Jay’s Bank
Application Penetration Testing

Jay’s Bank – 897-19
BUSINESS CONFIDENTIAL

Date: June 1st, 2024
Project: 897-19
Version 1.0

Table of Contents

Confidentiality Statement

This report is confidential and contains proprietary information of Pentester. The Report is provided solely for the use of authorized personnel of Pentester and its affiliates, and may not be disclosed to any third party without the prior written consent of Pentester.

By accessing or reviewing the Report, the recipient acknowledges that the Report contains confidential and proprietary information of Pentester, including but not limited to:

- Sensitive security information and vulnerabilities identified during the penetration testing;
- Proprietary methodologies and tools used during the engagement;
- Confidential business information and trade secrets of Pentester;
- Information about Pentester's systems, networks, and infrastructure.

Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.

The Report is a snapshot of the Client's security posture at the time of the Engagement and may not reflect the current security state. The Report is based on the Tester's observations, testing, and analysis, and may not be exhaustive or definitive.

Contact Information

Name	Title	Contact Information
Pentester		
Arsyad Rizantha Maulana Salim	Information technology Student	Office: (081) 234-567-890 Email: arsyad.rizantha@gmail.com

Assessment Overview

From May 28th, 2024 to June 01st, 2024, Jay's Bank engaged Pentester to evaluate the security posture of its infrastructure compared to current industry best practices that included an external penetration test. All testing performed is based on the NIST *SP 800-115 Technical Guide to Information Security Testing and Assessment, OWASP Testing Guide (v4), and customized testing frameworks*.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.

Scope and Focus

1. SQL Injection: Look for vulnerabilities that allow malicious SQL code to be executed within the application's database.
2. Cross-Site Scripting (XSS): Identify weaknesses that could allow attackers to inject malicious scripts into web pages viewed by other users.
3. Authentication and Authorization Issues: Examine the mechanisms used for user authentication and authorization to find any flaws that could be exploited to gain unauthorized access.

Exploitation Guidelines

- User Account Access: If possible, vulnerabilities should be exploited to demonstrate how an attacker might access other users' accounts within the application.
- Application Only: Exploitation should be confined to the application itself, without extending to the server or other infrastructure component.

Finding Severity Ratings

The following table defines levels of severity and corresponding Pentester score range that are used throughout the document to assess vulnerability and risk impact.

Severity	FT V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system - level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization’s attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Scope

Assessment	Details
External Penetration Test	167.172.75.216

Client Allowances

- Do not perform attacks that can damage data or application infrastructure.
- Do not exploit vulnerabilities that provide server access (e.g., RCE, privilege escalation).
- Avoid DoS/DDoS attacks that disrupt application availability.

Executive Summary

Pentester evaluated Jay’s Bank’s external security posture through an external network penetration test from May 28th, 2024 to June 01st, 2024. By leveraging a series of attacks, Pentester found critical level vulnerabilities that allowed full internal network access to the Jay’s Bank headquarter office. It is highly recommended that Jay’s Bank address these vulnerabilities as soon as possible as the vulnerabilities are easily found through basic reconnaissance and exploitable without much effort.

Security Strengths

During the assessment, the Jay’s Bank’s security team alerted Pentester engineers of detected vulnerability scanning against their systems. The team was successfully able to identify the Pentester engineer’s attacker IP address within minutes of scanning and was capable of blacklisting Pentester from further scanning actions.

Security Weaknesses

Missing Multi-Factor Authentication

Pentester leveraged multiple attacks against Jay's Bank Login forms using valid credentials harvested through open-source intelligence. Successful logins included employee email accounts through Outlook Web Access and internal access via Active Directory login on the VPN. The use of multi-factor authentication would have prevented full access and required Pentester to utilize additional attack methods to gain internal network access.

Weak Password Policy

Pentester successfully performed password guessing attacks against Jay's Bank Login forms, providing internal network access. A predictable password format of Summer2018! (season + year + special character) was attempted and successful.

