目录

[一、 暴力破解](#_Toc4980_WPSOffice_Level1) [4](#_Toc4980_WPSOffice_Level1)

[1. 基于表单的暴力破解](#_Toc24208_WPSOffice_Level2) [4](#_Toc24208_WPSOffice_Level2)

[2. 验证码绕过（on server）](#_Toc18558_WPSOffice_Level2) [4](#_Toc18558_WPSOffice_Level2)

[3. 验证码绕过（on client）](#_Toc2594_WPSOffice_Level2) [4](#_Toc2594_WPSOffice_Level2)

[4. Token防爆破](#_Toc27804_WPSOffice_Level2) [4](#_Toc27804_WPSOffice_Level2)

[(1) 什么是Token？](#_Toc24208_WPSOffice_Level3) [4](#_Toc24208_WPSOffice_Level3)

[(2) 绕过思路](#_Toc18558_WPSOffice_Level3) [4](#_Toc18558_WPSOffice_Level3)

[(3) 具体步骤](#_Toc2594_WPSOffice_Level3) [4](#_Toc2594_WPSOffice_Level3)

[第一个Payload设置载入密码字典](#_Toc7657_WPSOffice_Level2) [6](#_Toc7657_WPSOffice_Level2)

[二、 XSS](#_Toc24208_WPSOffice_Level1) [9](#_Toc24208_WPSOffice_Level1)

[1. 反射型（get）](#_Toc21249_WPSOffice_Level2) [9](#_Toc21249_WPSOffice_Level2)

[(1) 反射型XSS](#_Toc27804_WPSOffice_Level3) [9](#_Toc27804_WPSOffice_Level3)

[(2) 具体步骤](#_Toc7657_WPSOffice_Level3) [9](#_Toc7657_WPSOffice_Level3)

[2. 反射型（post）](#_Toc7982_WPSOffice_Level2) [9](#_Toc7982_WPSOffice_Level2)

[(1) 具体步骤](#_Toc21249_WPSOffice_Level3) [9](#_Toc21249_WPSOffice_Level3)

[3. 存储型](#_Toc25853_WPSOffice_Level2) [10](#_Toc25853_WPSOffice_Level2)

[(1) 存储型](#_Toc7982_WPSOffice_Level3) [10](#_Toc7982_WPSOffice_Level3)

[(2) 具体步骤](#_Toc25853_WPSOffice_Level3) [10](#_Toc25853_WPSOffice_Level3)

[4. DOM型xss](#_Toc3710_WPSOffice_Level2) [10](#_Toc3710_WPSOffice_Level2)

[(1) DOM型](#_Toc3710_WPSOffice_Level3) [10](#_Toc3710_WPSOffice_Level3)

[(2) 具体步骤](#_Toc29536_WPSOffice_Level3) [10](#_Toc29536_WPSOffice_Level3)

[5. DOM型xss-x](#_Toc29536_WPSOffice_Level2) [11](#_Toc29536_WPSOffice_Level2)

[(1) 具体步骤](#_Toc27134_WPSOffice_Level3) [11](#_Toc27134_WPSOffice_Level3)

[6. xss之盲打](#_Toc27134_WPSOffice_Level2) [12](#_Toc27134_WPSOffice_Level2)

[(1) 提示](#_Toc31236_WPSOffice_Level3) [12](#_Toc31236_WPSOffice_Level3)

[(2) 具体步骤](#_Toc833_WPSOffice_Level3) [12](#_Toc833_WPSOffice_Level3)

[7. Xss之过滤](#_Toc31236_WPSOffice_Level2) [13](#_Toc31236_WPSOffice_Level2)

[(1) 过滤<script>](#_Toc15568_WPSOffice_Level3) [13](#_Toc15568_WPSOffice_Level3)

[(2) 具体步骤](#_Toc19653_WPSOffice_Level3) [13](#_Toc19653_WPSOffice_Level3)

[8. Xss之htmlspecialchars](#_Toc833_WPSOffice_Level2) [14](#_Toc833_WPSOffice_Level2)

[(1) specialchars函数](#_Toc29099_WPSOffice_Level3) [14](#_Toc29099_WPSOffice_Level3)

[(2) 具体步骤](#_Toc9363_WPSOffice_Level3) [14](#_Toc9363_WPSOffice_Level3)

[9. Xss之href输出](#_Toc15568_WPSOffice_Level2) [15](#_Toc15568_WPSOffice_Level2)

[(1) 闭合a标签的href属性](#_Toc1853_WPSOffice_Level3) [15](#_Toc1853_WPSOffice_Level3)

[10. Xss之js输出](#_Toc19653_WPSOffice_Level2) [15](#_Toc19653_WPSOffice_Level2)

[(1) js标签](#_Toc3213_WPSOffice_Level3) [15](#_Toc3213_WPSOffice_Level3)

[三、 CSFR](#_Toc18558_WPSOffice_Level1) [17](#_Toc18558_WPSOffice_Level1)

[1. CSFR（GET）](#_Toc29099_WPSOffice_Level2) [17](#_Toc29099_WPSOffice_Level2)

[(1) 具体步骤](#_Toc22120_WPSOffice_Level3) [17](#_Toc22120_WPSOffice_Level3)

[2. CSFR（POST）](#_Toc9363_WPSOffice_Level2) [18](#_Toc9363_WPSOffice_Level2)

[(1) 具体步骤](#_Toc1640_WPSOffice_Level3) [18](#_Toc1640_WPSOffice_Level3)

[3. CSFR Token](#_Toc1853_WPSOffice_Level2) [21](#_Toc1853_WPSOffice_Level2)

[(1) Token验证](#_Toc15585_WPSOffice_Level3) [21](#_Toc15585_WPSOffice_Level3)

[四、 SQL注入](#_Toc2594_WPSOffice_Level1) [22](#_Toc2594_WPSOffice_Level1)

[1. 数字型（POST）](#_Toc3213_WPSOffice_Level2) [22](#_Toc3213_WPSOffice_Level2)

[(1) 具体步骤](#_Toc14456_WPSOffice_Level3) [22](#_Toc14456_WPSOffice_Level3)

[2. 字符型（GET）](#_Toc22120_WPSOffice_Level2) [24](#_Toc22120_WPSOffice_Level2)

[(1) 具体步骤](#_Toc20828_WPSOffice_Level3) [24](#_Toc20828_WPSOffice_Level3)

[3. 搜索型](#_Toc1640_WPSOffice_Level2) [25](#_Toc1640_WPSOffice_Level2)

[(1) 具体步骤](#_Toc27046_WPSOffice_Level3) [25](#_Toc27046_WPSOffice_Level3)

[4. xx型](#_Toc15585_WPSOffice_Level2) [26](#_Toc15585_WPSOffice_Level2)

[(1) 闭合方式的判断](#_Toc9750_WPSOffice_Level3) [26](#_Toc9750_WPSOffice_Level3)

[5. “insert/update”注入](#_Toc14456_WPSOffice_Level2) [26](#_Toc14456_WPSOffice_Level2)

[(1) insert语句注入](#_Toc5708_WPSOffice_Level3) [26](#_Toc5708_WPSOffice_Level3)

[(2) 具体步骤](#_Toc29901_WPSOffice_Level3) [26](#_Toc29901_WPSOffice_Level3)

[6. “delete”注入](#_Toc20828_WPSOffice_Level2) [27](#_Toc20828_WPSOffice_Level2)

[(1) 具体步骤](#_Toc11471_WPSOffice_Level3) [27](#_Toc11471_WPSOffice_Level3)

[7. “http header”注入](#_Toc27046_WPSOffice_Level2) [28](#_Toc27046_WPSOffice_Level2)

[(1) http头注入](#_Toc10088_WPSOffice_Level3) [28](#_Toc10088_WPSOffice_Level3)

[(2) 具体步骤](#_Toc14382_WPSOffice_Level3) [28](#_Toc14382_WPSOffice_Level3)

[8. 布尔盲注](#_Toc9750_WPSOffice_Level2) [29](#_Toc9750_WPSOffice_Level2)

[(1) 具体步骤](#_Toc2774_WPSOffice_Level3) [29](#_Toc2774_WPSOffice_Level3)

[9. 时间盲注](#_Toc5708_WPSOffice_Level2) [30](#_Toc5708_WPSOffice_Level2)

[(1) 具体步骤](#_Toc12256_WPSOffice_Level3) [30](#_Toc12256_WPSOffice_Level3)

[10. 宽字节注入](#_Toc29901_WPSOffice_Level2) [30](#_Toc29901_WPSOffice_Level2)

[(1) 具体步骤](#_Toc9711_WPSOffice_Level3) [31](#_Toc9711_WPSOffice_Level3)

[五、 RCE远程执行](#_Toc27804_WPSOffice_Level1) [31](#_Toc27804_WPSOffice_Level1)

[1. Exec](#_Toc11471_WPSOffice_Level2) [31](#_Toc11471_WPSOffice_Level2)

[(1) 命令执行](#_Toc8110_WPSOffice_Level3) [31](#_Toc8110_WPSOffice_Level3)

[(2) 具体步骤](#_Toc15453_WPSOffice_Level3) [31](#_Toc15453_WPSOffice_Level3)

[2. Eval](#_Toc10088_WPSOffice_Level2) [32](#_Toc10088_WPSOffice_Level2)

[(1) 代码执行](#_Toc12153_WPSOffice_Level3) [32](#_Toc12153_WPSOffice_Level3)

[(2) 具体步骤](#_Toc1500_WPSOffice_Level3) [32](#_Toc1500_WPSOffice_Level3)

[六、 文件包含](#_Toc7657_WPSOffice_Level1) [32](#_Toc7657_WPSOffice_Level1)

[1. 本地包含](#_Toc14382_WPSOffice_Level2) [32](#_Toc14382_WPSOffice_Level2)

[(1) 读取目录下隐藏文件](#_Toc4612_WPSOffice_Level3) [32](#_Toc4612_WPSOffice_Level3)

[(2) 读取系统文件](#_Toc9286_WPSOffice_Level3) [34](#_Toc9286_WPSOffice_Level3)

[(3) 执行getshell](#_Toc23357_WPSOffice_Level3) [34](#_Toc23357_WPSOffice_Level3)

[2. 远程包含](#_Toc2774_WPSOffice_Level2) [34](#_Toc2774_WPSOffice_Level2)

[(1) 读取系统文件](#_Toc30545_WPSOffice_Level3) [34](#_Toc30545_WPSOffice_Level3)

[(2) php伪协议的利用](#_Toc409_WPSOffice_Level3) [35](#_Toc409_WPSOffice_Level3)

[(3) 包含木马文件](#_Toc8934_WPSOffice_Level3) [36](#_Toc8934_WPSOffice_Level3)

[七、 文件下载](#_Toc21249_WPSOffice_Level1) [36](#_Toc21249_WPSOffice_Level1)

[八、 文件上传](#_Toc7982_WPSOffice_Level1) [36](#_Toc7982_WPSOffice_Level1)

[九、 越权](#_Toc25853_WPSOffice_Level1) [36](#_Toc25853_WPSOffice_Level1)

[十、 目录遍历](#_Toc3710_WPSOffice_Level1) [36](#_Toc3710_WPSOffice_Level1)

[十一、 敏感信息泄露](#_Toc29536_WPSOffice_Level1) [36](#_Toc29536_WPSOffice_Level1)

[十二、 PHP反序列化](#_Toc27134_WPSOffice_Level1) [36](#_Toc27134_WPSOffice_Level1)

[十三、 XXE](#_Toc31236_WPSOffice_Level1) [36](#_Toc31236_WPSOffice_Level1)

[十四、 URL重定向](#_Toc833_WPSOffice_Level1) [36](#_Toc833_WPSOffice_Level1)

[十五、 SSFR](#_Toc15568_WPSOffice_Level1) [36](#_Toc15568_WPSOffice_Level1)

1. 暴力破解
2. 基于表单的暴力破解
3. 验证码绕过（on server）
4. 验证码绕过（on client）
5. Token防爆破
   1. 什么是Token？

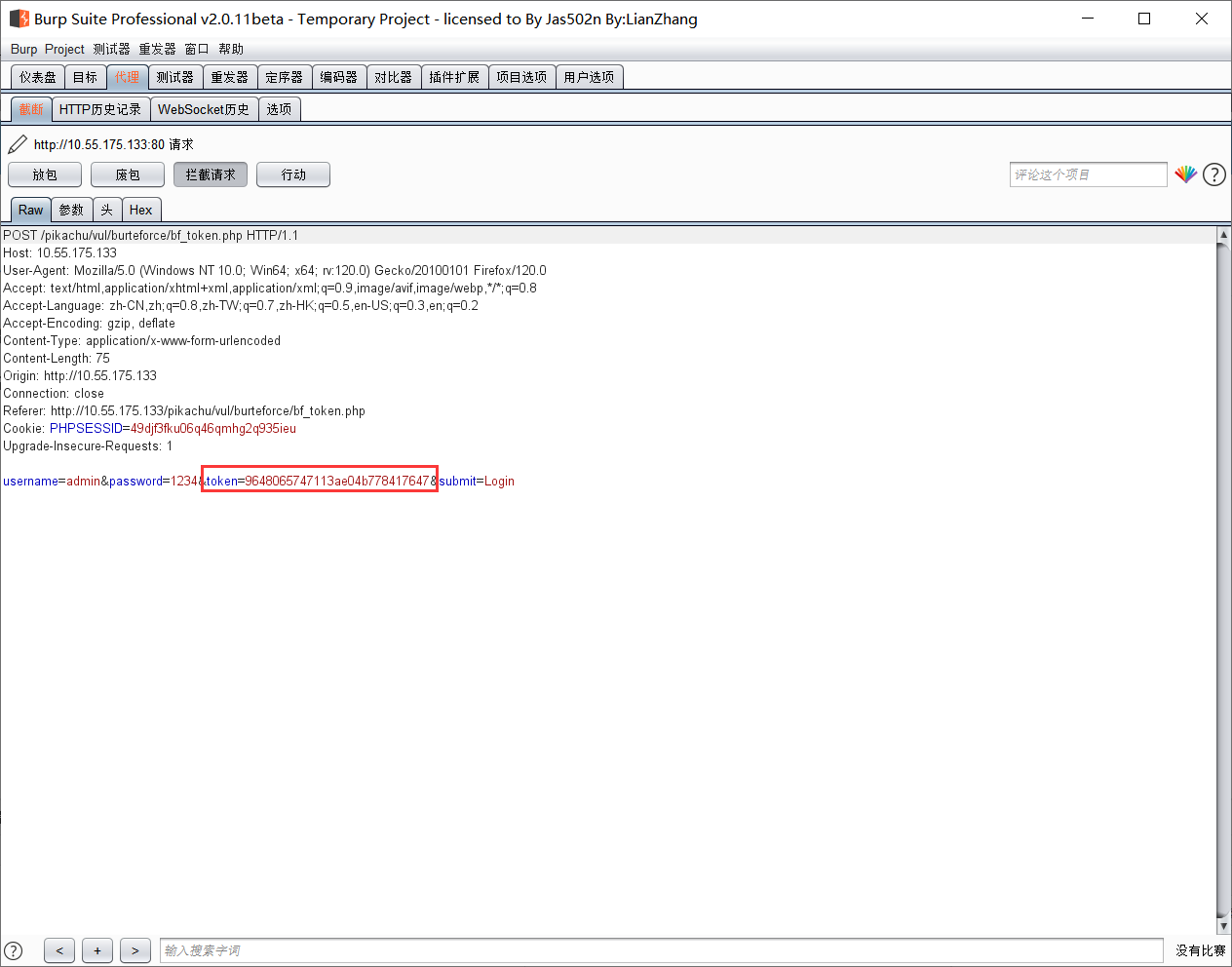
服务器给前端发的令牌，前端向服务器发送请求时都要带上这个身份证，服务器通过这个身份证来判断是否是合法请求

