**一、直接Log 打印，栈打印**

1. 打印调用栈

(new Exception()).printStackTrace();

2. Hack aidl生成的java文件

将aidl生成的文件(从out)拷贝到对应的文件夹, 然后在里面加入log, 将aidl文件从Android.mk里删除掉。

**二、 内存泄露**

Step1: Use DDMS to monitor heap change

<http://developer.android.com/tools/debugging/ddms.html>

Step2: Dump hprof file and convert format

Use DDMS to dump.

The hprof-conv tool converts the HPROF file that is generated by the Android SDK tools to a standard format so you can view the file in a profiling tool of your choice.

 hprof-conv [-z] <infile> <outfile>

You can use "-" for <infile> or <outfile> to specify stdin or stdout.

You can use "-z" to filter out zygote allocations shared by all applications.

Step3: Use MAT to analysis hprof

<http://www.eclipse.org/mat/>

**Native layer memory leak**

* Identification

How to judge it's different from java layer memory leak?

1. No OOM Exception in java layer

2. Kernel OOM killer triggered unexpected. (E.g. 2GB 1080p device, but lot of process get killed)

* Debug command

adb shell cat /proc/meminfo

adb shell procrank

* Debug process

1. Start test

2. print status using the above 2 commands and capture log

3. Wait some time (half time of the repro should be enough)

4. execute step2 again

**Too Many File Open Debug Sequence**[https://wiki.nvidia.com/wmpwiki/skins/common/images/tech-check.gif](https://wiki.nvidia.com/wmpwiki/index.php?title=Software/General/Android_SW_Platform_Team/Android_Frameworks/JunSu&action=revalidatesection&anchor_name=Too_Many_File_Open_Debug_Sequence&valid=1&bust=2112839128%23Too_Many_File_Open_Debug_Sequence)

* Step1: Log into DUT, use the following command to get the first log.

lsof > 1.log

* Step2: Follow the repro sequence, use the following command to get the second log.

lsof > 2.log

* Step3: Find out the biggest difference between 1.log and 2.log. You can get the file which get not closed.

**ANR issue**[https://wiki.nvidia.com/wmpwiki/skins/common/images/tech-check.gif](https://wiki.nvidia.com/wmpwiki/index.php?title=Software/General/Android_SW_Platform_Team/Android_Frameworks/JunSu&action=revalidatesection&anchor_name=ANR_issue&valid=1&bust=517374727%23ANR_issue)

* Common Debug command

adb pull /data/anr/traces.txt

adb bugreport

adb shell perf stat top -n 1 (derived from a MTS related ANR on <http://nvbugs/1489518>)

* What is ANR, there are 3 types of ANR

Key dispatch timeout 5s

broadcast timeout 10s

service timeout 20s

* Why ANR happens

Main thread stuck with one message in the main message queue and can't process the next message.

What is main thread doing: **UI, touch event.**

* How to debug ANR issues

2 kind of Necessary information:

- /data/anr/traces.txt (Would be the best)

- adb bugreport result