

Case Study 2a (Metasploitable)

Aloysius Egbo Ike

College of Arts & Sciences, Niagara University

CSO 530: Ethical Hacking

Dr. Glenn Papp

October 12, 2023

INTRODUCTION

The essence of this project is to ensure that Virtual Machines are running on our Hyper-V which is the host and they can communicate with one another.

In a typical cloud environment, multiple virtual machines (VMs) are generated on a single physical server through virtualization technology, and share the server's hardware resources (e.g., CPU, memory, and so on) in order to offer various services to the users. Various software technologies are available for supporting virtualization and among them is Hyper-V (Chen & Li, 2017). Hyper-V is Microsoft's hardware virtualization technology.

It is identified as one of the "Big Three" hypervisors that can allow multiple Virtual servers also known as virtual machines (VM) to run on a single physical server. Numerous users can use each of the VMs hosted on the physical servers.

For this assignment, the objective is to use a private network switch that will allow Kali and Metasploitable to ping each other. First, we need to extract the metasploitable folder and create the private switch in Hyper-V.

Extracting Metasploitable folder from the Class File Folder.

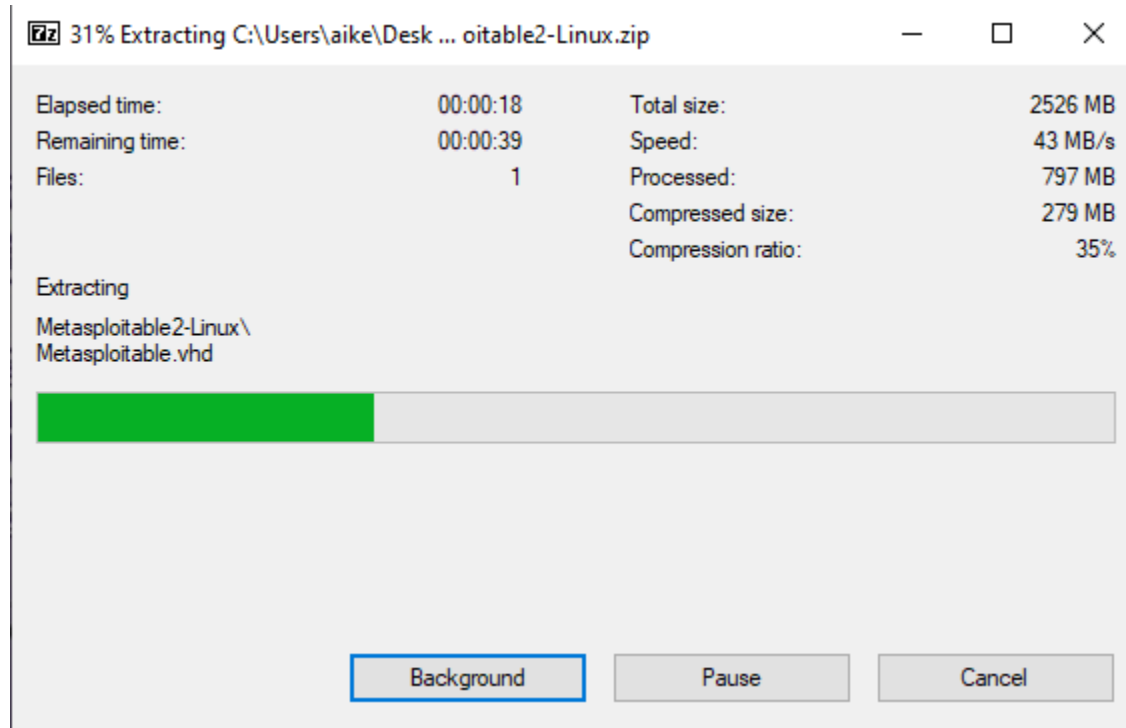


Fig 1. Screenshot showing extraction of Metasploitable folder from the class file folder.

From the extracted file, we will be using “.vhd” which is a virtual machine disk file used by Hyper-V. We will open Hyper-V, go to the right pane, and install the Metasploitable virtual machine using the new virtual machine wizard.