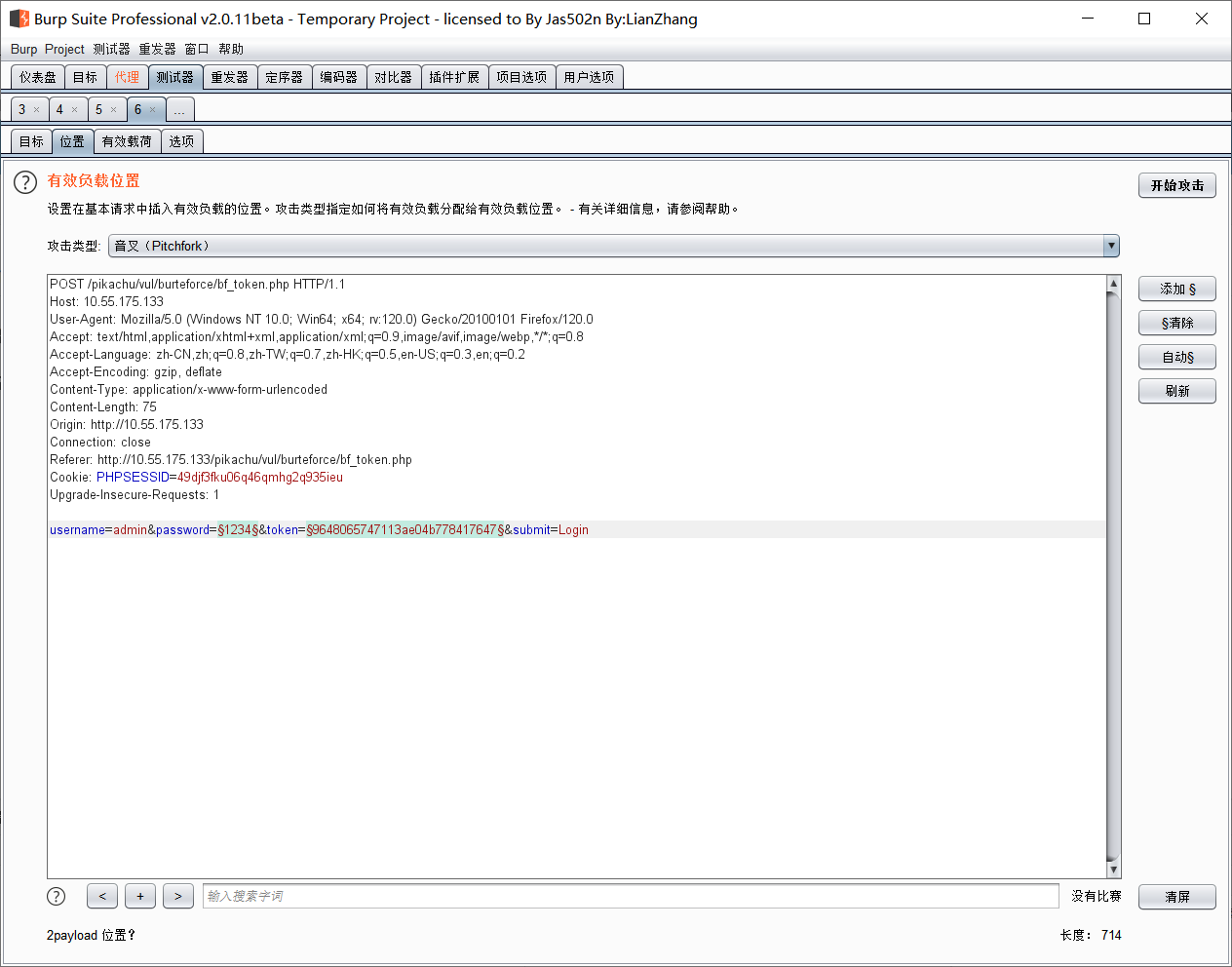
* 1. 绕过思路

通过Burp Suite的Intruder功能配置一些参数来绕过。

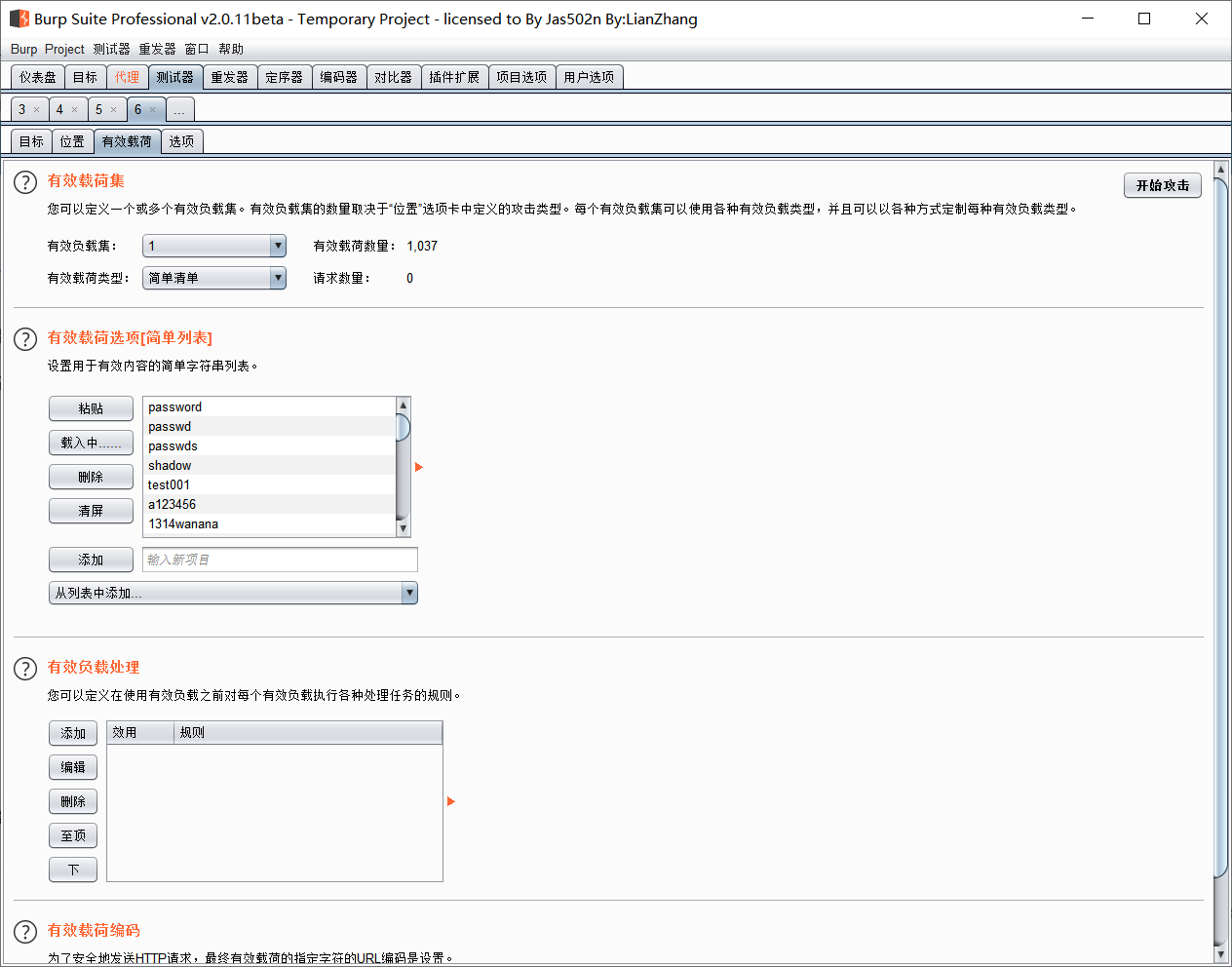
* 1. 具体步骤

使用Burp Suite截取数据包，发现多了token关键字

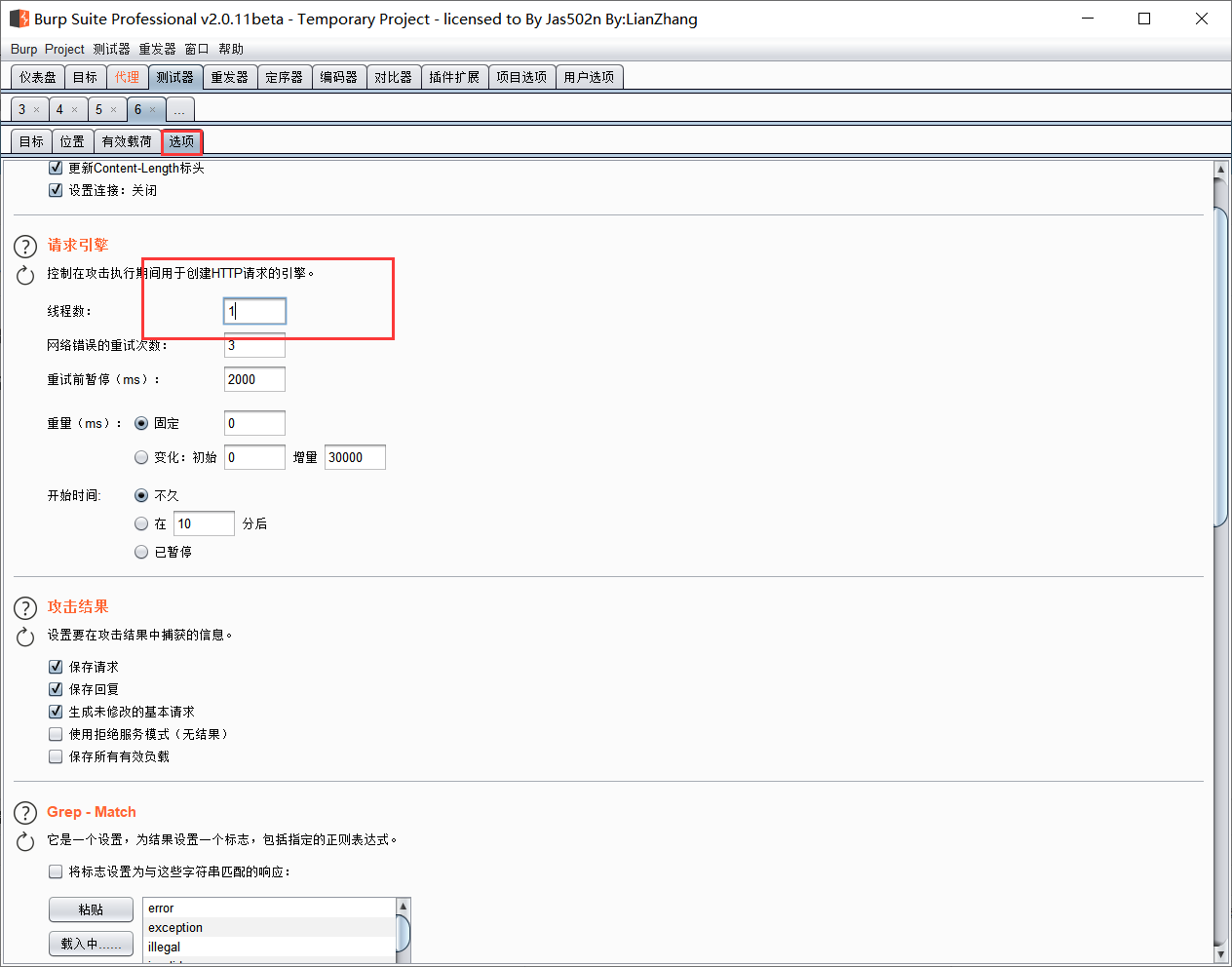


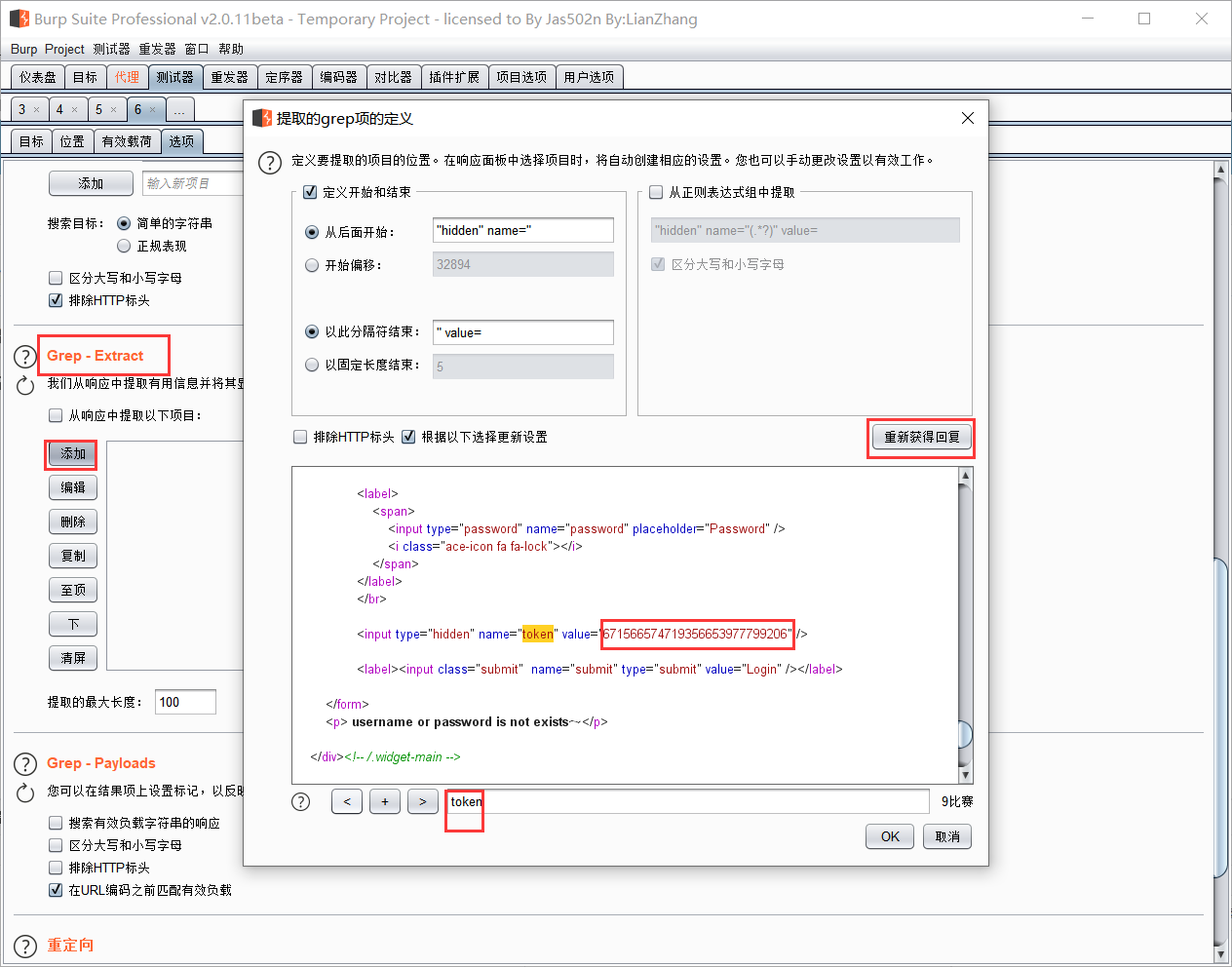
将数据包发送到Intrduer模块,将password和token参数设为爆破值，攻击类型设置音叉（Pitch fork）。

