. GSI烧写的方法
11. 测试VTS/CTS-ON-GSI
12. VTS
13. CTS-ON-GSI
14. CTS
15. CTS-Verifier
16. GTS
17. STS
18. GOATS (Performance Test)
19. APTS
20. BTS
21. 其他测试命令
22. GMS Express
 - 22.1 注意事项
 - 22.2 工程下载及配置
23. GTS本地媒体包测试
24. 开机向导Wi-Fi设置概率无法skip
25. 通过认证后, 无法显示已认证问题

1. 名词解释

- GMS (Google Mobile Service)
Google移动设备服务, 包含Google服务框架, Play商店, Chrome浏览器等一系列应用。

- 3PL
第三方实验室，它会根据Google CDD要求以及GMS Requirments，协助Google在全球不同区域
的GMS配置要求对产品进行认证。常见的有富士康/哈曼/风河/和硕等，一般来说自行联系，也
可以找业务引荐。
- Google Partner/MADA (Mobile Application Distribution Agreement)
签署MADA协议成为Google partner后，才能够访问Google Partner文档，才能够联系3PL进行
GMS的申请及测试。
- EEA
卖往欧盟国家的设备需要通过EEA的GMS认证，代码中配置的EEA类型则需要针对不同的情
况和签订的协议EMADA (European Mobile Application Distribution Agreement)进行配置。有关
EEA的相关说明，请参考Google Partner文档：
https://support.google.com/androidpartners_gms/answer/9071728?hl=en
- xTS
执行对应目录下的binary后，会进入xTS-tradefed命令行，此后执行下列的测试命令或retry命令
即可。
具体可参考Google官方网站：<https://source.android.com/compatibility/cts/run>
- retry
指在已有结果的基础上继续测试，进入tradefed后，执行 `l r` 即可查看之前的测试结果，第一
列为 `session id`，执行retry时接参数 `--retry session_id` 即可续测指定的结果。
- 单测
指单独测某些测试项，用于debug，测试后会产生测试结果及log，分别位于 `results` 及 `logs`
目录下。

2. 测前需知

想要在设备搭载Google移动服务，必须通过GMS认证，通过认证后能够享受Google的移动服务，设备的安全性及兼容性也能得到极大的保证。想要通过GMS认证，必须拥有资质，一般来说有2种方式能够通过GMS认证，使设备最终能够搭载GMS服务：

- 自己拥有MADA资质
这种情况下，会专门有一个Google的联系人，有关Partner的文档权限问题以及各种账号问题都能够咨询他，想要通过GMS认证时需要联系Google联系人或是联系3PL来申请。通过本文档的详细配置和自测通过所有xTS测试后，就能够搭载GMS服务了。
- 自己没有MADA资质
只有具有MADA资质才能申请并通过GMS认证。没有MADA资质时，请先联系具有ODM MADA资质的公司，可以自行联系或通过业务同事引荐。

2.1 注意事项

- 文档中绝大部分的外部链接，都需要MADA权限才可以访问。
- 目前2G及以下容量的DDR，均只能以Android Go过认证。体现在编译时选择 `lunch *go`，例如：

```
lunch rk3326_rgo-user
```

- 关于PX30/RK3568过GMS认证

PX30为RK3326的工业版本，功能大致相同，且Google备案处仅有RK3326，所以如果想要用PX30去过认证，请使用RK3326；RK3568同属RK356x系列，请使用RK3566。已PX30为例：

1. 向实验室提交sub-license申请时，与RK3326保持一致。
2. 编译请直接使用RK3326，例如PX30-2GB DDR版本，`lunch rk3326_rgo-user`
3. 如需使用一键编译(例如：`./build.sh -UCA`)，请先修改makefile中的

```
PRODUCT_UBOOT_CONFIG
```

```
device/rockchip/rk3326/BoardConfig.mk
```

```
PRODUCT_UBOOT_CONFIG := px30
```

3. GMS包的获取

1. 拥有MADA资质的客户，建议直接使用我们的Express分支，同步下来的GMS包是符合GMS Express Plus规定的，无需做太多修改。同步方法请移步至文档结尾处的GMS Express章节。
2. 没有MADA资质或想要自己下载GMS包的客户，请按以下步骤操作：

- 常规GMS包请到以下网址直接下载，解压后放到 `vendor/partner_gms`：

```
https://docs.partner.android.com/gms/building/integrating/gms-download
```

- mainline包请到网盘下载, Android 11及以上版本，没有打包好的压缩包，只有说明文档，需要自己使用repo进行同步，网盘/说明文档地址如下：

```
https://drive.google.com/drive/folders/1gxxMalTC99Y2SvyRDUfiDFvY\_k4Ozj05
```

例如：

```
# 下载manifest文件
repo init -u https://partner-
android.googleusercontent.com/platform/manifest -b
r-aml-prebuilt-release
# 开始同步，同步前，请先检查.repo/manifests/default.xml中是否有内容
repo sync -c -j8
```

下载指定版本（一般测试都需要使用最新的Approved版本），查看当前最新approved的文档的 `Release Summary`，找出 `git tag`，例如：

```
git tag mainline_m_2020_dec_preload_5
```

此时，需要手动将manifest中的默认同步分支修改为对应的tag：

```
diff --git a/default.xml b/default.xml
index 7belf37..01ccba9 100644
--- a/default.xml
+++ b/default.xml
@@ -4,7 +4,7 @@
     <remote name="ohd"
         fetch=".."
         review="https://partner-android-review.googleusercontent.com/"
     />

- <default revision="r-aml-prebuilt-release"
```

```
+ <default revision="refs/tags/mainline_m_2020_dec_preload_5"
    remote="ohd" />
```

```
<!-- Google-signed Mainline prebuilt module projects !-->
```

