ANDROID COMPILER FINGERPRINTING

CALEB FENTON - TIM "DIFF" STRAZZERE

07.22.2016

HITCON COMMUNITY 2016

REDNAGA

WHO ARE WE

RED NAGA?

- Banded together by the love of 0days and hot sauces
- Random out of work collaboration and pursuit of up-leveling the community
  - Disclosures / Code / Lessons available on GitHub
- rednaga.io
- github.com/RedNaga

WHO ARE WE

CALEB

- Researcher @ SentinelOne
- Former Researcher @ SourceClear
  Former Researcher @ Lookout
- Texan at heart, Californian based on shorts and sandals 24/7
- Creator of “Simplify”
- @CalebFenton
- github.com/CalebFenton

WHO ARE WE

DIFF

- Researcher @ SentinelOne
- Former Researcher @ Lookout
- Obfuscation and Packer Junkie
- Makes own hot sauce - cause why not?
- @timstrazz
- github.com/strazzere

WHY ARE WE HERE

More importantly - why should you care?

- Threat Intel is important!
- Used for many purposes:
  - What are people researching now?
  - What should you research next?
  - Anticipate attack patterns
  - Avoid overlap with others!
- We like drinking…

THE TAKE AWAYS

What should you learn from us today?

- **How to fingerprint compilers (generically)**
- Abnormalities in DEX structure or values
- Signals modification / tampering
- Compiler fingerprinting
- Sophisticated agents
- Related PC stuff
  - F.L.I.R.T. - https://www.hex-rays.com/products/ida/tech/flirt/index.shtml
  - PEID - http://www.aldeid.com/wiki/PEiD

CURRENT ANDROID TOOL LANDSCAPE

Tools and Evolution

REDNAGA

ANDROID APPLICATION PACKAGING (APK)

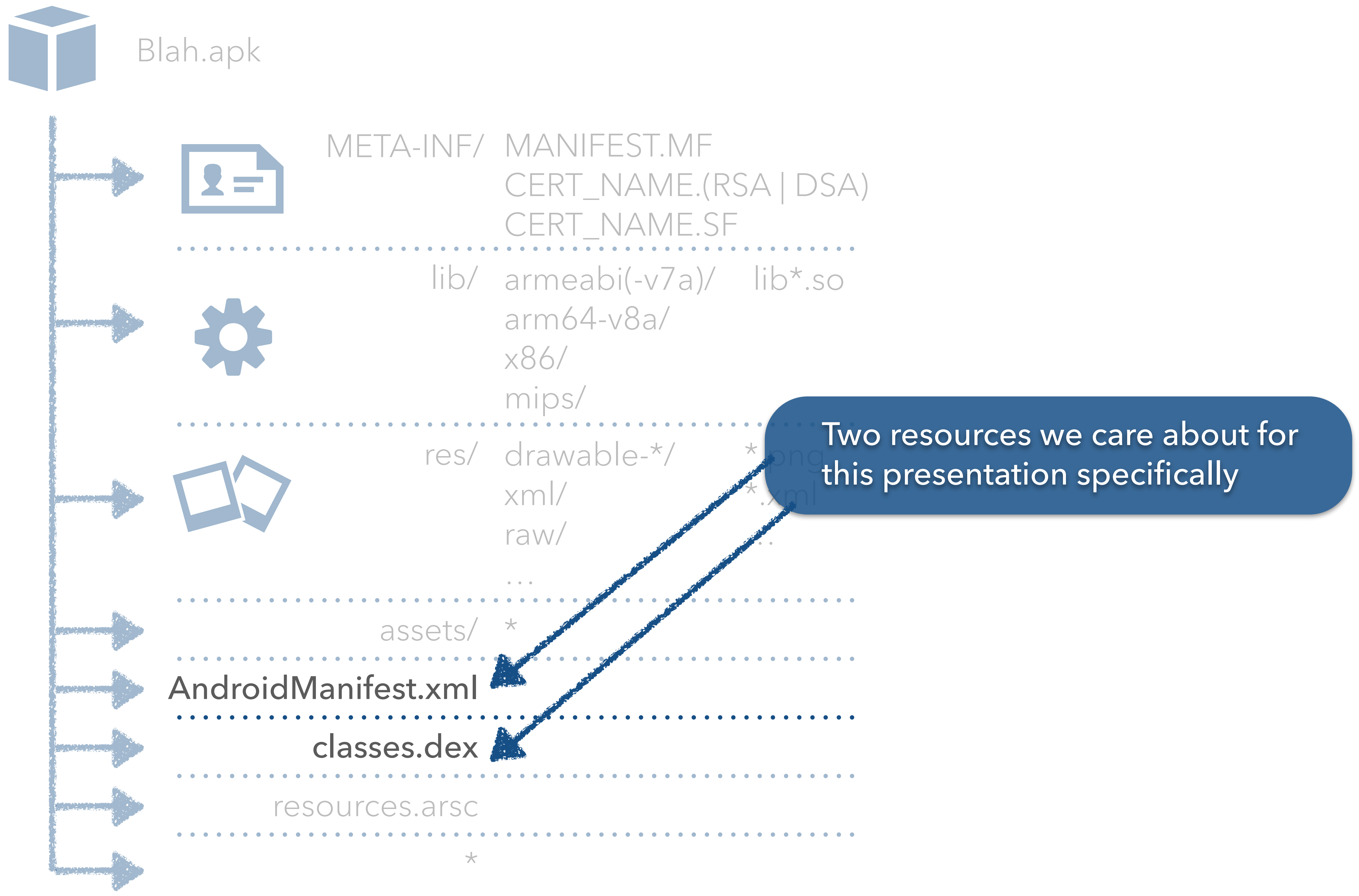
application/vnd.android.package-archive

Blah.apk

META-INF/	MANIFEST.MF	
	CERT_NAME.(RSA | DSA)	
	CERT_NAME.SF	
lib/	armeabi(-v7a)/	lib*.so
	arm64-v8a/	
	x86/	
	mips/	
res/	drawable-*/	*.png
	xml/	*.xml
	raw/	…
	…	
assets/	*	
AndroidManifest.xml		
classes.dex		
resources.arsc		
*		

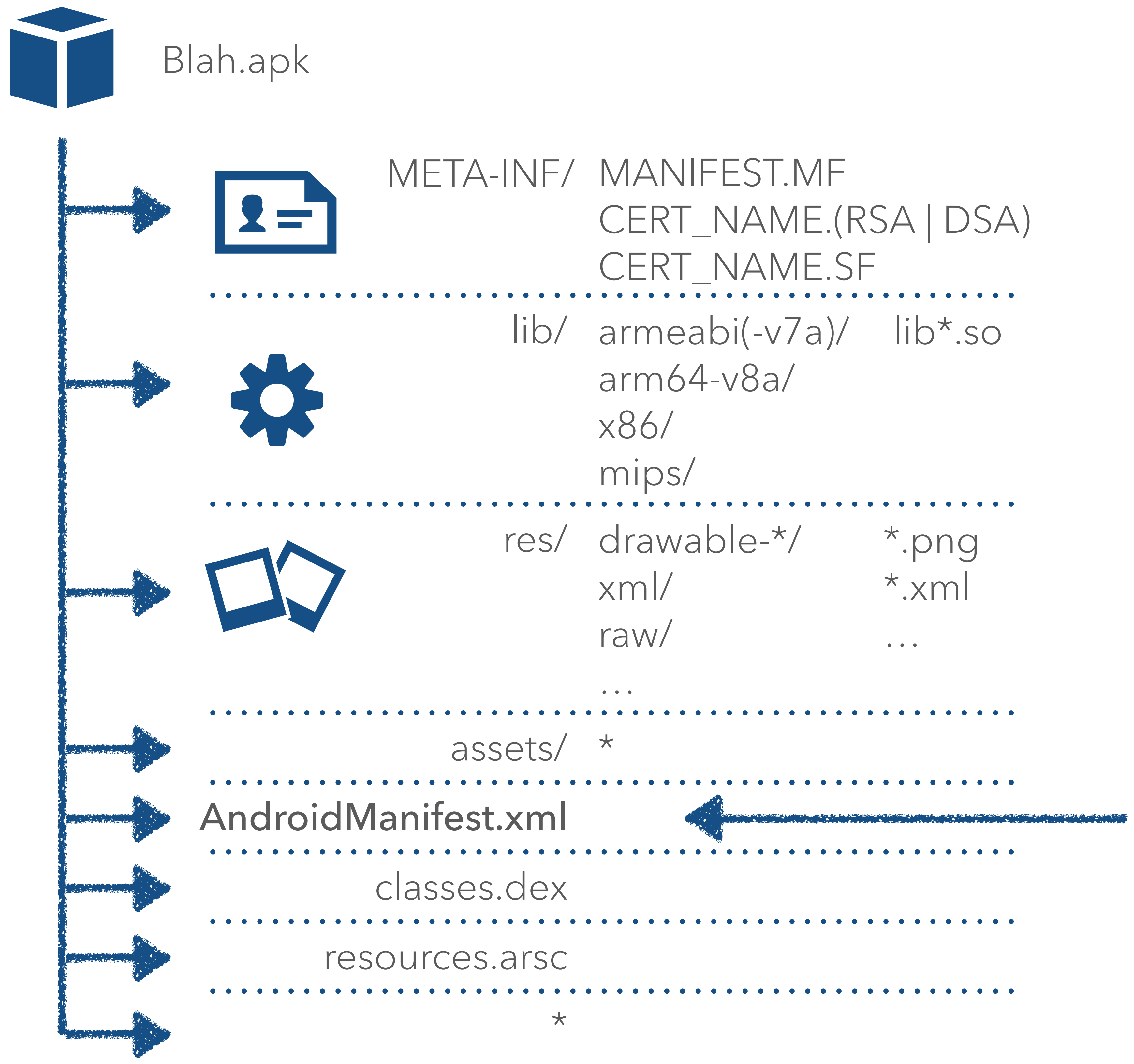
ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



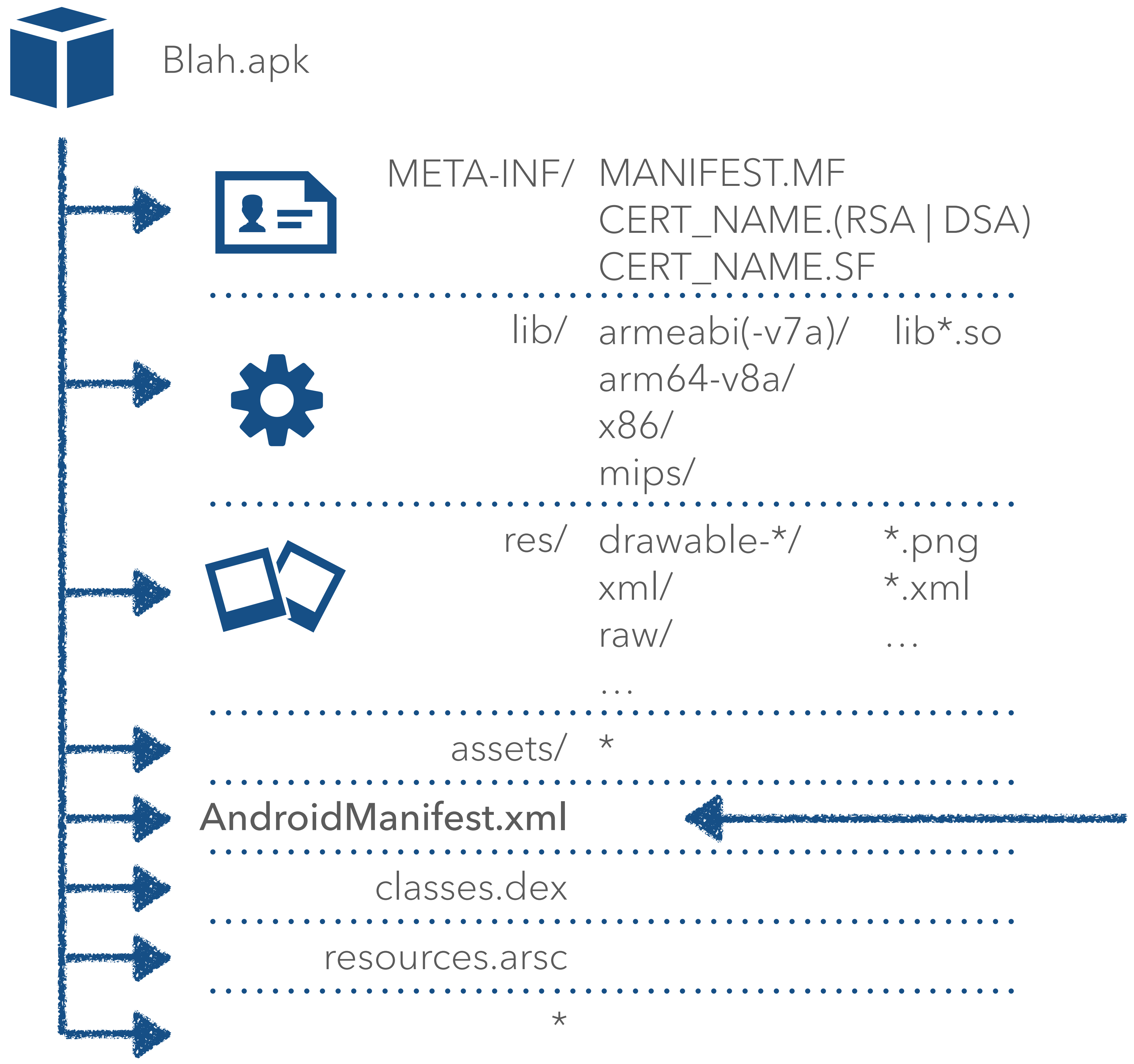
Android Manifest
Compiled AndroidXML

Contains:
- **entry points for app**
  - Activities
  - Services
  - Receivers
  - Intents
  - …
- **app permissions**
- **app meta-data**
  - **package name**
  - **version code/name**
  - **debuggable**
- **referenced libraries**

Reverse with:
- axmlprinter2
- apktool
- jeb / jeb2
- androguard
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



Blah.apk

META-INF/	MANIFEST.MF	
	CERT_NAME.(RSA | DSA)	
	CERT_NAME.SF	
lib/	armeabi(-v7a)/	lib*.so
	arm64-v8a/	
	x86/	
	mips/	
res/	drawable-*/	*.png
	xml/	*.xml
	raw/	…
	…	
assets/	*	
AndroidManifest.xml		
classes.dex		
resources.arsc		
*		

Android Manifest
Compiled AndroidXML

Created by:
- aapt
- axmlprinter2 (new ver)
- apktool
  - (axmlprinter2 mod)
- random Python scripts

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive

Blah.apk

META-INF/	MANIFEST.MF	
	CERT_NAME.(RSA | DSA)	
	CERT_NAME.SF	
lib/	armeabi(-v7a)/	lib*.so
	arm64-v8a/	
	x86/	
	mips/	
res/	drawable-*/	*.png
	xml/	*.xml
	raw/	…
	…	
assets/	*	
AndroidManifest.xml		
classes.dex		
resources.arsc		
*		

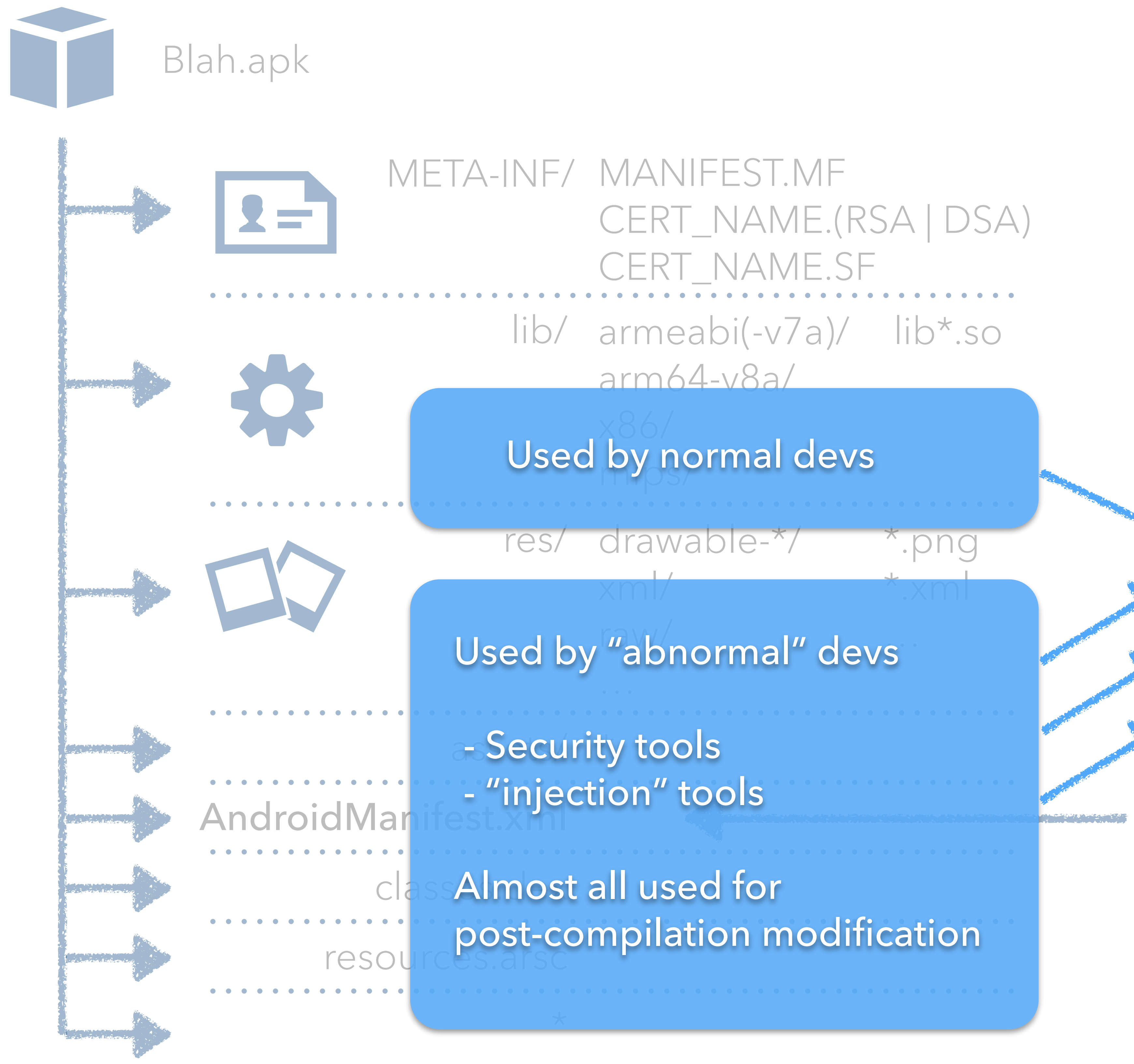
Used by normal devs

Android Manifest
Compiled AndroidXML

Created by:
- aapt
- axmlprinter2 (new ver)
- apktool
  (axmlprinter2 mod)
- random Python scripts

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



Used by normal devs

Used by "abnormal" devs

- Security tools
- "injection" tools

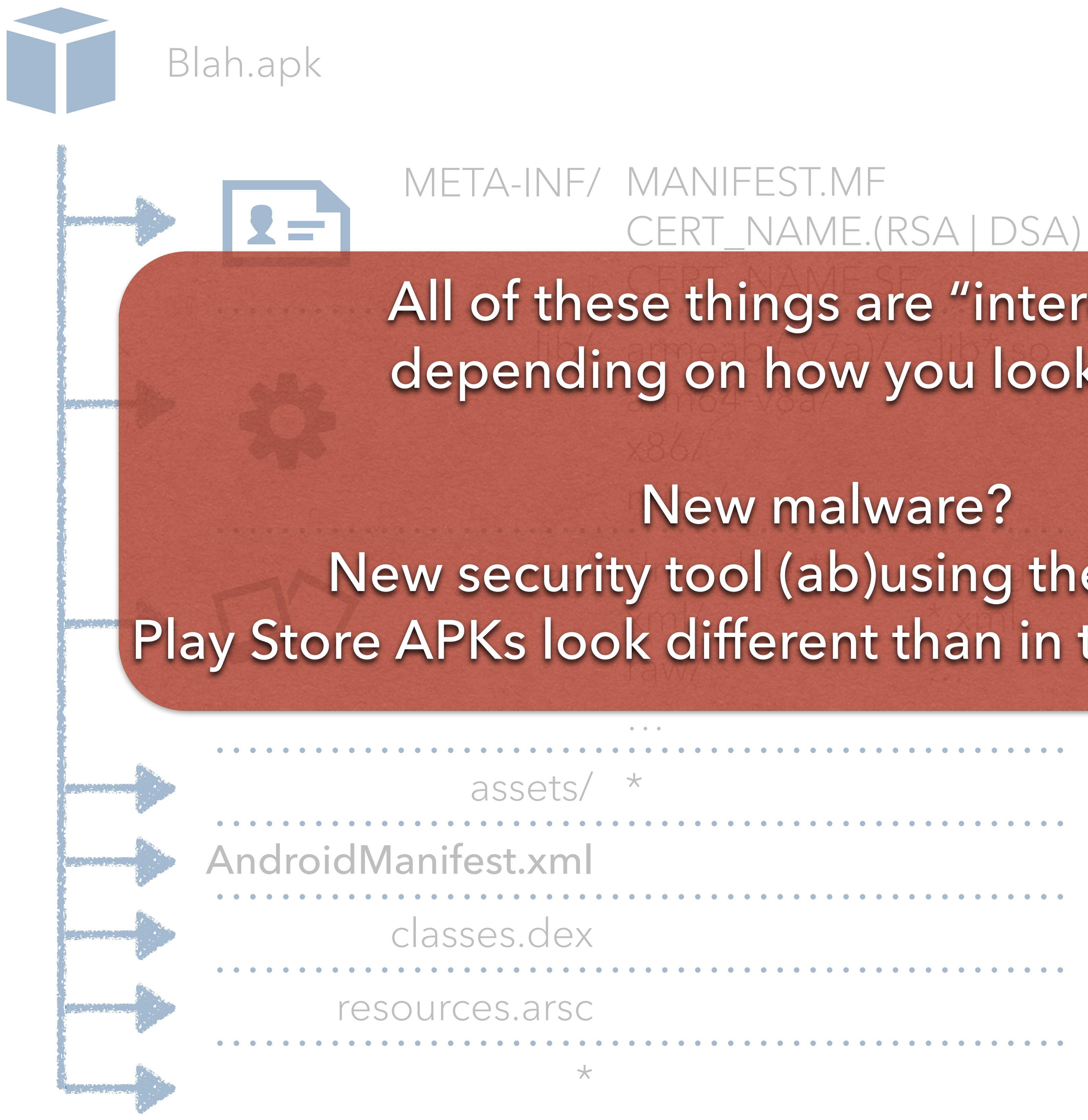
Almost all used for
post-compilation modification

Android Manifest
Compiled AndroidXML

Created by:
aapt
axmlprinter2 (new ver)
apktool
(axmlprinter2 mod)
random Python scripts

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



All of these things are "interesting" depending on how you look at it…

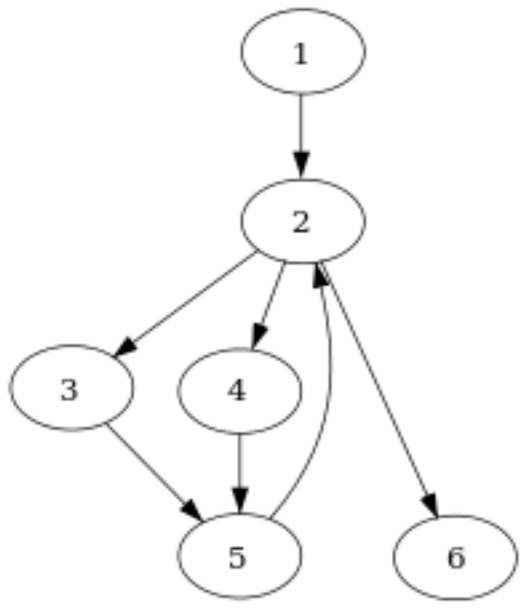
New malware?
New security tool (ab)using the system?
Play Store APKs look different than in the wild binaries?