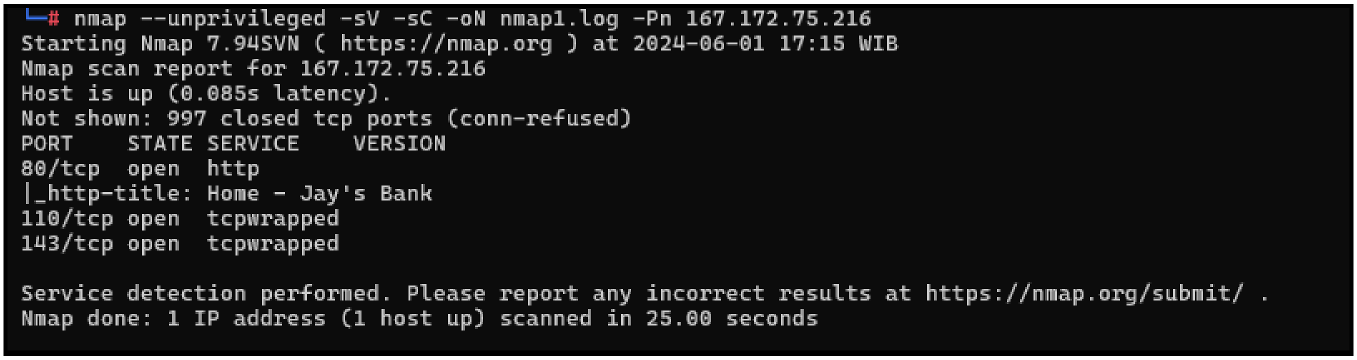
External Penetration Test Findings

Description:	Jay's Bank allowed unlimited login attempts against their Outlook Web App (OWA) services. This configuration allowed brute force and password guessing attacks in which TCMS used to gain access to Jay's Bank internal network.
Impact:	-
System:	http://167.172.75.216/

Exploitation Proof of Concept

- ## First Method

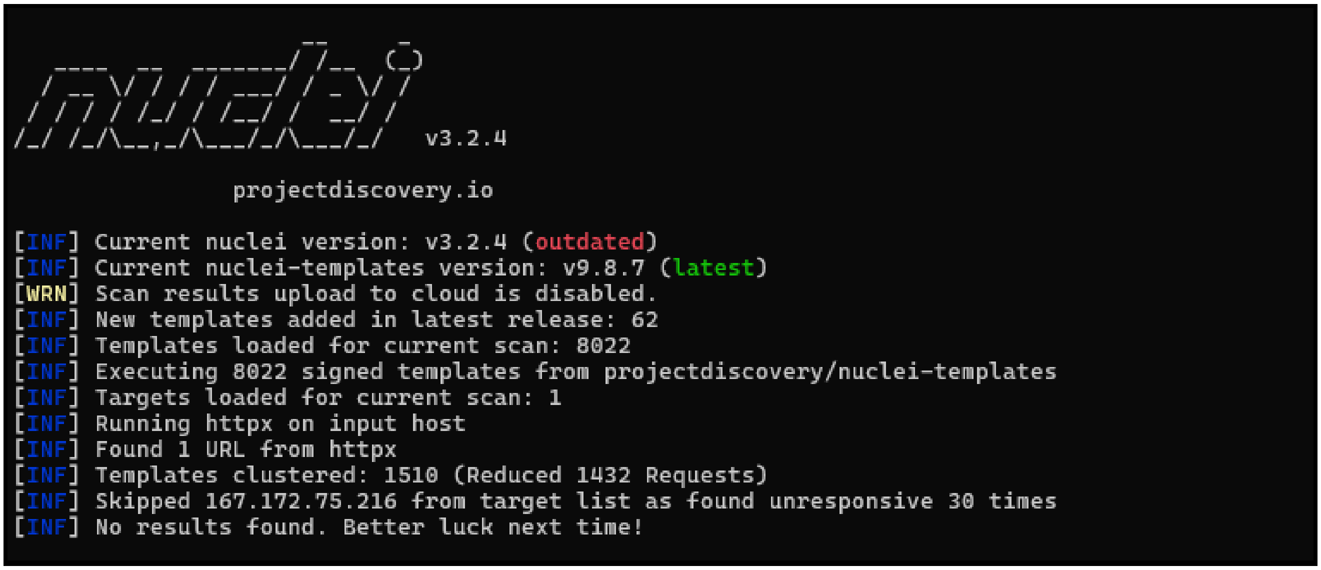
Try checking the IP address by entering the following command in the Kali terminal below: `nmap --unprivileged -sV -sC -oN nmap1.log -Pn 167.172.75.21`.



```
└─# nmap --unprivileged -sV -sC -oN nmap1.log -Pn 167.172.75.216
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-01 17:15 WIB
Nmap scan report for 167.172.75.216
Host is up (0.085s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT    STATE SERVICE    VERSION
80/tcp  open  http
|_http-title: Home - Jay's Bank
110/tcp open  tcpwrapped
143/tcp open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 25.00 seconds
```

