



# Mobile Security: Android

Saurabh Kumar  
Senior Research Scholar  
IIT Kanpur  
Date: 09/03/2022



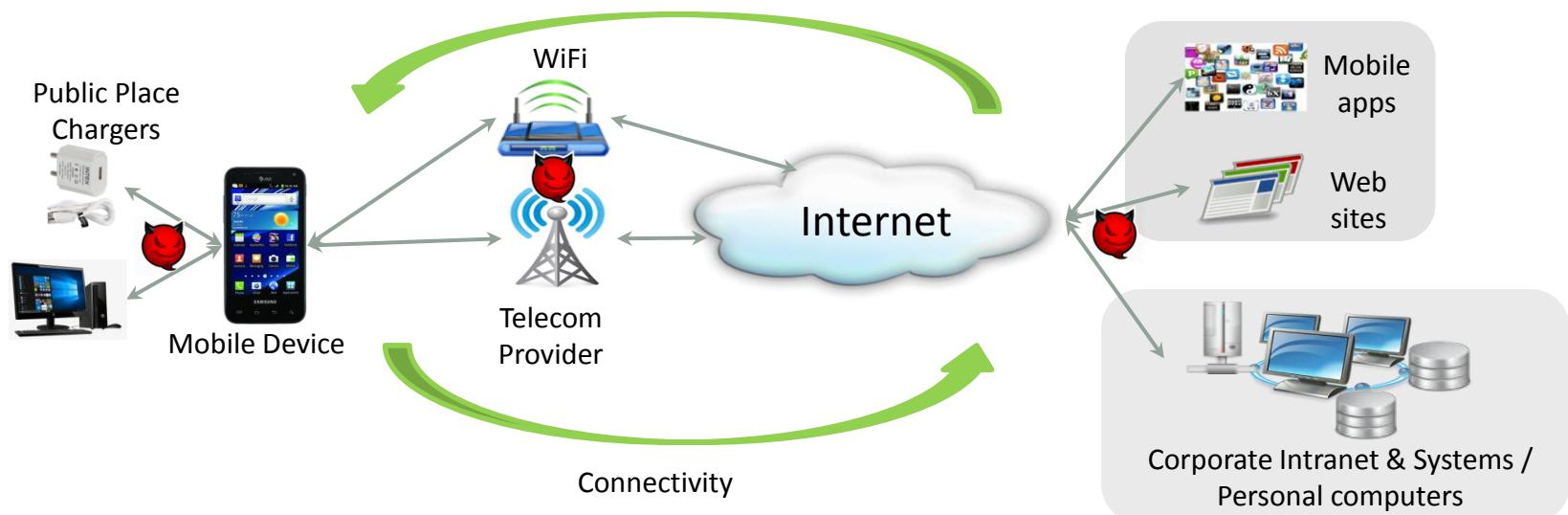
<https://github.com/skmtr1/Workshop-Mobile-Forensics-And-Security>

# MOTIVATION

---

# Why Mobile Security?

- User activity
- Valuable data
- Always on
- Multiple Attack Surfaces



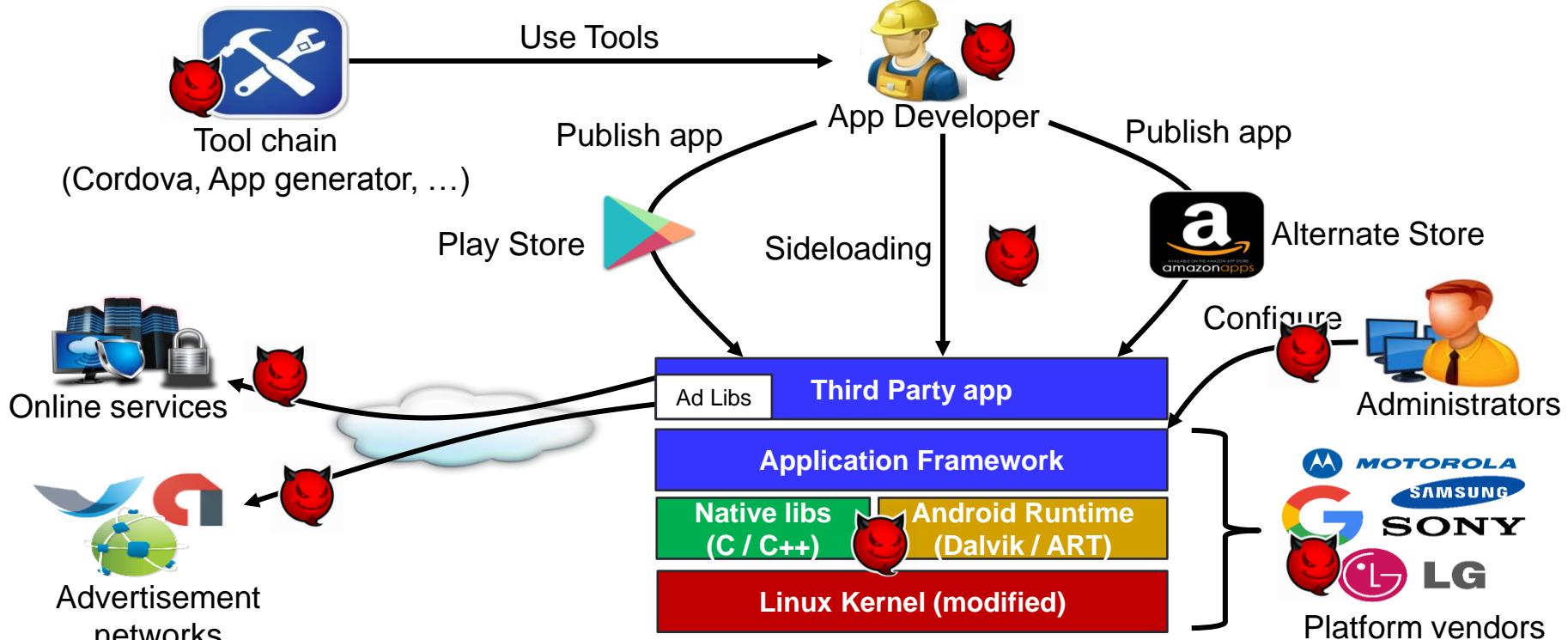
# Why Android?

1. Almost completely open source
2. Global smartphone market share

| Period | Android | iOS   | Others |
|--------|---------|-------|--------|
| 2020   | 84.1%   | 15.9% | 0%     |
| 2021   | 83.8%   | 16.2% | 0%     |
| 2022   | 84.1%   | 15.9% | 0%     |
| 2023   | 84.4%   | 15.6% | 0%     |
| 2024   | 84.7%   | 15.3% | 0%     |
| 2025   | 84.9%   | 15.1% | 0%     |

