Android PlayReady

开发指南

文档版本　　00B02

发布日期　　2014-10-17

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 PlayReady 工作原理，海思 Android 平台上 PlayReady 的开发过程以及注意事项。

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
Hi3716C 芯片	V200
Hi3719C 芯片	V100
Hi3719M 芯片	V100
Hi3716M 芯片	V400
Hi3798M 芯片	V100

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

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

修订日期	版本	修订说明
2014-06-16	00B01	第 1 次版本发布。
2014-10-17	00B02	1. 更新 SW 版本配置文件说明 2. 工厂生产方案，证书加密算法更改为 AES-CTR 模式

目 录

插图目录

1 概述

1.1 PlayReady 简介

PlayReady 为微软公司的新的 DRM 系统，它是 WMDRM（Windows Media DRM）的升级产品，兼容 WMDRM，可以为数字媒体提供内容保护支持。

微软 PlayReady 官方网站为 http://www.microsoft.com/playready/，可以从该网站获取 PlayReady 产品、技术、文档、License 及支持等信息；PlayReady 官方测试网站为 http://playready.directtaps.net/，可以使用该网站的测试用例和码流对用户的 PlayReady 产品进行基本功能测试。

海思 Android PlayReady 解决方案支持 PIFF/Smooth Streaming 格式流媒体（文件名为 Manifest）的网页播放，支持 ASF 格式（扩展名 pyv/pya）媒体文件的本地播放。

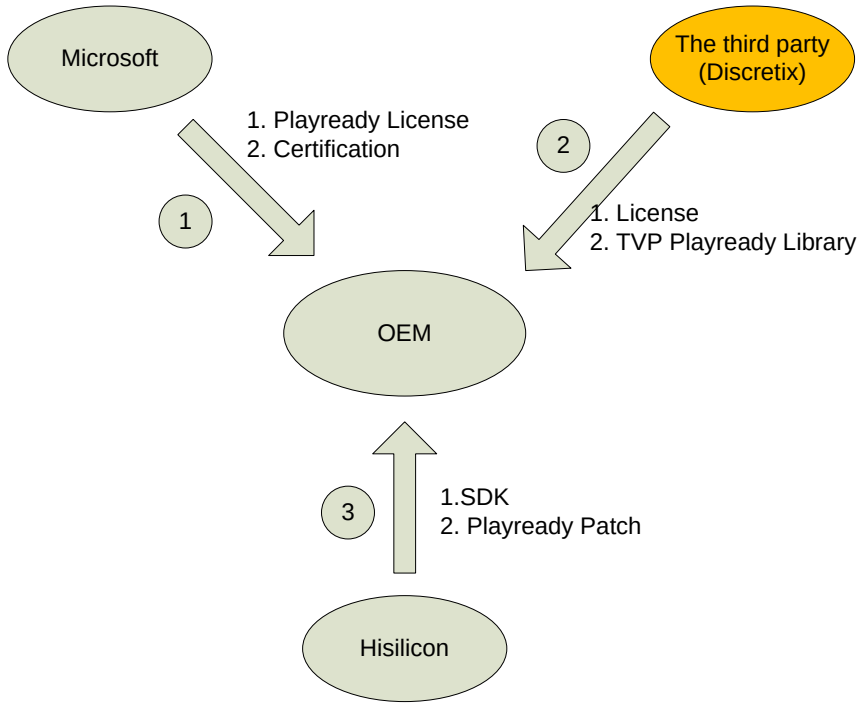
海思 PlayReady 方案分为 Hardware 版本（以下简称 HW 版本）和 Software 版本（以下简称 SW 版本）。SW 版本能为内容和证书提供基本和必要的保护，对芯片和方案没有特别要求，适用于内容提供商没有特殊要求的场景。HW 版本使用海思芯片的 Trustzone 硬件保护机制，对证书、加解密密钥以及解密后的码流提供更高级别的保护，HW 版本适用于内容提供商明确要求 Trustzone 特性或 Trust Video Path 的场景。

HW 版本 Playready 方案，为海思与第三方公司合作完成。第三方公司基于海思 Trustzone 平台提供符合 Trust Video Path 和 Android DRM Framework 要求的 playready 库和插件。OEM 厂商除了需要获取微软 Playready Final Product 授权，还需要获取第三方公司授权以及对应的 HW 版本的 Playready 库。目前的第三方合作公司为 Discretix (www.discretix.com)。

1.2 PlayReady 项目开发流程

1.2.1 项目关系图

图1-1 项目关系图

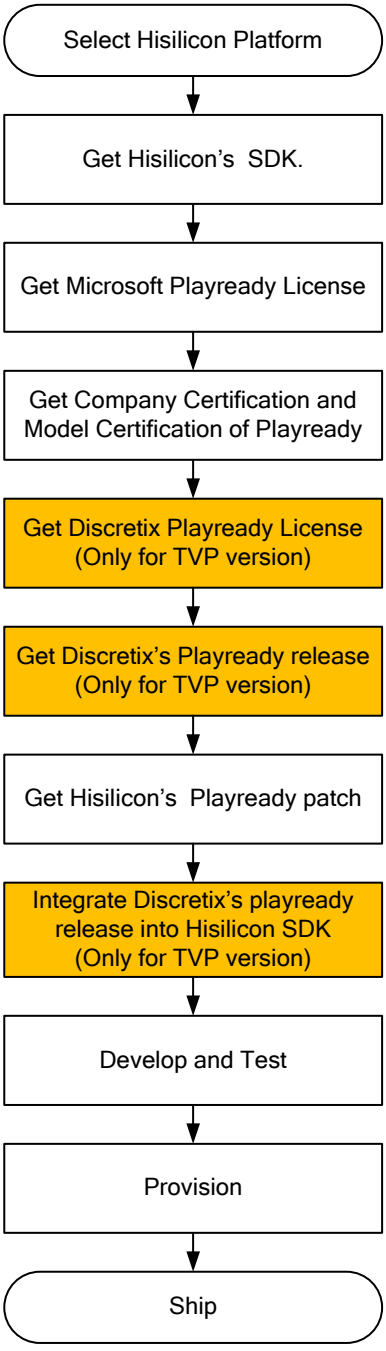


注意

SW 版本 PlayReady，海思提供的补丁包已包含所需库文件，客户只需要从微软获取 License 和证书；HW 版本 PlayReady，客户还需要获取第三方公司（Discretix）的 PlayReady 库文件和 License。

1.2.2 项目开发流程

图1-2 项目开发流程



步骤 1 选择海思平台，若是 HW 版本，请选择支持 Trustzone 特性的平台。

步骤 2 从海思获取支持 PlayReady 的 Android SDK。

步骤 3 从微软获取 PlayReady License 以及 PlayReady company 证书，并生成 Model 证书。

步骤 4 对于 HW 版本 PlayReady，还需要从 Discretix 获取 PlayReady license，并获取到与海思平台和 SDK 匹配的 PlayReady 发布包。

步骤 5 从海思获取与 SDK 匹配的 PlayReady 补丁包。

步骤 6 对于 HW 版本 PlayReady，将 Discretix 发布的 PlayReady 相关文件集成进海思 SDK。

步骤 7 参考 PlayReady Demo APK，OEM 厂商开发自己的 PlayReady 应用。

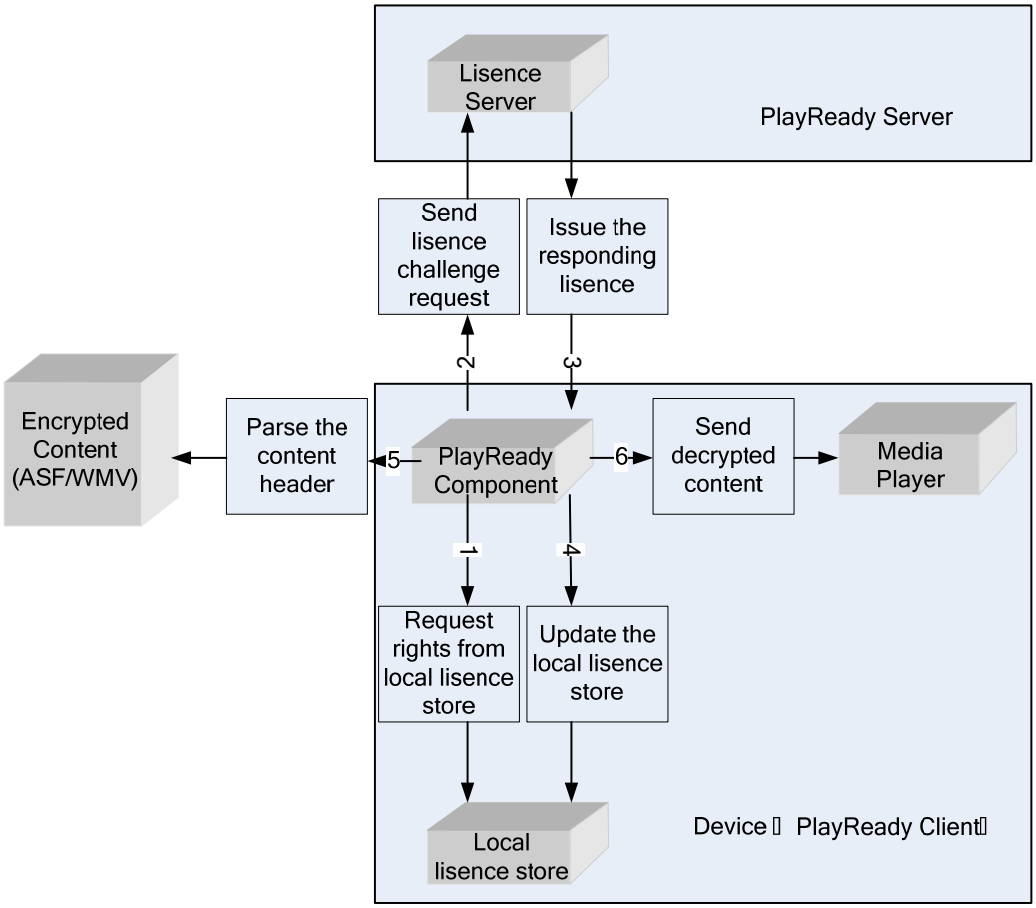
步骤 8 工厂生产，完成证书和密钥烧写。。

----结束

1.3 HW 版本 SW 版本 PlayReady 工作流程

PlayReady 的工作流程图如图 1-3 所示。

图1-3 PlayReady 工作流程



具体流程说明如下：

步骤 1 用户播放文件时，播放器向 PlayReady 组件查询该文件能否被播放。

步骤 2 PlayReady 组件在许可证库中搜寻是否有相应的有效许可证。

步骤 3 如果在许可证库中没有找到对应的许可证，PlayReady 组件会发送一个许可证请求至 License 服务器。

步骤 4 License 服务器返回包含内容解密密钥的许可证给客户端。

步骤 5 PlayReady 组件验证、解析许可证，获取解密密钥，将许可证放入许可证库。

步骤 6 PlayReady 组件解密数据包，将数据包发送给播放器解码播放。

----结束

1.4 设备证书

PlayReady 设备证书，用于码流 License 获取和解密，由 OEM 厂商向微软申请获得。

1.4.1 生成证书

注意

微软在协议中已经郑重申明：申请新的设备认证证书应该由 final product 公司提出申请（申请次数不限）。海思作为 Intermediate Product Distribution 只能够提出一次申请，并且只能是内部测试使用，不能分发给客户，否则违反 PlayReady 协议，所以，海思的客户所使用的 PlayReady 的证书应该由客户自己提出申请。

OEM 厂商需要先向微软申请 Company 证书，然后使用微软提供的工具和模板，用 Company 证书生成 Model 证书， Model 证书需要烧写到设备上用于码流 License 获取和解密。

由于 PlayReady 兼容 WMDRM，所以 PlayReady 的证书包括 WMDRM 证书。

证书申请和生成的具体方法，请参考 1.5 PlayReady 证书申请及使用 。

1.4.2 安全等级

微软对使用证书的等级做了不同的区分，非商业用途 PlayReady 证书的级别 security_level 为 150，商业用途 PlayReady 证书的级别 security_level 最高为 2000。

1.5 PlayReady 证书申请及使用

下面介绍 PlayReady 证书申请及使用流程：

步骤 1 首先根据微软 PlayReady 开发包中的 sample 工具 GenerateCompanyCertRequest.exe 生成 RequestFile 和 OEMPrivateKeyFile.xml，request 发送给微软，OEMPrivateKeyFile.xml 由申请者自己秘密保存。使用如下命令：

```
generatecompanycertrequest.exe -r:Request.xml -p:XXXOEMPrivateKeyFile.xml
-m:"Huawei Technologies Co."
```

申请表格请使用 Playready 协议中的 EXHIBIT B:"PlayReady Device Development and Intermediate Product Distribution License CERTIFICATE REQUEST FORM"，写入"Company Name"和"Effective Date and Agreement Number".

