

SIMULATED PHISHING CAMPAIGN AND ANALYSIS

USING KALI LINUX

This is a **practical lab activity** designed specifically for students who have **Kali Linux, Windows 11, OWASP tools, and Metasploitable VMs**. The exercise simulates a **controlled social engineering phishing attack and analysis**, reinforcing the key ideas from the lecture material. The activity connects directly to Week 5 slides discussing **phishing, emotional manipulation, and human-focused attacks** such as urgency, trust, and curiosity.

1. Lab Objectives

The objective of this lab is to demonstrate how **phishing attacks are used in social engineering to obtain sensitive information such as usernames and passwords**.

By completing this lab, students will:

- Understand how attackers manipulate users rather than exploiting systems directly.
- Simulate a **phishing attack in a controlled virtual environment**.
- Observe how credentials can be captured using Kali Linux tools.
- Identify warning signs that indicate a phishing attempt.

The lecture material highlights that **social engineering focuses on manipulating people rather than machines**, often exploiting trust or urgency to obtain confidential information

2. Lab Environment

Ensure the following virtual machines are running and connected to the same virtual network.

Virtual Machine	Role
Kali Linux	Attacker machine
Windows 11	Victim machine
OWASP Juice Shop (optional)	Target website

All systems must be connected using **Host-only or NAT networking**.

3. Lab Scenario

- An attacker sends a phishing link to a victim that appears to be a legitimate website login page.
- The victim unknowingly enters their credentials into the fake website.
- The attacker captures these credentials using the **Social Engineering Toolkit (SET)** on Kali Linux.
- This activity demonstrates how phishing attacks trick users into revealing confidential information, which is one of the most common social engineering techniques

5. Lab Tasks

Task 1: Identify the Attacker IP Address

1. Start the **Kali Linux virtual machine**.
2. Open a terminal window.

3. Run the following command to identify the IP address of Kali Linux:

```
root@osboxes: ~  
root@osboxes:~# ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host  
        valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000  
    link/ether 08:00:27:79:26:d7 brd ff:ff:ff:ff:ff:ff  
    inet 192.168.56.103/24 brd 192.168.56.255 scope global dynamic noprefixroute eth0  
        valid_lft 551sec preferred_lft 551sec  
    inet6 fe80::a00:27ff:fe79:26d7/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever  
root@osboxes:~#
```

4. Locate the network interface (usually eth0 or ens33) and record the IP address.

Example:

```
inet 192.168.56.103/24 brd 192.168.56.255 scope global dynamic noprefixroute eth0
```

This IP address will be used to host the phishing page.

Task 2: Launch the Social Engineering Toolkit

1. In the Kali Linux terminal, start the Social Engineering Toolkit: `sudo setoolkit`

```
root@osboxes: ~  
root@osboxes:~# sudo setoolkit  
[-] New set.config.py file generated on: 2026-03-10 13:29:37.783848  
[-] Verifying configuration update...  
[*] Update verified, config timestamp is: 2026-03-10 13:29:37.783848  
[*] SET is using the new config, no need to restart  
Copyright 2019, The Social-Engineer Toolkit (SET) by TrustedSec, LLC  
All rights reserved.  
  
Redistribution and use in source and binary forms, with or without modification, are permitted provided  
met:
```

2. When the menu appears, select yes and then select the following options:

- 1) Social Engineering Attacks

```
Select from the menu:  
  
1) Social-Engineering Attacks  
2) Penetration Testing (Fast-Track)  
3) Third Party Modules  
4) Update the Social-Engineer Toolkit  
5) Update SET configuration  
6) Help, Credits, and About  
  
99) Exit the Social-Engineer Toolkit  
  
set> 
```

3. Next choose:

2) Website Attack Vectors

```
1) Spear-Phishing Attack Vectors
2) Website Attack Vectors
3) Infectious Media Generator
4) Create a Payload and Listener
5) Mass Mailer Attack
6) Arduino-Based Attack Vector
7) Wireless Access Point Attack Vector
8) QRCode Generator Attack Vector
9) Powershell Attack Vectors
10) Third Party Modules

99) Return back to the main menu.

set> 
```

4. Then select:

3) Credential Harvester Attack Method

```
1) Java Applet Attack Method
2) Metasploit Browser Exploit Method
3) Credential Harvester Attack Method
4) Tabnabbing Attack Method
5) Web Jacking Attack Method
6) Multi-Attack Web Method
7) HTA Attack Method

99) Return to Main Menu

set:webattack> 
```

