燕十二的BLOG

个人资料

justFWD

访问： 81241次
积分： 1172
等级： BLOG 4
排名： 千里之外

原创： 32篇 转载： 16篇
译文： 0篇 评论： 34条

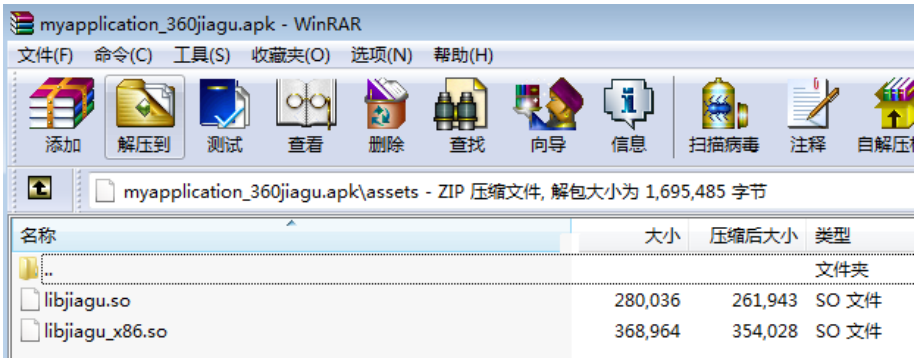
360加固之libjiagu.so脱壳及dex dump

标签： 360 加固 libjiagu.so dex dump

2015-11-17 13:57 8340人阅读 评论(5) 收藏 举报

分类： Android逆向（27）

360加固后的apk，在arm设备上首先会将assets目录下的libjiagu.so拷贝到files目录下,然后通过libjiagu.so动态加载原始dex



libjiagu.so的init_proc和init_array都无实质功能，真正的解密放在JNI_OnLoad里面

```
:00006630                       JNI_OnLoad                              ; DATA XREF: _:off_5DC↑o
:00006630 01 00 2D E9                           STMFD           SP!, {R0}
:00006634 D0 01 8F E2 04 00 80 E2               ADRL            R0, loc_6674
:0000663C 04 00 40 E2                           SUB             R0, R0, #4
:00006640 10 FF 2F E1                           BX              R0 ; loc_6670
:00006644                       ; ---------------------------------------------------------------------------
:00006644 02 00 F0 00                           RSCEQS          R0, R0, R2
:00006648 40 03 00 00                           ANDEQ           R0, R0, R0,ASR#6
:0000664C
:0000664C                       loc_664C                                ; DATA XREF: _:000024C0↑o
:0000664C 58 03 00 00                           ANDEQ           R0, R0, R8,ASR R3
:00006650 01 00 BD E8                           LDMFD           SP!, {R0}
:00006654
:00006654                       ; =============== S U B R O U T I N E =======================================
:00006654
```

JNI_OnLoad函数里有非常多的垃圾跳转指令干扰分析，后面还会有动态解密的函数运行，如果检测到调试器，会把动态函数置成非法代码，使程序CRASH

12306之梆梆加固libsec (2841)

评论排行

抢红包神器免费VIP版 (7)
win10 系统版本号获取的 (6)
360加固之libjiagu.so脱 (5)
fiddler Android下https抓 (4)
12306 Android客户端的 (4)
改造 Cydia Substrate 框 (2)
12306 2.2版本SO的分析 (2)
抢红包神器免费VIP开发 (1)
阿里系UTDID库生成唯一 (1)
让APK只包含指定的ABI (1)

最新评论

阿里系UTDID库生成唯一性ID分
Jugg_VS_Spe: 厉害啊。我的哥。讲解的很清楚。原理不复杂，但要寻根问底找到这里的代码可不容易。

fiddler Android下https抓包全攻
mux2: 厉害了 我的哥 OK了

让APK只包含指定的ABI
啊炆: 谢谢分享，支持一个

win10 系统版本号获取的三种方
shmiloveyou: @justFWD:可能是我的用法有问题吧，据知6.2.9xx也是Win10的版本号

win10 系统版本号获取的三种方
justFWD: @qq2399431200:我的测试平台也是win7,可以这么用啊。这个函数是调用ntdll里面的R...