第一个Payload设置载入密码字典

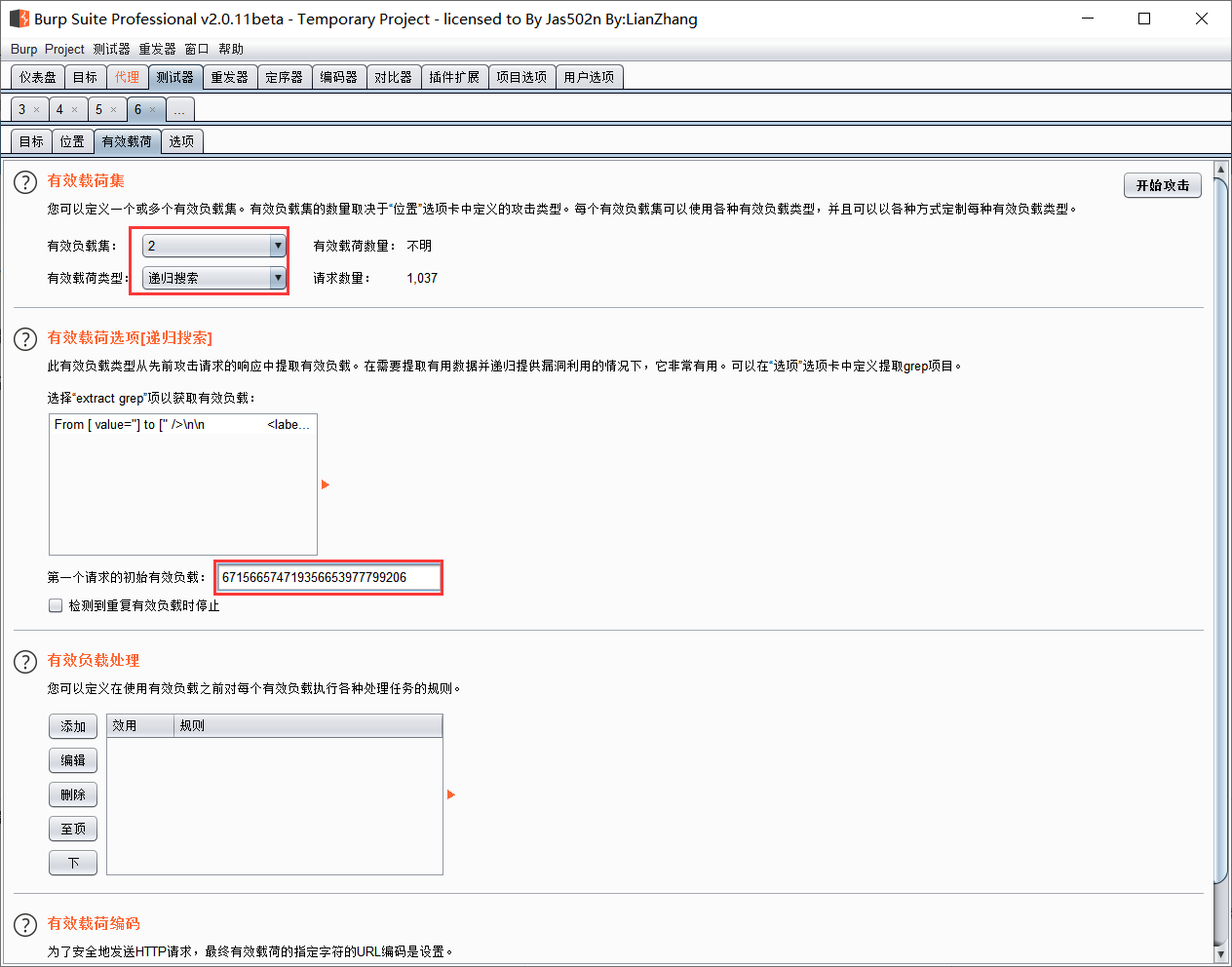


设置攻击线程数为1

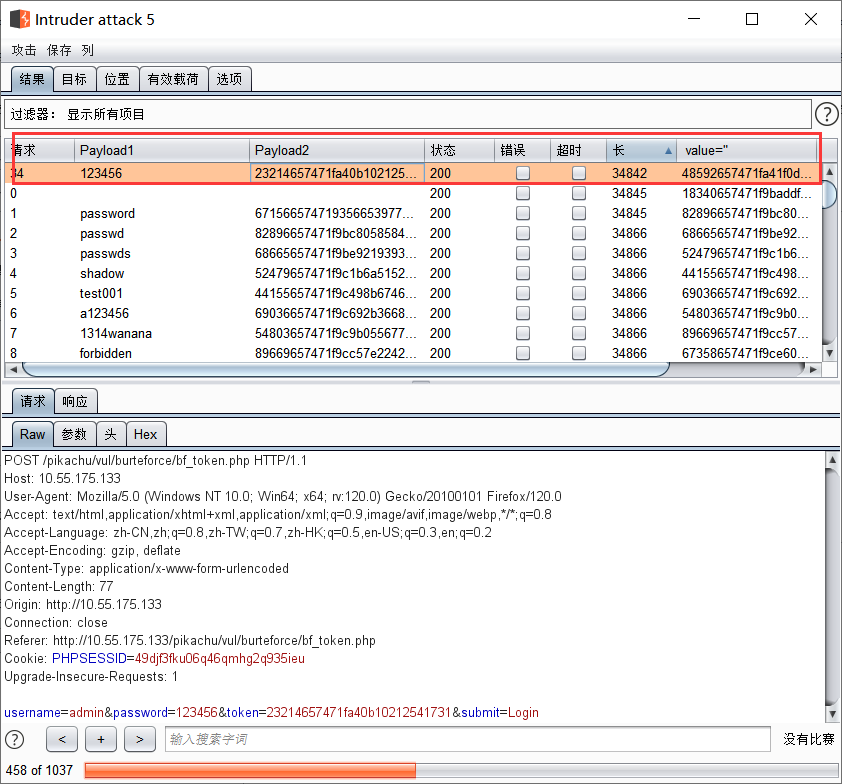


在Grep-Extract中，选择添加—获得回复—选中token的值复制，然后点击OK。

在Payload2的位置上，选择载荷类型为递归搜索，并且将复制的671566574719356653977799206粘贴到如图位置。



开始攻击，选择长度排序，找到被爆破出的密码了

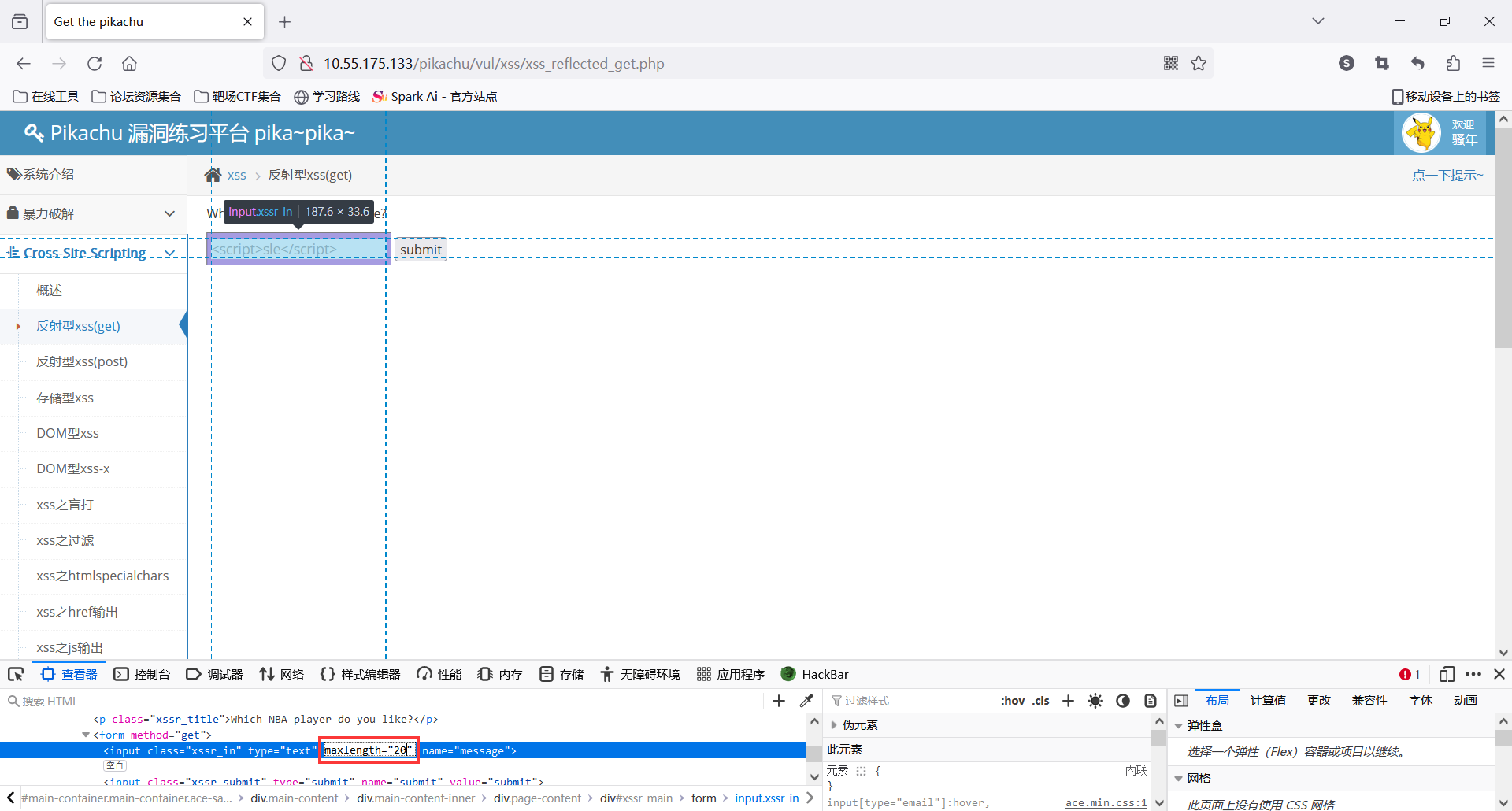


1. XSS
2. 反射型（get）
   1. 反射型XSS

又称非持久型XSS。之所以称为反射型XSS，是因为这种攻击方式的注入代码是从目标服务器通过错误信息、搜索结果等等方式“反射”回来的：发出请求时，XSS代码出现在URL中，作为输入提交到服务器端，服务器端解析后响应，XSS代码随响应内容一起传回给浏览器，最后浏览器解析执行XSS代码。

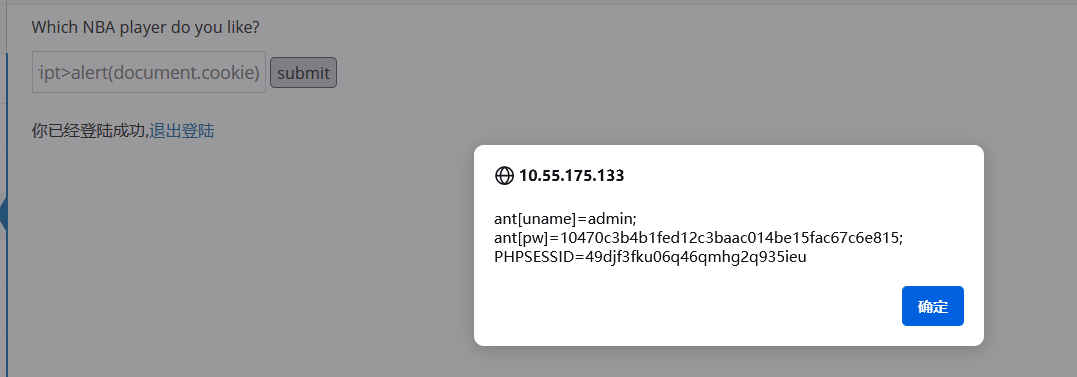
* 1. 具体步骤

这个input输入框，在前端做了限制输入数量的限制，修改maxlength的值或者直接删除即可。



1. 反射型（post）
   1. 具体步骤

先登录，输入弹Cookie的XSS代码<script>alert(document.cookie)</script>

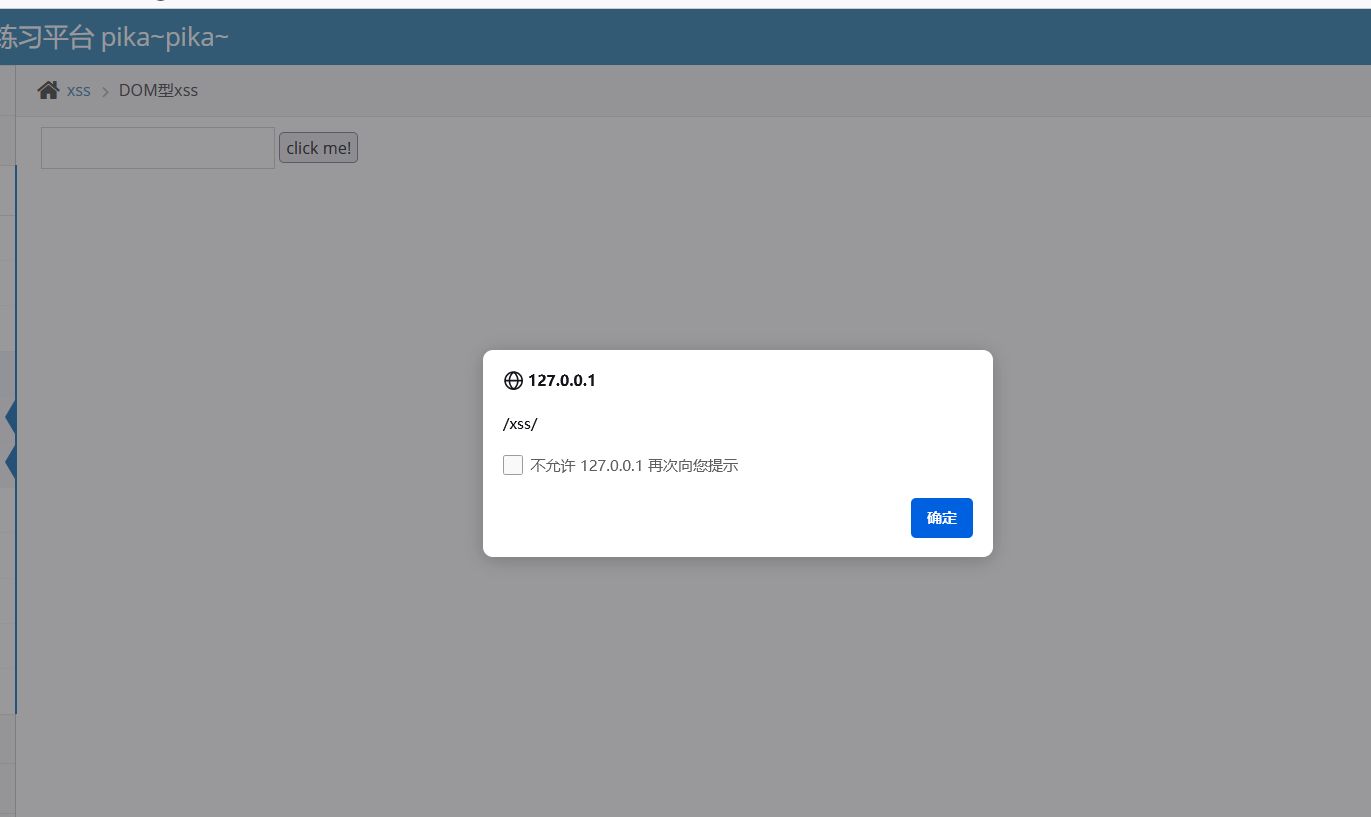


1. 存储型
   1. 存储型

又称持久型XSS，他和反射型XSS最大的不同就是，攻击脚本将被永久地存放在目标服务器端（数据库，内存，文件系统等），下次请求目标页面时不用再提交XSS代码。

* 1. 具体步骤

在留言板中输入XSS代码，下次访问留言板时会自动触发代码



1. DOM型xss
   1. DOM型

向文档对象传入xss代码参数，然后操作文档对象时就会触发xss攻击

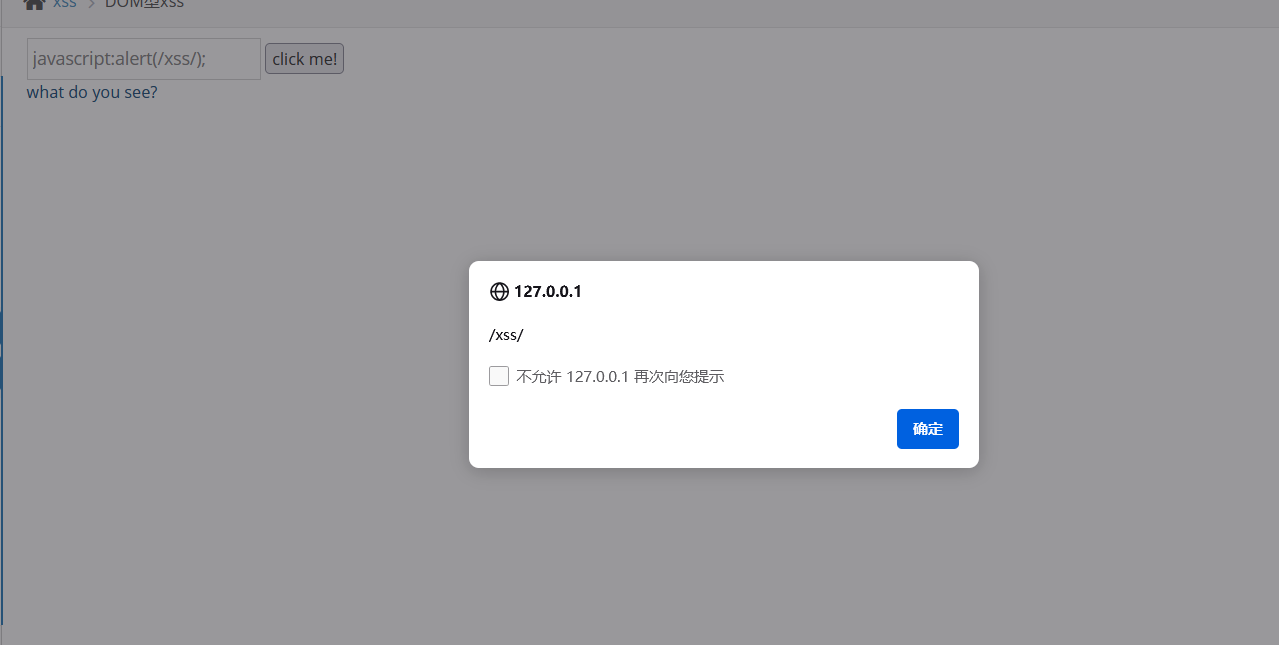
* 1. 具体步骤

在input框中输入XSS代码发现不能触发XSS，观察前端代码发现，xss代码被作为href的参数构成超链接了



现在我们需要构造payload先把href给闭合掉再执行XSS代码，这里我们使用利用javascript伪协议的方式构造xss，

<a href=”javascript:alert(/xss/);”>点我</a>点击超链接即可触发

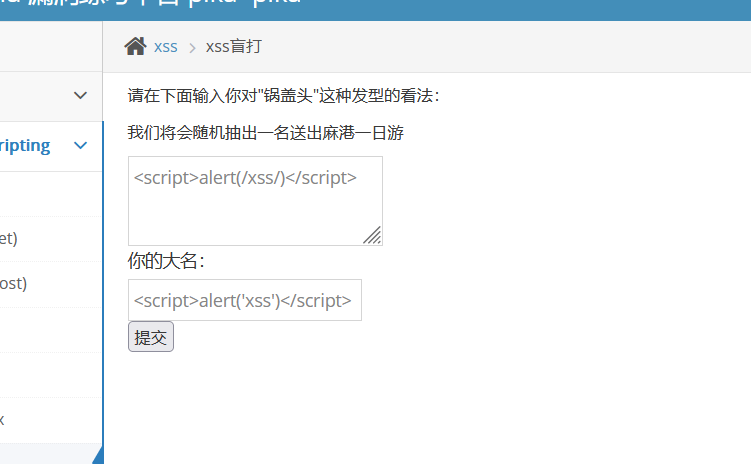


1. DOM型xss-x
2. xss之盲打
   1. 提示

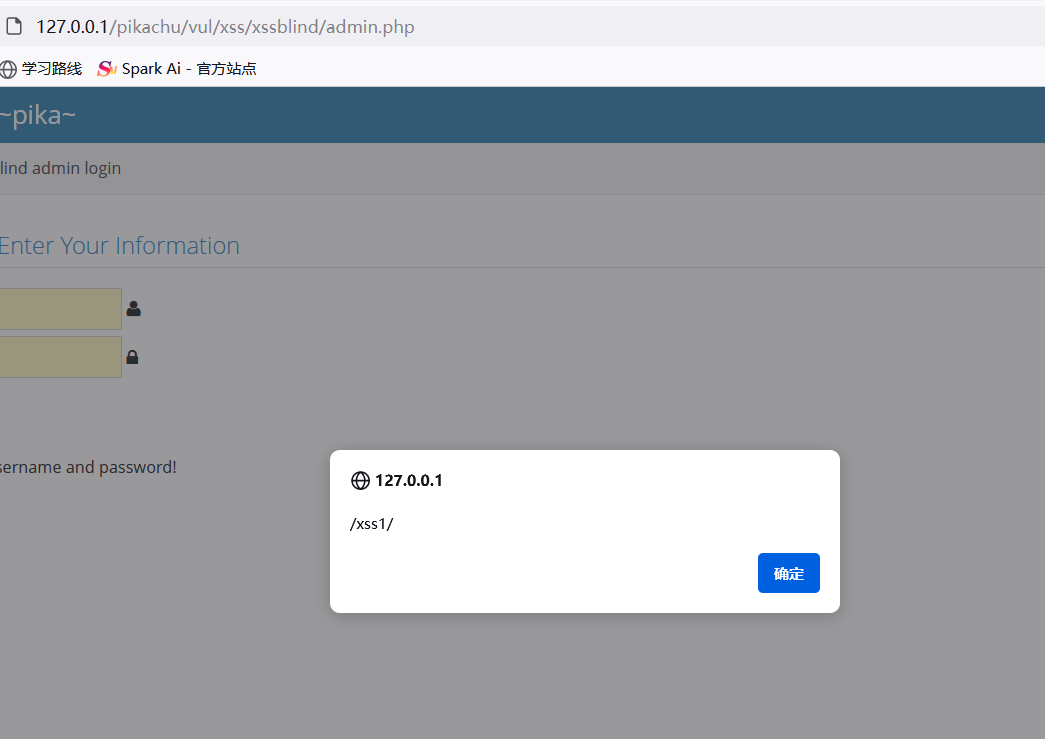
这里的盲打主要是不知道是反射型还是存储型，让我们见到输入框就输入XSS代码，管理员在后台会触发XSS代码

* 1. 具体步骤

在两个输入框中都输入XSS代码



管理员在后台会触发xss代码

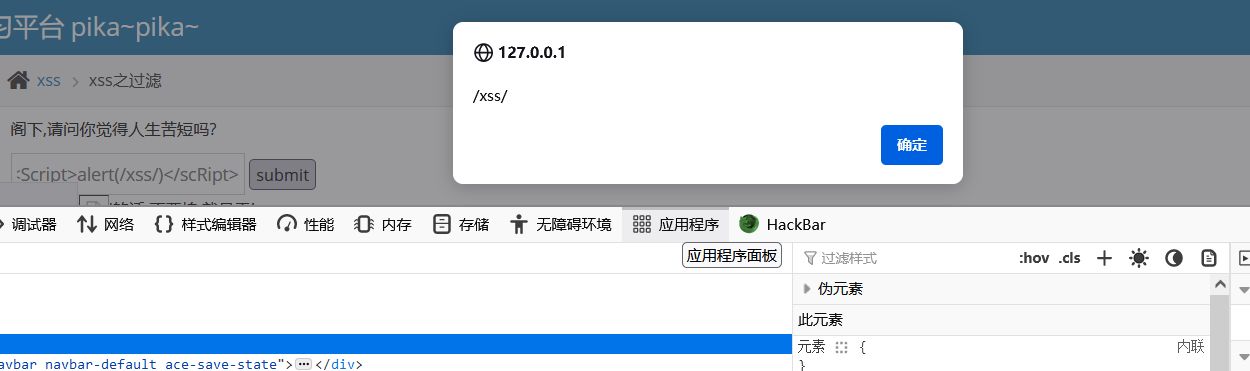


1. Xss之过滤
   1. 过滤<script>

输入<script>alert(/xss/)</script>之后，发现只剩下“ > ”了，<script>关键字被过滤了

* 1. 具体步骤

大小写绕过<Script>alert(/xss/)</scRipt>



使用非script标签，例如

<img src=1 onerror=alert(/xss/);>

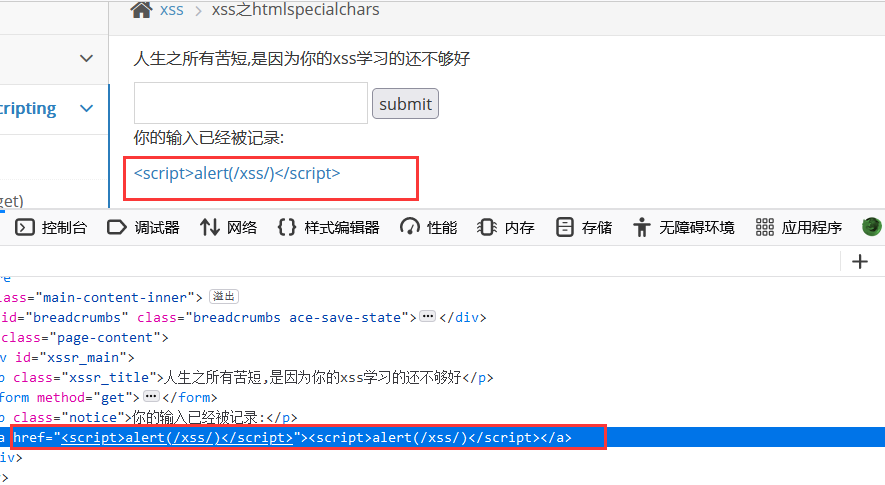


1. Xss之htmlspecialchars
   1. specialchars函数

htmlspecialchars() 函数把预定义的字符转换为 HTML 实体，同时会过滤单引号、双引号、尖括号、&几种符号

* 1. 具体步骤

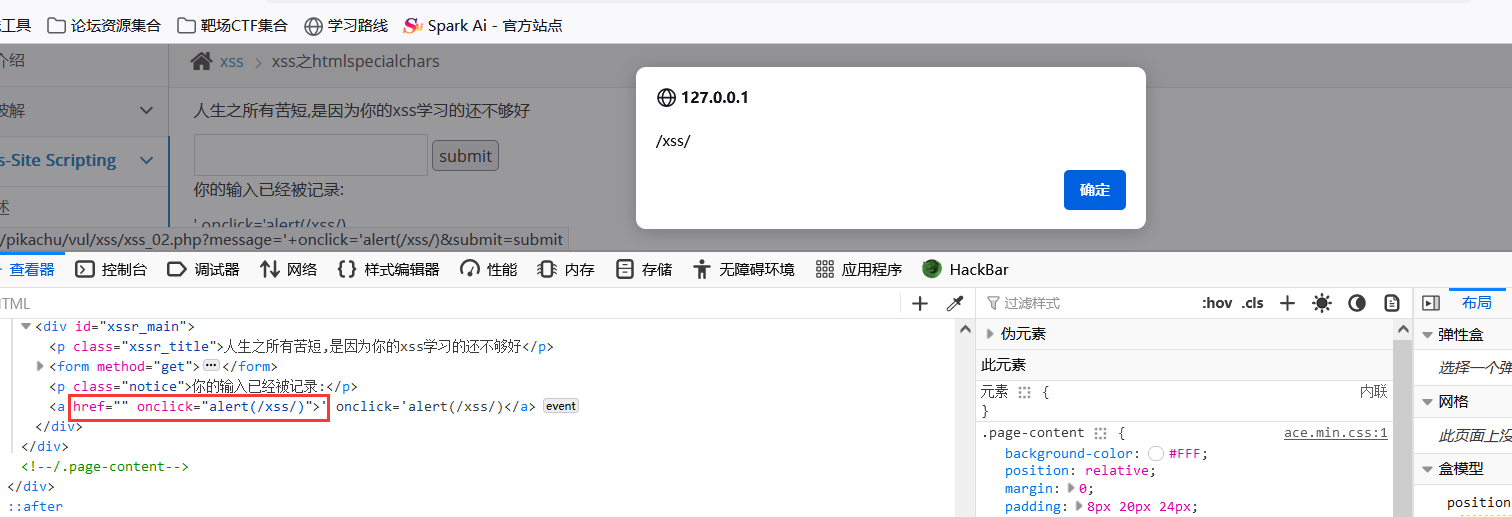
输入框输入的值会被作为a标签的href属性



使用第四题的payload可以直接执行XSS代码



或者修改第五题的payload也可以，' onclick='alert(/xss/)



