Android 4.0.3 cts 测试文档

CTS 测试全称 Compatibility Test Suite。通过 cts 测试可以保证设备具有良好的用户体验，保证大量优质的应用可以正常运行；也能够让应用开发者能够放心地制作高质量的应用程序，而且只有通过 cts 认证的设备才能合法的安装使用 google market 等 GOOGLE 的应用。

一． CTS 测试机的硬件配置要求：

1.LCD 屏分辨率至少在 426*320 以上，宽高比必须是 4：3 或者是 16：9

2.设备必须支持 OpenGL ES 1.0 and 2.0

3.必须具有至少一个 sensor（ Accelerometer，Magnetometer，GPS，Thermometer， Proximity Sensor，Gyroscope）

4.必须至少具备一个可以上网的无线连接（wifi，3G，bluetooth）

5.camera(前置，后置)。谷歌的测试用例中会测试后置 camera，如果没有后置摄像头需要把我们的虚拟的后置摄像头打开（至少在 r3 测试中仍然有测试用例去测试后置摄像头）。前置摄像头是可选的。

6.至少剩余 340M 可用物理内存（DDR），350M 的 data 空间（flash 的一个分区），1G 的存储器（可以是 sdcard 也可以是 flash 存储器）。我们建议使用 1G 以上的 ddr。

7.理论上要求一个 USB client port 和一个 USB host port。

具体请参考谷歌发布的 android-4.0-cdd.pdf

二．测试设备软件配置

1.请使用我们发布的可以过 CTS 的 SDK。

2.将要进行 cts 测试的工程必须在 user 模式下进行编译。请修改 andriod 工程根目录下 buildspec.mk

3. 编译前删除工程下已生产的编译结果，运行 make clean && rm out -r。

4.进行如下必要的配置，由于我们的机器把 flash mount 成为 sd 卡，所以不用插卡。

A．Settings > Security > Screen 选为 none 。

B．打开 adb ： Settings > Developer options > USB debugging 。

C．选中 Settings > Developer options > Stay Awake is checked 。

D．选中 Settings > Developer options > Allow mock locations 。

E．选中 Settings>Location services>Google ’ s location service。

F．连接 wifi 。 测试过程中会访问 youtube、google 等地址， 如果你的网络不允许最好等搭建 vpn ，否则会导致多项测试失败。

G．安装 android-cts/repository/testcases 目录下的两个 apk ：CtsDelegatingAccessibilityService.apk 和 CtsDeviceAdmin.apk 。

H．使能 Settings > Accessibility > Delegating Accessibility Service。

I．选中 Settings > Security > Device Administrators 下所有复选框。

J．下载 cts 测试用媒体文件 android-cts-media-1.0.zip 并按照其中说明 copy 文件到 device.

三．上位机配置及操作

1.必须是 Linux 操作系统。

2.下载 CTS 测试包 android-cts-4.0.3_r3-linux_x86-arm 并解压

3. 连接设备到 pc ，并保证设备与上位机的 adb 连接正常。

4. 运行 android-cts /tools/ cts-tradefed，带进入 CTS 控制台后，键入 “ run cts -- plan CTS ”进行测试。

cts-tradefed 控制台命令简介：

help 显示帮助

help all 显示详细帮助

exit 退出

run cts --plan test_plan_name 运行一个 plan ， 已有的 plan 在 android-cts /repository/plans 下，也可以编写自己的 plan

run cts --package/-p 以包为对象进行测试

run cts --class/-c [--method/-m] 以类或类的一个方法为对象进行测试

run cts --continue-session session_ID 继续一个之前的测试，以前的测试可以用命令 “ l r ”查看， session_ID 即第一列显示的内容

run cts [options] --serial/s device_ID 在指定设备号的设备上运行 cts

run cts [options] --shards number_of_shards 在多个设备上并行运行 cts ， 可提供测试速度 ， 测试用设备必须用相同的固件

run cts --help/--help-all 获取帮助

list devices 显示所有已连接的设备

list packages 显示所有进行测试的包

list plans 显示已有的计划

list invocations 显示当前在运行的测试

list commands 显示当前在运行的命令

add derivedplan --plan plane_name --session/-s session_id -r [pass/fail/notExecuted/timeout] 添加一个 plan ，继承之前的测试

dump logs 打印 log

四．关于 WideVine DRM

1.客户必须把品牌 ro.product.manufacturer 和产品型号 ro.product.model 注册到谷歌的认证服务器上之后才开始下面的移植步骤。

