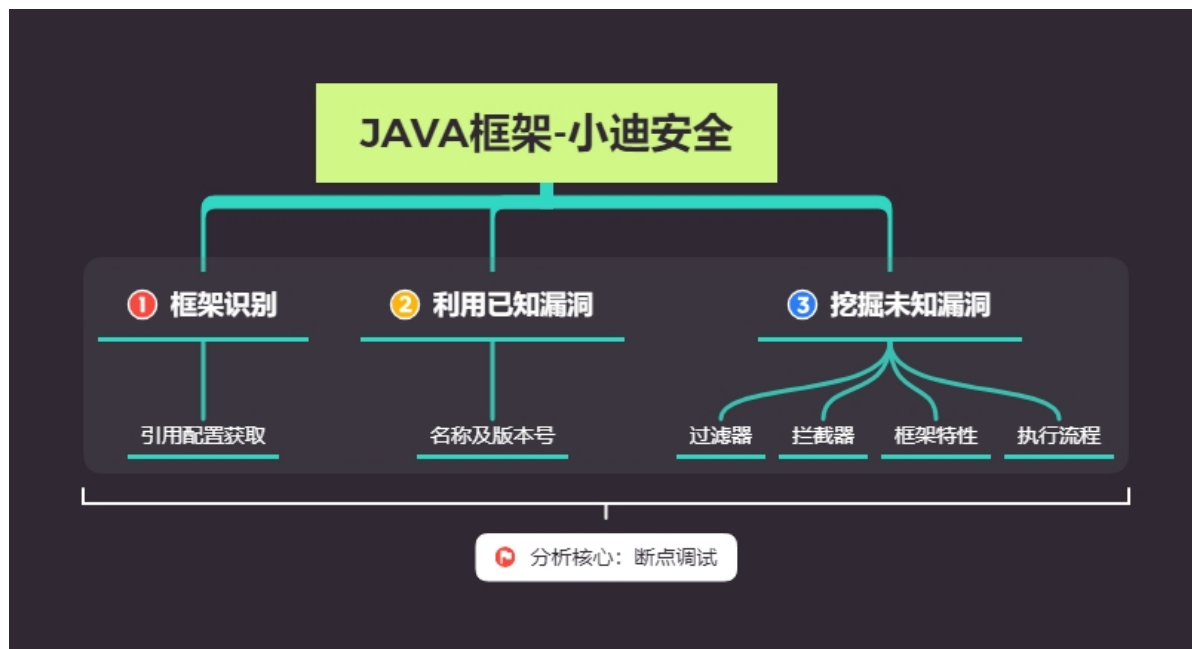


Day57 代码审计-JAVA项目 框架类漏洞分析报告



57.1 过滤器及拦截器相关区别解释

Filter是基于函数回调的，而Interceptor这是基于Java反射的。Filter依赖于Servlet容器，而Interceptor不依赖于Servlet容器。

Filter对几乎所有的请求起作用，而Interceptor只能对action请求起作用。

Interceptor可以访问Action的上下文，值栈里的对象，而Filter不能。

最重要的要记住他们的执行顺序：先Filter后Interceptor，另外在不同框架中有的是自带，有的是需要自写，具体可以查看开发资料。

57.2 其他知识点



- 1 简称 OGNL，对象导航图语言（Object Graph Navigation Language），是应用于 Java 中的一个开源的表达式语言（Expression Language），它被集成在 Struts2 等框架中，作用是对数据进行访问，它拥有类型转换、访问对象方法、操作集合对象等功能。
- 2 Spring Expression Language（缩写为 SpEL）是一种强大的表达式语言。在 Spring 产品组合中，它是表达式计算的基础。它支持在运行时查询和操作对象图，它可以与基于 XML 和基于注解的 Spring 配置还有 bean 定义一起使用。由于它能够在运行时动态分配值，因此可以为我们节省大量 Java 代码。