AXML OPEN SOURCE CYCLE

Who is using what?

AXMLPrinter2 is a very, very old project with bugs…

- Was the standard which people found breakages in
- Code used by APKTool (licenses appear stripped)
- JEB imported APKTool (seen in licenses)
- JEB author back ported fixed into APKTool
- Library to break them all! (until jeb2)



AXML OPEN SOURCE CYCLE

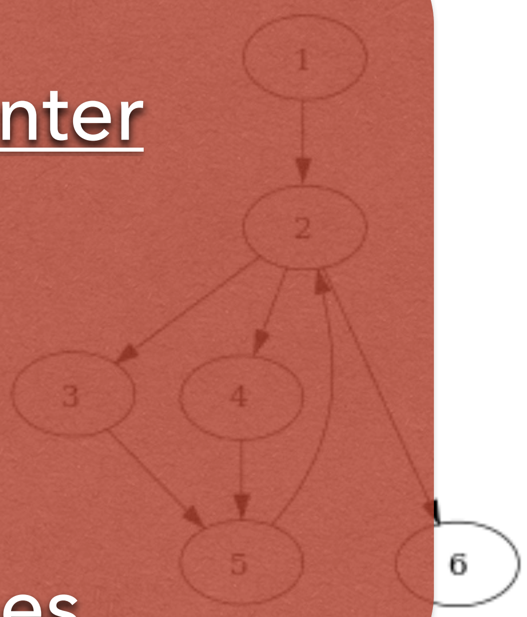
Who is using what?

AXMLPrinter2 is a very, very old project with bugs…

- Was the standard which people found breakages in
- Code used by APKTool (licenses appear stripped)
- JEB imported APKTool (seen in licenses)
- JEB author back ported fixed into APKTool
- Library to break them all! (until jeb2)

FOSS remake released;
https://github.com/rednaga/axmlprinter

~85% TCC
Allows reading / writing AXML
Avoids previous breakages
Can be used to detect these changes



ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive

Blah.apk

META-INF/	MANIFEST.MF	
	CERT_NAME.(RSA | DSA)	
	CERT_NAME.SF	
lib/	armeabi(-v7a)/	lib*.so
	arm64-v8a/	
	x86/	
	mips/	
res/	drawable-*/	*.png
	xml/	*.xml
	raw/	…
	…	
assets/	*	
AndroidManifest.xml		
classes.dex		
resources.arsc		
*		

Dalvik Executable
Compiled classes for DVM

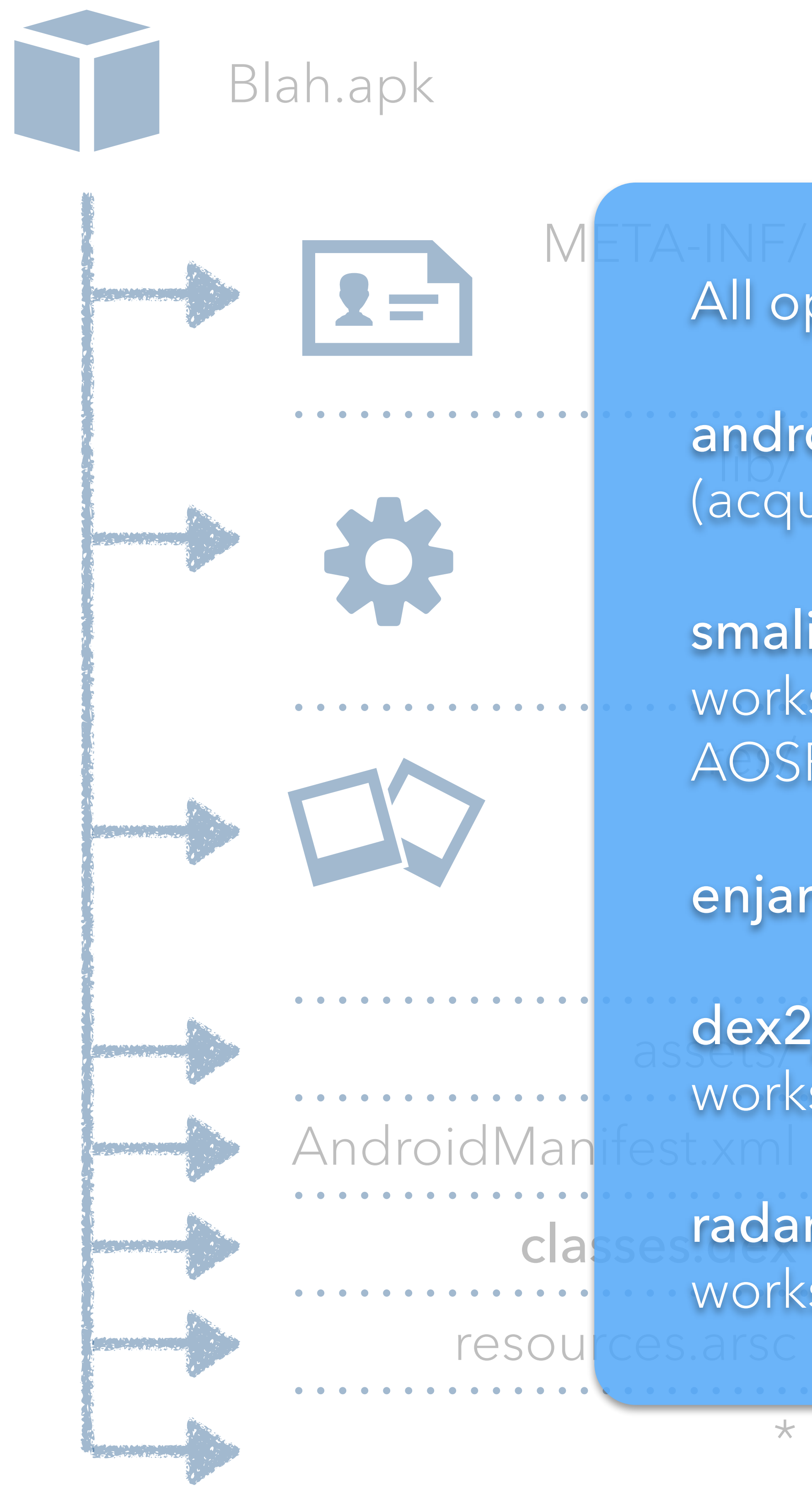
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



All open sourced tools

androguard used by VT (acquired by Google)

smali creator/maintainer now works at Google, used in AOSP

enjarify made by Google

dex2jar creator/maintainer works(ed?) at Trend

radare creator/maintainer works at NowSecure

Dalvik Executable
Compiled classes for DVM

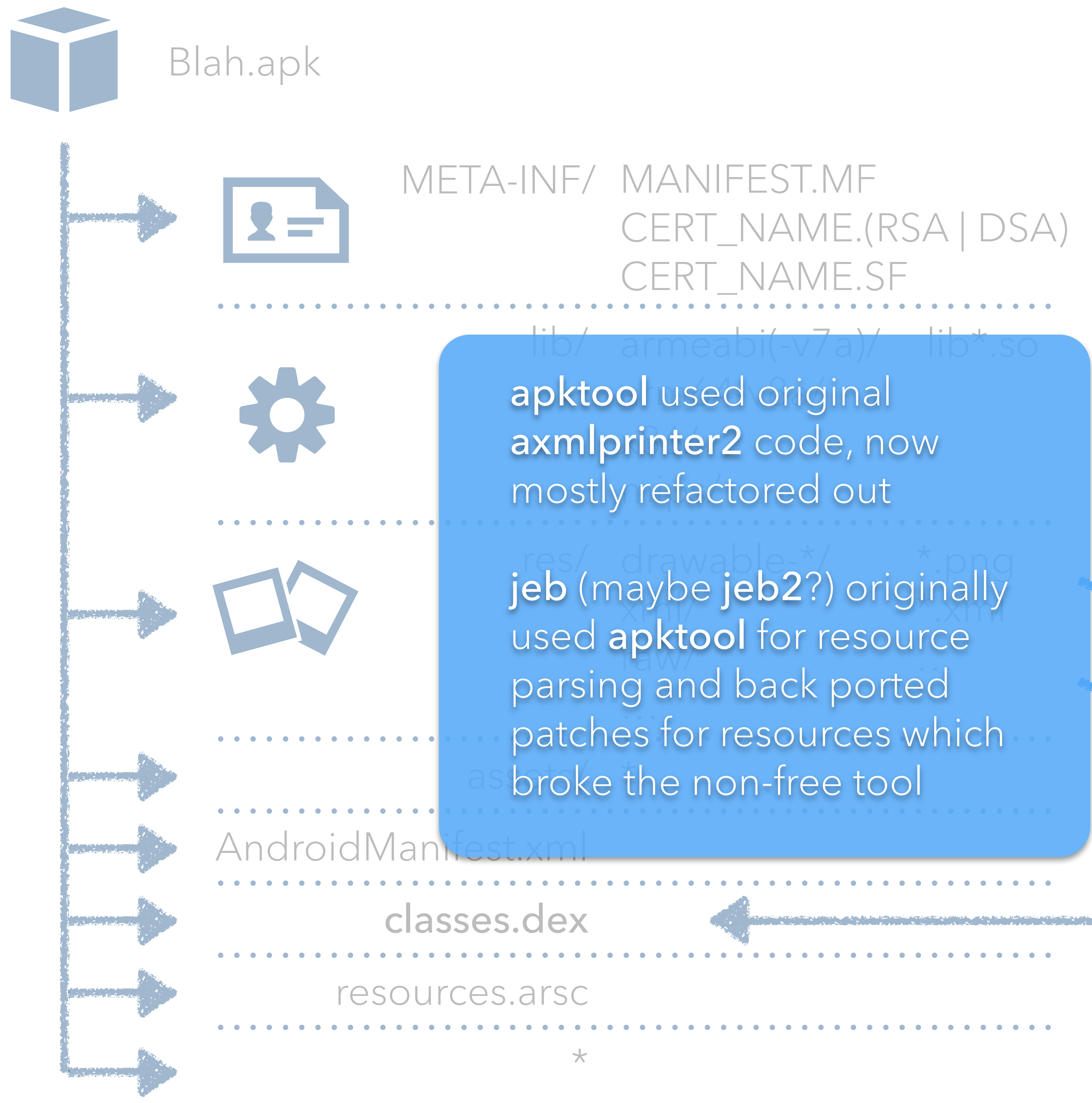
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



apktool used original **axmlprinter2** code, now mostly refactored out

jeb (maybe **jeb2**?) originally used **apktool** for resource parsing and back ported patches for resources which broke the non-free tool

Dalvik Executable
Compiled classes for DVM

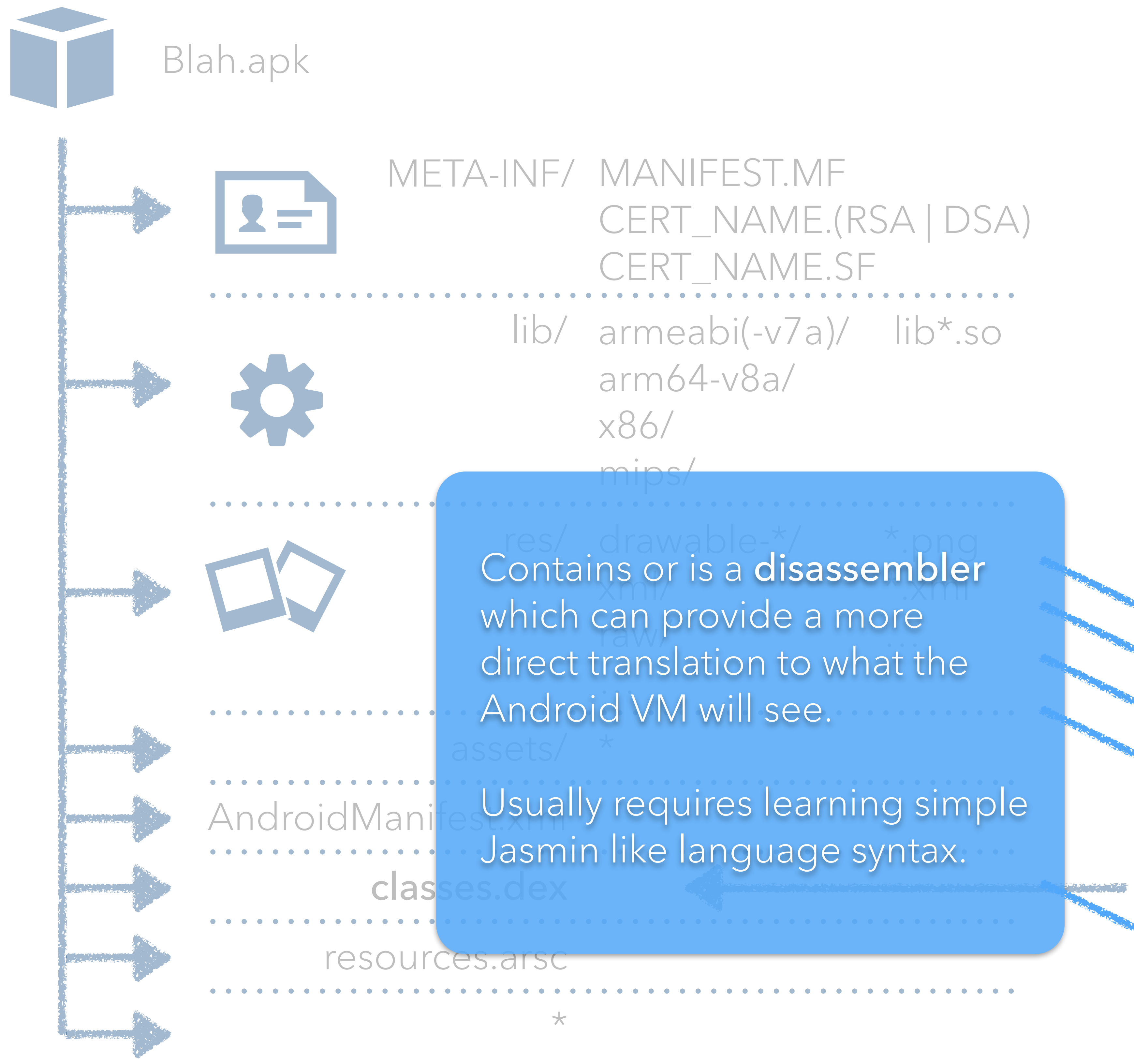
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



Contains or is a **disassembler** which can provide a more direct translation to what the Android VM will see.

Usually requires learning simple Jasmin like language syntax.

Dalvik Executable
Compiled classes for DVM

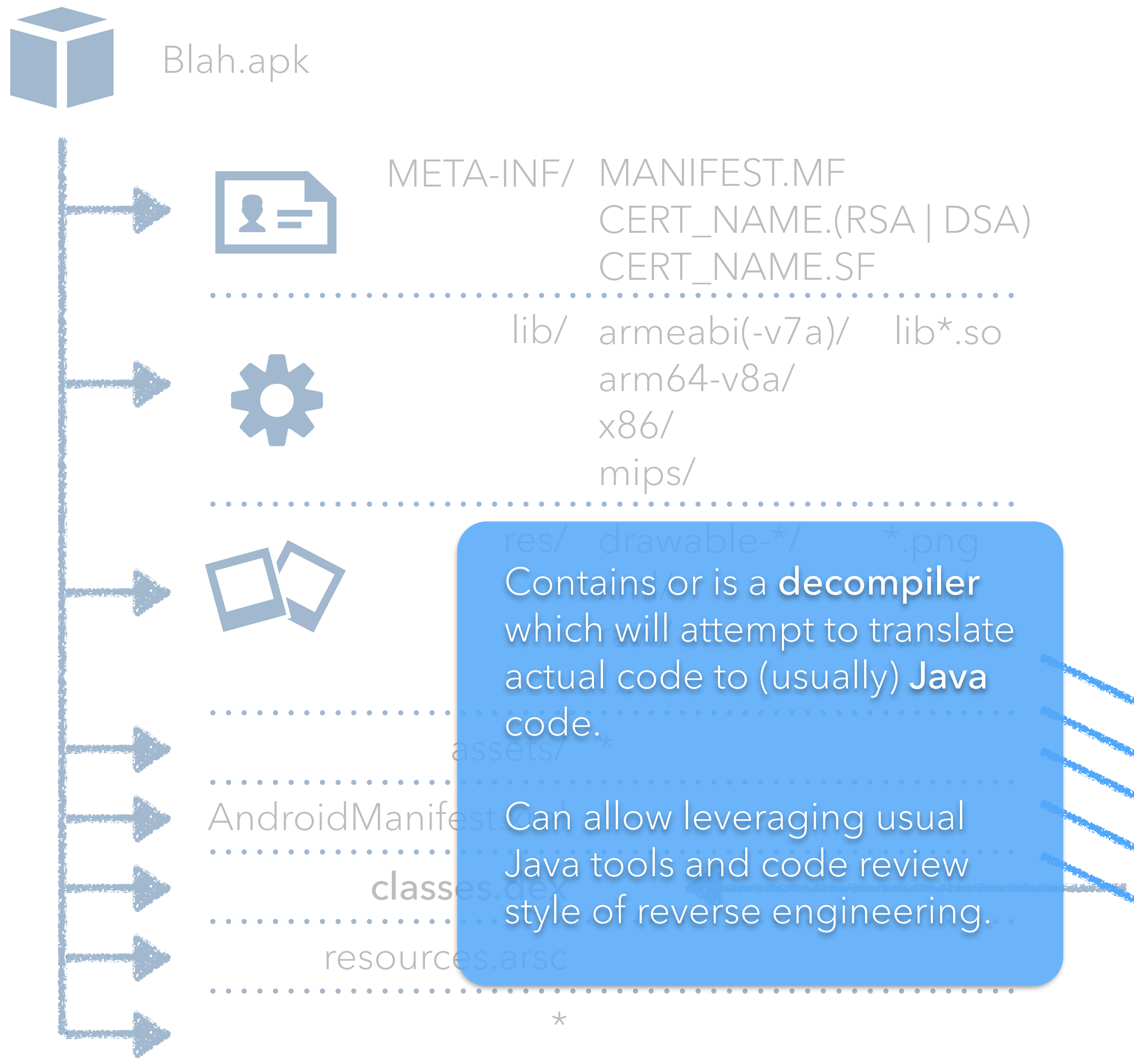
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



Contains or is a **decompiler** which will attempt to translate actual code to (usually) **Java** code.

Can allow leveraging usual Java tools and code review style of reverse engineering.

