What, exactly, is different or new about mobile security?

Dan S. Wallach, Rice University
The “computers inside the computer”

Every chip has one or more CPUs inside; they have exploitable bugs

Usability issues

Smaller screens mean fewer security indicators

The death of app isolation

Apps have full Internet access, sensitive privileges, and abuse them

Threat models: physical attacks

Or, defending against the San Bernadino iPhone attack
The computers inside your computer
Have you looked inside a phone lately?

Each chip has an embedded CPU, typically ARM

“Firmware” (i.e., software) baked in by vendor, not part of the OS distribution

(Google Pixel photos via iFixit)
Example: SD card firmware

Flash storage is incredibly complicated
High defect rates, wear leveling / block remapping, etc.
Allow a vanilla filesystem, designed for a hard drive, to “just work”

Cheaper to use a general-purpose CPU
Testing (defect mapping, binning) and runtime (load leveling, remapping) all done in software
Even if 80% of blocks are dead, can still sell as a lower-capacity card
Quality-control issues?

Andrew “Bunnie” Huang designed the Chumby

“I realized that all the units failing [in quality control] had Kingston microSD cards from a particular lot code.” (2009)
Quality-control issues?

Andrew “Bunnie” Huang designed the Chumby

“I realized that all the units failing [in quality control] had Kingston microSD cards from a particular lot code.” (2009)

“One [Shenzhen] vendor ... interested me; it was literally a mom, pop and one young child sitting in a small stall of the mobile phone market, and they were busily slapping dozens of non-Kingston marked cards into Kingston retail packaging. They had no desire to sell to me, but I was persistent; this card interested me in particular because it also had the broken ‘D’ logo but no Kingston marking.”
Counterfeit analysis

Bunnie bought a bunch of cheap SD cards in Shenzhen

“Normal”: OEM Toshiba

“Sketchy”: alternate OEM codes, etc.

Conclusion: Kingston resells lower-quality parts at tight margins
Counterfeit analysis

Bunnie bought a bunch of cheap SD cards in Shenzhen

“Normal”: OEM Toshiba

“Sketchy”: alternate OEM codes, etc.

Conclusion: Kingston resells lower-quality parts at tight margins

“Larger vendors will tend to offer more consistent quality, but even the largest players staunchly reserve the right to mix and match flash chips with different controllers, yet sell the assembly as the same part number — a nightmare if you’re dealing with implementation-specific bugs.”
SD firmware hacking

Bunnie and Sean “Xobs” Cross (2013)

Discovered firmware update command

Able to send 8051 machine code (no code signing, etc.)

MITM attacks from your storage?!
Bunnie and Sean “Xobs” Cross (2013)

Discovered firmware update command
Able to send 8051 machine code (no code signing, etc.)

MITM attacks from your storage?!

“It’s as of yet unclear how many other manufacturers leave their firmware updating sequences unsecured.”
Same thing for your networking chips

Modern network chips have embedded CPUs as well

Support “full stack” WiFi

Don’t interrupt the CPU as often

Exploitable from the outside!

No use of protection bits: every page is RWX (also no stack cookies, etc.)

(Source: Gal Beniamini, Google Project Zero, googleprojectzero.blogspot.com/2017/04/over-air-exploiting-broadcoms-wi-fi_4.html)
Attacking the main CPU from the NIC

Option 1: Attack the OS kernel
Heap overflow, vulnerable code pointer

Option 2: Direct memory access
PCIe devices can do DMA
IOMMUs not used to limit visible memory in the kernel

Arbitrary read/write to the OS kernel

(Source: Gal Beniamini, Google Project Zero, googleprojectzero.blogspot.com/2017/04/over-air-exploiting-broadcoms-wi-fi_11.html)
What about ARM TrustZone?

TrustZone is something of an OS layer below the kernel

Support for boot locking, DRM, etc.

Of course, it’s exploitable

(Also discovered by Gal Beniamini)

memcpy() buffer overwrite vulnerability

Messy process to build a ROP chain

Shellcode to read/interact with the “secure file system”

bits-please.blogspot.com/2016/05/qsee-privilege-escalation-vulnerability.html
TrustZone security engineering?