* 1. 具体步骤

和上一个步骤一样



利用JS事件构造payload为’ onclick=alert('xss')>

或者' onclick='alert(/xss/)



1. Xss之href输出
   1. 闭合a标签的href属性

参考四、五题目,但是这里过滤了单引号，于是只有js伪协议能用，但我还是把三种闭合href的方式都列出来

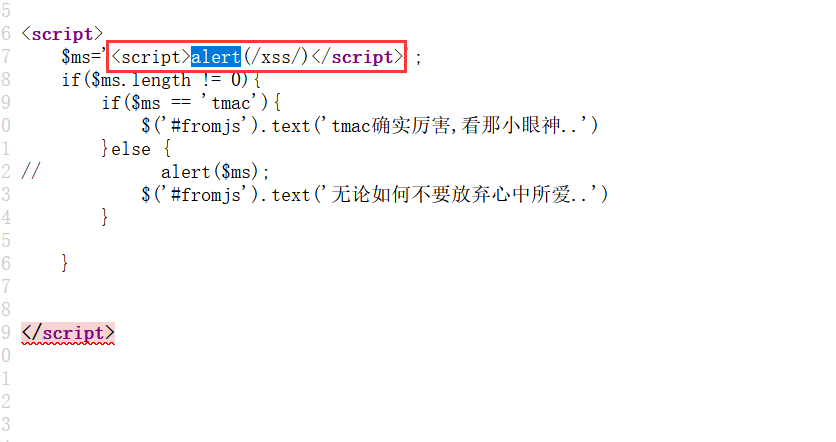
<a href=”javascript:alert(/xss/);”>点我</a>

' onclick=alert('xss')>

' onclick='alert(/xss/)

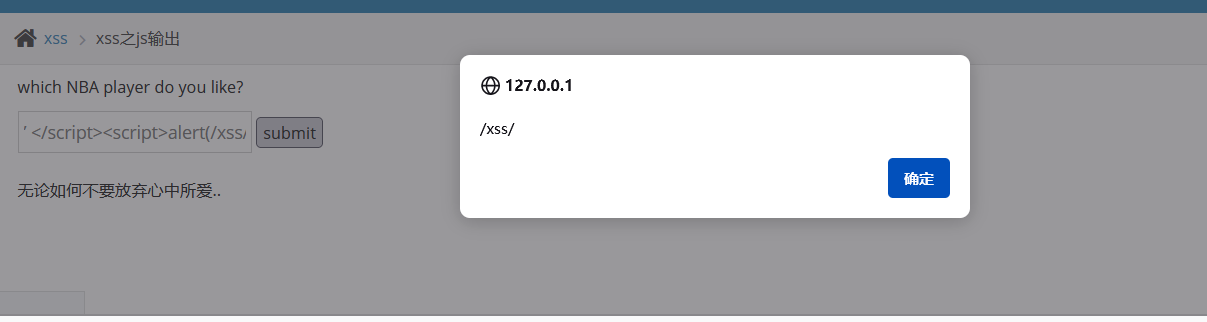
1. Xss之js输出
   1. js标签

输入xss代码后发现没有触发，右键查看源代码，发现在js标签中



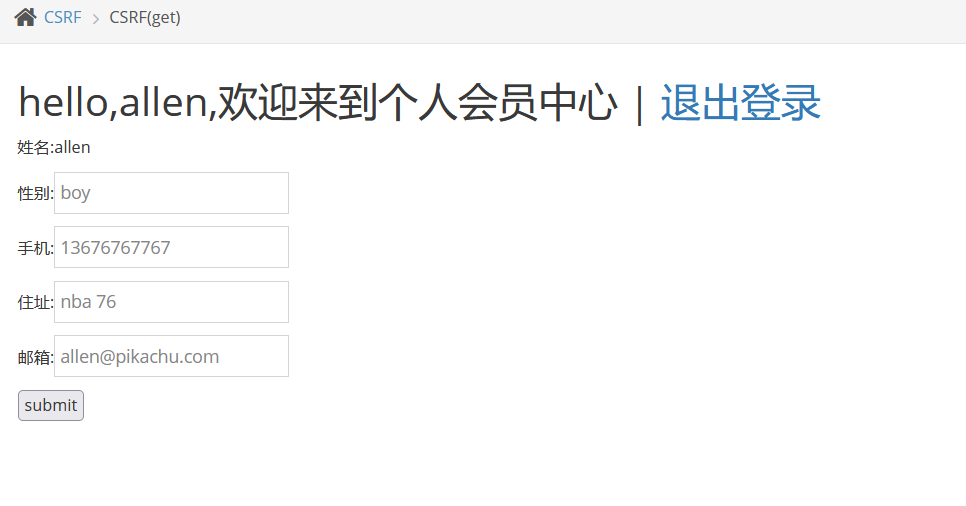
构造payload，第一个单引号用来闭合$ms的单引号，第一个</script>用来闭合script标签，剩下的内容就是被执行的XSS代码

’ </script><script>alert(/xss/)</script>

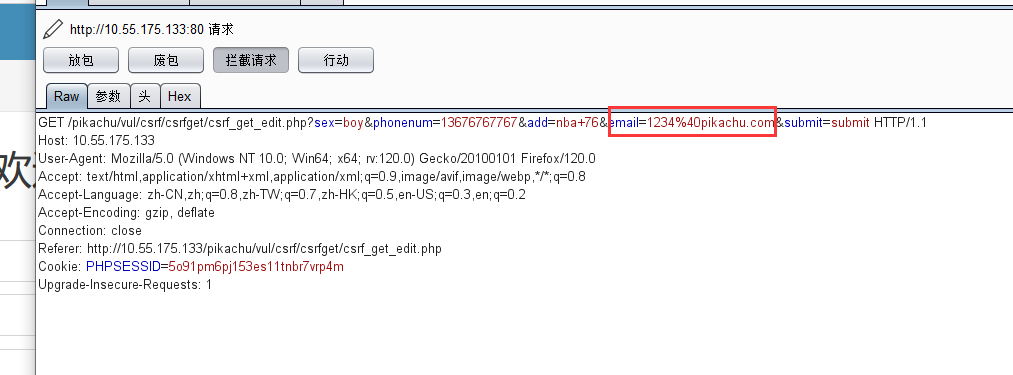


1. CSFR
2. CSFR（GET）
   1. 具体步骤

登录allen/123456,修改个人信息



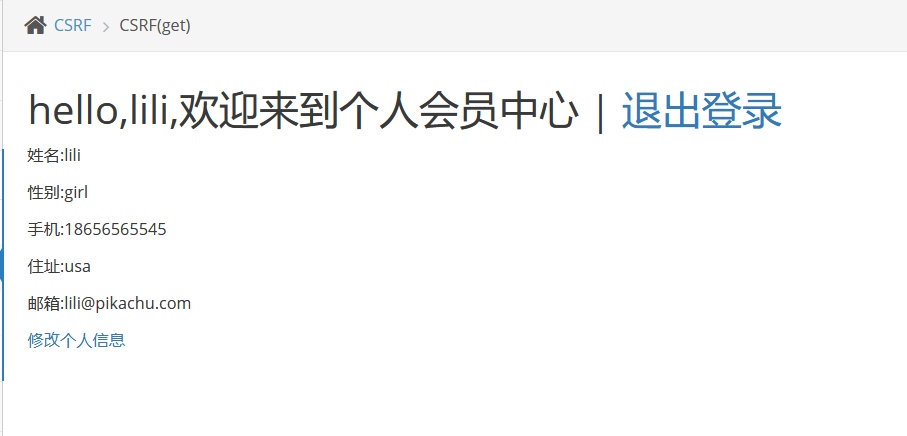
修改邮箱为1234@pikachu.com，提交数据后抓包



构造POC

<http://10.55.175.133/pikachu/vul/csrf/csrfget/csrf_get_edit.php?sex=boy&phonenum=13676767767&add=nba+76&email=1234%40pikachu.com&submit=submit>

