SQL 注入攻击

W4.1. 假设数据库的 password 和 eid 列只存储 SHA-256 哈希值。下面的 SQL 语句会发送到数据库，其中变量 $passwd 和 $eid 的值由用户提供。程序是否存在 SQL 注入漏洞？如果存在，请说明如何利用。

```
$sql = "SELECT * FROM employee
  WHERE eid=SHA2('$eid', 256) and password=SHA2('$passwd', 256)";
```

W4.2. 本题与问题 W4.1. 类似，但哈希值不在 SQL 语句中计算，而是由 PHP 代码调用 hash() 计算。修改后的程序是否仍存在 SQL 注入漏洞？

```
$hashed_eid = hash('sha256', $eid);
$hashed_passwd = hash('sha256', $passwd);
$sql = "SELECT * FROM employee
        WHERE eid='$hashed_eid' and password='$hashed_passwd'";
```

W4.3. 如果按以下方式构造 SQL 语句（注意 WHERE 子句中的换行），还能实施有效的 SQL 注入攻击吗？

```
   SELECT * FROM employee
   WHERE  eid= '$eid' AND
      password='$password'
```

W4.4. 以下 SQL 语句用于向数据库添加新用户，其中 $name 和 $passwd 变量的内容由用户提供，但 EID 和 Salary 字段是由系统设置的。恶意员工如何将自己的工资设置为高于 80000？

```
$sql = "INSERT INTO employee (Name, EID, Password, Salary)
        VALUES ('$name', 'EID6000', '$passwd', 80000)";
```

W4.5. 下面的 SQL 语句用于修改用户姓名和密码，其中变量 $eid、$name、$oldpwd 和 $newpwd 的内容都由用户提供。你希望把老板 Bob 的工资改为 $1（通过 Salary 字段），同时把他的密码改成一个已知值，以便之后登录他的账户。

```
$hashed_newpwd = hash('sha256', $newpwd);
$hashed_oldpwd = hash('sha256', $oldpwd);
```

```
$sql = "UPDATE employee
        SET name='$name', password='$hashed_newpwd'
        WHERE eid = '$eid' and password='$hashed_oldpwd'
```

W4.6. 下面的 SQL 语句被发送到数据库，其中 `$eid` 和 `$passwd` 包含用户提供的数据。攻击者想要尝试让数据库运行任意 SQL 语句。攻击者应将什么放入 `$eid` 或 `$passwd` 以实现该目标？假设数据库确实允许执行多个语句。

```
$sql = "SELECT * FROM employee
        WHERE eid='$eid' and password='$passwd'"
```

W4.7. MySQL 允许用分号连接两条 SQL 语句。能否利用 SQL 注入漏洞让受害服务器执行任意 SQL 语句？

W4.8. 为抵御 SQL 注入，某 Web 应用只在客户端实现过滤：用户输入数据的页面用 JavaScript 删除撇号、注释符和 SQL 保留字等特殊内容。假设过滤逻辑本身十分完善，能删除数据中的所有代码，这种方案能防止 SQL 注入吗？

W4.9. 以下 PHP 代码安全吗？

```
$conn = new mysqli("localhost", "root", "seedubuntu", "dbtest");
$sql = "SELECT Name, Salary, SSN
        FROM employee
        WHERE eid= '$eid' and password=?";

if ($stmt = $conn->prepare($sql)) {
   $stmt->bind_param("s", $pwd);
   $stmt->execute();

   ...
}
```

W4.10. 请使用预处理语句修改以下程序。

```
$sql = "UPDATE employee SET password='$newpwd'
        WHERE eid ='$eid' and password='$oldpwd'";
```

W4.11. SQL 注入使远程用户能够让数据库执行代码。在典型部署中，数据库只允许 Web 应用服务器访问，远程用户没有与数据库交互的直接通道。用户是如何把代码注入数据库的？

W4.12. 为了在 C 程序需要调用外部程序时抵御代码注入攻击，我们不应使用 `system()`；相反，我们应该使用 `execve()`。请描述该对策与防御 SQL 注入所使用的预处理语句有什么相似之处？