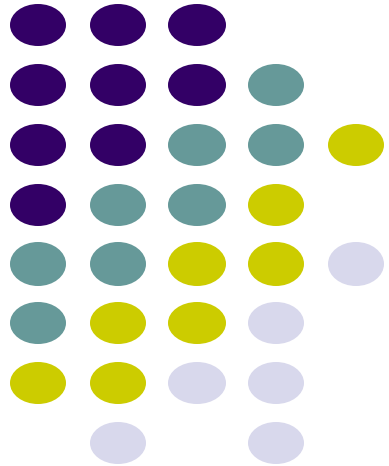
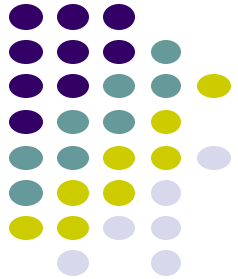
Mobile and Ubiquitous Computing on Smartphones

Lecture 10a: Mobile Security and Mobile Software Vulnerabilities

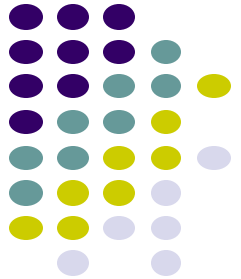
Emmanuel Agu



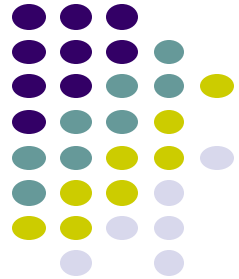


Mobile Security Issues

Introduction

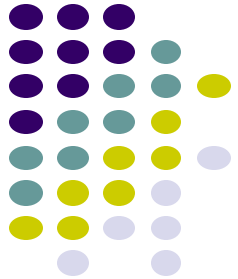


- So many cool mobile apps
- Access to web, personal information, social media, etc
- Security problems (not previously envisaged) have resulted
- Examples:
  - Malicious apps can steal your private information (credit card information, etc)
  - Jogging map generated from paths of Fitbit users can expose locations/behavioral habits of users. E.g. US soldiers at German base
  - Malware can lock your phone till you pay some money (ransomeware)
- Users/developers need better understanding of mobile security



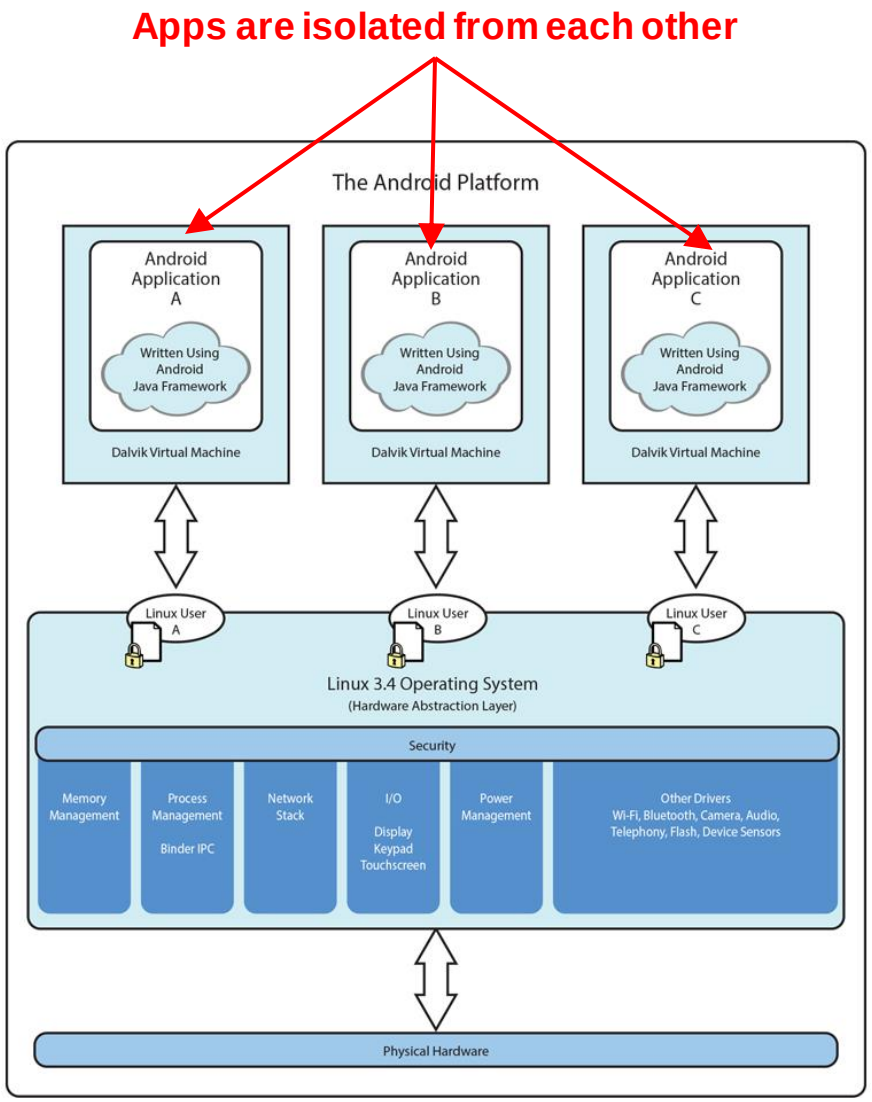
Android Security Model

Android Security



- Android security goals are to
  - Protect user data, system resources (hardware, software)
  - Isolate applications (e.g. app 1 from app 2)

- **Foundations of Android Security**
  1. **Application Isolation:**
     - Application sandboxing: App 1 cannot interact directly with app 2
     - Apps can only communicate using secure inter-process communication
  2. **Permission Requirement:**
     - Supports default system, and user-defined permissions
     - All apps must be signed: identifies author, ensures future updates are authentic

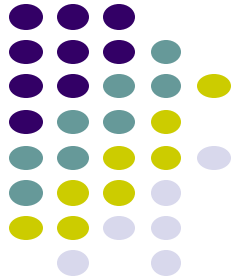


Ref: Introduction to Android Programming, Annuzzi, Darcey & Conder

Recall: Android Software Framework

- Each Android app runs in its own security sandbox (VM, minimizes complete system crashes)
- Android OS multi-user Linux system
- Each app is a different user (assigned unique Linux ID)
- Access control: only process with the app's user ID can access its files
- Apps talk to each other only via intents, IPC or ContentProviders

Android Encryption



- Encryption encodes data/information, unauthorized party cannot read it

- **Full-disk encryption:** Android 5.0+ supports full filesystem encryption
  - Single key used to encrypt all the user's data
  - User password needed to access files, even to boot device

- **File-based encryption:** Android 7.0+ allows specific files to be encrypted and unlocked independently
  - Different keys used to encrypt different files

iPhone vs Android Encryption

- iPhones encrypt automatically: almost all encrypted
- More iPhone versions encrypted as requirement vs Android