Dalvik Executable
Compiled classes for DVM

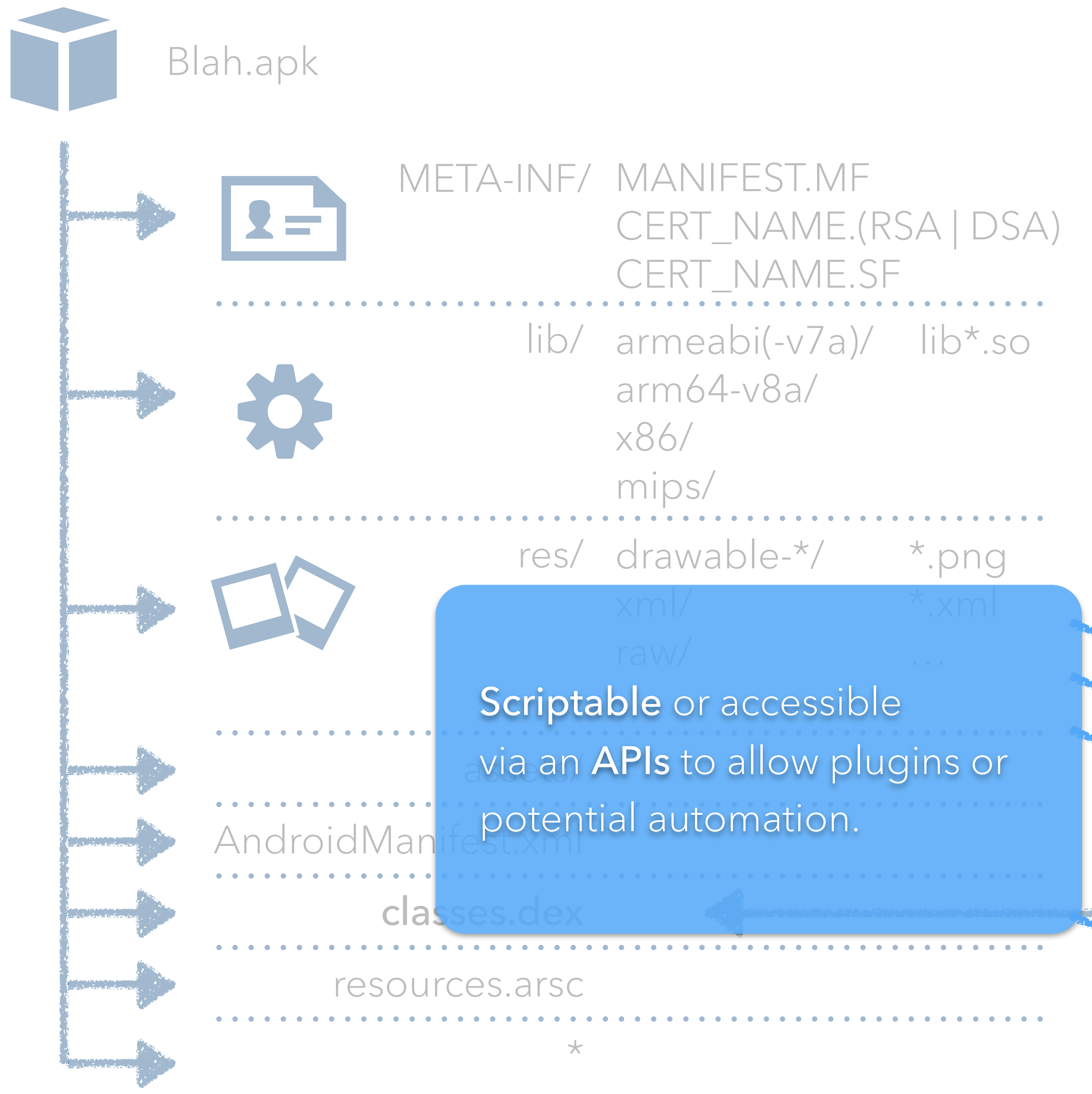
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



Scriptable or accessible via an **APIs** to allow plugins or potential automation.

Dalvik Executable
Compiled classes for DVM

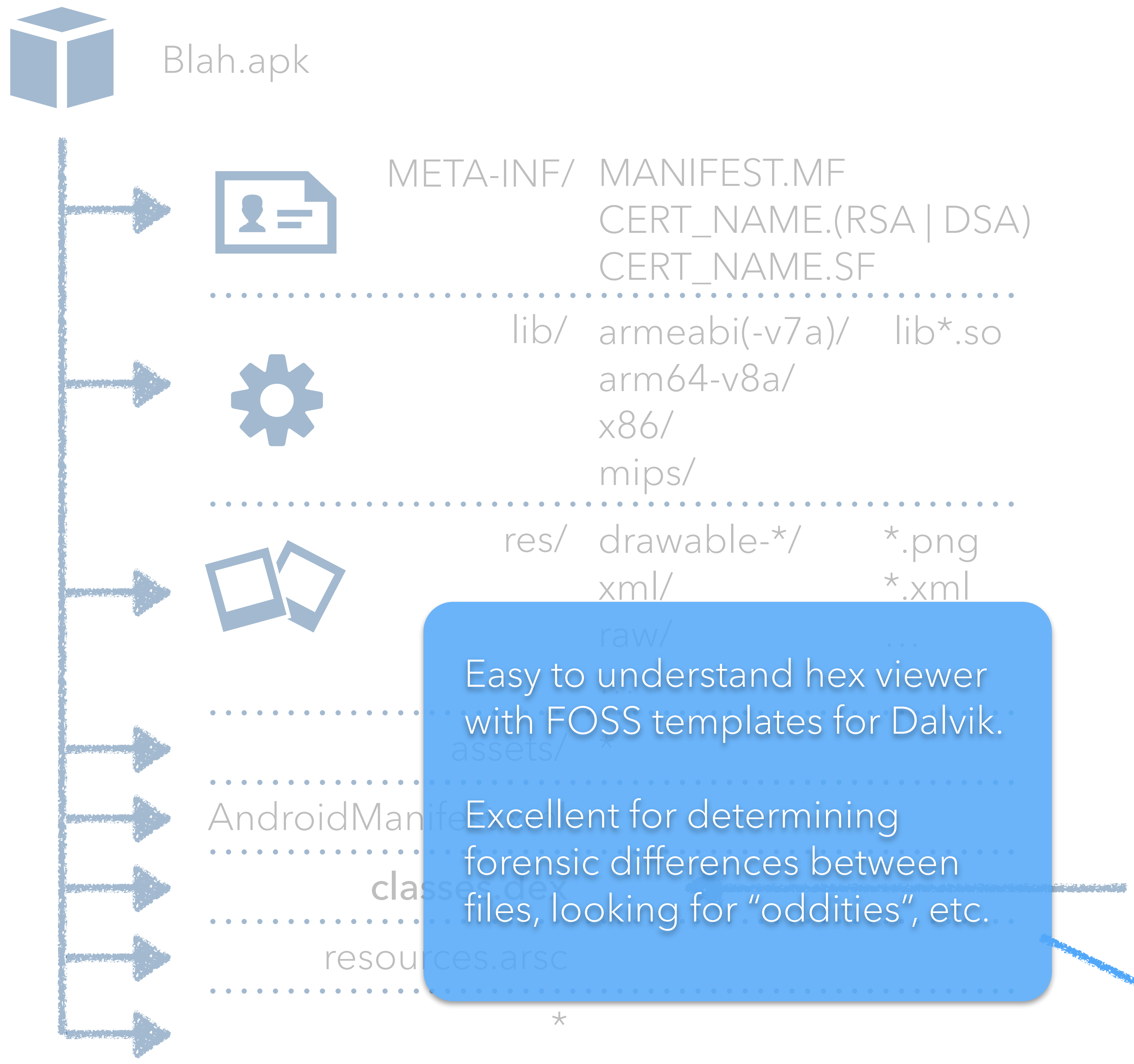
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



Easy to understand hex viewer with FOSS templates for Dalvik.

Excellent for determining forensic differences between files, looking for “oddities”, etc.

Dalvik Executable
Compiled classes for DVM

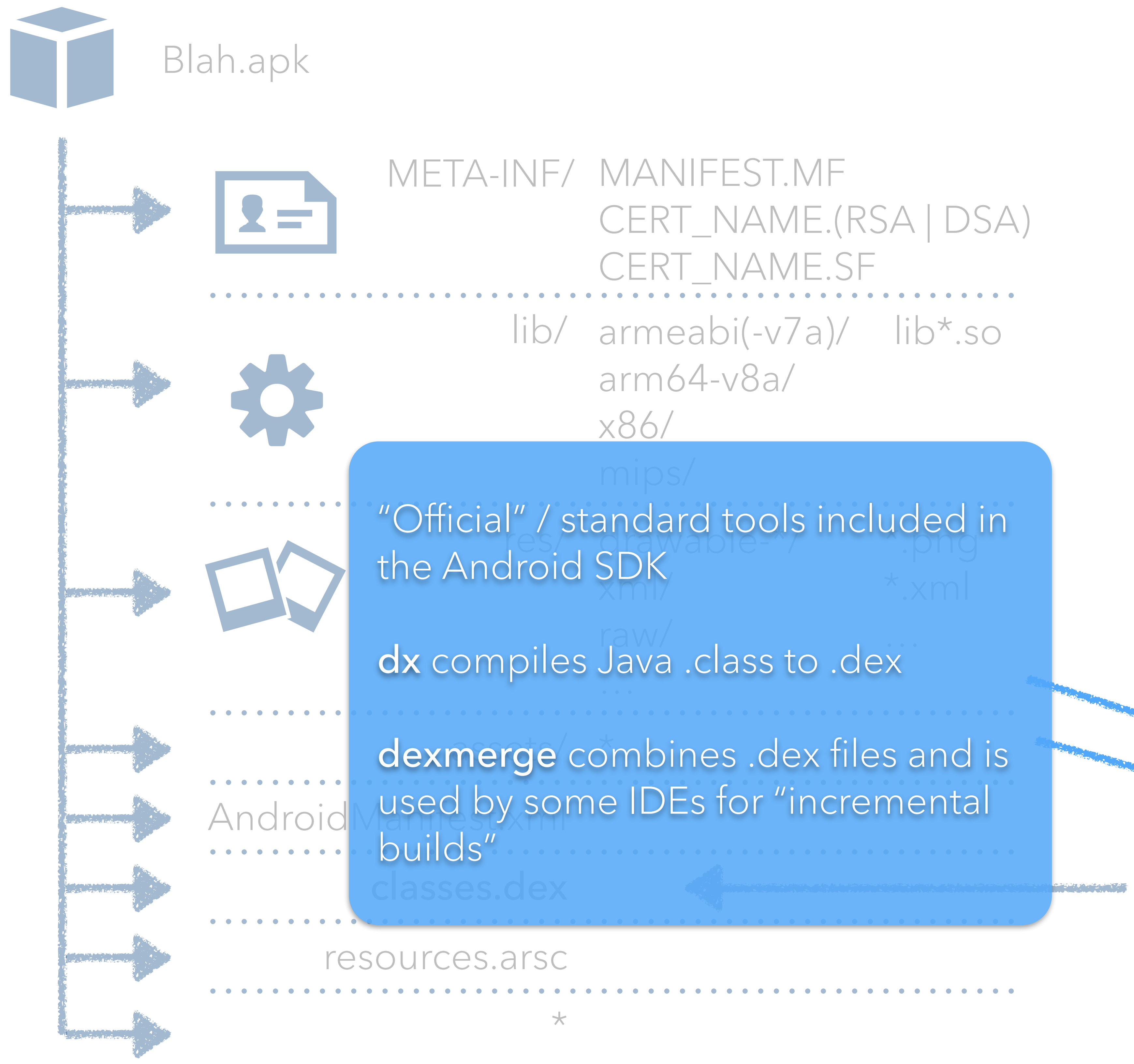
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



"Official" / standard tools included in the Android SDK

dx compiles Java .class to .dex

dexmerge combines .dex files and is used by some IDEs for "incremental builds"

Dalvik Executable
Compiled classes for DVM

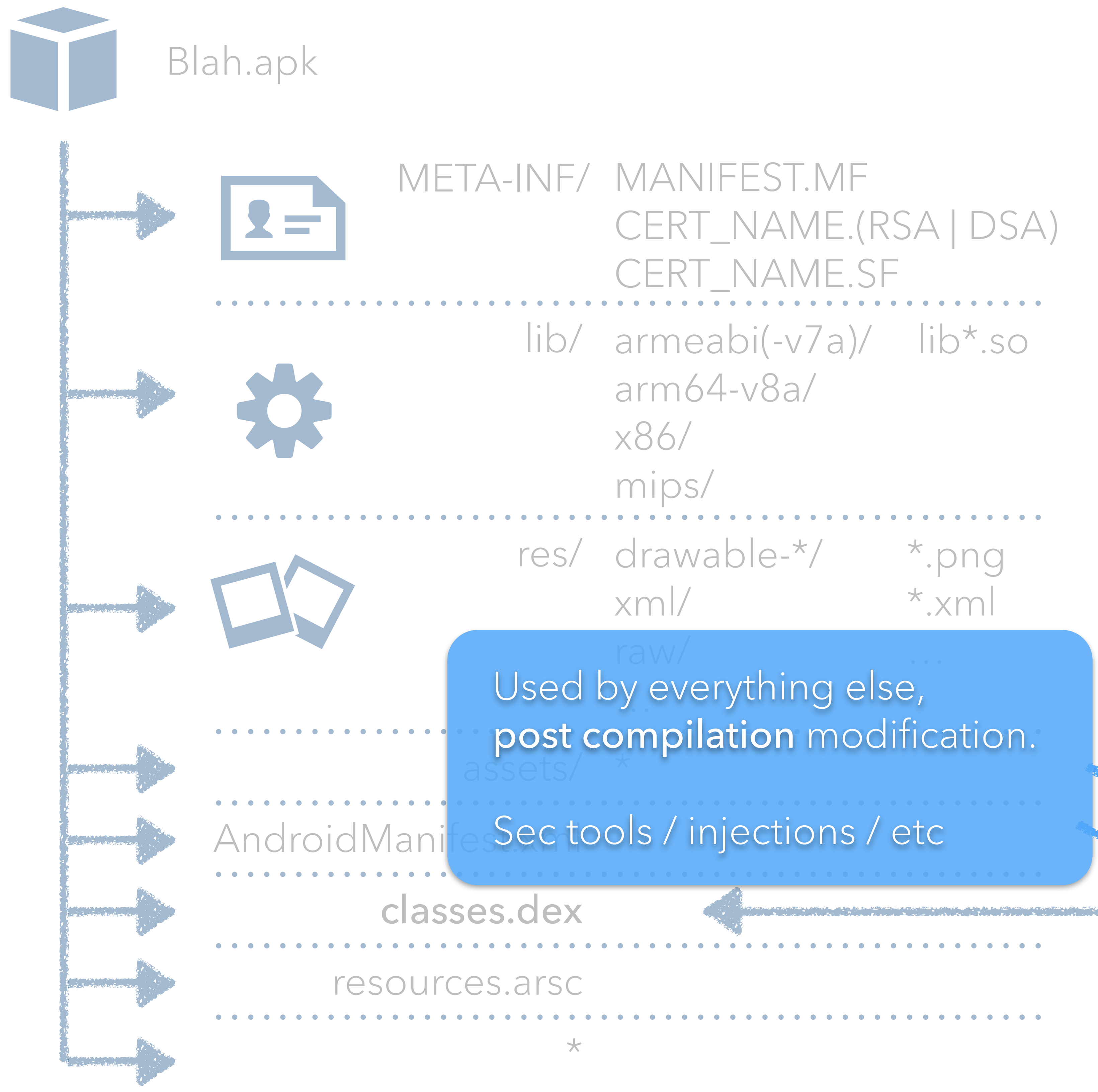
Contains executable Dalvik code

Optimized on install to:
- ODEX for DVM runtime
- OAT for ART runtime

Reverse with:
- smali / apktool
- IDA Pro
- jeb / jeb2
- androguard
- enjarify
- dex2jar + jad/jd
- jadx
- radare
- 010Editor Templates

ANDROID APPLICATION PACKAGING (APK)

application/vnd.android.package-archive



Used by everything else, **post compilation** modification.

Sec tools / injections / etc

Dalvik Executable
Compiled classes for DVM

Contains executable Dalvik code

Optimized on install to:
ODEX for DVM runtime
OAT for ART runtime

Created with:
dexmerge
dx
smali (dexlib1/2/2beta)
apktool (dexlib)

COMPILER FINGERPRINTING

diff / caleb

REDNAGA

AXML FILES

Relatively Simplistic…

- Normal tools create AXML file in a simple order
- AXML files don't need to be in a specific order
- Most tools **append** new structures to the file

AXML FILES

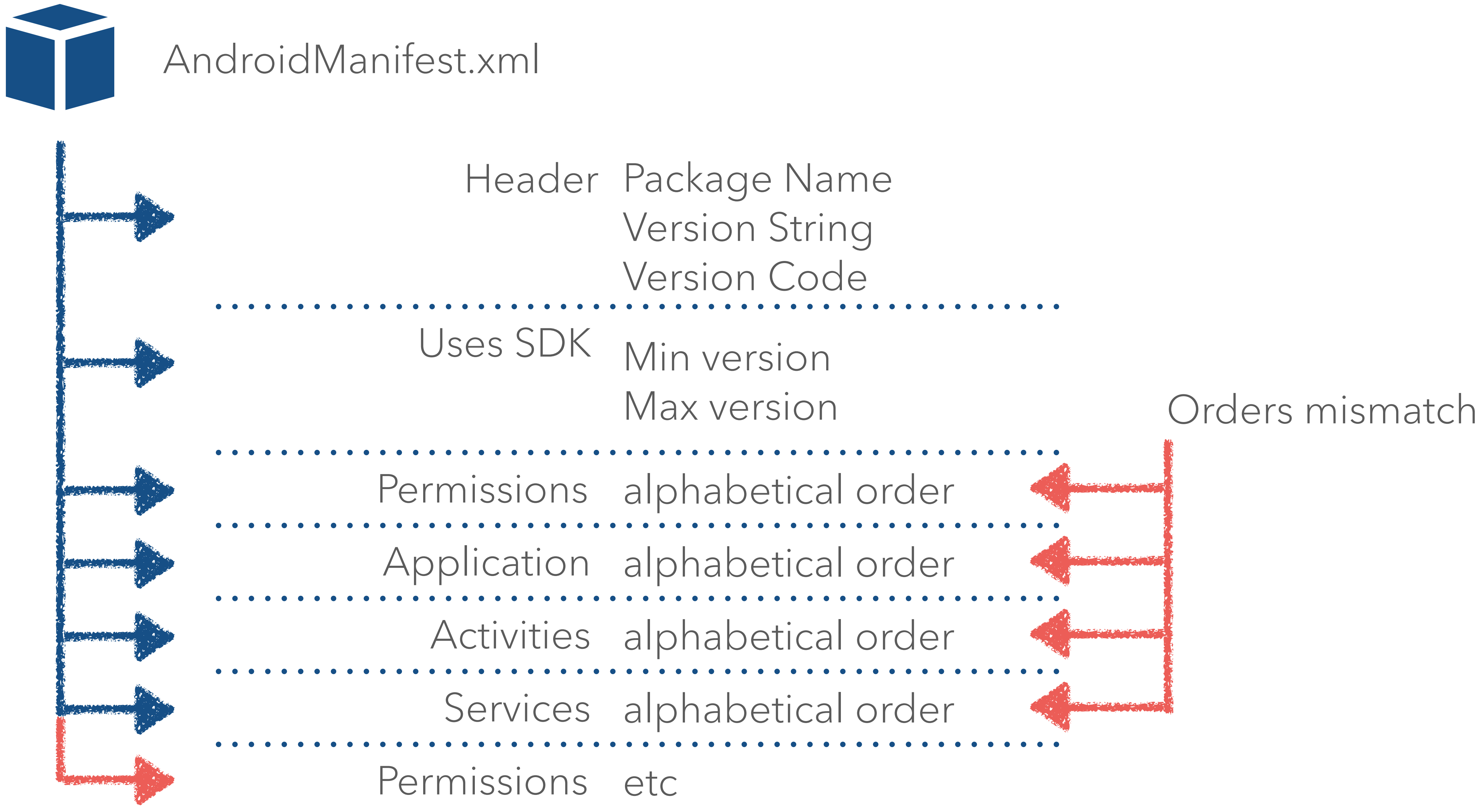
Normal Files

AndroidManifest.xml

Header	Package Name Version String Version Code
Uses SDK	Min version Max version
Permissions	alphabetical order
Application	alphabetical order
Activities	alphabetical order
Services	alphabetical order

