developers

Android App Hacking Workshop

What is this?

To prepare for the Android App Hacking workshop, please review and complete the content below. This includes installing necessary software for the workshop, material to review ahead of time, and a few warm up exercises.

Prerequisites

The following sections define prerequisites for the workshop, including suggested resources to review, and software that **must be installed before the workshop.**

Required Hardware

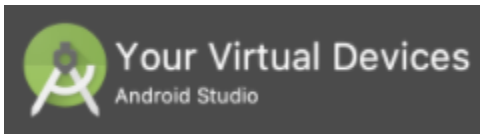
You will need to have your own Mac, Linux, or Windows computer for this workshop capable of running the software below.

Required Software

Before the workshop, please review the instructions below and install each piece of software.

Android Studio

To install Android Studio, visit https://developer.android.com/studio/ and follow the instructions to download and install it.



Android Virtual Device & Running an App

After installing and opening Android Studio, please follow these instructions to create an app, and create an Android Virtual Device and run your app. (If you'd like, you can continue the tutorial's next step to build a user interface, but this is optional.)

Adding commands to your "path"

Whether you are on a Mac, Linux, or Windows, as you install the various software below, you will need to add commands to your "path". This makes it so when you type the name of the tool (like adb or jadx etc.), you don't need to be in the same directory where the tool is now; you can type the name of the tool in any directory and it should work. See below for general instructions on how to do this on a Mac, Linux, and Windows. If these instructions don't make sense right now, that's okay; come back to them after reviewing how to download and install the specific tools below.

Mac

If you downloaded or installed software below, the binary will exist in a file path somewhere, e.g. ~/Downloads/nameOfDownload/someBinary.

First, you'll want to move the downloaded software to another more permanent location (in case you delete or clear your downloads folder in the future). Let's make a "tools" folder in our home directory, then move the downloaded software there.

1. Open Terminal
2. Type the following commands
3. `mkdir ~/tools`
4. `mv ~/Downloads/nameOfDownload ~/tools`

Once that's done, we can add the folder of that tool to our path:

5. First check if you have a `.bash_profile` by using the command **`cat ~/.bash_profile`** if the file doesn't exist create the file with **`touch ~/.bash_profile.`**
6. Secondly, this command will add the binary to the PATH variable and append it to your .bash_profile which is read when a shell is invoked. echo 'export `PATH=${PATH}:~/tools/binaryfolder' >> ~/.bash_profile`
7. Activate the changes in your current shell with source `~/.bash_profile`
8. Now every time the name of the binary is entered into the bash shell it will be recognized! Type the name of the binary, press enter, and the binary should show some kind of output (Usually a list of parameters that can be used with the binary).

Linux

If you've downloaded or installed software below, the binary will exist in a file path somewhere, e.g. `~/Downloads/nameOfDownload/someBinary.`

1. `mkdir ~/tools`
2. `mv ~/Downloads/nameOfDownload ~/tools`

Once that's done, we can add the folder of that tool to our path:

3. First check if you have a .bash_profile by using the command cat `~/.bash_profile` if the file doesn't exist create the file with touch `~/.bash_profile.`
4. This command will add the binary to the PATH variable and append it to your .bash_profile which is read when a shell is invoked. echo 'export `PATH=${PATH}:~/tools/binaryfolder' >> ~/.bash_profile`
5. Activate the changes in your current shell with source `~/.bash_profile`

6. Now every time the name of the binary is entered into the bash shell it will be recognized! Type `nmkdir ~/tools`
7. ame of the binary, press enter, and the binary should show some kind of output (Usually a list of parameters that can be used with the binary).

Windows

If you've downloaded or installed software below, the binary will exist in a file path somewhere, e.g. `C:\Users\B3nac\Downloads`. The process of adding binaries is a bit different in Windows.

1. Open cmd using Windows key + R and typing cmd in the run prompt. Alternatively you can pin cmd to the task bar and click the cmd shortcut to open a new cmd window.
2. Ensure you're in your account's home C:\Users\yourusername directory and type mkdir tools in cmd to create a tools folder
   a. `cd C:\Users\yourusername`
   b. `mkdir tools`
3. Move the binary from the downloads folder to the tools folder `C:\Users\yourusername\tools with copy C:\Users\yourusername\Downloads\binarydownload C:\Users\yourusername\tools`. Or use the file explorer.
4. Add the binary path to the Environment Variables Path variable by running the following:
   a. Open the run prompt by holding down Windows key + R.
   b. Type rundll32.exe sysdm.cpl,EditEnvironmentVariables into the run prompt and click OK, this will bring up the environment variables prompt.
   c. Double click on user variable Path or click on Path to highlight and click Edit.
   d. Add the path to the binary `C:\Users\yourusername\tools\binaryfolder`
   e. Apply the changes
5. Open a new CMD prompt with Windows key + R and typing cmd in the run prompt.
6. Now every time the name of the binary is entered it will be recognized! Type the name of the binary, press enter, and the binary should show some kind of output (Usually a list of parameters that can be used with the binary).