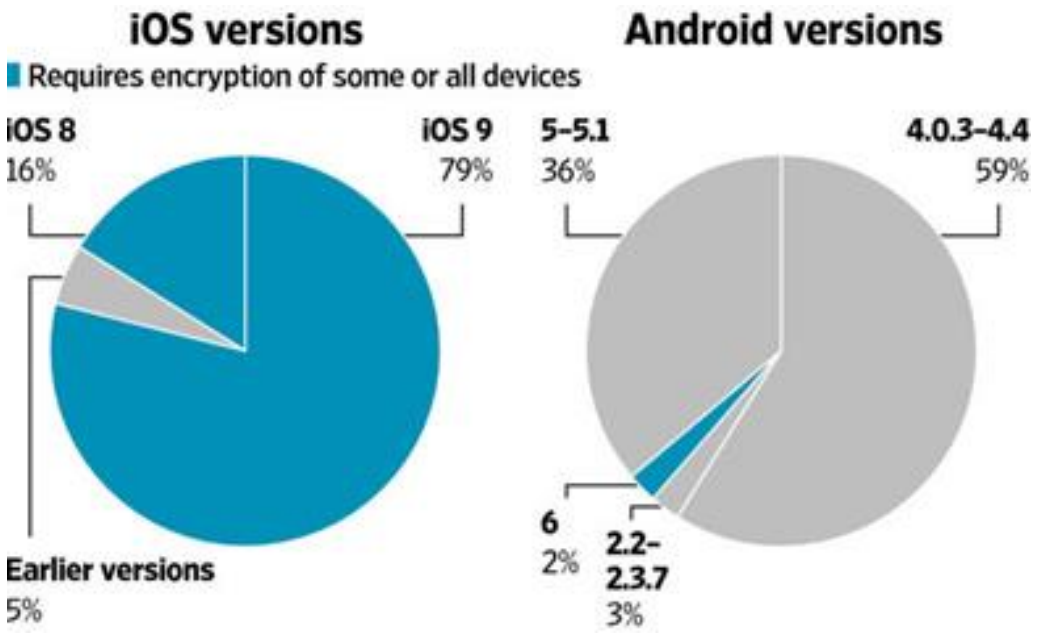
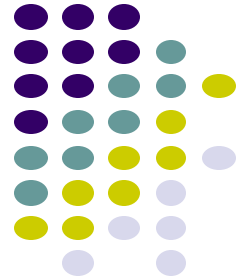


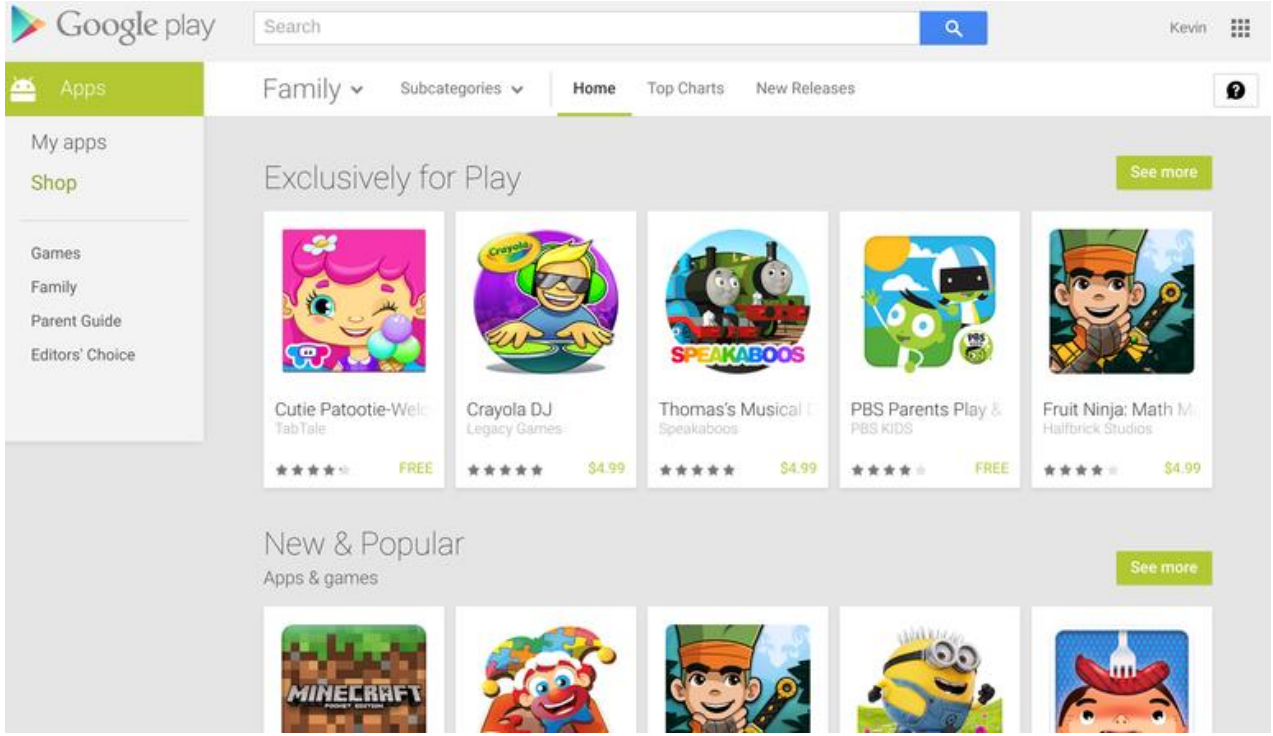
Image credit: wall street journal



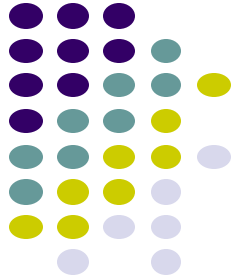
App Markets

App Markets & Distribution

- Major OS vendors manage their own markets for “certified” apps
  - Android: Google Play Store
  - iOS: App Store (only way to download iPhone apps)



App Market Scanning

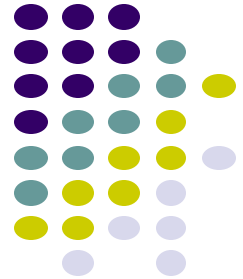


Google App Store: scanning called **Google Play Protect**

- Antivirus scans apps on Google Play for threats, malware
- New "peer grouping system:
  - similar apps (e.g. all calculators) are grouped on app market.
  - If an app requests more permissions than similar apps, human takes a look
- Also scans apps already installed on device, warns user if app looks malicious

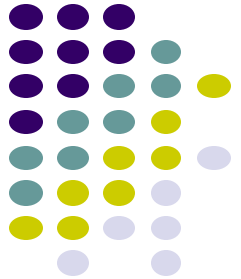
- Apple App Store
  - Highly regulated
  - All applications are reviewed by human
  - iOS devices can only obtain apps through official app store, unless jailbroken

- Many malware developers target third-party app stores (e.g. Amazon, getJar)
  - Weaker/no restrictions or analysis capabilities



