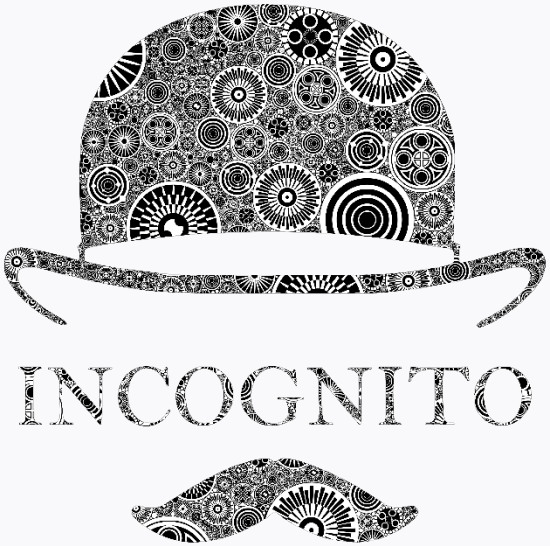
Android Deep DEX Analysis Technique

세종대학교 SSG

김남준



In this session..

- How Android translate DEX file format inside APK file
- Deep Analysis of DEX file (class, method recovery)
- Translate DEX bytecode
- How we extract information from malicious DEX file
- Apply to Real World (smishing.kr)

About Speaker

- 김남준 (20)

  세종대학교 정보보호학과 15 (고3 탈출!)

  + 작년 고등학생 인코 발표자

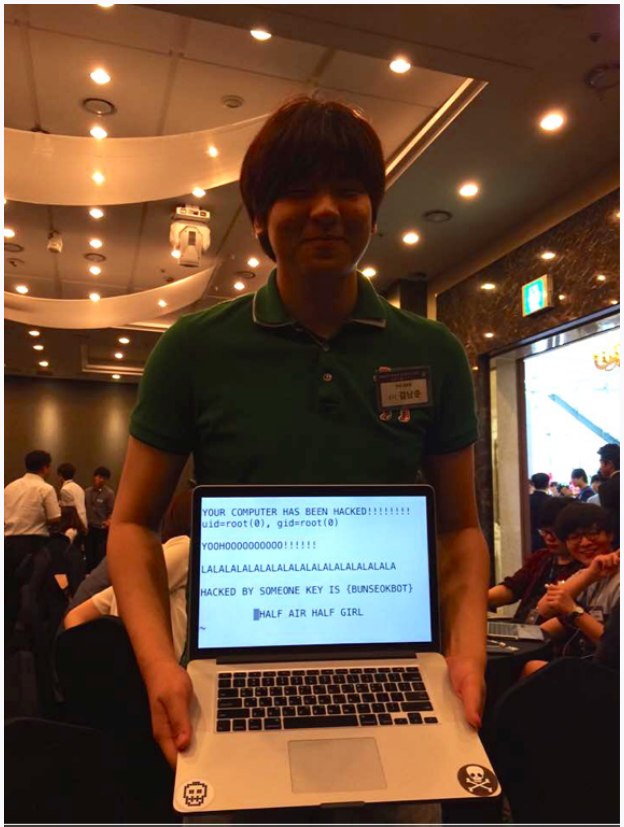
  Newbie of Sejong Security Group

  UpRoot Researcher

  KITRI Best of the Best 3기

  여친구함

E-mail : admin@smishing.kr
Blog : blog.smishing.kr
Web : smishing.kr



Special Thanks to @공기반여자반
Pictured by @유정빈

What is DEX file

- Dalvik Executable File Format
- Specially designed for Embedded System
- It can run on DVM inside Android
- Actually, DVM is different between JVM

Standard Android Translation Process

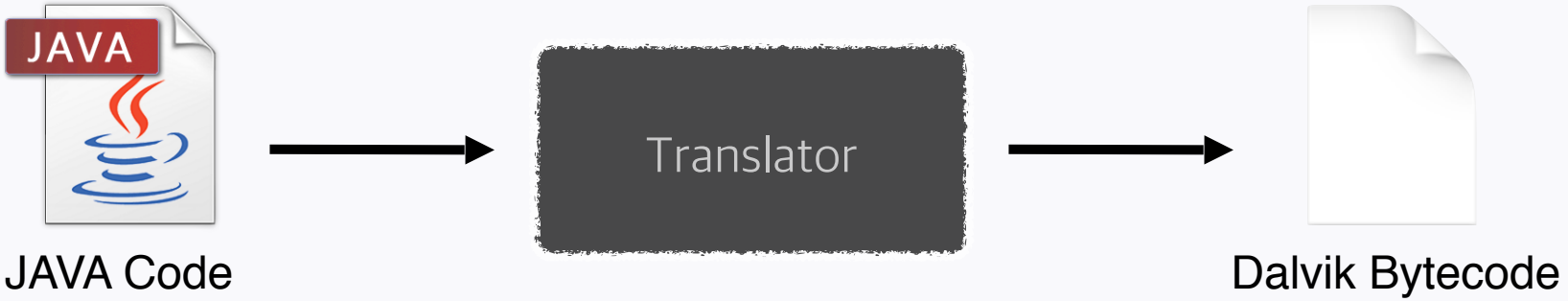
```
public class MainActivity.. {
    public static void onCreate(s..) {
        super.onCreate(savedInsta..)
        ..
        public string TAG = “S.S.G”;
        Log.i(TAG, “sejong S.S.G”);
    }
}
```

JAVA Code

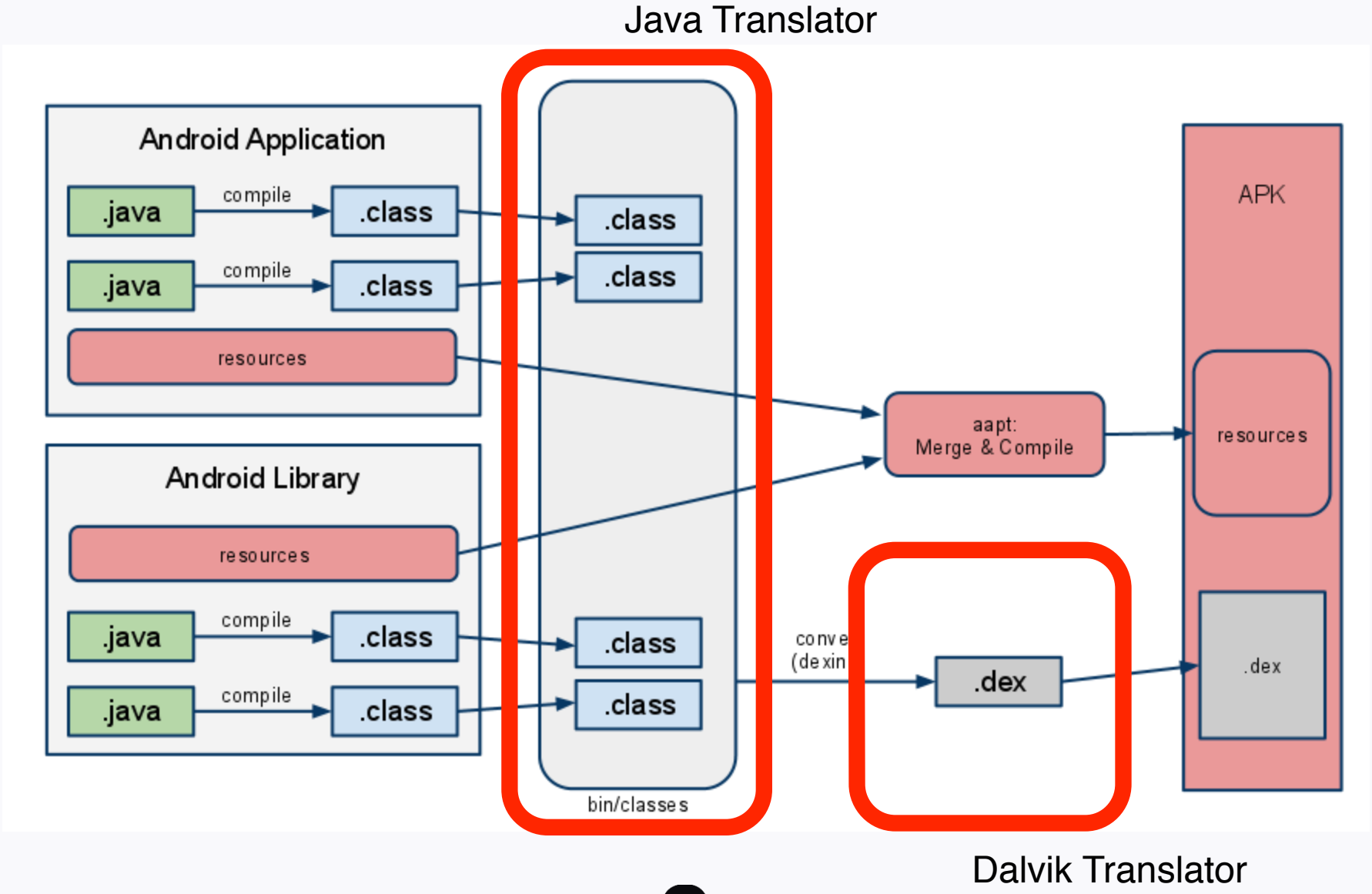
```
.class public Lbunseokbot/MainActivity
.source MainActivity.java

.method public static onCreate(..)V
const-string v0, “S.S.G”
const-string v1, “sejong S.S.G”
invoke-static {v0, v1}, L/android/util/Log ->
i(Ljava/lang/String;Ljava/lang/String;)I
return-void
```

