Hacking the Android APK

DEF CON 27



Copyright © 2019 Ben Hughes and Polito, Inc.

Hacking the Android APK
DC27
08/08/2019, Thursday, 1430-1830 in Flamingo, Red Rock V

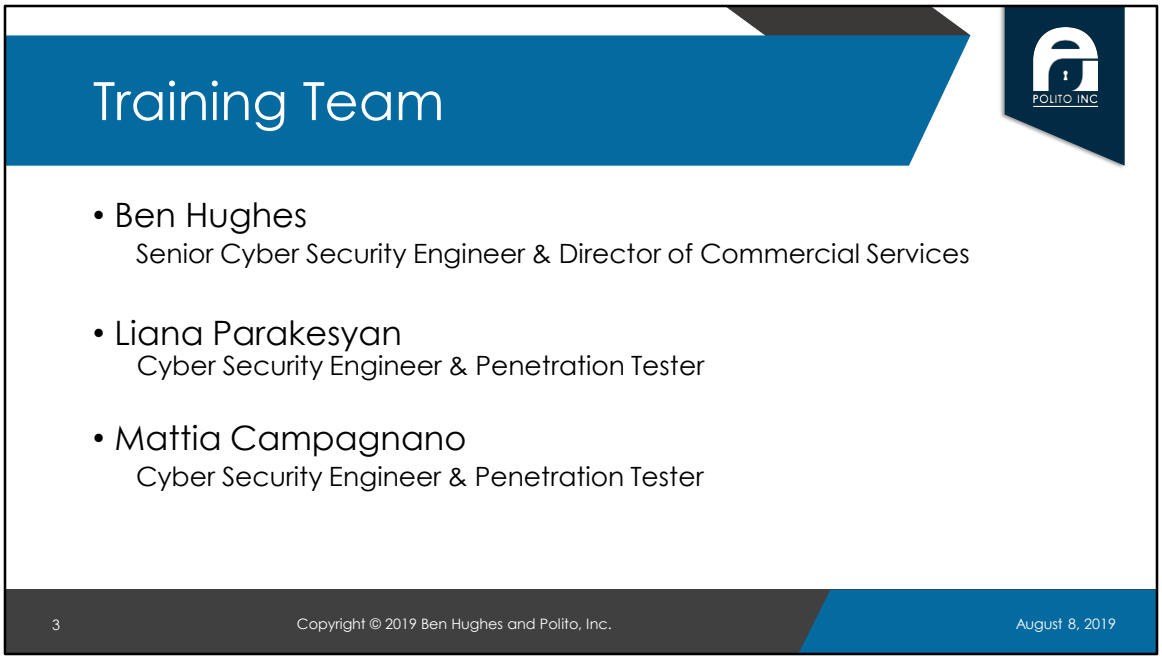
Schedule & Agenda



- 1430 Intro and VM
- 1500 Static and Dynamic Analysis
- 1630 Forensic Analysis
- 1700 Example APK Teardowns
- 1730 CTF!

Copyright © 2019 Ben Hughes and Polito, Inc.

Hacking the Android APK
DC27
08/08/2019, Thursday, 1430-1830 in Flamingo, Red Rock V



Trainer intros/bios

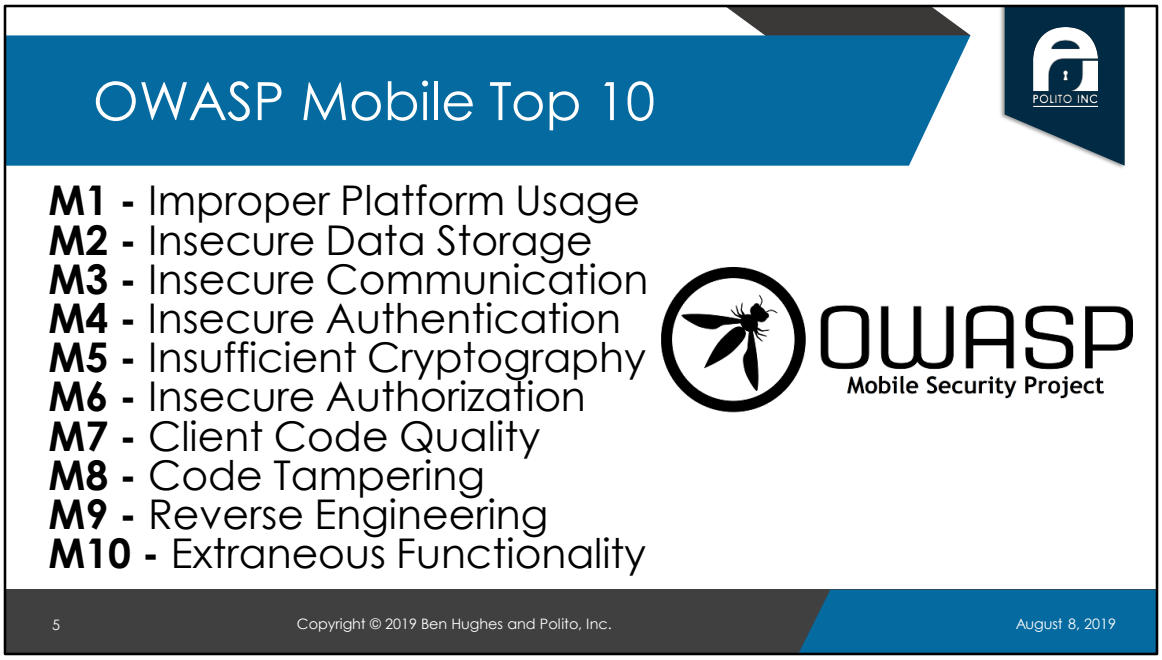
Ben (@CyberPraesidium) brings over 12 years of diverse experience in cyber security, IT, and law. He leads Polito's commercial services including vulnerability assessments, penetration testing, incident response, forensics, and threat hunting. Prior to joining Polito, Ben worked on APT hunt teams at federal and commercial clients. He holds CISSP, GCFA, GWAPT, and Splunk Power User certifications.

Liana has a wide range of experience in cybersecurity. She has created tailored cybersecurity frameworks for companies and federal agencies. She has a background in building cybersecurity labs for clients, consulting on Defense-in-Depth strategies based on threat modeling, and performing penetration testing. She holds a Master's degree in Cybersecurity and has earned the Security+, CEH, and CISSP certifications.

Mattia brings a wide range of experience in IT and cybersecurity, including as Desktop Support with the Italian agency for foreign trade and as a SOC analyst with a major US cybersecurity company. He has worked with SIEMs and conducted penetration testing. He has two Associate's of Applied Science degrees from Stark State College (Cyber Security & Forensics and Network Security, Linux Database Admin). He also

has an MBA from Università di Napoli Federico II (Italy) and Security+ certification.


