Hook技术在Android性能测试中的应用

MTSC2018

第四届中国移动互联网测试开发大会

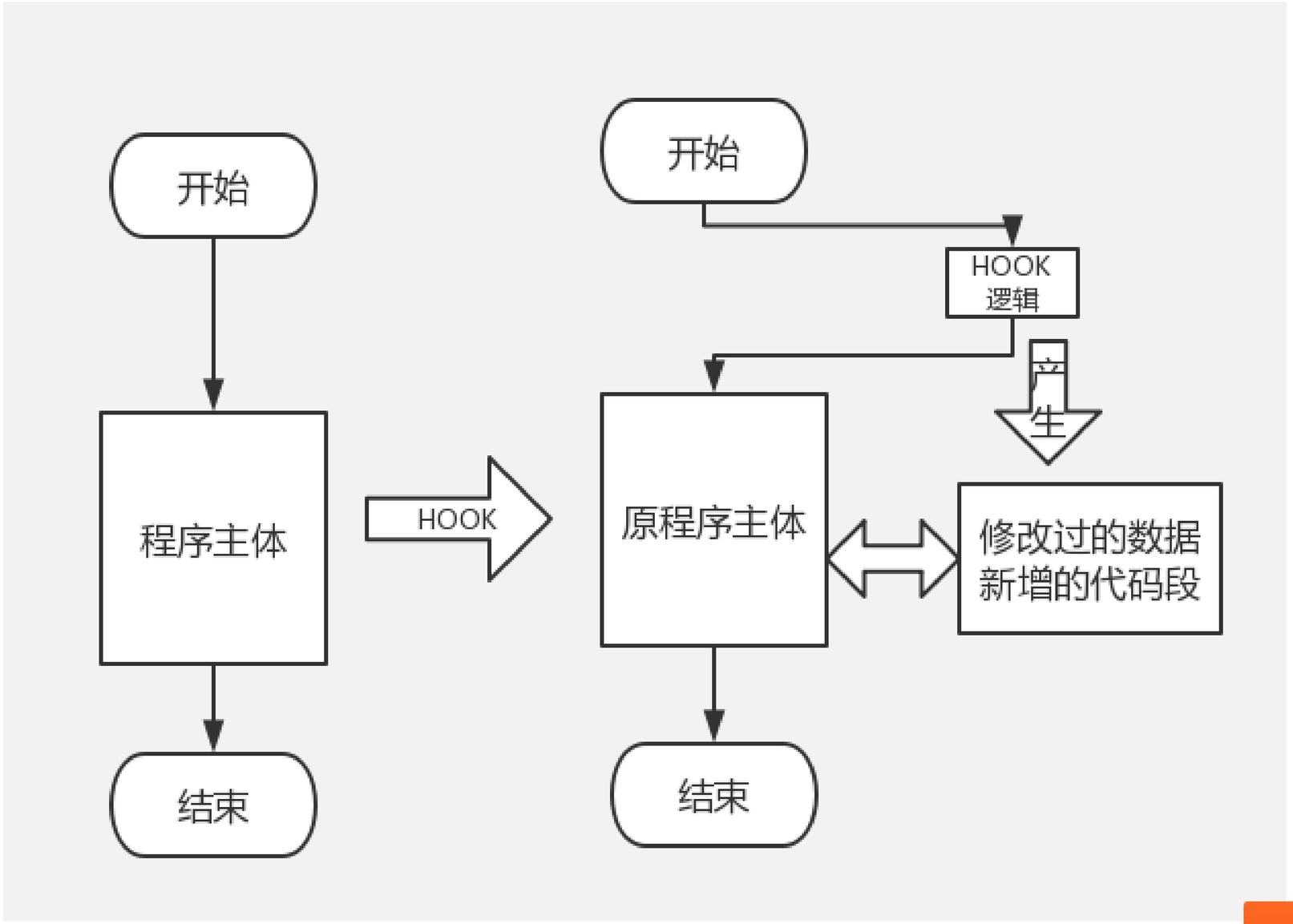
Hook技术介绍

Android性能

主线程耗时测试

Rom内存泄露测试

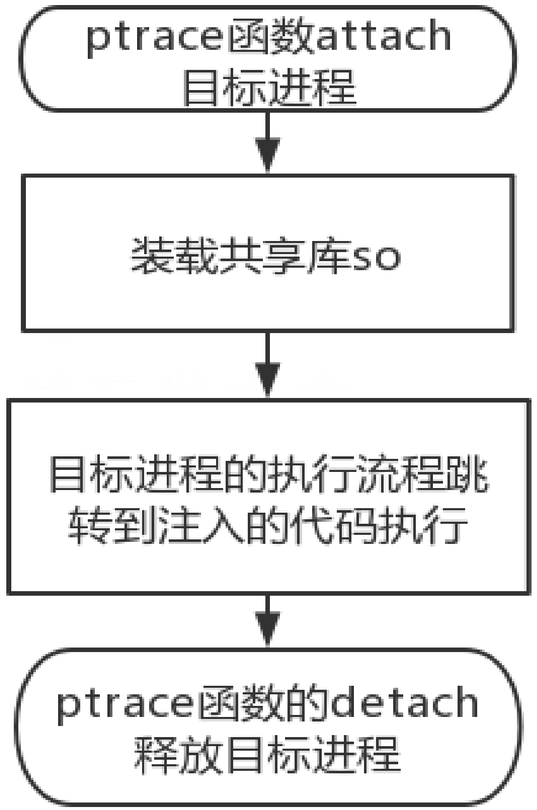
Hook技术介绍


开始
程序主体
结束
HOOK
开始
HOOK
逻辑
产
生
原程序主体
修改过的数据
新增的代码段
结束

Android相关内核函数：

ptrace_attach
ptrace_getregs
ptrace_call
ptrace_writedata
dlopen
mmap
ptrace_setregs
ptrace_detach

