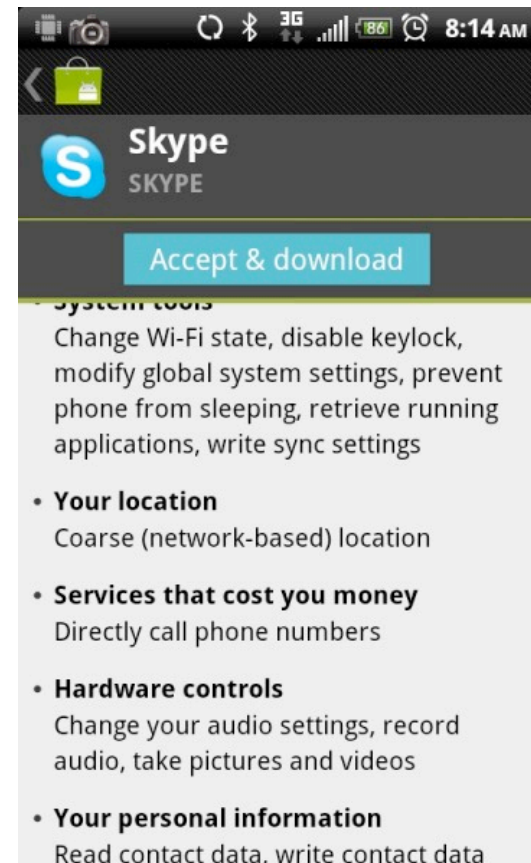


# ANDROID PERMISSIONS DEMYSTIFIED

**Adrienne Porter Felt**, Erika Chin,  
Steve Hanna, Dawn Song, David Wagner  
University of California, Berkeley

# ANDROID PERMISSIONS

- Apps are low-privilege by default
- Developers declare permissions upfront
- Users see permissions at installation



# POTENTIAL BENEFITS

- **Defense in depth**
  - Mitigate application vulnerabilities
- **User consent**
  - Users know the risks before installation

# LEAST PRIVILEGE

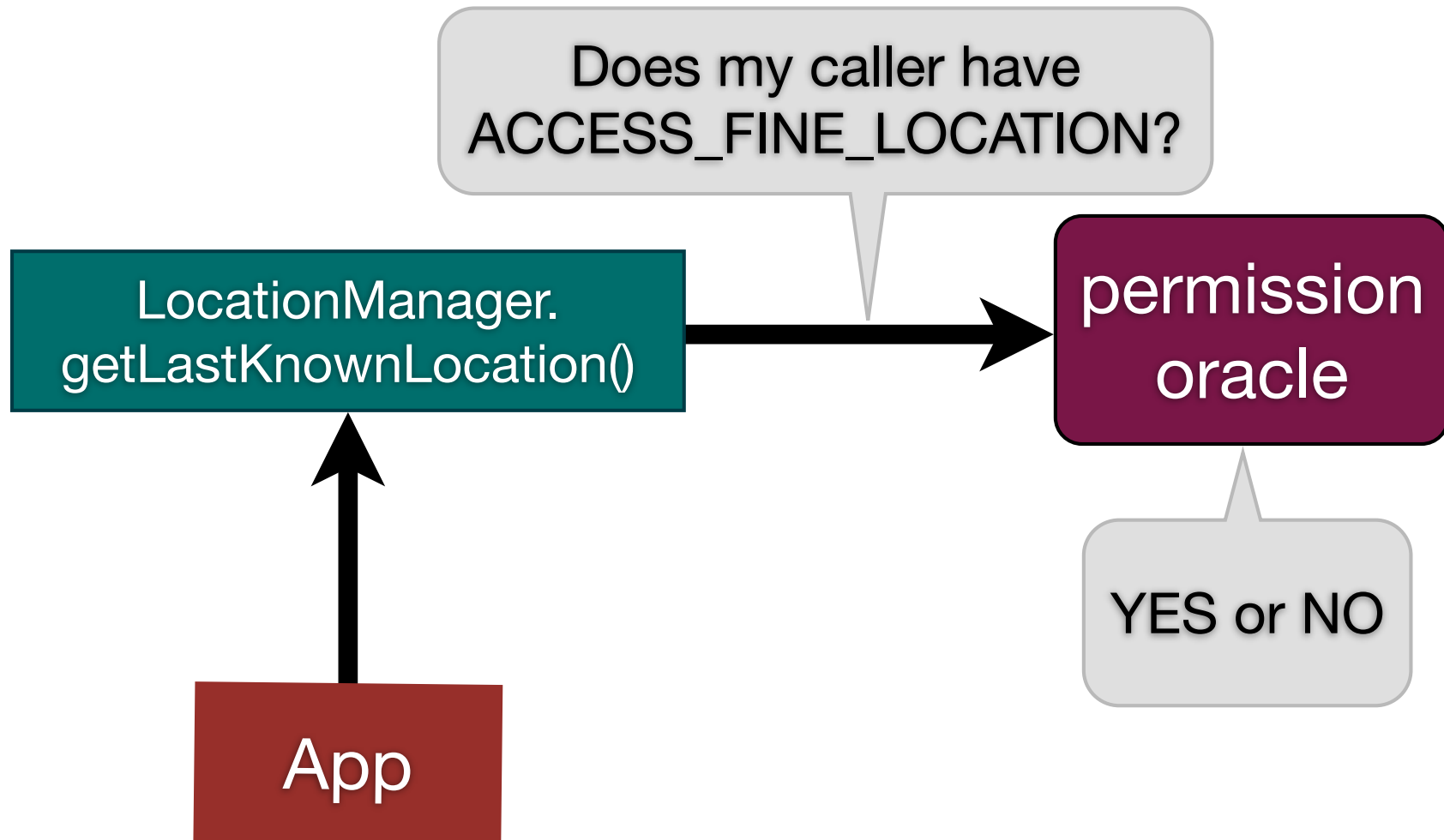
- Fewer permissions → more effective permission system
- Ideally, developers follow least privilege
- **Do Android developers follow least privilege in practice?**