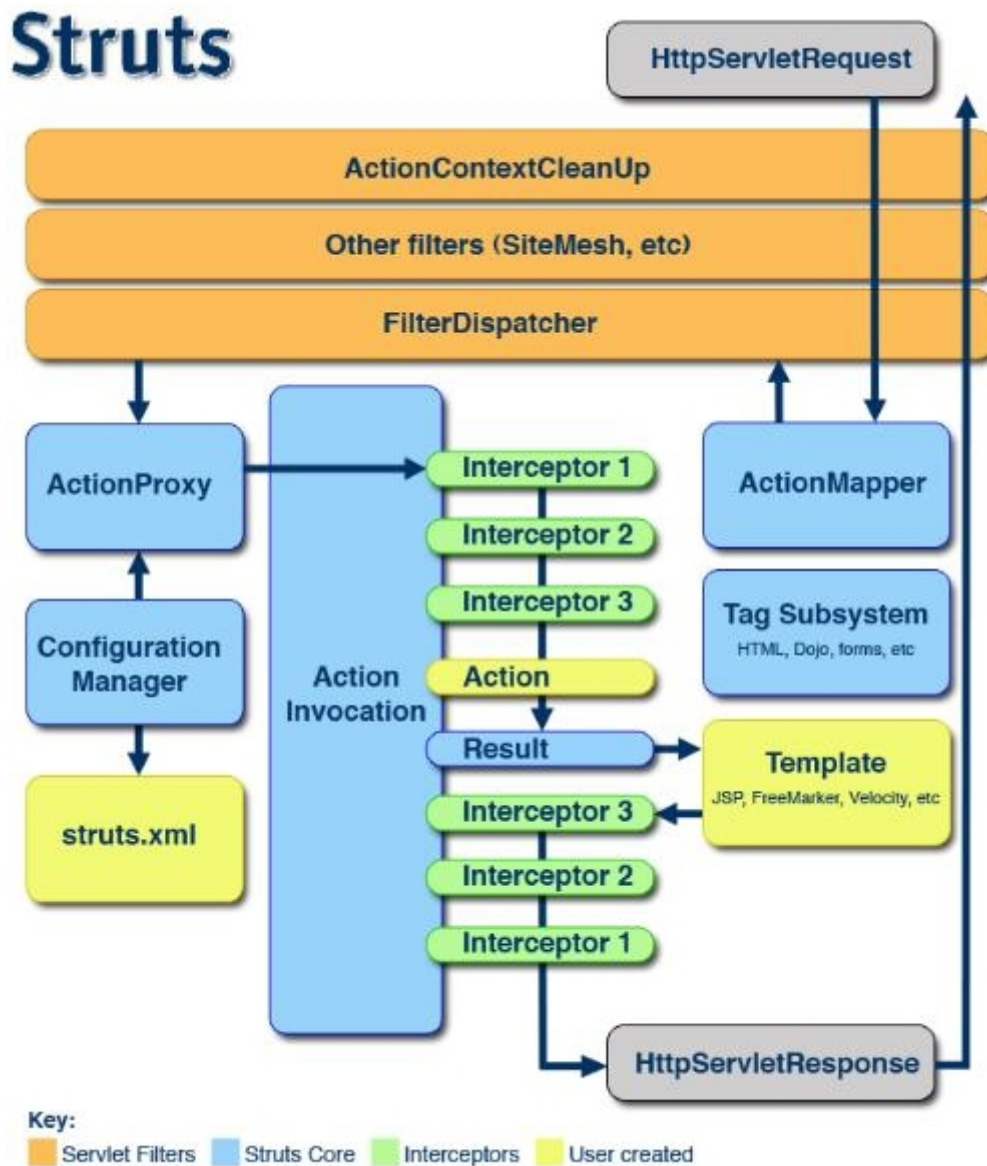


- 1 1、HttpServletRequest 请求信息
- 2 2、ActionContextCleanUp 不重要
- 3 3、Other filters 不重要，现在貌似不用了
- 4 4、Filter Dispatcher 过滤器，这个应该是最底层的过滤器
- 5 5、ActionMapper Struts2中主要检测请求信息是否需要 Struts2处理
- 6 6、ActionProxy 一个中间层，就是可以调用其他类什么的
- 7 7、ConfigurationManger 负责将Struts.xml配置文件映射到内存中去
- 8 8、Struts.xml Struts.xml配置文件需要程序员填写
- 9 9、ActionInvocation 包含四个属性分别获取前端传递的值，action，struts.xml信息，其他一些数据。
- 10 10、Interceptor 拦截器
- 11 11、Tag Subsystem Struts2自带标签库 没用
- 12 12、Templete struts2的前端模板，没用吧，不清楚
- 13 13、HttpServletResponse 响应用户的类

57.3 案例1：Struts2-016远程代码执行漏洞分析-黑盒流程

struts框架已过时

框架执行流程：



57.3.1 漏洞介绍

Struts2 有漏洞，一般在传统企业会用一点，比如：物流，erp等，面试的时候要求你了解Struts的执行流程，真实工作开发中一般用的少一些。BS架构web项目SpringMVC，SpringBoot，SpringCloud 企业中用的相对多一些。

