EECS 448 Smartphone Security

Android Permissions

Xusheng Xiao

Electrical Engineering and Computer Science
Case Western Reserve University

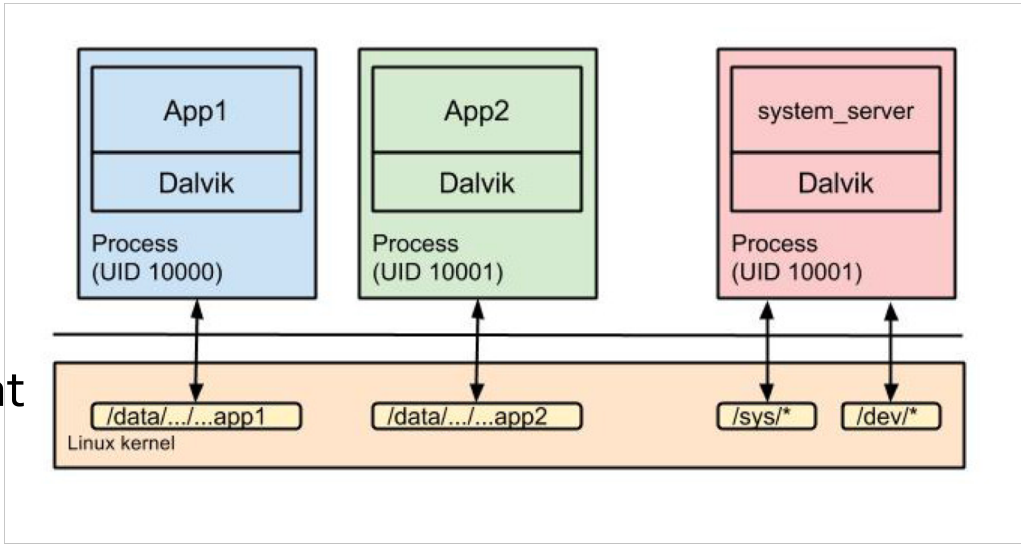
Last Lecture: Android Programming

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent">

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Hello World!"
       />

</RelativeLayout>
```

- **Layouts**: arranging UI objects automatically
- **Widgets**: building blocks to compose a user interface.
    - Show text or graphics
    - Buttons, text input controls, and checkboxes

App Sandbox

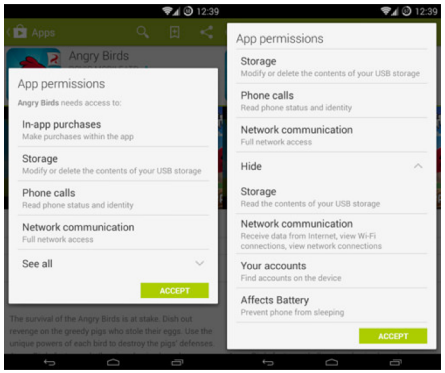
- Multi-user Linux system
- Each app is assigned a unique Linux user ID
  - User ID is unknown to the app
  - Only the user ID assigned to that app can access files in the app



- Each process has its own virtual machine (VM)
  - An app's code runs in isolation from other apps
- Every app runs in its own Linux process
  - Start a process for executing any of the app's components
  - Shut down the process when it's no longer needed or when the system must recover memory for other apps

Data Access

- Principle of least privilege
  - Each app has access only to the required components and no more



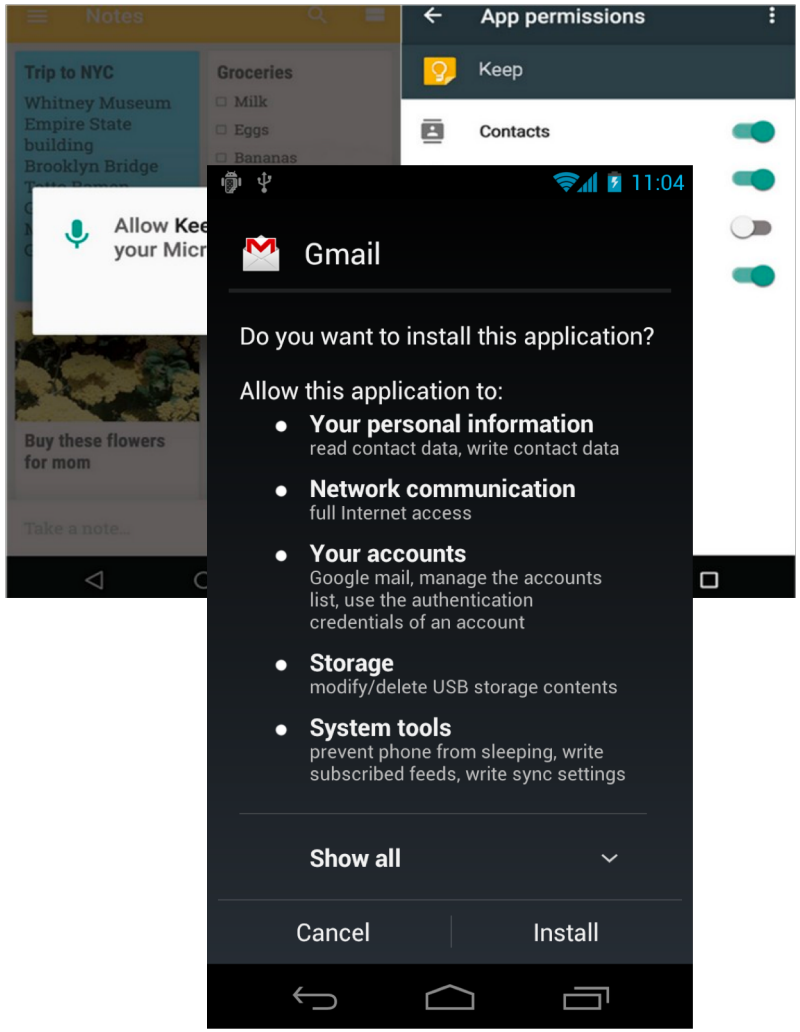
- Data sharing
  - Apps signed with the same certificate may run in the same process
  - **System permissions** are required to access device data (contacts, SMS)

Permission Protection Level

- Normal Permissions
  - ACCESS_NETWORK_STATE
  - BLUETOOTH
  - CHANGE_WIFI_STATE
- Dangerous Permissions
  - Contacts: READ_CONTACTS, WRITE_CONTACTS
  - Location: ACCESS_FINE_LOCATION