Malware Evolution

Threat Types: Malware, Grayware & Personal Spyware

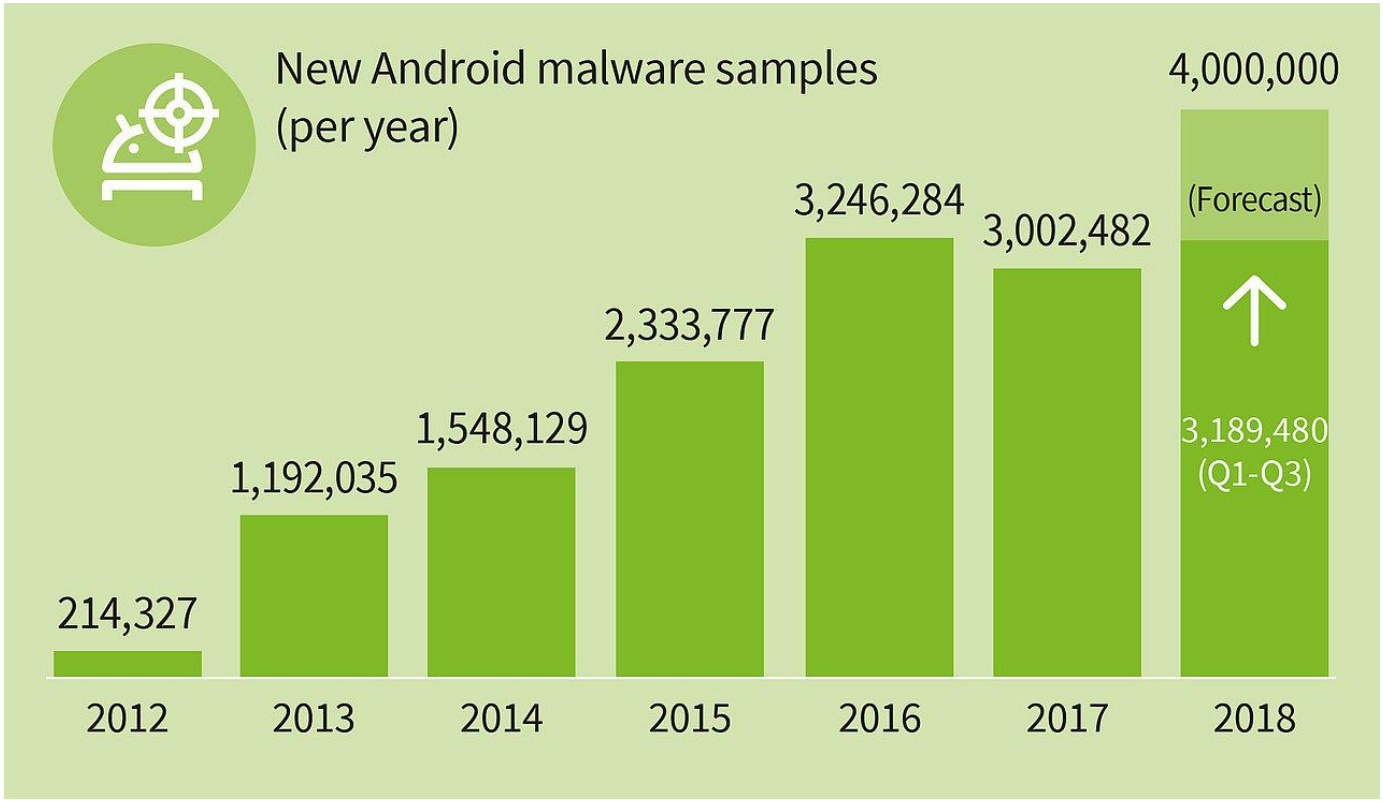


- **Malware:**
  - Gains access to a mobile device in order to steal data, damage device, or annoying the user, etc. **Malicious!!**

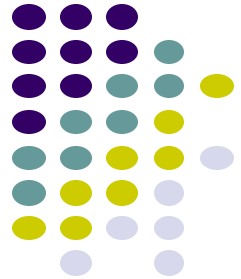
- **Personal Spyware:**
  - Collects user's personal information over of time
  - Sends information to app **installer** instead of author
  - E.g. spouse may install personal spyware to get info

- **Grayware:**
  - Collect data on user, but with no intention to harm user
  - E.g. for marketing, user profiling by a company

Growth of Android Malware



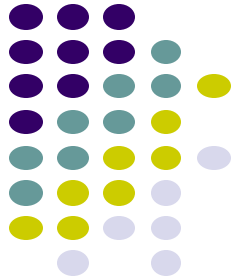
Ref: https://www.gdatasoftware.com/blog/2018/11/31255-cyber-attacks-on-android-devices-on-the-rise



Mobile Malware Survey (*Felt et al*)

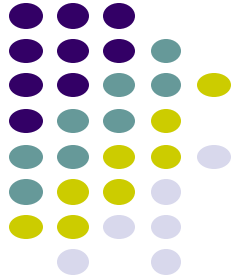
Mobile Malware Study?

A survey of mobile malware in the wild Adrienne Porter Felt, Matthew Finifter, Erika Chin, Steve Hanna, and David Wagner in Proc SPSM 2011



- First major mobile malware study in 2011 by Andrienne Porter Felt *et al*
  - Prior studies mostly focused on PC malware
- Analyzed 46 malwares that spread Jan. 2009 – June 2011
  - 18 – Android
  - 4 – iOS
  - 24 – Symbian (discontinued)
- Analyzed information:
  - in databases maintained by anti-virus companies
    - E.g., Symantec, F-Secure, Fortiguard, Lookout, and Panda Security
  - Discover malware based on mentions of malware in news sources
- Just analyzed malware. Did not analyze spyware and grayware

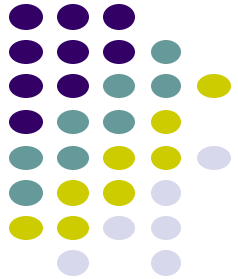
Categorized Apps based on Behaviors



1. **Novelty and amusement**
   - Causes minor damage
   - E.g. Change user's wallpaper

2. **Selling user information**
   - Malware obtains user's personal information via API calls
     - E.g. User's location, contacts, download + browser history/preferences
   - Information can be sold to advertisers
     - E.g. Dunkin Donuts may want to know users who visit their competitors
     - Price: $1.90 to $9.50 per user per month

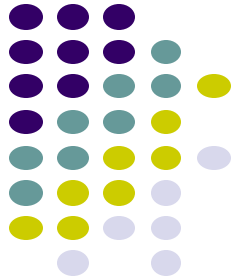
Categorized Apps based on Behaviors



3. **Stealing user credentials**
   - People use smartphones for activities that require them to input their passwords and payment information. E.g. shopping, banking, e-mail
   - Malwares can log keys typed by user (keylogging), scan their documents for username + password

   - User credentials can be sold
   - In 2008, black market price of:
     - Bank account credentials: $10 to $1, 000,
     - Credit card numbers: $.10 to $25,
     - E-mail account passwords: $4 to $30

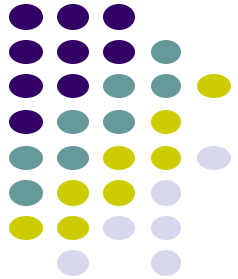
Categorized Apps based on Behaviors



4. **Make premium-rate calls and SMS**
   - Premium rate texts to specific numbers are expensive (E.g. 1-900.. Numbers)
   - Attacker can set up premium rate number, Malware sends SMS there
   - User is billed by their cell carrier (e.g. sprint), attacker makes money