目标进程注入代码过程

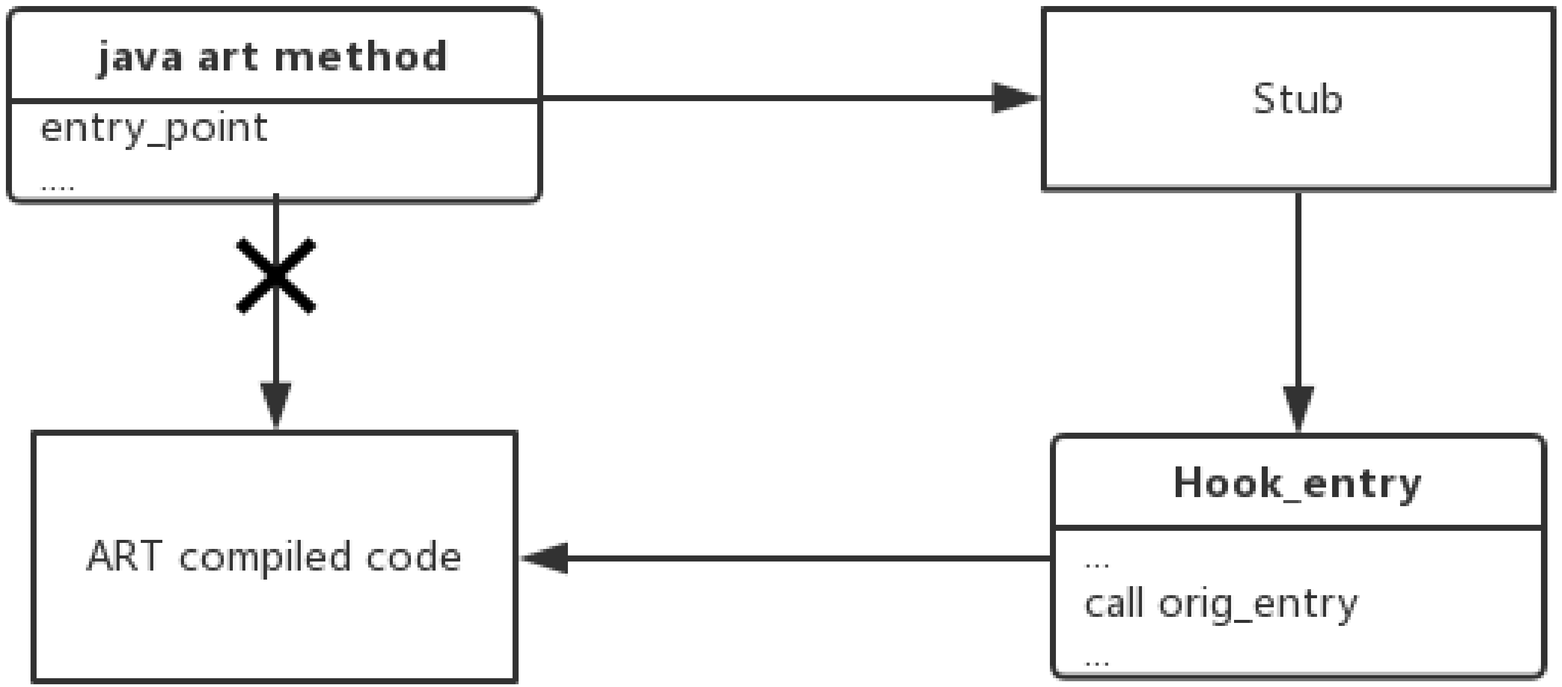


Hook在ART中的应用

Xposed框架

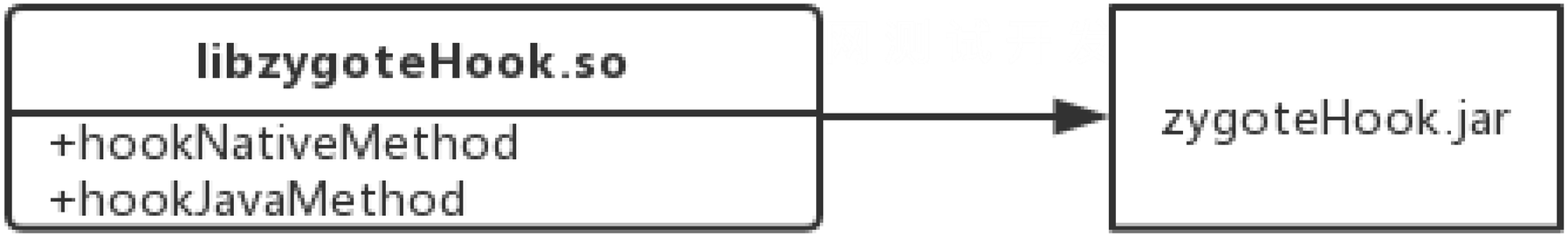
android java层 hook 框架

不需要修改apk



zygote_hook库的加载

ZygoteInit ->preloadSharedLibraries
|-> System.loadLibrary(libjnigraphics)
|-> dlopen
|-> libzygotehook
|-> hook JavaAPI