Dalvik bytecode

Building Android Application



Translator

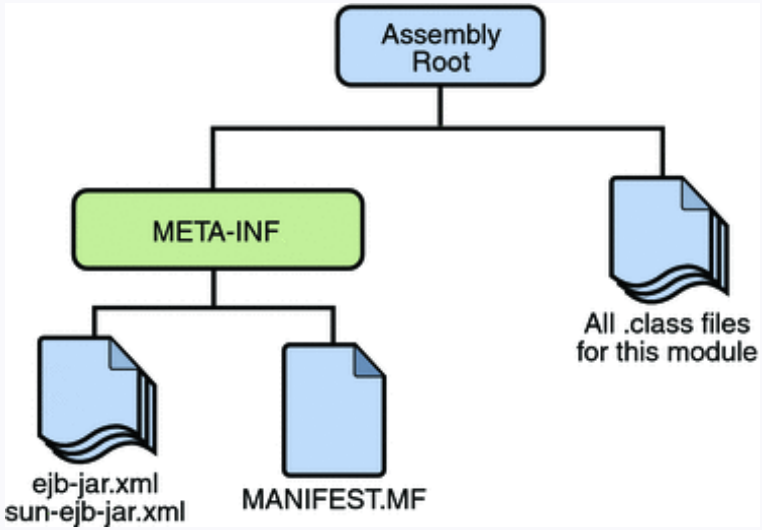

Java Translator
Android Application
.java
compile
.class
.java
compile
.class
resources
Android Library
resources
.java
compile
.class
.java
compile
.class
.class
.class
.class
.class
bin/classes
aapt:
Merge & Compile
conve
(dexin
.dex
APK
resources
.dex
Dalvik Translator

Java Language

- Object-Oriented Programming Language
- Easy to **“DECOMPILE”**
- Running on Java Virtual Machine as “JAR” File
- Actually, JVM is different between DVM
- use for “native” (NOT JNI) Android Programming

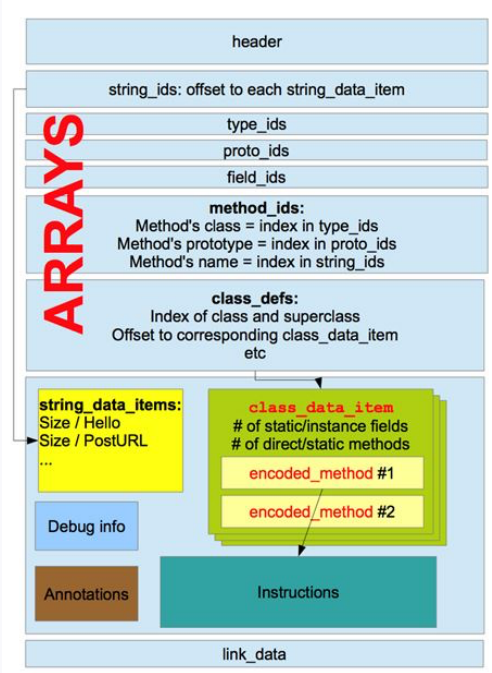
Different between JAR and DEX

- Java Archive File -> Package File Format
- Java class files + metadata + resources
- each class have independent class file
- “Class Level File Management”



Different between JAR and DEX

- Dalvik Execution File
- Single DEX file
- specified for Embedded..? (Mobile, PDA..?)



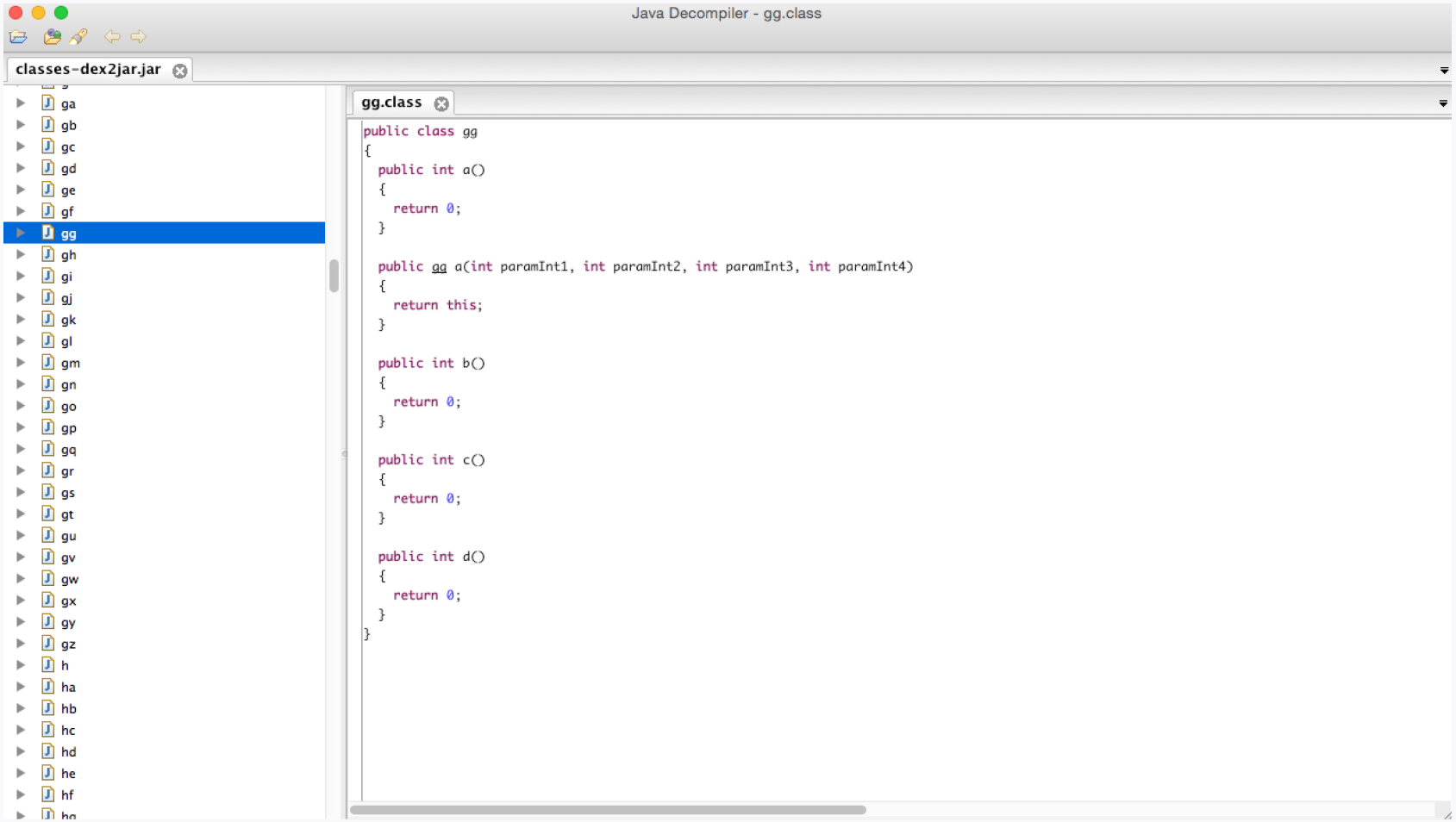
Different between JVM and DVM

Dalvik Virtual Machine	Java Virtual Machine
Register Based	Stack Based
Constant Pool per **Application**	Constant Pool per **Class**
Supply Just-In Time Execution	

We have to understand and Translate

DEX File as different technique

Normally.. we use “dex2jar”