5. **SMS spam**
   - Used for commercial advertising and phishing
   - Sending spam email is illegal in most countries
   - Attacker uses malware app on user's phone to send SPAM email
   - Harder to track down senders

Categorized Apps based on Behaviors



6. **Search Engine Optimization (SEO):**
   - Malware makes HTTP requests for specific pages to increase their search ranking (e.g. on Google)
   - Increases popularity of requested websites

7. **Ransomeware**
   - Possess device, e.g. lock screen till money is paid
   - *Kenzero* – Japanese virus inserted into pornographic games distributed on P2P networks
     - Publishes user's browser history on public website
     - Asked **5800 Yen** (~$60) to delete information from website
     - About 12 % of users (661 out of 5510) actually paid

Ransomware

Ransomware: Type of malware that prevents or limits users from accessing their system, by locking smartphone's screen or by locking the users' files till a ransom is paid

Source: Lookout Top Threats
https://www.lookout.com/resources/top-threats/scarepakage

Source: MalwareBytes "State of Malware Report" 2017
https://www.malwarebytes.com/pdf/white-papers/stateofmalware.pdf

Frequency of Malware Categories

Exfiltrates user information	28
Premium calls or SMS	24
Sends SMS advertisement spam	8
Novelty and amusement	6
Exfiltrates user credentials	4
Search engine optimization	1
Ransom	1

Table 1: We classify 46 pieces of malware by behavior. Some samples exhibit more than one behavior, and every piece of malware exhibits at least one.

Malware Detection based on Permissions

- Does malware request more permissions?
- Analyzed permissions of 11 Android malware

- **Findings: Yes!**
  - 8 of 11 malware request SMS permission (73%)
    - Only 4% of non-malicious apps ask for this
  - Dangerous permissions: requests for personal info (e.g. contacts), etc
  - Malware requests 6.18 dangerous permissions
    - 3.46 for Non-malicious apps

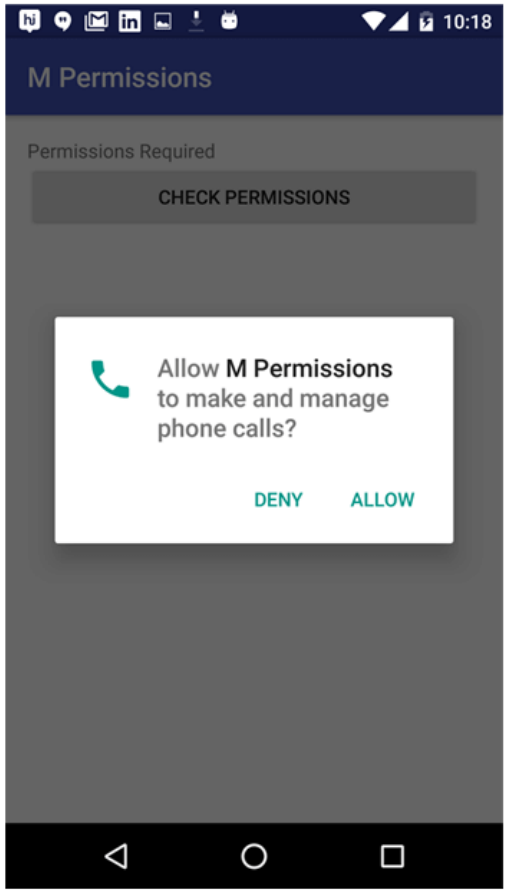
Number of Dangerous permissions	Number of non-malicious applications	Number of malicious applications
0	75 (8%)	-
1	154 (16%)	1
2	182 (19%)	1
3	152 (16%)	-
4	140 (15%)	2
5	82 (9%)	1
6	65 (7%)	-
7	28 (3%)	2
8	19 (2%)	1
9	21 (2%)	1
10	10 (1%)	1
11	6 (0.6%)	1
12	7 (0.7%)	-
13	4 (0.4%)	-
14	4 (0.4%)	-
15	2 (0.2%)	-
16	1 (0.1%)	-
17	1 (0.1%)	-
18	-	-
19	-	-
20	1 (0.1%)	-
21	-	-
22	-	-
23	1 (0.1%)	-
24	-	-
25	-	-
26	1 (0.1%)	-

Table 2: The number of "Dangerous" Android permissions requested by 11 pieces of malware and 956 non-malicious applications [28].

Android Run-Time Permissions Changed in Marshmallow (Android 6.0)

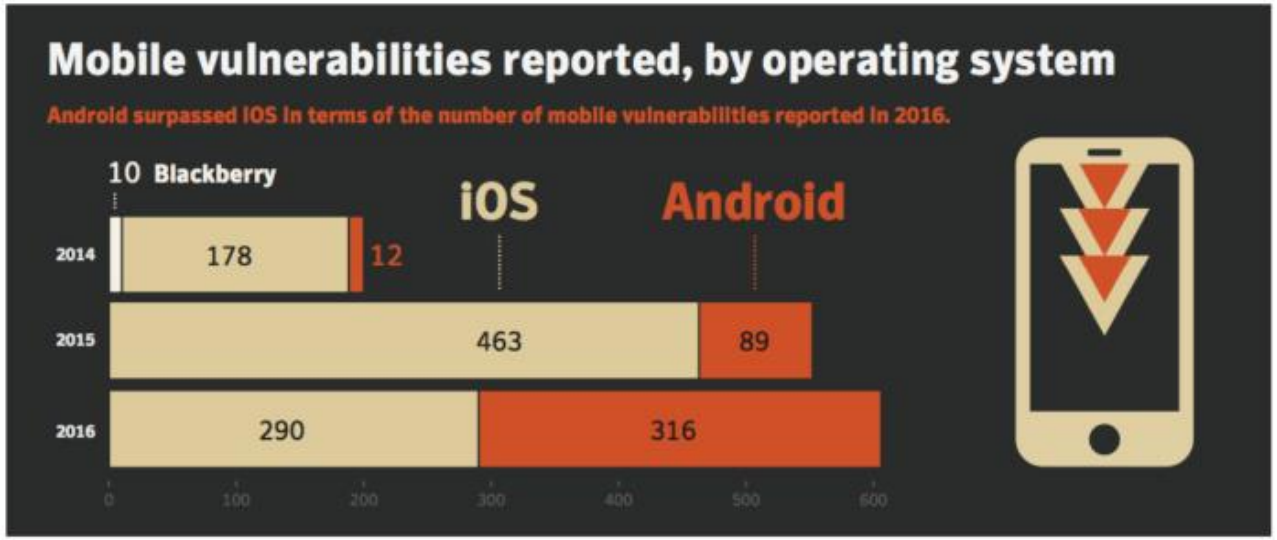
- Pre Android 6.0: Permissions during install
- Android 6.0: Changes!!
- "Normal" permissions don't require user consent
  - E.g. change timezone
  - Normal permissions can do very little to harm user
  - Automatically granted
- Dangerous permissions (e.g. access to contacts can harm user

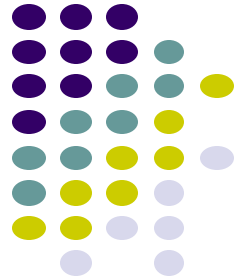
- Android 6.0: Run-time permissions now required for "dangerous" permissions



iOS Malware Review

- iOS generally fewer vulnerabilities (even till date)
  - All 4 pieces of Apple malware were spread through jailbroken devices;
  - not found on App Store
  - iOS: Human reviews all apps, more effective, but slower!!?





