Linux Networking

1. На Server_1 налаштувати статичні адреси на всіх інтерфейсах.

2. На Server_1 налаштувати DHCP сервіс, який буде конфігурувати адреси Int1 Client_1 та Client_2.

3. За допомогою команд ping та traceroute перевірити зв'язок між віртуальними машинами. Результат пояснити.

4. На віртуальному інтерфейсу lo Client_1 призначити дві IP адреси за таким правилом: 172.17.D+10.1/24 та 172.17.D+20.1/24. Налаштувати маршрутизацію таким чином, щоб трафік з Client_2 до 172.17.D+10.1 проходив через Server_1, а до 172.17.D+20.1 через Net4. Для перевірки використати traceroute.

5. Розрахувати спільну адресу та маску (summarizing) адрес 172.17.D+10.1 та 172.17.D+20.1, при чому префікс має бути максимально можливим. Видалити маршрути, встановлені на попередньому кроці та замінити їх об'єднаним маршрутом, якій має проходити через Server_1.

6. Налаштувати SSH сервіс таким чином, щоб Client_1 та Client_2 могли підключатись до Server_1 та один до одного.

7. Налаштуйте на Server_1 firewall таким чином:
• Дозволено підключатись через SSH з Client_1 та заборонено з Client_2
• З Client_2 на 172.17.D+10.1 ping проходив, а на 172.17.D+20.1 не проходив

Практична частина модуля Linux Networking передбачає створення засобами Virtual Box мережі, що показаний на рисунку 1.

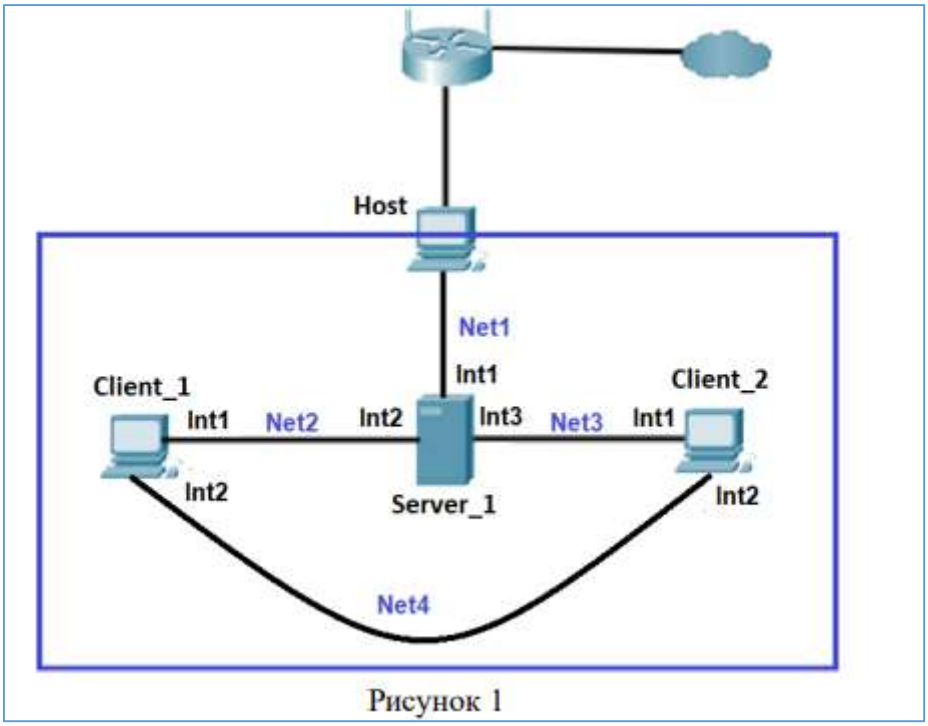


Рисунок 1

Host – це комп'ютер, на якому запущений Virtual Box;

Server_1 – Віртуальна машина, на якій розгорнуто ОС Linux. **Int1** цієї машини в режимі «**Мережевий міст**» підключений до мережі **Net1**, тобто знаходиться в адресному просторі домашньої мережі. IP-адреса **Int1** встановлюється статично відповідно до адресного простору, наприклад 192.168.1.200/24. Інтерфейси **Int2** та **Int3** відповідно підключено в режимі «**Внутрішня мережа**» до мереж **Net2** та **Net3**.

Client_1 та **Client_2** – Віртуальні машини, на яких розгорнуто ОС Linux (бажано різні дистрибутиви, наприклад Ubuntu та CentOS). Інтерфейси підключені в режимі «**Внутрішня мережа**» до мереж **Net2**, **Net3** та **Net4** як показано на рисунку 1.

Адреса мережі **Net2** – 10.Y.D.0/24, де **Y** – дві останні цифри з вашого року народження, **D** – дата народження.

