Android 逆向工程筆記

工具簡介

基本工具

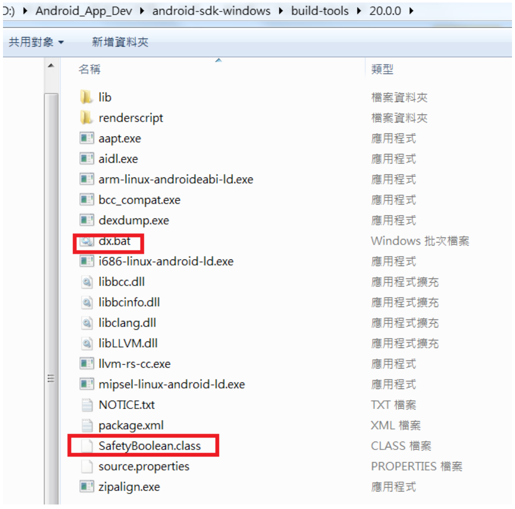
1. apktool: https://ibotpeaches.github.io/Apktool/
2. jd-gui: http://java-decompiler.github.io/
3. dex2jar: https://sourceforge.net/projects/dex2jar/
4. dedexer: http://dedexer.sourceforge.net/

SDK 工具

1. dx: <sdk-path>\build-tools\<sdk-version>
2. aapt: <sdk-path>\build-tools\<sdk-version>
3. dexdump: <sdk-path>\build-tools\<sdk-version>

範例

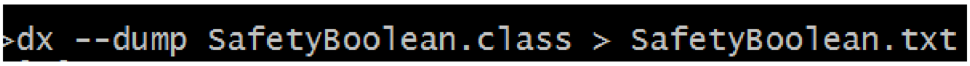
1. Dump class file



D:) ▸ Android_App_Dev ▸ android-sdk-windows ▸ build-tools ▸ 20.0.0 ▸

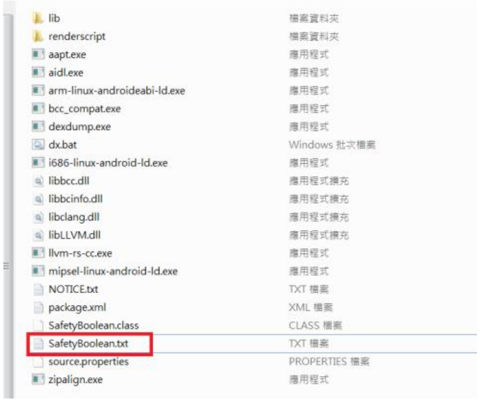
名稱	類型
lib	檔案資料夾
renderscript	檔案資料夾
aapt.exe	應用程式
aidl.exe	應用程式
arm-linux-androideabi-ld.exe	應用程式
bcc_compat.exe	應用程式
dexdump.exe	應用程式
dx.bat	Windows 批次檔案
i686-linux-android-ld.exe	應用程式
libbcc.dll	應用程式擴充
libbcinfo.dll	應用程式擴充
libclang.dll	應用程式擴充
libLLVM.dll	應用程式擴充
llvm-rs-cc.exe	應用程式
mipsel-linux-android-ld.exe	應用程式
NOTICE.txt	TXT 檔案
package.xml	XML 檔案
SafetyBoolean.class	CLASS 檔案
source.properties	PROPERTIES 檔案
zipalign.exe	應用程式

打開 cmd 並且下以下指令



```
>dx --dump SafetyBoolean.class > SafetyBoolean.txt
```

之後就會產生一個 dump file

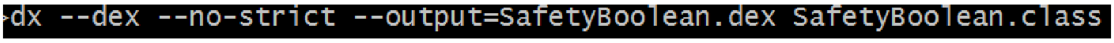


lib	檔案資料夾
renderscript	檔案資料夾
aapt.exe	應用程式
aidl.exe	應用程式
arm-linux-androideabi-ld.exe	應用程式
bcc_compat.exe	應用程式
dexdump.exe	應用程式
dx.bat	Windows 批次檔案
i686-linux-android-ld.exe	應用程式
libbcc.dll	應用程式擴充
libbcinfo.dll	應用程式擴充
libclang.dll	應用程式擴充
libLLVM.dll	應用程式擴充
llvm-rs-cc.exe	應用程式
mipsel-linux-android-ld.exe	應用程式
NOTICE.txt	TXT 檔案
package.xml	XML 檔案
SafetyBoolean.class	CLASS 檔案
SafetyBoolean.txt	TXT 檔案
source.properties	PROPERTIES 檔案
zipalign.exe	應用程式

