How Android Malware Fights (and we fight back!)

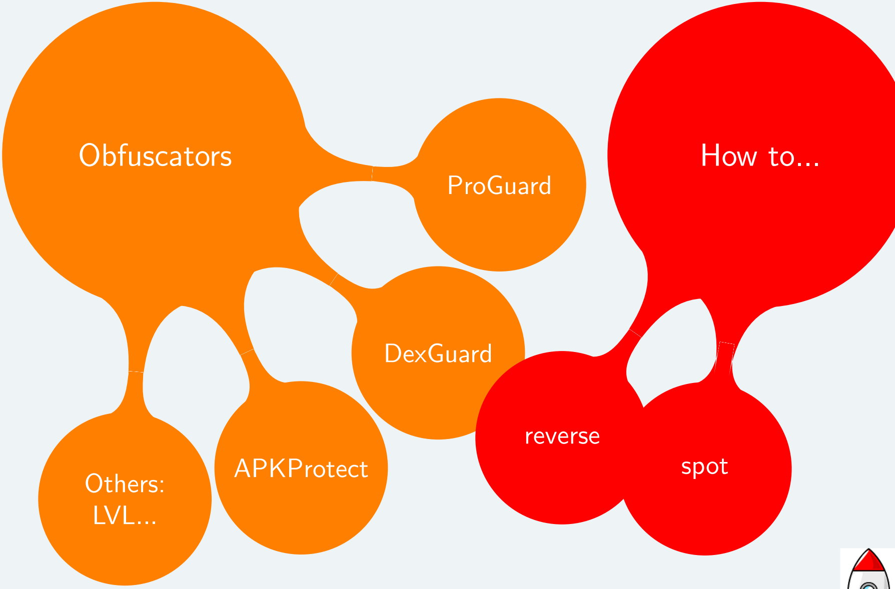
or Research on obfuscation in Android malware

Axelle Apvrille, Ruchna Nigam
Fortiguard Labs, Fortinet

CARO Workshop, Melbourne, Florida, USA
May 2014

Slides have been edited for public release.

Obfuscators


Obfuscators

ProGuard

DexGuard

APKProtect

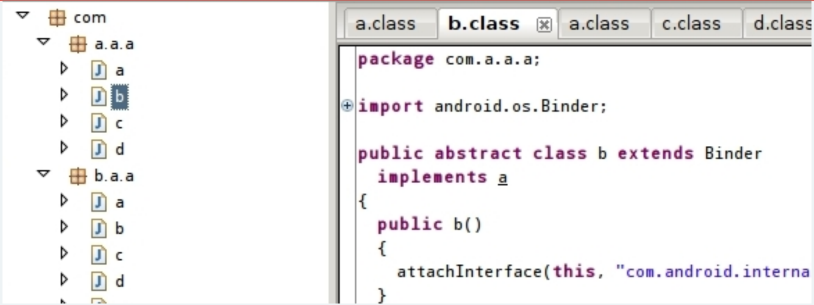
Others:
LVL...

How to...

reverse

spot

ProGuard



```
package com.a.a.a;

import android.os.Binder;

public abstract class b extends Binder
  implements a
{
  public b()
  {
    attachInterface(this, "com.android.interna
  }
```

- Spot the a's: **a/a;->a** ... (Example: Android/Pincer.A!tr.spy above)
- Use a **custom dictionary** `-obfuscationdictionary`, `-classobfuscationdictionary`, `-packageobfuscationdictionary`. Possibly generate with http://www.random.org/strings (e.g GinMaster.L).
- Approx 33,000/230,000 analyzed = 17 malicious samples using Proguard

License Verification Library (LVL)