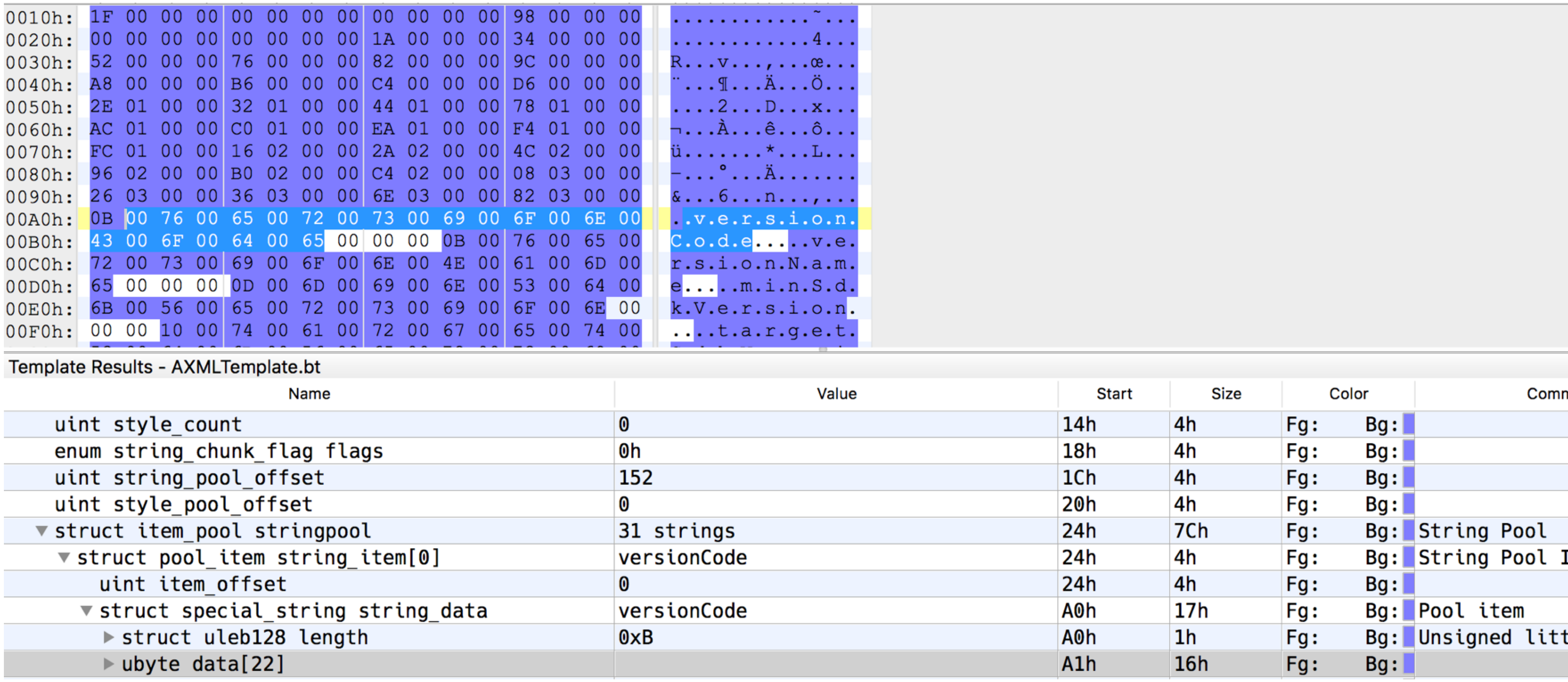
AXML FILES

Abnormal Files



AXML FILES

Normal Files



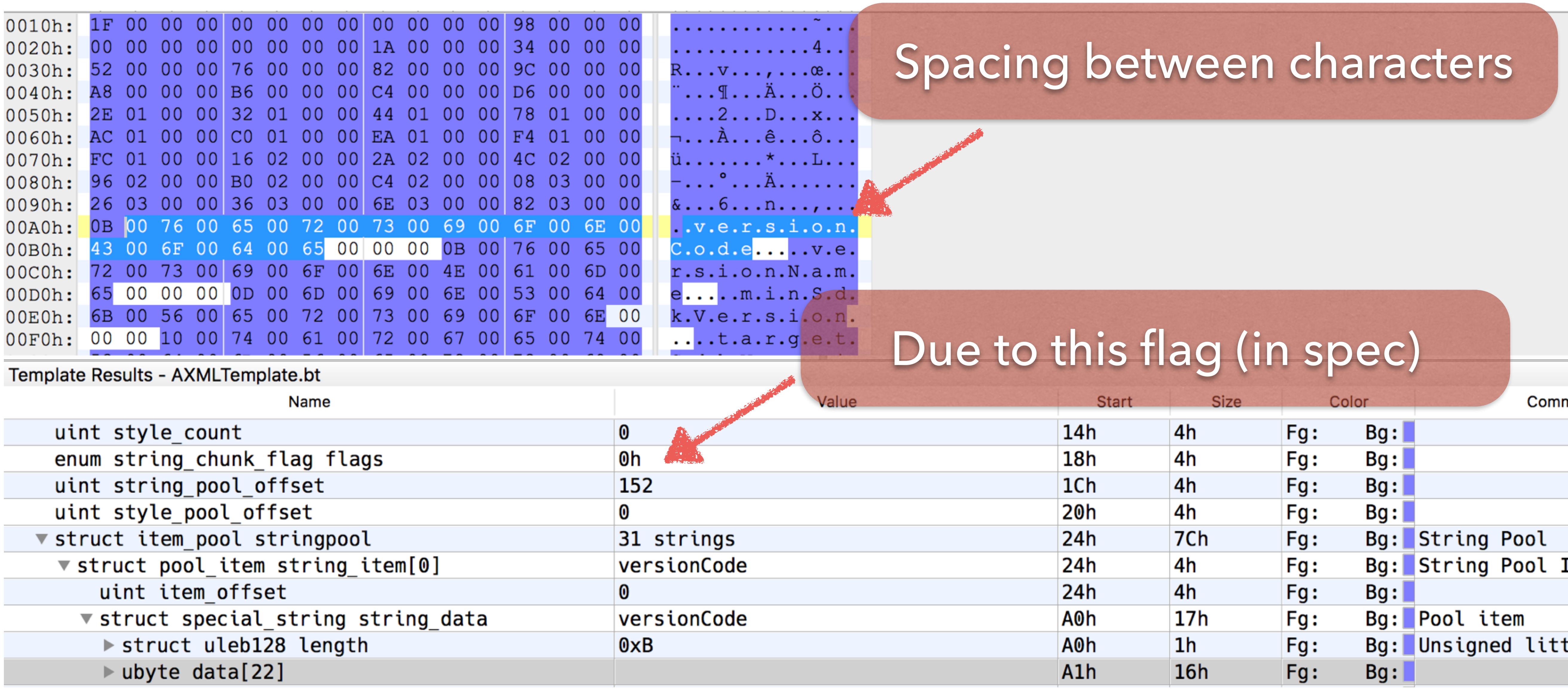
```
0010h: 1F 00 00 00 00 00 00 00 00 00 00 00 98 00 00 00  ............˜...
0020h: 00 00 00 00 00 00 00 00 1A 00 00 00 34 00 00 00  ............4...
0030h: 52 00 00 00 76 00 00 00 82 00 00 00 9C 00 00 00  R...v...‚...œ...
0040h: A8 00 00 00 B6 00 00 00 C4 00 00 00 D6 00 00 00  ¨...¶...Ä...Ö...
0050h: 2E 01 00 00 32 01 00 00 44 01 00 00 78 01 00 00  ....2...D...x...
0060h: AC 01 00 00 C0 01 00 00 EA 01 00 00 F4 01 00 00  ¬...À...ê...ô...
0070h: FC 01 00 00 16 02 00 00 2A 02 00 00 4C 02 00 00  ü.......*...L...
0080h: 96 02 00 00 B0 02 00 00 C4 02 00 00 08 03 00 00  –...°...Ä.......
0090h: 26 03 00 00 36 03 00 00 6E 03 00 00 82 03 00 00  &...6...n...‚...
00A0h: 0B 00 76 00 65 00 72 00 73 00 69 00 6F 00 6E 00  ..v.e.r.s.i.o.n.
00B0h: 43 00 6F 00 64 00 65 00 00 00 0B 00 76 00 65 00  C.o.d.e.....v.e.
00C0h: 72 00 73 00 69 00 6F 00 6E 00 4E 00 61 00 6D 00  r.s.i.o.n.N.a.m.
00D0h: 65 00 00 00 0D 00 6D 00 69 00 6E 00 53 00 64 00  e.....m.i.n.S.d.
00E0h: 6B 00 56 00 65 00 72 00 73 00 69 00 6F 00 6E 00  k.V.e.r.s.i.o.n.
00F0h: 00 00 10 00 74 00 61 00 72 00 67 00 65 00 74 00  ....t.a.r.g.e.t.
```

Template Results - AXMLTemplate.bt

Name	Value	Start	Size	Color	Comm
uint style_count	0	14h	4h	Fg: Bg: ▮	
enum string_chunk_flag flags	0h	18h	4h	Fg: Bg: ▮	
uint string_pool_offset	152	1Ch	4h	Fg: Bg: ▮	
uint style_pool_offset	0	20h	4h	Fg: Bg: ▮	
▼ struct item_pool stringpool	31 strings	24h	7Ch	Fg: Bg: ▮	String Pool
▼ struct pool_item string_item[0]	versionCode	24h	4h	Fg: Bg: ▮	String Pool I
uint item_offset	0	24h	4h	Fg: Bg: ▮	
▼ struct special_string string_data	versionCode	A0h	17h	Fg: Bg: ▮	Pool item
▶ struct uleb128 length	0xB	A0h	1h	Fg: Bg: ▮	Unsigned litt
▶ ubyte data[22]		A1h	16h	Fg: Bg: ▮	

AXML FILES

Normal Files



```
0010h: 1F 00 00 00 00 00 00 00 00 00 00 00 98 00 00 00  ............˜...
0020h: 00 00 00 00 00 00 00 00 1A 00 00 00 34 00 00 00  ............4...
0030h: 52 00 00 00 76 00 00 00 82 00 00 00 9C 00 00 00  R...v...‚...œ...
0040h: A8 00 00 00 B6 00 00 00 C4 00 00 00 D6 00 00 00  ¨...¶...Ä...Ö...
0050h: 2E 01 00 00 32 01 00 00 44 01 00 00 78 01 00 00  ....2...D...x...
0060h: AC 01 00 00 C0 01 00 00 EA 01 00 00 F4 01 00 00  ¬...À...ê...ô...
0070h: FC 01 00 00 16 02 00 00 2A 02 00 00 4C 02 00 00  ü.......*...L...
0080h: 96 02 00 00 B0 02 00 00 C4 02 00 00 08 03 00 00  –...°...Ä.......
0090h: 26 03 00 00 36 03 00 00 6E 03 00 00 82 03 00 00  &...6...n...‚...
00A0h: 0B 00 76 00 65 00 72 00 73 00 69 00 6F 00 6E 00  ..v.e.r.s.i.o.n.
00B0h: 43 00 6F 00 64 00 65 00 00 00 00 0B 00 76 00 65 00  C.o.d.e.....v.e.
00C0h: 72 00 73 00 69 00 6F 00 6E 00 4E 00 61 00 6D 00  r.s.i.o.n.N.a.m.
00D0h: 65 00 00 00 0D 00 6D 00 69 00 6E 00 53 00 64 00  e.....m.i.n.S.d.
00E0h: 6B 00 56 00 65 00 72 00 73 00 69 00 6F 00 6E 00  k.V.e.r.s.i.o.n.
00F0h: 00 00 10 00 74 00 61 00 72 00 67 00 65 00 74 00  ....t.a.r.g.e.t.
```

Template Results - AXMLTemplate.bt

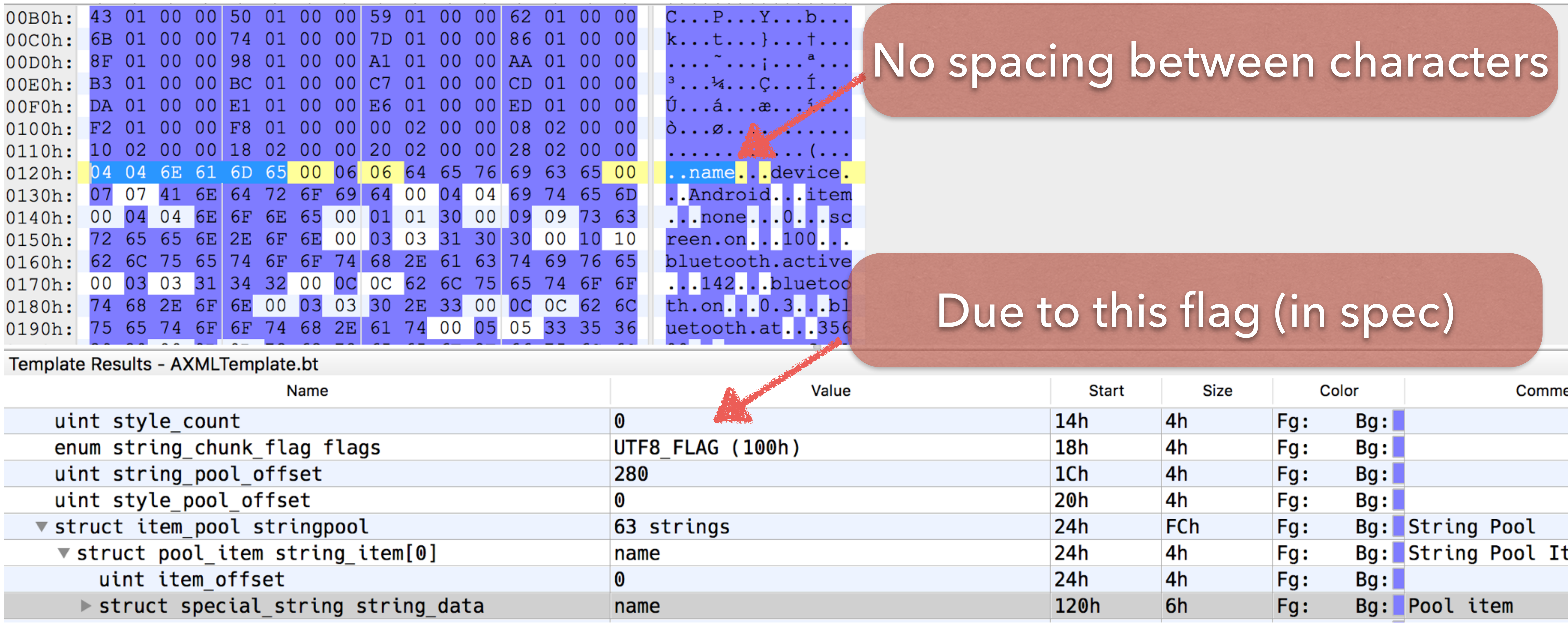
```
Name                                     Value            Start  Size  Color       Comm
   uint style_count                      0                14h    4h    Fg:  Bg:
   enum string_chunk_flag flags          0h               18h    4h    Fg:  Bg:
   uint string_pool_offset               152              1Ch    4h    Fg:  Bg:
   uint style_pool_offset                0                20h    4h    Fg:  Bg:
 ▼ struct item_pool stringpool           31 strings       24h    7Ch   Fg:  Bg:  String Pool
   ▼ struct pool_item string_item[0]     versionCode      24h    4h    Fg:  Bg:  String Pool I
       uint item_offset                  0                24h    4h    Fg:  Bg:
     ▼ struct special_string string_data versionCode      A0h    17h   Fg:  Bg:  Pool item
       ▶ struct uleb128 length           0xB              A0h    1h    Fg:  Bg:  Unsigned litt
       ▶ ubyte data[22]                                   A1h    16h   Fg:  Bg:
```

Spacing between characters

Due to this flag (in spec)

AXML FILES

Abnormal files which broke old AXMLPrinter2 lib



```
00B0h: 43 01 00 00 50 01 00 00 59 01 00 00 62 01 00 00  C...P...Y...b...
00C0h: 6B 01 00 00 74 01 00 00 7D 01 00 00 86 01 00 00  k...t...}...†...
00D0h: 8F 01 00 00 98 01 00 00 A1 01 00 00 AA 01 00 00  ....˜...¡...ª...
00E0h: B3 01 00 00 BC 01 00 00 C7 01 00 00 CD 01 00 00  ³...¼...Ç...Í...
00F0h: DA 01 00 00 E1 01 00 00 E6 01 00 00 ED 01 00 00  Ú...á...æ...í...
0100h: F2 01 00 00 F8 01 00 00 00 02 00 00 08 02 00 00  ò...ø...........
0110h: 10 02 00 00 18 02 00 00 20 02 00 00 28 02 00 00  ........ ...(...
0120h: 04 04 6E 61 6D 65 00 06 06 64 65 76 69 63 65 00  ..name...device.
0130h: 07 07 41 6E 64 72 6F 69 64 00 04 04 69 74 65 6D  ..Android...item
0140h: 00 04 04 6E 6F 6E 65 00 01 01 30 00 09 09 73 63  ...none...0...sc
0150h: 72 65 65 6E 2E 6F 6E 00 03 03 31 30 30 00 10 10  reen.on...100...
0160h: 62 6C 75 65 74 6F 6F 74 68 2E 61 63 74 69 76 65  bluetooth.active
0170h: 00 03 03 31 34 32 00 0C 0C 62 6C 75 65 74 6F 6F  ...142...bluetoo
0180h: 74 68 2E 6F 6E 00 03 03 30 2E 33 00 0C 0C 62 6C  th.on...0.3...bl
0190h: 75 65 74 6F 6F 74 68 2E 61 74 00 05 05 33 35 36  uetooth.at...356
```

No spacing between characters

Due to this flag (in spec)

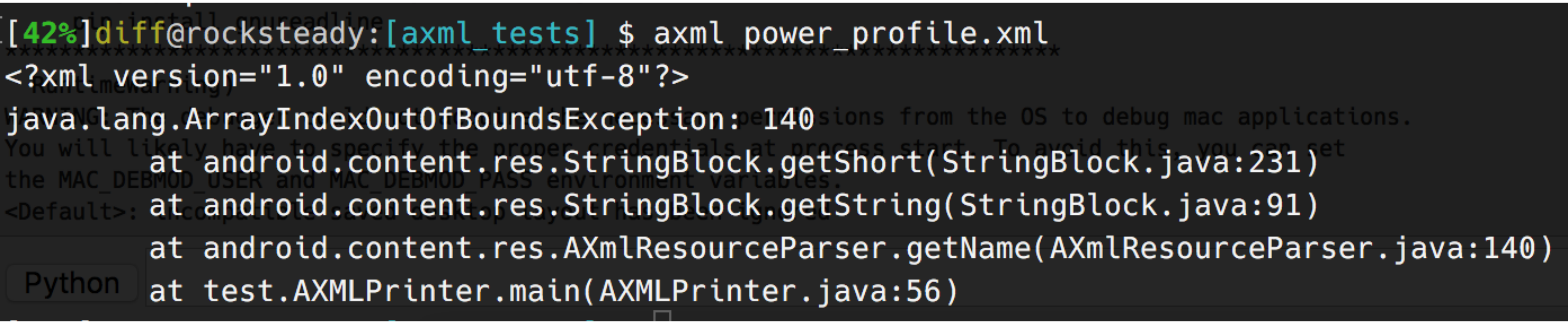
Template Results - AXMLTemplate.bt

Name	Value	Start	Size	Color		Comme
uint style_count	0	14h	4h	Fg:	Bg:	
enum string_chunk_flag flags	UTF8_FLAG (100h)	18h	4h	Fg:	Bg:	
uint string_pool_offset	280	1Ch	4h	Fg:	Bg:	
uint style_pool_offset	0	20h	4h	Fg:	Bg:	
▼ struct item_pool stringpool	63 strings	24h	FCh	Fg:	Bg:	String Pool
▼ struct pool_item string_item[0]	name	24h	4h	Fg:	Bg:	String Pool It
uint item_offset	0	24h	4h	Fg:	Bg:	
▶ struct special_string string_data	name	120h	6h	Fg:	Bg:	Pool item

This was back ported from JEB to APKTOOL…

AXML FILES

Protectors / Anti* tricks

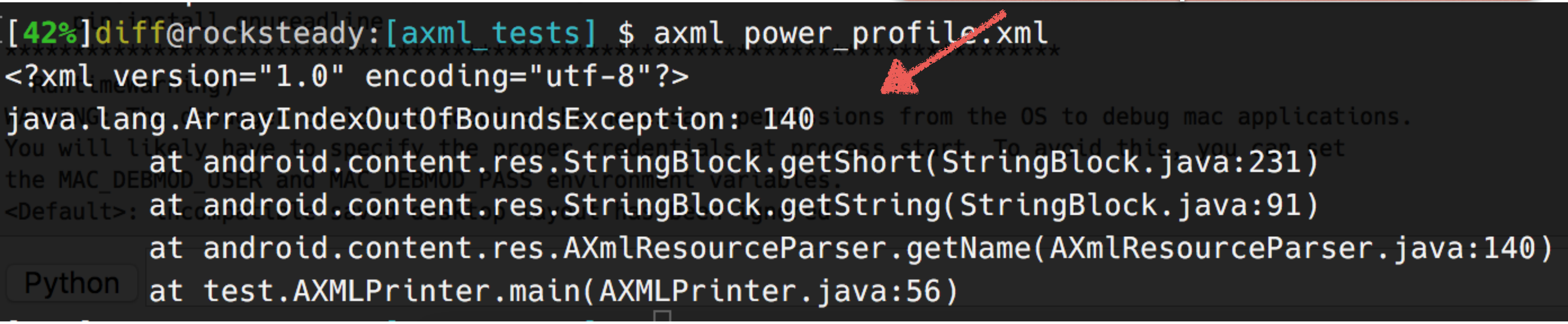


```
[42%]diff@rocksteady:[axml_tests] $ axml power_profile.xml
<?xml version="1.0" encoding="utf-8"?>
java.lang.ArrayIndexOutOfBoundsException: 140
        at android.content.res.StringBlock.getShort(StringBlock.java:231)
        at android.content.res.StringBlock.getString(StringBlock.java:91)
        at android.content.res.AXmlResourceParser.getName(AXmlResourceParser.java:140)
        at test.AXMLPrinter.main(AXMLPrinter.java:56)
```

AXML FILES

Protectors / Anti* tricks

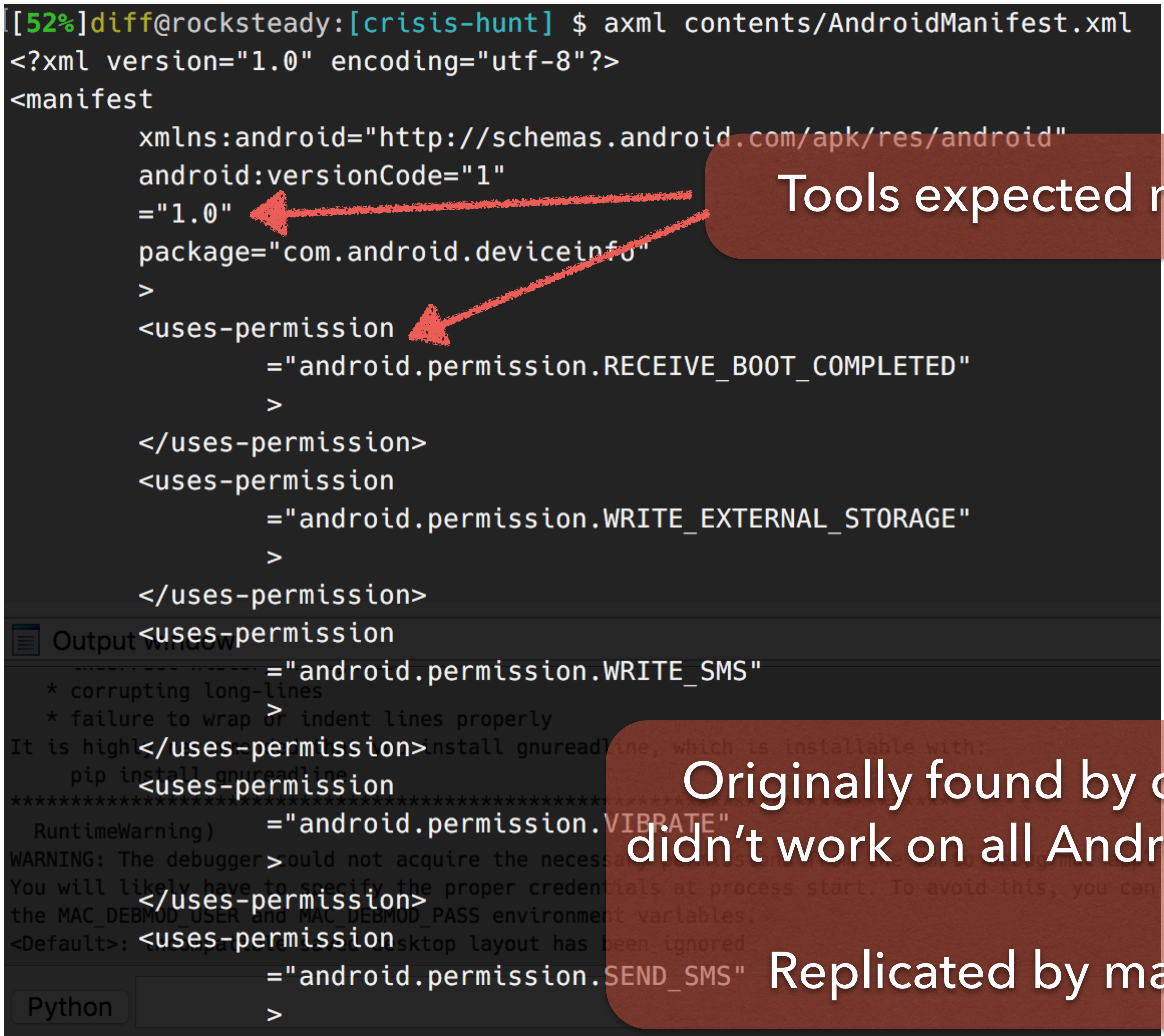
Expects certain values to be present



```
[42%]diff@rocksteady:[axml_tests] $ axml power_profile.xml
<?xml version="1.0" encoding="utf-8"?>
java.lang.ArrayIndexOutOfBoundsException: 140
        at android.content.res.StringBlock.getShort(StringBlock.java:231)
        at android.content.res.StringBlock.getString(StringBlock.java:91)
        at android.content.res.AXmlResourceParser.getName(AXmlResourceParser.java:140)
        at test.AXMLPrinter.main(AXMLPrinter.java:56)
```

