QARK
Who are we?

- **Penetration Testers at LinkedIn**
  - **Tony Trummer**
    - Staff Information Security Engineer
  - **Tushar Dalvi**
    - Senior Information Security Engineer
Reversing APKs

- Get Manifest
  - apktool d foo.apk

- Unzip APK
  - change apk to zip; unzip foo.zip

- Dalvik Bytecode
  - dex2jar classes.dex

- Java Bytecode
  - JD-GUI

- Java Class files
Communications

- WebViews
- AIDL
- Intents
- Binder
- Network Requests
- Deeplink URLs
What's (still) wrong with Android?

- Many sources – all the web bugs ++
- SSL/TLS fail – cert validation & plain-text HTTP
- Lots of old devices – glacial patch cycles
- Much client-side fail – no one will ever know…
What is QARK?

- Quick Android Review Kit
- Lots of stolen code
- Lots of stolen ideas
- Lots of stolen exploits
- Lots of (horribly written) Python
- A pinch of innovation
What QARK is not (yet)

Perfect

Finished

A forensics tool

A dynamic analysis tool
QARK motivation

- We’re lazy
- Our boss is lazy
- Developers are extremely lazy and ignore warnings
- I don’t like hate repeating bugs
- We have lots of apps to protect
- Lots of small dev shops (aka no security)
QARK’s mission

- Raise the bar for Android security
- Knowledge sharing
- Free SCA with validation
- Community involvement
- Motivate Google?
Under the hood

Parsing: PLYJ, BeautifulSoup, Minidom
Building: Android SDK
Decompiling: Procyon, JD Core, CFR
Code: Python (they made me do it)
Tools: adb, dex2jar, apktool
What does QARK do?

- Automates APK retrieval
- Decompresses APK
- Converts AndroidManifest.xml to text
- Parses AndroidManifest.xml
Tying it all together

- Identifies permissions issues, exported components, supported versions, etc.
- Parses Java classes
- Maps Manifest to classes
- Locates "entry point" methods
- Looks for sources of user-supplied data
What else?

Follows potentially tainted input through code

Looks for modifiers

Records any "sinks" encountered
Then what?

Combines the information gathered with manifest details for later use.

Examines WebView configurations and provides templated HTML files for validation of vulnerabilities.

Looks for vulnerabilities originating from within the app, inspecting Broadcast, Sticky and Pending Intents.
What else?

- Looks for WORLDREADABLE/WRITABLE files
- Looks for tapjacking defenses
- Looks for X.509 certificate validation issues
- Creates a “deliverable” HTML report of findings
The fun part

- Builds an APK for manual testing with a swiss-army knife style set of functionalities
- Automatically creates ADB commands to exploit discovered vulnerabilities
- Automatically builds a custom exploit APK based on its findings for point-and-click pwnage
Demo time!
Future plans

- Dynamic analysis functionality
- Contribute back to improve libraries and tools
- Handle obfuscated code
- Smali inspection
- Native code support
- Ask for your help
Where to get QARK?

LinkedIn’s GitHub

https://github.com/linkedin
Acknowledgements

- NVisium for the TapJacking code
- Rafay Blaloch, et al, for the WebView exploits
- MWR Labs for Drozer (inspiration)
- The authors and maintainers of all the opensource projects used in QARK
- Jason Haddix, Sam Bowne, et al, for supplying some vulnerable APKs to test against
Contact Info

www.secbro.com

Tony Trummer
www.linkedin.com/in/tonytrummer
@SecBro1

Tushar Dalvi
www.linkedin.com/in/tdalvi
@tushardalvi