# THIS TALK

- Our approach to overprivilege analysis
- Static analysis of applications
- Permission map
- Real-world overprivilege

# OUR APPROACH TO OVERPRIVILEGE

# ROLE OF PERMISSIONS



# OVERPRIVILEGE

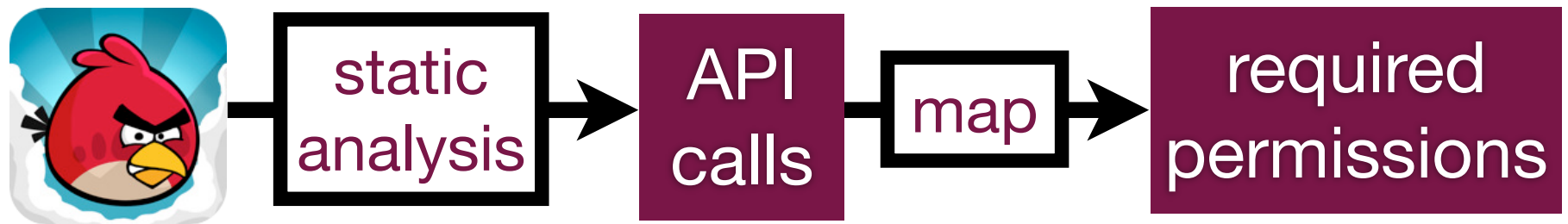
- **Least-privileged app:**  
asks only for the permissions it needs to make API calls
- **Overprivileged app:**  
asks for additional permissions

# STOWAWAY



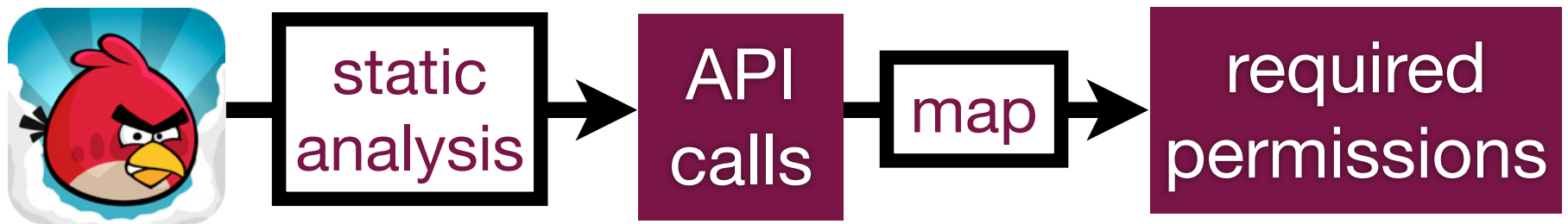
1. Find all the API calls made by an app

