Способы защиты формы и баз данных

Аннотация

Сорока А.С., Завадская Т.В. Способы защиты формы и баз данных.
В данной статье рассмотрены способы защиты формы, как защитится от SQL-инъекции шифровка данных, способы защиты от скриптов. Был проведен анализ этих способов и протестирован на динамическом сайте и CMS WordPress.

Ключевые слова: *CMS* WordPress, SQL-инъекции, база данных, MSQLI, хеш, md5.

Введение. Сейчас очень популярно и полезно иметь свой собственный сайт, будь то просто блог или бизнес сайт. Потребность в них растет т.к., в наше время большая часть времени и работы опирается на интернет. Всегда востребованы люди, которые не только могут создать нужный сайт, но и защитить его. В ходе разработки всегда появляются различные проблемы: защита сайта, его быстродействие, нагрузки, с которыми будет работать сервер.

Постановка проблемы. В любом сайте присутствуют уязвимые места, например: форма в PHP обработчике . Если она буде сделана без защиты, то она может отправить на сервер запрос, скрипт, опасный или заражённый файл, можно будет произвести SQL-инъекцию. В таких случаях можно потерять важные данные или лишиться своего сайта. Всегда при написании серверной части рассматривается безопасность данных и самого сайта, какой пользователь захочет пользоваться сайтом, если в любой момент он лишится всех своих данных.

Основные драйвера реляционных СУБД. Перед тем, как разработчик начет делать свою работу, он должен выбрать каким драйвером реляционной СУБД будет пользоваться. Есть следующие драйвера:

- Функции mysql являются процедурами и используют ручное экранирование. Данные функции уже не используют, т.к. с появлением PHP 5 начали возникать проблемы различного рода. Разработка была остановлена на Mysql 4.1.3, это расширение не поддерживает транзакции, имеет уязвимости при подстановке в запрос. Mysql функциям пришли на замену: MySQLi и PDO. [5,2,3]
- MySQLi является заменой функций mysql, с объектно-ориентированными и процедурными версиями. Он поддерживает подготовленные заявления. В отличие от PDO является прямым наследником, API Mysql. Имеет следующие улучшения: объектно-ориентированный интерфейс, поддержка множественных операторов, поддержка транзакций, расширенная поддержка отладки. [5,2]
- PDO (PHP Data Objects) - это общий уровень абстракции базы данных с поддержкой MySQL среди многих других баз данных. Он предоставляет подготовленные заявления и значительную гибкость в отношении того, как данные возвращаются. Очень хорошо подходит, если нужно сменить базу данных. Имеет возможность работать с несколькими БД, что позволяет “забыть” какую базу использовали, за нас это будут делать специальные драйвера. Из методов борьбы с sql-инъекциями появилось prepared. Prepared statement — это заранее

скомпилированное SQL-выражение, которое может быть многократно выполнено путём отправки серверу лишь различных наборов данных. [5,4]

Будем использовать MySqli т.к., PHP разработчики это рекомендуют.

Моделирование. Сперва рассмотрим SQL-инъекции и способы защиты от скриптов. SQL-инъекции – это один из распространённых способов взлома сайта, который отправляет запрос в базу данных и может получить нужную информацию с неё. Чтобы избежать ее можно использовать mysqli вместо mysql или PDO при вызове функций работы с базой данных.

На рис.1 можно увидеть форму в которой написан js скрипт, PHP скрипт, SQL запрос, обычный текст.

Рис.1 Форма с данными

В данном примере происходит обновление поля text, который находится в таблице articles по нужному id адресу, обрабатываемые поля описаны в коде ниже.