2.必须使用我们发布的可以 CTS 的 SDK，使用最新的工具。

RK30 必须使用 1.10 以上的 loader，RK29 必须使用 RK29xxLoader(L)_2.25 以上的 loader。因为 DRM 功能需要 loader 和 flash 驱动的支持。

3 对于已经申请到的 DMR 的源码，其中 widevine_source\proprietary\libcrypto\libcrypto.c 这里面定义的 4 个接口需要我们平台来实现，其他的都使用谷歌默认的实现即可。

OEMCryptoResult OEMCrypto_EncryptAndStoreKeyBox(OEMCrypto_UINT8 *keybox, OEMCrypto_UINT32 keyBoxLength)----------这个接口是用来保存获取到的 DRM 的 key，

现在我们的实现在把获取到的 key 保存在 flash 中的特定区域，保证不会被任何工具擦除。

OEMCryptoResult OEMCrypto_IdentifyDevice(OEMCrypto_UINT8* deviceID,
OEMCrypto_UINT32 idLength)----------这个接口是用来获取唯一区别一台设备的 ID，目前我们有两种实现方法，一种是提供给客户烧写序列号的工具，客户自行产生序列号并烧写，第二种是通过 wifi 的 mac 地址来产生一个唯一的序列号。客户需要根据自己的实现方法来修改我们默认的代码，我们默认提供的实现是获取随机产生 SN 码。根据 wifi 的 mac 地址随即产生序列号的方法请参考“序列号补丁”；使用工具烧写自定义的 SN 码的，请参考“regionTag 补丁”的实现，注意：这个并不是读取序列号的补丁，请根据写入的 SN 的长短和格式进行修改。

OEMCryptoResult OEMCrypto_GetKeyboxData(OEMCrypto_UINT8* buffer,
OEMCrypto_UINT32 offset, OEMCrypto_UINT32 length)-------这个接口是从设备中读取已经存储的 DRM 的 key。

OEMCryptoResult OEMCrypto_GetRandom(
OEMCrypto_UINT8* randomData, OEMCrypto_UINT32 dataLength)----这个接口是用来获取随机值。

4.整合了 DMR 的源码之后，需要用我们加密工具来产生一组密钥，并且用公钥加密 loader，用私钥加密 boot.img 和 recovery.img。注意:一款产品只能用同一组密钥，现在已经出了可以直接解密 update.img 的工具，不需要再分开加密了。
5. 在 loader 签名了并且获取到 DRM 的 key 之后，一旦 loader 检测到签名不对的 boot 和 recovery 就会关闭 KeyBox 并且删除 DRM 的 key,这个时候即使再烧回正确的固件也不行，只能手动的去开启 SecureBoot。所以在烧写的时候请注意要使用加密后的固件，并且同一台机器只能烧写同一个密钥加密过的固件。
如果烧了没加密的固件或者签名错误的固件，在启动的时候串口会显示的 SecureBootEn=0，这个时候需要手动去打开 SecureBoot。在烧写了正确加密的固件后，请将补丁包中的 SecureSwitch 文件 adb push 到 system/bin/目录下，并且在机器上执行 SecureSwitch -o 命令（如果权限不够，请 chmod 一下），然后重启，启动时串口打印信息中要看到 SecureBootEn=1，这样 DRM 才能正常工作。
6. 对于 RK29 的机器，必须打上 DRM 补丁\RK29\newDRM 补丁 516 这个补丁
对于 RK30 的机器，必须打上 DRM 补丁\RK30 这个补丁。