For this project, I'll be creating a private virtual switch as shown below.

Virtual machines (VMs) linked to a private virtual switch network are able to communicate with each other, but not with any resources located outside of the private virtual switch. The VM is isolated with a private virtual switch network.

There are three types of virtual switches available: external, internal, and private (Savill, 2014). They include;

- An external virtual switch can connect to the physical network adapter, which allows VMs to access a physical network. An external virtual switch enables communications between VMs on the same physical computer, VMs and the physical computer, as well as enables VMs to access physical network.
- An internal virtual switch enables communications between the VMs running on the same Hyper-V server, as well as between those VMs and the management OS. However, this type of switch does not provide access to a physical network.
- A private virtual switch only ensures connectivity between the VMs that run on the same Hyper-V host. This type of network provides an isolated mode of network connectivity.

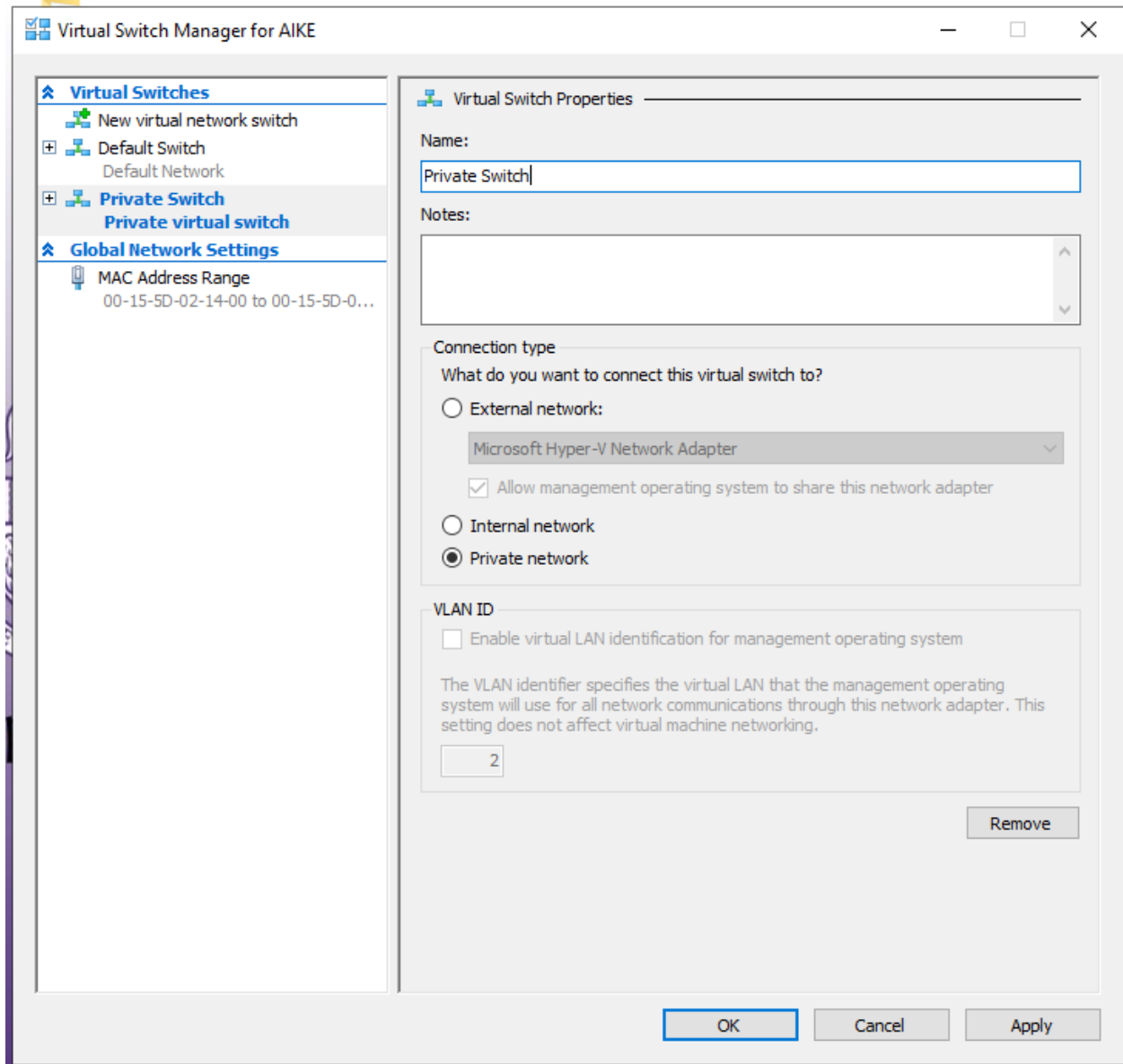
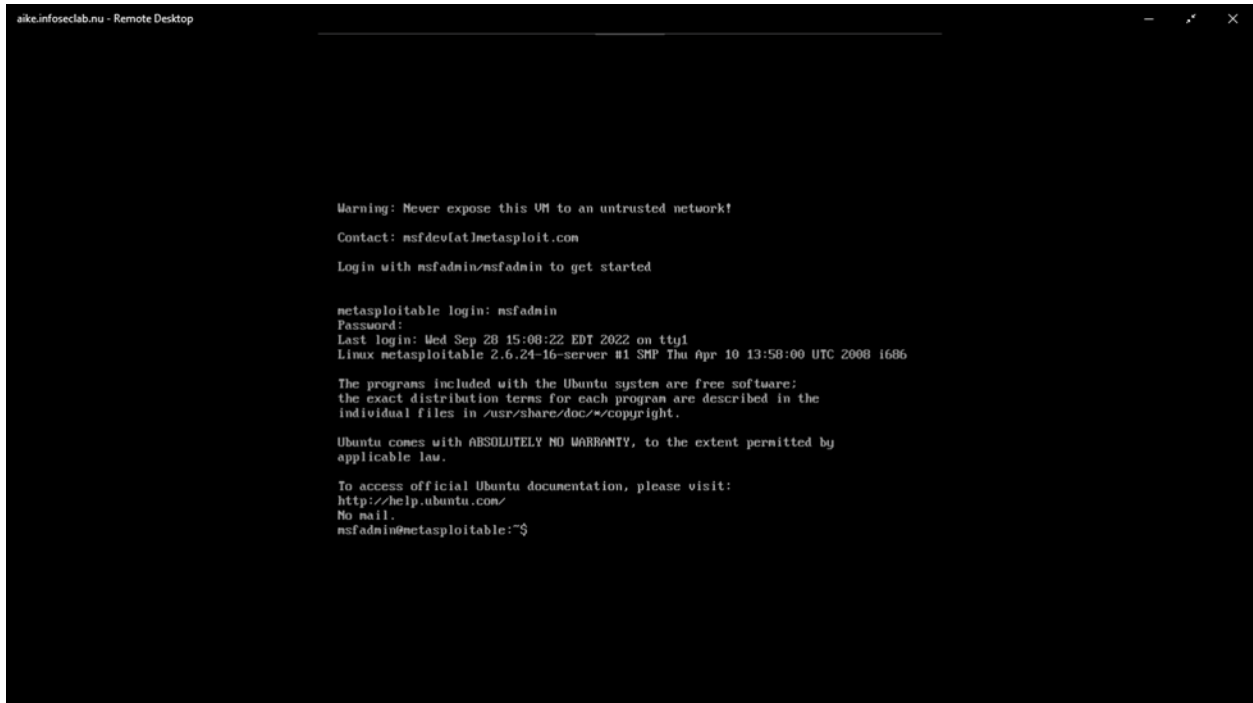