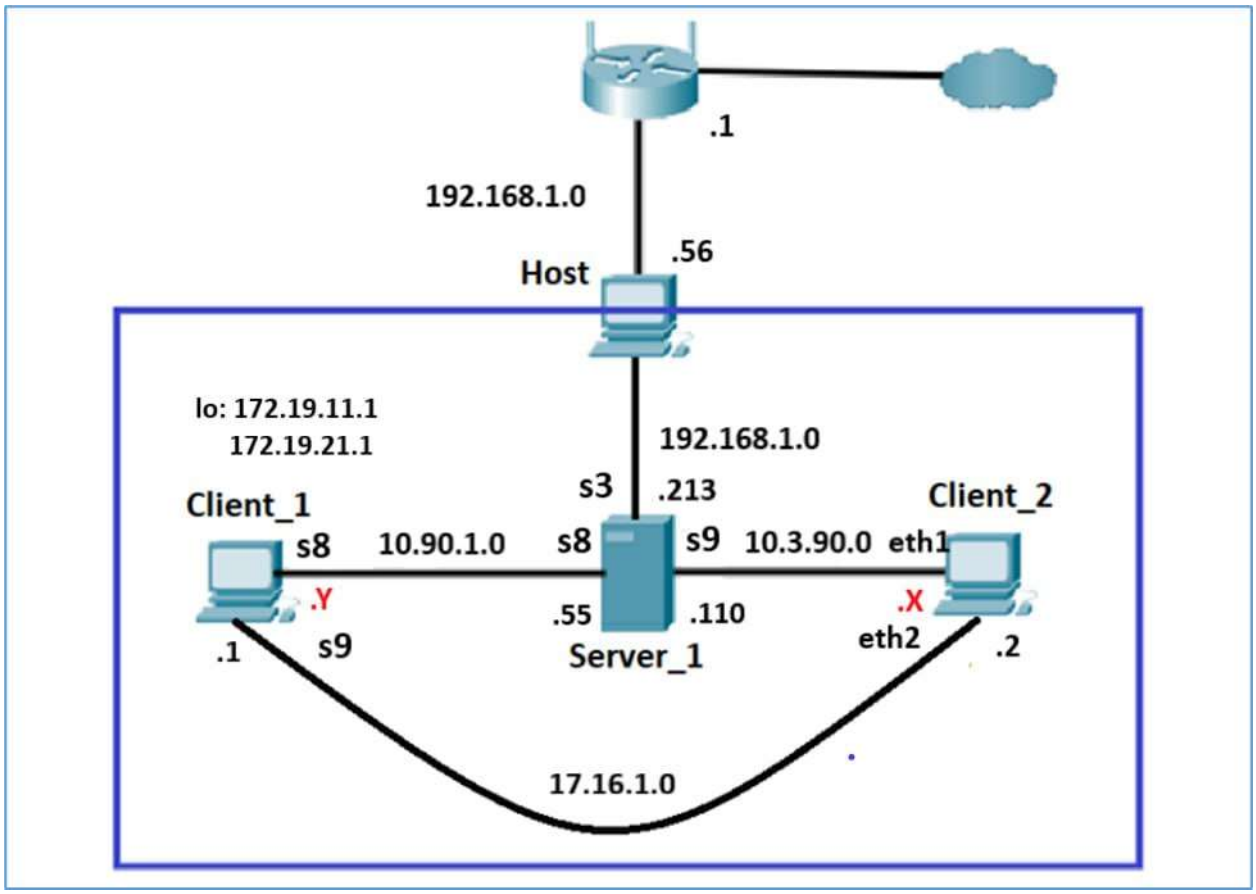
Адреса мережі **Net3** – 10.M.Y.0/24, де **M** – номер місяця народження.

Адреса мережі **Net4** – 172.16.D.0/24.

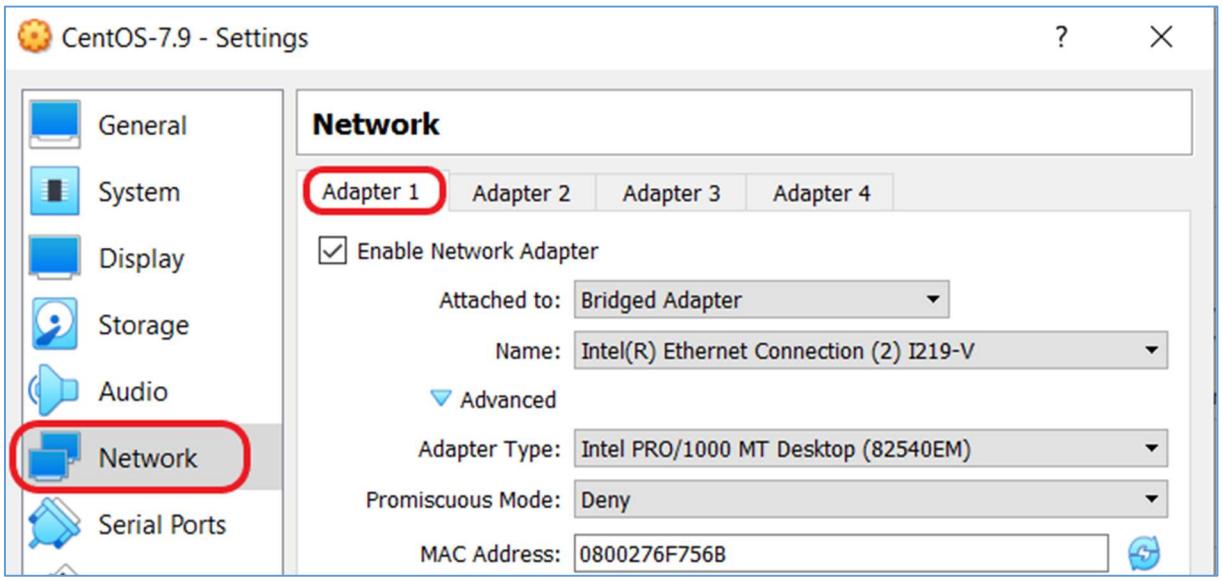
Увага! Якщо, адресний простір **Net2**, **Net3** або **Net4** перетинається з адресним простором **Net1** – відповідну адресу можна змінити на власний розсуд.

