

Beginners Guide to Android Reverse Engineering

(W)ORK-SH/OP:

sam@ccc.de

Ha11[14], Day 3 11:00h

Agenda

- Purpose
- Recommended or needed tools
- (De)construction of Android apps
- Obtaining APKs
- Decompiling
- Resource and code manipulation
- Repackaging and signing
- Where to go from here

Purpose

- Smartphone contains valuable data
- Many uncontrolled interfaces
- Security controls depend on user-given permissions

Malware

- Downloads from obscure sources
- Requested „support“ apps
- Piggybacked on Google Play downloads

Privacy leaks

- Address book sync on servers
- Content analysis for „smart“ search
- Ads!

Cheats

- Manipulation of Scores
- Or In-Game assets
- Cheating

Recommended or needed tools

Androguard Various Python scripts
(similarity, unpacking)

dex2jar Dalvik->Java converter

JD-GUI Visual Java decompiler

apktool Package decoder/builder

(De)construction of Android apps

- Coding chain
 - Coding in Java, using Android SDK/NDK
 - Compiling to Dalvik (DEX-Files), running in DVM
 - Packaging as signed APK
 - Distributed freely or via Google Play
- Decoding chain



Obtaining APKs

1. Pulling from device

- Connect USB-Cable
- Use **ADB** (Android Debug Bridge) from SDK
- No Google Play on emulator (AVD)

2. Directly downloading via **googleplay-pythonapi** from Google Play

- Configured Google Account with connected Android ID

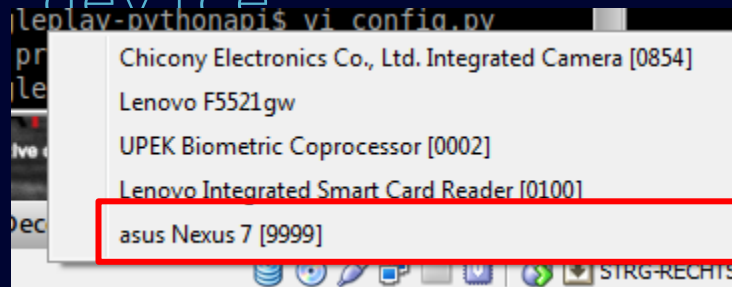
3. Download from Web

- Alternative source
- Capture transfer to device (Proxy, Wireshark)

Obtaining APKs: ADB (1)

Hands On

- Prepare device
 - Tap Settings/Device Info multiple times
 - Set developer options on device
 - Enable USB debugging
- Connect to Host
 - Connect to VM
- If device is “rooted”,
adb server must be started as root



```
$ adb devices -l
List of devices attached
emulator-5554          device product:sdk model:sdk device:generic
???????????????? no permissions usb:1-2
```

Obtaining APKs: ADB (2)

Hands On

```
root@santoku-VirtualBox:~# adb start-server
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
root@santoku-VirtualBox:~# adb devices -l
List of devices attached
emulator-5554          device product:sdk model:sdk device:generic
01 androidId x13      device usb:1-2 product:nakasi model:Nexus_7
device:grouper
```

```
root@santoku-VirtualBox:~# adb -d shell
shell@android:/ $ id
uid=2000(shell) gid=2000(shell)
groups=1003(graphics),1004(input),1007(log),1009(mount),1011(adb),1015
(sdcard_rw),1028(sdcard_r),3001(net_bt_admin),3002(net_bt),3003(inet),
3006(net_bw_stats)

root@santoku-VirtualBox:~# adb -d shell pm list packages -f > list
```


