Widevine L3

开发指南

文档版本　　00B03

发布日期　　2015-10-31

商标声明

 、HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市海思半导体有限公司

地址：　深圳市龙岗区坂田华为基地华为总部　邮编：518129

网址：　http://www.hisilicon.com

客户服务邮箱：　support@hisilicon.com

前 言

概述

本文档主要介绍海思 Widevine L3 的工作原理以及开发过程及注意事项。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3798M	V1XX
Hi3796M	V1XX
Hi3798C	V2XX
HiSTBAndroid	V600R001C00SPC060
HiSTBAndroid	V600R002

读者对象

本文档（本指南）主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师

作者信息

章节号	章节名称	作者信息
全文	全文	W00269678

修订记录

修订日期	版本	修订说明
2015-03-13	00B01	第 1 次临时发布。
2015-08-11	00B02	删除 2.2.2 章节。
2015-10-31	00B03	修正 Keybox 相关章节描述。
2016-05-11	00B04	新增第 3 章“测试场景”。

目 录

前 言 iii

1 概 述 1-3

1.1 Widevine 简介 1-3
1.2 Widevine 整体框架 1-3
1.3 工作原理 1-3
1.4 安全等级 1-3

2 开发指引 2-3

2.1 开发流程 2-3
2.1.1 Widevine L3 开发流程 2-3
2.2 环境配置 2-3
2.2.1 安全启动配置 2-3
2.3 固件安装 Keybox 2-3
2.3.1 概述 2-3
2.3.2 Keybox 定义 2-3
2.3.3 Keybox 获取请求及传送协议 2-3
2.3.4 固件安装 Keybox 2-3
2.3.5 获取 Device ID 2-3

3 测试场景 3-3

3.1 GTS 测试 3-3
3.2 ExoPlayer 播放器测试 3-3
3.3 其他网络播放器 3-3

插图目录

图 1-1 Android 系统 Widevine 架构 1-3
图 1-2 Widevine 的工作原理 1-3
图 2-1 widevine 开发流程图 2-3
图 2-2 Keybox xml 文件 2-3
图 2-3 固件安装简要示意图 2-3
图 2-4 Keybox 无效进行固件安装示意图 2-3
图 3-1 GTS 测试环境拓扑图 3-3

1 概 述

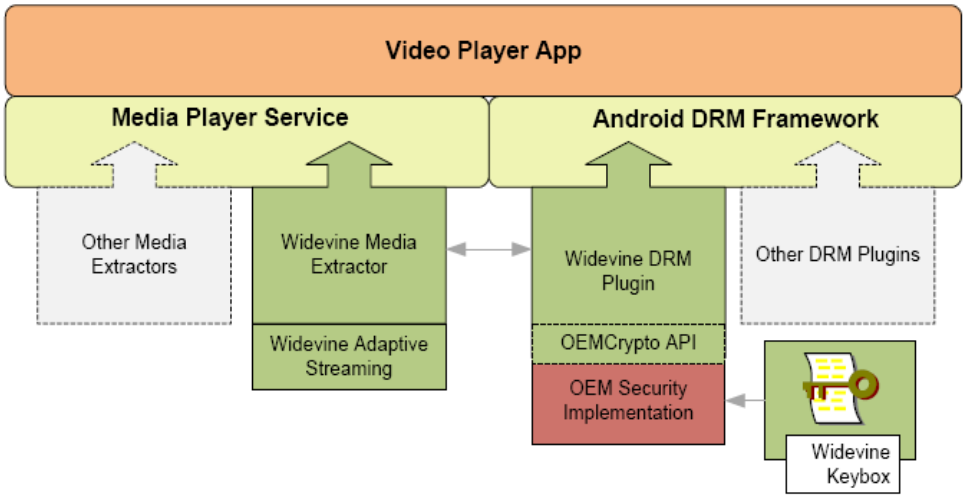
1.1 Widevine 简介

Widevine 是美国的一家专门提供流媒体数字版权保护（DRM）技术的公司，该公司的 DRM 技术被广泛地应用于数字流媒体领域，例如在线视频、数字电视等等。2010 年 9 月 3 日，谷歌收购了此公司，意图拓展自己的数字流媒体电影服务以及获得其 DRM 保护技术。