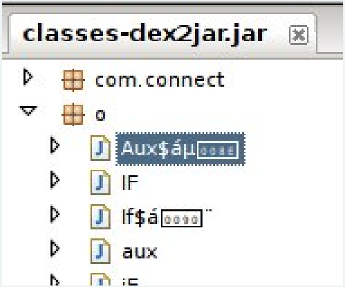
```
package com.android.vending.licensing;
...
public class AESObfuscator implements Obfuscator {
...
    private static final String CIPHER_ALGORITHM =
                "AES/CBC/PKCS5Padding";
    private static final byte[] IV = { 16, 74, 71, -80...
    private static final String header =
                "com.android.vending.licensing.AESObfuscator-1|"
```

- header prefix + string to obfuscate → AES → Base64
- used in Android/Plankton.B!tr

Tip: How to Spot

```
$ find . -type f -name *.smali -print |
   xargs grep "AESObfuscator-1"
```

DexGuard-ed samples are painful to reverse



Hard time for AV analysts

- UTF16 →
  - bad display with jd-gui,
  - no completion with Androguard...
- Example: used by Android/Dendroid.A!tr (March 2014)

Tip: How to spot

```
$ find . -type f -name "*.smali" -print |
   perl -ne 'print if /[$^$ [:ascii:]]/'
```

Tips to reverse DexGuard

Python decryption script template

- Adapt to each case
- Written by Nicolas Fallière
- Does not work with recent versions of DexGuard

Is it acceptable to modify the DEX?

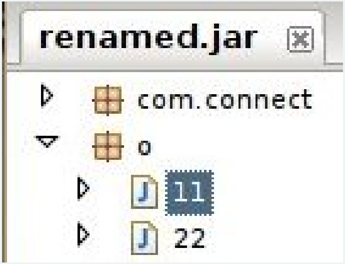
Insert logs in smali, then re-build

```
invoke-static {v1, v2}, Landroid/util/Log;->e(
Ljava/lang/String;Ljava/lang/String;)I
```

Handy: DEX Strings renaming

How does it work?

1. Parse `string_id_item[]`
2. Rename non printable strings, keep same size



renamed.jar

- com.connect
- o
  - 11
  - 22

Issues

- Make sure no duplicate strings
- Breaks string ordering - but we don't care

Download

https://github.com/cryptax/dextools/tree/master/hidex

```
$ ./hidex.pl --input classes.dex --rename-strings
```

Reversing Android/SmsSend.ND!tr

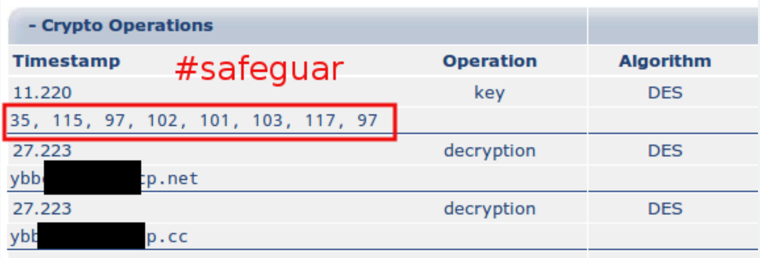
SmsSend.ND!tr - March 2014

- First **APKProtected** malware?
- Spot string `"APKProtect"`

Make your own custom decryption routine

```
$ java SmsDecrypt
Processing string: ==aFgIDU0oPWgoK...
d64xor: 96500db3f2242a4b2ac920e4...
Decrypting: ybbc[CENSORED]icp.cc
```

Andrubis does it :)