登录lili/123456

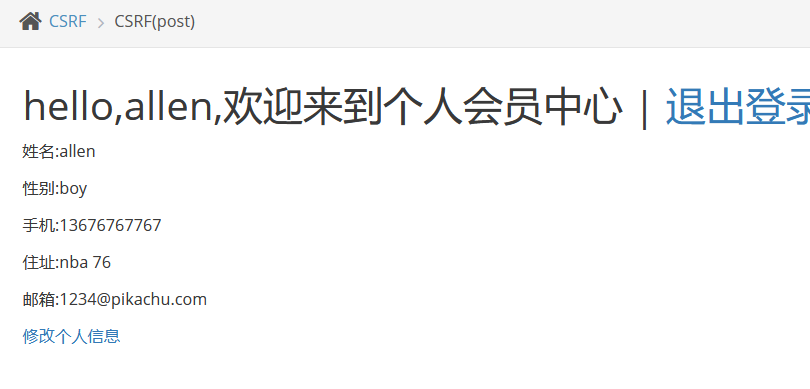


在登录状态下访问构造的POC，导致信息被修改

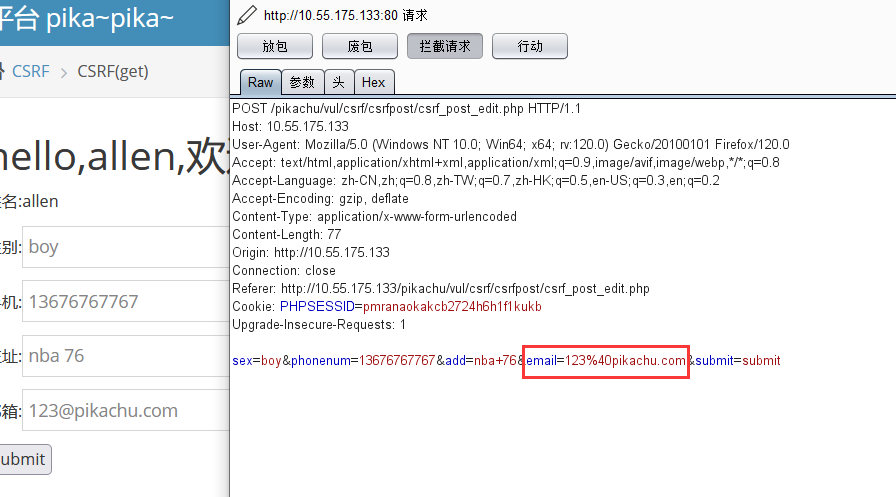


1. CSFR（POST）
   1. 具体步骤

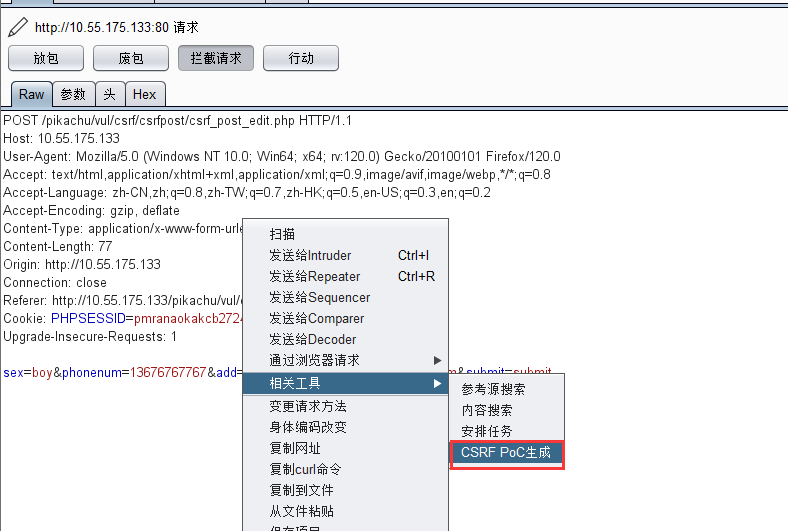
登录allen/123456,修改个人信息

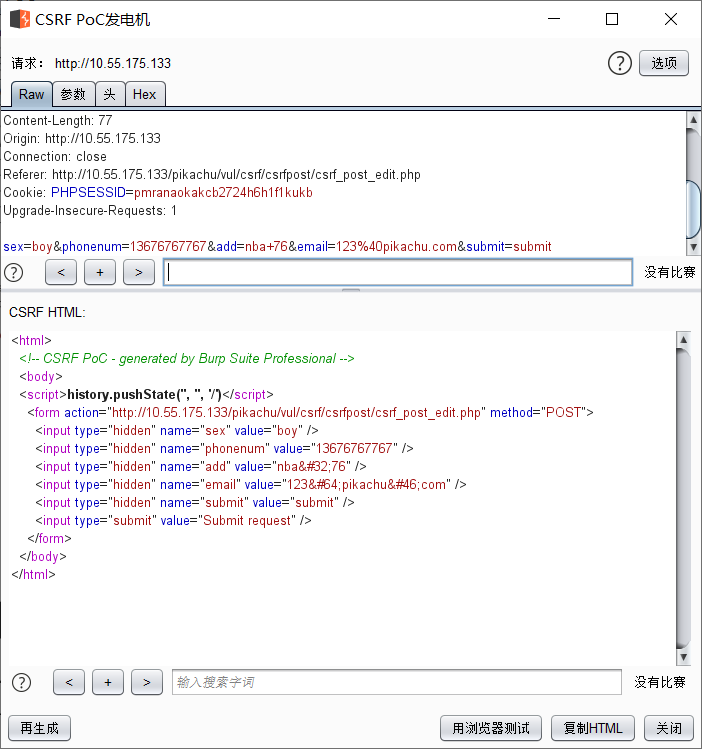


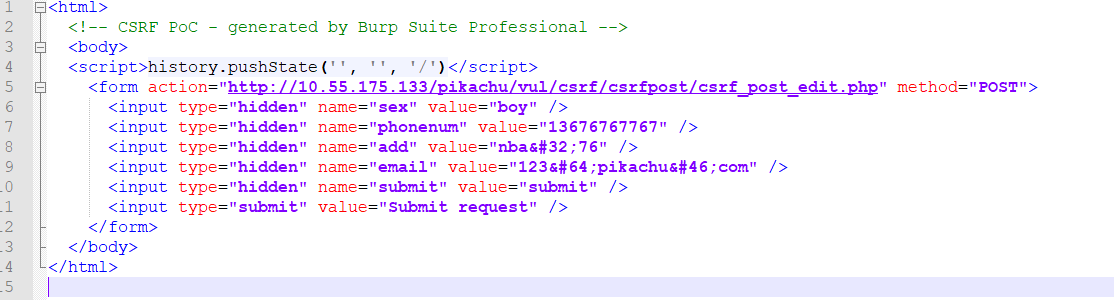
修改邮箱为123@pikachu.com，提交数据后抓包



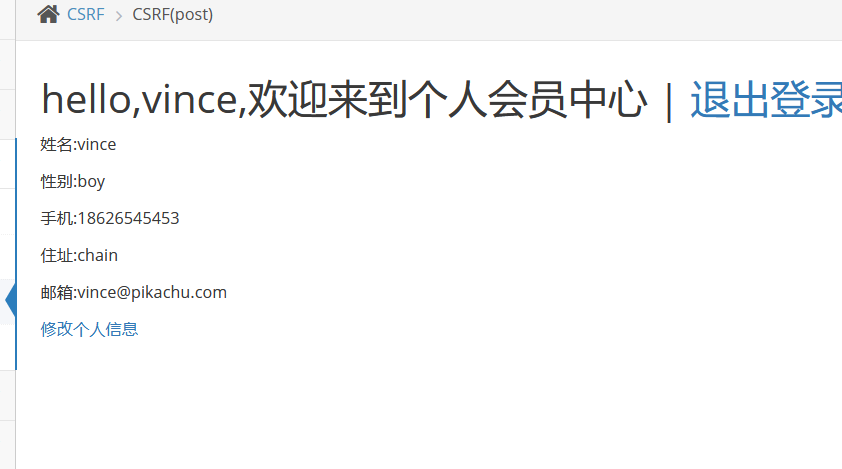
构造POC，新建一个.html页面，把构造的POC代码粘贴进去



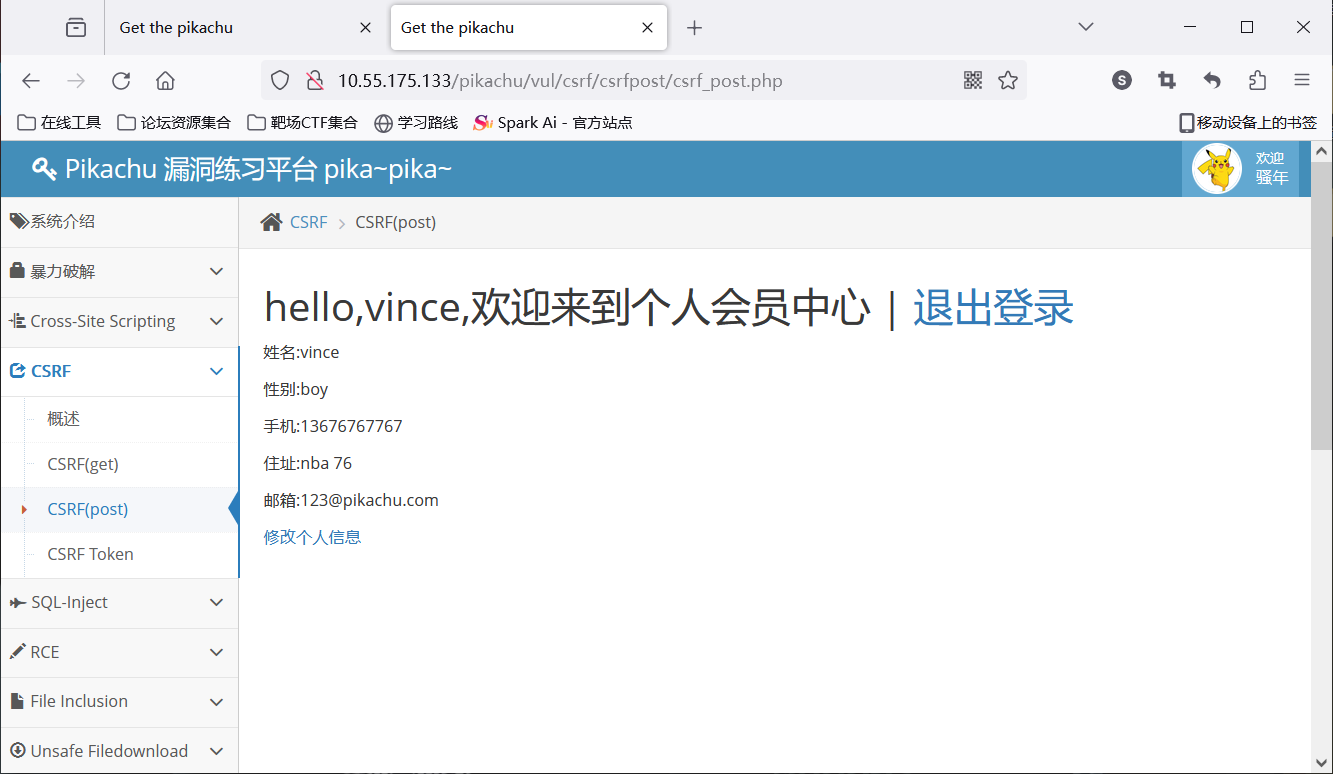




登录vince/123456



打开刚刚新建的html文件，发现vince的信息被修改



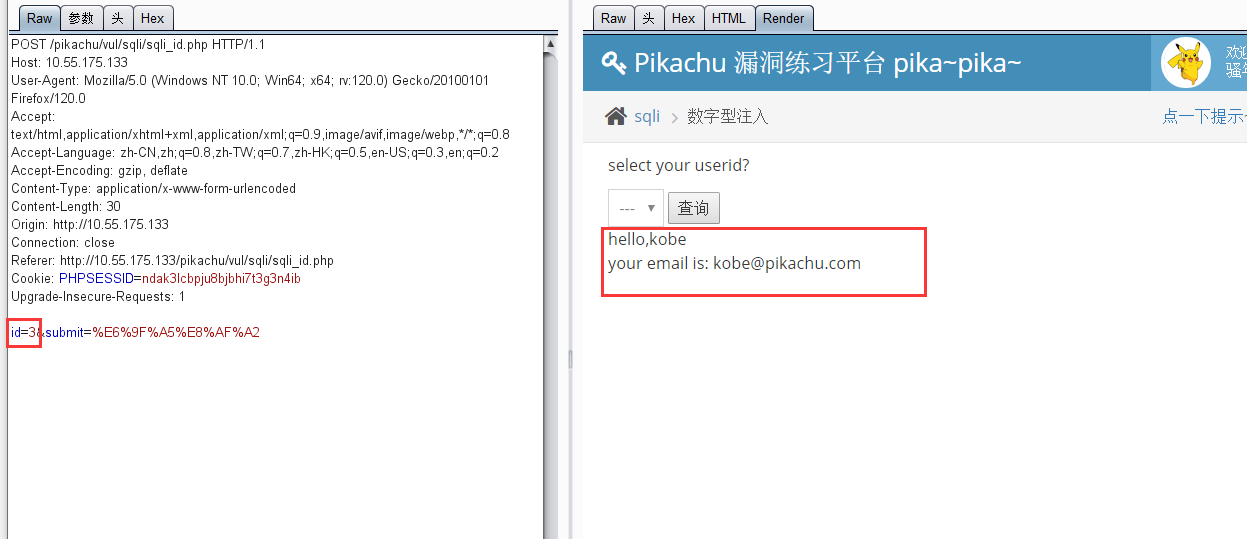
1. CSFR Token
   1. Token验证

存在Token验证机制，查看token\_get\_edit.php的源码, 发现有一个set\_token()函数, 该函数每次刷新页面都会被调用, 然后将SESSION中的token销毁, 并生成新的token发送至前端表单中，无法绕过

