Introduction

Objective:

The execution of the project focuses on a virtualized network employing Software-Defined Networking for centralized control, Network Function Virtualization for effectiveness of the service, automation providing efficient network management, and embedded redundancy for enhanced dependability and reliability. Resulting a resilient network architecture, a scalable and flexible network design and architecture.

Scope:

- **SDN:** The recommended SDN architecture establishes advanced controllers for centralized handling of network traffic via an SDN controller, such as OpenDaylight or ONOS.
- **NFV:** It enables the filtering and hosting of applications such as firewalls, load balancing, VPN to allow docker and virtual machines.
- **Automation:** Modifications and control of setups using scripts and Traffic flows, Python, and Ansible consisting of scripts.
- **Redundancy:** Safeguards measures and provision of standby connectivity to test the network at times of failure situations.

Network Design and Architecture

Technologies Used:

- **Network Simulators:** It can be explored using the GNS3 or EVE-NG as well as for the designing and simulating network environments in the project.
- **SDN Controller:** In the Network Operating System that have been suggested will be the OpenDaylight or ONOS in charge of the central management of the network provided for this project.
- **NFV Tools:** This includes installing Docker for the hosting of VNF like the firewalls, VPNs and other load balancers in the ubuntu.
- **Automation Tools:** In automation we have used the Python and Ansible in particular for scripting of network configurations and for managing tasks that are automated that are needed in this project.
- **Visualization Tools:** Lucidchart, Draw.io or Microsoft Visio for the creation of network diagrams as suggested.