Using Hand Gestures to Curb Mobile Malware (*Shrestha et al*)

Malware Protection using Hand Movements

Curbing Mobile Malware Based on User-Transparent Hand Movements Babins Shrestha,Manar Mohamed, Anders Borg, Nitesh Saxena and Sandeep Tamrakar in Proc IEEE Percom 2015

- **General idea:** Use real world hand movements to distinguish malware from real user

- Real user will make certain natural hand gestures when:
  - Making phone call
  - Taking a picture
  - Swiping to use NFC reader

- These hand gestures will be missing if activity is by malware

- **Main idea:** Check for these gestures (gesture recognition) to distinguish malware requests from valid user requests

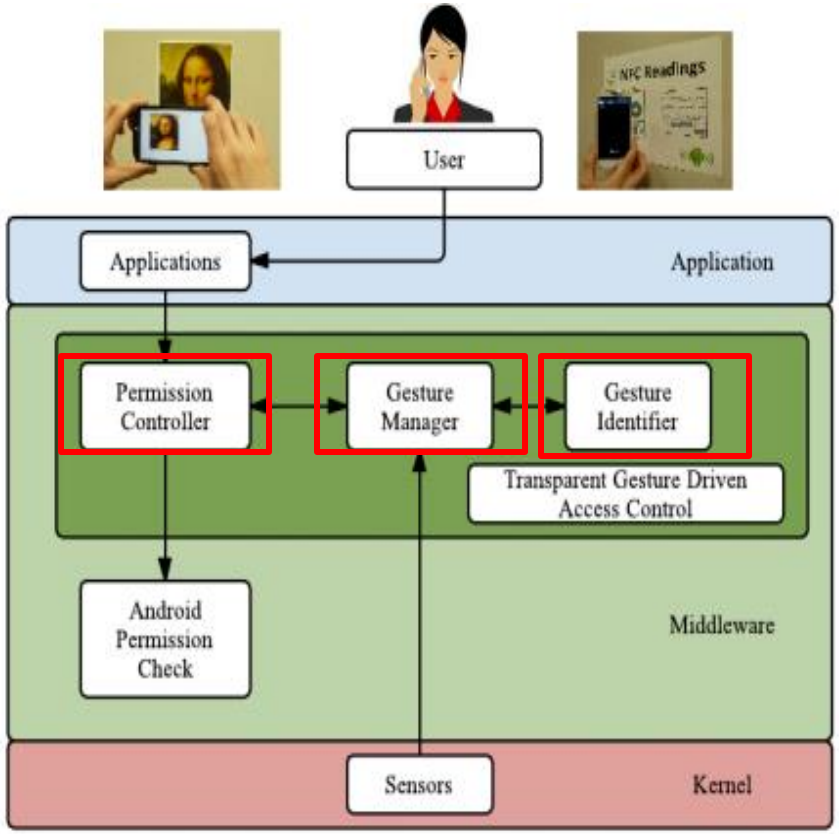
Sensors used for Gesture Identification

- Gesture Identifier used sensors to detect natural hand movements associated with phone dialing, taking picture, NFC usage
  - **Motion Sensors:** Accelerometer and gyroscope
  - **Position Sensors:** Magnetometer and orientation sensors
  - **Environmental Sensors:** Temperature, pressure and illuminance

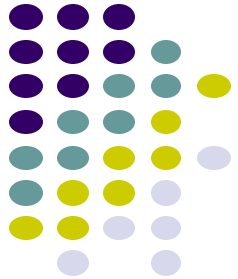
TABLE I. SENSORS UTILIZED FOR GESTURE DETECTION

Type	Sensor	Description
Motion	Accelerometer (A)	The acceleration force including gravity
Motion	Gyroscope (Gy)	The rate of rotation
Motion	Linear Acceleration (LA)	The acceleration force excluding gravity
Motion	Rotation Vector (R)	The orientation of a device
Motion	Gravity (G)	The gravity force on the device
Position	Game Rotation (GR)	Uncalibrated rotation vector
Position	Magnetic Field (M)	The ambient magnetic field
Position	Orientation (O)	The device orientation
Environment	Pressure (P)	The ambient air pressure

System Architecture



- 3 Entities
  - **Gesture Identifier:** classifier to identify gesture
  - **Permission Controller:** checks permissions granted by Android
  - **Gesture Manager:** compares gestures with permissions
- **Results:** > 85% accuracy (user gesture detection)



