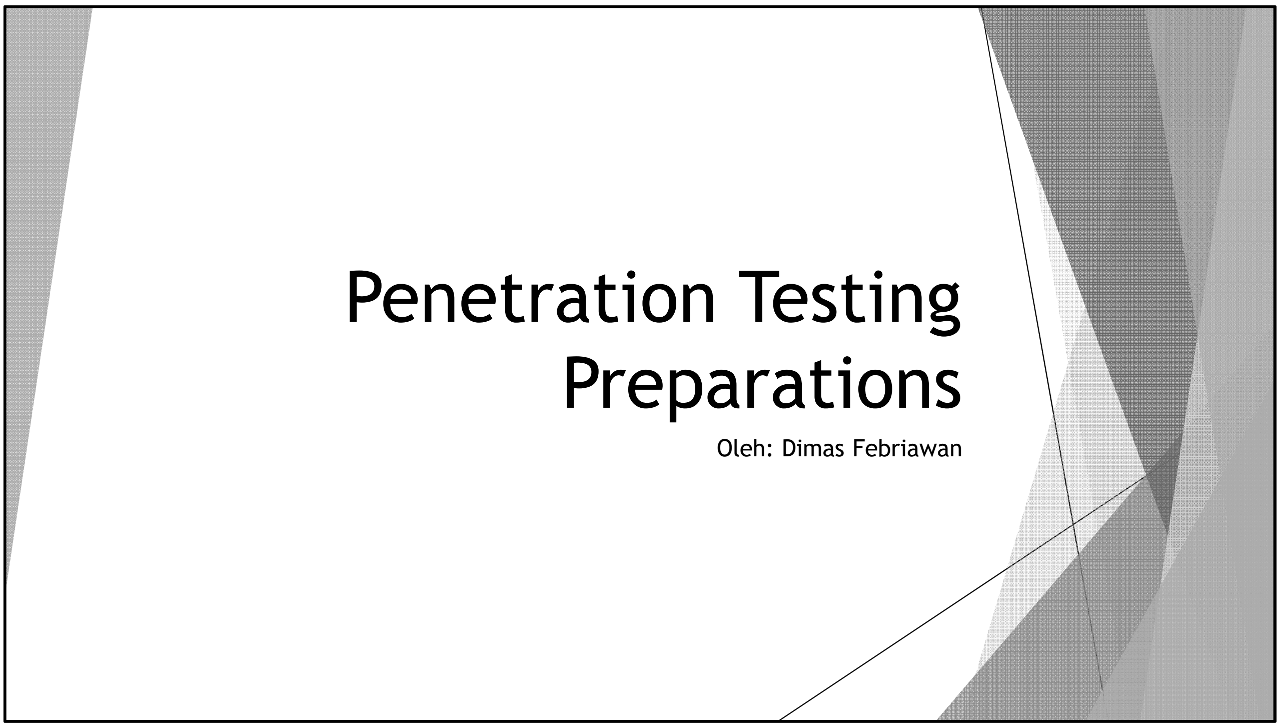
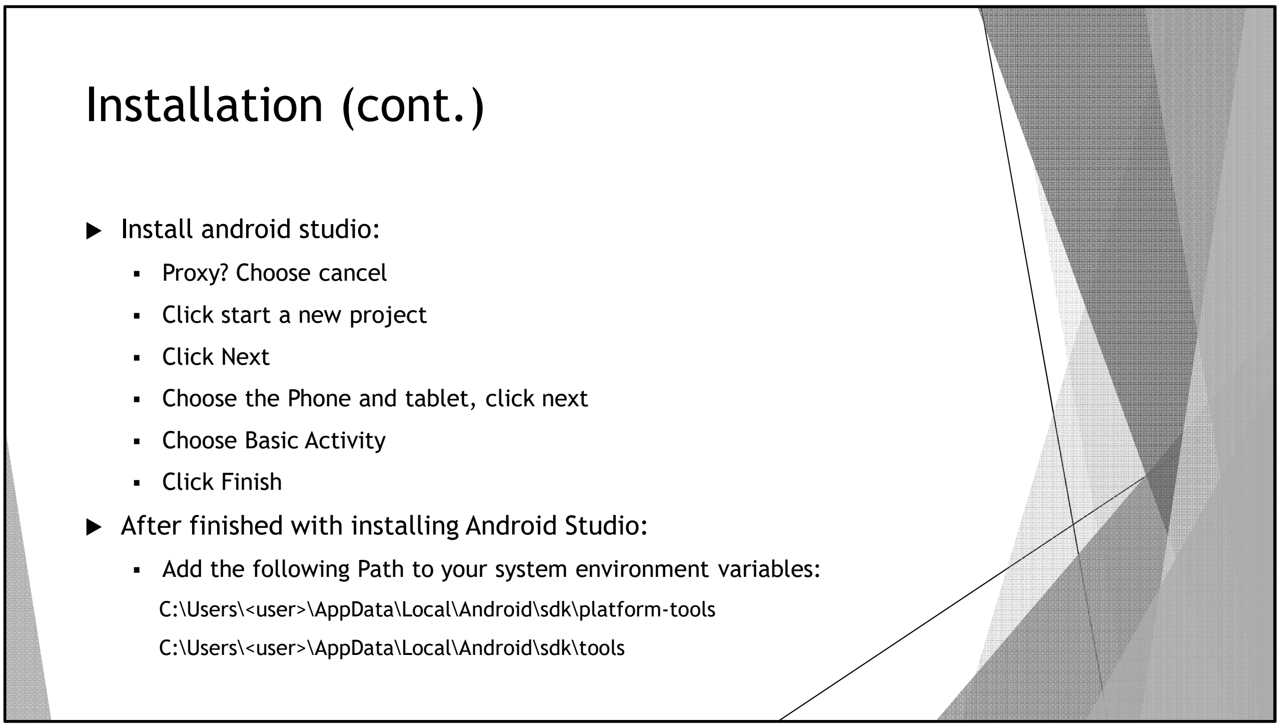


Penetration Testing Preparations

Oleh: Dimas Febriawan

Installation

- Download Java JDK from:
  http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html
- Download dex2jar from:
  https://sourceforge.net/projects/dex2jar/
- Download android Studio:
  https://developer.android.com/studio/index.html



Installation (cont.)

- Install android studio:
  - Proxy? Choose cancel
  - Click start a new project
  - Click Next
  - Choose the Phone and tablet, click next
  - Choose Basic Activity
  - Click Finish
- After finished with installing Android Studio:
  - Add the following Path to your system environment variables:

```
C:\Users\<user>\AppData\Local\Android\sdk\platform-tools
C:\Users\<user>\AppData\Local\Android\sdk\tools
```

Installation (cont.)

- Create new Virtual Device:
  - Click Tools -> Android -> AVD Manager
  - Click Create New Virtual Device
  - Click Next

  For PCs that don’t support Intel’s hardware virtualization (VT-x):
  - Click Other Images Tab
  - Download the armv64 (for 64-bit computers) or armeabi images
  - After finished with selecting the system images, click next, finish
  - Click the green button to Run the emulator

Installation (cont.)

- 2nd option, connect your physical phone to Android Studio:
  - First, you need to root your phone!
  - Follow: https://developer.android.com/studio/run/device.html
  - And: https://developer.android.com/studio/run/oem-usb.html
  - Open Windows CLI: adb devices

  If device is unauthorized:
  - From your device, go to the developer options on the phone and click "Revoke USB debugging authorization“
  - Then restart adb server from CLI:

```
adb kill-server
adb start-server
```

Installation (cont.)

