

Java Reversing

Nathan



```
sigpwny{java_reversing_101}
```



Announcements

- Last content meeting of the semester is this Sunday
 - Hacking Mobile Apps
 - Next thursday will be social
- Next Semester Logistics
 - UIUCTF
 - Research
 - Meetings



How does Java work?

- .jar files
 - This is the Java “executable”
 - Literally just a zip file renamed to .jar
 - Contains compiled classes (*.class) and metadata (META-INF)
- .class files
 - Represents a compiled .java file
 - Contains JVM bytecode
- Metadata
 - In file META-INF/MANIFEST.MF
 - Tells the JVM which class has the main method (where to start)



Jar example

```
(base) nathan@desktop:~/Downloads$ java -jar challenge.jar  
Please enter the password to continue: █
```

```
(base) nathan@desktop:~/Downloads$ unzip challenge.jar  
Archive:  challenge.jar  
  creating:  META-INF/  
  inflating:  META-INF/MANIFEST.MF  
  inflating:  com/sigpwny/Main.class  
(base) nathan@desktop:~/Downloads$ █
```



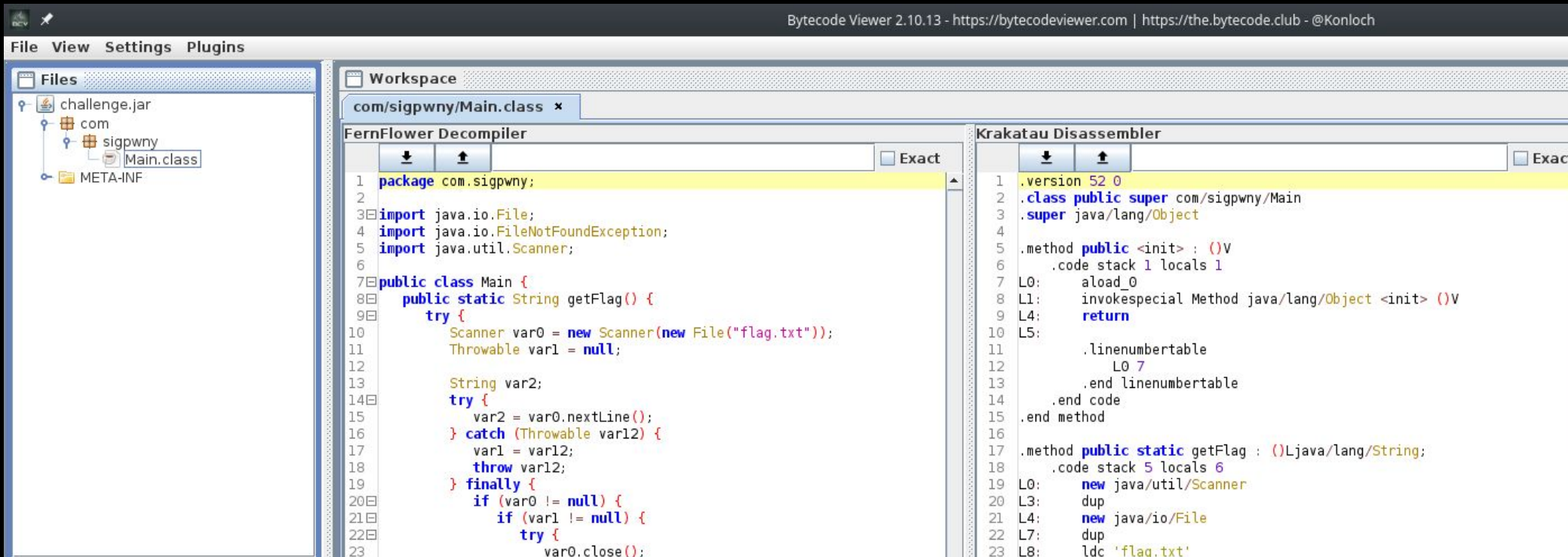
Good news about Java

- Very easy to decompile accurately
 - You can keep:
 - Class names
 - Function names
 - Class variable names
 - Line numbers!!!
 - (Used for printing exception stack traces, but makes decompilation even easier)
 - You might lose:
 - Local variable names
 - Comments



How to decompile

- Use Bytecode-Viewer
 - <https://github.com/Konloch/bytecode-viewer>
 - Procyon, FernFlower decompilers included