Mobile Ad Vulnerabilities

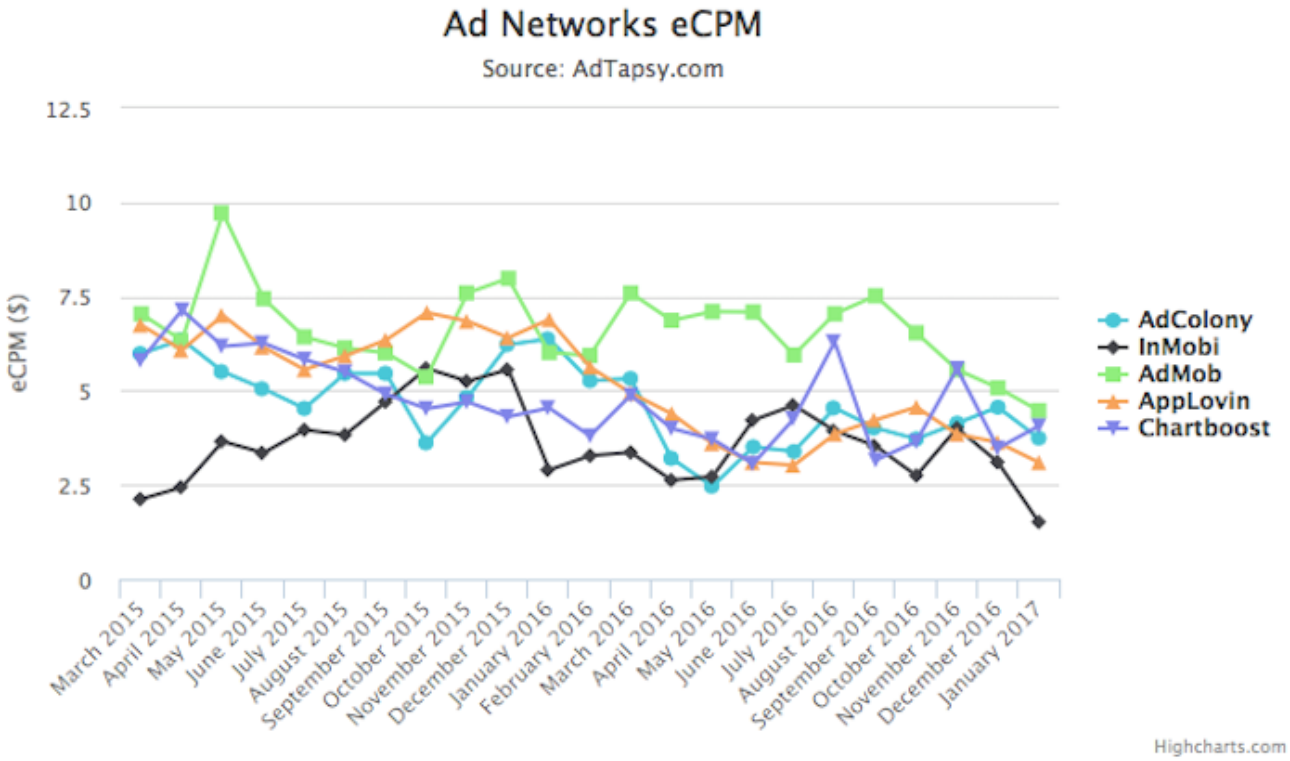
Ad Services

- App developers make money from apps in 2 main ways:
  - Charge users fee for apps
  - Getting $$$ from advertisers to include ads in apps
- To make money from ads, app author integrates ad services into app
- Mobile ad company serves ads to device

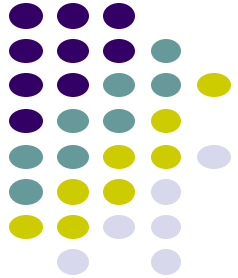
Ads

AdMob

- AdMob: Most popular mobile ad company
  - Acquired by Google in 2009



Permissions Requested by Ad Services

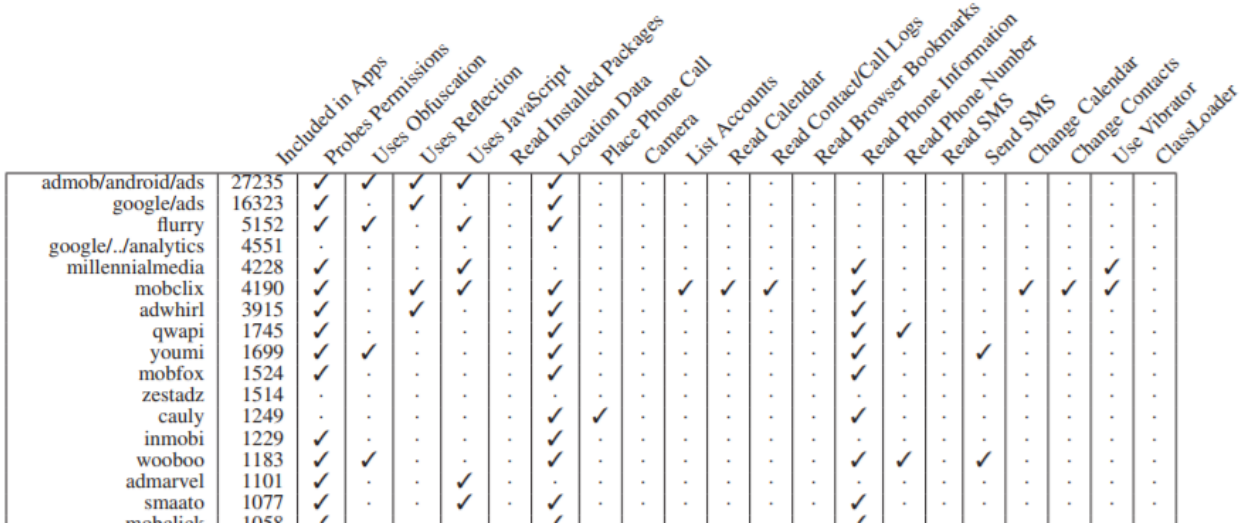


- Ad Services can also add requests to app's Android Manifest file
- Total permissions an app's AndroidManifest.xml

  = permissions requested by app **+ permissions requested by ad service**

Rogue? Ad Services

- Google is careful about permissions requested by AdMob
- Some other mobile ad libraries require more permissions:
  - Access location data, camera, account details, calendar, call logs, browser bookmarks, contact lists, phone information, phone number, SMS, etc
  - Make phone calls, send SMS messages, vibrate
  - Change calendar and contacts

