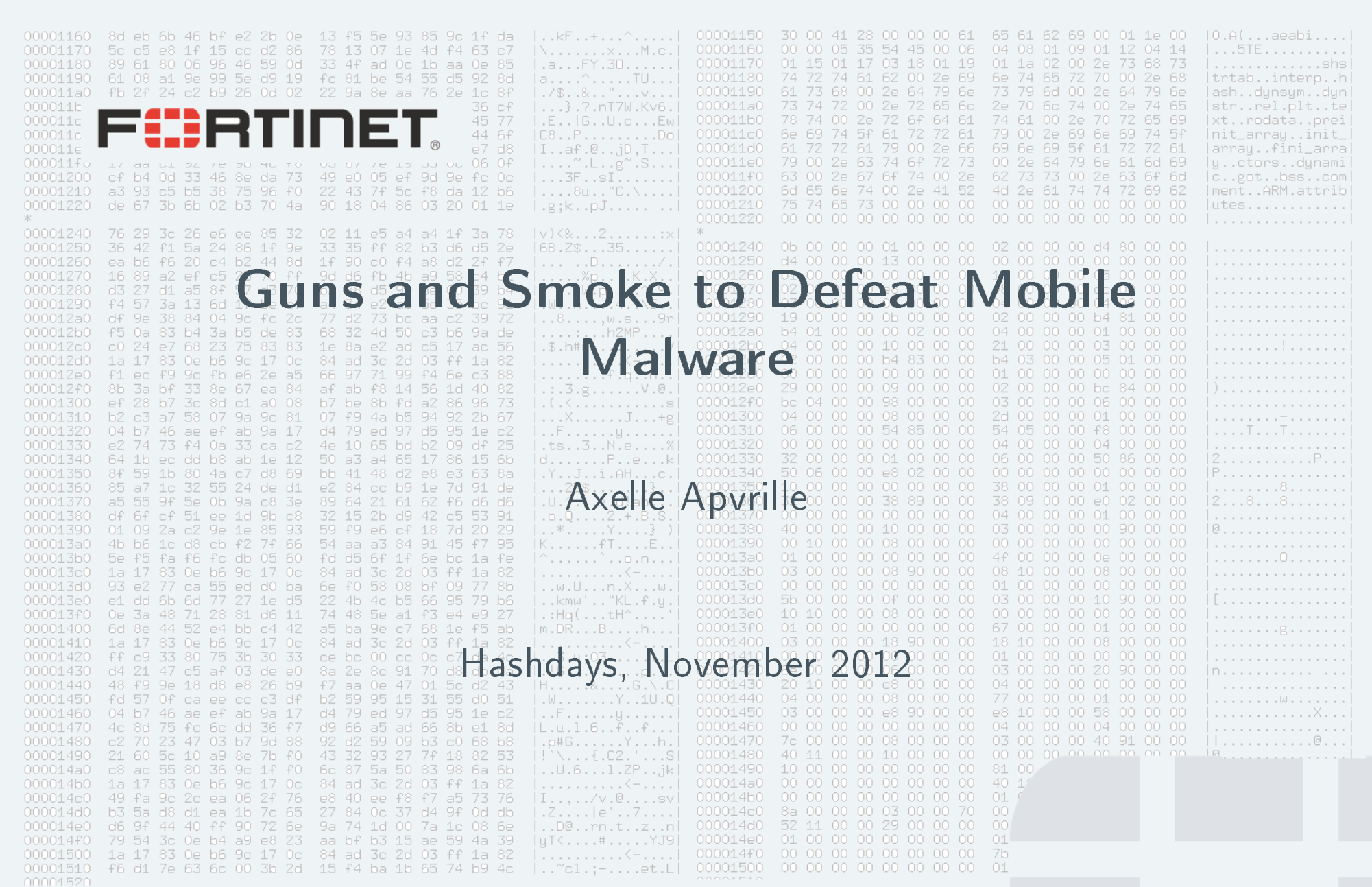


Guns and Smoke to Defeat Mobile Malware

Axelle Apvrille

Hashdays, November 2012

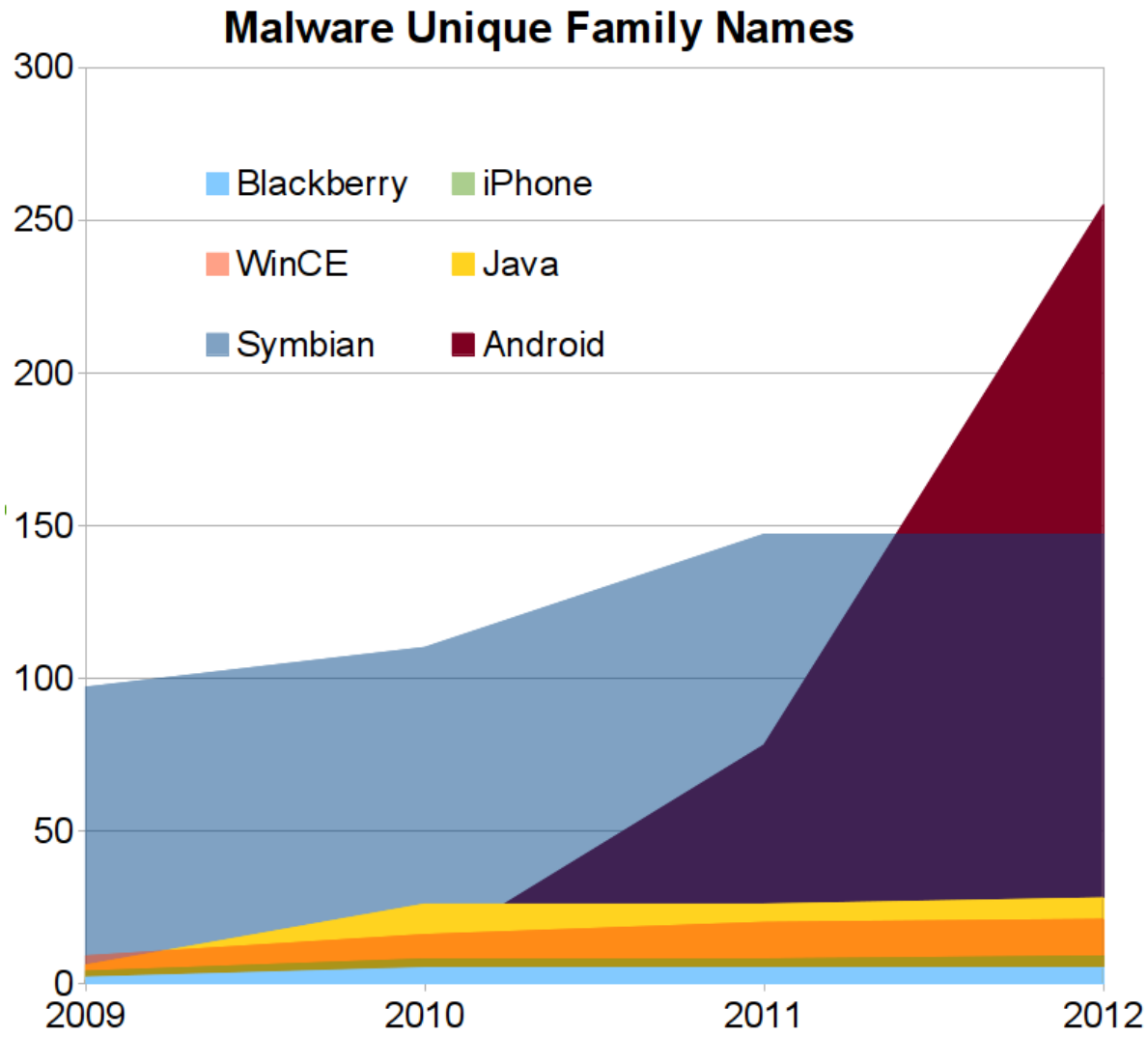
Who am i

whoami

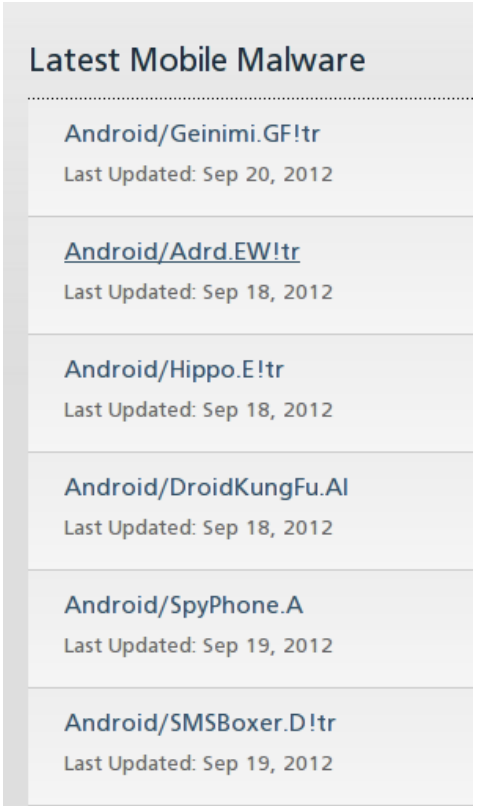
```
#!/usr/bin/perl -w

my $self = {
    realname => 'Axelle Apvrille',
    nickname => 'Crypto Girl',
    twitter => '@cryptax',
    job => 'Mobile Malware Analyst and Researcher',
    # reverse engineering of incoming mobile malware
    # research and tools in related areas
    title => 'Senior', # white hair
    company => 'Fortinet, FortiGuard Labs',
    before => 'Security software eng.: protocols, crypto...',
    languages => 'French, English, Hexadecimal :)'
};
```