2. Dump dex to file

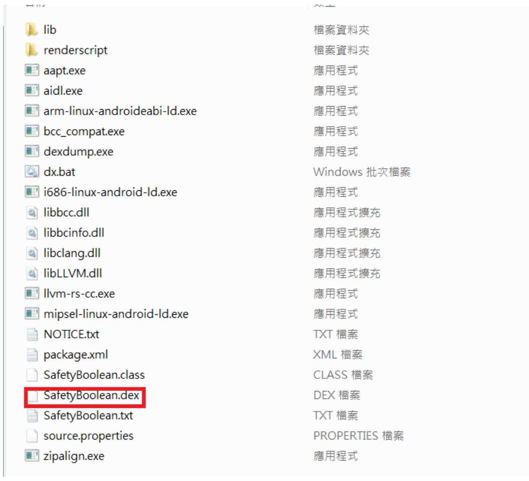
Class to dex file

打開 cmd 並且下以下指令



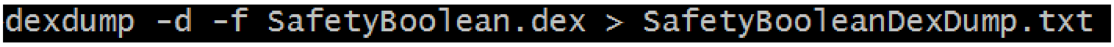
```
>dx --dex --no-strict --output=SafetyBoolean.dex SafetyBoolean.class
```

之後就會產生一個 DEX file



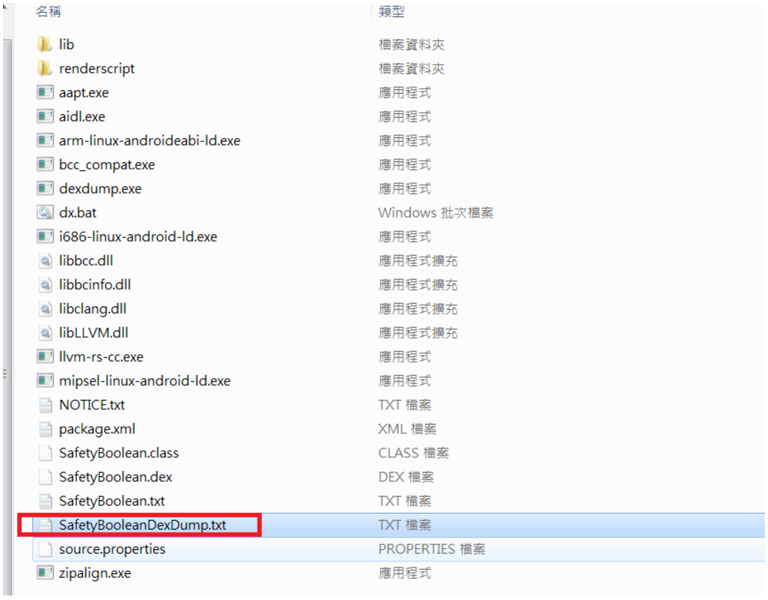
lib	檔案資料夾
renderscript	檔案資料夾
aapt.exe	應用程式
aidl.exe	應用程式
arm-linux-androideabi-ld.exe	應用程式
bcc_compat.exe	應用程式
dexdump.exe	應用程式
dx.bat	Windows 批次檔案
i686-linux-android-ld.exe	應用程式
libbcc.dll	應用程式擴充
libbcinfo.dll	應用程式擴充
libclang.dll	應用程式擴充
libLLVM.dll	應用程式擴充
llvm-rs-cc.exe	應用程式
mipsel-linux-android-ld.exe	應用程式
NOTICE.txt	TXT 檔案
package.xml	XML 檔案
SafetyBoolean.class	CLASS 檔案
SafetyBoolean.dex	DEX 檔案
SafetyBoolean.txt	TXT 檔案
source.properties	PROPERTIES 檔案
zipalign.exe	應用程式

Dex to file



```
dexdump -d -f SafetyBoolean.dex > SafetyBooleanDexDump.txt
```