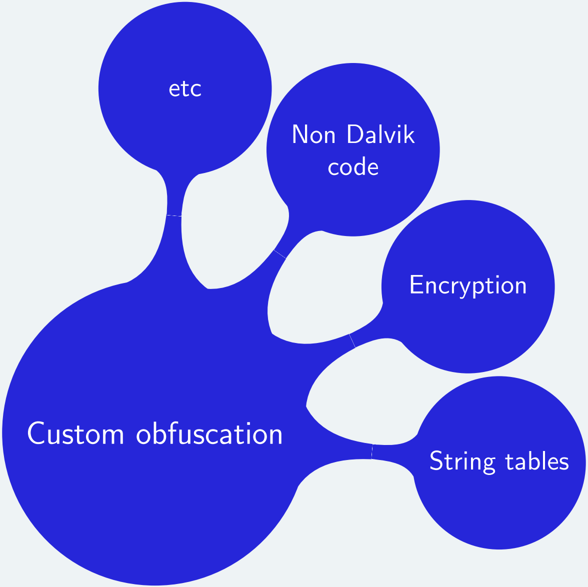


- Crypto Operations

Timestamp	Operation	Algorithm
11.220	key	DES
35, 115, 97, 102, 101, 103, 117, 97		
27.223	decryption	DES
ybb[redacted]p.net		
27.223	decryption	DES
yb[redacted]p.cc		

#safeguar

Custom obfuscation



Android/GinMaster.L (Dec 2013) and string tables

Builds its own string table:

```
package Eg9Vk5Jan;
 class x18nAzukp {
    final private static char[][] OGqHAYq8N6Y6tswt8g;
    static x18nAzukp()
    {
        v0 = new char[][48];
        v1 = new char[49];
        v1 = {97, 0, 110, 0, 100, 0, 114, 0, 111, 0, 105,
        0, 100, 0, 46, 0, 97, 0, 112, 0, 112, 0, 46, 0, 67, 0,
        ...
        v0[0] = v1;
        v2 = new char[56];
        v2 = {97, 0, 110, 0, 100, 0, 114, 0, 111, 0, 105,
        0, 100, 0, 46, 0, 97, 0, 112, 0, 112, 0, 46, 0, 65, 0,
      ...
```

String tables: an attempt to hide strings in code

Using the string table

```
protected static String rLGAEh9JeCgGn73A(int p2) {
return new String(
  Eg9Vk5Jan.x18nAzukp.0GqHAYq8N6Y6tswt8g[p2]);
}
...
new StringBuilder(x18nAzukp.rLGAEh9JeCgGn73A(43))...
```

At first, the analyst only sees a reference (e.g 43)

Procyon sees it better

```
class x18nAzukp {
    private static final char[][] 0GqHAYq8N6Y6tswt8g;
    static {
        0GqHAYq8N6Y6tswt8g = new char[][] { { 'a', 'n', 'd', 'r', 'o',
'a', 'p', 'p', '.', 'C', 'o', 'n', 't', 'e', 'x', 't',...
```

or use Python snippet like `"".join(map(chr, bytes))`

Encryption in malware

See Cryptography for mobile malware obfuscation, RSA 2011

Example

Android/SmsSpy.HW!tr (Feb 2014): Blowfish encrypted asset is XML configuration file

Stats

27 of malware use encryption - stats collected from 460,493 malicious samples

NB. sometimes encryption is used in legitimate portions

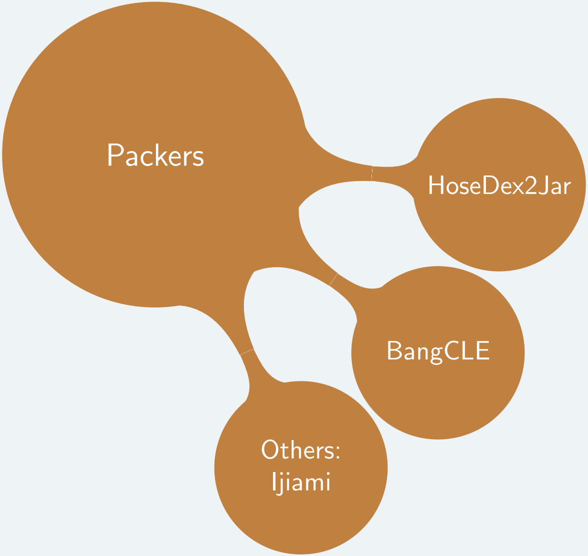
Loading non Dalvik code

- **ELF**: Android/DroidKungFu.B, C, E and G process commands in native code.
  Android/DroidCoupon hides Rage Against The Cage exploit in a file with PNG extension

```
ratc.png: ELF 32-bit LSB executable, ARM, version 1 (SYSV),
dynamically linked (uses shared libs), stripped
rbb.png:  gzip compressed data, was "busybox", from Unix, . . .
```