Network Diagram

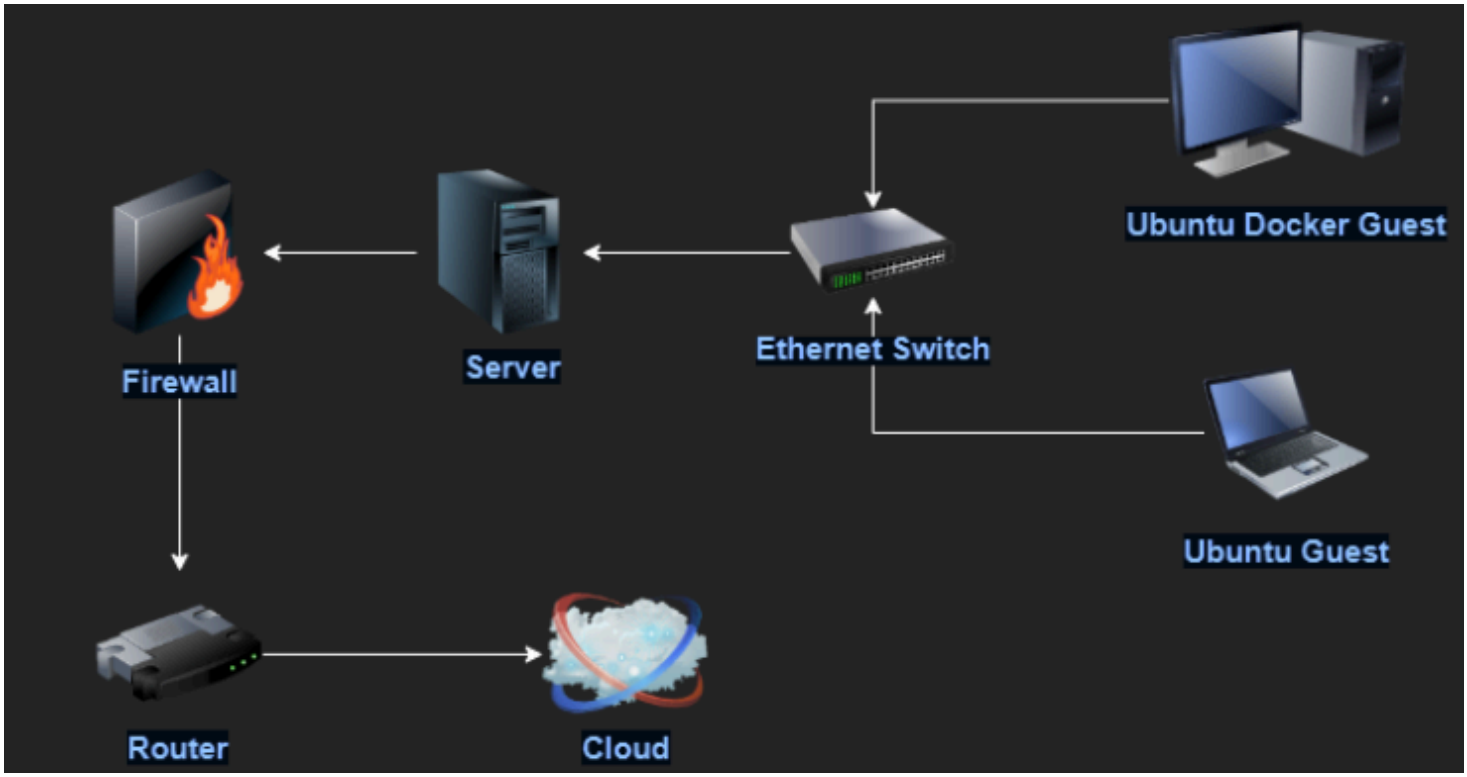

Figure 1. Simple Network Architecture Diagram

Implementation and Configuration

Setup Environment:

```
vboxuser@week7:~$ sudo apt-get install gns3-server gns3-gui
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  cpu-checker cpulimit dmeventd dynamips i965-va-driver intel-media-va-driver ipxe-qemu ipxe-qemu-256k-compat-efi-roms
  libaacs0 libaio1t64 libavahi-gobject0 libavahi-ui-gtk3-0 libavcodec60 libavformat60 libavutil58 libb2-1 libbcg729-0
  libbdplus0 libbluray2 libboost-iostreams1.83.0 libboost-thread1.83.0 libcacard0 libchromaprint1 libcjson1
  libcodec2-1.2 libdav1d7 libdaxctl1 libdevmapper-event1.02.1 libdouble-conversion3 libfdt1 libfreerdp2-2t64 libgme0
  libgsm1 libgtk-vnc-2.0-0 libgvnc-1.0-0 libhwy1t64 libigdgmm12 libiscsi7 libjack-jackd2-0 libjxl0.7 liblua5.2-0
  liblvm2cmd2.03 libmbedcrypto7t64 libmd4c0 libminizip1t64 libndctl6 libnghttp3-3 libnorm1t64 libnss-mymachines
  libopencore-amrnb0 libopenmpt0t64 libpcre2-16-0 libpgm-5.3-0t64 libphodav-3.0-0 libphodav-3.0-common libpmem1
  libpmemobj1 libqt5core5t64 libqt5dbus5t64 libqt5designer5 libqt5gui5t64 libqt5help5 libqt5network5t64
  libqt5printsupport5t64 libqt5qml5 libqt5qmlmodels5 libqt5quick5 libqt5sql5-sqlite libqt5sql5t64 libqt5svg5
  libqt5test5t64 libqt5waylandclient5 libqt5waylandcompositor5 libqt5websockets5 libqt5widgets5t64 libqt5xml5t64
  libqt6core5compat6 libqt6core6t64 libqt6dbus6t64 libqt6gui6t64 libqt6multimedia6 libqt6network6t64 libqt6opengl6t64
  libqt6printsupport6t64 libqt6qml6 libqt6qmlmodels6 libqt6quick6 libqt6svg6 libqt6waylandclient6
  libqt6waylandcompositor6 libqt6waylandeglclienthwintegration6 libqt6waylandeglcompositorhwintegration6
  libqt6widgets6t64 libqt6wlshellintegration6 librabbitmq4 librados2 librav1e0 librbd1 librdmacm1t64 librist4
  libsdl2-2.0-0 libshine3 libslirp0 libsmi2t64 libsnappy1v5 libsodium23 libsoxr0 libspandsp2t64
  libspice-client-glib-2.0-8 libspice-client-gtk-3.0-5 libspice-server1 libsrt1.5-gnutls libssh-gcrypt-4
  libsvtav1enc1d1 libswresample4 libswscale7 libtk8.6 libtpms0 libts0t64 libudfread0 liburing2 libusbredirhost1t64
  libusbredirparser1t64 libva-drm2 libva-x11-2 libva2 libvdpau1 libvirglrenderer1 libvirt-clients libvirt-daemon
  libvirt-daemon-config-network libvirt-daemon-config-nwfilter libvirt-daemon-driver-qemu libvirt-daemon-system
  libvirt-daemon-system-systemd libvirt-l10n libvirt0 libvncclient1 libvncserver1 libvpl2 libwinpr2-2t64
  libwireshark-data libwireshark17t64 libwiretap14t64 libwsutil15t64 libx264-164 libx265-199 libxcb-xinerama0
  libxcb-xinput0 libxml2-utils libxvidcore4 libyajl2 libzmq5 libzvbi-common libzvbi0t64 lvm2 mdevctl mesa-va-drivers
  mesa-vdpau-drivers msr-tools mtools ocl-icd-libopencl1 ovmf python3-pyqt5 python3-pyqt5.qtsvg
  python3-pyqt5.qtwebsockets python3-pyqt5.sip qemu-block-extra qemu-system-common qemu-system-data qemu-system-gui
  qemu-system-modules-opengl qemu-system-modules-spice qemu-system-x86 qemu-utils qt5-gtk-platformtheme
  qt6-gtk-platformtheme qt6-qpa-plugins qt6-translations-l10n qt6-wayland qttranslations5-l10n qtwayland5 seabios
```

Figure 1. Installation of GNS3 or EVE-NG

```
vboxuser@week7:~$ sudo apt-get install default-jre-headless
[sudo] password for vboxuser:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java java-common openjdk-21-jre-headless
Suggested packages:
  default-jre fonts-dejavu-extra fonts-ipafont-gothic fonts-ipafont-mincho
  fonts-wqy-microhei | fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  ca-certificates-java default-jre-headless java-common
  openjdk-21-jre-headless
0 upgraded, 4 newly installed, 0 to remove and 90 not upgraded.
Need to get 46.4 MB of archives.
After this operation, 202 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu noble/main amd64 ca-certificates-java
all 20240118 [11.6 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu noble/main amd64 java-common all 0.75+
exp1 [6,798 B]
Get:3 http://ph.archive.ubuntu.com/ubuntu noble-updates/main amd64 openjdk-21-jr
e-headless amd64 21.0.5+11-1ubuntu1~24.04 [46.4 MB]
55% [3 openjdk-21-jre-headless 26.3 MB/46.4 MB 57%]              1,349 kB/s 14s
```