步骤 2 一般在十个工作日后，微软处理完成此证书申请请求，然后返回给申请者 DeviceCompanyCert.exe 文件。

步骤 3 解压 DeviceCompanyCert.exe 文件生成 NDResponse.crt，PDDACResponse.dat，PRDACResponse.dat 三个文件，仅需要使用后面两个文件。

生成 PlayReady 未签名模板 xml 文件 UnsignedTemplate.xml，注意必须保存为 UTF-16 little endian 格式。（可以使用 windows 记事本来保存，未签名模板文件参考 PlayReady 开发包中的 PlayReady 开发文档 playReady.chm ，PlayReady Device Porting Kit v2.0 > Getting Started > Obtaining Certificates > Creating a Model Certificate > Creating the Unsigned Template > PlayReady and WMDRM-PD Device Template Sample）。或者直接使用如下的设置：

```
<UNSIGNEDTEMPLATE>
    <NAME>Hi3716</NAME>
    <MANUFACTURER>Hisilicon</MANUFACTURER>
    <MAKE>UnknownMake</MAKE>
    <DISTRIBUTOR>WideWorldImporters</DISTRIBUTOR>
    <MODEL>Hi3716X</MODEL>
    <SECURITYLEVEL>2000</SECURITYLEVEL>
    <HARDWARE_VER_MAJOR>2</HARDWARE_VER_MAJOR>
    <HARDWARE_VER_MINOR>1</HARDWARE_VER_MINOR>
    <FIRMWARE_VER_MAJOR>1</FIRMWARE_VER_MAJOR>
    <FIRMWARE_VER_MINOR>3</FIRMWARE_VER_MINOR>

    <FEATURES>
        <CLOCK>2</CLOCK>
        <SECURECLOCK>
            <URL>http://go.microsoft.com/fwlink/?LinkID=149408</URL>

<PUBLICKEY>!CNhvvz1WaNV1AFUmetxkvm9iD4UrE9cnGUi!qcqdxMiXmD1*ikYGA==</PUBL
ICKEY>
        </SECURECLOCK>
        <METERING>1</METERING>
        <LICENSE_ACQ>1</LICENSE_ACQ>
        <LICENSE_SYNC>1</LICENSE_SYNC>
        <ENCRYPTION>1</ENCRYPTION>
        <SYMMETRIC_OPT>1</SYMMETRIC_OPT>
        <SUPPORT_REVOCATION>0</SUPPORT_REVOCATION>
    </FEATURES>
```

```
   <LIMITS>
        <MAXCHAINDEPTH>2</MAXCHAINDEPTH>
       <MAXLICENSESIZE>10240</MAXLICENSESIZE>
     <MAXHEADERSIZE>5120</MAXHEADERSIZE>
</LIMITS>

</UNSIGNEDTEMPLATE>
```

注意

必须把模板中的以下字段去掉：

<TRANSMITTER>1</TRANSMITTER>

<RECEIVER>1</RECEIVER>

步骤 4 使用 sample 工具 generatemodelcert.exe ，键入如下命令：

```
generatemodelcert.exe -d:PDDACResponse.dat  -b:PRDACResponse.dat -
u:UnsignedTemplate.xml -f:OEMPrivateKeyFile.xml -t:devcerttemplate.dat -
g:bgroupcert.dat -h:zgpriv.dat  -k:priv.dat
```

说明

保证以上各个文件都在当前目录下面。

运行后，提示 success，即为生成成功。

生成文件：

- devcerttemplate.dat、priv.dat（为 WMDRM 证书和密钥）
- bgroupcert.dat、zgpriv.dat（为 PlayReady 证书和密钥）

至此，bgroupcert.dat、zgpriv.dat、devcerttemplate.dat、priv.dat 即为 PlayReady 设备证书。

----结束

注意

默认的设备证书安全等级为 150，这个等级最低，在 LS（License server）添加一些限制之后，就无法授权播放相关媒体了。所以需要申请相应安全等级的安全证书。

1.6 私钥与安全性

PlayReady 的保密性依赖于生成证书的公私钥对，公钥存放于设备证书中，私钥需要用户安全保管和存储。

注意

每个设备上的证书的存放需要各个厂商安全存放。

1.7 Compliance Rules and Robustness Rules for Microsoft PlayReady

微软规定，所有签订 PlayReady final product licenses 的公司必须要满足 Compliance Rules and Robustness Rules。

注意

Compliance Rules and Robustness Rules **中定义了不同** OPLs **时的音视频输出的限制，作为** final product license **必须要遵守，例如** HDMI **不能输出** OPLs **超过** 300 **的解密后的音视频内容 。**

此文档可以在 http://www.microsoft.com/PlayReady/Licensing/compliance.mspx **获得。**

1.8 安全时钟

在一些通过时间限制（例如：限制播放的开始日期和失效日期）来限制使用媒体内容的许可证权限中，获得一个安全的系统时钟就显得非常必要。在租赁和预定媒体内容的需求场景中，系统的安全时钟显得尤为重要。

Playready 组件通过连接微软的安全时钟服务来设置系统时间为安全时钟。当系统时间为可信任状态时，设备可以使用 PlayReady 组件来访问媒体内容。当系统时间为不可信任状态时，设备不允许使用 playready 组件来尝试播放和获取媒体内容的许可证。

一般情况下，设备会周期性的更新系统时钟为安全时钟。

1.9 Domains

PlayReady 允许一组设备作为一个 domain 的成员来播放所有绑定到这个 domain 的所有媒体。从而达到 domain 里面的媒体共享的目的。一个设备可以成为多个 domain 的成员。

通过加入一个 PlayReady domain，一个设备可以获取访问此 domain 里面的媒体内容，从而允许设备来播放绑定到此 domain 的所有媒体。

domain 服务管理者会限制加入到此 domain 的数量，因此，用户必须要把设备从一个 domain 中移除。用户在购买一个新的设备来替换旧的设备之前应该把旧的设备从 domain 中移除。

1.10 Metering

在媒体内容订购服务中，PlayReady 的 Metering 功能用来向媒体内容的拥有者报告用户的播放次数，以此来收取相关的费用。Metering 提供了很多便利，例如可以大幅削减在线内容提供商的版税，通常情况下按次数单独购买比一次性永久转让更经济实惠。

playready metering 报告一般分为如下几个步骤：

步骤 1 设备检查本地是否存储有该媒体的 MID 的 metering certificate。

- 如果有，转向步骤 3。
- 如果没有，设备向 metering certificate service 请求所需要的 metering certificate。

步骤 2 设备收集 metering data（具有此 MID），然后加密此 metering data 并发送到由 metering certificate 中所提供的 metering service URL。

步骤 3 metering service 确认此 metering 报告，然后通知设备，将设备上的 metering 计数器清零。

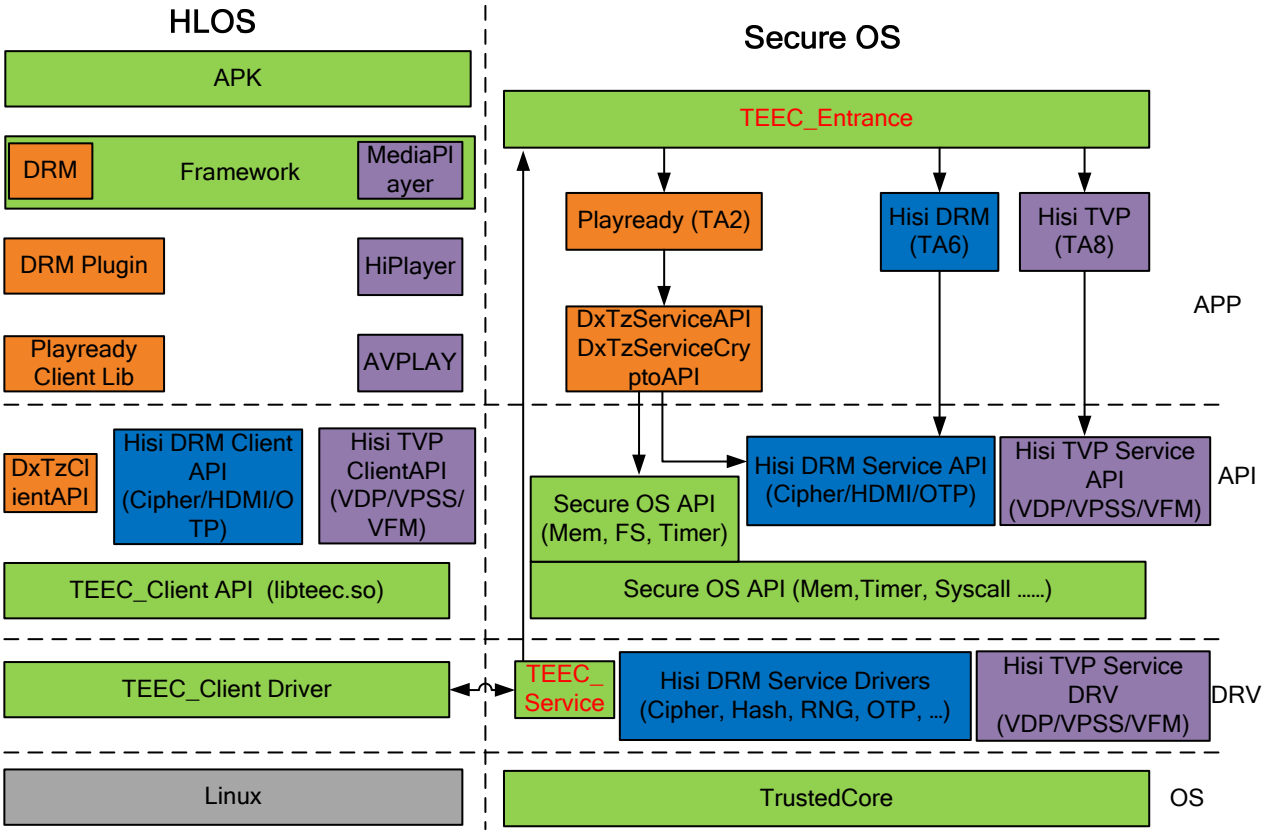
如果 metering data 数据量很大，则设备可以将其分为几个部分，重复步骤 2、3 分别发送。

----结束

2 HW 版本 PlayReady 开发

2.1 方案框图

图2-1 HW 版本 PlayReady 方案框图



HW 版本 PlayReady 基于海思芯片 Trustzone 硬件保护机制，对证书，密钥以及解密后的数据提供更高级别保护，解密和解码过程都运行在安全域。

上图中橙色部分负责将数据解密到安全 Memory，紫色部分负责解密后数据的解码，播放和显示。解密与解码过程通过 Android 的 Framework 层进行关联。

2.2 准备工作

步骤 1 从 Discretix 获取到与 SDK 匹配的 HW 版本的 PlayReady 发布包。

步骤 2 从海思获取到 SDK 对应的 HW 版本 Playready 补丁包，并放到 Android SDK 根目录。

步骤 3 在 Android SDK 根目录解压缩海思 PlayReady HW 版本补丁包。

步骤 4 在 Android SDK 根目录执行./pack_install.sh ${ANDROID_BUILD_TOP}命令，将补丁包内容合入 SDK。

步骤 5 参考 device/hisilicon/bigfish/prebuilts/drm/readme_cn.txt 说明，将 Discretix 发布包中的库文件、配置文件及 provision 相关文件更新到 SDK 中。

- 更新 PlayReady 库文件

  拷贝 Discretix playready 发布包/Software/SW/*.so 到 SDK 的 device/hisilicon/bigfish/prebuilts/drm 目录

- 更新 playready TA

  拷贝 Discretix playready 发布包 /Software/Service/obj/EXECUTABLES/task_QA_TZService_intermediates/task_QA_TZService.elf 到 SDK 的 device/hisilicon/bigfish/sdk/source/plugin/tvp/trustedcore/release/internal_tasks 目录。

- 更新 Provision 工具
  - 拷贝 Discretix playready 发布包/Provisioning/SecuredProvisioning 目录下以下文件到 SDK 的 device/hisilicon/bigfish/prebuilts/drm/Provisioning 目录。
    - cek.bin
    - cekVerifier.bin
    - DxDrmConfig.txt
    - DxPrdyPkgEnc
  - 拷贝 Discretix playready 发布包

    /Provisioning/SecuredProvisioning/provTest/libs/armeabi/securedProvTest 文件到 SDK 的 device/hisilicon/bigfish/prebuilts/drm/Provisioning 目录。