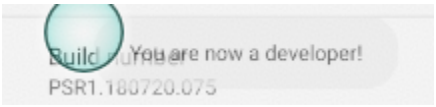
ADB

Once Android Studio and your virtual device are set up, you'll need to install adb (Android Debug Bridge). We will use adb to run commands on our virtual device, including There's detailed info on adb here. This guide on XDA Developers provides step by step instructions on how to install ADB on Windows, Mac, and Linux (please note this guide is referenced for your information and convenience, and does not constitute endorsement, recommendation, or

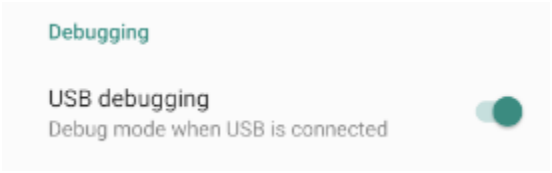
favoring by Google). We've also included more specific instructions below to ensure adb is working correctly on Mac, Windows, and Linux.

Put your virtual device in Developer Mode

1. In your Android Virtual Device, go to Settings
2. Go to system settings
3. Go to about emulated device
4. Scroll to bottom, repeatedly tap on build number
5. You should see a toast message pop up that says "You are now a developer!"



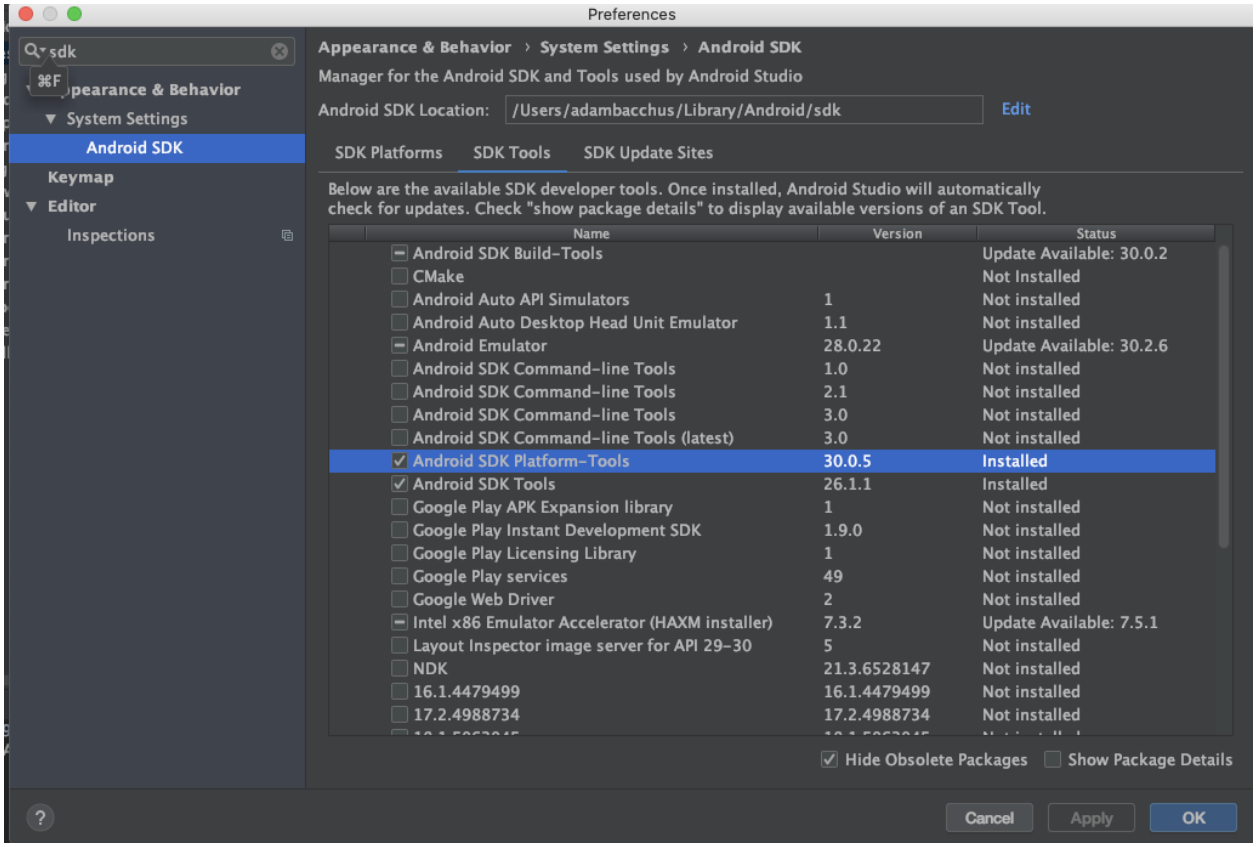
6. Under system, advanced settings, developer options, debugging, ensure USB debugging is turned on