1 Family = several variants = several samples



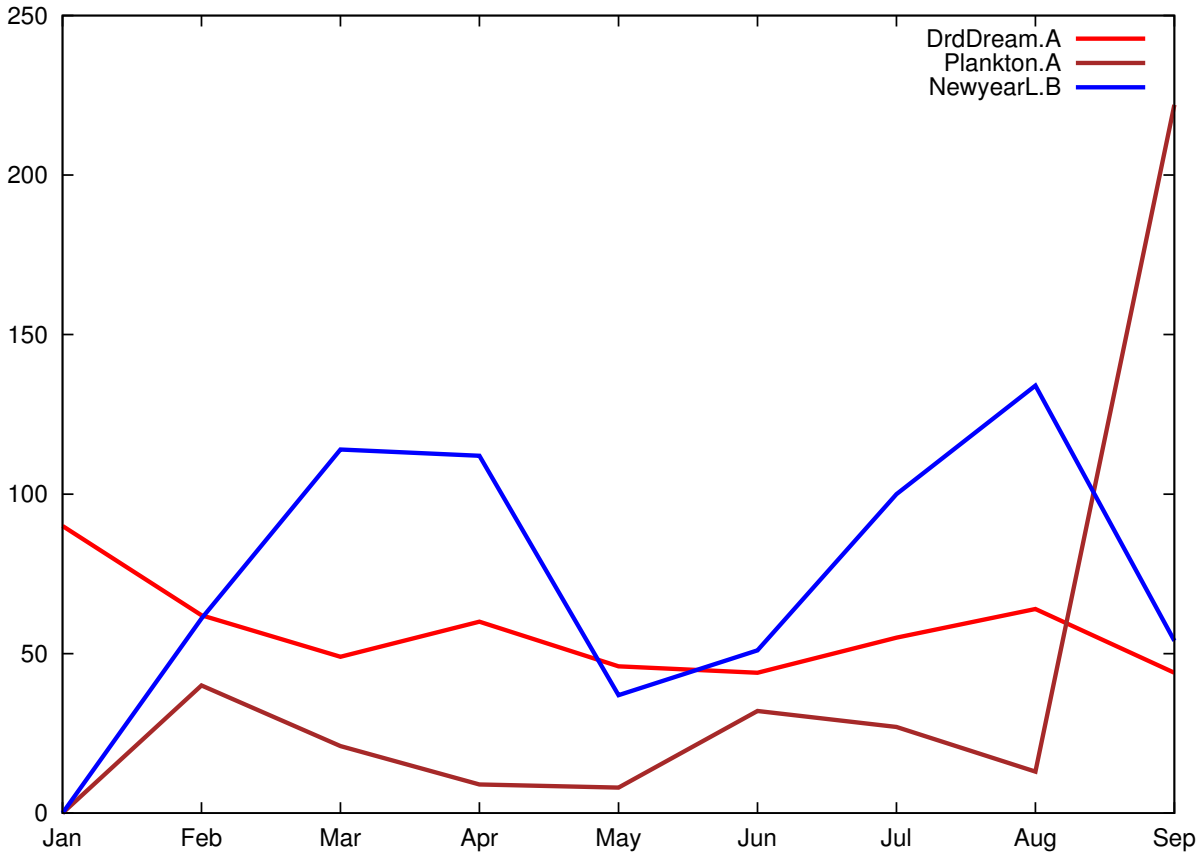
Recent Android malware /excerpt/



- Android/Fakelash: fake Flash Player, spies SMS
- Android/Fakemart: propagates in France, SMS toll fraud
- Android/Vidro: porn video viewer, SMS toll fraud
- Android/SMSZombie: malicious wallpaper, SMS toll fraud
- Android/Spyoo: location, SMS, call, browsing spyware
- Android/Luckycat: remote backdoor
- Android/Zitmo: mobile 'version' of ZeuS, grabs mTANs
- Android/Vdloader: wallpaper, commands from C&C

Recent detections: http://www.fortiguard.com/library.html

Top EMEA Android Malware Download Hits in 2012



Download hits?

- downloads via Internet
- hits per *signature* per *month*

Far below reality

- for FortiGates, *when stats enabled.*
- EMEA only. **Worldwide: DroidDream > 300 hits / month etc.**

Previously - @ Insomni'Hack 2012

- Reading the Manifest: AXMLPrinter
- Reading the resources: aapt dump, apktool
- Disassembling: dexdump, smali/baksmali, dedexer, ded
- Java output: dex2jar, jd-gui
- Swiss knives: Androguard, Apktool
- Adding logs
- Customizing emulator IMEI and IMSI
- Permissions, similarities, visualization with Androguard
- Misc: APKInspector, DroidBox, AndBug

Download my slides

http://www.fortiguard.com/sites/default/files/insomnidroid.pdf

Today

- A bit more on *decompilers*
- Tools to play with the DEX format **New stuff inside**
  - Reading the header, parsing a dex
  - De-obfuscating obfuscated bytecode
  - Disassembling at a given offset
  - Make your own dex
- Demo :)
- Playing with the *Device ID*
- Evading *emulator detection* **New stuff inside**
- Eavesdropping malicious *SMS messages*

Comparing Decompilers

Tests

- Test with the same Android malware: Android/Fakemart
- Decompile the same method: `com/twodboy/worldofgoofull/function;->GetKeyCode()`
- 4 decompilers: Java Decompiler, DJ, DED, DAD

1- Java Decompiler

- Free: download from http://java.decompiler.free.fr/
- Not specific to Dalvik: provide .jar or .class (e.g use dex2jar)

2- DJ

- Free to try, but not free. There are older free versions.
- Works with Wine on Unix
- Not specific to Dalvik: provide .class (e.g use dex2jar)

Other decompilers

3- DED

- http://siis.cse.psu.edu/ded/
- Provide APK or DEX
- Option -c to decompile:
  `./ded.sh -d output -c my.apk`

4- DAD

- Located in Androguard
- Select class, then method to decompile
- Or use via Androlyze:
  `AnalyzeAPK('xyz.apk', decompiler="dad")`

Java Decompiler

```
public static void GetKeyCODE(String paramString, Context paramContext)
  {
    while (true)
    {
      String str1;
      try
      {
        if (GetMyMXNot("FNB", paramContext).equals("123786"))
        {
          str1 = DownloadFromUrl("http://script. [....]");
          if (str1.equals("Erreur Of The Dead"))
          {
            StringTOP("USN", "PASS", paramContext);
```

Issues

- Strange while(true) loop decompilation
- Failures or crashes to decompile some other functions

DJ decompiler

```
public static void GetKeyCODE(String s, Context context)
{
    com/twodboy/worldofgoofull/function;
    JVM INSTR monitorenter ;
    if(!GetMyMXNot("FNB", context).equals("123786")) goto _L2; else got
_L1:
    String s1 = DownloadFromUrl("http://script. [...]");
    if(!s1.equals("Erreur Of The Dead")) goto _L4; else goto _L3
_L3:
    StringTOP("USN", "PASS", context);
```

Issues

- Unexpected monitorenter instruction
- Many gotos

Oops

```
Exception in thread "main" java.lang.RuntimeException:
 Could not verify approximated Synchronized body!
```

Another function

```
r22 = (Object[]) (Object[]) r6.get("pdus");
r23 = new SmsMessage[r22.length];
r9 = SmsMessage.createFromPdu((byte[]) (byte[]) r22[0]).getMessageBody(
function.GetKeyCODE(r1.getPackageName(), r1);
```

Issues

- Fails to decompile some classes :(
- Casts not very readable

DAD

```
public static synchronized void GetKeyCODE(String p18, android.content.
    {
        synchronized(com.twodboy.worldofgoofull.function) {
            if(com.twodboy.worldofgoofull.function.GetMyMXNot("FNB", p1
                v2 = com.twodboy.worldofgoofull.function.DownloadFromUr
                if(v2.equals("Erreur Of The Dead") == 0) {
                    if(v2.indexOf("<span id="sksmskeyword" class="bigte
```

Issues

- Strange synchronized start
- Long lines because full path for all methods

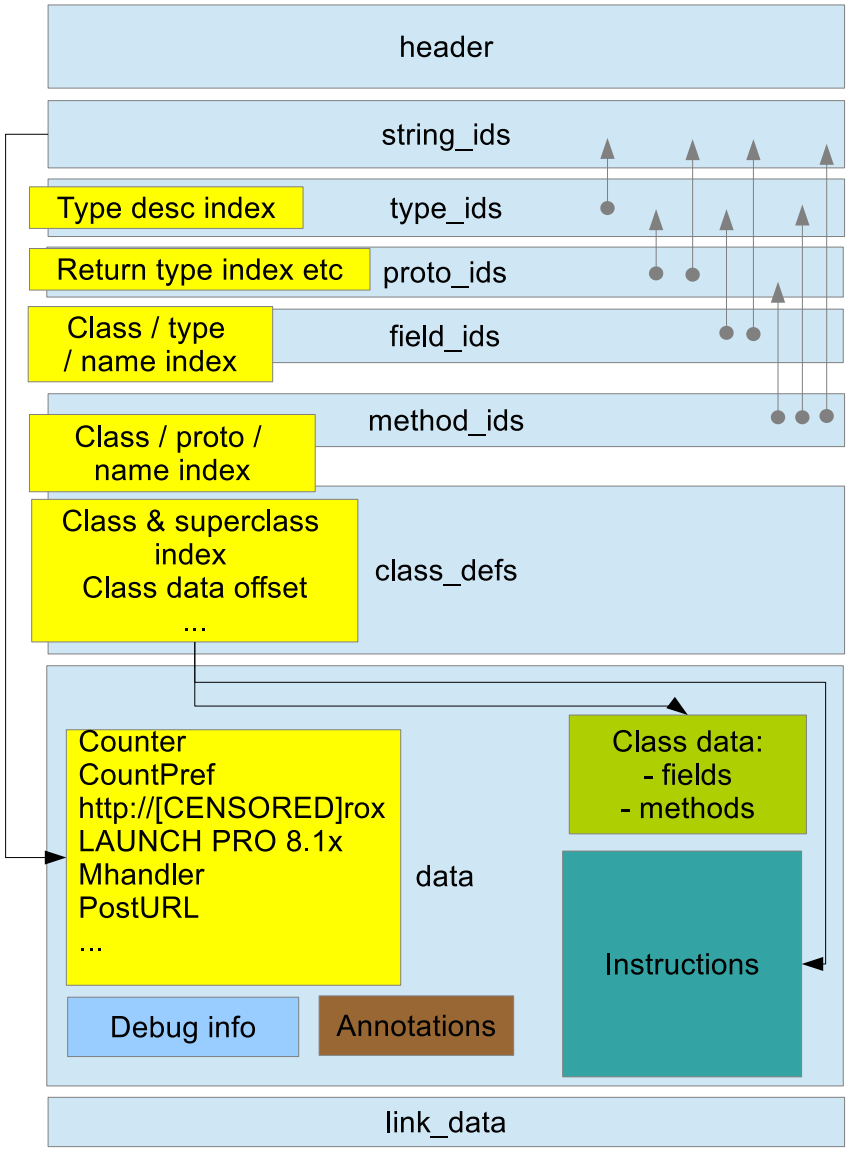
No decompiler is perfect

Good to have several decompilers and know their pros and cons.
Smali, dalvik bytecode to read.