- **Basic4PPC** in WinCE/Redoc. No malware using Basic4Android yet?
- **Javascript** for click fraud in Android/FakePlay.B!tr

Packers



Packers

HoseDex2Jar

BangCLE

Others:
Ijiami

HoseDex2Jar

Tip: How To Spot - hidex

```
$ ~/dev/hideandseek/hidex/hidex.pl --input classes.dex-hosed
WARNING: strange header size: 136080
DEX Header of file:
Magic      : 6465780a30333500
```

De-hose

https://github.com/strazzere/dehoser ;)

BangCLE packer - *Ruchna Nigam*

- Online packing service www.bangcle.com

BangCLE packer - *Ruchna Nigam*

- Online packing service www.bangcle.com
- Encountered in Android/Feejar.B (2014)

BangCLE packer - *Ruchna Nigam*

- Online packing service www.bangcle.com
- Encountered in Android/Feejar.B (2014)
- Real application decrypted at runtime only

BangCLE packer - *Ruchna Nigam*

- Online packing service www.bangcle.com
- Encountered in Android/Feejar.B (2014)
- Real application decrypted at runtime only
- (Most of) packing job done by native libraries - obfuscated too

BangCLE packer - *Ruchna Nigam*

- Online packing service www.bangcle.com
- Encountered in Android/Feejar.B (2014)
- Real application decrypted at runtime only
- (Most of) packing job done by native libraries - obfuscated too

BangCLE packer - *Ruchna Nigam*

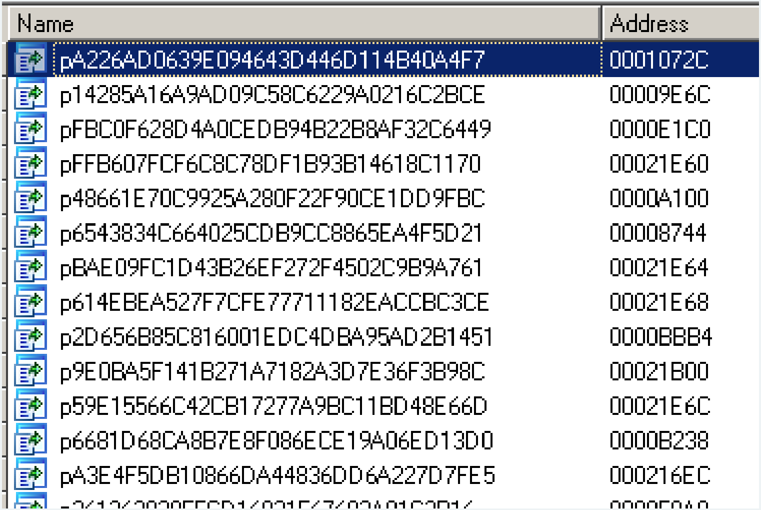
- Online packing service `www.bangcle.com`
- Encountered in Android/Feejar.B (2014)
- Real application decrypted at runtime only
- (Most of) packing job done by native libraries - obfuscated too

How to detect?

- Presence of libsecmain.so, libsecexe.so, bangcle_classes.jar
- `com.secapk.wrapper.ApplicationWrapper`
- Classes named FirstApplication, MyClassLoader, ACall...