POLITO INC
Introduction to Android and Mobile Security Fundamentals
4
Copyright © 2019 Ben Hughes and Polito, Inc.
August 8, 2019



OWASP Mobile Top 10 (2016) Categories:
M1 - Improper Platform Usage
M2 - Insecure Data Storage
M3 - Insecure Communication
M4 - Insecure Authentication
M5 - Insufficient Cryptography
M6 - Insecure Authorization
M7 - Client Code Quality
M8 - Code Tampering
M9 - Reverse Engineering
M10 - Extraneous Functionality

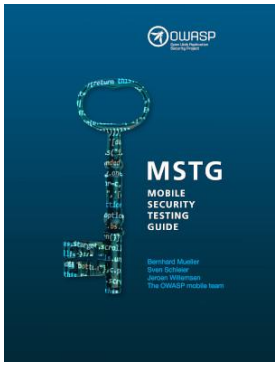
Walk through examples for each category.

References:
https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10
https://www.owasp.org/index.php/OWASP_Mobile_Security_Project
https://github.com/OWASP/owasp-mstg

OWASP Mobile Security Testing Guide (MSTG)

POLITO INC

- "The MSTG is a comprehensive manual for mobile app security testing and reverse engineering. It describes technical processes for verifying the controls listed in the OWASP Mobile Application Verification Standard (MASVS)."



Copyright © 2018 Ben Hughes and Polito, Inc.

Reference:
https://github.com/OWASP/owasp-mstg

Android vs. iOS Pen Testing



- For hackers, pen testers, and security researchers there are pros and cons with each platform
- Some generalizations:
  - For better or worse, the Android platform is not the iOS walled garden offered by Apple
  - Unlocking and rooting Android devices is easier and provides more options than jailbreaking iOS devices
  - Reversing APKs is easier than IPAs; e.g. often can reverse complete cleartext Java classes for APKs, but typically not going to see much actual source code for IPAs unless provided separately by devs
  - Tampering with / modifying APKs is easier than IPAs.
  - There are more open source / free and commercial tools for the Android platform
    - Tooling for iOS often breaks after major iOS updates; many iOS open source / free tools stopped working reliably or at all several iOS versions ago

Copyright © 2019 Ben Hughes and Polito, Inc.

Android vs. iOS Pen Testing



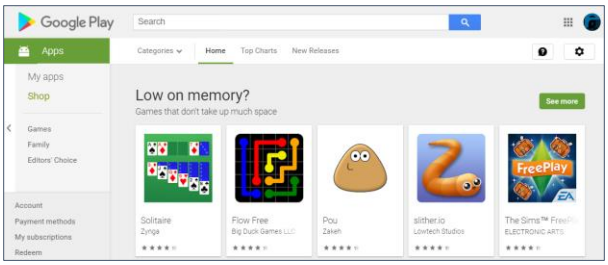
- Those general differences aside:
  - If the same app is available as an APK and IPA, start with analyzing the APK
  - The network/web/API traffic is often identical or virtually identical
  - The main SQLite databases and other on-device artifacts are often identical or virtually identical
  - Cross-compiling mobile apps for Android and iOS platforms is becoming increasingly popular

Copyright © 2019 Ben Hughes and Polito, Inc.

Android Ecosystem



To publish an app, a developer creates a Google Play developer account, fills out some forms in the Play Console about the app and store listing, and uploads the APK to be released.



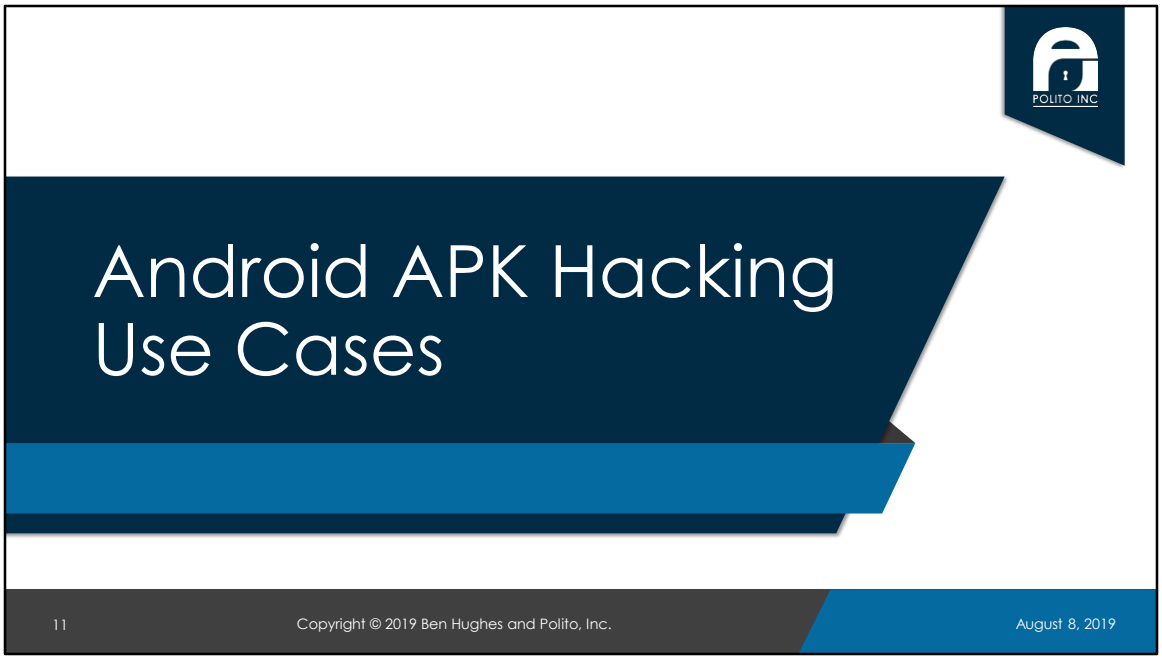
Copyright © 2019 Ben Hughes and Polito, Inc.

Where to Obtain APKs



- Google Play Store
- APK repos
- GitHub
- Android Devices and Emulators
- Developers (test or pre-release versions)

Copyright © 2019 Ben Hughes and Polito, Inc.


POLITO INC
Android APK Hacking Use Cases
11
Copyright © 2019 Ben Hughes and Polito, Inc.
August 8, 2019

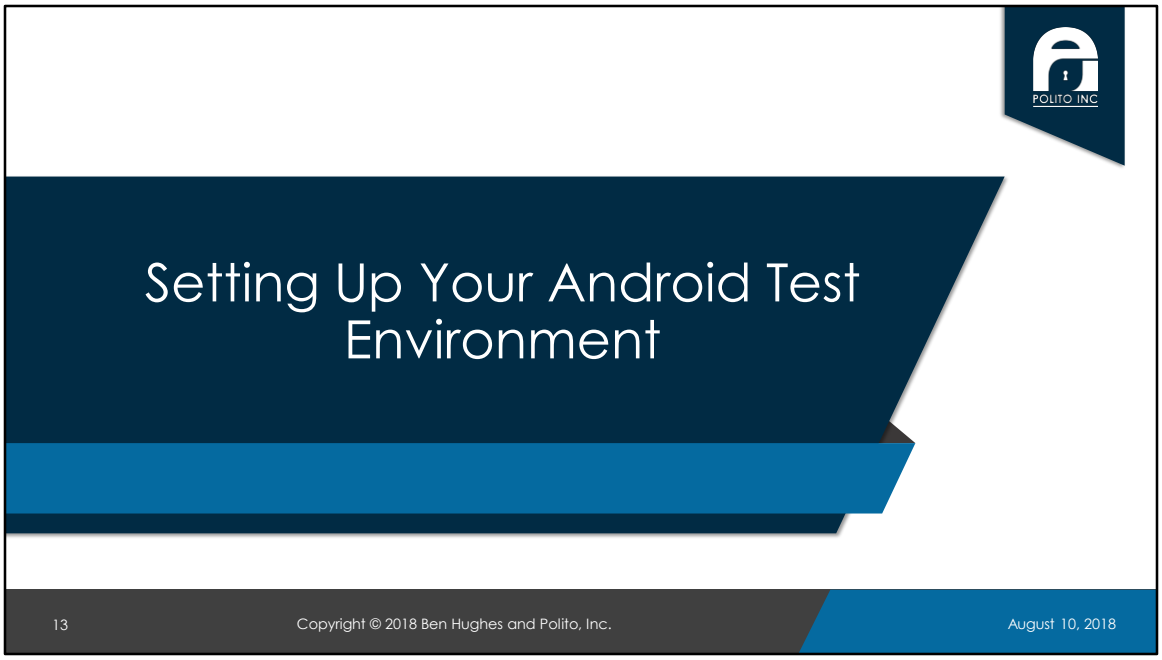
Why Analyze APKs?