  - Phone: READ_PHONE_STATE, READ_CALL_LOG
  - SMS: READ_SMS, SEND_SMS

Permission Protection Level

- Normal permissions
  - Automatically granted by the system
- Dangerous permissions
  - Version 6.0+ (23+): granted by users at run time
    - Revoked by any time in the setting.
  - Version <6.0 (<23): granted by users at installation time
    - New permissions granted during update
    - Revoked by uninstalling the apps



Permission Protection Level

- Signature
  - Highest privilege, can only be obtained if the app is signed with the device manufacturer's certificate.

- Signature or System
  - Apps that are in the Android system image or are signed with the same certificate in the system image.
  - Only pre-installed apps from vendors, e.g., HTC, Motorola, Samsung,and LG

Permission Groups

Permission Group	Permissions
CALENDAR	• READ_CALENDAR • WRITE_CALENDAR
CAMERA	• CAMERA
CONTACTS	• READ_CONTACTS • WRITE_CONTACTS • GET_ACCOUNTS
LOCATION	• ACCESS_FINE_LOCATION • ACCESS_COARSE_LOCATION

- No permission are granted for a group -> ask users
- Any permission is already granted for a group -> no interaction with users

Using Permissions

- A basic app has no permissions associated
- Request specific permissions in manifest file

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.app.myapp" >
    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    ...
</manifest>
```

- Permission failure results in SecurityException
  - sendBroadcast may not get one

Example: Accessing Contacts

- Requesting permission

```
<uses-permission android:name="android.permission.READ_CONTACTS" />
```