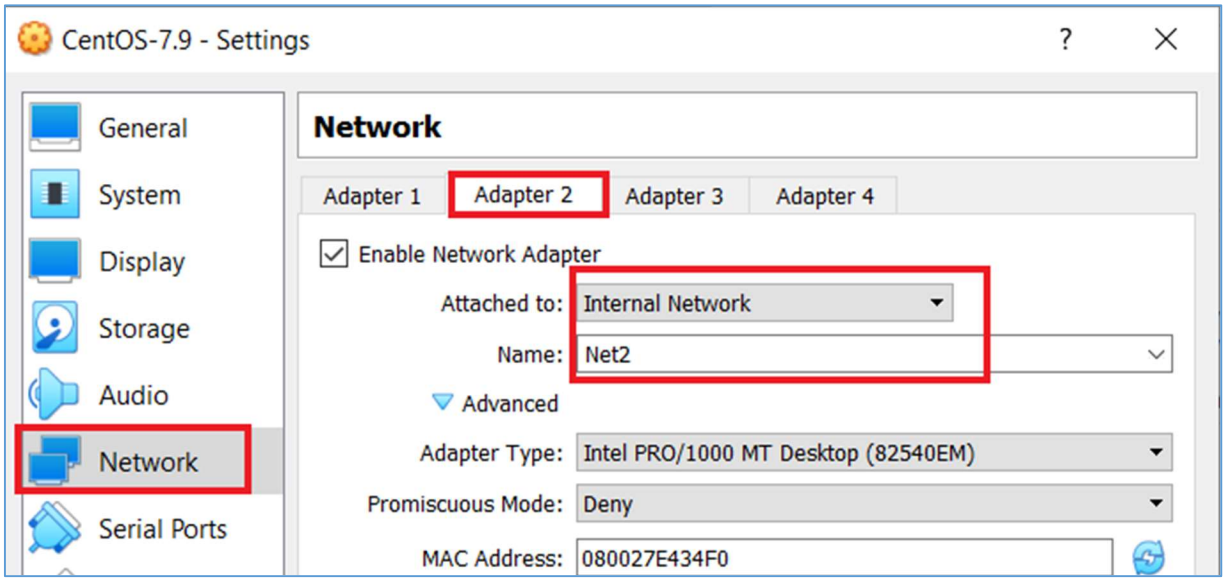
Адреса мережі **Net1** – 192.168.1.0/24
Адреса мережі **Net2** – 10.90.1.0/24
Адреса мережі **Net3** – 10.3.90.0/24
Адреса мережі **Net4** – 172.16.1.0/24

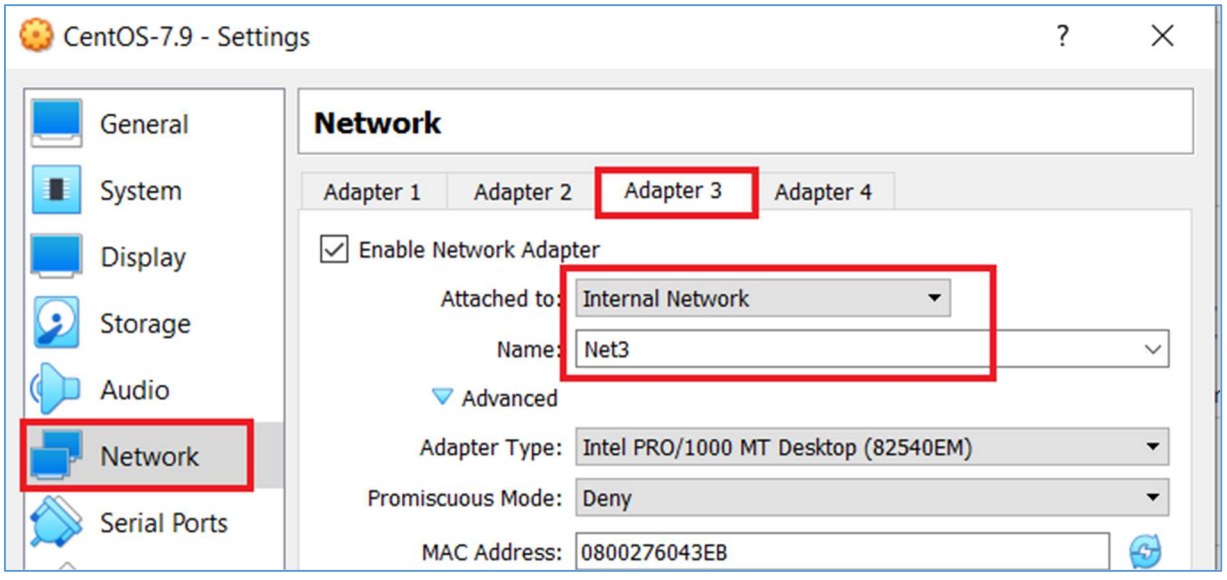
- **Network topology:**



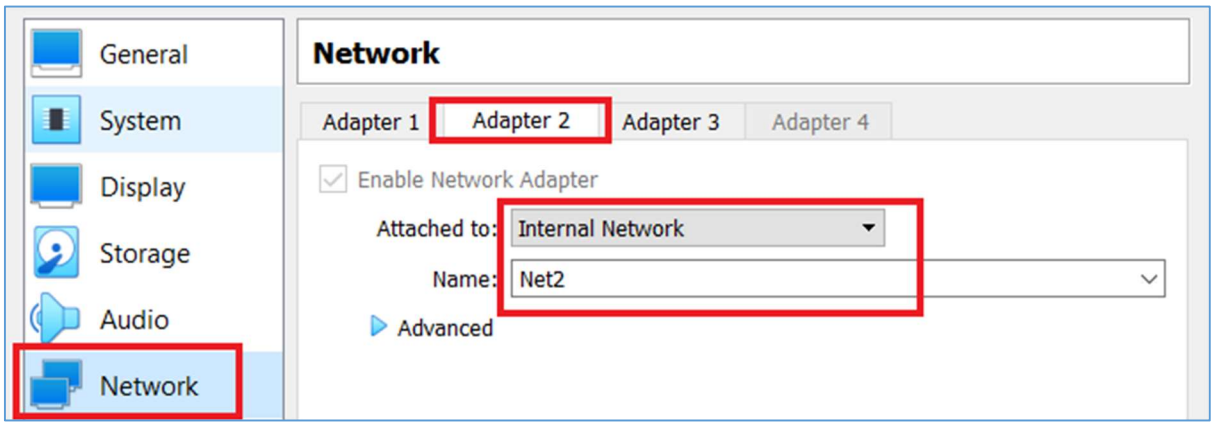
- **Setting network on Server1 (CentOS 7):**

