史上最全面 Android逆向培训之__Xposed使用

刚招来个Android，干了半个月辞职了，他走之后，成堆的bug被测了出来，都是这个新人代码都没看懂就开始改的一塌糊涂，还给提交了。
实在是让人头疼，清理了一个月多月才把他半个月写的bug清理个差不多。实在是得不偿失。
有了这个前车之鉴，不得不重视新人的岗前培训，毕竟面试找个来了就能上手的实在是太难了。
《写给新手入职的培训资料》-- by 齐浩 2019-07-18 18:20:30

> 教程目录:
> 第一章: Android JAVA 逆向基础
> 课时1 :Android环境配置与常用工具介绍
> 课时2 :调试方法及Smali文件结构
> 课时3 :新版本调试方法及Smali函数文件修改
> 实验3 新版本调试方法及Smali函数文件修改
> 课时4 :JD-Gui进行代码快速阅读分析
> 课时5 :实战演练如何去去除应用中的广告
> 课时6 :分析神器JEB使用方法
> 课时7 :常用Android快速定位关键点方法介绍
> 实验7 常用Android快速定位关键点方法介绍
> 课时8 :从0开始打造自己的破解代码库
> 第二章: Android Hook 插件开发
> 课时1 :Android 结构基础讲解
> 课时2 :快速Hook代码搭建之 Cydia Substrate
> 课时3 :快速Hook代码搭建之 Xposed
> 第三章: 阶段考核
> 课时1 :阶段考核
> 实验1 阶段考核
> 第四章: Android 系统编译
> 课时1 :安装部署Android源码编译环境
> 课时2 :Android源码目录结构与修改引导
> 课时3 :Android源码修改与刷机介绍
> 课时4 :Android Jni 编程
> 第五章: Android arm native 逆向
> 课时1 :arm 汇编代码讲解1
> 课时2 :arm 汇编代码讲解2
> 课时3 :arm 汇编代码讲解3
> 课时4 :arm 汇编代码讲解4
> 课时5 :arm 汇编代码讲解5

> 第六章：Android 应用初步编程保护
> 课时1 :class.dex文件格式讲解
> 课时2 :Android 动态代码自修改原理
> 课时3 :Android 动态代码自修改实现1
> 课时4 :Android 动态代码自修改实现2
> 第七章：Android 应用脱壳
> 课时1 :Android脱壳中的思路，技巧
> 课时2 :elf结构详解：动态运行库so文件的文件组成结构
> 课时3 :elf结构详解：加载so文件的流程
> 课时4 :elf文件变形与保护
> 课时5 :elf文件修复分析
> 课时6 :so加壳文件修复
> 课时7 :常用调试检测方法与过检测方法
> 课时8 :Android源码定制添加反反调试机制
> 课时9 :Android dvm 脱壳2
> 课时10 :Android dvm 脱壳3
> 课时11 alvik dex处理分析
> 课时12 :IDA脱壳脚本编写
> 课时13 :Odex修复方法
> 课时14 :IDAOdex修复脚本编写
> 第八章：Android 应用保护
> 课时1 :Android 加壳原理
> 课时2 :Android 加壳保护工具编写1
> 课时3 :Android 加壳保护工具编写2
> 课时4 :Android 加壳保护工具编写3
> 第九章：Android 应用hook
> 课时1：hook原理
> 课时2：root环境进行hook
> 课时3：ASP代码注入式写法
> 课时4：Android 免root进行hook
> 第十章：Android 虚拟机技术
> 课时1：虚拟机原理，多开原理讲解。
> 课时2：apk内部写一个虚拟机，在apk内安装apk
> 课时3：虚拟机适配各种手机
> 课时4：自己动手开发最简单的虚拟机代码

逆向课程都是我在公司内部培训新人时讲过的一些东西，内容再多太繁琐了，如果有同学喜欢学习这块内容，可以加我微信，都是免费的，在我这没啥秘密……欢迎共同探讨

官网地址： http://repo.xposed.info/

源码地址： https://github.com/rovo89

Xposed，Xposed的C++ 部分，主要是用来替换/system/bin/app_process，并为XposedBridge提供JNI方法
XposedBridge,Xposed 提供的jar文件，app_process启动过程中会加载该jar包，其他的Modules的开发都是基于该jar包
XposedInstaller,Xposed的安装包，提供对基于Xposed框架的Modules的管理

