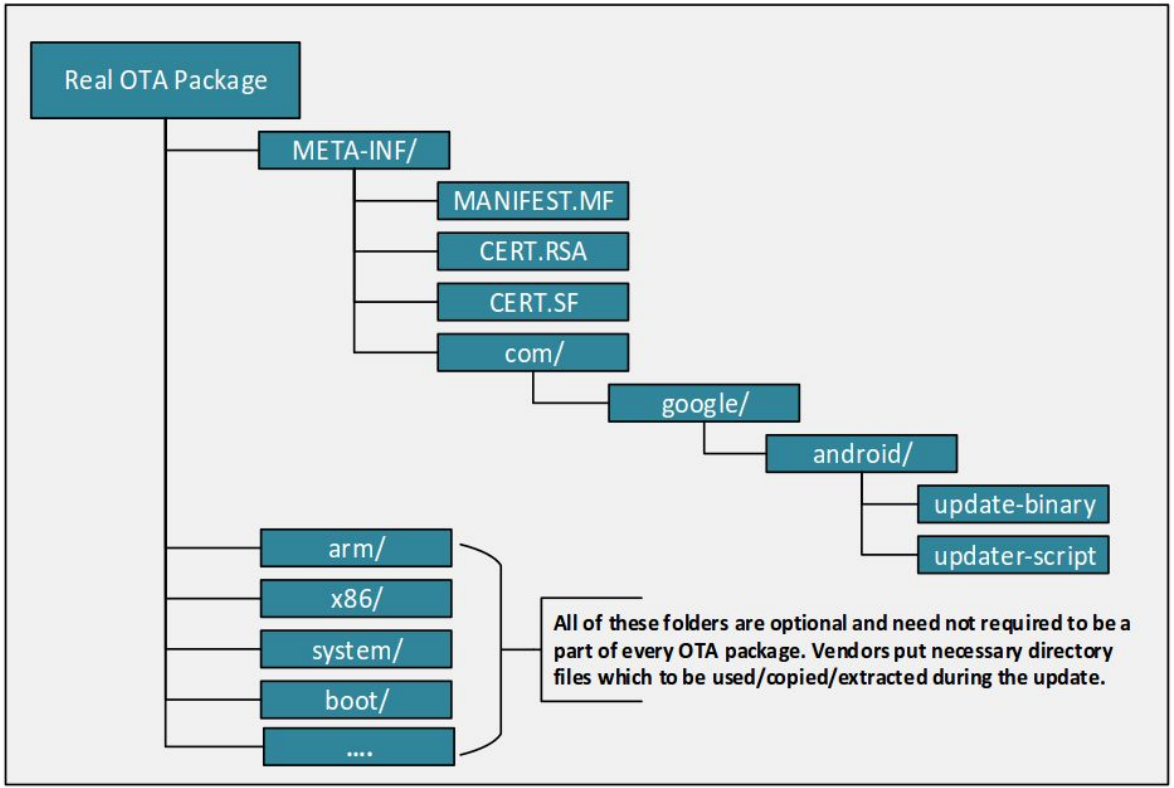
CSE 643
Android Rooting Help Session

Agenda

- OTA Structure
- Go through Task 1
- Go through NDK compilation for Task 2
- Demo Task 3

Goal of Rooting Lab

Write OTA package and root Android using Recovery OS



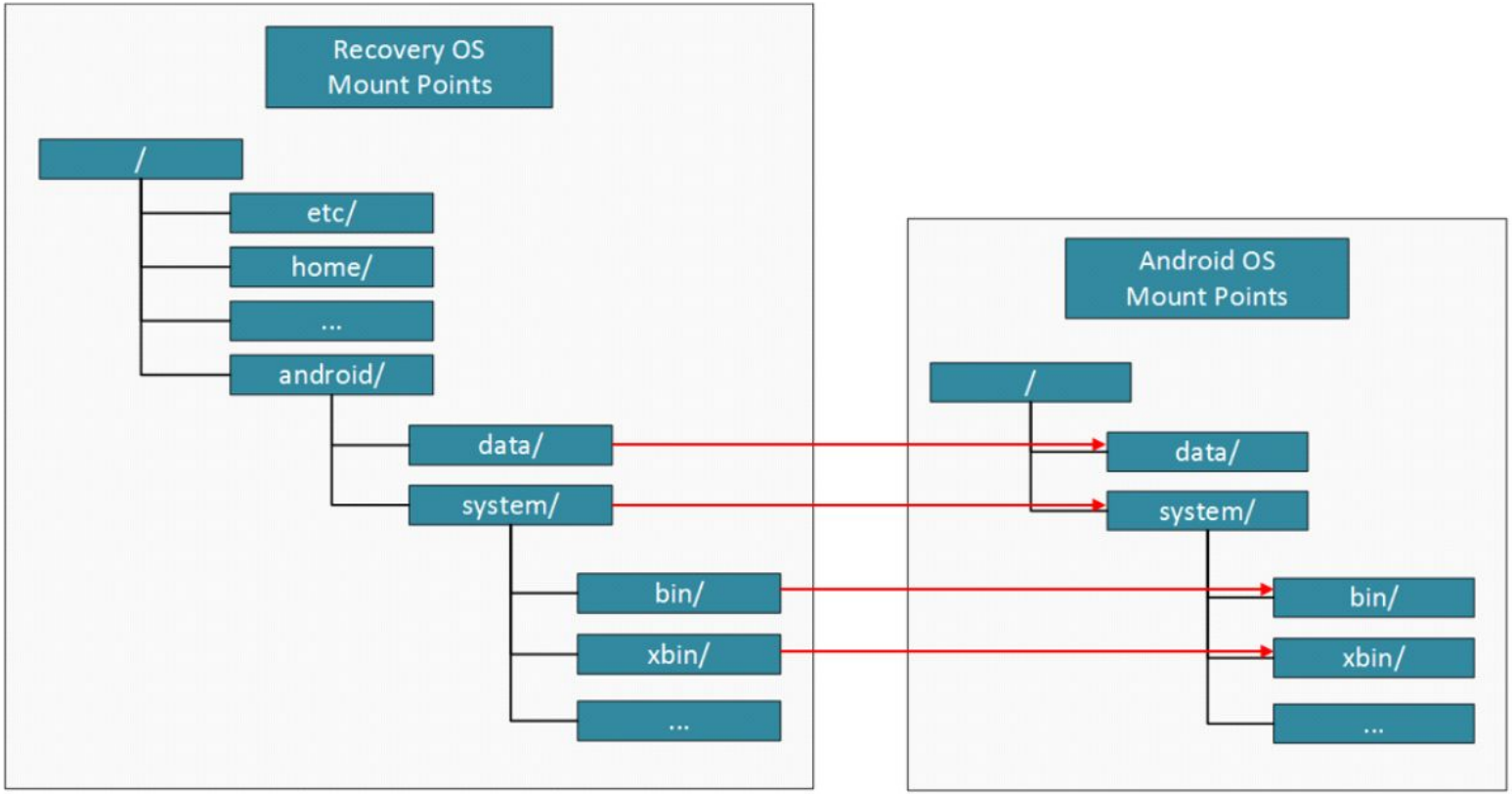
Remember !

