

#### 1. Dex 转 Odex

优化: vm/analysis/Optimize.cpp --> dvmOptimizeClass

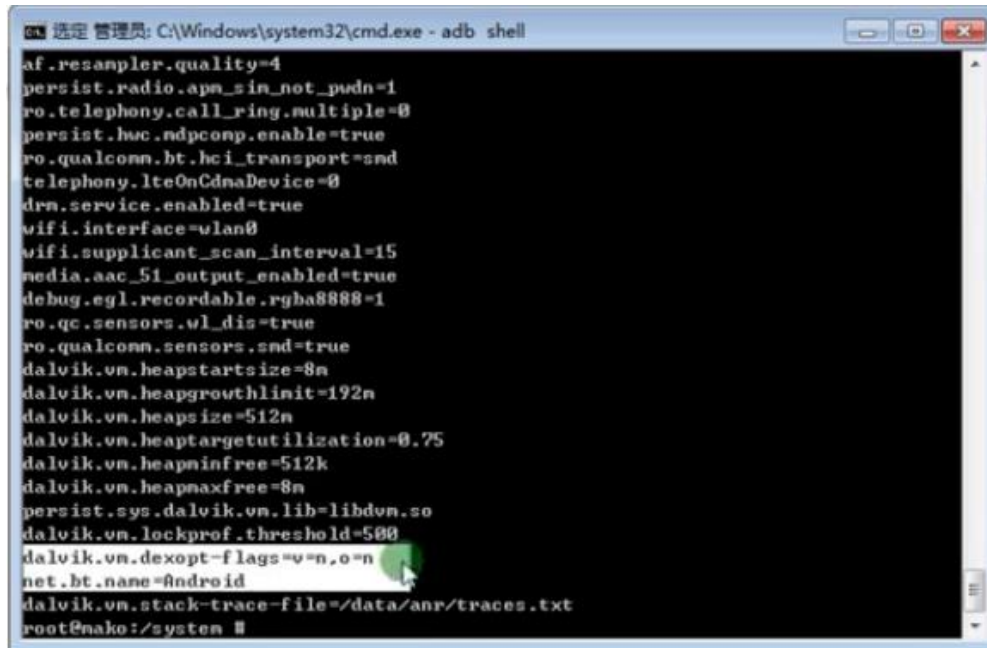
#### 2. Dex 校验

vm/analysis/DexVerify.cpp --> dvmVerifyClass

#### 3. 取消非必要优化与校验

/system/build.prop

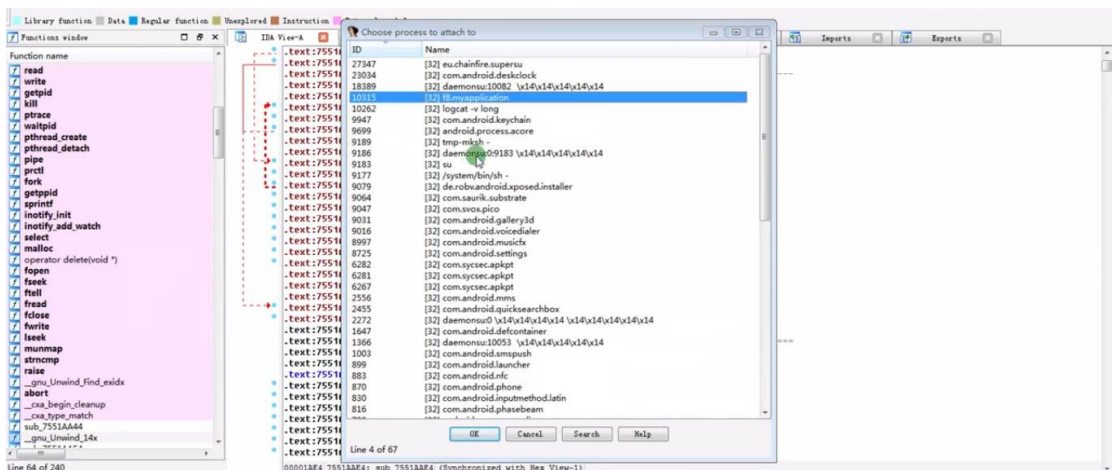
Dalvik.vm.dexopt-flag=v=n, o=n

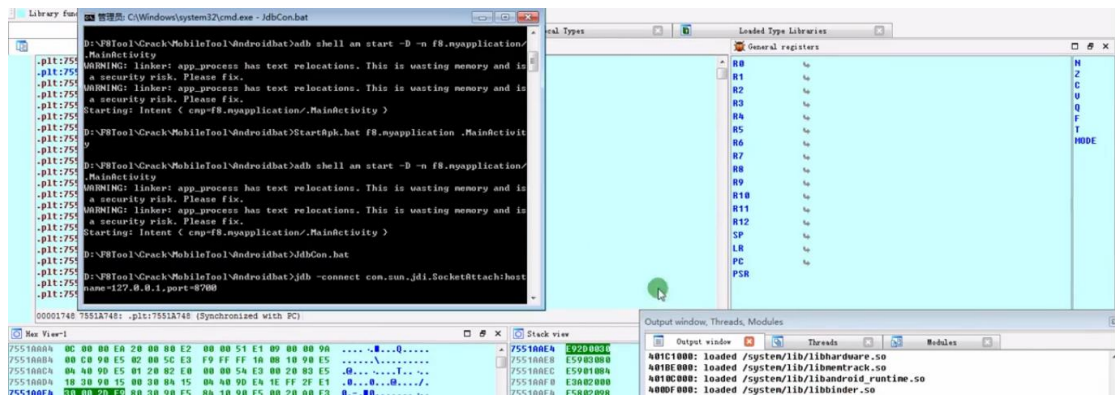


```
af.resampler.quality=4
persist.radio.apn_sim_not_pwdn=1
ro.telephony.call_ring.multiple=0
persist.hwc.ndpcomp.enable=true
ro.qualcomm.bt.hci_transport=snd
telephony.lteOnCdmaDevice=0
drm.service.enabled=true
wifi.interface= wlan0
