**Практическая работа № 16.**

**Подмена значения в SQL в PHP**

Давайте рассмотрим следующую уязвимость. Она связана с тем, что ваш код не отслеживает допустимые значения параметров и злоумышленник может произвольно их изменять.

Посмотрим на примере. Пусть в таблице с юзерами поле role имеет значение 1 для админа и 2 для обычного юзера.

Пусть у вас есть следующая форма для регистрации. В этой форме вы жестко задаете, что новый юзер будет обычным, не админом. Вы решили сделать это с помощью скрытого инпута:

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

<input name="login">

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

<input name="role" type="hidden" value="2">

<input type="submit">

</form>

По нажатию на кнопку отправки форму логин, пароль и роль пользователя должны занестись в базу данных с помощью INSERT запроса, вот так:

<?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);

?>

Проблема тут в том, что злоумышленник может легко поменять роль нового пользователя. Ведь то, что поле скрыто - совсем не мешает увидеть его в отладчике браузера и отредактировать его значение. Таким образом злоумышленник может сменить значение на 1 и зарегистрировать нового пользователя в качестве администратора.

Чтобы избавиться от такой уязвимости, мы должны жестко прописать роль нового пользователя в SQL запросе:

<?php

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

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

$query = "INSERT INTO users SET

login='**$login**',

password='**$password**',

role=2

";

mysqli\_query($link, $query);

?>

№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 уявимостью. Устраните уязвимость.