Security & Pie

Android 9.0 & APK Security

sourcetoad

Plan of Attack

- Start at the hardware
- Work up to Android OS
- Climb into the Play Store
- Discuss Application (APK)

Google Play

sourcetoad

Connor Tumbleson

Senior Software Engineer

@Sourcetoad

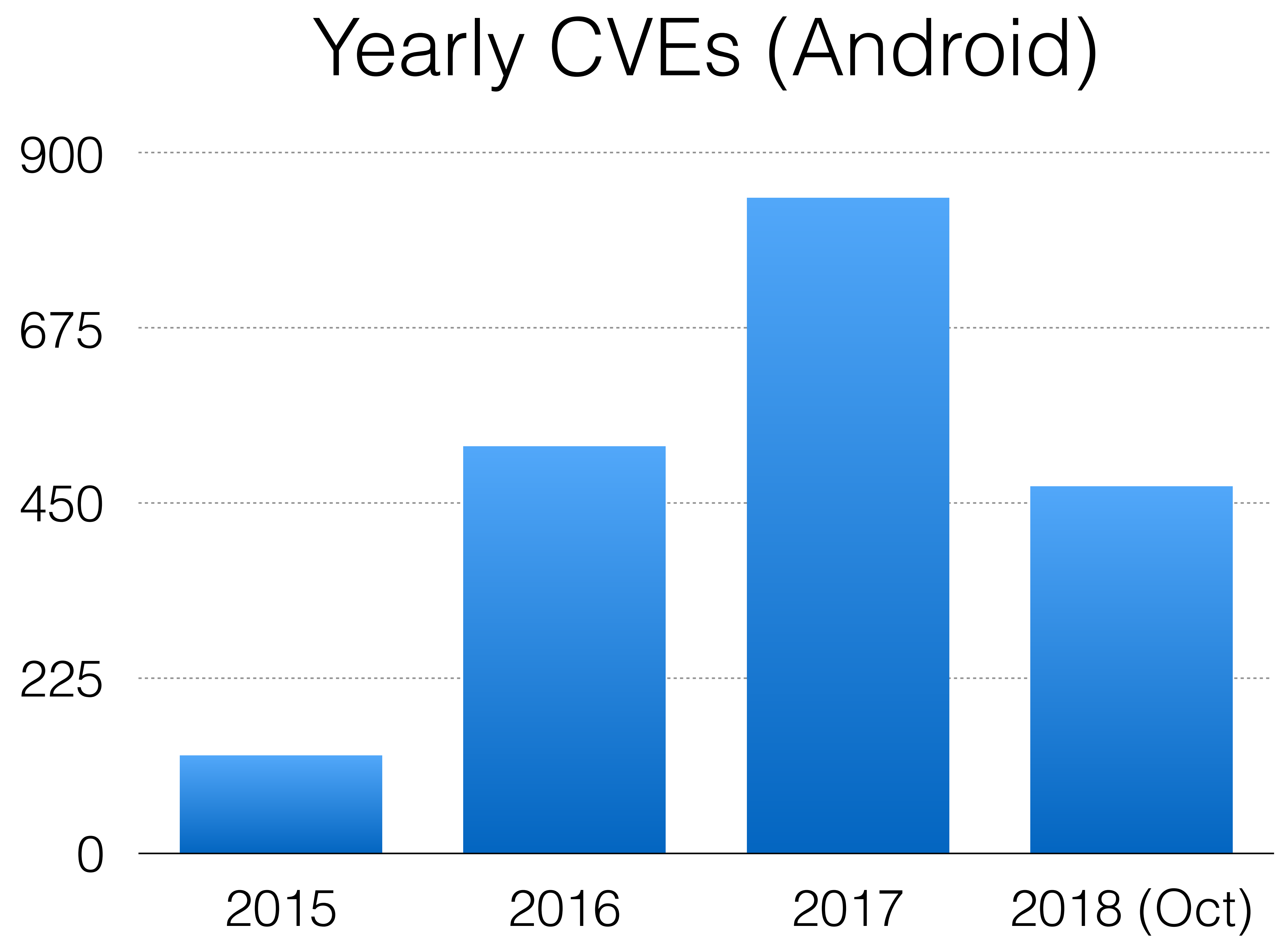
Apktool Maintainer

@iBotPeaches

connortumbleson.com

Some History

- Google I/O 2017 - **2 billion monthly** devices
- Popular target



sourcetoad

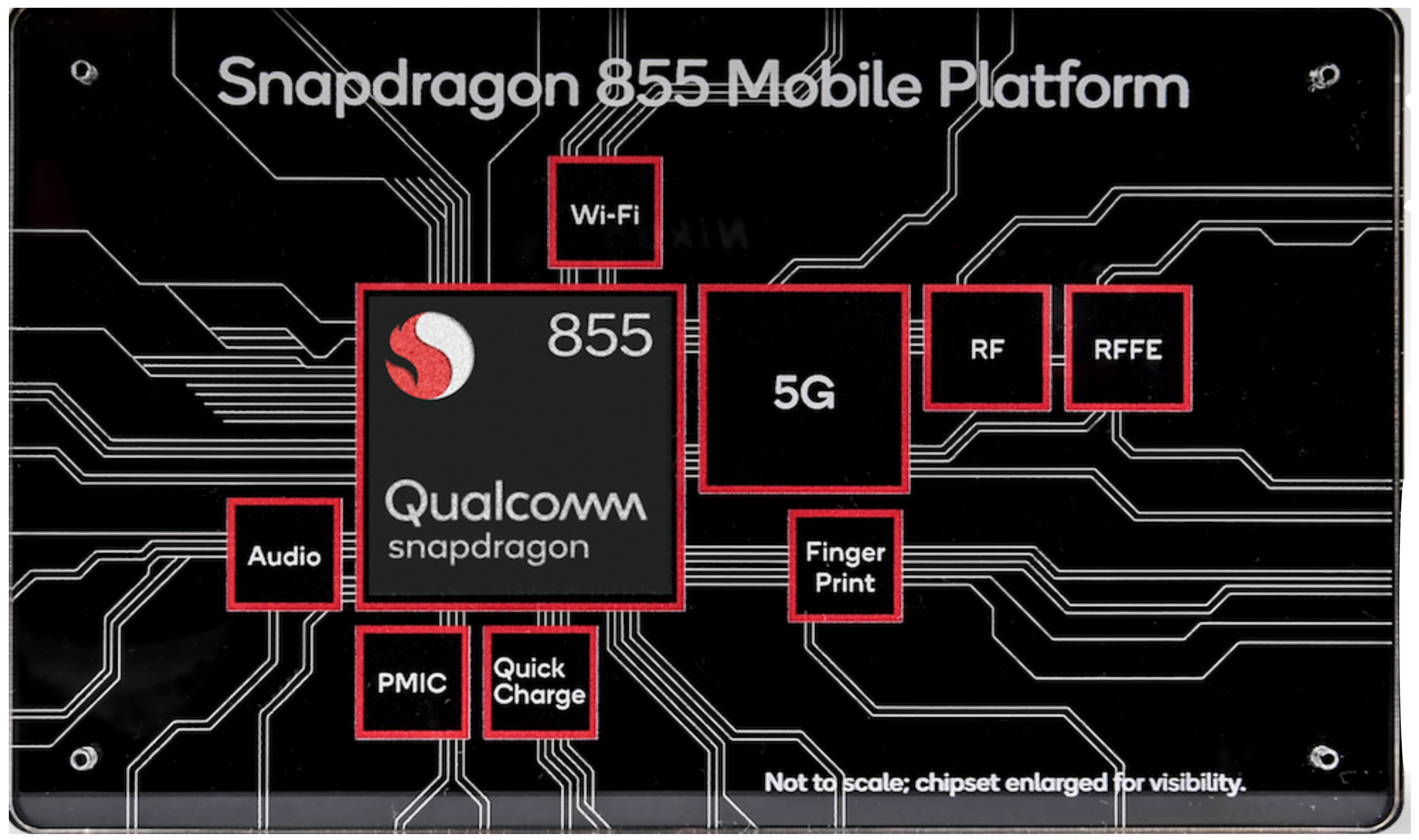
The Mobile World

- Bank applications
- PayPal / Venmo
- Medical apps
- 2 Factor Authentication
- Travel + Lodging

Hardware

Starting Line: **Hardware - SOC**

- Broadcom - BCM
- Intel - Atom
- MediaTek - MT
- NVIDIA - Tegra
- Qualcomm - Snapdragon
- Samsung - Exynos


Snapdragon 855 Mobile Platform
Wi-Fi
855
Qualcomm snapdragon
5G
RF
RFFE
Audio
Finger Print
PMIC
Quick Charge
Not to scale; chipset enlarged for visibility.

sourcetoad

Snapdragon - Qualcomm

- **SPU - S**ecure **P**rocessing **U**nit
  - Isolated RAM/CPU/Power
  - Vault-like
- **TEE - T**rusted **E**xecution **E**nvironment
  - **HLOS** - **H**igh **L**evel **O**perating **S**ystem
  - Trusted execution of code