DEX format

Structure of a DEX

- A header + several arrays (strings, types, prototypes...)
- Data is located in the data section
- Arrays store offsets to the data section, or indexes into other arrays



DEX references

- Dex Format http://source.android.com/tech/dalvik/dex-format.html
- Bytecode for the Dalvik VM http://www.netmite.com/android/mydroid/dalvik/docs/dalvik-bytecode.html
- Dalvik VM Instruction Formats http://www.netmite.com/android/mydroid/dalvik/docs/instruction-formats.html

Reading the DEX header

Name	Format
magic	ubyte[8] = DEX_FILE
checksum	uint
signature	ubyte[20]
file_size	uint
header_size	uint = 0x70
endian_tag	uint = ENDIAN_CONST
link_size	uint
link_off	uint
map_off	uint
string_ids_size	uint
string_ids_off	uint
type_ids_size	uint
type_ids_off	uint
proto_ids_size	uint
proto_ids_off	uint
field_ids_size	uint
field_ids_off	uint
method_ids_size	uint
method_ids_off	uint
class_defs_size	uint
class_defs_off	uint
data_size	uint
data_off	uint

DexNfo

- Get source from Tim Strazzere's blog
- Compile: `javac DexNfo.java`
- Run: takes the Dex file as argument

Androlyze - Anthony Desnos et al.

```
d = DalvikVMFormat( open('classes.dex',
                'rb').read())
header = d.get_header_item()
header.show()
```

Malware

Haven't seen malware with magic != dex 035

Parsing the DEX format: 010 Editor

- 010 Editor: not free, but free trial http://www.sweetscape.com/010editor/
- Dex Template for 010 Editor: https://github.com/jlarimer/android-stuff

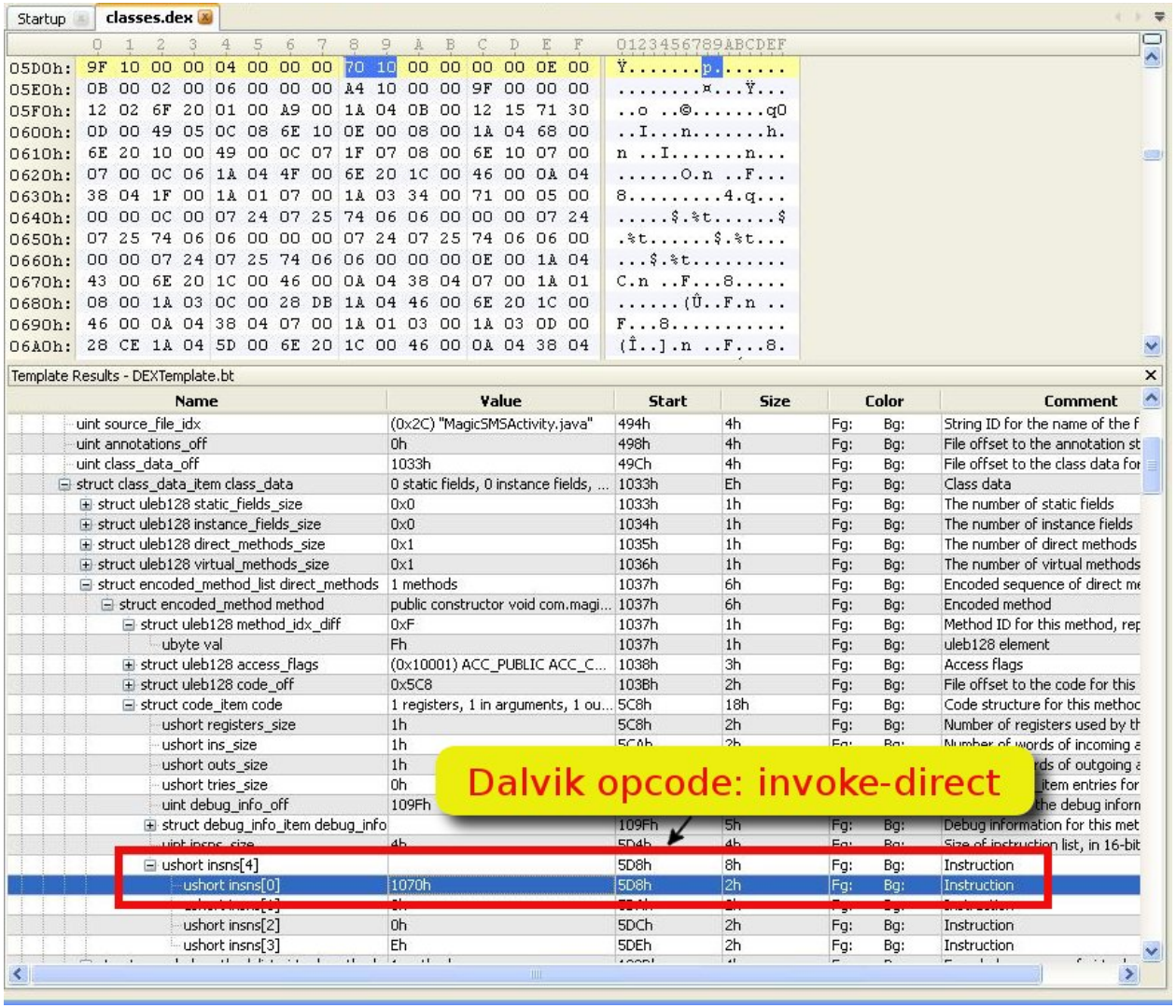
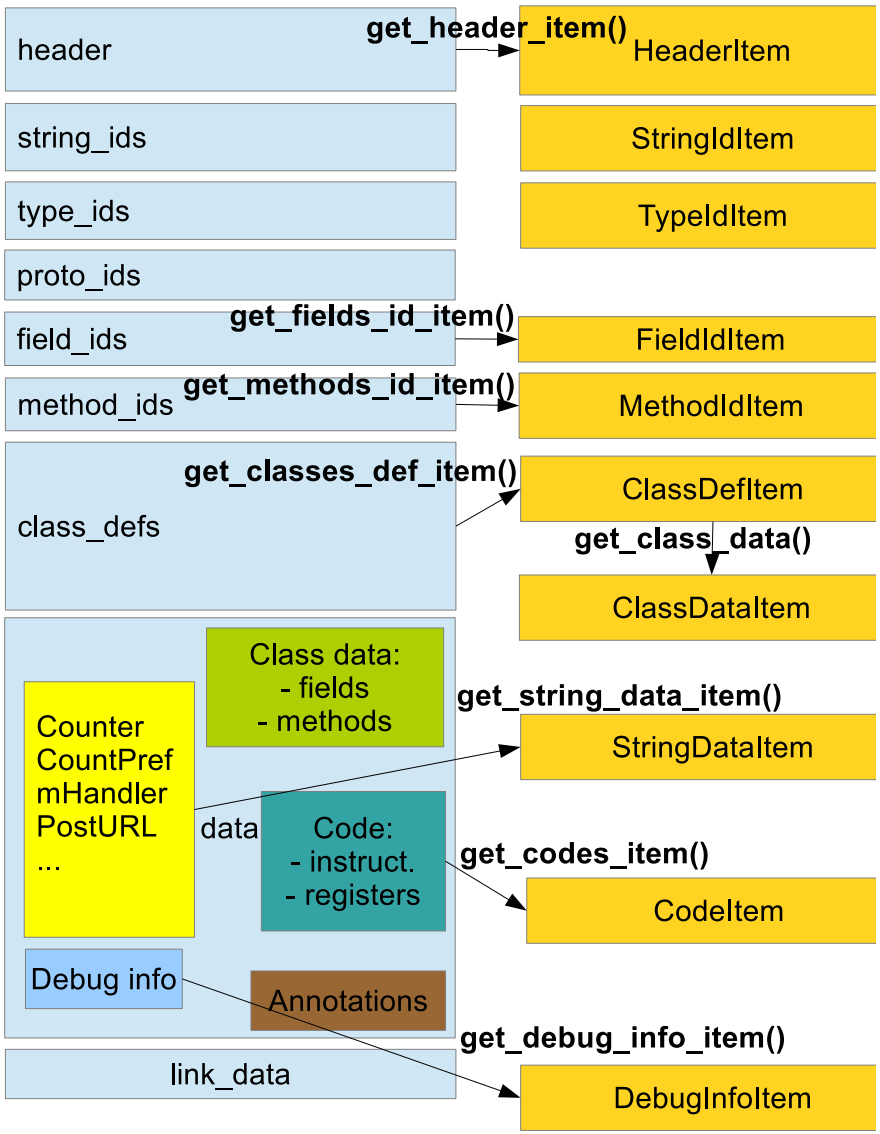


Figure: Dalvik bytecode for Android/Foncy.A!tr

Parsing the DEX format with Androguard



Androguard

- Reads DEX files + has classes & methods for each part, see dvm.py

Classes

- DalvikVMFormat: use this to parse & decompile the DEX
- ClassManager: access to all elements of the DEX. Usually, don't use directly but provide object to other methods.
- H = list of : Type**H**IdItem list of TypeIdItem, Proto**H**IdItem list of ProtoIdItem ...

Useful methods of Androguard

Methods

- show, pretty_show: prints the contents
- get_raw: raw buffer (hexadecimal)
- get/set_off: offset from the beginning of the file
- get/set_idx: manipulate array indexes. For instructions, 16-bit units!
- shortcuts from DalvikVMFormat: get_strings() returns all referenced strings, get_classes() returns list of ClassDefItem, get_methods() returns list of EncodedMethod...

Can't remember names?