7. You're done with this step! Please proceed to your OS below for instructions on how to install ADB.

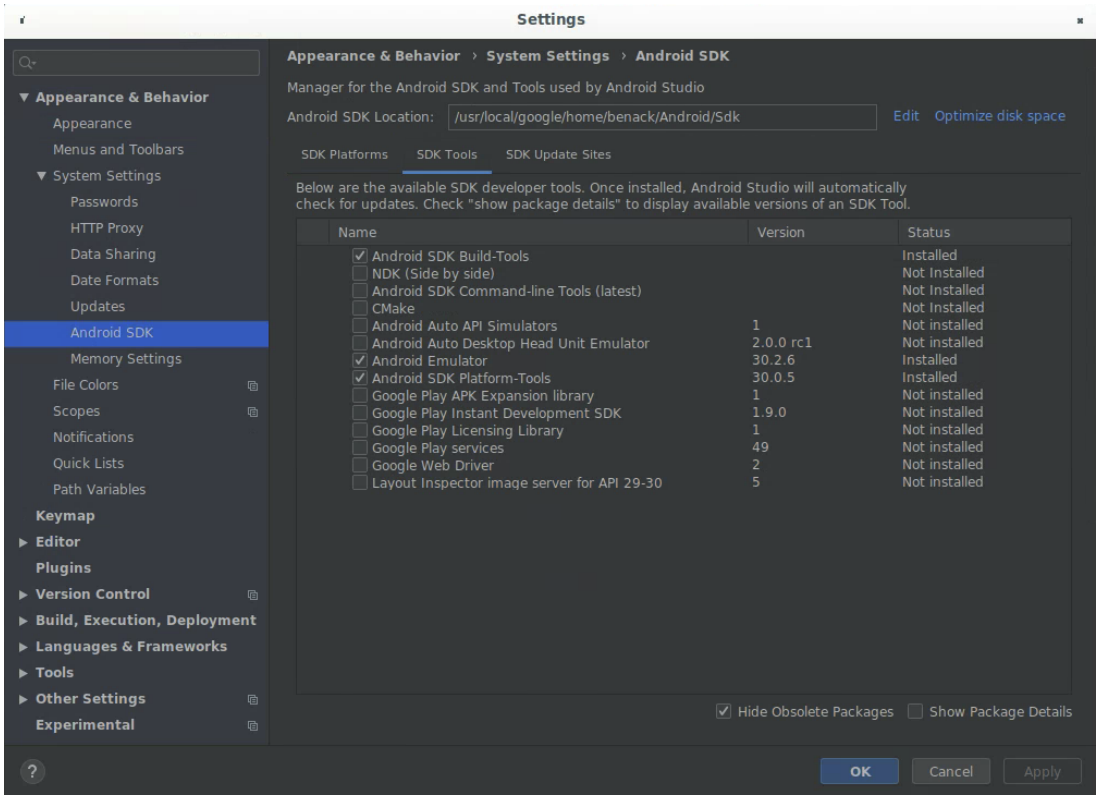
ADB on Mac

1. In Android Studio, click the Android Studio menu in the upper left, then click Android Studio -> Settings
2. In the search bar type sdk
3. This should bring you to System Settings -> Android SDK
4. Click on SDK Tools
5. Look for Android SDK Platform-Tools and check the box next to it, click Apply, and follow the prompts to ensure the Android SDK Platform-Tools are installed
6. Android SDK can be accessed from System Settings