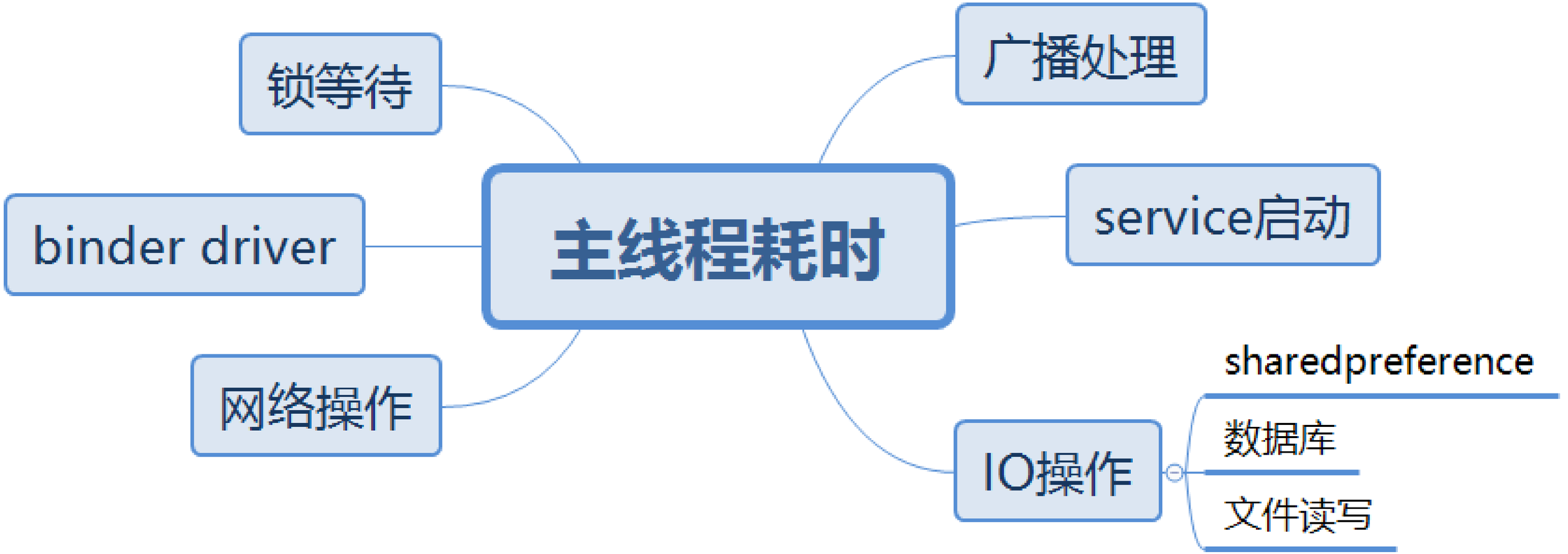
优势	不足
白盒测试	不稳定
无痕	其他开销
灵活	实现困难

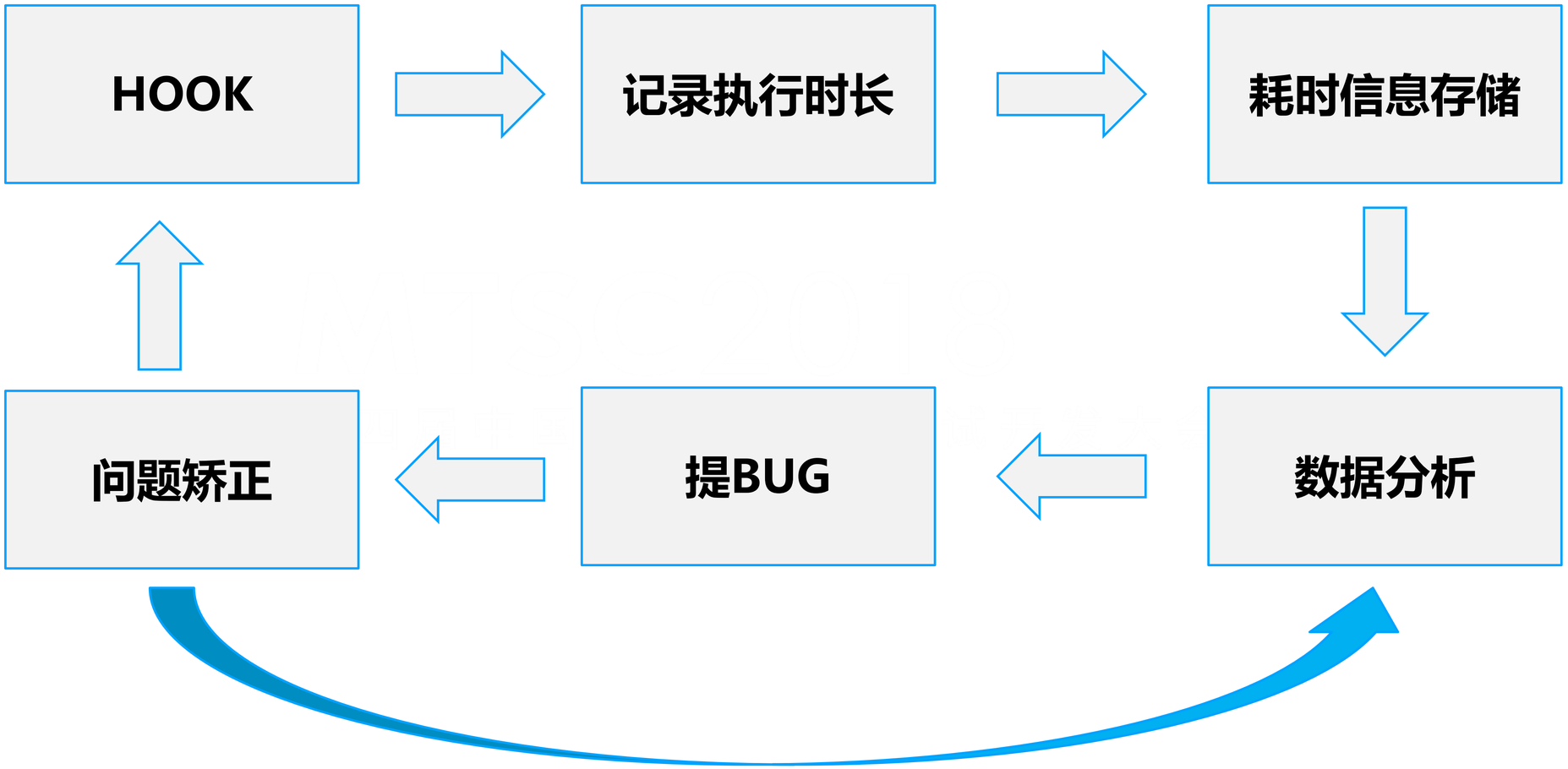
Android性能


Android rom性能
应用启动
冷启动
热启动
流畅度
帧率
绘制时长
其他响应时间
解锁时长
亮灭屏时间
跟手性
...