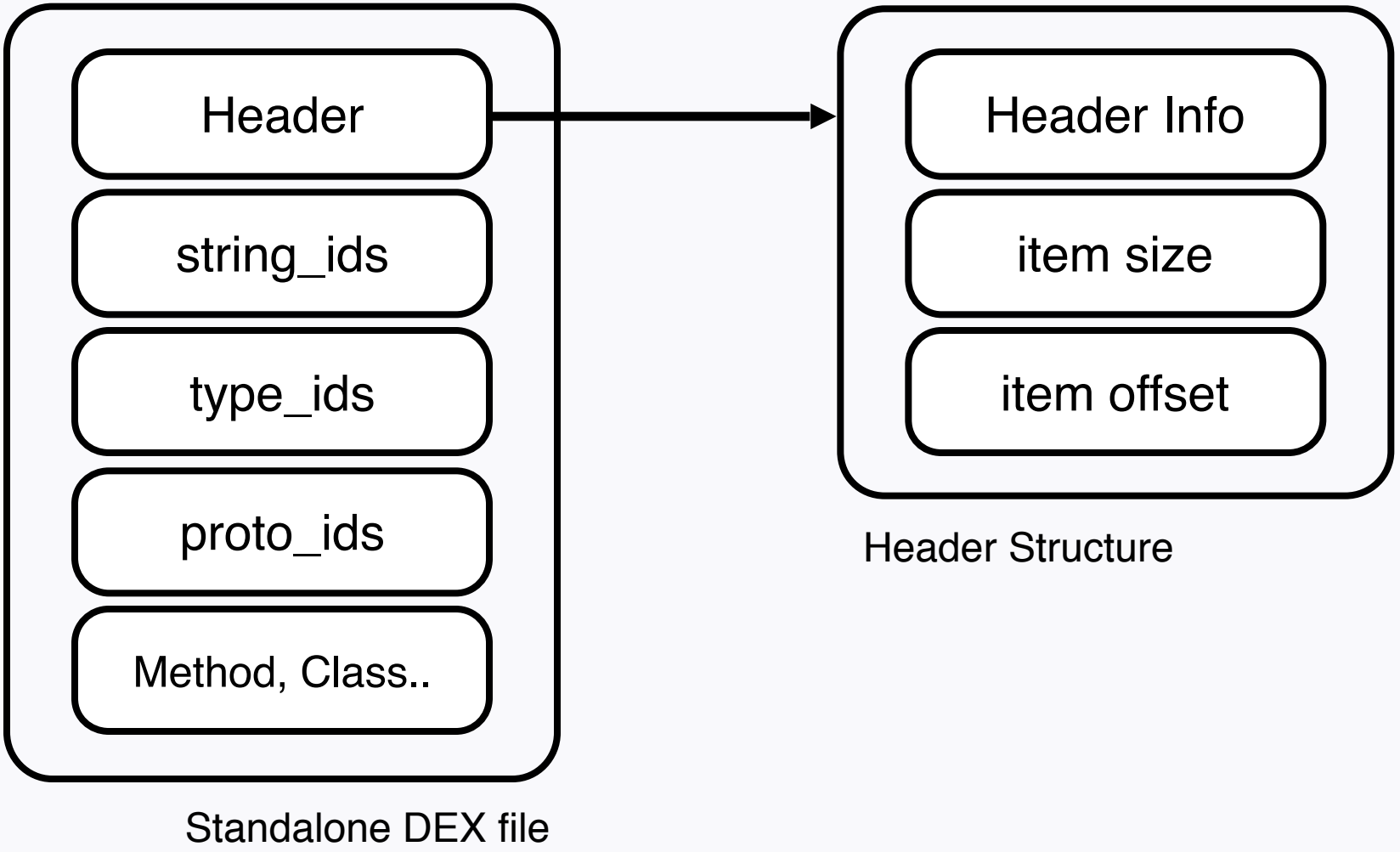
or.. baksmali!

```
mac@bunseokbot-Macbook ~/Desktop/Tools/smali/out/com/example/google/service cat Repeater.smali
.class public Lcom/example/google/service/Repeater;
.super Ljava/lang/Object;
.source "Repeater.java"


# direct methods
.method public constructor <init>()V
    .registers 1

    .prologue
    .line 9
    invoke-direct {p0}, Ljava/lang/Object;-><init>()V

    return-void
.end method

.method public static sendUpdateBroadcastRepeat(Landroid
    .registers 9
    .param p0, "ctx"    # Landroid/content/Context;

    .prologue
    const/4 v4, 0x0

    .line 11
    new-instance v7, Landroid/content/Intent;

    const-class v1, Lcom/example/google/service/TaskRequ

    invoke-direct {v7, p0, v1}, Landroid/content/Intent;

    .line
    .local 7, "intent":Landroid/content/Intent;
    invoke-static {p0, v4, v7, v4}, Landroid/app/Pending
;I)Landroid/app/PendingIntent;

    move-result-object v6

    .line 15
    .local v6, "pendingIntent":Landroid/app/PendingInten
```

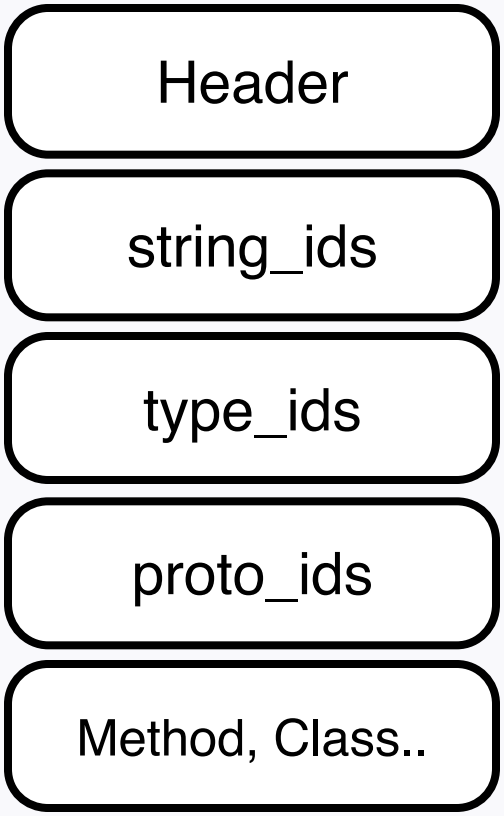
```
mac@bunseokbot-Macbook ~/Desktop/Tools/smali/out/com/example/google/service cat Contacts.smali
.class public Lcom/example/google/service/Contacts;
.super Ljava/lang/Object;
.source "Contacts.java"


# instance fields
.field private _Context:Landroid/content/Context;


# direct methods
.method public constructor <init>(Landroid/content/Context;)V
    .registers 2
    .param p1, "context"    # Landroid/content/Context;

    .prologue
    .line
    invoke-direct {p0}, Ljava/lang/Object;-><init>()V

    .line 11
    iput-object p1, p0, Lcom/example/google/service/Contacts;->_Context:Landroid/content/Context;

    .line 12
    return-void
.end method


# virtual methods
.method public ForwardContacts()V
    .registers 10

    .prologue
    .line 15
    new-instance v0, Lcom/example/google/service/ContactsHelper;

    iget-object v6, p0, Lcom/example/google/service/Contacts;->_Context:Landroid/content/Context;

    invoke-direct {v0, v6}, Lcom/example/google/service/ContactsHelper;-><init>(Landroid/content/Context;)V

    .line 16
```

How baksmali translate it?

