

今天国庆的第六天，西安淅淅沥沥小雨下个不停，索性卧在床上不如品一杯西湖龙井，更新一篇博客，一来帮助需要之人；二来加深自己的理解。

### 问题篇：

在上一篇中讲到了关于 Android so 的动态调试，没看的可以点这里：

[点击打开链接](#)；

我自认为写的还是挺全的，在上文中我们说到关于最后一步 jdb 附加调试时，很多时候都会出现附加不上的问题，使人很闹心。。。于是这一篇就是专门关于这个问题进行展开的。解决这个问题方法有很多，我是按照自己认为的优良答案顺序展开的，都是借鉴网上的各路大神而总结的。

### 解决篇：

根据 android 的官方文档，如果调试一个 APK，必须满足以下两个条件中的任何一个：

1. APK 的 AndroidManifest.xml 文件中的 Application 标签包含 `android:debuggable="true"`；
2. /default.prop 中的 ro.debuggable 的值为 1；

### 方法一:

在已经 root 的手机安装 Xposed 框架和 xinstaller 插件

目的: 就是利用 Xposed 的 HOOK 插件 xinstaller 开启系统中所有应用的调试功能。

使用方法:

第一步: 下载 Xposed 框架, 并激活, 再下载 xinstaller 插件安装;

第二步: 开启模块, 点击 xinstall 插件设置专家模式, 进入其他设置, 开启调试应用, 最后在 xposed 中激活重启, OK!!

关于框架和 xinstaller 插件会放在附件中。

### 方法二:

修改 `android:debuggable="true"`

用 AK 反编译以后在 `AndroidManifest.xml` 文件中的 `Application` 标签中加入 `android:debuggable="true"`; 然后回编译。

这个方法虽然简单, 但是问题多, 比如:

第一: 有的反编译, 签名验证等等。

第二: 如果说软件已经爱加密或加壳了, 修改 XML 几乎是不可能的, 因为改了也不能回包

于是引入了方法三.

### 方法三:

如果我们在真机，则可以修改根目录下的 default.prop 文件，将里面的 ro.debuggable =1。

第一： 从 Google 官方网站下载到 boot.img;

第二： 使用工具 (abootimg, gunzip, cpio) 把 boot.img 完全解开，获取到 default.prop;

第三： 修改 default.prop;

第四： 把修改后的文件重新打包成 boot\_new.img;

第五： 使用 fastboot 工具把 boot\_new.img 刷入设备 (fastboot flash boot boot\_new.img) ;

优点是可以永远调试，不再担心此问题;

缺点是我没有 Nexus 5 手机 😡，这个问题很严肃；于是引出了方法四。

#### 方法四:

我们没有谷歌的亲儿子，但是我们有神器的小工具。

首先我们看到如图所示 ro.debuggable=0;

```
1|root@android:/ # cat default.prop
#
# ADDITIONAL_DEFAULT_PROPERTIES
#
ro.secure=1
ro.allow.mock.location=0
ro.adb.secure=1
ro.debuggable=0
persist.sys.timezone=Asia/Shanghai
root@android:/ #
```

init 进程会解析这个 default.prop 文件，然后把这些属性信息解析到内存中，给所有 app 进行访问使用，所以在 init 进程的内存块中是存在这些属性值的，那么这时候我们可以利用进程注入技术，我们可以使用 ptrace 注入到 init 进程，然后修改内存中的这些属性值，只要 init 进程不重启的话，那么这些属性值就会起效。当然这个工具已经写好，我会放在后 main 附件中。

解决方法：

第一步：拷贝 mprop 到/data/目录下；

第二步：./mprop ro.debuggable 1；

第三步：getprop ro.debuggable；（查看此时 ro.debuggable 在内存中的值）

第四步：stop;start(重启 adbd 进程)；

```
shared
ssh
system
time
tombstones
user
wiper
wpstyles
root@android:/data # ./mprop ro.debuggable 1
properties map area: 401b4000-401c4000
00000000 ed 00 00 00 35 2c 00 00 50 52 4f 50 76 4f 43 45 ...5,..PROPvOCE
00000010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000020 00 08 00 10 80 08 00 0c 00 09 00 10 80 09 00 10 .....
00000030 00 0a 00 0b 80 0a 00 0b 00 0b 00 0b 80 0b 00 0d .....
00000040 00 0c 00 0b 80 0c 00 0b 00 0d 00 15 80 0d 00 0e .....
00000050 00 0e 00 0b 80 0e 00 0e 00 0f 00 15 80 0f 00 09 .....
00000060 00 10 00 16 80 10 00 0d 00 11 00 0d 80 11 00 14 .....
00000070 00 12 00 10 80 12 00 15 00 13 00 16 80 13 00 0b .....
00000080 00 14 00 13 80 14 00 1c 00 15 00 14 80 15 00 19 .....
00000090 00 16 00 18 80 16 00 0d 00 17 00 11 80 17 00 0d .....
000000a0 00 18 00 0d 80 18 00 0d 00 19 00 0d 80 19 00 10 .....
```

注意：

我们在上面的第三步的时候查看好像没有改过来，分析原因因为：

该工具是通过 ptrace 修改 init 进程中的内存，然而 4.X 系统强制开启了 selinux；因此这个时候我们需要设置 Selinux 的状态。需要个 APK，会放在后面的附件中。

**BY：雪一梦**

**2016.10.06**