Hands on BangCLE

Mangled export names



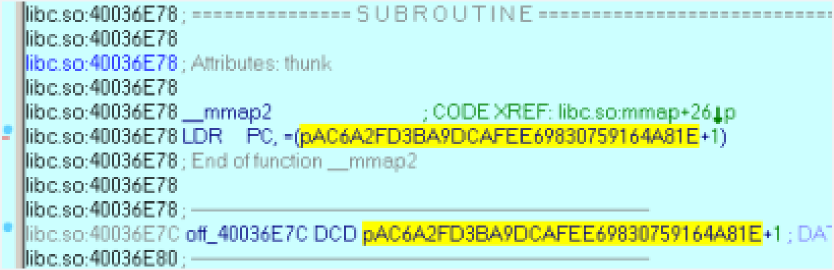
Name	Address
pA226AD0639E094643D446D114B40A4F7	0001072C
p14285A16A9AD09C58C6229A0216C2BCE	00009E6C
pFBC0F628D4A0CEDB94B22B8AF32C6449	0000E1C0
pFFB607FCF6C8C78DF1B93B14618C1170	00021E60
p48661E70C9925A280F22F90CE1DD9FBC	0000A100
p6543834C664025CDB9CC8865EA4F5D21	00008744
pBAE09FC1D43B26EF272F4502C9B9A761	00021E64
p614EBEA527F7CFE77711182EACCBC3CE	00021E68
p2D656B85C816001EDC4DBA95AD2B1451	0000BBB4
p9E0BA5F141B271A7182A3D7E36F3B98C	00021B00
p59E15566C42CB17277A9BC11BD48E66D	00021E6C
p6681D68CA8B7E8F086ECE19A06ED13D0	0000B238
pA3E4F5DB10866DA44836DD6A227D7FE5	000216EC
[illegible]	[illegible]

Mmap is hooked

Hands on BangCLE

Mangled export names

Mmap is hooked



```
libc.so:40036E78 ; ================ S U B R O U T I N E ===============================
libc.so:40036E78
libc.so:40036E78 ; Attributes: thunk
libc.so:40036E78
libc.so:40036E78 __mmap2                          ; CODE XREF: libc.so:mmap+26↓p
libc.so:40036E78 LDR     PC, =(pAC6A2FD3BA9DCAFEE6983075916 4A81E+1)
libc.so:40036E78 ; End of function __mmap2
libc.so:40036E78
libc.so:40036E78 ; ---------------------------------------------------------------
libc.so:40036E7C off_40036E7C DCD pAC6A2FD3BA9DCAFEE69830759164A81E+1 ; DAT
libc.so:40036E80 ; ---------------------------------------------------------------
```

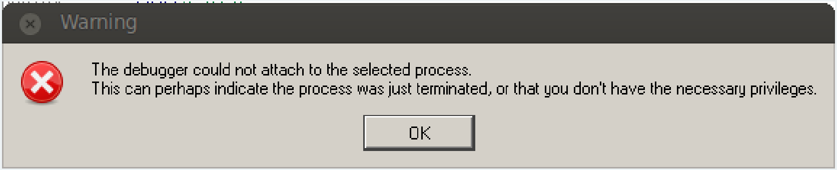
Anti-debugging?

Hands on BangCLE

Mangled export names

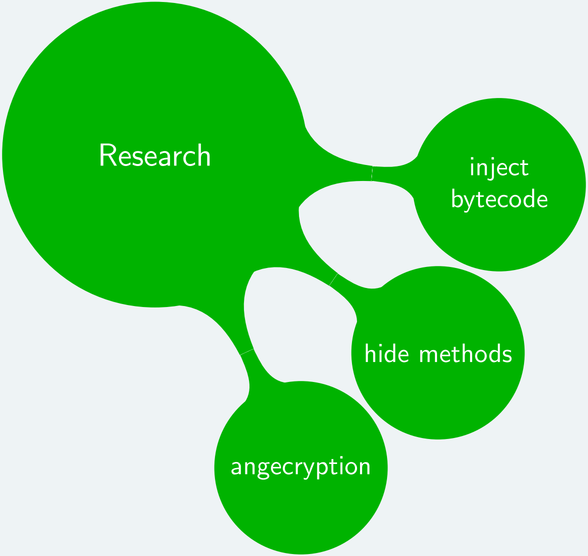
Mmap is hooked

Anti-debugging?



The debugger could not attach to the selected process.
This can perhaps indicate the process was just terminated, or that you don't have the necessary privileges.