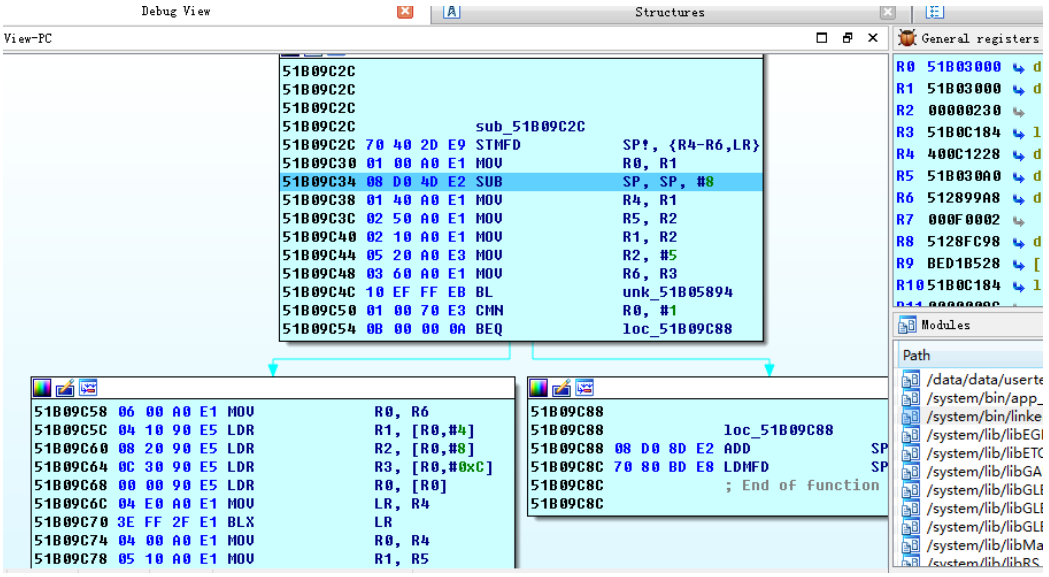
win10 系统版本号获取的三种方
shmiloveyou: @justFWD:不好意打错了，我的开发平台是Windows7，测试平台有Win XP/7/8/8....

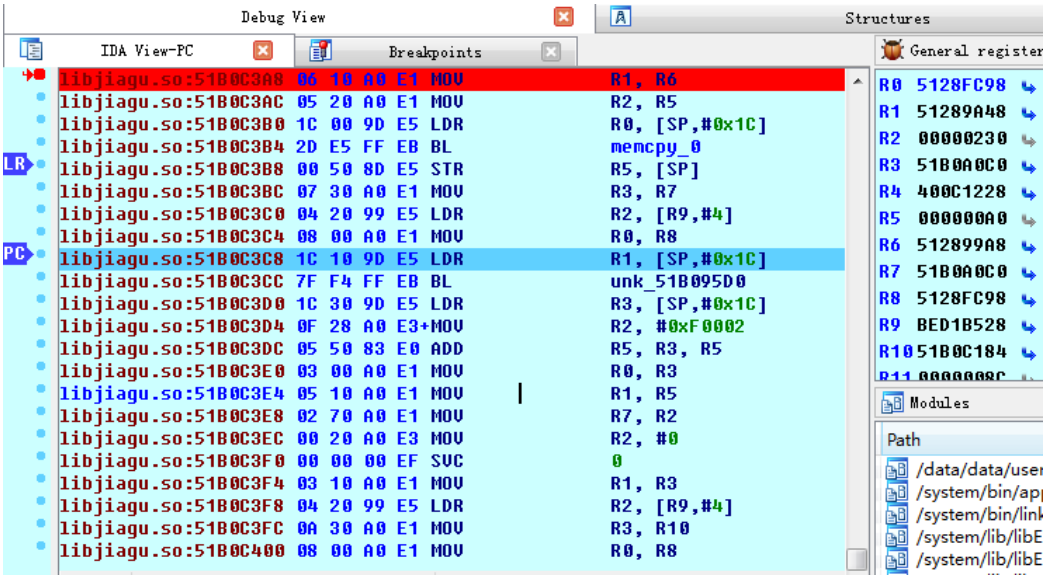
win10 系统版本号获取的三种方
shmiloveyou: @justFWD:我的测试平台为Windows 7首先包含VersionHelpers.h摘取了您...

win10 系统版本号获取的三种方
justFWD: @qq2399431200:不会吧，系统提供的头文件也不行？你是怎么用的？

win10 系统版本号获取的三种方
shmiloveyou: 尝试过上面的第三种方法，没能成功获取正确的Win10版本号，在此提供一个读取注册表的方法，已经验证过...

360加固之libjiagu.so脱壳及dex
a87462272: 膜拜！





51b0c404这个位置就会跑飞。这样分析比较累人。换个思路，对一些关键函数进行hook,进行hook log分析

Hook dlopen和dlsym LOG打印出libjiagu.so的加载基址和大小

```
dlopen /data/data/usertest.com.myapplication/files/libjiagu.so
dlopen_after addr:400c10f9 retV:400d2838 R0:5129bb80 R1:00000001 R2:0
dlopen after:libjiagu.so base:51b3b000 size:46000
libjiagu base:51b3b000 size:46000
raw func:400c1065
jmp Addr:51b3a001
dlsym_before addr:400c1065 retA:408ca9fb R0:400d2838 R1:4090e87b R2:5
dlsym_after addr:400c1065 retV:51b41630 R0:400d2838 R1:4090e87b R2:59
dlsym JNI_OnLoad funcP:51b41630  cont:e92d0001
```