Figure 1. Installation of OpenDaylight

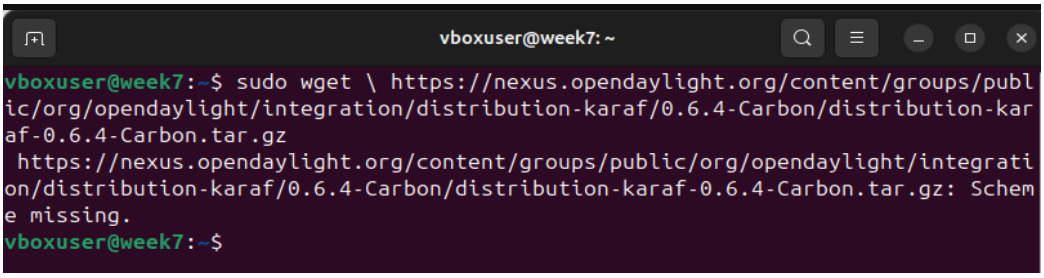

Figure 2. Installation of OpenDaylight

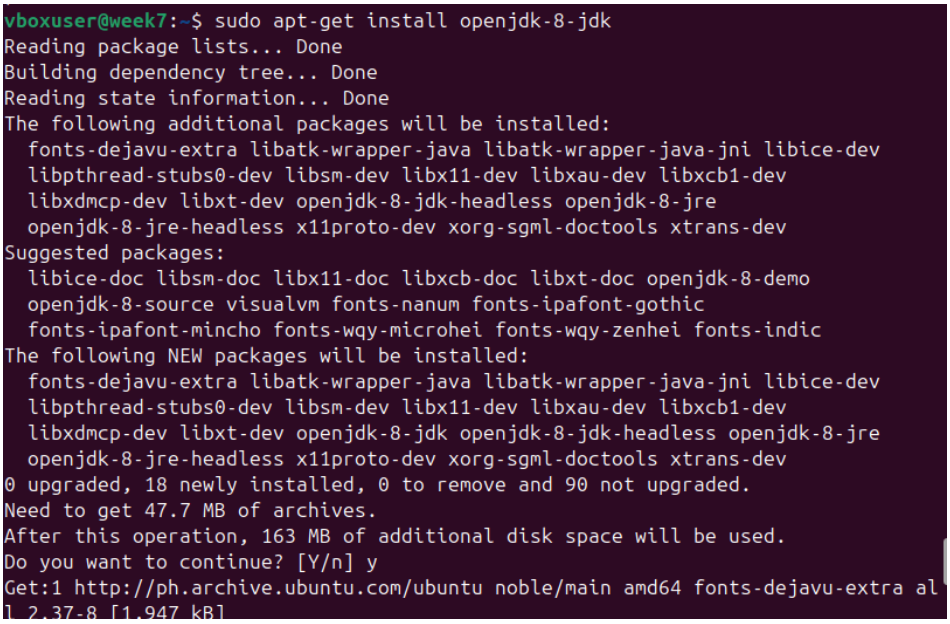

Figure 1. Installation of Java Development Kit (JDK)

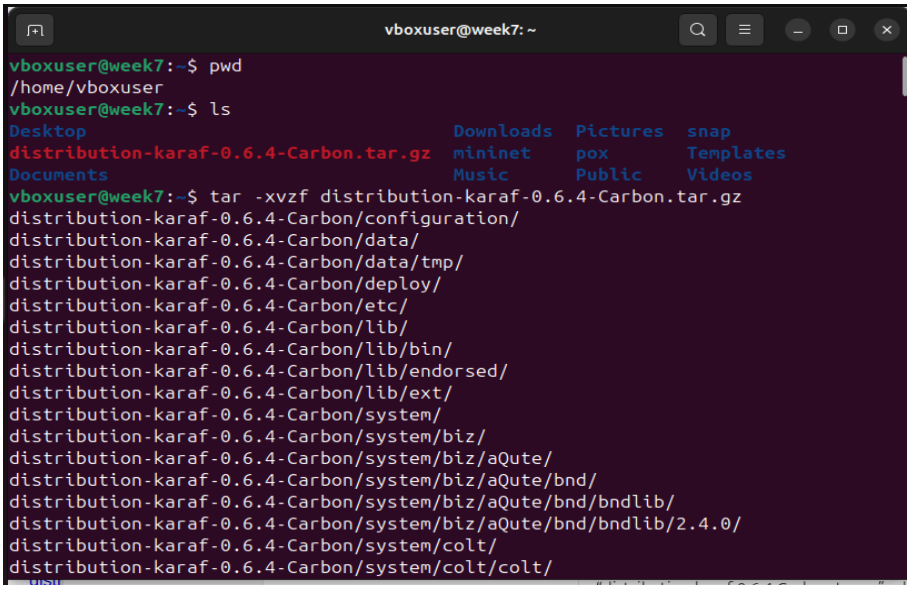

Figure 2. Installation of Java Development Kit (JDK)