7. You've installed adb, but typing adb in a Terminal might not work, so you'll want to add adb to your "path" - you can do this by…
    a. Open up a Terminal (command + spacebar, type Terminal, hit enter)
    b. Copy/paste this into your Terminal
    c. `echo 'export ANDROID_HOME=/Users/$USER/Library/Android/sdk' >> ~/.bash_profile`
    d. Hit enter, then copy/paste the following command
    e. echo 'export `PATH=${PATH}:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools' >> ~/.bash_profile`
    f. Hit enter
    g. Copy/paste the following command
    h. cat ~/.bash_profile
    i. Hit enter - you should see the two "export" strings above in your .bash profile - we did this so whenever you type adb in your Terminal, it should just work
    j. Try it! Type adb and hit enter

ADB on Linux

1. In Android Studio, click on File in the upper left, then click Settings
2. In the Settings menu click on the arrow drop-down for System Settings
3. Click on Android SDK in the drop-down menu
4. Click on SDK Tools
5. Look for Android SDK Platform-Tools and check the box next to it, click Apply, and follow the prompts to ensure the Android SDK Platform-Tools are installed
6. 

7. You've installed adb, but typing adb in a Terminal might not work, so you'll want to add adb to your "path" - you can do this by…
    a. Open up a Terminal (command + spacebar, type Terminal, hit enter)
    b. Copy/paste this into your Terminal
    c. echo 'export ANDROID_HOME=/Users/$USER/Library/Android/sdk' >> ~/.bash_profile
    d. Hit enter, then copy/paste the following command
    e. echo 'export PATH=${PATH}:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools' >> ~/.bash_profile
    f. Hit enter
    g. Copy/paste the following command
    h. cat ~/.bash_profile

i. Hit enter - you should see the two “export” strings above in your bash profile - we did this so whenever you type adb in your Terminal, it should just work
j. Try it! Type adb and hit enter

ADB on Windows

1. In Android Studio, click on Tools, then click SDK Manager
2. In the Settings menu click on the arrow drop-down for System Settings
3. Click on Android SDK in the drop-down menu
4. Click on SDK Tools
5. Look for Android SDK Platform-Tools and check the box next to it, click Apply, and follow the prompts to ensure the Android SDK Platform-Tools are installed
6. Test if adb was added to your path by typing adb in a cmd prompt after the install is complete.

If adb wasn’t added to your path automatically we have to add the SDK location to the Path environment variable on Windows:

1. Open the run prompt by holding down Windows key + R.
2. Type rundll32.exe sysdm.cpl,EditEnvironmentVariables into the run prompt and click OK, this will bring up the environment variables prompt.
3. Double click on user variables Path or click on Path to highlight and click Edit.
4. Add the path to the binary C:\Users\yourusername\AppData\Local\Android\Sdk\platform-tools
5. Apply the changes
6. Open a new CMD prompt with Windows key + R and typing cmd in the run prompt.
7. Now every time adb is entered the binary will be recognized! Type adb, press enter, and the binary should show some kind of output (Usually a list of parameters that can be used with the binary).

apktool

Apktool is used to reverse engineer APK files (Android apps) by decoding all their embedded files. We’ll use apktool to decompile Android apps in the workshop. To install apktool, visit https://ibotpeaches.github.io/Apktool/ and follow the installation instructions.

apktool on Mac

Visit https://ibotpeaches.github.io/Apktool/ and follow the installation instructions.

apktool on Windows

Visit https://ibotpeaches.github.io/Apktool/ and follow the installation instructions.

apktool on Linux

Visit https://ibotpeaches.github.io/Apktool/ and follow the installation instructions.

Python & pip

Pip is a package management tool that'll make it easier to install other software. Please install pip with the instructions below.

Python & pip on Mac

You'll want Python3 for this workshop - to see if you already have it installed, open a terminal (command key + spacebar, type terminal, hit enter) then type python3 and hit enter. If the output of your command is "you already have it - you can type exit() to exit", pip is already installed if you are using Python 2 >=2.7.9 or Python 3 >=3.4 downloaded from python.org or if you are working in a Virtual Environment created by virtualenv or pyvenv.

```
$ python3
Python 3.6.8 (v3.6.8:3c6b436a57, Dec 24 2018, 02:04:31)
[GCC 4.2.1 Compatible Apple LLVM 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
```

If you don't have it installed, you'll need to follow the instructions here

Python & pip on Windows

1. Download the latest version of Python from https://www.python.org/downloads/
2. Double click on the installer in the Downloads folder and follow the prompts
3. After the installer is finished type python in cmd to check if Python was added to the PATH environment variables

Python & pip on Linux

1. Check if Python is installed with by typing python3 in the terminal

   If not installed:
   a. sudo apt update
   b. sudo apt install python3.8
   c. Verify the installation was successful by running python3 – –version

developers

JDK 11

You'll need to install JDK (Java Development Toolkit) as a variety of the other software we'll install will require it.