- Android 12及以上版本还需要prebuilts包，请务必下载和mainline包相同的版本，否则编译会报错。例如：

```
vendor/partner_modules
T1003604/mainline_m_2022_feb_preload_5

prebuilts/module_sdk
T1003604/mainline_m_2022_feb_preload_5
```

编译报错示例：

```
ERROR: Hidden API flags are inconsistent:
< prebuilts/module_sdk/IPsec/current/hiddenapi/all-flags.csv
> out/soong/hiddenapi/hiddenapi-flags.csv

ERROR: Hidden API flags are inconsistent:
< prebuilts/module_sdk/Media/current/hiddenapi/stub-flags.csv
> out/soong/hiddenapi/hiddenapi-stub-flags.txt

ERROR: Hidden API flags are inconsistent:
< prebuilts/module_sdk/Media/current/hiddenapi/stub-flags.csv
> out/soong/hiddenapi/hiddenapi-stub-flags.txt

ERROR: Hidden API flags are inconsistent:
< prebuilts/module_sdk/Connectivity/current/hiddenapi/all-flags.csv
> out/soong/hiddenapi/hiddenapi-flags.csv

ERROR: Hidden API flags are inconsistent:
< prebuilts/module_sdk/Media/current/hiddenapi/all-flags.csv
> out/soong/hiddenapi/hiddenapi-flags.csv

ERROR: Hidden API flags are inconsistent:
< prebuilts/module_sdk/Wifi/current/hiddenapi/stub-flags.csv
> out/soong/hiddenapi/hiddenapi-stub-flags.txt
```

4. GMS所有测试项

1. CTS
2. CTS verifier
3. GTS
4. VTS
5. CTS (ReferencePlan, CTS-ON-GSI)
6. STS
7. 性能测试 Performance Test (仅Go版本需要)
8. 性能测试 APTS (仅Android 12及以上版本Go需要)

5. GMS SDK参数设置

5.1 内核编译

在编译kernel时，请确认使用Clang进行编译。如果是编译Android Go版本，请使用android-10-go.config代替android-10.config，Android 11版本请把10更换为11，例如：

```
make ARCH=arm64 rockchip_defconfig android-10-go.config rk3326.config
make ARCH=arm64 rockchip_defconfig android-11-go.config rk3326.config
```

如果提示rkxxxx.config不存在，可以不加。

以下部分涉及Android编译，注意到device目录中检查device/rockchip/rkxxxx/BoardConfig.mk的配置

5.2 Widevine

注意只有RK3566及RK3399已经支持Widevine L1，而平板设备一般L3即可满足要求。如果需要使L1，请与我司联系获取最新L1支持情况以及L1补丁。

```
BOARD_WIDEVINE_OEMCRYPTO_LEVEL := 3
```

5.3 EEA

针对欧盟出售的设备，只有做EEA的设备才需要配置，其他设备请留空

```
BUILD_WITH_EEA := true
BUILD_WITH_EEA_TYPE := type1 (根据自己的EEA类型配置)
```

5.4 GMS包使能

```
BUILD_WITH_GOOGLE_MARKET := true
# 含radio的设备请配置以下配置，其余设备保持false
BUILD_WITH_GOOGLE_MARKET_ALL := true
# 编译user固件时，请把下面的宏设成false，RK的压力测试工具将会移除，否则无法正常启动。
PRODUCT_HAVE_RKAPPS := false
```

如果需要使用Android Go的2G版本，请

device/rockchip/common/device.mk中添加这个修改：

```
diff --git a/device.mk b/device.mk
index 63bb4ac..4340170 100644
--- a/device.mk
+++ b/device.mk
@@ -899,7 +899,7 @@ ifeq ($(strip $(BUILD_WITH_GOOGLE_MARKET)), true)
    OVERRIDE_TARGET_FLATTEN_APEX := true
```

```

PRODUCT_PROPERTY_OVERRIDES += ro.apex.updatable=false

# 2G A Go
- #TMP_GMS_VAR := $(TMP_GMS_VAR)_2gb
+ TMP_GMS_VAR := $(TMP_GMS_VAR)_2gb
endif
ifeq ($(strip $(BUILD_WITH_EEA)),true)
    BUILD_WITH_GOOGLE_MARKET_ALL := true

```

5.5 Keymaster & Optee

申请Sublicense时，填写Optee等级时，要注意只有RK3326/RK356x/RK3588平台填写Optee V2，其他平台均为V1。

```
PRODUCT_HAVE_OPTEE := true
```

5.6 Hardware Features

请仔细和硬件及驱动同事确认设备所支持的硬件，如果不支持（如陀螺仪，BLE等），软件上一定要移除。注意：**Android 11及以上版本要求Nearby Share，该功能依赖BLE，设计硬件时务必注意！**可以通过以下的宏控制，没有宏的，请到frameworks/native/data/etc/下将对应的feature.xml删除：

```

BOARD_GRAVITY_SENSOR_SUPPORT := true
BOARD_COMPASS_SENSOR_SUPPORT := true

```

例如，删除BLE:

```
BOARD_BLUETOOTH_LE_SUPPORT := false
```

5.7 FRP (Factory Reset Protection)

GMS需要启用FRP。有关此功能的详细介绍，请参阅随附的SDK文档：

Rockchip_Introduction_Android_Factory_Reset_Protection_CN&EN.pdf

```
BUILD_WITH_GOOGLE_FRP := true
```