- Mobile appsec and mobile penetration testing
  - Specific APKs and Android platform itself
  - For pivoting elsewhere
- Bug bounties
- Mobile malware/adware/APT research
- Mobile forensics
- Curiosity – what are all the apps on your phone doing with your data?



Copyright © 2019 Ben Hughes and Polito, Inc.



For testing an Android app, you can utilize a physical device or an emulator, such as the one provided by Android Studio.

Pros and cons of each approach

Setting Up Your Android Test Environment - VM



- You can create a dedicated VMware or VirtualBox VM to be used as your Android Test Environment.
- Your Test Environment can be hosted on Linux or Windows.
- For example, you can download then customize a pre-built virtual machine from the following websites:
  - Kali Linux: https://www.kali.org/downloads/
  - Windows: https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/

Copyright © 2019 Ben Hughes and Polito, Inc.

Setting Up Your Android Test Environment



Android Studio and adb tools

Android Studio is available for Windows, macOS and Linux and allows you to build and analyze an Android app in APK format and to run it in a device emulator. It can be downloaded from: https://developer.android.com/studio/

Android Debug Bridge (adb) is a versatile command-line tool providing a command shell on the Android device. It is included in the Android SDK Platform-Tools package, available within the SDK Manager, or as a standalone package: https://developer.android.com/studio/releases/platform-tools.html.

Copyright © 2019 Ben Hughes and Polito, Inc.

Setting Up Your Android Test Environment

Android Studio Emulator: Android Studio includes an emulator for all supported API levels / Android versions, allowing you to install any Android app by simply dragging and dropping it to the virtual device. Emulators have their pros and cons, compared with a physical Android device.



Copyright © 2019 Ben Hughes and Polito, Inc.

Setting Up Your Android Test Environment



Physical rooted device: You'll need an Android device with readily available (and relatively trustworthy) root exploits and unlocked/unlockable bootloaders. Some examples of suitable devices include:

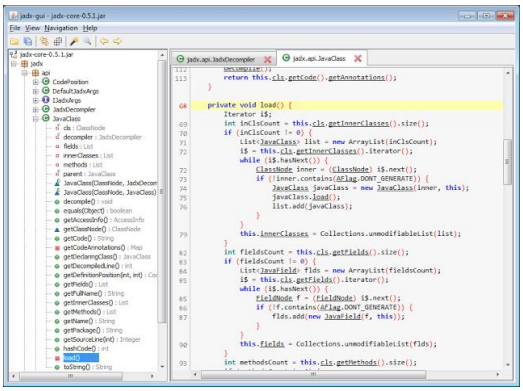
- Google Nexus - older models
- Samsung Galaxy S3 (rooted Verizon version)
- Some Motorola models

Copyright © 2019 Ben Hughes and Polito, Inc.

Additional Tools – Popular Free Tools

- Apktool
- dex2jar
- JD-GUI
- Jadx
- Drozer

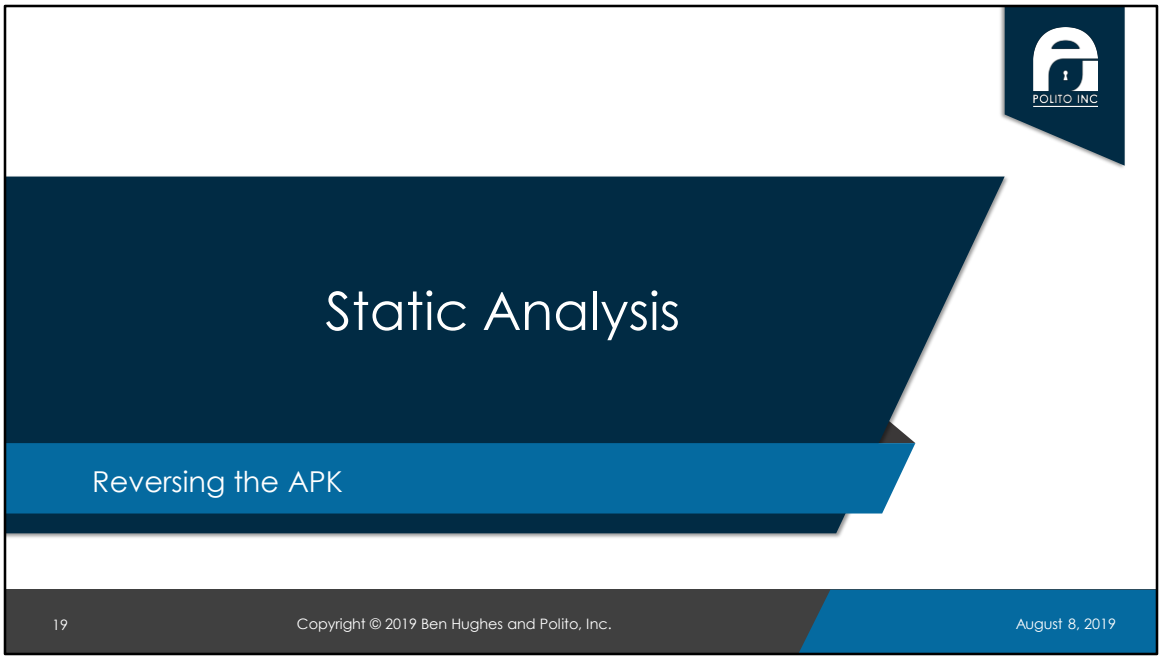




Copyright © 2018 Ben Hughes and Polito, Inc.