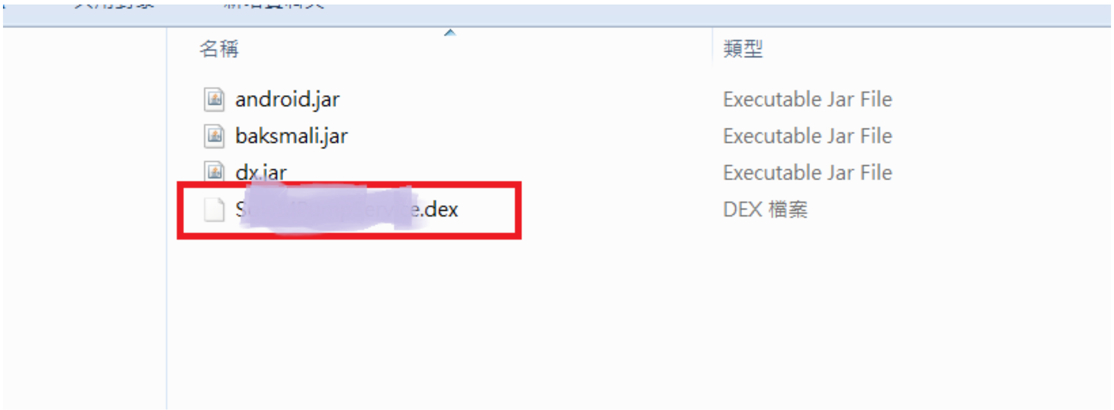
之後就會產生一個 DEX dump file



名稱	類型
lib	檔案資料夾
renderscript	檔案資料夾
aapt.exe	應用程式
aidl.exe	應用程式
arm-linux-androideabi-ld.exe	應用程式
bcc_compat.exe	應用程式
dexdump.exe	應用程式
dx.bat	Windows 批次檔案
i686-linux-android-ld.exe	應用程式
libbcc.dll	應用程式擴充
libbcinfo.dll	應用程式擴充
libclang.dll	應用程式擴充
libLLVM.dll	應用程式擴充
llvm-rs-cc.exe	應用程式
mipsel-linux-android-ld.exe	應用程式
NOTICE.txt	TXT 檔案
package.xml	XML 檔案
SafetyBoolean.class	CLASS 檔案
SafetyBoolean.dex	DEX 檔案
SafetyBoolean.txt	TXT 檔案
SafetyBooleanDexDump.txt	TXT 檔案
source.properties	PROPERTIES 檔案
zipalign.exe	應用程式

3. Java convert to smali file

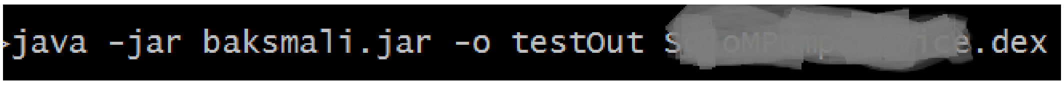
把 dex file 放到 AndroidDumpTool\lib



名稱	類型
android.jar	Executable Jar File
baksmali.jar	Executable Jar File
dx.jar	Executable Jar File
S[redacted]e.dex	DEX 檔案

打開 cmd 並且下以下指令

java -jar baksmali.jar -o 資料夾名稱 dex file



```
java -jar baksmali.jar -o testOut S[redacted]ce.dex
```

產生 smali file

Ps: baksmali 可以下載 https://github.com/fourbrother/java2smali

附件

AndroidDumpTool.rar

參考網站

http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html
http://huli.logdown.com/posts/661513-android-apk-decompile
https://codertw.com/android-%E9%96%8B%E7%99%BC/354008/
http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html
https://docs.oracle.com/javase/specs/jvms/se7/html/jvms-6.html#jvms-6.5
https://github.com/fourbrother/java2smali
https://r8.googlesource.com/?format=HTML
http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html

Adb 指令

1. 在沒有 root 手機的情況下讀取 Data 目錄下的檔

After android 8.0

使用 adb 命令時的錯誤

如果直接使用 adb 命令會產生以下錯誤:

```
127|shell@android:/ $ cd /data
cd /data
shell@android:/data $ ls
ls
opendir failed, Permission denied
```

你是沒有許可權的。

正確使用 adb 讀取 data 目錄下的檔方式

```
shell@android:/data $ run-as com.your.package
run-as com.your.package
shell@android:/data/data/com.your.package $ cd /data/data/com.your.package
cd /data/data/com.your.package
shell@android:/data/data/com.your.package $ ls
ls
cache
databases
lib
```

```
shared_prefs
shell@android:/data/data/com.your.package $ cd databases
cd databases
shell@android:/data/data/com.your.package/databases $ ls
yourpackagename.db
$ cat preferences.db > /mnt/sdcard/yourpackagename.db
```