----结束

2.3 编译

HW 版本 PlayReady 依赖支持 Trustzone 特性的安全芯片和 SDK，以 Hi3719CV100 芯片为例， 在 SDK 根目录依次执行以下命令：

步骤 1 source build/envsetup.sh

步骤 2 lunch Hi3719CV100-eng 或 lunch Hi3719CV100-user

步骤 3 ./device/hisilicon/Hi3719CV100/build/tvp.sh，该操作同时打开 SDK 的安全配置选项。

步骤 4 make bigfish -j32 2>&1 | tee bigfish.log

----结束

2.4 签名

请参考《Trustzone 开发指南》完成对 boot 镜像和 trustedcore 镜像（trustedcore.img）进行签名。

2.5 烧写

请参考《Trustzone 开发指南》，使用 HiTool 工具烧写签名后的安全 boot 镜像, trustedcore.img 及其它镜像。

2.6 测试说明

请参考 device/hisilicon/bigfish/prebuilts/drm/readme_cn.txt 说明，使用 PlayReady Demo APK 对 PlayReady 官网测试用例进行测试。

加密文件的播放，需要机顶盒能访问微软 License 服务器。

2.6.1 测试前准备

- 请确认机顶盒以下文件存在
  - /system/lib/libDxDrmServer.so
  - /system/lib/libdxprdyDrmOem.so
  - /system/lib/drm/libDxPrRecommended.so
  - /system/app 或/data/app 目录下的 DrmAssist-Recommended.apk
  - /system/etc/DxDrmConfig.txt 或者 /data/DxDrm/DxDrmConfig.txt
  - /data/DxDrm/cek.bin
  - /data/DxDrm/cekVerifier.bin
  - /data/DxDrm/DxPrdyPkgEnc
  - /data/DxDrm/securedProvTest
  - /data/DxDrm/run_secure_prov.sh
- 请确认/system/build.prop 中 drm.service.enabled=true。
- 请确认后台服务 drmserver 在运行。
- 配置文件注意事项
  - DxDrmConfig.txt 的默认位置为/system/etc/，也可以是/data/DxDrm。放置在/system/etc，可避免被误删除。
  - DxDrmConfig.txt 中，安全文件系统 sfs 所在位置配置项为:

    SfsLocation = /data/sfs

    /data/sfs 位置仅用于调试，sfs 位置默认配置为/drmsfs，推荐将独立的 flash 分区挂载到/drmsfs，避免安全文件系统数据被误删除。

    修改 SfsLocation，务必要修改 run_secure_prov.sh 中对应路径。

2.6.2 证书初始化

开发调试时，首次上电时需要在单板上手工执行证书初始化操作，方法如下：

步骤 1 执行 cd /data/DxDrm。

步骤 2 执行 chmod 755 run_secure_prov.sh。

步骤 3 执行./run_secure_prov.sh。

步骤 4 确认 log 提示未出现证书初始化失败的情况。

步骤 5 确认/data/sfs/dxprdy/dxprdy/dxprdy.sfs 存在，且属性为 666。

----结束

注意

- **工厂生产时有专门的 Provision 流程执行证书烧写和初始化操作，不需要执行以上开发调试用的证书手工初始化操作。**
- **/data/DxDrm 下的数据未破坏的情况下，证书初始化操作通常只需要执行一次。**

2.6.3 Smoothstreaming 码流网页播放

步骤 1 运行 Playready Demo APK（DrmAssist-Recommended.apk)，图标和名字均为 DRM。

步骤 2 选择 UrlPlay，进入微软 smoothstreaming 测试网页 http://playready.directtaps.net/smoothstreaming/ 。

步骤 3 选择任一 Manifest 文件播放。

----结束

2.6.4 ASF 文件本地播放

步骤 1 从微软的 playready 服务器下载测试码流到本地。

打开 http://playready.directtaps.net/网页，选择 PYV/PYA 选项，对应页面下即为 playready 对应官方测试用例和码流，下载需要的测试文件到本地。

步骤 2 运行 Playready Demo APK（DrmAssist-Recommended.apk)，图标和名字均为 DRM。

步骤 3 选择 LocalPlay，进入测试文件所在目录，选择需要播放的文件。

步骤 4 用'CheckRights'检查权限是否有效，若权限无效，选择“GetRightsByPlugin”或“GetRightsByApplication”以获取权限。

步骤 5 选择 Play 播放文件。

----结束

2.7 工厂生产

HW 版本 PlayReady 方案，要求支持安全启动和 Trustzone 特性，工厂生产时需要完成以下数据烧写：

- 烧写 HDCP Root Key，加密 HDCP Key 并保存到 Flash，用于 HDMI 输出保护。
- 烧写 RSA Root Public Key，用于芯片启动时校验 boot 镜像 Key Area 数据。
- 烧写 PlayReady SFS Key，用于 PlayReady 安全文件系统数据加解密保护。
- 烧写和初始化 PlayReady 证书。
- 安全启动和 Trustzone 标记使能。

SDK 的 device/hisilicon/bigfish/sdk/sample/advca/sample_factory_drm.c 文件，提供了以上数据烧写的参考代码。

2.7.1 芯片 OTP 数据列表

表 2-1 中列出了 TVP Playready 方案需要烧写到芯片 OTP 中的数据。

表2-1 芯片 OTP 数据列表

名称	位置	大小	说明
RSA_RootKey	0x2f0~0x3ff	E: 16Bytes N: 256Bytes	RSA Root Public key, 用来校验 boot 镜像的 key_area 数据 RSA_Ext_Key。 Lock 后无法读出。
HDCP_RootKey	0xc0~0xcf	16Bytes	HDCP 数据加解密密钥。 Lock 后无法读出。
pr_sfs_key	0x130~0x13f	16Bytes	PlayReady 证书及其它 PlayReady 本地存储数据加解密密钥。 位于 otp 的 trustzone 区域，otp_tz_area_enable 使能后只有安全 CPU 才可以访问。
otp_tz_area_enable	0x4[2]	1bit	Otp trustzone 区域使能标记。 使能后，只有安全 CPU 才可以访问 otp 的 trustzone 区域数据。
soc_tz_enable	0x4[1]	1bit	芯片 trustzone 功能使能标志。
SCS_activation	0x18[0]	1bit	安全启动使能标志。

2.7.2 数据准备

2.7.2.1 RSA Root Key 公钥

使用“2.4 签名”章节对 boot 镜像签名时生成的 RSA Root Key 公钥数组文件 root_rsa_pub.h 中的数组，更新 sample_factory_drm.c 文件 rsa_root_key_pub[]数组的内容。

2.7.2.2 OEM Seed Key

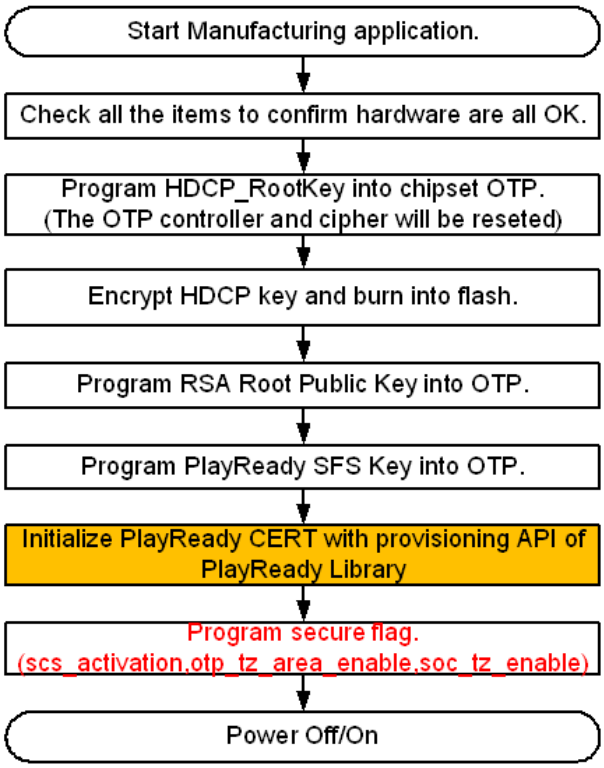
sample_factory_drm.c 文件中的 oem_seed_key[]数组，为各加解密密钥的种子密钥，由 OEM 厂商自行定义，该种子密钥可以根据 OEM 厂商的需要生成 HDCP_RootKey，R2R_RootKey，pr_sfs_key。

密钥生成模式有以下三种，请参考 HI_FAC_KEY_MODE_E 定义：

- HI_FAC_KEY_MODE_UNIFIED
  所有机顶盒芯片使用相同加解密密钥
- HI_FAC_KEY_MODE_UNIQUE
  每个机顶盒芯片使用不同加解密密钥。
  通过固定算法，根据种子密钥、加解密密钥类型、芯片 ChipID，计算出对应的加解密密钥。
- HI_FAC_KEY_MODE_RANDOM
  机顶盒芯片使用随机加解密密钥。

2.7.3 数据烧写流程

图2-2 HW 版本 Playready 工厂生产流程



图中橙色部分为烧写和初始化 Playready 证书，具体方案和流程请参考“2.7.5 PlayReady 证书烧写”。

2.7.4 工厂生产 API 说明

SDK 的 device/hisilicon/bigfish/sdk/sample/advca/sample_factory_drm.c 文件为 HW 版本 PlayReady 工厂生产 sample，提供了以下 OTP 数据烧写 API 接口：

- HI_FAC_Set_HDCP_RootKey：烧写 HDCP Root Key。
- HI_FAC_Encrypt_HdcpKey：加密 HDCP Key。
- HI_FAC_Set_RSA_RootKey：烧写 RSA Root Public key。
- HI_FAC_Set_PR_SFS_Key：烧写 PlayReady 安全文件系统加解密密钥。
- HI_FAC_Set_Secure_Flag：烧写 otp_tz_area_enable，soc_tz_enable 以及 SCS_activation 标记，使能芯片 trustzone 和安全启动功能。

2.7.5 PlayReady 证书烧写

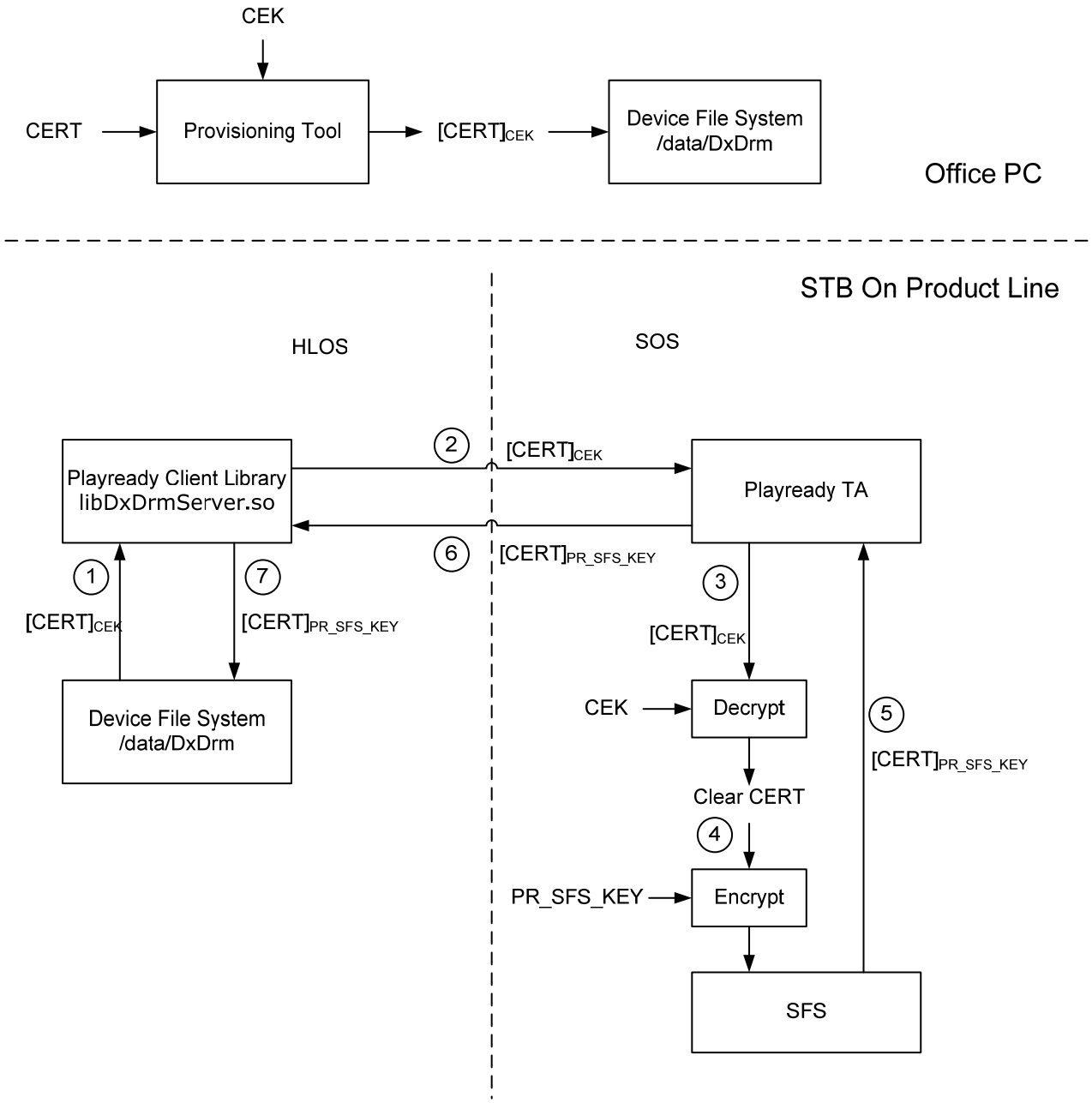
2.7.5.1 PlayReady 证书烧写方案

OEM 厂商使用 Discretix 提供的 PC 端 Provisioning Tool 加密 PlayReady Model 证书（以下简称 CERT），加密密钥为 CEK（Customer Encryption Key），然后将加密的证书放入机顶盒文件系统。

