Struts S2-045 漏洞调试及分析

Auth : Cryin’
Date : 2016.3.9

漏洞公告

首先看官方给出的漏洞公告信息:

“Possible Remote Code Execution when performing file upload based on **Jakarta Multipart parser.**”

问题原因:

“It is possible to perform a RCE attack with a **malicious Content-Type value.** If the Content-Type value isn’t valid **an exception is thrown** which is then used to **display an error message** to a user.”

从公告信息可以得到几个漏洞的重点信息:

- 存在漏洞的模块是 **Jakarta**
- 漏洞产生的点是恶意的 **Content-Type 头**
- 恶意的 Content-Type 头会造成程序**抛出异常**，在**显示错误消息给用户**时造成 RCE

补丁对比

查看 Struts2 版本 2.3.32 在 github 上的 commit(Uses default error key if specified key doesn’t exist)并对比修改内容：

https://github.com/apache/struts/commit/352306493971e7d5a756d61780d57a76eb1f519a

可以看到对 LocalizedTextUtil.findText 方法进行了重写，并添加了判断。Struts 2 RCE 漏洞的根本原因是程序将用户可控数据带入 OGNL 表达式解析并执行，而 OGNL(Object Graph Navigation Language)对象图形导航语言是一个功能强大的表达式语言，可以用来获取和设置 Java 对象的属性，但同时也可以对服务端对象进行修改，绕过沙盒甚至可以执行系统命令。

所以，从补丁分析来看 LocalizedTextUtil.findText 函数很可能是 OGNL 表达式的传入点，在调试分析时可在此处下断点跟踪分析。

关于 jakarta

Jakarta 是 struts2 默认处理 multipart 报文的解析器，该组件定义在 struts-default.xml 中：

默认使用 org.apache.struts2.dispatcher.multipart.JakartaMultiPartRequest 类对上传数据进行解析并调用第三方组件 commons-fileupload.jar 完成上传操作。

开发人员可以通过配置 struts.multipart.parser 属性，修改指定不通的解析类。如图:

```
65  ### Parser to handle HTTP POST requests, encoded using the MIME-type multipart/form-data
66  # struts.multipart.parser=cos
67  # struts.multipart.parser=pell
68  # struts.multipart.parser=jakarta-stream
69  struts.multipart.parser=jakarta
70  # uses javax.servlet.context.tempdir by default
71  struts.multipart.saveDir=
72  struts.multipart.maxSize=2097152
```

所以，只要不修改默认的 struts.multipart.parser 属性,且有 commons-fileupload.jar 包。在受影响 struts2 版本范围内则都受此漏洞影响。不用实现上传功能，只要具备上述条件的 struts2 程序，即可进行漏洞调试、跟踪漏洞触发过程进行深入分析。

漏洞触发过程

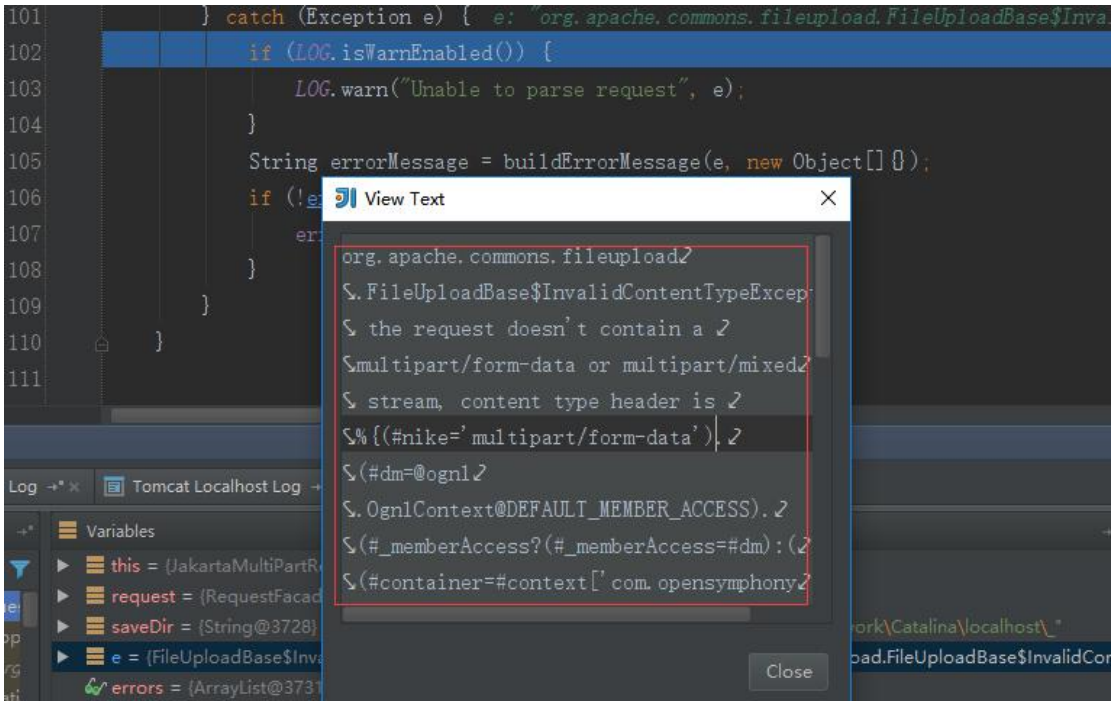
由于不需要具体实现代码，这里建一个空工程即可，下载 struts 2 源代码并关联，这里使用 struts-2.2.30 版本，并在上面提到的 LocalizedTextUtil.findText 函数、JakartaMultiPartRequest 类的 parse 函数以及解析前的 content_type 校验处下断点，启动调试并运行 POC 测试脚本，在 org.apache.struts2.dispatcher.Dispatcher.java 文件中的 wrapRequest 函数对 content_type 校验处截断开始单步跟踪调试。

```
837         String content_type = request.getContentType();  request: RequestFacade@3
838         if (content_type != null && content_type.contains("multipart/form-data"))
839             MultiPartRequest mpr = getMultiPartRequest();
840             LocaleProvider provider = getContainer().getInstance(LocaleProvider.cl
841             request = new MultiPartRequestWrapper(mpr, request, getSaveDir(), prov
```