Android Rom性能测试中的问题	解决思路	方法
复现困难	一次性发现原因	在测试中自动分析问题 通过插桩找到耗时路径
问题太多，分析量大	测试左移	
同类问题居多	自动化分析原因	

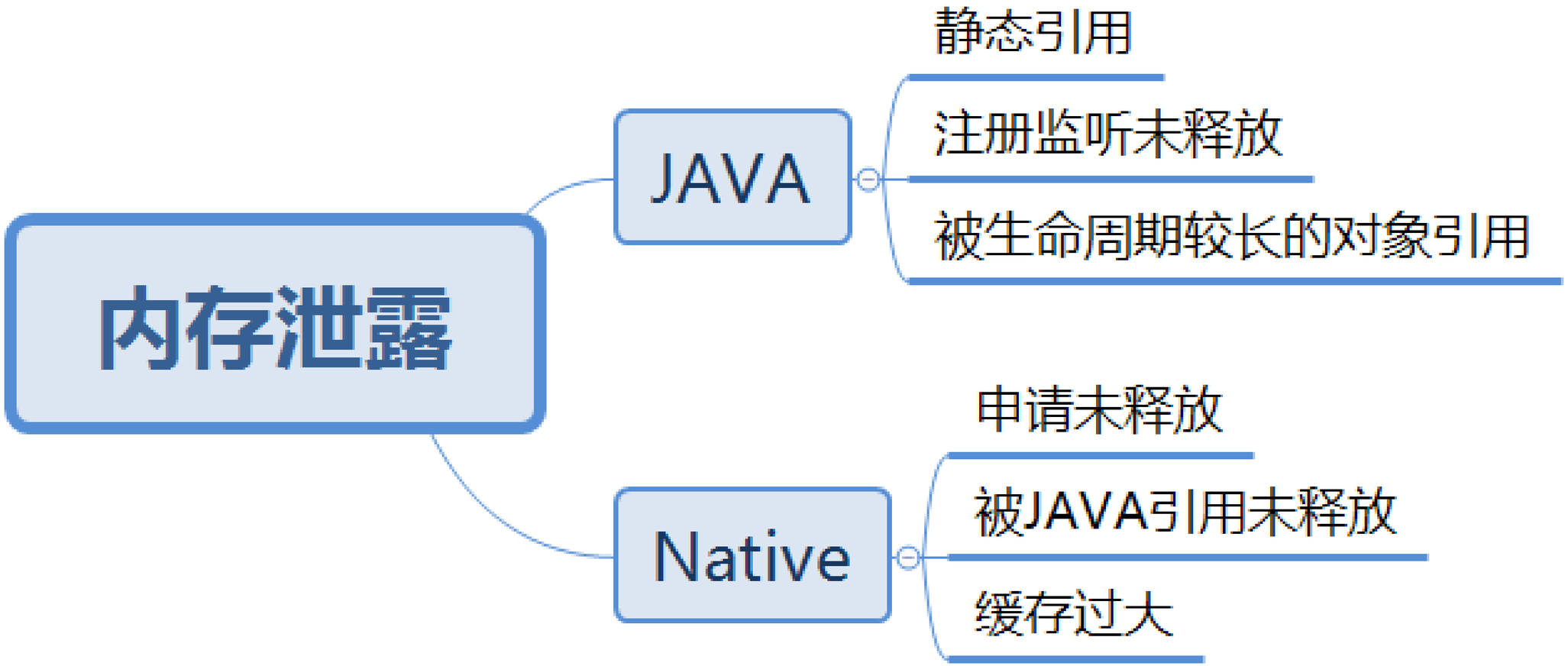
主线程耗时测试


主线程耗时
锁等待
广播处理
binder driver
service启动
网络操作
IO操作
sharedpreference
数据库
文件读写

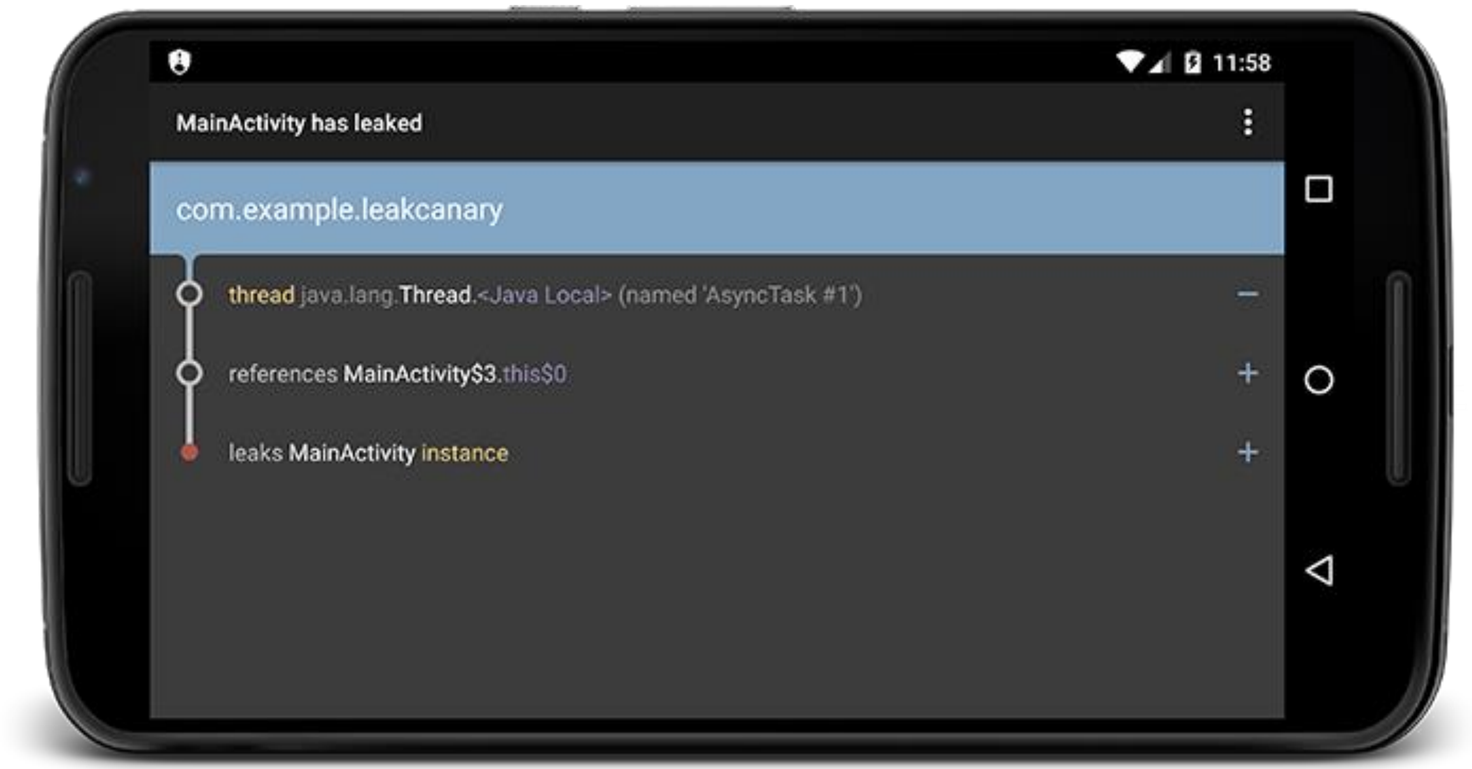

HOOK
记录执行时长
耗时信息存储
数据分析
提BUG
问题矫正

com.miui.[redacted]的mainThread 使用 onReceive 耗时打点统计结果如下：
app Version : v2017090290(M[redacted]o-UN)
最大耗时 : 1236ms
超bug(70ms)阈值次数： 1次
最大耗时 action&ReceiveClass :
com.xiaomi.[redacted]
$MMReceiver