JDK 11 on Mac

1. Create an Oracle account to download JDK 11 or higher (An account is now required to download). Note: OpenJDK could also probably be used on Mac. https://www.oracle.com/java/technologies/javase-jdk11-downloads.html

2. Install JDK 11 after the download is finished by double clicking the .dmg file.

JDK 11 on Windows

1. Create an Oracle account to download JDK 11 or higher (An account is now required to download). Note: OpenJDK could also probably be used on Windows. https://www.oracle.com/java/technologies/javase-jdk11-downloads.html

2. Install JDK 11 after the download is finished by double clicking the executable in the Downloads folder.

JDK 11 on Linux

1. Install openjdk 11 with `sudo apt install openjdk-11-jdk`
2. Check if openjdk 11 was added to your path with `echo $PATH`

jadx

Jadx is a tool we will use to decompile "Dex" (Dalvik bytecode) to Java, which will make it easier for us to read and understand the underlying code inside of an Android app.

developers

Jadx on Mac

1. To install Jadx, visit https://github.com/skylot/jadx, look for “Releases,” and find the latest release.
2. Scroll to the bottom of the latest release page and download the jadx-latest version.zip file (the version # may be different if new releases have come out)
3. Open the zip file
4. Open your terminal by hitting command + spacebar, typing terminal, hitting enter
5. Type cd ~/Downloads/jadx-latest version/bin, hit enter
6. Type ls and hit enter - this will show you the files in the directory you’re in
7. Type ./jadx-gui & and hit enter
8. If you see a message that says “java” would like to access files in your Downloads folder.” hit ok
9. This should open the jadx GUI
10. From here you can open up .apk files, which are Android apps, and look at the contents inside of it.

Jadx on Windows

Setting up environment variables on Windows

1. Ensure JDK 11 is added to your path environment variables
2. Open the run prompt by holding down Windows key + R.
3. Type rundll32.exe sysdm.cpl,EditEnvironmentVariables into the run prompt and click OK, this will bring up the environment variables prompt.
4. Double click on user variables Path or click on Path to highlight and click Edit.
5. All the path variables should now be listed and the path to JDK 11 should have been added by the installer.

Installing Jadx on Windows

1. To install Jadx, visit https://github.com/skylot/jadx, look for “Releases,” and find the latest release.
2. Open this url in your browser to download the compiled binaries for jadx latest version.zip
3. Open the zip file in the Downloads folder and extract all files
4. Open your terminal by holding down Windows key + R, typing cmd in the run prompt, and click OK.
5. Type cd C:\Users\yourusername\Downloads\jadx-latest version\bin, hit enter
6. Type dir and hit enter - this will show you the files in the directory you’re in

7. Type start jadx-gui.bat and hit enter
8. If you see a message that says “java would like to access files in your Downloads folder.” hit ok
9. This should open the jadx GUI
10. From here you can open up .apk files, which are Android apps, and look at the contents inside of it.

Jadx on Linux

1. To install Jadx, visit https://github.com/skylot/jadx, look for “Releases,” and find the latest release.
2. Download the compiled binaries for jadx-latest version.zip
    a. Or use wget via command line:
    wget https://github.com/skylot/jadx/releases/download/latest version/jadx-latest version.zip
3. Open the zip file via GUI and extract the files
    a. Or navigate to the zip file via terminal and use the unzip command
4. Open a terminal with CTRL + ALT + T
5. Type cd ~/Downloads/jadx-latest version/bin, hit enter
6. Type ./jadx-gui & and hit enter
7. The Jadx GUI will load and from there everything is awesome

Burp Suite

Burp Suite is a tool that lets you “proxy” traffic between your device and the server. We will use Burp Suite to intercept requests from the app on your phone before they make it to the server.

Burp Suite on Mac

1. Download Burp Suite Community Edition here https://portswigger.net/burp/releases/community/latest
2. Select Mac OSX and click download
3. After the download is complete the file burpsuite_community_macos_v2020_11_3.dmg will be added to your Downloads folder.
4. Run the installer in your Downloads folder.
5. Setup Burp Suite to work with your favorite browser using these instructions https://portswigger.net/burp/documentation/desktop/getting-started/proxy-setup/browser
6. After Burp Suite is setup with your browser navigate to http://burpsuite and download the certificate
7. Navigate to ~/Downloads and change the file extension of the downloaded certificate to .cer.