Fig. 2. Screenshot showing creating on a Private Network Switch using Hyper-V

Metasploitable is an open source virtual machine with the Ubuntu operating system for testing security tools and demonstrating common vulnerabilities. It comes with security vulnerabilities to enable users in conducting security training, security tools testing, and practice common penetration testing techniques (Alhijawi et al., 2022). We will be installing the Metasploitable next so as to be able to have a vulnerable machine we can use for this project.



```
Warning: Never expose this VM to an untrusted network!  
Contact: msfdev[at]metasploit.com  
Login with msfadmin/msfadmin to get started  
  
metasploitable login: msfadmin  
Password:  
Last login: Wed Sep 28 15:08:22 EDT 2022 on tty1  
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686  
  
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/  
No mail.  
msfadmin@metasploitable:~$
```

Fig 3. Screenshot showing completion of the installation of Metasploit

After installation, it is important to edit the network configuration of Metasploitable so as to allow it have a successful ping with Kali. This is done using nano command on /etc/network/interfaces folder.

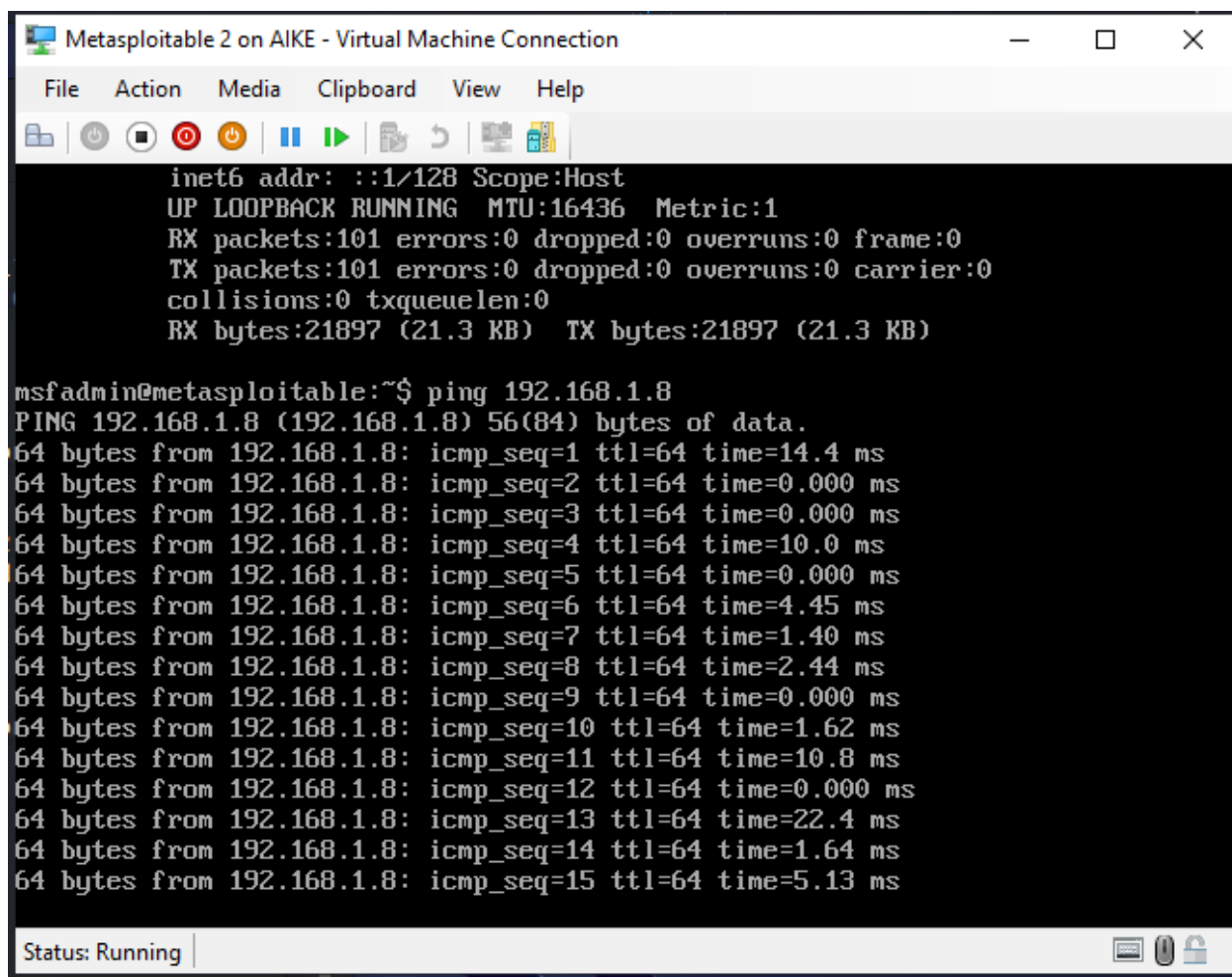
```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:15:5d:02:14:12
          inet addr:192.168.1.9  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::215:5dff:fe02:1412/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:27 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:2954 (2.8 KB)
          Interrupt:11 Base address:0xec00

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:101 errors:0 dropped:0 overruns:0 frame:0
          TX packets:101 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:21897 (21.3 KB)  TX bytes:21897 (21.3 KB)

msfadmin@metasploitable:~$
```

