**Практическая работа № 17.**

**Список значений в SQL в PHP**

В предыдущем уроке мы решили проблему с подменой значения, просто жестко задав его в коде. Это, однако, не всегда работает. Может быть так, что пользователю разрешено выбрать одно из нескольких значений.

К примеру, у нас на сайте может быть несколько ролей: админ, ученик и учитель. Пусть пользователь может зарегистрироваться по своему выбору: или учеником, или учителем. Сделаем для этого выпадающий список:

<form action="" method="POST">

<input name="login">

<input name="password" type="password">

<select name="role">

<option value="2">ученик</option>

<option value="3">учитель</option>

</select>

<input type="submit">

</form>

Теперь у нас нет вариантов - мы обязательно должны вставлять роль из переменной:

<?php

$login = **$\_POST**['login'];

$password = **$\_POST**['password'];

$role = **$\_POST**['role'];

$query = "INSERT INTO users SET

login='**$login**',

password='**$password**',

role=**$role**

";

mysqli\_query($link, $query);

?>

Но мы опять возвращаемся к пробеме с подменой. Ведь злоумышленник опять может поменять значение поля, в нашем случае уже селекта, на 1 и стать админом.

В этом случае есть два варианта решения проблемы: черный список и белый список. Давайте их рассмотрим.

Черный список

Под *черным списком* имеется ввиду набор значений, которые мы запрещаем. В нашем случае мы хотим запретить роль со значением 1. В этом случае защищенный код будет следующем:

<?php

$login = **$\_POST**['login'];

$password = **$\_POST**['password'];

$role = **$\_POST**['role'];

if ($role !== 1) { // 1 в черном списке

$query = "INSERT INTO users SET

login='**$login**',

password='**$password**',

role=**$role**

";

mysqli\_query($link, $query);

} else {

// попытка взлома

}

?>

Белый список

Черные списки, однако, как правило - плохое решение. Более безопасно делать списки с разрешенными значениями. Такие списки называются *белыми*.

Давайте исправим нашу уязвимость с помощью белого списка разрешенных значений:

<?php

$login = **$\_POST**['login'];

$password = **$\_POST**['password'];

$role = **$\_POST**['role'];

$list = [2, 3]; // белый список

if (in\_array($role, $list)) {

$query = "INSERT INTO users SET

login='**$login**',

password='**$password**',

role=**$role**

";

mysqli\_query($link, $query);

} else {

// попытка взлома

}

?>

Практические задачи

№1

Воспроизведите пример, приведенный в уроке. Проверьте наличие уязвимости. Устраните ее.

№2

В приведенном коде также есть возможность провести SQL-инъекцию. Придумайте, как ее сделать. Воспользуйтесь уязвимостью. Устраните уязвимость.