Apparently, there are no vulnerabilities detected. Next, try checking the vulnerabilities using this command : `nuclei -u 167.172.75.216 -o testing.txt`.



```
                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.2.4

                projectdiscovery.io

[INF] Current nuclei version: v3.2.4 (outdated)
[INF] Current nuclei-templates version: v9.8.7 (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 62
[INF] Templates loaded for current scan: 8022
[INF] Executing 8022 signed templates from projectdiscovery/nuclei-templates
[INF] Targets loaded for current scan: 1
[INF] Running httpx on input host
[INF] Found 1 URL from httpx
[INF] Templates clustered: 1510 (Reduced 1432 Requests)
[INF] Skipped 167.172.75.216 from target list as found unresponsive 30 times
[INF] No results found. Better luck next time!
```

There are no results found as well.

- ## Second Method

The next method is to go to the IP website which is http://167.172.75.216/, then register by entering the username and password. It can also use the code below to do the register process:

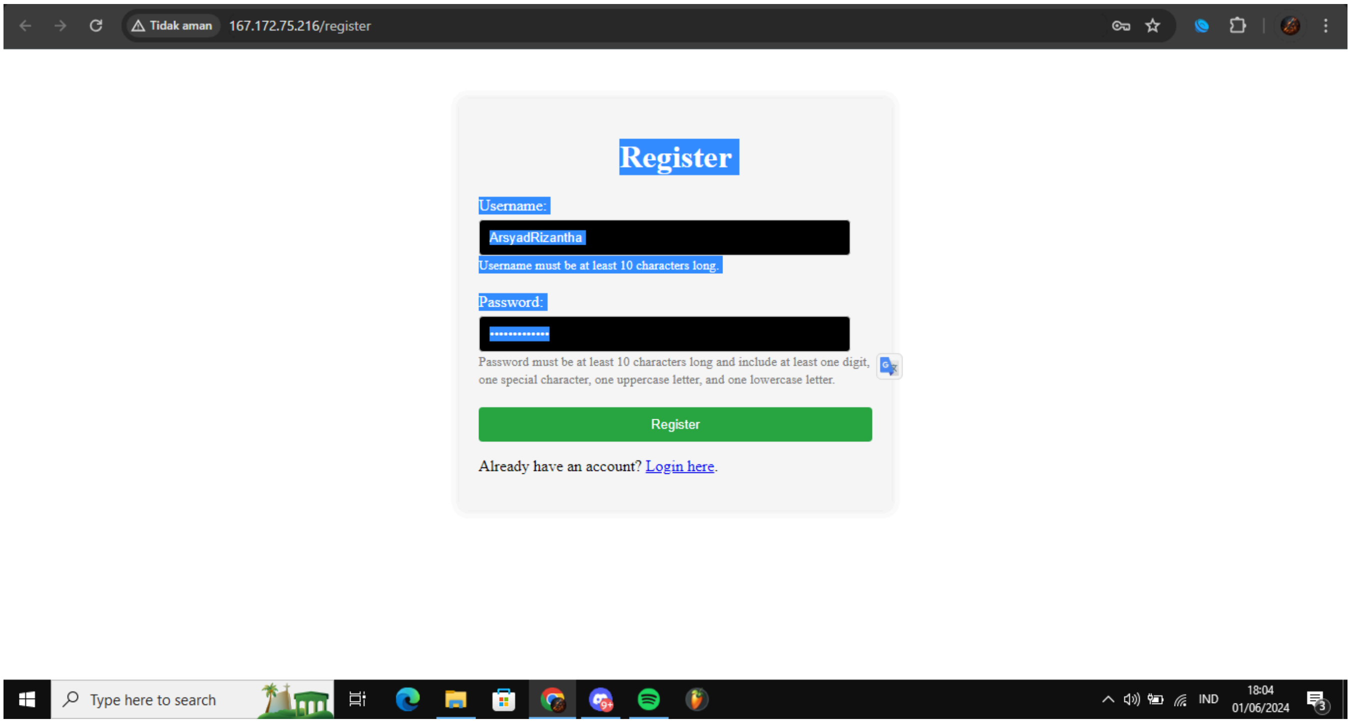
```
import requests
import json

url = "http://167.172.75.216/register"

data = {
    'username': 'ArsyadRizantha',
    'password': 'Anjaymabar12!'
}

response = requests.post(url, headers={'Content-Type': 'application/json'}, data=json.dumps(data))
```

```python
if response.status_code == 200:
    response_data = response.json()
    if response_data.get('success'):
        print("Registration successful!")
    else:
        print(f"Registration failed: {response_data.get('message')}")
else:
    print(f"Registration failed with status code: {response.status_code}")
    print("Response:", response.text)
```