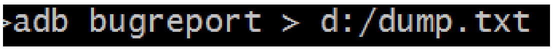
將你要訪問的 package 目錄下的 db 檔拷貝到 sdcard 中，這樣就可以正常訪問了！

作者：Snowball
來源：CSDN
原文：https://blog.csdn.net/yangzl2008/article/details/8498196
版權聲明：本文為博主原創文章，轉載請附上博文連結！

2. Android adb bugreport 工具分析和使用

Before android 8.0

adb bugreport > [file path] ps: Suggestion use the sdk version before android 8.0

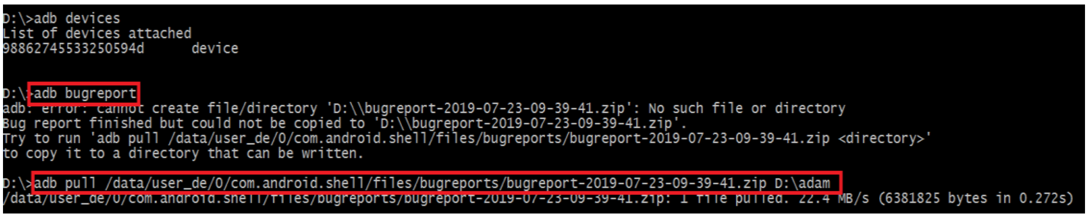


```
>adb bugreport > d:/dump.txt
```

After android 8.0

adb bugreport

adb pull [bug report zip file] [dest_file_path]



```
D:\>adb devices
List of devices attached
988627455332505594d      device


D:\>adb bugreport
adb: error: cannot create file/directory 'D:\\bugreport-2019-07-23-09-39-41.zip': No such file or directory
Bug report finished but could not be copied to 'D:\\bugreport-2019-07-23-09-39-41.zip'.
Try to run 'adb pull /data/user_de/0/com.android.shell/files/bugreports/bugreport-2019-07-23-09-39-41.zip <directory>'
to copy it to a directory that can be written.

D:\>adb pull /data/user_de/0/com.android.shell/files/bugreports/bugreport-2019-07-23-09-39-41.zip D:\adam
/data/user_de/0/com.android.shell/files/bugreports/bugreport-2019-07-23-09-39-41.zip: 1 file pulled. 22.4 MB/s (6381825 bytes in 0.272s)
```

異常 1

這裡我出現一個異常：當前 SDK 最新版本為 7.0(25)，而我的手機系統版本為 5.0，所以使用命令列：**adb bugreport > report.txt** 時出現了如下異常：

Failed to get bugreportz version, which is only available on devices running Android 7.0 or later.

這是由於 SDK 版本太新了，既然手機系統不能升級，那就只好把 SDK 降級，其實這裡只是使用了 plantform 下的 adb 命令，所以從新下載 **platform-tools** 即可。我下載和手機系統版本一致的 platform-tools，然後解壓。使用新下載的 platform-tools 的 adb 命令執行 adb bugreport > out.txt 時提示沒有設備。

異常 2

又使用 **adb devices** 查看列表仍然沒有。重啟 adb 也沒用效果。
因為我的是 Mac 設備，所以命令列執行 system_profiler SPUSBDataType 時會顯示一系列外接設備資訊，找到你的 Android 設備，複製 Vendor ID。
進入你下載的低版本 **platform-tools** 資料夾下，修改或創建 **adb_usb.ini** 檔，把剛才複製的 Vendor ID 粘貼到這裡。再次執行 **adb devices** 就可以查看到設備清單了。然後再次執行 adb bugreport > out.txt 就會產生一個設備耗能統計資訊檔。

