Fine-grained Access Control for Mobile Platforms
Seminar on Information and System Security

Thomas Nyman
Department of Computer Science
University of Helsinki
thomas.nyman@cs.helsinki.fi

March 13, 2013
Motivation

- Properties that make smartphones interesting targets
  - Private or sensitive information
    - Personal messages, contacts, etc.
    - Location
    - Electronic ID
  - Mobile phone payments
    - Premium numbers & SMS
    - Online banking
    - App Stores & in-app purchases
    - Near Field payments
  - Internet connectivity
  - Third-party applications

- Over 1 billion smartphones in use worldwide in 2012 [4]
- Other types of devices as well (feature phones, tablets etc.)
Outline

- Introduction
- Mobile Software Platforms
- Architecture
- Application Security
- Conclusion
Background

● Threat model includes
  ▶ Software defects
  ▶ Malware
  ▶ Malicious users

● Reasons behind current platform security measures
  ▶ Regulatory requirements
    ★ Secure storage of cellular network parameters
    ★ Theft deterrent mechanisms
  ▶ Business requirements
    ★ Subsidy locks & other vendor lock-in mechanism
    ★ DRM
  ▶ User requirements
    ★ Information privacy
    ★ Resistance against bugs & malware

● Influences on multiple levels
  ▶ Hardware security mechanisms
  ▶ Platform security architecture
What can be done?

- Reduce attackable surface area
- Follow Principle of least privilege
- Avoid software defects
- Enforce Mandatory Access Controls
- Manage & isolate resources
- Ensure software integrity
Mobile Software Platforms
### Android

- Operating system for smartphones and tablets
- Developed by the Open Handset Alliance led by Google
- Marketshare of over 70% of worldwide smartphone sales [7]
- Close to 80% of mobile threats target Android [5]
- Applications written in Java and run on the Dalvik VM
Tizen

- Aim to provide consistent user experience across different devices
  - Smartphones
  - Tablets
  - Netbooks
  - Smart TVs
  - *In Vehicle Infotainment* (IVI) systems
- Development governed within the Linux Foundation by Intel and Samsung
- Continuation of MeeGo formerly developed by Nokia and Intel
- Applications mainly web applications (widgets), supports native applications
Architecture
Android Architecture [1]
Android Access Control

- **Application Sandbox**
  - Enforced by kernel through Linux user access control mechanisms
  - Each application assigned unique low-privilege UID
  - Applications cannot interact with other processes
  - Application cannot access files of other applications

- **System resources accessed through middleware services**
  - Install-time permissions to access services
  - Enforced by *Android Application Framework*
  - Some permissions enforced by kernel through Linux groups
    - INTERNET
    - BLUETOOTH
    - WRITE_EXTERNAL_STORAGE
Tizen Architecture [2]
Tizen Access Control

- Applications run under one low-privilege UID
- Privilege escalation to root not allowed
- Access control done on two levels:
  - Kernel level application sandbox through Simple Mandatory Access Control Kernel (SMACK)
  - Fine grained access control of JavaScript APIs through Web RunTime (WRT) Access Control Engine (ACE)
- Each application part of Security Domain enforced by SMACK
- Access to middleware services subject to user space access control
SMACK

- Terminology:
  - **Subject**: Active entity accessing resources
  - **Object**: Passive entity that is accessed
  - **Access**: Read, write, execute, append
  - **Label**: Defines security characteristics of subjects or objects

- Two types of labels:
  - Object labels for filesystem objects
  - Process labels for executables

- Rules of the form: `subjectLabel objectLabel accessMode`
  - `Album Camera r`

- Support for RPM, Xorg, D-Bus and udev
Sample Manifest File [9]

```xml
<manifest>
  <define>
    <domain name="Camera" policy="shared" />
    <provide>
      <label name="Camera::timings" />
      <label name="Camera::public" />
      <label name="Camera::dbus-access" />
    </provide>
    <request>
      <smack request="Graphics" type="w" />
      <smack request="System" type="w" />
      <smack request="Camera::timings" type="rw" />
      <smack request="Camera::public" type="rw" />
      <smack request="Camera::dbus-access" type="rw" />
    </request>
  </define>
  <assign>
    <filesystem path="/opt/share/Camera_timings" label="Camera::timings" />
    <filesystem path="/opt/share/Camera_public/" label="Camera::public" />
    <dbus name="com.tizen.camera" own="Camera" bus="system">
      <node name="/com/tizen/camera">
        <interface name="com.tizen.camera">
          <annotation name="com.tizen.smack" value="Camera::dbus-access" />
        </interface>
      </node>
    </dbus>
  </assign>
  <request>
    <domain name="Camera"/>
  </request>
</manifest>
```
Tizen User Space Access Control [8]
Application Security
Android Permissions