1.2 Widevine 整体框架

从 Android3.0 开始，Google 对 Drm 做了较大的增强，新增加了 Drm 框架，而且谷歌已经将 Widevine 功能集成在了 DRM 框架中。Widevine 在 Android 平台中的具体位置如图 1-1 所示。

图1-1 Android 系统 Widevine 架构



Android 平台的 Widevine 由主要是由两个组件构成：

- Widevine Media Extractor

  该模块主要是用于对流媒体的适配以及对加密的流媒体的解析操作。

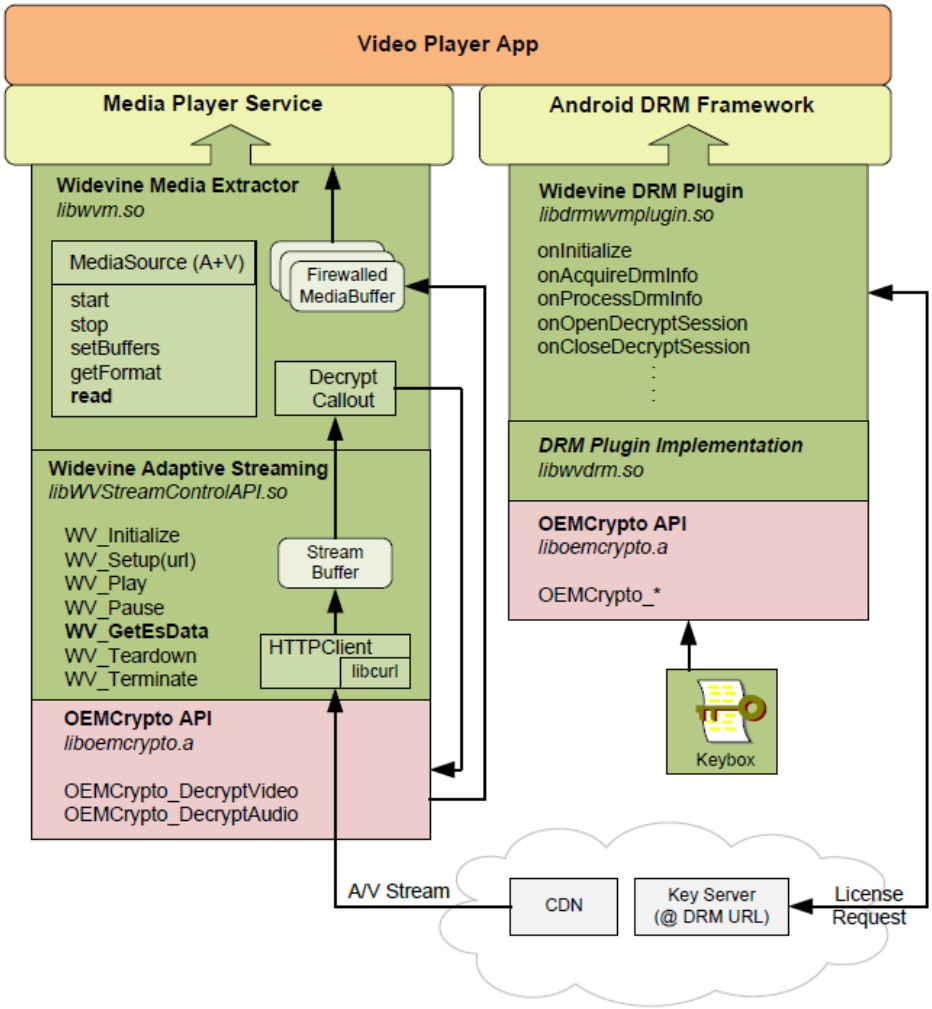
- Widevine DRM 插件

  该模块是作为 Android Framework 的一个插件，主要用于密钥管理、权限认证以及加密流媒体的解密。

1.3 工作原理

Widevine 的工作原理如图 1-2 所示。

图1-2 Widevine 的工作原理



Widevine 工作步骤如下：

步骤 1　Widevine Plugin 负责密钥管理，播放前，需要向 DRM server 申请 license，并获得认证通过。

步骤 2 认证通过后，媒体服务向内容服务器下载加密的内容，并通过 Widevine 解析器将加密码流解析出来，通过 Widevine plugin 解密后送到播放器播放。