可以看到JNI_OnLoad的地址还是在libjiasu.so内存范围内的

Hook RegisterNatives函数，打印如下LOG

```
RegisterNatives_before addr:408b5985 retA:51e2f17d R0:40c46850 R1:80f00029 R2:51e6d004 R3:00000002 Arg0:40c46850
RegisterNatives name:interface7 sig:(Landroid/app/Application;Landroid/content/Context;)Z Func:51e2f211
fwrite:43f0e92d fb01461f
open:b503b40e f4419903
write:e92d0090 e3a07004 ef000000 e8bd0090
start dump image:/data/[redacted]51e25000-4b000__interface7.dmp
memSize:4b000 pageLeft:0
leftSize:4b000 writeCount:4b
RegisterNatives_after addr:408b5985 retV:ffffffff R0:40c46850 R1:80f00029 R2:51e6d004 R3:00000002 Arg0:40c46850
```

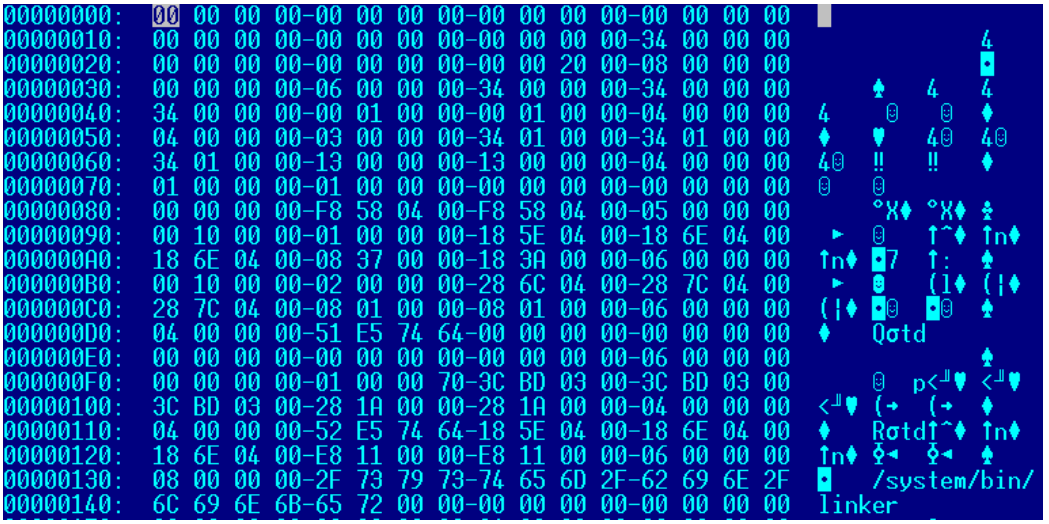
这个LOG可以看到RegisterNatives有被调用，注册了一个interface7函数,函数地址是51e2f211,仔细看这个地址发现，这个地址已经是thumb指令集了，跟libjiagu.so使用的arm指令集不太相符，而且可以看到函数地址已经超出了libjiagu.so内存范围了.