①Xposed 安装

首先，需要有一个可 ROOT 的 android 手机。
在 ROOT 的时候，需要对手机进行解锁。玩过ROOT的都知道。
注：如果你实在不知道怎么刷机，请直接找 比如:http://www.7to.cn/ ROOT市场走一圈就明白了。

Xposed 有几种安装方式：
1. 通过官网下载 http://repo.xposed.info/module/de.robv.android.xposed.installer
2. ROOT 过的手机，一把都自带有 Xposed ，并且是激活版（我的是小米 5S ）
3. 经过 Root 的手机应用市场也可以搜索到，直接搜索安装即可。但记得给 Xposed 给与 root 权限；
截止目前，Xposed 是需要 root 全部权限的。
另外，再附上一份安装方法（https://www.cnblogs.com/QUSIR/p/6912032.html）
如果你安装完成后，重启开机xposed页面时绿色的，那恭喜你。Xposed 框架安装成功!

②Xposed 仓库插件

在 Xposed 主界面的左上角点击，我们可以看到（我的版本为: version89），
在这里简单介绍一下各个菜单的用途：

框架：为 Xposed 主界面；
模块：主要是你自己开发的或从 Xposed 仓库
（仓库地址：http://repo.xposed.info/module-overview）中，下载安装的模块。
自己开发的模块也可以在这个位置中找到。注：稍后文章会重点介绍模块的开发流程。
下载：这里主要是与 Xposed 仓库地址打通的，仓库中提供了一下可下载安装的开源 Xposed 模块。
其中有一些比较常用的有：Alipay InstallB 支付宝装X模块、抖音插件等!
日志：主要为我们自己开发的模块在代码中打印的日志都会出现在这里。
日志很重要，方便我们调试和查看信息的关键入口，你也可以通过 Log.d 的方式来调试!
设置：主要是针对 Xposed 的简单配置。
重点设置对象为“禁用资源钩子”。在不影响使用的情况下，建议不要禁用!
支持和关于：略过。

③模块安装

这里以 抖音插件 为例，说明安装方法。
首先，进入到下载中，在搜索栏中输入“抖音插件”。
点击进入后。首先看到的是描述 Tab，
同时还可以看到源码和模块仓库的地址信息。
如果要安装，直接跳到 “版本” Tab 中，下载稳定版进行安装，安装成功后。
需要切换到 “模块” 菜单栏，并勾选自己刚才新安装的模块。
接下来重启手机，已使的插件生效并被 Xposed 框架加载！
接下来就可以正常使用该插件了。
切记，每次安装模块后，都需要先激活并且重启手机才能正常使用！

④开发环境搭建：

1、导入包：

```
compileOnly 'de.robv.android.xposed:api:82'
compileOnly 'de.robv.android.xposed:api:82:sources'//源码
```

2、AndroidManifest.xml中加入如下配置

```
<application
……略
>

    <meta-data
        android:name="xposedmodule"
        android:value="true" />
    <meta-data
        android:name="xposeddescription"
        android:value="卖客微信中控" />
    <meta-data
        android:name="xposedminversion"
        android:value="30" />

</application>
```

xposedmodule：value为true，表示自己是一个xposed模块
xposeddescription：value中的文字就是对模块的描述，这些能够在手机上的Xposed框架中看到，举个栗子
xposedminversion：xposed最低版本，这些应该都是向下兼容的吧？所以直接填最低版本好了

3、标记入口

在assets中new一个file，文件名为xposed_init（文件类型选text），
并在其中写上入口类的完整路径（下面是举例类路径，你们填自己的 如：com.longji.XposedHookUtil）

这样，xposed框架就能够读取 xposed_init 中的信息来找到模块的入口

注意事项:

请确保禁用Instant Run（File -> Settings -> Build, Execution, Deployment -> Instant Run），
否则你的类不会直接包含在APK中，导致HOOK失败！！！