五．客户申请 CTS 测试的流程以及遇到问题时的处理方法
1. 目前进行 CTS 测试根据客户性质的不同主要有两种方式：
A、客户拥有 google 授予的 GMS license；
B、普通客户，无 GMS license；
第一种客户可以直接像 google 申请测试，期间 DRM 代码、GMS 包他们都是可以自己拿到的。目前全球只有 45 家拥有 GMS license（目前已知的是华为、中兴、海信、TCL、爱可视、anydata 等）。
第二种客户只能通过拥有 GMS license 的方案公司进行 GMS 移植合成，然后通过他们向 google 提交测试申请。如 anydata（目前 CVT 的项目就是通过他），这个是收费的。

具体申请的细节，市场部门苗立峰比较清楚，业务部如要进一步了解可以找他。

2. 在 CTS 认证过程中，RK 和厂家各自要做的工作和责任

A、RK 负责提供可以过 CTS 测试的公版软件和定制文档、DRM 源码修改范例、加密工具、CTS 的相关补丁以及必要的技术支持。

B、厂家要负责如下工作

1. 如果厂家不是 Google 的会员，则必须通过第三方厂家来申请 CTS 认证，包括申请 GMS、WideVine 包。

2.拿到 WideVine 源码后，并根据 DRM 源码修改范例参照自己的实现方式修改 WideVine 源码并通过测试，修改方法如上第四部分所述。

3.根据定制文档修改公版软件并通过 CTS 测试，CtsVerifier 测试。

4.第三方厂家需要支持他们的客户移植 Widevine，而不是直接找我们来支持。

3.测试 CTS 时遇到问题的处理方法。

a. 遇到问题的时候，先对照本文档第二部分以及第四部分的配置，看是否配置不当导致的。

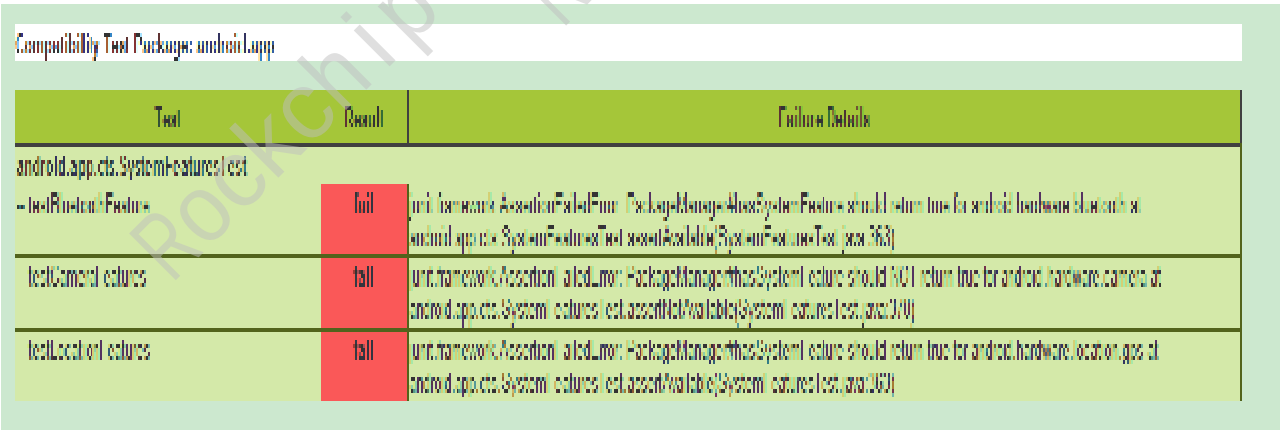
b. 如果确认配置没有错，排除一下是不是网络问题引起的。详见第四部分的描述

c. 单独测试一下这个测试用例，如果可以通过再单独测试一下这个类或者测试包。有个别的测试用例可能会出现概率性的失败。

d. 如果单侧还是不能通过，重启机器，再单独测试这个测试用例，如果还是不能通过，请单独抓一下这个测试用例的 log 并跟我们联系。

六. 客户机器定制

因为客户的机器的硬件配置各有不同，必须根据如下进行配置



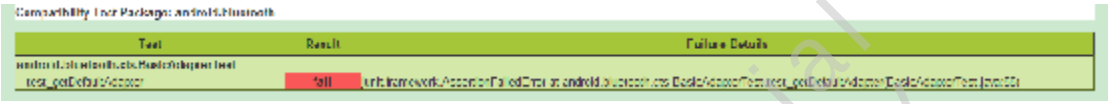
