攻防世界ｗｅｂ新手部分，进阶部分

原创

舞动的獾 于 2019-04-08 20:41:29 发布 4400 收藏

分类专栏： ｗｅｂ 文章标签： ｃｔｆ

本文链接：https://blog.csdn.net/Yu_csdnstory/article/details/89107000

---

title: 攻防世界第一题

date: 2019-04-08 19:31:23

author: 舞动之獾

top: false

cover: true

coverImg: https://img-blog.csdnimg.cn/20190408193755785.PNG

categories: Markdown

tags:

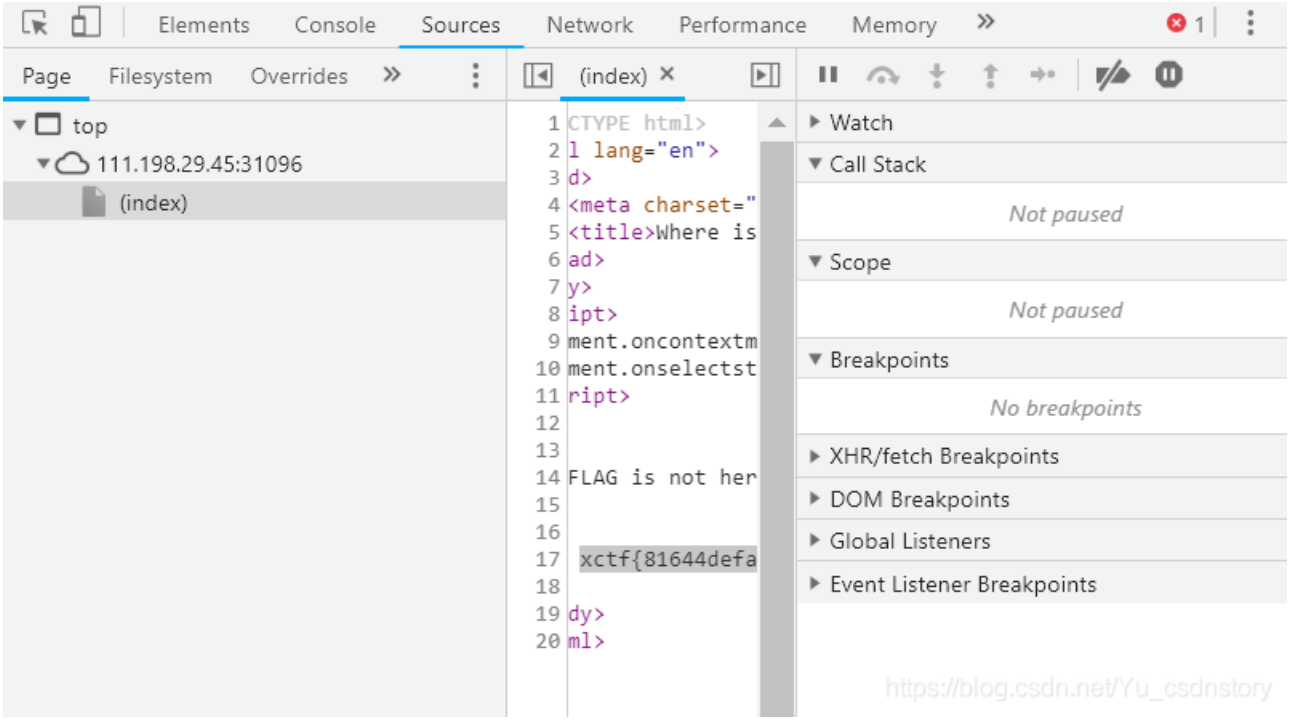
- web
- 网络安全

攻防世界新手第一题writeup

点击传送地址为：http://111.198.29.45:31096/
复制到浏览器打开：

FLAG is not here

按下f12,出现控制台界面，找到source的index,发现flag



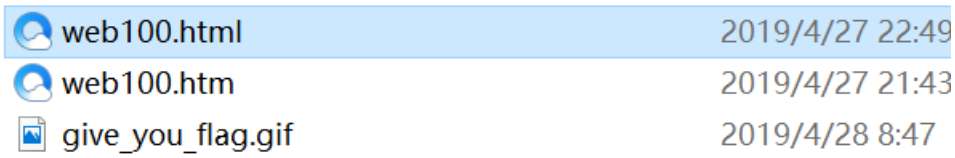
flag为：`xctf{81644defa1410baf36fb7d6f89ac4333}`