S2-001 远程代码执行漏洞

S2-005 远程代码执行漏洞

S2-007 远程代码执行漏洞

S2-008 远程代码执行漏洞

S2-009 远程代码执行漏洞

S2-012 远程代码执行漏洞

S2-013 远程代码执行漏洞

S2-015 远程代码执行漏洞

S2-016 远程代码执行漏洞

S2-032 远程代码执行漏洞

S2-045 远程代码执行漏洞

S2-046 远程代码执行漏洞

S2-048 远程代码执行漏洞

S2-052 远程代码执行漏洞

中文版本(Chinese version)

The Apache Struts frameworks, when forced, performs double evaluation of attributes' values assigned to certain tags attributes such as id so it is possible to pass in a value that will be evaluated again when a tag's attributes will be rendered. With a carefully crafted request, this can lead to Remote Code Execution (RCE).

Affected Version: Struts 2.0.0 - Struts 2.5.20

References:

- <https://cwiki.apache.org/confluence/display/WW/S2-059>
- <https://securitylab.github.com/research/ognl-apache-struts-exploit-CVE-2018-11776>

Setup

Start the Struts 2.5.16 environment:

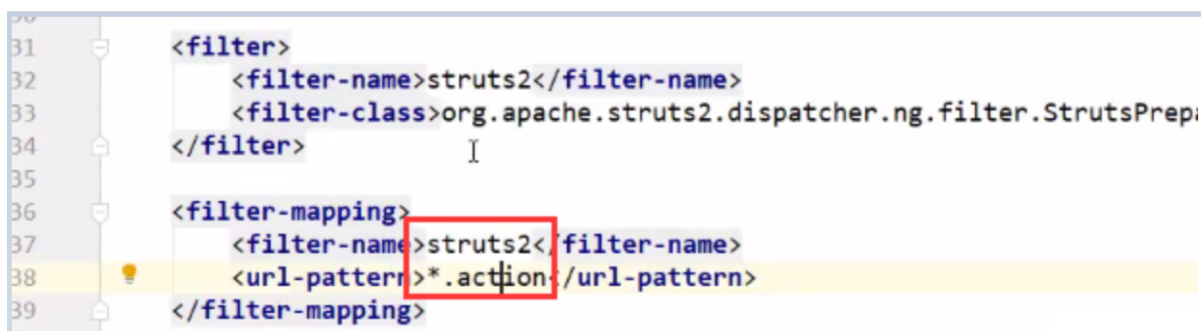
```
docker-compose up -d
```

After the environment is started, visit `http://your-ip:8080/?id=1` and you will see the Struts2 test page.

- 1 本文以S2-016 RCE为例:
- 2 Struts2-016远程代码执行漏洞官方描述:
- 3 struts2的defaultActionMapper支持一种方法, 可以使用“action:”, “redirect:”, “redirectAction:”对输入信息进行处理, 从而改变前缀参数, 这样操作的目的是方便表单中的操作。在2.3.15.1版本以前的struts2中, 没有对“action:”, “redirect:”, “redirectAction:”等进行处理, 导致ognl表达式(类似php里面的eval)可以被执行。

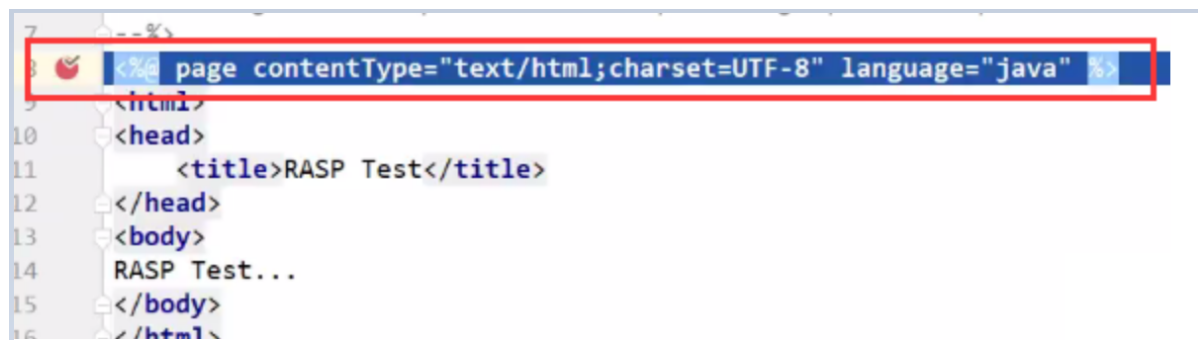
- 1 该漏洞利用st2特性: 1. 路由; 2. ONGL表达式; 3. 执行流程, 断点调试查看执行代码文件分析