References:

https://ibotpeaches.github.io/Apktool/
https://github.com/pxb1988/dex2jar
https://github.com/java-decompiler/jd-gui
https://github.com/skylot/jadx
https://labs.mwrinfosecurity.com/tools/drozer/


POLITO INC
Static Analysis
Reversing the APK
19
Copyright © 2019 Ben Hughes and Polito, Inc.
August 8, 2019

Static Analysis

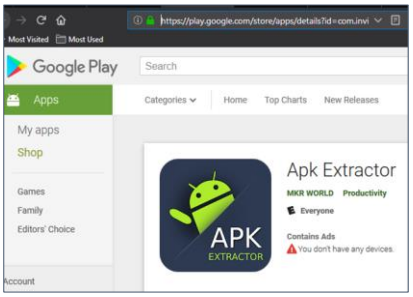


- Static analysis is conducted to review the APK contents including the source code of the mobile application
- This type of analysis can potentially identify the following common issues:
  - M1 - Improper Platform Usage
  - M2 - Insecure Data Storage
  - M3 - Insecure Communication
  - M5 - Insufficient Cryptography
  - M7 - Client Code Quality
  - M8 - Code Tampering
  - M9 - Reverse Engineering
  - M10 - Extraneous Functionality

Copyright © 2019 Ben Hughes and Polito, Inc.

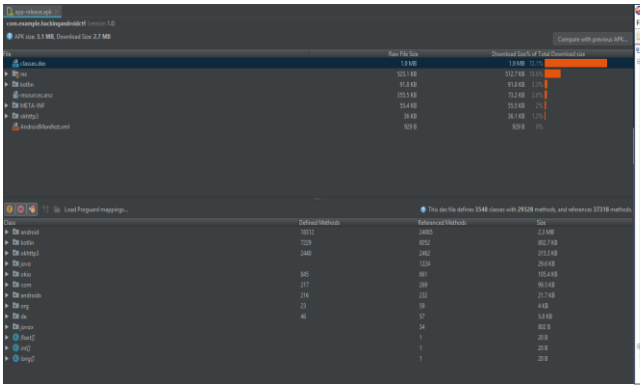
Obtaining the APK

- Install the app that you want to analyze on your rooted device or emulator
- Install APK Extractor from the Google Play Store on the same device https://play.google.com/store/apps/details?id=com.invincible.apkextractor&hl=en
  - Use the APK Extractor app to extract the APK of the target app
- ADB (Android Debug Bridge) Tool
  - Connect rooted device to laptop and use ADB shell to extract the .apk file to your computer
  - `adb devices -l`
  - `adb pull /data/app/<filename.apk> /root/APK`

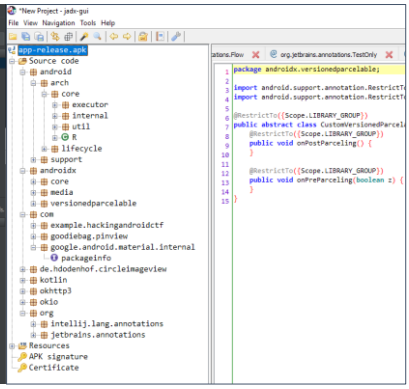


Copyright © 2019 Ben Hughes and Polito, Inc.

APK Structure and File Contents



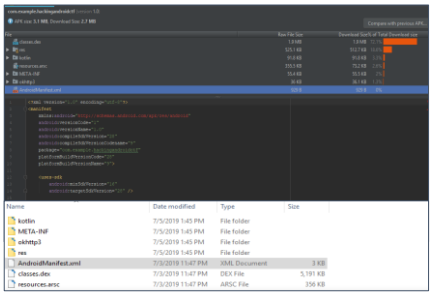
Android Studio



jadx

Copyright © 2019 Ben Hughes and Polito, Inc.

Decoding and Reserving an APK



- *Example 1:* Android Studio – For example, decoding and analyzing AndroidManifest.xml
  - Open .apk file in Android Studio to view the AndroidManifest.xml and analyze
- *Example 2:* apktool and JD-GUI
  - Decode the .apk with apktool https://ibotpeaches.github.io/Apktool
  - Convert the classes.dex into a .jar file using dex2jar https://sourceforge.net/p/dex2jar/wiki/UserGuide
  - Download JD-GUI http://java-decompiler.github.io and open the new .jar file to view the classes and code
- *Example 3:* jadx
  - https://github.com/skylot/jadx
  - Combines dex2jar and JD-GUI functionality into 1 CLI and GUI app, as it decodes/deobfuscates and then displays the APK contents including its manifest.xml and DEX files (Java classes)

Copyright © 2019 Ben Hughes and Polito, Inc.

AndroidManifest.xml



- Contains official package name, app ID, app components, permissions, etc.
- Sometimes contains hard-coded secrets or other sensitive data
- Multiple tools and methods to decode the manifest file and view cleartext contents

```
sudo java -jar axmlprinter-0.1.7.jar AndroidManifest.xml
> AndroidManifest_decoded.txt
```


Copyright © 2018 Ben Hughes and Polito, Inc.

References:
https://developer.android.com/guide/topics/manifest/manifest-intro
https://github.com/rednaga/axmlprinter

Looking for Interesting Strings



- Look for API levels, versions, app components and intents, etc.
- Permissions - any excessive or unnecessary permissions?
- Look for hard-coded API keys, passwords, certificates, URLs, databases, and other potentially sensitive data
  - `"crypt"`
  - `"https?://"`
  - `"password"`
  - `"key"`
  - `"cert"`
  - `"ssl"`
  - `"modulus"`
  - `"User-Agent"`
  - `"database"`
  - `"sqlite"`
  - `apikey`

```java
jSONObject.put("username", "guest");
jSONObject.put("password", "busyreindeer78");
jSONObject.put("type", "usage");
jSONObject.put("activity", str);
jSONObject.put("udid", Secure.getString(context.getContentResolver(), "android_id"));
new Thread(new Runnable() {
        public void run() {
            b.a(context.getString(R.string.analytics_usage_url), jSONObject);
        }
    }).start();
```


Copyright © 2019 Ben Hughes and Polito, Inc.

Reference:
https://www.holidayhackchallenge.com/2016/winners/grigorescu/

Decoding and Disassembling DEX



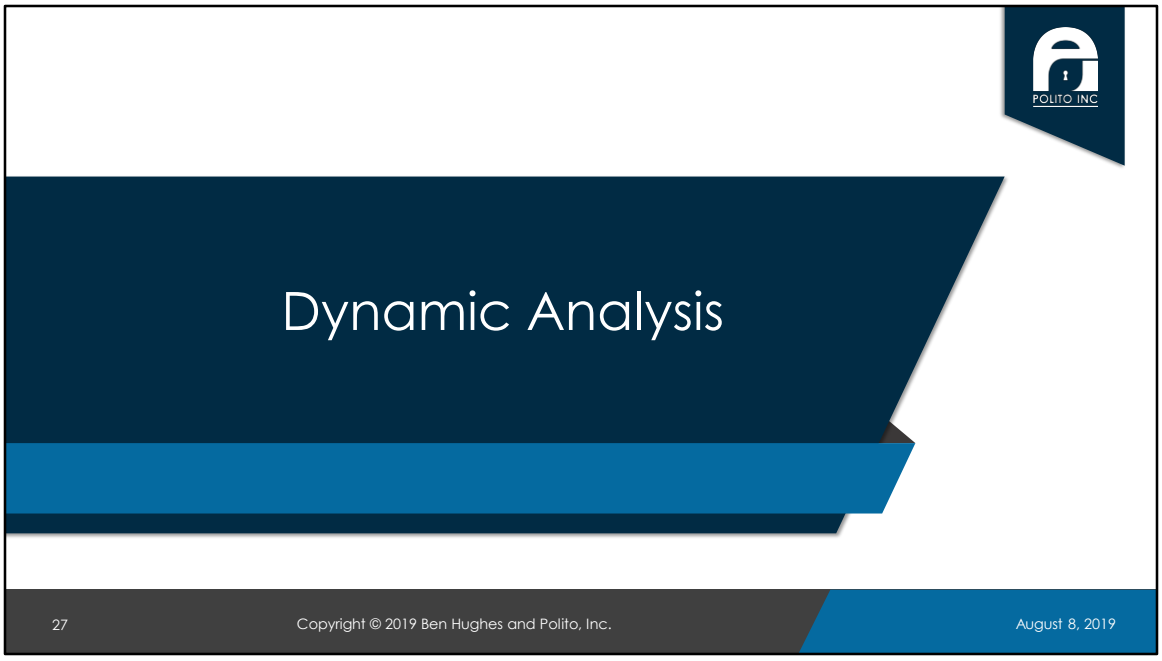
- jadx is recommended
- Start with searching for low-hanging fruit (interesting strings)
- Review interesting Java classes
- Review dependencies and 3rd party integrations
- Review for code vulnerabilities
  - Automated code scanners
  - Manual review – triage interesting/important classes
- Sometimes the code has been obfuscated…