- Preparing the URIs

```
Uri CONTENT_URI = ContactsContract.Contacts.CONTENT_URI;
String _ID = ContactsContract.Contacts._ID;
String DISPLAY_NAME = ContactsContract.Contacts.DISPLAY_NAME;
String HAS_PHONE_NUMBER = ContactsContract.Contacts.HAS_PHONE_NUMBER;

Uri PhoneCONTENT_URI = ContactsContract.CommonDataKinds.Phone.CONTENT_URI;
String Phone_CONTACT_ID = ContactsContract.CommonDataKinds.Phone.CONTACT_ID;
String NUMBER = ContactsContract.CommonDataKinds.Phone.NUMBER;

Uri EmailCONTENT_URI = ContactsContract.CommonDataKinds.Email.CONTENT_URI;
String EmailCONTACT_ID = ContactsContract.CommonDataKinds.Email.CONTACT_ID;
String DATA = ContactsContract.CommonDataKinds.Email.DATA;
```

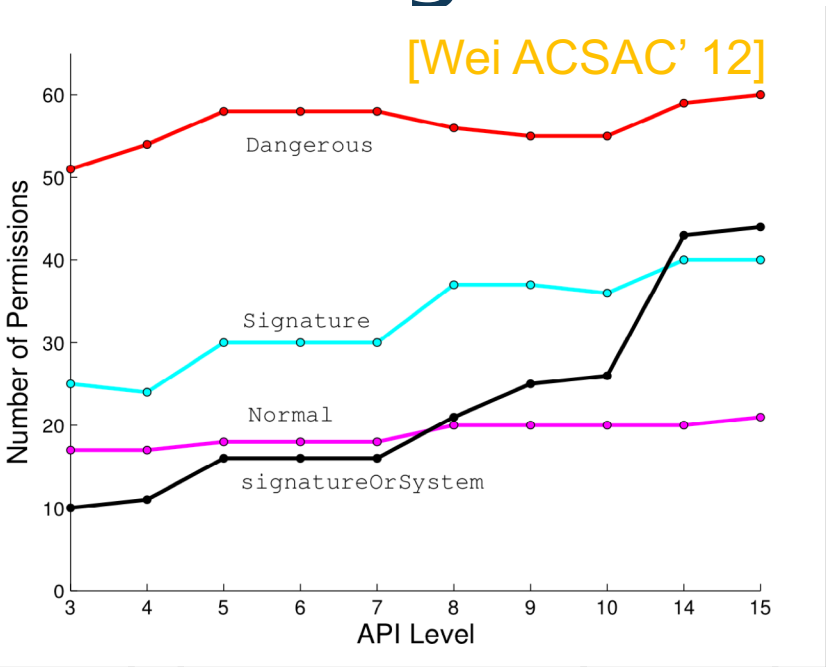
Example: Accessing Contacts

- Fetching data from Contacts content provider

```
Cursor cursor = contentResolver.query(CONTENT_URI, null, null, null, null);
while (cursor.moveToNext()) {
   String contact_id = cursor.getString(cursor.getColumnIndex(_ID));
   String name = cursor.getString(cursor.getColumnIndex(DISPLAY_NAME));
   int hasPhoneNumber =
Integer.parseInt(cursor.getString(cursor.getColumnIndex(HAS_PHONE_NUMBER)));
   if (hasPhoneNumber > 0) {
      Cursor phoneCursor = contentResolver.query(PhoneCONTENT_URI, null, Phone_CONTACT_ID
+ " = ?", new String[]{contact_id}, null);

      while (phoneCursor.moveToNext()) {
         phoneNumber = phoneCursor.getString(phoneCursor.getColumnIndex(NUMBER));
         output.append("\n Phone number:" + phoneNumber);
      }
   }
}
```

Permission Evolving

API level	Android platform	SDK codename	Total permissions	Release (mm-dd-yy)
15	4.0.3	Ice Cream Sandwich MR1	165	12-16-11
14	4.0.2 4.0.1	Ice Cream Sandwich	162	11-28-11 10-19-11
10	2.3.4 2.3.3	Gingerbread MR1	137	04-28-11 02-09-11
9	2.3.2 2.3.1 2.3	Gingerbread	137	12-06-10
8	2.2.x	Froyo	134	05-20-10
7	2.1.x	Eclair MR1	122	01-12-10
6	2.0.1	Eclair 0 1	122	12-03-09
5	2.0	Eclair	122	10-26-09
4	1.6	Donut	106	09-15-09
3	1.5	Cupcake	103	04-30-09



- Dataset: 237 popular apps with 1,703 versions that span at least three years
- Permissions are growing

Permission Evolution in the Android Ecosystem
Xuetao Wei, Lorenzo Gomez, Iulian Neamtiu, and Michalis Faloutsos.
28th Annual Computer Security Applications Conference (**ACSAC 2012**), December 2012.