# STOWAWAY



2. Map API calls to their permissions

# STOWAWAY



3. Compare required and requested permissions to determine overprivilege

# STATIC ANALYSIS OF APPLICATIONS

# FINDING API CALLS

- **Goal:** find all API calls in an app's code
- Regular API calls are simple to find
- **Reflective calls are a big challenge**
  - Dynamically specify the target of an invocation

# JAVA REFLECTION

```
Object vib = getSystemService("vibrator");
```

```
Class vibratorclass =
```

```
    Class.forName("android.os.Vibrator");  
    android.os.Vibrator.vibrate(300);
```

```
Method vibmeth = vibratorclass
```

```
    .getDeclaredMethod("vibrate");
```

```
vibmeth.invoke(vib, 300);
```

# REFLECTION IN ANDROID

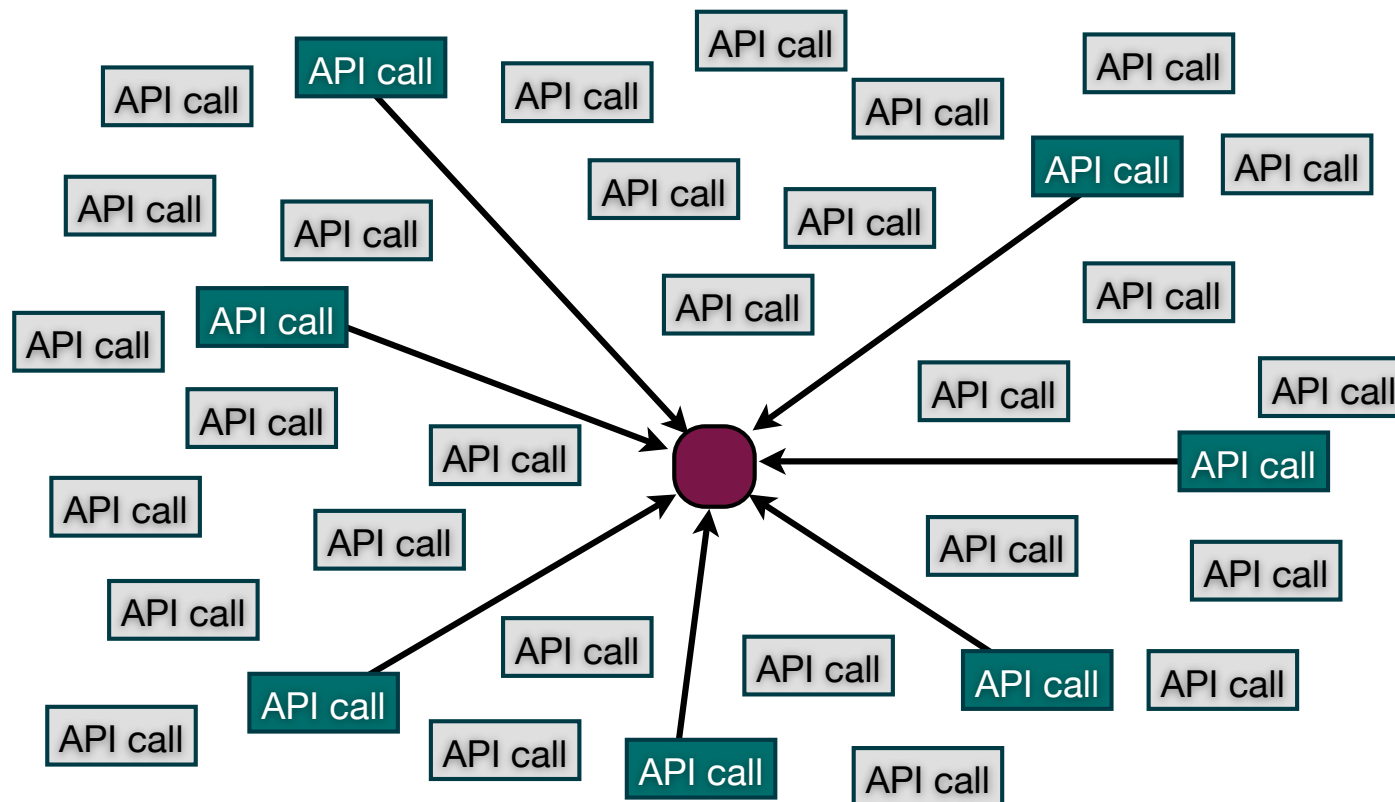
- Missing a reflective call might cause an error in the overprivilege analysis
- **61%** of apps make reflective calls
- Stowaway can't fully handle **12%** of apps