以上是来自官方的警告，一定要注意，死活hook不了，日志也输出不了，就是因为这个！！！
如果你不这样做，你会惊喜地发现xposed日志反复给你抛出类似这样一个错误:
xposed didn't find class on dexpathlist：……（省略一长串）
而这个错误是你百度到死（google也没用）也不一定查的到解决方案的（这个故事提醒我们要仔细阅读官方文档）
在GitHub随便找个demo，打包运行Run成功后……

可以发现，打开xposed在模块里出现了我们的模块，勾选并且重启手机
到此，你就算是入坑xposed了。

当然，上面的例程比较简单，实际的项目要复杂得多，自己可以到github找些开源项目学习。

不懂的地方多多参照官方API文档:
http://api.xposed.info/reference/packages.html
https://github.com/rovo89/XposedBridge/wiki/Development-tutorial

常用hook方法举例

```
/**
* hook res资源和Java类需要实现如下两个不同的接口
* IXposedHookLoadPackage, IXposedHookInitPackageResources
*
* @author longji
* @date on 2019/5/4 15:28
*/

public class XposedHookUtil implements IXposedHookLoadPackage,
IXposedHookInitPackageResources {
String class_name = "com.longji.xposdedemo.MainActivity";
private Context context;
```

```
private String pkgName;
private String processName;
private String versionName;

// 开机时开始hook
@Override
public void handleLoadPackage(final XC_LoadPackage.LoadPackageParam lpparam) throws
Throwable {
pkgName = lpparam.packageName;
processName = lpparam.processName;

//hook 微信进程
if (processName.equals("com.tencent.mm")) {
Log.e("肉鸡信息2", "微信");
}

//获取当前上下文
context = (Context) XposedHelpers.callMethod(
XposedHelpers.callStaticMethod(
XposedHelpers.findClass("android.app.ActivityThread", null),
"currentActivityThread",
new Object[0]),
"getSystemContext",
new Object[0]);

//获取版本号
versionName = context.getPackageManager().getPackageInfo(pkgName, 0).versionName;
Log.e("肉鸡信息1", "pkgName:" + pkgName + ",processName=" + processName + ",versionName=" +
versionName);

// 举例如下
if (processName.equals("com.longji.xposdedemo")) {
//不能通过Class.forName()来获取Class , 在跨应用时会失效
Class clazz = lpparam.classLoader.loadClass(class_name);

// hook具体方法
XposedHelpers.findAndHookMethod(clazz, "getTTAd", new XC_MethodReplacement() {
@Override
protected Object replaceHookedMethod(MethodHookParam methodHookParam) throws Throwable {
Log.e("肉鸡信息2", "pkgName:" + pkgName + ",processName=" + processName + ",versionName=" +
versionName);
return "广告被拦截了";
}
});
}

if (processName.equals("com.youku.phone")) {
Log.e("肉鸡信息2", "优酷");
}
```

```
XposedHelpers.findAndHookMethod(
"android.widget.ListView",
lpparam.classLoader,
"setAdapter",
Class.forName("android.widget.ListAdapter"),
new XC_MethodHook() {
@Override
protected void afterHookedMethod(MethodHookParam param) throws Throwable {
super.afterHookedMethod(param);
if (param.thisObject instanceof ListView) {
XposedBridge.log(param.args[0].toString());
}
}
});

}

// 在资源布局初始化时进行hook
@Override
public void handleInitPackageResources(XC_InitPackageResources.InitPackageResourcesParam
resparam) throws Throwable {
Log.e("肉鸡信息22", "PackageResources");
resparam.res.hookLayout(resparam.packageName, "layout", "activity_main", new
XC_LayoutInflated() {
@Override
public void handleLayoutInflated(LayoutInflatedParam liparam) throws Throwable {
Log.e("肉鸡信息22", "PackageResources");
}
});
resparam.res.hookLayout(resparam.packageName, "layout", "view_demo", new XC_LayoutInflated()
{
@Override
public void handleLayoutInflated(LayoutInflatedParam liparam) throws Throwable {
XposedBridge.log("hook view_demo");
}
});
}
}
```