WPS Android 专业版加解密功能使用手册

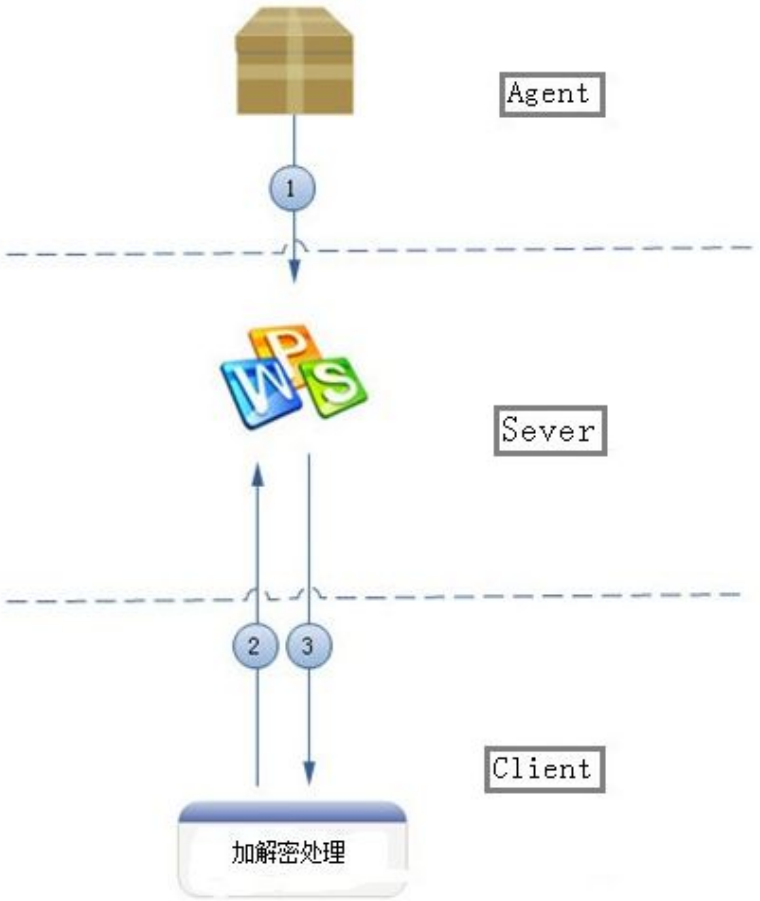
一、专业版加解密概述

WPS Android 专业版加解密是指 WPS 连接用户自己编写的加密工具，使用加密工具对数据进行加解密，加解密算法是有用户自定义的，用户只需要实现连接 WPS 的接口就可以对 WPS 打开文件的数据进行加解密控制，从而提高数据的安全性。

下面展示使用用户自己编写的 Demo 通过第三方调用打开 WPS，同时通过判断调用 WPS 的应用的包名来决定是否进行加解密，实现指定应用打开和保存加密的文档工作。

