**loki方案破解**

**0x00 原理**

Loki 的R解决方案由如下五个exploit来实现:

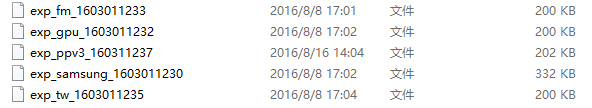


图1

通过逆向分析可知，这五个exploit在R完之后，都会去调用一个相同的静态链接的安装函数，如下图所示（我把它重命名为install）

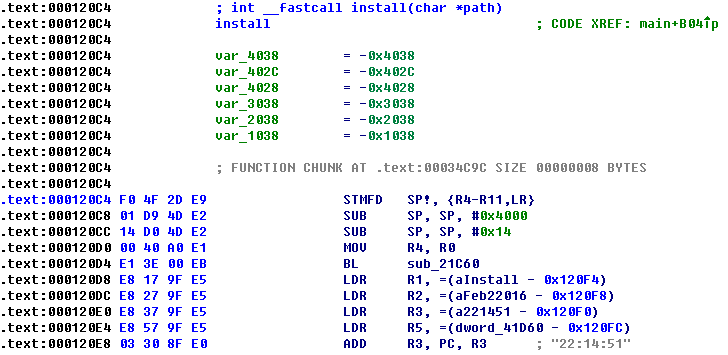


图2 ppv3的install函数

这个函数会释放loki的so文件以及安装apk文件到系统中，而在这个函数之前都是R逻辑，因此我们可以选择修改这个函数并调用execve来运行我们的程序，修改点选择在install这个函数上有如下两个好处：

1.可以去掉loki的安装流程，让exploit程序完全和loki业务无关

2.该函数的原型如下：

int \_\_fastcall install(char \*path)

参数path通过寄存器R0传入，并且这个path来源于命令行参数，我们可控。

**0x01 修改过程**

这里以PPV3为例，通过查找字符串"Loki-OTA not found (%s)\n"的交叉引用，很快能找到install函数，如上图2所示，我们选择在地址为0x120D4处插入我们的shellcode，shellcode的代码如下：

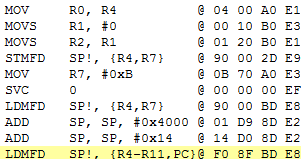


图3 shellcode

exceve函数的系统调用号为11（即0xB）:

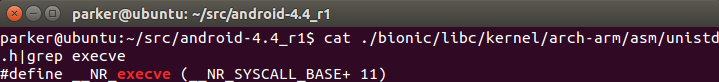


图4 execve的系统调用号

# 接下来我们需要把shellcode转化成arm机器码，我们可以用capstone，也可以用<http://armconverter.com/>提供的Online ARM To Hex Converter

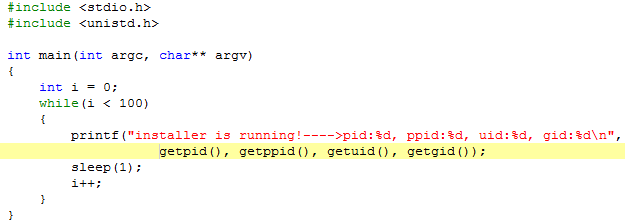
# C:\Users\thinkpad\AppData\Local\Temp\C8D7.tmp.png

# 然后我们找到地址0x120D4对应的文件偏移即offset=0x20D4处修改之，ppv3修改前后对比如下：

# C:\Users\thinkpad\AppData\Local\Temp\3379.tmp.png

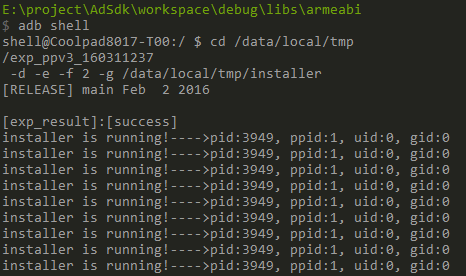
# 0x02 测试运行

我们先编写一个简单的测试程序installer，让它不断打印当前进程的uid，gid等，如下：



我们把installer和修改后的ppv3传到测试手机上并运行，运行结果如下：

./exp\_ppv3\_160311237 -d –e –f 2 –g /data/local/tmp/installer



uid和gid变为0，说明R成功并且运行了我们的程序。