Widevine L3

Development Guide

Issue	**00B04**
Date	**2016-05-11**

Trademarks and Permissions

 , HISILICON, and other HiSilicon icons are trademarks of HiSilicon Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between HiSilicon and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

HiSilicon Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: http://www.hisilicon.com

Email: support@hisilicon.com

About This Document

Overview

This document introduces the working principle, development process, and precautions for HiSilicon Widevine L3.

Related Versions

The following table lists the product versions related to this document.

Product Name	Version
Hi3798M	V1XX
Hi3796M	V1XX
Hi3798C	V2XX
HiSTBAndroid	V600R001C00SPC060
HiSTBAndroid	V600R002

Intended Audience

This document is intended for:

- Technical support engineers
- Software development engineers

Change History

Changes between document issues are cumulative. Therefore, the latest document issue contains all changes made in previous issues.

Issue 00B04 (2016-05-11)

This issue is the fourth draft release, which incorporates the following changes:

Chapter 3 is added.

Issue 00B03 (2015-11-02)

This issue is the third draft release, which incorporates the following changes:

Sections about Keybox are modified.

Issue 00B02 (2015-08-11)

This issue is the second draft release, which incorporates the following change:

Section 2.2.2 is deleted.

Issue 00B01 (2015-03-13)

This issue is the first draft release.

Contents

Figures

1 Overview

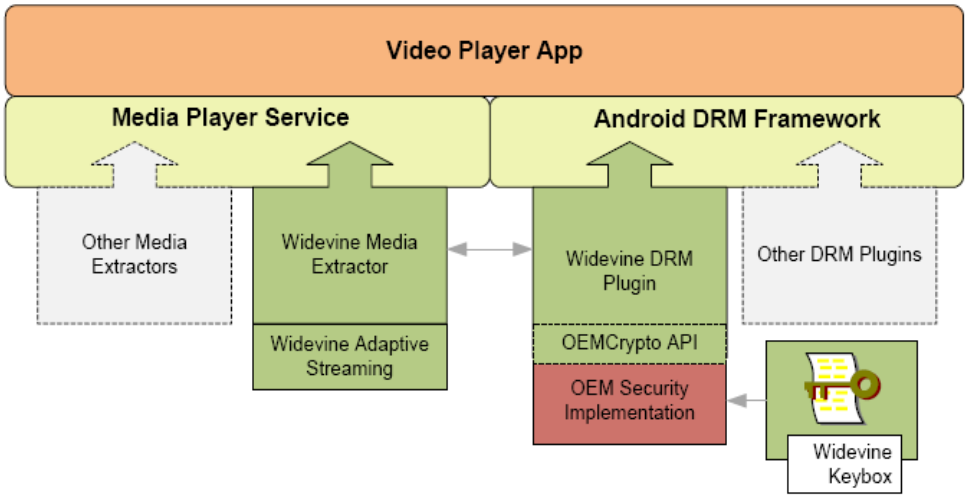
1.1 Widevine Introduction

Widevine is an American company that is dedicated to providing the technology of streaming media Digital Rights Management (DRM), which has been widely used in the digital streaming media field, such as the online video and digital TV. On September 3, 2010, Google purchased the company, aiming to expand the digital streaming media movie services and obtain the DRM technology.

1.2 Widevine Overall Framework

From Android3.0, Google has greatly enhanced the DRM with added DRM framework, which is provided with integrated Widevine functions. Figure 1-1 shows the detailed location of the Widevine in the Android platform.

Figure 1-1 Widevine structure of the Android system



The Widevine of the Android platform mainly consists of the following components:

- Widevine Media Extractor

This module is mainly used to adapt to the streaming media and parse the encrypted streaming media.