Register

Username:

ArsyadRizantha

Username must be at least 10 characters long.

Password:

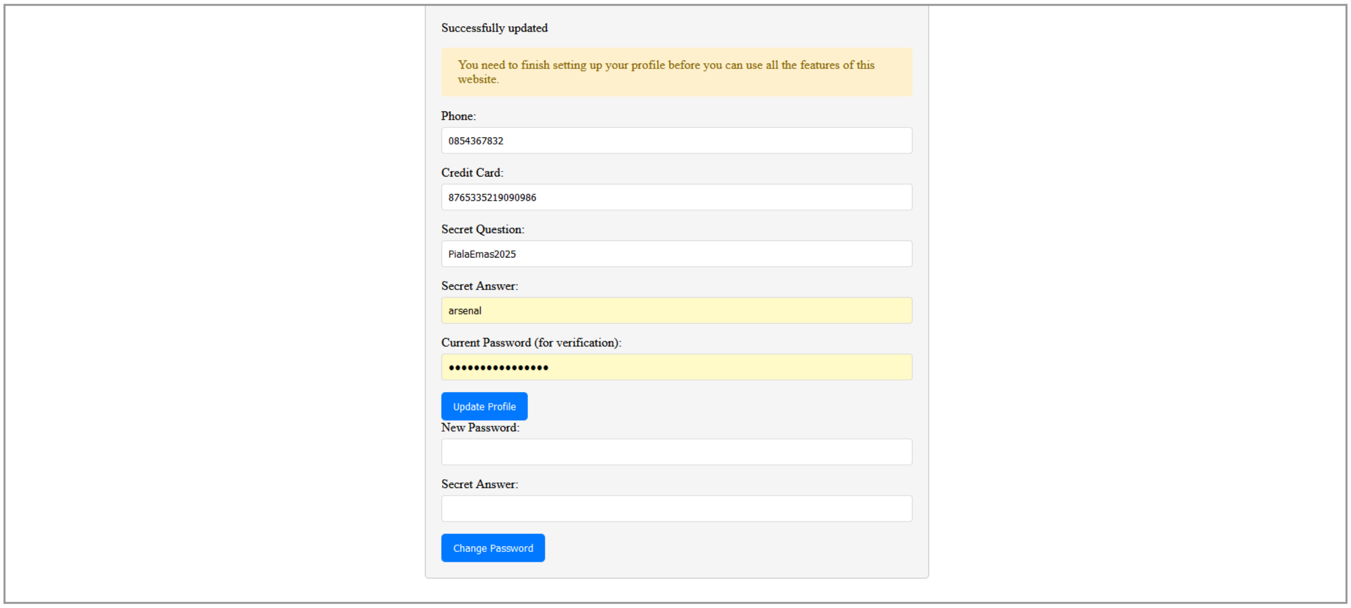
•••••••••••••

Password must be at least 10 characters long and include at least one digit, one special character, one uppercase letter, and one lowercase letter.