机顶盒厂测程序执行库文件 libDxDrmServer.so 提供的 Provisioning API，完成对证书的解密，并使用烧写到芯片 OTP 中的 PR_SFS_KEY 重新加密证书。

以下为 PlayReady 证书烧写方案框图：

图2-3 Playready 证书烧写方案



HLOS：非安全文件系统 Linux。

SOS：安全文件系统。

2.7.5.2 PlayReady 证书烧写步骤

步骤 1 使用 Discretix 提供 PC 端 Provisioning Tool 加密 playready model 证书 CERT，加密密钥为 CEK。

步骤 2 将 CEK 加密的证书$[CERT]_{CEK}$放入机顶盒/data/DxDrm 目录。

步骤 3 执行库文件 libDxDrmServer.so 提供的 Provisioning API SetProvisioningCEK()设置 $[CERT]_{CEK}$ 解密密钥 CEK。

步骤 4 执行库文件 libDxDrmServer.so 提供的 Provisioning API VerifyStoredCEK ()，确认 $[CERT]_{CEK}$ 解密密钥 CEK 成功设置。

步骤 5 执行库文件 libDxDrmServer.so 提供的 Provisioning API StartSecuredProvisioning ()完成证书烧写动作。

Playready 库将按照图 2-3 中①~⑦所示，将$[CERT]_{CEK}$送到安全内存解密，然后用存放在 OTP 安全区域的 PR_SFS_KEY 加密，最后将$[CERT]_{PR_SFS_KEY}$送回非安全侧，存放到机顶盒文件系统中。

步骤 6 执行库文件 libDxDrmServer.so 提供的 Provisioning API VerifySecuredProvisioning ()确认证书烧写成功。

----结束

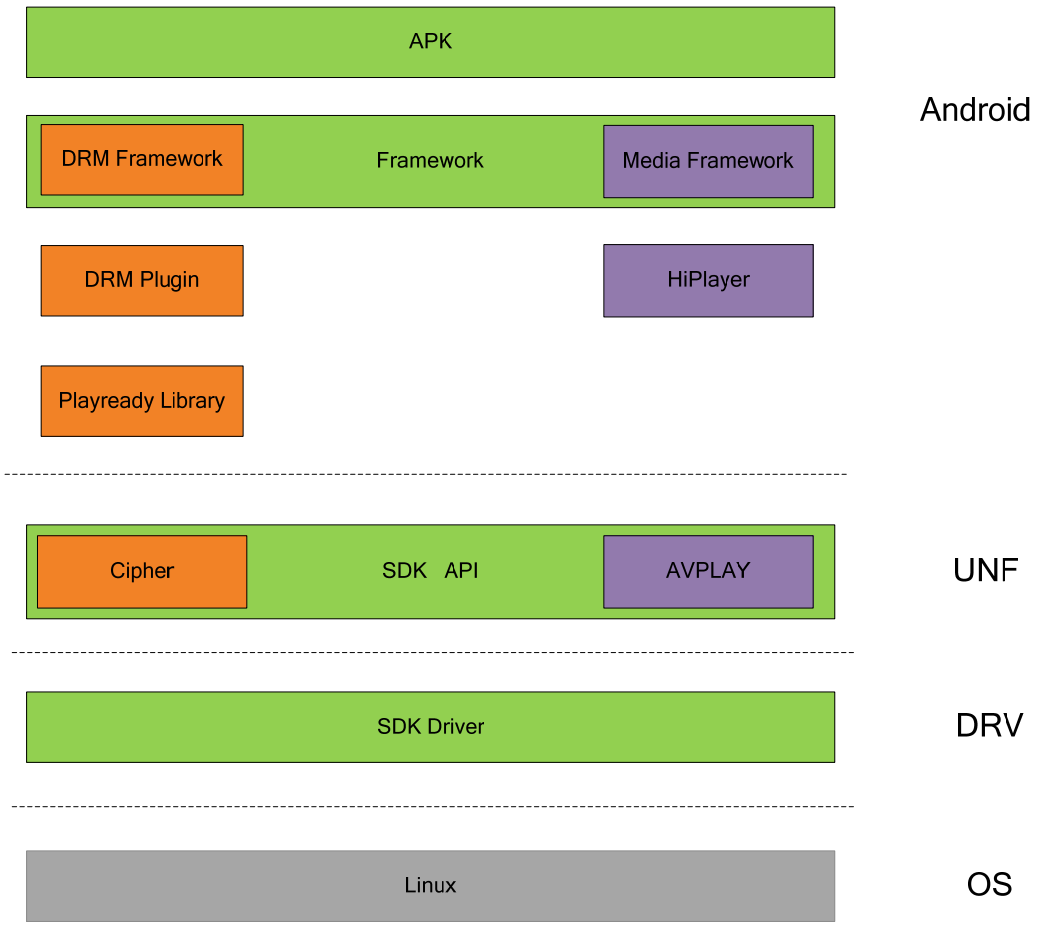
2.7.5.3 PlayReady 证书烧写 API 说明

请参考 Discretix 的 PlayReady 发布包中的《Secured Provisioning design.pdf》。

3 SW 版本 PlayReady 开发

3.1 方案框图

图3-1 SW 版本 PlayReady 方案框图



3.2 准备工作

- 从 Microsoft 申请到 Playready Company 证书并生成 Model 证书。

  bgroupcert.dat、zgpriv.dat、devcerttemplate.dat、priv.dat
- 从海思获取到 SW 版本 PlayReady 补丁包，包含以下库文件。

  libhi_playready.so

  libhi_playready_protect.so

  libplayreadyplugin.so

3.3 编译说明

以 Hi3719CV100 芯片为例：

- 不要编译 HW 版本。

  不要运行 device\hisilicon\Hi3719CV100\build\tvp.sh 脚本，确保 device\hisilicon\Hi3719CV100\BoardConfig.mk 中 HISI_TRUST_VIDEO_PATH := false
- 确认 device\hisilicon\Hi3719CV100\device.mk 中，PRODUCT_PROPERTY_FOR_DRM:=true

3.4 测试说明

3.4.1 配置文件说明

可使用配置文件 playreadyconfig.txt 对 SW 版本 PlayReady 的运行环境进行配置：

- playreadyconfig.txt 在单板的默认路径为/system/etc 或/data/playready。
- playreadyconfig.txt 内容示例如下，用户可更改“=”后面的值，“=”前后不要有空格。

```
CertPath=/data/playready/prpd/          /*证书存放路径*/
LocalStorePath=/data/playready/prdata.localstore   /*临时数据存放路径*/
KeyFilePath=/data/playready/prpd/       /*设备证书存放路径*/
CertProtection=enable       /*证书是否加密保存，有效值disable, enable*/
CertRootKeyType=stb_root_key        /*证书加密密钥的类型*/
LogLevel=error          /*log信息输出级别设置，有效值no, error, info*/
```

- 若单板不存在配置文件 playreadyconfig.txt，或配置项无效时，将使用如下默认配置，证书明文存储，仅用于开发调试：

```
CertPath=/data/playready/prpd/
LocalStorePath=/data/playready/prdata.localstore
KeyFilePath=/data/playready/prpd/
CertProtection=disable
```

```
LogLevel=error
```

- 若证书需要加密存储，playreadyconfig.txt 中请按照如下配置：

```
CertProtection=enable
CertRootKeyType=stb_root_key
```

- 证书加密存储时，对应的密钥烧写和证书加密方法，请参考 3.5 章节内容。
- 证书加密密钥 CertRootKey 保存在芯片 OTP 中，lock 后 CPU 无法读出。

3.4.2 测试前准备

- 将海思 SW 版本 PlayReady 补丁包中的库文件拷贝到单板以下目录：
  - /system/lib/libhi_playready.so
  - /system/lib/libhi_playready_protect.so
  - /system/lib/*drm*/libplayreadyplugin.so
- 参考 3.4.1 章节配置 PlayReady 运行环境。
- 单板创建/data/playready/prpd 目录，并确保/data/playready 及其子目录有读写权限，然后拷贝证书文件到/data/playready/prpd 目录。若证书配置为加密存储，请参考 3.5 章节内容烧写密钥和证书。
- 确认单板/system/app/DrmAssist-Recommended.apk 存在。
- 确认/system/build.prop 中 drm.service.enabled=true 和 drm.client.enabled=true。
- 确认后台服务 drmserver 在运行。

3.4.3 Smoothstreaming 码流网页播放

步骤 1 运行 Playready Demo APK（DrmAssist-Recommended.apk），图标和名字均为 DRM。

步骤 2 选择 UrlPlay，进入微软 smoothstreaming 测试网页 http://playready.directtaps.net/smoothstreaming/ 。

步骤 3 选择任一 Manifest 文件播放。

----结束

3.4.4 ASF 文件本地播放

步骤 1 从微软的 playready 服务器下载测试码流到本地。

打开 http://playready.directtaps.net/网页，选择 PYV/PYA 选项，对应页面下即为 playready 对应官方测试用例和码流，下载需要的测试文件到本地。

步骤 2 运行 Playready Demo APK（DrmAssist-Recommended.apk），图标和名字均为 DRM。

步骤 3 选择 LocalPlay，进入测试文件所在目录，选择需要播放的文件。

步骤 4 用 CheckRights 检查权限是否有效，若权限无效，选择“GetRightsByPlugin”或“GetRightsByApplication”以获取权限。

步骤 5 选择 Play 播放文件。

----结束

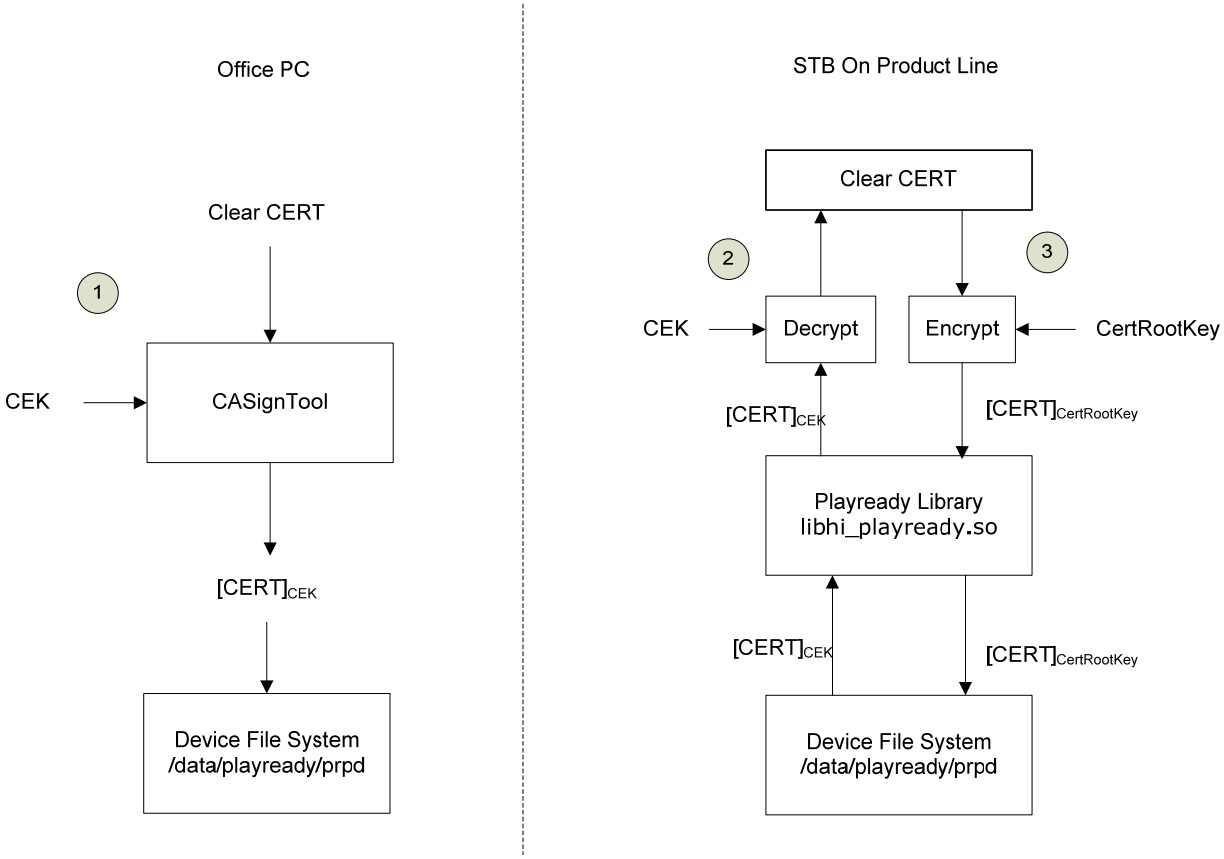
3.5 工厂生产

SW 版本 PlayReady 工厂生产，用于证书加密存储的场景，主要完成以下三个功能：

- 证书加密，用于从 OEM 厂商到生产工厂之间的安全传输。
- 烧写证书加密密钥 CertRootKey 到芯片 OTP。
- 用 CertRootKey 加密证书保存到单板 Flash。