Source: International Data Corporation (IDC), October 2021

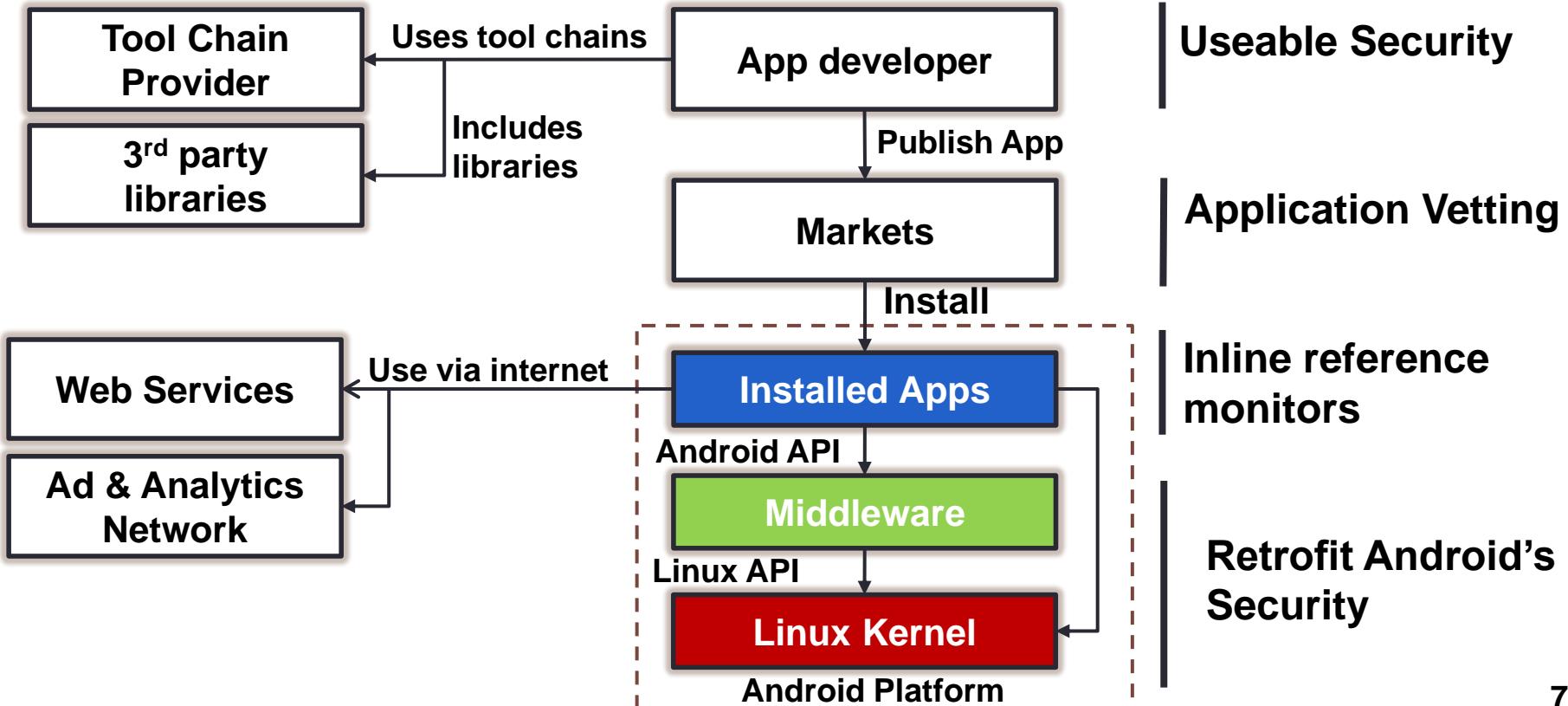
# Actors in the Android Ecosystem



# Security Impact of an Actor Over Others

| Actor               | OS Developer | H/W Vendor | Library Providers | S/W Developer | Toolchain Providers | S/W Publisher | S/W Market | End User |
|---------------------|--------------|------------|-------------------|---------------|---------------------|---------------|------------|----------|
| OS Developer        | --           | Partial    | Full              | Full          | Partial             | Full          | Full       | Full     |
| H/W Vendor          | None         | --         | Full              | Full          | None                | None          | None       | Full     |
| Library Provider    | None         | None       | --                | Full          | None                | None          | None       | Full     |
| S/W Developer       | None         | None       | Partial           | --            | None                | None          | None       | Full     |
| Toolchain Providers | None         | None       | None              | Full          | --                  | None          | None       | Partial  |
| S/W Publisher       | None         | None       | Partial           | Partial       | None                | --            | Partial    | Full     |
| S/W Market          | None         | None       | Partial           | Partial       | None                | None          | --         | Full     |
| End User            | None         | None       | None              | None          | None                | None          | None       | --       |

# Where to Improve Security?



# MOTIVATION: SUMMARY

- ❑ Feature-rich smartphones and appification have induced security research on various new aspects
- ❑ Android's market share has made Android the #1 target for malware authors and makes improved security & privacy mechanisms imperative
- ❑ Various actors in the ecosystem with (strong) influence on security and privacy

# ANDROID BACKGROUND

---

# Android Software Stack

**Default apps**

Contacts

SMS

**Third party apps**

paytm

linkedin

**Application Framework**

**Native libs  
(C / C++)**

**Android Runtime  
(Dalvik / ART)**

**Linux Kernel (modified)**

# Application Packages (APK)

- ❑ APK is simply a packaging format like **JAR**, **ZIP** and **TAR**
- ❑ Component of Application

- Activity
- Content Provider
- Services
- Broadcast Receiver

- ❑ Native Code (C/C++ shared libraries)
- ❑ Resources
- ❑ META-INF
- ❑ Application Manifest



# ANDROID SECURITY ARCHITECTURE

---

- Package Integrity
- Sandboxing
- Permission and Least Privilege

# Package Integrity: Package Manifest

- ❑ Created with **jarsigner**
- ❑ META-INF
  - Manifest.mf, Cert.sf, Cert.{RSA,DSA}

File

Manifest.mf

```
Manifest-Version: 1.0
Built-By: Generated-by-ADT
Created-By: Android Gradle 3.0.1

Name: res/mipmap-hdpi-v4/ic_launcher.png
SHA1-Digest: 2zklQdtvIXqEHSTVOVuwBQ18als=
```

hash

Cert.sf

```
Signature-Version: 1.0
Created-By: 1.0 (Android)
SHA1-Digest-Manifest:
h9xNIIN3bQiTJ8RQyPUWBojRKD8=
X-Android-APK-Signed: 2

Name: res/mipmap-hdpi-v4/ic_launcher.png
SHA1-Digest: L8RpX5x8pChJbucqml+hMt9D9CQ=
```



hash

ic\_launcher.png

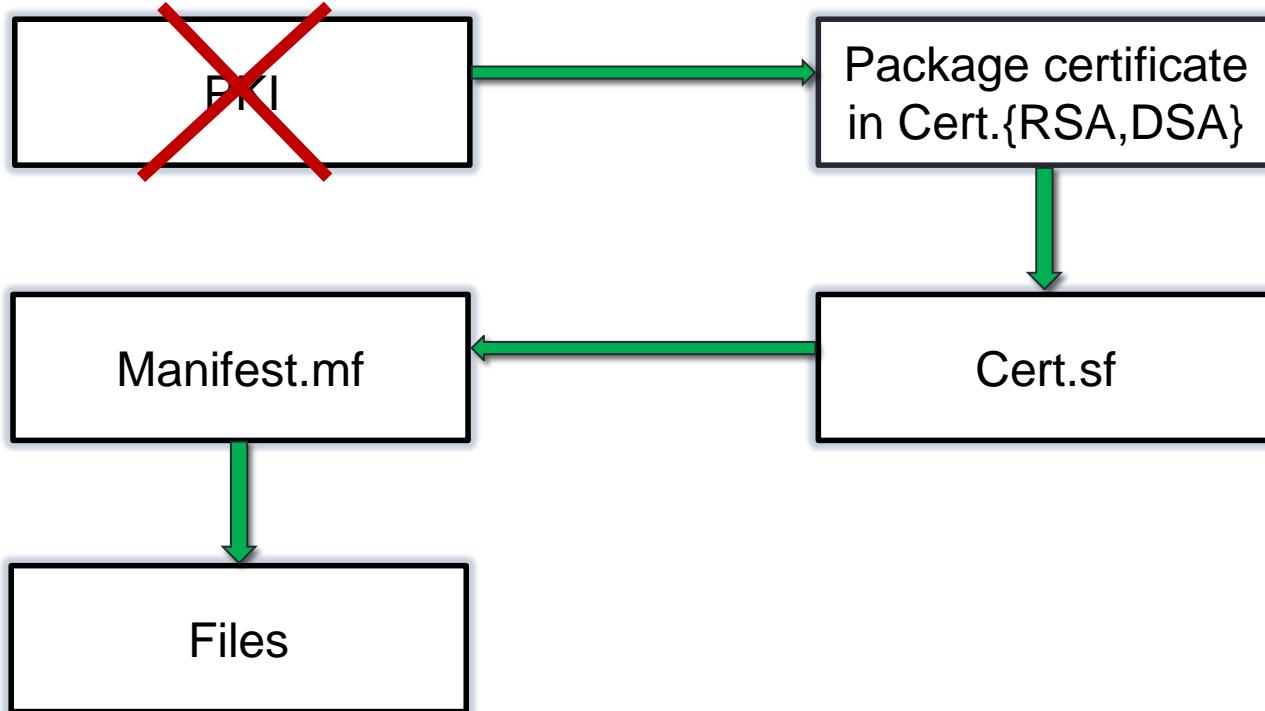
Certificate

Cert.sf signature

CERT.{RSA,DSA}

# Verifying of package manifest

Chain of trust:

