TIZEN™ DEVELOPER CONFERENCE
MAY 7–9, 2012

Tizen Security Overview

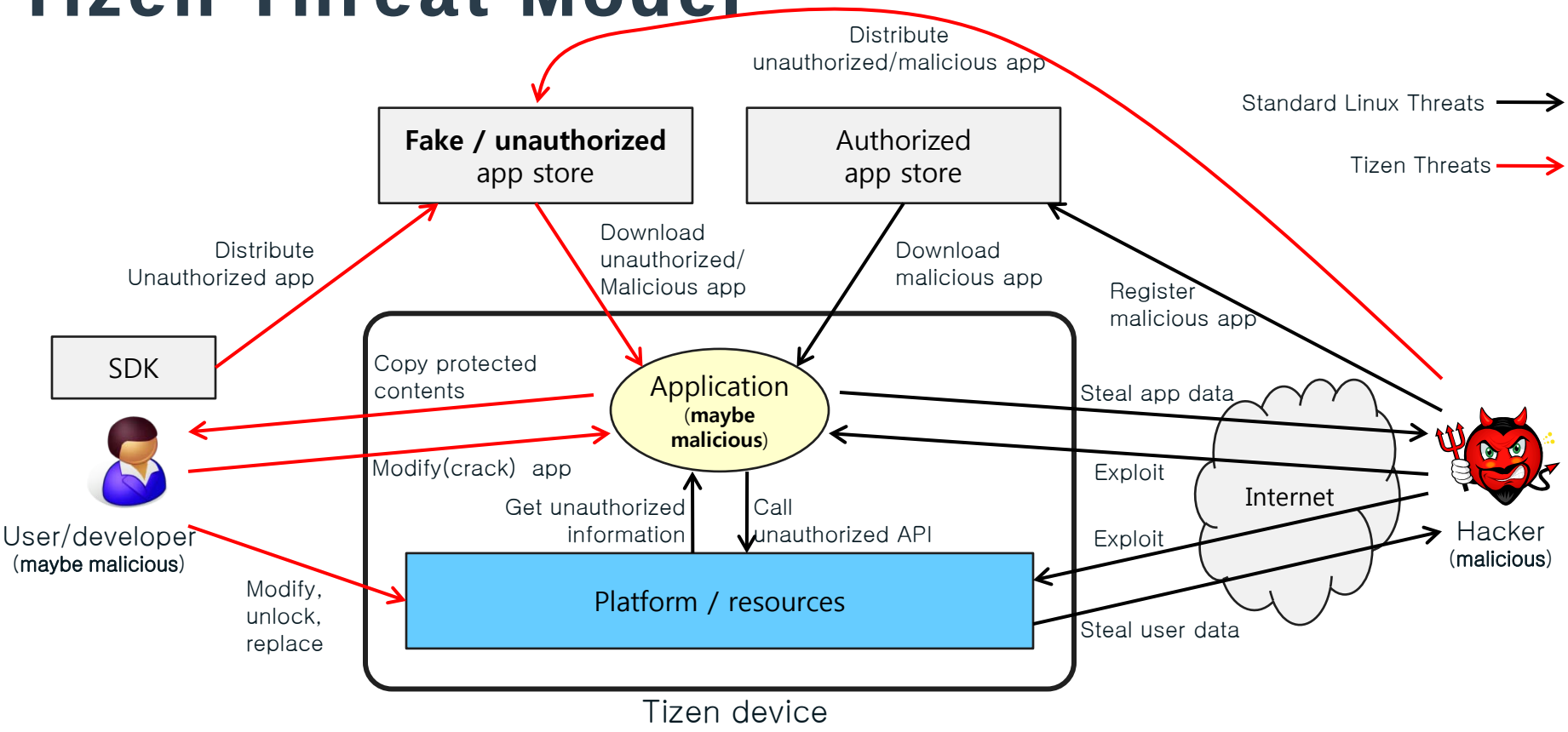
Bumjin Im, Ryan Ware

Contents

- Security Model of Tizen
- Enforcing Access Control
  - Kernel space access control by Smack
  - User space access control
- Application Security
  - Package Signing
  - Manifest Permission
- Future Plans

TIZEN SECURITY MODEL

Tizen Threat Model



High Level Security Principles

- No application running under root privilege
  - Most system daemons currently running under root privilege
    - Will be minimized soon
- No application can interrupt each other
  - Applications are sandboxed
  - Also possible to have application family
- Important system resources are protected by access control mechanism
  - Kernel space objects: DB files, network interface, IPCs, signals, ...
  - User space objects: Make phone call, Send SMS, ...
- All applications should have least privilege
  - Each application has different set of privileges