加密的码流通过 HTTPClient 和 Stream Buffer 送到 Widevine Media Extractor，然后调用 Widevine plugin 中的 OEMCrypto API 接口对码流进行解密，最后送到播放器播放。

----结束

1.4 安全等级

由于 Widevine 的 DRM 技术需要硬件的支持，由于现有设备并非都具备该技术所需的硬件设备支持，故 Widevine 指定了三个安全等级，区别如表 1-1 所示。

表1-1 谷歌 Widevine 安全等级

安全等级	安全启动	Widevine keybox 装配	需要安全硬件 Trustzone	Widevine keybox 及视频密钥处理	硬件视频路径
Level 1	是	工厂安装	是	密钥不以明文暴露给 CPU	视频流通过硬件保护，输出在 TEE 中
Level 2	是	工厂安装	是	密钥不以明文暴露给 CPU	解码器直接获得明文视频流
Level 3	是	软件安装	否	密钥以明文暴露给 CPU	解码器直接获得明文视频流

- 根据安全要求，需要安全启动，这部分需要芯片厂商或者 OEM 厂商自行定制，满足安全即可。
- Widevine 定义了能够与底层安全硬件进行交互的硬件抽象层 OEMCrypto API。对于不同安全等级，与底层进行的硬件交互也有所不同，故 OEMCrypto API 会随之变化。
- Widevine L1 要求硬件级别的视频通路保护，需要 OEM 厂商自行定制，一般带防火墙的视频缓冲，或者其他硬件保护方案，有硬件 Keyladder 或者软件运行 Trustzone 中，达到隔离保护作用。