Android

Android Platform

- Encryption
- Kernel
- Sandboxing
- SELinux
- Userspace
- Boot

bit.ly/2SJI5xk

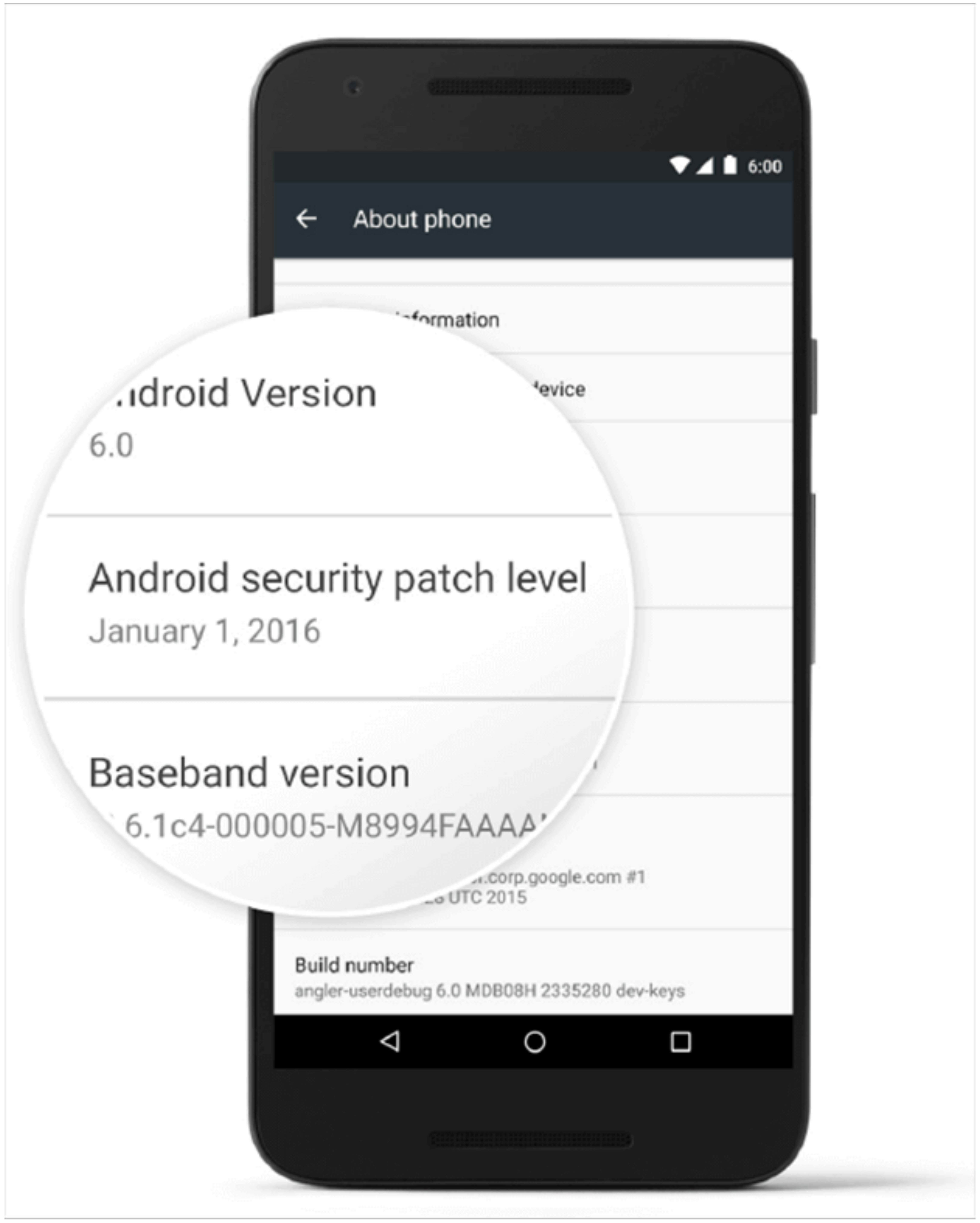
Android Security
2017 Year In Review

March 2018

android

Android Platform

- Monthly updates
- Security Patch level
  - Easier to follow
- OEMs follow
  - Or try too…



About phone

Android Version
6.0

Android security patch level
January 1, 2016

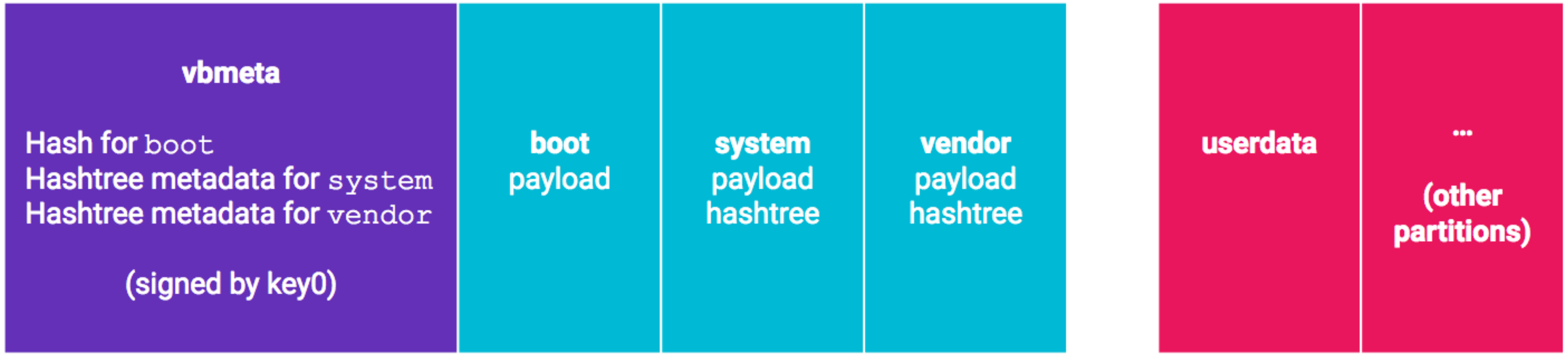
Baseband version
6.1c4-000005-M8994FAAAA

.corp.google.com #1
UTC 2015

Build number
angler-userdebug 6.0 MDB08H 2335280 dev-keys

Android Boot

- **AVB - A**ndroid **V**erified **B**oot
- Integrity of software during boot



bit.ly/2rBWm3E

sourcetoad

Android Userspace - Before ASLR

- Take some memory
- We want the **secrets**
- Overflow
- Goal to take from 0x2
- Retry. Retry. Retry.
- Profit.

0x1 - memory
0x2 - secrets
0x3 - memory
0x4 - app
0x5 - memory
0x6 - memory
0x7 - memory

sourcetoad

Android **ASLR**

- **A**ddress
- **S**pace
- **L**ayout
- **R**andomization

??? - memory

??? - memory

??? - app

??? - memory

??? - memory

??? - secrets

??? - memory

Android **ASLR** Example

```
6704d000-67144000 r-xp 00000000 b3:17 465        /system/lib/libstagefright.so
67144000-67145000 ---p 00000000 00:00 0
67145000-6714b000 r--p 000f7000 b3:17 465        /system/lib/libstagefright.so
6714b000-6714c000 rwxp 000fd000 b3:17 465        /system/lib/libstagefright.so
6714c000-6714d000 rw-p 000fe000 b3:17 465        /system/lib/libstagefright.so
6714d000-67161000 r-xp 00000000 b3:17 287        /system/lib/libdrmframework.so
67161000-67164000 r--p 00013000 b3:17 287        /system/lib/libdrmframework.so
67164000-67167000 rw-p 00000000 00:00 0
671b0000-671b2000 r-xp 00000000 b3:17 487        /system/lib/libstagefright_yuv.so
671b2000-671b3000 r--p 00001000 b3:17 487        /system/lib/libstagefright_yuv.so
```