Obfuscation in the future: what could it be like?



Inject arbitrary Dalvik bytecode

How does it work?

- Jurrian Bremer - "Abusing Dalvik Beyond Recognition"
- Injecting bytecode:
  1. Dalvik bytecode represented as UTF16 string
  2. Instantiate a class object (class with virtual method)
  3. **iput-quick** (0xf5): Overwrite address code of virtual function with address of string
  4. **invoke-virtual**: call the method

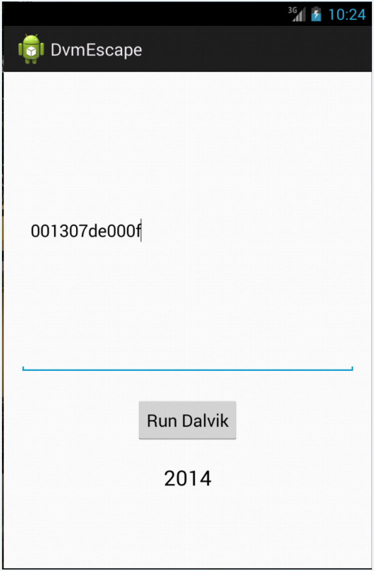
Example: injecting 0013 07de 000f

Dalvik bytecode:

```
const/16 v0, #7de
return v0
```

- 0x13: const/16
- 0x07de = 2014
- 0x0f: return

Injecting bytecode PoC - DvmEscape - Bremer



001307de000f

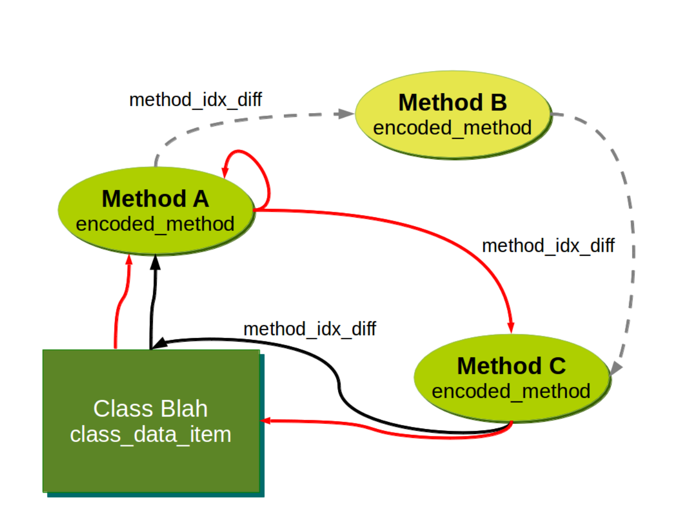
2014

Status

PoC return integers only

Not seen in malware (yet?).

Hidex: hiding methods to dissassemblers



Detect with `hidex.pl`

AngeCryption to hide a APK

- **Attack**: decrypt a PNG and it becomes an APK
- PoC tool at http://corkami.googlecode.com/svn/trunk/src/angecryption/angecrypt.py

It works!!! (after a few hacks)

```
$ python angecrypt.py test.apk pic.png modified.apk
  ’key....’ aes ...
```

- Duplicate EOCD After all the central directory entries comes the end of central directory (EOCD) record, which marks the end of the .ZIP file
- Pad to 16 bytes

Alert!

Keep an eye on it in the future!

Thank You !

FortiGuard Labs

Follow us on twitter: **@FortiGuardLabs**
or on our blog http://blog.fortinet.com
Me: **@cryptax** or `aapvrille` at `fortinet dot com`
Ruchna: **@_r04ch_** or `rnigam` at `fortinet dot com`
Hidex:
`https://github.com/cryptax/dextools/tree/master/hidex`

Many thanks to: Ange Albertini, Jurriaan Bremer, Anthony Desnos.

Are those PowerPoint slides? No way! It's LaTeX+ TikZ + Beamer + Lobster