inside DEX file



inside DEX file



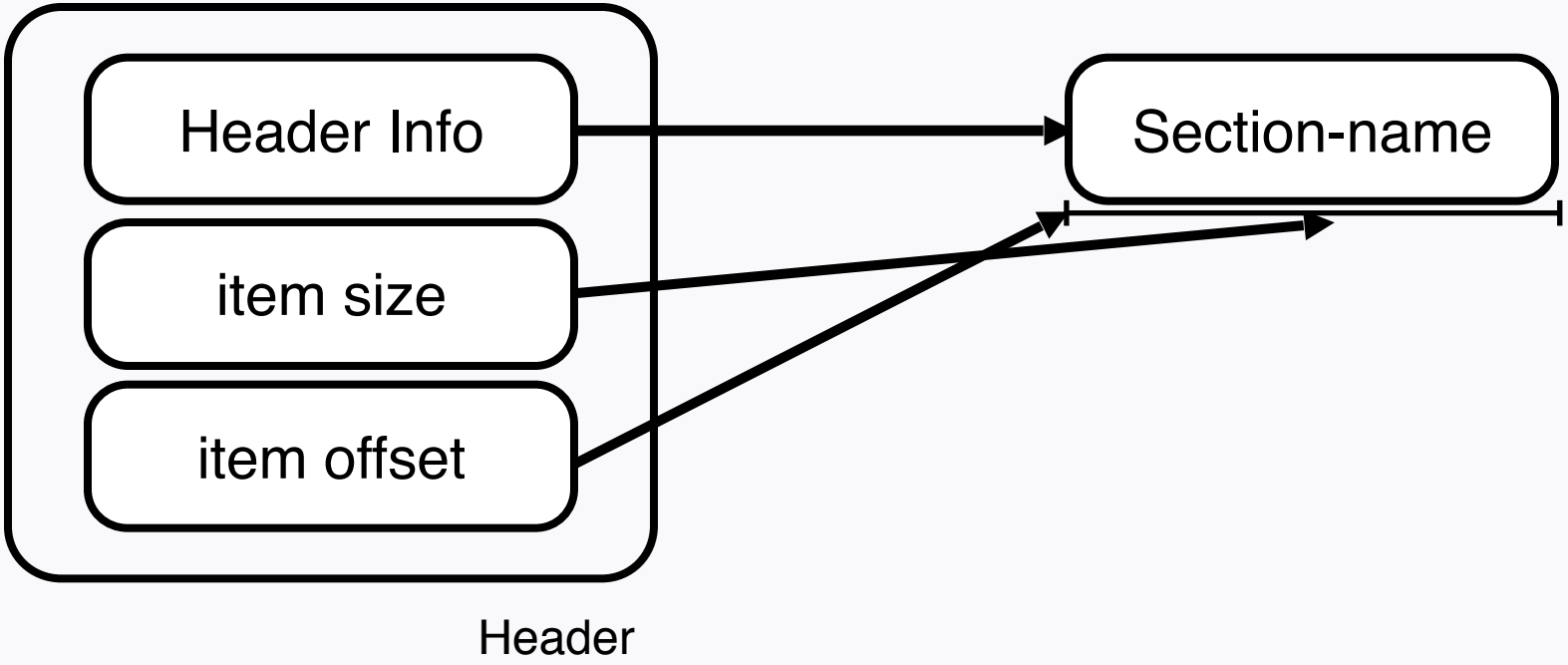
String where code contains **(only Address)**

Class, Method type container **(only Address)**

Class, Method parameter return info **(only Address)**

real Method, Class, Field, Data container

inside DEX file



Now, We analyze DEX file

```
public class MyActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Log.i("S.S.G", "DEX File Analyze Testing");

    }
}
```

```
Log/i S.S.G DEX File Analyze Testing
```

How to Analysis DEX file

- DEX Parser for Python v0.0.1
- https://github.com/bunseokbot/dexparser
- pip install dexparser

1. DEX Header

```
from dexparser import Dexparser
import sys

FILE_DIR = sys.argv[1]

d = Dexparser(FILE_DIR)
print d.header_info()
```

```
{'type_ids_size': 896, 'string_ids_off': 112, 'file_size': 718444, 'type_ids_off': 2
5276, 'field_ids_off': 42504, 'data_off': 116384, 'method_ids_off': 53776, 'data_siz
e': 602060, 'map_off': 116384, 'field_ids_size': 1409, 'method_ids_size': 5458, 'pro
to_ids_off': 28860, 'header_size': 112, 'signiture': '\xda\xcf\x13\'\x1c\xa2\xbc*\x9
0K\xfe\xf1\x894\xddp\xcd"\xc5M', 'endian_tag': 305419896, 'string_ids_size': 6291, '
magic': 'dex\n035\x00', 'link_size': 0, 'checksum': 1181462038, 'link_off': 0, 'clas
s_defs_off': 97440, 'class_defs_size': 534, 'proto_ids_size': 1137}
```

1. DEX Header


```
{'type_ids_size': 896, 'string_ids_off': 112, 'file_size': 718444, 'type_ids_off': 2
5276, 'field_ids_off': 42504, 'data_off': 116384, 'method_ids_off': 53776, 'data_siz
e': 602060, 'map_off': 116384, 'field_ids_size': 1409, 'method_ids_size': 5458, 'pro
to_ids_off': 28860, 'header_size': 112, 'signiture': '\xda\xcf\x13\'\x1c\xa2\xbc*\x9
0K\xfe\xf1\x894\xddp\xcd"\xc5M', 'endian_tag': 305419896, 'string_ids_size': 6291, '
magic': 'dex\n035\x00', 'link_size': 0, 'checksum': 1181462038, 'link_off': 0, 'clas
s_defs_off': 97440, 'class_defs_size': 534, 'proto_ids_size': 1137}
```


d.header_info() -> Dictionary Type
d.header_info()[‘string_ids_size’]
d.header_info()[‘string_ids_off’]

```
string_ids SIZE 6291
string_ids OFFSET 112
```

2. Parse string_ids section

- 1. extract string_ids section offset, size
- 2. parse string_data offset for access string_data_item
- 3. translate to Language from string_data_item

2. Parse string_ids section

string_id_item

appears in the string_ids section

alignment: 4 bytes

Name	Format	Description
string_data_off	uint	offset from the start of the file to the string data for this item. The offset should be to a location in the `data` section, and the data should be in the format specified by "`string_data_item`" below. There is no alignment requirement for the offset.

2. Parse string_ids section

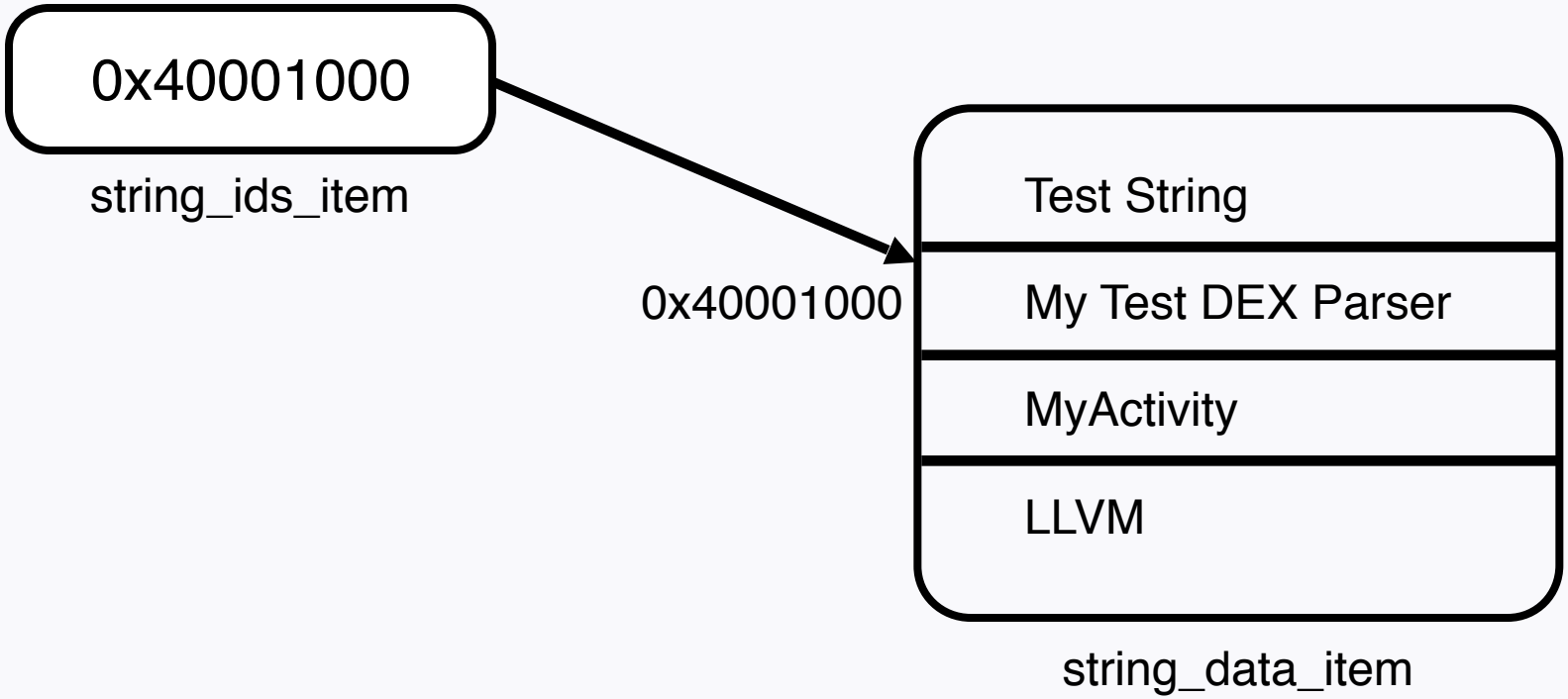
string_data_item

appears in the data section

alignment: none (byte-aligned)

Name	Format	Description
utf16_size	uleb128	size of this string, in UTF-16 code units (which is the "string length" in many systems). That is, this is the decoded length of the string. (The encoded length is implied by the position of the `0` byte.)
data	ubyte[]	a series of MUTF-8 code units (a.k.a. octets, a.k.a. bytes) followed by a byte of value `0`. See "MUTF-8 (Modified UTF-8) Encoding" above for details and discussion about the data format. > **Note:** It is acceptable to have a string which includes (the encoded form of) UTF-16 surrogate code units (that is, `U+d800` … `U+dfff`) either in isolation or out-of-order with respect to the usual encoding of Unicode into UTF-16. It is up to higher-level uses of strings to reject such invalid encodings, if appropriate.

