SQL注入简介

< 上一节 下一节 >

SQL 注入是一种代码渗透技术，是最常用的网络黑客技术之一。SQL 注入非常危险，可能会导致数据库中的数据被暴露，甚至被损坏。

通过网页输入框（<input> 标签、<textarea> 标签等）将恶意 SQL 代码提交给服务器是最常见的 SQL 注入方式之一。

当网站要求输入诸如用户名（用户ID）之类的内容时，通常会发生 SQL 注入。黑客会输入一条 SQL 语句，而不是用户名/用户ID，当页面被提交以后，我们将不知不觉地在数据库上运行这条恶意的 SQL 语句。例如，下面的代码会将 userID 参数拼接到 SQL 语句中，从而构建 SELECT 查询，从数据库中获取当前用户的所有信息。

```
01. demoUserID = getrequestString("userID");
02. demoSQL = "SELECT * FROM users WHERE id =" + demoUserID;
```

SQL 注入的危害

SQL 注入会带来很多危害，包括但不限于：

- 骗过登录校验，查看用户登录后的详细信息（例如发布的评论、购买的商品、邮寄地址等），这是 SQL 注入的最简单形式；
- 更新、删除和插入记录，破坏数据库中的数据；
- 在服务器上执行命令，该命令可以下载和安装木马等恶意程序；
- 将有价值的用户数据（例如邮箱、密码、信用卡等）导出到攻击者的远程计算机。

SQL 注入示例

现在有一个查看员工信息的页面，该页面允许所有员工通过输入自己的 ID 来查看个人信息。假设员工 ID 在数据表中的字段名为 id，现在有黑客在 <input> 文本框中输入以下内容：

```
236893238 OR 1=1
```

它将被拼接成下面的 SQL 语句：


```
SELECT * FROM employee WHERE id = 236893238 OR 1=1;
```

这条 SQL 代码是有效的，将从 employee 表中返回所有符合条件的记录。`1=1` 始终成立，这条 SQL 语句将返回 employee 表中的所有记录，这意味着，所有的员工信息都将被泄露。

类似的，黑客还可以骗过登录校验，使用无效的用户名和密码登录：

```
SELECT * FROM employee WHERE (username="" or 1=1) AND (password="" or 1=1);
```

有些数据库支持批处理 SQL 语句，也即一组由分号 `;` 分隔的两条或者多条 SQL 语句。下面给出的 SQL 语句将返回 employee 表的所有行，然后删除 employee_add 表：

```
SELECT * FROM employee; DROP TABLE employee_add;
```

防止 SQL 注入

SQL 注入不能杜绝，只能尽力防止，因为即使最优秀的程序员也会犯错。Web 防火墙可以检测和阻止最基本的 SQL 注入攻击，但是它仅仅是一种预防手段，我们还要从自己的代码入手，检测用户输入的内容。永远不要信任用户提供的数据，仅在校验通过后才能将数据提交给数据库。

通常使用模式匹配（Pattern Matching），借助正则表达式来校验用户输入的数据，几乎每种编程语言都提供了模式匹配函数。

下面是一段 PHP 代码，它使用 preg_match() 校验用户输入的数据，限定用户名只能包含汉字、字母、数字、下划线 `_` 和连字符 `-`：

```
01. if (preg_match("/^[\x{4e00}-\x{9fa5}0-9A-Za-z_\-]{2,20}$/u", $_POST['username'],
$matches)) {
02.     $result = mysql_query("SELECT * FROM user WHERE name = $matches[0]");
03. } else {
04.     echo "Tips from c.biancheng.net: User name not accepted!";
05. }
```

此外，您还可以结合 mysql_real_escape_string() 函数，它用来转义 SQL 语句中的特殊字符（在特殊字符前面加反斜杠 `\`），比如 `'` 和 `"`，请看下面的例子：

```
01. // 去除斜杠
02. if (get_magic_quotes_gpc()) {
03.    $name = stripslashes($name);
04. }
05. // 对特殊字符进行转义
```

```
$name = mysql_real_escape_string($name);
mysql_query("SELECT * FROM user WHERE name='{$name}'");
```

对于 LIKE 查询，应该使用 addcslashes() 函数对用户输入的 % 和 _ 字符进行转义。addcslashes() 允许用户指定要转义的字符，请看下面的代码：

```
$sub = addcslashes(mysql_real_escape_string("%str"), "%_");
// 转换以后的 $sub == \%str\_
mysql_query("SELECT * FROM messages WHERE subject LIKE '{$sub}%'");
```

关注公众号「站长严长生」，在手机上阅读所有教程，随时随地都能学习。本公众号由C语言中文网站长亲自运营，长期更新，坚持原创。

微信扫码关注公众号

< 上一节 下一节 >

优秀文章

Go语言type关键字（类型别名）
VS2015使用教程（使用VS2015编写C语言程序）
Java冒泡排序法（非常重要）
什么是查找表
C语言strcmp()函数：比较两个字符串
Java项目实战：计算平均成绩
PHP new：实例化对象
结合实例分析Linux权限对指令执行的影响
Servlet的部署和访问
Spring MVC表单标签库