Security Model

- Non root applications
  - All applications run under same non-root user ID
  - Most of the middleware daemons will run as non-root user
- Application sandboxing
  - All applications are sandboxed by Smack
    - Each application has its own Smack label
  - Each application unable to send IPC and signal, r/w other application files, …
- Resource access control
  - Important system objects are Smack labeled
  - System daemons will make use of Smack and enforce access control
  - All files owned by root
    - Applications only able to write to home directory
- Least privilege
  - All applications will have manifest file describing permissions
    - Manifest file describes SMACK labels and rule of the application to get proper privileges

Tizen Access Control Model

- 1 root & 1 user account
  - No login
    - All applications will run under predefined non-root user ID
    - Application Utility Library (AUL) Launchpad *forks* and *setuids* to user ID
  - No privilege escalation allowed
    - There is no "su" command
    - System service daemons support privileged features
      - i.e.) change time, change routing table, shutdown, …
- Mandatory access control powered by Smack
  - Each application is Smack labeled and has proper Smack rules
    - Assigned and maintained by manifest file from each package
  - Application based sandboxing
    - Each application is able to write to home directory only
      - Basically application can read system files but not able to read other application's files

ACCESS CONTROL

Smack

- Tizen includes the Simple Mandatory Access Control Kernel (Smack)
  - Upstream Linux Security Module
  - Simple {subject, object, permission} access control model
  - Provides method of creating appropriate security domains without wholly understanding a 900,000 line reference policy
- New Features:
  - Long Label Support
    - Smack labels up to 255 characters instead of 23
  - Recursive Transmute
    - Isolation of application tmpfs contents
  - SmackFS Access
    - Mechanism allowing userspace apps to check Smack permissions

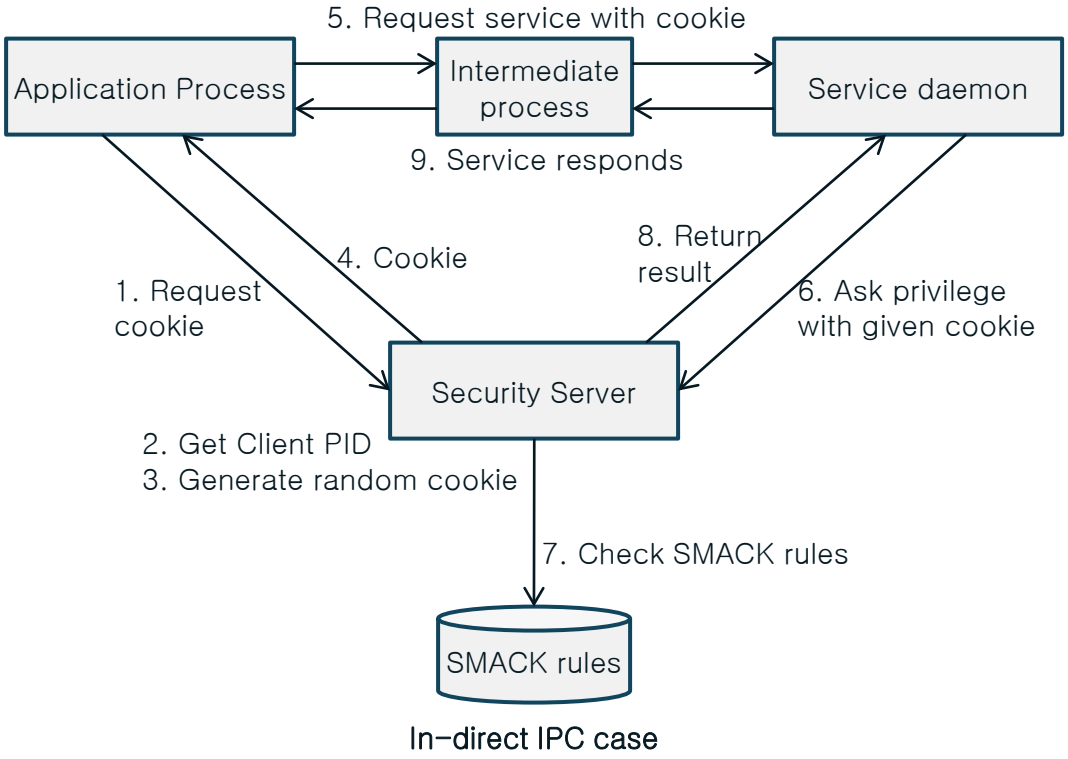
Smack Enabled Services

- Xorg
  - X Access Control Extension
- DBus
  - Utilizes SmackFS Access
  - Allows DBus to enforce Smack protections itself
- Udev
  - Create device nodes with Smack labels
- RPM
  - Security Manifest Plugin
  - Sets Smack Labels on Install