Copyright © 2018 Ben Hughes and Polito, Inc.

Source code reversing triage techniques and tips

Disclaimer: I am not a developer ☺

References:
https://posts.specterops.io/dont-you-forget-about-re-e2c92d67c641


POLITO INC
Dynamic Analysis
27
Copyright © 2019 Ben Hughes and Polito, Inc.
August 8, 2019

Dynamic Analysis



- Dynamic analysis is conducted to review the application while it is running
- This type of analysis can potentially identify the following issues:
  - M1 - Improper Platform Usage
  - M2 - Insecure Data Storage
  - M3 - Insecure Communication
  - M4 - Insecure Authentication
  - M5 - Insufficient Cryptography
  - M6 - Insecure Authorization
  - M8 - Code Tampering

Copyright © 2019 Ben Hughes and Polito, Inc.

Conducting Dynamic Analysis



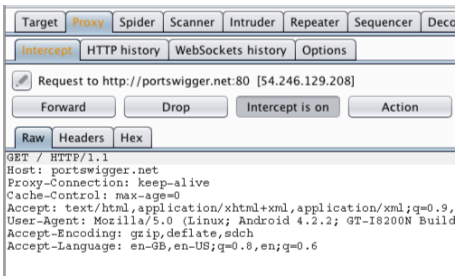
- Review the mobile application
  - Go through all the pages, buttons, features
  - Learn what the app does
  - Test unauthenticated and authenticated; create users and review the app as a logged in user (if possible)
- Execute application while device is connected to a controlled network to observe traffic
  - Proxy Options
    - Burp Suite https://portswigger.net/burp
    - Frida/Brida https://github.com/federicodotta/Brida
    - Drozer https://labs.mwrinfosecurity.com/tools/drozer
- Run Android Studio and/or adb logcat to debug and view logs while the application is running

Copyright © 2019 Ben Hughes and Polito, Inc.

Proxying Traffic with Burp Suite

POLITO INC

- Many mobile apps can be proxied as easily as a typical web app; the traffic is often very similar too
- Mobile-specific user-agents are common
- Some apps are proxy aware (bypasses exist)
- Certificate pinning may be an obstacle, especially on newer versions of Android (bypasses exist)

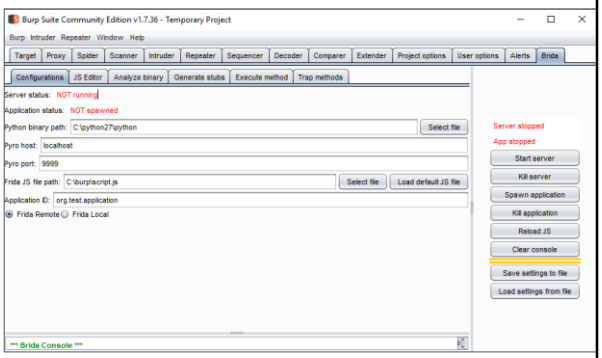


Copyright © 2018 Ben Hughes and Polito, Inc.

References:
https://support.portswigger.net/customer/portal/articles/1841102-installing-burp-s-ca-certificate-in-an-android-device
https://support.portswigger.net/customer/portal/articles/1841101-Mobile%20Set-up_Android%20Device.html
https://blog.ropnop.com/configuring-burp-suite-with-android-nougat/

Brida

- Brida - Burp Suite to Frida
  - Brida extension in Burp to proxy mobile app traffic for analysis
  - This extension works as a bridge between Burp Suite and Frida
  - Allows to view and tamper with the traffic exchanged between the applications and their back-end services/servers

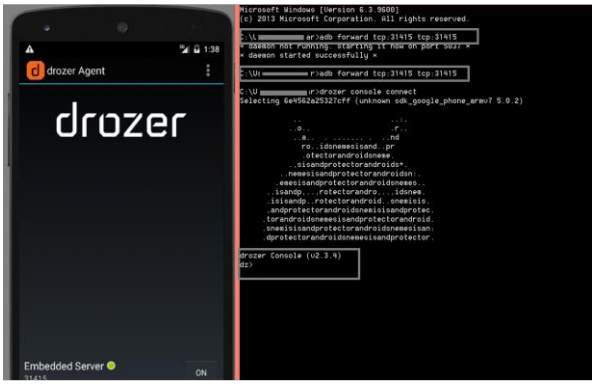


Copyright © 2019 Ben Hughes and Polito, Inc.

Reference:
https://github.com/federicodotta/Brida

Drozer

- **Drozer** - security testing framework for Android.
  - Drozer allows to test for security vulnerabilities in apps and devices by assuming the role of an app and interacting with the Dalvik VM, other apps' IPC endpoints and the underlying OS
  - Install Drozer on the analysis host, start emulator, and install agent on rooted device: **`adb install drozer.apk`**
  - Start Drozer, observe and interact with the traffic



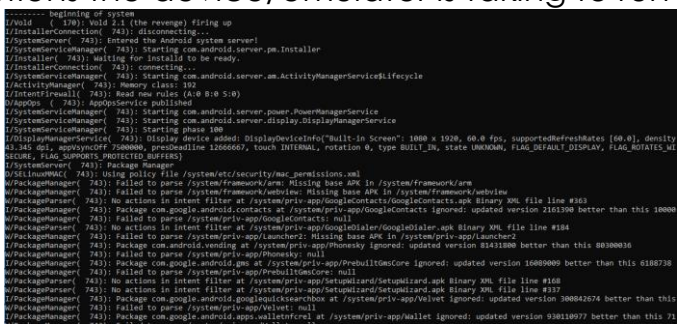
Copyright © 2019 Ben Hughes and Polito, Inc.