Permission Growth in Categories

API level	Dev tools	Sys tools	Accounts	Cost Money	Hardware Controls	Location	Messages	Network	Personal Info	Phone calls	Storage	Default
3	36	35	1	2	6	4	5	5	6	3		
4	-1	+2,-2				+1					+1	+2
5		+3	+4						+2			+7
6												
7												
8		+7										+6, -1
9					+1			+2	-2			+2
10												
14		+2		+1	+2,-1		+1	+1	+5	+1	+1	+12
15		+1						+1				+1
Overall	-1	+13	+4	+1	+2	+1	+1	+4	+5	+1	+2	+29

- All categories are growing
- Default, Sys tools increase the most
  - More capabilities added through versions

Added Dangerous Permissions

Dangerous permission	Category
READ_HISTORY_BOOKMARKS	Personal Info
WRITE_HISTORY_BOOKMARKS	Personal Info
READ_USER_DICTIONARY	Personal Info
READ_PROFILE	Personal Info
WRITE_PROFILE	Personal Info
READ_SOCIAL_STREAM	Personal Info
WRITE_SOCIAL_STREAM	Personal Info
WRITE_EXTERNAL_STORAGE	Storage
AUTHENTICATE_ACCOUNTS	Accounts
MANAGE_ACCOUNTS	Accounts
USE_CREDENTIALS	Accounts
NFC	Network
USE_SIP	Network
CHANGE_WIFI_MULTICAST_STATE	System Tools
CHANGE_WIMAX_STATE	System Tools

- Largest group, and still growing
  - Most of them belong to personal information
- More channels for personal information

Why Added or Deleted Permissions

- Offer more functionality
  - E.g., in API 9, NFC (near-field communication)
  - E.g., in API 15, WiMAX is added for 4G
- Accommodate new features
  - E.g., in API 8, READ_OWNER_DATA removed, but READ_PROFILE and READ_SOCIAL_STREAM added in API 14
  - E.g., BACKUP_DATA added in API 5 deleted in API 8
- Facilitate development of pre-installed apps

Effectiveness of Permission Systems

[Felt et al. WebApps' 11]

- Traditional user-based permission system
  - Assign users' full privileges to all applications

- Android's permission system: application permissions
  - Each app has its own set of permissions
  - Two major types
    - Time-of-use: grant at run time
    - Install-time: grant at installation

The effectiveness of application permissions
AP Felt, K Greenwood, D Wagner
Proc. of the USENIX Conference on Web Application Development, 2011