wifi.suppliment_scan_interval=15
media.aac_51_output_enabled=true
debug.egl.recordable. rgba8888=1
ro.qc.sensors.wl_dis=true
ro.qualcomm.sensors.snd=true
dalvik.vm.heapstartsize=8m
dalvik.vm.heapgrowthlimit=192m
dalvik.vm.heapsize=512m
dalvik.vm.heaptargetutilization=0.75
dalvik.vm.heapminfree=512k
dalvik.vm.heapmaxfree=8m
persist.sys.dalvik.vm.lib=libdvm.so
dalvik.vm.lockprof.threshold=500
dalvik.vm.dexopt-flags=v=n,o=n
net.bt.name=Android
dalvik.vm.stack-trace-file=/data/anr/traces.txt
root@nako:/system
```

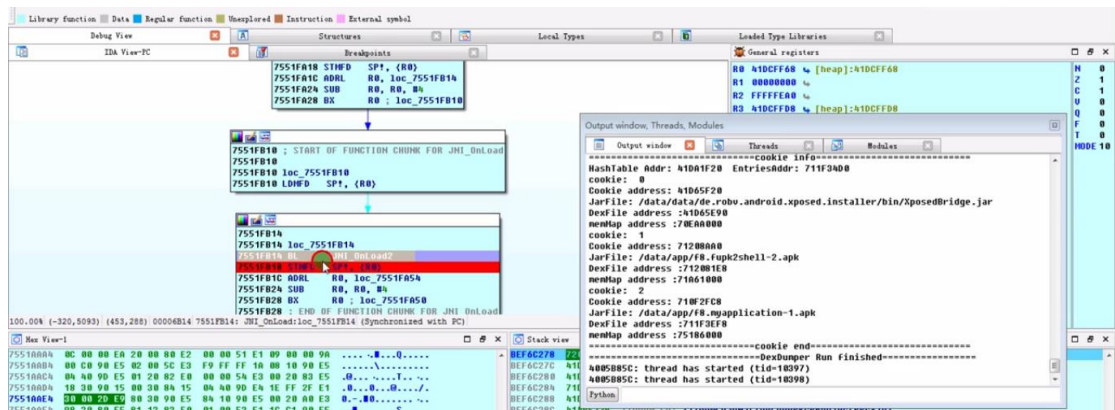
#### 4. Demo 演示 (360 壳)

寻找脱壳点(手动调试)