5.8 AVB (Android Verified Boot), A/B升级

GMS需要启用AVB。有关此功能的详细介绍，请参阅随附的SDK文档：

Rockchip_Introduction_Android_Verify_Boot_CN&EN.pdf

```
BOARD_AVB_ENABLE := true
```

为方便开发与调试，SDK默认代码没有开启AVB，也没有锁定bootloader。这会导致通过认证后也无法显示已认证，请确认在量产前在uboot目录打上以下补丁，以确保刷key时能够同时锁定bootloader，正常显示已认证，调试阶段建议先不加补丁，否则烧写不是同一套固件时，会无法启动。

```
RKDocs/android/patches/gms/0001-libavb-Lock-the-device-when-the-device-init-  
or-write.patch
```

- 在Android 11及以上版本中需要启用虚拟A/B，例如：

```
BOARD_USES_AB_IMAGE := true  
BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := true  
  
# 下面这段在配置时如果没有，请一起添加  
ifeq ($(strip $(BOARD_USES_AB_IMAGE)), true)  
    include device/rockchip/common/BoardConfig_AB.mk  
    TARGET_RECOVERY_FSTAB := device/rockchip/rk356x/rk3566_rgo/recovery.fstab_AB  
endif
```

如果编译时提示默认的分区分大小不够，可以手动配置：

```
ifeq ($(strip $(BOARD_USES_AB_IMAGE)), true)  
    include device/rockchip/common/BoardConfig_AB.mk  
    TARGET_RECOVERY_FSTAB := device/rockchip/rk356x/rk3566_rgo/recovery.fstab_AB  
endif  
+ # 以下这段overlay必须要放到`include device/rockchip/common/BoardConfig_AB.mk`之  
+ 后  
+BOARD_SUPER_PARTITION_SIZE := 3263168512  
+BOARD_ROCKCHIP_DYNAMIC_PARTITIONS_SIZE := $(shell expr  
+ $(BOARD_SUPER_PARTITION_SIZE) - 4194304)
```

- 开启虚拟A/B时，uboot中也要一起打开A/B的配置，例如：RK3566/RK3568

```
diff --git a/configs/rk3568_defconfig b/configs/rk3568_defconfig  
index b7d433a..a87e2e3 100644  
--- a/configs/rk3568_defconfig  
+++ b/configs/rk3568_defconfig  
@@ -204,6 +204,7 @@ CONFIG_AVB_LIBAVB=y  
    CONFIG_AVB_LIBAVB_AB=y  
    CONFIG_AVB_LIBAVB_ATX=y  
    CONFIG_AVB_LIBAVB_USER=y  
+CONFIG_ANDROID_AB=y  
    CONFIG_RK_AVB_LIBAVB_USER=y  
    CONFIG_OPTEE_CLIENT=y  
    CONFIG_OPTEE_V2=y
```

5.9 SELinux

GMS需要启动SELinux。开启SELinux后可能会导致某些功能不正常，具体说明及修改方法请参考随附的SDK文档：

Rockchip_Developer_Guide_Android_SELinux(Sepolicy)_CN.pdf


```
device/rockchip/common/BoardConfig.mk:
BOARD_SELINUX_ENFORCING ?= true
```

对于Android 12及以上版本, 我们提供了一个工具可以处理常见的问题, 请务必使用该工具检查下, 工具位于:

```
RKTools/linux/Android_Devices_SELinux_Tool
```

具体使用方法请参照目录中的 `README.md`

5.10 固定fingerprint (可选)

在测试过程中, 可能需要调整固件。建议在测试之前用以下补丁固定指纹, 以免在更新固件后无法继续测试。

```
diff --git a/prebuild.mk b/prebuild.mk
index 28391f6..7f38922 100644
--- a/prebuild.mk
+++ b/prebuild.mk
@@ -4,3 +4,4 @@ $(warning You can disable this by removing this and setting
BOARD_RECORD_COMMIT_
$(shell test -d .repo && .repo/repo/repo manifest -r -o
$(OUT_DIR)/commit_id.xml)
-include $(TARGET_DEVICE_DIR)/prebuild.mk

+ROCKCHIP_BUILD_NUMBER := 202001 (写一串数字即可, 自己定)
```

5.11 安全补丁等级 Security Patch level

对于安全补丁的获取及说明, 请参考随附的SDK文档:

`Rockchip_Introduction_Android_Security_Patch_CN.pdf`

通过GMS时, 请留意GMS认证窗口的要求, 一般来说安全补丁有效期为三个月, 送测时固件的安全补丁不能过旧。检查此处获取安全补丁等级:

```
build/make:
core/version_defaults.mk
PLATFORM_SECURITY_PATCH := 2020-03-05
```

5.12 Attestation key (Keybox)

进行GMS测试前请烧attestation key。首先要通过3PL向Google申请keybox。拿到keybox后使用瑞芯微提供的keybox打包工具进行打包。打包成烧写的格式, 再使用烧写工具写到设备中, 相关的工具在工程目录:

`RKTools/linux/Linux_AttestationKeyboxPack_Tool.rar`

`RKTools/windows/KeyBoxWrite_v1.53.zip`

详细的操作步骤请参考随附的SDK文档:

`Rockchip_User_Guide_KeyWrite_CN.pdf`

编译固件前, 请检查uboot中烧写attestation key的功能有没有打开, 例如在3566/3568中打开烧写功能:

```
diff --git a/configs/rk3568_defconfig b/configs/rk3568_defconfig
index 46440b6..22831d2 100644
--- a/configs/rk3568_defconfig
+++ b/configs/rk3568_defconfig
@@ -205,6 +205,10 @@ 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_OPTEE_CLIENT=y
 CONFIG_OPTEE_V2=y
+CONFIG_ANDROID_WRITE_KEYBOX=y
+CONFIG_ANDROID_KEYMASTER_CA=y
+CONFIG_OPTEE_ALWAYS_USE_SECURITY_PARTITION=y
```