Register

Already have an account? Login here.

After registering, the next step is log in/enter on the web login page. After entering, file the data in the table according to what you want, with a maximum number of telephone numbers of 10 and a maximum of 16 credit card numbers.



Successfully updated

You need to finish setting up your profile before you can use all the features of this website.

Phone:

0854367832

Credit Card:

8765335219090986

Secret Question:

PialaEmas2025

Secret Answer:

arsenal

Current Password (for verification):

••••••••••••••••

New Password:

Secret Answer:

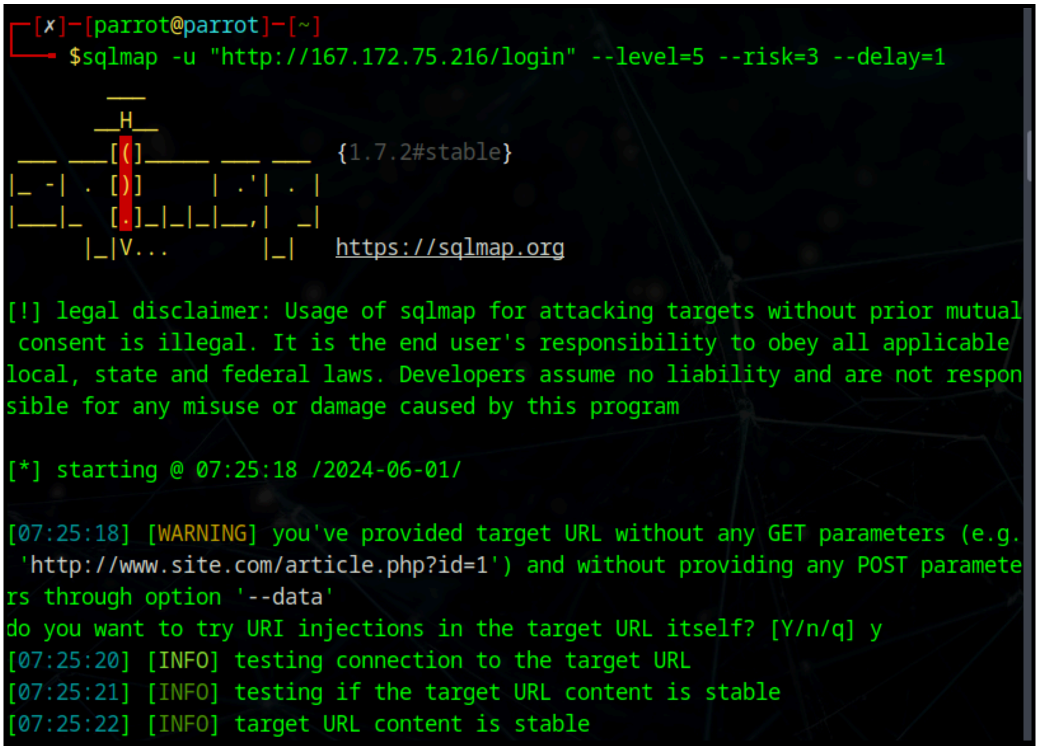
After that, save then check on the dashboard, whether it was successful or not.

- **Third Method**

Use SQLmap to detect what types of SQL are on that IP using the command in the parrot terminal:

```
sqlmap -u "http://167.172.75.216/login" --level=5 --risk=3 --delay=1
```

And then check (it may take quite a long time because the tool is for mapping the database, and exploiting the database).