其中 testBluetoothFeature 这个是因为这个产品没有蓝牙，解决方法：在 device.mk 中吧所有关于 bluetooth 的设置关掉，同时在 SystemServer.java 中把 230 左右，把 bluetooth 的 service 启动的地方去掉

testCameraFeatures 是因为这个产品有后置摄像头（如果只有前置摄像头的设备打开了虚拟的后置摄像头也要拷贝，在 hardware/rk29/camera/CameraHal.h 中把 #define CONFIG_CAMERA_SINGLE_SENSOR_FORCE_BACK_FOR_CTS 1 来打开虚拟后置摄像头），所以要在 device.mk 中增加这个拷贝

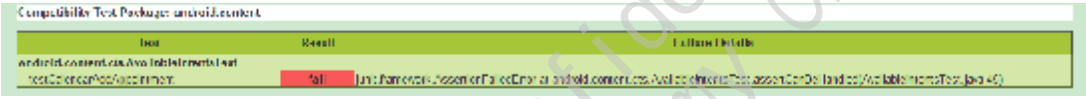
frameworks/base/data/etc/android.hardware.camera.xml:system/etc/permissions/android.hard

ware.camera.xml \
注意：如果后置摄像头支持 auto_focus 和 flash 的功能，则必须拷贝对应的配置文件
如果机器有前置摄像头则要再拷贝 android.hardware.camera.front.xml
如果机器只有前置摄像头，必须如上所述打开虚拟的后置摄像头，并拷贝后置摄像头的配置文件。

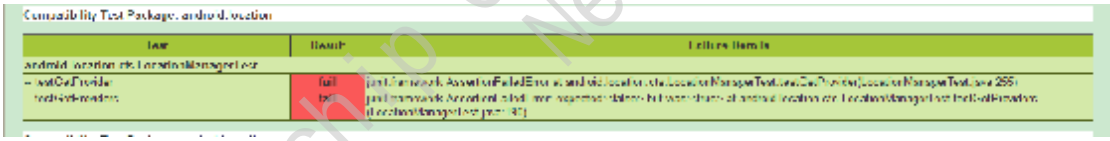
testLocationFeatures 这个是因为这个产品没有 gps，所以要在 device.mk 中把相关 gps 的 feature 以及跟 gps 有关的东西都去掉，frameworks/base/data/etc/android.hardware.location.gps.xml:system/etc/permissions/android.hardware.location.gps.xml \
同时在 frameworks/base/data/etc/handheld_core_hardware.xml 中把 + <!--feature name="android.hardware.location.network" /--> 注释掉,因为这个产品没有 google map 等应用，如果有的话就要加上这个 feature。