MobileCore (Samsung)
No ASLR, no stack cookies

QSEE (Qualcomm): slightly better
9-bit ASLR, no guard page between stack, BSS, heap

Trustlets: Proprietary code, bugs can linger
Many trustlets directly exposed to userland through proxy services

(Source: Gal Beniamini talk, BlueHat Israel 2017, microsoftrnd.co.il/Press%20Kit/BlueHat%20IL%20Decks/GalBeniamini.pdf)
Example: Android Full Disk Encryption

KeyMaster app manages keys

Vulnerabilities in other trustlets

Privilege escalation

Lack of separation across trustlets

Master keys can leak

Qualcomm, others support hardware-fused keys

Not currently used by KeyMaster

Maybe in Android “O”?
Kernel bugs increasingly targeted

What kinds of bugs?

- Missing/incorrect bounds check
- Null pointer dereference
- Information leak
- Missing permission check
- Use after free
- Race condition
- Memory corruption (other)
- Other
- Integer overflow
- Uninitialized data

44.8%

Missing/incorrect bounds check
Addressed by hardened usercopy, backported to Android kernel 3.18+

If we used a safe programming language, plenty of PL and systems research that addresses these remaining concerns!
All the computers inside the computer are vulnerable.

All the same attack types (buffer overflow, heap grooming, ROP, etc.)

Less competitive pressure → less use of standard defenses

OS kernels tend to trust their devices to act reasonably.

An “evil component” has a large attack surface

IOMMUs can help limit this

Unclear whether vendor isolation layer (Android “O” Treble) will help
Challenges so far

All the usual vulnerabilities that come from C programming.
Can we please get rid of C? Is Rust a good alternative?
At least most Android apps and many system services are in Java.

**Vulnerability discovery, patch delivery.**
If Beniamini can do it, so can others. Are similar vulns being exploited?

**Supply chain integrity.**
Are you even getting the chips you expect?
The death of app isolation
Default security policies

Every web page has an origin (DNS name, protocol, etc.)
Separation enforced by browser’s same origin policy
Network connections limited (unless the receiving server allows it)
Limited visibility of native OS resources

Android apps have private storage, but unlimited networking
Scan your internal network? Why not?
Easy to abuse privileges
Example: exfiltration of contacts list

The Wrong Way: Path Uploads iOS Users' Address Books Without Permission

Tuesday, February 7th, 2012

What started as a bit of aimless tinkering for developer Arun Thampi ultimately unearthed something very surprising about personal life-sharing service Path. As a fan of the app, Thampi took it upon himself to look at the API calls that the app made to Path’s service and found that his “entire address book (including full names, emails and phone numbers) was being sent as a plist to Path.”

Puzzled, Thampi created an entirely new Path and tried again, only to be faced with the same results. Feel free to try it for yourself if you’re curious, as Thampi has written up the test procedures on his blog.

According to a comment left by Path co-founder and CEO Dave Morin, uploading the user’s address book is meant simply to connect users with each other. As VentureBeat points out, this isn’t exactly a secret — the practice is pointed out in the company’s Wikipedia entry. Still, it’s not exactly the easiest information to come across unless you’re actively looking for it, especially when no mention of it is made during the initial sign-up process.
When asked why Path didn’t give users the choice to opt-in right from the start, [Path CEO] Morin responded with the following:

*This is currently the industry best practice and the App Store guidelines do not specifically discuss contact information. However, as mentioned, we believe users need further transparency on how this works, so we’ve been proactively addressing this.*

ADS!
dynamo

dynamo [dahy-nuh-moh]

—noun, pl. -mos.

1. an electric generator, especially for direct current.
2. an energetic, hardworking, forceful person.

Origin: 1882; short for dynamoelectric

Cost: Free

dynamo

dynamo [dahy-nuh-moh]

—noun, pl. -mos.

1. an electric generator, especially for direct current.
2. an energetic, hardworking, forceful person.

Origin: 1882; short for dynamoelectric

Cost: $2.99
dynamo

dynamo [dahy-nuh-moh]

—noun, pl. -mos.

1. an electric generator, especially for direct current.
2. an energetic, hardworking, forceful person.

Origin: 1882; short for dynamoelectric

dynamo-

1. variant of dyna-:
   dynamometer.