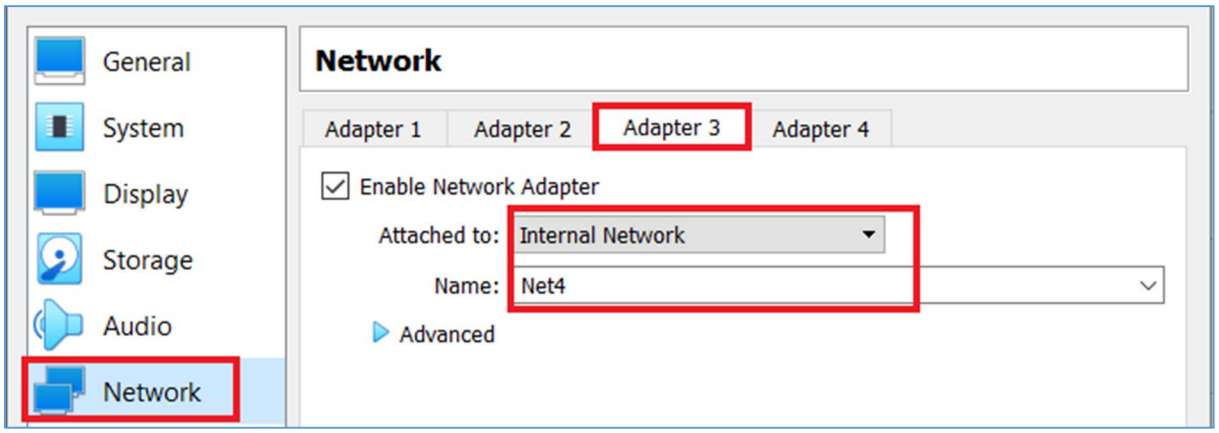




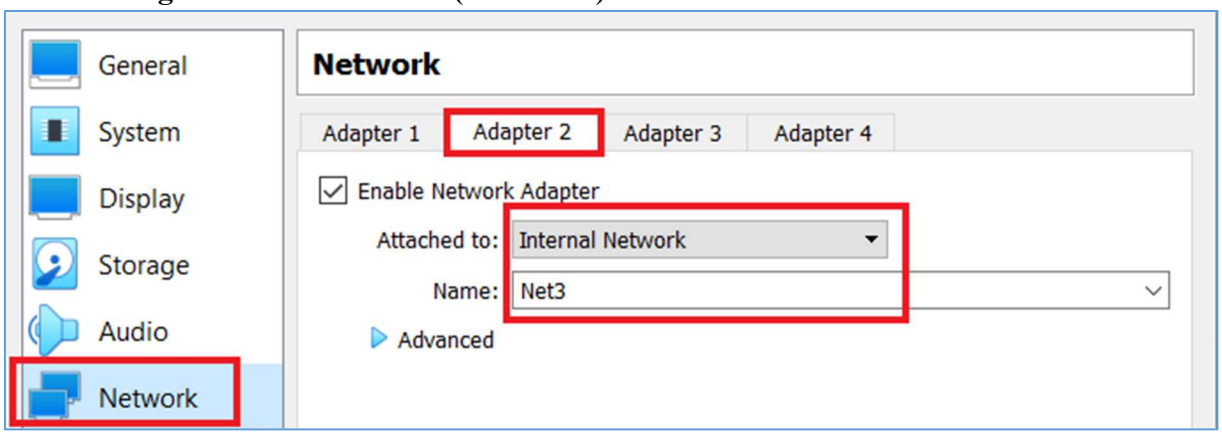


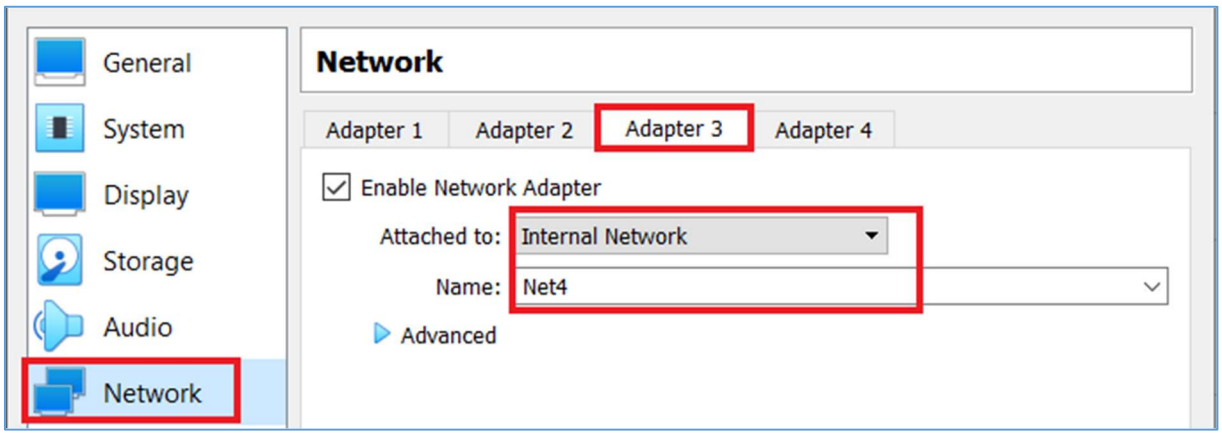
- **Setting network on Client1 (Ubuntu 22.04):**





- **Setting network on Client2 (CentOS 7):**





1. На Server1 налаштувати статичні адреси на всіх інтерфейсах.

