

## 0x01 漏洞利用环境

web代码 <https://github.com/spring-projects/spring-mvc-showcase>

中间件 jetty

## 0x02 细节分析

这个漏洞存在下面几个限制条件

1. 要使用file协议打开资源文件目录
2. Windows平台
3. 不能使用Tomcat或者wildfy等中间件

漏洞是出在Spring Framework处理自定义静态文件的功能处。通过阅读官方手册得知，在Spring Framework中有两种方式去定义资源文件。一种是配置XML文件，经常可以在Spring项目中看到如下配置

```
<mvc:resources mapping="/resources/**" location="/public, classpath:/static/" cache-period="31556926" />
```

另外一种方式就是通过重写WebMvcConfigurer中的addResourceHandlers方法来添加新的资源文件路径。其中使用addResourceLocations属性来设置资源文件内容file等协议的。例如如下代码

```
registry.addResourceHandler("/resources/**").addResourceLocations("http://www.resources.com/", "/resources/");
```

其含义就是你在访问<http://www.xxx.com/resources/1.txt>时，Spring会去寻找 www.resources.com 是否存在此文件。当然file也同理。我们使用spring自己提供的demo，只要org.springframework.samples.mvc.config.WebMvcConfig添加以下代码即可

```
registry.addResourceHandler("/resources/**").addResourceLocations("file:///D:/static/", "/resources/");
```

当前漏洞环境在D盘，我在D盘根目录创建一个123.txt，通过下面的URL即可访问。

<http://127.0.0.1:8080/spring-mvc-showcase/resources/xxxx/..%5c/..%5c/..%5c/123.txt>



123

漏洞只能在windows上存在原因是因为org.springframework.web.servlet.resource.ResourceHttpRequestHandler中的String path = (String)

request.getAttribute(HandlerMapping.PATH\_WITHIN\_HANDLER\_MAPPING\_ATTRIBUTE);会将URL中的..%2f去除,然后在Windows中能通过..\的方式来跳目录%5c(/)的URL直接http 400，若需要处理此类型的URL

则需要在tomcat配置文件中添加Dorg.apache.tomcat.util.buf.UDecoder.ALLOW\_ENCODED\_SLASH=true Spring Framework处理资源文件的整个流程如下图：

```
getFile:86, FileUrlResource (org.springframework.core.io)
exists:52, AbstractFileResolvingResource (org.springframework.core.io)
getResource:185, PathResourceResolver (org.springframework.web.servlet.resource)
getResource:156, PathResourceResolver (org.springframework.web.servlet.resource)
resolveResourceInternal:135, PathResourceResolver (org.springframework.web.servlet.resource)
resolveResource:48, AbstractResourceResolver (org.springframework.web.servlet.resource)
resolveResource:61, DefaultResourceResolverChain (org.springframework.web.servlet.resource)
getResource:535, ResourceHttpRequestHandler (org.springframework.web.servlet.resource)
handleRequest:439, ResourceHttpRequestHandler (org.springframework.web.servlet.resource)
handle:53, HttpRequestHandlerAdapter (org.springframework.web.servlet.mvc)
doDispatch:991, DispatcherServlet (org.springframework.web.servlet)
doService:925, DispatcherServlet (org.springframework.web.servlet)
processRequest:978, FrameworkServlet (org.springframework.web.servlet)
doGet:870, FrameworkServlet (org.springframework.web.servlet)
service:687, HttpServlet (javax.servlet.http)
service:855, FrameworkServlet (org.springframework.web.servlet)
service:790, HttpServlet (javax.servlet.http)
handle:864, ServletHolder (org.eclipse.jetty.servlet)
doFilter:1655, ServletHandler$CachedChain (org.eclipse.jetty.servlet)
doFilterInternal:100, CsrfFilter (org.springframework.security.web.csrf)
doFilter:107, OncePerRequestFilter (org.springframework.web.filter)
invokeDelegate:357, DelegatingFilterProxy (org.springframework.web.filter)
doFilter:270, DelegatingFilterProxy (org.springframework.web.filter)
doFilter:1642, ServletHandler$CachedChain (org.eclipse.jetty.servlet)
doFilter:215, WebSocketUpgradeFilter (org.eclipse.jetty.websocket.server)
doFilter:1642, ServletHandler$CachedChain (org.eclipse.jetty.servlet)
doHandle:533, ServletHandler (org.eclipse.jetty.servlet)
handle:146, ScopedHandler (org.eclipse.jetty.server.handler)
```



在经过一系列判断之后在org.springframework.util.ResourceUtils中的getFile方法读取了文件。值得注意的是虽然在存在URL检查中存在如下代码

```
if (path.contains("WEB-INF") || path.contains("META-INF")) {
    if (logger.isTraceEnabled()) {
        logger.trace("Path contains \"WEB-INF\" or \"META-INF\".");
    }
    return true;
}
```

但是在Windows平台下是不会区分大小写的，所以还是可以读取配置文件。但是如何构造URL得看在配置资源文件中设置的location是在何处了。

This XML file does not appear to have any style information associated with it. The document tree is shown below.

<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.springframework.org/schema/bear



官方修复方式是在CheckResource方法中添加了如下过滤

```
protected String processPath(String path) {  
    path = StringUtils.replace(path, "\\ ", "/");  
    path = cleanDuplicateSlashes(path);  
    return cleanLeadingSlash(path);  
}
```

替换了\之后都成为了../ 自然就不能跨目录读文件了。

### 0x03 参考

<https://docs.spring.io/spring-framework/docs/5.0.4.RELEASE/spring-framework-reference/web.html#mvc-config-static-resources>  
<https://pivotal.io/security/cve-2018-1271>  
<https://threathunter.org/topic/5acc0902ec721b1f1966f582>

点击收藏 | 2 关注 | 1

[上一篇：浅谈OAuth2.0协议安全性](#) [下一篇：利用Angr分析恶意软件的通信协议](#)

1. 0 条回复

- 动动手指，沙发就是你的了！

[登录](#) 后跟帖

先知社区

[现在登录](#)

[技术文章](#)

[社区小黑板](#)

[目录](#)

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)