- Application permissions declared in manifest file
- Four permission levels
  - **Normal**  Harmless functionality
  - **Dangerous**  Potentially harmful functionality
  - **Signature**  Restricted to OEM / Application developer
  - **SignatureOrSystem**  Restricted to pre-installed applications
- Permissions cover wide range of functionality
  - Voice calls, messaging, device state, Internet, Bluetooth, sensors etc.
  - Access to Content Providers
  - Access to device settings
- User authorizes dangerous features during application install
  - Cannot selectively refuse permissions
  - Permission changes during updates must be authorized separately
Tizen Widget Access Control Model

- Each widget has its own Security Domain (unique SMACK label)
- Widgets need authorization to invoke restricted JavaScript APIs
  - Declaration in manifest file
  - Authorization by user confirmation of prompt type according to WRT ACE policy
- W3C Widget Access Request Policy (WARP)
  - Network accesses by widgets denied by default
  - Protocol, domain, sub-domains or wildcard declared in manifest
- Policies defined Operators and OEMs
- Users can affect policy through preferences in a limited way
Sample Widget Manifest [3]

```xml
<?xml version="1.0" encoding="UTF-8"?>
<widget xmlns="http://www.w3.org/ns/widgets" xmlns:tizen="http://tizen.org/ns/widgets" version="1.0" id="http://YourDomain.com/SampleContact" viewmodes="fullscreen">
  <icon src="icon.png"/>
  <name>SampleContact</name>
  <content src="index.html"/>
  <description>Sample application for Tizen contact module.</description>
  <license/>
  <feature name="http://tizen.org/api/tizen" required="true"/>
  <feature name="http://tizen.org/api/contact" required="true"/>
  <feature name="http://tizen.org/api/contact.read" required="true"/>
  <feature name="http://tizen.org/api/contact.write" required="true"/>
  <access origin="http://jquerymobile.com" subdomains="true"/>
</widget>
```
Sample Policy File [3]

```xml
<policy-set id="Tizen-Policy" combine="first-matching-target">
  <policy id="Tizen-Policy-Trusted" description="Tizen's policy for trusted domain"
    combine="permit-overrides">
    <rule effect="prompt-session"> <!-- rules for specific resources -->
      <condition combine="and">
        <condition combine="or">
          <resource-match attr="device-cap" func="equal" match="XMLHttpRequest" />
          <resource-match attr="device-cap" func="equal" match="externalNetworkAccess" />
          <resource-match attr="device-cap" func="equal" match="messaging.send" />
        </condition>
        <environment-match attr="roaming" match="true" />
      </condition>
    </rule>
    <rule effect="permit" /> <!-- all other matches -->
  </policy>
</policy-set>
```
Prompt types [3]

- **Blanket Prompt**: Widget requires access to: contact.read. Options: Deny, Permit.
- **Session Prompt**: Widget requires access to: contact.read. Options: Deny, Permit. Additional options: Remember for one run.
- **One-Shot Prompt**: Widget requires access to: contact.read. Options: Deny, Permit.
Conclusion
Conclusion

- Attack Surface Reduction
- Principle of Least Privilege
- Software defect avoidance
- Mandatory Access Controls
- Resource management & isolation
- Integrity measurement
References

Android Security Overview.

Architecture of Tizen.
Retrieved 27.02.2013.

O. Acicmez and A. Blaich.
Understanding the permissions and access control model for tizen application sandboxing.
http://download.tizen.org/misc/media/conference2012/wednesday/seacliff/2012-05-09-0945-1025-understanding_the_permission_and_access_control_model_for_tizen_application_sandboxing.pdf, may 2012.
Tizen Developer Conference.

S. Analytics.
Worldwide Smartphone Population Tops 1 Billion in Q3 2012.
Retrieved 05.02.2012.

F-Secure.
Mobile Thread Report Q4 2012.

Android permissions demystified.

Gartner.
Gartner Says Worldwide Mobile Phone Sales Declined 1.7 Percent in 2012.

B. Im and R. Ware.
Tizen Security Overview.
Tizen Developer Conference.

E. Reshetova.
Application Installation and Manifest.
Retrieved on 05.03.2012.