8. Use adb push cert.cer sdcard/Download to transfer the cert via usb to your testing device's Downloads folder.
9. These instructions will show how to enable permissions and install the certificate on your android device https://portswigger.net/support/installing-burp-suites-ca-certificate-in-an-android-device
10. After setting up the certificate these instructions show how to set up your device to forward traffic to your Burp Suite proxy on your PC. https://portswigger.net/support/configuring-an-android-device-to-work-with-burp
11. After these steps are completed all traffic should now be intercepted by Burp Suite. An easy test is opening the browser on your device and trying to navigate to a website. With intercept enabled this request should show up in Burp Suite.

Burp Suite on Windows

1. Download Burp Suite Community Edition here https://portswigger.net/burp/releases/community/latest
2. Select Windows 64 bit and click download
3. After the download is complete the file burpsuite_community_windows-x64_v2020_11_1.exe will be added to your Downloads folder.
4. Run the installer in your Downloads folder.
5. Setup Burp Suite to work with your favorite browser using these instructions https://portswigger.net/burp/documentation/desktop/getting-started/proxy-setup/browser
6. After Burp Suite is setup with your browser navigate to http://burpsuite and download the certificate
7. Navigate to ~/Downloads and change the file extension of the downloaded certificate to .cer.
8. Use adb push cert.cer sdcard/Download to transfer the cert via usb to your testing device's Downloads folder.
9. These instructions will show how to enable permissions and install the certificate on your android device https://portswigger.net/support/installing-burp-suites-ca-certificate-in-an-android-device
10. After setting up the certificate these instructions show how to set up your device to forward traffic to your Burp Suite proxy on your PC. https://portswigger.net/support/configuring-an-android-device-to-work-with-burp
11. After these steps are completed all traffic should now be intercepted by Burp Suite. An easy test is opening the browser on your device and trying to navigate to a website. With intercept enabled this request should show up in Burp Suite on your PC.

Burp Suite on Linux

1. Download Burp Suite Community Edition here https://portswigger.net/burp/releases/community/latest
2. Select Linux 64 bit and click download
3. After the download is complete the file burpsuite_community_linux_v2020_11_1.sh will be added to your ~/Downloads folder.
4. Permissions will need to be changed in order to run BurpSuite with chmod +x filename
5. In your terminal use ./filename & to start BurpSuite
6. Setup Burp Suite to work with your favorite browser using these instructions https://portswigger.net/burp/documentation/desktop/getting-started/proxy-setup/browser
7. After Burp Suite is setup with your browser navigate to http://burpsuite and download the certificate
8. Navigate to ~/Downloads and change the file extension of the downloaded certificate to .cer.
9. Use adb push cert.cer sdcard/Download to transfer the cert via usb to your testing device's Downloads folder.
10. These instructions will show how to enable permissions and install the certificate on your android device https://portswigger.net/support/installing-burp-suites-ca-certificate-in-an-android-device
11. After setting up the certificate these instructions show how to set up your device to forward traffic to your Burp Suite proxy on your PC. https://portswigger.net/support/configuring-an-android-device-to-work-with-burp
12. After these steps are completed all traffic should now be intercepted by Burp Suite. An easy test is opening the browser on your device and trying to navigate to a website. With intercept enabled this request should show up in Burp Suite on your PC.

Wireshark

Network protocol analyzer.

Wireshark on Mac

1. Download the installer macOS Intel 64-bit .dmg from https://www.wireshark.org/#download
2. Double click installer in the downloads directory to start setting up Wireshark
3. Follow the setup instructions and ensure Npcap is installed
4. When the installer is finished start Wireshark and select the Wifi interface
5. Find your phones internal private ip address 192.168. 0.0/16 with adb shell ifconfig wlan0
6. Filter outgoing and incoming traffic from that ip by inputting this filter into the "Apply a display" filter search box ip.addr == yourphoneip traffic to and from your phone over wifi should now start showing up in Wireshark!

developers

Wireshark on Windows

1. Download the Windows 32 or 64 bit installer that matches your architecture from https://www.wireshark.org/#download
2. Double click installer in the downloads directory to start setting up Wireshark
3. Click yes to allow Wireshark to install on Windows
4. Follow the setup instructions and ensure Npcap is installed
5. When the installer is finished start Wireshark and select the Wifi interface
6. Find your phones internal private ip address 192.168. 0.0/16 with adb shell ifconfig wlan0
7. Filter outgoing and incoming traffic from that ip by inputting this filter into the “Apply a display” filter search box ip.addr == yourphoneip
8. Traffic to and from your phone over wifi should now start showing up in Wireshark!