Obtaining APKs: googleplay-pythonapi

Hands On

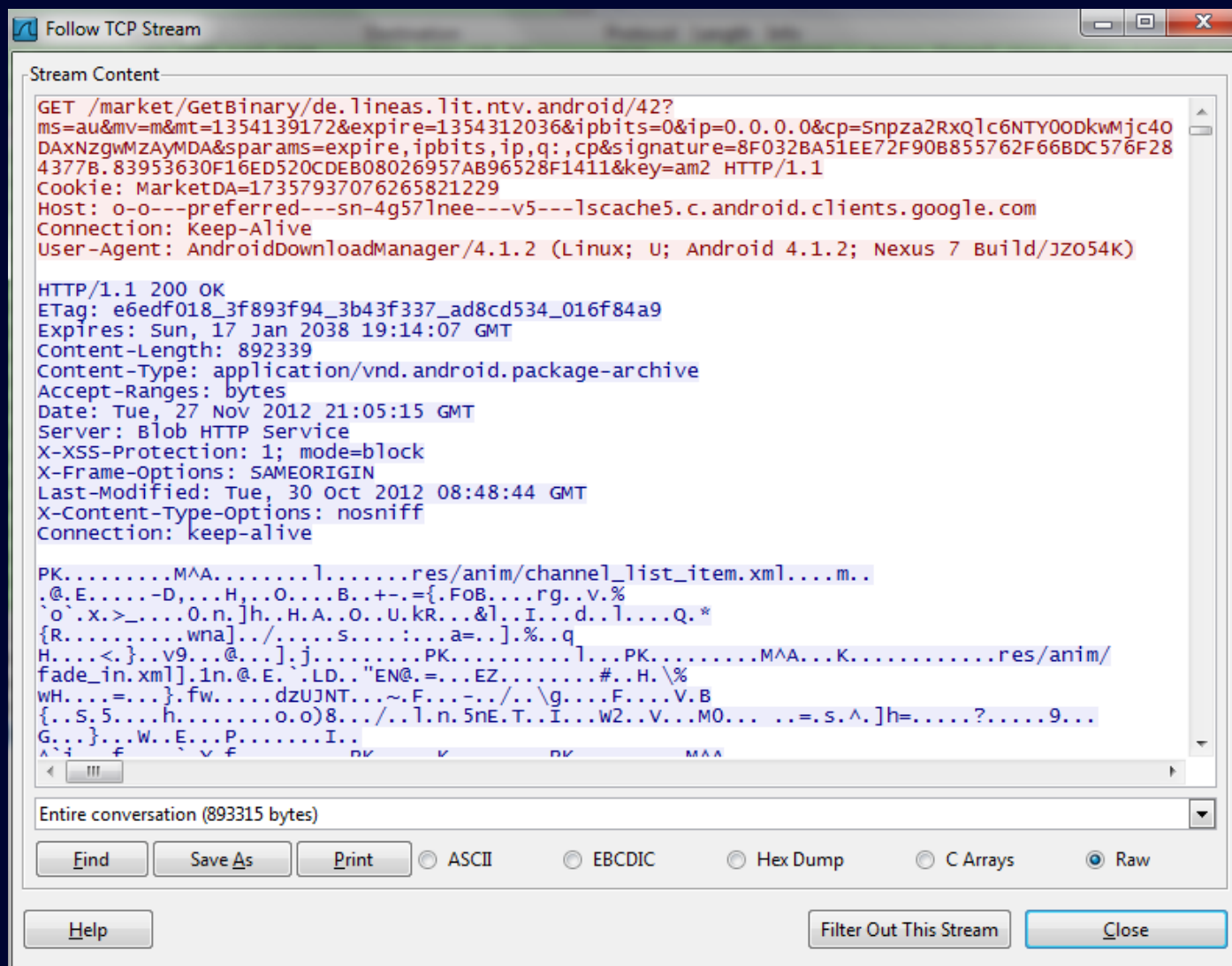
```
googleplay.py: self.androidId = "xxxxxxxxxxxxxx" # change me
config.py:      GOOGLE_LOGIN = ""
config.py:      GOOGLE_PASSWORD = ""
(sudo easy_install protobuf)
```

```
$ python search.py wetter
Title;Package name;Creator;Super Dev;Price;Offer Type;Version
Code;Size;Rating;Num Downloads
wetter.com;com.wetter.androidclient;wetter.com
AG;0;Kostenlos;1;23;7.1MB;4.27;5.000.000+
Yahoo! Wetter;com.yahoo.mobile.client.android.weather;Yahoo!
Inc.;1;Kostenlos;1;91590009;5.8MB;4.37;1.000.000+
WETTER.NET;com.lsapplications.qmet;Q.met
GmbH;0;Kostenlos;1;30;4.5MB;3.81;100.000+
```

```
$ python download.py com.wetter.in.de
Downloading 2.5MB... Done
$
```