The screenshot displays the Bytecode Viewer application interface. The top bar shows the title "Bytecode Viewer 2.10.13" and the URL "https://bytecodeviewer.com". The left sidebar shows a file tree with "challenge.jar" containing "com/sigpwny/Main.class". The main workspace is split into two panes: "FernFlower Decompiler" and "Krakatau Disassembler".

```
1 package com.sigpwny;
2
3 import java.io.File;
4 import java.io.FileNotFoundException;
5 import java.util.Scanner;
6
7 public class Main {
8     public static String getFlag() {
9         try {
10             Scanner var0 = new Scanner(new File("flag.txt"));
11             Throwable var1 = null;
12
13             String var2;
14             try {
15                 var2 = var0.nextLine();
16             } catch (Throwable var12) {
17                 var1 = var12;
18                 throw var12;
19             } finally {
20                 if (var0 != null) {
21                     if (var1 != null) {
22                         try {
23                             var0.close();
```

```
1 .version 52 0
2 .class public super com/sigpwny/Main
3 .super java/lang/Object
4
5 .method public <init> : ()V
6     .code stack 1 locals 1
7 L0:   aload_0
8 L1:   invokespecial Method java/lang/Object <init> ()V
9 L4:   return
10 L5:
11     .linenumbertable
12         L0 7
13     .end linenumbertable
14 .end code
15 .end method
16
17 .method public static getFlag : ()Ljava/lang/String;
18     .code stack 5 locals 6
19 L0:   new java/util/Scanner
20 L3:   dup
21 L4:   new java/io/File
22 L7:   dup
23 L8:   ldc 'flag.txt'
```



Java Obfuscation

- Obfuscators can
 - Change control flow
 - Rename classes/variables/functions
 - Remove + add functions
 - Encrypt strings
 - Make it impossible to use a decompiler
- Popular obfuscators:
 - ZKM, Allatori, Proguard



Java Deobfuscation

- Pro strat:
 - Run the obfuscated .jar through a deobfuscator
 - Run the deobfuscated jar through your decompiler
 - Typically only works if obfuscation is not custom
 - Still lose class/variable/function names
- Use <https://github.com/java-deobfuscator/deobfuscator>
- See <https://youtu.be/oal1F7D2Z2A?t=428> for a tutorial



Java Deobfuscation

Original

Obfuscated

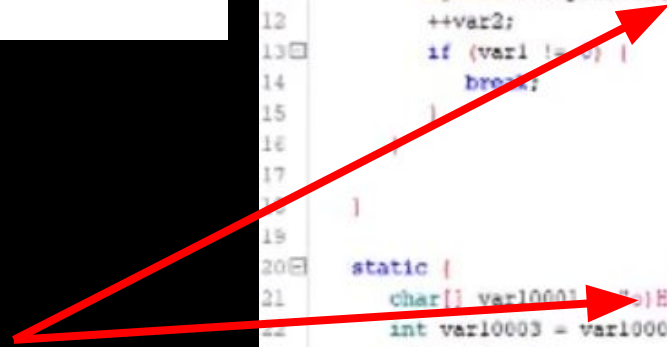
Deobfuscated

```
Fernflower Decompiler - Editable: false
1 package com.sigpwny;
2
3 public class Main {
4     public static void main(String[] args) {
5         for(int i = 0; i < 5; ++i) {
6             System.out.println("Hello world: " + i);
7         }
8     }
9 }
10
11
```

```
Fernflower Decompiler - Editable: false
1 public class a {
2     public static int a;
3     public static int b;
4     private static final String c;
5
6     public static void main(String[] var0) {
7         int var2 = 0;
8         int var1 = b;
9
10        while(var2 < 5) {
11            System.out.println(c + var2);
12            ++var2;
13            if (var1 != 0) {
14                break;
15            }
16        }
17    }
18
19
20    static {
21        char[] var10001 = "H2R\nEH>B:\u0007\n".toCharArray();
22        int var10003 = var10001.length;
23        int var0 = 0;
24        char[] var10002 = var10001;
25        int var2 = var10003;
26        String var1;
27        char[] var4;
28        int var10004;
29        boolean var5;
```

```
Fernflower Decompiler - Editable: false
1 public class a {
2     public static int a;
3     public static int b;
4     private static final String c = "Hello world: ";
5
6     public static void main(String[] var0) {
7         int var2 = 0;
8         int var1 = b;
9
10        while(var2 < 5) {
11            System.out.println("Hello world: " + var2);
12            ++var2;
13            if (var1 != 0) {
14                break;
15            }
16        }
17    }
18
19
```

Encrypted
"Hello world"



Tackling Custom Obfuscation

- See how far regular deobfuscators get you
- Use ObjectWeb ASM library to parse + manipulate class bytecode yourself
 - Prepare to spend a lot of time on this
- Newer constructs in later JVM standards (invokedynamic) have created opportunities for new crazy obfuscators
 - See <https://itzsomebody.xyz/2021/07/11/javaisez3-writeup.html>



Next Meetings

Weekend Seminar: Mobile Hacking with Pete

- Mobile hacking

Next Thursday: Social @ Murphys

- Good luck on finals!



Go try for yourself!

<https://ctf.sigpwny.com>

- Start with Java Reversing 1
- Bytecode-Viewer can solve Java Reversing 1, 1.5, and 2
- Practice practice practice! Ask for help!