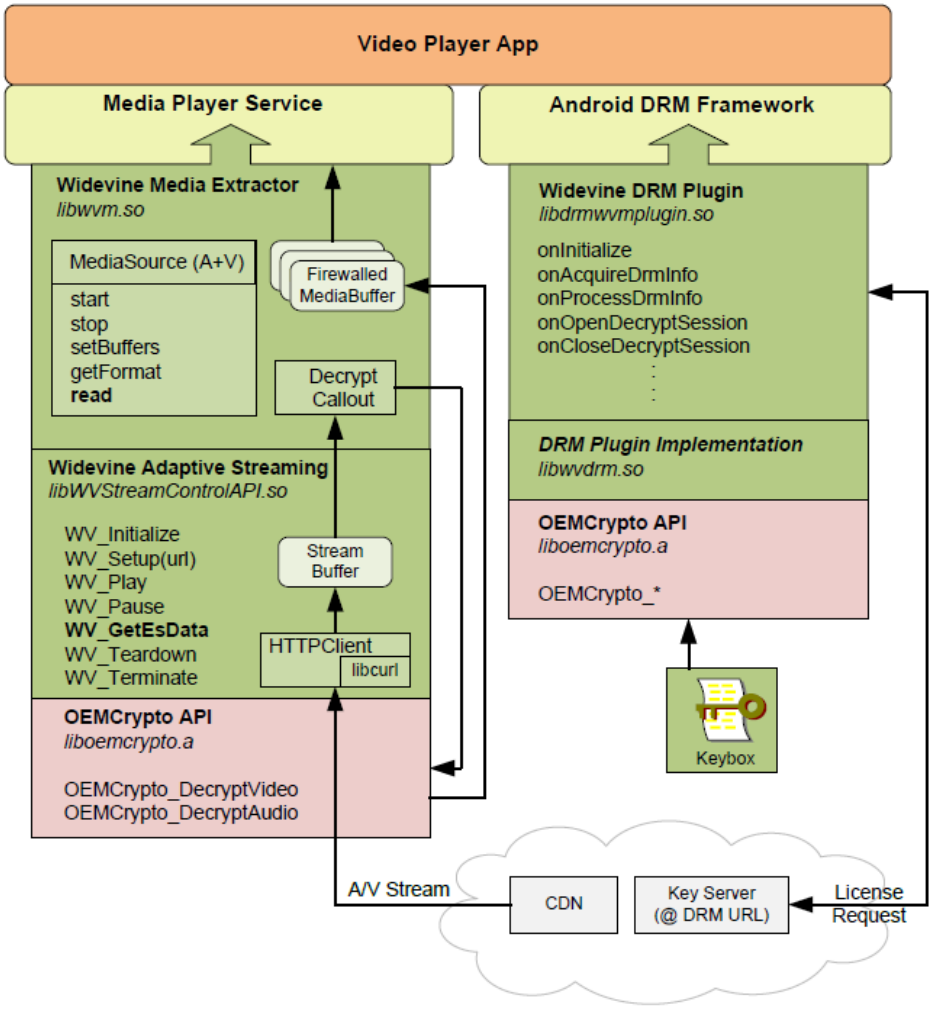
- Widevine DRM plug-in

  This module is used as a plug-in of the Android Framework to manage keys, approve permissions, and decrypt the encrypted streaming media.

1.3 Working Principle

Figure 1-2 shows the working principle of Widevine.

Figure 1-2 Working principle of Widevine



The working procedure of Widevine is as follows:

Step 1 The Widevine plug-in is responsible for key management. Before playback, a license must be obtained from and authenticated by the DRM server.

Step 2 After the license is authenticated, the media server downloads the encrypted contents from the content server, parses the encrypted code through the Widevine parser, and sends the contents to the player after the Widevine plug-in decrypts the contents.

The encrypted code stream is sent to the Widevine media extractor through the HTTP client and stream buffer. Then, the OEMCrypto API in the Widevine plug-in is invoked to decrypt the stream. Finally, the steam is sent to the player.

----End

1.4 Security Level

The DRM technology of Widevine needs hardware support. However, not all current devices have the required hardware devices to provide the support. Therefore, the Widevine designated three security levels, as shown in Table 1-1.

Table 1-1 Google Widevine security levels

Security Level	Whether the Boot can be securely loaded	Widevine Keybox Assembly	Whether Security Hardware or Trust Zone Is Required	Widevine Keybox and Video Key Processing	Hardware Video Path
Level 1	Yes	Installed in factory	Yes	Keys are not disclosed to CPU in plaintext.	Video streams are protected through hardware and output in the TEE.
Level 2	Yes	Installed in factory	Yes	Keys are not disclosed to CPU in plaintext.	The decoder directly obtains the video streams in plain text.
Level 3	Yes	Installed through software	No	Keys are disclosed to CPU in plaintext.	The decoder directly obtains the video streams in plain text.

