HACKING TIZEN
THE OS OF EVERYTHING
AJIN ABRAHAM | @ajinabraham
WHOAMAI

- Application Security Engineer, Yodlee
- Blogs at opensecurity.in
- Spoken at NULLCON, ClubHack, OWASP AppSec, BlackHat, Ground Zero Summit….
- Loves to learn NEW things.
DISCLAIMER

- All Images, Logos and Trademark belongs to their respective owners.
- All vulnerabilities discussed are responsibly disclosed to Tizen Security community.
- Personal View/Research, doesn’t reflect the views of my employer.
AGENDA

- What is Tizen
- Why Tizen?
- Types of Tizen Application
- Tizen Architecture
- Tizen Application Structure
- Tizen Security Model
- Sandbox – SMACK
- WebKit2 on Tizen
- Quick Comparison – Android vs Tizen vs iOS

- Hacking Tizen
  * Android vs Tizen Web App
  * Shellshock
  * Issues in DEP
  * Broken ASLR
  * CSP Bypass
  * URL Spoofing/Content Injection

- Pentesting Methodology
  * Static Analysis
  * Dynamic Analysis
  * Network Analysis

- Security Concerns in Tizen
- Conclusion
TIZEN: The OS of Everything

IoT (Internet of Things)
Tizen – A Linux Foundation Project.
Why TIZEN?
Micromax beats Samsung, becomes India's No. 1 mobile vendor: Report

Anupam Saxena, TOI Tech | Aug 4, 2014, 07.04PM IST

NEW DELHI: Micromax has overtaken Samsung to become the largest mobile phone supplier in India in Q2 2014, according to independent market research and consulting firm, CounterPoint Research.

As per the report, Micromax's handset shipments share was 16.6% in the quarter while Samsung's share was 14.4%. This is the first time that Samsung has been displaced from the pole position.

Nokia was at the third position with a 10.9% shipments share, followed by domestic brands Karbonn and Lava which had a 9.5% and 5.6% share, respectively.

Samsung and Intel find 36 more companies to back Tizen, their Android competitor

By Rich McCormick on November 12, 2013 04:36 am  ➤ Email

Source: http://www.theverge.com/2013/11/12/5093588/tizen-open-operating-system-partners-with-36-companies
Samsung 2015 Tizen TV range now available at Curry’s in the UK

TYPES OF TIZEN APPLICATIONS

Native

Hybrid

Supports Android application with Tizen Application Compatibility Layer (ACL).
TIZEN ARCHITECTURE

Framework
- Tizen Web App .wgt
  Tizen Web Framework (HTML5 + Tizen Web API)
- Tizen Native App .tpk
  Tizen Native Framework (C++ API)

Core
- App Framework
- Graphics & UI
- Multimedia
- Location
- Messaging
- Web
- Security
- System
- Base
- Connectivity
- Telephony
- PIM

Kernel
- Linux Kernel & Drivers
Web API = Standard HTML5 + Tizen Device API
TIZEN APPLICATION STRUCTURE
WEB APPS (.WGT)
TIZEN SECURITY MODEL

• **Non root applications**
  • All applications run under same non-root user ID, app.
  • Most of the middleware daemons will run as non-root user.

• **Application sandboxing**
  • All applications are sandboxed by SMACK.
  • An application is allowed to read or write files in it’s home directory and shared media directory (/opt/usr/media)
  • Each application unable to send IPC and sockets, r/w other application files.

• **Permission Model/Least privilege**
  • All applications will have manifest file describing privileges.
  • Manifest file describes also SMACK labels and rule.

• **Application Signing** – Author and Distributor

• **Tizen CSP for Web Apps** – Web Apps have additional layer of security with Content Security Policy.

• **Encrypt HTML, JS and CSS stored in Device** - Encrypts at Install time and Runtime decryption.

• **Content Security Framework** – Provides API for AVs.
SMACK
SIMPULIFIED MANDATORY ACCESS CONTROL KERNEL

“what's mine is mine; what's yours is yours.”

SMACK allows you to add controlled exception to this basic rule.
SMACK TERMS

- **Subject** → Any Running Process (Have Smack Label)
- **Object** → File, IPC, Sockets, Process
- **Access** → Read (r), Write (w), Execute (e), Append (a), Lock (l), Transmute (t)