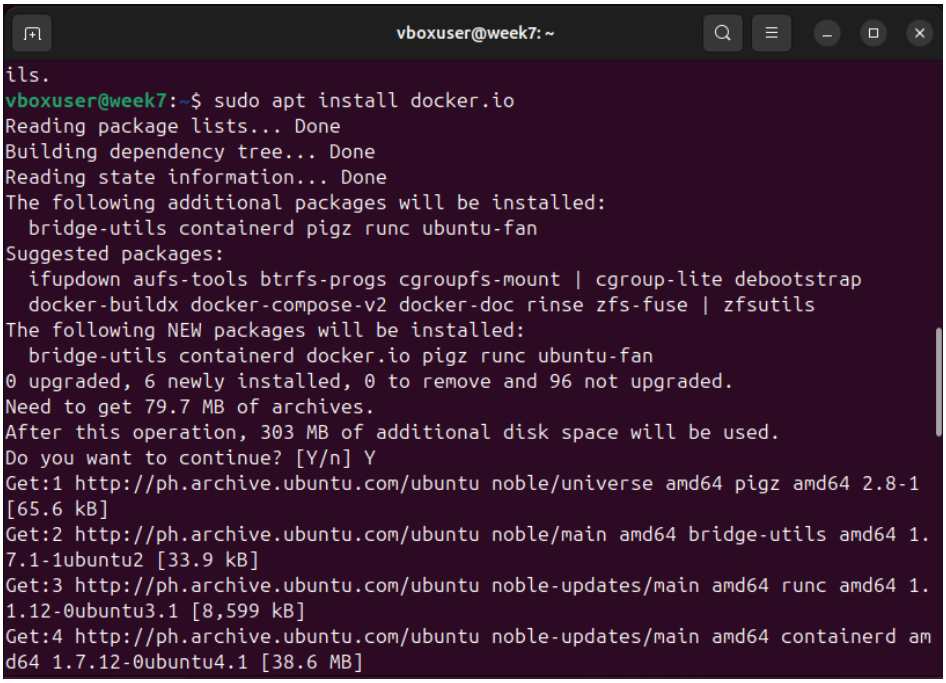

Figure 1. Installation of Docker for WSL2

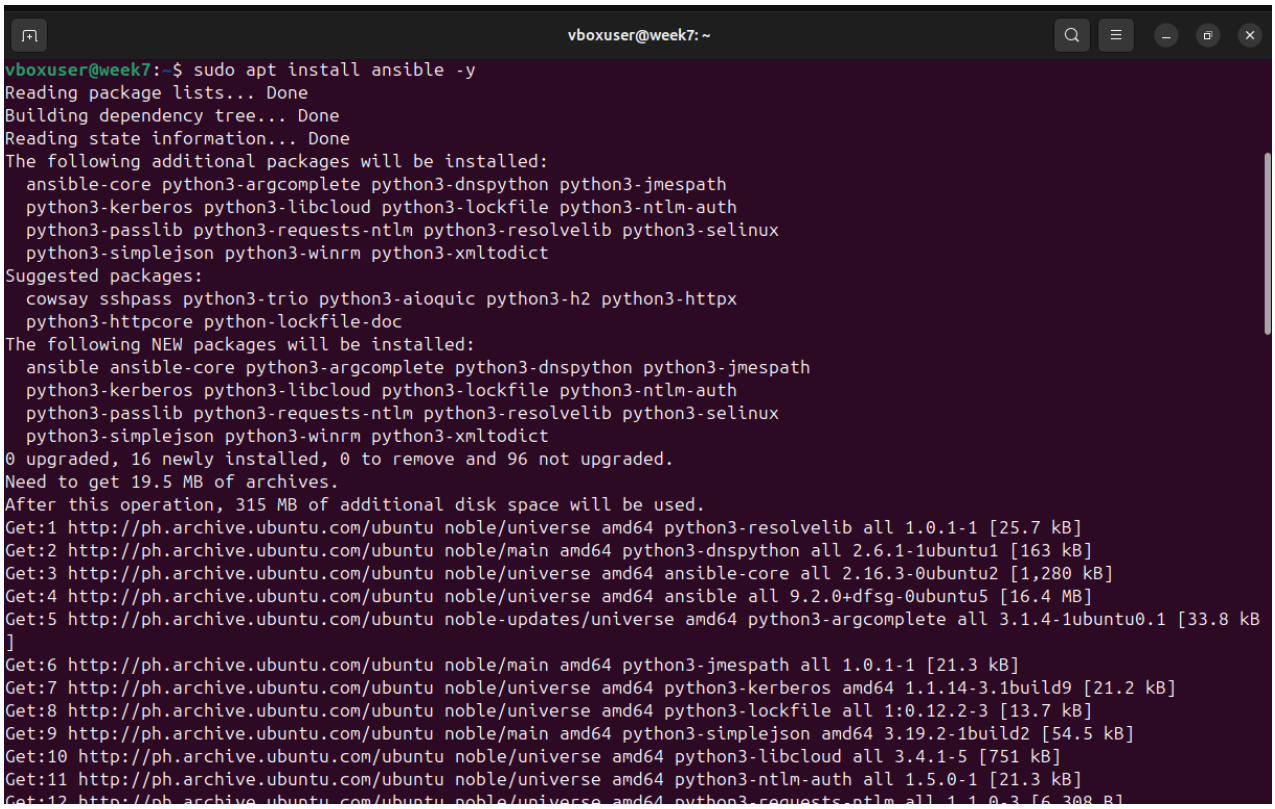

Figure 1. Installation of Ansible using the Ubuntu WSL2

SDN Configuration:

Figure 1. Configuration of OpenDaylight in the Ubuntu

Apache Karaf, a small and OSGi-based container is used to start the OpenDaylight controller. The user can view the bundle statistics and most importantly opens a Karaf shell by typing *(./karaf)* in the terminal and can see that the system is live and ready for the implementation of SDN (Software Defined Networks).