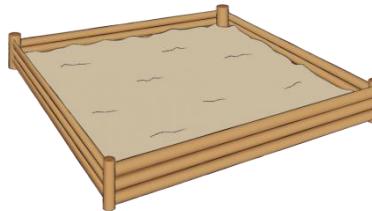


# ANDROID SECURITY ARCHITECTURE

---

- Package Integrity
- Sandboxing
- Permission and Least Privilege

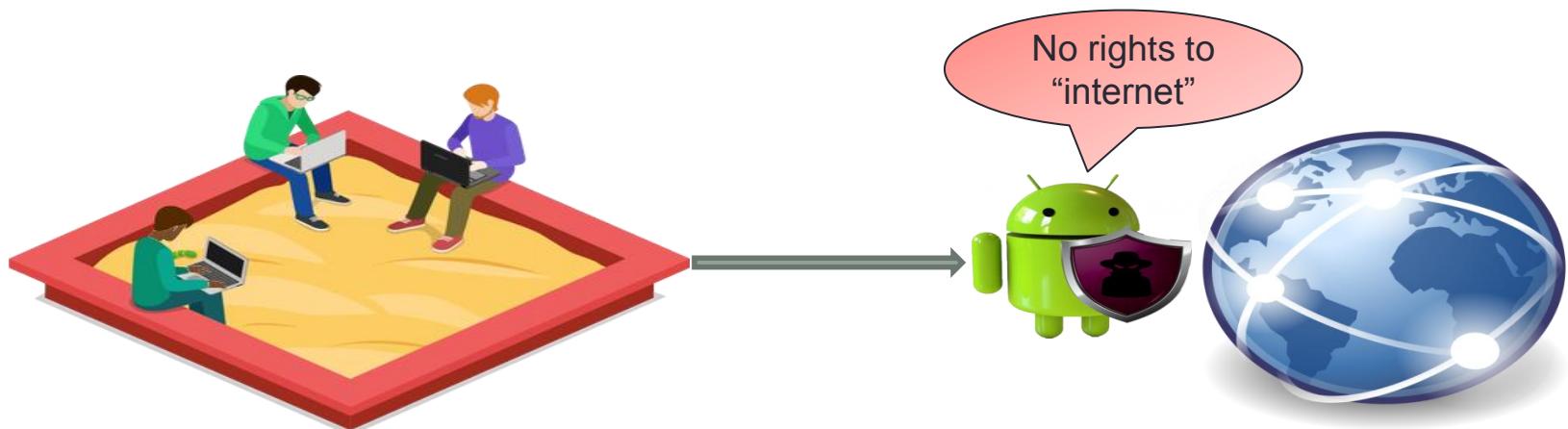
# Sandboxing



- ❑ The application sandbox **specifies** which system **resources** the application is allowed to access
- ❑ An **attacker** can only perform **actions** defined in the sandbox

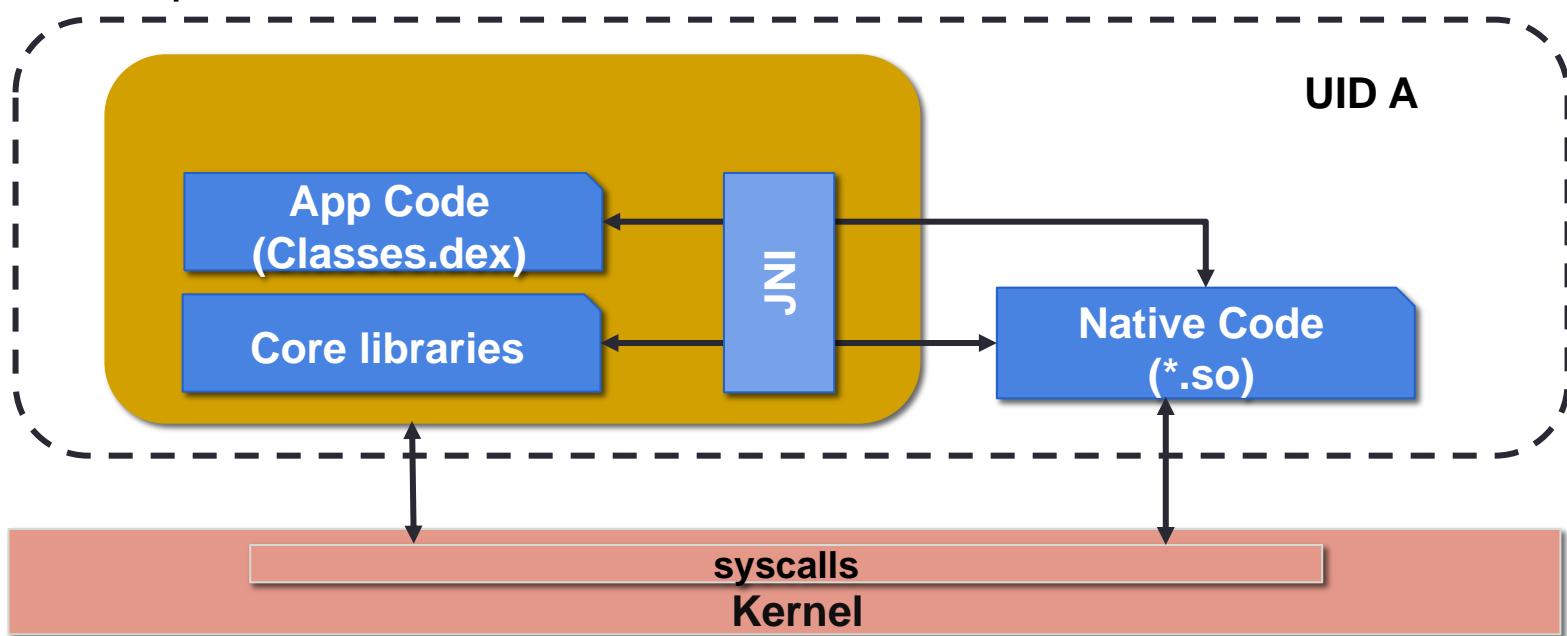
# Application Isolation by Sandboxing

- ❑ Each Application is **isolated** in its own **environment**
  - **Applications** can access only its **own resources**
  - Access to **sensitive resources** depends on the **application's rights**
- ❑ **Sandboxing** is enforced by **Linux**



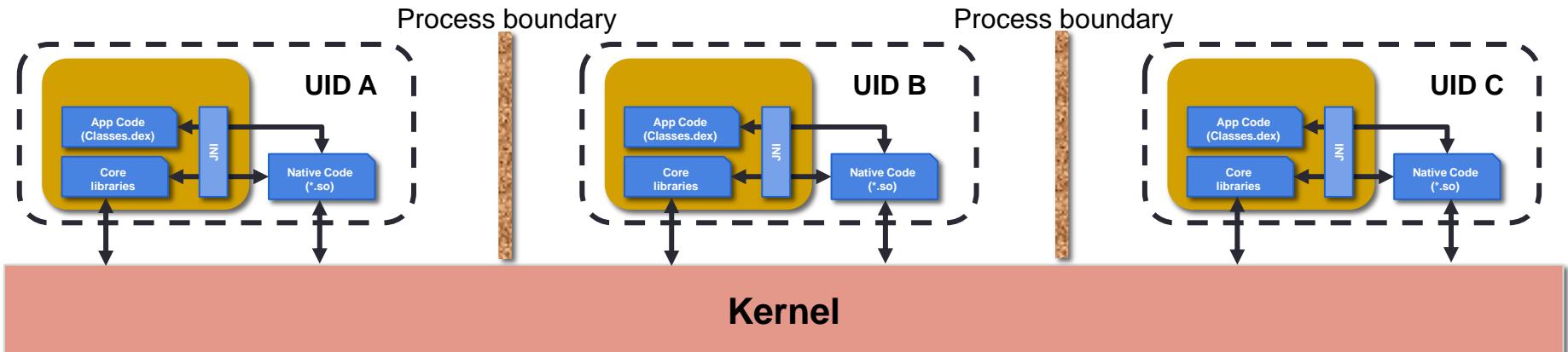
# Application sandbox

- ❑ Isolation: Each installed App has a separate user ID



# Application sandbox

- ❑ Isolation: Each installed App has a separate user ID
  - Each App lives in its own sandbox