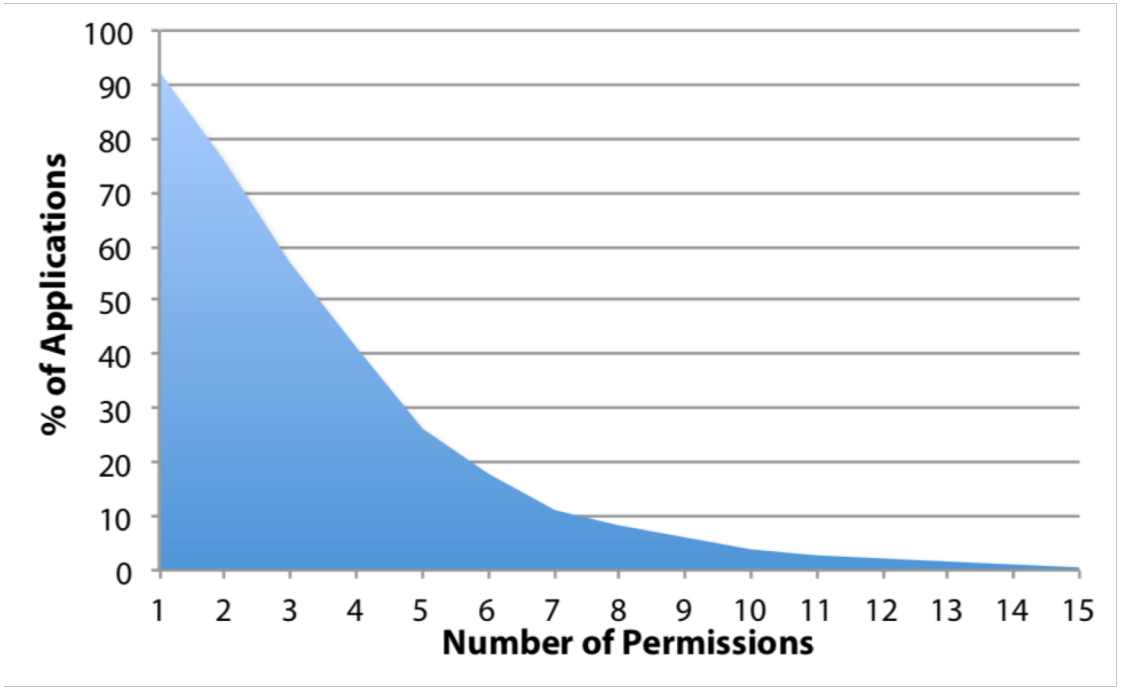
Advantages of Application Permissions

- Assumption: most apps requesting less than full privileges
- User consent: security-conscious users may be hesitant to grant access
- Defense in depth: limiting the vulnerabilities to the applications' privileges
- Review triaging: facilitating central review by ignoring low-privilege apps

Study Dataset

- 956 Android apps
- 100 paid apps and 856 free apps from Android Market
- 756 most popular apps and 100 most recent apps

Prevalence of Dangerous Permissions



- 93% of free and 82%of paid apps have at least one dangerous permission
- Max is 26, less than half of 56 total permissions
- Average is low: 3.99 for paid apps and 3.46 for free apps

Dangerous Permissions in Categories

(a) Prevalence of Dangerous permissions, by category.

Category	Free	Paid
NETWORK**	87.3 %	66 %
SYSTEM_TOOLS	39.7 %	50 %
STORAGE**	34.1 %	50 %
LOCATION**	38.9 %	25 %
PHONE_CALLS	32.5 %	35 %
PERSONAL_INFO	18.4 %	13 %
HARDWARE_CONTROLS	12.5 %	17 %
COST_MONEY	10.6 %	9 %
MESSAGES	3.7 %	5 %
ACCOUNTS	2.6 %	2 %
DEVELOPMENT_TOOLS	0.35 %	0 %

(b) The most frequent Dangerous permissions and their categories.

Permission (Category)	Free	Paid
INTERNET** (NETWORK)	86.6 %	65 %
WRITE_EXTERNAL_STORAGE** (STORAGE)	34.1 %	50 %
ACCESS_COARSE_LOCATION** (LOCATION)	33.4 %	20 %
READ_PHONE_STATE (PHONE_CALLS)	32.1 %	35 %
WAKE_LOCK** (SYSTEM_TOOLS)	24.2 %	40 %
ACCESS_FINE_LOCATION (LOCATION)	23.4 %	24 %
READ_CONTACTS (PERSONAL_INFO)	16.1 %	11 %
WRITE_SETTINGS (SYSTEM_TOOLS)	13.4 %	18 %
GET_TASKS* (SYSTEM_TOOLS)	4.4 %	11 %

- INTERNET is heavily used
  - 14% of free and 4% of paid apps have INTERNET as the only permission
  - Free apps often need INTERNET to load ads
- Potential risks in leaking information
  - 97% of 225 apps ask for INTERNET and ACCESS_FINE_LOCATION
  - 94% of apps that ask for READ_CONTACTS also ask for INTERNET
  - Significantly more free apps ask for both internet access and location data, leaking to ads

Evaluation of Application Permissions

- User consent
  - Nearly all apps request for 1+ dangerous permissions
    - Users are accustomed to it, and tend to ignore
  - Important categories are requested infrequently
    - PERSONAL_INFO and COST_MONEY
- Defense in depth
  - Most apps ask for less than 7 dangerous permissions
  - Only 10% ask for permissions that would cost users' money
- Review triaging
  - 18% of paid and 7% of free apps can be exempted
  - 22% of paid and 21% of free apps excluding INTERNET