AXML FILES

Protectors / Anti* tricks



```
[52%]diff@rocksteady:[crisis-hunt] $ axml contents/AndroidManifest.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:versionCode="1"
        ="1.0"
        package="com.android.deviceinfo"
        >
        <uses-permission
                ="android.permission.RECEIVE_BOOT_COMPLETED"
                >
        </uses-permission>
        <uses-permission
                ="android.permission.WRITE_EXTERNAL_STORAGE"
                >
        </uses-permission>
        <uses-permission
                ="android.permission.WRITE_SMS"
                >
        </uses-permission>
        <uses-permission
                ="android.permission.VIBRATE"
                >
        </uses-permission>
        <uses-permission
                ="android.permission.SEND_SMS"
                >
```

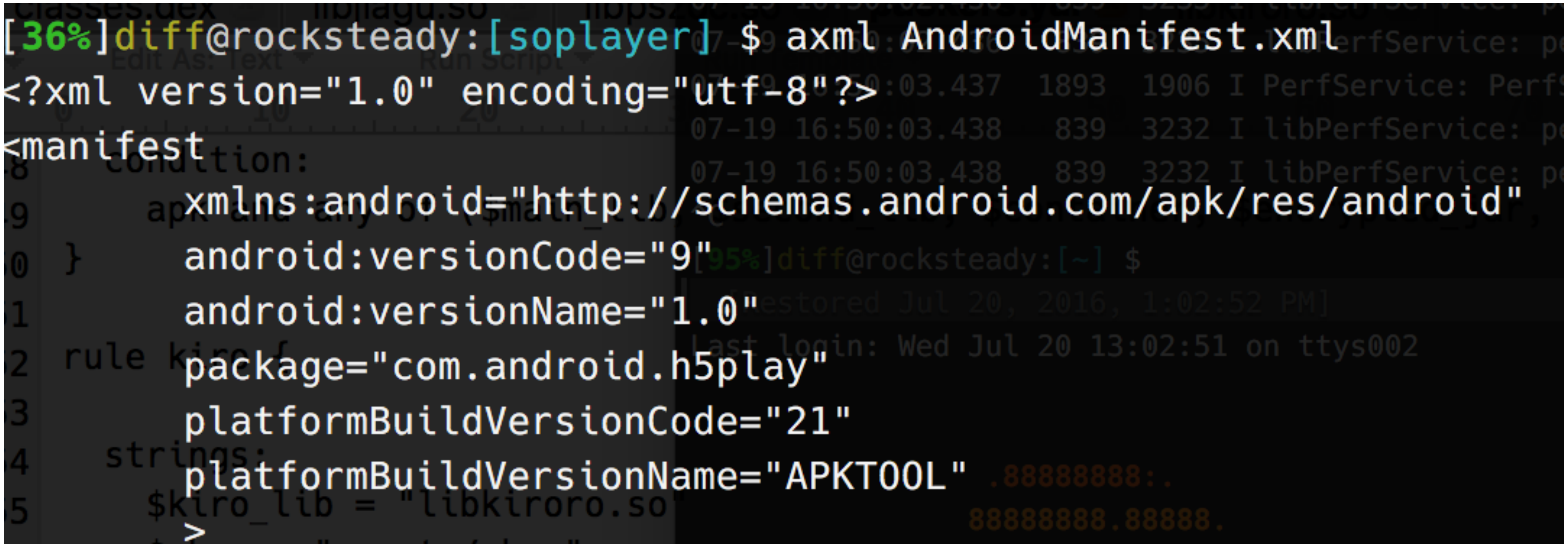
Tools expected name tags

Originally found by dexguard, didn't work on all Android versions

Replicated by malware

AXML FILES

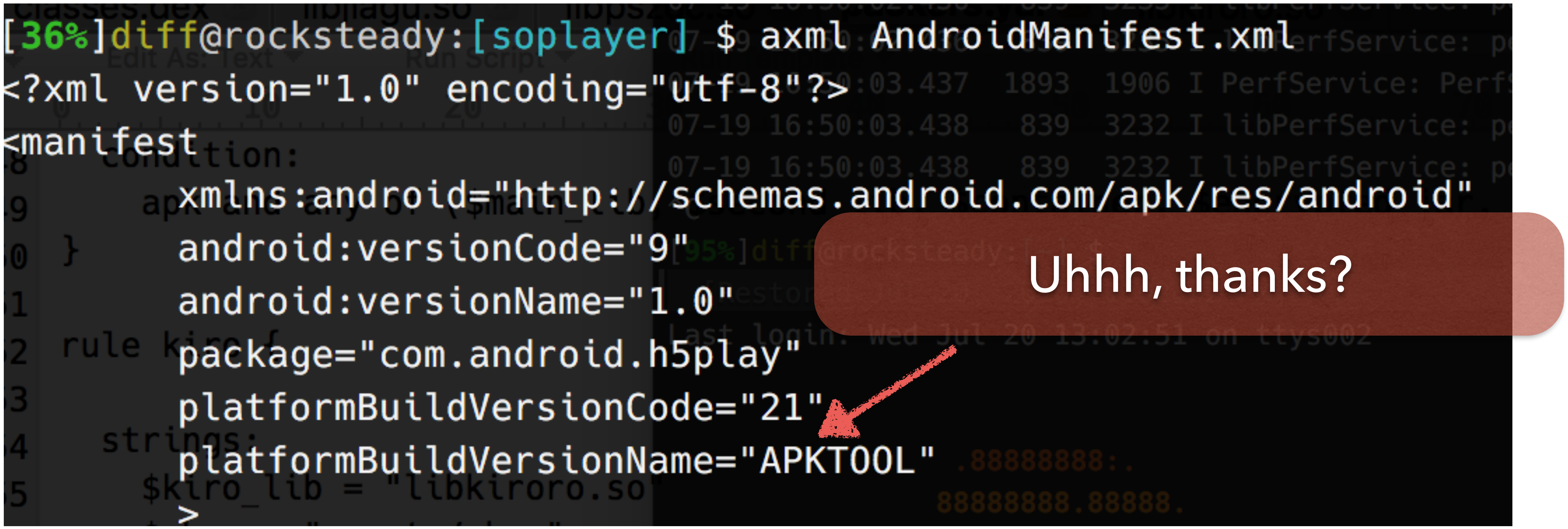
APKTOOL Specifics… easy, easy



```
[36%]diff@rocksteady:[soplayer] $ axml AndroidManifest.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:versionCode="9"
        android:versionName="1.0"
        package="com.android.h5play"
        platformBuildVersionCode="21"
        platformBuildVersionName="APKTOOL"
        >
```

AXML FILES

APKTOOL Specifics… easy, easy



```
[36%]diff@rocksteady:[soplayer] $ axml AndroidManifest.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:versionCode="9"
        android:versionName="1.0"
        package="com.android.h5play"
        platformBuildVersionCode="21"
        platformBuildVersionName="APKTOOL"
        >
```

DEX FILES

- DEX format is … flexible
- Only a few different compilers
- Slight variations between each one
- Obfuscators do really weird stuff too

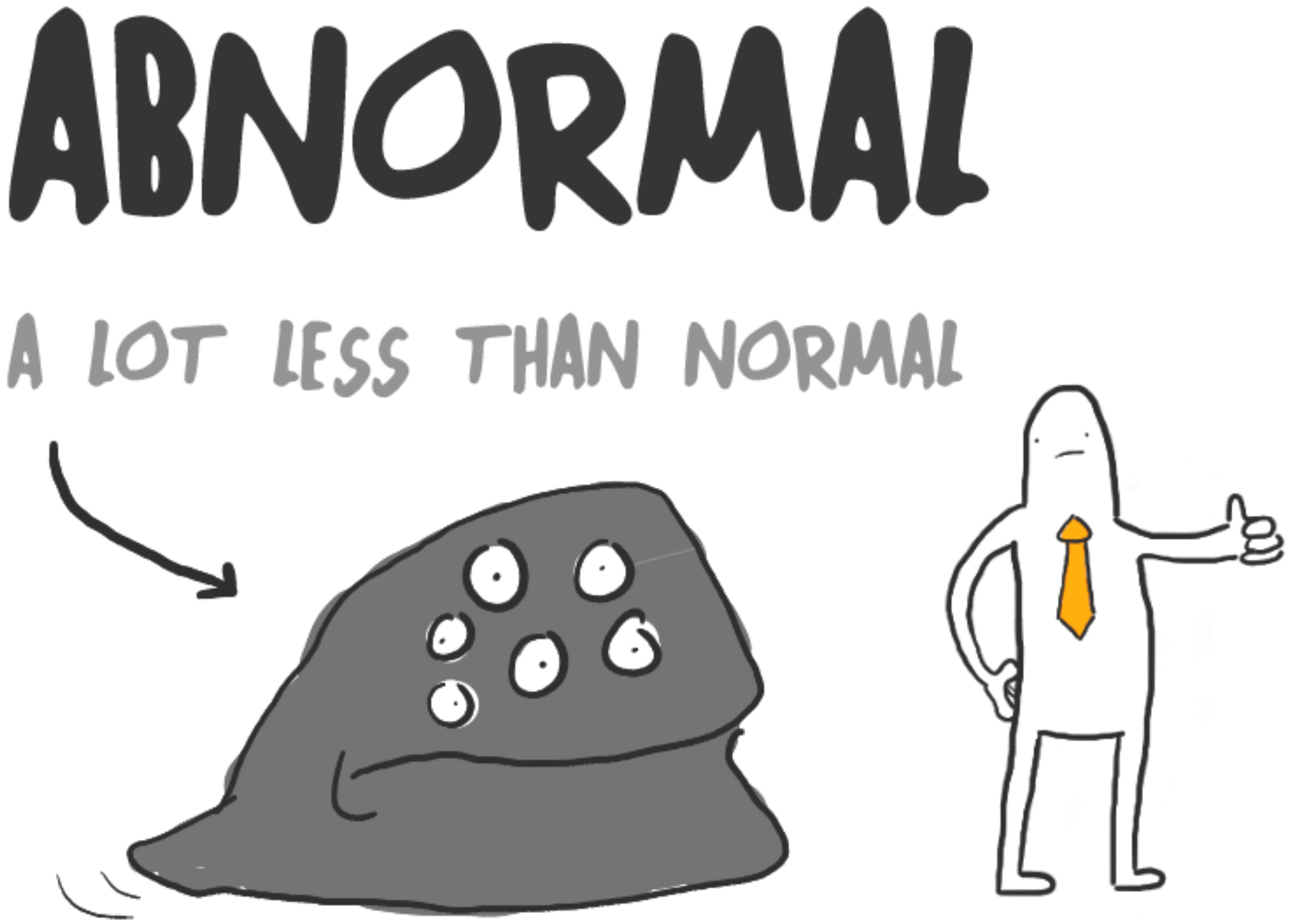
INVESTIGATION

- Built lots of DEX files with different tools
- Compared files with 010Editor
- Found some differences but wanted to know **all** of them
- Read DEX format specification
- Gave up since it doesn’t include enough detail
- Very carefully read the source code
- Found many fingerprintable “characteristics”

CHARACTERISTICS

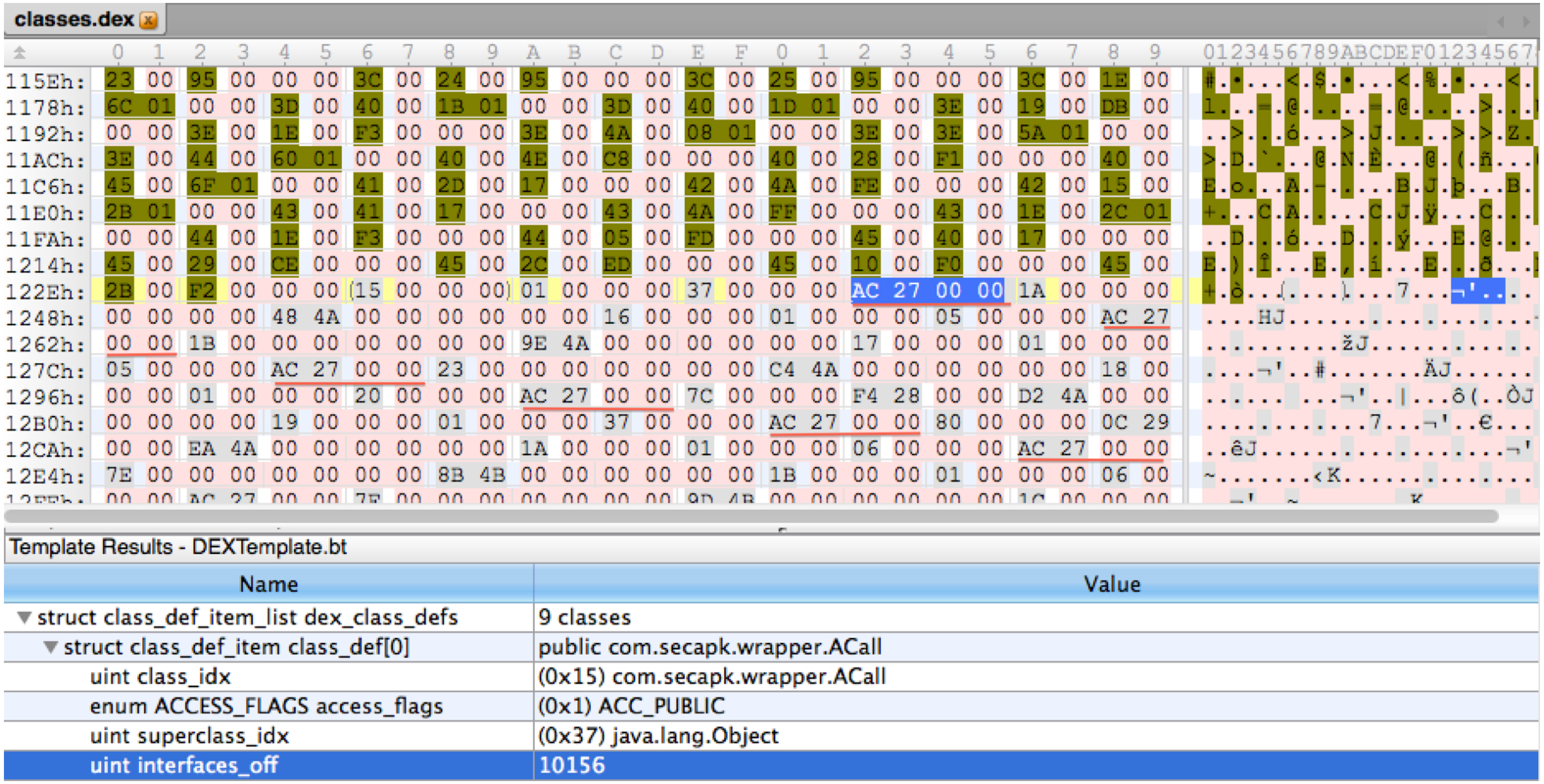
These may be abnormal…

1. Class interfaces
2. Class paths
3. Endian tag
4. Header size
5. Link section
6. String sorting
7. Map type order
8. Section contiguity



ABNORMAL_CLASS_INTERFACES

- Implies: early dexlib 2.x (smali)



classes.dex

```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0  1  2  3  4  5  6  7  8  9   0123456789ABCDEF01234567
115Eh: 23 00 95 00 00 00 3C 00 24 00 95 00 00 00 3C 00 25 00 95 00 00 00 3C 00 1E 00   #.•...<.$.•...<.%.•...<.
1178h: 6C 01 00 00 3D 00 40 00 1B 01 00 00 3D 00 40 00 1D 01 00 00 3E 00 19 00 DB 00   l...=.@.....=.@.....>...
1192h: 00 00 3E 00 1E 00 F3 00 00 00 3E 00 4A 00 08 01 00 00 3E 00 3E 00 5A 01 00 00   ..>...ó...>.J.....>.>.Z.
11ACh: 3E 00 44 00 60 01 00 00 40 00 4E 00 C8 00 00 00 40 00 28 00 F1 00 00 00 40 00   >.D.`...@.N.È...@.(.ñ...
11C6h: 45 00 6F 01 00 00 41 00 2D 00 17 00 00 00 42 00 4A 00 FE 00 00 00 42 00 15 00   E.o...A.-.....B.J.þ...B.
11E0h: 2B 01 00 00 43 00 41 00 17 00 00 00 43 00 4A 00 FF 00 00 00 43 00 1E 00 2C 01   +...C.A.....C.J.ÿ...C...
11FAh: 00 00 44 00 1E 00 F3 00 00 00 44 00 05 00 FD 00 00 00 45 00 40 00 17 00 00 00   ..D...ó...D...ý...E.@...
1214h: 45 00 29 00 CE 00 00 00 45 00 2C 00 ED 00 00 00 45 00 10 00 F0 00 00 00 45 00   E.).Î...E.,.í...E...ð...
122Eh: 2B 00 F2 00 00 00(15 00 00 00)01 00 00 00 37 00 00 00 AC 27 00 00 1A 00 00 00   +.ò...(....)...7...¬'...
1248h: 00 00 00 00 48 4A 00 00 00 00 00 00 16 00 00 00 01 00 00 00 05 00 00 00 AC 27   ....HJ..................
1262h: 00 00 1B 00 00 00 00 00 00 00 9E 4A 00 00 00 00 00 00 17 00 00 00 01 00 00 00   ..........žJ............
127Ch: 05 00 00 00 AC 27 00 00 23 00 00 00 00 00 00 00 C4 4A 00 00 00 00 00 00 18 00   ....¬'..#.......ÄJ......
1296h: 00 00 01 00 00 00 20 00 00 00 AC 27 00 00 7C 00 00 00 F4 28 00 00 D2 4A 00 00   ...... ...¬'..|...ô(..ÒJ
12B0h: 00 00 00 00 19 00 00 00 01 00 00 00 37 00 00 00 AC 27 00 00 80 00 00 00 0C 29   ...........7...¬'..€....
12CAh: 00 00 EA 4A 00 00 00 00 00 00 1A 00 00 00 01 00 00 00 06 00 00 00 AC 27 00 00   ..êJ..................¬'
12E4h: 7E 00 00 00 00 00 00 00 8B 4B 00 00 00 00 00 00 1B 00 00 00 01 00 00 00 06 00   ~.......‹K..............
```

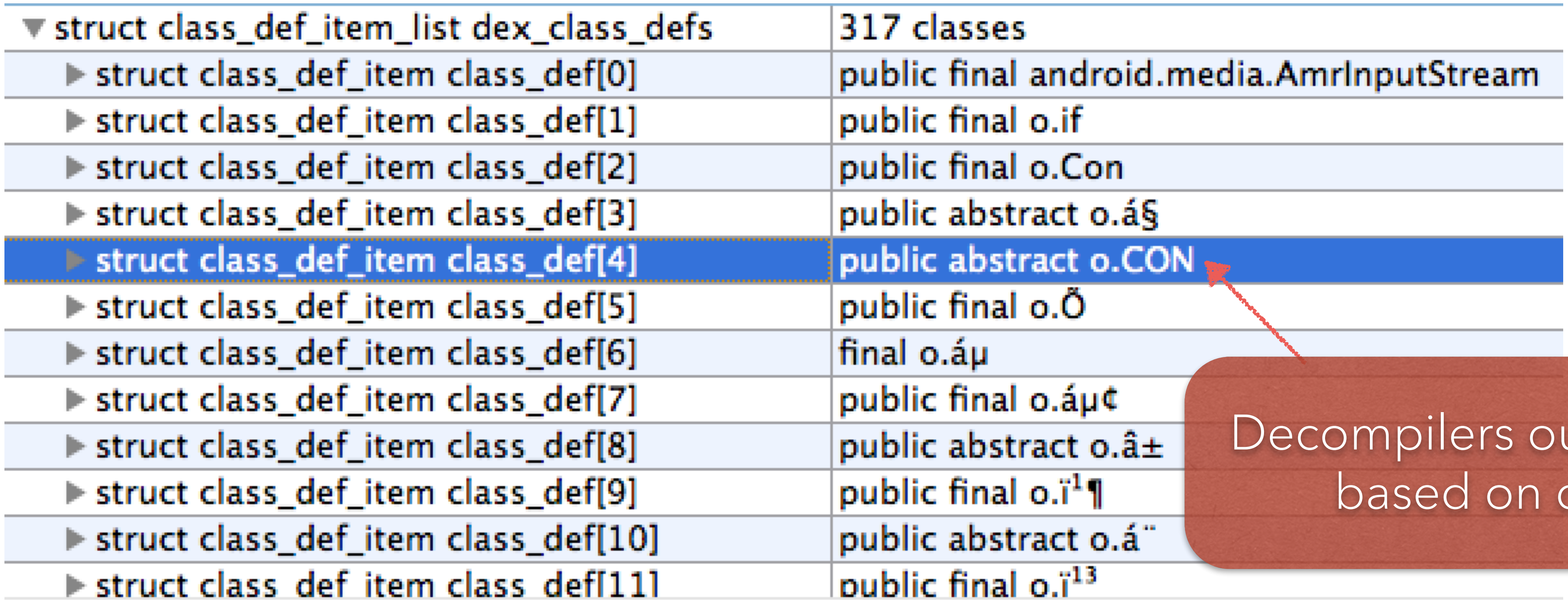
Template Results - DEXTemplate.bt

Name	Value
struct class_def_item_list dex_class_defs	9 classes
struct class_def_item class_def[0]	public com.secapk.wrapper.ACall
uint class_idx	(0x15) com.secapk.wrapper.ACall
enum ACCESS_FLAGS access_flags	(0x1) ACC_PUBLIC
uint superclass_idx	(0x37) java.lang.Object
uint interfaces_off	10156