注意

匯出電量統計資料是一個持續輸出的過程，就是說如果你不手動停止，那會一直統計。所以看到檔中已經寫入內容後就可以 CTRL+C 了。

Bugreport 分析

Reference: https://blog.csdn.net/createchance/article/details/51954142

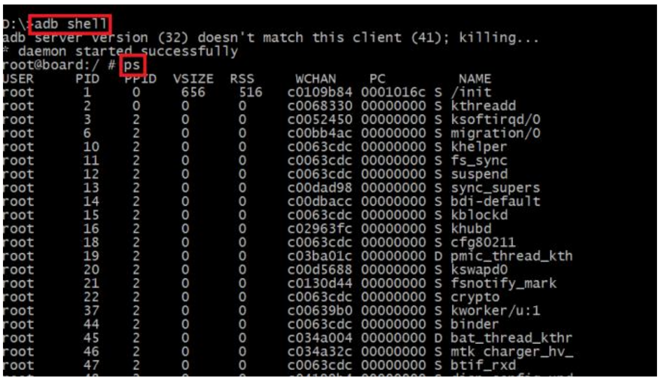
3. 利用 adb shell 來觀看 device 中的 process 和 thread

Before android 8.0

Open the command line and check adb connection status by adb devices

Process

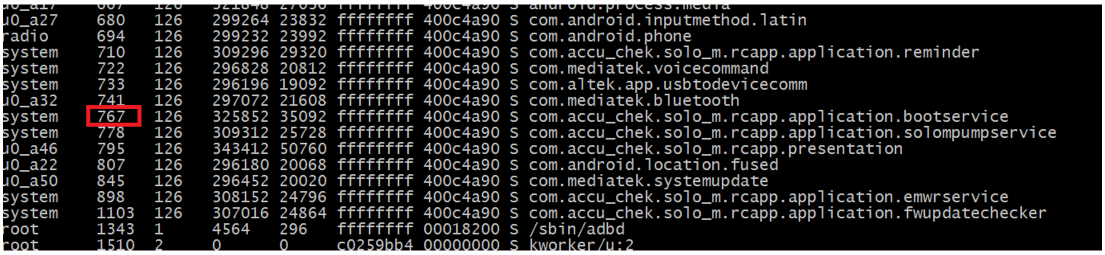
- adb shell
- ps



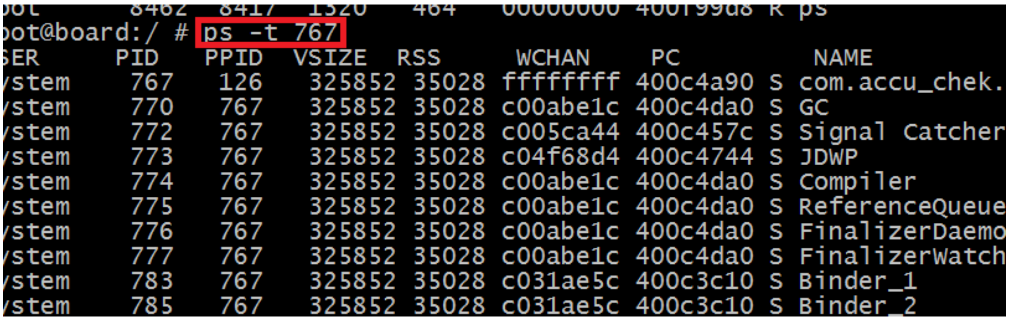
```
D:\>adb shell
adb server version (32) doesn't match this client (41); killing...
* daemon started successfully
root@board:/ # ps
USER     PID   PPID  VSIZE  RSS     WCHAN    PC         NAME
root      1     0     656    516   c0109b84 0001016c S /init
root      2     0     0      0     c0068330 00000000 S kthreadd
root      3     2     0      0     c0052450 00000000 S ksoftirqd/0
root      6     2     0      0     c00bb4ac 00000000 S migration/0
root      10    2     0      0     c0063cdc 00000000 S khelper
root      11    2     0      0     c0063cdc 00000000 S fs_sync
root      12    2     0      0     c0063cdc 00000000 S suspend
root      13    2     0      0     c00dad98 00000000 S sync_supers
root      14    2     0      0     c00dbacc 00000000 S bdi-default
root      15    2     0      0     c0063cdc 00000000 S kblockd
root      16    2     0      0     c02963fc 00000000 S khubd
root      18    2     0      0     c0063cdc 00000000 S cfg80211
root      19    2     0      0     c03ba01c 00000000 D pmic_thread_kth
root      20    2     0      0     c00d5688 00000000 S kswapd0
root      21    2     0      0     c0130d44 00000000 S fsnotify_mark
root      22    2     0      0     c0063cdc 00000000 S crypto
root      37    2     0      0     c00639b0 00000000 S kworker/u:1
root      44    2     0      0     c0063cdc 00000000 S binder
root      45    2     0      0     c034a004 00000000 D bat_thread_kthr
root      46    2     0      0     c034a32c 00000000 S mtk charger_hv_
root      47    2     0      0     c0063cdc 00000000 S btif_rxd
```