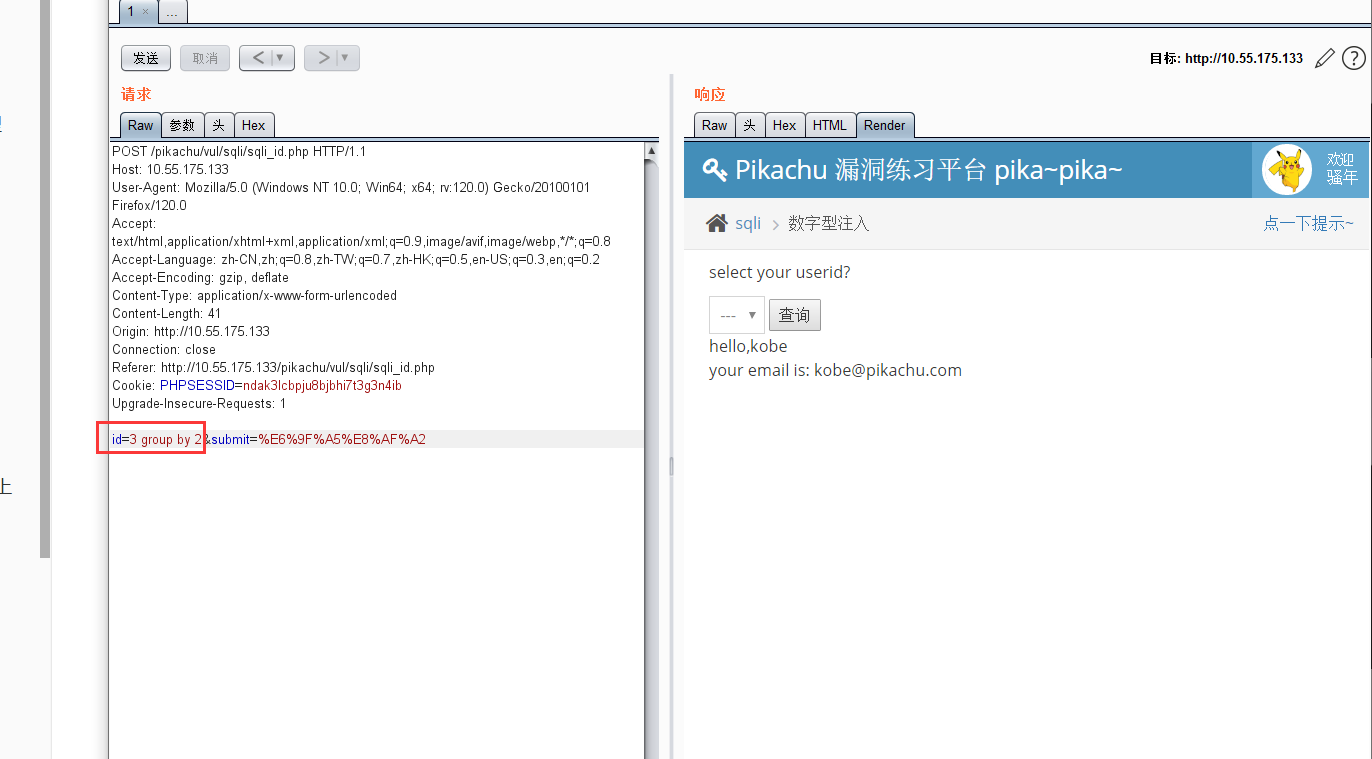


1. SQL注入
2. 数字型（POST）
   1. 具体步骤

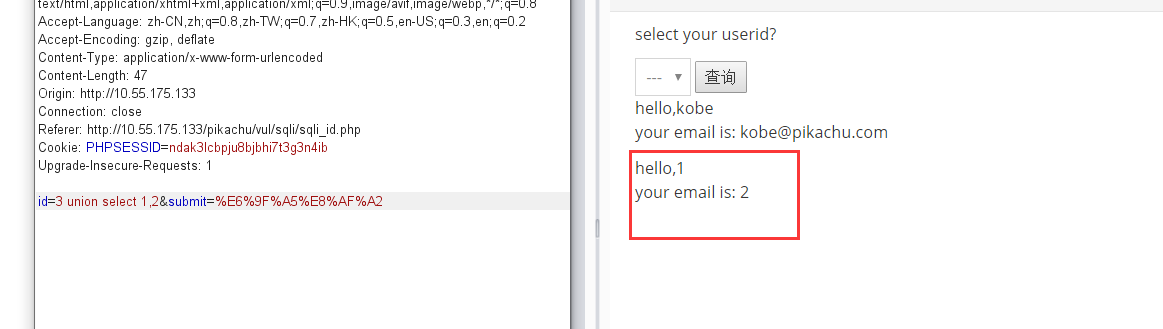
Post传参的数字型注入，通过修改id参数，回显内容也不同



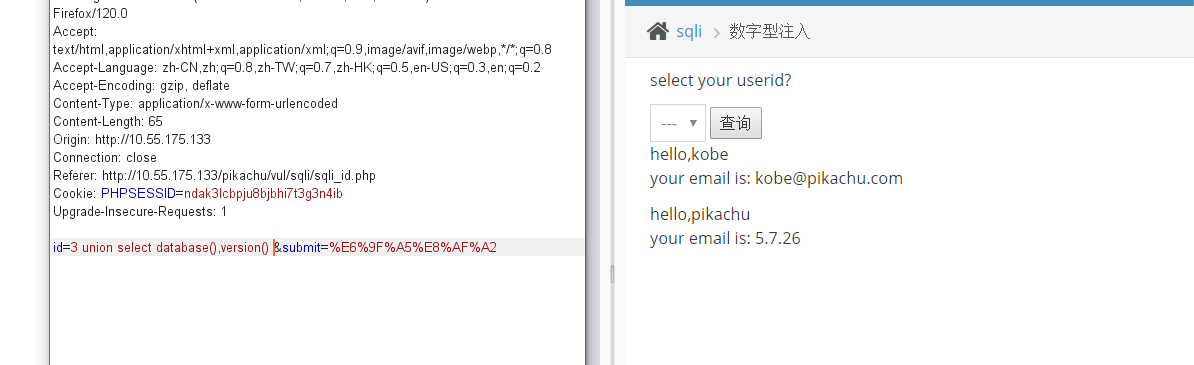
判断回显列数



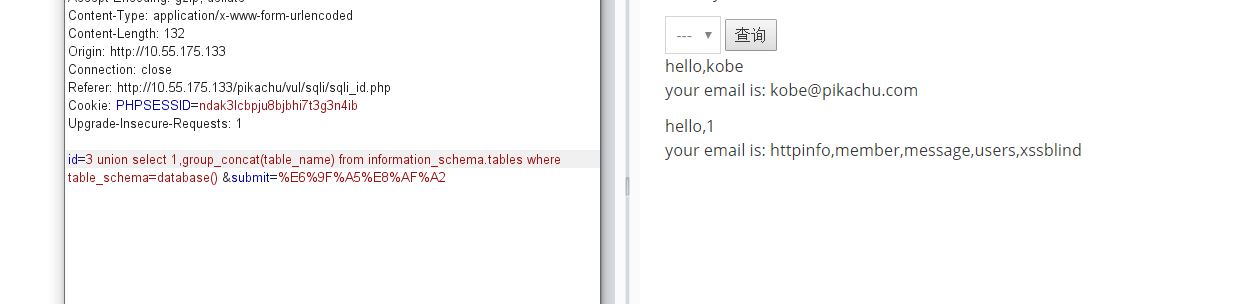
查询具体回显位



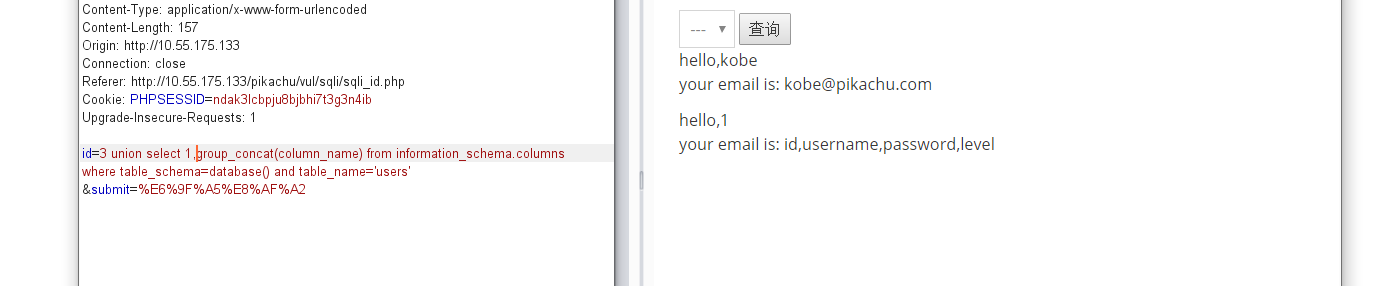
查询当前数据库和数据包版本信息



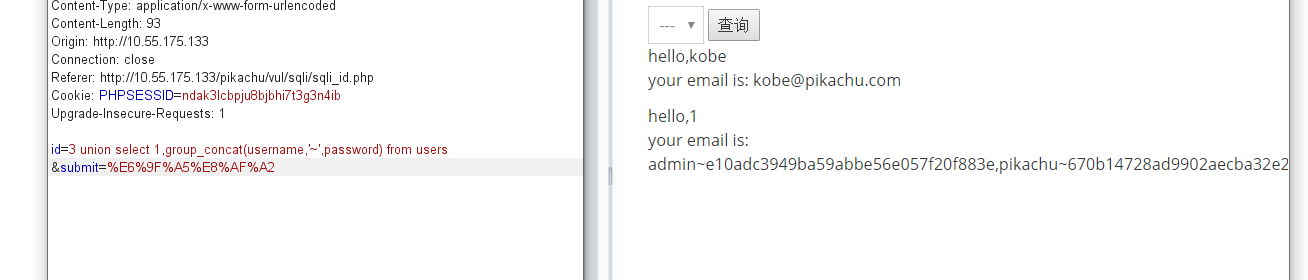
查询当前数据库下存在的表



查询users表下的列名

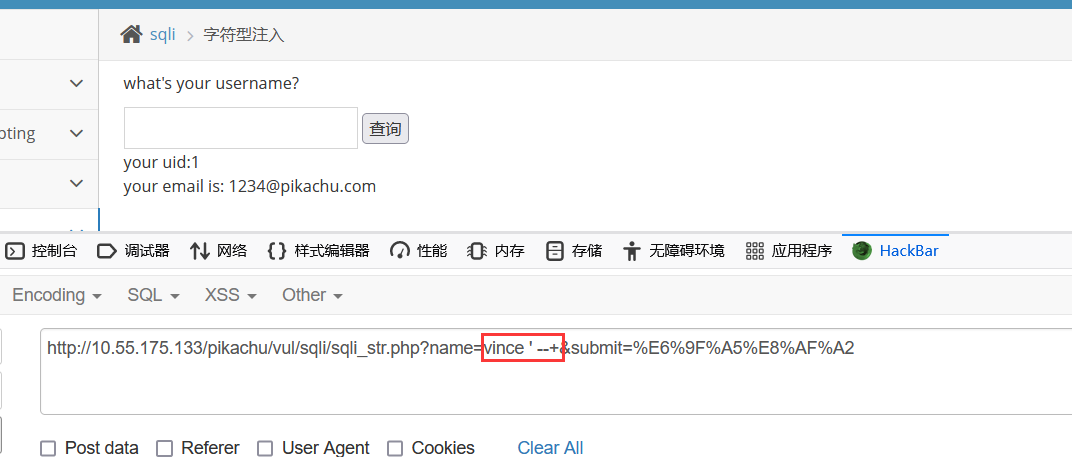


查询users表下，username和passeord列的具体信息



1. 字符型（GET）
   1. 具体步骤

和数字型（POST）类似，注意是GET类型提交数据和闭合、注释payload











1. 搜索型
   1. 具体步骤









1. xx型
   1. 闭合方式的判断

使用\判断闭合方式，返回的报错信息中\后面的内容就是闭合方式



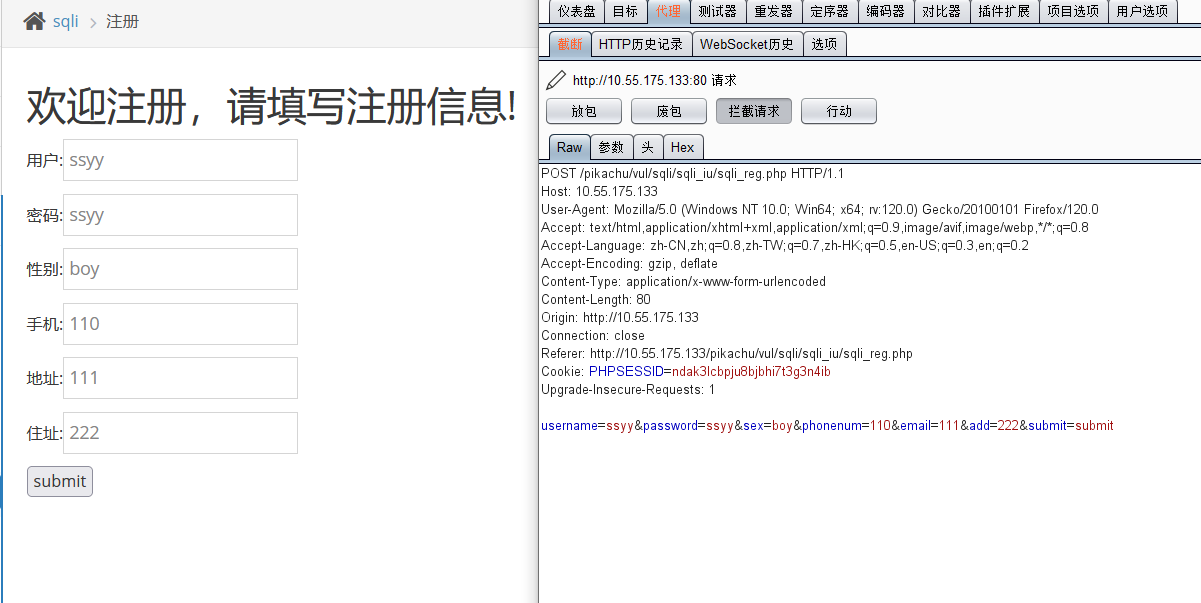
后续步骤与字符型注入一样

1. “insert/update”注入
   1. insert语句注入

这一关是注册账号，后端使用insert语句而不是select语句，因此没有回显，需要利用报错注入

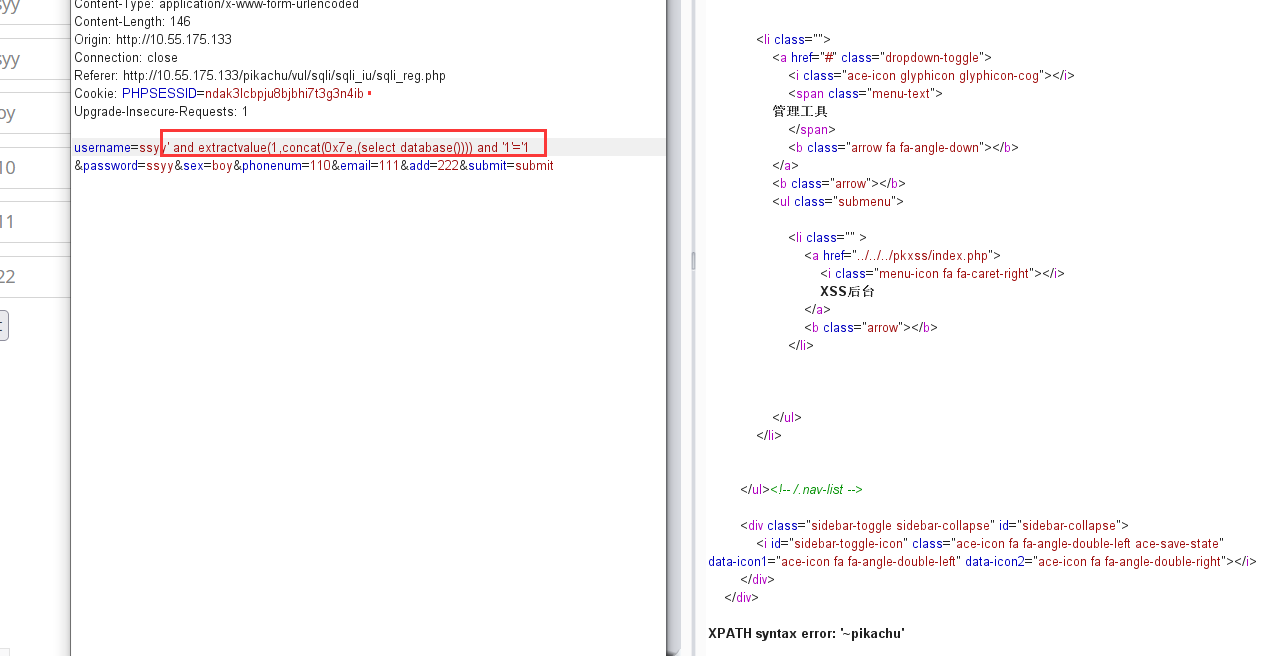
* 1. 具体步骤

抓取注册的数据包



对post数据包的内容进行修改构造payload，

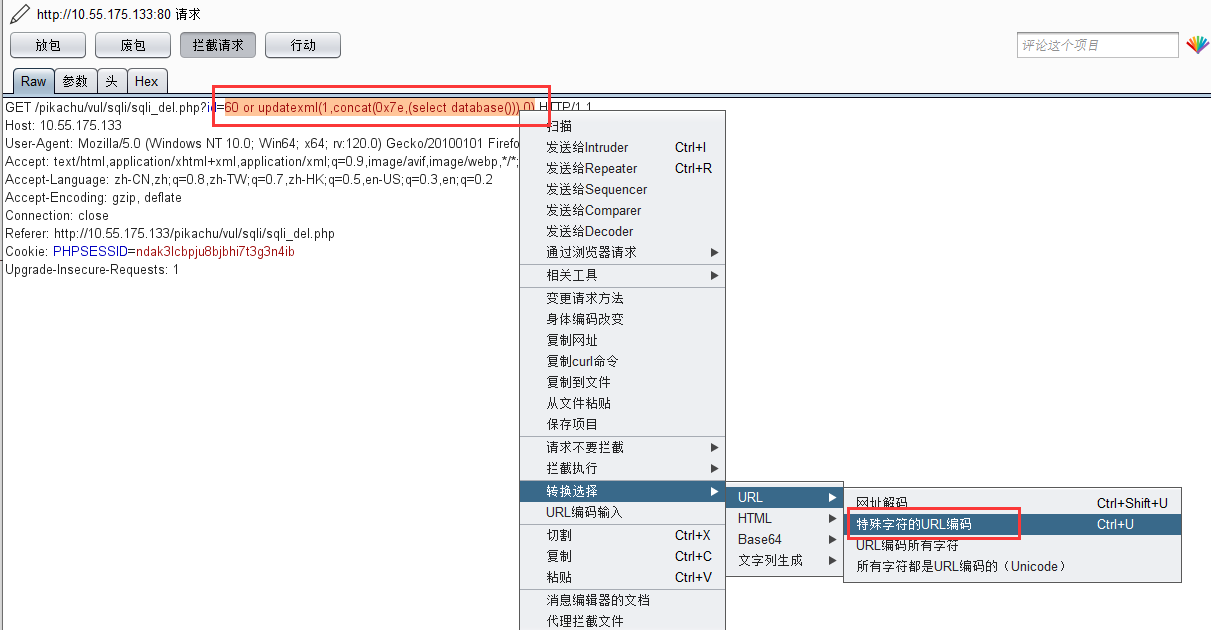
' and extractvalue(1,concat(0x7e,(select database()))) and '1'='1

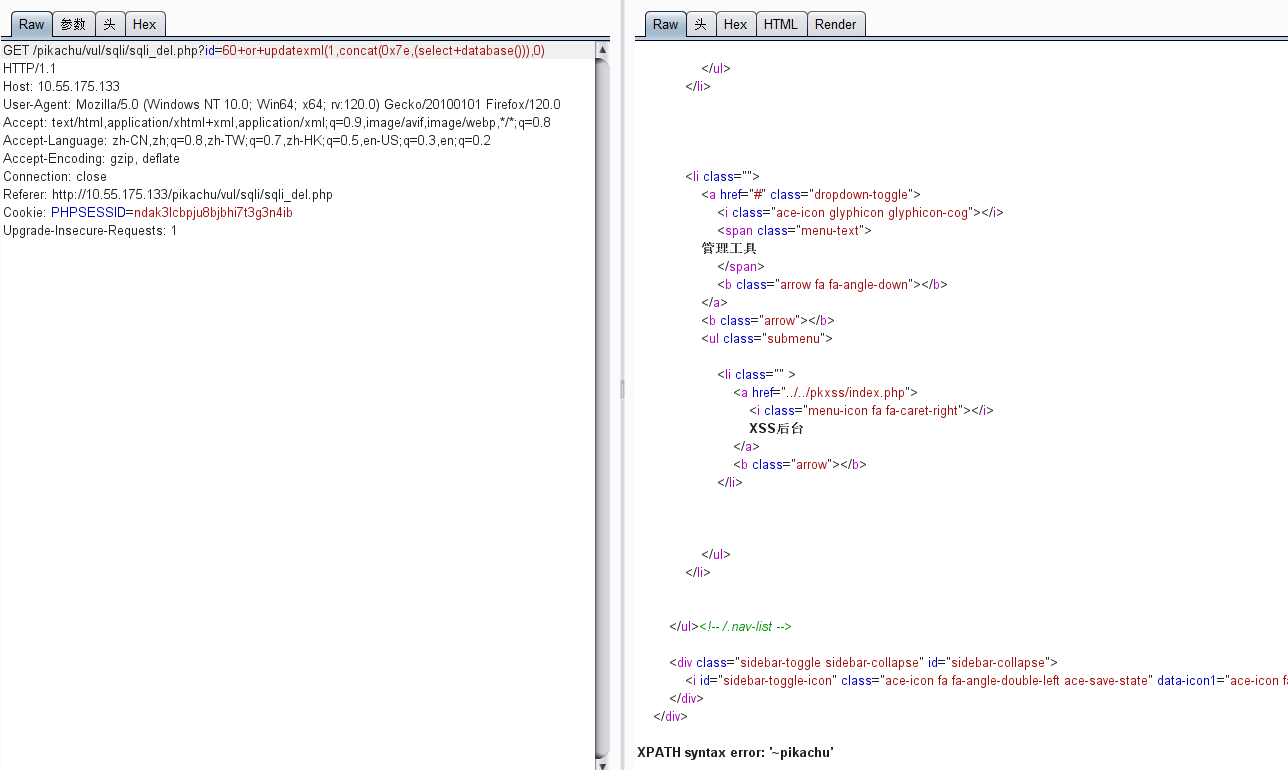


1. “delete”注入
   1. 具体步骤

和上一题一样使用报错注入，

构造payload为or updatexml(1,concat(0x7e,(select database())),0)