2. Parse string_ids section



2. Parse string_ids section

```
from dexparser import Dexparser
import sys

FILE_DIR = sys.argv[1]

d = Dexparser(FILE_DIR)

print d.string_list()
```

```
['', '1.0', '<init>', 'AppTheme', 'BUILD_TYPE', 'BuildConfig.java', 'DEBUG', 'DEX Testing'
, 'FLAVOR', 'I', 'ILL', 'Landroid/app/Activity;', 'Landroid/os/Bundle;', 'Landroid/util/Lo
g;', 'Lbunseokbot/smishing/kr/test/BuildConfig;', 'Lbunseokbot/smishing/kr/test/MyActivity
;', 'Lbunseokbot/smishing/kr/test/R$attr;', 'Lbunseokbot/smishing/kr/test/R$dimen;', 'Lbun
seokbot/smishing/kr/test/R$drawable;', 'Lbunseokbot/smishing/kr/test/R$id;', 'Lbunseokbot/
smishing/kr/test/R$layout;', 'Lbunseokbot/smishing/kr/test/R$menu;', 'Lbunseokbot/smishing
/kr/test/R$string;', 'Lbunseokbot/smishing/kr/test/R$style;', 'Lbunseokbot/smishing/kr/tes
t/R;', 'Ldalvik/annotation/EnclosingClass;', 'Ldalvik/annotation/InnerClass;', 'Ldalvik/an
notation/MemberClasses;', 'Ljava/lang/Object;', 'Ljava/lang/String;', 'MyActivity.java', '
PACKAGE_NAME', 'R.java', 'SSG', 'V', 'VERSION_CODE', 'VERSION_NAME', 'VI', 'VL', 'Z', 'acc
essFlags', 'action_settings', 'activity_horizontal_margin', 'activity_my', 'activity_verti
cal_margin', 'app_name', 'attr', 'bunseokbot.smishing.kr.test', 'dimen', 'drawable', 'hell
o_world', 'i', 'ic_launcher', 'id', 'layout', 'menu', 'my', 'name', 'onCreate', 'release',
 'savedInstanceState', 'setContentView', 'string', 'style', 'this', 'value']
```

3. Parse type_ids section

type_id_item

appears in the type_ids section

alignment: 4 bytes

Name	Format	Description
descriptor_idx	uint	index into the `string_ids` list for the descriptor string of this type. The string must conform to the syntax for *TypeDescriptor*, defined above.

search from string_ids

3. Parse type_ids section

```
from dexparser import Dexparser
import sys

FILE_DIR = sys.argv[1]

d = Dexparser(FILE_DIR)

print d.typeid_list()
```

```
 mac@bunseokbot-Macbook  ~/Desktop/SSG/dex-disassembler  master ●  python inco.py /Us
ers/mac/Desktop/classes.dex
[9, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 34, 39]
```

this is descriptor_idx list

3. Parse type_ids section

```
from dexparser import Dexparser
import sys

FILE_DIR = sys.argv[1]

d = Dexparser(FILE_DIR)

s_list = d.string_list()
t_list = d.typeid_list()

for descriptor_idx in t_list:
        print s_list[descriptor_idx]
```

3. Parse type_ids section

```
I
Landroid/app/Activity;
Landroid/os/Bundle;
Landroid/util/Log;
Lbunseokbot/smishing/kr/test/BuildConfig;
Lbunseokbot/smishing/kr/test/MyActivity;
Lbunseokbot/smishing/kr/test/R$attr;
Lbunseokbot/smishing/kr/test/R$dimen;
Lbunseokbot/smishing/kr/test/R$drawable;
Lbunseokbot/smishing/kr/test/R$id;
Lbunseokbot/smishing/kr/test/R$layout;
Lbunseokbot/smishing/kr/test/R$menu;
Lbunseokbot/smishing/kr/test/R$string;
Lbunseokbot/smishing/kr/test/R$style;
Lbunseokbot/smishing/kr/test/R;
Ldalvik/annotation/EnclosingClass;
Ldalvik/annotation/InnerClass;
Ldalvik/annotation/MemberClasses;
Ljava/lang/Object;
Ljava/lang/String;
V
Z
```

TypeDescriptor Semantics

This is the meaning of each of the variants of *TypeDescriptor*.

Syntax	Meaning
V	`void`; only valid for return types
Z	`boolean`
B	`byte`
S	`short`
C	`char`
I	`int`
J	`long`
F	`float`
D	`double`
L*fully/qualified/Name*;	the class `fully.qualified.Name`
[*descriptor*	array of `descriptor`, usable recursively for arrays-of-arrays, though it is invalid to have more than 255 dimensions.

http://source.android.com/devices/tech/dalvik/dex-format.html#top

4. Parse proto_ids section

proto_id_item

appears in the proto_ids section

alignment: 4 bytes

Name	Format	Description
shorty_idx	uint	index into the `string_ids` list for the short-form descriptor string of this prototype. The string must conform to the syntax for *ShortyDescriptor*, defined above, and must correspond to the return type and parameters of this item.
return_type_idx	uint	index into the `type_ids` list for the return type of this prototype
parameters_off	uint	offset from the start of the file to the list of parameter types for this prototype, or `0` if this prototype has no parameters. This offset, if non-zero, should be in the `data` section, and the data there should be in the format specified by `"type_list"` below. Additionally, there should be no reference to the type `void` in the list.

4. Parse proto_ids section

parameters_offset

parameters_off	uint	offset from the start of the file to the list of parameter types for this prototype, or `0` if this prototype has no parameters. This offset, if non-zero, should be in the `data` section, and the data there should be in the format specified by `"type_list"` below. Additionally, there should be no reference to the type `void` in the list.

Access to “data” section directly and parse offset

4. Parse proto_ids section

```
from dexparser import Dexparser
import sys
import struct

FILE_DIR = sys.argv[1]

d = Dexparser(FILE_DIR)

s_list = d.string_list()
t_list = d.typeid_list()
m = d.mmapdata()
p_list = d.protoids_list()

for shorty_idx, return_type_idx, parameters_off in p_list:

        if parameters_off == 0:
                param = 0
        else:
                param = struct.unpack('<L', m[parameters_off:parameters_off+4])[0]

        print s_list[shorty_idx], s_list[t_list[return_type_idx]], s_list[t_list[param]]
```

4. Parse proto_ids section

```
 mac@bunseokbot-Macbook  ~/Desktop/SSG/dex-disassembler  master ●  python inco.py /Us
ers/mac/Desktop/classes.dex
ILL I Landroid/os/Bundle;
V V I
VI V Landroid/app/Activity;
VL V Landroid/app/Activity;
```

5. get Method Information

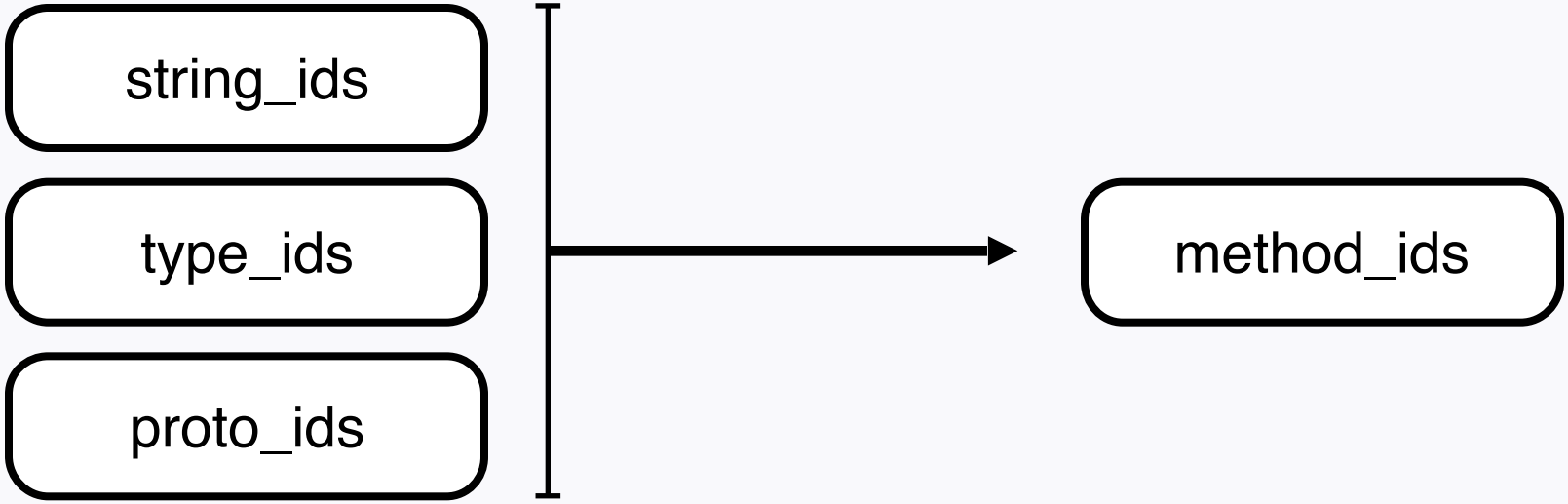
method_id_item

appears in the method_ids section

alignment: 4 bytes

Name	Format	Description
class_idx	ushort	index into the `type_ids` list for the definer of this method. This must be a class or array type, and not a primitive type.
proto_idx	ushort	index into the `proto_ids` list for the prototype of this method
name_idx	uint	index into the `string_ids` list for the name of this method. The string must conform to the syntax for *MemberName*, defined above.