# PERMISSION MAP

# PERMISSION MAP

- **Next step:** map application's API calls to permissions
- But how do we know what API calls need what permissions?
  - No documentation or test suite

# PERMISSION CHECKS



**Queries to the oracle are scattered throughout the API implementation**

# STATIC ANALYSIS?

- Look for calls to the oracle in API code
- **Challenges:**
  - API implemented in both C++ and Java
  - IPC interface between apps & system
  - Linux groups for INTERNET, BLUETOOTH
  - Unreachable code

# STATIC ANALYSIS?

- Look for calls to the oracle in API code
- **Challenges:**
  - API implemented in both C++ and Java
  - IPC interface between apps & system
  - Linux groups for INTERNET, BLUETOOTH
  - Unreachable code

# DYNAMIC APPROACH

- Modified oracle to log invocations
- Executed and logged API calls

`BluetoothDevice.<init>(String)`

`BluetoothDevice.getServiceChannel(ParcelUuid)`

`android.permission.BLUETOOTH`

# COVERAGE

## Randoop

- Feedback-directed testing
- 60% API coverage



## Custom tool

- Tests corrected by human supervisor
- +25% coverage

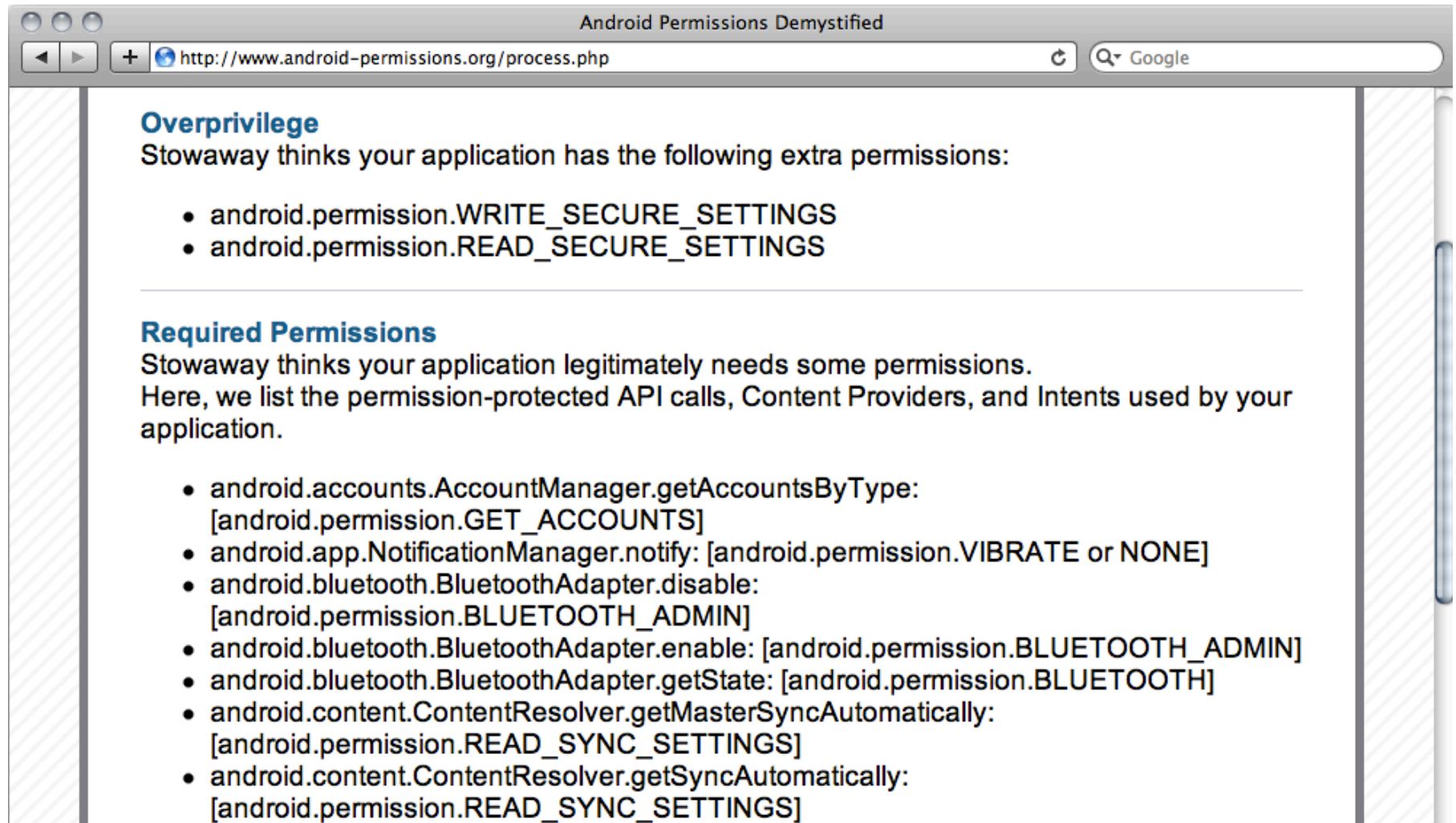


# RESULTS

- **Dynamic testing:** **1,259** API calls with permission checks
- **Documentation:** **78** API calls with permission checks
  - 8% error rate

# REAL-WORLD OVERPRIVILEGE

# STOWAWAY SERVICE



Android Permissions Demystified

http://www.android-permissions.org/process.php

Google

## Overprivilege

Stowaway thinks your application has the following extra permissions:

- android.permission.WRITE\_SECURE\_SETTINGS
- android.permission.READ\_SECURE\_SETTINGS

---

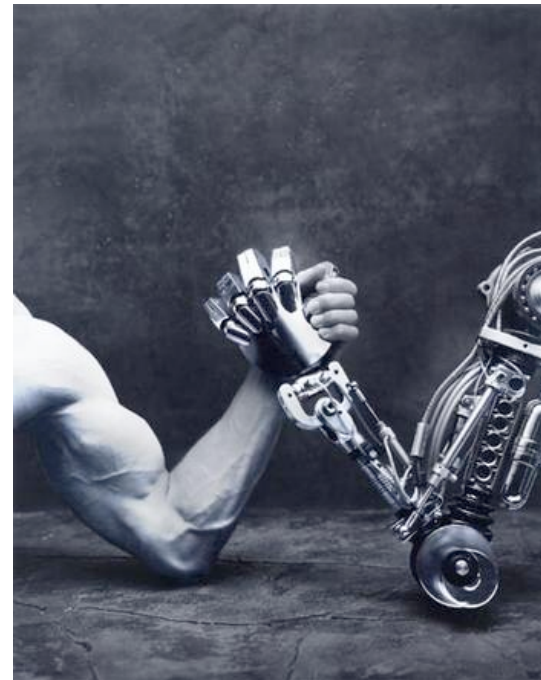
## Required Permissions

Stowaway thinks your application legitimately needs some permissions. Here, we list the permission-protected API calls, Content Providers, and Intents used by your application.