Status: Running

Fig. 4. Screenshot showing the edited network configuration for Metasploitable



The screenshot shows a terminal window titled "Metasploitable 2 on AIKE - Virtual Machine Connection". The terminal displays the output of the `ifconfig` command for the `lo` interface, followed by a `ping` command to `192.168.1.8`. The ping results show 15 successful packets with varying response times.

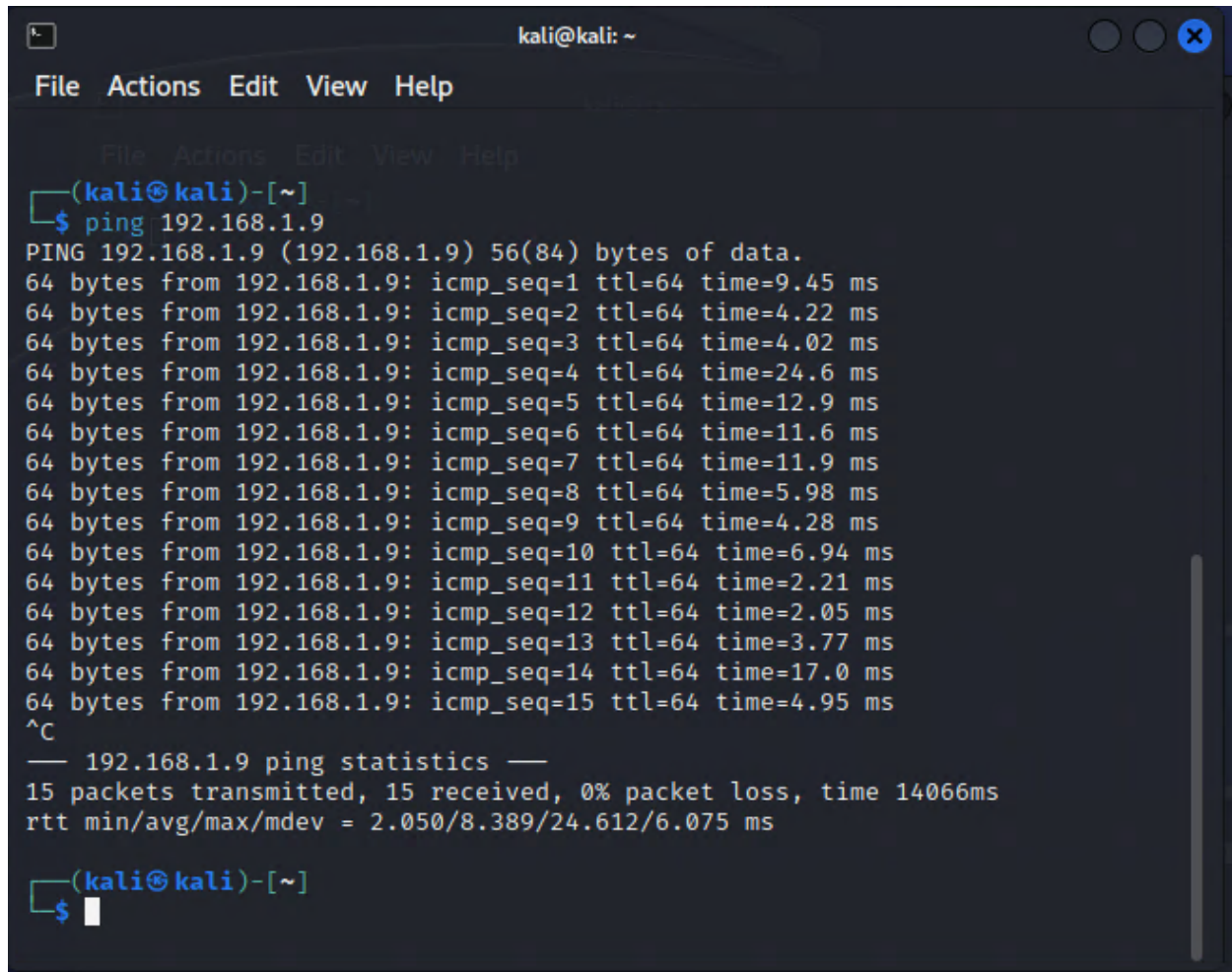
```
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:101 errors:0 dropped:0 overruns:0 frame:0
TX packets:101 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:21897 (21.3 KB) TX bytes:21897 (21.3 KB)

msfadmin@metasploitable:~$ ping 192.168.1.8
PING 192.168.1.8 (192.168.1.8) 56(84) bytes of data.
64 bytes from 192.168.1.8: icmp_seq=1 ttl=64 time=14.4 ms
64 bytes from 192.168.1.8: icmp_seq=2 ttl=64 time=0.000 ms
64 bytes from 192.168.1.8: icmp_seq=3 ttl=64 time=0.000 ms
64 bytes from 192.168.1.8: icmp_seq=4 ttl=64 time=10.0 ms