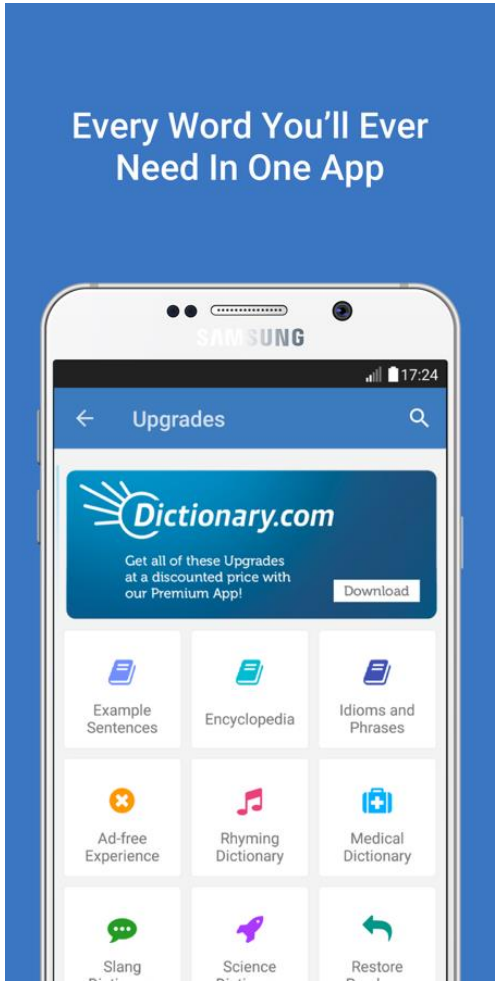


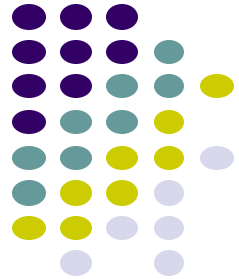
		Included in Apps	Probes Permissions	Uses Obfuscation	Uses Reflection	Uses JavaScript	Read Installed Packages	Location Data	Place Phone Call	Camera	List Accounts	Read Calendar	Read Contact/Call Logs	Read Browser Bookmarks	Read Phone Information	Read Phone Number	Read SMS	Send SMS	Change Calendar	Change Contacts	Use Vibrator	ClassLoader
admob/android/ads	27235	✓	✓	✓	✓	·	✓	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
google/ads	16323	✓	·	✓	·	·	✓	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
flurry	5152	✓	✓	·	✓	·	✓	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
google/../analytics	4551	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
millennialmedia	4228	✓	·	·	✓	·	·	·	·	·	·	·	·	✓	·	·	·	·	·	✓	·	·
mobclix	4190	✓	·	✓	✓	·	✓	·	·	✓	✓	✓	·	✓	·	·	·	✓	✓	✓	·	·
adwhirl	3915	✓	·	✓	·	·	✓	·	·	·	·	·	·	✓	·	·	·	·	·	·	·	·
qwapi	1745	✓	·	·	·	·	✓	·	·	·	·	·	·	✓	✓	·	·	·	·	·	·	·
youmi	1699	✓	✓	·	·	·	✓	·	·	·	·	·	·	✓	·	·	✓	·	·	·	·	·
mobfox	1524	✓	·	·	·	·	✓	·	·	·	·	·	·	✓	·	·	·	·	·	·	·	·
zestadz	1514	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
cauly	1249	·	·	·	·	·	✓	✓	·	·	·	·	·	✓	·	·	·	·	·	·	·	·
inmobi	1229	✓	·	·	·	·	✓	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
wooboo	1183	✓	✓	·	·	·	✓	·	·	·	·	·	·	✓	✓	·	✓	·	·	·	·	·
admarvel	1101	✓	·	·	✓	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
smaato	1077	✓	·	·	✓	·	✓	·	·	·	·	·	·	✓	·	·	·	·	·	·	·	·
mobclick	1058	✓					✓							✓								

Ref: Unsafe exposure analysis of mobile in-app advertisements
[M. Grace, W. Zhou, X. Jiang, A.-R. Sadeghi; WiSec 2012]

Final Words: Mobile Ad Services

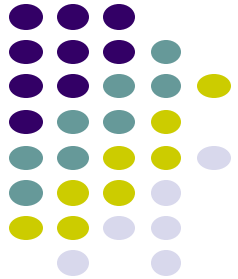
- Many apps use multiple ad services
  - Angry Birds app (a game) includes 7+ ad services
- Example of rogue requests:
  - One version of the Dictionary.com app requests permissions to **monitor phone calls** and **access location**





Android Analysis Tools

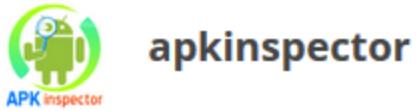
Analyzing Android Apps



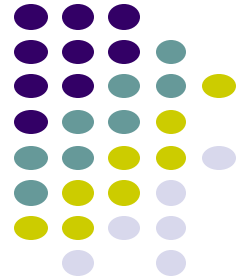
- Attacker can use analysis tools to get more information about an Android app
- **Source code recovery:** generate app source code from executable
- **Static analysis (binaries or source code):** Understand app design without running it.
  - Examine application logic, flow, APIs used
- **Dynamic analysis:** Observe how app executes
  - App memory usage, network usage, response time, performance, etc
- Many available (open source?) tools for all of the above!

Android Analysis Tools

- APKinspector
- Androguard
- AndroBugs
- Qark
- Epicc / IC3
- FlowDroid
- DidFail
- DroidBox
- MobSF



- **Scary!!**



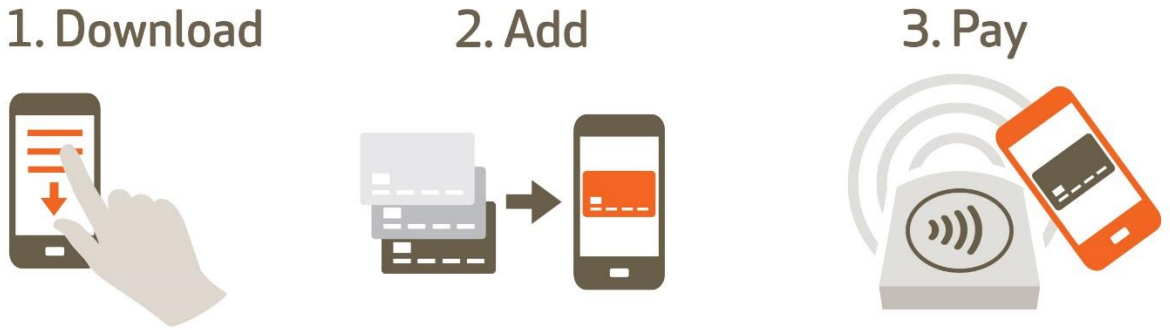
Android Pay using NFC

Android Pay

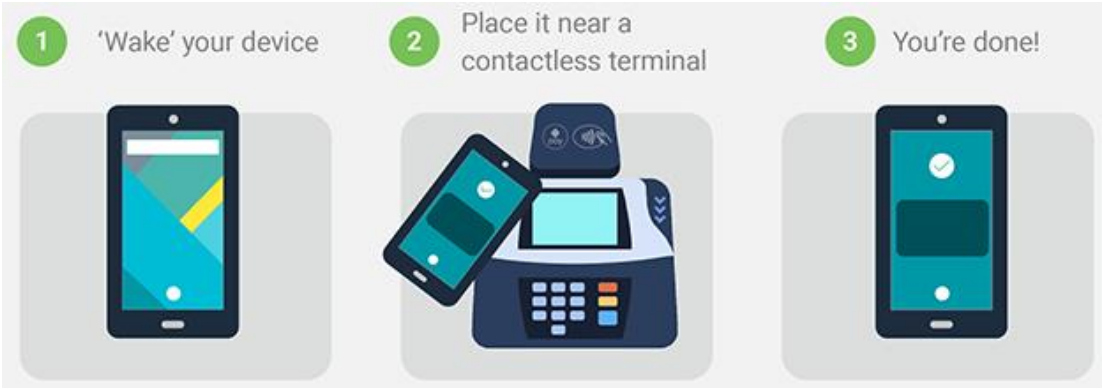
- Google Wallet → Android Pay (Sept 2015 initial release)
- **Vision:** Use smartphone to pay in stores
- E.g. Pay for donuts at Dunkin Donuts
- Easier way to track expenses, get rewards
  - Integrates with financial apps (banking, personal finance, etc)