3.5.1 工厂生产方案

图3-2 工厂生产流程



说明

也可在单板上使用 CertRootKey 直接加密明文证书，即略过图中第 3 步操作可以单独执行。

3.5.2 工厂生产操作步骤

步骤 1 在办公室 PC 使用 CASignTool 工具（2.15 及以后版本）加密 1.5 章节生成的 PlayReady 设备证书 bgroupcert.dat、zgpriv.dat、devcerttemplate.dat、priv.dat。

加密算法为 AES128-CTR，密钥为客户生成的 16bytes 密钥 CEK。

步骤 2 生产线厂测程序拷贝 CEK 加密后的证书到单板 PlayReady 证书目录。

步骤 3 烧写 CertRootKey 到芯片 OTP。

步骤 4 执行证书初始化函数，用 CEK 密钥解密证书到内存，然后再用 CertRootKey 加密后写回 Flash。

----结束

3.5.3 工厂生产 API 概览

libhi_playready.so 提供了如下工厂生产 API 函数：

- HI_PLAYREADY_Provision_IsCertRootKeyWrited：检查证书加密密钥 CertRootKey 是否已经烧写。
- HI_PLAYREADY_Provision_SetCertRootKey：烧写证书加密密钥到芯片 OTP。
- HI_PLAYREADY_Provision_CertInit：证书初始化，先使用 CEK 解密证书，然后再用 OTP 中的 CertRootKey 加密证书。

3.5.4 工厂生产 API 描述

HI_PLAYREADY_Provision_IsCertRootKeyWrited

【描述】

检查证书加密密钥 CertRootKey 是否已经烧写。

【语法】

```
HI_S32 HI_PLAYREADY_Provision_IsCertRootKeyWrited(HI_BOOL *bWrited);
```

【参数】

参数名称	描述	输入/输出
bWrited	指示 CertRootKey 是否被烧写	输出

【返回值】

返回值	描述
HI_SUCCESS	获取 CertRootKey 烧写状态成功
其它	获取 CertRootKey 烧写状态失败

【需求】

库：libhi_playready.so

【注意】

无

【举例】

无

【相关主题】

HI_PLAYREADY_Provision_SetCertRootKey

HI_PLAYREADY_Provision_SetCertRootKey

【描述】

烧写 16 字节 CertRootKey 到芯片 OTP。

【语法】

```
HI_S32 HI_PLAYREADY_Provision_SetCertRootKey(HI_U8 *pu8CertRootKey);
```

【参数】

参数名称	描述	输入/输出
pu8CertRootKey	16 字节 CertRootKey。 若为空指针，烧写随机值到芯片 OTP。	输入

【返回值】

返回值	描述
HI_SUCCESS	烧写 CertRootKey 成功
其它	烧写 CertRootKey 失败

【需求】

库：libhi_playready.so

【注意】

该函数在烧写完成后会锁定 CertRootKey，CPU 无法读出。

若无特殊需求，推荐使用随机值以加强密钥和证书的安全性。

注意

CertRootKey 烧写后，务必要先删除动态生成的设备证书文件 /data/playready/prpd/keyfile.dat 和 license 文件/data/playready/prdata.localstore，否则 PlayReady 库会因为设备证书密钥不匹配而运行失败。

【举例】

无

【相关主题】

HI_PLAYREADY_Provision_IsCertRootKeyWrited

HI_PLAYREADY_Provision_CertInit

【描述】

证书初始化，先使用 CEK 解密证书，然后再用 OTP 中的 CertRootKey 加密证书。

【语法】

```
HI_S32 HI_PLAYREADY_Provision_CertInit(HI_U8 *pu8CEK);
```

【参数】

参数名称	描述	输入/输出
pu8CEK	16 字节 CEK，证书加密密钥。 若为空指针，表示证书未用 CEK 加密，直接对明文证书进行加密。	输入

【返回值】

返回值	描述
HI_SUCCESS	证书初始化成功
其它	证书初始化失败

【需求】

库：libhi_playready.so

【注意】

CEK 加密的证书需要保存到 3.4.1 章节所描述的证书存放路径。

【举例】

无

【相关主题】

无

4 开发应用

4.1 PlayReady Demo APK 说明

PlayReady Demo APK 提供了 ASF 文件本地播放，以及 PlayReady 官方测试网站 SmoothStreaming 码流网页播放的功能，支持码流权限获取。

PlayReady Demo APK 基于 Android DRM Framework 和 Media Framwork JAVA 层标准接口实现，其中权限获取基于 DrmManagerClient.java 实现。

海思 Android SDK 的 device/hisilicon/bigfish/development/apps/PlayReadyDemo 目录下为 PlayReady Demo APK 源码，编译出的 APK 名字为 DrmAssist-Recommended.apk，图标和标题均为 DRM。

4.2 Drm Header 获取

Drm Header 数据用于加密流权限获取，PlayReady 的 Drm Header 信息即 PlayReady Header Objects 中的 Rights Management Header 数据。PlayReady Header Objects 的规范请到 http://www.microsoft.com/playready/documents/ 下载。

PlayReady 插件可以从存放在本地的 ASF 格式文件和 SmoothStreaming 的 Manifest 文件中解析和获取 Drm Header 信息。

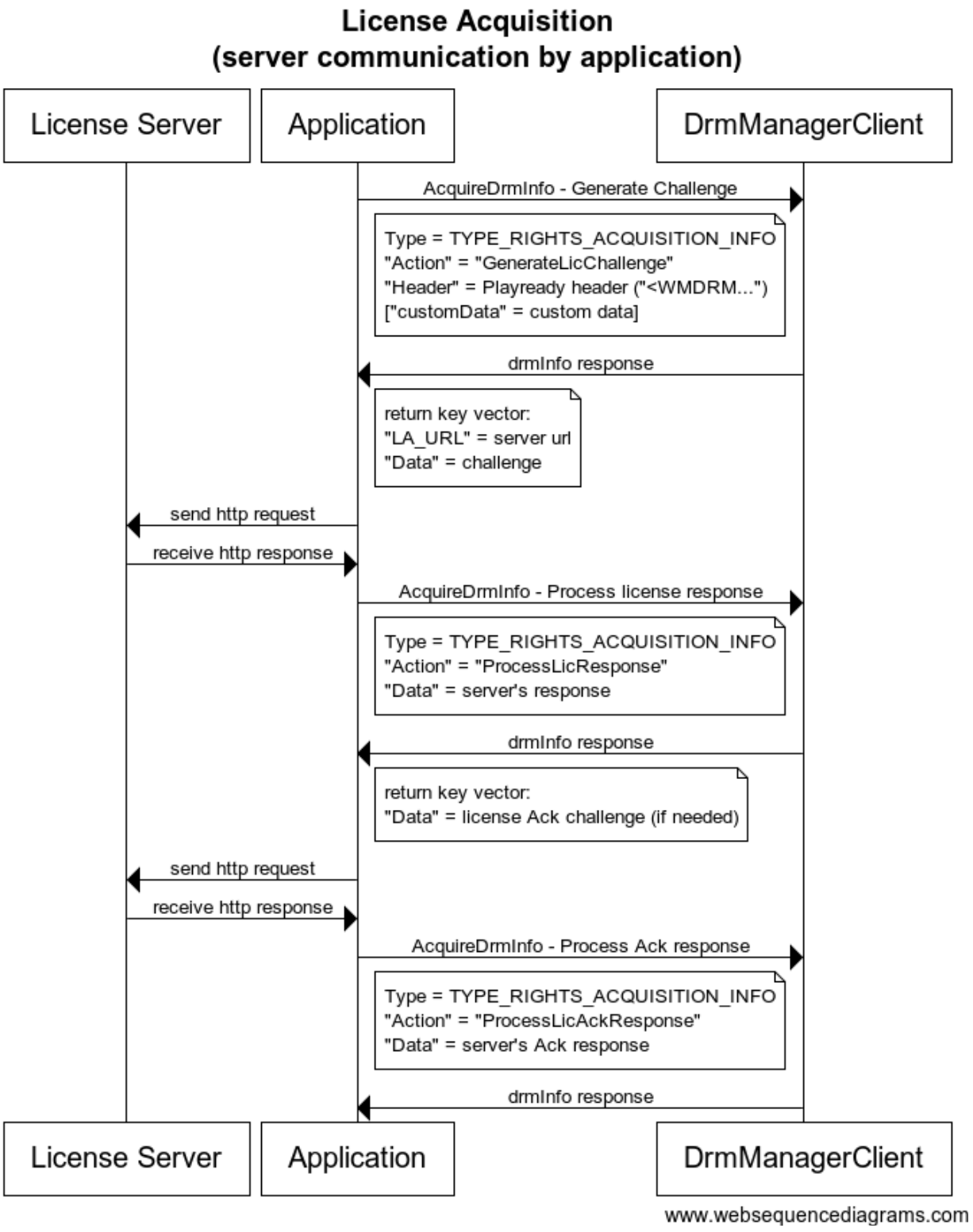
APK 可以调用 DrmManagerClient 的 acquireDrmInfo()接口，通过 PlayReady 插件获取到字符串型的 Drm Header 数据，用于加密流权限获取。对应参考代码可参考 PlayReady Demo APK 代码中的 WebActivity.java 或 AcquireRightsByApp.java 文件。

注意

因 PlayReady 插件无法解析网络文件的内容，APK 需要先下载 SmoothStreaming 的 Manifest 文件到本地并保存为*.ismc 文件， 然后用其完整路径（含文件名）作为获取 DrmHeader 时的输入参数。

4.3 权限获取流程

图4-1 License 获取流程



权限获取流程代码可参考 PlayReady Demo APK 代码中的 WebActivity.java 或 AcquireRightsByApp.java 文件。