64 bytes from 192.168.1.8: icmp_seq=5 ttl=64 time=0.000 ms
64 bytes from 192.168.1.8: icmp_seq=6 ttl=64 time=4.45 ms
64 bytes from 192.168.1.8: icmp_seq=7 ttl=64 time=1.40 ms
64 bytes from 192.168.1.8: icmp_seq=8 ttl=64 time=2.44 ms
64 bytes from 192.168.1.8: icmp_seq=9 ttl=64 time=0.000 ms
64 bytes from 192.168.1.8: icmp_seq=10 ttl=64 time=1.62 ms
64 bytes from 192.168.1.8: icmp_seq=11 ttl=64 time=10.8 ms
64 bytes from 192.168.1.8: icmp_seq=12 ttl=64 time=0.000 ms
64 bytes from 192.168.1.8: icmp_seq=13 ttl=64 time=22.4 ms
64 bytes from 192.168.1.8: icmp_seq=14 ttl=64 time=1.64 ms
64 bytes from 192.168.1.8: icmp_seq=15 ttl=64 time=5.13 ms
```

Status: Running

Fig 5. Screenshot showing ping on Kali from Metasploitable

The screenshot above shows that Metasploitable could successfully ping to Kali. If it had shown that “Network is unreachable”, that would mean that the ping to Kali is unsuccessful and some configurations need to be looked into.