Thread

- Get the specified pid by ps command
- Ps －t [pid]



```
u0_a27    680   126   299264 23832 ffffffff 400c4a90 S com.android.inputmethod.latin
radio     694   126   299232 23992 ffffffff 400c4a90 S com.android.phone
system    710   126   309296 29320 ffffffff 400c4a90 S com.accu_chek.solo_m.rcapp.application.reminder
system    722   126   296828 20812 ffffffff 400c4a90 S com.mediatek.voicecommand
system    733   126   296196 19092 ffffffff 400c4a90 S com.altek.app.usbtodevicecomm
u0_a32    741   126   297072 21608 ffffffff 400c4a90 S com.mediatek.bluetooth
system    767   126   325852 35092 ffffffff 400c4a90 S com.accu_chek.solo_m.rcapp.application.bootservice
system    778   126   309312 25728 ffffffff 400c4a90 S com.accu_chek.solo_m.rcapp.application.solompumpservice
u0_a46    795   126   343412 50760 ffffffff 400c4a90 S com.accu_chek.solo_m.rcapp.presentation
u0_a22    807   126   296180 20068 ffffffff 400c4a90 S com.android.location.fused
u0_a50    845   126   296452 20020 ffffffff 400c4a90 S com.mediatek.systemupdate
system    898   126   308152 24796 ffffffff 400c4a90 S com.accu_chek.solo_m.rcapp.application.emwrservice
system    1103  126   307016 24864 ffffffff 400c4a90 S com.accu_chek.solo_m.rcapp.application.fwupdatechecker
root      1343  1     4564   296   ffffffff 00018200 S /sbin/adbd
root      1510  2     0      0     c0259bb4 00000000 S kworker/u:2
```



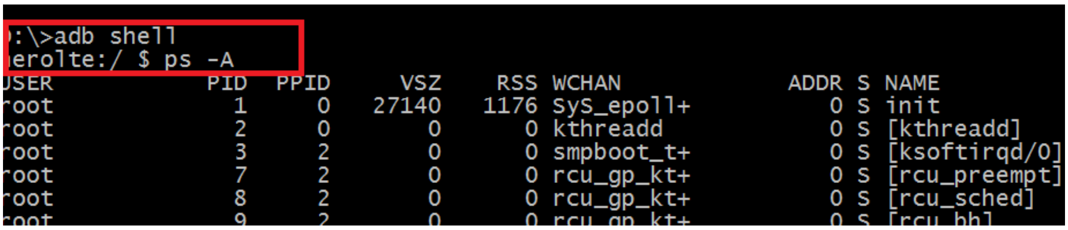
```
oot       8462  8417  1520   464   00000000 400f99d8 R ps
oot@board:/ # ps -t 767
SER      PID   PPID  VSIZE  RSS     WCHAN    PC         NAME
ystem     767   126   325852 35028 ffffffff 400c4a90 S com.accu_chek.
ystem     770   767   325852 35028 c00abe1c 400c4da0 S GC
ystem     772   767   325852 35028 c005ca44 400c457c S Signal Catcher
ystem     773   767   325852 35028 c04f68d4 400c4744 S JDWP
ystem     774   767   325852 35028 c00abe1c 400c4da0 S Compiler
ystem     775   767   325852 35028 c00abe1c 400c4da0 S ReferenceQueue
ystem     776   767   325852 35028 c00abe1c 400c4da0 S FinalizerDaemo
ystem     777   767   325852 35028 c00abe1c 400c4da0 S FinalizerWatch
ystem     783   767   325852 35028 c031ae5c 400c3c10 S Binder_1
ystem     785   767   325852 35028 c031ae5c 400c3c10 S Binder_2
```