Android **ASLR** Example

```
670b0000-671a7000 r-xp 00000000 b3:17 465        /system/lib/libstagefright.so
671a7000-671a8000 ---p 00000000 00:00 0
671a8000-671ae000 r--p 000f7000 b3:17 465        /system/lib/libstagefright.so
671ae000-671af000 rwxp 000fd000 b3:17 465        /system/lib/libstagefright.so
671af000-671b0000 rw-p 000fe000 b3:17 465        /system/lib/libstagefright.so
671b0000-671c4000 r-xp 00000000 b3:17 287        /system/lib/libdrmframework.so
671c4000-671c7000 r--p 00013000 b3:17 287        /system/lib/libdrmframework.so
671c7000-671c9000 rw-p 00000000 00:00 0
67216000-67228000 r-xp 00000000 b3:17 470        /system/lib/libstagefright_omx.so
67228000-67229000 ---p 00000000 00:00 0
67229000-6722a000 r--p 00012000 b3:17 470        /system/lib/libstagefright_omx.so
6722a000-6722b000 rwxp 00013000 b3:17 470        /system/lib/libstagefright_omx.so
```

Android **ASLR + DEP**

- **DEP - D**ata **E**xecution **P**revention
- In short - Prevents stack execution
- **ASLR** randomizes a lot.
  - Stack, Heap, Libs, Linker, Execs, etc

Android **SELinux**

- **S**ecurity-**E**nhanced
- 20+ years old
- Created by NSA
- Separation of information
- Constantly upgraded

Android **SELinux - History**

- 4.3 - Permissive “Warn, don’t block”
- 4.4 - Partially Enforced
- 5.0 - Fully Enforced
- 6.0 - Isolation between users
- 7.0 - Mediaserver
- 8.0 - Support with Treble

Android 9.0 **SELinux**

- **Per App Sandbox :)**
  - Non-Privileged Apps run in individual containers
  - No more leaking data, if >= API 28
  - Share data via Content Providers

Devs do this!

sourcetoad

Android **Encryption**

- **Full Disk** based (4.4 - *Deprecated*)
  - Entire disk with one key.
- **File** based (7.0)
  - File based with different keys
- **Metadata** based (9.0)
  - Everything else with single key

Android 9.0 - **Metadata Encryption**

- What is everything else?
  - Directory Layouts
  - File sizes, permissions, creation time
- Key protected in Keymaster which is protected with **A**ndroid **V**erified **B**oot

sourcetoad

Hold up. What is **Keymaster?**

- Trusted environment for secrets.
- v1 - Access Controls for keys
- v2 - Version Binding
- v3 - ID Attestation (Serial, Name, IMEI)
- v4 - Strongbox (?)

Android 9.0 - **Strongbox**

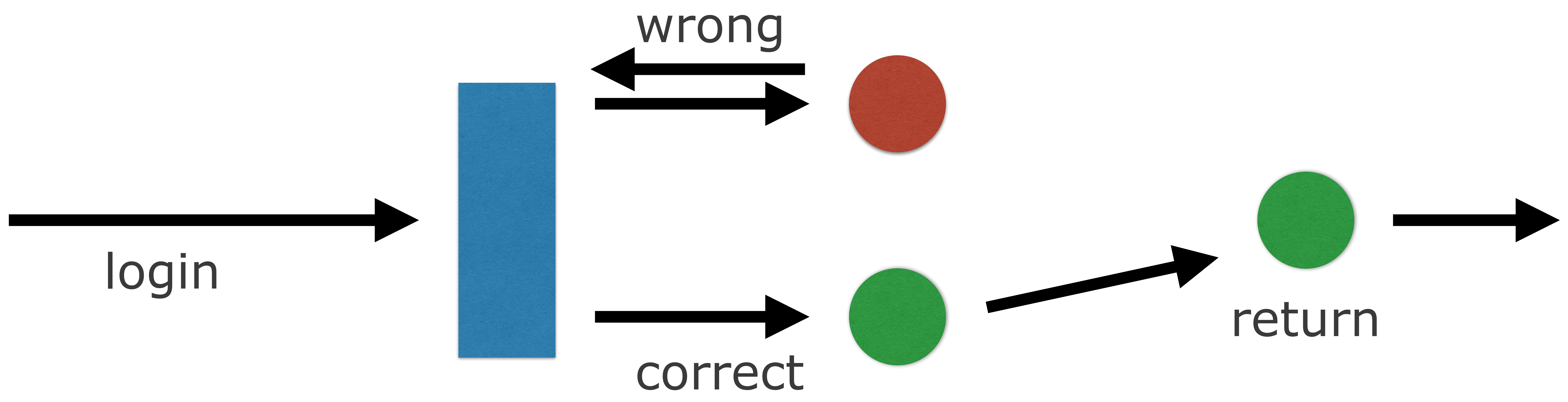
- Physical separate CPU
- Secure Storage
- True Random
- Tamper resistant
- Side channel protection

Android 9.0 - **CFI**

- **C**ontrol **F**low **I**ntegrity
- *As of 2016, 86% of vulnerabilities on Android are memory safety related.*
- So what is it?