图4-2 Join Domain 流程

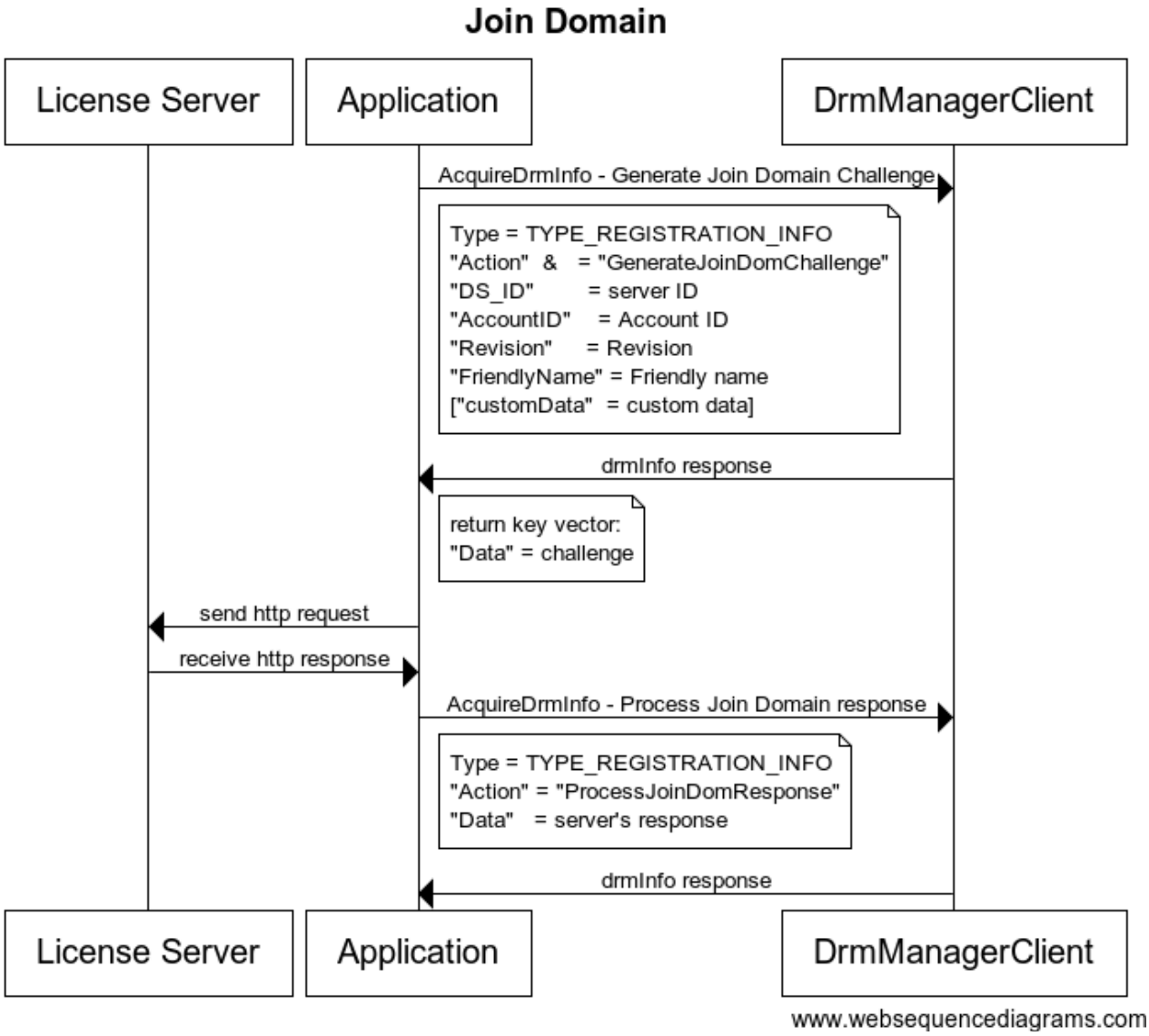


图4-3 Leave Domain 流程

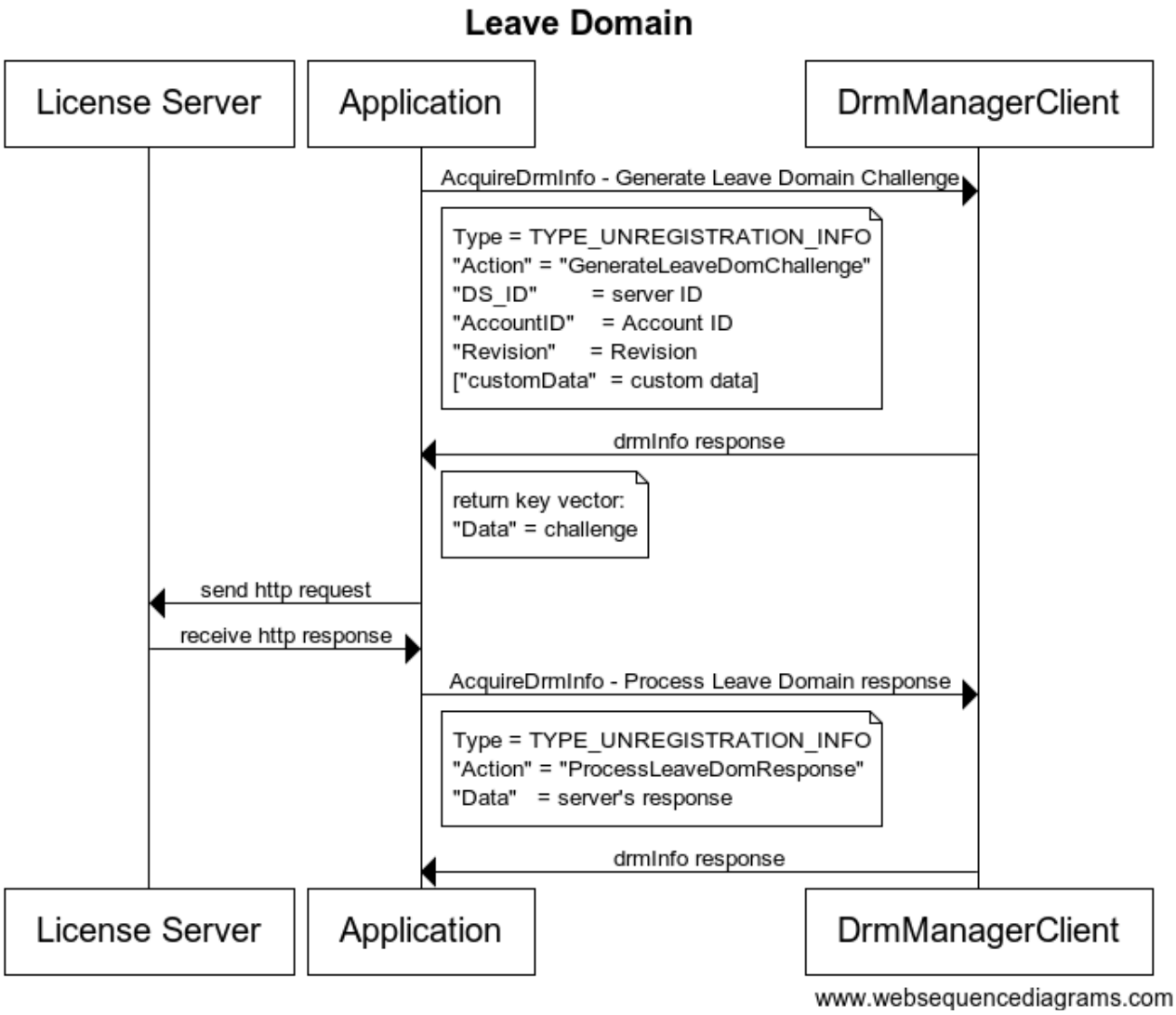
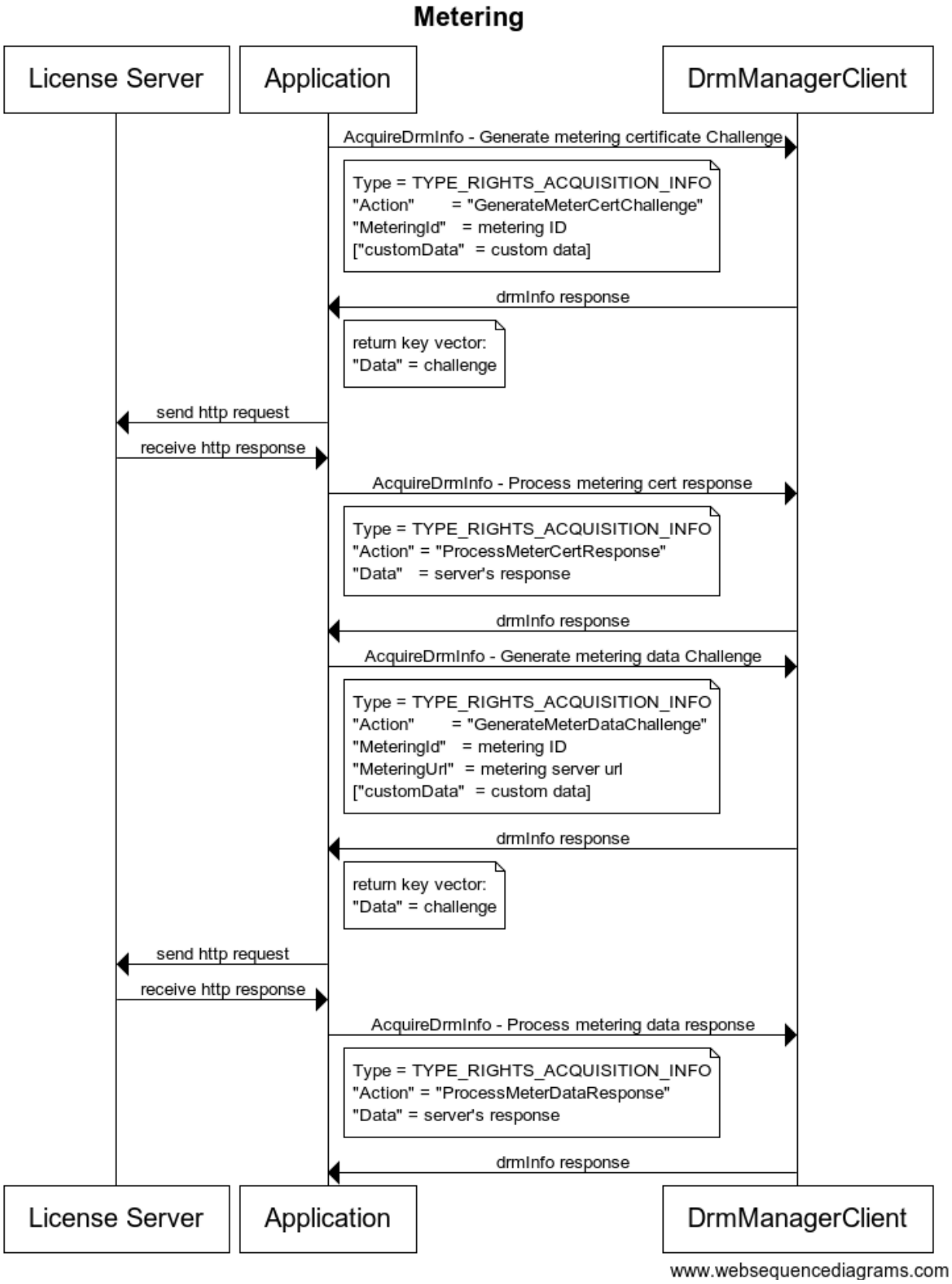


图4-4 Metering 流程



4.4 媒体播放解密流程

媒体播放器调用 Drm Framework 层的 C++接口完成 PlayReady 加密流的解密和播放，对应 API 定义请参考 DrmManagerClient.h。

4.4.1 本地文件解密流程

步骤 1 打开解密会话。

```
sp<DecryptHandle> openDecryptSession(int fd, off64_t offset, off64_t length, const char* mime);
```

或

```
sp<DecryptHandle> openDecryptSession(const char* uri, const char* mime);
```

步骤 2 消费权限。

```
status_t consumeRights(sp<DecryptHandle> &decryptHandle, int action, bool reserve);
```

该接口每个解密会话只能调用一次。

步骤 3 重复调用以下两个接口读取和解密数据直至播放结束。

```
mOriginalMediaSource->read(MediaBuffer **buffer, const ReadOptions *options)
```

和

```
status_t decrypt(sp<DecryptHandle> &decryptHandle, int decryptUnitId, const DrmBuffer* encBuffer, DrmBuffer** decBuffer, DrmBuffer* IV = NULL);
```

步骤 4 播放结束，关闭解密会话。

```
status_t closeDecryptSession(sp<DecryptHandle> &decryptHandle);
```

----结束

4.4.2 网页码流解密流程

步骤 1 打开解密会话。

```
sp<DecryptHandle> openDecryptSession(const char* uri, const char* mime);
```

注意

对于 SmoothStreaming **网页播放**，uri **必须是固定字符串**“ms-smooth-streaming”。

步骤 2 设置 DrmHeader

```
status_t initializeDecryptUnit(sp<DecryptHandle> &decryptHandle, int
decryptUnitId, const DrmBuffer* headerInfo);
```

注意 headerInfo 参数传入的是整个 PlayReady Header Objects 的内容。

步骤 3 消费权限。

```
status_t consumeRights(sp<DecryptHandle> &decryptHandle, int action, bool
reserve);
```

该接口每个解密会话只能调用一次。

步骤 4 重复调用以下两个接口读取和解密数据直至播放结束。

```
mOriginalMediaSource->read(MediaBuffer **buffer, const ReadOptions
*options)
```

和

```
status_t decrypt(sp<DecryptHandle> &decryptHandle, int decryptUnitId,
const DrmBuffer* encBuffer, DrmBuffer** decBuffer, DrmBuffer* IV = NULL);
```