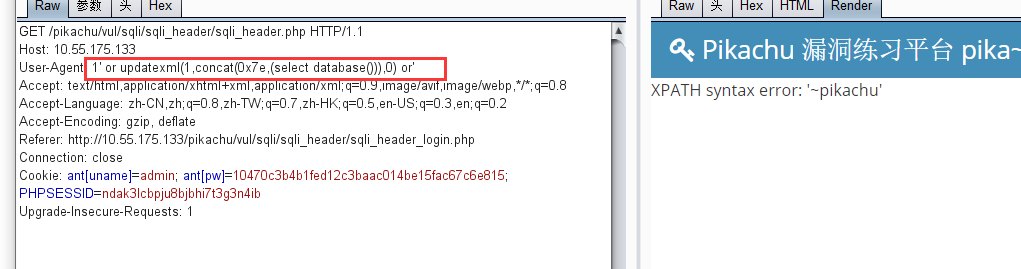




1. “http header”注入
   1. http头注入
   2. 具体步骤

抓包，修改user-agent头部信息





1. 布尔盲注
   1. 具体步骤

获取数据库长度为7



使用bp进行半自动化注入



位置1选择1-7（数据库的长度）



位置2选择0-127



1. 时间盲注
   1. 具体步骤

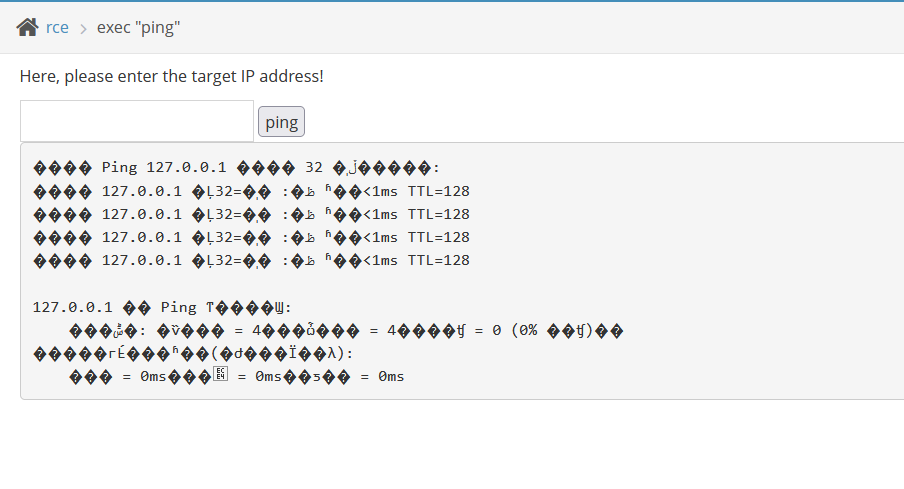


1. 宽字节注入
   1. 具体步骤
2. RCE远程执行
3. Exec
   1. 命令执行

利用exec()函数进行代码执行,输入的内容会在cmd中执行

* 1. 具体步骤

输入127.0.0.1，会去ping这个地址并且返回结果



但是输入127.0.0.1 | whoami，管道符|后面的内容也会被执行

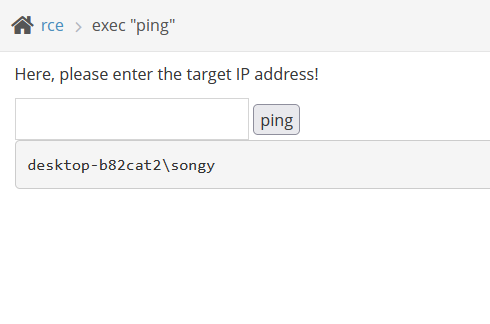
这里简述四种管道符

“|”左边命令的输出就会作为“|”右边命令的输入

“||”前面命令执行成功,不执行后面命令；前面命令执行失败,执行后面命令

&不管a命令是否执行成功, 都会去执行b命令

&&前面命令执行成功,才会执行后面命令；前面命令执行失败,则不执行后面的命令

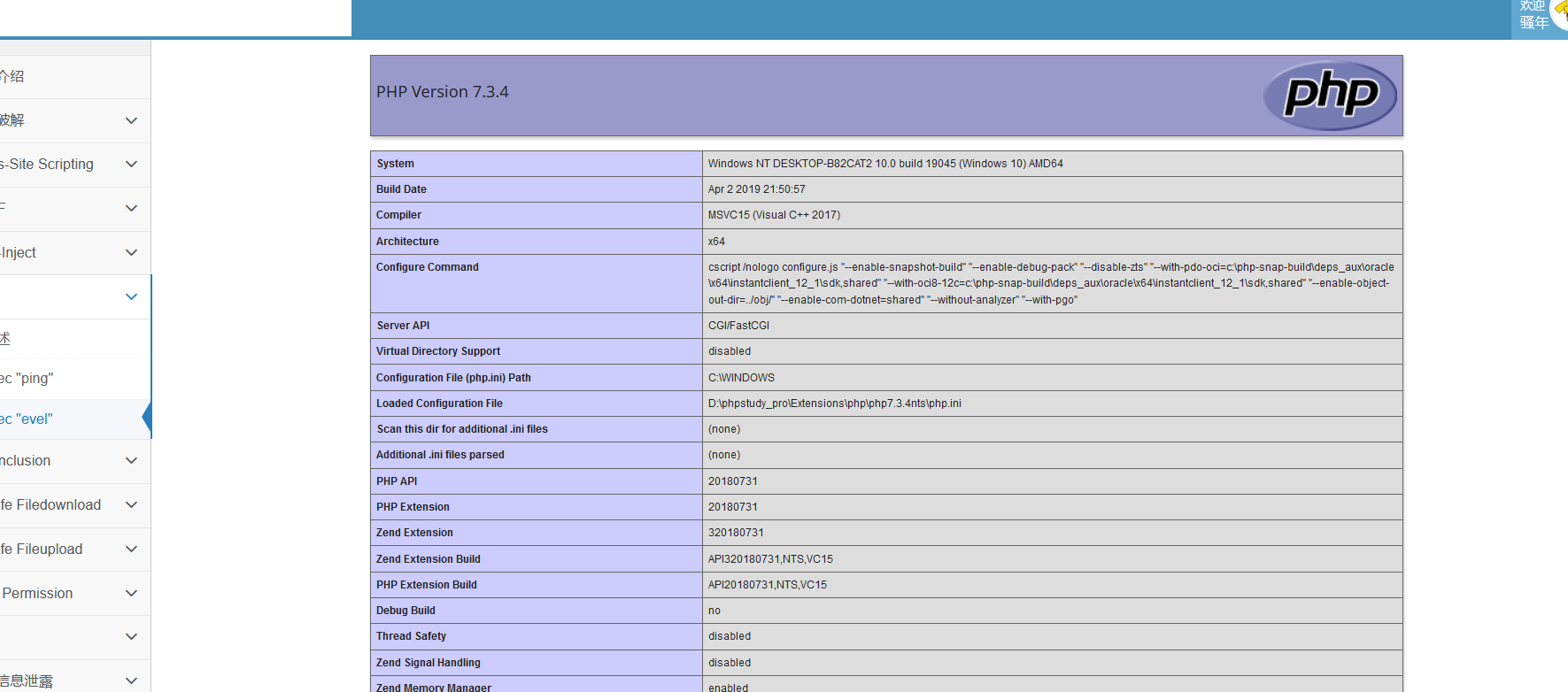


1. Eval
   1. 代码执行

eval()函数输入的内容会作为代码被解释器执行，这里是php代码

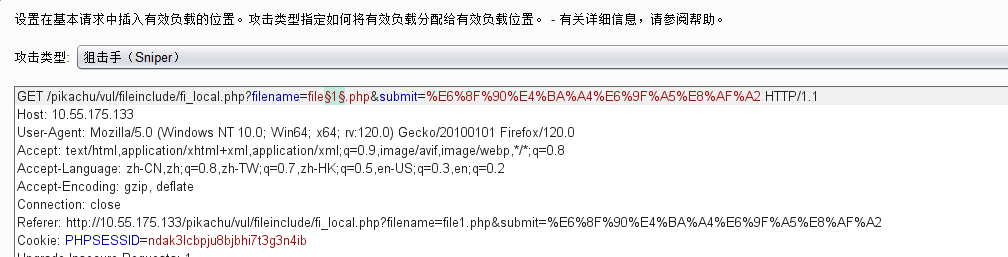
* 1. 具体步骤

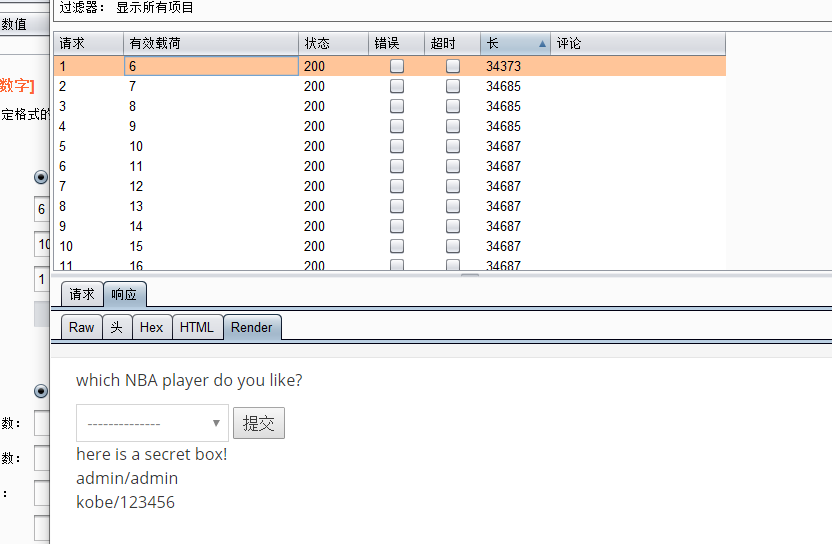
输入phpinfo();，会被作为代码执行



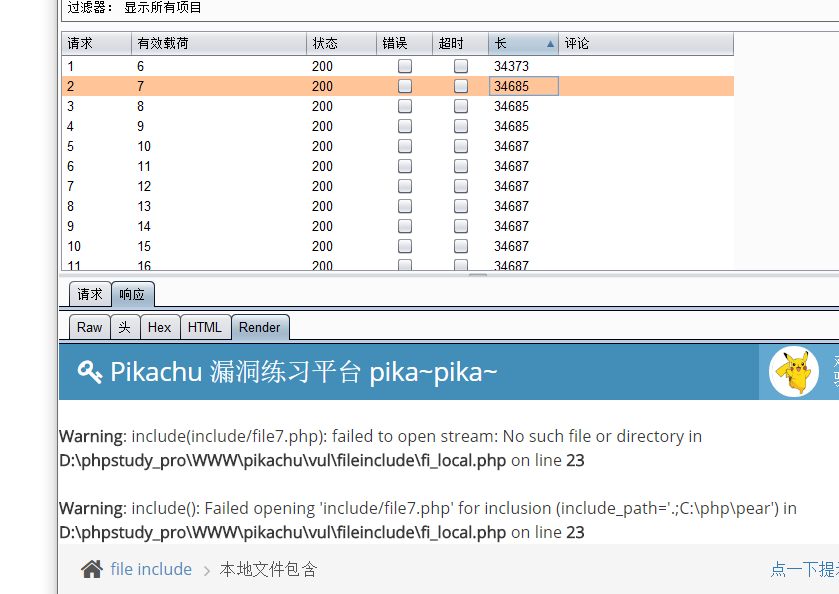
1. 文件包含
2. 本地包含
   1. 读取目录下隐藏文件

对url中的filex.php进行爆破，可以发现还存在隐藏文件file6.php





通过本地包含file7.php可以知道，include()函数是包含include文件夹下的文件



* 1. 读取系统文件

使用相对路径，可以读取到系统本地文件，包括hosts等配置文件



* 1. 执行getshell

包含“文件上传”上传的图片马

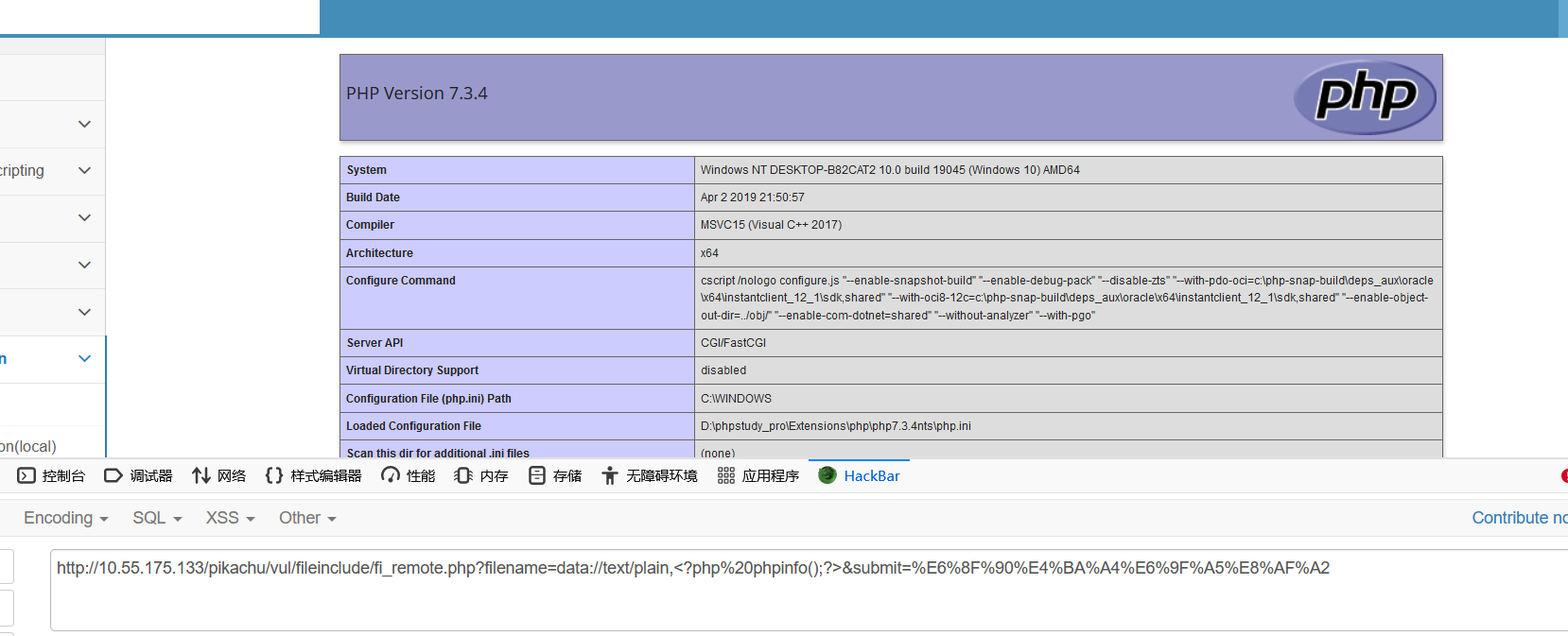
1. 远程包含
   1. 读取系统文件

由于本体包含的路径中含有“include/”，因此可以使用绝对路径包含

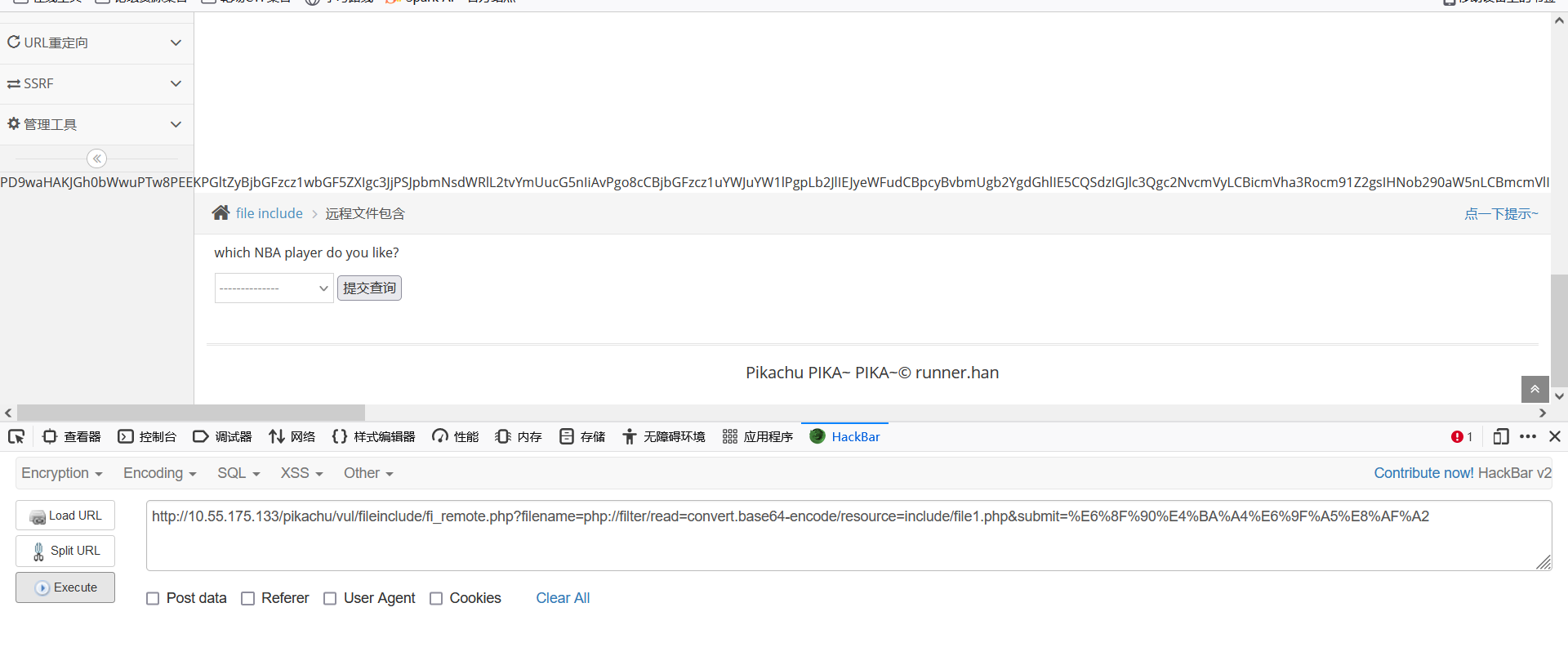


* 1. php伪协议的利用

执行php代码



读取文件源码，以base64加密的格式显示出来，例如读取include/file1.php文件



读取本地文件

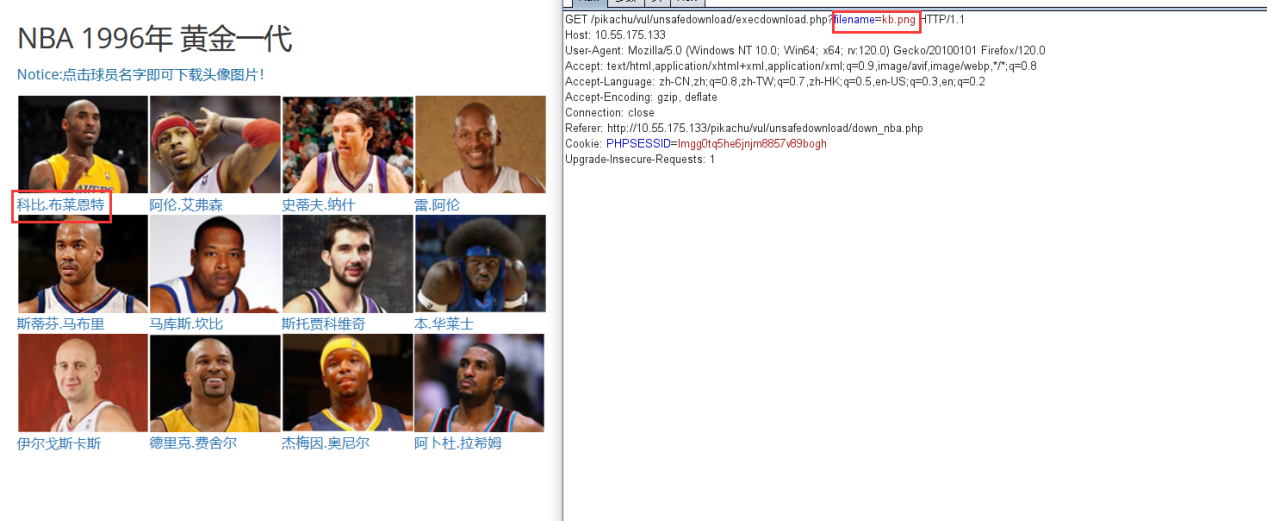


* 1. 包含木马文件

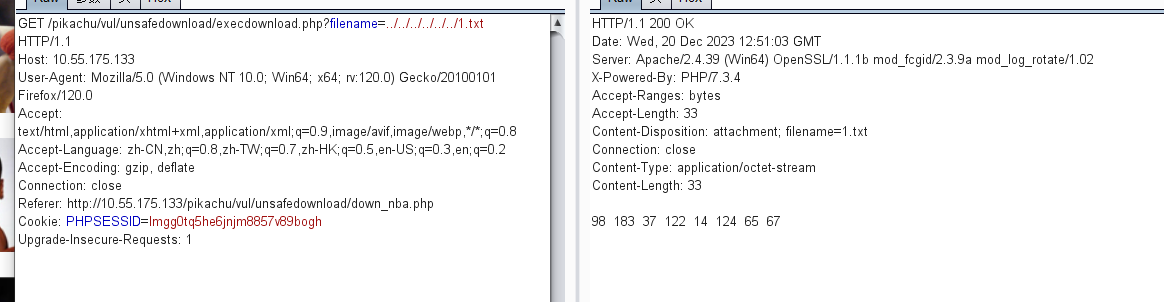
配合文件上传

1. 文件下载
2. 下载任意文件
   1. 步骤

点击图片下方文字，进行下载



通过构造filename后面的参数名称，可以下载任意文件，例如filename=../../../../../../1.txt,下载D盘下的1.txt文件



1. 文件上传
2. 前端验证
   1. 步骤

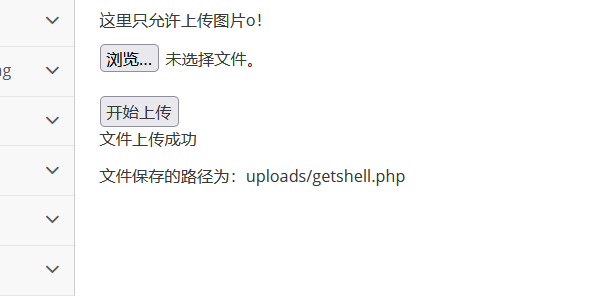
正常上传php木马文件，会报错



直接在前端JS代码中删除对文件进行校验和相关代码



上传成功



1. MIME验证
   1. MIME

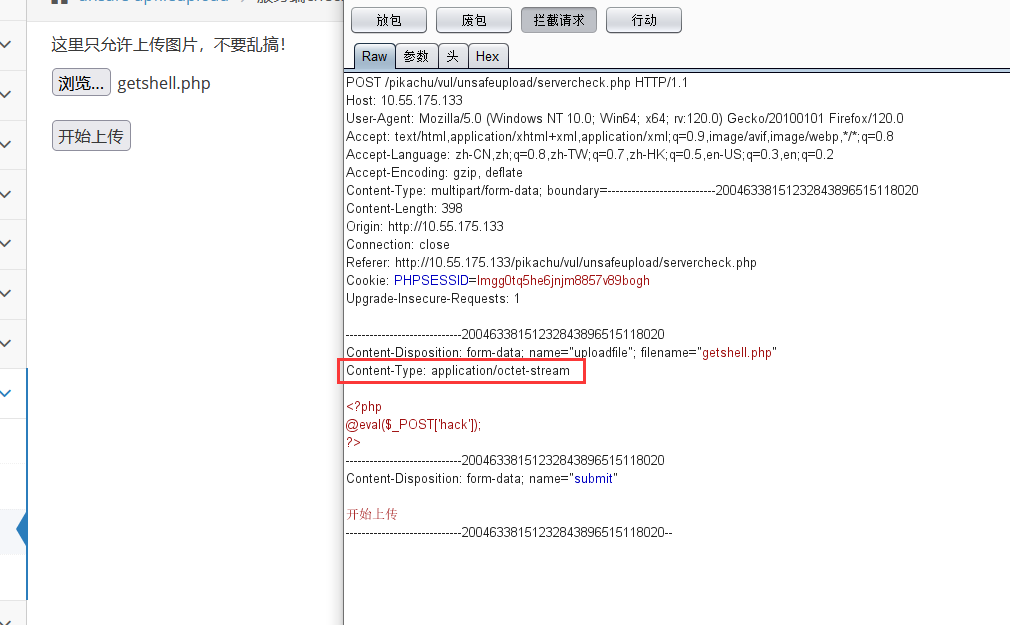
MIME (Multipurpose Internet Mail Extensions) 是描述消息内容类型的标准，用来表示文档、文件或字节流的性质和格式。

MIME 消息能包含文本、图像、音频、视频以及其他应用程序专用的数据。

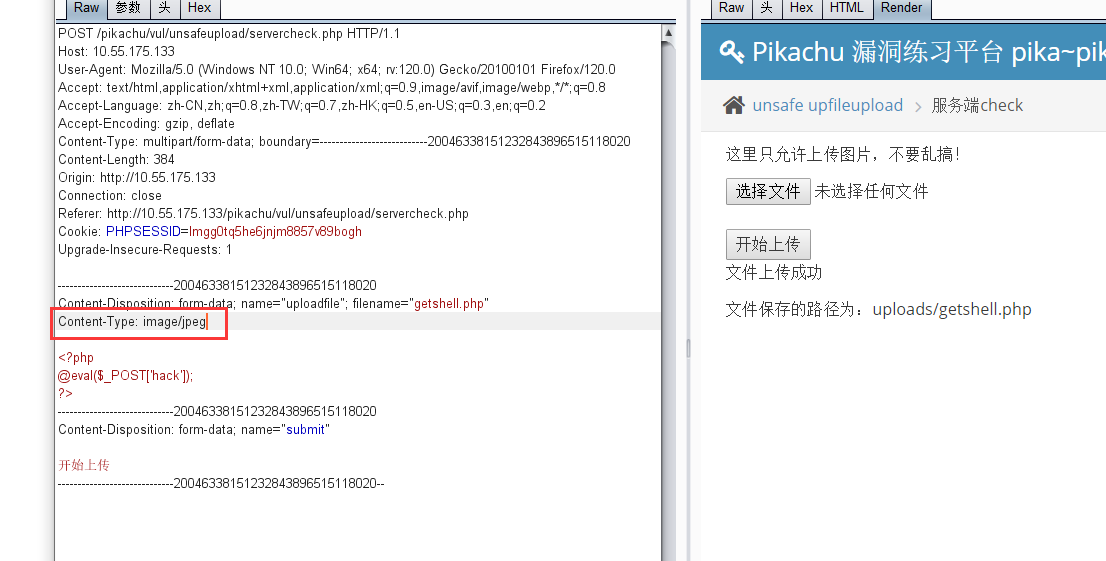
浏览器通常使用 MIME 类型（而不是文件扩展名）来确定如何处理URL，因此 We b服务器在响应头中添加正确的 MIME 类型非常重要。