注意:

使用烧写工具key时, 提示烧写完成后, 不要直接断电, 需要等待8秒左右, 等待烧写相关流程执行完成。可以通过串口打印查看烧写状态(需要确保u-boot代码更新到2022-03)。如 (0表示烧写完成):

```
write attestation key: RSA ret_rsa=0
```

如果要申请keybox, 请提供以下材料(具体需求请和3pl确认):

1. 一份报告 (一条case即可, 截止到文档更新时间, 测试是GtsEdiHostTestCases)
2. 设备信息 (问3pl要模板)
3. 一个txt文件 (一行一个device id对应一个设备)

Google Request to use Device ID to apply the key:

Device ID within the file should meet the following properties:

1. Unique and cannot be duplicate of another Device IDs within the file
2. Must be between 1-32 characters in length
3. Only following characters allowed [a-z][A-Z][0-9][_][-][.]
4. No whitespaces allowed

Device ID files have the following requirements:

1. ASCII text file in unix format.
2. File name should be created with the following characters [a-z][A-Z][0-9][_][-][.] in a meaningful way (e.g. Make_Model_Date_Quantity.txt)
3. Must only contain Device IDs. No comments, headers, or other information
4. One Device ID per line
5. No duplicate Device IDs within file
6. No blank lines
7. No white spaces

5.13 有关Android 12的内核版本使用说明

Android 12同时支持RK3588, 但内核版本于其他平台有所不同。内核源码目录如下表:

平台	kernel-4.19	kernel-5.10
RK3326	Y	N
RK356x	Y	N
RK3399	Y	N
RK3588	N	Y
RK3288	Y	N

编译、修改内核源码，驱动时，请注意不要把目录搞错。

5.14 有关incremental_fs的说明

Android 12版本中，GMS认证要求支持 `incremental_fs v2`，这需要内核驱动支持。为保证兼容所有平台，我司没有将V2的更新合并到 `kernel-4.19` 的主线，而是将 `incremental_fs_v2` 编译成了ko，放在以下目录：

```
vendor/rockchip/common/modular_kernel/4.19$ tree user*
user
|-- incremental_fs.ko
userdebug
|-- incremental_fs.ko

0 directories, 2 files
```

内核版本更新，ko有可能更新不及时，导致无法正确被加载。可以用以下方法确认驱动是否被正确加载，列表中包含 `incremental_fs` 即加载成功：

```
adb shell lsmod
Module                Size  Used by
bcmhdhd                1150976  0
incremental_fs         77824  0
```

如果没有被正确加载，则需要手动更新，所有V2的补丁 `*.patch` 以及补丁基于的提交点 `base.txt`，都可以在以下位置找到：

```
vendor/rockchip/common/modular_kernel/4.19
```

确认 `base` 提交点的方法：

```
git log fs/incfs/
```

如果您不考虑兼容其他Android 12以前的版本，您也可以将这些补丁默认合并到自己的主线中，做 `built-in` 编译。确认编译时是否做了 `built-in` 的方法如下，显示 `is not set` 即未做 `built-in`，显示 `=y` 则做了 `built-in`，这时可以不需要编译 `ko`：

```
cat kernel-4.19/.config |grep CONFIG_INCREMENTAL_FS
# CONFIG_INCREMENTAL_FS is not set
```

5.15 固件签名的生成、替换方法

固件/apk签名生成方法，参考如下说明，信息请自行更换：

```
build/target/product/security/README
```

例如：

```
development/tools/make_key testkey
'/C=CN/ST=Fujian/L=Fuzhou/O=Rockchip/OU=Rockchip/CN=Rockchip/emailAddress=gms@rock-chips.com'
生成: testkey.pk8 testkey.x509.pem

openssl pkcs8 -inform DER -nocrypt -in testkey.pk8 -out testkey.pem
生成:
testkey.pem
```

一般需要自己生成替换以下四组key, 每组3个文件，分别是 pem/pk8/x509.pem：
media platform shared testkey，共计12个文件。直接替换到如下目录：

```
build/target/product/security/
```

5.16 Apex签名的生成、替换方法

一般来说，做Regular GMS设备，需要修改Apex签名，否则会产生如下fail:

```
CtsAppSecurityHostTestCases
android.appsecurity.cts.ApexSignatureVerificationTest#testApexPubKeyIsNotWellKnownKey
java.lang.AssertionError: must not use well known pubkey
Expected: must not match well known key
```

生成方法可以在[Google官网](#)查看，此处拿 com.android.i18n 举例：

```
cd libcore/apex/
rm com.android.i18n.pem com.android.i18n.avbpubkey
openssl genrsa -out com.android.i18n.pem 4096
avbtool extract_public_key --key com.android.i18n.pem --output
com.android.i18n.avbpubkey
```

通常需要修改以下三个Apex，具体可以测试结果的log为准，log会直接打印在测试终端：

Apex module name	Path-Android 11.0	Path-Android 12.0
apex_com.android.i18n	libcore/apex	packages/modules/RuntimeI18n/apex
apex_com.android.runtime	bionic/apex	bionic/apex
apex_com.android.art	art/build/apex	packages/modules/ArtPrebuilt

注意：如果更换后仍不能pass，可以换测试主机试下，有部分客户遇到PC主机导致的fail.