41,000 SMACK Rules in Tizen 2.2.1 !!
From Tizen 3.X: (Smack Three domain Model, Cynara)
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<Manifest xmlns="http://schemas.tizen.org/2012/12/manifest">
  <Id>BEyf9tNAUG</Id>
  <Version>2.0.0</Version>
  <Type>C++App</Type>
  <Requirements>
    <Feature Name="http://tizen.org/feature/screen.size.normal">true</Feature>
  </Requirements>
  <Author/>
  <Descriptions/>
  <Url/>
  <DeviceProfile/>
  <Apps>
    <ApiVersion>2.0</ApiVersion>
    <Privileges>
      <Privilege>http://tizen.org/privilege/socket</Privilege>
      <Privilege>http://tizen.org/privilege/wifi.wifidirect.read</Privilege>
      <Privilege>http://tizen.org/privilege/wifi.wifidirect.admin</Privilege>
      <Privilege>http://tizen.org/privilege/network.connection</Privilege>
      <Privilege>http://tizen.org/privilege/wifi.admin</Privilege>
    </Privileges>
    <UiApp Main="True" Name="TizenNative" MenuItemVisible="True" />
WEB APPS – CONFIG.XML

---

### API Group

<table>
<thead>
<tr>
<th>Feature / Device Capability</th>
<th>API Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td><a href="http://tizen.org/api/time">http://tizen.org/api/time</a></td>
</tr>
<tr>
<td></td>
<td><a href="http://tizen.org/api/time.read">http://tizen.org/api/time.read</a></td>
</tr>
<tr>
<td></td>
<td><a href="http://tizen.org/api/time.write">http://tizen.org/api/time.write</a></td>
</tr>
<tr>
<td></td>
<td>All</td>
</tr>
<tr>
<td></td>
<td>All except setCurrentDateTime()</td>
</tr>
<tr>
<td></td>
<td>setCurrentDateTime()</td>
</tr>
</tbody>
</table>

---

### JavaScript:

```javascript
... var current_dt = tizen.time.getCurrentDateTime(); var is_leap = tizen.time.isLeapYear(current_dt.getFullYear()); if (is_leap) console.log("This year is a leap year."); ...
```