NaNNaNNaNNaN-Batman

首先进入场景有个附件，直接保存下载到本地

将它重命名为各种格式，最终发现.html的后缀显示的不是一堆乱码

如图所示：



在浏览器里打开，是一个输入框：

查看网页儿源代码，发现有个eval()，将eval()评估改为alert()，弹出还原后的函数，JavaScript语句。

```
function $(){var e=document.getElementById("c").value;
if(e.length==16)
 if(e.match(/^be0f23/)!=null)
  if(e.match(/233ac e98aa$/)!=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 _
```

拿到flag当然不需要按照函数去计算，首先输入的是16位，按照正则，从e.match()内容里，正则的话^为开始符号，$为结尾符号，拼接一下：be0f233ac7be98aa，输入到输入框，得到flag。

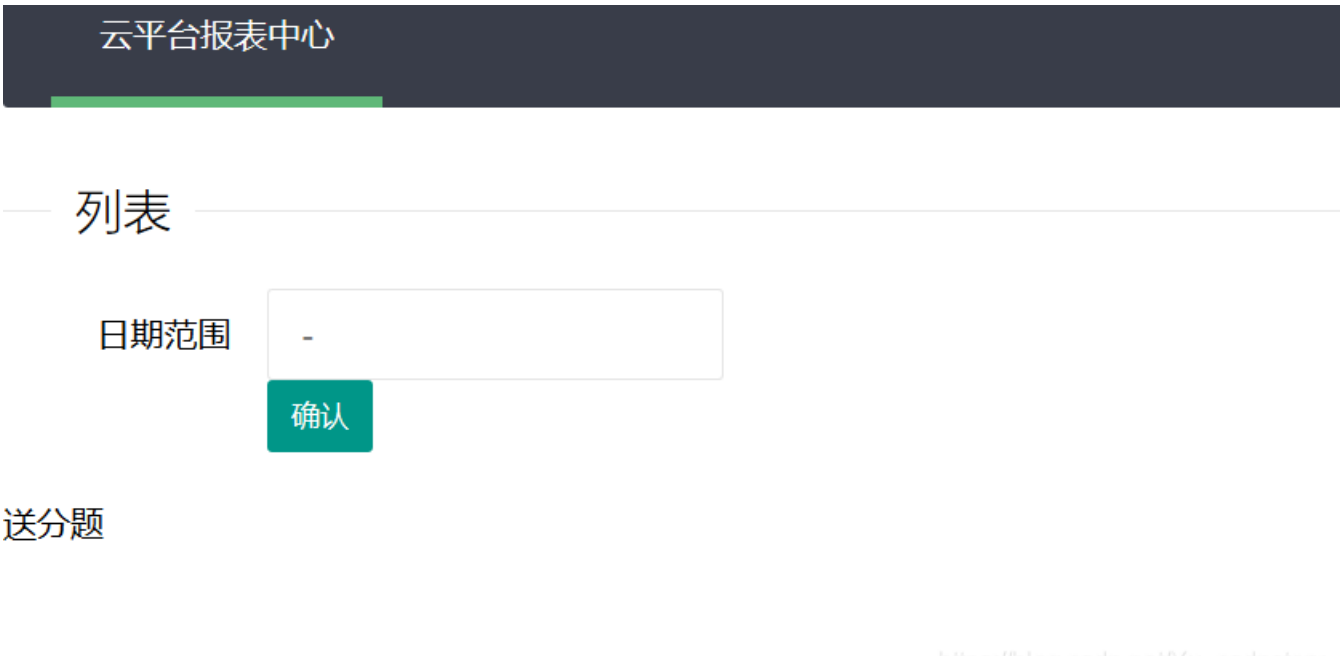
ics-06

难度系数：
题目来源： XCTF 4th-CyberEarth
题目描述：云平台报表中心收集了设备管理基础服务的数据，但是数据被删除了，只有一处留下了入侵者的痕迹。