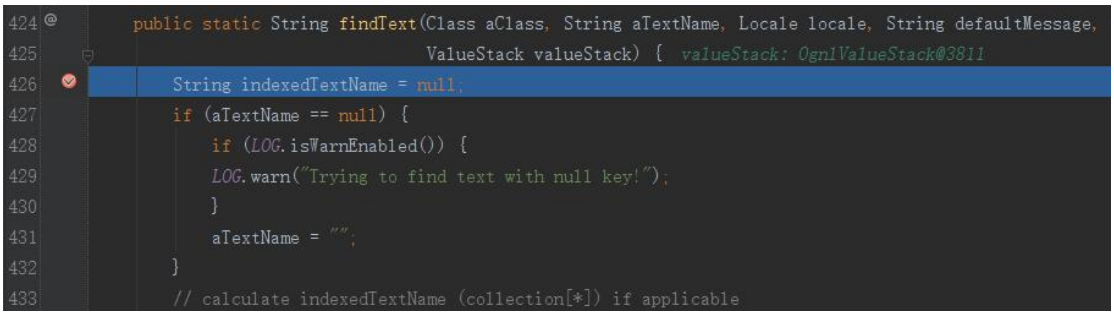
此处对 content_type 头进行检查，判断是否包含 multipart/form-data 字段，判断成功后进行 JakartaMultiPartRequest 的 parse 函数进行解析，这就是为什么该漏洞 POC 都包含 multipart/form-data 字段的原因。继续单步跟进，进入 parse 函数的断点。

```
89  public void parse(HttpServletRequest request, String saveDir) throws IOException {  request: RequestFacade@3
90      try {
91          setLocale(request);  request: RequestFacade@3749
92          processUpload(request, saveDir);
93      } catch (FileUploadBase.SizeLimitExceededException e) {
94          if (LOG.isWarnEnabled()) {
95              LOG.warn("Request exceeded size limit!", e);
96          }
97          String errorMessage = buildErrorMessage(e, new Object[]{e.getPermittedSize(), e.getActualSize()});
98          if (!errors.contains(errorMessage)) {
99              errors.add(errorMessage);
100         }
101     } catch (Exception e) {
102         if (LOG.isWarnEnabled()) {
103             LOG.warn("Unable to parse request", e);
104         }
105         String errorMessage = buildErrorMessage(e, new Object[]{});
106         if (!errors.contains(errorMessage)) {
```

继续单步运行，在 processUpload 函数位置抛出异常，并跳转到异常流程中，抛出的异常信息为:Invalid ContentType Exception: the request doesn't contain a multipart/form-data or multipart/mixed stream.因为 ContentType 未包含 multipart/form-data 或 multipart/mixed 流内容，造成程序异常。其中可以看到异常消息中同时也包含了测试 POC 中 ContentType 头中的 payload。



接着继续单步跟踪，程序流程进入 buildErrorMessage 函数并传入了异常消息。该函数就应该是漏洞公告中提到的显示错误信息给用户的这部分功能代码。继续跟进进入了下好断点的 LocalizedTextUtil.findText 函数。



查看 LocalizedTextUtil.findText 函数的介绍 https://struts.apache.org/maven/struts2-core/apidocs/com/opensymphony/xwork2/util/LocalizedTextUtil.html：

“If a message is found, it will also be interpolated. Anything within ${...} will be treated as an OGNL expression and evaluated as such.”