步骤 5 播放结束，关闭解密会话。

```
status_t finalizeDecryptUnit(sp<DecryptHandle> &decryptHandle, int
decryptUnitId);
```

和

```
status_t closeDecryptSession(sp<DecryptHandle> &decryptHandle);
```

----结束

4.5 DrmManagerClient API 说明

4.5.1 API 函数参考域

本文档用 9 个域对 API 参考信息进行描述，如表 4-1 所示。

表4-1 API 参考域说明

参数域	作用
目的	简要描述 API 的主要功能。
语法	显示 API 的语法样式。
描述	简要描述 API 的工作过程。
参数	列出 API 的参数、参数说明及其属性。
返回值	列出 API 的返回值及返回值说明。

参数域	作用
错误码	列出 API 的错误码及错误码说明。
需求	列出 API 要包含的头文件和 API 编译时要链接的库文件。
注意	使用 API 时应注意的事项。
举例	使用 API 的实例。

4.5.2 结构体参考域

本文档用 5 个域对结构体进行描述，如表 4-2 所示。

表4-2 结构体参考域

参数域	作用
目的	简单描述结构体所实现的功能。
语法	列出结构体的定义。
参数	描述结构体各个参数的意义。
描述	列举结构体的使用方法和使用中的注意事项。
参考	描述相关参考信息。

4.5.3 API 概览

4.5.3.1 DrmManagerClient.java 中 License 获取相关 API 简介

- getMetadata：获取内容的权限元数据。
- getOriginalMimeType：获取原始内容 MIME type。
- canHandle：检查 MIME type 或内容是否支持。
- acquireRights：执行完整的权限获取流程，包括与 License 服务器的通讯。
- acquireDrmInfo：返回注册，取消注册，权限请求信息。
- processDrmInfo：处理给定的 DRM 信息，如权限请求, Initiators 等。***PlayReady 插件不支持该接口。***
- getConstraints：返回内容的权限约束信息。
- setOnInfoListener：注册信息监听回调函数，可以使 DRM 插件在注册或权限请求时返回状态或警告信息。
- setOnEventListener：注册事件监听回调函数，可以使 DRM Framework 返回 DRM 处理信息。
- setOnErrorListener：注册错误监听回调函数，可以使 DRM Framework 返回错误信息。

- getDrmObjectType：返回内容或 MIME type 的权限保护类型。***PlayReady 插件不支持该接口。***
- checkRightsStatus：检查内容的权限是否有效。
- removeRights：移除内容的权限。
- removeAllRights：移除 DRM 插件的所有权限信息。

4.5.3.2 DrmManagerClient.h 中解密相关 API 简介

- openDecryptSession：打开解密会话，用于受保护内容的解密。
- closeDecryptSession：关闭句柄对应的解密会话。
- initializeDecryptUnit：初始化解密单元，设置 DRM Header 信息。
- finalizeDecryptUnit：去初始化解密单元。
- consumeRights：消费内容的权限。
- pread：从打开的 DRM 文件中读取指定字节数据。 ***PlayReady 插件不支持该接口。***
- decrypt：解密。

4.5.4 DrmManagerClient.java 权限获取 API 描述

getMetadata

【描述】

获取内容的权限元数据。

【语法】

```
ContentValues getMetadata(URI uri);
```

【参数】

参数名称	描述	输入/输出
uri	内容的 URI	输入

【返回值】

返回值	描述
ContentValues 实例	表示元数据的 key-value 对。 Playready 插件支持下面的元数据 key: Key=”CONTENT_MIME_TYPE”, value=uri 指定内容的 MIME type.
NULL	失败。

【需求】

类：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

getOriginalMimeType

【描述】

获取原始内容 MIME type。

【语法】

```
String getOriginalMimeType(String path);
```

【参数】

参数名称	描述	输入/输出
path	受保护内容的路径	输入

【返回值】

返回值	描述
MIME type	原始内容的 MIME type，例如"video/mpeg"。
空字符串	失败。

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

canHandle

【描述】

检查 MIME type 或内容是否支持。

【语法】

```
boolean canHandle(String path, String mimeType);
```

【参数】

参数名称	描述	输入/输出
path	受保护内容的路径	输入
mimetype	对象的 MIME type	输入

【返回值】

返回值	描述
true	插件能够处理给定的 MIME type 或路径。
false	插件不能处理给定的 MIME type 和路径。

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

acquireRights

【描述】

执行完整的权限获取流程，包括与 License 服务器的通讯。

【语法】

```
int acquireRights(DrmInfoRequest drmInfoRequest);
```

【参数】

参数名称	描述	输入/输出
drmInfoRequest	权限获取请求信息	输入

DrmInfoRequest 用户数据——权限获取

infoType	key	value	描述
TYPE_RIGHTS_ACQUISITION_INFO	Action	AcquireLicenseFull	自动获取权限请求，与服务器通讯由插件内部完成。 所需参数 (key-value)： “Header” = <drm header> "CustomData" = custom data for license

【返回值】

返回值	描述
ERROR_NONE	成功
ERROR_UNKNOWN	失败

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

acquireDrmInfo

【描述】

返回注册，取消注册，权限请求信息。

【语法】

```
DrmInfo acquireDrmInfo(DrmInfoRequest drmInfoRequest);
```

【参数】

参数名称	描述	输入/输出
drmInfoRequest	权限获取请求信息	输入

DrmInfoRequest 用户数据——权限获取

infoType	key	value	描述
TYPE_RIGHTS_ACQUISITION_INFO	Action	GenerateLicChallenge	请求插件生成权限获取 challenge 数据。 所需参数 (key-value)： “Header” = <drm header>或者“Data” = <filepath> "CustomData" = custom data for challenge 返回信息 (key-value): “LA_URL” = url of license server “Data” = license challenge buffer
TYPE_RIGHTS_ACQUISITION_INFO	Action	ProcessLicResponse	请求插件处理服务器权限获取应答数据。 所需参数 (key-value)： “Data” = response buffer 返回信息 (key-value): “Data” = License Ack challenge buffer
TYPE_RIGHTS_ACQUISITION_INFO	Action	ProcessLicAckResponse	请求插件处理服务器 ACK 应答数据。 所需参数 (key-value)： “Data” = ack response buffer

infoType	key	value	描述
TYPE_RIGHTS_ACQUISITION_INFO	Action	GenerateMeterCertChallenge	请求插件生成 Metering 证书 challenge 数据。 所需参数 (key-value)： "MeteringId" = metering ID "CustomData" = custom data for challenge 返回信息 (key-value): “Data” = challenge buffer
TYPE_RIGHTS_ACQUISITION_INFO	Action	ProcessMeterCertResponse	请求插件处理服务器 Metering 证书应答数据。 所需参数 (key-value)： "MeteringId" = metering ID "CustomData" = custom data for challenge 返回信息 (key-value): “Data” = response buffer
TYPE_RIGHTS_ACQUISITION_INFO	Action	GetDrmHeader	请求插件获取本地内容的 WRMHEADER 数据。 所需参数 (key-value)： “path” = path to content 返回信息 (key-value): “Data” = drm header buffer
TYPE_RIGHTS_ACQUISITION_INFO	Action	GenerateMeterDataChallenge	请求插件生成 Metering Data 的 challenge 数据。 所需参数 (key-value)： "MeteringId" = metering ID "CustomData" = custom data for challenge 返回信息 (key-value): "Data" = challenge buffer

infoType	key	value	描述
TYPE_RIGHTS_ACQUISITION_INFO	Action	ProcessMeterDataResponse	请求插件处理服务器 Metering Data 应答数据。 所需参数 (key-value)： “Data” = response buffer
TYPE_REGISTRATION_INFO	Action	ProcessJoinDomInitiator	请求插件处理 jion-domain initiator 文件。 Initiator 文件 sample: http://playready.directtaps.net/pr/initiator.aspx?p=0&type=JOIN 所需参数 (key-value)： “Data” = initiator text (String)
TYPE_REGISTRATION_INFO	Action	GenerateJoinDomChallenge	请求插件生成 Join Domain 的 challenge 数据。 通常 challenge 应该被发送到 domain controller URL，这个 URL 在 initiator 文件中。 所需参数 (key-value)： “DS_ID” “AccountID” “Revision” “FriendlyName” “CustomData” 返回信息 (key-value): "Data" = challenge buffer
TYPE_REGISTRATION_INFO	Action	ProcessJoinDomResponse	请求插件处理 Join Domain 应答数据。 所需参数 (key-value)： “Data” = response buffer

infoType	key	value	描述
TYPE_REGISTRATION_INFO	Action	ProcessLeaveDomInitiator	请求插件处理 leave domain initiator 文件。 Initiator 文件 sample: http://playready.directtaps.net/pr/initiator.aspx?p=0&type=LEAVE 所需参数 (key-value)： “Data” = initiator text (String)
TYPE_REGISTRATION_INFO	Action	Get_OPL_Miracast_Values	返回 OPL 和 Miracast 数值。 当内容绑定到 license 后 OPL 和 Miracast 数值才被设置。因此，这个操作只有在解密流程中调用 ConsumeRights() 后才是有效的。 当前实现仅支持返回一个解密会话的 OPL 和 Miracast enabler 数值。 AcquireDrmInfo()返回的 OPL value 为 OPL 数字，例如“300”。 当 PlayReady License 包含 Miracast Play Enabler Type 对象，且 Play Enabler Type 域的值为{A340C256-0941-4D4C-AD1D-0B6735C0CB24}，DRM_MIRACAST_ENABLER 字符串将为"true"，否则为"false"。 注意： 1. 返回的 OPL 和 Miracast values 只与当前解密会话有关。 2. openDecryptSession()和 AcquireDrmInfo()必须使用相同的 DrmManagerClient 实例。 返回值(key-value): OPL_AUDIO_COMPRESSED_LEVEL OPL_AUDIO_UNCOMPRES

infoType	key	value	描述
			SED_LEVEL OPL_VIDEO_COMPRESSED_LEVEL OPL_VIDEO_UNCOMPRESSED_LEVEL OPL_VIDEO_ANALOG_LEVEL DRM_MIRACAST_ENABLER
TYPE_UNREGISTRATION_INFO	Action	GenerateLeaveDomChallenge	请求插件生成 Leave Domain 的 challenge 数据。 所需参数 (key-value)： “DS_ID” “AccountID” “Revision” “CustomData” 返回信息 (key-value): "Data" = challenge buffer
TYPE_UNREGISTRATION_INFO	Action	ProcessLeaveDomResponse	请求插件处理 Leave Domain 应答数据。 所需参数 (key-value)： “Data” = response buffer
TYPE_UNREGISTRATION_INFO	Action	RemoveAllLicenses	请求移除所有 Playready licenses
TYPE_UNREGISTRATION_INFO	Action	isRemoveAllLicenses	检查是否支持 "RemoveAllLicenses"选项，如果支持 acquireDrmInfo 返回”Status” = ”ok”。

【返回值】

返回值	描述
ERROR_NONE	成功
ERROR_UNKNOWN	失败

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

getConstraints

【描述】

返回内容的权限约束信息。

【语法】

```
ContentValues getConstraints(URI uri, int action);
```

【参数】

参数名称	描述	输入/输出
uri	内容的 URI	输入
action	Actions 定义为 Action::DEFAULT, Action::PLAY 等等 这个参数不会被用到。	输入

【返回值】

返回值	描述
ContentValues 实例	表示约束值的 key-value 对。 Playready 插件支持下面的约束信息: • DrmConstraints：MAX_REPEAT_COUNT 最大重复次数。 • DrmConstraints：REMAINING_REPEAT_COUNT 剩余重复次数 • DrmConstraints：LICENSE_START_TIME 权限开始时间，在这个时间之前受保护的内容不能播放。数据单位为秒，从 1970 年 1 月 1 日的开始计时。 • DrmConstraints：LICENSE_EXPIRY_TIME 权限终止时间，在这个时间之后受保护的内容不能播放。数据单位为秒，从 1970 年 1 月 1 日的开始计时。 • DrmConstraints：LICENSE_AVAILABLE_TIME 权限有效时间，适用于 FIRST_PLAY_EXPIRATION 权限。 • DrmConstraints：EXTENDED_METADATA 内容是否可以作为铃音(ringtone)播放。
NULL	失败。

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

checkRightsStatus

【描述】

对于指定 Action，检查受保护内容的权限是否有效。

【语法】

```
int checkRightsStatus(String path, int action);
```

【参数】

参数名称	描述	输入/输出
path	受保护内容的路径	输入
action	执行的 Action。 如 Action::DEFAULT，Action::PLAY 等等。	输入