Also dynam-

Cost: Free

Cost: $2.99
Downloads: 100,000 – 500,000
dynamo

*dynamo* [dahy-nuh-moh]


1. an electric generator, especially for direct current.
2. an energetic, hardworking, forceful person.

Origin: 1882; short for *dynamoelectric*

dynamo-

1. variant of dyna-:
   *dynamometer.*

Also dynam-.

Cost: **Free**
Downloads: 10,000,000 – 50,000,000

Cost: **$2.99**
Downloads: 100,000 – 500,000
Ads are widely used
Ads are widely used

(and advertising uses 75% of the power budget - Pathak et al., Eurosys 2012)
Dubious Android apps may not be malware--just ads

Verizon-affiliated ICSA Labs steps into the controversy over Android apps that Symantec identified as malware.

by Elenor Mills  |  February 1, 2012 1:21 PM PST

Deal or BE Millionaire
Ogre Games

More from developer
Measuring permission usage

Separate library code from application code

Simple static analysis of library code

Stowaway (Felt et al., 2011)

Map API calls to Android permissions

Scout (Au et al., 2012)


Retrieve ads
Report usage
Vibrate
Notifies you about important ads!
Wake Lock
Video API calls
"Dangerous" Collection of Permissions
“Dangerous” Permissions
“Dangerous” Permissions

Get Tasks
See what else is running
“Dangerous” Permissions

Read History and Bookmarks
What are your favorite web pages?
“Dangerous” Permissions

Get Accounts
your Google ID... and Facebook, too!
“Dangerous” Permissions

Read
Contacts
Getting to know you...
“Dangerous” Permissions

Change WiFi State
Load those video ads!
"Dangerous" Permissions

Record Audio
Just listening!
“Dangerous” Permissions

Camera

Smile!
Nearly 60K Low-Quality Apps Booted From Google Play Store In February, Points To Increased Spam-Fighting

Google has stepped up its efforts to remove spammy or otherwise non-compliant applications from its mobile application marketplace, Google Play. In recent weeks, app deletions hit a record high in February, with 60,000 apps removed during the course of the month – the largest round of app deletions to date. The news of this massive app removal comes just ahead of the rumored launch of Google Play Pass, which could open the floodgates for even more thefts.
# Google’s actions vs. ad library

<table>
<thead>
<tr>
<th>Ad Library</th>
<th>Percent of Apps Removed</th>
</tr>
</thead>
<tbody>
<tr>
<td>EverBadge</td>
<td>60.5%</td>
</tr>
<tr>
<td>Hunt Mobile</td>
<td>45.5%</td>
</tr>
<tr>
<td>AirPush</td>
<td>40.7%</td>
</tr>
<tr>
<td>SendDroid</td>
<td>31.2%</td>
</tr>
<tr>
<td>Waps</td>
<td>29.7%</td>
</tr>
<tr>
<td>TapIt</td>
<td>28.4%</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>11.6%</strong></td>
</tr>
</tbody>
</table>
Ad libraries have sensitive APIs

<table>
<thead>
<tr>
<th>Classification</th>
<th>API Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keywords</td>
<td>void setKeywords(String)</td>
</tr>
<tr>
<td>Keywords</td>
<td>void setSearchString(String)</td>
</tr>
<tr>
<td>Gender</td>
<td>void setGender(GenderType)</td>
</tr>
<tr>
<td>Location</td>
<td>void setCurrentLocation(Location)</td>
</tr>
<tr>
<td>Age</td>
<td>void setAge(int)</td>
</tr>
<tr>
<td>Multiple Factors</td>
<td>void setRequestParams(Map)</td>
</tr>
<tr>
<td>Postal Code</td>
<td>void setPostalCode(String)</td>
</tr>
<tr>
<td>Enable Location</td>
<td>void setLocationInquiryAllowed(boolean )</td>
</tr>
<tr>
<td>Income</td>
<td>void setIncome(int)</td>
</tr>
<tr>
<td>Interests</td>
<td>void setInterests(String)</td>
</tr>
<tr>
<td>Area Code</td>
<td>void setAreaCode(String)</td>
</tr>
<tr>
<td>Education</td>
<td>void setEducation(EducationType)</td>
</tr>
<tr>
<td>Ethnicity</td>
<td>void setEthnicity(EthnicityType)</td>
</tr>
</tbody>
</table>