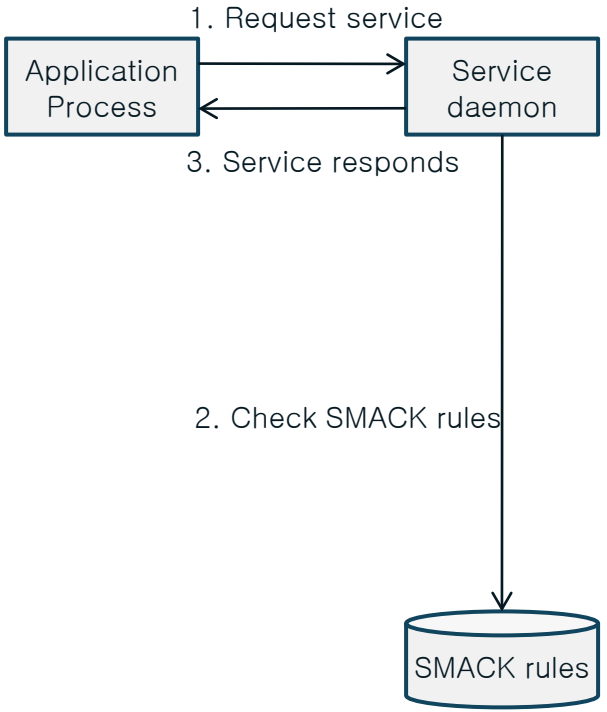
User Space Access Control

- User Space Objects
  - System resources consist of client-server
    - Use IPC for request/response channel & access occurs in server side
      - i.e) telephony, network, messaging, email, system f/w, …
    - Not possible to use kernel space access control
- Mechanism
  - Direct IPC: Server identifies client by kernel feature (SO_PEERSEC)
    - Check Smack rule for client label and continue service
  - Indirect IPC:
    - Trusted Daemon (security-server) identifies client and issues random cookie
      - Cookie identifies a Smack label
      - Security of the cookie value is important
      - Need garbage collection mechanisms for closed processes
    - Client uses cookie with request and server checks cookie with security server
    - Can be used as service with single DBus interface as IPC for multiple services

User Space Access Control (cont)



In-direct IPC case



Direct IPC case

APPLICATION SECURITY

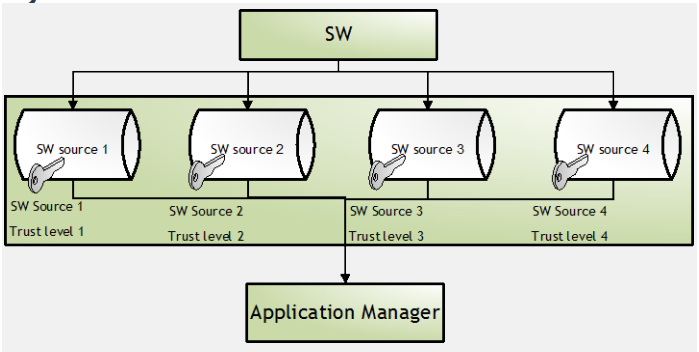
Manifests

- Included as part of package

```
<define>
  <domain name="Camera" policy="shared" />
    <provide>
          <label name=”Camera::statistics” />
          <description>A new label for labelling camera statistic
data</description>
          <label name=”Camera::timings” />
          <description>A new label for labelling camera data that contains
information about timings</description>
          <label name=”Camera::public” />
          <description>A new label for labelling camera public
data</description>
   </provide>
</define>
```

Secure Software Distribution

- Ensure SW updates originate from any one of multiple authenticated sources
  - RPM Signed Updates
- Trust Levels Assigned to Sources
  - Updates for installed software can only come from the same source or a source with higher trust
- OEM configurable
- Remotely updateable



Conclusion & QA

- Tizen Cannot Be Secure Without Your Help!!!
  - We can’t have eyes everywhere!
- If you have a security concern of any kind:
  - security@tizen.org
- There will be opportunities to participate in Tizen security soon

TIZEN™ DEVELOPER CONFERENCE
MAY 7–9, 2012