二、基本架构

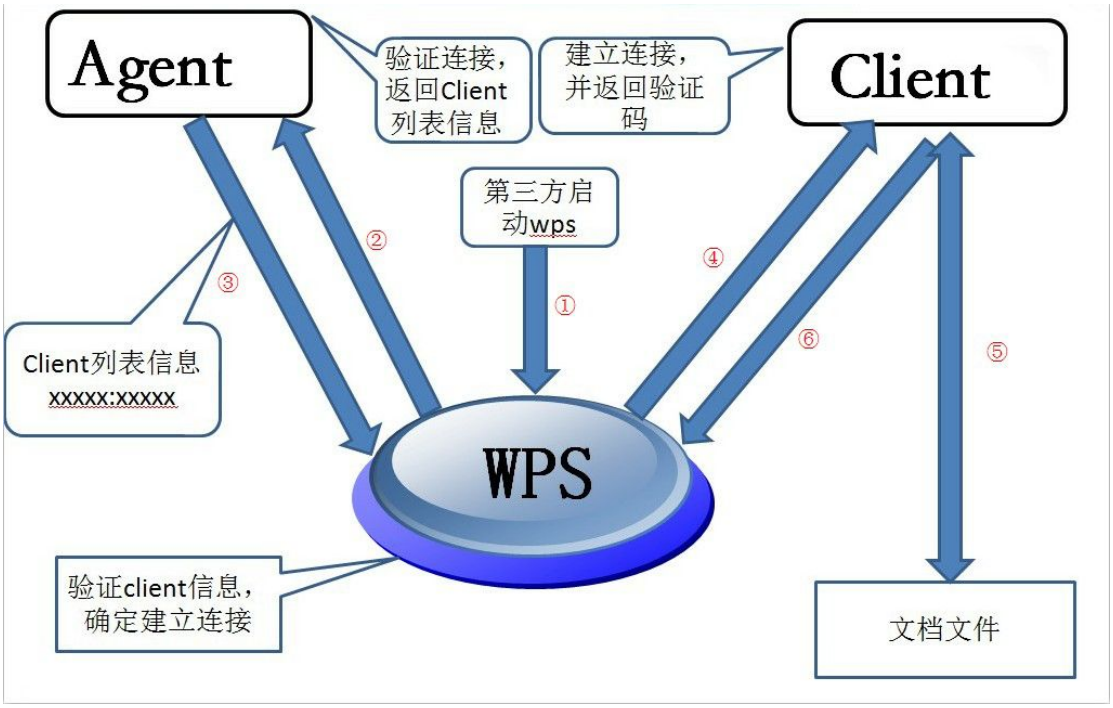
以下图为专业版加解密机制基本架构图



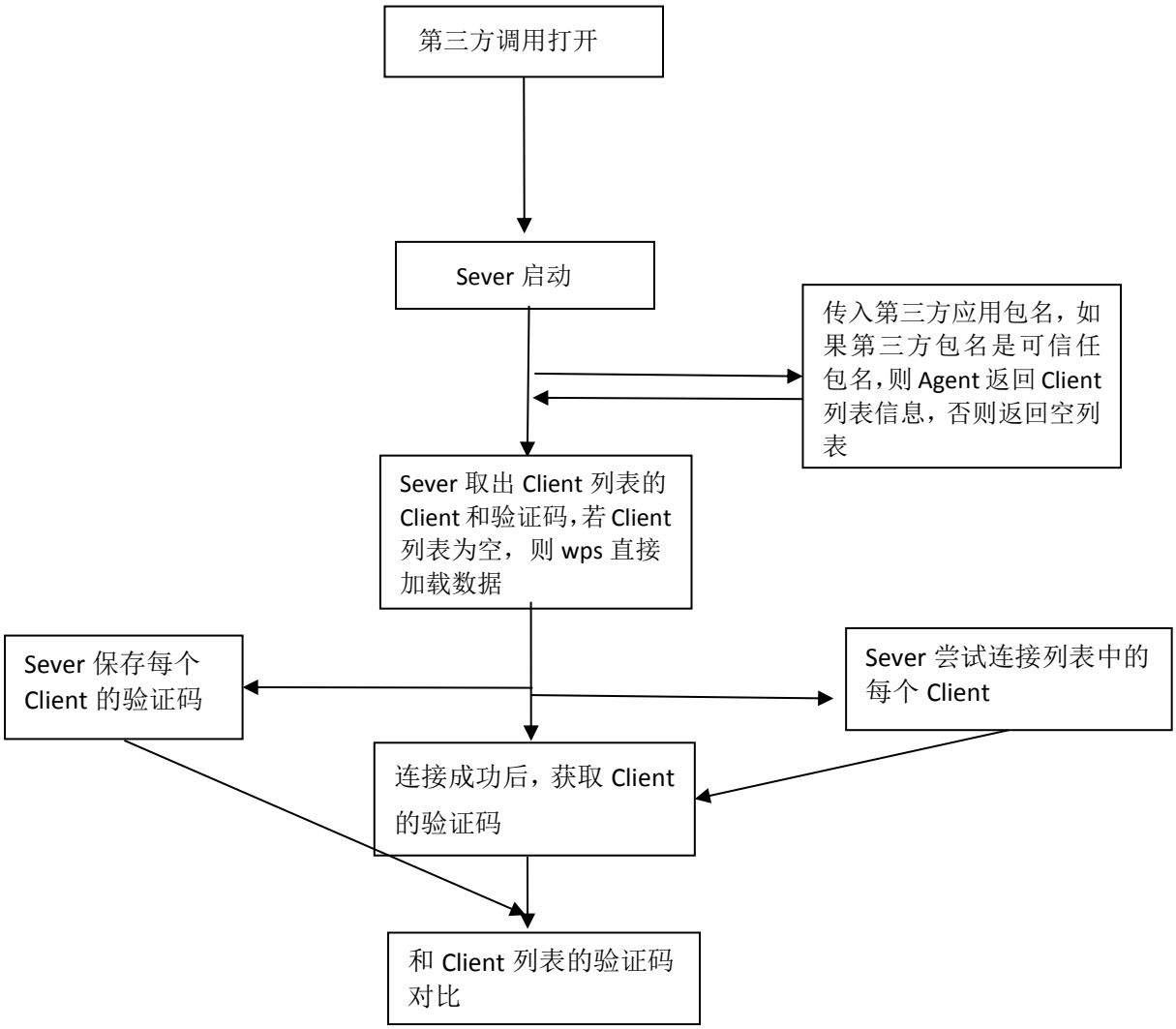
专业版加解密机制架构图

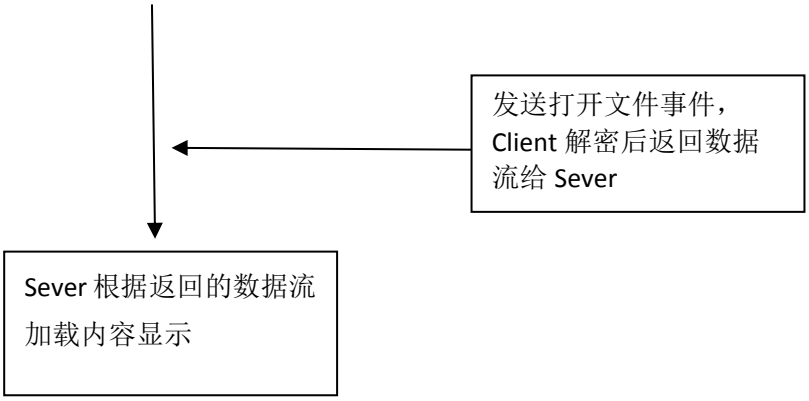
专业版加解密机制由三个模块组成：Agent，Sever，Client。

下面是是整个工程的连接流程图：



整个工作流程操作如下图（以打开加密文档为例）：





WPS 打开一个加密的文档的详细流程

流程描述：

示例：打开具有特殊加密的文档

1、从第三方应用中启动 Server（即 WPS，以下称 WPS）打开某个文件，启动 WPS 时会传入第三方包名。（第三方启动 WPS 详见“第三方操作 WPS 文档.doc”文档）
2、WPS 首先绑定 Agent，同时传入调用 WPS 的第三方包名，如果绑定成功且第三方包名验证正确，WPS 向 Agent 请求 Client 认证信息列表（即 JSON 数据包）。
3、WPS 根据返回的 Client 认证信息列表，解析出 Client 的信息，然后依次绑定 Client。
4、若有一个 Client 绑定成功后，获取该 Client 的验证码和 Client 列表汇总的验证对比，并验证该 Client 的权限。
5、认证成功后，WPS 将打开文件的事件传递给 Client。
6、Client 对文件进行解密，并将解密结果传递给 WPS