If class has no interface, dx uses interfaces_off = 0
dexlib gives offset to address with null bytes (10156 is null)

ABNORMAL_CLASS_PATH

· Implies: anti-decompiler



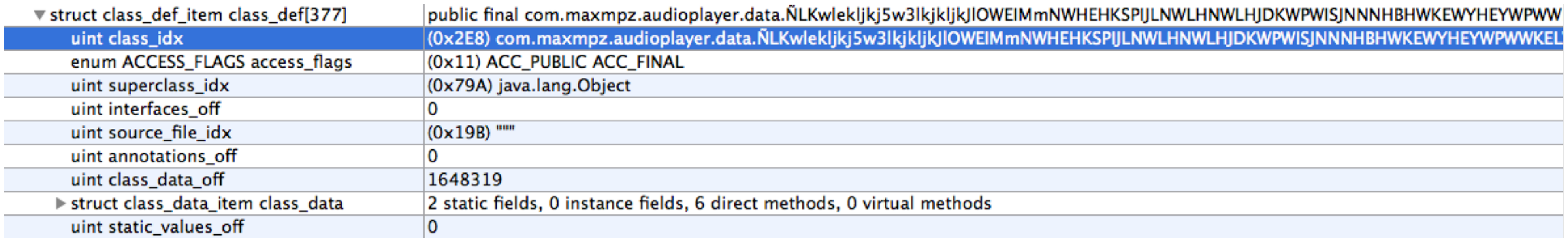
▼ struct class_def_item_list dex_class_defs	317 classes
▶ struct class_def_item class_def[0]	public final android.media.AmrInputStream
▶ struct class_def_item class_def[1]	public final o.if
▶ struct class_def_item class_def[2]	public final o.Con
▶ struct class_def_item class_def[3]	public abstract o.á§
▶ struct class_def_item class_def[4]	public abstract o.CON
▶ struct class_def_item class_def[5]	public final o.Ő
▶ struct class_def_item class_def[6]	final o.áµ
▶ struct class_def_item class_def[7]	public final o.áµ¢
▶ struct class_def_item class_def[8]	public abstract o.â±
▶ struct class_def_item class_def[9]	public final o.ï¹¶
▶ struct class_def_item class_def[10]	public abstract o.á¨
▶ struct class_def_item class_def[11]	public final o.ï¹³

Decompilers output filenames based on class name

Invalid Windows filenames:
CON, PRN, AUX, CLOCK$, NUL
COM1, COM2, COM3, COM4
LPT1, LPT2, LPT3, LPT4

ABNORMAL_CLASS_PATH

- Implies: anti-decompiler



▼ struct class_def_item class_def[377]	public final com.maxmpz.audioplayer.data.ÑLKwlekljkj5w3lkjkljkJlOWEIMmNWHEHKSPIJLNWLHNWLHJDKWPWISJNNNHBHWKEWYHEYWPWW
uint class_idx	(0x2E8) com.maxmpz.audioplayer.data.ÑLKwlekljkj5w3lkjkljkJlOWEIMmNWHEHKSPIJLNWLHNWLHJDKWPWISJNNNHBHWKEWYHEYWPWWKEL
enum ACCESS_FLAGS access_flags	(0x11) ACC_PUBLIC ACC_FINAL
uint superclass_idx	(0x79A) java.lang.Object
uint interfaces_off	0
uint source_file_idx	(0x19B) """"
uint annotations_off	0
uint class_data_off	1648319
▶ struct class_data_item class_data	2 static fields, 0 instance fields, 6 direct methods, 0 virtual methods
uint static_values_off	0

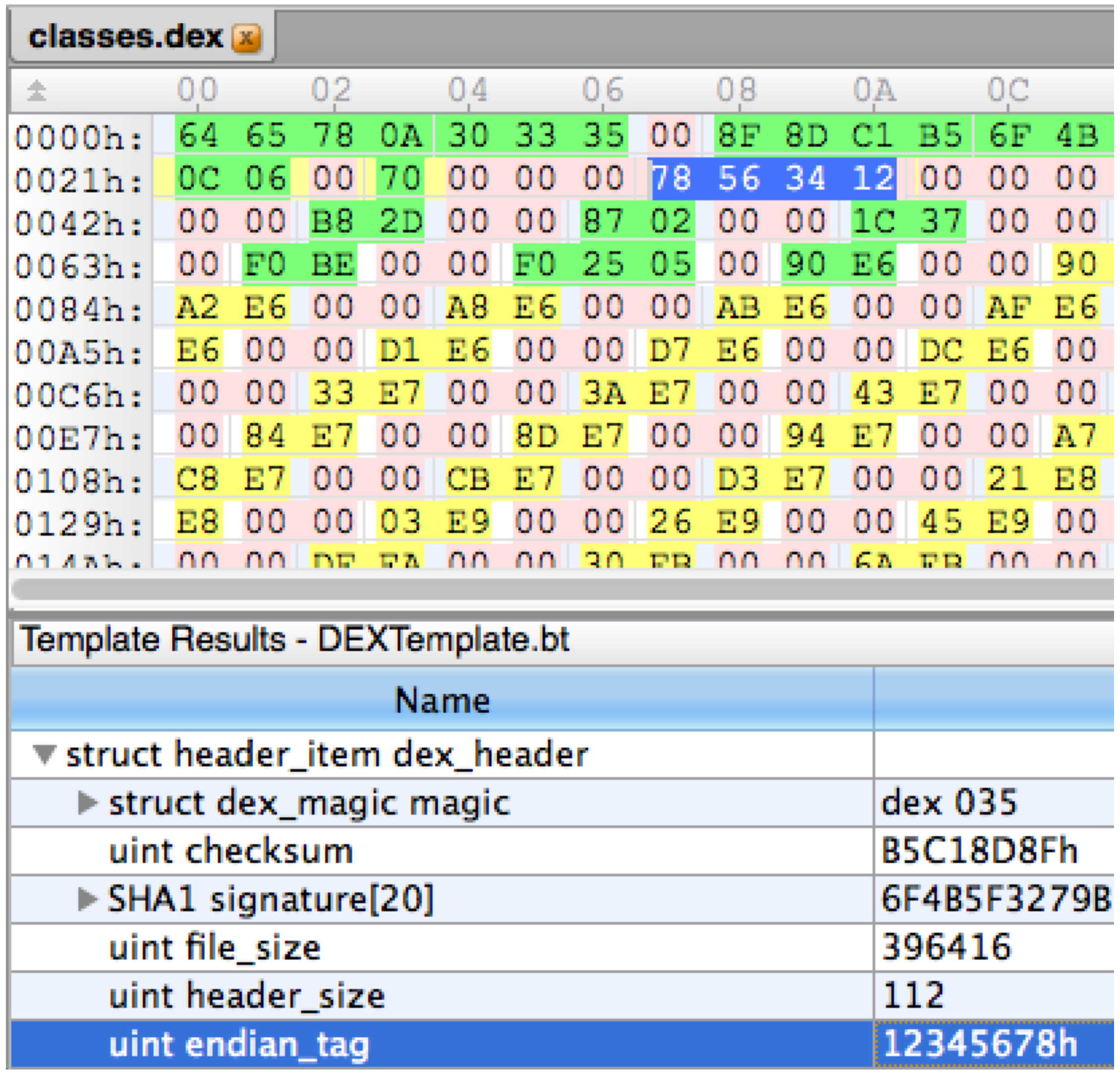
"com.maxmpz.audioplayer.data.ÑLKwlekljkj5w3lkjkljkJlOWEIMmNWHEHKSPIJLNWLHNWLHJDKWPWISJNNNHBHWKEWYHEYWPWWKELWJEKWEWNELWJEJHWELKWEWUEWIEKWLRJFKWNENWKJEJKWHEKWJEJHWKEWJHRKLWHJEKWJEJHWJEHWHEKWEHWHEHjehhwkjrhwerwnbewnrwemnrwkjh5n4m4mwn54mnkhjJNdenrrrr3453nmNMEWERTENRNERMERJEJRNNWKJEWNEWWKEJWEÐ˜"

Looks legit!

Class name used for filename!
Too long for Windows
Most Linux file systems have no limit
NTFS limited to 255 characters per part

ABNORMAL_ENDIAN_MAGIC

Implies: weird, shouldn't run on any Android device



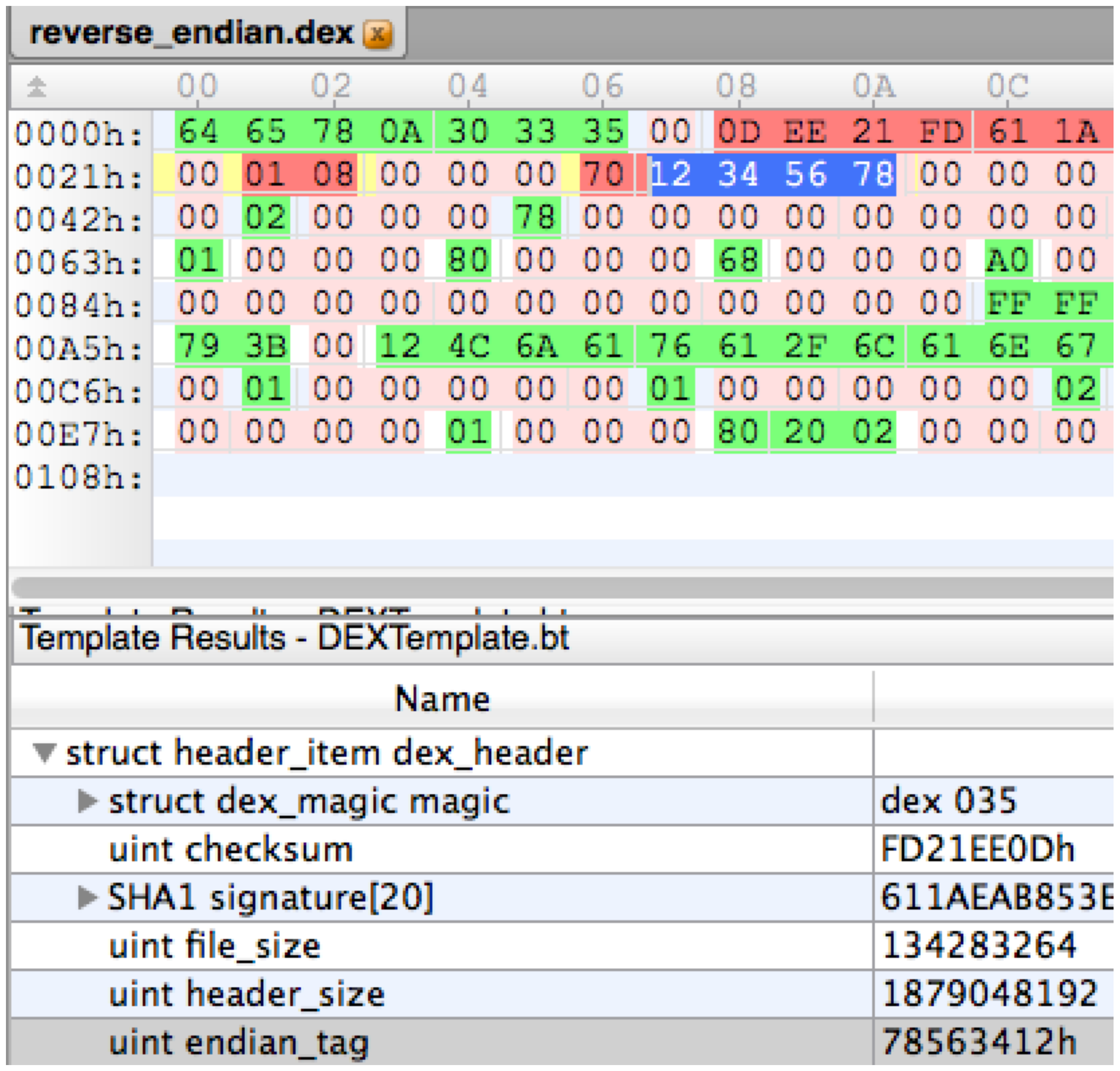
classes.dex

```
       00    02    04    06    08    0A    0C
0000h: 64 65 78 0A 30 33 35 00 8F 8D C1 B5 6F 4B
0021h: 0C 06 00 70 00 00 00 78 56 34 12 00 00 00
0042h: 00 00 B8 2D 00 00 87 02 00 00 1C 37 00 00
0063h: 00 F0 BE 00 00 F0 25 05 00 90 E6 00 00 90
0084h: A2 E6 00 00 A8 E6 00 00 AB E6 00 00 AF E6
00A5h: E6 00 00 D1 E6 00 00 D7 E6 00 00 DC E6 00
00C6h: 00 00 33 E7 00 00 3A E7 00 00 43 E7 00 00
00E7h: 00 84 E7 00 00 8D E7 00 00 94 E7 00 00 A7
0108h: C8 E7 00 00 CB E7 00 00 D3 E7 00 00 21 E8
0129h: E8 00 00 03 E9 00 00 26 E9 00 00 45 E9 00
014Ah: 00 00 DF EA 00 00 30 EB 00 00 6A EB 00 00
```

Template Results - DEXTemplate.bt

Name	
▼ struct header_item dex_header	
▶ struct dex_magic magic	dex 035
uint checksum	B5C18D8Fh
▶ SHA1 signature[20]	6F4B5F3279B
uint file_size	396416
uint header_size	112
uint endian_tag	12345678h

Big Endian
(Weird)



reverse_endian.dex

```
       00    02    04    06    08    0A    0C
0000h: 64 65 78 0A 30 33 35 00 0D EE 21 FD 61 1A
0021h: 00 01 08 00 00 00 70 12 34 56 78 00 00 00
0042h: 00 02 00 00 00 78 00 00 00 00 00 00 00 00
0063h: 01 00 00 00 80 00 00 00 68 00 00 00 A0 00
0084h: 00 00 00 00 00 00 00 00 00 00 00 00 FF FF
00A5h: 79 3B 00 12 4C 6A 61 76 61 2F 6C 61 6E 67
00C6h: 00 01 00 00 00 00 00 01 00 00 00 00 00 02
00E7h: 00 00 00 00 01 00 00 00 80 20 02 00 00 00
0108h:
```

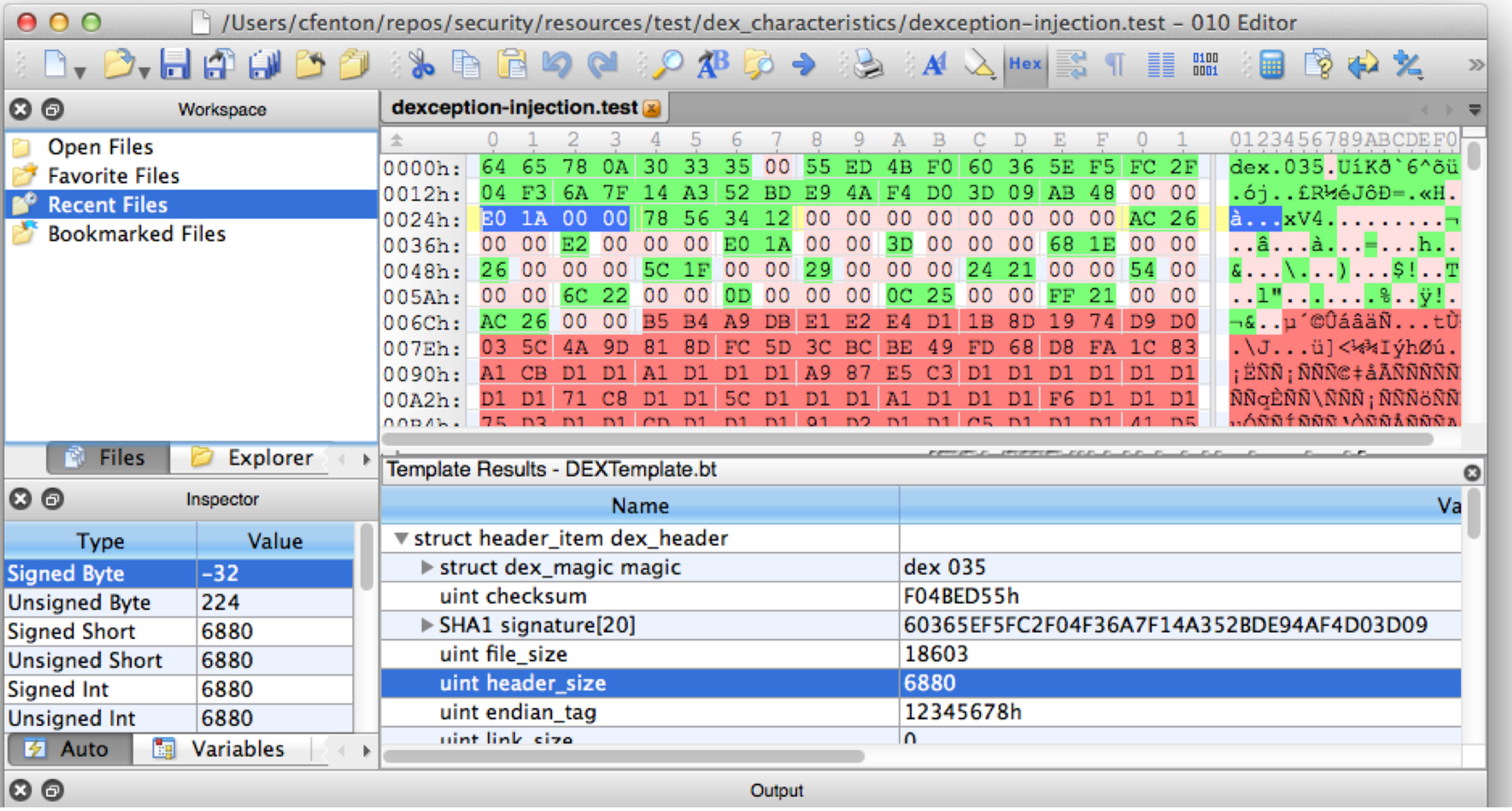
Template Results - DEXTemplate.bt

Name	
▼ struct header_item dex_header	
▶ struct dex_magic magic	dex 035
uint checksum	FD21EE0Dh
▶ SHA1 signature[20]	611AEAB853E
uint file_size	134283264
uint header_size	1879048192
uint endian_tag	78563412h

Little Endian
(Normal)

ABNORMAL_HEADER_SIZE

Implies: weird, possibly hiding data after header before string table


/Users/cfenton/repos/security/resources/test/dex_characteristics/dexception-injection.test - 010 Editor

dexception-injection.test

```
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0  1   0123456789ABCDEF01
0000h: 64 65 78 0A 30 33 35 00 55 ED 4B F0 60 36 5E F5 FC 2F  dex.035.UíKð`6^õü/
0012h: 04 F3 6A 7F 14 A3 52 BD E9 4A F4 D0 3D 09 AB 48 00 00  .ój..£R½éJôÐ=.«H..
0024h: E0 1A 00 00 78 56 34 12 00 00 00 00 00 00 00 00 AC 26  à...xV4.........¬&
0036h: 00 00 E2 00 00 00 E0 1A 00 00 3D 00 00 00 68 1E 00 00  ..â...à...=...h...
0048h: 26 00 00 00 5C 1F 00 00 29 00 00 00 24 21 00 00 54 00  &...\...)...$!..T.
005Ah: 00 00 6C 22 00 00 0D 00 00 00 0C 25 00 00 FF 21 00 00  ..l"......%..ÿ!..
006Ch: AC 26 00 00 B5 B4 A9 DB E1 E2 E4 D1 1B 8D 19 74 D9 D0  ¬&..µ´©ÛáâäÑ...tÙÐ
007Eh: 03 5C 4A 9D 81 8D FC 5D 3C BC BE 49 FD 68 D8 FA 1C 83  .\J...ü]<¼¾IýhØú..
0090h: A1 CB D1 D1 A1 D1 D1 D1 A9 87 E5 C3 D1 D1 D1 D1 D1 D1  ¡ËÑÑ¡ÑÑÑ©‡åÃÑÑÑÑÑÑ
00A2h: D1 D1 71 C8 D1 D1 5C D1 D1 D1 A1 D1 D1 D1 F6 D1 D1 D1  ÑÑqÈÑÑ\ÑÑÑ¡ÑÑÑöÑÑÑ
```

Inspector

Type	Value
Signed Byte	-32
Unsigned Byte	224
Signed Short	6880
Unsigned Short	6880
Signed Int	6880
Unsigned Int	6880