- According to the security requirements, secure boot is required. This must be customized by chip or ORM manufacturers to meet the security requirements.
- Widevine has defined the hardware abstraction layer OEMCrypto API which interacts with the bottom layer security hardware. The hardware varies with different security levels. Therefore, the OEMCrypto API changes with the hardware.
- Widevine L1 requires hardware-level video channel protection, which needs to be customized by the OEM manufacturers. Generally, video cache with firewalls, or other hardware protection strategy with key ladder or software running in the Trustzone can be used to implement isolation and protection.

Google suggests manufactures to implement level 1 security on newly produced devices. Level 3 security implemented through firmware upgrade can only be used to legacy devices which have not been preset with the Widevine Keybox on the production line.

Currently, HiSilicon's Android Widevine only supports level 3 security. This document provides guidance for the development of HiSilicon Android Widevine L3.

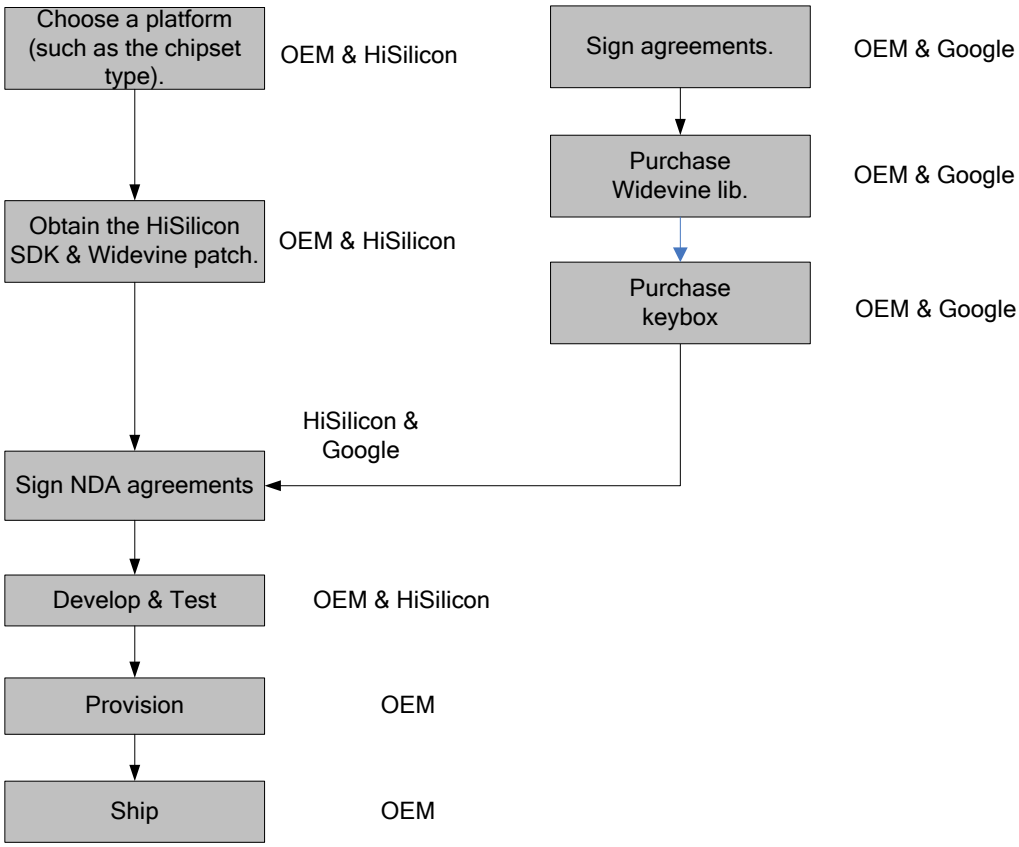
2 Development Guide

2.1 Development Procedure

2.1.1 Widevine L3 Development Procedure

Figure 2-1 shows Widevine L3 development procedure

Figure 2-1 Widevine development procedure



The development procedure is as follows:

Step 1 The OEM manufacturer signs the IDA agreement with Google and obtains the Widevine development package.