A screenshot of a Kali Linux terminal window. The window title is 'kali@kali: ~'. The menu bar shows 'File', 'Actions', 'Edit', 'View', and 'Help'. The terminal shows a user prompt '(kali@kali)-[~]' followed by the command '\$ ping 192.168.1.9'. The output shows 15 successful ping requests to 192.168.1.9, each with 64 bytes of data and varying response times. The statistics at the bottom show 15 packets transmitted, 15 received, 0% packet loss, and a total time of 14066ms. The user prompt is followed by a dollar sign and a cursor.

```
kali@kali: ~
File Actions Edit View Help

(kali@kali)-[~]
$ ping 192.168.1.9
PING 192.168.1.9 (192.168.1.9) 56(84) bytes of data.
64 bytes from 192.168.1.9: icmp_seq=1 ttl=64 time=9.45 ms
64 bytes from 192.168.1.9: icmp_seq=2 ttl=64 time=4.22 ms
64 bytes from 192.168.1.9: icmp_seq=3 ttl=64 time=4.02 ms
64 bytes from 192.168.1.9: icmp_seq=4 ttl=64 time=24.6 ms
64 bytes from 192.168.1.9: icmp_seq=5 ttl=64 time=12.9 ms
64 bytes from 192.168.1.9: icmp_seq=6 ttl=64 time=11.6 ms
64 bytes from 192.168.1.9: icmp_seq=7 ttl=64 time=11.9 ms
64 bytes from 192.168.1.9: icmp_seq=8 ttl=64 time=5.98 ms
64 bytes from 192.168.1.9: icmp_seq=9 ttl=64 time=4.28 ms
64 bytes from 192.168.1.9: icmp_seq=10 ttl=64 time=6.94 ms
64 bytes from 192.168.1.9: icmp_seq=11 ttl=64 time=2.21 ms
64 bytes from 192.168.1.9: icmp_seq=12 ttl=64 time=2.05 ms
64 bytes from 192.168.1.9: icmp_seq=13 ttl=64 time=3.77 ms
64 bytes from 192.168.1.9: icmp_seq=14 ttl=64 time=17.0 ms
64 bytes from 192.168.1.9: icmp_seq=15 ttl=64 time=4.95 ms
^C
— 192.168.1.9 ping statistics —
15 packets transmitted, 15 received, 0% packet loss, time 14066ms
rtt min/avg/max/mdev = 2.050/8.389/24.612/6.075 ms

(kali@kali)-[~]
$
```

Fig 6. Screenshot of Ping from Kali

Conclusion

This project aimed to establish effective communication between virtual machines (VMs) using virtualisation technology such as Hyper-V host through the implementation of a private virtual switch. Virtualization technology, which is increasingly becoming popular in many fields at present, has many advantages including reducing costs, unified management, mobile applications, cross platform, etc (Wang et al., 2014). Leveraging on one of the major virtualisation technologies; Microsoft's Hyper-V technology and following a systematic process, the Metasploitable virtual machine was successfully installed and configured, allowing it to communicate with Kali. The private virtual switch ensured an isolated network environment for the VMs, demonstrating the capability of VMs to interact exclusively with each other while remaining disconnected from external resources.

The successful ping between Metasploitable and Kali, as depicted in the provided screenshots, validates the project's objectives. This project not only highlights the practical application of virtualization technology but also emphasizes the importance of network configurations in

ensuring secure and controlled communication within virtualized environments. The outcomes of this project contribute to the broader understanding of virtualization technologies, particularly Hyper-V, and their role in facilitating secure and efficient communication among virtual machines.

References

Alhijawi, B., Almajali, S., Elgala, H., Bany Salameh, H., & Ayyash, M. (2022). A survey on DOS/ddos mitigation techniques in SDNS: Classification, comparison, solutions, testing tools and datasets. *Computers and Electrical Engineering*, 99, 107706. <https://doi.org/10.1016/j.compeleceng.2022.107706>

Chen, Y.-R., & Li, J.-S. (2017). Staggered approach for alleviating TCP Incast in simultaneous Multi-VM migration. *Computer Communications*, 106, 24–32. <https://doi.org/10.1016/j.comcom.2017.02.015>

McCoy, F. (2012). Resources for digital defenders: WHAT YOU NEED TO READ, VISIT AND KNOW TO GET AHEAD. *US Black Engineer and Information Technology*, 36(4), 75–78. <http://www.jstor.org/stable/43772623>

Savill, J. (2014). *Mastering hyper-v 2012 r2 with system center and windows azure*. John Wiley & Sons, Incorporated.

Wang, F., Sun, X., Li, S., Wang, Y., Xiao, B., & Chang, S. (2014). The implementation of virtualization technology in East Data System. *Fusion Engineering and Design*, 89(5), 766–769. <https://doi.org/10.1016/j.fusengdes.2014.04.003>