Tizen Security

Fourteenforty Research Institute, Inc.
http://www.fourteenforty.jp

Senior Research Engineer
Shuichiro Suzuki
As alternative OS for smartphones and tablets Tizen and Firefox OS are becoming remarkable OS
A vendor in Japan may release a device with Tizen OS in this or next year
In Android security has been concerned after its release
How about the security in new OS?
Research on the security of Tizen
Research in this slides are done by using an emulator in Tizen SDK 2.1
About Tizen

- Overview
  - Open source OS for mobile devices
  - One project in Linux Foundation
  - Development is mainly lead by Samsung and Intel

- Features
  - Linux based OS
  - It supports both web and native application development from Tizen 2.0
Tizen Architecture

Supports web and native applications

From [http://upload.wikimedia.org/wikipedia/commons/c/c3/What_is_tizen_architecture.png](http://upload.wikimedia.org/wikipedia/commons/c/c3/What_is_tizen_architecture.png)
Tizen Package

- There are 2 application package format
  - Tizen Package (.tpk): Native Applications
  - Tizen Web Package (.wgt): Web Applications
Common Feature of Tizen Web/Native application

- Installed into /opt/apps/(AppId)
- AppId is 10 byte length string
- Under /opt/apps/(AppId) directory there are
  - bin/
  - data/
  - res/
Tizen Web package

- Zip archive with .wgt extension
- .wgt file constitutes the structure shown below

From https://developer.tizen.org/documentation/articles/tizen-application-packaging-overview

- bin directory has a file which is a symbolic link to Web Runtime (WRT)
- A web application is hosted by this Web Runtime
Tizen Native package

- Zip archive with .tpk extension
- .tpk file constitutes the whole structure under (AppId) directory

- bin directory has an application binary

https://developer.tizen.org/documentation/articles/tizen-application-packaging-overviewより引用
Security

- Research on following security features
  - OS level Security
    - Access Control
    - Vulnerability Protection
    - Content Security Framework
  - Application Rights Isolation
    - Privileges
    - Feature
Overview of Access Control

- All the processes run with one of 2 UIDs which corresponds following accounts
  - root
  - app
- Both web and native application run with UID ‘app’
- It has mandatory access control by SMACK
  - All the applications are labeled by SMACK
Using 2 UIDs

Run with UID ‘app’

Web App1

Web App2

Native App (UID: app)

Web Runtime (UID: app)

Kernel

Processes require high rights run with UID ‘root’

Service (UID: root)
Application Isolation

All the applications run with UID ‘app’

↓

Can they access their private file each other??

Access control by SMACK
(They cannot access their private file each other by default)
SMACK

- One of the implementations of LSM (Linux Security Modules)
- Labeling to Subject (process) and Object (file) and controlled by access rules between them
- Example of a rule:

```
TopSecret   Secret   rx
```
Subject labeled “TopSecret”
To object labeled “Secret”
Can read and execute
Utilize SMACK Label

Run with UID ‘app’

Processes require high rights run with UID ‘root’

Web App1

Web App2

Native App1

Service1 (UID: root)

Web App1

Web App2

Native App1

Service1

File1

File2

Cannot access to a file labeled WebApp2
Vulnerability Protection

- Supports native (C/C++) application development from Tizen 2.0
- There may be a typical buffer overflow vulnerability
- Main possible protections are ASLR and DEP
DEP

- Tested following code as a Tizen Native Application

```c
int func(){
    int a = 10;
    int b = 20;
    return a+b;
}

_EXPORT_ int OspMain(int argc, char *pArgv[])
{
    AppLog("Application started.");
    char buf[1024];
    int (*f)();
    memcpy(buf, (char*)func, 1024);
    f = (int (*)())buf;
    int b = f();
    ArrayList args(SingleObjectDeleter);
    args.Construct();
    ..... Prepare buffer on the stack
    Copy func() into the buffer
    Run the cond on the stack
    This is executed without any errors
    DEP is not enabled
    (on Tizen SDK 2.1 x86 Emulator)```
ASLR

- The value of /proc/sys/kernel/randomize_va_space is 2
  - Which means ASLR is enabled

- But actually… (Tizen Native App on emulator)
  - Run a same program 2 times. Main modules, heap and stack addresses in /proc/[pid]/maps are the same

```
09e0e000-09e70000 rw-p 00000000 00:00 0          [heap]
09e70000-09f80000 rw-p 00000000 00:00 0          [heap]
b36e7000-b36ec000 r-xp 00000000 fe:00 73077 /opt/usr/apps/hNLQM52K10/bin/MySample7.exe
b36ec000-b36ed000 rw-p 00000000 00:00 0          /opt/usr/apps/hNLQM52K10/bin/MySample7.exe
b36ed000-b36f0000 r-xp 00000000 fe:00 73094 /opt/usr/apps/hNLQM52K10/bin/MySample7.exe
b36f0000-b36f1000 rw-p 00000000 00:00 0          /opt/usr/apps/hNLQM52K10/bin/MySample7.exe
bfdcf000-bfdf0000 rw-p 00000000 00:00 0          [stack]
```

- Not Randomized (On Tizen SDK 2.1 x86 Emulator)

- The value of /proc/self/personality is 00040000 (ADDR_NO_RANDOMIZE)
  - ASLR is disabled by this setting
Content Security Framework (CSF)

- This framework makes it easier to provide security check feature into Tizen
- CSF defines API and Plug-in interface
- Security check feature is provided as a Plugin (libengine.so)
- A Plug-in provides features to check file, URL, Web site (HTML, JavaScript)
- Applications provides the Plug-in (Security Application Package) must have trusted signature
Application Rights Isolation

- Privilege
  - Tizen divides APIs into 3 categories according to its right
    - Public – All the developers can use
    - Partner – Partner developers who registered Tizen application store can use
    - Platform – For managing Tizen platform (Limited developers can use)
  - Application without proper privilege can not use API
  - Application manifest file has the description of privileges

- Feature
  - This is like a permission on Android platform
  - Write features to use (Access contacts, camera and so on) in manifest file
  - Web Runtime has Access Control Engine (ACE)
  - It controls an access to each features
Web Application Sandbox

2 stage access control via Web Runtime and SMACK

Even if there is a vulnerability in Web Runtime it can prevent to access to device file to which the process does not have an permission (Have not confirmed that all the access control level is the same between WRT and SMACK rules)
Summary

- SMACK is the core part of access control in Tizen
- It has 2 stage access control with WRT and SMACK for web based application
- It can flexibly provides security check feature by Content Security Framework
- There may be a classical vulnerability like buffer overflow since it allows to develop an application by native(C/C++) code.
- There are some spaces to improve in memory protection such as ASLR or DEP
Reference

- 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
- http://www.youtube.com/watch?v=GtiAQOo4beg