bit.ly/2CkX4IP

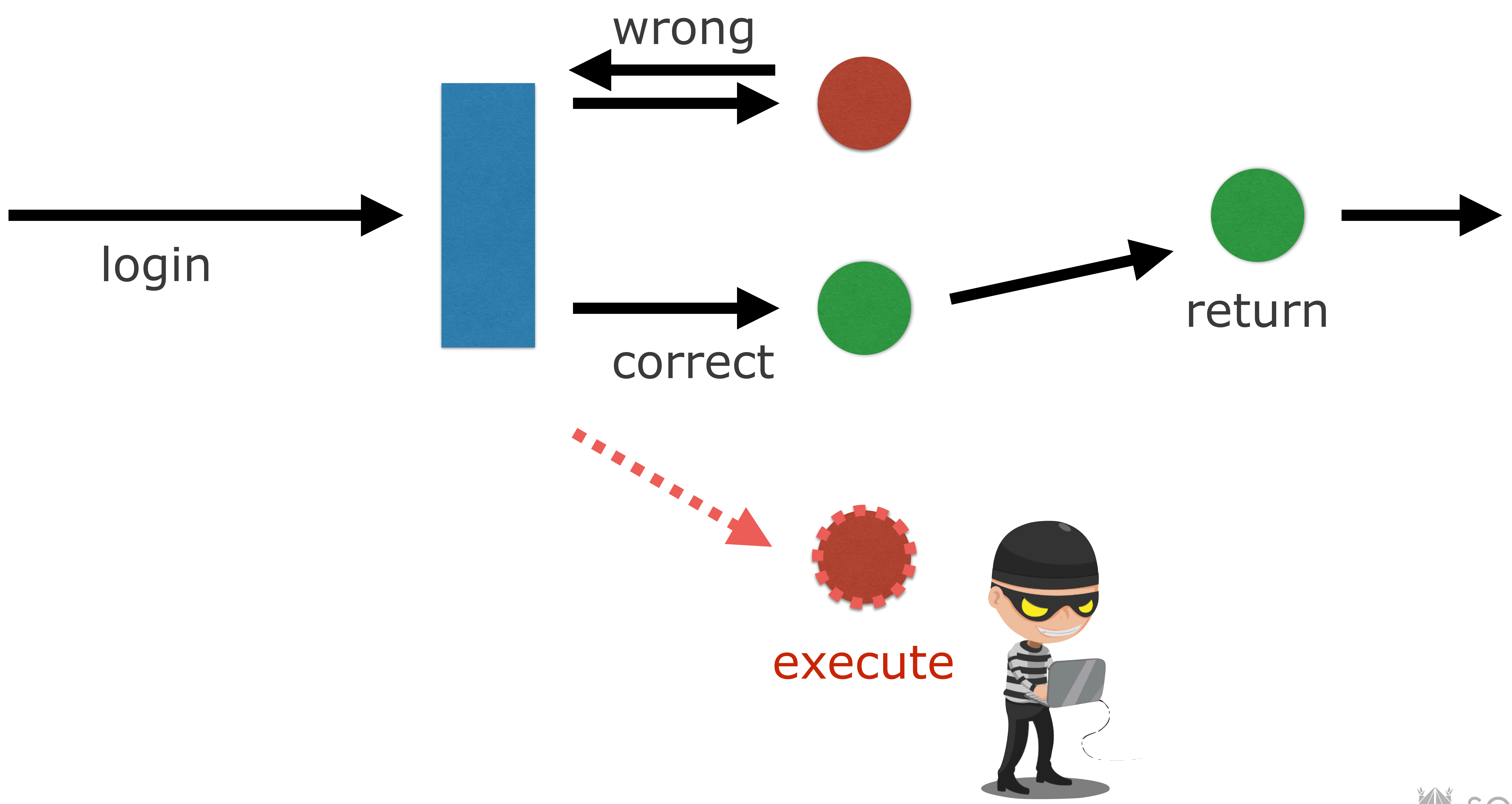
CFI - **Example Program**



- Basic program
- Fail login, must retry.
- If successful, move onward.

sourcetoad

CFI - **Example Program (Attacker)**



sourcetoad

CFI - **Example Program (Attacker)**

wrong
login
return
correct
execute

- **CFI** knows

sourcetoad

Android 9.0 - **CFI**

- Disallows changes to original control flow
- 9.0 - Enabled in components & kernel
- Requires **L**ink-**T**ime **O**ptimization
- Tough with shared libraries

sourcetoad

Android Platform - **Conclusion**

- Protection of Data
- Strong storage
- Self Protection (Kernel)
- Enforcement (SELinux)
- Verified Boot

Google PlayStore

PlayStore - Lets talk **PHA**

- **P**otentially **H**armful **A**pplication
- Google Play Protect
  - Finds lost devices
  - Blocks deceptive websites
  - Detects and removes **PHA**s

So what is a **PHA**?

- **Nothing good.**
- Fraud
- Phishing
- Trojan
- Spyware
- Ransomware

Known **PHAs** (2017 Report)

- **Chamois** - sms fraud + botnet
- **IcicleGum** - spyware
- **BreadSMS** - sms fraud
- **JamSkunk** - toll fraud
- **ExpensiveWall** - sms fraud
- **BambaPurple** - toll fraud + ads