libjiagu.so的基址是51b3b000,大小是46000,最大内存地址是51B81000

那么，基本可以确定，它在内存中加载了另一个so

从/proc/%pid%/maps里找到函数地址所在的内存块，把周围连续的内存一起dump出来

在这片内存里找到如下一块内存



这其实就是elf heaer了，不过一些elf结构信息已经被抹掉了。除了基本的结构信息还缺失

如下三个数据：

0x20 e_shoff

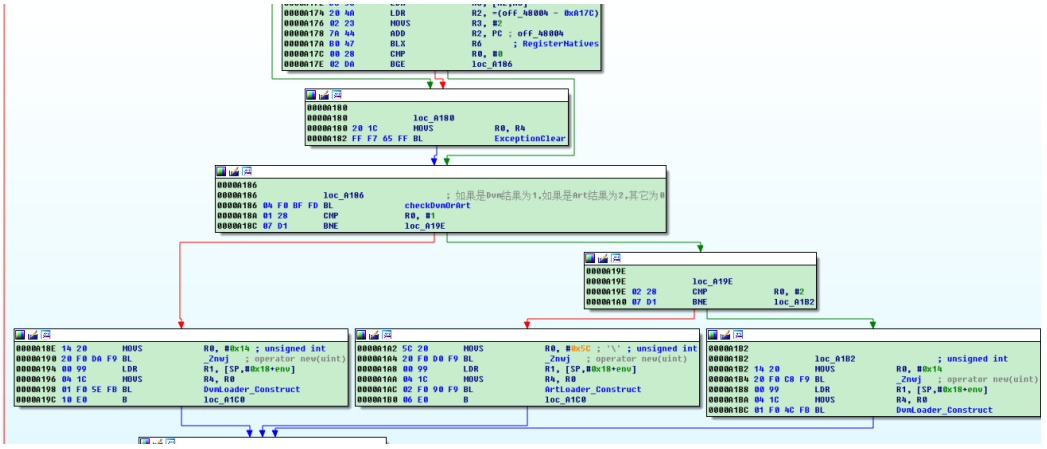
0x30 e_shnum

0x32 e_shtrndx

接下来的工作就是对这个残缺的内存进行修复了，通过分析上面三个数据，dlopen其实都没有用到，其值对elf运行没有影响。

经过修复后的文件可以替换掉原来的libjiagu.so，重新签名后使apk正常运行，用于IDA动态调试分析。

下面是经过修复后的libjiagu.so用ida分析的流程图片段



360在dex加载的时候，并没有调用dvmDexFileOpenPartial，而是自实现了此函数的功能，因此直接在这个函数上下断点是不能脱壳的。它的实现方式基本是照抄了dalvik源码

```
DexFile* dexFileParse(const u1* data, size_t length, int flags)
{
    DexFile* pDexFile = NULL;
    const DexHeader* pHeader;
    const u1* magic;
    int result = -1;

    if (length < sizeof(DexHeader)) {
        ALOGE("too short to be a valid .dex");
        goto bail;      /* bad file format */
    }

    pDexFile = (DexFile*) malloc(sizeof(DexFile));
    if (pDexFile == NULL)
        goto bail;      /* alloc failure */
    memset(pDexFile, 0, sizeof(DexFile));

    /*
     * Peel off the optimized header.
     */
    if (memcmp(data, DEX_OPT_MAGIC, 4) == 0) {
        magic = data;
        if (memcmp(magic+4, DEX_OPT_MAGIC_VERS, 4) != 0) {
            ALOGE("bad opt version (0x%02x %02x %02x %02x)",
                 magic[4], magic[5], magic[6], magic[7]);
            goto bail;
        }
```

如上面源码所示，使用dex所在内存的时候都会使用memcmp校验MAGIC是否为dex,所以只要hook memcmp，然后在过滤函数里校验是否为dex，然后dump出来即是原始的dex

(创建了一个**Android**逆向分析群，欢迎有兴趣的同学加入，群号码:376745720)

顶 2　　踩 0

上一篇　让APK只包含指定的ABI

下一篇　360加固之libjiagu.so dump修复

我的同类文章

Android逆向（**27**）

- 关于烧饼游戏修改器的分析　2017-01-09　阅读 87
- 某梆企业版加固脱壳及抽代…　2016-04-15　阅读 6670
- Android安全–从defineClas…　2016-03-17　阅读 715
- uleb128、sleb128和uleb1…　2016-03-08　阅读 475
- 如何无keystore获取apk的…　2016-02-26　阅读 524
- 抢红包神器免费VIP版　2016-01-14　阅读 1109
- Android smalidea无源码调试　2016-09-07　阅读 725
- 深入理解Android（二）：J…　2016-03-23　阅读 720
- 12306 2.2版本SO的分析和…　2016-03-15　阅读 1657
- Dex文件格式　2016-03-08　阅读 555
- 阿里系UTDID库生成唯一性…　2016-01-20　阅读 6428

更多文章

参考知识库

Android知识库

30834 关注 | 2623 收录

猜你在找

ARM裸机程序开发入门
GPIO和LED-1.4.ARM裸机第四部分
ARM那些你得知道的事儿-1.1.ARM裸机第一部分视频课程
1.14.ARM裸机第十四部分-LCD显示器
NandFlash和iNand-1.11.ARM裸机第十一部分
360加固保动态脱壳
Apk脱壳圣战之---脱掉360加固的壳
12306之梆梆加固libsecexeso的脱壳及修复
鬼哥的一次简单脱壳实例dex
安卓 dex 通用脱壳技术研究四

错误

查看评论

5楼 a87462272 2016-07-06 15:52发表

膜拜！

4楼 sluoyeqiu 2016-05-12 22:33发表

楼主你好，360的壳现在使用了so混淆，ida打开什么都看不到，请问您是怎样对混淆的so进行修复的呢？

3楼 杨锁锁 2016-05-01 11:58发表

感谢分享，为楼主点个赞！

2楼 tbfly 2016-01-06 15:05发表

群主通过一下qq群申请啊

1楼 ctl3n 2015-12-09 14:13发表

博主的方法的确有效，拿到了脱壳的libjiagu.so，再修复一下就可以用IDA分析。厉害！

您还没有登录,请[登录]或[注册]

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack
VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery
BI HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML LBS Unity
Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack
FTC coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App SpringSide Maemo
Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP HBase Pure Solr
Angular Cloud Foundry Redis Scala Django Bootstrap