5. get Method Information



5. get Method Information

```
from dexparser import Dexparser
import sys
import struct

FILE_DIR = sys.argv[1]

d = Dexparser(FILE_DIR)

s_list = d.string_list()
t_list = d.typeid_list()
m = d.mmapdata()
p_list = d.protoids_list()
n_p_list = []

for shorty_idx, return_type_idx, parameters_off in p_list:

        if parameters_off == 0:
                param = 0
        else:
                param = struct.unpack('<L', m[parameters_off:parameters_off+4])[0]

        n_p_list.append([s_list[shorty_idx], s_list[t_list[return_type_idx]], s_list[t_list[param]]])


m_list = d.method_list()
for class_idx, proto_idx, name_idx in m_list:
        print s_list[t_list[class_idx]], n_p_list[proto_idx], s_list[name_idx]
```

5. get Method Information

```
mac@bunseokbot-Macbook ~/Desktop/SSG/dex-disassembler master ● python inco.py /Us
ers/mac/Desktop/classes.dex
Landroid/app/Activity; ['V', 'V', 'I'] <init>
Landroid/app/Activity; ['VL', 'V', 'Landroid/app/Activity;'] onCreate
Landroid/util/Log; ['ILL', 'I', 'Landroid/os/Bundle;'] i
Lbunseokbot/smishing/kr/test/BuildConfig; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/MyActivity; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/MyActivity; ['VL', 'V', 'Landroid/app/Activity;'] onCreate
Lbunseokbot/smishing/kr/test/MyActivity; ['VI', 'V', 'Landroid/app/Activity;'] setContentV
iew
Lbunseokbot/smishing/kr/test/R$attr; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R$dimen; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R$drawable; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R$id; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R$layout; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R$menu; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R$string; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R$style; ['V', 'V', 'I'] <init>
Lbunseokbot/smishing/kr/test/R; ['V', 'V', 'I'] <init>
Ljava/lang/Object; ['V', 'V', 'I'] <init>
```

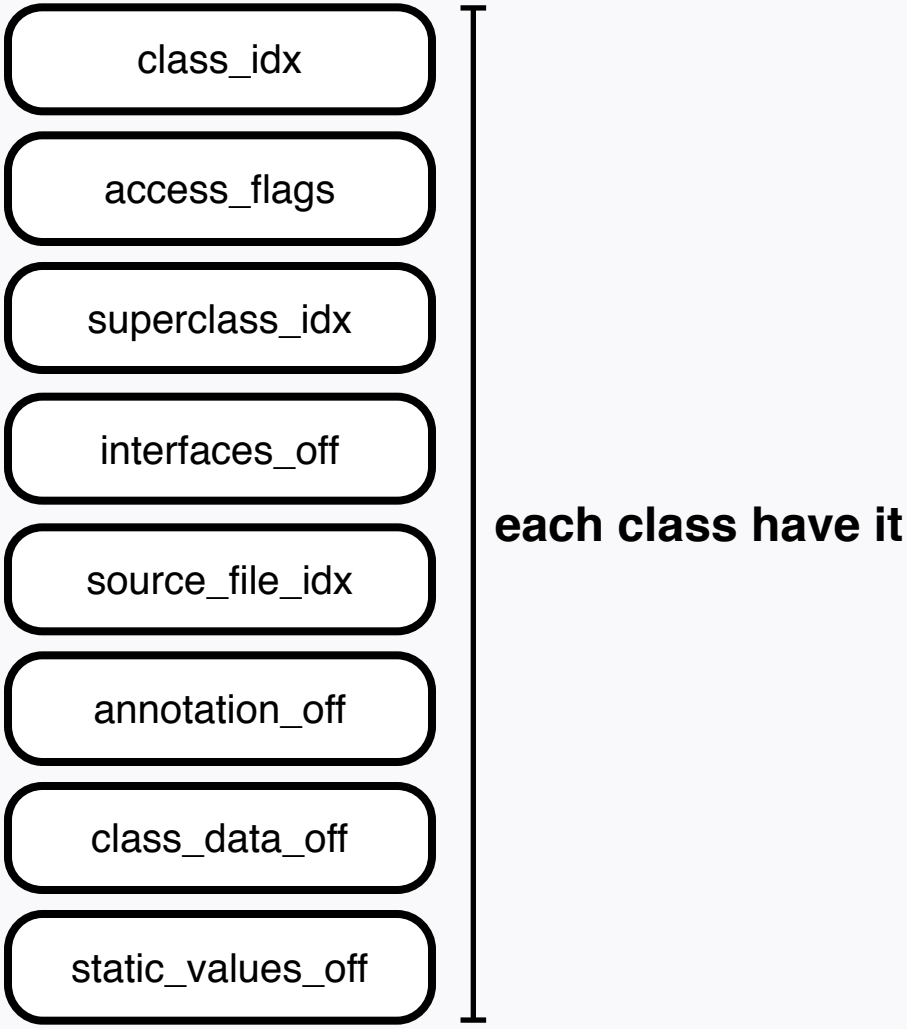
6. get Class Information

class_def_item

appears in the class_defs section

alignment: 4 bytes

Name	Format	Description
class_idx	uint	index into the `type_ids` list for this class. This must be a class type, and not an array or primitive type.
access_flags	uint	access flags for the class (`public`, `final`, etc.). See "`access_flags` Definitions" for details.
superclass_idx	uint	index into the `type_ids` list for the superclass, or the constant value `NO_INDEX` if this class has no superclass (i.e., it is a root class such as `Object`). If present, this must be a class type, and not an array or primitive type.
interfaces_off	uint	offset from the start of the file to the list of interfaces, or `0` if there are none. This offset should be in the `data` section, and the data there should be in the format specified by "`type_list`" below. Each of the elements of the list must be a class type (not an array or primitive type), and there must not be any duplicates.
source_file_idx	uint	index into the `string_ids` list for the name of the file containing the original source for (at least most of) this class, or the special value `NO_INDEX` to represent a lack of this information. The `debug_info_item` of any given method may override this source file, but the expectation is that most classes will only come from one source file.
annotations_off	uint	offset from the start of the file to the annotations structure for this class, or `0` if there are no annotations on this class. This offset, if non-zero, should be in the `data` section, and the data there should be in the format specified by "`annotations_directory_item`" below, with all items referring to this class as the definer.
class_data_off	uint	offset from the start of the file to the associated class data for this item, or `0` if there is no class data for this class. (This may be the case, for example, if this class is a marker interface.) The offset, if non-zero, should be in the `data` section, and the data there should be in the format specified by "`class_data_item`" below, with all items referring to this class as the definer.
static_values_off	uint	offset from the start of the file to the list of initial values for `static` fields, or `0` if there are none (and all `static` fields are to be initialized with `0` or `null`). This offset should be in the `data` section, and the data there should be in the format specified by "`encoded_array_item`" below. The size of the array must be no larger than the number of `static` fields declared by this class, and the elements correspond to the `static` fields in the same order as declared in the corresponding `field_list`. The type of each array element must match the declared type of its corresponding field. If there are fewer elements in the array than there are `static` fields, then the leftover fields are initialized with a type-appropriate `0` or `null`.