- Reconnect the device

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

- Re-check the device from CLI: adb devices

- 3rd option, use linux based OS computer

  Check to see if CPU supports hardware virtualization for Ubuntu:

  https://help.ubuntu.com/community/KVM/Installation

  If the CPU supports it, then see the next slide

Installation for Ubuntu

- Install Java JDK:
  - https://www.digitalocean.com/community/tutorials/how-to-install-java-on-ubuntu-with-apt-get
  - https://docs.oracle.com/javase/8/docs/technotes/guides/install/linux_jdk.html
- Install Android Studio:
  - https://itsfoss.com/install-android-studio-ubuntu-linux/
  - https://developer.android.com/studio/install.html -> Choose Instructions for: Linux
  - For 64-bit architecture computer, run:

```
sudo apt-get install libc6:i386 libncurses5:i386 libstdc++6:i386 lib32z1 libbz2-1.0:i386
```

Installation for Ubuntu (cont.)

- After installing Android Studio, make sure that Android SDK has been installed as well.
  - Check it from Configure -> SDK Manager
- After SDK has been properly installed:
  - Click start a new project
  - Click Next
  - Choose the Phone and tablet, click next
  - Choose Basic Activity
  - Click Finish
  - Click AVD Manager icon and then Create New Virtual Devices

Android .apk
Reverse Engineering

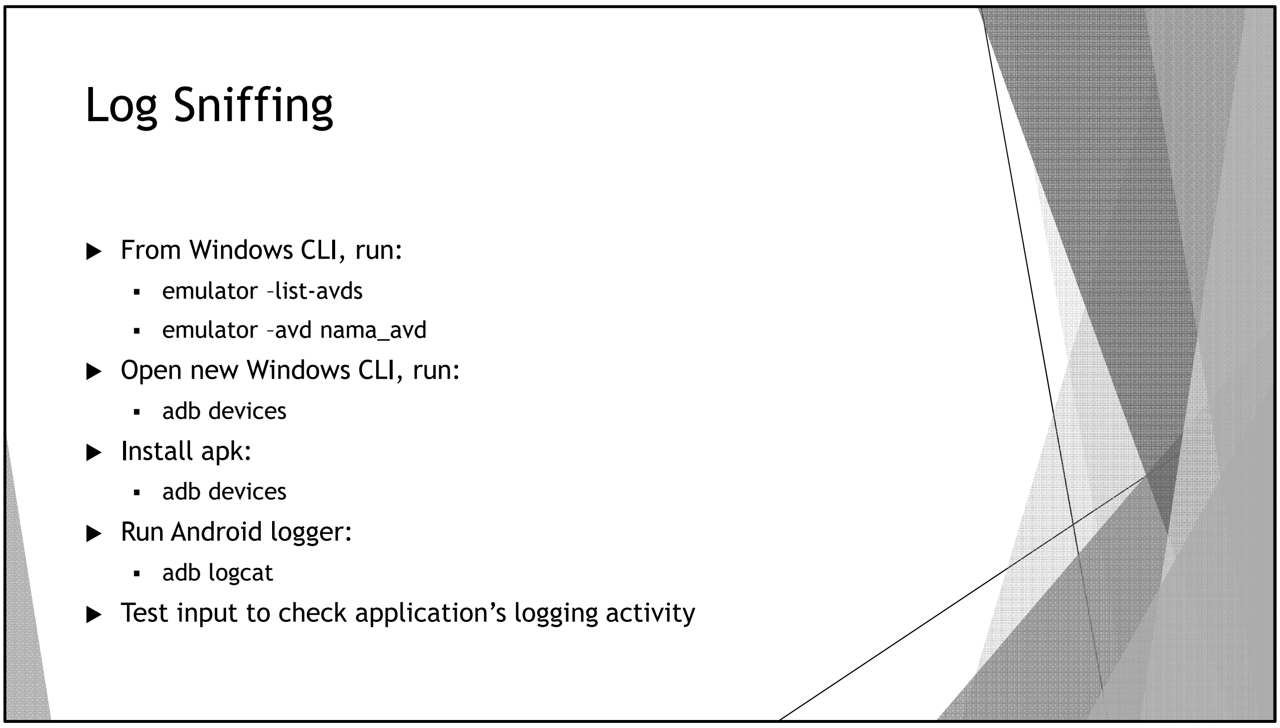
Reverse Engineering

- Download Diva apk from: http://www.payatu.com/wp-content/uploads/2016/01/diva-beta.tar.gz
- Run from Linux: tar zxvf diva-beta.tar.gz
- Run from Windows CLI: d2j-dex2jar.bat diva-beta.apk
- Run from Windows CLI : d2j-dex2smali.bat diva-beta-dex2jar.jar
- Put all the files (.apk, .jar and the diva-out folder) under 1 folder
- Run Android Studio and then open the diva-beta.apk

Reverse Engineering

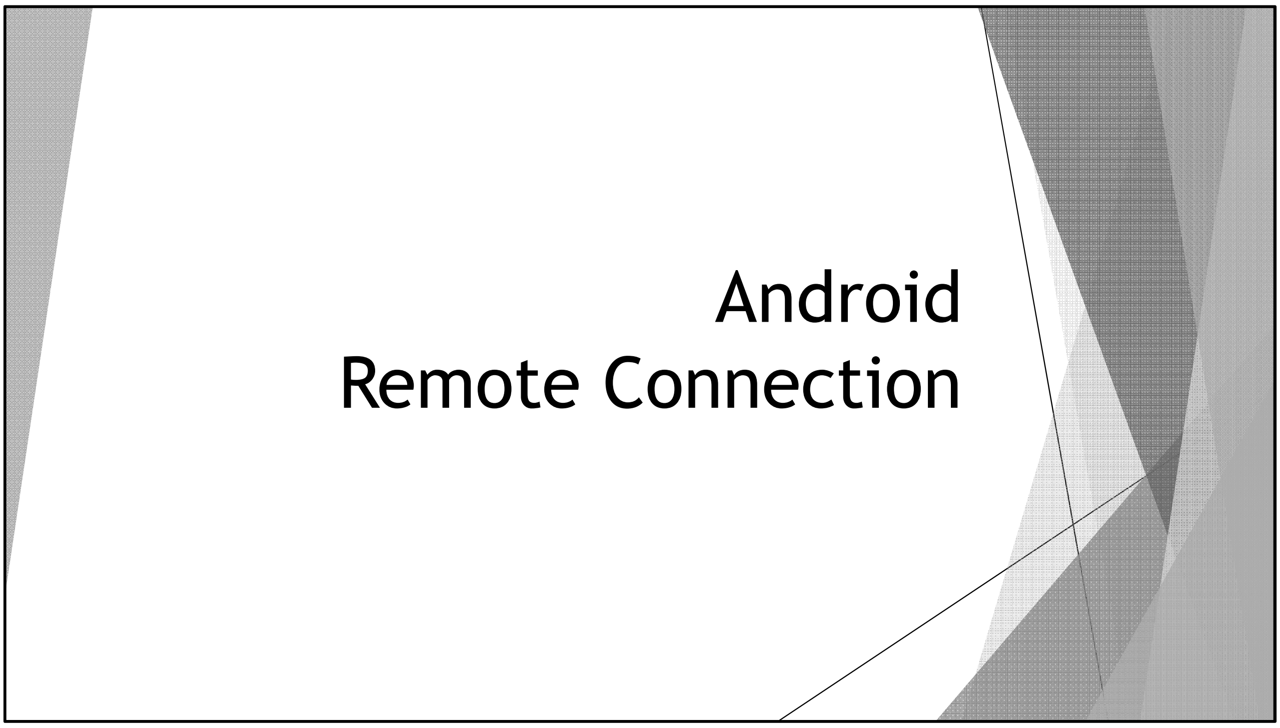
- Other option, download and extract JD GUI:
  http://jd.benow.ca/
- Open the JD GUI application
- Open the .jar file from JD GUI