Obtaining APKs: Download intercept

Notice



The screenshot shows a 'Follow TCP Stream' window with the following content:

```
Stream Content
GET /market/GetBinary/de.lineas.lit.ntv.android/42?
ms=au&mv=m&mt=1354139172&expire=1354312036&ipbits=0&ip=0.0.0.0&cp=Snpza2RxQ1c6NTY0ODkwMjc4O
DAXNzgWMZAYMDA&sparams=expire,ipbits,ip,q:,cp&signature=8F032BA51EE72F90B855762F66BDC576F28
4377B.83953630F16ED520CDEB08026957AB96528F1411&key=am2 HTTP/1.1
Cookie: MarketDA=17357937076265821229
Host: o-o---preferred---sn-4g57lnee---v5---lscache5.c.android.clients.google.com
Connection: Keep-Alive
User-Agent: AndroidDownloadManager/4.1.2 (Linux; U; Android 4.1.2; Nexus 7 Build/JZO54K)

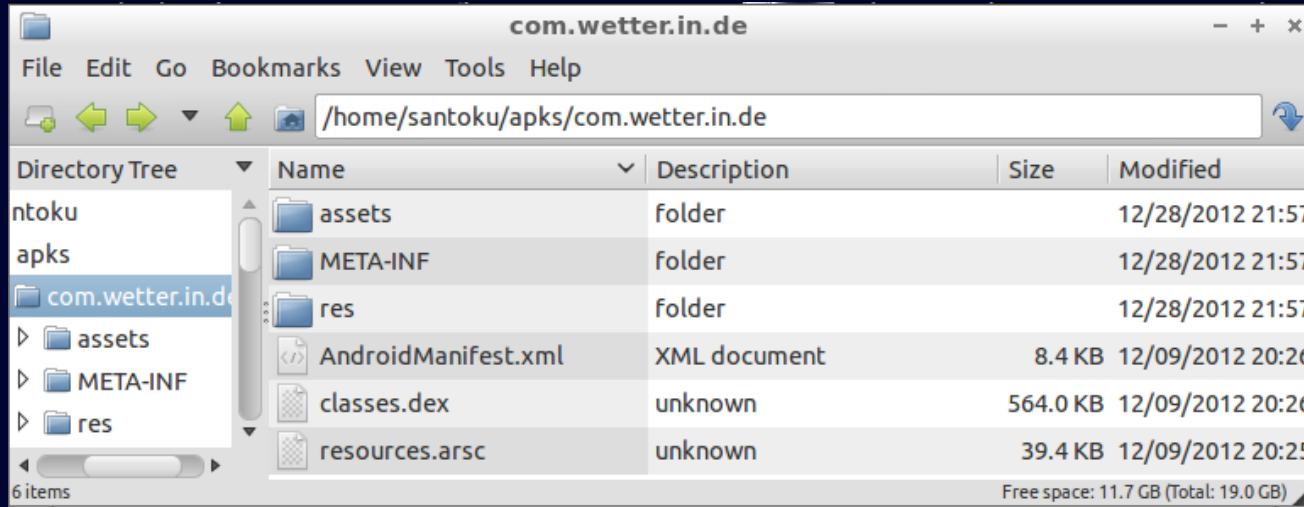
HTTP/1.1 200 OK
ETag: e6edf018_3f893f94_3b43f337_ad8cd534_016f84a9
Expires: Sun, 17 Jan 2038 19:14:07 GMT
Content-Length: 892339
Content-Type: application/vnd.android.package-archive
Accept-Ranges: bytes
Date: Tue, 27 Nov 2012 21:05:15 GMT
Server: Blob HTTP Service
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Last-Modified: Tue, 30 Oct 2012 08:48:44 GMT
X-Content-Type-Options: nosniff
Connection: keep-alive

PK.....M^A.....l.....res/anim/channel_list_item.xml....m..
.@.E.....D....H....O....B..+..={.FoB....rg..v.%
`o`.x.>....O.n.]h..H.A.O..U.kR...&l..I..d..l....Q.*
{R.....wna]../.s.....a=..].%.q
H....<}.v9...@...].j.....PK.....l...PK.....M^A...K.....res/anim/
fade_in.xml].1n.@.E..LD..EN@.=...EZ.....#.H.\%
wH....=...}.fw.....dzUJNT...~.F...-.../..g....F...V.B
{.S.5....h.....o.o)8.../.l.n.5nE.T..I...w2..V...MO... ..=.s.^.]h=.....?.....9...
G...}.W..E..P.....I..
^..i..f..v..f.....nk.....v.....nk.....M^A..
```