# ANDROID SECURITY ARCHITECTURE

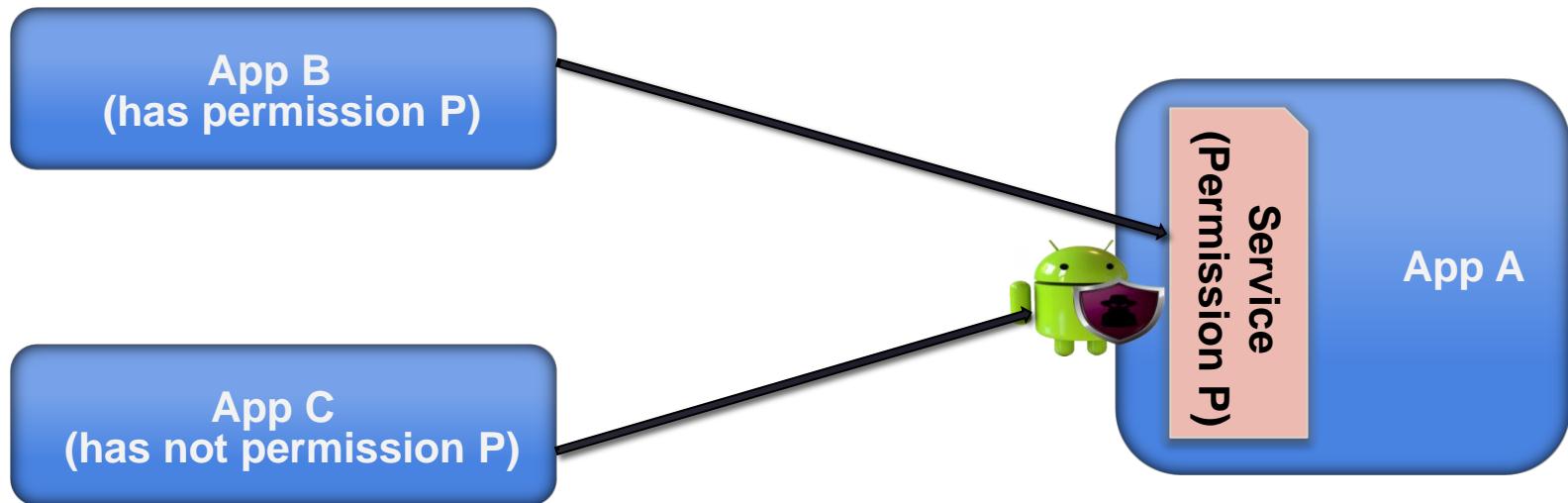
---

- Package Integrity
- Sandboxing
- Permission and Least Privilege

# Android Permission System

- ❑ **Access rights** in Android's application framework
  - Permissions are required to **gain** access to
    - System interfaces (Internet, send SMS, etc.)
    - System resources (logs, battery, etc.)
    - Sensitive data (SMS, contacts, etc.)
  - Currently more than 140 default permissions defined in Android
- ❑ Permissions are **assigned** to sandbox
- ❑ Application developers can also **define** their **own** permissions

# Android Permission: Example

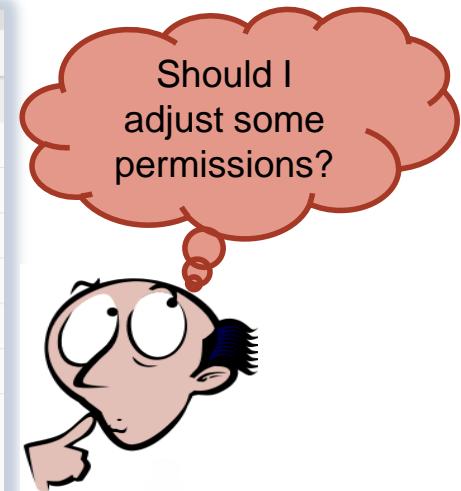
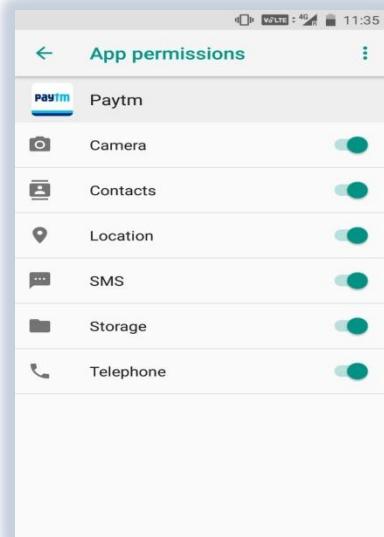
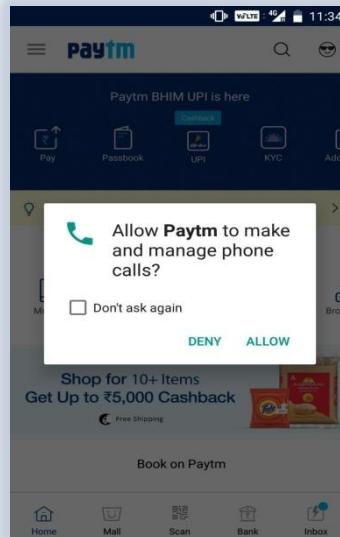
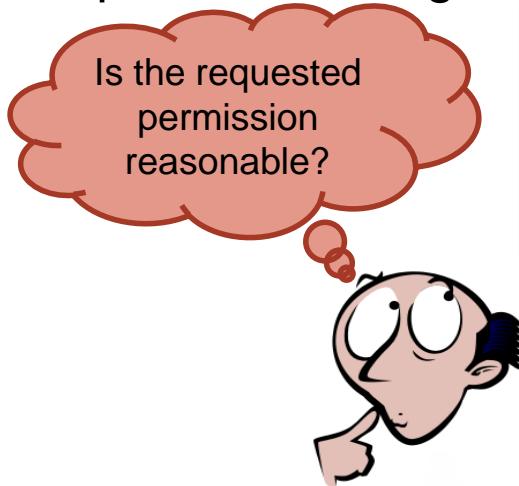


# Permissions' Protection Level

- Normal
- Dangerous
- Signature
- SignatureOrSystem

# Dynamic Permissions ( $\geq$ Android 6.0)

- ❑ App developers must **check** if their apps hold required **dangerous** permission, otherwise request them at runtime
- ❑ User can **grant** permissions at runtime and also **revoke** once granted permissions again