Reference:
https://labs.mwrinfosecurity.com/tools/drozer/

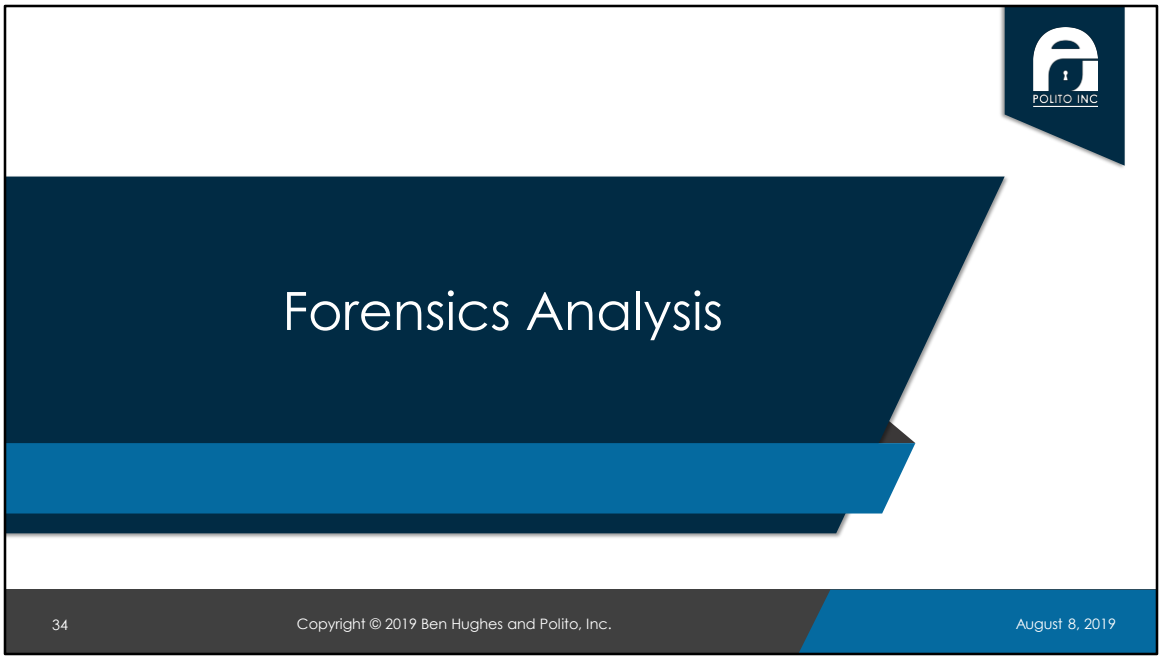
Logs and Debug



- To debug app and view logs run **`adb logcat`**
- This will show actions the device/emulator is taking to run the application



Copyright © 2019 Ben Hughes and Polito, Inc.

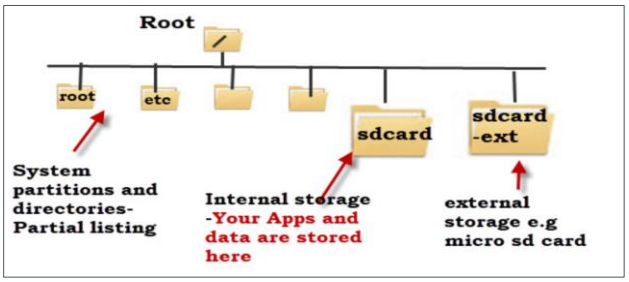


Relation to static and dynamic analysis

Forensic Analysis

Android filesystem and directory structure (brief overview)

Android uses the Linux file system structure, which has a single root.
(Image credit: http://www.stevesandroidguide.com/android-files/)



Copyright © 2019 Ben Hughes and Polito, Inc.

Rooted vs. unrooted access
Physical vs. logical image

Reference:
http://www.stevesandroidguide.com/android-files/

Rooting Android Device



Rooted vs. non-rooted devices:

On a non-rooted device, system partitions and directories are protected and cannot be directly accessed, even though some file managers will display them.

To get around these restrictions, you need to root the device.

Rooting an Android device:

- Once root exploit achieved, install SuperSU
- Install RootChecker (optional) - to verify it is rooted properly (also good for checking root status occasionally as OS updates or other changes to device can break root)

Copyright © 2019 Ben Hughes and Polito, Inc.

Rooted vs. unrooted access
Physical vs. logical image

References:

https://www.pcmag.com/article2/0,2817,2459892,00.asp

Android File System Structure



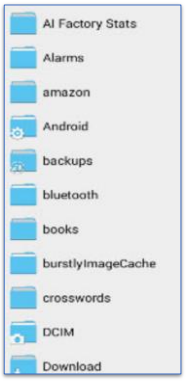
Physical disks and partitions appear under the root as a directories, but do not have a drive letter as in Windows.

Android does not always come with a default file manager, and so you will need to install a file manager app.

The **sdcard** partition is the main storage area for user data and files and it also contains app settings and data.

It gets created whether or not a physical SD card is present.

*(partial listing of **sdcard** partition- Image credits: http://www.stevesandroidguide.com/android-files/)*



Copyright © 2019 Ben Hughes and Polito, Inc.

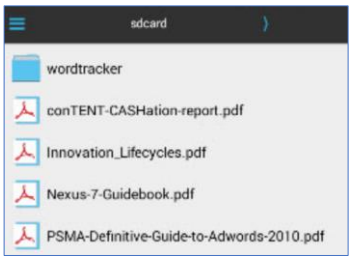
Reference:
http://www.stevesandroidguide.com/android-files/

Android FileSystem Structure

Android filesystem structure

The **ext-sdcard** partition will only be visible if your device supports external storage, usually using a microSD slot.

External sd cards use either the FAT, FAT32 or exFAT file system formats. Most devices support FAT and FAT32, but support for exFAT is limited.



Copyright © 2019 Ben Hughes and Polito, Inc.

Physical vs. Logical Acquisition



- **Physical acquisition** (or forensic image) is a bit-by-bit copy of the entire contents of the flash memory of a mobile device. This allows for the collection of all live data and also data that has been deleted or is protected/hidden. However, physical acquisition is not always possible, requiring root access to the device.
- **Logical acquisition**: Performed through forensic tools using the application API's to communicate with the mobile device's operating system and request the data from the system. This method allows for the acquisition of most of the user-accessible live data on the device, as it would be done for a live targeted collection of computer, and allows recovering live data, but not deleted or protected files.

Copyright © 2019 Ben Hughes and Polito, Inc.

Reference:
http://blog.specialcounsel.com/ediscovery/three-types-of-mobile-device-extractions-and-what-each-contains/

Filesystem Acquisition



- **Filesystem acquisition**: Unlike logical acquisitions, the forensic tool(s) can directly access the files on the mobile device's internal memory, which allows to extract all files present in the internal memory, including database files, system files and logs. Filesystem extractions allow examining file structure, web browsing history and app usage history of a mobile device and grant full access to the database files on a mobile device, and to existing and deleted files for apps such as iMessage, SMS, MMS, Calendar and others. (*Reference: http://blog.specialcounsel.com/ediscovery/three-types-of-mobile-device-extractions-and-what-each-contains/*)

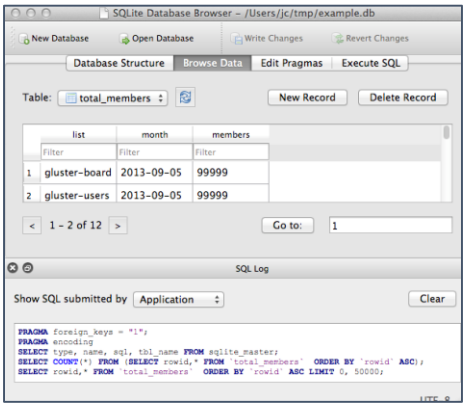
Copyright © 2019 Ben Hughes and Polito, Inc.