How Android Pay Works

- First need to download Android Pay app, add credit cards

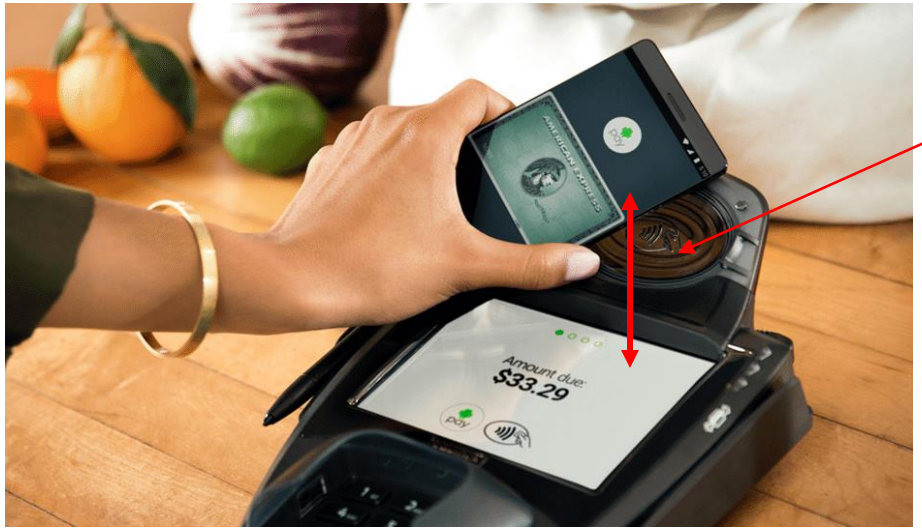


- To pay, place smartphone near Android pay terminal

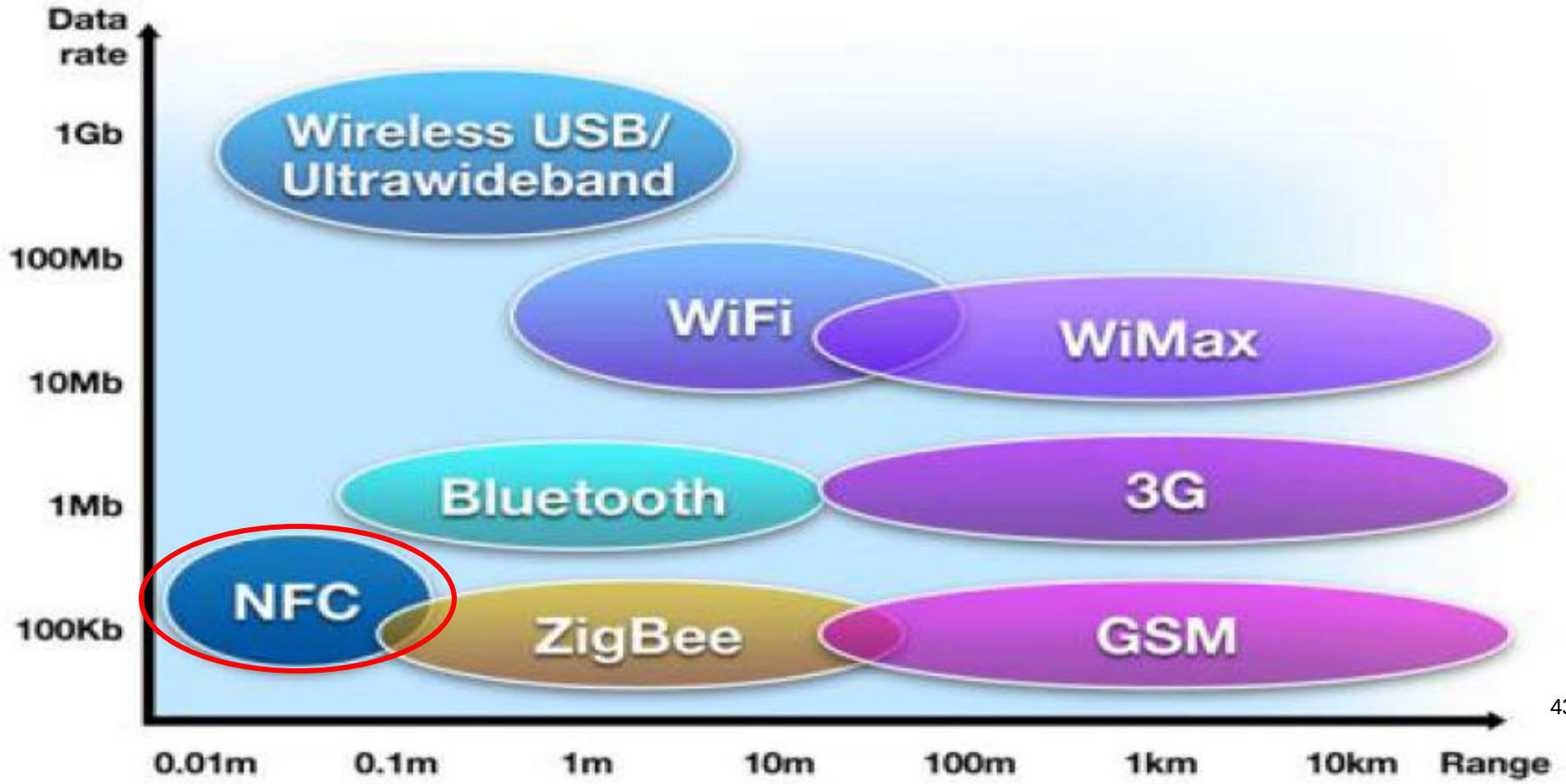


Mobile Pay Uses NFC

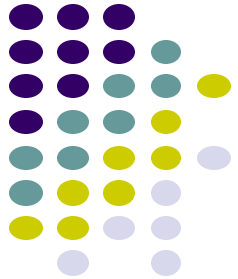
- Mobile payment (e.g. Android Pay) typically uses NFC for transaction
- NFC: Near Field Communication: short-range, low-rate wireless
  - For communication between devices in close proximity
- Utilized by many smartphone mobile pay systems (e.g. Google Pay)
  - E.g. pay at Dunkin donuts



Wireless Comparison



NFC: Short range, low bitrate



Android Pay using NFC

- Proximity makes it easier to verify payee
- **Convenient:** store all credentials inside the phone
- Integrates with other mobile services: eBooks, music downloads, barcodes, etc. (easier payments)

Types of NFC Devices

- **Active Device: E.g. Smartphone**
  - Can read + send
  - Can read information from target and also send information to target
  - **2-way** communication possible

- **Passive Device: E.g. NFC tag**
  - Cannot send, can only be read
  - Information on passive device can only be read.
  - Cannot initiate communication

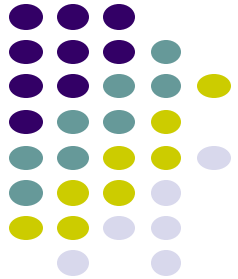
NFC Modes of Interaction

- **Reader/Writer:**
  - Active NFC device reads/writes from/to passive NFC tag (One way)

- **Peer-to-Peer:**
  - Active NFC devices interact with each other bi-directionally
  - Take turns being active vs passive

- **Card Emulation:**
  - An NFC device emulates a passive NFC tag that is read by an active NFC device

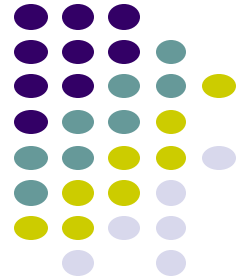
NFC Security / Threats



- NFC has similar threats as other wireless communications
  - Eavesdropping
  - Data modification / insertion / corruption
  - Man-in-the-middle attacks (attacker alters communication between 2 devices)

- Eavesdropping: Another device listening to transaction
  - NFC itself provides no explicit protection against eavesdropping
  - Passive exchange < 1m between devices, active exchange < 10m
  - Harder to eavesdrop on passive exchange due to shorter range

Data Modification & Injection



- Attacker modifies bits in flight e.g., flip 0s to 1s

- **Data Injection:**
  - Attacker responds faster than intended target
  - Possible defenses:
    - Secure handshake w/ verifiable response

- MitM is difficult in NFC due to:
  - Close proximity requirement (MitM needs to be closer than tag)
  - Attacker can use sheet of Aluminum to block legitimate sender