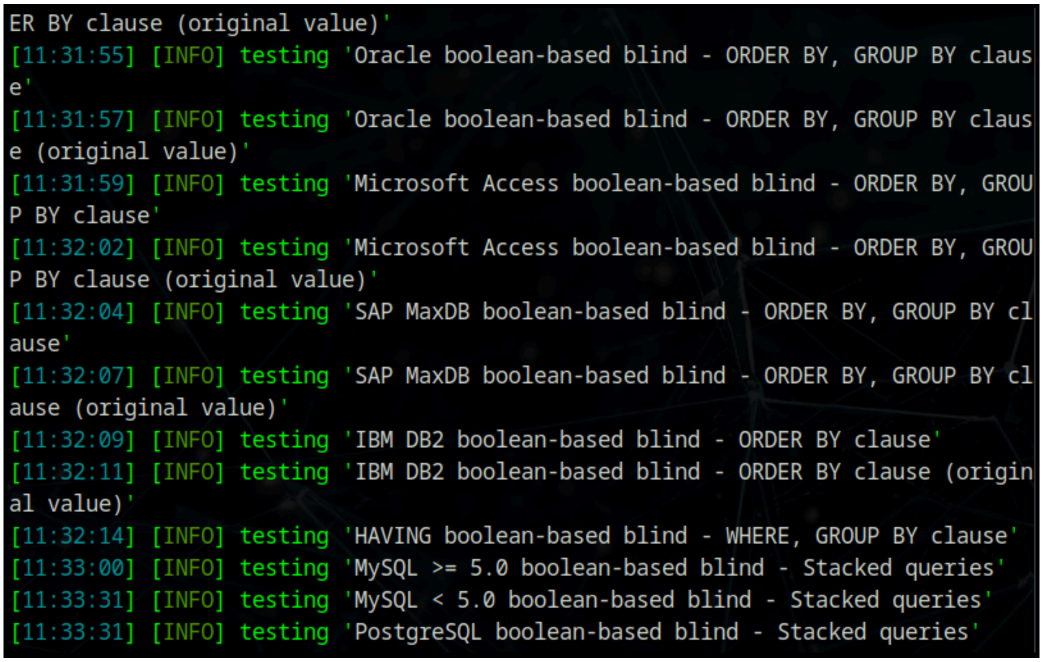


```
┌─[✗]─[parrot@parrot]─[~]
└──╼ $sqlmap -u "http://167.172.75.216/login" --level=5 --risk=3 --delay=1
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.7.2#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] 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 respon
sible for any misuse or damage caused by this program

[*] starting @ 07:25:18 /2024-06-01/

[07:25:18] [WARNING] you've provided target URL without any GET parameters (e.g.
 'http://www.site.com/article.php?id=1') and without providing any POST paramete
rs through option '--data'
do you want to try URI injections in the target URL itself? [Y/n/q] y
[07:25:20] [INFO] testing connection to the target URL
[07:25:21] [INFO] testing if the target URL content is stable
[07:25:22] [INFO] target URL content is stable
```