DRM Framework 的 Action 与 PlayReady 操作之间的关系如下表：

DRM Framework Action	PlayReady 操作	是否允许
DEFAULT	播放	取决于证书
PLAY	播放	取决于证书
RINGTONE	铃音	取决于证书
OUTPUT	未定义	不允许
PREVIEW	未定义	不允许
EXECUTE	执行	取决于证书
DISPLAY	未定义	不允许
TRANSFER	拷贝内容到其它设备	允许

【返回值】

返回值	描述
RightsStatus::RIGHTS_VALID	权限有效。
RightsStatus::RIGHTS_EXPIRED	权限过期。
RightsStatus::RIGHTS_INVALID	没有权限或发生错误时。

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

removeRights

【描述】

移除内容的权限。

【语法】

```
int removeRights(String path);
```

【参数】

参数名称	描述	输入/输出
path	受保护内容的路径。 • path 参数使用以下字符串以删除过期的或其它证书： • 所有证书："playready://AllLicenses.playready" • 所有过期证书："playready://AllExpiredLicenses.playready" • 某一内容的证书："playready://xxx.playready", xxx 为内容的路径。	输入

【返回值】

返回值	描述
ERROR_NONE	成功。
ERROR_UNKNOWN	失败。

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

removeAllRights

【描述】

移除 DRM 插件的所有权限信息。用于恢复出厂设置。

【语法】

```
int removeAllRights();
```

【参数】

参数名称	描述	输入/输出
无	无	

【返回值】

返回值	描述
ERROR_NONE	成功
ERROR_UNKNOWN	失败

【需求】

头文件：android.drm.DrmManagerClient;

【注意】

无

【举例】

无

【相关主题】

无

4.5.5 DrmManagerClient.h 解密 API 描述

openDecryptSession

【描述】

打开解密会话，用于本地文件受保护内容的解密。

【语法】

```
sp<DecryptHandle> openDecryptSession(int fd, off64_t offset, off64_t
length, const char* mime);
```

【参数】

参数名称	描述	输入/输出
fd	受保护内容的文件描述符。	输入
offset	内容的起始位置。 这个参数不会被用到。	输入
length	受保护内容的长度。 这个参数不会被用到。	输入
mime	如果不为空，则设置内容的 mime type；	输入

【返回值】

返回值	描述
DecryptHandle	解密会话句柄。 DecryptHandle 会被该 API 赋值： • decryptApiType = DecryptApiType::ELEMENTARY_STREAM_BASED • status = DRM_NO_ERROR • extendedData = Plug-in name (Key=”pluginName”) • mimeType 如果 API 输入参数 mime 不为空，mimeType = mime; 如果 mime 为空，mimeType = 内容的 mime type; 如果插件无法获取内容的 mime type，则设置为默认值”video/ismv”。
NULL	打开解密会话失败。

【需求】

头文件： DrmManagerClient.h

【注意】

- 这个 API 用于应用打开的本地文件的播放。
- 这个 API 不可以用于 Smoothstreaming 文件（例如 ismv/isma）播放。

【举例】

无

【相关主题】

无

openDecryptSession

【描述】

打开解密会话，用于本地或网络码流受保护内容的解密。

【语法】

```
sp<DecryptHandle> openDecryptSession(const char* uri, const char* mime);
```

【参数】

参数名称	描述	输入/输出
uri	受保护内容的 URI。 对于网络码流，使用以下固定字符串： • "ms-smooth-streaming" • "http-stream-download"	输入
mime	如果不为空，则设置内容的 mime type；	输入

【返回值】

返回值	描述
DecryptHandle	解密会话句柄。 • DecryptHandle 会被该 API 赋值： • decryptApiType = DecryptApiType::ELEMENTARY_STREAM_BASED • status = DRM_NO_ERROR • extendedData = Plug-in name (Key=”pluginName”) • mimeType 如果 API 输入参数 mime 不为空，mimeType = mime; 如果 mime 为空，若是网络码流则设置为默认值”video/ismv”；若是本地文件，mimeType = 内容的 mime type，如果插件无法获取内容的 mime type，则设置为默认值”video/ismv”。
NULL	打开解密会话失败。

【需求】

头文件：DrmManagerClient.h;

【注意】

无

【举例】

无

【相关主题】

无

closeDecryptSession

【描述】

关闭句柄对应的解密会话。

【语法】

```
status_t closeDecryptSession(sp<DecryptHandle> &decryptHandle);
```

【参数】

参数名称	描述	输入/输出
decryptHandle	解密会话的句柄。	输入

【返回值】

返回值	描述
DRM_NO_ERROR	成功
DRM_ERROR_UNKNOWN	失败

【需求】

头文件：DrmManagerClient.h;

【注意】

无

【举例】

无

【相关主题】

无

initializeDecryptUnit

【描述】

初始化解密单元，设置 DRM Header 信息。

【语法】

```
status_t initializeDecryptUnit(sp<DecryptHandle> &decryptHandle, int
decryptUnitId, const DrmBuffer* headerInfo);
```

【参数】

参数名称	描述	输入/输出
decryptHandel	解密会话句柄。	输入
decryptUnitId	定义解密单元的 ID，例如 Track ID。 这个参数未使用。	输入
headerInfo	PlayReady 文件的 WMDRM header 信息。 传入的是整个 PlayReady Header Objects 的内容。 PlayReady Header Objects 的规范请到 http://www.microsoft.com/playready/documents/ 下载。	输入

【返回值】

返回值	描述
DRM_NO_ERROR	成功
DRM_ERROR_UNKNOWN	失败。

【需求】

头文件：DrmManagerClient.h;

【注意】

只有当播放网络码流时才可以被调用，播放本地文件时不可以调用。

【举例】

无

【相关主题】

无

finalizeDecryptUnit

【描述】

去初始化解密单元。

【语法】

```
status_t finalizeDecryptUnit(sp<DecryptHandle> &decryptHandle, int
decryptUnitId);
```

【参数】

参数名称	描述	输入/输出
decryptHandel	解密会话句柄。	输入
decryptUnitId	定义解密单元的 ID，例如 Track ID。 这个参数未使用。	输入

【返回值】

返回值	描述
DRM_NO_ERROR	成功
DRM_ERROR_UNKNOWN	失败。

【需求】

头文件：DrmManagerClient.h;

【注意】

只有当播放网络码流时才可以被调用，播放本地文件时不可以调用。

【举例】

无

【相关主题】

无

consumeRights

【描述】

消费内容的权限。

【语法】

```
status_t consumeRights(sp<DecryptHandle> &decryptHandle, int action, bool
reserve);
```

【参数】

参数名称	描述	输入/输出
decryptHandel	解密会话句柄。	输入
action	执行的 Action，例如 Action::DEFAULT, Action::PLAY，等等。	输入
reserve	如果权限应该被保留，设置为 true	输入

【返回值】

返回值	描述
DRM_NO_ERROR	消费权限成功
DRM_ERROR_UNKNOWN	消费权限失败。
DRM_ERROR_NO_LICENSE	没有权限
DRM_ERROR_LICENSE_EXPIRED	权限过期
DRM_ERROR_CANNOT_HANDLE	无法处理

【需求】

头文件：DrmManagerClient.h;

【注意】

对于每个解密会话，该 API 只能被调用一次。

【举例】

无

【相关主题】

无

decrypt

【描述】

解密受保护内容。

【语法】

```
status_t decrypt(sp<DecryptHandle> &decryptHandle, int decryptUnitId,
const DrmBuffer* encBuffer, DrmBuffer** decBuffer, DrmBuffer* IV = NULL);
```

【参数】

参数名称	描述	输入/输出
decryptHandel	解密会话句柄。	输入
decryptUnitId	解密单元 ID，如 Track ID。 对于 HW 版本 PlayReady 插件，这个参数表示解密的是音频还是视频数据： Video – trackId = 0，数据解密到安全内存。 Audio – trackId = 1，数据解密到非安全内存。	输入
encBuffer	加密数据 Buffer。 对于 HW 版本 PlayReady 插件： • 硬件解密模块直接使用物理地址进行解密以避免非安全侧与安全侧之间的多次数据拷贝。 • encBuffer 负责把加密数据 buffer 和解密数据 buffer 的物理地址发送给 PlayReady 插件处理。 • 加密数据 buffer 为 HI_MMZ_Malloc()分配的不带 cache 的非安全侧 buffer。 • 若为视频数据，解密数据 buffer 为 HI_SEC_MMZ_New()分配的不带 cache 的安全侧 buffer；若为音频数据，解密数据 buffer 为 HI_MMZ_Malloc()分配的非安全侧 buffer。 • encBuffer.data 对应的数据结构为 DrmExtractor.h 中定义的私有数据结构 PhysicalBuffer。	输入
decBuffer	解密数据 Buffer。 对于 HW 版本 PlayReady 插件： 若为视频数据，解密数据 buffer 为 HI_SEC_MMZ_New()分配的不带 cache 的安全侧 buffer；若为音频数据，解密数据 buffer 为 HI_MMZ_Malloc()分配的非安全侧 buffer。	输出
IV	IV 向量 buffer。 IV.data 指向私有数据结构 DxMultiSampleHeader，后面紧跟几个 DxSubSample。DxSubSample 的数目由 DxMultiSampleHeader 中的 dwSubSamplesNum 确定。	输入

【返回值】

返回值	描述
DRM_NO_ERROR	成功
DRM_ERROR_UNKNOWN	失败
DRM_ERROR_SESSION_NOT_OPENED	会话未打开
DRM_ERROR_LICENSE_EXPIRED	权限过期

返回值	描述
DRM_ERROR_DECRYPT_UNIT_NOT_INITIALIZED	解密单元未初始化
DRM_ERROR_DECRYPT	解密错误

【需求】

头文件：DrmManagerClient.h, DRMExtractor.h

【注意】

对于 HW 版本 PlayReady 插件，decrypt()函数及其参数的更多说明请参考 Discretix 的 PlayReady 插件 API 手册《Discretix Android DRM Framework PlayReady Plugin for recommended plugin User Manual.pdf》。

【举例】

无

【相关主题】

无

4.5.6 数据类型概览

这里只描述解密接口用到的几个私有数据类型：

- PhysicalBuffer：HW 版本 PlayReady 插件定义的用于传递加解密数据 buffer 物理地址的数据结构。
- DxMultiSampleHeader：IV 向量及多 NAL 数据结构 Header 定义。
- DxSubSample： NAL 数据结构定义。

4.5.7 数据类型描述

PhysicalBuffer

【说明】

HW 版本 PlayReady 插件定义的用于传递加解密数据 buffer 物理地址的数据结构。

在 DrmExtractor.h 中定义。

【定义】

```
typedef struct
{
    unsigned int encPhyAddr;
    unsigned int encVirAddr;
    int encDataSize;
    unsigned int decPhyAddr;
```

```
    unsigned int decVirAddr;
    int decDataSize;
} PhysicalBuffer;
```

【成员】

成员名称	描述
encPhyAddr	加密数据 buffer 物理地址。
encVirAddr	加密数据 buffer 虚拟地址。
encDataSize	加密数据长度。
decPhyAddr	解密数据 buffer 物理地址。
decVirAddr	解密数据 buffer 虚拟地址。 若解密数据 buffer 为安全侧 buffer，该参数不使用。
decDataSize	解密数据长度。

【注意事项】

无

DxMultiSampleHeader

【说明】

IV 向量及多 NAL 数据结构 Header 定义。

【定义】

```
typedef struct
{
    uint64_t qwInitializationVector;
    uint32_t dwOutBufferOffset;
    uint32_t dwMediaOffset;
    uint32_t dwSubSamplesNum;
} DxMultiSampleHeader;
```

【成员】

成员名称	描述
qwInitializationVector	从 piff 或 asf 文件中获取的 8bytes 的 IV 向量。
dwOutBufferOffset	输出 buffer 偏移地址，作为数据拷贝或解密的起始地址。

成员名称	描述
dwMediaOffset	加密数据偏移地址，通常是 0。 当多次调用解密函数解密同一帧数据中的多个 NAL 单元时，该偏移地址不为 0。
dwSubSamplesNum	DxMultiSampleHeader 后面紧跟的 DxSubSample 结构体数目。

【注意事项】

无

DxSubSample

【说明】

NAL 数据结构定义。

【定义】

```
typedef struct
{
    uint32_t dwClearDataSize;
    uint32_t dwEncryptedDataSize;
} DxSubSample;
```

【成员】

成员名称	描述
dwClearDataSize	需要从加密数据 buffer 拷贝到解密数据 buffer 的非加密数据大小。
dwEncryptedDataSize	需要解密数据的大小。

【注意事项】

DxMultiSampleHeader ,DxSubSample 与多 NAL 数据关系图。

图4-5 多 NAL 数据结构图