NOTE

IDA is short for Integration and Distribution Agreement. Google only signs this agreement with OEM manufactures other than HiSilicon because HiSilicon is a chip manufacturer with no deliveries.

Step 2 The OEM manufacturer determines the chip and SDK version to be used, and obtains the SDK and Widevine patch from HiSilicon.

Step 3 The OEM manufacturer applies to Google for Keybox for mass production.

Step 4 HiSilicon confirms with Google to check whether the OEM manufacturer signs the IDA agreement with Google.

Step 5 After obtaining the Keybox and the HiSilicon SDK that supports Widevine, the OEM manufacturer implements integrated development with HiSilicon.

Step 6 The software is release after it survives the Google-provided online code stream test and Google Mobile Services Test Suite (GTS) test.

Step 7 The OEM produces and delivers the software.

----End

2.2 Environment Configuration

2.2.1 Configuring the Secure Boot

Widevine needs to support secure boot, which has been realized in HiSilicon level 2 security solutions. Therefore, you only need to compile the version for level 2 security solutions. For detailed procedures, see the compilation configuration in the *Level 2 Security Solution for the HiSilicon Intelligent STB User Guide*.

2.3 Keybox Firmware Installation

The Keybox is installed in Widevine L3 in firmware installation mode. That is, the Keybox is automatically installed when the OEM manufacturer applies to the DRM Server for the license. The production line does not need to install the Keybox.

2.3.1 Overview

The Widevine Keybox is installed in the device to establish absolute trust with the device, to protect the contents on the device. After the Keybox is installed, the security hardware on the device can be used to protect the Keybox contents. To decrypt the media to be played on the device, the device key in the Keybox is required.

The Keybox can be preset into devices through firmware or the product line. The installation through firmware is intended to realize level 3 security on legacy devices with no Keybox preset through the production line, or devices with no security hardware to protect keys. Level 1 and 2 security must be installed on the production line. In addition, the Keybox must be encrypted by the unique AES device key written into the chips and stored into the flash area that cannot be erased.

Each Widevine Keybox must be associated with one device ID, which must be unique for each device as required by Google. For the installation through the firmware, the device ID can be obtained by using the corresponding interface.

In addition to the device ID, the system ID assigned by the Widevine is used to ensure the uniqueness of the Keybox for different manufacturers. Because of the uniqueness of the system ID, two manufacturers can use the same device ID. Google assigns system IDs based on the manufacturer information and device model in manufacturers' Keybox requests. For devices using firmware installation, the firmware installation client on the device is used to generate the corresponding system ID.

2.3.2 Definition

Widevine Keybox includes a unique device ID, device key, encryption key data, and two valid fields (constant field and CRC) used to verify the Keybox, as described in Table 2-1.

Table 2-1 Keybox definition

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 Obtaining Request and Transmission Protocol

All Keybox-related requests and files transferred between the device manufacturer and Widevine must be encrypted using the PGP technology.

The manufacturer sends emails to widevine-keyboxrequest@google.com to request for the Keybox.

Request Email Format

The manufacturer must provide the following Keybox request information:

- Request body:
  - Device Manufacturer
  - Device Model
  - Number of Keyboxes
  - Date: in the format of mmddyyyy
  - Contact Email: valid email address

  For example, if the XYZ company requests for two keyboxes for BD1234 on December 25, 2009, the following information must be included:
  - Device Manufacturer: XYZ
  - Device Model: BD1234
  - Number of keyboxes: 2

- Date: 12252009
- Contact Email: contact@xyz.com

- Device IDs files:

  Files with information of the device ID must be encrypted by PGP and attached in the email.

  The name of the file must be in the format of **MFGR_MODEL_DATE_#OFIDS.ids**. In the file, each line has one device ID, which contains one alphanumeric character string. The maximum length of a device ID is 31 bytes.

  For example, the XYZ company should attach a file named as **XYZ_BD1234_12252009_2.ids** and contain device serial number in the device ID:

  - XYZ_BD1234_808KVJH008324
  - XYZ_BD1234_808KVJH008325

Reply File

After the device manufacturer requests for the Keybox, the Widevine generates a corresponding number of Keyboxes and place them in one XML file named in the format of **MFGR_MODEL_DATE_#OFIDS.keybox**, such as **XYZ_BD1234_12252009_2.keybox**. The PGP-encrypted Keybox files are replied to the manufacturer.

