S2—045漏洞分析

potato / 2017-03-09 09:42:00 / 浏览数 4425 安全技术 漏洞分析 顶(0) 踩(0)

s2-045 分析

1. 漏洞分析
   以下分析基于版本 struct 2.3.20 版本

struct 使用java
ee中的Filter去拦截请求，并实现自己的功能。也就是说，用户所发出的请求，首先会在`org.apache.struts2.dispatcher.ng.filter`中的StrutsPrepareAndExecut

我们跟入这个函数

会执行`dispatcher.wrapRequest`这个函数。于是继续跟入

在这里，将会判断一下是否包含multipart/form-data这个头。注意，在这里，并没去区分是什么http方法，也就是说，get方法强行包含一个带有恶意代码的content-type
在这里，首先判断一下是否为null和是否包含multipart/form-data。如果不包含，也就是get方法和post表单提交，则执行else里面的内容。如果包含，也就是说可能有文件
在if中，首先获取到默认的解析器

可以查看一下multipartHandlerName的默认值

也就是获取默认配置中的值，而在struct的默认配置(org.apache.struct2的default.properties)中，这个值恰好为jakarta，默认配置为

回到dispatcher.wrapReques中，获取到默认的解析器后，将会把MultiPartRequestWrapper这个类实例化，执行MultiPartRequestWrapper的构造函数。跟入MultiPart

在这里，将会执行muiti.pase函数，因为这里的multipartrequest为JakartaMultiPartRequest类，于是将会执行JakartaMultiPartRequest.parse方法，跟入之

这里是重点，在这里将会执行java的捕获异常，如果捕获到异常的花，将会执行builderrorMessage方法，也就是产生漏洞的地方。继续跟入，将会执行processUpload函数

在这里，将会调用parseRequest对请求进行解析，跟入

在这里，将会调用upload.parseRequest函数，跟入

现在已经进入org.apache.commons.fileupload的FileUploadBase类，注意在这里，没有对FileItemStreamImpl进行捕获异常，于是跟入FileItemStreamImpl的构造函数中

在这里，将会得到content-type头，并且检查一下是否以multipart开头。所以，exp中并不会以multipart开头。于是在这里将会抛出异常，并将content-type信息放入异常
这里并没有任何代码捕获异常，于是向函数调用方继续抛出，到org.apache.commons.fileupload的FileUploadBase类的parseRequest方法，但是parseRequest方法中，

在这里将会执行LocalizedTextUtil.findText。谷歌一下这个函数的信息

发现这里可能会执行ognl语句，跟入这个函数

将会执行getDefaultMessage，参数为包含payload的content-type头，跟入

在这里，将会执行TextParseUtil.translateVariables，谷歌一下这个函数

到这里，将会把包含payload的content-type头作为ognl语句去执行。再继续跟，就是表达式处理的过程了。于是，漏洞就产生了

s2-045 分析.pdf.zip (1.2 MB) 下载附件
点击收藏 | 0 关注 | 1
上一篇：如何优雅的把LFI转化为RCE（2... 下一篇：CSRF漏洞挖掘

1. 0 条回复
   - 动动手指，沙发就是你的了！

目录