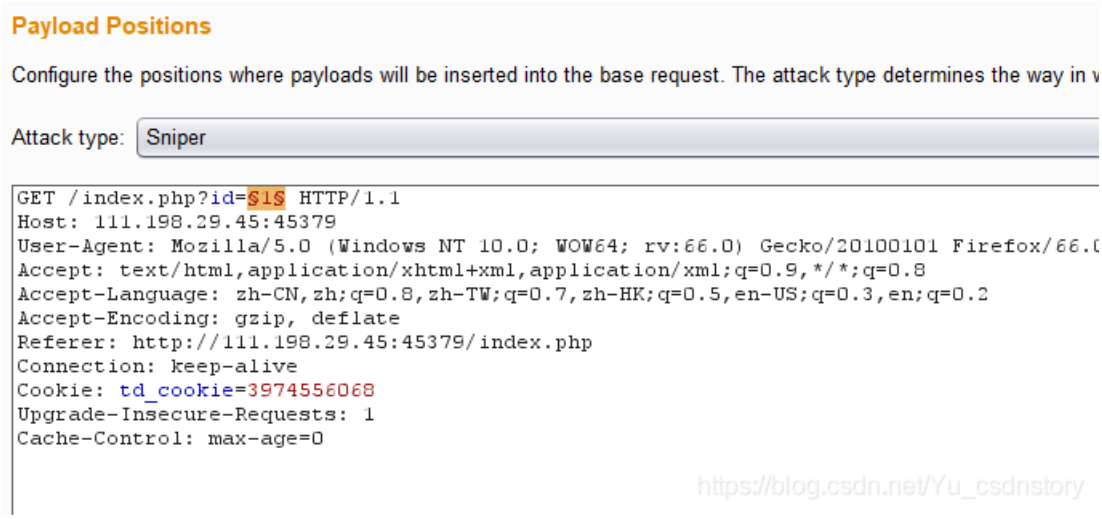
对这道题也是无语了，扫描，获取源码，抓包都试了，还是一无所获，最终还是查看了writeup,分享给大家，别在这道题上浪费时间，真的是没啥意义。wp如图：