谷歌建议厂商新生产的设备实现 1 级安全等级。通过固件升级实现的 3 级安全等级只推荐那些未在产线上预置 Widevine Keybox 的遗留设备。

目前海思的 Android 版本 Widevine 只支持 3 级安全，本文档主要是针对海思 Android 版本 Widevine level3 的开发指导。

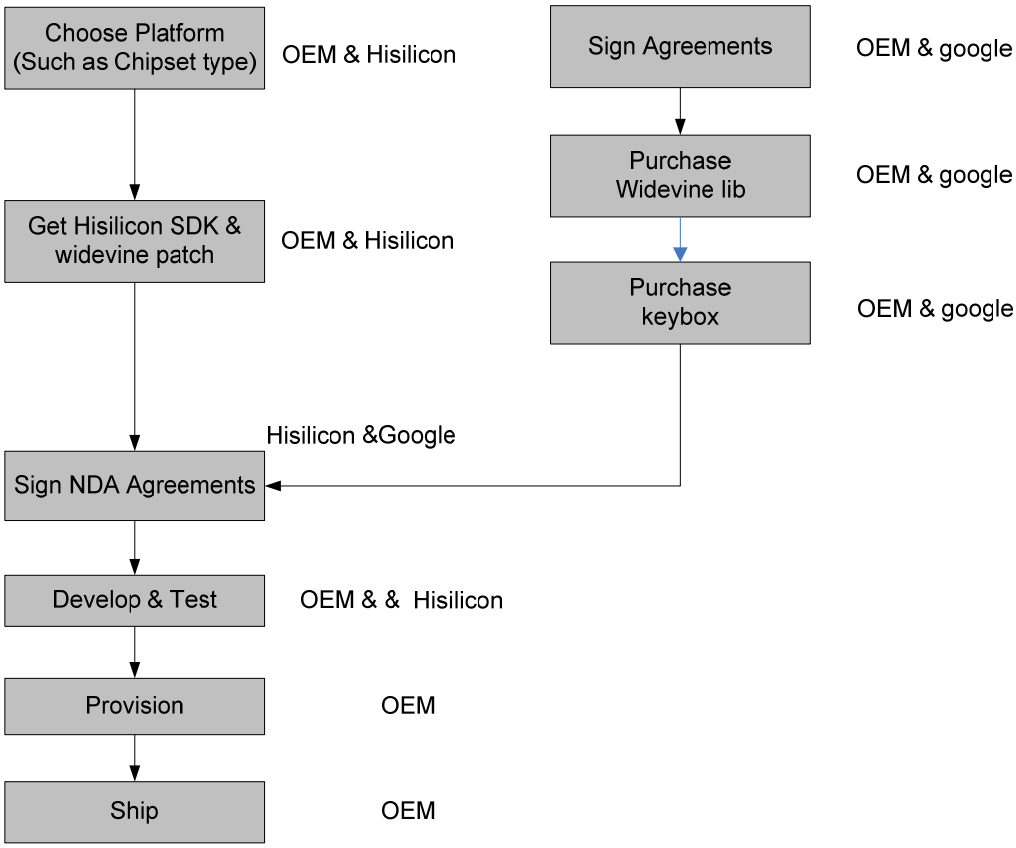
2 开发指引

2.1 开发流程

2.1.1 Widevine L3 开发流程

Widevine L3 开发流程如图 2-1 所示。

图2-1 widevine 开发流程图



开发步骤如下：

步骤 1 OEM 厂商首先向 google 签订 IDA 协议，并获取 Widevine 开发包。

说明

IDA 协议：Integration and Distribution Agreement for Device Manufacturers， 即产品集成发布协议。google 只会与 OEM 厂商签订，不会直接与海思签订，因为海思是芯片厂商，不直接出货。

步骤 2 OEM 厂商确定使用的芯片和 SDK 版本，并在步骤 1 完成后向 Hisilicon 获取 SDK 和 Widevine patch。

步骤 3 向 Google 申请用于量产的 Keybox。

步骤 4 OEM 厂商提供自己的名称，Hisilicon 确认 OEM 厂商是否跟 Google 签署过 IDA 协议。

步骤 5 OEM 厂商拿到 Keybox 和 Hisilicon 支持 Widevine 的版本 SDK 后，与 Hisilicon 集成开发。

步骤 6 需要通过 Google 提供的在线码流测试，Google 的 GTS 测试等，发布软件。

步骤 7 OEM 产线生产和出货。

----结束

2.2 环境配置

2.2.1 安全启动配置

Widevine 需要支持 boot 的安全启动，而 boot 的安全启动在海思 2 级安全方案中已经实现，所以只需要编译出 2 级安全方案的版本即可。具体步骤请参考文档《海思智能机顶盒 2 级安全方案 使用指南》中编译配置一节。

2.3 固件安装 Keybox

Widevine L3 是通过固件安装的方式安装 Keybox 的，即在向 DRM Server 申请 license 时自动安装的，不需要产线装备 Keybox。

2.3.1 概述

Widevine Keybox 安装在设备中用于与设备建立一种绝对信任，从而实现对设备上内容的保护。当 keybox 安装之后，设备上的安全硬件能够用于对 keybox 内容进行保护。当解密设备上播放的媒体内容时，需要使用到 Keybox 中的 device key 。

Keybox 可以通过固件安装或者产线安装的形式预置进设备中。固件安装形式只适用于实现三级安全等级的那些没有在产线上预置 keybox 的遗留设备，或者那些没有安全硬件来保护密钥的设备。一级及二级安全等级一定要在产线上进行装配，并且 keybox 要用写入芯片的唯一 AES 设备密钥进行加密后存储于不可擦除的 flash 区域。

每个 Widevine keybox 与一个 device Id 关联起来。谷歌规定每个设备应该有一个唯一的识别 ID。对于固件安装的形式，device id 应能通过相应的接口功能来获取。

除了 Device Id，在 keybox 中还有一个 Widevine 分配的 system id 用来保证 keybox 在不同厂商之间的唯一性。由于 system id 的唯一性，所以允许两个不同厂商使用相同的 device id。谷歌根据厂商提供的 keybox 请求中的厂商信息及设备型号信息来分配 system id。对于固件安装的设备，设备上的固件安装客户端负责产生相应的 system id。