- Use completion of commands in `androlyze`
- Online documentation http://doc.androguard.re
- Export parsed DEX entries to Python: dalvikvm.create_python_export()

Androguard Examples

Listing all classes

```
In [1]: d = DalvikVMFormat( open('classes.dex', 'rb').read())
In [2]: d.get_classes_names()
..
 'Lcom/twodboy/worldofgoofull/SmsReceiver$1;',
 'Lcom/twodboy/worldofgoofull/SmsReceiver;',
 'Lcom/twodboy/worldofgoofull/WorldofGoo;',
 'Lcom/twodboy/worldofgoofull/function;',
..
```

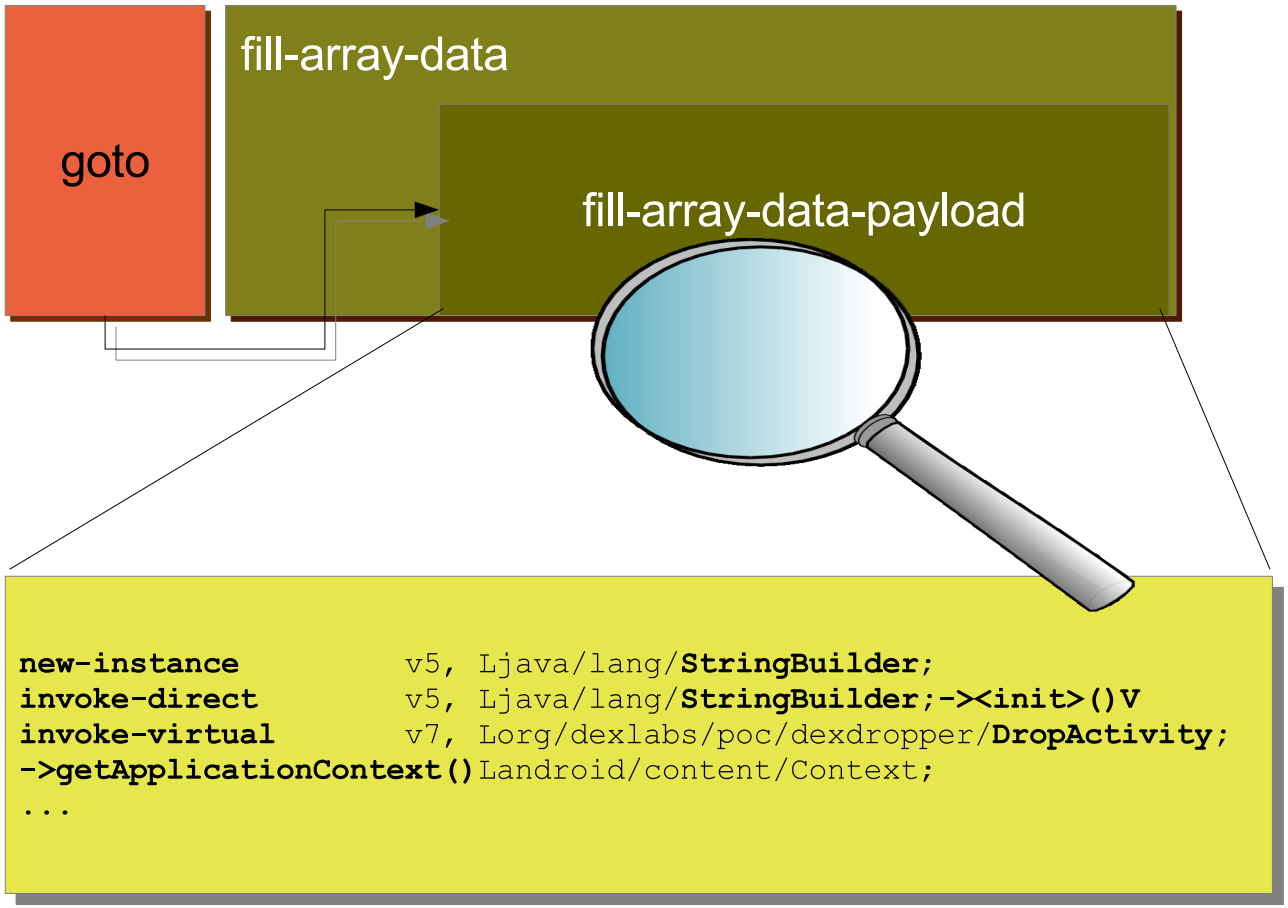
Listing all methods

```
In [1]: d = DalvikVMFormat( open('classes.dex', 'rb').read())
In [2]: for current_method in d.get_methods():
              print current_method.show_info()
```

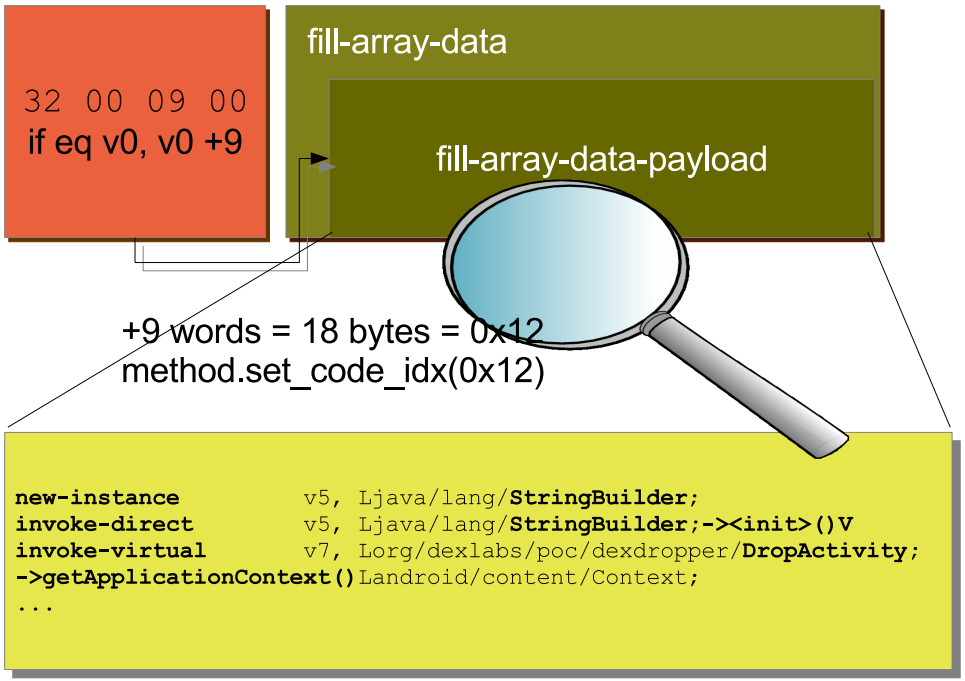
De-obfuscating obfuscated bytecode

Dexlabs challenge - @thuxnder

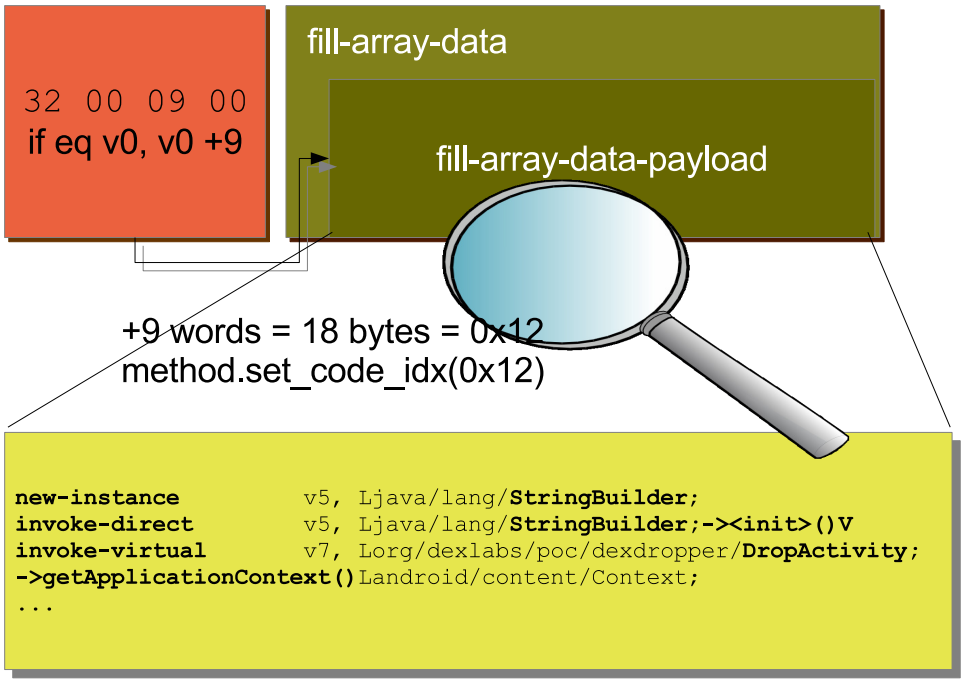
Get the crackme from Dexlabs



"Official" solution with Androguard



"Official" solution with Androguard



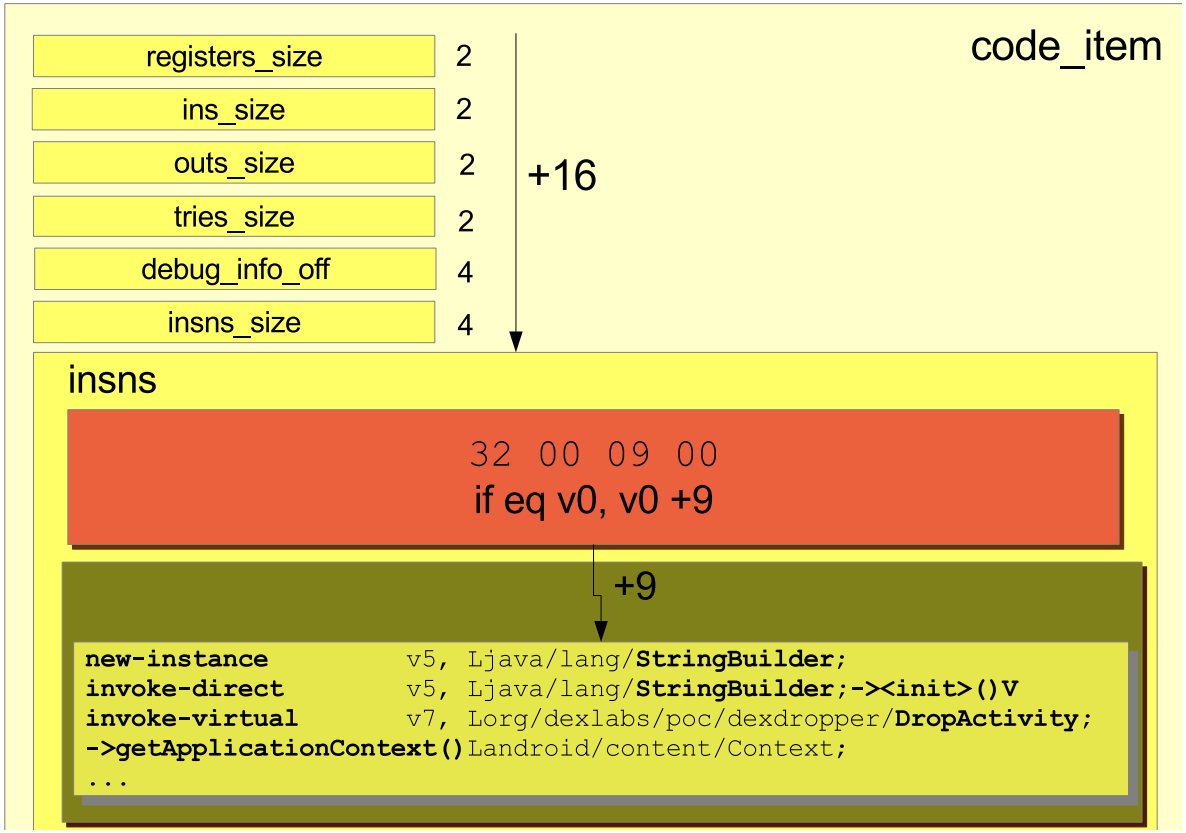
What if...