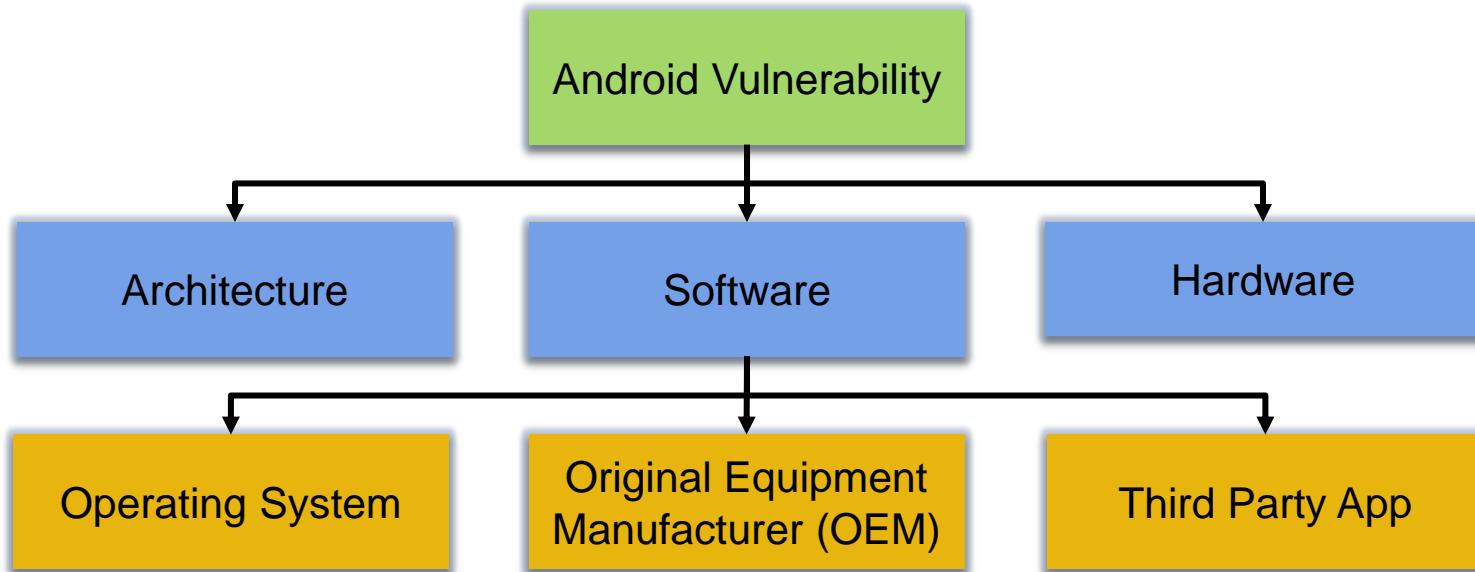


# ANDROID VULNERABILITIES

---

- Architecture Based
- Software Based
- Hardware Based

# Vulnerability Classification



# ANDROID VULNERABILITIES

---

- Architecture Based
- Software Based
- Hardware Based

# Application-Level Privilege Escalation Attacks



Malicious App



Confused Deputy App



Confused  
Deputy  
Attack



Malicious App

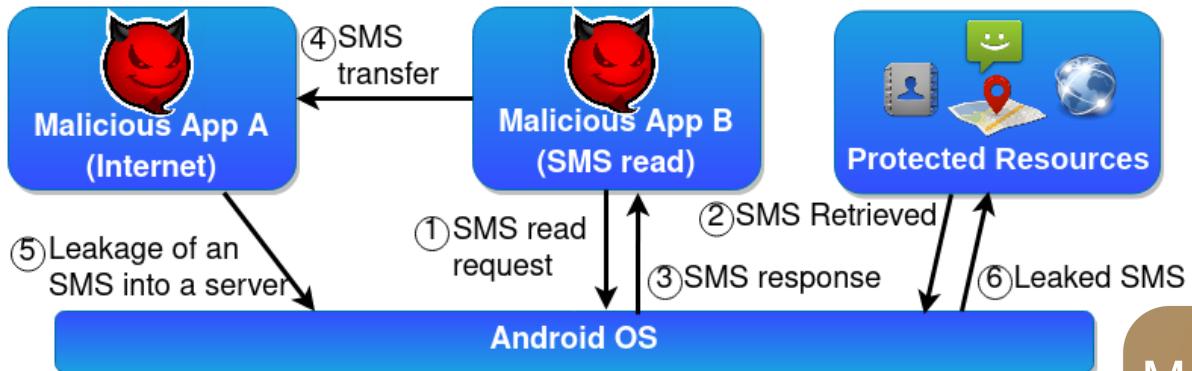


Malicious App



Collusion  
Attack

# Collusion Attack



Malicious apps **collude** in order to **merge** their respective **permissions**

## ❑ Variants:

- Apps communicate directly
- Apps communicate via covert channels in Android

# ANDROID VULNERABILITIES

---

- Architecture Based
- Software Based
- Hardware Based

# Dirty COW



- ❑ Existed in the Linux Kernel for **9 years**
- ❑ A **local** Privilege Escalation Vulnerability
- ❑ Exploits a race condition in the implementation of the **copy-on-write** mechanism
- ❑ Turns a **read-only** mapping of a file into a writable mapping

Android malware ZNIU exploits  
DirtyCOW vulnerability

29 SEP 2017 0

Android, Google, Malware, SophosLabs, Vulnerability

# Media Projection Service Issue

Vulnerabilities

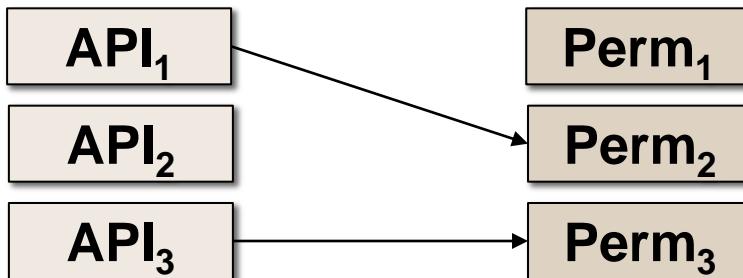
## Android issue allows attackers to capture screen and record audio on 77% of all devices

 November 20, 2017  Eslam Medhat  14 Views  0 Comments  android, MediaProjection

Source: <https://latesthackingnews.com/2017/11/20/android-issue-allows-attackers-to-capture-screen-and-record-audio-on-77-of-all-devices/>

# Over-privileged Apps

- ❑ Many apps request permissions that their **functionality** does not **require**
- ❑ Suspected root cause: API **documentation/naming** convention
  - Solution: API Permissions Maps
    - Can be integrated into lint tools



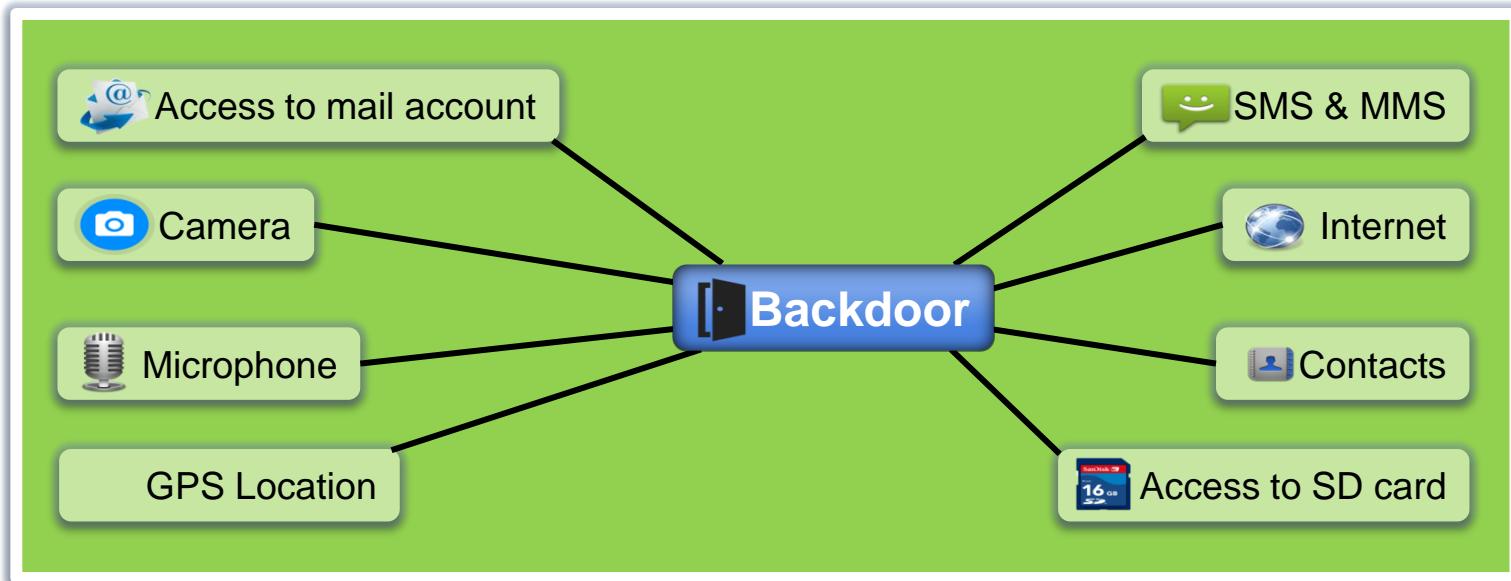
# Confused Deputy Attack



- ❑ A privileged app is fooled into **misusing** its privileges on behalf of another (malicious) **unprivileged app**
  
- ❑ Example:
  - **Unauthorized** phone calls
  - Various confused deputies in **system apps**

# Confused Deputy Introduce by OEMs

- ❑ Several **confused deputies** found in Samsung devices' **firmware**
  - One deputy running with system privileges provided **root shell service** to any app



# ANDROID VULNERABILITIES

---

- Architecture Based
- Software Based
- Hardware Based

# Broadcom Wi-Fi SoC Flaw

BIZ & IT —

## Android devices can be fatally hacked by malicious Wi-Fi networks

Broadcom chips allow rogue Wi-Fi signals to execute code of attacker's choosing.