Android Log Sniffing



Log Sniffing

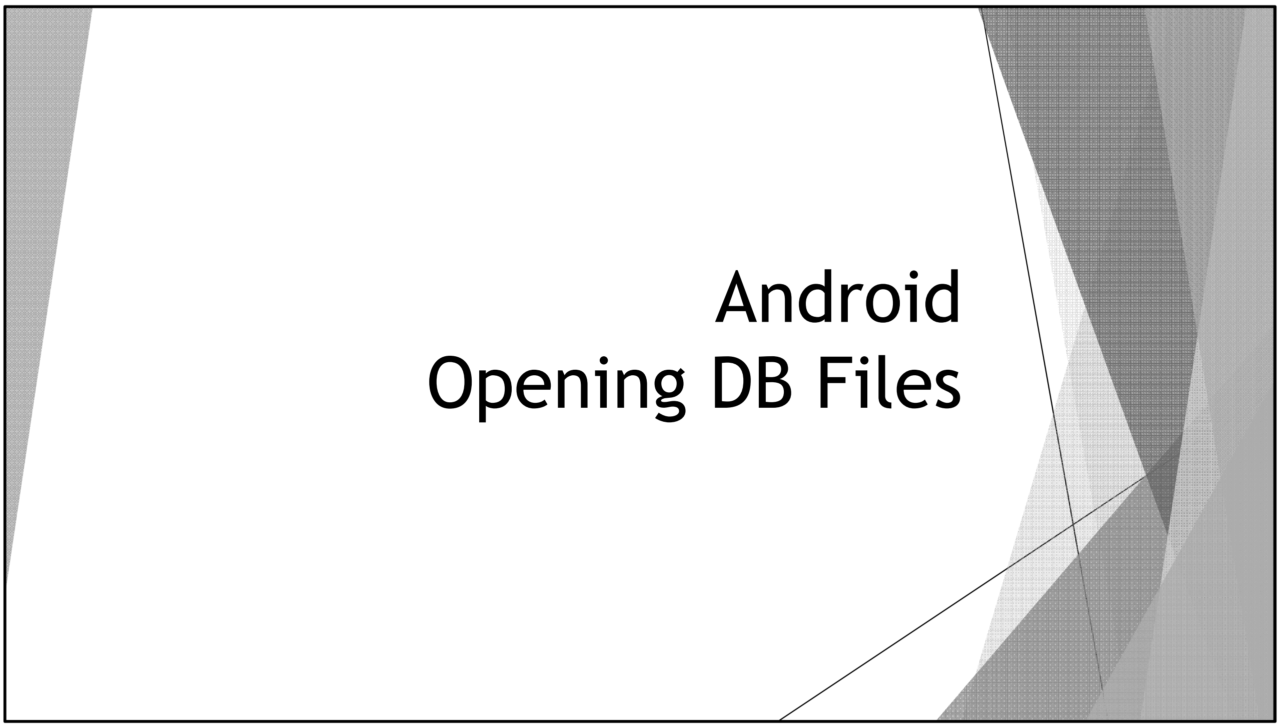
- From Windows CLI, run:
  - emulator –list-avds
  - emulator –avd nama_avd
- Open new Windows CLI, run:
  - adb devices
- Install apk:
  - adb devices
- Run Android logger:
  - adb logcat
- Test input to check application’s logging activity



Android
Remote Connection

Remote Connection

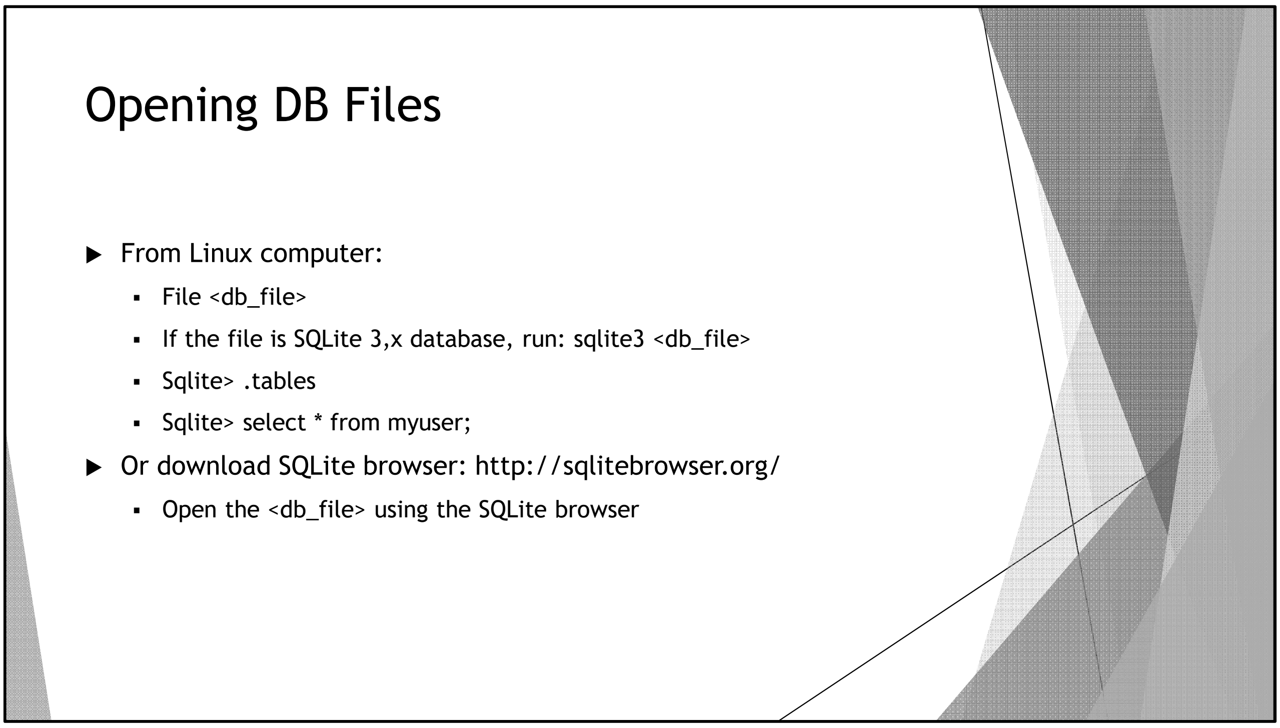
- Test application’s input
- From Windows CLI, run:
  - Adb shell
  - Ls –al (find the most recent modified folder then browse to that folder)