PHA - Chamois

- Largest PHA to date.
- Multiple stages
- Features
  - Generating invalid traffic (ads)
  - Automatic app installs
  - SMS fraud (premium texts)

bit.ly/2Cs57U1

SafetyNet

Google's **SafetyNet** Overview

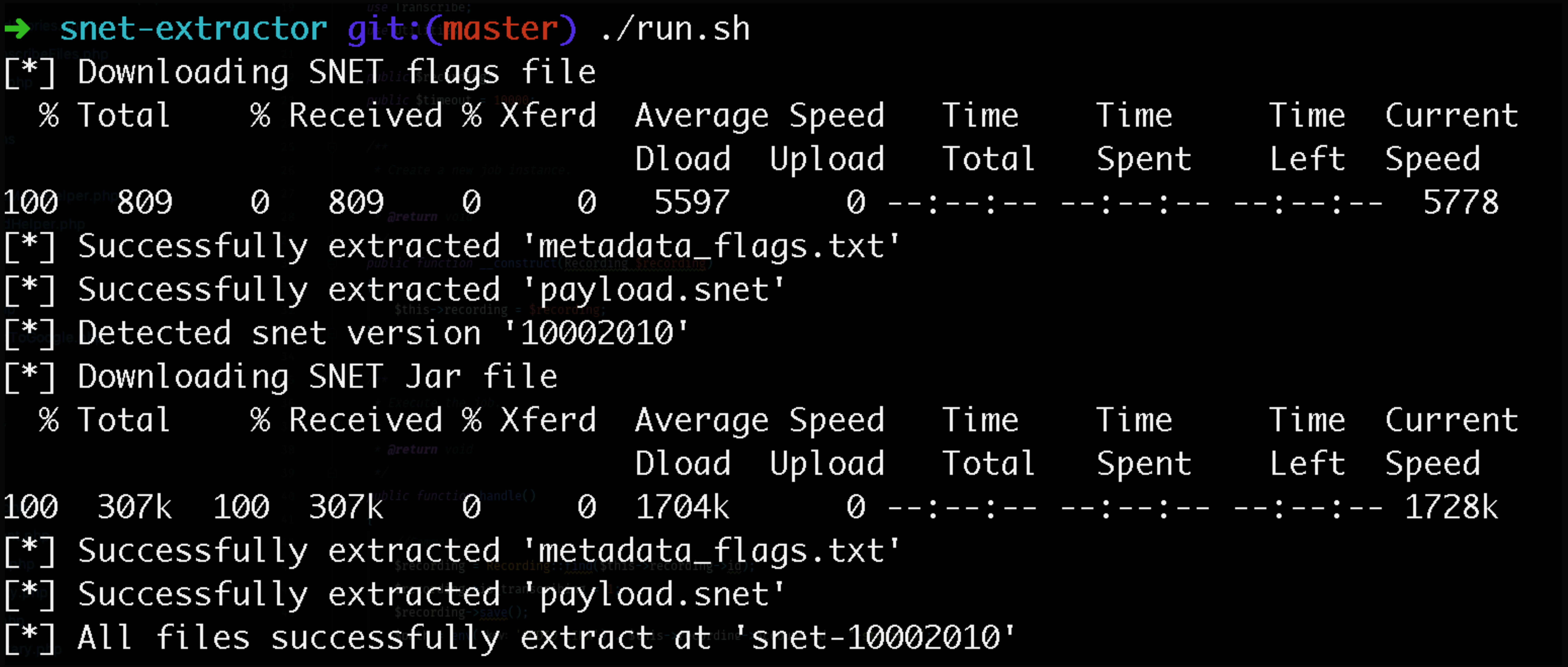
- Marketed as…
  - Verify Apps API
  - Google Play Protect
- The brains: **SafetyNet**
  - Features: always changing

sourcetoad

SafetyNet Internals

- Thanks to **@ikoz** (John Kozyrakis)
- Researches SafetyNet for years
- koz.io <— plenty of blogs about it
- First we need to get the binary.

SafetyNet Download (Research)



```
➜  snet-extractor git:(master) ./run.sh
[*] Downloading SNET flags file
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   809    0   809    0     0   5597      0 --:--:-- --:--:-- --:--:--  5778
[*] Successfully extracted 'metadata_flags.txt'
[*] Successfully extracted 'payload.snet'
[*] Detected snet version '10002010'
[*] Downloading SNET Jar file
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  307k  100  307k    0     0  1704k      0 --:--:-- --:--:-- --:--:-- 1728k
[*] Successfully extracted 'metadata_flags.txt'
[*] Successfully extracted 'payload.snet'
[*] All files successfully extract at 'snet-10002010'
```

bit.ly/2CrO98i

SafetyNet Explained

- Runs under **G**oogle **M**obile **S**ervices
- Google involved for **M**achine **L**earning
- Updates outside of OEM
- Complex
- Module based

SafetyNet Modules

- default_packages
- su_files
- settings
- locale
- ssl_handshake
- sslv3_fallback
- proxy
- setuid_files
- selinux_status
- apps
- logcat
- attest

SafetyNet Modules…

- system_ca_cert
- gmscore
- event_log
- device_state
- mount_options
- app_dir_wr
- phone sky
- internal_logs
- app_ops
- snet_network
- snet_verify_apps
- and more…

sourcetoad

SafetyNet - So what are those?