1、查看web.xml可知, 所有的.action请求都经过struts2过滤器

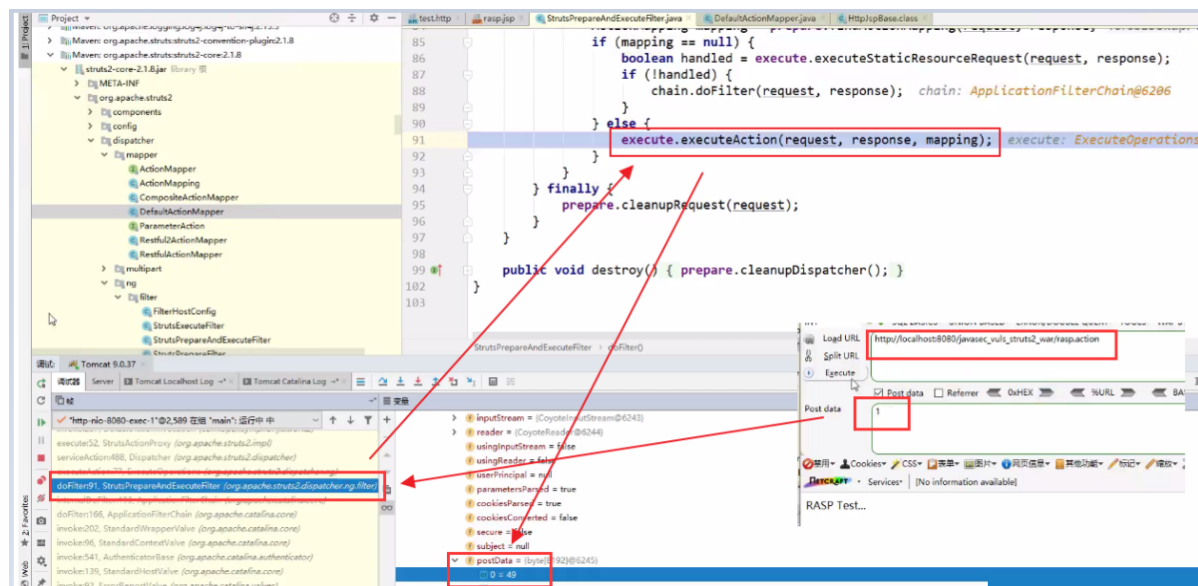


2、断点调试，未触发关键字-断点文件-调试到过滤器

我们在rasp.jsp页面第一行代码处打一个断点

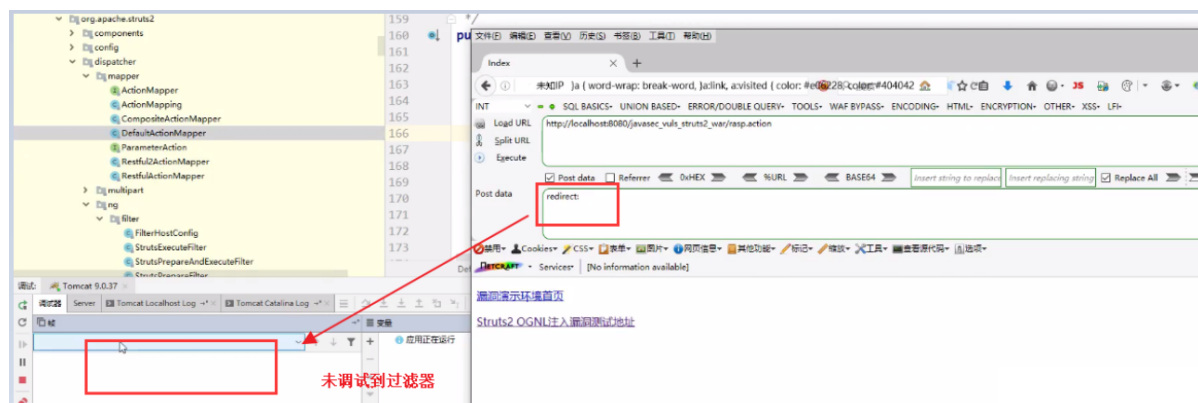


发送一个普通请求（未触发关键字），发现调试到过滤器，执行了execute方法，1的ascii是49



3、断点调试，触发关键字-断点文件-未调试到过滤器

同样在rasp.jsp页面第一行代码处打一个断点，发送一个包含关键字redirect的请求（触发关键字），发现未调试到过滤器，也未执行execute方法



总结漏洞成因，简单来说，结束本来所有.action请求都经过过滤器，执行execute方法，但是由于st2框架自身特性，当请求中有redirect、redirectAction、method、action四个关键字时，请求会绕过过滤器，直接与系统交互，从而造成漏洞。