Android
Opening DB Files

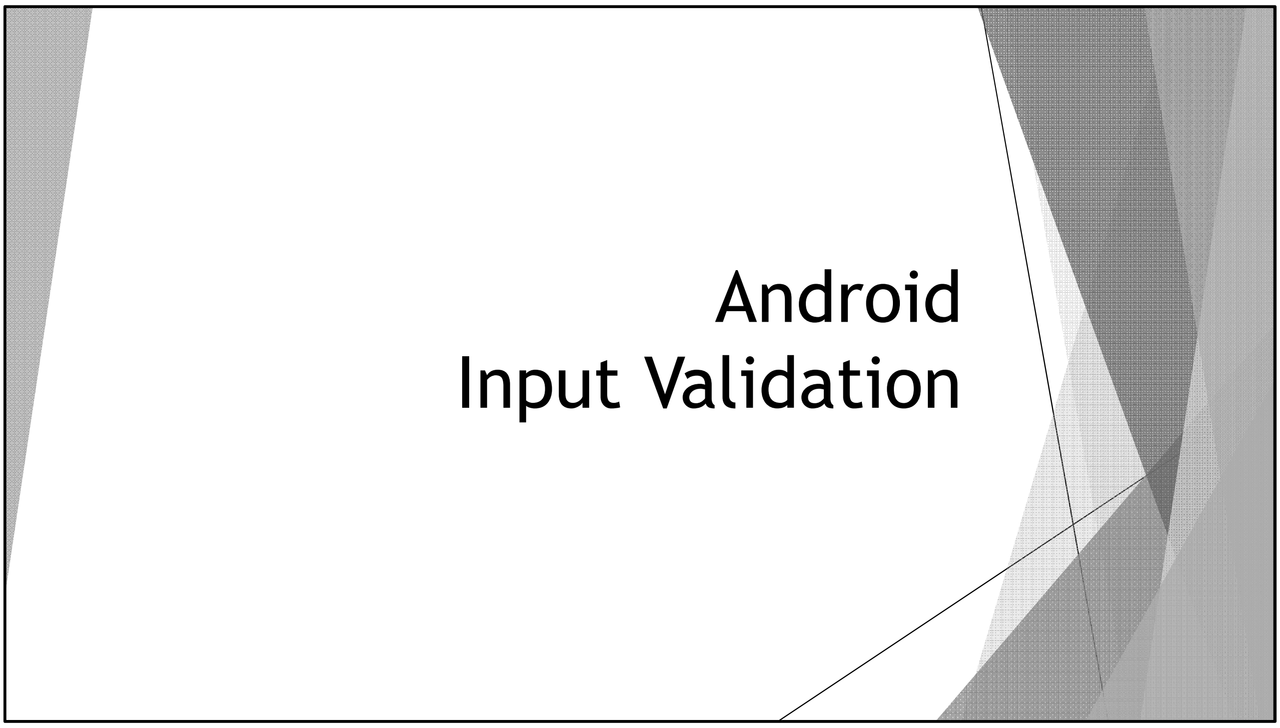
Opening DB Files

- From Windows CLI, run:
  - Adb shell
  - Ls –al (find the most recent modified folder then browse to that folder)
  - Browse to the application’s db folder, then change the permission:
    chmod 666 <db_file>
  - Exit from the shell, then pull the file to the local computer:
    adb pull <path>/<db_file>

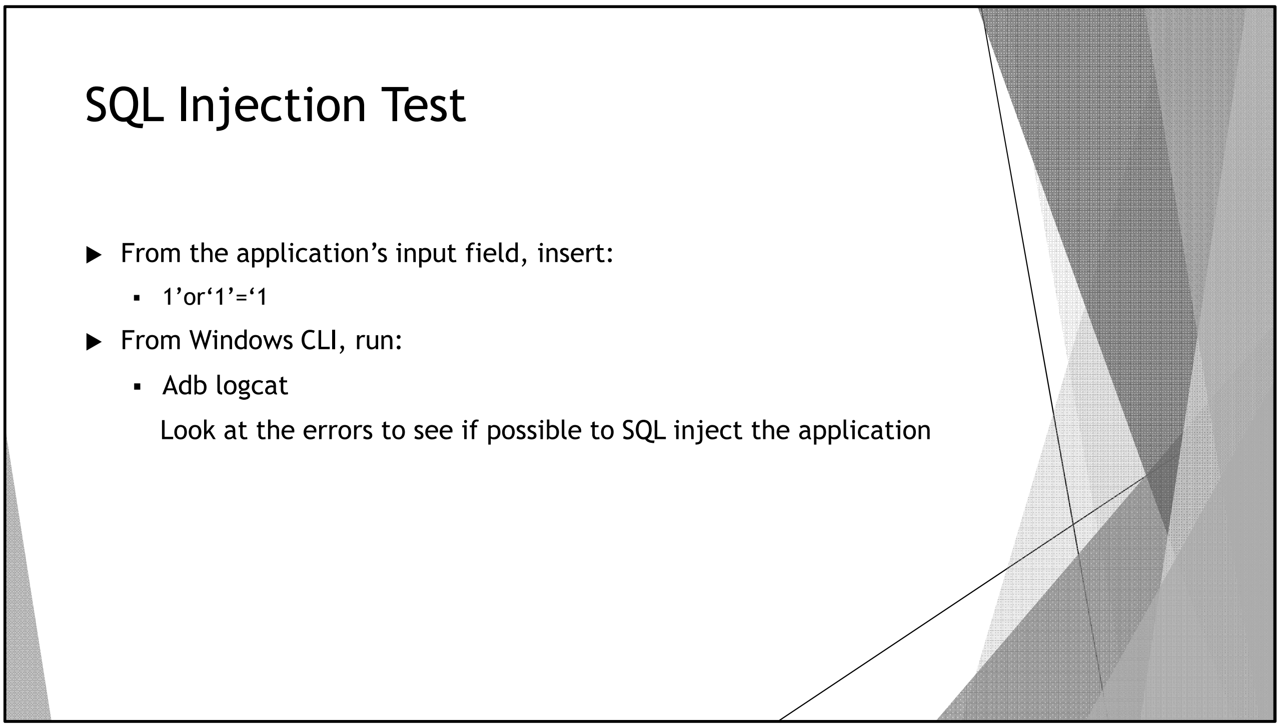


Opening DB Files

- From Linux computer:
  - File <db_file>
  - If the file is SQLite 3,x database, run: sqlite3 <db_file>
  - Sqlite> .tables
  - Sqlite> select * from myuser;
- Or download SQLite browser: http://sqlitebrowser.org/
  - Open the <db_file> using the SQLite browser



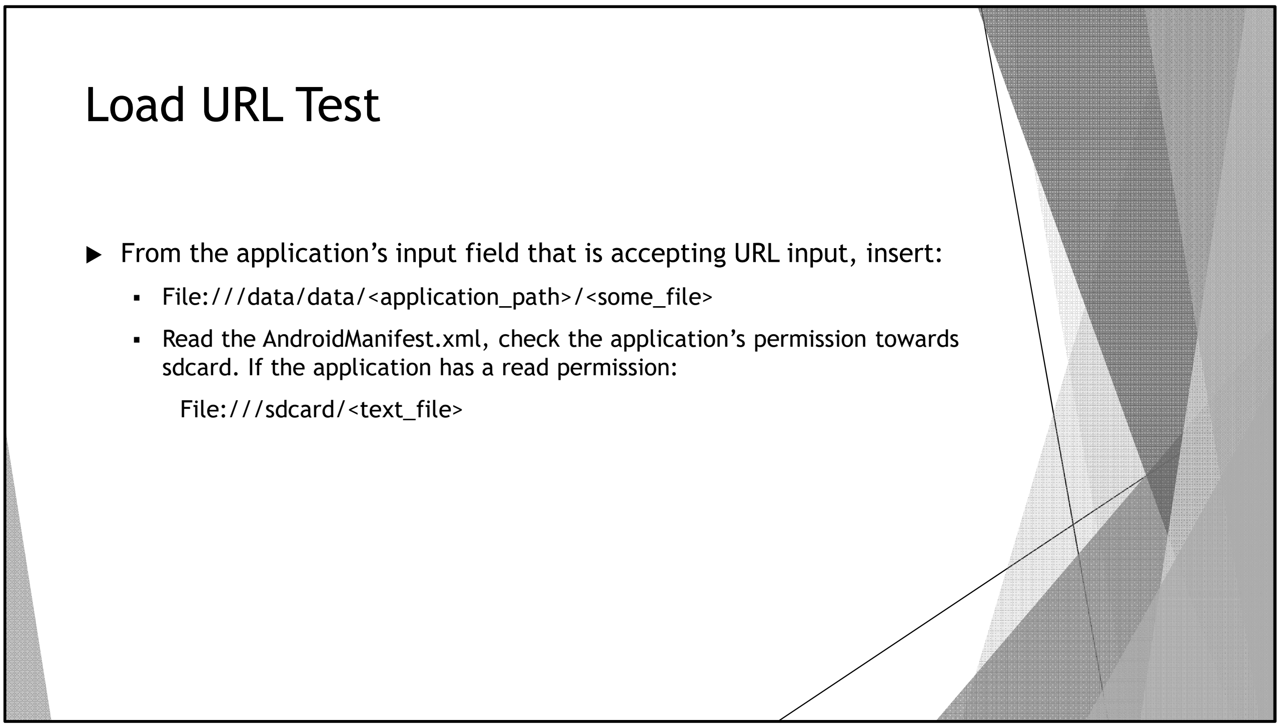
Android
Input Validation



SQL Injection Test

- From the application’s input field, insert:
  - 1’or‘1’=‘1
- From Windows CLI, run:
  - Adb logcat