6. get Class Information

class_idx	
access_flags	**ACCESS_FLAG**
superclass_idx	
interfaces_off	
source_file_idx	**Source File index**
annotation_off	
class_data_off	**associated class data offset**
static_values_off	**class level static value offset**

6. calculate ACCESS_FLAG

Name	Value	For Classes (and `InnerClass` annotations)	For Fields	For Methods
ACC_PUBLIC	0x1	`public`: visible everywhere	`public`: visible everywhere	`public`: visible everywhere
ACC_PRIVATE	0x2	* `private`: only visible to defining class	`private`: only visible to defining class	`private`: only visible to defining class
ACC_PROTECTED	0x4	* `protected`: visible to package and subclasses	`protected`: visible to package and subclasses	`protected`: visible to package and subclasses
ACC_STATIC	0x8	* `static`: is not constructed with an outer `this` reference	`static`: global to defining class	`static`: does not take a `this` argument
ACC_FINAL	0x10	`final`: not subclassable	`final`: immutable after construction	`final`: not overridable
ACC_SYNCHRONIZED	0x20			`synchronized`: associated lock automatically acquired around call to this method. **Note:** This is only valid to set when `ACC_NATIVE` is also set.
ACC_VOLATILE	0x40		`volatile`: special access rules to help with thread safety	
ACC_BRIDGE	0x40			bridge method, added automatically by compiler as a type-safe bridge

VARIOUS TYPE OF ACCESS_FLAG!

6. calculate ACCESS_FLAG

ACC_PUBLIC	0x1
ACC_PRIVATE	0x2
ACC_PROTECTED	0x4
ACC_STATIC	0x8

public static void myFunction() {}

ACC_PUBLIC = 0x1
ACC_STATIC = 0x8

ACCESS_FLAG = 0x9

6. calculate ACCESS_FLAG

```
access_flag_methods = {
    0x1:     'public',
    0x2:     'private',
    0x4:     'protected',
    0x8:     'static',
    0x10:    'final',
    0x20:    'synchronized',
    0x40:    'bridge',
    0x80:    'varargs',
    0x100:   'native',
    0x400:   'abstract',
    0x800:   'strictfp',
    0x1000:  'synthetic',
    0x10000: 'constructor',
    0x20000: 'declared_sy
}
```

```
ACCESS_ORDER = [0x1, 0x4, 0x2, 0x400, 0x8, 0x10,
                0x80, 0x40, 0x20, 0x100, 0x800,
                0x200, 0x1000, 0x2000, 0x4000,
                0x10000, 0x20000]
```

6. class_data_item

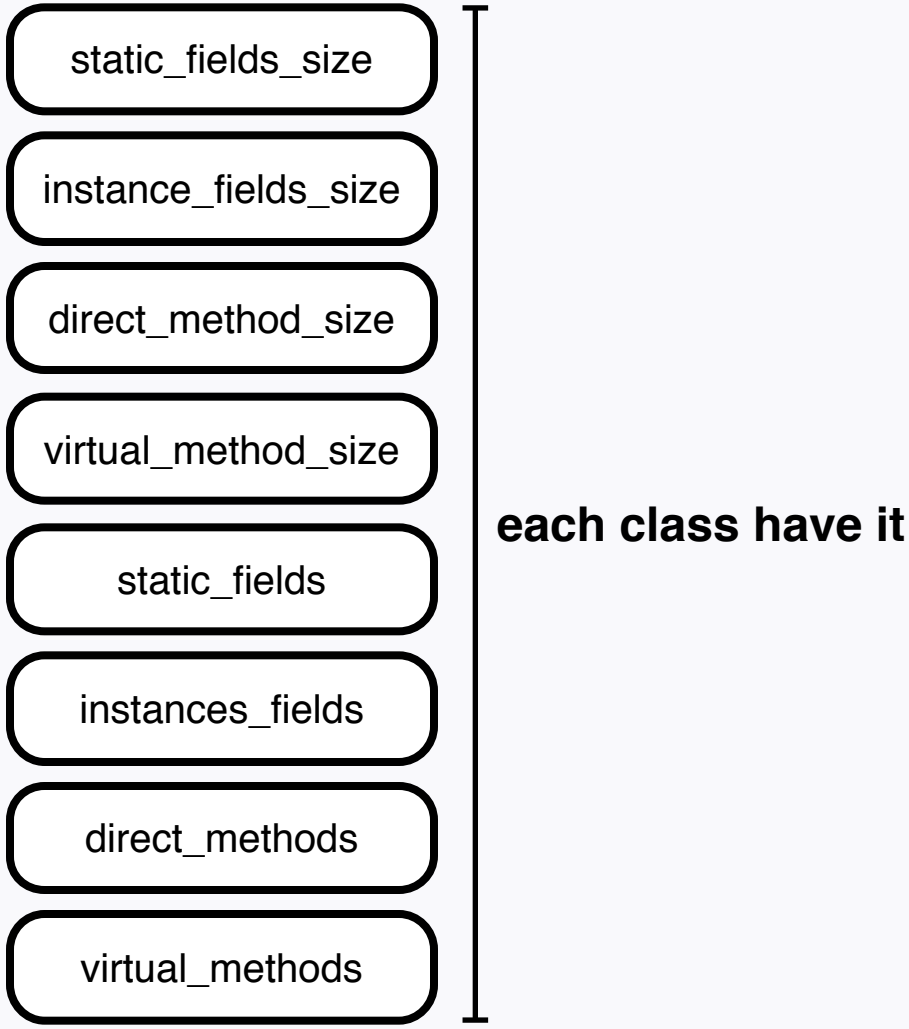
class_data_item

referenced from class_def_item

appears in the data section

alignment: none (byte-aligned)

Name	Format	Description
static_fields_size	uleb128	the number of static fields defined in this item
instance_fields_size	uleb128	the number of instance fields defined in this item
direct_methods_size	uleb128	the number of direct methods defined in this item
virtual_methods_size	uleb128	the number of virtual methods defined in this item
static_fields	encoded_field[static_fields_size]	the defined static fields, represented as a sequence of encoded elements. The fields must be sorted by `field_idx` in increasing order.
instance_fields	encoded_field[instance_fields_size]	the defined instance fields, represented as a sequence of encoded elements. The fields must be sorted by `field_idx` in increasing order.
direct_methods	encoded_method[direct_methods_size]	the defined direct (any of `static`, `private`, or constructor) methods, represented as a sequence of encoded elements. The methods must be sorted by `method_idx` in increasing order.
virtual_methods	encoded_method[virtual_methods_size]	the defined virtual (none of `static`, `private`, or constructor) methods, represented as a sequence of encoded elements. This list should *not* include inherited methods unless overridden by the class that this item represents. The methods must be sorted by `method_idx` in increasing order.



6. class_data_item

Name	Format	Description
static_fields_size	uleb128	the number of static fields defined in this item
instance_fields_size	uleb128	the number of instance fields defined in this item
direct_methods_size	uleb128	the number of direct methods defined in this item
virtual_methods_size	uleb128	the number of virtual methods defined in this item

6. leb128?

- Little-Endian Based 128
- Type : Unsigned & Signed
- following DWARF Format
- for encoding variable-length, arbitrary signed or unsigned int

Bitwise diagram of a two-byte LEB128 value															
First byte								Second byte							
1	bit_6	bit_5	bit_4	bit_3	bit_2	bit_1	bit_0	0	bit_{13}	bit_{12}	bit_{11}	bit_{10}	bit_9	bit_8	bit_7

6. leb128?

```
#uleb128 decoder!
def uleb128_value(m, off):
        size = 1
        result = ord(m[off+0])
        if result > 0x7f :
                cur = ord(m[off+1])
                result = (result & 0x7f) | ((cur & 0x7f) << 7)
                size += 1
                if cur > 0x7f :
                        cur = ord(m[off+2])
                        result |= ((cur & 0x7f) << 14)
                        size += 1
                        if cur > 0x7f :
                                cur = ord(m[off+3])
                                result |= ((cur & 0x7f) << 21)
                                size += 1
                                if cur > 0x7f :
                                        cur = ord(m[off+4])
                                        result |= (cur << 28)
                                        size += 1
        return result, size
```

return value is **(result, size)**

6. get Class Information!

```
1
class_data_read_start!
length of direct_method is  11
[[464, 65544, 165140], [465, 65538, 165228], [466, 9, 165252], [467, 9, 165344], [468, 9, 165560], [469, 9, 165676], [470, 9, 165708], [471, 9, 165740], [472, 9, 165772], [473, 9, 1
65804], [474, 9, 165836]]
direct_method_idx_diff 0x1d0
direct_method_access_flag ['static', 'constructor']
direct_method_code_off 0x28514
direct_method_idx_diff 0x1d1
direct_method_access_flag ['private', 'constructor']
direct_method_code_off 0x2856c
direct_method_idx_diff 0x1d2
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x28584
direct_method_idx_diff 0x1d3
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x285e0
direct_method_idx_diff 0x1d4
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x286b8
direct_method_idx_diff 0x1d5
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x2872c
direct_method_idx_diff 0x1d6
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x2874c
direct_method_idx_diff 0x1d7
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x2876c
direct_method_idx_diff 0x1d8
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x2878c
direct_method_idx_diff 0x1d9
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x287ac
direct_method_idx_diff 0x1da
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x287cc

2
class_data_read_start!
length of direct_method is  6
[[475, 65536, 165868], [476, 9, 165892], [477, 9, 165920], [478, 9, 165948], [479, 9, 165976], [480, 9, 166004]]
direct_method_idx_diff 0x1db
direct_method_access_flag ['constructor']
```