- set_code_idx() had not been added?
- disassemble at an arbitrary location?

Disassembling at a given offset

Feasible without patch

- Get code offset: *method*.get_code_off()
- Find offset of real beginning: +16 (1st code_item fields) + $9 * 2$ (skipping array)



Disassembling at a given offset

Androdis - an Androguard Python script

- Standalone Python script that uses the Androguard API
- Usage: `./androdis.py -i DEXFILE -o OFFSET`
- Provide correct offset (or disassembly won't make sense)
- Will add to Androguard ASAP - or ask me for source :)

Demo

Time to wake up

Hey, it's demo time!

In case the demo does not work :) lol

Androlyze

```
In [1]: a, d, dx = AnalyzeAPK( 'crackme-obfuscator.apk', decompi
In [2]: execm = d.CLASS_Lorg_dexlabs_poc_dexdropper_DropActivity
In [3]: mydex = a.get_dex()
In [4]: execm.get_code_off()
Out[4]: 219900
In [5]: execm.get_code().get_insns_size()
Out[5]: 108
In [6]: DCode(d.CM, 108-9, mydex[219900+16+18:230000]).show()
```

Used against Dexlabs challenge :)

```
$ ./androdis.py --input dexlabs-classes.dex --offset 0x35b1e
0 0x0 new-instance v5, Ljava/lang/StringBuilder;
1 0x4 invoke-direct v5, Ljava/lang/StringBuilder;-><init>()V
2 0xa invoke-virtual v7, Lorg/dexlabs/poc/dexdropper/DropActivity;
   ->getApplicationContext()Landroid/content/Context;
3 0x10 move-result-object v6
4 0x12 invoke-virtual v6, Landroid/content/Context;
   ->getFilesDir()Ljava/io/File;
```

Modify a DEX and re-package

Fix DEX header

- Re-compute the modified DEX sha1 (skip magic, checksum and sha1)
- Re-compute checksum (skip magic and checksum)
- dexrehash: https://github.com/cryptax/dextools (Perl)
- A Java version exists Tim's Re-DEX

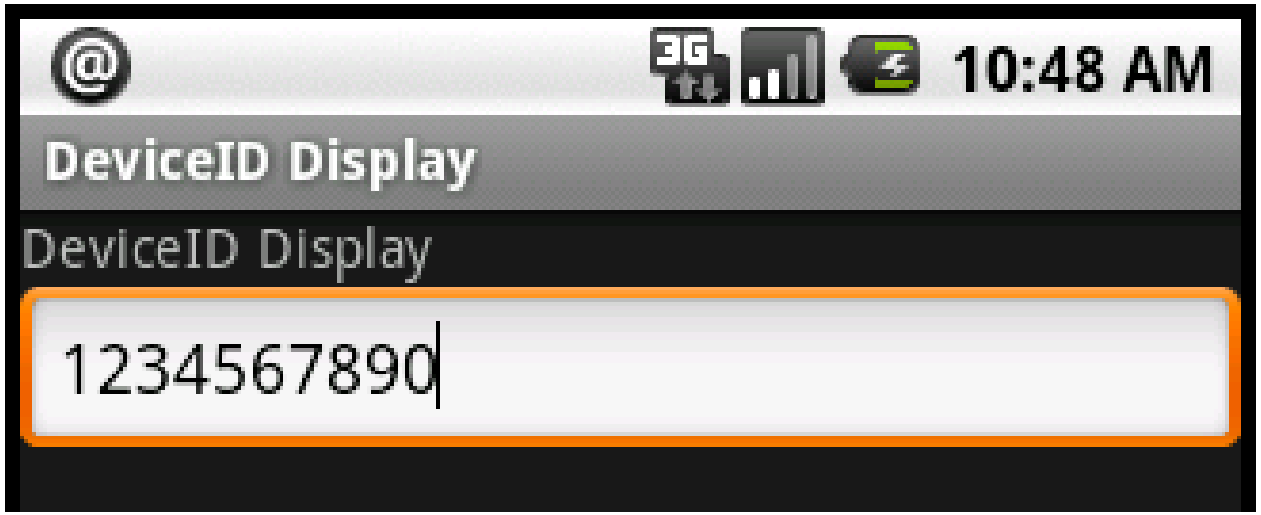
Re-package APK

Copy:

- modified classes.dex
- compiled resources: resources.arsc + res/* (layout and images)
- binary AndroidManifest.xml
- assets

Then zip all and sign with jarsigner. **Use a Makefile**

What's my Device ID?




```
import android.provider.Settings.Secure;
..
udid = Secure.getString(getContentResolver(),
  Secure.ANDROID_ID);
```

Modifying the Device ID

adb shell - need to be root

```
# cd /data/data/com.android.providers.settings/databases/
# sqlite3 settings.db
..
sqlite> update secure set value='1234567890' where name='android
sqlite> select * from secure;
..
146|android_id|1234567890
```

Evade Emulator Detection

Detecting the emulator

Typically done by checking system properties:

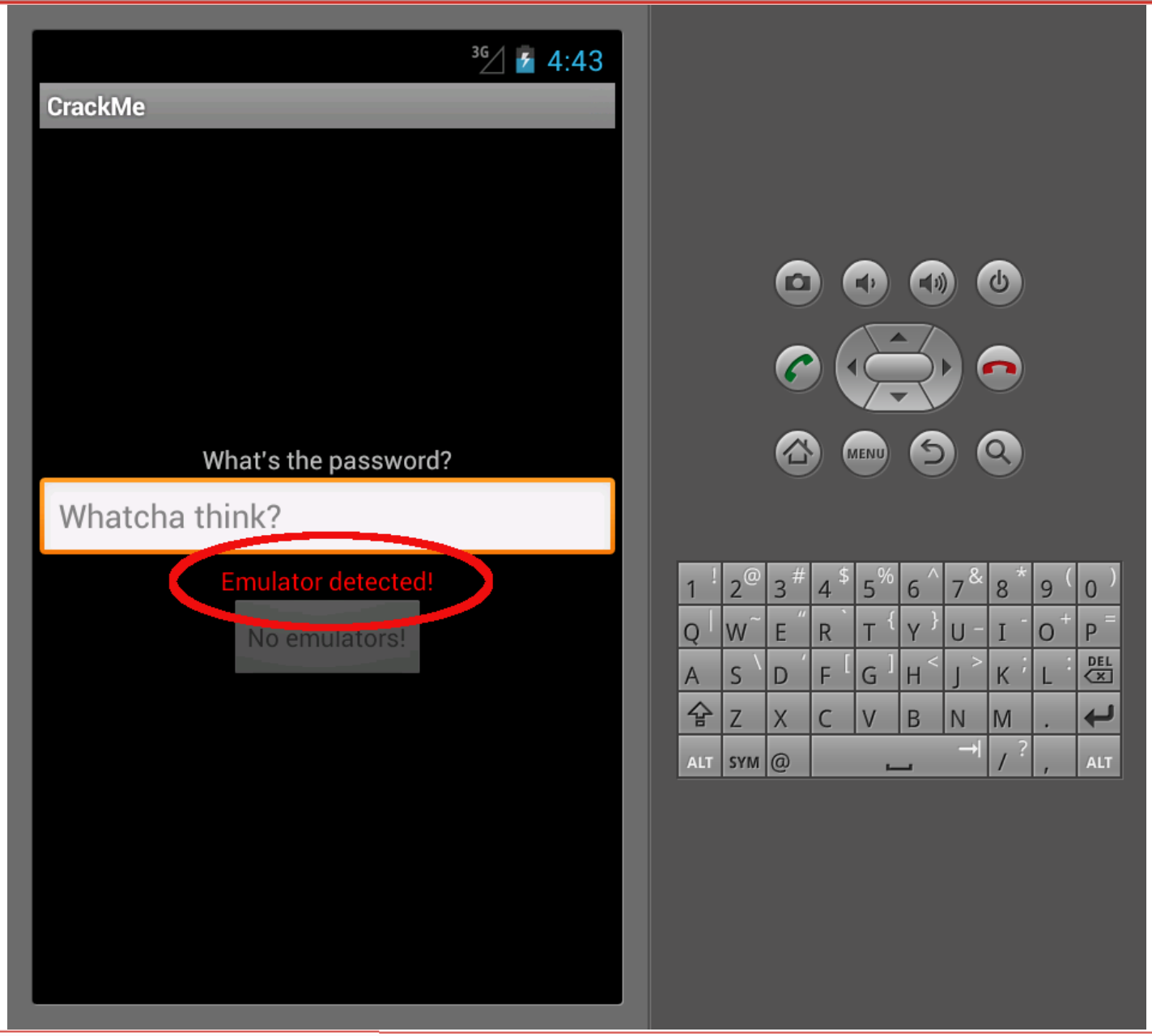
- gsm.version.ril-impl: android reference-ril 1.0
- ro.product.brand: generic
- ro.product.name: sdk
- ro.product.model: sdk
- ro.kernel.qemu: 1
- ...