$text = htmlentities(mysqli_real_escape_string($connection, $_POST['text-post']));

mysqli_query($connection, "UPDATE articles SET text='$text' WHERE id='(int)$id'")

В $text заносим информацию с формы, но перед этим произойдет обработка через mysqli_real_escape_string, которая заданную строку кодирует в экранированную SQL строку, используя текущий набор символов подключения, а также htmlentities, который преобразует все возможные символы в соответствующие HTML-сущности. Переменной id укажем, что это целое число. В тех случаях если идет обработка формы через post или get, то полям с числами лучше указывать, что они целочисленные. Основное различие методов GET и POST состоит в способе передачи данных веб-формы обрабатывающему скрипту, а именно:

- Метод GET отправляет скрипту всю собранную информацию формы как часть URL
- Метод POST передает данные таким образом, что пользователь сайта уже не видит передаваемые скрипту данные

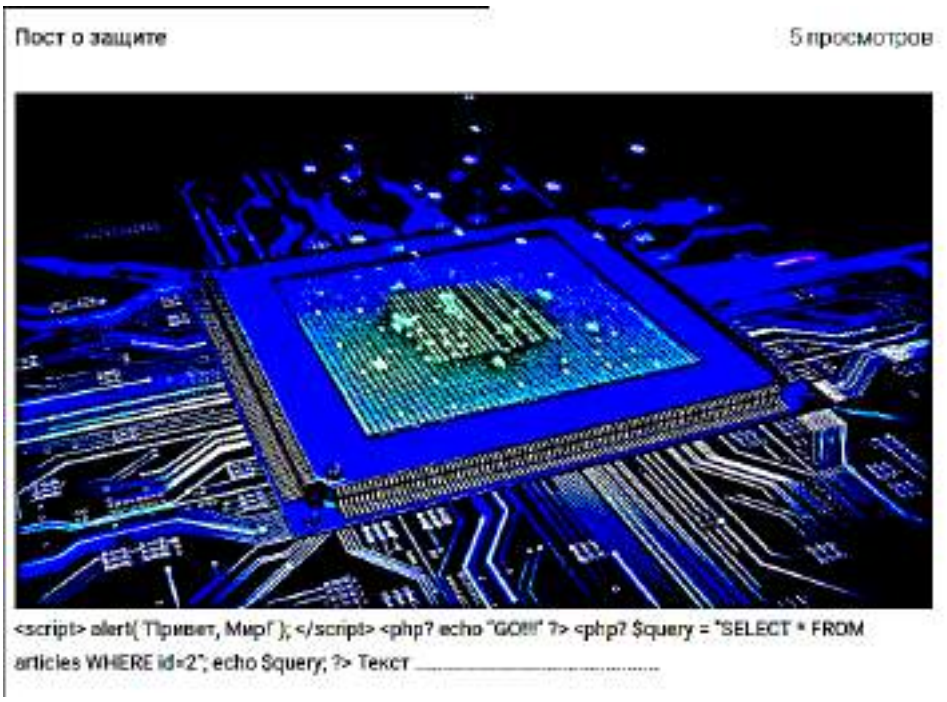

Рис.2 Результат вывода

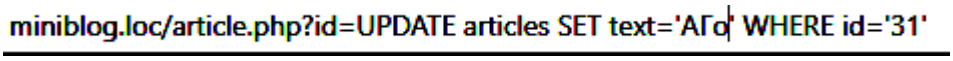

Рис. 3 Пример SQL-инъекции в строке браузера

На рисунке 2 мы можем увидеть, что скрипты не сработали, а преобразовались в строку. Они преобразуются тогда, когда идет отправка через форму. В тех случаях, если зададут SQL-инъекцию в строке поиска (пример на Рис.3), то действий не будет . Методу get было указано принимать в качестве id адреса целое число. В таких случаях нам напишут, что страница не существует, или выдаст белый экран.

Многим формам нужна капча, она поможет от спама в комментариях или от подбора пароля ботом. Для нее лучше всего ставить шрифт, который бот не сможет прочитать. Шрифт всегда можно проверить на определенных сайтах. [3]

Всегда, когда устанавливается CMS, есть пункт префикса таблиц в базе данных (бд), которые пользователи оставляют стандартными. Одним из способов затруднить инъекции в CMS или обычным сайтам – это добавление или изменение префикса таблиц. Такой способ не даст простой доступ к таблице т.к. злоумышленник не будет знать, как называется нужная таблица. [1]

Рис.4 Стандартный префикс

Рис.5 Новый префикс

Получить 100% защиту не так просто, всегда нужно учитывать, что сайт или аккаунт могут взломать и в таких случаях нужно обезопасить данные пользователя. Учетную запись пользователя можно защитить, например: высылкой кодов на телефон или почту, программами аутентификации, способом защищённости пароля.