Configuring static network addresses on Server1 (CentOS 7):

```
→  ~ hostname
Server1
→  ~
→  ~ ls /sys/class/net
enp0s3  enp0s8  enp0s9  lo  virbr0  virbr0-nic
→  ~
→  ~ ip a sh dev enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:6f:75:6b brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.213/24 brd 192.168.1.255 scope global noprefixroute dynamic enp0s3
       valid_lft 85790sec preferred_lft 85790sec
    inet6 fe80::5ff4:2b6c:cbc0:4d97/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
→  ~
→  ~ ip a sh dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:e4:34:f0 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::ae41:b84a:62f4:b212/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
→  ~
→  ~ ip a sh dev enp0s9
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:60:43:eb brd ff:ff:ff:ff:ff:ff
    inet6 fe80::6aba:ea64:11f9:cb5c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
→  ~
```

```
→  network-scripts cat ifcfg-enp0s3
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="dhcp"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s3"
UUID="fec476c2-0e6d-4ea3-964a-e5186f263cd3"
DEVICE="enp0s3"
ONBOOT="yes"
```

```
→  network-scripts cat ifcfg-enp0s8
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="none"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s8"
UUID="fec476c2-0e6d-4ea3-964a-e5186f263cd3"
DEVICE="enp0s8"
ONBOOT="yes"
IPADDR=10.90.1.55
NETMASK=255.255.255.0
GATEWAY=10.90.1.1
DNS1=8.8.8.8
```

```
→  network-scripts cat ifcfg-enp0s9
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="none"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s9"
UUID="fec476c2-0e6d-4ea3-964a-e5186f263cd3"
DEVICE="enp0s9"
ONBOOT="yes"
IPADDR=10.3.90.110
NETMASK=255.255.255.0
GATEWAY=10.3.90.1
DNS1=8.8.8.8
```

```
→  network-scripts ip addr sh dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:e4:34:f0 brd ff:ff:ff:ff:ff:ff
    inet 10.90.1.55/24 brd 10.90.1.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::ae41:b84a:62f4:b212/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
→  network-scripts
→  network-scripts ip addr sh dev enp0s9
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:60:43:eb brd ff:ff:ff:ff:ff:ff
    inet 10.3.90.110/24 brd 10.3.90.255 scope global enp0s9
       valid_lft forever preferred_lft forever
    inet6 fe80::6aba:ea64:11f9:cb5c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
→  network-scripts
```

For these changes to take effect, you must restart the network interfaces and service with the following commands:

```
$ sudo systemctl stop NetworkManager
$ sudo systemctl disable NetworkManager
$ sudo ifdown <<interface_name>>
$ sudo ifup <<interface_name>>
$ sudo systemctl restart network
```

Configuring The Default Gateway (CentOS 7):

The **default gateway** is determined by the network scripts which parse the **/etc/sysconfig/network** file first and then the network interface **ifcfg** files for interfaces that are "up". The **ifcfg** files are parsed in numerically ascending order, and the **last GATEWAY** directive to be read is used to compose a **default route** in the **routing table**.

In special cases where it is necessary to influence NetworkManager's selection of the exit interface to be used to reach a gateway, make use of the `DEFROUTE="no"` command in the **ifcfg** files for those interfaces which do not lead to the default gateway.

```
→  sysconfig cat network
# Created by anaconda
NETWORKING=yes
GATEWAY=192.168.1.1
```

2. На Server1 налаштувати DHCP, який буде конфігурувати адреси Int1 Client1 та Client2.

Configuring DHCP server on Server1 (CentOS 7):

The following is a **/etc/dhcp/dhcpd.conf** file, for a server that has two network interfaces, **enp0s8** in a 10.90.1.0/24 network, and **enp0s9** in a 10.3.90.0/24 network. Multiple subnet declarations allow you to define different settings for multiple networks:

```
  GNU nano 2.3.1                                File: dhcpd.conf

# dhcpd.conf
#
# Sample configuration file for ISC dhcp

default-lease-time 600;
max-lease-time 7200;
authoritative;

# Configuration for an internal subnet Net2 - Client1 Int1.
subnet 10.90.1.0 netmask 255.255.255.0 {
  range 10.90.1.5 10.90.1.50;
  option domain-name-servers 192.168.1.1;
  # option domain-name "internal.example.org";
  option routers 10.90.1.1;
  option broadcast-address 10.90.1.255;
}

# Configuration for an internal subnet Net3 - Client2 Int1.
subnet 10.3.90.0 netmask 255.255.255.0 {
  range 10.3.90.5 10.3.90.50;
  option domain-name-servers 192.168.1.1;
  # option domain-name "internal.example.org";
  option routers 10.3.90.1;
  option broadcast-address 10.3.90.1;
}


# Fixed IP addresses can also be specified for hosts.
host Clien1 {
  hardware ethernet 08:00:27:4C:7C:F9;
  fixed-address 10.90.1.51;
}

host Clien2 {
  hardware ethernet 08:00:27:31:9B:10;
  fixed-address 10.3.90.51;
}
```