DAN GOODIN - 4/6/2017, 1:16 AM

Source: <https://arstechnica.com/information-technology/2017/04/wide-range-of-android-phones-vulnerable-to-device-hijacks-over-wi-fi/>

# MALWARE ANALYSIS

---

# WHY MALWARE ANALYSIS?

## This data-stealing Android malware infiltrated the Google Play Store, infecting users in 196 countries

At least 100,000 people downloaded apps distributing MobSTSPY malware, which also leverages a phishing

**First Android Clipboard Hijacking Crypto Malware Found On Google**

## Android banking malware hitting more users than ever

Source: <https://www.techradar.com/news/android-banking-malware-hitting-more-users-than-ever>

By Anthony Spadafora 22 days ago [View comments](#)

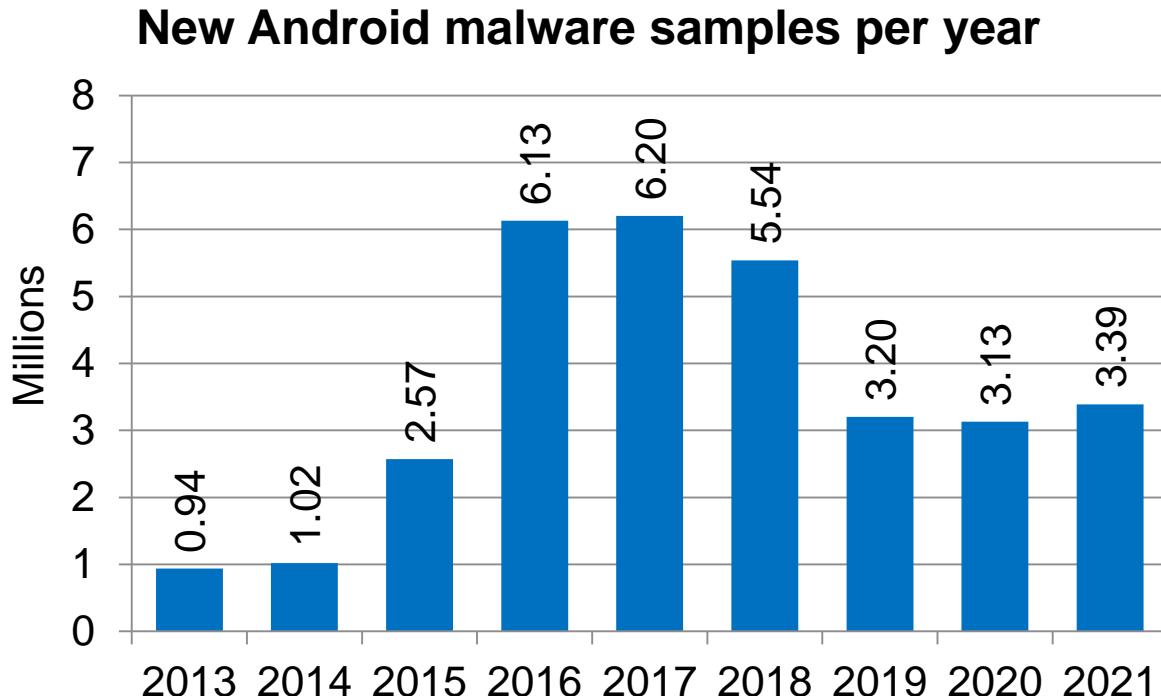
Fake banking apps could be more effective than banking Trojans

**Several Popular Beauty Camera Apps Caught Stealing Users' Photos**

 February 04, 2019  Swati Khandelwal

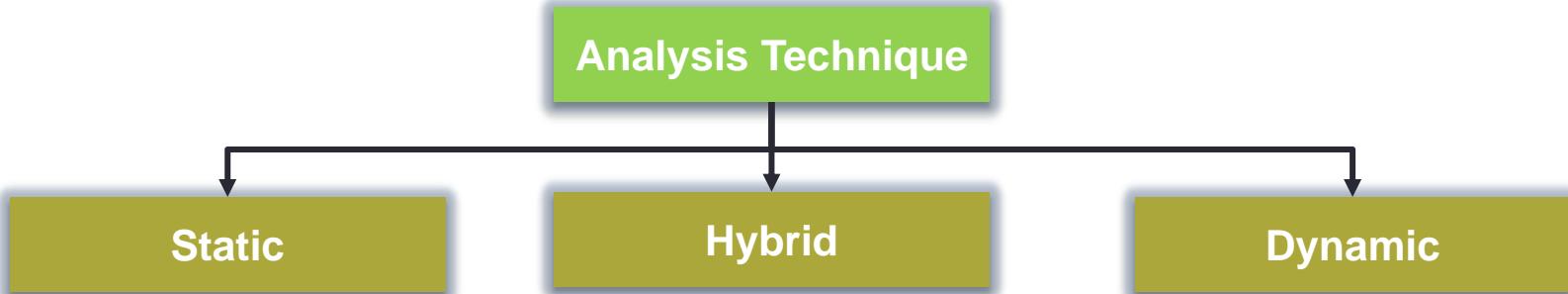
Source: <https://thehackernews.com/2019/02/beauty-camera-android-apps.html>

# Android Malware Statistics



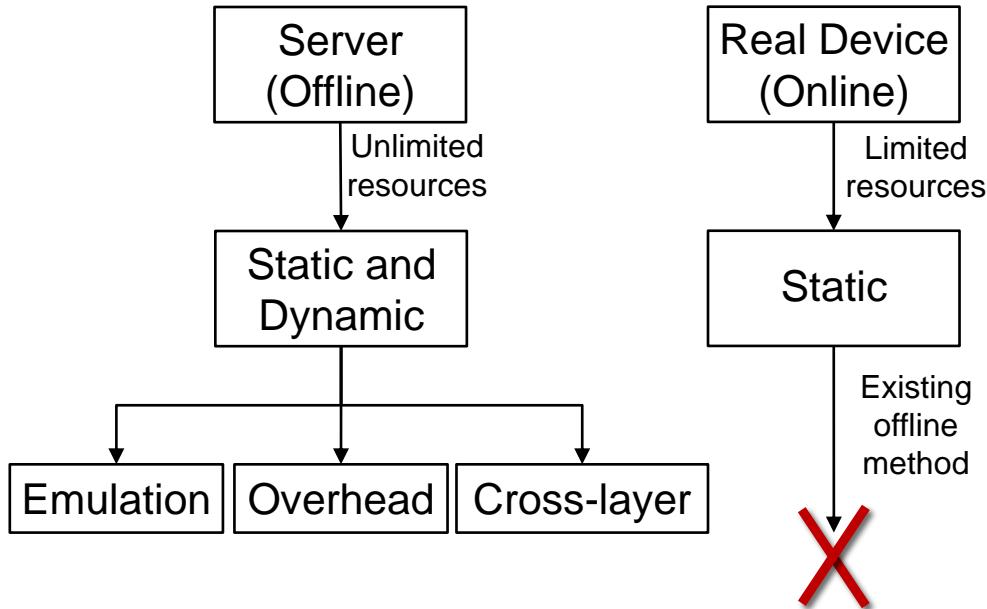
❑ In every 10 seconds, A new Android malware is born.

# Analysis Techniques



# Malware Analysis

- ❑ Many work has been proposed
- ❑ Deployed on
  - Server
  - Real Device
- ❑ Offline analysis can be bypassed
- ❑ On a real device, existing offline method cannot be used
  - High resources requirement



# Challenges: Dynamic Analysis

# Android Emulator

- ❑ A virtual mobile device
- ❑ Use Case:
  - Prototype, develop and test an application
  - Dynamic Analysis of malware
    - Used by security companies



# Emulation-Detection

- ❑ Detection methods are classified in 5 category
  - Unique Device Information (basic and **smart**)
  - Sensors Reading
  - GPS Information
  - Device State Information
  - Distributed Detection

# Unique Device Information

## Basic

- Unrealistic/null value for IMEI, Phone No. etc.



**IMEI**

123456XXXXX2347

**Phone No.**

901XXXXX36

**ICCID**

89XXXXX5611117910720



null/00000000000

155XXXXX554

89XXXXX3211118510720

## Smart

- Realistic but fixed values



3514XXXXX401216

972XXXXX243

89XXXXX0082067415160



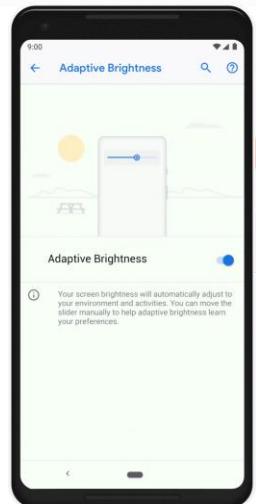
3514XXXXX401216

972XXXXX243

89XXXXX0082067415160

# Sensors

- ❑ Different sensors in a smart phone
  - Motion Sensors: accelerometer, gyroscope
  - Environmental Sensors: illumination (light), humidity
- ❑ Detection:
  - Count: At least 6-7 or more sensors in a smartphone
  - Reading: No change in sensors reading