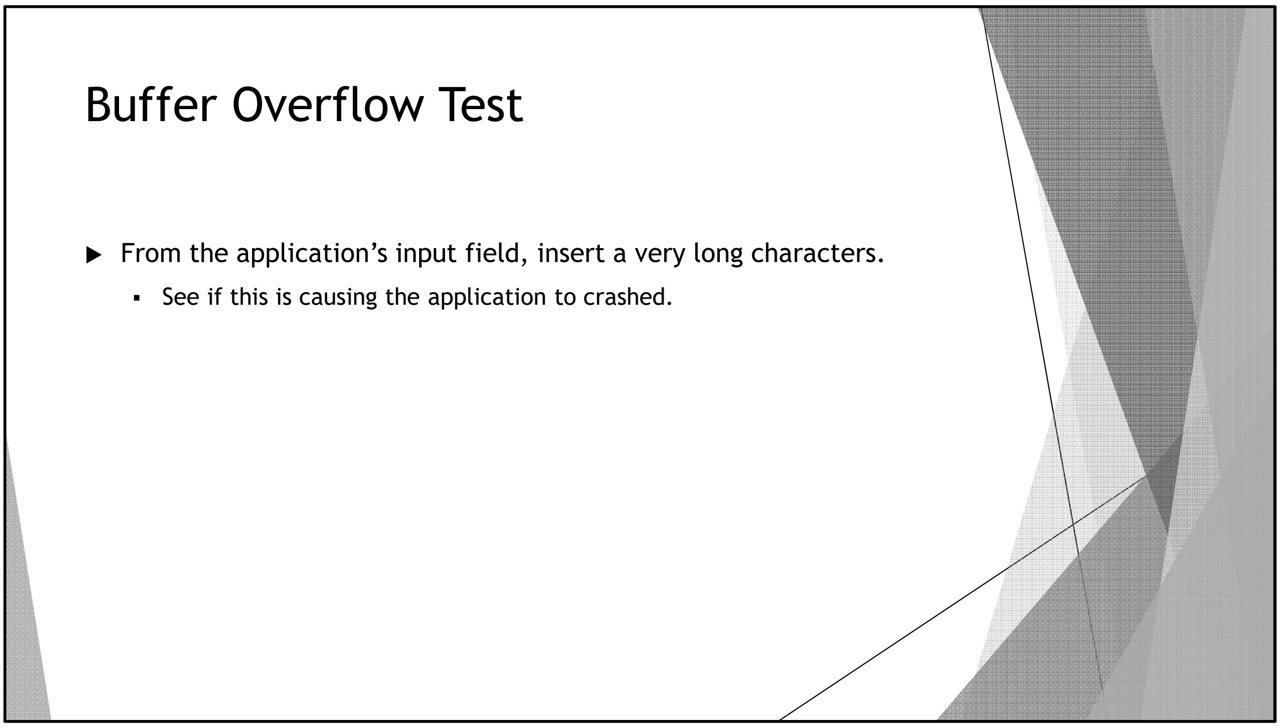
    Look at the errors to see if possible to SQL inject the application



Load URL Test

- From the application’s input field that is accepting URL input, insert:
  - File:///data/data/<application_path>/<some_file>
  - Read the AndroidManifest.xml, check the application’s permission towards sdcard. If the application has a read permission:

    File:///sdcard/<text_file>



Buffer Overflow Test

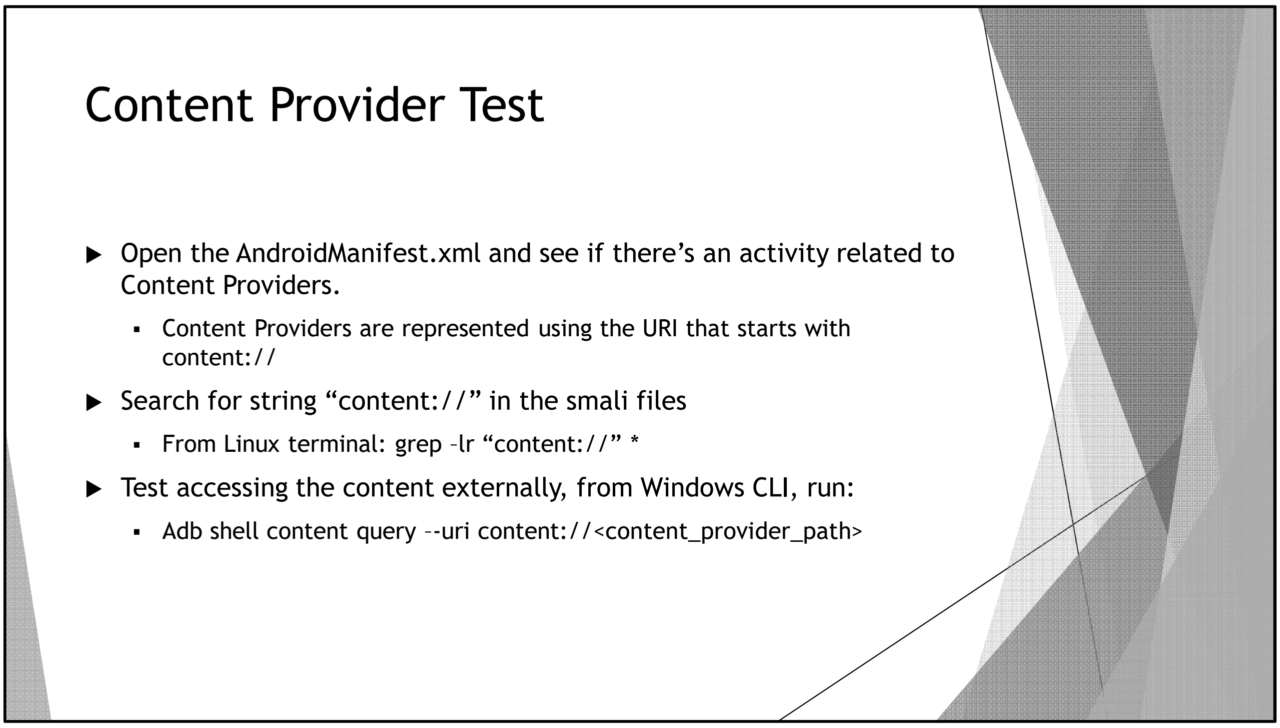
- From the application’s input field, insert a very long characters.
  - See if this is causing the application to crashed.

