后盾网 人人做后盾

www.houdunwang.com

MySQL安全处理

后盾网 2011-2013 v3.0

以下SQL有安全隐患

$sql = “select uid,username,tel from user where uid=$uid”；

其实以上SQL如果有字符串处理，是根本不会这么写的，但有些时间没有字符串时你可能也习惯这么写，如果真这么写那么问题就来了

假设提交参数为

http://localhost/index.php?uname=1 or 1=1 /*&pwd=123*/注入成功

SQL会变成：

Select id from user where uname=1 or 1=1 /* and passwd=123*/

SQL安全

通过上页我们知道，加引号是多么重要的事情，加上引号看看效果会有什么变化，SQL变为如下形式

$db->query(“set names gbk”);

$uid= $_GET[‘uid’];

$sql = “select id from user where uid= ‘$uid’ ”;

如果我们传入参数如下：

http://localhost/index.php?uid=1' or 1=1%23

结果SQL会变为如下格式，依然可以成功注入

Select id from user name= ‘1’ or 1=1

注：%23为#号 即sql语法中的注释符号

SQL安全

经过上面的操作，发现单纯加引号，还是有注入问题，所以我们要将引号进行转义

PHP.INI的配置文件中有一个选择magic_quotes_gpc，如果这个选项开启时会转义所有外部的GET及POST参数

```
If(!get_magic_quotes_gpc()){
        $uname=addslashes($_GET[ ‘name’ ]);
}
```

经过以上转义处理，引号就不可以注入了，但是隐患依然存在

如果以set names方式设置字符集

如果传入参数如：http://localhost/index.php?name=1%d5%27 or 1=1%23

以上说明:%d5为誠字的高位字节%27为单引号，最终SQL会变成

Select d from user where name= ‘1誠’ or 1=1 #’

依然注入成功

SQL安全

如果设置提交字符集为

```
$sql ='SET
    CHARACTER_SET_CLIENT=BINARY,CHARACTER_SET_CONNECTI
    ON=UTF8,CHARACTER_SET_RESULTS=UTF8'
```

- 表示发送的客户端的字符集为BINARY（二进制），因为二进制不存在字符集问题，所以MSYQL不会将内部的ox5c即转义符\理解为转义符，就不存在注入问题了

SQL安全

Mysqli::set_charset(“gbk”)

- 设置字符集

Mysqli::real_escape_string()

- 考虑到连接的当前字符集的特殊字符转义

如果字符串是gbk编码的，当对oxbf27进行addslashes()时会产生为oxbf5c27这个字符，mysql会将oxbf5c当做一个字符，而把ox27当做单引号处理，有了单引号就可以注入SQL语句，而real_escape_string()方法在处理是是考虑字符集的，从而不会出现这种情况

SQL安全

1. $db= new mysqli(“localhost” ,” root” ,” ” ,” houdunwang”);
2. $db->set_charset(“gbk”);
3. If(get_magic_quotes_gpc()){
4. $title=stripslashes($_POST[‘uname’]);
5. }
6. $uname=$db->real_escape_string($uname);

针对SQL：$sql = “select * from name= ‘$name’ ” ;

如果地址址传参为?title=%bf%27 or 1=1%23则，addslashes将产生注入问题而通过set_charset结合real_escape_string方法操作则不会产生注入危险

SQL安全

- gbk相对注入情况多些，尽量选择UTF8
- 用预准备语句，可以很好的屏蔽注入危险
- 如果使用addslashs进入转义处理，发送客户端字符集为二进制类型
- 可以通过方法Mysqli::set_charset(“gbk”)设置字符集，再通过Mysqli::real_escape_string（ ）函数进行转义处理，屏蔽注入

总结