# 1 前言

为了增加APK文件的破解难度，很多比较重要的功能都是通过native实现，与反编译不同的是，java的伪代码可以很清楚的得到程序的逻辑关系，但是so文件反编译得到的是汇编代码，使用ida这样的神器可以得到c的伪代码。使用ida实现so的动态调试，可以在关键地方下断点，对于一些变换，比如有的时候输入的内容与经过一些列的变换后的结果比较，这个时候我们不需要关心中间的变换过程，在动态调试的时候，在比较的地方下个断点，然后在内存中查看此时比较的常量。正好最近做了CTF的一个re。顺便熟悉下ida的动态调试步骤。

# 2 APK逻辑的简单分析

在apk的manifest文件里，我们得到apk的入口avtivity。包括包名和主类。

通过一步步分析，得知：用户输入 用户名和 密码， 其中用户名必须是 <string name=”username”>secl-007</string> ，而密码的判断逻辑是调用了native函数（Java\_com\_ssctf\_seclreg\_Seclo0o\_getpl ）去判断，这个时候用ida静态分析，打开apk加载的so文件，通过分析，我们发现对于输入的密码没有进行任何处理，只是判断输入密码的长度，如果长度是39，然后和一个常量做比较，是否正确。而这个常量在运行过程经过一些列的变化，因此我们不需要关心变化的过程。有两个方法：

1. 输入任意长度为39的内容，然后dump整个程序的内存，这个时候在dump内容中寻找，（因为我们知道这个常量特定方式SSCTF{\*\*\*\*\*）
2. 使用IDA动态调试，在比较的地方下断点，得到此时常量的内容。这篇文章我们使用第二种方法实现

# 3 IDA动态调试SO

关于IDA动态调试步骤，首先你的手机需要ROOT。

1. 把ida目录下的android\_server文件放到手机的目录，adb pull android\_server /data/local/tmp
2. 给与android\_server 777 权限， chmod 777 android\_server
3. 端口的转发 adb forward tcp:23946 tcp:23946
4. 打开ddms adb forward tcp:8700 **jdwp:5996**
5. 运行android\_server ./android\_server
6. 将要调试的apk文件以调试的方式运行 adb shell am start -D -n com.ssctf.seclreg/.Seclreg   包名/.类名
7. 这个时候手机显示等到调试连接模式，打开ida，debugger-attach-remote android server
8. 选择ok，并且F9运行，这个时候，jdb附加程序，jdb -connect com.sun.jdi.SocketAttach:hostname=127.0.0.1,port=8700
9. 这样程序就可以动态调试了，我们根据SO基址和函数的偏址找到调试时候函数的地址，然后F2 下断点，

这个时候在APK输入，然后程序运行到我们设置断点的地方，

然后我们在数据窗口里查看数据，得到FLAG：）

原文地址： https://www.ijiza.cn/2016/03/03/android%E4%B8%ADso%E6%96%87%E4%BB%B6%E5%8A%A8%E6%80%81%E8%B0%83%E8%AF%95/