Reference:
http://blog.specialcounsel.com/ediscovery/three-types-of-mobile-device-extractions-and-what-each-contains/

Obtaining and Reviewing SQLite Databases

Recovering SQLite databases from a mobile app is a priority, because they could contain sensitive data.

One of the best tools for the job is DB Browser for SQLite (aka *sqlitebrowser*, available at https://sqlitebrowser.org/)



Copyright © 2019 Ben Hughes and Polito, Inc.

Reference:
https://sqlitebrowser.org/

Obtaining and Reviewing SQLite Databases

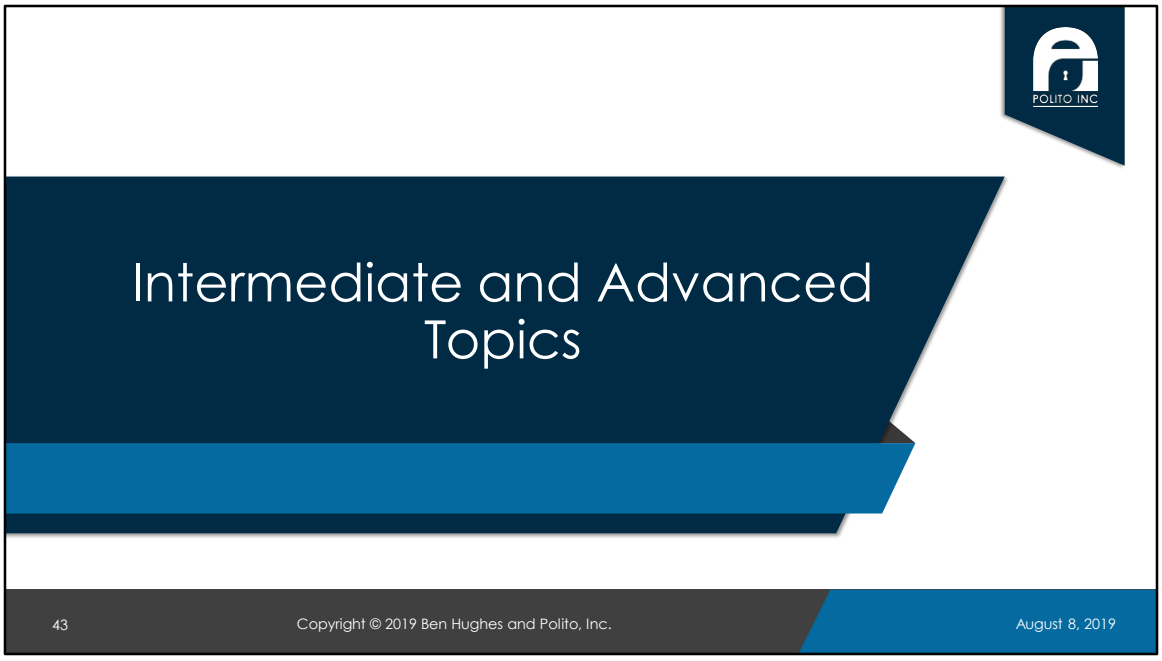


DB Browser for SQLite (DB4S) is a high quality, visual, open source tool to create, design, and edit database files compatible with SQLite.
DB4S uses a familiar spreadsheet-like interface, and complicated SQL commands do not have to be learned. It is a tool to be used by both developers and end users, and must remain as simple to use as possible in order to achieve these goals.

Alternatively, use **Android Studio, adb, or commercial mobile forensics tools such as Magnet Axiom** to grab and analyze app SQLite databases and related forensic artifacts:

Copyright © 2019 Ben Hughes and Polito, Inc.

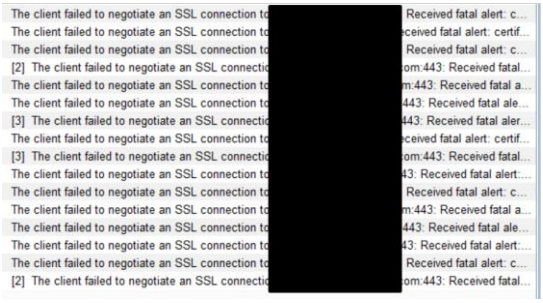
Reference:
https://sqlitebrowser.org/


POLITO INC
Intermediate and Advanced Topics
43
Copyright © 2019 Ben Hughes and Polito, Inc.
August 8, 2019

Bypassing Certificate Pinning



- Apps that leverage certificate pinning and newer versions of Android sometimes prevent proxying the mobile traffic
- There are bypasses…



Copyright © 2019 Ben Hughes and Polito, Inc.

Reference:
https://blog.ropnop.com/configuring-burp-suite-with-android-nougat/

Modifying and Re-signing APKs



- APKs are signed
- However, you can modify and re-sign an APK
- This is required for one approach to bypassing certificate pinning and related proxy issues
- Can be beneficial for other use cases too

Copyright © 2019 Ben Hughes and Polito, Inc.

Deobfuscating Source Code



- Sometimes the Java classes have been obfuscated
- Some tools like jadx have limited deobfuscation capabilities
  - Sometimes the deobfuscation attempts fail or even make things worse (i.e. the code can become more obfuscated)

Copyright © 2019 Ben Hughes and Polito, Inc.

References:
https://www.evilsocket.net/2016/04/18/how-i-defeated-an-obfuscated-and-anti-tamper-apk-with-some-python-and-a-home-made-smali-emulator/
https://posts.specterops.io/dont-you-forget-about-re-e2c92d67c641

Looking for Network and Server-side Issues

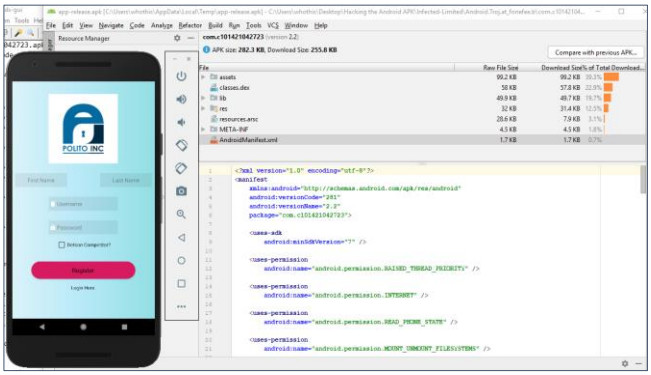


- Mobile app traffic
- Web and app servers
- APIs
- App infrastructure

- Potential pivots during a pen test (e.g. hard-coded credentials and keys FTW)

Copyright © 2019 Ben Hughes and Polito, Inc.