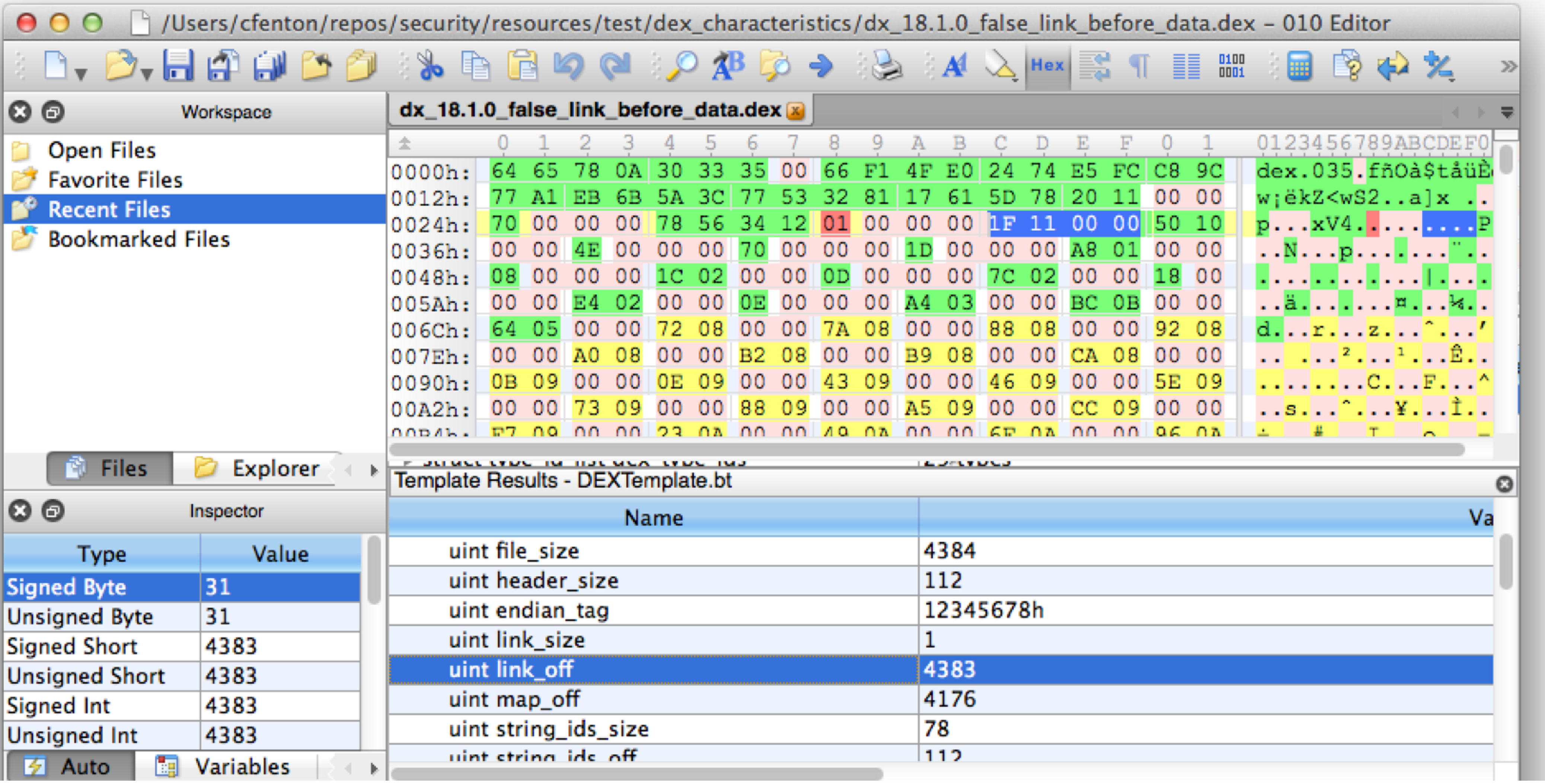
Template Results - DEXTemplate.bt

Name	Va
struct header_item dex_header	
struct dex_magic magic	dex 035
uint checksum	F04BED55h
SHA1 signature[20]	60365EF5FC2F04F36A7F14A352BDE94AF4D03D09
uint file_size	18603
uint header_size	6880
uint endian_tag	12345678h
uint link_size	0

header_size normally
0x70 (112) bytes

ABNORMAL_LINK_SECTION

- Implies: anti-decompiler



/Users/cfenton/repos/security/resources/test/dex_characteristics/dx_18.1.0_false_link_before_data.dex – 010 Editor

dx_18.1.0_false_link_before_data.dex

```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0  1   0123456789ABCDEF0
0000h: 64 65 78 0A 30 33 35 00 66 F1 4F E0 24 74 E5 FC C8 9C   dex.035.fñOà$tåüÈ
0012h: 77 A1 EB 6B 5A 3C 77 53 32 81 17 61 5D 78 20 11 00 00   w¡ëkZ<wS2..a]x ..
0024h: 70 00 00 00 78 56 34 12 01 00 00 00 1F 11 00 00 50 10   p...xV4.........P
0036h: 00 00 4E 00 00 00 70 00 00 00 1D 00 00 00 A8 01 00 00   ..N...p.......¨..
0048h: 08 00 00 00 1C 02 00 00 0D 00 00 00 7C 02 00 00 18 00   ............|....
005Ah: 00 00 E4 02 00 00 0E 00 00 00 A4 03 00 00 BC 0B 00 00   ..ä.......¤...¼..
006Ch: 64 05 00 00 72 08 00 00 7A 08 00 00 88 08 00 00 92 08   d...r...z...^...’
007Eh: 00 00 A0 08 00 00 B2 08 00 00 B9 08 00 00 CA 08 00 00   .. ...².......Ê..
0090h: 0B 09 00 00 0E 09 00 00 43 09 00 00 46 09 00 00 5E 09   ........C...F...^
00A2h: 00 00 73 09 00 00 88 09 00 00 A5 09 00 00 CC 09 00 00   ..s...^...¥...Ì..
```

Template Results - DEXTemplate.bt

Name	Va
uint file_size	4384
uint header_size	112
uint endian_tag	12345678h
uint link_size	1
uint link_off	4383
uint map_off	4176
uint string_ids_size	78

Inspector

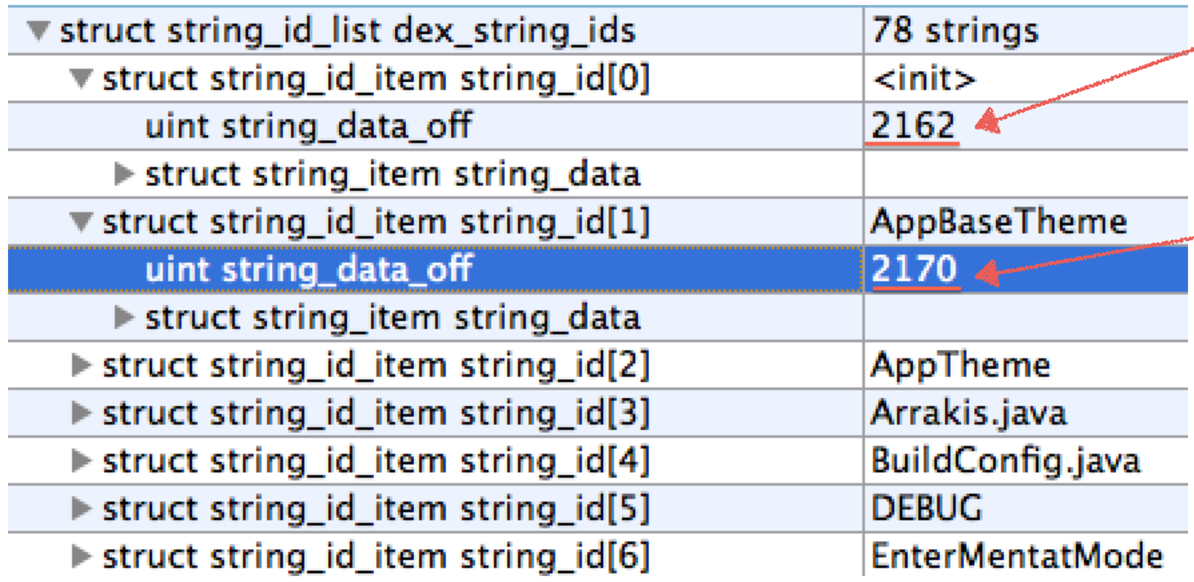
Type	Value
Signed Byte	31
Unsigned Byte	31
Signed Short	4383
Unsigned Short	4383
Signed Int	4383
Unsigned Int	4383

link_offset and size always **0** in DEX files

ABNORMAL_STRING_SORT

- Implies: dexlib 1.x

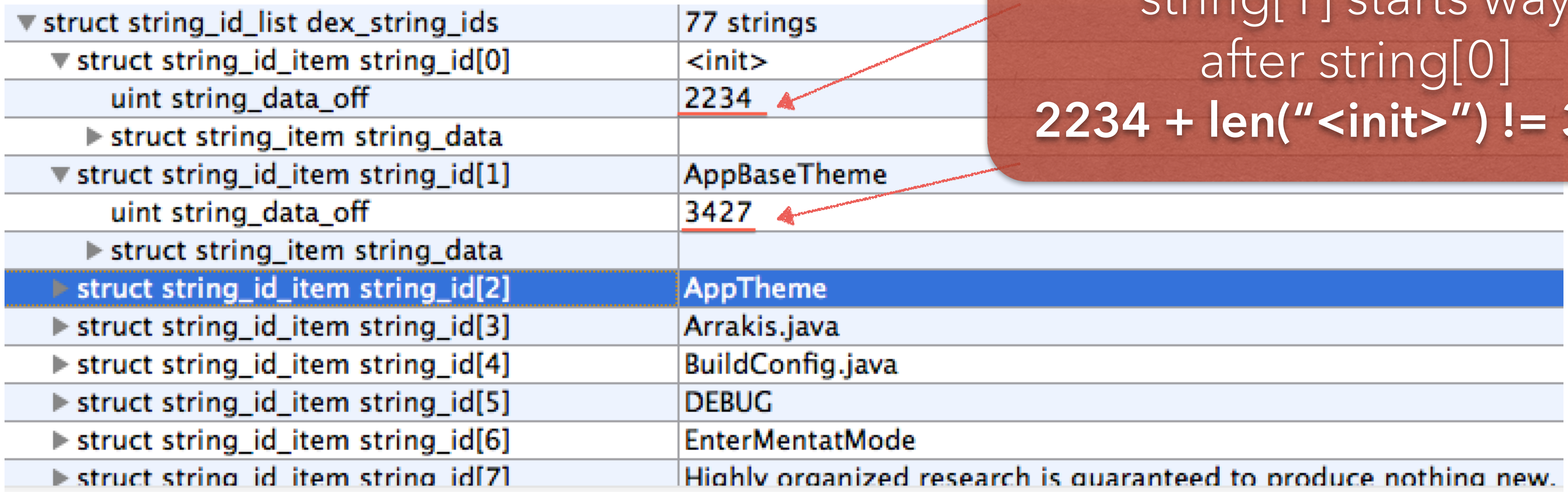
Normal



▼ struct string_id_list dex_string_ids	78 strings
▼ struct string_id_item string_id[0]	<init>
uint string_data_off	2162
▶ struct string_item string_data	
▼ struct string_id_item string_id[1]	AppBaseTheme
uint string_data_off	2170
▶ struct string_item string_data	
▶ struct string_id_item string_id[2]	AppTheme
▶ struct string_id_item string_id[3]	Arrakis.java
▶ struct string_id_item string_id[4]	BuildConfig.java
▶ struct string_id_item string_id[5]	DEBUG
▶ struct string_id_item string_id[6]	EnterMentatMode

string[0] starts @2162
string[1] starts immediately after string[0]

Abnormal



▼ struct string_id_list dex_string_ids	77 strings
▼ struct string_id_item string_id[0]	<init>
uint string_data_off	2234
▶ struct string_item string_data	
▼ struct string_id_item string_id[1]	AppBaseTheme
uint string_data_off	3427
▶ struct string_item string_data	
▶ struct string_id_item string_id[2]	AppTheme
▶ struct string_id_item string_id[3]	Arrakis.java
▶ struct string_id_item string_id[4]	BuildConfig.java
▶ struct string_id_item string_id[5]	DEBUG
▶ struct string_id_item string_id[6]	EnterMentatMode
▶ struct string_id_item string_id[7]	Highly organized research is guaranteed to produce nothing new.

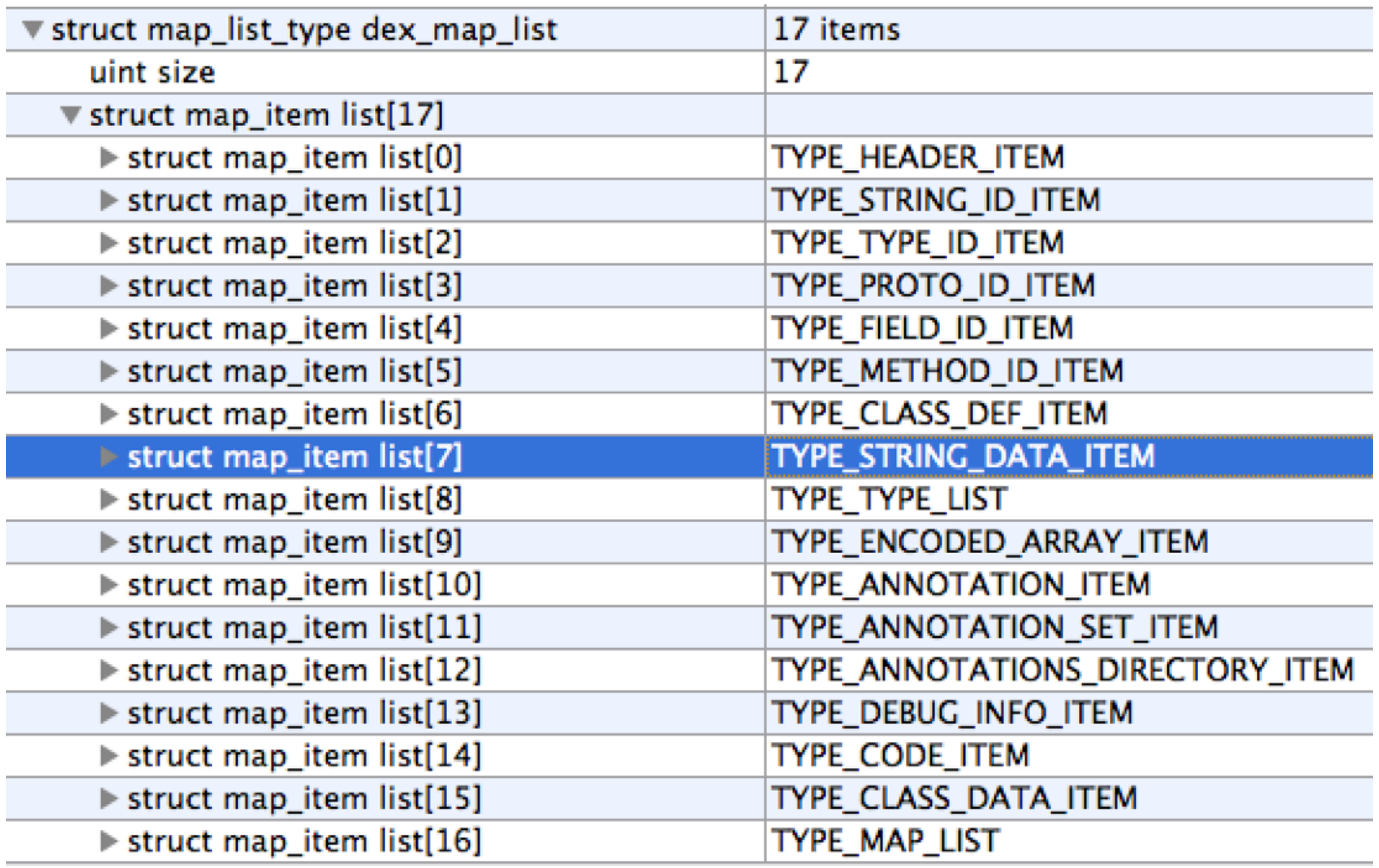
string[1] starts way after string[0]
2234 + len("<init>") != 3427

ABNORMAL_TYPE_ORDER

- Implies: something other than dx or dexmerge

dx Map Item Order

1. HEADER_ITEM
2. STRING_ID_ITEM
3. TYPE_ID_ITEM
4. PROTO_ID_ITEM
5. FIELD_ID_ITEM
6. METHOD_ID_ITEM
7. CLASS_DEF_ITEM
8. ANNOTATION_SET_REF_LIST
9. ANNOTATION_SET_ITEM
10. CODE_ITEM
11. ANNOTATIONS_DIRECTORY_ITEM
12. TYPE_LIST
13. STRING_DATA_ITEM
14. DEBUG_INFO_ITEM
15. ANNOTATION_ITEM
16. ENCODED_ARRAY_ITEM
17. CLASS_DATA_ITEM
18. MAP_LIST



▼ struct map_list_type dex_map_list	17 items
uint size	17
▼ struct map_item list[17]	
▶ struct map_item list[0]	TYPE_HEADER_ITEM
▶ struct map_item list[1]	TYPE_STRING_ID_ITEM
▶ struct map_item list[2]	TYPE_TYPE_ID_ITEM
▶ struct map_item list[3]	TYPE_PROTO_ID_ITEM
▶ struct map_item list[4]	TYPE_FIELD_ID_ITEM
▶ struct map_item list[5]	TYPE_METHOD_ID_ITEM
▶ struct map_item list[6]	TYPE_CLASS_DEF_ITEM
▶ struct map_item list[7]	TYPE_STRING_DATA_ITEM
▶ struct map_item list[8]	TYPE_TYPE_LIST
▶ struct map_item list[9]	TYPE_ENCODED_ARRAY_ITEM
▶ struct map_item list[10]	TYPE_ANNOTATION_ITEM
▶ struct map_item list[11]	TYPE_ANNOTATION_SET_ITEM
▶ struct map_item list[12]	TYPE_ANNOTATIONS_DIRECTORY_ITEM
▶ struct map_item list[13]	TYPE_DEBUG_INFO_ITEM
▶ struct map_item list[14]	TYPE_CODE_ITEM
▶ struct map_item list[15]	TYPE_CLASS_DATA_ITEM
▶ struct map_item list[16]	TYPE_MAP_LIST

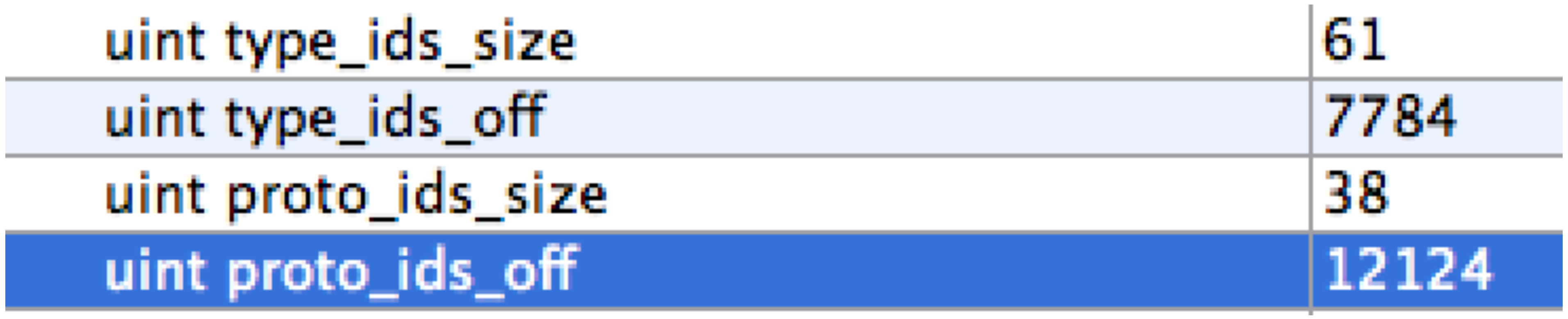
not made with **dx** or **dexmerge,** probably **dexlib** because TYPE_STRING_DATA_ITEM comes after TYPE_CLASS_DEF_ITEM

dexmerge Map Item Order

1. HEADER_ITEM
2. STRING_ID_ITEM
3. TYPE_ID_ITEM
4. PROTO_ID_ITEM
5. FIELD_ID_ITEM
6. METHOD_ID_ITEM
7. CLASS_DEF_ITEM
8. MAP_LIST
9. TYPE_LIST
10. ANNOTATION_SET_REF_LIST
11. ANNOTATION_SET_ITEM
12. CLASS_DATA_ITEM
13. CODE_ITEM
14. STRING_DATA_ITEM
15. DEBUG_INFO_ITEM
16. ANNOTATION_ITEM
17. ENCODED_ARRAY_ITEM
18. ANNOTATIONS_DIRECTORY_ITEM

NON_CONTIGUOUS_SECTION

- Implies: weird, maybe dexmerge



uint type_ids_size	61
uint type_ids_off	7784
uint proto_ids_size	38
uint proto_ids_off	12124

proto_ids should come after type_ids

type_id_item size = 4 bytes
type_ids_size * 4 = 244
type_ids_off + 244 = 8028
proto_ids *actually* starts 12124! weird!

MALWARE AND PIRACY DETECTION

caleb

REDNAGA

THE QUESTION

Three main compilers:

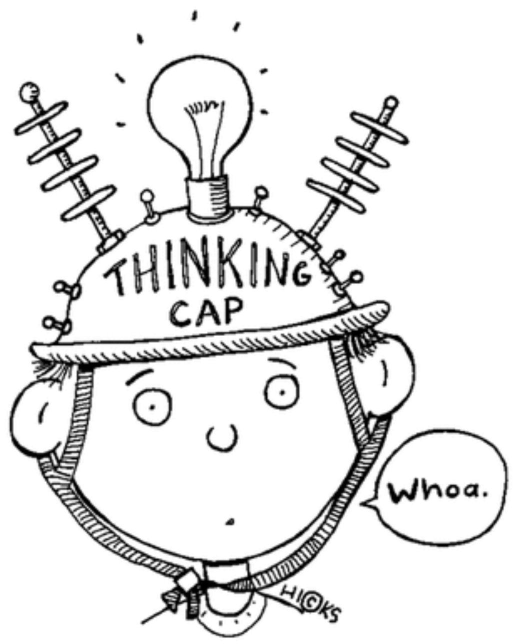
1. dx ← Java .class files (source code)
2. dexmerge ← Not used manually, only by IDEs (source code)
3. smali (dexlib) ← DEX files (**not source code**)

Why would a legitimate developer ever need to use smali?

They have the source.