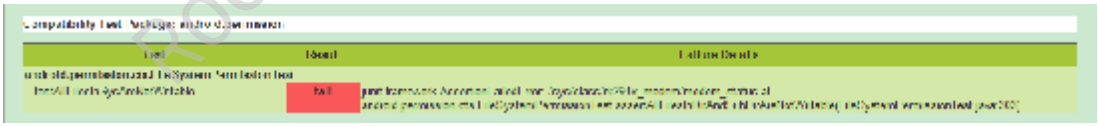
这个只要把 bluetooth 的 service 去掉后就可以通过了



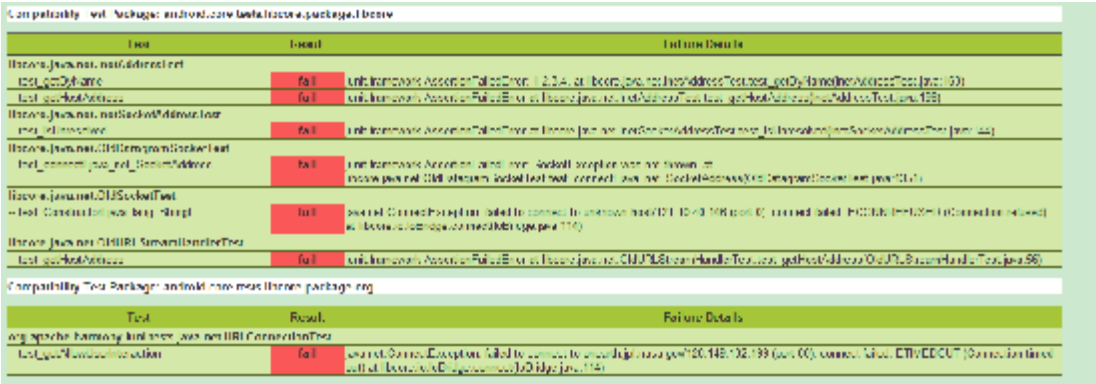
这个是因为他们在这个产品中把 Calendar.apk 去掉了，加上去即可通过。



这个只要按照第一个问题中的方法修改即可通过



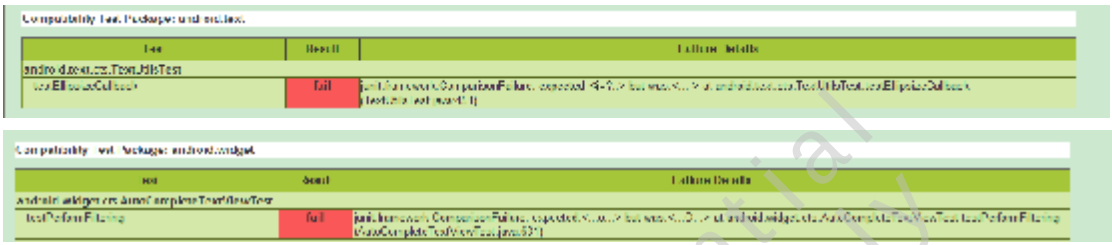
这个是因为 kernel 节点的权限问题造成的，只要把相应的节点的 other 的写权限去掉就可以了。可能存在其他节点也有写权限的问题。



Compatibility Test Package: android.core.tests.libcore.package.tests

Test	Result	Failure Details
tests.api.javax.net.ssl.SSLSocketFactoryTest		
--test_createSocket	fail	junit.framework.AssertionFailedError: java.net.ConnectException: failed to connect to [illegible]/121.10.40.154 (port [illegible]): connect failed: ECONNREFUSED (Connection refused) was thrown instead of UnknownHostException at tests.api.javax.net.ssl.SSLSocketFactoryTest.test_createSocket(SSLSocketFactoryTest.java:[illegible])
tests.api.javax.net.ssl.SSLSocketTest		
--testConstructor_04	fail	java.net.ConnectException: failed to connect to [illegible]/121.10.40.154 (port [illegible]): connect failed: ECONNREFUSED (Connection refused) at libcore.io.IoBridge.connect(IoBridge.java:114)
--testConstructor_05	fail	java.net.ConnectException: failed to connect to [illegible]/121.10.40.154 (port [illegible]): connect failed: EINVAL (Invalid argument) at libcore.io.IoBridge.connect(IoBridge.java:114)
tests.api.javax.net.SocketFactoryTest		
--test_createSocket_02	fail	junit.framework.AssertionFailedError: java.net.ConnectException: failed to connect to [illegible]/121.10.40.154 (port [illegible]): connect failed: ECONNREFUSED (Connection refused) was thrown instead of UnknownHostException at tests.api.javax.net.SocketFactoryTest.test_createSocket_02(SocketFactoryTest.java:[illegible])
--test_createSocket_05	fail	junit.framework.AssertionFailedError: java.net.ConnectException: failed to connect to [illegible]/121.10.40.154 (port [illegible]): connect failed: EINVAL (Invalid argument) was thrown instead of UnknownHostException at tests.api.javax.net.SocketFactoryTest.test_createSocket_05(SocketFactoryTest.java:[illegible])