- **su_files** - Checks for SU binaries
- **ssl_handshake** - Detects MITM
- **mx_record** - Detects spoofed DNS
- **google_page_info** - Detects JS injection
- **proxy** - Detects known bad locations

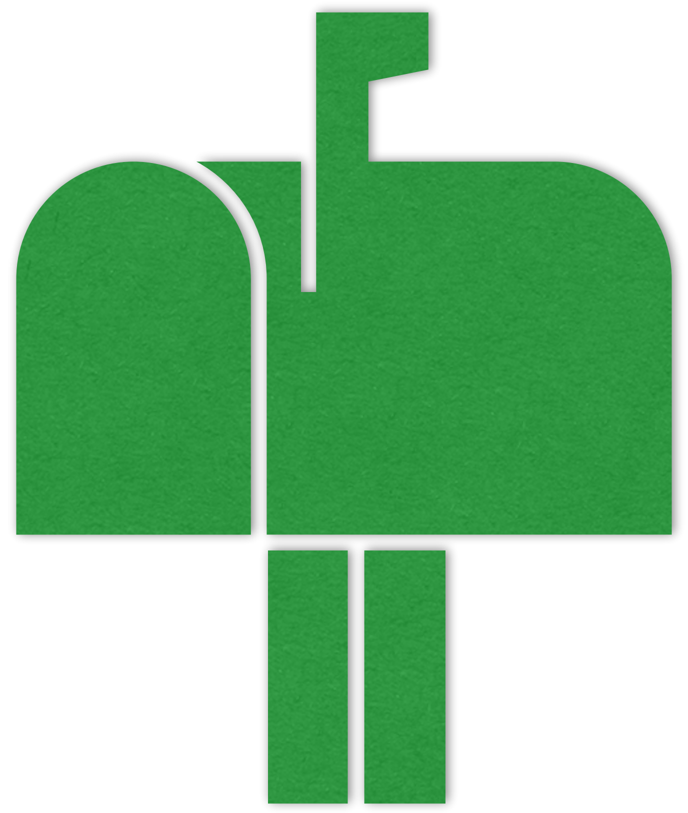
SafetyNet - **DroidGuard**

- Secret Weapon - DroidGuard
- Native blob of magic
  - Tough to RE
  - Growing with features
  - Anti-malware
- Not talked about a lot. Quite hidden

Applications (APKs)

APK Basics

- Think ZIP file.
- Collection of resources and source
- Assets, libraries, etc
- One big package isolated for each app.



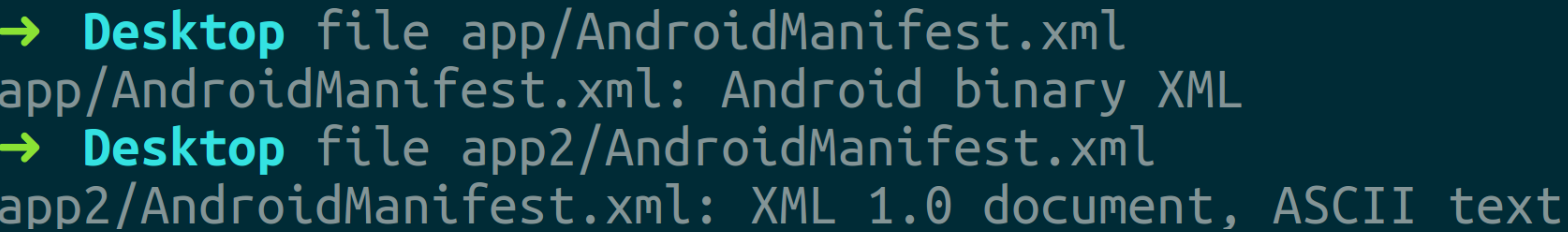
APK Basics - Just **unzip** it!

```
➜  app ls -ls
total 4284
   4 -rw-rw-rw-   1 ibotpeaches ibotpeaches    2292 Dec 31  1979 AndroidManifest.xml
4000 -rw-rw-r--   1 ibotpeaches ibotpeaches 4093496 Dec 31  1979 classes.dex
   4 drwxrwxr-x   7 ibotpeaches ibotpeaches    4096 Dec 27 07:11 kotlin
   4 drwxrwxr-x   2 ibotpeaches ibotpeaches    4096 Dec 27 07:11 META-INF
   4 drwxrwxr-x  41 ibotpeaches ibotpeaches    4096 Dec 27 07:11 res
 268 -rw-rw-rw-   1 ibotpeaches ibotpeaches  271380 Dec 31  1979 resources.arsc
```

APK Basics - or **Apktool** it!

```
→  app2 ls -la
total 36
drwxrwxr-x   6 ibotpeaches ibotpeaches 4096 Dec 27 07:12 .
drwxr-xr-x  17 ibotpeaches ibotpeaches 4096 Dec 27 07:12 ..
-rw-rw-r--   1 ibotpeaches ibotpeaches  971 Dec 27 07:12 AndroidManifest.xml
-rw-rw-r--   1 ibotpeaches ibotpeaches 7108 Dec 27 07:12 apktool.yml
drwxrwxr-x   7 ibotpeaches ibotpeaches 4096 Dec 27 07:12 kotlin
drwxrwxr-x   3 ibotpeaches ibotpeaches 4096 Dec 27 07:12 original
drwxrwxr-x 148 ibotpeaches ibotpeaches 4096 Dec 27 07:12 res
drwxrwxr-x   6 ibotpeaches ibotpeaches 4096 Dec 27 07:12 smali
```

AXML vs XML



```
→  Desktop file app/AndroidManifest.xml
app/AndroidManifest.xml: Android binary XML
→  Desktop file app2/AndroidManifest.xml
app2/AndroidManifest.xml: XML 1.0 document, ASCII text
```

Apktool - Reverse Engineering APKs

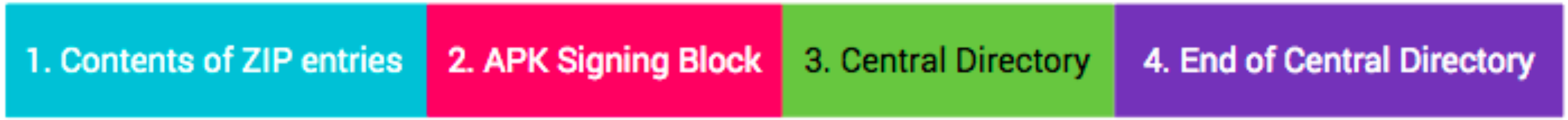
- Open source. Free.
- Decodes AXML, 9patch and dex files.
- Thanks to smali project

APK Internals

- **.dex** - source files (Java)
- **.arsc** - resources (strings, layouts, themes)
- **libs** - native libraries
- **res** - images, raw, xml, etc
- and more.