Android
Access Control Test

Access Control Test

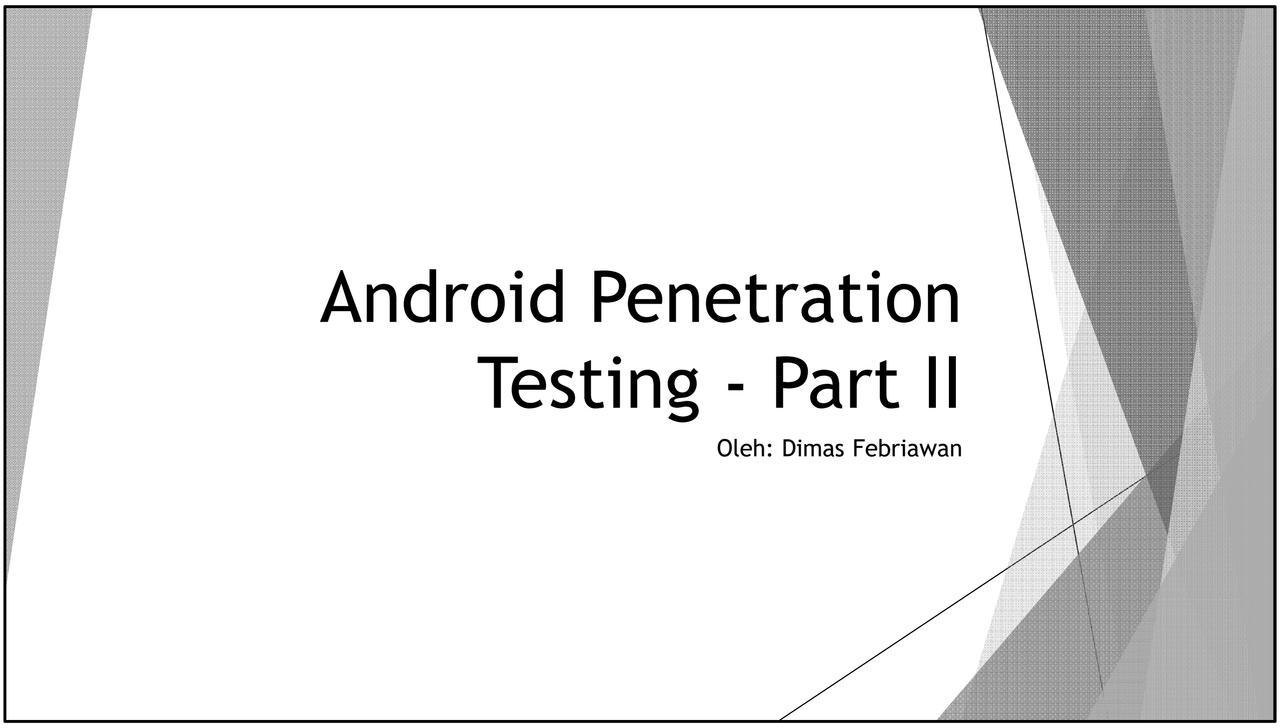
- Open the AndroidManifest.xml and see if there's an activity related to Intent Filters.
  - Using an intent filter is not a secure way to control access rights.
- From Windows CLI, run:
  - Adb shell am start –n jakhar.aseem.diva/.APICredsActivity –a jakhar.aseem.diva.action.VIEW_CREDS

Adb shell	: to get shell access to emulator / device
AM	: activity manager tool
Start	: launch an activity
-n	: activity name
- A	: activity action



Content Provider Test

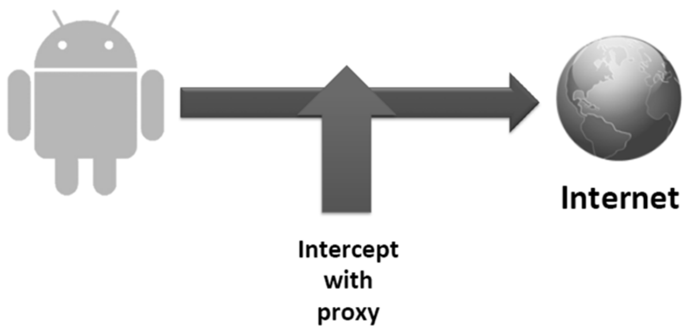
- Open the AndroidManifest.xml and see if there’s an activity related to Content Providers.
  - Content Providers are represented using the URI that starts with content://
- Search for string “content://” in the smali files
  - From Linux terminal: grep –lr “content://” *
- Test accessing the content externally, from Windows CLI, run:
  - Adb shell content query --uri content://<content_provider_path>