Below the stream content, there is a section for the 'Entire conversation (893315 bytes)' with a dropdown menu. At the bottom, there are buttons for 'Find', 'Save As', 'Print', and radio buttons for 'ASCII', 'EBCDIC', 'Hex Dump', 'C Arrays', and 'Raw' (which is selected). There are also 'Help', 'Filter Out This Stream', and 'Close' buttons.

Decompiling (1)

1. UNZIP to get overview



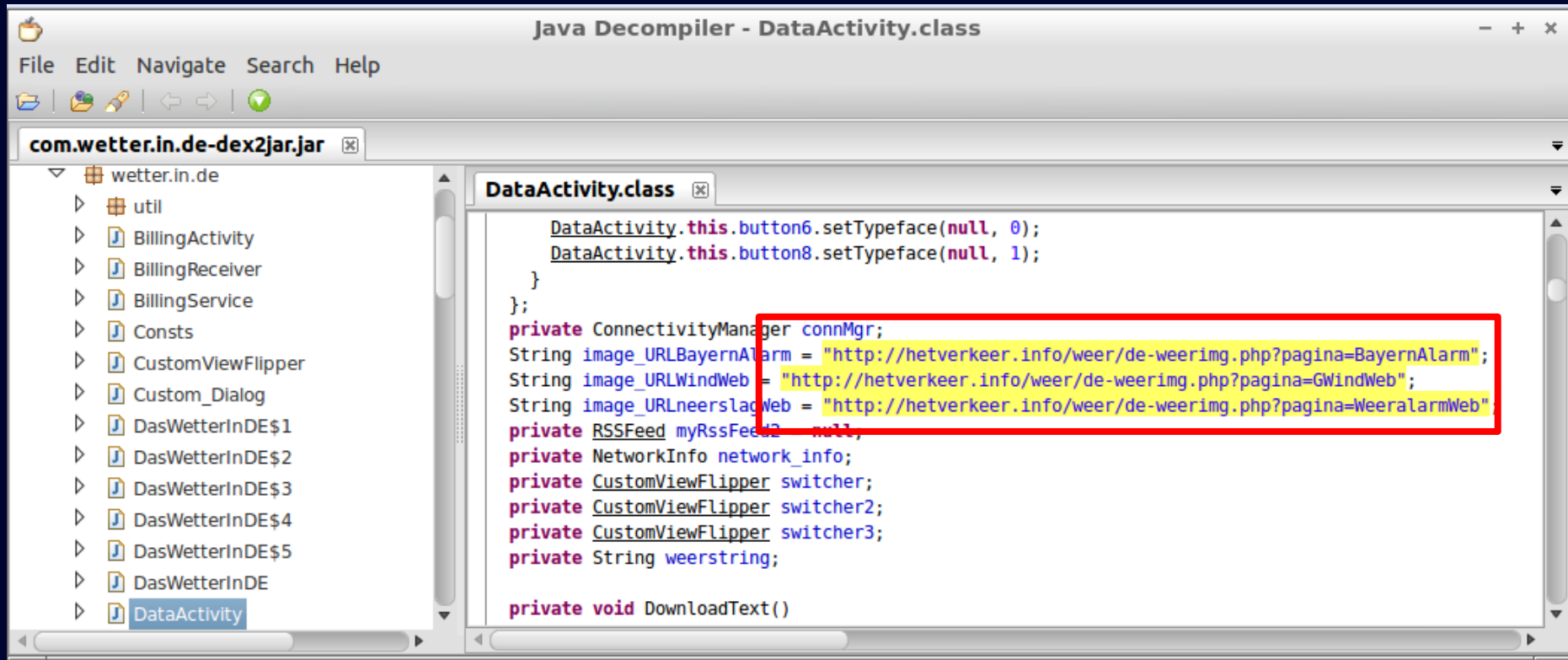
2. DEX2JAR to create JAR

```
$ ./dex2jar.sh de.ratiopharm.pollenradar.apk
this cmd is deprecated, use the d2j-dex2jar if possible
dex2jar version: translator-0.0.9.9
dex2jar de.ratiopharm.pollenradar.apk ->
de.ratiopharm.pollenradar_dex2jar.jar
Done.
```