Wireshark on Linux

1. Install wireshark with sudo apt install wireshark
    a. The capture session won’t be initiated until the running the following commands:
    b. Sudo dpkg-reconfigure wireshark-common
    c. Selecting <Yes> in response to “Should non-superusers be able to capture packets?”
    d. Add yourself to the “wireshark” group with: sudo usermod -a -G wireshark yourusername
    e. Logout and log back in again.
2. Find your phones internal private ip address 192.168.0.0/16 with adb shell ifconfig wlan0
3. Filter outgoing and incoming traffic from that ip by inputting this filter into the “Apply a display” filter search box ip.addr == yourphoneip
4. Traffic to and from your phone over wifi should now start showing up in Wireshark!

Frida

Free dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers. This is another tool we’ll use in the exercises in the workshop.

Frida on Mac

1. Run the following command pip3 install frida-tools --user via terminal
2. Follow the quick setup instructions to ensure frida is setup correctly https://frida.re/docs/installation/
3. Setup frida-server on device with adb https://frida.re/docs/android/
    a. Note: Phone architecture needs to match frida-server architecture.

4. Download the frida-server that matches your phone's architecture from the releases page https://github.com/frida/frida/releases
5. Rename the download to frida-server with the mv command mv downloadname frida-server
6. Use the following adb commands to push the server to your device, make the binary executable, and run the server so that Frida can connect
7. adb root # might be required
8. adb push frida-server /data/local/tmp/
9. adb shell "chmod 755 /data/local/tmp/frida-server"
10. adb shell "/data/local/tmp/frida-server &"
11. Test the connection with frida-ps -U
    a. Add the path to your frida binary to invoke frida commands from any location. Example location on Mac: /Users/yourusername/Library/Python/3.8/bin
    b. You can list the location with pip3 show package-name

Frida on Windows

1. Run the following command pip install frida-tools via cmd
2. Follow the quick setup instructions to ensure frida is setup correctly https://frida.re/docs/installation/
3. Setup frida-server on device with adb https://frida.re/docs/android/
    a. Note: Phone architecture needs to match frida-server architecture.
4. Download the frida-server that matches your phone's architecture from the releases page https://github.com/frida/frida/releases
5. Rename the download to frida-server with the mv command mv downloadname frida-server
6. Use the following adb commands to push the server to your device, make the binary executable, and run the server so that Frida can connect
7. adb root # might be required
8. adb push frida-server /data/local/tmp/
9. adb shell "chmod 755 /data/local/tmp/frida-server"
10. adb shell "/data/local/tmp/frida-server &"
11. Test the connection with frida-ps -U

Frida on Linux

1. Run the following command pip install frida-tools via bash terminal
2. Follow the quick setup instructions to ensure frida is setup correctly https://frida.re/docs/installation/
3. Setup frida-server on device with adb https://frida.re/docs/android/
    a. Note: Phone architecture needs to match frida-server architecture.

4. Download the frida-server that matches your phone's architecture from the releases page https://github.com/frida/frida/releases
5. Rename the download to frida-server with the mv command mv downloadname frida-server
6. Use the following adb commands to push the server to your device, make the binary executable, and run the server so that Frida can connect
7. adb root # might be required
8. adb push frida-server /data/local/tmp/
9. adb shell "chmod 755 /data/local/tmp/frida-server"
10. adb shell "/data/local/tmp/frida-server &"
11. Test the connection with frida-ps -U

Ghidra

A software reverse engineering (SRE) suite of tools developed by NSA's Research Directorate in support of the Cybersecurity mission.

Ghidra on Mac

1. Make sure Java is installed by running java in the terminal.
2. Download Ghidra from https://ghidra-sre.org/
3. Unzip the downloaded file in your home directory.
4. Go into the ghidra directory and run ./ghidraRun to start the Ghidra.

Ghidra on Windows

1. Make sure a supported JDK is installed. **JDK 11+ is required as indicated here https://ghidra-sre.org/InstallationGuide.html**
2. Download Ghidra from https://ghidra-sre.org/
3. Unzip downloaded ZIP file
4. In the root directory of the unzipped Ghidra package you can find a batch script called ghidraRun.bat.
5. Start Ghidra with start ghidraRun.bat

Ghidra on Linux

1. Make sure you have Java installed
2. Download Ghidra from https://ghidra-sre.org/
3. Unzip downloaded ZIP file
4. In the root directory of the unzipped Ghidra package you can find a shell script called ghidraRun.