* 1. 步骤

后端会对如图所示的内容进行验证，必须是图片格式的MIME类型才能通过验证



修改为image/jpeg后，上传成功

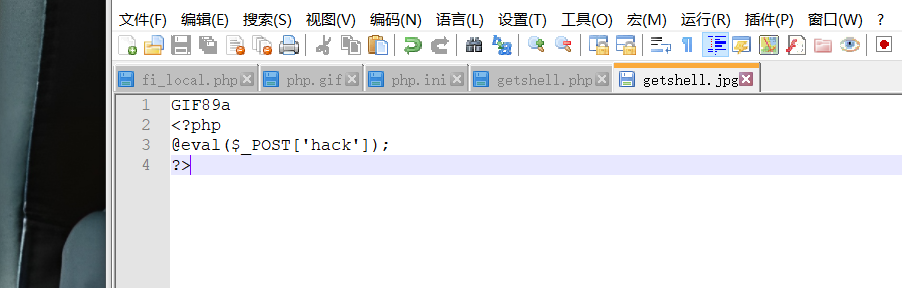


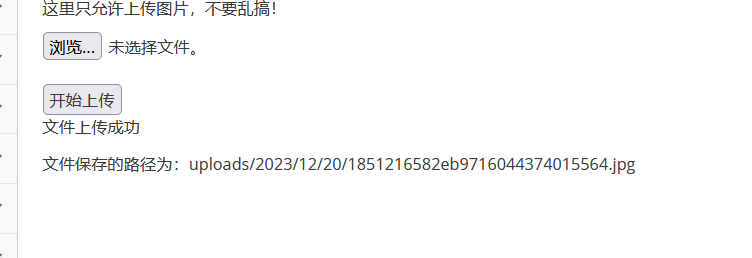
1. 文件内容验证
   1. getimagesize()

getimagesize() 函数用于获取图像大小及相关信息，成功返回一个数组，失败则返回 FALSE 并产生一条 E\_WARNING 级的错误信息。

* 1. 步骤

由于后端会对文件的内容进行验证，在这里需要在文件中添加图片头文件来绕过

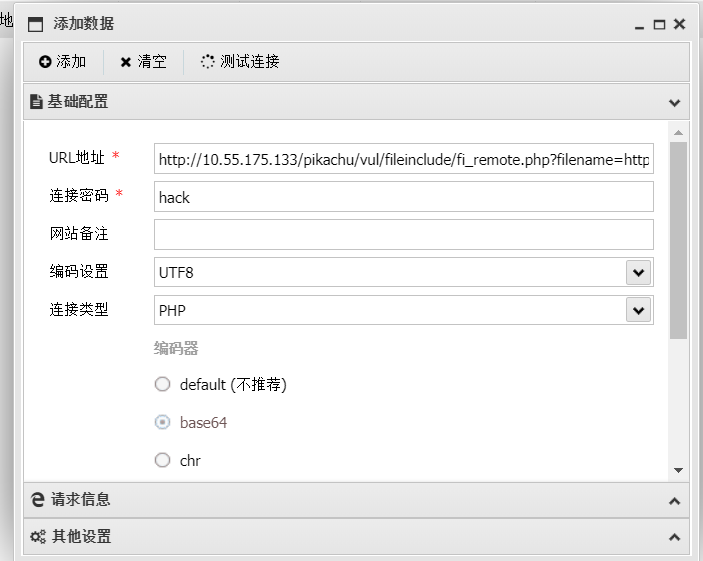


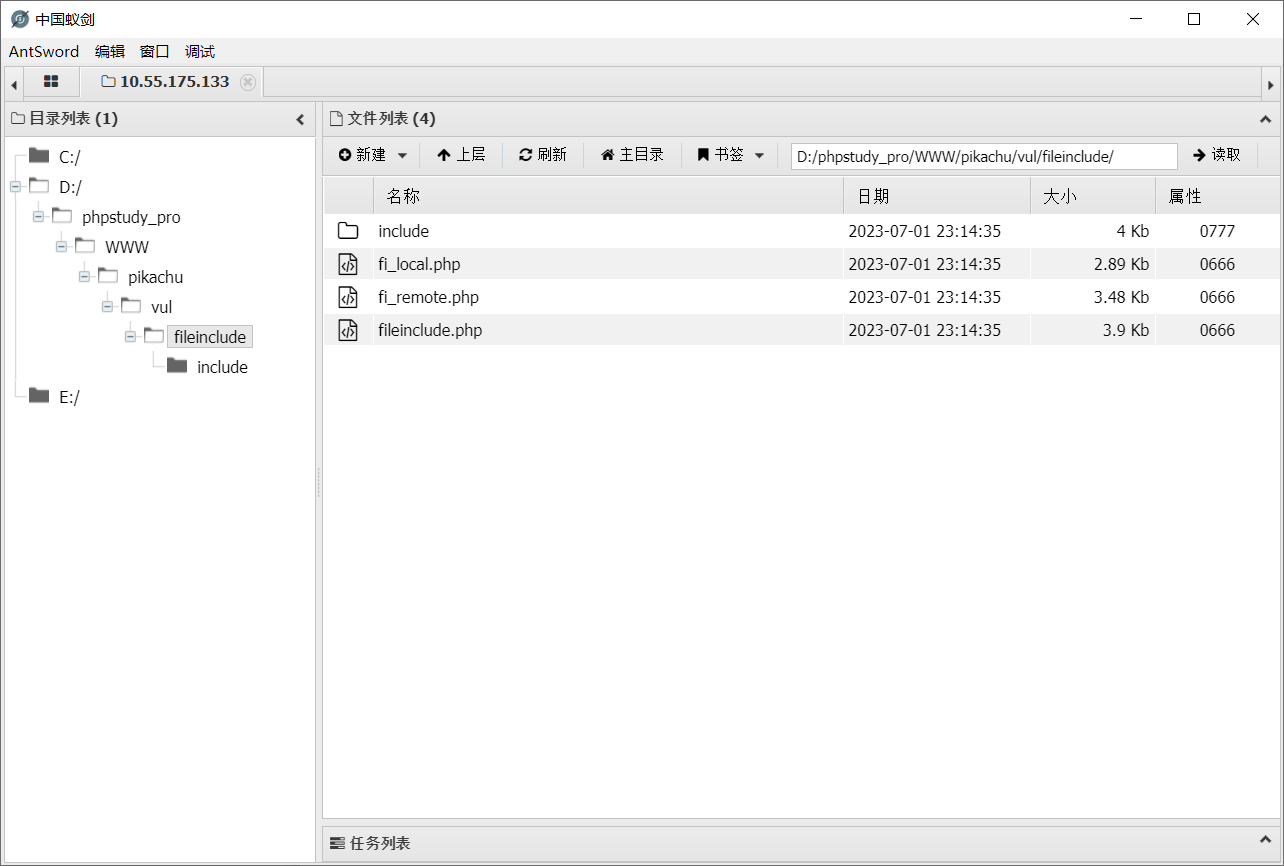
上传成功

利用文件包含漏洞，远程包含执行图片马中的恶意PHP代码



使用蚁剑连接





1. 越权
2. 水平越权
   1. 水平越权

A借助B的权限完成操作

* 1. 具体步骤

登录lucy/123456，显示lucy的信息



在url中修改username,可以直接显示其他人的信息

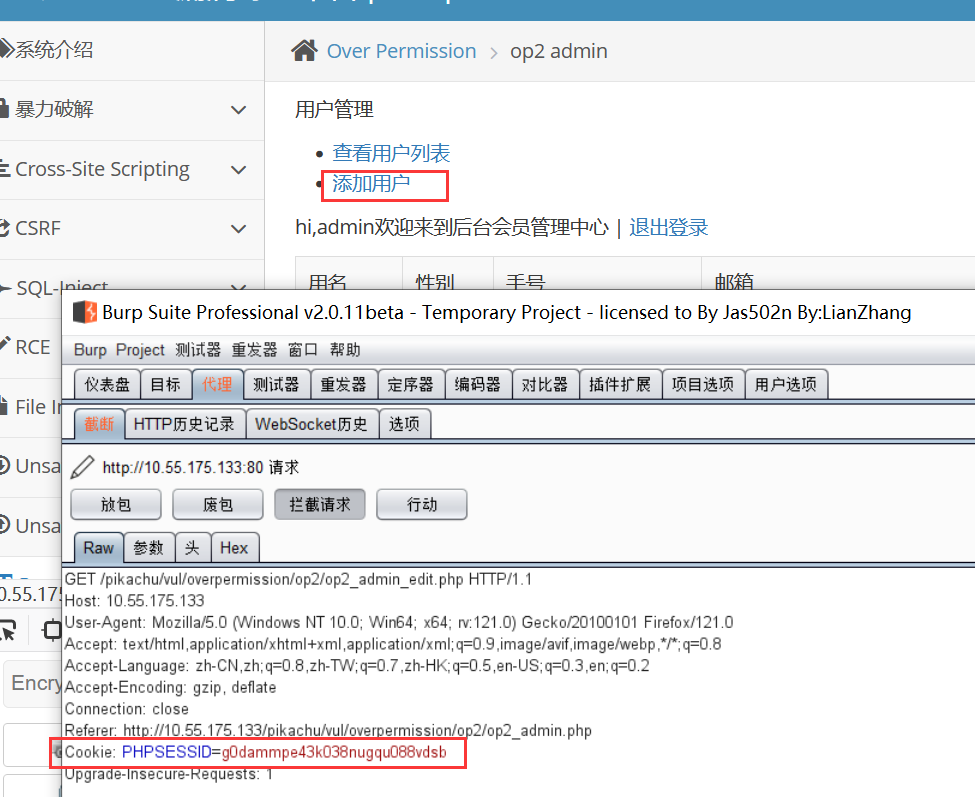


1. 垂直越权
   1. 垂直越权

A借助管理员的权限完成操作

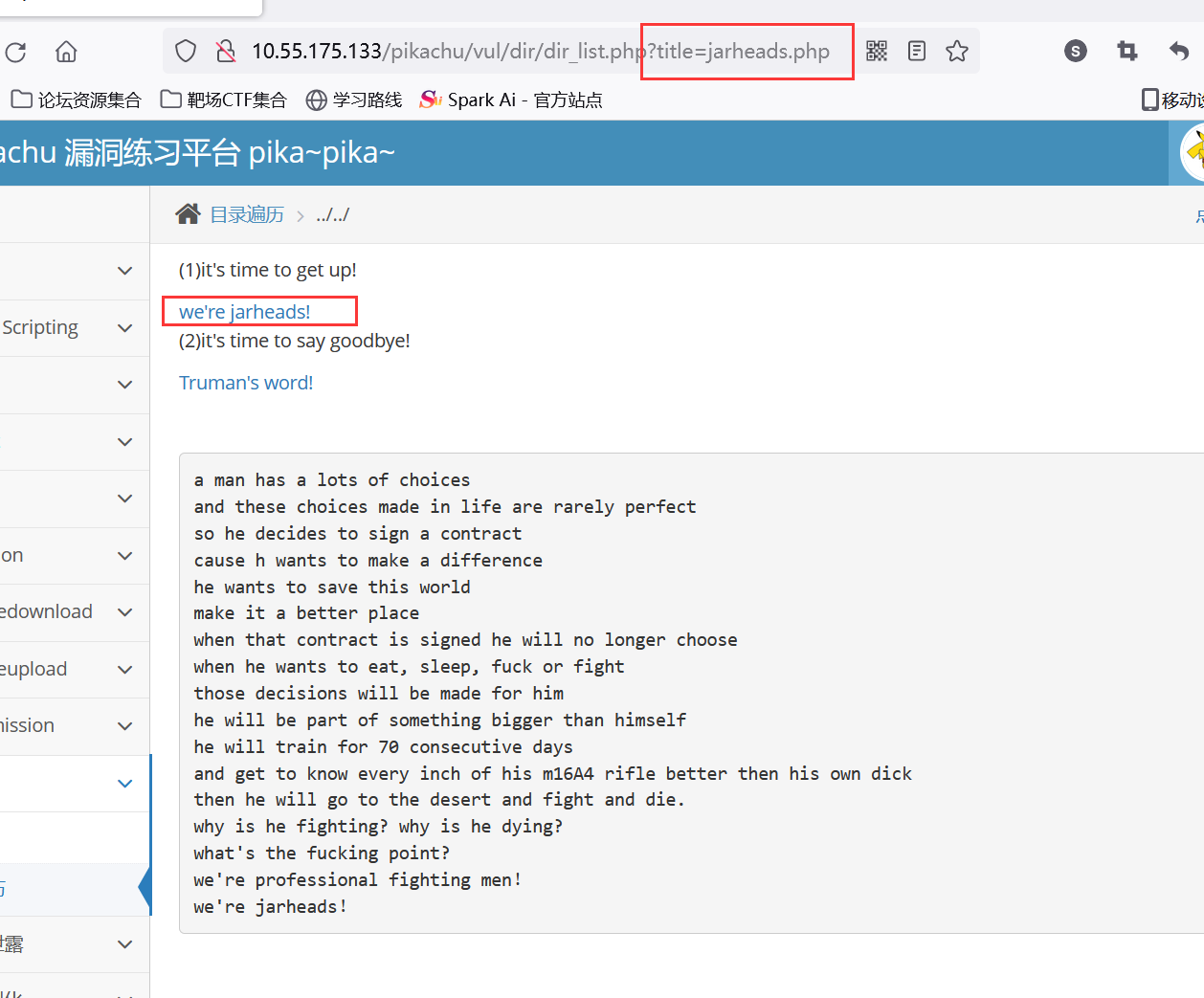
* 1. 具体步骤

这里一共存在两个用户，admin和pikachu，登录之后存在两个不同的cookie，通过替换cookie来实现操作其他用户的权限，先登录admin/123456，执行“添加用户的操作”

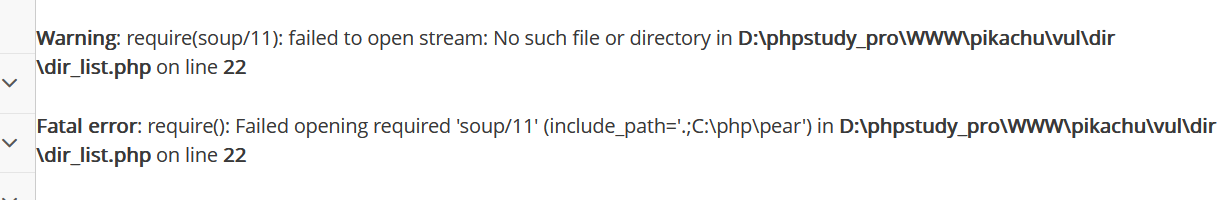


1. 目录遍历
2. 目录遍历
   1. 具体步骤

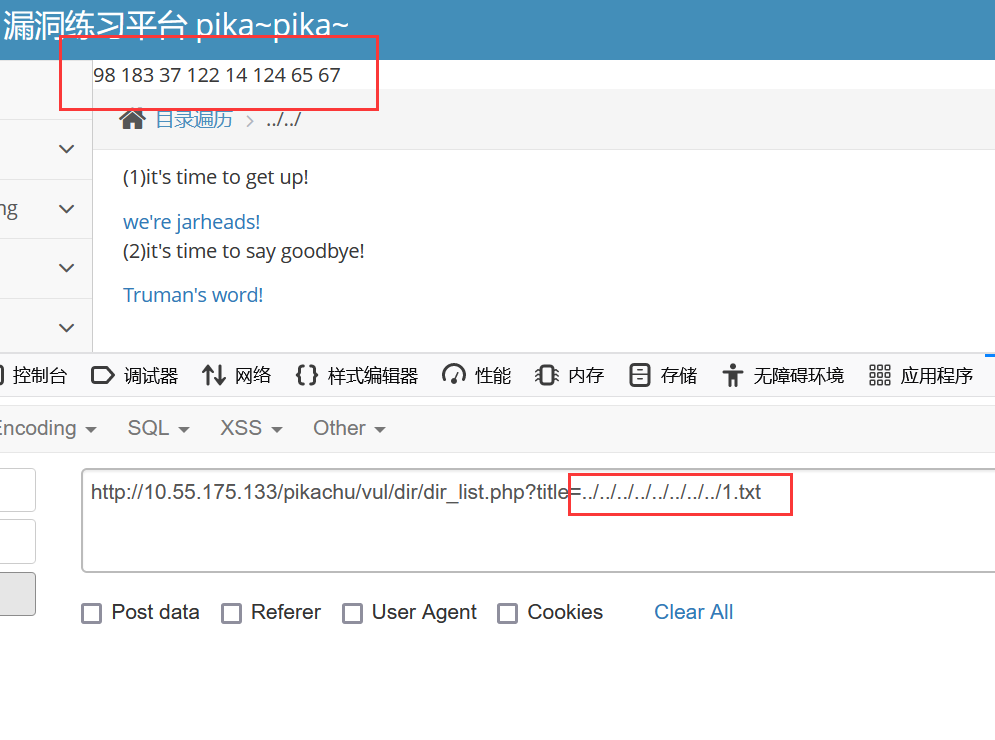
点击文字，网页会把特点的文件内容显示在这里，通过修改文件的位置，能够显示系统的敏感文件



输入任意内容，判断当前文件位置



构造POC读取D盘下的某个文件，../../../../../../../../1.txt



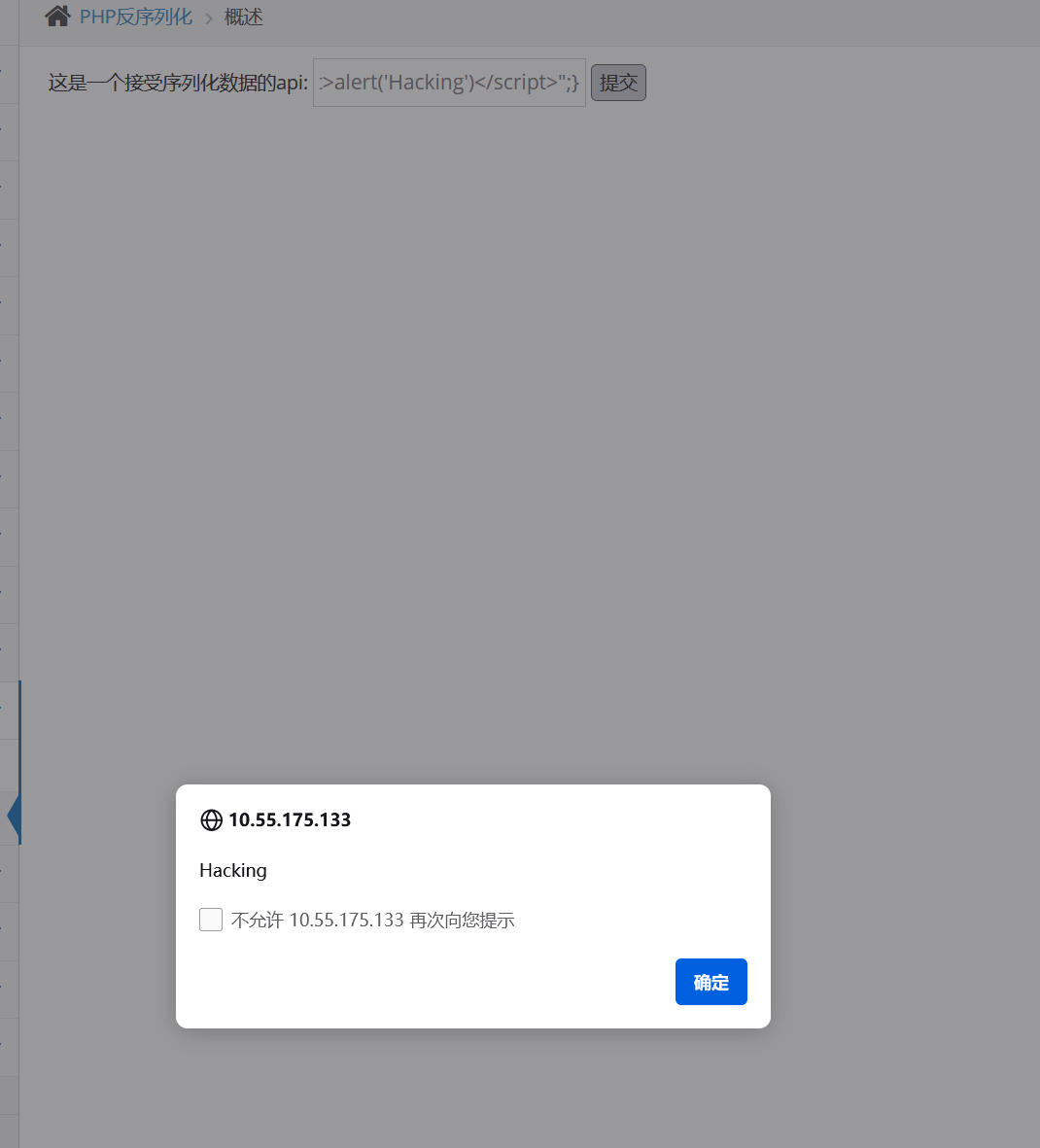
1. 敏感信息泄露
2. 敏感信息泄露
   1. 具体步骤

查看网页源代码，发现账号密码



1. PHP反序列化
2. PHP反序列化
   1. 具体步骤

对S类的test属性进行构造



1. XXE
2. XXE漏洞
3. URL重定向
4. 不安全的URL跳转
5. SSFR
6. SSRF(curl)
7. SSRF(file\_get\_content)