APK Teardown Example 1



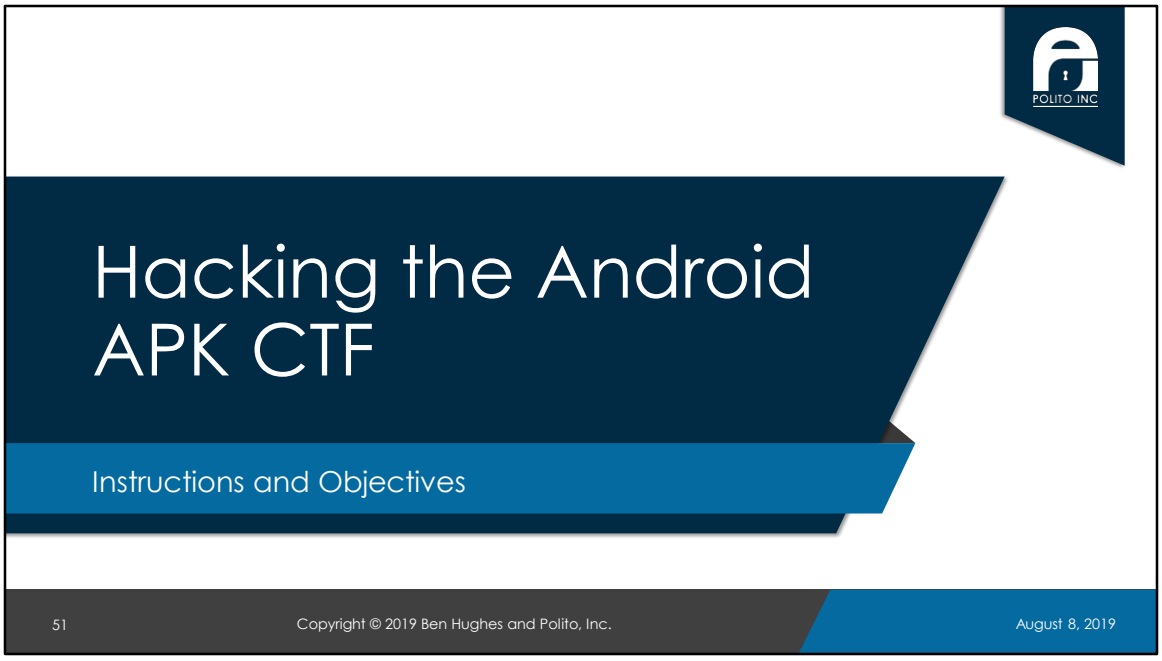
Copyright © 2019 Ben Hughes and Polito, Inc.

APK Teardown Example 2



Copyright © 2019 Ben Hughes and Polito, Inc.


POLITO INC
Questions?
Q&A Session
50
Copyright © 2019 Ben Hughes and Polito, Inc.
August 8, 2019


POLITO INC
Hacking the Android APK CTF
Instructions and Objectives
51
Copyright © 2019 Ben Hughes and Polito, Inc.
August 8, 2019

CTF Scenario



- Super secure banking app
- You are a pen tester (or hacker?) tasked with finding security weaknesses
- Any mobile app fails can likely lead to fun and profit

Copyright © 2019 Ben Hughes and Polito, Inc.

CTF Instructions



- Flags have been planted in the custom APK
- Static, dynamic, and/or forensic analysis will be required to solve the challenges and find the flags
- Challenge difficulty ranges from very easy to very hard
- Prizes for the winner(s)

Copyright © 2019 Ben Hughes and Polito, Inc.

Will provide short link to CTFd instance during workshop
Add login instructions (individual and team self-register)
Do live demo walkthrough

CTFd Server



- **https://www.politoinc.com/defcon-ctf**
- Click **Register** to create your own account
  - Provide Team Name, Email and Password
  - Can compete individually or join a team

Copyright © 2019 Ben Hughes and Polito, Inc.

Redirect will be operational during workshop

CTF Scoring Rules



- Challenges are worth 100-400 points each
  - 100 Easy
  - 200 Moderate
  - 300 Hard
  - 400+ Really hard
- No point deductions for wrong answers, but limited number of wrong submissions allowed
- -50 points for every viewed hint

Copyright © 2019 Ben Hughes and Polito, Inc.

Scoring system subject to change

CTF Prizes



- Top Team
  - Max 3 members
- Top Individual

Copyright © 2019 Ben Hughes and Polito, Inc.

Prizes likely to be rooted Android test devices

CTF – Do's and Don'ts



- Do collaborate and team up.
- Do use the hints if you get stuck and need help. (but remember hefty point penalty per hint)
- Don't intentionally interfere with the WiFi, CTFd server, app server, or other competitors. ☺

Copyright © 2019 Ben Hughes and Polito, Inc.

CTF – Ready… Set… Hack!



Copyright © 2019 Ben Hughes and Polito, Inc.

Future Workshops



- Dates and Locations TBD

Copyright © 2019 Ben Hughes and Polito, Inc.

How Did We Do?



www.politoinc.com/feedback

- We would greatly appreciate your honest feedback
- Submit your e-mail address to receive a copy of this presentation

Copyright © 2019 Ben Hughes and Polito, Inc.

References



- https://github.com/OWASP/owasp-mstg
- https://ibotpeaches.github.io/Apktool
- https://sourceforge.net/p/dex2jar/wiki/UserGuide
- https://github.com/java-decompiler/jd-gui/releases
- https://github.com/skylot/jadx
- https://github.com/federicodotta/Brida
- https://labs.mwrinfosecurity.com/tools/drozer
- https://resources.infosecinstitute.com/android-penetration-tools-walkthrough-series-drozer
- https://portswigger.net/burp
- https://conference.hitb.org/hitbsecconf2018ams/materials/D1T1%20-%20Federico%20Dotta%20and%20Piergiovanni%20Cipolloni%20-%20Brida%20When%20Burp%20Suite%20Meets%20Frida.pdf
- https://techblog.mediaservice.net/2018/04/brida-a-step-by-step-user-guide
- https://support.portswigger.net/customer/portal/articles/1841101-configuring-an-android-device-to-work-with-burp

Copyright © 2019 Ben Hughes and Polito, Inc.

References:

- https://github.com/OWASP/owasp-mstg
- https://ibotpeaches.github.io/Apktool
- https://sourceforge.net/p/dex2jar/wiki/UserGuide
- https://github.com/java-decompiler/jd-gui/releases
- https://github.com/skylot/jadx
- https://github.com/federicodotta/Brida
- https://labs.mwrinfosecurity.com/tools/drozer
- https://resources.infosecinstitute.com/android-penetration-tools-walkthrough-series-drozer
- https://portswigger.net/burp
- https://conference.hitb.org/hitbsecconf2018ams/materials/D1T1%20-%20Federico%20Dotta%20and%20Piergiovanni%20Cipolloni%20-%20Brida%20When%20Burp%20Suite%20Meets%20Frida.pdf
- https://techblog.mediaservice.net/2018/04/brida-a-step-by-step-user-guide
- https://support.portswigger.net/customer/portal/articles/1841101-configuring-an-android-device-to-work-with-burp

Thank You!



E-mail
info@politoinc.com

Website
www.politoinc.com

Blog
www.politoinc.com/blog

Twitter
info@politoinc.com

GitHub
www.github.com/politoinc

Lead Trainer
Ben Hughes
ben@politoinc.com
@CyberPraesidium



Copyright © 2019 Ben Hughes and Polito, Inc.