2.3.2 Keybox 定义

Widevine Keybox 包含了一个唯一的 device ID，device key，加密密钥数据以及两个用于校验 Keybox 有效性的字段：常数字段及 CRC。如表 2-1 所示。

表2-1 Keybox 定义

Field	Description	Size (bytes)
Device ID	C character string identifying the device, null terminated.	32
Device Key	128 bit AES key assigned to device, generated by Widevine.	16
Key Data	Encrypted data	72
Magic	Constant code used to recognize a valid keybox: “kbox” (0x6b626f78)	4
CRC	CRC-32-IEEE 802.3 validates integrity of the key data field	4
	Total Size	***128***

2.3.3 Keybox 获取请求及传送协议

所有在设备厂商与 Widevine 之间传递的关于 keybox 请求及相关文件必须通过 PGP 技术进行加密。

厂商通过发送 email 给 widevine-keyboxrequest@google.com 来请求 keybox。

2.3.3.1 Keybox 请求 Email 格式

厂商提供包含以下 keybox 请求信息：

- 请求信息正文：
  - Device Manufacturer：公司名字
  - Device Model：产品型号
  - Number of keyboxes：请求的 keybox 数量
  - Date：格式 mmddyyyy
  - Contact Email：有效的 email 地址

  例如，XYZ 公司在 12/25/2009 为设备型号为 BD1234 的产品请求 2 个 keybox 将在 email 中包含如下的信息
  - Device Manufacturer：XYZ
  - Device Model：BD1234
  - Number of keyboxes：2
  - Date：12252009
  - Contact Email：contact@xyz.com

- Device IDs 文件：

  包含 Device ID 信息的文件必须用 PGP 进行加密并附在请求 email 中。

  文件名的格式为 MFGR_MODEL_DATE_#OFIDS.ids。文件中必须是每个 Device Id 一行，每个 device id 需要包含一个字母数字形式的字符串。最大的 device id 长是 31 个字节。

  例如，厂商 XYZ 将附上一个名为 XYZ_BD1234_12252009_2.ids，并使用设备序列号作为 device id 的一部分：

  - XYZ_BD1234_808KVJH008324
  - XYZ_BD1234_808KVJH008325

2.3.3.2 Keybox 回复文件

在设备厂商请求了 keybox 之后，Widevine 会产生相应数量的 keybox 并置于一个 XML 格式的文件中。这个文件的文件名格式为 MFGR_MODEL_DATE_#OFIDS.keybox，例如 XYZ_BD1234_12252009_2.keybox。keybox 文件通过 PGP 加密后回复给厂商。

2.3.3.3 Keybox XML 文件格式

Keybox 文件的例子如图 2-2 所示。

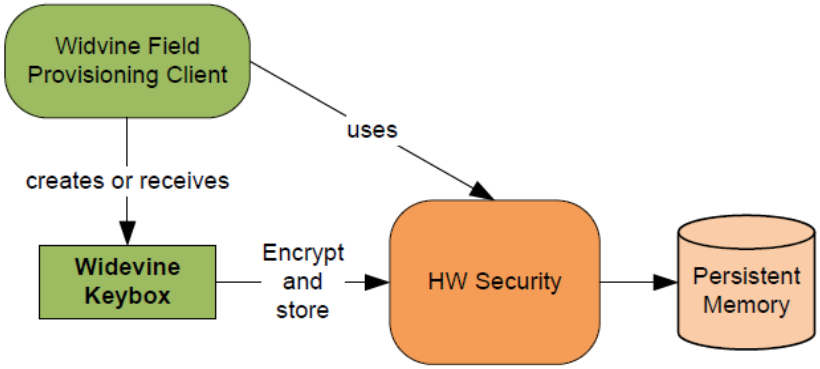
图2-2 Keybox xml 文件

```
<?xml version="1.0"?>
<Widevine>
<NumberOfKeyboxes>2</NumberOfKeyboxes>
<Keybox
DeviceID="mfg_mod123_0000001"><Key>c5f5cf3c2cb2ce175f2f5337a2f8f8ab</Key>
<ID>9d56e4931762b52aa21e4e590df477b5c81c683e0579f041ffa21f875c4c5e4a1cd4c2331
e27e3f4a49352fb432557336f63b1cb62549fddc9224b84d0c0364c827365fc217d9cb0</ID>
<Magic>6b626f78</Magic>
<CRC>0b11b841</CRC>
</Keybox>
<Keybox
DeviceID="mfg_mod123_0000002"><Key>73e38eb4f313e4fce8a5ab547cc7e2c0</Key>
<ID>215a40a9d13da3a9648335081a182869cbe78f607ce3ceb7506f351a22f411ae3f324ab5f
5bfb7c542ffcd38ec09438e7f92855149b02921463153c441332d7a21f875c4c5e4a1cd </ID>
<Magic>6b626f78</Magic>
<CRC>2b4c5e9f</CRC>
</Keybox>
</Widevine>
```