The DHCP daemon listens on all network interfaces unless otherwise specified. Use the **/etc/sysconfig/dhcpd** file to specify which network interfaces the **DHCP** daemon listens on. The following **/etc/sysconfig/dhcpd** example specifies that the **DHCP** daemon listens on the **enp0s8** and **enp0s9** interfaces:

```
→  sysconfig cat dhcpd
# WARNING: This file is NOT used anymore.

# If you are here to restrict what interfaces should dhcpd listen on,
# be aware that dhcpd listens *only* on interfaces for which it finds subnet
# declaration in dhcpd.conf. It means that explicitly enumerating interfaces
# also on command line should not be required in most cases.

# If you still insist on adding some command line options,
# copy dhcpd.service from /lib/systemd/system to /etc/systemd/system and modify
# it there.
# https://fedoraproject.org/wiki/Systemd#How_do_I_customize_a_unit_file.2F_add_a_custom_unit_file.3F

# example:
# $ cp /usr/lib/systemd/system/dhcpd.service /etc/systemd/system/
# $ vi /etc/systemd/system/dhcpd.service
# $ ExecStart=/usr/sbin/dhcpd -f -cf /etc/dhcp/dhcpd.conf -user dhcpd -group dhcpd --no-pid <your_inte
# $ systemctl --system daemon-reload
# $ systemctl restart dhcpd.service

DHCPDARGS="enp0s8 enp0s9";
```

Now that you have configured your **DHCP** server daemon, you need to **start** the service for the mean time and **enable** it to start automatically from the next system boot, and **check** if its up and running using following commands:

```
$ sudo systemctl start dhcpd
$ sudo systemctl enable dhcpd
$ sudo systemctl status dhcpd
```

```
→  dhcp sudo systemctl start dhcpd
[sudo] password for yevhen_yakymov:
→  dhcp
→  dhcp
→  dhcp
→  dhcp sudo systemctl enable dhcpd
Created symlink from /etc/systemd/system/multi-user.target.wants/dhcpd.service to /usr/lib/systemd/system/dhcpd.service.
→  dhcp
→  dhcp sudo systemctl status dhcpd
● dhcpd.service - DHCPv4 Server Daemon
   Loaded: loaded (/usr/lib/systemd/system/dhcpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Sun 2023-01-22 13:59:03 EET; 1min 1s ago
     Docs: man:dhcpd(8)
           man:dhcpd.conf(5)
 Main PID: 4118 (dhcpd)
   Status: "Dispatching packets..."
   CGroup: /system.slice/dhcpd.service
           └─4118 /usr/sbin/dhcpd -f -cf /etc/dhcp/dhcpd.conf -user dhcpd -group dhcpd --no-pid
```

Next, permit requests to the **DHCP** daemon on **Firewall**, which listens on port **67/UDP**, by running:

```
$ firewall-cmd --zone=public --permanent --add-service=dhcp
$ firewall-cmd --reload
```

```
→  dhcp
→  dhcp sudo firewall-cmd --zone=public --permanent --add-service=dhcp
[sudo] password for yevhen_yakymov:
success
→  dhcp
→  dhcp sudo firewall-cmd --reload
success
→  dhcp
→  dhcp
```

Enable or disable IP forwarding:

Switch on routing is needed only on transit devices. To check out routing enable use **sysctl net.ipv4.conf.all.forwarding** command. To switch “on” or “off” routing you must edit **/etc/sysctl.conf** file or use **sysctl -w net.ipv4.ip_forward=1(0)** command.

```
$ sysctl net.ipv4.ip_forward
$ sysctl -w net.ipv4.ip_forward=1

0  - Disable
1  - Enable
```

Configuring DHCP clients.

Client1 – Ubuntu 22.04:

On Ubuntu 18.04 and later, networking is controlled by the **Netplan** program. You need to edit the appropriate file under the directory **/etc/netplan/**. Then enable **dhcp4** under a specific interface for example under ethernets, **enp0s8**, and comment out static IP related configs:

```
vagrant@Client1:~$
vagrant@Client1:~$ cd /etc/netplan/
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ ll
total 16
drwxr-xr-x   2 root root 4096 Jan  4 20:38 ./
drwxr-xr-x 105 root root 4096 Jan 21 13:36 ../
-rw-r--r--   1 root root  485 Nov 11 08:54 50-cloud-init.yaml
-rw-r--r--   1 root root   90 Jan  4 20:38 50-vagrant.yaml
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ cat 50-vagrant.yaml
---
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s8:
      dhcp4: true
    enp0s9:
      addresses: [172.16.1.1/24]
```

Save the changes and run the following command to effect the changes:

```
$ sudo netplan apply
```

Client2 – CentOS 7:

On CentOS, the interface config files ate located at **/etc/sysconfig/network-scripts/:**

```
 GNU nano 2.3.1                                File: ifcfg-eth1

#VAGRANT-BEGIN
# The contents below are automatically generated by Vagrant.
NM_CONTROLLED=yes
BOOTPROTO=dhcp
ONBOOT=yes
IPADDR=
NETMASK=255.255.255.0
DEVICE=eth1
```

```
 GNU nano 2.3.1                                File: ifcfg-eth2

#VAGRANT-BEGIN
# The contents below are automatically generated by Vagrant.
NM_CONTROLLED=yes
BOOTPROTO=none
ONBOOT=yes
IPADDR=172.16.1.2
NETMASK=255.255.255.0
DEVICE=eth2
```

Save the file and restart network service (or reboot the system):

```
$ sudo systemctl restart network
```

3. За допомогою команд ping та traceroute перевірити зв'язок між віртуальними машинами. Результат пояснити.