Checking properties

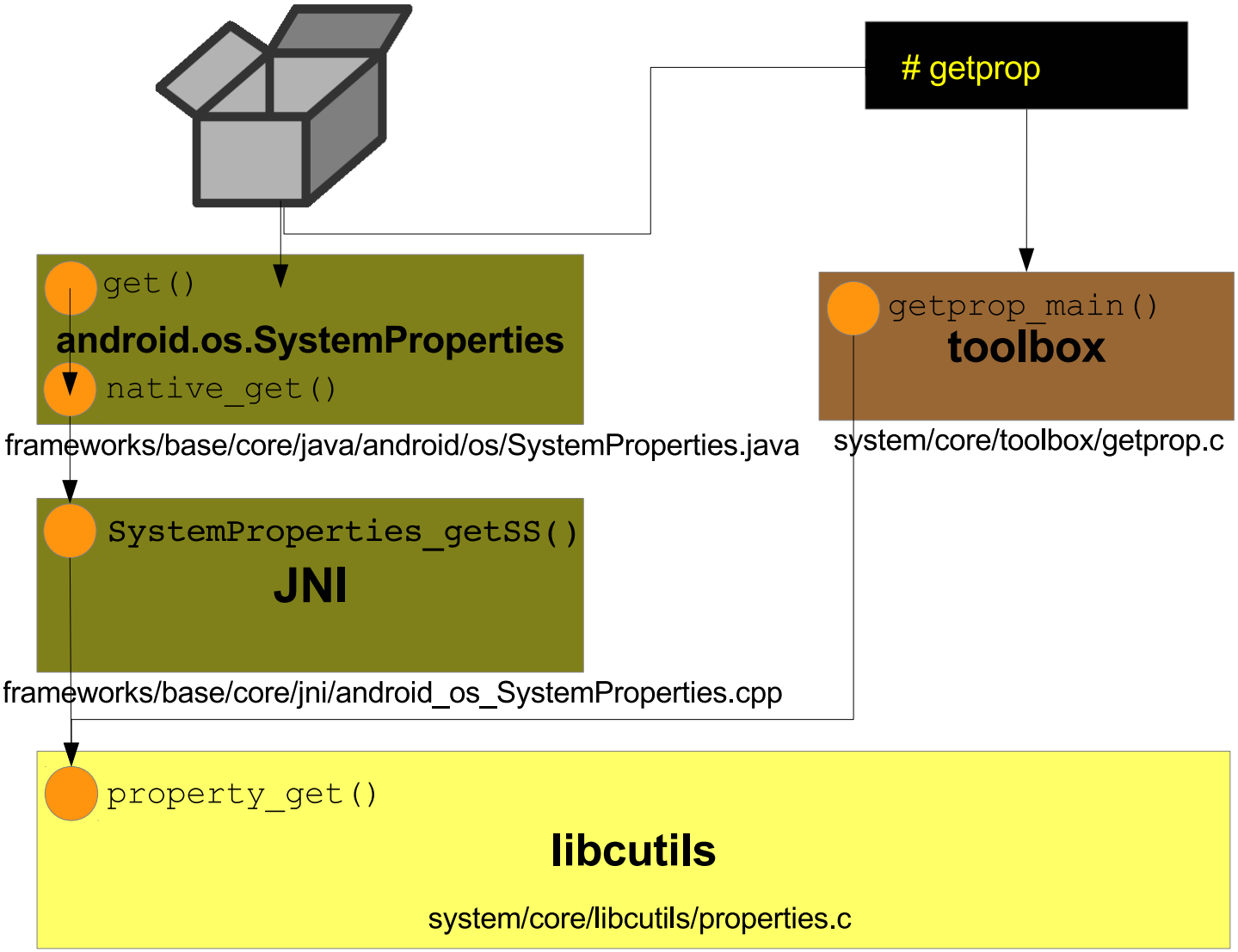
- Calling `android.os.SystemProperties.get()` via Reflection
- Executing the `getprop` command

```
Runtime.getRuntime().exec("getprop...")
```

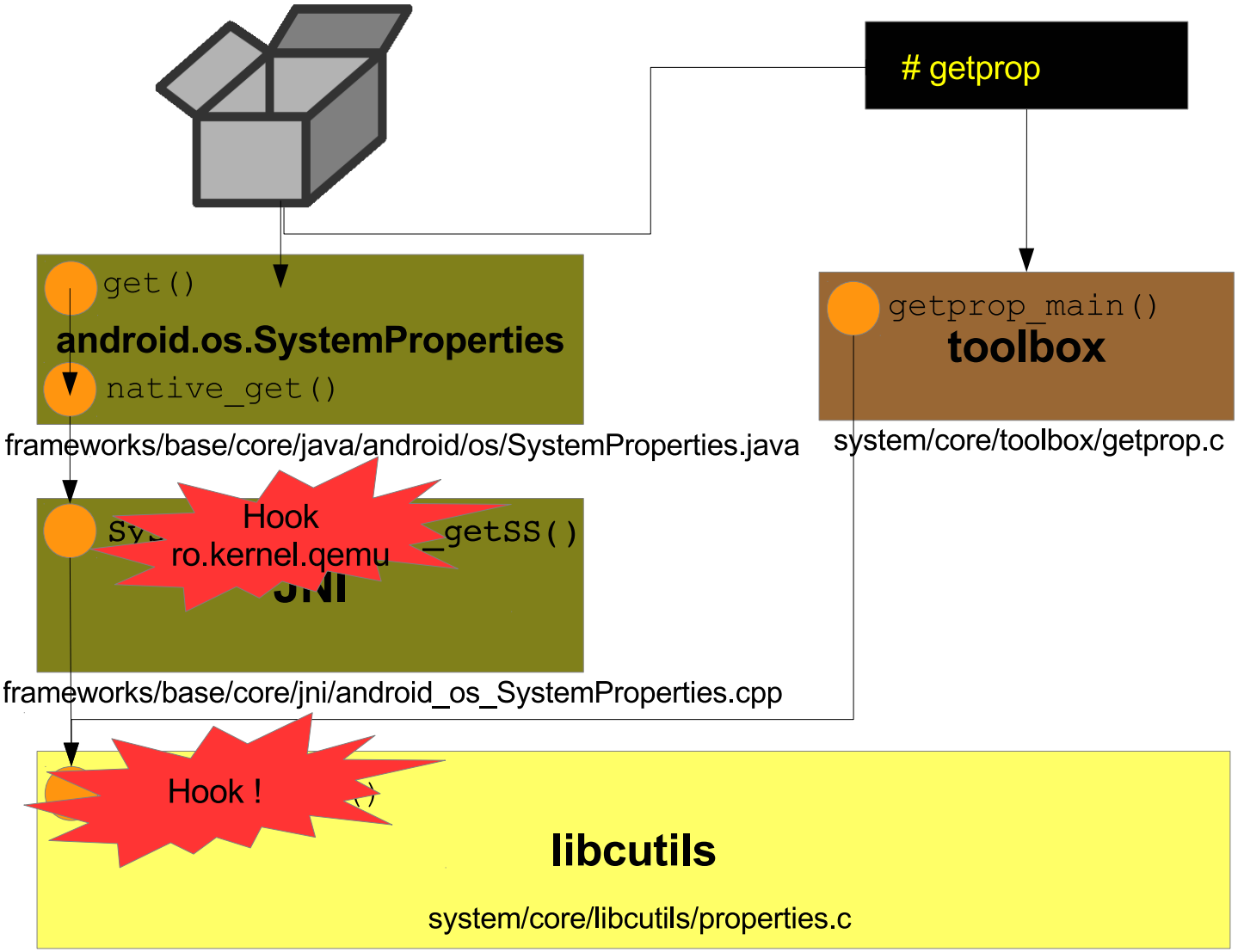
Emulator Detected



Properties - Implementation



Properties - Hooking



Where to hook?

Location

- Hook in libcutils, except ro.kernel.qemu - or emulator won't boot
- Hook ro.kernel.qemu in JNI for e.g

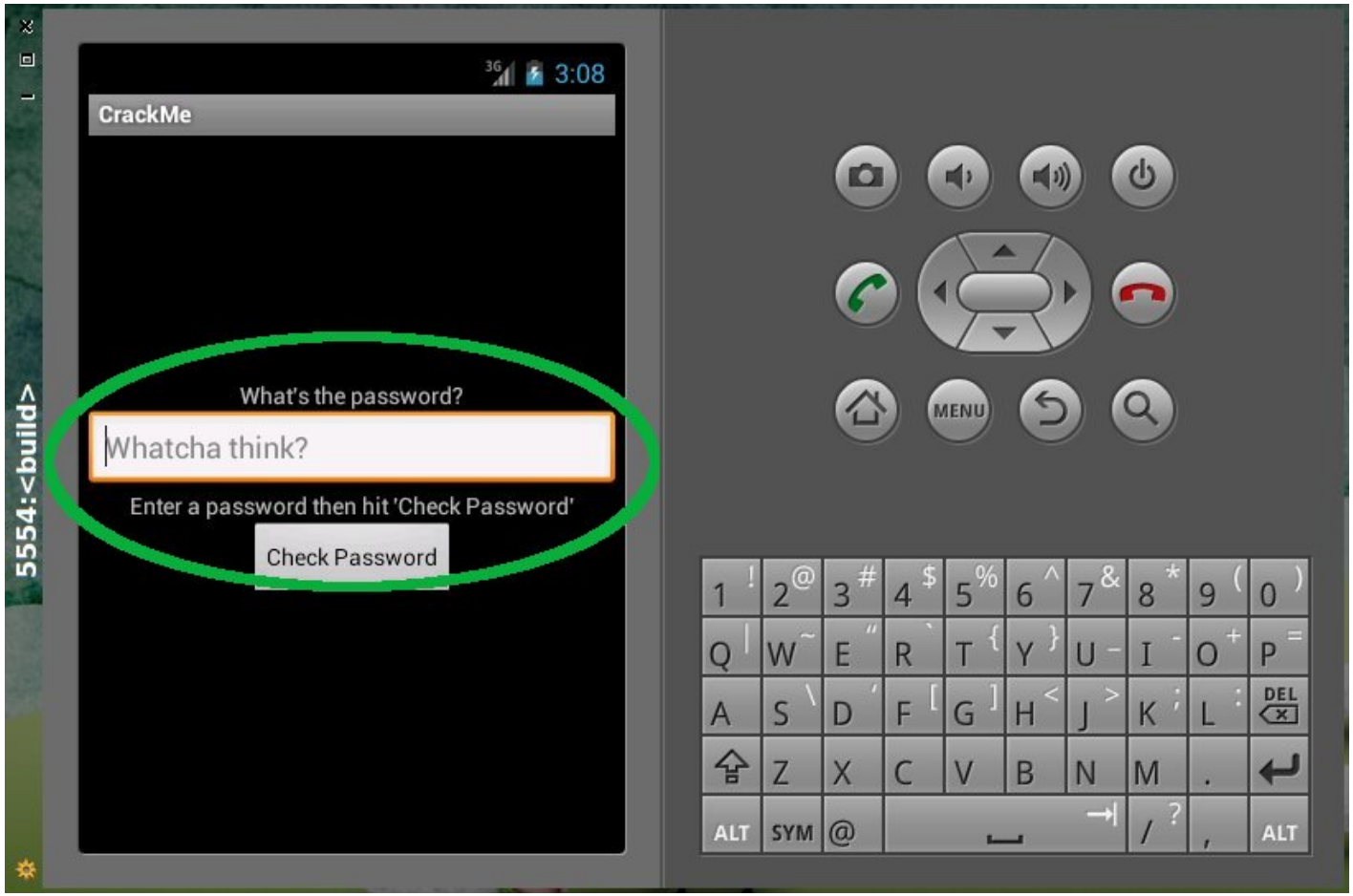
Build custom emulator

Get AOSP, then:

```
$ . build/envsetup.sh
$ lunch full-eng
$ make
$ ./out/host/linux-x86/bin/emulator -debug-init
-sysdir out/target/product/generic/
-system out/target/product/generic/system.img
-ramdisk out/target/product/generic/ramdisk.img
-data out/target/product/generic/userdata.img
-no-audio -no-boot-anim
-partition-size 1024 &
```