After android 8.0

Open the command line and check adb connection status by adb devices

Process

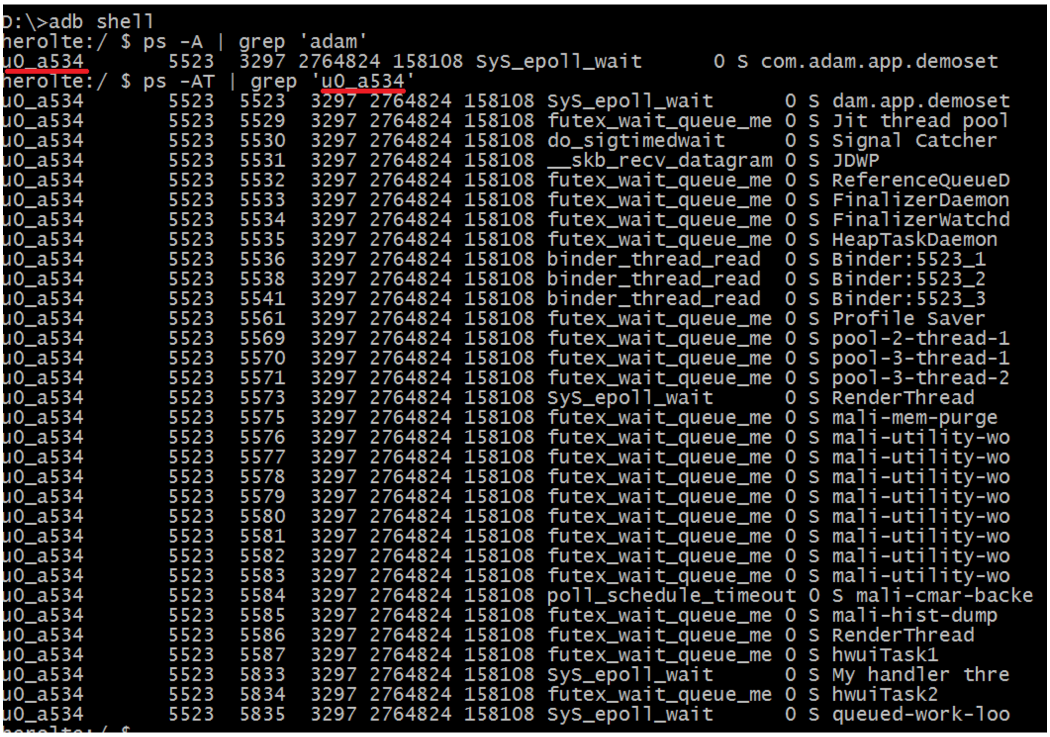
- adb shell
- ps -A



```
D:\>adb shell
erolte:/ $ ps -A
USER            PID   PPID     VSZ    RSS WCHAN            ADDR S NAME
root              1      0   27140   1176 SyS_epoll+          0 S init
root              2      0       0      0 kthreadd            0 S [kthreadd]
root              3      2       0      0 smpboot_t+          0 S [ksoftirqd/0]
root              7      2       0      0 rcu_gp_kt+          0 S [rcu_preempt]
root              8      2       0      0 rcu_gp_kt+          0 S [rcu_sched]
root              9      2       0      0 rcu_gp_kt+          0 S [rcu_bh]
```