THE HYPOTHESIS

- If app compiled with dexlib, probably tampered
- If tampered, probably was not the developer
- Tampered apps are likely either:
  - 🔓 pirated / cracked
  - 💀 malware



∴ ***app is tampered -> app is interesting*** 🔎

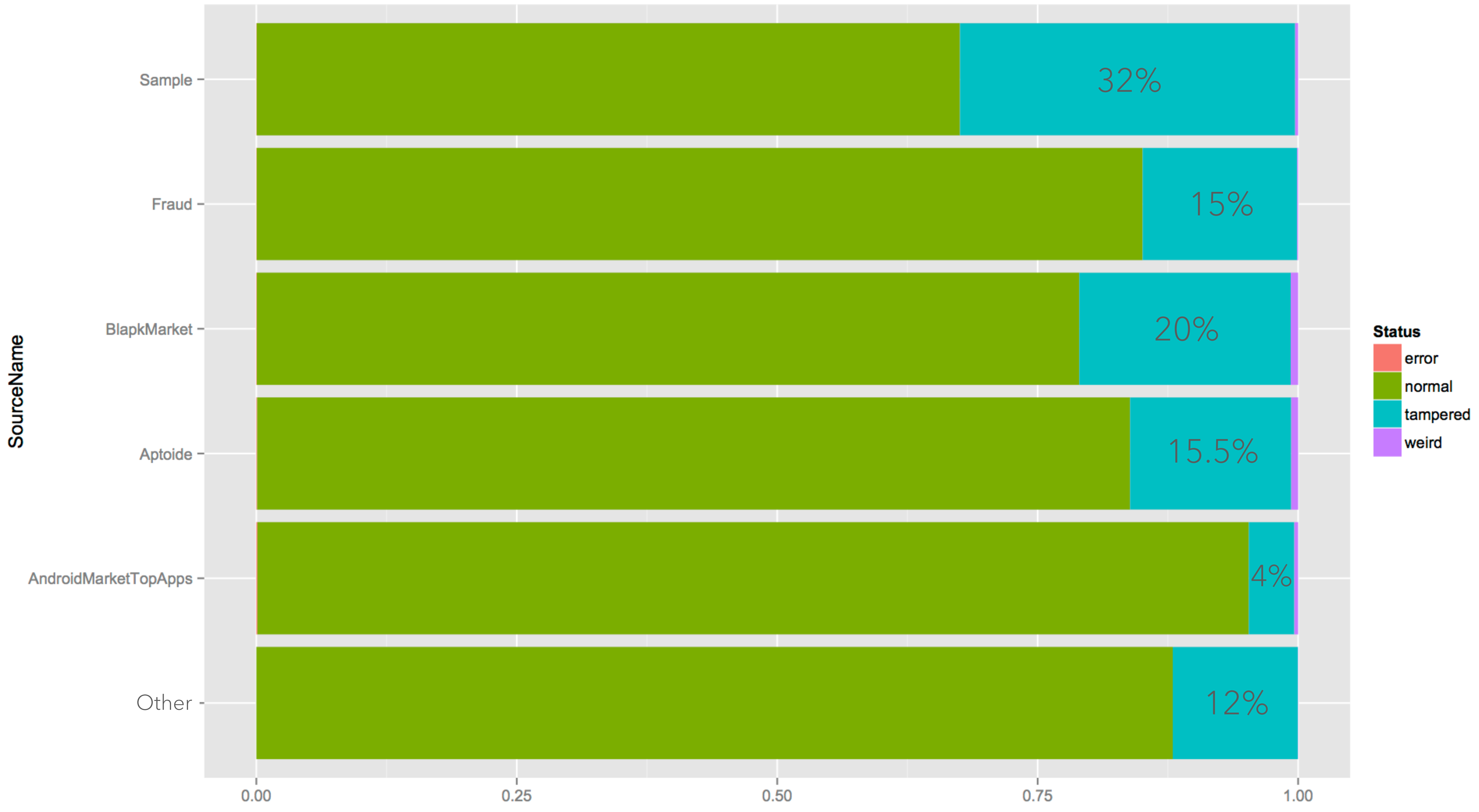
SAMPLE SET

- 20,000 APKs from each market
  - Top Play Apps, Aptoide, BlapkMarket, etc.
- 10,000 highest scoring “fraudulent” apps
  - Scored by experimental statical model
  - Fraud may just mean modified XML (not DEX)
- Up to 10 APKs per variant of all malware families

THE METHOD

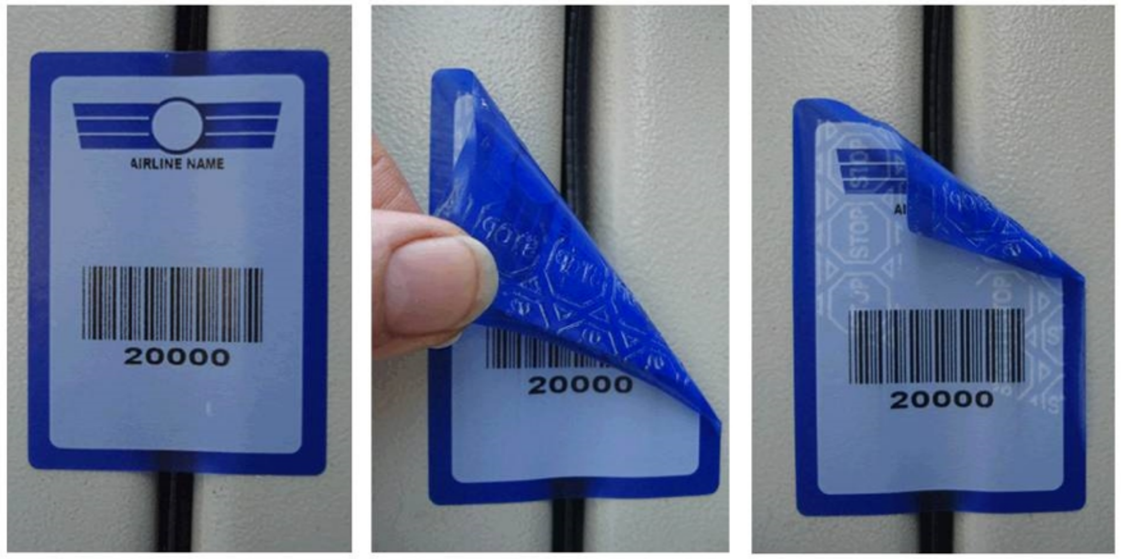
- Scanned the DEX of each APK
- Did not scan AXML files
- Tampered means:
    - abnormal string sort, class path, type order
- Weird means:
    - abnormal endian magic, header size, type descriptor, class path

RESULTS: SOURCE TAMPERING



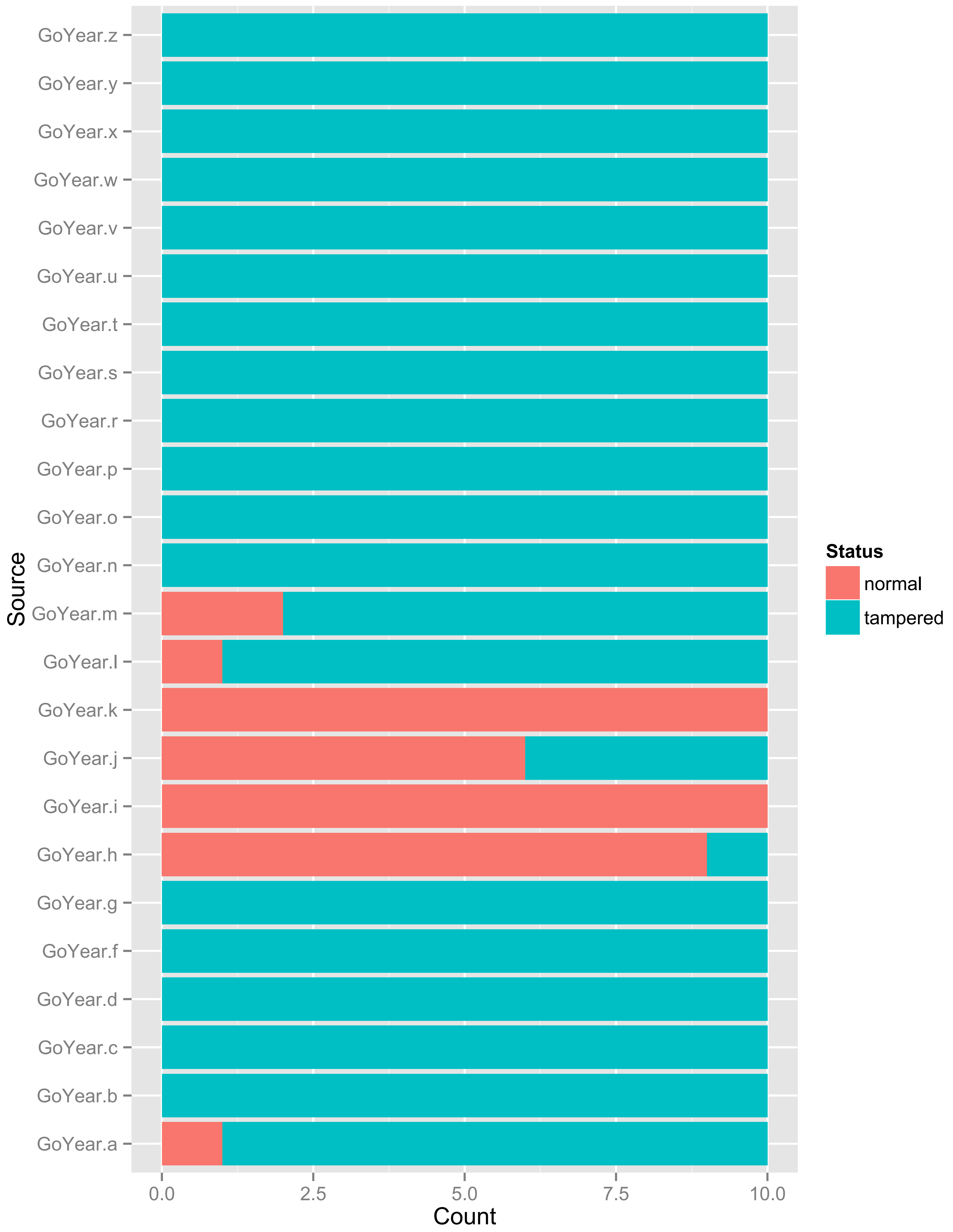
RESULTS: MALWARE TAMPERING

- 756 malware families (many variants each)
- 17508 malicious APKs
- 50% families have some tampering
- 50% families have *no* tampering
- 85 families are 100% tampered
- Each family has a tampering *profile*

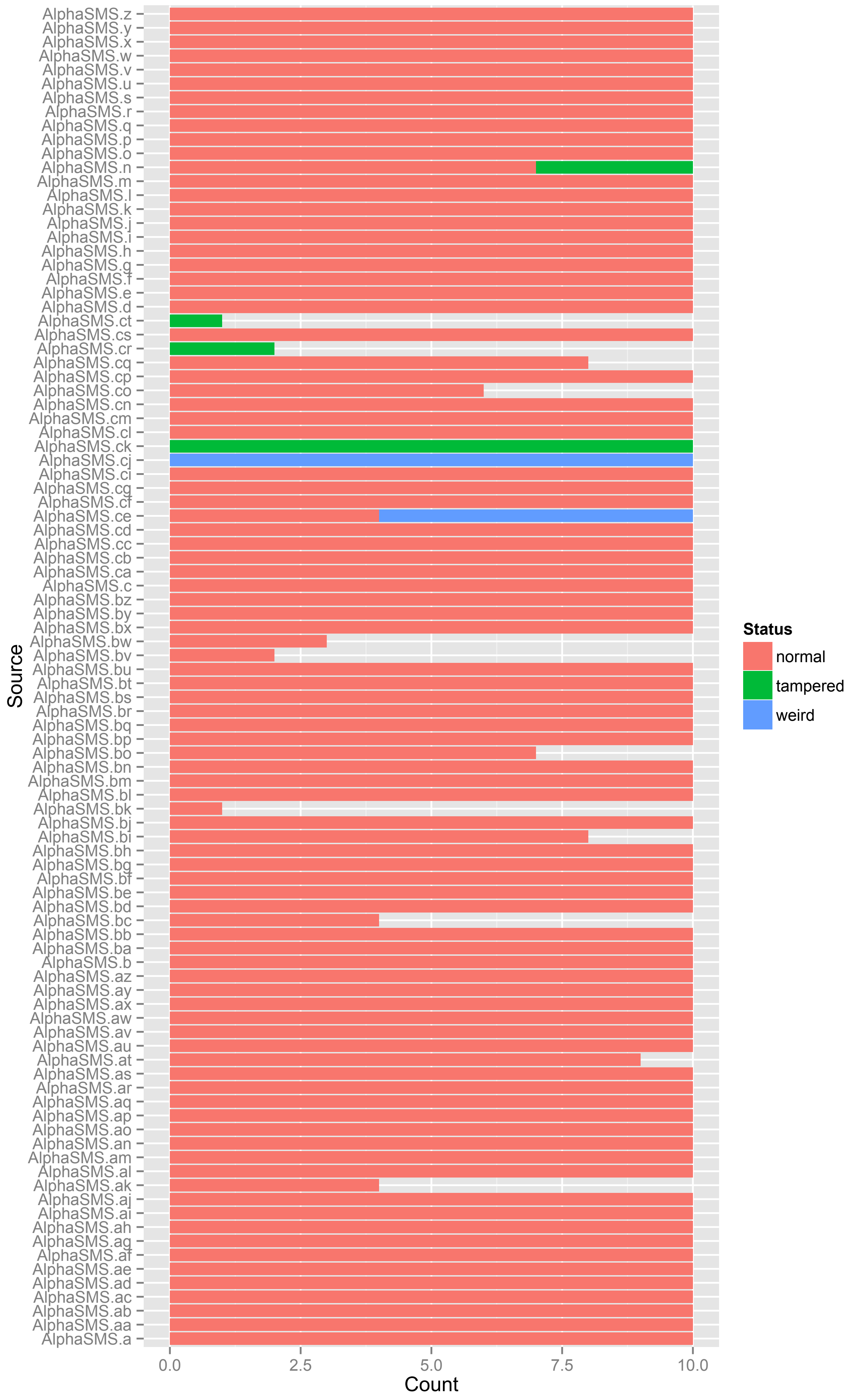


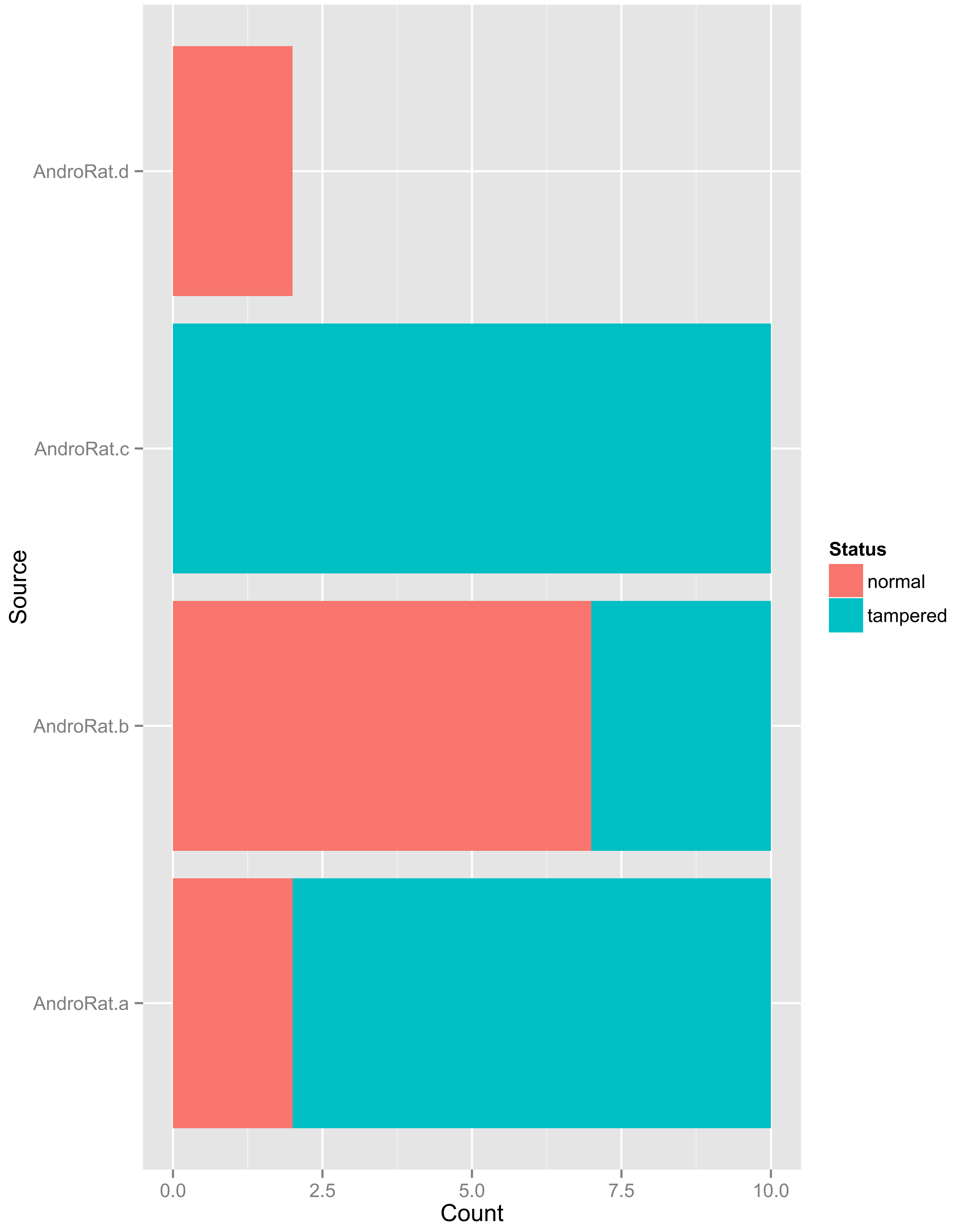
100% TAMPERED FAMILIES

AdultFreedom Alsalah AncientSMSThief AppleService AvariceYY BadSerial BadSub Badaccents BankMirage Bgserv BiggBoss CastilStyle CataChar CnSky CoinKrypt DeCerTasks DidStall DirtyAir DoubleZero EasyPine EdeFraud EmmentalCrupt ErrthangSms Euroxbox ExplicitHorse FadeSMS FakeActivate FakeKakao FastUninstallRepackaged FineFocusAds FlaccidForest GauerCloud Geinimi GoGuangGo Gone60 ImAdPush KHSms Kakabet KhpowSms KrabBot Krysanec LidLocker LoveMii MMarketPay MaClickFraud MirvspySMS MixedSmoke MmsMore MocheYY Moghava Obad OccupyYourPrivacy PVAFraud PhoXinhSms PicSysCom PirateShame PlusTV PopTest RootSmart RuSms Samsapo SandroRat SimpleTemai SixPointFourSMS SlyInstall Smsmonitur SneakyBeeSMS Stask Stoqx StorgeSMS SwfScam SwiftLogger Taotobo Tornika UniversalAndrootRepackaged VDLoader Vchargelet VideoBoss VservSubscription WinAdOffers WrongPath XSider Xybot YobaSMS ZxtdPay



Source
GoYear.z
GoYear.y
GoYear.x
GoYear.w
GoYear.v
GoYear.u
GoYear.t
GoYear.s
GoYear.r
GoYear.p
GoYear.o
GoYear.n
GoYear.m
GoYear.l
GoYear.k
GoYear.j
GoYear.i
GoYear.h
GoYear.g
GoYear.f
GoYear.d
GoYear.c
GoYear.b
GoYear.a
0.0
2.5
5.0
7.5
10.0
Count
Status
normal
tampered


AlphaSMS.z
AlphaSMS.y
AlphaSMS.x
AlphaSMS.w
AlphaSMS.v
AlphaSMS.u
AlphaSMS.s
AlphaSMS.r
AlphaSMS.q
AlphaSMS.p
AlphaSMS.o
AlphaSMS.n
AlphaSMS.m
AlphaSMS.l
AlphaSMS.k
AlphaSMS.j
AlphaSMS.i
AlphaSMS.h
AlphaSMS.g
AlphaSMS.f
AlphaSMS.e
AlphaSMS.d
AlphaSMS.ct
AlphaSMS.cs
AlphaSMS.cr
AlphaSMS.cq
AlphaSMS.cp
AlphaSMS.co
AlphaSMS.cn
AlphaSMS.cm
AlphaSMS.cl
AlphaSMS.ck
AlphaSMS.cj
AlphaSMS.ci
AlphaSMS.cg
AlphaSMS.cf
AlphaSMS.ce
AlphaSMS.cd
AlphaSMS.cc
AlphaSMS.cb
AlphaSMS.ca
AlphaSMS.c
AlphaSMS.bz
AlphaSMS.by
AlphaSMS.bx
AlphaSMS.bw
AlphaSMS.bv
AlphaSMS.bu
AlphaSMS.bt
AlphaSMS.bs
AlphaSMS.br
AlphaSMS.bq
AlphaSMS.bp
AlphaSMS.bo
AlphaSMS.bn
AlphaSMS.bm
AlphaSMS.bl
AlphaSMS.bk
AlphaSMS.bj
AlphaSMS.bi
AlphaSMS.bh
AlphaSMS.bg
AlphaSMS.bf
AlphaSMS.be
AlphaSMS.bd
AlphaSMS.bc
AlphaSMS.bb
AlphaSMS.ba
AlphaSMS.b
AlphaSMS.az
AlphaSMS.ay
AlphaSMS.ax
AlphaSMS.aw
AlphaSMS.av
AlphaSMS.au
AlphaSMS.at
AlphaSMS.as
AlphaSMS.ar
AlphaSMS.aq
AlphaSMS.ap
AlphaSMS.ao
AlphaSMS.an
AlphaSMS.am
AlphaSMS.al
AlphaSMS.ak
AlphaSMS.aj
AlphaSMS.ai
AlphaSMS.ah
AlphaSMS.ag
AlphaSMS.af
AlphaSMS.ae
AlphaSMS.ad
AlphaSMS.ac
AlphaSMS.ab
AlphaSMS.aa
AlphaSMS.a
Source
0.0
2.5
5.0
7.5
10.0
Count
Status
normal
tampered
weird


AndroRat.d
AndroRat.c
AndroRat.b
AndroRat.a
Source
0.0
2.5
5.0
7.5
10.0
Count
Status
normal
tampered

CONCLUSION

- Few legitimate apps are tampered
- Tampering good signal for malware / piracy
- Better able to understand malware family evolution

APKID DEMO

REDNAGA

EXTENDED READING

https://github.com/rednaga/training

http://www.strazzere.com/papers/DexEducation-PracticingSafeDex.pdf

https://github.com/strazzere/anti-emulator/tree/master/slides

https://github.com/strazzere/android-unpacker/blob/master/AHPL0.pdf

http://www.droidsec.org/wiki/#whitepapers

http://calebfenton.github.io/

http://androidcracking.blogspot.com/

REDNAGA

THANKS!

TIM “DIFF” STRAZZERE
@TIMSTRAZZ

CALEB FENTON
@CALEB_FENTON

Special Thanks for Jacob Soo and Mikachu for all your assistance!

Join use on Freenode on #droidsec and #rednaga

Good people to follow on Twitter for
Android /reversing /malware / hacking information:

@_jsoo_ @droidsec @jcase @marcwrogers @moong1ider @msolnik
@osxreverser @PatrickMcCanna @rotlogix @snare @tamakikusu @trimosx
#MalwareMustDie

07.22.2016

HITCON COMMUNITY

REDNAGA