继续跟进，最后可以看到异常消息被传入 TextParseUtil.ParsedValueEvaluator 对象的 evaluate 方法中。

evaluate

```
Object  evaluate(String  parsedValue)
```

Evaluated the value parsed by Ognl value stack.

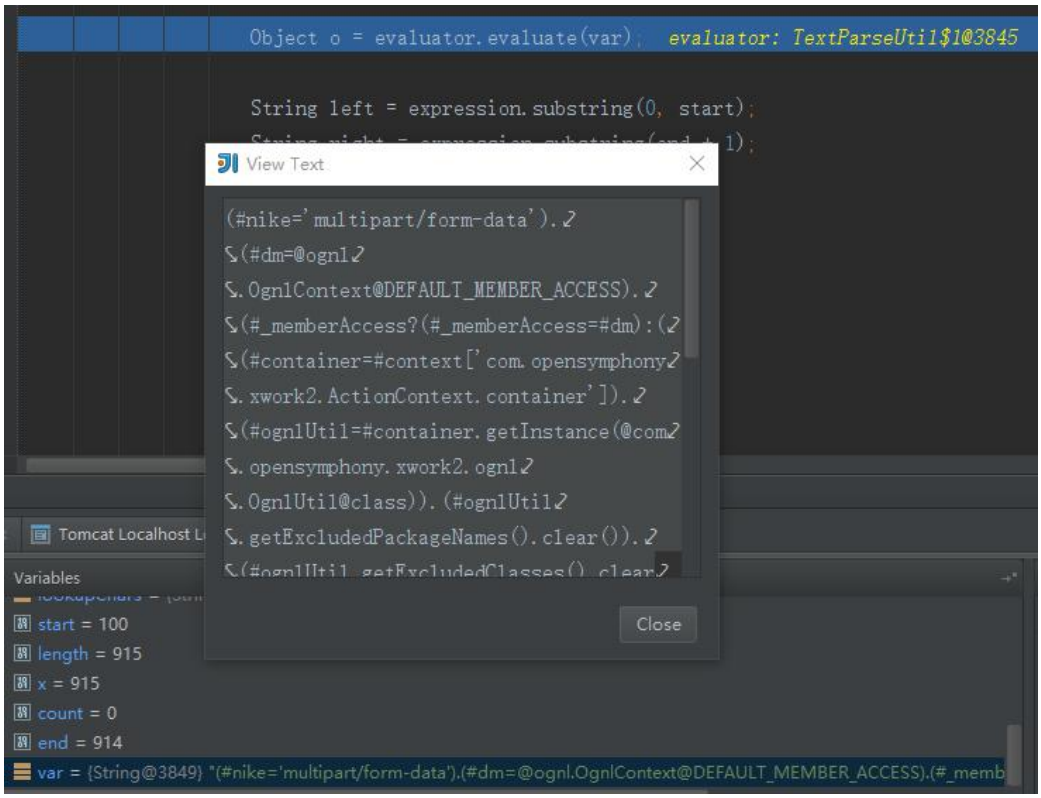
Parameters:

parsedValue – – value parsed by ognl value stack

Returns:

return the evaluted value.

这里提到 Ognl 值栈，可以看到包含构造 Ognl 语句的异常消息进入该函数，从而导致命令执行。



防护建议

- 升级 struts2 版本到 2.3.32 或 2.5.10.1
- 使用 pell、cos 等其它 multipart 解析器
- 弃坑，使用 SpringMVC

参考

[1] http://www.secbox.cn/hacker/program/205.html
[2] http://paper.seebug.org/241/
[3] http://blog.csdn.net/u011721501/article/details/60768657
[4] http://blog.csdn.net/three_feng/article/details/60869254
[5] https://github.com/apache/struts/
[6] http://blog.nsfocus.net/apache-struts2-remote-code-execution-vulnerability-analysis-program/
[7] https://cwiki.apache.org/confluence/display/WW/S2-045
[8] http://archive.apache.org/dist/struts/