2.3.4 固件安装 Keybox

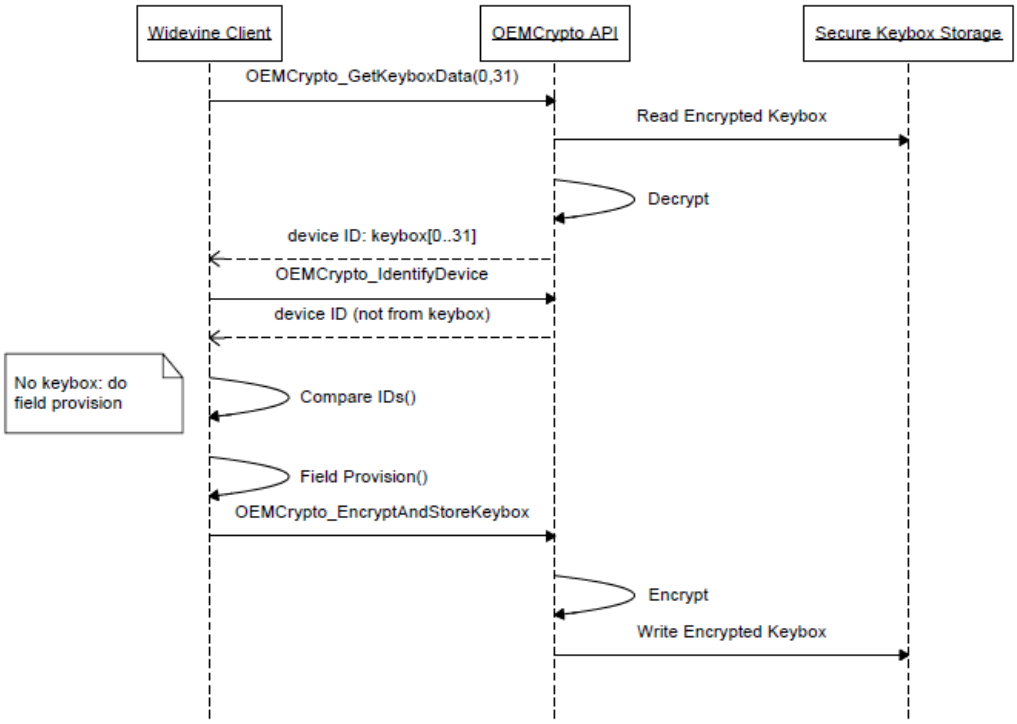
在固件安装的形式中，Widevine 固件安装客户端可以产生或接收一个 Widevine keybox，然后通过 OEMCrypto API 加密并存储在一个永久存储区，如图 2-3 所示。

图2-3 固件安装简要示意图



当 Widevine 固件安装客户端在设备上激活了以后，它首先会验证是否加载了一个有效的 keybox。通过检查 device id，magic kebox identifier（“kbox”）和 32 位 CRC 校验码来鉴别 keybox 是否有效。如果上述的任何一个字段被检测是无效的，那么设备会初始化一个固件安装操作，如图 2-4 所示。

图2-4 Keybox 无效进行固件安装示意图



2.3.5 获取 Device ID

由于每个客户的 Device ID 不同，需要 OEM 厂商自己实现获取 Device ID 的接口，接口具体位置在如下文件函数中：

device/hisilicon/bigfish/hidolphin/component/drm/source/widevine/proprietary/hisilicon/liboemcrypto/OEMCrypto.c