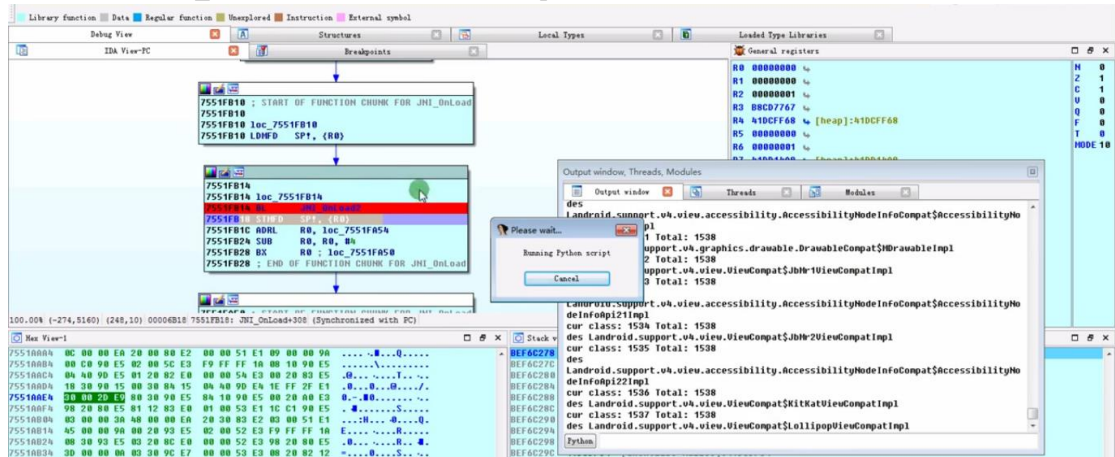




使用脚本分析加载了多少 dex 文件



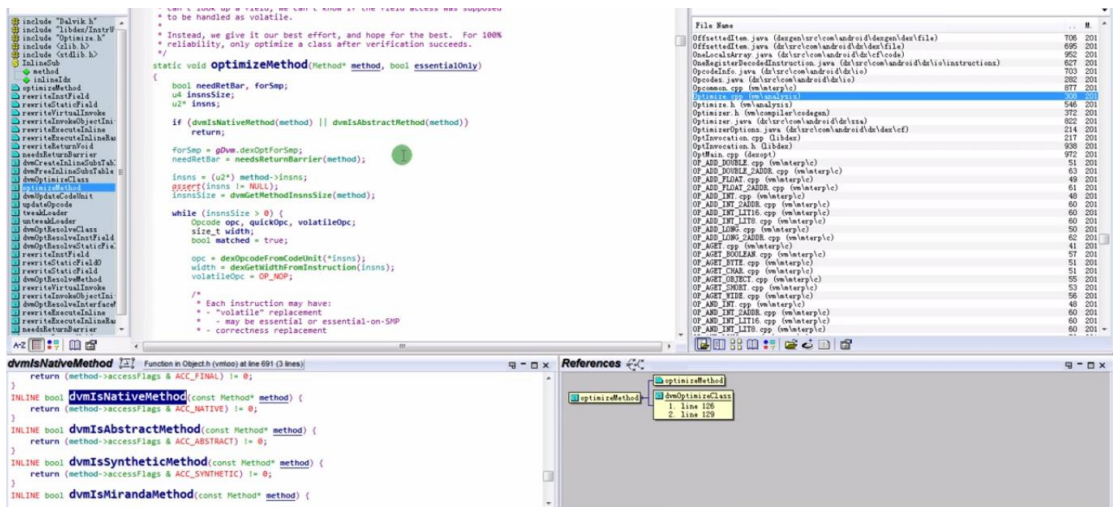
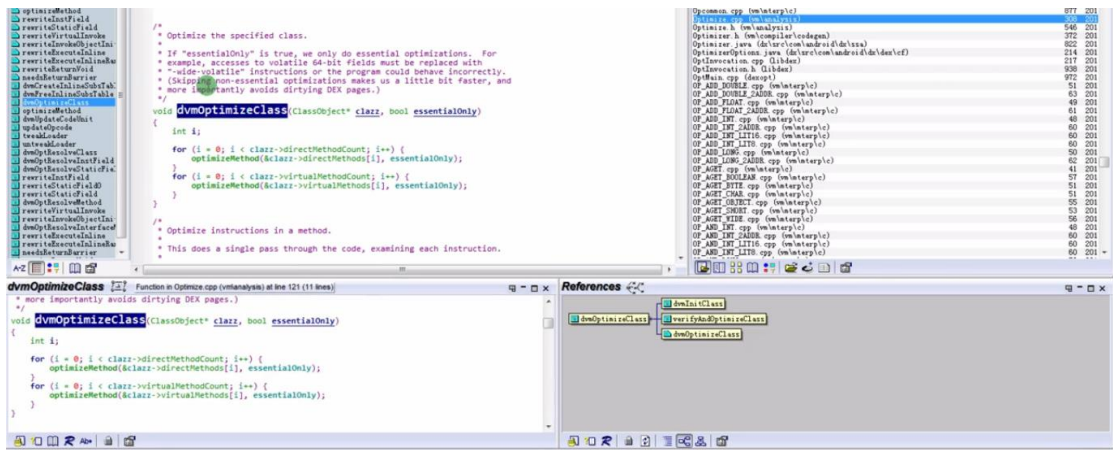
找到脱壳点 JNI\_OnLoad2 并执行脚本，dump 出原始 dex 文件