NFV Configuration:

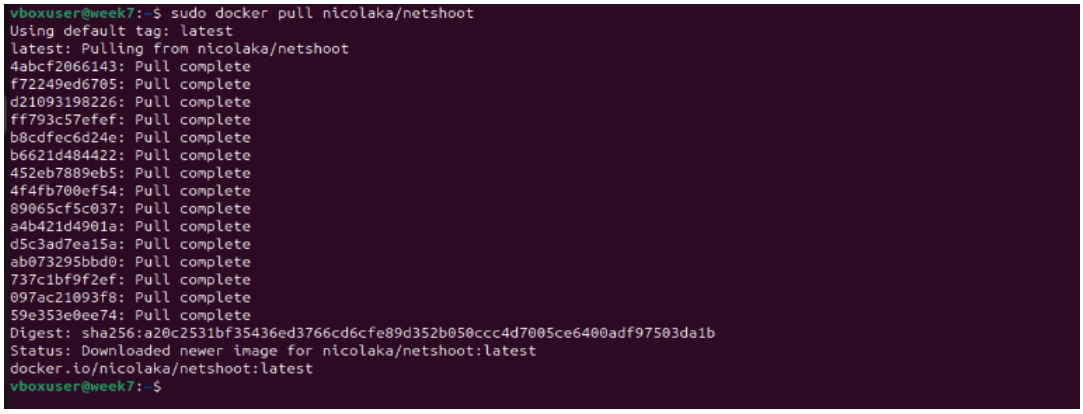

Figure 1. Configuration of Virtual Firewall in the Ubuntu

This picture demonstrates how to utilize Docker to pull the most recent rendition of the image and all the layers contained which makes containerized system availability and network troubleshooting: docker pull nicolaka/netshoot. In the configuration of VPN, the picture shows how to employ the kylemanna/openvpn Docker image to bring up an OpenVPN server.

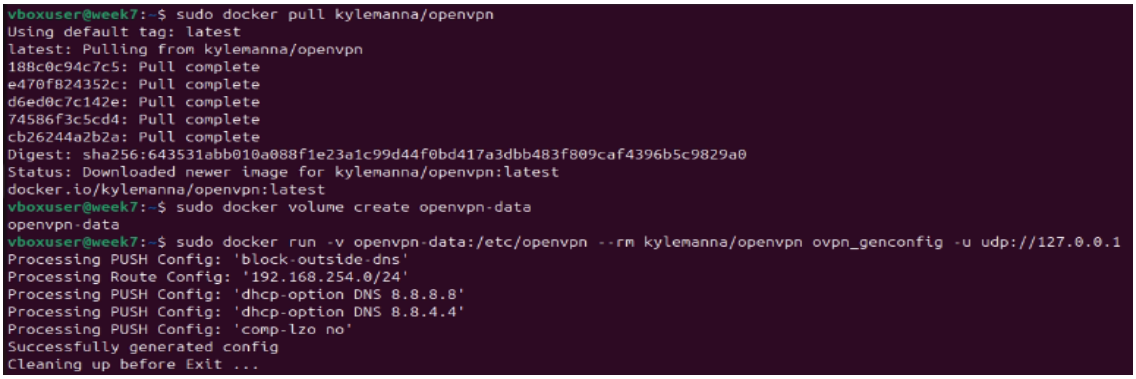

Figure 2. Configuration of VPN in the Ubuntu

The latest image of OpenVPN is then fetched from Docker Hub using the following command; *sudo docker pull kylemanna/openvpn.* Following that, to save the condition information of the server, the command used is sudo docker volume create openvpn-data. Finally, using the UDP protocol and 127.0.0.1 as the server IP, the command *sudo docker run -v openvpn-data:docker run -v /etc/openvpn --rm kylemanna/openvpn ovpn_genconfig -u udp://127.0.0.1* opens a Docker instance which is used to generate the OpenVPN configurations needed for the routing. This working involves setting DNS servers, routes, disabling compression for security purposes before launching the removal of the container.

Automation Scripts:

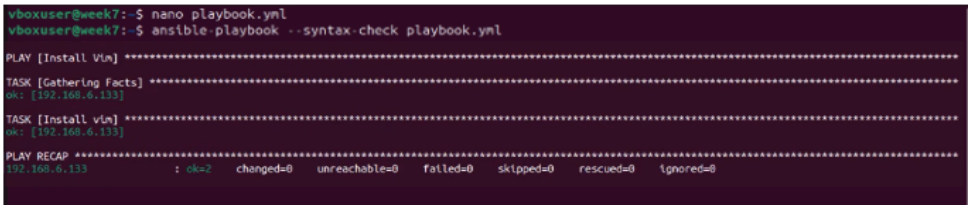

Figure 1. Installation of Ansible using the Ubuntu WSL2

This output confirms there is no syntactical error with the playbook *(playbook.yml)* which means the file is set for Ansible execution *(ansible-playbook -- syntax -- check playbook.yml)*. It checks the syntax of an Ansible playbook using the name playbook.yml and does execute it *(-automation)*.