5. Finally choose:

2) Site Cloner

```
1) Web Templates
2) Site Cloner
3) Custom Import

99) Return to Webattack Menu

set:webattack> 
```

Task 3: Configure the Phishing Attack

1. When prompted, enter the **IP address of the Kali Linux machine**.

Example: 192.168.56.103

2. Next, enter the URL of the website to clone.

```
Enter the IP address for POST back in Harvester/Tabnabbing: 192.168.56.3
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:
```

Example: `http://www.thisisafakesite.com`

3. SET is unable to clone the website because **HTTP/HTTPS** access through ports **80** and **443** is blocked, closed, or unavailable. As a result, the tool cannot reach or copy the target site over the network.

```
Enter the IP address for POST back in Harvester/Tabnabbing: 192.168.56.3
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:http://www.thisisafakesite.com

[*] Cloning the website: http://www.thisisafakesite.com
[*] This could take a little bit...
[*] Error. Unable to clone this specific site. Check your internet connection.

Press <return> to continue
```

4. To verify if port 80 and 443 are actually closed we execute this command `nmap -p 80, 443 10.0.2.15` on kali Linux terminal. As you can see both ports are closed.

```
root@osboxes:~# nmap -p 80, 443 192.168.56.3
Starting Nmap 7.80 ( https://nmap.org ) at 2026-07-09 16:53 EDT
setup_target: failed to determine route to 443 (0.0.1.187)
Nmap scan report for 192.168.56.3
Host is up (0.00013s latency).

PORT      STATE SERVICE
80/tcp    closed http

Nmap done: 1 IP address (1 host up) scanned in 13.07 seconds
root@osboxes:~# nmap -p 443 192.168.56.3
Starting Nmap 7.80 ( https://nmap.org ) at 2026-07-09 16:54 EDT
Nmap scan report for 192.168.56.3
Host is up (0.000046s latency).

PORT      STATE SERVICE
443/tcp   closed https

Nmap done: 1 IP address (1 host up) scanned in 13.05 seconds
root@osboxes:~#
```

Task 4: Create a folder for the local practice page

Type: `mkdir -p ~/mocksite`

```
root@osboxes:~# mkdir -p ~/mocksite
root@osboxes:~#
```

Task 5: Create the HTML file

Copy and paste this whole block into the terminal:

```
cat > ~/mocksite/index.html <<'EOF'
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Training Login</title>
</head>
<body>
  <h1>Training Login</h1>
  <form method="get" action="#">
    <label>User <input name="user"></label><br>
    <label>Pass <input name="pass" type="password"></label><br>
    <button type="submit">Sign in</button>
  </form>
</body>
</html>
EOF
```

```
root@osboxes:~# cat > ~/mocksite/index.html <<'EOF'
> <!doctype html>
> <html>
> <head>
>   <meta charset="utf-8">
>   <title>Training Login</title>
> </head>
> <body>
>   <h1>Training Login</h1>
>   <form method="get" action="#">
>     <label>User <input name="user"></label><br>
>     <label>Pass <input name="pass" type="password"></label><br>
>     <button type="submit">Sign in</button>
>   </form>
> </body>
> </html>
> EOF
root@osboxes:~#
```

Task 6: Confirm the file exists

Run: `ls -l ~/mocksite`

You should see `index.html` listed.

```
root@osboxes:~# ls -l ~/mocksite
total 12
-rw-r--r-- 1 root root 342 Jul  9 17:08 index.html
-rw-r--r-- 1 root root 342 Mar 21 04:18 index.html.save
-rw-r--r-- 1 root root 413 Mar 21 04:20 index.html.save.1
root@osboxes:~#
```

Task 7: Move into that folder

Run: `cd ~/mocksite`

```
root@osboxes:~# cd ~/mocksite
root@osboxes:~/mocksite#
```

Task 8: Start the local web server

Run: `python3 -m http.server 8080`

If it works, you should see something like:

Serving HTTP on 0.0.0.0 port 8080 ...