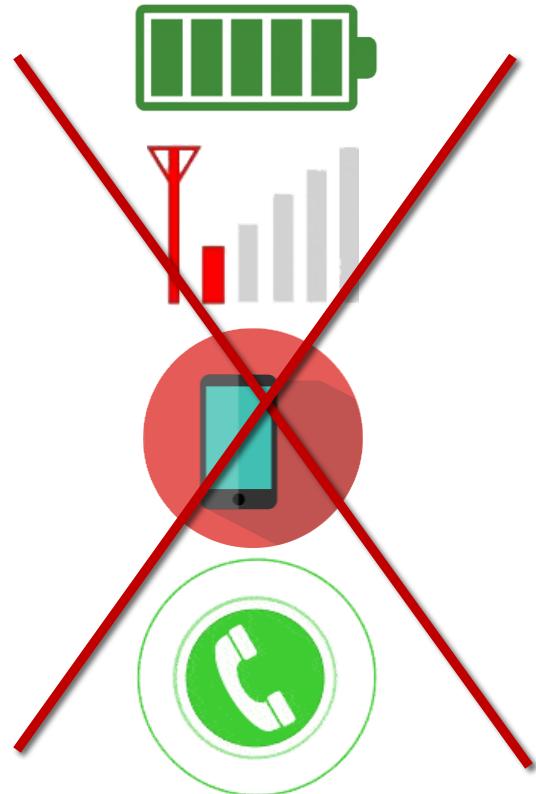
# GPS Information

- No change in GPS location
  
- Use of mock location API to provide fake location
  
- No correlation with BTS geo-location



# Device State Information

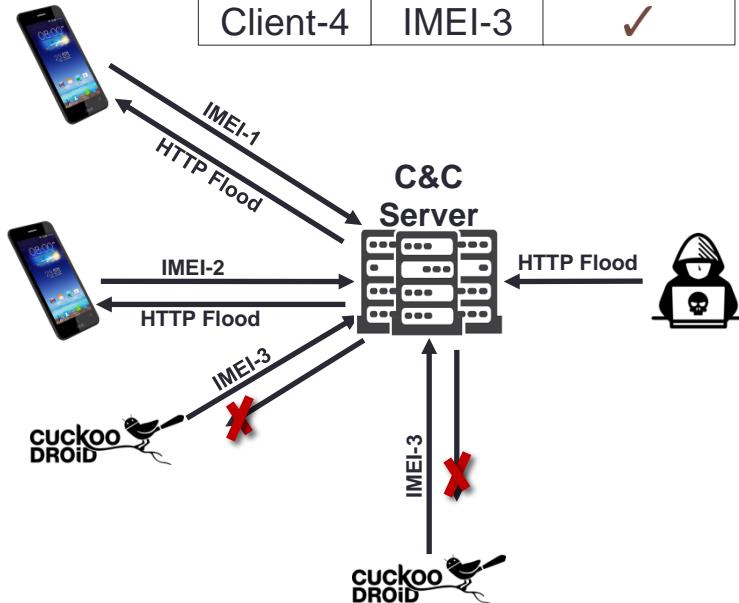
- Smartphone state may change due to:
  - Battery power
  - Signal Strength
  - SMS
  - Call
- No state change in emulated platform



# Distributed Detection

- ❑ Detection on server
  - App communicates with server
  - Observing identical information for multiple device like IMEI
  
- ❑ Example:
  - Botnet analysis

| Client No. | IMEI   | Emulated? |
|------------|--------|-----------|
| Client-1   | IMEI-1 | ✗         |
| Client-2   | IMEI-2 | ✗         |
| Client-3   | IMEI-3 | ✗         |
| Client-4   | IMEI-3 | ✓         |



# Existing Frameworks Evaluation

| Detection Type        | Emulator | DroidBox | CuckooDroid | MobSF |
|-----------------------|----------|----------|-------------|-------|
| Unique ID (Basic)     | ✓        | ✗        | ✗           | ✗     |
| Unique ID (Smart)     | ✓        | ✓        | ✓           | ✓     |
| Sensors reading       | ✓        | ✓        | ✓           | ✓     |
| Device State          | ✓        | ✓        | ✓           | ✓     |
| GPS                   | ✓        | ✓        | ✓           | ✓     |
| Distributed Detection | ✓        | ✓        | ✓           | ✓     |

Every framework fails to defend against all the detection method except for basic unique ID

# Summary: Emulation Detection

- ❑ Existing framework fails to defend against detection method:
  - Smart unique device information
  - Sensors and GPS information
  - Device state
  - Distributed detection
  
- ❑ Need a robust anti-emulation-detection system:
  - Hides underline emulated platform
  - Remain undetected when attack is performed from any layer

# Reference for More Details

- ❑ Robust Anti-Emulation-Detection

<https://www.youtube.com/watch?v=ahAgW4Wj3qc>

- ❑ On-Device Android Malware Detection

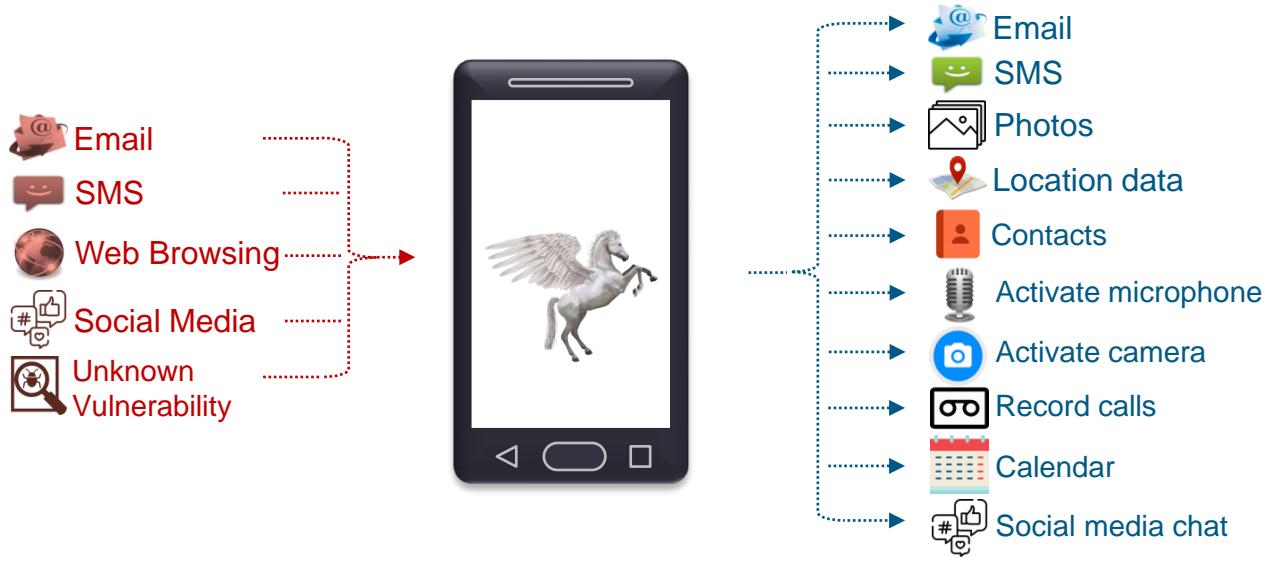
<https://www.youtube.com/watch?v=ziwlJGttkYg>



# CASE STUDY: ANALYSIS OF PEGASUS MALWARE

---

# Pegasus: Attack Vector and Capabilities



Attack Vector.

Capabilities

# Data Collection

- Samples were collected from CloudSek
- Total 5 Apps
- App-1 and App-3 are same only file name is different

| App ID | File Name  |
|--------|--|
| App-1  | 9fae5d148b89001555132c896879652fe1ca633d35271db34622248e048c78ae.apk |
| App-2  | 144778790d4a43a1d93dff6b660a6acb3a6d37a19e6a6f0a6bf1ef47e919648e.apk |
| App-3  | cc9517aafb58279091ac17533293edc1.apk                                 |
| App-4  | d257cfde7599f4e20ee08a62053e6b3b936c87d373e6805f0e0c65f1d39ec320.apk |
| App-5  | bd8cda80aaee3e4a17e9967a1c062ac5c8e4aefd7eaa3362f54044c2c94db52a.apk |

# Analysis Type and Environment

## ❑ Static

- Androguard

## ❑ Dynamic

- STDNeut: Neutralizing Sensor, Telephony System and Device State Information on Emulated Android Environments
- Xposed framework to monitor API calls
- SysCallMon: A system call monitoring Kernel module

# Analysis Result

# App-1 and App-3

## Meta Information

- Package Name: com.binary.sms.receiver
- Modification Date: 2 June, 2014
- Hash: 9fae5d148b89001555132c896879652fe1ca633d35271db34622248e048c78ae

## Server Communication

| IP/URLs        | Port | Geo Location                      |
|----------------|------|-----------------------------------|
| 142.XXX.27.188 | 443  | Mountain View,<br>California, USA |
|                | 5228 |                                   |

# App-1 and App-3 cont..

## System Command

- chmod, mount, su

## Capability

- Install new applications
- Make a call, listen or record incoming/outgoing call
- Read/Write contacts, bookmark,
- Many more...