5.17 检查脚本

对于以上大部分要求，Android 12版本已增加一键检查脚本，请在 `lunch` 自己的项目后，使用该脚本进行检查。例如：

```
source build/envsetup.sh
lunch rk3326_sgo-user

source device/rockchip/common/scripts/check_gms_env.sh
```

对于Android SDK `lunch`和编译，默认会有检查App权限的提示，请确认所有App没有申请特殊权限后再注释掉该报错！例如：

```
device/rockchip/common/modules/gms.mk:31: warning:
*****
device/rockchip/common/modules/gms.mk:32: error: Please note that all your
apps MUST be able to get permissions, Otherwise android cannot boot!.
17:38:09 dumpvars failed with: exit status 1
```

5.18 送测前需要检查的部分

送到实验室正式测试之前，一定要根据GMS Requirements仔细核对，以节约时间，对于部分要求给出检查方法：

- 安全模式
在开机动画阶段按住 `音量-` 或 `重启`时按住重启的图标，即可进入安全模式。
- Lockdown 模式

1. 添加一个锁屏密码
2. 打开 `设置`
3. 下滑滚动并点击 `安全和位置`
4. 在设备安全下方点击 `锁屏偏好`
5. 点击 `显示lockdown选项`

- 磁盘加密/文件加密 `Full Disk Encryption/File Base Encryption`
设备启动后请在以下选项查询状态：

```
settings->security & location->Encryption & credentials->Encrypt tablet
```

6. 测前准备

1. 请按Rockchip Android SDK发布说明中的刷机方法，为测试机烧写固件；
2. 由于集成了GMS包，烧写结束后第一次启动会比较慢，请耐心等待，开机启动后完成GMS Setup Wizard中的设置，设置默认语言为United States English，wifi部分请先跳过，时区请选择美国，设置用户信息，进入Home主界面；
3. 确保机器按以下配置，Settings->Wi-Fi连接wifi，连接vpn网络环境，主机也要连接vpn；

4. 开始测试CTS/GTS之前，切记不要登录GMS账号，否则会有fail；如果是EEA Go项目，其中GtsRegulationComplianceTestCases 单项测试要求，测试前登陆EEA测试帐号，并选择非Google搜索引擎（例如：DuckDuckGo）。具体请以EEA官方文档为准：

```
https://docs.partner.android.com/gms/policies/restricted/eea-builds?hl=en#choice-screen
```

5. Settings->Security->Screenlock选择None；
6. 如果产品（如laptop类产品）带物理键盘，Languages & input->Physical keyboard->Show virtual keyboard，勾选该选项；
7. 连续点击Settings->About tablet（phone）->Build Number，使被隐藏的Developer Options显示出来；
8. Settings->Developeroptions->Stayawake，勾选该选项；
9. Settings->Location打开定位服务（默认是打开的，不要关闭）；
10. Settings->Display->Sleep设置成最长时间，将亮度调节到最暗（测试时间较长，以节省电量）；
11. 需使用android-cts-media-version新版的媒体资源包。
12. 拷贝媒体资源文件，执行媒体包下的脚本 `source copy_media.sh && source copy_image.sh` 拷贝媒体文件(或将媒体包放到以下位置： `/tmp/android-cts-media/android-cts-media-1.4`)。android 11及以上版本需要 media-1.5。
13. 查看Sensor校准状态，校准状态永久有效，如果该机器做过校准则不需要再做。测VTS前务必要确认校准状态，查看方法：`cat /sys/class/sensor_class/accel_calibration`，如果有值打印，类似 `accel calibration: -604, 131, 535`，则说明校准成功。未校准的机器将其水平静止放置，输入命令 `echo 1 > /sys/class/sensor_class/accel_calibration` 即可校准，校准后请确认是否校准成功；
14. 物理竖屏的机器要竖屏放置，物理横屏的机器横屏放置；
15. 每次重新测试，都要将机器进行如上配置。
- 注：Android 11及以上版本测试GTS前，必须在向导中连接wifi，帐号不要登陆，否则Gts会有fail.

7. 主机配置

需要使用Ubuntu主机并自行配置好java，Python，adb，fastboot，aapt环境。

FTP服务器：

地址	账户	密码
ftp://ftp.rock-chips.com	xzj-guest	V3qTwMQ4YV

对Android 12及以上版本，我们将在FTP中持续更新客户遇到的常见问题及工具，可以从FTP的以下位置获取：

```
GMS-Test-Suite/Exp+
```

推荐使用RK-FTP中的Android_O_cts_env.tar.gz，解压后配置环境变量：

Ubuntu下可以加载到~/.bashrc或者编辑以下内容到文件中如env，测试前source env：

```
export JAVA_HOME=/home/Your_Name/Software/jdk1.8.0_77
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH
export PATH=/home/Your_Name/Software/android-sdk-linux/tools:$PATH
export PATH=/home/Your_Name/Software/android-sdk-linux/platform-tools:$PATH
export PATH=/home/Your_Name/Software/android-sdk-linux/build-
tools/19.0.0:$PATH
```

配置后，请按照下面Q&A中的方法升级adb、aapt及fastboot工具，如果不满足最低版本要求，测试时将会产生各种问题！

各个Android版本要求的adb/aapt/fastboot版本如下：

Android版本	adb版本最低要求	fastboot版本最低要求
Android 10.0	29.0.6+	29.0.6+
Android 11.0	30	30
Android 12.0	31	31
Android 12.1 (Android 12L)	31	31