Ping and traceroute from Server1 (Int2-10.90.1.55) to Client1 (Int1-10.90.1.5):

```
→  ~ hostname
Server1
→  ~ ping 10.90.1.5
PING 10.90.1.5 (10.90.1.5) 56(84) bytes of data.
64 bytes from 10.90.1.5: icmp_seq=1 ttl=64 time=0.876 ms
64 bytes from 10.90.1.5: icmp_seq=2 ttl=64 time=2.31 ms
64 bytes from 10.90.1.5: icmp_seq=3 ttl=64 time=2.68 ms
64 bytes from 10.90.1.5: icmp_seq=4 ttl=64 time=1.94 ms
^C
--- 10.90.1.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3061ms
rtt min/avg/max/mdev = 0.876/1.956/2.684/0.676 ms
→  ~
→  ~ traceroute 10.90.1.5
traceroute to 10.90.1.5 (10.90.1.5), 30 hops max, 60 byte packets
 1  10.90.1.5 (10.90.1.5)  0.788 ms  1.080 ms  0.576 ms
```

Ping and traceroute from Server1 (Int3-10.3.90.110) to Client2 (Int1-10.3.90.5):

```
→  ~ hostname
Server1
→  ~
→  ~ ping 10.3.90.5
PING 10.3.90.5 (10.3.90.5) 56(84) bytes of data.
64 bytes from 10.3.90.5: icmp_seq=1 ttl=64 time=0.812 ms
64 bytes from 10.3.90.5: icmp_seq=2 ttl=64 time=0.736 ms
^C
--- 10.3.90.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1004ms
rtt min/avg/max/mdev = 0.736/0.774/0.812/0.038 ms
→  ~
→  ~ traceroute 10.3.90.5
traceroute to 10.3.90.5 (10.3.90.5), 30 hops max, 60 byte packets
 1  10.3.90.5 (10.3.90.5)  0.764 ms  0.664 ms  0.584 ms
```

Ping and traceroute from Client1 (Int1-10.90.1.5) to Server1 (Int2 - 10.90.1.55):

```
vagrant@Client1:/etc/netplan$ hostname
Client1
vagrant@Client1:/etc/netplan$ ping 10.90.1.55
PING 10.90.1.55 (10.90.1.55) 56(84) bytes of data.
64 bytes from 10.90.1.55: icmp_seq=1 ttl=64 time=0.901 ms
64 bytes from 10.90.1.55: icmp_seq=2 ttl=64 time=0.980 ms
^C
--- 10.90.1.55 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.901/0.940/0.980/0.039 ms
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ traceroute 10.90.1.55
traceroute to 10.90.1.55 (10.90.1.55), 64 hops max
  1   10.90.1.55  1.104ms !*  0.557ms !*  0.536ms !*
vagrant@Client1:/etc/netplan$
```

Ping and traceroute from Client2 (Int1-10.3.90.5) to Server1 (Int3 - 10.3.90.110):

```
[vagrant@Client2 network-scripts]$ hostname
Client2
[vagrant@Client2 network-scripts]$ ping 10.3.90.110
PING 10.3.90.110 (10.3.90.110) 56(84) bytes of data.
64 bytes from 10.3.90.110: icmp_seq=1 ttl=64 time=0.822 ms
64 bytes from 10.3.90.110: icmp_seq=2 ttl=64 time=1.71 ms
64 bytes from 10.3.90.110: icmp_seq=3 ttl=64 time=1.52 ms
^C
--- 10.3.90.110 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 0.822/1.355/1.716/0.387 ms
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ traceroute 10.3.90.110
traceroute to 10.3.90.110 (10.3.90.110), 30 hops max, 60 byte packets
 1  10.3.90.110 (10.3.90.110)  0.551 ms !X  0.326 ms !X  0.468 ms !X
[vagrant@Client2 network-scripts]$
```

Ping and traceroute from Client1 (Int1-10.90.1.5) to Server1 (Int1 - 192.168.1.213):

```
vagrant@Client1:/etc/netplan$ hostname
Client1
vagrant@Client1:/etc/netplan$ ping 192.168.1.213
PING 192.168.1.213 (192.168.1.213) 56(84) bytes of data.
64 bytes from 192.168.1.213: icmp_seq=1 ttl=63 time=1.14 ms
64 bytes from 192.168.1.213: icmp_seq=2 ttl=63 time=3.08 ms
^C
--- 192.168.1.213 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1003ms
rtt min/avg/max/mdev = 1.136/2.110/3.084/0.974 ms
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ traceroute 192.168.1.213
traceroute to 192.168.1.213 (192.168.1.213), 64 hops max
  1   10.0.2.2  0.405ms  0.002ms  0.002ms
  2   *  *  *
  3   *  *  *
```

Ping and traceroute from Client2 (Int1-10.3.90.5) to Server1 (Int1 - 192.168.1.213):

```
[vagrant@Client2 network-scripts]$ hostname
Client2
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ ping 192.168.1.213
PING 192.168.1.213 (192.168.1.213) 56(84) bytes of data.
64 bytes from 192.168.1.213: icmp_seq=1 ttl=63 time=1.61 ms
64 bytes from 192.168.1.213: icmp_seq=2 ttl=63 time=3.93 ms
64 bytes from 192.168.1.213: icmp_seq=3 ttl=63 time=4.85 ms
^C
--- 192.168.1.213 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 1.613/3.468/4.852/1.363 ms
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ traceroute 192.168.1.213
traceroute to 192.168.1.213 (192.168.1.213), 30 hops max, 60 byte packets
 1  gateway (10.0.2.2)  0.290 ms  0.259 ms  0.274 ms
 2  * * *
 3  * * *
```