57.3.2 代码审计

1.获取配置信息

- 1 由源码找过滤器（拦截器），框架包，触发请求，框架特性（ognl，路由等）

2.打开st2对应过滤器，配合st2路由配置文件，解读

- 1 分析开源框架的漏洞还是从其源码入手，问题出在了DefaultActionMapper上，这个类主要是用来处理一些灵活的URL调用，比如处理Action中动态调用方法的形式，如：
- 2 `http://www.foo.com/bar/hello.action?`
`jinyouxin!bar`
- 3 `jinyouxin!bar`这种形式是动态的调用action中的方法，其中jinyouxin是action，bar是方法名，但是调用的前提是在struts.xml中事先进行配置。当然这只是一种，这个类还有个重要的作用就是处理redirect、redirectAction、method、action：
- 4 `redirect`一旦写定，同样不会执行默认action中的execute方法，而是重定向到其他的页面，内部通过ServletRedirectResult完成执行。`redirectAction`同样会屏蔽默认action的方法，而是重定向到其他的Action，同样依靠ServletRedirectResult实现任务。

3.payload加上路由关键字进行OGNL表达式接受处理执行



4.总结审计思路-struts

- 1 已知漏洞：找框架自身的版本，利用漏洞库去确定漏洞
- 2 未知漏洞：执行流程，过滤器，拦截器，框架特性
- 3 验证：采用断点调试，确定.action文件断点后，无法拦截执行请求，说明满足路由规则请求方法后跳出过滤拦截器，绕过实现ongl表达式漏洞。

57.4 案例2：SpringBoot-SpEL表达式注入漏洞分析-白盒思路

57.4.1 SpEL介绍

Spring Expression Language（缩写为SpEL）是一种强大的表达式语言。在Spring产品组合中，它是表达式计算的基础。它支持在运行时查询和操作对象图，它可以与基于XML和基于注解的Spring配置还有bean定义一起使用。由于它能够在运行时动态分配值，因此可以为我们节省大量Java代码。

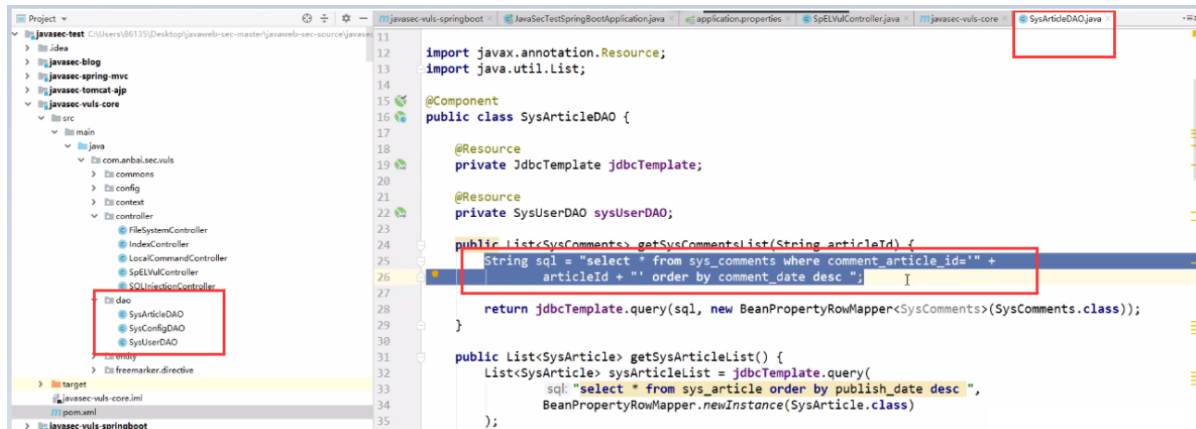
57.4.2 审计思路

- 已知漏洞：找框架自身的版本（xml），利用漏洞库去确定漏洞，有spel表达式注入漏洞
- 未知漏洞：从自身源码分析到文件sql语句中有注入漏洞，尝试成功（未发现过滤器及拦截器）

57.4.3 SpringBoot SpEL表达式注入漏洞-分析与复现

1 <https://www.cnblogs.com/litlife/p/10183137.html>

57.4.4 sql注入漏洞



57.5 总结

- 1.先确定源码中是否存在框架-
 - 执行流程，特性（表达式，拦截器或过滤器自带或自写?）
 - 已知的框架版本漏洞 利用
- 2.确定源码中是否存在过滤器-自带或引用或自定义拦截规则
- 3.过滤器中怎么触发，过滤器规则