如果出现aapt不可用，请安装c++兼容库：

```
sudo apt-get install lib32stdc++6 lib32z1
```

1. 安装Python开发包：

```
$ sudo apt-get install python-dev
```

2. 安装 Protocol Buffer工具：

```
$ sudo apt-get install python-protobuf
$ sudo apt-get install protobuf-compiler
```

3. 安装 Python 虚拟环境相关工具：

```
$ sudo apt-get install python-virtualenv
$ sudo apt-get install python-pip
```

4. 如果你是python3，可能需要单独装下面这个：

```
$ sudo apt install virtualenv
```

8. 主机环境常见问题及注意事项

1. 出现无法创建虚拟环境

VTs测试时无法创建虚拟环境，可以尝试以下命令强制指定pip为pip2并通过pip安装virtualenv：

```
sudo apt autoremove python-virtualenv
sudo apt autoremove virtualenv
sudo ln -sf ~/.local/bin/pip2 /usr/bin/pip
pip install --user virtualenv
```

2. 升级adb、aapt及fastboot

使用SDK中提供的adb、aapt和fastboot工具替换。参考随附的SDK文档：

Rockchip_Android_*_development_guide_V*_CN.pdf

Q：替换adb/fastboot

A：以adb为例，在终端键入：

whereis adb

```
$ adb: /home/rockchip/Software/android-sdk-linux/platform-tools/adb
```

```
adb kill-server
```

得知adb/aapt/fastboot的位置后，替换adb/aapt/fastboot的二进制文件，二进制文件可在[SSDK/RKTools/linux/Linux_adb_fastboot](#)获取。

9. 多台机器协作测试

在以下的测试中大多支持协作测试，命令为：

```
--shard-count
```

- 如3台机器同时跑cts：

```
run cts --shard-count 3 -s SN1 -s SN2 -s SN3
```

注意：多台机器在测试时，如果其中一台机器掉电或死机，那这台机器中的报告将全部丢失，测试总数量会异常！建议先将机器调试稳定后，先已一台机器测试，测十分钟就拔掉，生成一份完整报告，再基于这个报告多台机器retry测试！

10. GSI烧写的方法

确认机器解锁后，进入fastbootd，只需要烧写GSI中的system.img及固件中的misc.img，烧写后会进入recovery进行恢复出厂设置。下面附上整个烧写流程：

1. 重启至bootloader，未解锁->机器解锁：

```
adb reboot bootloader
fastboot oem at-unlock-vboot ## 对于烧写过avb公钥的客户，请参考对应的文档解锁。
```

2. 恢复出厂设置，重启至fastbootd：


```
fastboot flash misc misc.img
fastboot reboot fastboot ## 此时将进入fastbootd
```

3. 开始烧写GSI

```
## (可选) 对于分区空间紧张的设备, 可以先执行本条命令删除product分区后再烧写GSI
fastboot delete-logical-partition product
## 对于使用A/B, 虚拟A/B的设备, 需要同时删除product_a/product_b
fastboot flash system system.img
## 烧写成功后, 重启
fastboot reboot
```

- 注1: 也可以使用DSU(Dynamic System Updates)烧写GSI, 目前Rockchip平台已经默认支持DSU。由于该功能需要消耗大量内存, 不建议1G DDR及以下的设备使用, 有关DSU的说明和使用, 请参考[Android官网](#);
- 注2: VTS测试时, 需要同时烧写编译出的boot-debug.img到boot分区;
- 注3: CTS-ON-GSI测试时则不需要烧boot-debug.img;
- 注4: 测试时请使用Google官方发布的, 带有-signed结尾的GSI镜像;

11. 测试VTS/CTS-ON-GSI

根据[Google GMS文档](#)的要求, 测试时需要将机器按下列情况锁定/解锁。

- CTS: 原有的OEM固件, 即正常编译出的完整固件。
 - CTS-on-GSI: 解锁机器, 刷写GSI, 锁定机器后再进行CTS-on-GSI的测试。
 - VTS: 解锁机器, 刷写boot-debug和GSI, 再进行VTS的测试。
- 注意: 在CTS-on-GSI的情况时, 务必要确保GSI为签名(-signed)版本, 且GSI的安全补丁等级要等于或高于SDK的安全补丁等级, 否则烧写后再锁定设备, 系统将无法启动!
- 注意: Android 11测试CTS或CTS-on-GSI时, 务必将设备按上面所说的进行锁定。否则会产生如下fail项:

```
CtsKeystoreTestCases
android.keystore.cts.KeyAttestationTest#testEcAttestation
android.keystore.cts.KeyAttestationTest#testRsaAttestation
android.keystore.cts.KeyAttestationTest#testEcAttestation_DeviceLocked
android.keystore.cts.KeyAttestationTest#testRsaAttestation_DeviceLocked
```

12. VTS

需要额外烧写GSI(即AOSP的system-xxx-signed.img, 可从Google官网/3PL或Rockchip安全补丁FTP获取, 要使用signed的镜像)以及boot-debug.img (开启AVB后打包固件会打包到rockdev/Image-xxx中)。

测试包	测试命令	retry命令	单测命令
android-vts-xxx-arm_64.zip	<code>run vts</code>	<code>run vts -- retry 0</code>	<code>run vts -m module_name -t case_name</code>

13. CTS-ON-GSI

需要额外烧写GSI(即AOSP的system-xxx-signed.img, 可从Google官网/3PL或Rockchip安全补丁FTP获取, 要使用signed的镜像), 烧写方法同上。

测试包	测试命令	retry命令	单测命令
android-vts-xxx-arm_64.zip	<code>run cts-on-gsi</code>	<code>run retry -- retry 0</code>	<code>run cts-on-gsi -m module_name - t case_name</code>

