Фильтрация данных в PHP.

Каждый web-мастер должен уметь не только писать скрипты, но и грамотно организовывать защиту своих творений. Одним из важнейших навыков является умение правильно фильтровать всю информацию, поступающую от пользователя. Об этом и пойдет речь в моей статье.

Прежде всего, следует фильтровать данные, которые передает пользователь осознанно - в основном, это данные различных форм. Это может быть пара логин-пароль для входа, пункт голосования и т.п. Например, такая форма

```
<form action="index.php" method="GET">
<input type="text" name="login">
<input type="text" name "pass">
<input type="submit" value="OK">
</form>
```

После нажатия кнопки "OK" передаст скрипту index.php два значения - $login и $pass. Как их можно отфильтровать? Пример для переменной $login:

```
if($login)
{
$login = htmlspecialchars((stripslashes($login)), ENT_QUOTES);
$login = str_replace("/","",$login);
$login = str_replace(".","",$login);
$login = str_replace("`","",$login);
}
else
{
echo "Логин не введен!";
}
```

В первой строке мы проверяем существование переменной $login, если она существует - идем дальше, если нет - выводим сообщение об ошибке. Затем с помощью функции **htmlspecialchars** заменяем в этой переменной спецсимволы на их HTML мнемоники. То есть знак `<` меняется на `<`, `&` меняется на `&` и т.д. Функция **stripslashes** вырезает знак обратного слеша `\`. Далее с помощью **str_replace** вырезаем знак прямого слеша, точку (иногда бывает полезно) и обратную кавычку.

Если вы знакомы с регулярными выражениями, то предыдущий пример можно записать гораздо короче:

```
if($login)
{
if (preg_match("/[0-9a-z_]/i", $login))
{
// … действия над логином …
}
else
{
```

```
echo "Логин введен неверно!";
}
}
else
{
echo "Логин не введен!";
}
```

Этот фрагмент кода будет проверять введенный логин на соответствие регулярному выражению `/[0-9a-z_]/i`, которое означает: все цифры + все латинские буквы в любом регистре + знак подчеркивания. Если логин содержит другие символы, то будет показано сообщение об ошибке.

Аналогично фильтруются переменные, получаемые скриптом через URL. В движках сайтов можно встретить что-то вроде таких ссылок:

http://www.site.com/index.php?module=news

Если не фильтровать переменную $module (или $_GET[`module`], если **register_globals** отключен), то над сайтом могут вытворяться не очень хорошие вещи, вроде XSS. Нужно применять первый приведенный мной скрипт-чистильщик, разумеется, убрав сообщения об ошибках.

Следующее, на чем бы я хотел остановиться - это фильтрация кукисов. Думаю, что даже если вы начинающий программист, то с "плюшками" вы сталкивались, а насчет их проверки даже не задумывались. Зря! Если вы используете SQL-базы данных, то отсутствие проверки кукисов может привести к использованию хакерами SQL-injection. Так как в кукисах, в основном, мы используем определенный тип данных, например, только числа, то проверку данных можно проводить с помощью все тех же регулярных выражений. Допустим, у нас есть кукис "id", в котором хранятся числовые данные. Его проверка:

```
if($_COOKIE[`id`])
{
if (preg_match("/[0-9]/", $_COOKIE[`id`])
{
// … действия над кукисом …
}
else
{
echo "Хм, странный кукис. Не пойдет!";
}
}
```

В сети есть огромное количество документации посвященной взлому сайтов, в частности, там показаны различные случаи использования XSS и приемы обхода фильтрации. Но мы то с вами уже умеем защищаться от непрошенных гостей ;)

Автор статьи: **SapienS**
Сайт: **http://wcode.ru/**