Decompiling (2)

3. Java Decompiler: JD-GUI or JAD

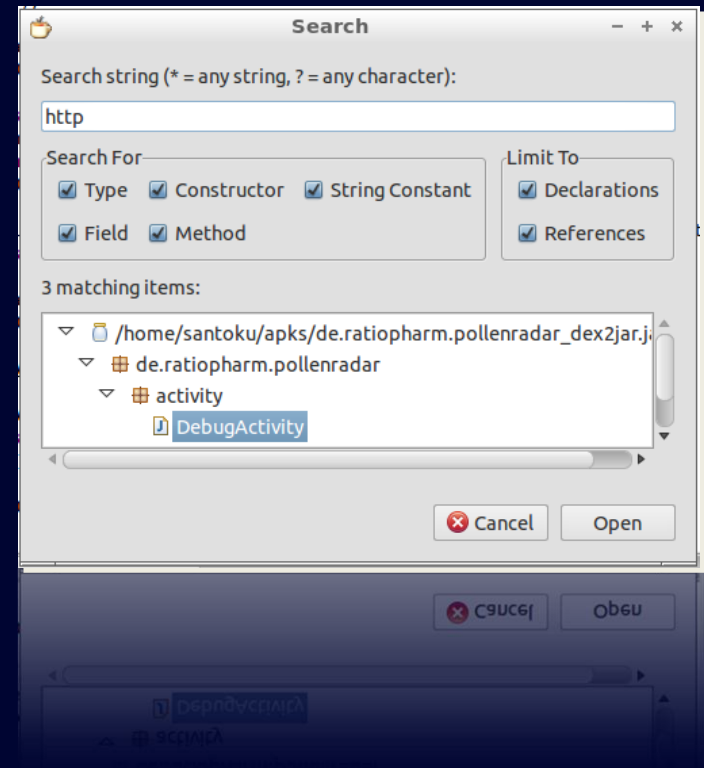
- JD-GUI sometimes stumbles
- JAD is closed source



Decompiling (3)

Hints

- Search broadly for well-known strings
- Search in code and resources (XML)
- Sometimes “http” is obfuscated



Resource and code manipulation

Easy



Edit installed resource on device

- Look for /sdcard/data/data
- Local data usually in XML-files

Medium



Edit resource in APK

- UNZIP, edit, ZIP
- Sign with self-created certificate

Hard



Change essential code

- APKTOOL: (BAK)SMALI into macro-language
- Sign with ... certificate

Repackaging and signing

1. Download current Java **SDK** (JDK)
2. Generate certificates with JDK keytool

```
santoku@santoku-VirtualBox:~$ keytool -genkey -alias GoogleInc -keyalg  
RSA -keysize 2048 -validity 100000 -keystore keystore
```

3. Sign JAR with JDK jarsigner

```
$ jarsigner -verbose -keystore ~/keystore -signedjar demo_new.apk  
demo.apk GoogleInc
```

ADD-ON: SMALI

- Macro language
- Easy to read (?)
- Easy to re-assemble

```
.method private
OpenHttpConnection(Ljava/lang/String;)Ljava/i
o/InputStream;
    .locals 7
    .parameter "strURL"
    .annotation system
Ldalvik/annotation/Throws;
        value = {
            Ljava/io/IOException;
        }
    .end annotation

    .prologue
    .line 333
    const/4 v3, 0x0

    .line 334
    .local v3,
inputStream:Ljava/io/InputStream;
    new-instance v4, Ljava/net/URL;

    invoke-direct {v4, p1}, Ljava/net/URL;-
><init>(Ljava/lang/String;)V
```


Where to go from here

- APKTOOL, SMALI and BAKSMALI
- Mercury
 - Device component (APK)
 - Host component
 - Dynamic analysis of running system
 - New version 2.0 of Dec. 2012
- Exploits (Rage against the cage) and VM manipulation

Thank you

E-mail: `sam@ccc.de`

Jabber: `sam@jabber.ccc.de`