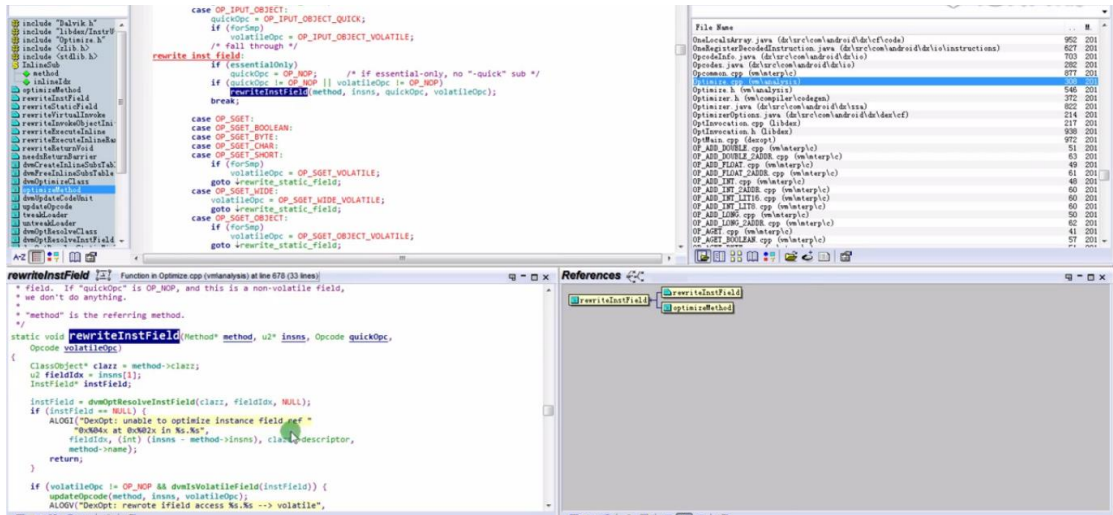
加载前后的 dex 文件数据比较



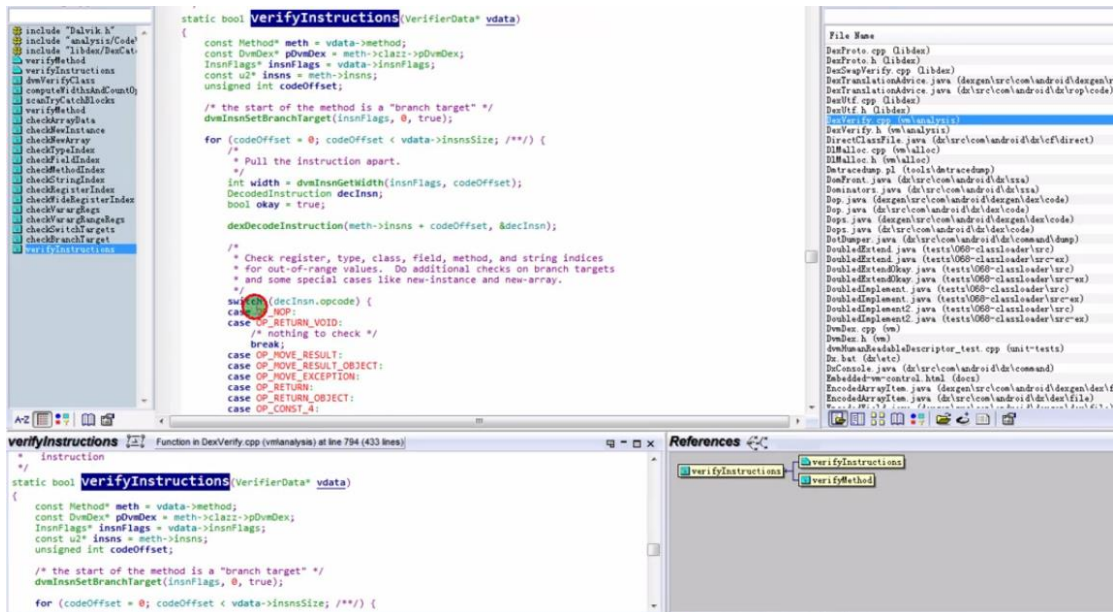
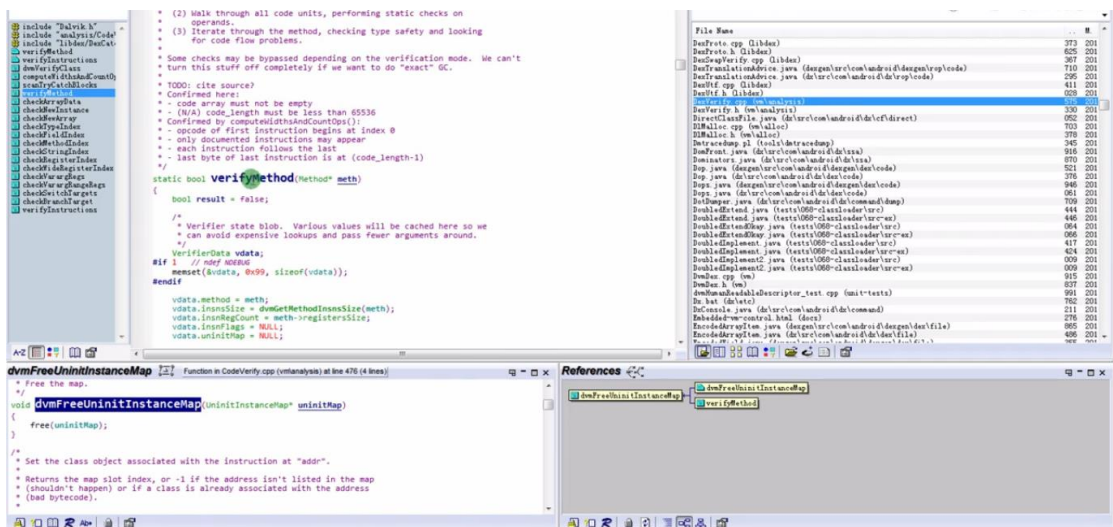