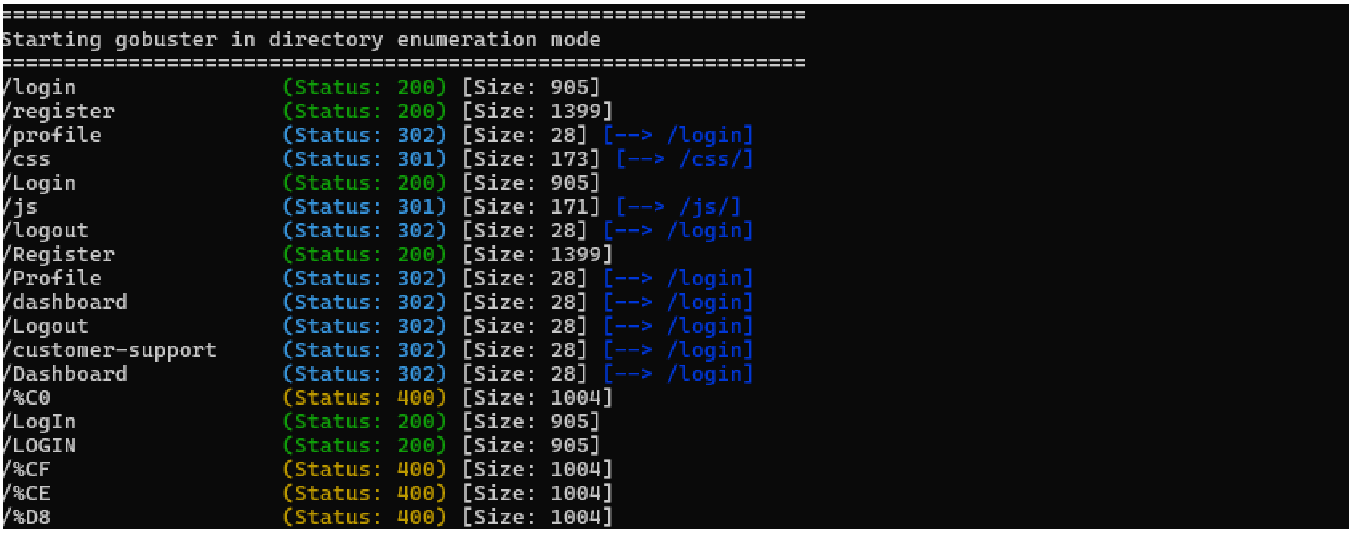


```
ER BY clause (original value)'
[11:31:55] [INFO] testing 'Oracle boolean-based blind - ORDER BY, GROUP BY claus
e'
[11:31:57] [INFO] testing 'Oracle boolean-based blind - ORDER BY, GROUP BY claus
e (original value)'
[11:31:59] [INFO] testing 'Microsoft Access boolean-based blind - ORDER BY, GROU
P BY clause'
[11:32:02] [INFO] testing 'Microsoft Access boolean-based blind - ORDER BY, GROU
P BY clause (original value)'
[11:32:04] [INFO] testing 'SAP MaxDB boolean-based blind - ORDER BY, GROUP BY cl
ause'
[11:32:07] [INFO] testing 'SAP MaxDB boolean-based blind - ORDER BY, GROUP BY cl
ause (original value)'
[11:32:09] [INFO] testing 'IBM DB2 boolean-based blind - ORDER BY clause'
[11:32:11] [INFO] testing 'IBM DB2 boolean-based blind - ORDER BY clause (origin
al value)'
[11:32:14] [INFO] testing 'HAVING boolean-based blind - WHERE, GROUP BY clause'
[11:33:00] [INFO] testing 'MySQL >= 5.0 boolean-based blind - Stacked queries'
[11:33:31] [INFO] testing 'MySQL < 5.0 boolean-based blind - Stacked queries'
[11:33:31] [INFO] testing 'PostgreSQL boolean-based blind - Stacked queries'
```

- **Fourth Method**

Using gobuster dir in kali with the command,

```
gobuster dir -u http://167.172.75.216/ -w /home/grk/gobuster/KaliLists/
dirbuster/directory-list-2.3-medium.txt
```



```
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/login                (Status: 200) [Size: 905]
/register             (Status: 200) [Size: 1399]
/profile              (Status: 302) [Size: 28] [--> /login]
/css                  (Status: 301) [Size: 173] [--> /css/]
/Login                (Status: 200) [Size: 905]
/js                   (Status: 301) [Size: 171] [--> /js/]
/logout               (Status: 302) [Size: 28] [--> /login]
/Register             (Status: 200) [Size: 1399]
/Profile              (Status: 302) [Size: 28] [--> /login]
/dashboard            (Status: 302) [Size: 28] [--> /login]
/Logout               (Status: 302) [Size: 28] [--> /login]
/customer-support     (Status: 302) [Size: 28] [--> /login]
/Dashboard            (Status: 302) [Size: 28] [--> /login]
/%C0                  (Status: 400) [Size: 1004]
/LogIn                (Status: 200) [Size: 905]
/LOGIN                (Status: 200) [Size: 905]
/%CF                  (Status: 400) [Size: 1004]
/%CE                  (Status: 400) [Size: 1004]
/%D8                  (Status: 400) [Size: 1004]
```

After the recon was carried out, it turned out that there were several endpoints that might be accessible.

Last Page