Important: **leave this terminal open.** Do not close it. Do not press `Ctrl+C`.

```
root@osboxes:~/mocksite# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
█
```

Task 9: Open a second terminal window

1. Open another terminal while the first one is still running the server.
2. In the second terminal, run: `curl -I http://127.0.0.1:8080`

If it is working, you should get something like: `HTTP/1.0 200 OK`

```
root@osboxes:~# curl -I http://127.0.0.1:8080
HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.7.4
Date: Thu, 09 Jul 2026 21:13:08 GMT
Content-type: text/html
Content-Length: 342
Last-Modified: Thu, 09 Jul 2026 21:08:57 GMT

root@osboxes:~#
```

Task 10: Open it in the browser

1. Still in the second terminal, run:

```
firefox http://127.0.0.1:8080
```

That should open your local training page.

The screenshot displays two terminal windows and a browser window. The left terminal window shows the output of a curl command and the execution of the firefox command. The right terminal window shows the HTML source code of the 'Training Login' page, which includes a form with 'User' and 'Pass' input fields and a 'Sign in' button. Below the browser window, the terminal shows the output of the ls command in the ~/mocksite directory, listing index.html, index.html.save, and index.html.save.1. Finally, the terminal shows the output of the python3 -m http.server 8080 command, indicating that the HTTP server is running on port 8080.

```
root@osboxes:~# curl -I http://127.0.0.1:8080
HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.7.4
Date: Thu, 09 Jul 2026 21:13:08 GMT
Content-type: text/html
Content-Length: 342
Last-Modified: Thu, 09 Jul 2026 21:08:57 GMT

root@osboxes:~# firefox http://127.0.0.1:8080

> <html>
> <head>
>   <meta charset="utf-8">
>   <title>Training Login</title>
> </head>
> <body>
>   <h1>Training Login</h1>
>   <form method="get" action="#">
>     <label>User <input name="user"></label><br>
>     <label>Pass <input name="pass" type="password"></label><br>
>     <button type="submit">Sign in</button>
>   </form>
> </body>
> </html>
> EOF
root@osboxes:~# ls -l ~/mocksite
total 12
-rw-r--r-- 1 root root 342 Jul  9 17:08 index.html
-rw-r--r-- 1 root root 342 Mar 21 04:18 index.html.save
-rw-r--r-- 1 root root 413 Mar 21 04:20 index.html.save.1
root@osboxes:~# cd ~/mocksite
root@osboxes:~/mocksite# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
127.0.0.1 - - [09/Jul/2026 17:13:08] "HEAD / HTTP/1.1" 200 -
127.0.0.1 - - [09/Jul/2026 17:14:49] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [09/Jul/2026 17:14:49] code 404, message File not found
127.0.0.1 - - [09/Jul/2026 17:14:49] "GET /favicon.ico HTTP/1.1" 404 -
```