Rom内存泄露测试


内存泄露
JAVA
静态引用
注册监听未释放
被生命周期较长的对象引用
Native
申请未释放
被JAVA引用未释放
缓存过大

JAVA泄露检测



监听activity等
关键对象生命周期

↓

HOOK 所有应用
检测关键对象

↓

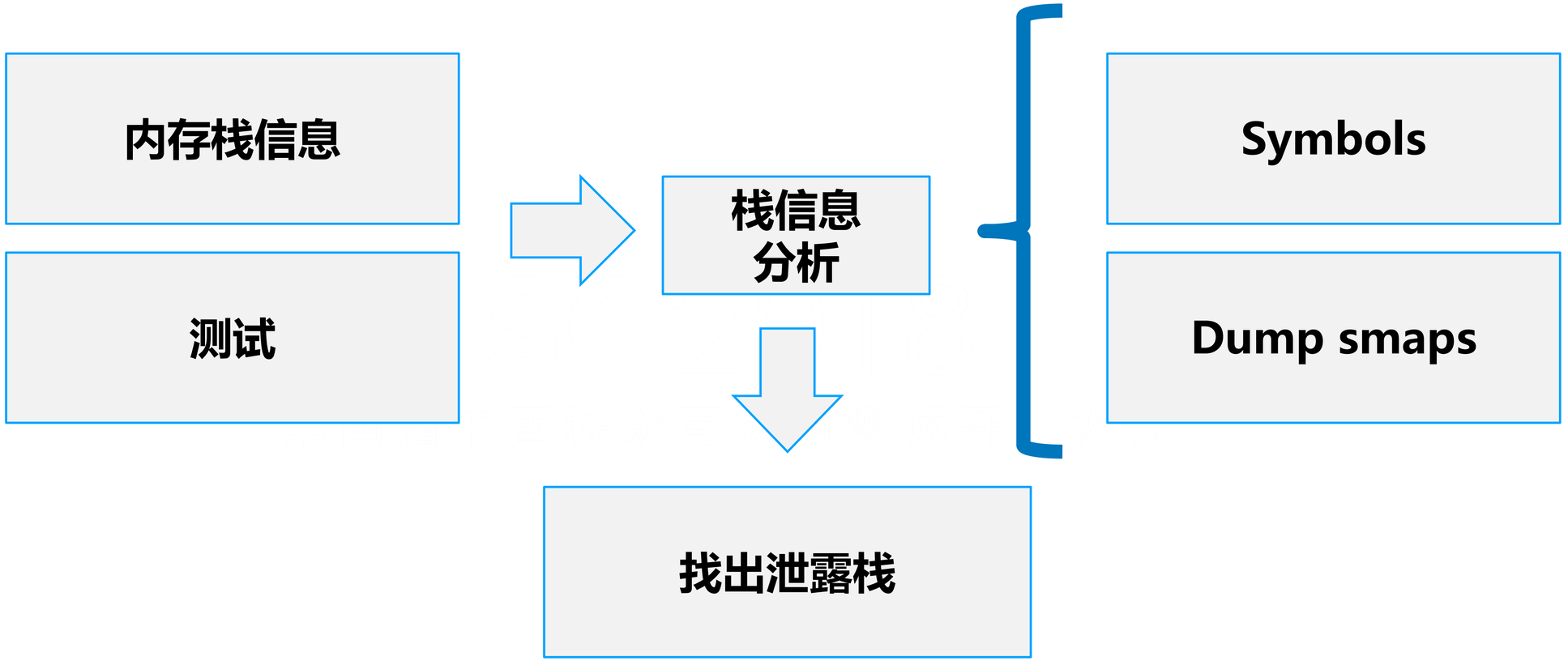
Monkey
用例测试

↓

输出泄露

测试结果:
1.Android Version: 7.1
2.Device: xxx
3.LeakCount: 5
4.LeakMemorySize: 70 KB
5.FileName: com.miui.xxxx_2018-01-25_21-09-36_494.hprof_leakInfo.txt
6.LeakInfoSummary: ['In com.miui.xxxx:v2018012090(Mixxxx-ROM):2018012090.', '* com.miui.xxxx.HomeActivity has leaked:', '* GC ROOT static android.app.ActivityThread.sCurrentActivityThread', '* references android.app.ActivityThread.mActivities', '* references android.util.ArrayMap.mArray', '* references array java.lang.Object[].[1]', '* references android.app.ActivityThread$ActivityClientRecord.nextIdle', '* references android.app.ActivityThread$ActivityClientRecord.activity', '* leaks com.miui.xxxx.HomeActivity instance']