- Please take a snapshot of the Android VM at beginning of the lab
- Use this snapshot at the beginning of each task
- Make sure Ubuntu 16.04 VM and Android VM belong to same NAT network

How to Boot Recovery OS

- Start Android VM
- Hold Left Shift when you see Virtual Box logo
- Keep Left Shift Pressed Until you see Grub menu
- Choose Ubuntu to enter Recovery OS
- Username / password:
    - seed, dees
    - root, root

Mount Points inside Recovery OS



Task 1: Simple OTA package

Goal of OTA is to create a dummy file in /system directory in Android

Step 1: Create OTA structure

$ mkdir -p task1/META-INF/com/google/android

Step 2: Create dummy.sh

$ cd task1/META-INF/com/google/android/

$ gedit dummy.sh

Add to file:

echo hello > /system/testfile

Step 3: Create update-binary

In same folder as step 2:

$ gedit update-binary

Add to file:

```
cp dummy.sh /android/system/xbin
chmod a+x /android/system/xbin/dummy.sh
sed -i "/return 0/i/system/xbin/dummy.sh" /android/system/etc/init.sh
```

$ chmod a+x update-binary

Step 4: Zip OTA

Switch to directory containing task1/

$ cd ../../../../../

$ zip -r task1.zip task1

The OTA is prepared. Boot the Recovery OS and find its IP address.

Step 5: Copy OTA to Recovery OS

(Assuming 10.0.2.97 is IP address of Recovery OS)

$ scp task1.zip seed@10.0.2.97:/tmp

Switch to Recovery OS

$ cd /tmp

$ unzip task1.zip

Step 6: Run OTA and verify result

(In Recovery OS)

$ cd /tmp/task1/META-INF/com/google/android

$ sudo ./update-binary

$ sudo reboot

Let VM boot into Android OS. Check whether file is created in /system

Task 2: Inject code via app_process

Here we will try the compilation step make sure students don’t have any issue to prepare the binary

In Ubuntu 16.04 VM, create directory task2_code.

In task2_code:

- Create my_app_process.c (code in lab description)
- Application.mk (code in lab description)

Task 2 (Contd.)

Create Android.mk:

```
LOCAL_PATH := $(call my-dir)
include $(CLEAR_VARS)
LOCAL_MODULE := my_app_process
LOCAL_SRC_FILES := my_app_process.c
include $(BUILD_EXECUTABLE)
```

Compile using NDK (follow 2 steps in lab description)

The resulting binary will be in: task2_code/libs/x86

Task 2 OTA

- You will create a task 2 OTA directory structure similar to Task 1:

  $ mkdir -p task2/META-INF/com/google/android

- Copy my_app_process binary from libs/x86/ (previous slide) into the task 2 OTA android sub-folder
- **Your task is to write the update-binary script inside the task 2 OTA android sub-folder**
- Once done, zipping OTA, sending to Recovery, and running OTA in Recovery is similar to Task 1.

Task 3

Download and compile SimpleSU (http://www.cis.syr.edu/~wedu/seed/Labs_16.04/Mobile/Android_Rooting/SimpleSU.zip)

$ unzip SimpleSU.zip

$ cd SimpleSU/

$ bash compile_all.sh

The generated binaries are in:

- SimpleSU/mydaemon/libs/x86/mydaemon
- SimpleSU/mysu/libs/x86/mysu

Task 3 OTA

- You will create a task 3 OTA directory structure similar to Task 1:

  $ mkdir -p task3/META-INF/com/google/android

- Create a x86 folder under task 3:

  $ mkdir -p task3/x86/

- Copy mydaemon and mysu binaries (from previous slide) into task 3 OTA x86 sub-folder
- **Your task is to write the update-binary script inside the task 3 OTA android sub-folder.** Once done, zipping OTA, sending to Recovery, and running OTA in Recovery is similar to Task 1

Task 3 Result

- Once update-binary from task 3 OTA has been executed in Recovery, reboot into Android (follows step from task 1)
- Open Terminal App
- Type: $ mysu
- If you see a root shell, then you have succeeded

END