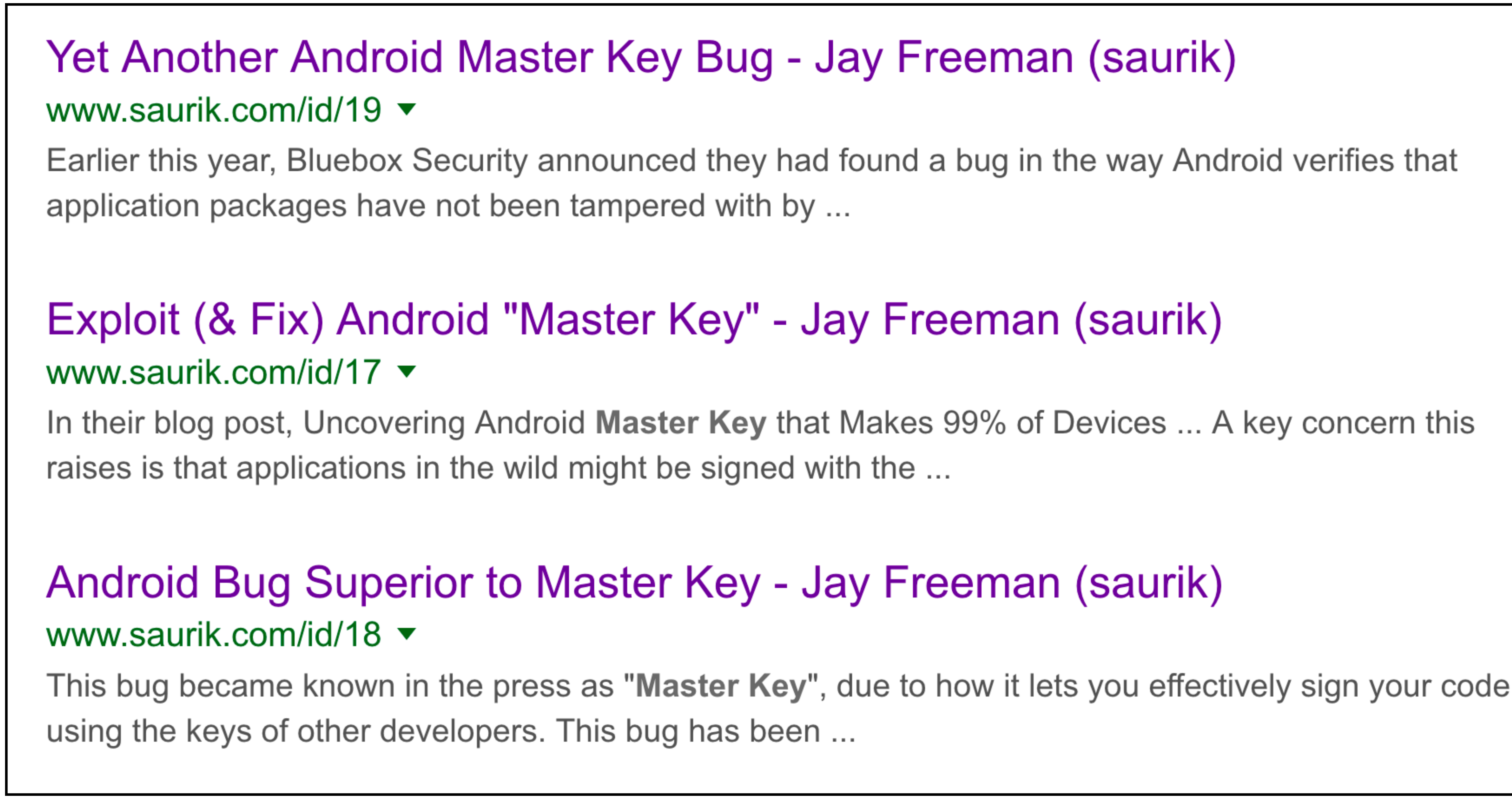
APK Signatures

- 1.0 - JAR Signature
- ??? (security fixes)
- 7.0 - APK Signature Block v2
- 9.0 - APK Signature Block v3



APK **"Master Key"** Woes

- APKs unzipped on Android
- Bug after bug
- Led to v2



Yet Another Android Master Key Bug - Jay Freeman (saurik)
www.saurik.com/id/19 ▾
Earlier this year, Bluebox Security announced they had found a bug in the way Android verifies that application packages have not been tampered with by ...

Exploit (& Fix) Android "Master Key" - Jay Freeman (saurik)
www.saurik.com/id/17 ▾
In their blog post, Uncovering Android **Master Key** that Makes 99% of Devices ... A key concern this raises is that applications in the wild might be signed with the ...

Android Bug Superior to Master Key - Jay Freeman (saurik)
www.saurik.com/id/18 ▾
This bug became known in the press as "**Master Key**", due to how it lets you effectively sign your code using the keys of other developers. This bug has been ...

Android 9.0 - **v3 Signature**

- Key Rotation
- Update keys as part of APK update
- Think company acquiring app
- Minor, big change was v2

In **Closing**

- Take those monthly updates
- Stay within the Play Store
- Leave those slow OEMs behind

Thanks!

sourcetoad

@iBotPeaches

connortumbleson.com