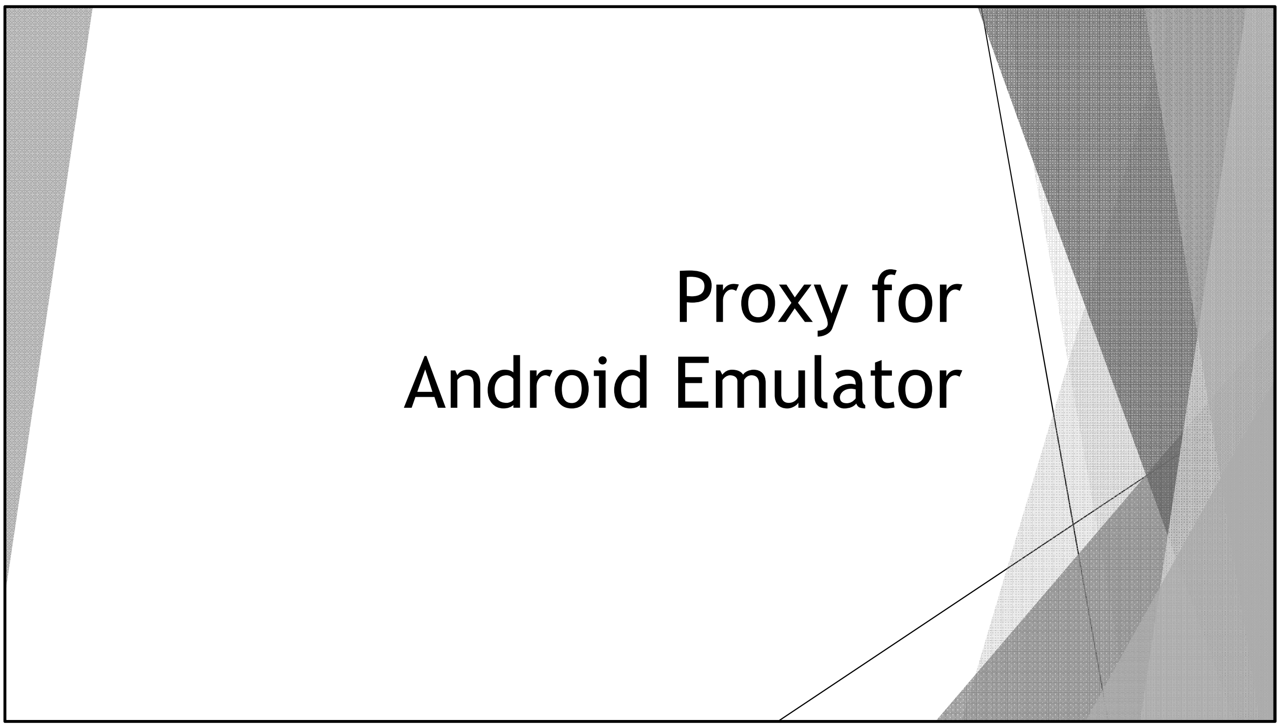


Android Penetration Testing - Part II

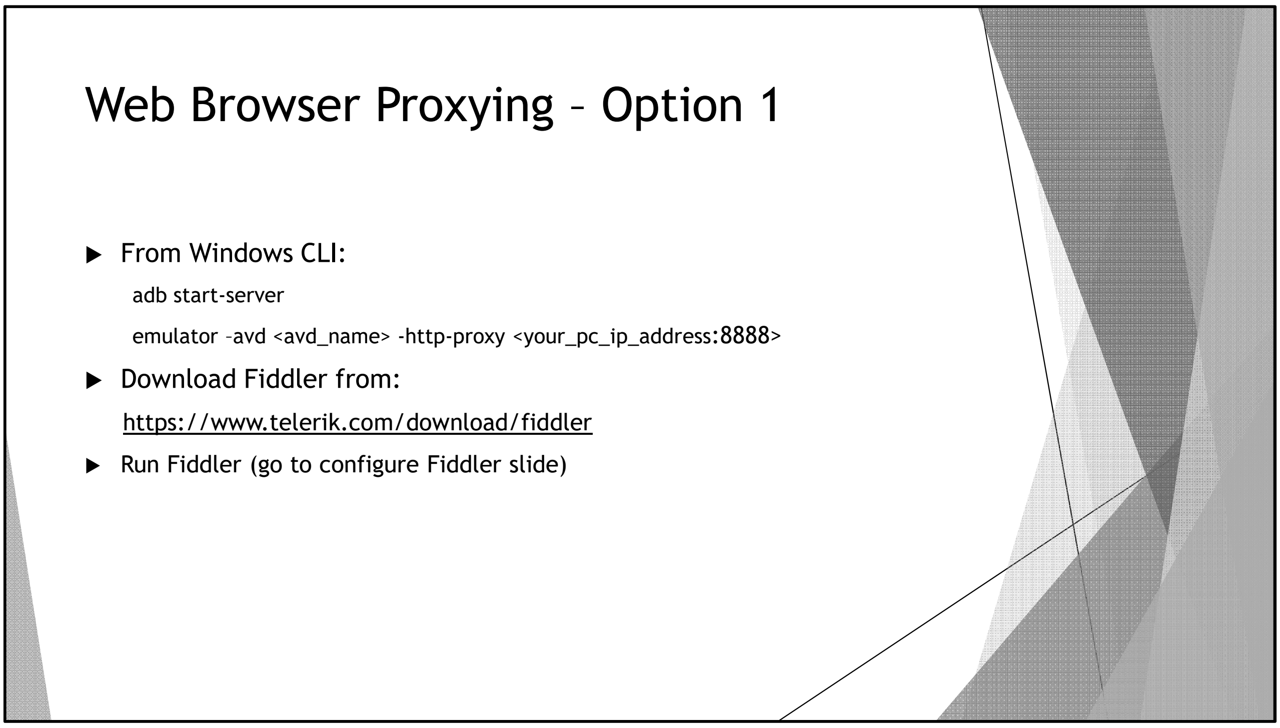
Oleh: Dimas Febriawan

Android Proxying





Proxy for Android Emulator



Web Browser Proxying – Option 1

- From Windows CLI:

```
adb start-server
emulator –avd <avd_name> -http-proxy <your_pc_ip_address:8888>
```

- Download Fiddler from:

  https://www.telerik.com/download/fiddler

- Run Fiddler (go to configure Fiddler slide)

Web Browser Proxying – Option 2

- Change the emulator's APN settings from Settings -> Wireless & Networks -> Mobile Networks -> Access Point Names:

  Name: Internet

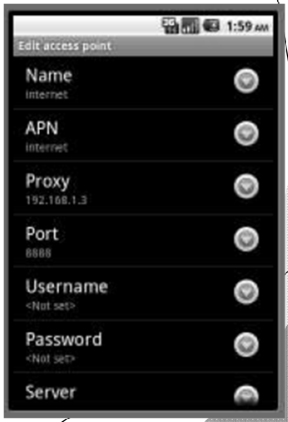
  APN: Internet

  Proxy: IP Address of your computer

  Port: 8888

  Username: <not set>

  Password: <not set>


Edit access point

Name
internet

APN
internet

Proxy
192.168.1.3

Port
8888

Username
<Not set>

Password
<Not set>

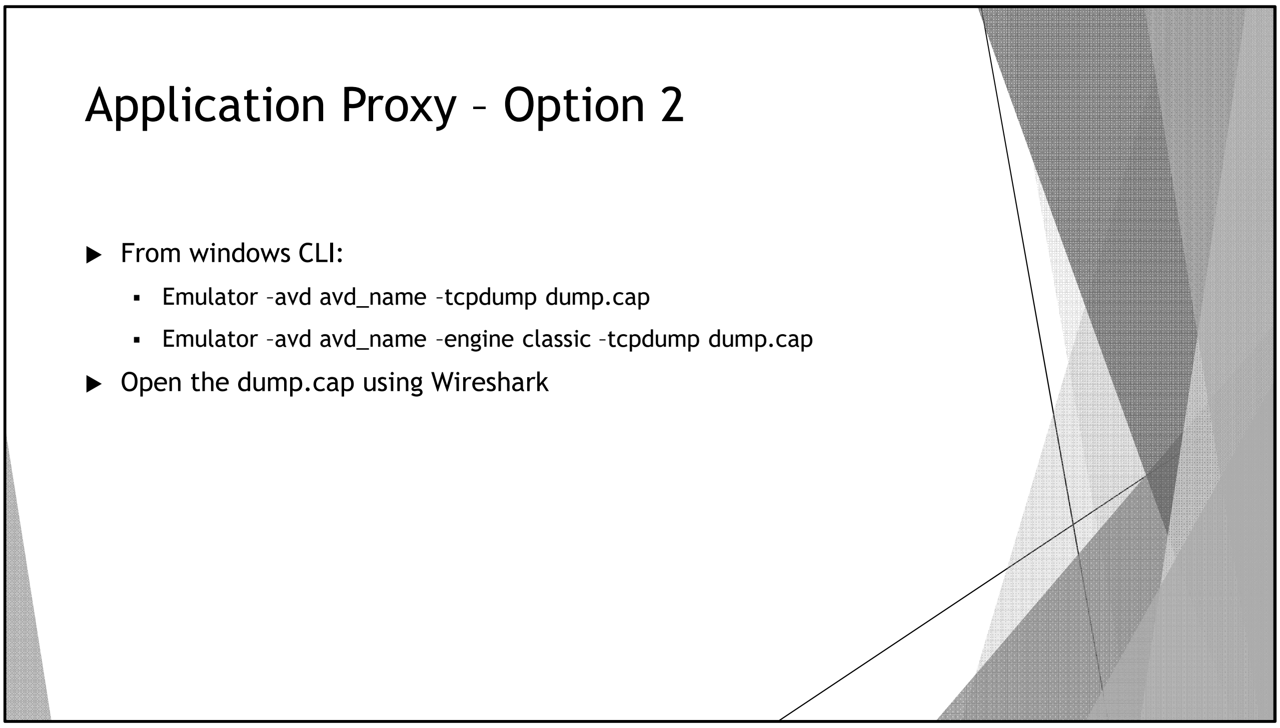
Server

Web Browser Proxying – Option 3

- Set the emulator’s environment variable.
- From Windows CLI:
  - Adb shell
  - Export HTTP_PROXY=http://your_ip_address:8888

Application Proxy – Option 1

- Download tsocks:
  https://sourceforge.net/projects/tsocks/files/tsocks/1.8%20beta%205/
- From windows CLI:
  - Adb push tsocks-1.8beta5.tar.gz /mnt/sdcard/Download
  - Adb shell
  - Cd /mnt/sdcard/Download
  - Tar –zxvf tsocks-1.8beta5.tar.gz
  - Cd tsocks-1.8
  - ./install-sh
- Modify the settings in: /etc/tsocks.conf