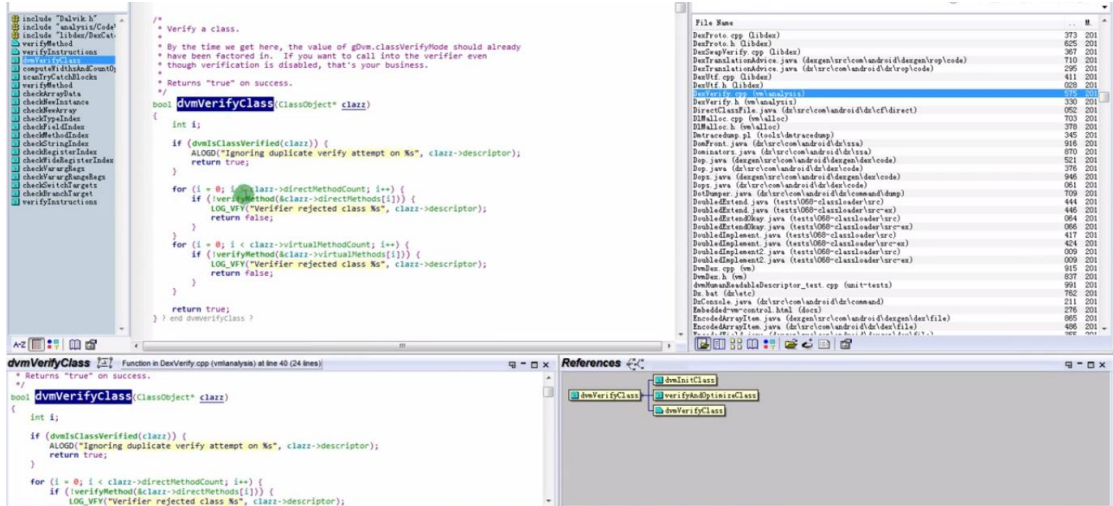




通过 while 中的 switch 判断来对所有能进行 opcode 的代码进行优化



## 6. 分析 dvmVerifyClass



如果校验失败则返回 false，显示校验失败并且不会加载