Table 1: Privacy-related API calls found in the InMobi API

Goal: enumerate use of these APIs in top libraries from large corpus of Android apps
Calls vs. Install Count
Calls vs. Install Count

Popular apps benefit from additional revenue
Calls vs. Install Count

Top apps can’t get away with misbehavior

Popular apps benefit from additional revenue
Fine, I’ll just deny them permissions
Fine, I’ll just deny them permissions
Fine, I’ll just deny them permissions
Fine, I’ll just deny them permissions
The OS should provide privacy features

Cyanogen / LineageOS have a “PrivacyGuard” feature

Example: Provides a contacts list with zero entries
To root or not to root...

Rooted phones can install ad blockers (e.g., AdAway)
More control, better security

Rooted phones can violate DRM
Also, malicious apps can abuse superuser privs
Game cheats as well

Android “O” attestation features effectively block rooting
Google Plans Ad-Blocking Feature in Popular Chrome Browser

Filter could strip out ads that provide bad experiences for users

Alphabet Inc.'s Google is planning to introduce an ad-blocking feature in the mobile and desktop versions of its popular Chrome web browser, according to people familiar with the company's plans.

The ad-blocking feature, which could be switched on by default within Chrome, would filter out certain online ad types deemed to provide bad experiences for users as they move around the web.

By Jack Marshall
Updated April 19, 2017 7:18 p.m. ET

What about Android-native ad libraries?
Summary so far

Advertising-supported free apps want to make money
More user information = more money

**OS permission requests only partially protect users**
Some apps really do need to read your contacts or learn your location
Some apps *refuse to run if you deny them permissions*

Very little that third-party researchers can do here
Usability: trusted path
Old-school idea: trusted path

Unforgeable labels
Prevent apps from spoofing one another

Trusted user input paths
Uninterruptible path for user to speak to the system
(Example: Ctrl-Alt-Del in older Windows NT for login.)

Screenshot: Compartmented Mode Workstation (early 1990's)
GDC4S SME PED

- Color Touchscreen Display
  2.8 in TFT QVGA

- 5-Way Navigation Key
  Joystick navigates & selects lists and menus, unlock keyboard

- Application Launch Keys
  Messaging, Windows Start, Context, Web Browser

- Phone / Send Key
  Initiate, place and answer phone calls

- QWERTY Keyboard
  Full QWERTY keyboard & phone dialpad with backlighting

- Unclassified Key
  Selects Unclassified PDA

- Earpiece
  Used for phone calls when using handset

- LEDs for SCIF-Friendly, Alert & Unlocked
  - SCIF-Friendly mode activated
  - Alerts for network connection, incoming calls, messages, appointments
  - Unlocked (PIN entered)

- Power / End Key
  Turn device on/off, exits to previous menu and end phone calls

- Classified Key
  Selects Classified PDA or Security Menu

- Trusted Display
  Security menus
Trusted path features

Separate display, managed by crypto module
Trusted path features

Separate display, managed by crypto module

Dedicated mode selectors
OAuth phishing

We want to hide security indicators
Users probably wouldn’t notice, even if prominent

Google’s solution?
Better anti-spam features
“Google” in name now special
OAuth phishing

We want to hide security indicators

Users probably wouldn’t notice, even if prominent

Google’s solution?

Better anti-spam features

“Google” in name now special
Phishing on mobile

Web browsers try to get out of the way
Less chance for chrome context to help you

Apps are, by nature, full-screen
Home button is still a “trusted path” feature
(Not that this is obvious to users.)

Central control from app stores can help
Misbehaving apps will be globally uninstalled!
Experts have been warning for years about security blunders in the Signaling System 7 protocol – the magic glue used by cellphone networks to communicate with each other. These shortcomings can be potentially abused to, for example, redirect people’s calls and text messages to miscreants’ devices. Now we’ve seen the first case of crooks exploiting the design flaws to line their pockets with victims’ cash.

O2-Telefónica in Germany has confirmed to Süddeutsche Zeitung that some of its customers have had their bank accounts drained using a two-stage attack that exploits

3 May 2017 at 20:02, Iain Thomson

After years of warnings, mobile network hackers exploit SS7 flaws to drain bank accounts

O2 confirms online thefts using stolen 2FA SMS codes

Maybe two-factor auth will help?
And pairing is a huge problem

Long, complicated instructions

Nest Protect: scan QR code

Nest Thermostat: dial in your WiFi password

Rachio / Electric Imp: screen flashing to a light sensor

Needs to be easier!