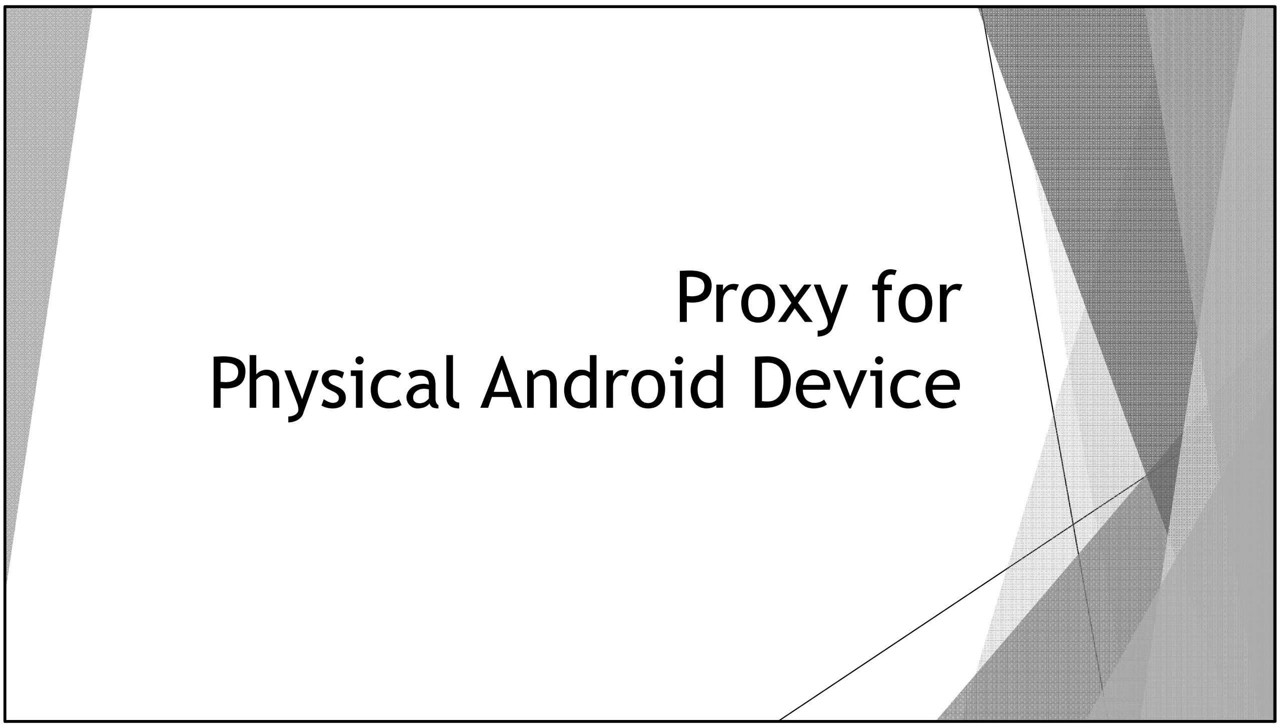


Application Proxy – Option 2

- From windows CLI:
  - Emulator –avd avd_name –tcpdump dump.cap
  - Emulator –avd avd_name –engine classic –tcpdump dump.cap
- Open the dump.cap using Wireshark

Application Proxy – Option 3

- Follow:
  - http://www.devineloper.com/2013/08/28/setup-socks-proxy-android-without-root/



Proxy for Physical Android Device

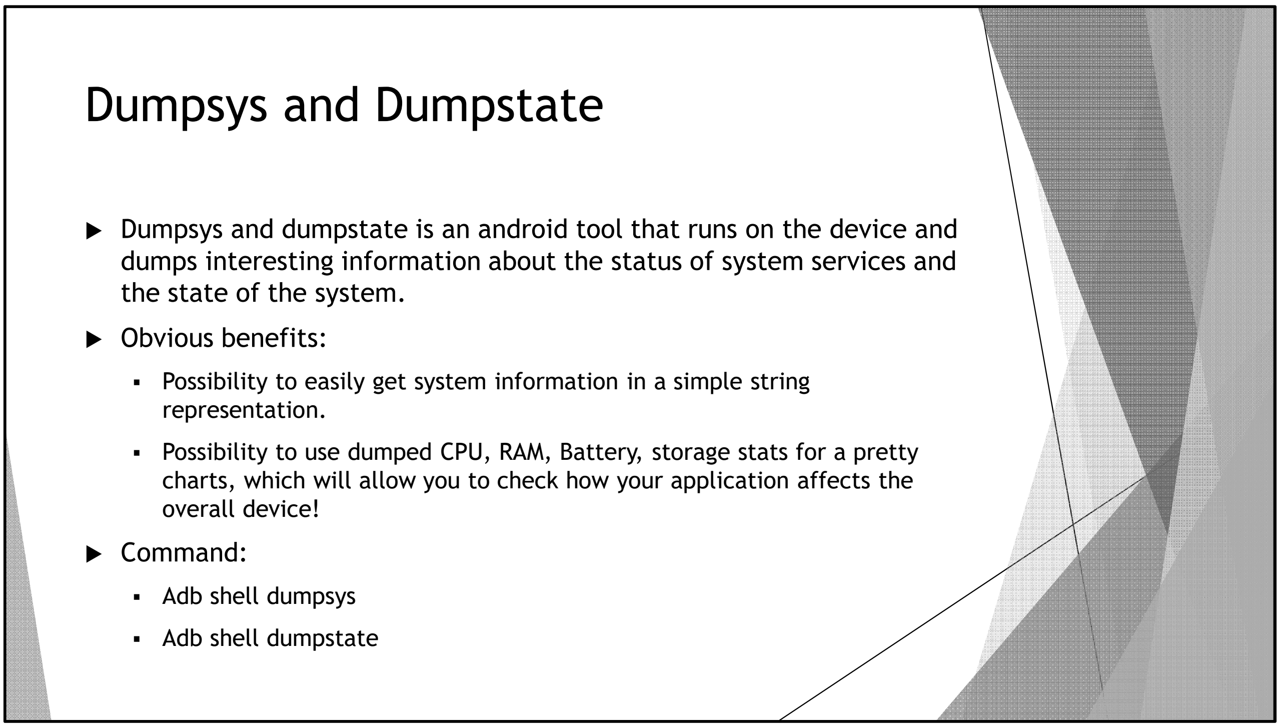
Android Proxy

- Install Autoproxy from Play Store
- Configure Autoproxy:
  - Host: your_computer_ip_address
  - Port: 8888
  - Type: HTTP
- Download Fiddler from:
  https://www.telerik.com/download/fiddler
- Run Fiddler (go to configure Fiddler slide)

Configure Fiddler

- Click WinConfig, Exempt All, Save
- Click Tools, Options:
  - Connections: check the “Allow remote computers to connect”
  - HTTPS: check the capture and decrypt HTTPS
  - Accept certificate (yes to all)
- Restart Fiddler
- From Android phone or emulator:
  - Open browser and browse to: http://your_computer:8888
  - Download the Fiddler root certificate

Dumpsys and Dumpstate on Android



Dumpsys and Dumpstate

- Dumpsys and dumpstate is an android tool that runs on the device and dumps interesting information about the status of system services and the state of the system.
- Obvious benefits:
  - Possibility to easily get system information in a simple string representation.
  - Possibility to use dumped CPU, RAM, Battery, storage stats for a pretty charts, which will allow you to check how your application affects the overall device!
- Command:
  - Adb shell dumpsys
  - Adb shell dumpstate

End of Mobile Application Penetration Testing

Any Questions?