2. Enter test credentials such as:

Username: student1

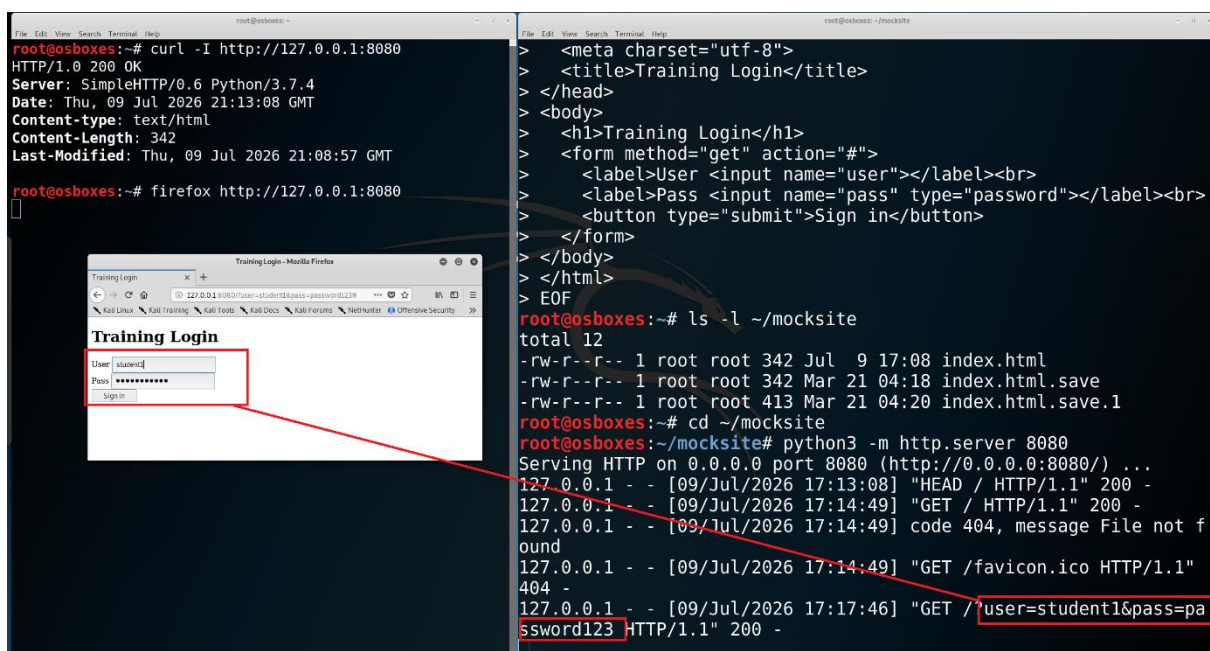
Password: password123

Submit the login form.

Task 5: Capture the Credentials

1. Return to the **Kali Linux terminal 1**.
2. Observe the captured credentials displayed.

Example output:



The screenshot shows a Kali Linux terminal window on the left and a web browser window on the right. The terminal window displays the following commands and output:

```
root@osboxes:~# curl -I http://127.0.0.1:8080
HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.7.4
Date: Thu, 09 Jul 2026 21:13:08 GMT
Content-type: text/html
Content-Length: 342
Last-Modified: Thu, 09 Jul 2026 21:08:57 GMT

root@osboxes:~# firefox http://127.0.0.1:8080
```

The web browser window shows a "Training Login" form with fields for "User" (containing "student1") and "Pass" (containing "password123"). A red box highlights the "Pass" field. The browser's developer tools show the HTML source code of the page, which includes the following form:

```
<meta charset="utf-8">
<title>Training Login</title>
<body>
  <h1>Training Login</h1>
  <form method="get" action="#">
    <label>User <input name="user"></label><br>
    <label>Pass <input name="pass" type="password"></label><br>
    <button type="submit">Sign in</button>
  </form>
</body>
</html>
EOF
```

The terminal window also shows the following commands and output:

```
root@osboxes:~# ls -l ~/mocksite
total 12
-rw-r--r-- 1 root root 342 Jul  9 17:08 index.html
-rw-r--r-- 1 root root 342 Mar 21 04:18 index.html.save
-rw-r--r-- 1 root root 413 Mar 21 04:20 index.html.save.1

root@osboxes:~# cd ~/mocksite
root@osboxes:~/mocksite# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
127.0.0.1 - - [09/Jul/2026 17:13:08] "HEAD / HTTP/1.1" 200 -
127.0.0.1 - - [09/Jul/2026 17:14:49] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [09/Jul/2026 17:14:49] code 404, message File not found
127.0.0.1 - - [09/Jul/2026 17:14:49] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [09/Jul/2026 17:17:46] "GET /?user=student1&pass=password123 HTTP/1.1" 200 -
```

Task 6: Identify Warning Signs

Students should examine the phishing page and identify suspicious indicators such as:

- Incorrect website URL
- Absence of HTTPS security
- Unexpected login request
- Slight differences in website design

According to the lecture material, phishing attacks often contain **suspicious links or unexpected requests for sensitive information**, which can help users recognize potential attacks.

6. Lab Questions

Answer the following questions after completing the lab.

1. What technique was used to obtain the victim's credentials?
2. Why did the victim trust the phishing page?
3. What signs indicated that the website was fake?
4. What type of social engineering attack was demonstrated in this lab?
5. How can users protect themselves from phishing attacks?

7. Discussion

- Phishing is one of the most common social engineering attacks used by cybercriminals.

- Attackers exploit **human emotions such as trust, fear, urgency, or curiosity** to manipulate victims into revealing confidential information.
- Even when technical systems are secure, human error can still lead to security breaches. Therefore, organizations must provide **security awareness training and enforce strong security policies**.

8. Learning Outcomes

After completing this lab, students should be able to:

- Explain how **phishing attacks work**.
- Use Kali Linux tools to simulate a **credential harvesting attack**.
- Recognize common signs of **social engineering attacks**.
- Understand the importance of **user awareness in cybersecurity**.