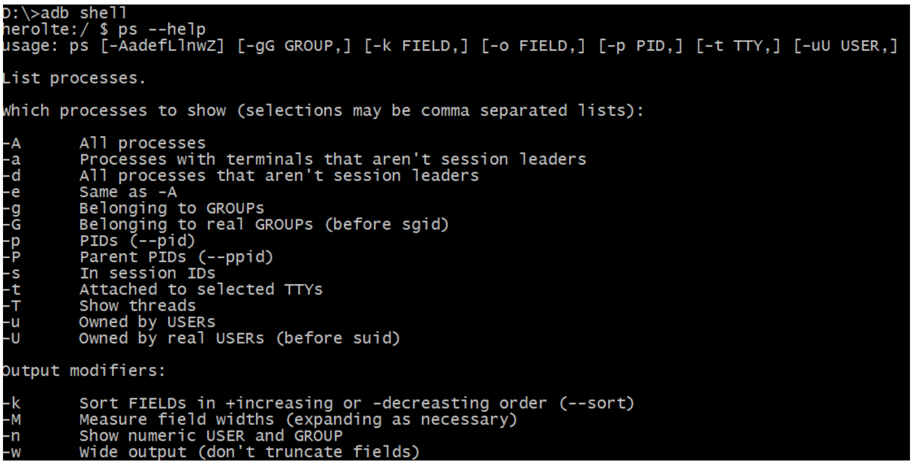
Thread

- adb shell
- ps -A
- get user name from process list
- ps -AT | grep [user name]



```
D:\>adb shell
herolte:/ $ ps -A | grep 'adam'
u0_a534         5523  3297 2764824 158108 SyS_epoll_wait      0 S com.adam.app.demoset
herolte:/ $ ps -AT | grep 'u0_a534'
u0_a534         5523  5523  3297 2764824 158108 SyS_epoll_wait      0 S dam.app.demoset
u0_a534         5523  5529  3297 2764824 158108 futex_wait_queue_me 0 S Jit thread pool
u0_a534         5523  5530  3297 2764824 158108 do_sigtimedwait     0 S Signal Catcher
u0_a534         5523  5531  3297 2764824 158108 __skb_recv_datagram 0 S JDWP
u0_a534         5523  5532  3297 2764824 158108 futex_wait_queue_me 0 S ReferenceQueueD
u0_a534         5523  5533  3297 2764824 158108 futex_wait_queue_me 0 S FinalizerDaemon
u0_a534         5523  5534  3297 2764824 158108 futex_wait_queue_me 0 S FinalizerWatchd
u0_a534         5523  5535  3297 2764824 158108 futex_wait_queue_me 0 S HeapTaskDaemon
u0_a534         5523  5536  3297 2764824 158108 binder_thread_read  0 S Binder:5523_1
u0_a534         5523  5538  3297 2764824 158108 binder_thread_read  0 S Binder:5523_2
u0_a534         5523  5541  3297 2764824 158108 binder_thread_read  0 S Binder:5523_3
u0_a534         5523  5561  3297 2764824 158108 futex_wait_queue_me 0 S Profile Saver
u0_a534         5523  5569  3297 2764824 158108 futex_wait_queue_me 0 S pool-2-thread-1
u0_a534         5523  5570  3297 2764824 158108 futex_wait_queue_me 0 S pool-3-thread-1
u0_a534         5523  5571  3297 2764824 158108 futex_wait_queue_me 0 S pool-3-thread-2
u0_a534         5523  5573  3297 2764824 158108 SyS_epoll_wait      0 S RenderThread
u0_a534         5523  5575  3297 2764824 158108 futex_wait_queue_me 0 S mali-mem-purge
u0_a534         5523  5576  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5577  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5578  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5579  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5580  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5581  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5582  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5583  3297 2764824 158108 futex_wait_queue_me 0 S mali-utility-wo
u0_a534         5523  5584  3297 2764824 158108 poll_schedule_timeout 0 S mali-cmar-backe
u0_a534         5523  5585  3297 2764824 158108 futex_wait_queue_me 0 S mali-hist-dump
u0_a534         5523  5586  3297 2764824 158108 futex_wait_queue_me 0 S RenderThread
u0_a534         5523  5587  3297 2764824 158108 futex_wait_queue_me 0 S hwuiTask1
u0_a534         5523  5833  3297 2764824 158108 SyS_epoll_wait      0 S My handler thre
u0_a534         5523  5834  3297 2764824 158108 futex_wait_queue_me 0 S hwuiTask2
u0_a534         5523  5835  3297 2764824 158108 SyS_epoll_wait      0 S queued-work-loo
herolte:/ $
```

Type the commend ps – help



```
D:\>adb shell
herolte:/ $ ps --help
usage: ps [-AadefLlnwZ] [-gG GROUP,] [-k FIELD,] [-o FIELD,] [-p PID,] [-t TTY,] [-uU USER,]

List processes.

Which processes to show (selections may be comma separated lists):

-A      All processes
-a      Processes with terminals that aren't session leaders
-d      All processes that aren't session leaders
-e      Same as -A
-g      Belonging to GROUPs
-G      Belonging to real GROUPs (before sgid)
-p      PIDs (--pid)
-P      Parent PIDs (--ppid)
-s      In session IDs
-t      Attached to selected TTYs
-T      Show threads
-u      Owned by USERs
-U      Owned by real USERs (before suid)

Output modifiers:

-k      Sort FIELDs in +increasing or -decreasting order (--sort)
-M      Measure field widths (expanding as necessary)
-n      Show numeric USER and GROUP
-w      Wide output (don't truncate fields)
```

Reference: https://blog.csdn.net/RadianceBlau/article/details/77855149