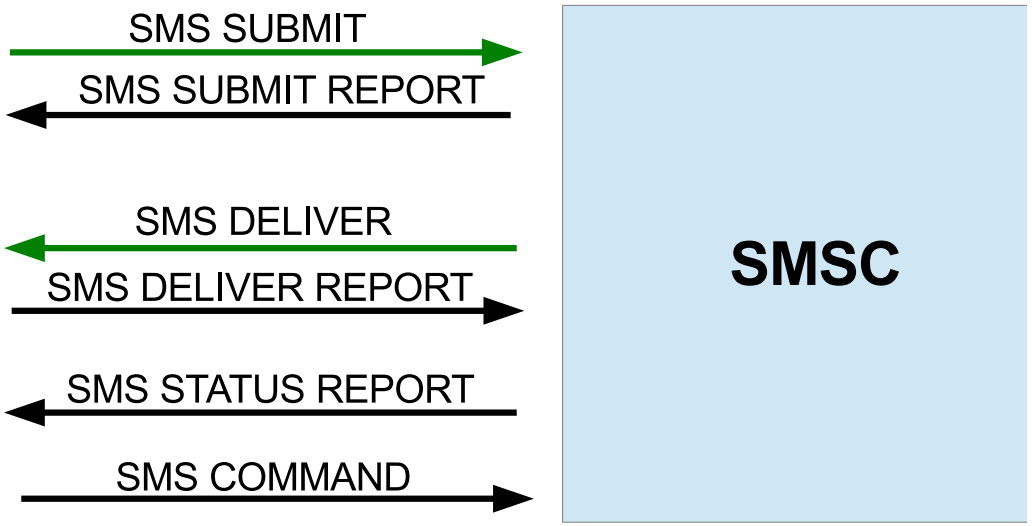
Hacked Emulator :)

The app does not detect it is running on an emulator :)

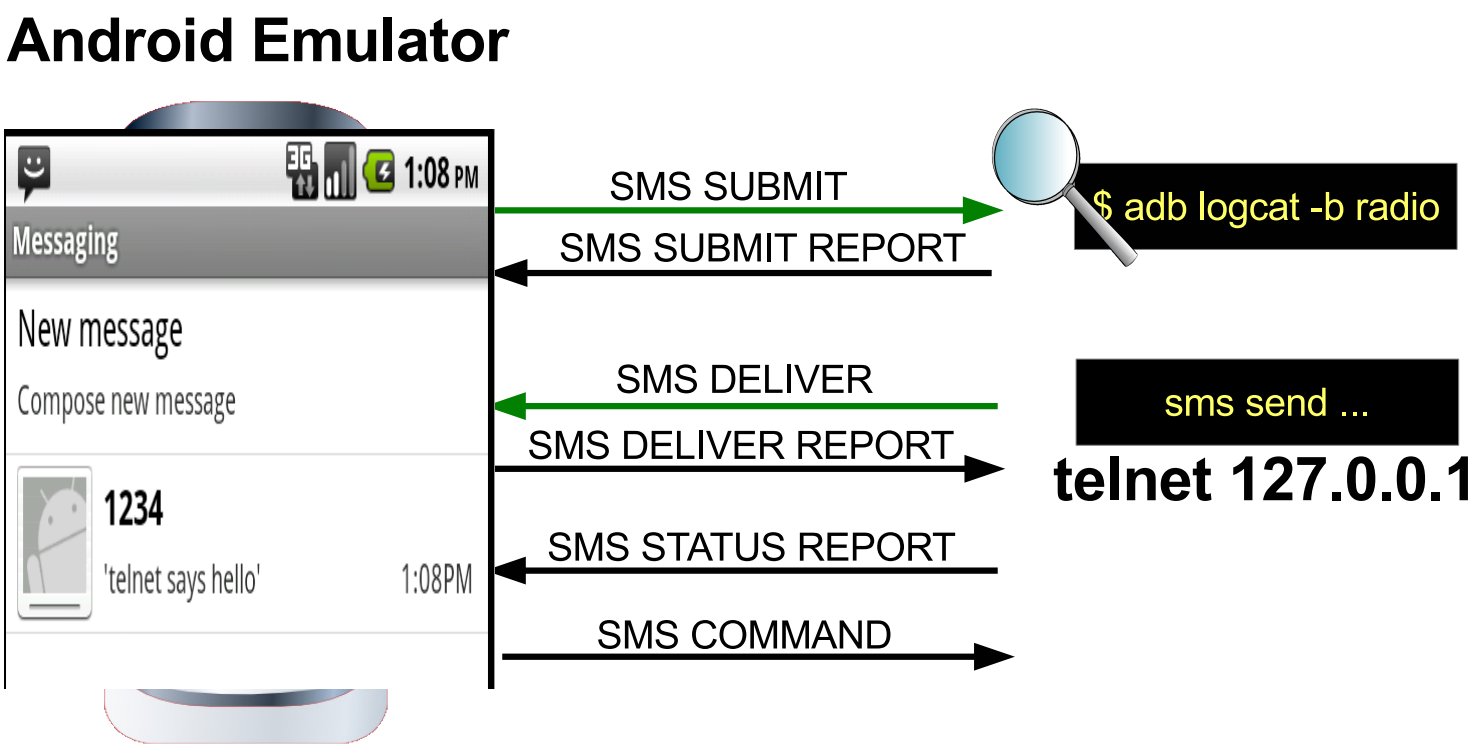


SMS messaging

Cell phone



Eavesdropping SMS



Eavesdropping SMS: Examples

From telnet

```
sms send 1234 'telnet says hello'
```

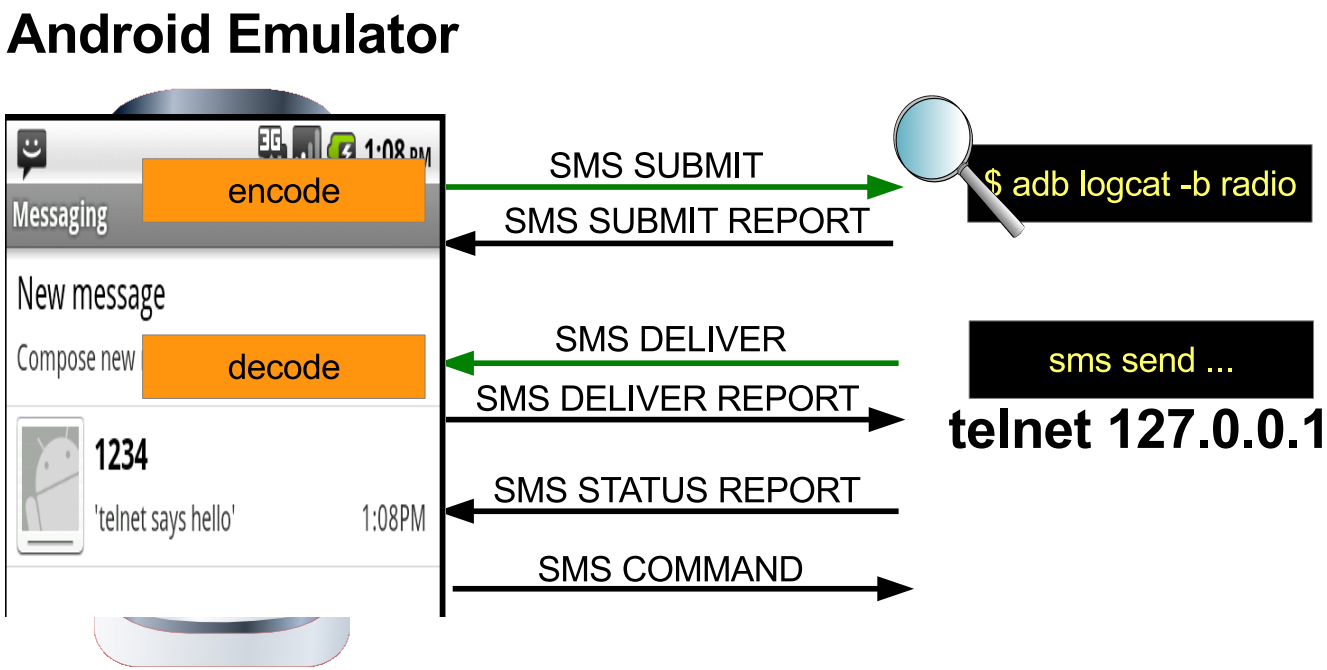
In logcat

```
D/RILJ    (  118): [0491]< SIGNAL_STRENGTH {7, 99, 0, 0, 0, 0, 0
D/AT      (   32): AT< +CMT: 0
D/AT      (   32): AT< 00200481214300002101906152538013277a99ed2
D/RILJ    (  118): [UNSL]< UNSOL_RESPONSE_NEW_SMS
D/GSM     (  118): SMS SC timestamp: 1349792735000
D/SMS     (  118): New SMS Message Received
```

Look for:

- AT+CMGS: send a SMS
- AT+CIMI: retrieve the IMSI
- AT+CGSN: get the IMEI
- AT+CPIN?: test if the PIN is set or not