XML File Format

Figure 2-2 shows the example of Keybox files.

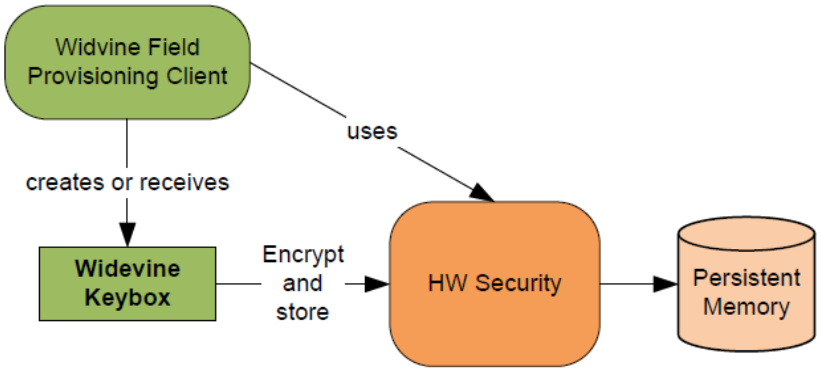
Figure 2-2 Keybox xml files

```
<?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 Installing the Keybox Firmware

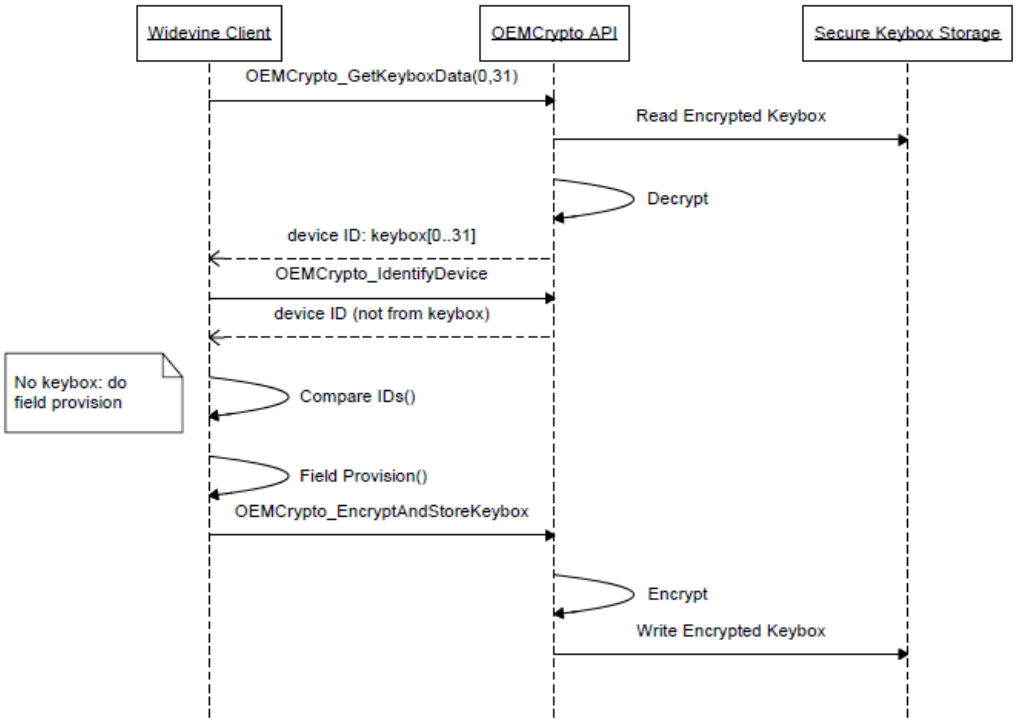
In the installation through firmware, the Widevine firmware installation client can generate or receive one Widevine keybox, encrypt it through OEMCrypto API, and store it in a permanent storage area, as shown in Figure 2-3.

Figure 2-3 Brief diagram of firmware installation



When the Widevine firmware installation client is activated on the device, the client verifies whether a valid Keybox is loaded by checking the device ID, magic keybox identifier ("kbox"), and 32-byte CRC verification code. If any of the aforesaid fields is invalid, the device initiates the firmware installation, as shown in Figure 2-4.

Figure 2-4 Diagram for firmware installation upon invalid Keyboxes



2.3.5 Obtaining the Device ID

Because customers have different device IDs, the OEM manufacturer needs to activate the interface to obtain the device ID. The specific location of the interface is shown in the following file function:

```
device/Hisilicon/bigfish/hidolphin/component/drm/source/widevine/proprietary/Hisili
con/liboemcrypto/OEMCrypto.c
Function HI_S32  HI_ADP_GetDeviceID(HI_U8 au8deviceid[32]);
```

3 Test Scenario

The test scenario for Widevine L3 is divided into three parts: GTS test, ExoPlayer test, and test of other network players.

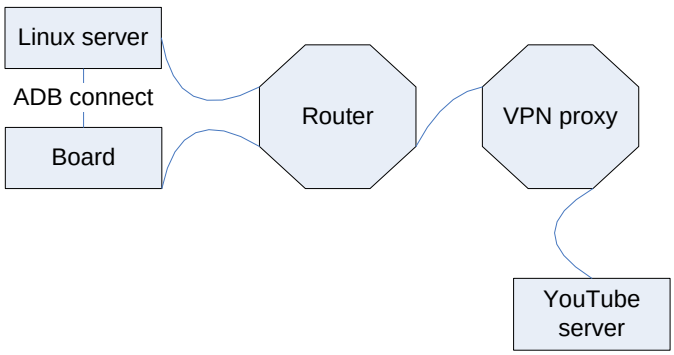
3.1 GTS Test

GTS is a automatic test suite provided by Google and contains the video test (including resolution and bit rate). The current test version is gts 3.0_r2. The GTS test process is as follows:

Step 1 Configure the virtual private network (VPN) proxy environment of the router. Ensure that the network speed is high, the network connection is stable, and the website www.youtube.com can be visited.

Step 2 Connect the Linux server and the board to the router, as shown in Figure 3-1.

Figure 3-1 Topology of the GTS test environment



Step 3 Check whether the board can connect to YouTube by running **ping www.youtube.com**.

Step 4 Check whether the board connects to the Linux server.

Step 5 Connect the server to the board by using the adb command. For example, if the board IP address is 192.168.001.010, run the following command:

```
adb connect 192.168.1.10
```

Step 6 Copy the GTS to the server and decompress it to a directory.

Step 7 Find the **xts-tradefed** tool in the **gts-3.0_r2/android-xts/tools** directory of the server, and run the following commands:

```
cd gts-3.0_r2/android-xts/tools

./xts-tradefed

run xts –-plan XTS
```

NOTE

Other test commands are as follows:

- Execute one test suite at a time (taking google.media as an example):

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

- Execute one method:

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

Step 8 The test result **xtsTestResult.xml** is generated in **gts-3.0_r2/android-xts/repository/results/<start time>.zip**.

----End

NOTE

You need only to pay attention to the test result of test items in the **google.media** test suite, because only these test items are relevant to Widevine.

3.2 ExoPlayer Test

ExoPlayer is an open-source player used to verify the DASH+CENC streams, and is integrated in the Widevine version by default. The process of the ExoPlayer test is as follows:

Step 1 Configure the VPN environment of the router. Ensure that the network speed is high, the network connection is stable, and the website www.youtube.com can be visited.

Step 2 Open the ExoPlayer APK, and click to play the Widevine streams in the WIDEVINE GTS DASH list one by one.

----End

3.3 Test of Other Network Players

Customers may use other players that support Widevine streams. In such scenarios, the player directly calls the standard Widevine plug-in interfaces. The test process of such network players is as follows:

Step 1 Configure the VPN environment of the router. Ensure that the network speed is high, the network connection is stable, and the permission (such as the membership permission) to access streams is granted.

Step 2 (Optional) Open the player, select the streams to be played, and click to obtain the permission. (This step is not required for some players.)

Step 3 Click the playback option to view the selected stream.

Step 4 If the stream cannot be played, check whether the stream can be played by using the GTS. If the stream can be played by using the GTS but fails to be played by using the APK, contact the APK provider or operator to check whether the APK version and account ID permission are correct.

----End