Redundancy Techniques:

These redundancy measures were built into the SDN architecture, taking advantage of the dynamic traffic flow to reroute traffic and around failed components. Advanced tools like; SDN controllers, OpenDaylight, and ONOS, also backed up redundancy through the efficient handling of backup pathways in a centralized manner. This design gives a chance to have a strong network design that can still not be deemed highly available even in a failure situation.

Testing and Validation

Testing Plan:

Ping Tests: These were carried out using the command interfaces including CMD and Ubuntu terminal to initialize hardware connection checks for confirmatory network connectivity of the nodes in the network.

Traffic Flow Validation: While not fully stated this presumably entails the application of various means to check correct routing and delivery of the packets through the network's canopy.

Log Collection: The logs were obtained with Windows PowerShell for diagnostic purposes. This involves tracking for errors, monitoring the networks and satisfying ourselves that certain configurations were done properly.

Docker Troubleshooting: Then for the Virtualized Network Functions (VNFs), the troubleshooting was done by managing Docker containers. For instance, rebooting the virtual firewall with specific Docker commands and checking all the containers' functionality.

Network Simulator Testing: Tools like GNS3 or EVE-NG can only have been used for emulating the network virtualization and the configurations tested before their implementation here.

Automation Verification: All the scripts that were written in Python and Ansible were run to check for any syntax errors, and overall test to see whether automation was achieved as planned.

Results:

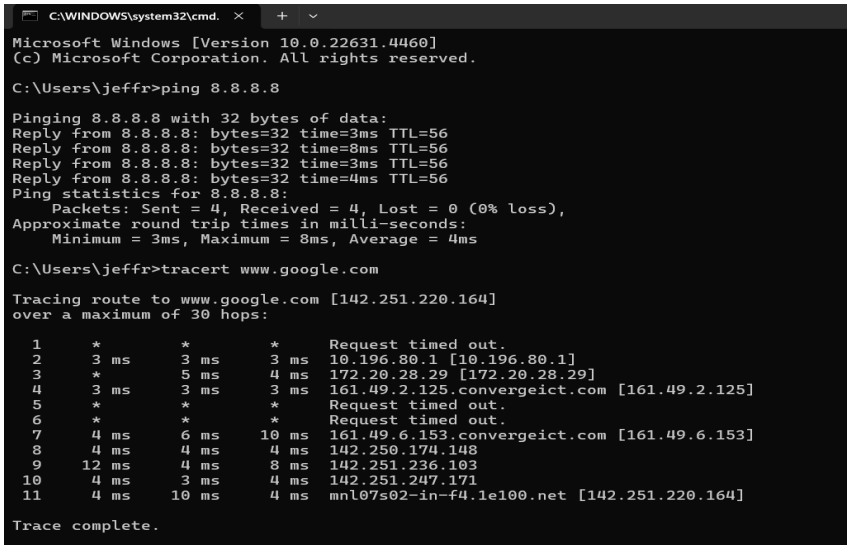

Figure 1. Ping and Connectivity Tests using the CMD

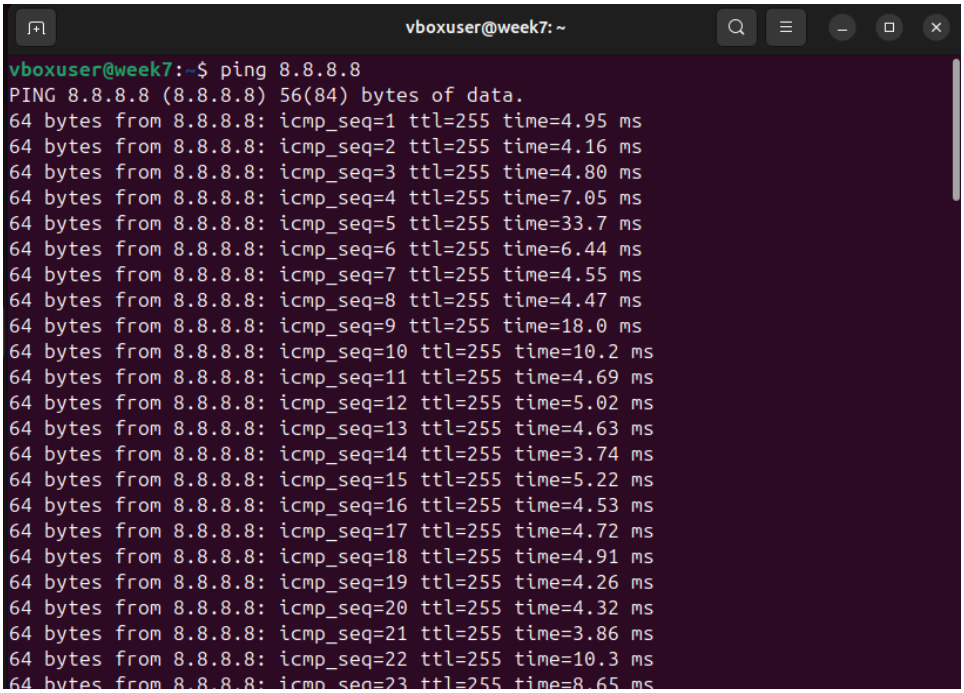

Figure 1. Ping and Connectivity Tests using the Ubuntu

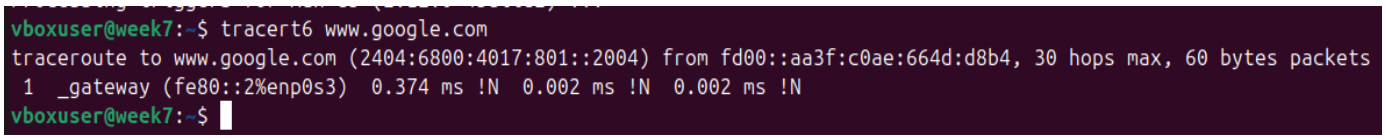

Figure 1.1. Ping and Connectivity Tests using the Ubuntu

```
PS C:\Users\jeffr> Get-Eventlog -LogName System -Newest 20

   Index Time          EntryType   Source                 Insta
                                                          nceID
   ----- ----          ---------   ------                 -----
   17201 Nov 29 19:20  Information Service Control M... ...64
   17200 Nov 29 19:16  Information Service Control M... ...64
   17199 Nov 29 19:09  Warning     DCOM                   10016
   17198 Nov 29 19:04  Information Service Control M... ...64
   17197 Nov 29 18:59  Information Service Control M... ...64
   17196 Nov 29 18:59  Information Microsoft-Windows...    19
   17195 Nov 29 18:59  Information Microsoft-Windows...    43
   17194 Nov 29 18:59  Information Microsoft-Windows...    44
   17193 Nov 29 18:55  Information Microsoft-Windows...    16
   17192 Nov 29 18:55  Information Microsoft-Windows...    16
   17191 Nov 29 18:55  Information Microsoft-Windows...    16
   17190 Nov 29 18:55  Information Microsoft-Windows...    16
   17189 Nov 29 18:55  Information Microsoft-Windows...    16
   17188 Nov 29 18:53  Warning     Microsoft-Windows...  1014
   17187 Nov 29 18:53  Information Microsoft-Windows...   102
   17186 Nov 29 18:53  Information Microsoft-Windows...   232
   17185 Nov 29 18:53  Information Microsoft-Windows...   233
   17184 Nov 29 18:52  Information Service Control M... ...69
   17183 Nov 29 18:52  Information Microsoft-Windows...   233
   17182 Nov 29 18:52  Information Microsoft-Windows...   234
```

Figure 1. Log Collection using the Windows Powershell

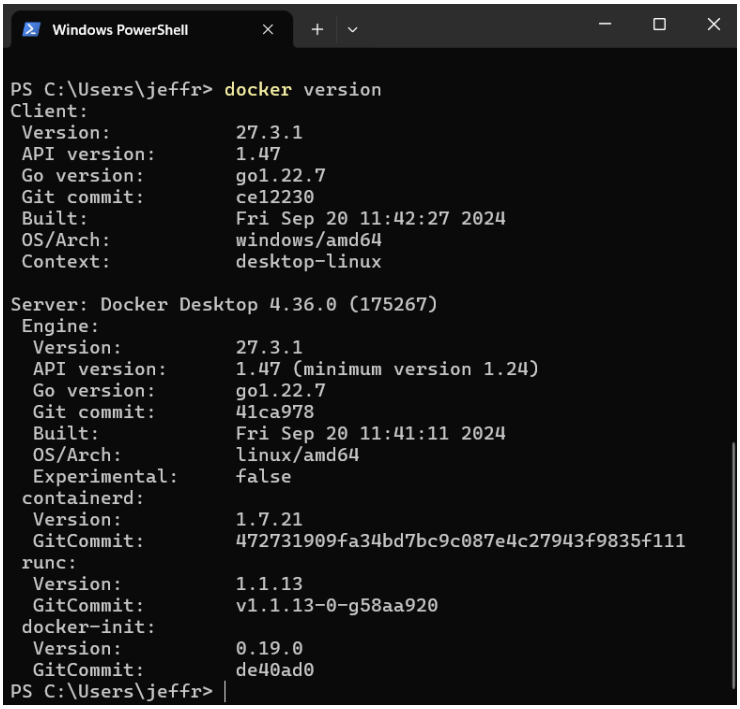

Figure 1. Docker Troubleshooting using the Windows Powershell

Appendix

OpenDaylight

```
vboxuser@week7:~$ sudo apt-get install maven
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java
  libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java
  libcommons-parent-java liberror-prone-java
  libgeronimo-annotation-1.3-spec-java libgeronimo-interceptor-3.0-spec-java
  libguava-java libguice-java libjansi-java libjsr305-java
  libmaven-parent-java libmaven-resolver-java libmaven-shared-utils-java
  libmaven3-core-java libplexus-cipher-java libplexus-classworlds-java
  libplexus-component-annotations-java libplexus-interpolation-java
  libplexus-sec-dispatcher-java libplexus-utils2-java libsisu-inject-java
  libsisu-plexus-java libslf4j-java libwagon-file-java
  libwagon-http-shaded-java libwagon-provider-api-java
Suggested packages:
  libatinject-jsr330-api-java-doc libel-api-java libcommons-io-java-doc
  libasm-java libcglib-java libjsr305-java-doc libmaven-shared-utils-java-doc
  liblogback-java libplexus-utils2-java-doc junit4 testng
  libcommons-logging-java liblog4j1.2-java
The following NEW packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java
  libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java
```