Native泄露检测



libc malloc debug 工具

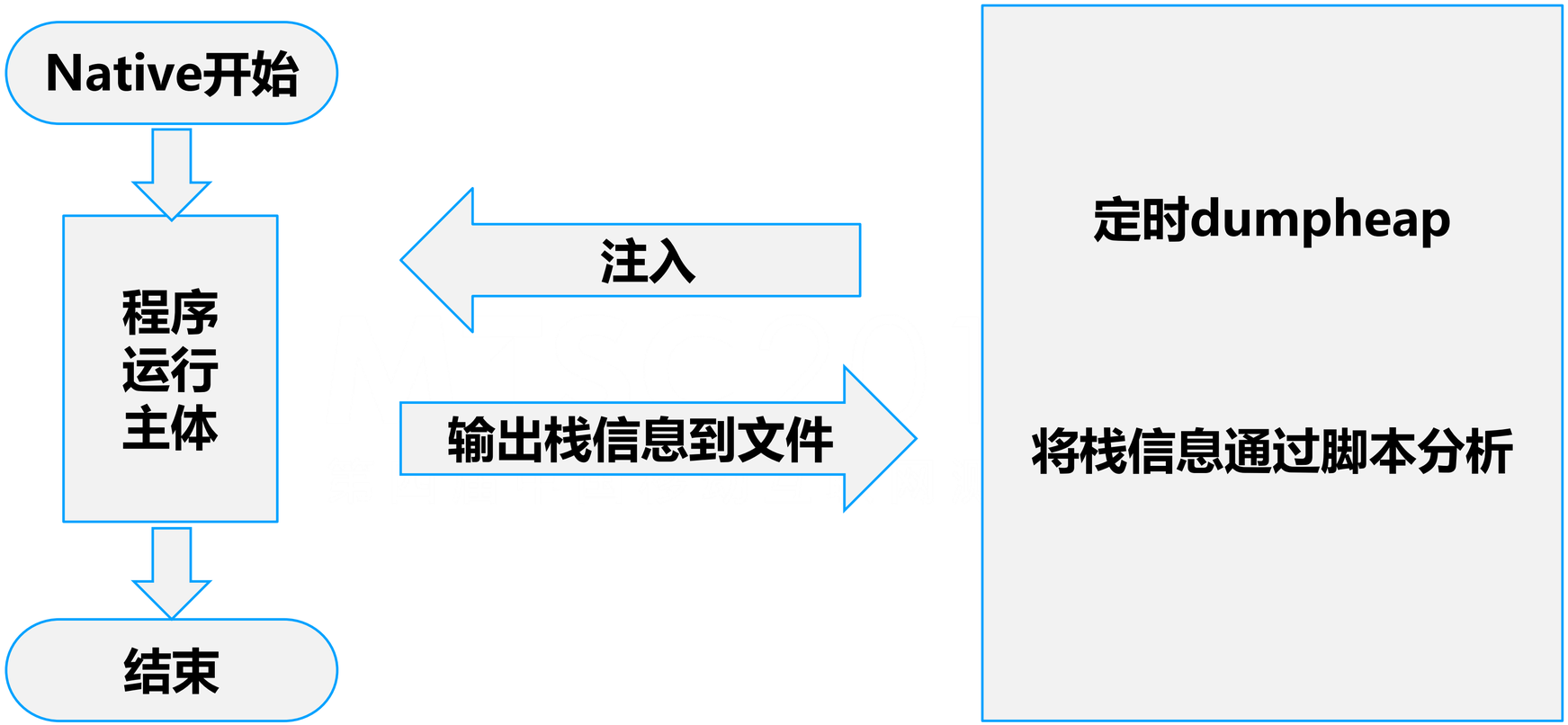
AOSP https://source.android.com/devices/tech/debug/native-memory?hl=zh-cn

```
adb shell stop
adb shell setprop libc.debug.malloc.program app_process
adb shell setprop libc.debug.malloc.options backtrace
adb shell start

#启动需要 debug 的 APP

adb shell am dumpheap -n <PID_TO_DUMP> /data/local/tmp/heap.txt
```

Native进程内存泄露检测



inject64 [pid] dumpHeap libinject.so
inject [pid] dumpHeap .so

nativeDiff脚本

Google native heapdump viewer	脚本修改
结合smaps可以找函数名的栈	内存大小排序
统计各个调用栈申请内存大小	内存现场对比
	运行过程中内存栈信息分析

```
Begin trace
size 3917520  count:5441    parentcnt:0

0x00008c0c    debug_malloc                    bionic/libc/malloc_debug/malloc_debug.cpp:310
/system/lib64/libc_malloc_debug.so
0x00024db4    get_wifi_radio_stats(wifi_radio_stat*, nlattr**)
hardware/xxxx/wlan/xxxx/wifi_hal/llstats.cpp:717    /system/lib64/libwifi-service.so
0x00023a28    WifiCommand::response_handler(nl_msg*, void*)
hardware/xxxx/wlan/xxxx/wifi_hal/cpp_bindings.cpp:688    /system/lib64/libwifi-service.so
0x0000a544    recvmsgs                        external/libnl/include/netlink-local.h:113
/system/lib64/libnl.so
0x00023920    WifiCommand::requestResponse(WifiRequest&)
hardware/xxxx/wlan/xxxx/wifi_hal/cpp_bindings.cpp:619    /system/lib64/libwifi-service.so
0x00025f1c    wifi_get_link_stats
…

End Trace
```

谢 谢!

MTSC2018

第四届中国移动互联网测试开发大会

TesterHome | IT大咖说

TesterHome