**Практическая работа № 13.**

**SQL инъекция в строковый параметр в PHP**

Когда вы выполняете SQL запрос с данными, полученными от пользователя, этот пользователь может быть злоумышленником и осуществить атаку на сайт. Эта атака называется *SQL инъекция*. Давайте посмотрим, как проводится такая атака, чем это грозит и как он ее защититься.

Пусть у нас есть форма, в которую вводятся логин и пароль для авторизации пользователя:

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

<input name="login">

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

<input type="submit">

</form>

Получим введенные пользователем данные в переменные:

<?php

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

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

?>

Осуществим теперь SQL запрос на предмет того. прошел пользователь авторизацию или нет:

<?php

$query = "SELECT \* FROM users WHERE login='**$login**' AND password='**$password**'";

$res = mysqli\_query($link, $query);

$user = mysqli\_fetch\_assoc($res);

if (!empty($user)) {

// прошел авторизацию

} else {

// неверно ввел логин или пароль

}

?>

Пусть наш код работает с тестовой таблицей users, которая была в предыдущих уроках. Пусть должен авторизоватся админ. В этом случае мы ожидаем, что будет выполнен следующий SQL код:

**SELECT** \* **FROM** users **WHERE** login='admin' **AND** password='abcde'

Наш код, однако, имеет уязвимость. Давайте посмотрим, как злоумышленник может воспользоваться ей и авторизоваться под админом, не зная его пароля (и даже логина).

Вариант 1

Пусть злоумышленник знает логин админа. Как правило, это вполне реальная ситуация, так как логин - это открытая информация и видна, например, в переписке.

В этом случае злоумышленник может ввести в инпут для логина следующий текст:

admin' --

Так как мы сразу подставляем текст из инпутов в запрос, то фактически у нас выполнится следующий SQL:

**SELECT** \* **FROM** users **WHERE** login='admin' --' AND password='abc'

Давайте разберем полученный запрос. Текст -- является комментарием SQL, а значит дальнейшая часть запроса просто не будет выполнена.

И фактически у нас получится следующий запрос:

**SELECT** \* **FROM** users **WHERE** login='admin'

То есть злоумышленник отсек пароль в запросе и спокойно входит на сайт без пароля!

Вариант 2

Можно также авторизоваться под админом, вообще не зная логина. Но зная, что поле role содержит 1 для админа.

В этом случае в поле с логином следует ввести следующее значение:

' or role=1 --

В итоге наш запрос станет таким:

**SELECT** \* **FROM** users **WHERE** login='' **OR** role=1 --' AND password=''

А фактически вот таким, если отбросить закомментированную часть:

**SELECT** \* **FROM** users **WHERE** login='' **OR** role=1

Вариант 3

Можно также авторизоваться не под амидном, а вообще под любым юзером по его id. Для этого нужно вбить в поле с логином следующий текст:

' or id=1 --

В итоге запрос станет таким:

**SELECT** \* **FROM** users **WHERE** login='' **OR** id=1 --' AND password=''

А фактически вот таким:

**SELECT** \* **FROM** users **WHERE** login='' **OR** id=1

Защита

Давайте теперь разберемся, как защититься от такой инъеции. Для защиты от инъекции такого типа нужно обрабатывать все входящие строковые данные функцией mysqli\_real\_escape\_string:

<?php

$login = mysqli\_real\_escape\_string($link, **$\_POST**['login']);

$password = mysqli\_real\_escape\_string($link, **$\_POST**['password']);

?>

Замечания

Здесь следует подчеркнуть, что речь идет именно про строковые параметры. С числовыми параметрами **этот способ не работает**. Защиту числовых параметров мы рассмотрим в следующем уроке.

Также следуюет подчеркнуть, что использование mysqli\_real\_escape\_string - самый примитивный вариант. Более продвинутым вариантом является использование *подготовленных запросов*. Работу с ними мы будем изучать в разделе, посвященном расширению PDO для работы с базами данных.

Практические задачи

№1

Опробуйте все описанные варианты использования уязвимости. Устраните уязвимость. Убедитесь, что она исчезла.

№2

Устраните уязвимость в следующем коде:

<?php

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

$query = "SELECT \* FROM users WHERE login='**$login**'";

$res = mysqli\_query($link, $query);

$row = mysqli\_fetch\_assoc($res);

var\_dump($user);

?>