Java Development Kit (JDK)

```
vboxuser@week7:~$ sudo apt-get install openjdk-8-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  fonts-dejavu-extra libatk-wrapper-java libatk-wrapper-java-jni libice-dev
  libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-8-demo
  openjdk-8-source visualvm fonts-nanum fonts-ipafont-gothic
  fonts-ipafont-mincho fonts-wqy-microhei fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  fonts-dejavu-extra libatk-wrapper-java libatk-wrapper-java-jni libice-dev
  libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-8-jdk openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 18 newly installed, 0 to remove and 90 not upgraded.
Need to get 47.7 MB of archives.
After this operation, 163 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu noble/main amd64 fonts-dejavu-extra al
l 2.37-8 [1,947 kB]
```

Docker for Windows

```
vboxuser@week7:~$ sudo systemctl enable docker
vboxuser@week7:~$ sudo systemctl start docker
vboxuser@week7:~$ sudo apt install -y docker-compose
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  python3-compose python3-docker python3-dockerpty python3-docopt python3-dotenv python3-texttable
  python3-websocket
The following NEW packages will be installed:
  docker-compose python3-compose python3-docker python3-dockerpty python3-docopt python3-dotenv
  python3-texttable python3-websocket
0 upgraded, 8 newly installed, 0 to remove and 88 not upgraded.
Need to get 297 kB of archives.
After this operation, 1,589 kB of additional disk space will be used.
Get:1 http://ph.archive.ubuntu.com/ubuntu noble/universe amd64 python3-websocket all 1.7.0-1 [38.1 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu noble-updates/universe amd64 python3-docker all 5.0.3-1ubuntu1.1 [89
.1 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu noble/universe amd64 python3-dockerpty all 0.4.1-5 [11.4 kB]
Get:4 http://ph.archive.ubuntu.com/ubuntu noble/universe amd64 python3-docopt all 0.6.2-6 [26.1 kB]
Get:5 http://ph.archive.ubuntu.com/ubuntu noble/universe amd64 python3-dotenv all 1.0.1-1 [22.3 kB]
Get:6 http://ph.archive.ubuntu.com/ubuntu noble/universe amd64 python3-texttable all 1.6.7-1 [11.0 kB]
Get:7 http://ph.archive.ubuntu.com/ubuntu noble/universe amd64 python3-compose all 1.29.2-6ubuntu1 [84.6 kB]
Get:8 http://ph.archive.ubuntu.com/ubuntu noble/universe amd64 docker-compose all 1.29.2-6ubuntu1 [14.0 kB]
Fetched 297 kB in 1s (242 kB/s)
Selecting previously unselected package python3-websocket.
(Reading database ... 179806 files and directories currently installed.)
Preparing to unpack .../0-python3-websocket_1.7.0-1_all.deb ...
```

Ansible

```
vboxuser@week7:~$ sudo apt-get install software-properties-common
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
software-properties-common is already the newest version (0.99.49.1).
software-properties-common set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
vboxuser@week7:~$ sudo apt-add-repository --yes --update ppa:ansible/ansible
Repository: 'Types: deb
URIs: https://ppa.launchpadcontent.net/ansible/ansible/ubuntu/
Suites: noble-date
Components: main
'
Description:
Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy. Avoid wr
iting scripts or custom code to deploy and update your applications— automate in a language that approaches plain Englis
h, using SSH, with no agents to install on remote systems.

http://ansible.com/

If you face any issues while installing Ansible PPA, file an issue here:
https://github.com/ansible-community/ppa/issues
More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible
Adding repository.
Get:1 file:/cdrom noble InRelease
Ign:1 file:/cdrom noble InRelease
Get:2 file:/cdrom noble Release
Err:2 file:/cdrom noble Release
  File not found - /cdrom/dists/noble/Release (2: No such file or directory)
Hit:3 http://ph.archive.ubuntu.com/ubuntu noble InRelease
Hit:4 http://ph.archive.ubuntu.com/ubuntu noble-updates InRelease
```

Pinging Test

```
vboxuser@week7:~$ sudo ufw allow from 127.0.0.1
Rule added
vboxuser@week7:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Anywhere                   ALLOW       127.0.0.1

vboxuser@week7:~$ sudo ufw allow 80
Rule added
Rule added (v6)
vboxuser@week7:~$ sudo ufw allow 53
Rule added
Rule added (v6)
vboxuser@week7:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Anywhere                   ALLOW       127.0.0.1
80                         ALLOW       Anywhere
53                         ALLOW       Anywhere
80 (v6)                    ALLOW       Anywhere (v6)
53 (v6)                    ALLOW       Anywhere (v6)

vboxuser@week7:~$
```

Enabling Firewall

```
vboxuser@week7:~$ sudo ufw enable
[sudo] password for vboxuser:
Firewall is active and enabled on system startup
```