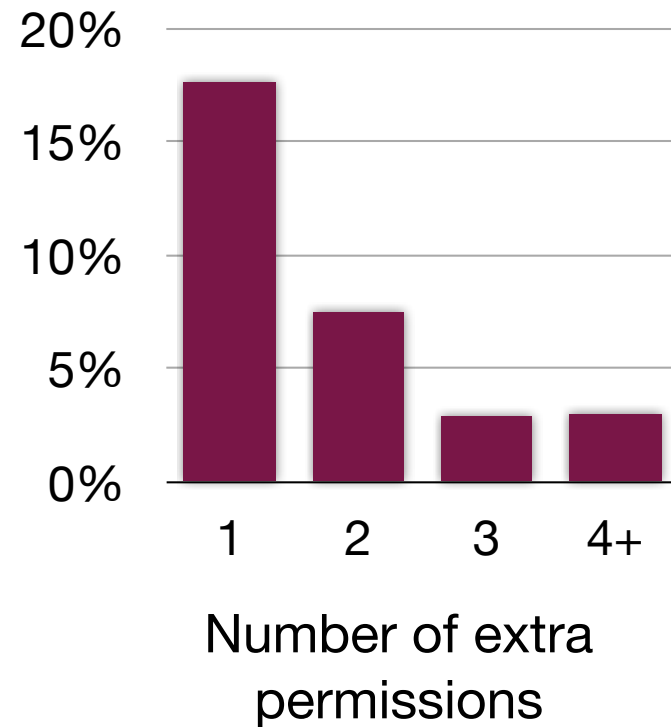
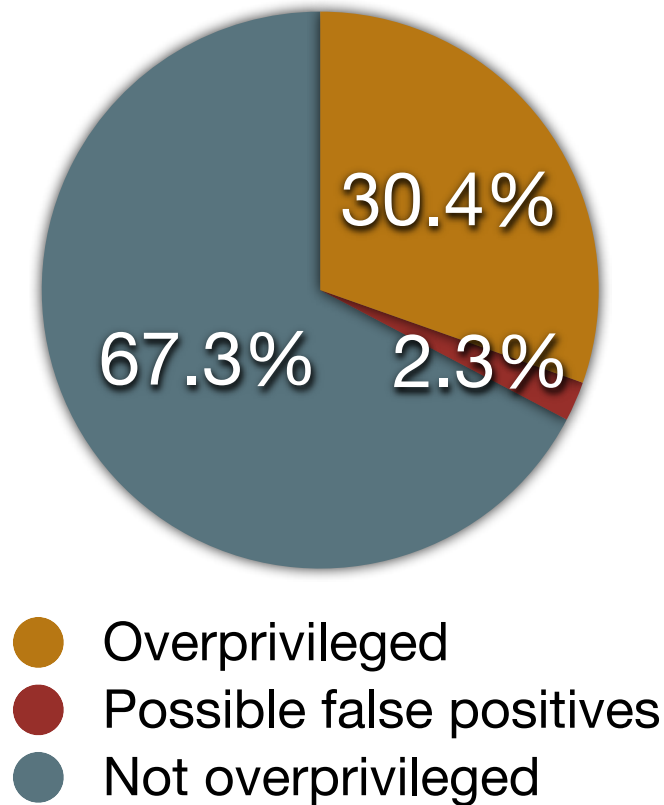
- android.accounts.AccountManager.getAccountsByType: [android.permission.GET\_ACCOUNTS]
- android.app.NotificationManager.notify: [android.permission.VIBRATE or NONE]
- android.bluetooth.BluetoothAdapter.disable: [android.permission.BLUETOOTH\_ADMIN]
- android.bluetooth.BluetoothAdapter.enable: [android.permission.BLUETOOTH\_ADMIN]
- android.bluetooth.BluetoothAdapter.getState: [android.permission.BLUETOOTH]
- android.content.ContentResolver.getMasterSyncAutomatically: [android.permission.READ\_SYNC\_SETTINGS]
- android.content.ContentResolver.getSyncAutomatically: [android.permission.READ\_SYNC\_SETTINGS]

# FALSE POSITIVES

- Reviewed 40 apps, and compared to Stowaway
- 7% false positive rate



# OVERPRIVILEGE RATES



Set of 795 apps with no reflection errors

# COMMON ERRORS

- **Permission names**

- ACCESS\_WIFI\_STATE or ACCESS\_NETWORK\_STATE?

- **Deputies**

- Apps that ask the **browser** to open a web site do not need INTERNET



# RELATED WORK

## Dynamic

- ✓ Missed 3 API calls
- ✓ 0 unreachable permission checks

## Static

- Missed 247 API calls
- 143 unreachable permission checks

Static analysis from Bartel et al.

# CONCLUSION

- 1/3 of apps have extra permissions
  - Most developers try to follow least privilege
- [www.android-permissions.org](http://www.android-permissions.org)
  - Run Stowaway on your app
  - Permission map data