Reducing Application Privileges

- Review process
  - More dangerous permission -> More review time
- Pressure from users
  - Users dislike permission requests
- Automatic updates
  - Updates with more permissions require re-grant permissions
    - Apps may request unnecessary permissions
    - Improve update UIs to minimize users' efforts

Developer Errors

- Manual review of 36 apps from 18 categories
- 4 over-privileged:
  - 3 INTERNET and 1 LOCATION
- 4 may avoid unnecessary permissions
  - "DocsToGo" needs INTERNET to update, but Android Market can do that
  - "Jesus Hates Zombie" could store its small set of data locally

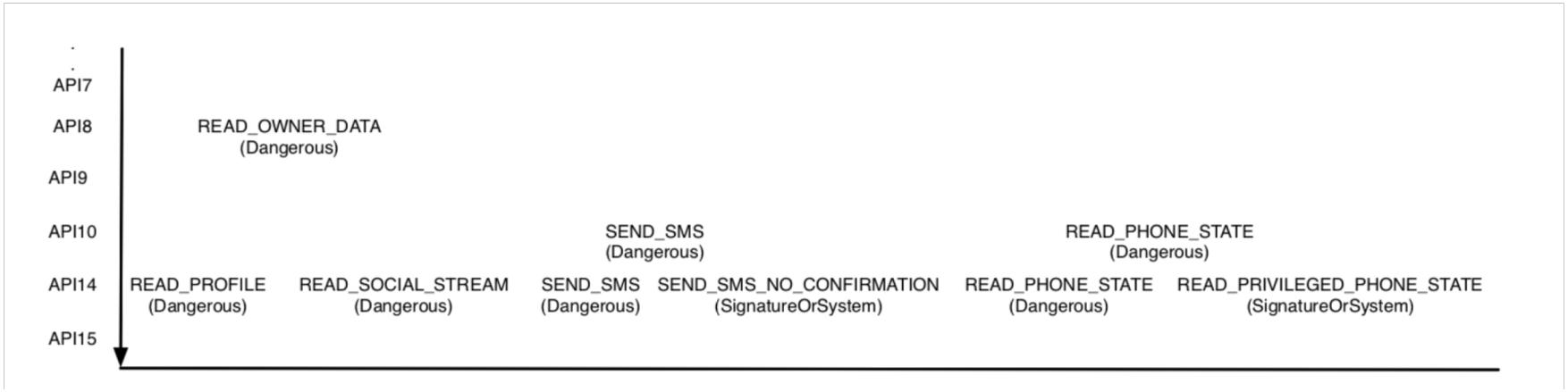
Permission Granularity

- Categories
  - No apps request all permissions in one category
  - STORAGE category has only one permission
- Read/Write
  - 149 apps requests one Contacts Read/Write, but not both
  - 10 of 19 apps requests both Read/Write
  - 6 of 53 apps requests all three permissions in SMS
- Location
  - FINE: GPS location, COARSE: cell location
  - 358 apps request either FINE(225) or COARSE (133)

Coarse-Grained Android Permissions

- INTERNET
  - HTTPS, WebView, Arbitrary destinations and ports
- 27 of 36 apps request INTERNET
  - 13 only for HTTPS
  - 14 only for AdSense in WebView
- Many apps can tolerate with limited Internet access
  - 52% of 27 apps would still work

No Tendency towards Finer-Grained



- One potential example
  - READ_OWNER -> READ_PROFILE and READ_SOCIAL_STREAM
- More flexibility and control to vendor apps
  - SEND_SMS -> SEND_SMS_NO_CONFIRMATION
  - READ_PHONE_STATE -> READ_PRIVILEGED_PHONE_STATE

User Prompts

- Risks of dangerous permissions
  - Cost the user money (e.g., send text messages)
  - Pertain to private information (e.g., location, contacts, and the calendar)
  - Eavesdrop on phone calls
- De-emphasizing less threatening permissions
  - WAKE_LOCK (26%)
    - Keep the phone awake, such as playing music
  - WRITE_EXTERNAL_STORAGE (35.7%)
    - Control access to SD card
  - INTERNET
    - Less severe warning for limited access, further notification for advertising domains

Presentations

- What is the problem
- Why it is important
- What are the challenges
- How to solve them
- What are the results

Thank You !

Questions ?