Первые два способа сложны и требуют много времени реализации, и не всегда нужны такие меры защиты, т.к. для обычного блога будет затратно отправлять СМС

пользователям. Большую часть взломов производят через ботов, которые занимаются подбором паролей и даже не всегда может помочь капча от них. Обычно в различные CMS или обычных сайтах, в базе данных пароли хранят в кодировке MD5(алгоритм MD5 RSA Data Security), который хеширует строки и возвращает этот хэш, который представляет собой 32-значное шестнадцатеричное число. Для более надежной защиты пароля можно использовать несколько MD5 сразу, например:

```
md5('Tortik'.md5('666'.$p1.'228').md5('013'.$p2.'007'));
```

Данный способ задет ключ ` Tortik `, а также первую переменную (пароль) соединяет еще с двумя ключами, при этом хешируя пароль и к этому добавим операнд два (логин), который будет иметь еще два ключа и тоже хеширует его. После того как получи две соединённые хешированные строки мы ее еще раз применим алгоритм md5 и получаем зашифрованный пароль. В результате чтобы раскодировать пароль, нужно знать какие были ключи, а также, на какие части разбить пароль, чтобы убрать лишнее. Такой способ затрудняет взлом. На рисунке 6 можно увидеть зашифрованный пароль:

user_pass
PB1tzPcbWnbkmG93L1rRJ3q1cvAbR5k/

Рис.6 зашифрованный пароль

Иногда нужно отправлять на сервер файлы мультимедиа. В этих файлах могут быть скрипты или они могут быть заражёнными. Рассмотрим случай, когда нам нужно загрузить аватар пользователя:

```
function LoadImg($path)
{ $types = array('image/gif', 'image/png', 'image/jpeg','image/jpg');
   if ($_SERVER['REQUEST_METHOD'] == 'POST')
   { if (!in_array($_FILES['picture']['type'], $types))   return 5;
    if (!@copy($_FILES['picture']['tmp_name'], $path . $_FILES['picture']['name'])) return 3;
    else return 4;}}
```

Данная функция загружает один мультимедийный файл при этом, чтобы не загрузить файл с неверным форматом сделаем массив $types с разрешёнными форматами. В тех случаях, если файл неверный нам вернет результат ошибки формата. Данную функцию можно доработать, сделав так, чтобы файл был переименован. Такой способ позволит исключить скрипты в имени. [1]

Выводы: Подводя итоги было получена защита: формы от инъекций , пароля с помощью шифрования, была разобрана капча и таблицы в базе данных, исследована функция загрузки файлов на сервер и сайт. В результате было положено начало исследования защиты форм, паролей, таблиц бд от несанкционированного доступа. Каждый из методов можно доработать или усложнить на усмотрение пользователя. Данные методы были использованы в реализации модуля. В дальнейшем можно сделать защиту SESSION и COOKIE

Литература:

1. PHP: Простой учебник - Manual [Электронный ресурс]: Руководство по PHP — Режим доступа: http://php.net/manual/ru/index.php

2. Все о PHP, MySQL и не только ! [Электронный ресурс]: Изучение PHP — Режим доступа: http://www.php.su/learnphp/

3. Маклафлин Б. PHP и MySQL. Исчерпывающее руководство. — СПб.: Питер, 2013. — 512 с.

4. PHP 7 / Котеров Д.В., И.В. Симдяков – СПб.: БХВ – Петербург, 2016. – 1088с.: ил. – (В подлиннике)

5. Разница между MySQL, MySQLi и PDO[Электронный ресурс]: stack.io -— Режим доступа: http://qaru.site/questions/54419/what-is-the-difference-between-mysql-mysqli-and-pdo