SMS messaging with python-messaging



https://github.com/pmarti/python-messaging

Encoding/Decoding SMS with python-messaging

Encoding a SMS SUBMIT

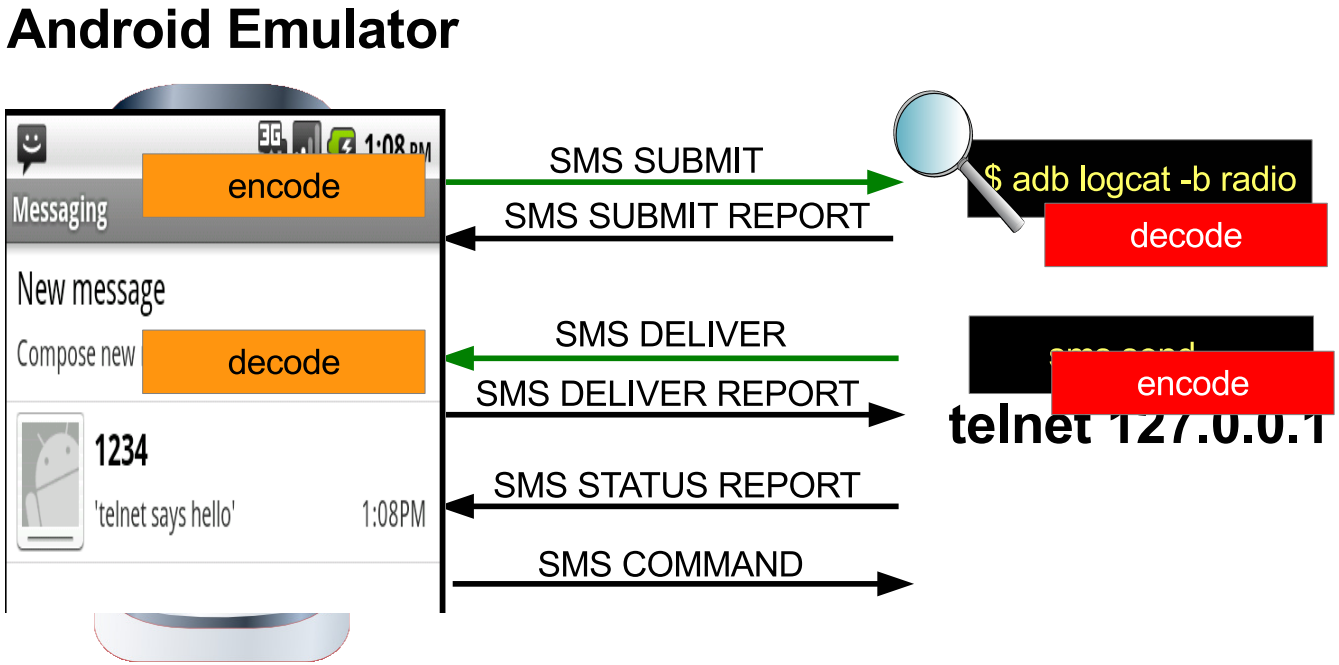
```
from messaging.sms import SmsSubmit
sms = SmsSubmit("+33610203040", "Hi there")
pdu = sms.to_pdu()[0]
print pdu.length, pdu.pdu
20 0001000B913316203040F0000008C834888E2ECBCB
```

Decoding SMS DELIVER

```
from messaging.sms import SmsDeliver
pdu = "0020048121430000210190413100801 0e8329bfd36cbdfedf1db3d7fb
sms = SmsDeliver(pdu)
print sms.data
{'csca': None, 'sr': None, 'type': None, 'date':
datetime.datetime(2012, 10, 9, 12, 13),
'text': u'hellofromconsole', 'fmt': 0, 'pid': 0,
'dcs': 0, 'number': '1234'}
```

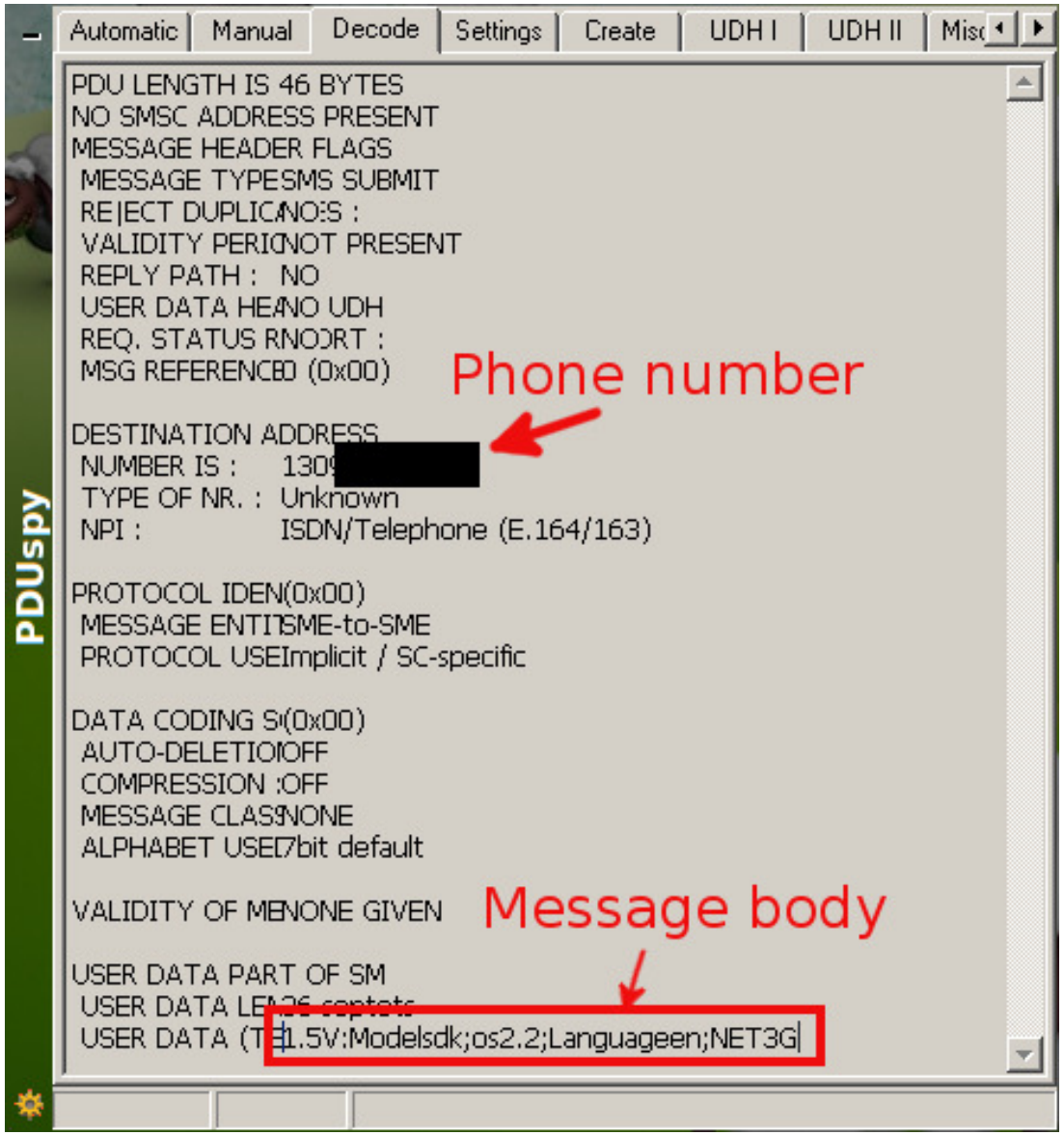
Missing parts in python-messaging

The really useful [red] parts are (currently) missing...



Do it with PDU Spy

Decode PDUs with http://www.nobbi.com/pduspy.html

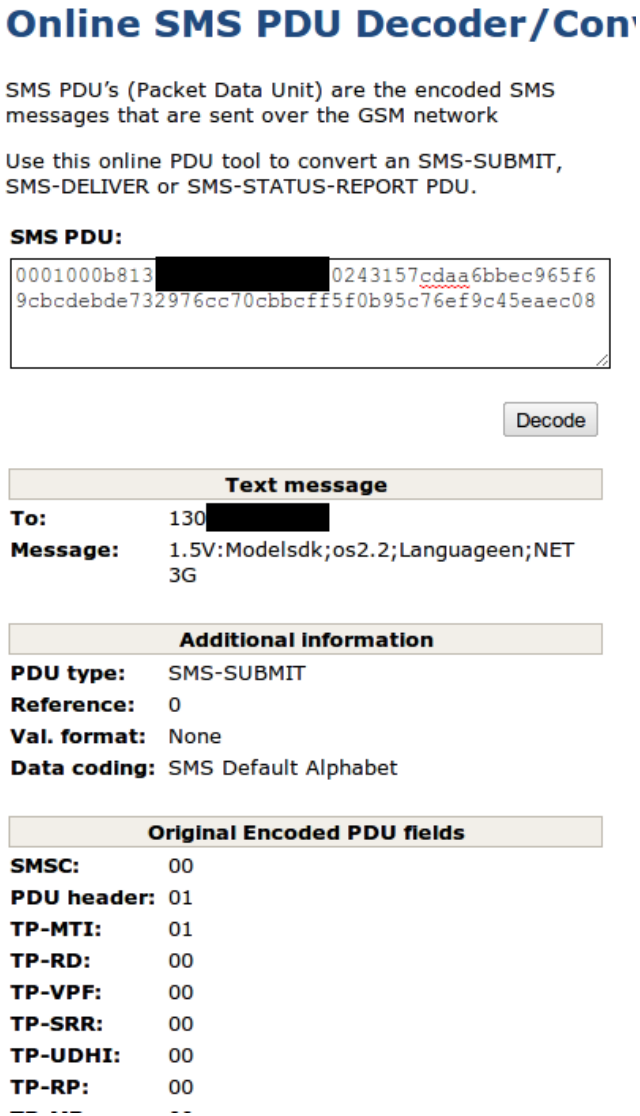


Alternative: online decoding

Online decoder

Copy paste PDU from logcat

http://www.diafaan.com/sms-tutorials/gsm-modem-tutorial/online-sms-pdu-decoder/



Online SMS PDU Decoder/Con

SMS PDU's (Packet Data Unit) are the encoded SMS messages that are sent over the GSM network

Use this online PDU tool to convert an SMS-SUBMIT, SMS-DELIVER or SMS-STATUS-REPORT PDU.

SMS PDU:

0001000b813[redacted]0243157cdaa6bbec965f6
9cbcdebde732976cc70cbbcff5f0b95c76ef9c45eaec08

Decode

Text message	
To:	130[redacted]
Message:	1.5V:Modelsdk;os2.2;Languageen;NET 3G

Additional information	
PDU type:	SMS-SUBMIT
Reference:	0
Val. format:	None
Data coding:	SMS Default Alphabet

Original Encoded PDU fields	
SMSC:	00
PDU header:	01
TP-MTI:	01
TP-RD:	00
TP-VPF:	00
TP-SRR:	00
TP-UDHI:	00
TP-RP:	00

Hashdays Android Challenge

Enjoy our Android Challenge and win a FortiGate with AV/IPS/spam filtering updates for 12 months

How to win

https://www.hashdays.ch/downloads/hashdays-challenge.apk

sha256:

8acfac2d1646b7689e09aab629a58ba66029b295068ca76cdaccbdc92b4e5ea9

Be the 1st to bring back secret code to Fortinet's booth

Provide a write-up in the next few days

Thank You !

FortiGuard Labs

Follow us on twitter: **@FortiGuardLabs**
or on our blog http://blog.fortiguard.com

Me

twitter: @cryptax
e-mail: aapvrille at fortinet dot com
https://github.com/cryptax/dextools

Slides edited with LOBSTER=LaTeX+*Beamer* + *Editor*