---

### Manifest File:

```xml
...<feature name="http://tizen.org/api/tizen"/>
<feature name="http://tizen.org/api/time.read"/>
...`
WEBKIT2 on Tizen

- Tizen WebApps runs on WebKit2
- New API Layer over WebKit
- Supports Split Process Model, Like your Chrome Tabs
QUICK COMPARISON

**Android**
- Apps identified by UID
- Permission: AndroidManifest.xml
- Binder IPC using Intents
- SELinux
- Signed by Developer

**iOS**
- All Apps run under user “mobile”.
- No permission model. Ask for Permission at Runtime.
- URL Schemes, x-callback URL, Extension, XPC based IPC
- Powerbox, Seatbelt
- Signed by Distributor
RESEARCH FOCUS

- Tizen 2.2.1 and IVI 3.0
- OS Memory Protection
- Tizen CSP and WebKit
ANDROID WEB APP vs. TIZEN WEB APP

• Tizen Web Apps are powerful and feature rich.
• In Android Web Apps in WebView and can interact with Device features using `addJavascriptInterface`.
• In Tizen, It provides Web API that allows to leverage Device features and are protected using privileges and CSP.
OVER PRIVILEGED ANDROID APP VS TIZEN APP

**Android**

- WebView
- ADDJAVASCRIPTINTERFACE
- BLUETOOTH PERMISSION
- NFC PERMISSION
- DEVELOPER EXPOSES API TO BRIDGE
- BLUETOOTH
- NFC

**Tizen**

- WebApp
- BLUETOOTH PRIVILEGE
- NFC PRIVILEGE
- BLUETOOTH API
- NFC API
- BLUETOOTH
- NFC
**SCENARIO : XSS**

- **WebView**
  - XSS
  - AddJavaScriptInterface
  - Bluetooth Permission
  - NFC Permission
  - Developer Exposes API to Bridge

- **WebApp**
  - XSS
  - CSP
  - Bluetooth Privilege
  - NFC Privilege
  - Bluetooth API
  - NFC API

- **Android**
  - Bluetooth
  - NFC

- **Tizen**
  - Bluetooth
  - NFC
LIKE ANY LINUX DISTRO: SHELLSHOCK
• When Data Execution Prevention is enabled, data on stack should be non-executable.
• Prevents Shellcode at Stack from Executing.
• But DEP is not seen in action.
ASLR

• As per documentation ASLR is fully implemented in Tizen 2.1 itself.
• Already Broken in Tizen 2.1, discovered by Shuichiro Suzuki
• `/proc/sys/kernel/randomize_va_space` is set to 2 which tell us that ASLR is enabled.
• The personality value at `/proc/self/personality` is set to 00040000, which corresponds to (ADDR_NO_RANDOMIZE) disables ASLR.
• InTizen 2.2, `/proc/self/personality` is set to 00000000

Pie (position-independent executable). So this will make the native application ASLR enabled.
• But due to implementation issues, it was still found that ASLR is still in broken state.
• `/proc/<pid>/maps` – Address of heap, stack and main modules remain the same.
Open a new window with URL https://facebook.com and assign it to a variable w.

Try to write “<h1>You've been Hacked</h1>” to DOM using w.document.write()

Focus the window.
DEMO
CSP BYPASS

Content-Security-Policy: default-src 'self'; script-src 'self'

• We create a script tag with JavaScript nullbyte prepended to a SCRIPT URL.
• Tricks the browser and load the Script from a different domain and Bypass CSP.
DEMO
PENTESTING METHODOLOGIES

Whitebox
Access to Source and Knowledge about the application

Blackbox
No access to Source and no idea about the application

Further Classification
- Static Analysis
- Dynamic Analysis
- Network Analysis
STATIC ANALYSIS

• **Certificate Signature Analysis** – Developer and Distributor
• **Manifest Analysis** – manifest.xml/config.xml
  * Unwanted Privileges.
  * CSP is proper or not.
  * Smack Labels and Rules

• **Decompile Native App**
  * Apps Compiled with CLANG/CLANG++ compiler.
  * LLVM decompiler - tizen_tpk_decompiler.py (make use of Retdec API).

• **Code Review**
  * Weak Encryption, Crypto, Plaintext Information, SSL Overriding, Insecure File Storage, Client Side SQLi/XSS.
  * Pretty much OWASP Mobile Top 10.

• **Couple of tools** - https://github.com/ajinabraham/tizen-security
DYNAMIC ANALYSIS

• Enable Developer Mode - *#84936#
• Run the App in Device/Tizen VM or Web Simulator.
• Sensitive data shared during IPC, Sensitive files written at Runtime, Temp files etc.
• Directories/ Files/DB with chmod 777 access.
• Tools: Dynamic Analyzer much like android ddms/Android Device Monitor, sdb – The adb equivalent for Tizen.
Usage: sdb [option] <command> [parameters]

options:
- -e, --emulator
  direct command to the only running emulator
  return an error if more than one emulator is running
- -d, --device
  direct command to the only connected USB device
- -s, --serial <serial_number>
  direct command to the USB device or emulator with the given serial number

commands:
- sdb root <on | off>
  switch to root or developer account mode
  'on' means to root mode, and vice versa
- sdb status-window
  continuously print device status for a specified device
- sdb get-serialno
  print: <serial-number>
- sdb get-state
  print: offline | locked | device
- sdb kill-server
  kill the server if it is running
- sdb start-server
  ensure that there is a server running
- sdb version
  show version num
- sdb help
  show this help message
- sdb forward <local> <remote>
  forward socket connections
  For example: sdb forward tcp:9999 tcp:9999
- sdb uninstall <pkg_id>
  uninstall an app from the device
  the <pkg_id> is an unique 10-digit unique identifier for the application. The Ex.) sdb uninstall ko983dw33q
- sdb install <pkg_path>
  push package file and install it
- sdb dlog [<filter_spec>]
  view device log
- sdb shell [command]
  if argument is null, run remote shell interactively
  if argument is not null, run command in the remote shell
- sdb pull <remote> [<local>]
  copy file/dir from device
- sdb push <local> <remote> [--with-utf8]
  copy file/dir to device
  (--with-utf8 means to create the remote file with utf-8 character encoding)
- sdb disconnect [<host>[:<port>]]
Choose a target and template

**Targets**
- mobile-2.3

**Template**
- Bottleneck
- Memory Leaks
- Process Activity
- File
- Thread Activity
- Wait Status
- Network
- OpenGL
- Energy
- Custom

**Bottleneck**
This template shows where can be the most bottleneck point while you are using your program. With the CPU and process chart, you can easily find where the application uses the CPU a lot. And the function profiling and call trace information shows the bottleneck point with the view of function level.
• Installing SSL Certificate and HTTPS Traffic Decryption with a Proxy like Burp/Fiddler.
• OWASP Top 10 Web Risks
INSTALLING CA CERT TO TRUSTED CERT STORE

- Installing CA in Device
- Trusted CA Certificates are stored under /etc/ssl/certs
- Filename: `<8HEXChars.0>` in PEM format.
- Copy the CA certificate to /etc/ssl/certs and it’s trusted.
Currently Supports Android and iOS.
Tizen support is on the way.

SECURITY CONCERNS

- WebKit = Bugs!!

- "WebKit is basically a collection of use-after-frees that somehow manages to render HTML (probably via a buffer overflow in WebGL)" - the gruggq

- HTML Web APIs are powerful, Improper CSP and XSS=owned !!

- Too much SMACK Rules – High chance that developers will mess up. Will be reduced from Tizen 3.
CONCLUSION

- Security Model/Architecture wise they put lot of effort compared to Android or other Operating Systems.

- They made it so complex (SMACK rules) that people can easily mess up.

- Looks promising if they can fix some silly implementation bugs.
THANKS

- Thanks to Yodlee and my awesome manager, Sachin for all the support and encouragement.
- Presentation template by SlidesCarnival & Unsplash
QUESTIONS?

Ajin Abraham
@ajinabraham

http://opensecurity.in