Scan the QR code

The Nest app will turn on your phone or tablet’s camera. Use it to scan the QR code on the back of your Nest Protect.

The Nest app will automatically recognize it.

My phone’s camera won’t scan the QR code, what should I do? >
Threat models
“I’m still clinging to my BlackBerry,” Mr. Obama said Wednesday [7 Jan ’09]. “They’re going to pry it out of my hands.”

The New York Times
In person vs. remote attacks

Do we need to defend devices against “local” threats?

- Storage encryption?
- Fingerprint vs. PIN?
- Privacy from shoulder surfing
- Privacy from gov’t search
- Radio emissions?
Whose job is it to protect you?

The hardware vendor? The OS vendor? The chipset vendor?

What about your cloud services?

Can the government compel a vendor to add a backdoor?

Who provides ongoing security updates?

Example: Mirai webcam botnet
Internet Giants Erect Barriers to Spy Agencies

By DAVID E. SANGER and NICOLE PERLROTH  JUNE 6, 2014

MOUNTAIN VIEW, Calif. — Just down the road from Google's main campus here, engineers for the company are accelerating what has become the newest arms race in modern technology: They are making it far more difficult — and far more expensive — for the National Security Agency and the intelligence arms of other governments around the world to pierce their systems.

As fast as it can, Google is sealing up cracks in its systems that Edward J. Snowden revealed the N.S.A. had brilliantly exploited. It is encrypting more data as it moves among its servers and helping customers encode their own emails. Facebook, Microsoft and Yahoo are taking similar steps.

After years of cooperating with the government, the immediate goal now is to thwart Washington — as well as Beijing and Moscow. The strategy is also intended to preserve business overseas in places like Brazil and Germany that have threatened to entrust data only to local providers.

Google, for example, is laying its own fiber optic cable under the world's oceans, a project that began as an effort to cut costs and extend its influence, but now has an added purpose: to assure that the company will have more control over the movement of its customer data.
Internet Giants Erect Barriers to Spy Agencies

By DAVID E. SANGER and NICOLE PERLROTH  JUNE 6, 2014

MOUNTAIN VIEW, Calif. — Just down the road from Google’s main campus, engineers for the company are accelerating what has become the newest arms race in modern technology: They are making it far more difficult — and far more expensive for the National Security Agency and the intelligence agencies of the United Kingdom, France, Germany and others to spy on them.

Eric Grosse, Google’s security chief, suggested in an interview that the N.S.A.’s own behavior invited the new arms race.

“I am willing to help on the purely defensive side of things,” he said, referring to Washington’s efforts to enlist Silicon Valley in cybersecurity efforts. “But signals intercept is totally off the table,” he said, referring to national intelligence gathering.

“No hard feelings, but my job is to make their job hard,” he added.
Open challenges
Ease of use

Internet of Things are hard to install
Pre-installed trust (at purchase time)?

Power user features vs. security lockdown
Apple: one app store
Google: you can install a 3rd-party store
The computers inside the computer

*Disaggregated computing:* Our definition of a computer is changing

Embedded computers need to be exposed, managed

**Nasty challenges**

What should it mean to “boot” a computer?

What does it mean to not trust one of your own devices?

How to protect vendor “intellectual property”? 
Code correctness

Buffer overflows have been known since the 1980’s, maybe earlier.
We have tools that try to make C safe (e.g., Coverity)
Inherently safe systems tend to require GC memory (e.g., Java)

Maybe it’s time to go with something else?
Even tiny embedded CPUs are insanely fast and have lots of RAM*

* If you’re old enough to remember the bad old days.
Code correctness

Buffer overflows have been known since the 1980's, maybe earlier. We have tools that try to make C safe (e.g., Coverity). Inherently safe systems tend to require GC memory (e.g., Java). Maybe it's time to go with something else?

Even tiny embedded CPUs are insanely fast and have lots of RAM*.

Redox OS: written from scratch in Rust.

* If you're old enough to remember the bad old days.
We’ve got a lot of work to do