在Android 11中, 按以下格式:

测试包	测试命令	retry命令	单测命令
android-cts-xxx-arm_64.zip	<code>run cts-on-gsi</code>	<code>run retry -- retry 0</code>	<code>run cts-on-gsi -m module_name - t case_name</code>

14. CTS

使用完整编译出的user固件进行测试, 固件配置请按文档前面所说检查和配制好。

测试包	测试命令	retry命令	单测命令
android-cts-xxx-linux_x86-arm.zip	<code>run cts</code>	<code>run retry -- retry 0</code>	<code>run cts -m module_name -t case_name</code>

15. CTS-Verifier

使用完整编译出的user固件进行测试, 固件配置请按文档前面所说检查和配制好。

测试包	测试命令	retry命令	单测命令
android-cts-verifier-10_xx-linux_x86-arm.zip	手动测试	手动测试	手动测试

1. Android 10.0及以上, 请在安装CtsVerifier前, 执行以下命令:

```
adb shell settings put global hidden_api_policy 1
```

2. 安装CtsVerifier, 请务必使用这个命令安装, 且在上一步执行之后

```
adb install -r -g CtsVerifier.apk
```

3. Android 10.0及以上需要额外执行

```
adb shell appops set com.android.cts.verifier android:read_device_identifiers allow
```

4. Android 11.0及以上，需要额外执行这个命令

```
adb shell appops set com.android.cts.verifier MANAGE_EXTERNAL_STORAGE 0
```

5. 设置好系统的时间

16. GTS

使用完整编译出的user固件进行测试，固件配置请按文档前面所说检查和配制好。测试前务必确认好主机环境，是否配置好gts key，可通过`echo $APE_API_KEY`查看。

- GTS key配置方法
从3PL获取后，将key放好后，添加到环境变量中，如：

```
$ vi .bashrc
export APE_API_KEY=/path/to/key.json
```

测试GTS前，务必在向导时连接vpn-wifi, 帐号不要登陆，否则会导致以下fail:

```
com.google.android.gts.setupwizard.SetupWizardZeroTouchTest
```

测试包	测试命令	retry命令	单测命令
android-gts-xx.zip	<code>run gts</code>	<code>run retry --retry 0</code>	<code>run gts -m module_name -t case_name</code>

17. STS

使用完整编译出的userdebug固件进行测试，固件配置请按文档前面所说检查和配制好，有关fail项的处理，请参考随附的SDK文档 [Android安全补丁说明文档](#)

测试包	测试命令	retry命令	单测命令
android-sts-xx.zip	<code>run sts-engbuild</code>	<code>run retry --retry 0</code>	<code>run sts -m module_name -t case_name</code>

18. GOATS (Performance Test)

性能测试，使用完整编译出的userdebug固件进行测试，固件配置请按文档前面所说检查和配制好。编译内核必须用Clang编译（编译方法请看开发文档）kernel中打入如下补丁

```
diff --git a/drivers/video/rockchip/rga2/Kconfig
b/drivers/video/rockchip/rga2/Kconfig
index efc1ef6..3458051 100644
--- a/drivers/video/rockchip/rga2/Kconfig
+++ b/drivers/video/rockchip/rga2/Kconfig
@@ -1,5 +1,5 @@
 # SPDX-License-Identifier: GPL-2.0
 -menu "RGA2"
 +menu "RGA2G"
     depends on ARCH_ROCKCHIP

config ROCKCHIP_RGA2
```

19. APTS

Android 12及以上的机器，需要测试新的性能测试，不再测试GOATS。

测试包	测试命令（填写对应user固件的fingerprint）	单测命令，不支持retry
android- apts- go.zip	<code>run test/approval-go --fingerprint-swap google/wembley/wembley:12/SP1A.210712.001/7539480:user/release</code>	<code>run test/approval-go -- test-case app- start-cold-3p</code>

20. BTS

自2018年4月1日起，所有的设备过认证都要进行Android BTS (Build Test Suite) 测试，请参考Google Partner文档：

<https://support.google.com/androidpartners/gms/answer/9027630?hl=en>

上传固件给Google时，不要上传打包后的 `update.img`，否则会出现Google无法解析镜像的问题，具体细节请自行查看文档和询问3pl。RK建议的BTS打包方法为：将以下列表中的镜像放到文件夹中，命名为fingerprint(其中所有的":","/"请更换为 "~"), 然后再压缩为zip格式。

```
rockchip~rk3326_qgo~rk3326_qgo~10~QD1A.190821.014.C2~201911~user~release-keys$
ls
boot-debug.img boot.img dtbo.img misc.img recovery.img super.img
vbmeta.img
```

关于PHA：有客户曾经出现过GMS测试全部通过，但因BTS中检测出PHA（有害应用）被拒绝的情况，根据Google香港峰会中安全会议所说，可以先将需要预置的app上传到Google Play进行检测，具体咨询3PL或TAM，避免BTS出现类似的情况。

21. 其他测试命令

单独跳过某一项：

```
run cts --exclude-filter "CtsViewTestCases  
android.view.cts.SurfaceViewSyncTests#testEmptySurfaceView"
```

添加fail子项plan：

help add查看

```
add subplan --session xx -n plan_name --result-type type
```

按新建的子项plan测试：

```
run cts -o -a arm64-v8a --skip-all-system-status-check --subplan xxxx
```

22. GMS Express

为建立良好的Android生态，使得市面上所有的Andrid设备均能及时应用到Google每月release的安全更新，Google推出了GMS Express Program。