三、模块划分及术语定义

业务系统方：指对 Office 使用流程有根据特定业务（如加解密）进行定制要求的一方。

Agent：一个本地应用程序，由业务系统开发方提供，负责提供 Client 的认证信息列表，为 json 数据格式。

Sever：即 WPS Office for Android（以下简称 WPS） 应用程序，是流程的发起者，事件的发送者。

Client：接收事件的应用程序，由业务系统开发方提供。必须预先通过业务系统方管理员注册到认证列表之后，才能通过 Agent 被 Server 获取到，才能接收到由 Server 发送的事件。

四、模块具体功能

4.1 Agent 规范

Agent 是一个本地应用程序。它的包名是由业务系统方管理员预订的（如 cn.wps.moffice.agent.OfficeServiceAgent ,另外Agent的包名必须是wps指定包名，现在默认为”cn.wps.moffice.demo” ），此信息将被固化到 Sever 中。

Agent 必须按照以下 AIDL 实现一个接口，Server 将调用此接口中的方法，获取可用的 Client 的列表。

AIDL 定义：

OfficeServiceAgent 对象及方法

package cn.wps.moffice.agent
interface OfficeServiceAgent
{
int getClients(out string[] clients, out int[] expireDays);
boolean isValidPackage(String originalPackage, String realThirdPackage)
}

AIDL 说明： getClients 方法 方法说明：用于获取一个 json 数据包和 Client 连接的过期时间。 首先，要判断由 WPS 传入的包名是否可信（可信判断由业务系统方自己决定，如判断包名是否等于”cn.wps.demo”等），根据判断结果，如果为 true，则返回包含 Client 认证列表的 JSON 数据包；如果为 false，则返回空的 json 数据包。 clients 参数：返回一个认证信息列表，其中包含了一个或多个 Client 的认证信息。 expireDays 参数：返回一个过期时间。在次过期时间之内，Server 将不会再次调用此方法来更新认证信息列表。此参数是为了避免每次启动时都需要进行一次认证信息列表获取动作，以提高启动速度。此参数若小于等于 0，则会导致每次启动时都需要进行一次获取动作。若大于 0，则在此数字规定的天数之后，Server 会再次调用此方法来更新认证信息列表。 isValidPackage 方法： 该方法是为了验证多个第三方包名定制，特殊需要，如果不需要可以直接返回 false

注：
获取第三方包名代码如下：

```
Bundle bundle = intent.getExtras();
m3rdPartyPackageName = bundle.getString(Define.THIRD_PACKAGE);
```

Agent 可以将 Client 认证信息列表固化在自身程序内部，也可以实现为连接到某个有效的服务器获取。 这由业务系统方自行决定。

不论如何，返回给调用者的 Client 认证信息列表必须符合以下格式要求：

```
{
   "name"：String Value,
    "type" : "Package-ID",
    "id" : String Value,
    "Security-Level" : "Full-access",
    "Authorization" : String Value
 }
```

格式说明:

name：业务流程处理程序名称,目前为 client 对应 Service 的 intent-filter 的参数。

type：调用方式，目前必须为"Package-ID"。

id： 根据调用方式确定的唯一名称，目前必须为 Client 的程序包名。

Security-Level：目前必须为"Full-access"

Authorization：认证码，将会与 Client 程序返回的认证码作对比，用于验证 Client 程序有效性。

注：这些参数中最重要的是 name，因为这个将是绑定 Client 的依据，其次是 Authorization，这个参数会在绑定 client 之后校验。

假设 Agent 存放（或通过其他方式获取的）的 Client 信息为下表内容：

信息列表				
name	type	id	Security-Level	Authorization
King Test App1	"Package-ID"	cn.wps.moffice.test.App1	Full-access	"abxxdsewrwsds3232ss"
King Test App2	"Package-ID"	cn.wps.moffice.test.App2	Full-access	"abxxdseqqds3ssw32ss"
King Test App3	"Package-ID"	cn.wps.moffice.test.App3	Full-access	"affxdseqqds3ssws2ss"