5. Start Ghidra with ./ghidraRun

Pre-workshop Exercises

Warm up

To get ready for the workshop, please try out the following exercises.

Testing out commands & adding to path

Mac

1. Press COMMAND + SPACE BAR to open the search box.
2. Type Terminal and press enter to open the terminal window.
3. Run adb in the terminal to make sure it is working.

```
$ adb devices
$ adb shell
$ adb shell pm list packages
$ adb install /path/to/your/app.apk
$ adb uninstall package.name
```

Windows

1. Open command prompt with Windows key + R and type cmd
2. Type adb or adb --help in the cmd windows and press enter
3. If the list of commands display then everything is good to go
   If there's an error:
   a. Open the run prompt by holding down Windows key + R.
   b. Type rundll32.exe sysdm.cpl,EditEnvironmentVariables into the run prompt and click OK, this will bring up the environment variables prompt.
   c. Double click on user variables Path or click on Path to highlight and click Edit.
   d. Enter the path to your adb binary and click ok.
   e. Try typing adb or adb --help in cmd again and a list of commands should show up

Linux

1. Press CTRL + ALT + T to open a new terminal window.
2. Type adb or adb --help in terminal cmd window and press enter
3. If the list of commands display then everything is good to go

If there's an error:

a. Check if adb is in your path with echo $PATH the adb binary should be located at /usr/bin
b. If adb is not in /usr/bin copy/paste this into your Terminal
c. echo 'export ANDROID_HOME=/Users/$USER/Library/Android/sdk' >> ~/.bash_profile
d. Hit enter, then copy/paste the following command
e. echo 'export PATH=${PATH}:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools' >> ~/.bash_profile
f. Hit enter
g. Copy/paste the following command
h. cat ~/.bash_profile
i. Hit enter - you should see the two "export" strings above in your bash profile - we did this so whenever you type adb in your Terminal, it should just work
j. Try it! Type adb and hit enter

```
# print help of adb command
$adb --help

# run python3 then exit python session
$ python3
>>> exit()
or use CTRL-D with your keyboard
>>> Ctrl-D

# launch jadx
$ ./jadx-gui &
```

List the content of the /sdcard repository

1. Ensure device is connected via usb or an emulator is started
2. Ensure adb server is started
    a. adb start-server
3. 1st way:
    a. adb shell
    b. ls /sdcard
4. 2nd way:
    a. adb shell "ls /sdcard"

List all the package names installed on your device

1. adb shell "pm list packages"

2. To list with associated paths adb shell “pm list packages -f”

Installing apps

We will provide access to custom APKs built for the workshop when it starts, but in the meantime, please try the following steps to install an app and then extract its APK from your virtual or physical device.

Installing an app from Play

Pleases download the following application from the Google Play store: Google Authenticator

- ★ After you have downloaded the Google Authenticator application pull the apk from your device to your pc with adb.
  - adb shell pm path com.google.android.apps.authenticator2
  - adb pull pathtoapk

- ★ The com.google.android.apps.authenticator2 application should now be on your pc with the file name of base.apk. Rename base.apk to com.google.android.apps.authenticator2.apk so the app doesn’t accidentally get overwritten the next time we adb pull an application off of a device.

Now we are ready to test out some of the tools that are setup on your machine.

List all the permissions requested by the com.google.android.apps.authenticator2 application

1. An APK file is a ZIP file
   - so you can make a copy of it and change its extension and extract the archive and open the file AndroidManifest.xml. The content can not be read immediately as the file is encoded in a binary format. We need a new tool to deal with the content of APK files.
2. Use apktool
   - apktool d com.google.android.apps.authenticator2.apk
   - Display the content of AndroidManifest.xml to get the permissions requested by the app.
3. Use Jadx
   - Open the com.google.android.apps.authenticator2.apk file with Jadx
   - On the left panel, unroll com.google.android.apps.authenticator2.apk > Resources and click on AndroidManifest.xml

Additional Resources

Build your first app - a step by step guide that shows you how to build an Android app and run it on an emulated Android device with Android Studio. Please note this is not a prerequisite for the workshop, but is helpful to learn the basics of how to build and run an app on an emulated device.

Ubuntu command line for beginners - guide that shows how to use the command line in Ubuntu. Ubuntu is an open source operating system on Linux. You do not need to install Ubuntu, but if you'd like to learn how to, see this guide.