函数 HI_S32 HI_ADP_GetDeviceID(HI_U8 au8deviceid[32]);

3 测试场景

对 widevine L3 的测试场景主要分为 3 部分：GTS 测试，ExoPlayer 播放器测试，其他网络播放器测试。

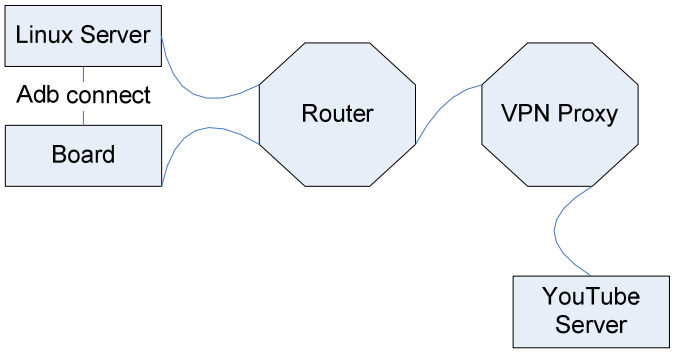
3.1 GTS 测试

GTS: GMS test suite，是 Google 提供的一套自动化的测试套，包括视频测试（含分辨率和码率），目前测试的版本是 gts 3.0_r2。GTS 测试流程如下：

步骤 1 设置好路由器的 VPN 翻墙环境，最好有比较高的网速，比较稳定的网络连接，并且能够访问 www.youtube.com；

步骤 2 将 Linux 服务器以及单板连接到已经翻墙的路由器，如图 3-1 所示。

图3-1 GTS 测试环境拓扑图



步骤 3 看单板和 youtube 是否能连上 ping www.youtube.com；

步骤 4 看单板和 Linux 服务器是否连通；

步骤 5 服务器通过 adb 连接到单板，例如单板 IP 为 192.168.001.010，执行：

```
adb connect 192.168.1.10
```

步骤 6 拷贝 GTS 测试包到服务器并解压到一个目录；

步骤 7 在服务器上找到 gts-3.0_r2/android-xts/tools 下的 xts-tradefed 工具并执行：

```
cd gts-3.0_r2/android-xts/tools
./xts-tradefed
run xts --plan XTS
```

说明

其他测试命令：

- 一次执行某个包(以 goole.media 为例)

  ```
  run xts -p google.media
  ```

- 某次执行一个方法

  ```
  run xts -c <class name> -m <method name>
  ```

步骤 8 测试结果 xtsTestResult.xml 会生成在 gts-3.0_r2/android-xts/repository/results/<start time>.zip 中。

----结束

说明

我们只需要关注 google.media 这个测试包中的测试项是否通过，因为只有 google.media 这些测试项会和 widevine 有关。

3.2 ExoPlayer 播放器测试

ExoPlayer 是一个开源的验证 DASH+CENC 码流的一个播放器，已经默认集成到 Widevine 的版本中。测试步骤如下：

步骤 1 设置好路由器的 VPN 翻墙环境，最好有比较高的网速，比较稳定的网络连接，并且能够访问 www.youtube.com；

步骤 2 打开 ExoPlayer apk，依次点击播放 WIDEVINE GTS DASH 列表中的 widevine 码流播放即可。

----结束

3.3 其他网络播放器

客户场景下可能用的是其他支持 widevine 码流的播放器，播放器直接调用标准 widevine plugin 的接口实现，这种场景也是支持的，测试步骤如下：

步骤 1 设置好路由器的 VPN 翻墙环境，最好有比较高的网速，比较稳定的网络连接，确认具有访问码流的权限（比如会员权限等）；

步骤 2 打开播放器，选择要观看的码流，点击获取权限。（此步骤需视情况而定，有的播放器不需要）；

步骤 3 点击播放即可观看。

步骤 4 如果播放不出来，首先确定 GTS 播放码流是否 OK。如果 GTS 播放 OK 但通过 APK 还是播不出来的话，需要跟 APK 提供商或者运营商确认 APK 版本，账户 ID 权限等是否正确。

----结束