则调用上述 API 后返回的结果为如下 json 格式的数据包：

4.2 Client 规范

Client 需要根据一套 AIDL 接口描述来实现一系列接口，以供 Sever 调用。

Sever 在获得 Agent 的 Client 认证列表后，会试着连接列表中的每个 Client，直到有一个成功连接。

Client 主要实现有三部分组成：OfficeServiceClient 、OfficeAuthorization、OfficeClientEventListener。

OfficeServiceClient 返回 OfficeAuthorization 和 OfficeClientEventListener 两个类的实例，以便进行操作。

OfficeAuthorization 主要是返回该 Client 的验证码给 Sever 以验证 Client 的有效性。

OfficeClientEventListener 主要是实现响应 Sever 发送的事件（如打开文件）。

下面是这些类的 AIDL 详细定义：

OfficeServiceClient 对象及方法
package cn.wps.moffice.client; import cn.wps.moffice.client.OfficeAuthorization; import cn.wps.moffice.client.OfficeEventListener;

```
interface OfficeServiceClient
{
    OfficeAuthorization getAuthorization();
    OfficeEventListener getOfficeEventListener();
}
```

OfficeAuthorization getAuthorization()
Client 实现此接口被 Server 调用，返回值为 OfficeAuthorization 对象，当 Client 不在服务列表时，返回值为 null,则当前 Client 对象不会得到任何 Server 的服务及调用请求。
OfficeEventListener getOfficeEventListener()
Client 实现此接口被 Server 调用，返回值为 OfficeEventListener 对象，当 Client 不在服务列表时，返回值为 null,则当前 Client 对象不会得到任何 Server 的服务及调用请求。

OfficeAuthorization 对象及方法

package cn.wps.moffice.client
interface OfficeAuthorization
{
int getAuthorization(out String[] auth_code);
}

AIDL 说明： getAuthorization 方法： 方法说明： 用于向调用者返回自身的认证码 。Server 调用此方法，检测授权码是否正确。如果不正确，该 Client 将不会接受到任何事件。

OfficeClientEventListener 对象及方法
package cn.wps.moffice.client

interface OfficeEventListener
{
int onOpenFile(in String path, in OfficeOutputStream output);

int onSaveFile(in OfficeInputStream input, in String path);
int onCloseFile();
}

onOpenFile(in String path, in OfficeOutputStream output)
onOpenFile 方法： 方法说明：打开文件事件，打开文件时会调用此方法。 path 参数：文件路径。 output 参数：Client 对 path 所对应的文件解密，将解密结果写入到 output 对象中，写入完成之后，返回 0。解密失败返回-1. 返回值：返回值为-1 时，Server 将告诉用户，文件无法打开。
int onCloseFile()
onCloseFile 方法： 方法说明：关闭文件事件。
int onSaveFile(in OfficeInputStream input, in String path)
方法说明：保存文件事件，保存文件时会调用此方法。 path 参数：文件路径。 input 参数：Client 从 input 对象中读取数据，对 path 对应的文件进行加密，加密完成之后，保存到 path 指向的路径，返回 0。加密失败返回-1。 返回值：返回值为-1 时，Server 将告诉用户，文件无法保存。
boolean isActionAllowed(String path, ActionType type)
方法说明：wps 外部编辑事件功能，即外部控制 wps 中某些功能可否操作 Path 参数：当前打开文档路径 Type 参数：当前事件的类型 返回值：返回该事件是否可以执行。该方式是控制 wps 某些操作（如保存功能）可否进行，如果返回 false，则 wps 中文档不能保存，否则可以保存操作，如果不需要用到该功能，则直接返回 true
boolean isValidPackage(String originalPackage, String thirdPackage)
特殊定制需求，直接返回 false 即可。

注：OfficeInputStream 、OfficeOutputStream 类由 Sever 实现，在 OfficeClientEventListener 打开和保存文件方法使用时，可以像 FileInputStream、FileOutputStream 一样使用。