这些问题都是由于国内网络问题造成的，到香港测试可通过

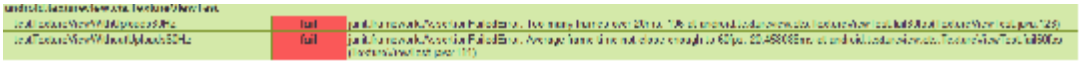


这两个测试项跟默认输入法有关系，如果默认输入法设置成谷歌拼音的话，在测试这两个用例的时候，Instrumentation 发送的 key 会被输入法截获，然后再转到测试的控件上面。所以默认输入法必须保证是英文的。

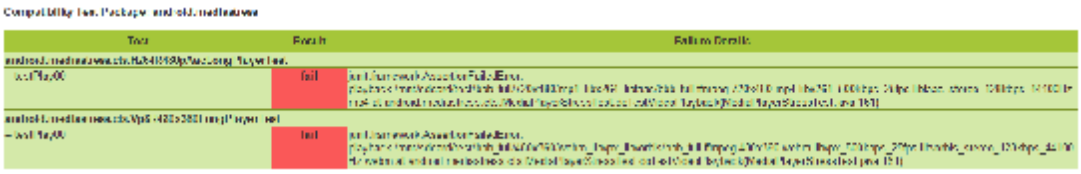
这个是由于内置的 apkinstaller.apk 是用的谷歌自带的 key 来签名的，只要把 apk 删掉或者把 apk 的源码放在最新的 SDK 代码中重新编译即可。

Device driver->network device support->universal TUN/TAP device driver support 这个选项打开，重现编译生成内核即可

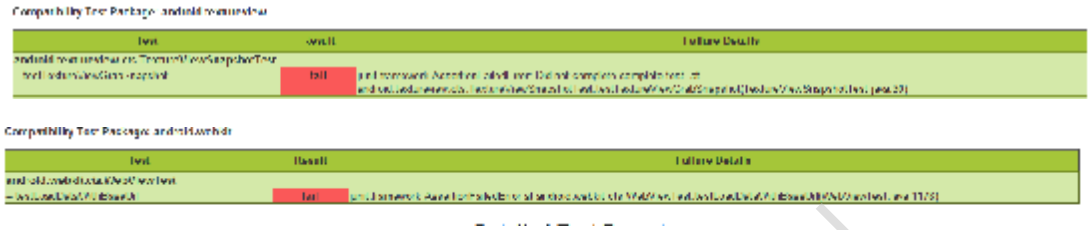
这个要打上补丁 testBrowserPrivateDataAccess 补丁



这个如果是 30 的机器，要打上发布的 v1.1 补丁，如果是 29 的机器则要打上 RK29_R3_GPU 这个补丁。



Five 碰到这两个问题是因为在 kernel 中打印了太多的 log，导致 codec 输出变慢，最后导致测试用例超时。把 kernel 中多余的 log 去掉就可以了。



这两个测试是谷歌测试用例的问题，可以向谷歌申请豁免

USB Accessory Test of the CtsVerifier.apk 的这个测试不能通过，因为 CtsVerifler.apk 中对 usb 的 pid 和 uid 做了限制，所以必须修改 init.rk29board.usb.rc 中

- write /sys/class/android_usb/android0/idVendor 2207

- write /sys/class/android_usb/android0/idProduct 0004

+ write /sys/class/android_usb/android0/idVendor 18D1

+ write /sys/class/android_usb/android0/idProduct 2D00

即可通过测试。

对于机器中不支持的 Sensor，要在 frameworks/base/data/etc/handheld_core_hardware.xml 中，把没有的 feature 去掉，例如 campass 等等。对于 sensor，出了这个地方，还要修改 hardware 中 sensor 的定义 hardware/rk29/sensor/st/sensors.c 中 sSensorList 的定义，把没用的去掉。