# App-1 and App-3 cont..

## Observation:

- Tries to get root privilege
- Change file permissions
- Mount system partition as R/w
- Intercept incoming/outgoing SMS and Calls
- Obtain information about installed and running apps
- Can install new apps
- Read other information like contacts, history bookmarks,
- Read/write system settings,
- Process outgoing calls and send new sms
- Delete call logs and many more.

# App-2

## Meta Information

- Package Name: com.lenovo.safecenter
- Modification Date: 16 Dec, 2010
- Hash: 144778790d4a43a1d93dff6b660a6acb3a6d37a19e6a6f0a6bf1ef47e919648e

## Server Communication

| IP/URLs         | Port | Geo Location                      |
|-----------------|------|-----------------------------------|
| 142.XXX.102.188 | 443  | Mountain View,<br>California, USA |
|                 | 5228 |                                   |
| 142.XXX.5.188   | 443  | Mountain View,<br>California, USA |
|                 | 5228 |                                   |

# App-2 cont..

## System Command

- app\_process, bind, cat, chmod, chown, close, connect, date, dumpsys, echo, exit, gzip, id, iptables, kill, log, logcat, ls, mkdir, mount, mv, notify, open, pm, ps, pwd, read, reboot, sdcard, select, service, sh, socket, start, su, system\_server, times, uptime, write

## Capability

- Make a call, send new SMS
- Read/Write contacts, system settings,
- Process outgoing calls
- Access location data
- Kill background processes
- Many more...

# App-2 cont..

## Observation:

- Capable to bypass dynamic analysis using device information
- Tries to get root privilege
- Can change files permission
- Mount system partitions as RW
- Open network sockets
- Get running process information and kill any process
- Dynamically load code, end an incoming call, kill background processes
- Remove any app
- Register a broadcast receiver to intercept incoming SMS

# App-4

## Meta Information

- Package Name: com.xxGameAssistant.pao
- Modification Date: 15 Nov, 2013
- Hash: d257cfde7599f4e20ee08a62053e6b3b936c87d373e6805f0e0c65f1d39ec320

## System Command

- Chmod, dd, ln, mkdir, mount, stop, su

## Capability

- Read Phone state
- Access location data
- Listen to boot complete event
- Read/Write to external storage

# App-4

## Server Communication

| IP/URLs                          | Port/Protocol | Geo Location                   |
|----------------------------------|---------------|--------------------------------|
| http://tdcv3.talkingdata.net/g/d | HTTP          | Kansas City, Missouri, USA     |
| http://tdcv3.talkingdata.net     | DNS           |                                |
| 35.XXX.63.213                    | --            |                                |
| 142.XXX.188.196                  | --            |                                |
| 142.XXX.102.188                  | --            | Mountain View, California, USA |
| 142.XXX.27.188                   | --            |                                |
| 142.XXX.5.188                    | --            |                                |

# App-4 cont..

## Observation:

- Tries to get root privilege
- Can change files permission
- Mount system partitions as R/W
- Capable to bypass dynamic analysis using device & CPU information
- Can install apps,
- Get information about the currently installed/running App, processes and tasks
- Track location and steal sensitive information like device Ids, phone numbers and others
- Listen to BOOT COMPLETE event so that it can run a code or background process when a phone restarts.

# App-5

## Only static analysis

- Dex file is tempered, hence no dynamic analysis

## Meta Information

- Package Name: sec.dujmehn.qdtheyt
- Modification Date: 10 Nov, 2018 based on last modified content
- Hash: bd8cda80aaee3e4a17e9967a1c062ac5c8e4aefd7eaa3362f54044c2c94db52a

# App-5 cont..

## Capability

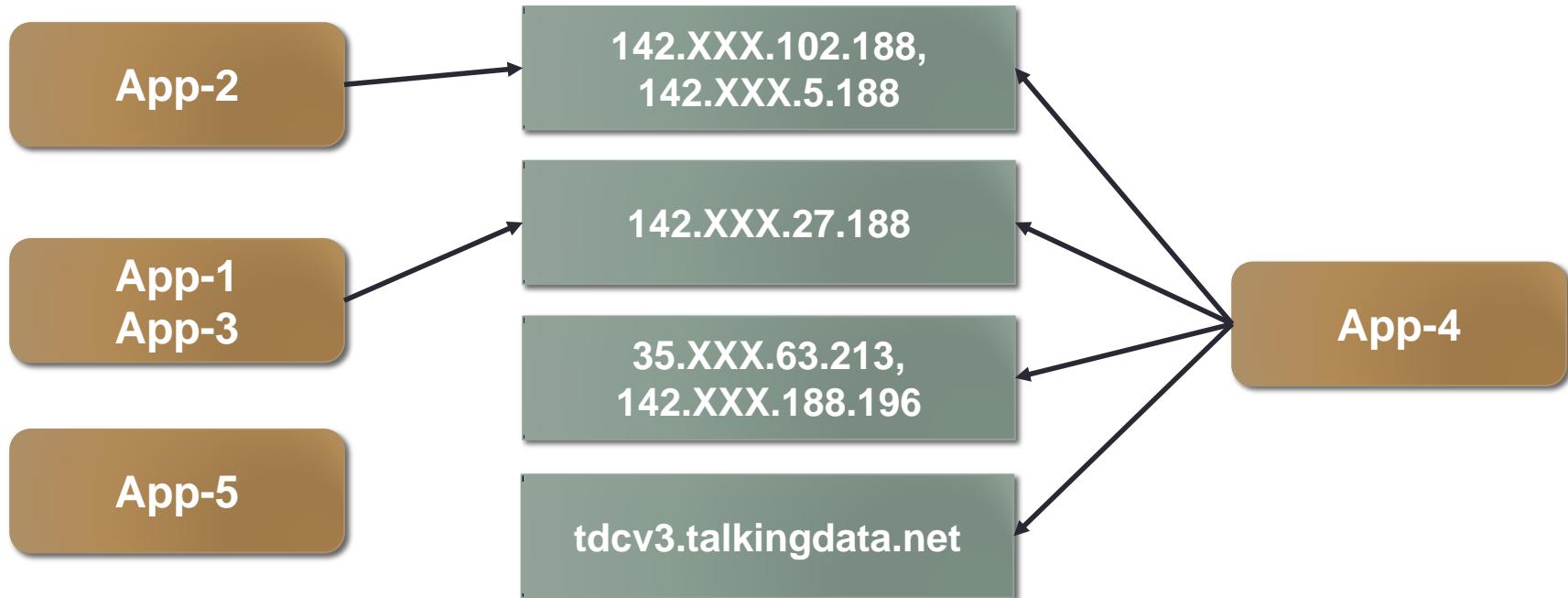
- Install new applications
- Make a call, listen or record incoming/outgoing call
- Read/Write contacts, bookmark,
- Access to location data
- Send and read SMS
- Kill background process
- Set fake location information
- Many more...

# App-5 cont..

## Observation:

- Can change files permissions
- Mount system partitions as R/W.
- Can get information about currently installed/running apps, processes and tasks
- Track location and steal sensitive information like device Ids, phone numbers and others.
- Listen to BOOT COMPLETE, NEW SMS, OUTGOING CALLS, BATTERY STATUS CHANGED, and many other events
  - Can run a code or background process when any of such event occurs
- Ability to change system configuration, R/W contacts, bookmark history,
- Record audio in background, install apps

# Connection Between Apps



# Detection of Pegasus

- ❑ Used DeepDetect, machine learning based Android malware detector
- ❑ Static features from Manifest File and Dex code
- ❑ Results

| App ID | Detection Result |
|--------|------------------|
| App-1  | ✓                |
| App-2  | ✓                |
| App-3  | ✓                |
| App-4  | ✓                |
| App-5  | ✓                |

**(Only based on Android Manifest file)**



<https://github.com/skmtr1/Workshop-Mobile-Forensics-And-Security>

# Thank You