没办法，又打开了burpsuite，进行爆破测试

首先，打开传送地址，看到只有一个报表中心处还能显示出点东西，如下图：



抓到这个网页的数据包之后，发送到intruder，选择自己写的三位和四位数的密码本，变量为id(也只有它是数字了）

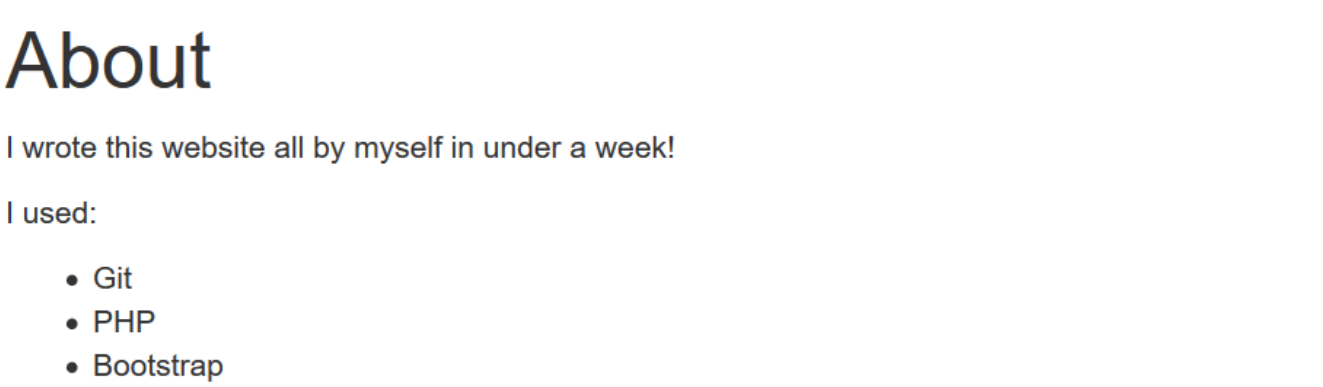


最终在response得到flag

```
cyberpeace{4f2ae395100cf9501bf152deabda88f7}
```

mfw

这道题看着大佬的博客做出来了，费了好长时间，不过学到了点儿知识
看这道题，进入链接，发现很简单，没有什么过多的代码，到about里去看看



试了以下，竟然可以拿到源码，发现了flag.php
打开，发现并没有那么简单，是空的，只好找到index.php,对它进行分析，发现关键

```php
$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!");

?>
<!DOCTYPE html>
```

首先看file这个变量等于template/page/php

首先看file这个变量等于 template/page/php

然后，用了strpos判断file变量里面有没有… 返回true就结束并且输出Detected hacking attempt!

之后判断这个文件是否存在，不存在就结束，并且输出That file doesn't exist!

这几个函数的意思如下

bool **assert** (mixed $assertion [, string $description])

PHP 7

bool **assert** (mixed $assertion [, Throwable $exception])

assert() 会检查指定的 assertion 并在结果为 **FALSE** 时采取适当的行动。

Traditional assertions (PHP 5 and 7)

如果 assertion 是字符串，它将会被 **assert()** 当做 PHP 代码来执行。 assertion 是字符串的优势是当禁用断言时它的开销会更小，并且在断言失败时消息会包含 assertion 表达式。 这意味着如果你传入了 boolean 的条件作为 assertion，这个条件将不会显示为断言函数的参数；在调用你定义的 assert_options() 处理函数时，条件会转换为字符串，而布尔值 **FALSE** 会被转换成空字符串。

断言这个功能应该只被用来调试。 你应该用于完整性检查时测试条件是否始终应该为 **TRUE**，来指示某些程序错误，或者检查具体功能的存在（类似扩展函数或特定的系统限制和功能）。

断言不应该用于普通运行时操作，类似输入参数的检查。 作为一个经验法则，在断言禁用时你的代码也应该能够正确地运行。

strpos（a,b）

判断b是否在a中，是则返回第一次位置，否则返回false

file_exists

(PHP 4, PHP 5, PHP 7)

file_exists — 检查文件或目录是否存在

说明

bool **file_exists** (string $filename)

检查文件或目录是否存在。

参数

filename

文件或目录的路径。

在 Windows 中要用 *//computername/share/filename* 或者 *\\computername\share\filename* 来检查网络中的共享文件。

返回值

如果由 filename 指定的文件或目录存在则返回 **TRUE**，否则返回 **FALSE**。

发现可以构造闭合绕过，大体思路为利用assert函数会将里面的字符串当作php语句，可以构造

?page=ss',‘123’)===false and system(‘cat template/flag.php’)and strpos('

目的是让它执行system的语句，然后在源代码显示，这里的page为**ss',‘123’)===false and system(‘cat template/flag.php’)and strpos('**

在执行第一个assert时，它返回false，执行构造的代码，只要保证strope括号里面的值返回false就能执行system里面的内容，得到flag,执行第二个assert时没有那个文件，输出That file doesn't exist!

```
?page=ss',g)===false and system('cat templates/flag.php')and strpos('
```

查看源代码，得到flag

```
<?php $FLAG="cyberpeace{aa35ec51c23ca62f2a0f73ca33a95fdf}"; ?>
That file doesn't exist!
```