Rockchip全系列平台Android 12.0平板方案将全面支持，如以下表格所示，本文档介绍Rockchip GMS Express技术相关内容。名单会不断更新，请以[Google网站名单](#)为准。

芯片平台	Go Express	Regular Express
RK3326	支持	不支持
RK3566	支持	支持
RK3399	不支持	即将支持
RK3588	不支持	即将支持
RK3288	不支持	即将支持

22.1 注意事项

- 使用Rockchip GMS Express Baseline的客户，请务必确保与Google有签署MADA协议，若是集成Android Go请务必确保有签署Go的补充MADA；集成EDLA，请确保签署EDLA协议，不按照协议滥用设备所造成的一切法律风险需自行承担。
- Rockchip GMS Express baseline，会按Google要求每月都会持续更新（包括安全更新、GMS包的更新、Google提供的AOSP重要补丁等），每个月的baseline正式发布后会正式通知，请务必跟进更新，并将更新按ota方式推送给终端用户。若未及时更新我司将不予提供GMS相关技术支持。

22.2 工程下载及配置

请联系RK FAE以及对外SDK相关接口人，获取GMS Express Baseline的repo下载地址及相关下载权限的开通，注意：**这不是RK常规发布的SDK**，否则将获取不到定期安全更新及GMS包更新。Android 10及以上版本中，所有平台代码统一，获得Rockchip SDK后，更新至Express baseline的方法如下，不必重新下载：

```
repo init -m Android10_Express.xml

# Android 11请使用Android 11的xml
repo init -m Android11_Express.xml

# Android 12请使用Android 12的xml
repo init -m Android12_express.xml
```

- 直接获取Express baseline的方法如下：

```
repo init --repo-url=ssh://git@www.rockchip.com.cn:2222/repo-
release/tools/repo.git -u
ssh://git@www.rockchip.com.cn:2222/Android_Qt/manifests.git -m
Android10_Express.xml

# Android 11
repo init --repo-url=ssh://git@www.rockchip.com.cn:2222/repo-
release/tools/repo.git -u
ssh://git@www.rockchip.com.cn:2222/Android_R/manifests.git -m
Android11_Express.xml

# Android 12
repo init --repo-url=ssh://git@www.rockchip.com.cn:2222/repo-
release/tools/repo.git -u
ssh://git@www.rockchip.com.cn:2222/Android_S/manifests.git -m
Android12_express.xml
```

- 请确保产品目录Makefile中做如下配置：

```
BUILD_WITH_GOOGLE_GMS_EXPRESS := true
```

这个宏配置打开后，**我们不预置任何RK的应用，有需要自行添加**。编译结束后，默认会配置com.google.android.feature.GMSEXPRESS_BUILD的flag，我们的baseline满足Express Plus的要求。谷歌鼓励Express Plus的产品，过GMS认证时可以拿到相关补贴支持，详情请咨询3PL。在具体项目二次开发后，请参考[Google Exp+要求文档](#)确认是否满足Express Plus的要求。若满足，修改以下文件：

vendor/rockchip/common/gms-express.xml

```
diff --git a/gms-express.xml b/gms-express.xml
index 78f4d99..de93557 100644
--- a/gms-express.xml
+++ b/gms-express.xml
@@ -16,5 +16,5 @@
-->
<!-- These are configurations that should exist on GMS Express devices. -->
<config>
-   <feature name="com.google.android.feature.GMSEXPRESS_BUILD" />
+   <feature name="com.google.android.feature.GMSEXPRESS_PLUS_BUILD" />
</config>
```

23. GTS本地媒体包测试

- 使用自己搭建的GTS媒体包最大的好处就是能够排除网络问题，实现视频秒播。
 - 必须是与vpn环境在同一网段下的服务器(Nginx或其他)
 - https必须是有证书单/双向认证的
 - http也可以
 - 具体可以参考文档 [GTS_CTS Media Test Guide v1.2.pdf](#)
- 附一份使用http的nginx配置:

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /home/bjc/www_gms;

    # Add index.php to the list if you are using PHP
    index index.html index.htm index.nginx-debian.html;

    server_name _;

    location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ =404;
    }
}
```

最终目录结构:

```
bjc@bjc-HP-Compaq-dx7518-MT:~/www_gms$ tree
.
|-- client-key.pem
|-- exoplayer-test-media-1
|   |-- gen-4
|       |-- common-encryption
|           |-- tears-aes-cbc1.mpd
|           |-- tears-aes-cbcs.mpd
|   ...
|       |-- dynamic-config-1.0.json
|       |-- dynamic-config-sdcard-1.0.json
|       |-- exoplayer-gts-media.zip
|       |-- gts
|           |-- exoplayer
|           |-- screens
|               |-- dash-vod-single-segment
|                   |-- audio-140-enc.mp4
|   ...
|-- index.html
|-- youtube-test-media
|   |-- gts
|       |-- GtsYouTubeTestCases-media-1.0.zip
|       |-- dynamic-config-1.0.json
```

24. 开机向导Wi-Fi设置概率无法skip

性能差导致，可以打这个补丁：

```
patch/frameworks/base/increase_waiting_time_for_setup_wizard.diff
```

25. 通过认证后，无法显示已认证问题

不能显示已认证，一般有两种情况：

1. 机器未烧key
2. 机器未锁定

为方便开发及调试，默认将机器的状态设为解锁，这里提供一个补丁，默认在烧写key时将机器锁定。

待生产前，请务必加上补丁以保证烧attestation key时能够锁定机器。

```
patches/gms/0001-libavb-Lock-the-device-when-the-device-init-or-write.patch
```