6. get Class Information!

```
124
class_data_read_start!
length of direct_method is  2
[[1688, 65536, 204748], [1689, 9, 204772]]
direct_method_idx_diff 0x698
direct_method_access_flag ['constructor']
direct_method_code_off 0x31fcc
direct_method_idx_diff 0x699
direct_method_access_flag ['public', 'static']
direct_method_code_off 0x31fe4
```

Apply to Real World

- smishing.kr

isolate Real Permission

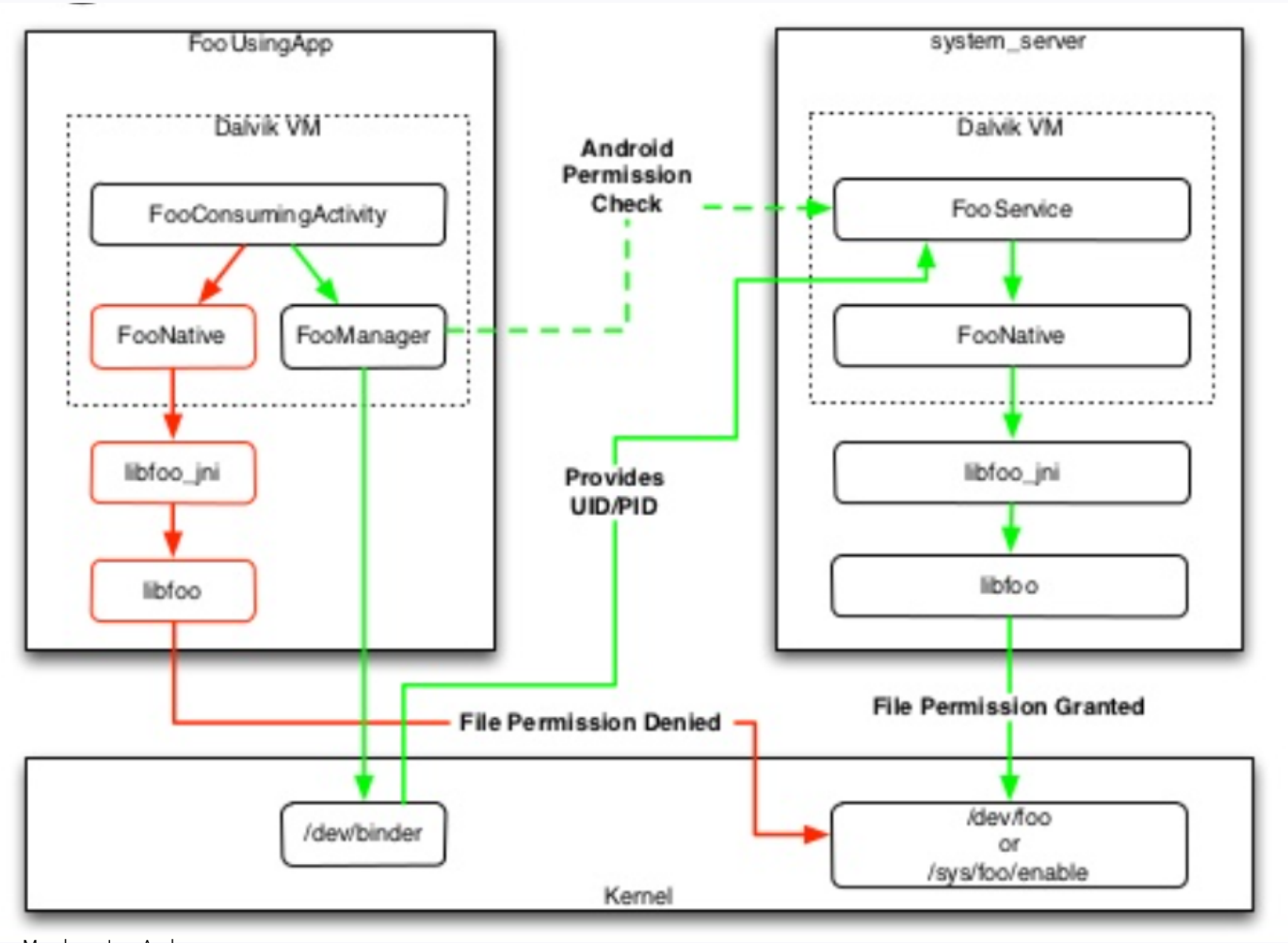
- Usually, Android Malware Analyzer's check this thing!
  "Searching Permission info from AndroidManifest.xml"
- BUT..
- Recently, Attacker register all uses-permission to AndroidManifest.xml

isolate Real Permission

android.permission.*

- If application want to use Permission such as INTERNET, ACCESS_COURSE_LOCATION. They must call **“Authorized Function”**

Logical Permission Enforcement



Marakana Inc. Andsec

isolate Real Permission

android.permission.*

- invoke-* [parameter] [call method]
  -> direct
  -> virtual
  -> super
  -> static
  -> interface

isolate Real Permission

android.permission.*

- invoke-* **[parameter] [call method]**
  -> direct
  -> virtual
  -> super
  -> static
  -> interface

For Example..

android.permission.INTERNET

invoke-direct {v0, v1}, Lorg/../HttpGet;-><init>(Ljava/lang/String;)V

Lorg/../HttpGet class “**Approve**” to access Internet Module

Network connection **success**!

Where to get Function Mapping..

- inside Android Source…?
- “Androguard” have it
- https://github.com/androguard/androguard

in Androguard

- in androguard/core/bytecodes/api_permission.py

```
"INTERNET" : {
        "Lcom/android/http/multipart/FilePart;" : [
                ("F", "sendData", "(Ljava/io/OutputStream;)"),
                ("F", "sendDispositionHeader", "(Ljava/io/OutputStream;)"),
        ],
        "Ljava/net/HttpURLConnection;" : [
                ("F", "<init>", "(Ljava/net/URL;)"),
                ("F", "connect", "()"),
        ],
        "Landroid/webkit/WebSettings;" : [
                ("F", "setBlockNetworkLoads", "(B)"),
                ("F", "verifyNetworkAccess", "()"),
        ],
        "Lorg/apache/http/impl/client/DefaultHttpClient;" : [
                ("F", "<init>", "()"),
                ("F", "<init>", "(Lorg/apache/http/params/HttpParams;)"),
                ("F", "<init>", "(Lorg/apache/http/conn/ClientConnectionManager; Lorg/apache/http/params/HttpParams;)"),
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest;)"),
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler; Lorg/apache/http/protocol/HttpContext;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler; Lorg/apache/http/protocol/HttpContext;)"),
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/protocol/HttpContext;)"),
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/protocol/HttpContext;)"),
        ],
        "Lorg/apache/http/impl/client/HttpClient;" : [
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest;)"),
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler; Lorg/apache/http/protocol/HttpContext;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler; Lorg/apache/http/protocol/HttpContext;)"),
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/protocol/HttpContext;)"),
                ("F", "execute", "(Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/client/ResponseHandler;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest;)"),
                ("F", "execute", "(Lorg/apache/http/HttpHost; Lorg/apache/http/client/methods/HttpUriRequest; Lorg/apache/http/protocol/HttpContext;)"),
        ],
        "Lcom/android/http/multipart/Part;" : [
                ("F", "send", "(Ljava/io/OutputStream;)"),
                ("F", "sendParts", "(Ljava/io/OutputStream; [Lcom/android/http/multipart/Part;)"),
                ("F", "sendParts", "(Ljava/io/OutputStream; [Lcom/android/http/multipart/Part; [B)"),
                ("F", "sendStart", "(Ljava/io/OutputStream;)"),
                ("F", "sendTransferEncodingHeader", "(Ljava/io/OutputStream;)"),
        ],
        "Landroid/drm/DrmErrorEvent;" : [
                ("C", "TYPE_NO_INTERNET_CONNECTION", "I"),
        ],
        "Landroid/webkit/WebViewCore;" : [
                ("F", "<init>", "(Landroid/content/Context; Landroid/webkit/WebView; Landroid/webkit/CallbackProxy; Ljava/util/Map;)"),
        ],
        "Ljava/net/URLConnection;" : [
                ("F", "connect", "()"),
                ("F", "getInputStream", "()"),
        ],
```

Conclusion

- DEX have “Simple” structure
- source.android.com have specific DEX file info
- If you use “dexparser” in Python, You’ll be easy to parse DEX and analyze.
- DEXCrypt, DEXGuard have same structure!
- DEX Research is very fun!

감사합니다