Ping and traceroute from Client1 (Int1-10.90.1.5) to Client2 (Int1 – 10.3.90.5):

```
vagrant@Client1:/etc/netplan$ hostname
Client1
vagrant@Client1:/etc/netplan$ ping 10.3.90.5 -c3
PING 10.3.90.5 (10.3.90.5) 56(84) bytes of data.
^C
--- 10.3.90.5 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2047ms

vagrant@Client1:/etc/netplan$ traceroute 10.3.90.5
traceroute to 10.3.90.5 (10.3.90.5), 64 hops max
  1   10.0.2.2  0.003ms  0.002ms  0.001ms
  2   *  *  *
```

Ping and traceroute from Client2 (Int1-10.3.90.5) to Client1 (Int1 - 10.90.1.5):

```
[vagrant@Client2 network-scripts]$ hostname
Client2
[vagrant@Client2 network-scripts]$ ping 10.90.1.5 -c3
PING 10.90.1.5 (10.90.1.5) 56(84) bytes of data.
^C
--- 10.90.1.5 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2001ms

[vagrant@Client2 network-scripts]$ traceroute 10.90.1.5
traceroute to 10.90.1.5 (10.90.1.5), 30 hops max, 60 byte packets
 1  gateway (10.0.2.2)  0.373 ms  0.203 ms  1.588 ms
 2  * * *
 3  * * *
```

Ping and traceroute from Client1 (Int2-172.16.1.1) to Client2 (Int2 – 172.16.1.2):

```
vagrant@Client1:/etc/netplan$ hostname
Client1
vagrant@Client1:/etc/netplan$ ping 172.16.1.2 -c3
PING 172.16.1.2 (172.16.1.2) 56(84) bytes of data.
64 bytes from 172.16.1.2: icmp_seq=1 ttl=64 time=1.40 ms
64 bytes from 172.16.1.2: icmp_seq=2 ttl=64 time=0.760 ms
64 bytes from 172.16.1.2: icmp_seq=3 ttl=64 time=0.736 ms

--- 172.16.1.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 0.736/0.966/1.403/0.308 ms
vagrant@Client1:/etc/netplan$ traceroute 172.16.1.2
traceroute to 172.16.1.2 (172.16.1.2), 64 hops max
  1   172.16.1.2  0.919ms  0.539ms  0.775ms
```

Ping and traceroute from Client2 (Int2 – 172.16.1.2) to Client1 Int2-172.16.1.1):

```
[vagrant@Client2 network-scripts]$ hostname
Client2
[vagrant@Client2 network-scripts]$ ping 172.16.1.1 -c3
PING 172.16.1.1 (172.16.1.1) 56(84) bytes of data.
64 bytes from 172.16.1.1: icmp_seq=1 ttl=64 time=0.824 ms
64 bytes from 172.16.1.1: icmp_seq=2 ttl=64 time=0.664 ms
64 bytes from 172.16.1.1: icmp_seq=3 ttl=64 time=0.655 ms

--- 172.16.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.655/0.714/0.824/0.080 ms
[vagrant@Client2 network-scripts]$ traceroute 172.16.1.1
traceroute to 172.16.1.1 (172.16.1.1), 30 hops max, 60 byte packets
 1  172.16.1.1 (172.16.1.1)  0.459 ms  0.562 ms  0.727 ms
```

Conclutions:

The **Client1** cannot ping **Client2** via **Server1**, because they are not in the same subnet: **10.90.1.0/24** for the **Client1(enp0s8)** network and **10.3.90.0/24** for the **Client2(eth1)** network. As the two hosts are not part of the same subnet, the ping command goes to the default gateway. So we should add static routes for **Client1** and **Client2.**

And the same time **Client1** can ping **Client2** from interface **enp0s9**, because they are in the same subnet: **172.16.1.0/24** for the **Client1(enp0s9)** network and **172.16.1.0/24** for the **Client2(eth2).**

Route table for Server1:

```
→  ~ hostname
Server1
→  ~ ip route show
default via 192.168.1.1 dev enp0s3
10.3.90.0/24 dev enp0s9 proto kernel scope link src 10.3.90.110
10.90.1.0/24 dev enp0s8 proto kernel scope link src 10.90.1.55
169.254.0.0/16 dev enp0s3 scope link metric 1002
169.254.0.0/16 dev enp0s8 scope link metric 1003
169.254.0.0/16 dev enp0s9 scope link metric 1004
192.168.1.0/24 dev enp0s3 proto kernel scope link src 192.168.1.213
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
```

Route table for Client1:

```
vagrant@Client1:/etc/netplan$ hostname
Client1
vagrant@Client1:/etc/netplan$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG    100    0        0 enp0s3
0.0.0.0         10.90.1.1       0.0.0.0         UG    100    0        0 enp0s8
10.0.2.0        0.0.0.0         255.255.255.0   U     0      0        0 enp0s3
10.0.2.2        0.0.0.0         255.255.255.255 UH    100    0        0 enp0s3
10.90.1.0       0.0.0.0         255.255.255.0   U     0      0        0 enp0s8
10.90.1.1       0.0.0.0         255.255.255.255 UH    100    0        0 enp0s8
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
172.18.0.0      0.0.0.0         255.255.0.0     U     0      0        0 br-21c069d61709
vagrant@Client1:/etc/netplan$
```

Route table for Client2:

```
[vagrant@Client2 network-scripts]$ hostname
Client2
[vagrant@Client2 network-scripts]$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG    103    0        0 eth0
0.0.0.0         10.3.90.1       0.0.0.0         UG    104    0        0 eth1
10.0.2.0        0.0.0.0         255.255.255.0   U     103    0        0 eth0
10.3.90.0       0.0.0.0         255.255.255.0   U     104    0        0 eth1
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
[vagrant@Client2 network-scripts]$
```

Temporary route adding

Client1:

```
vagrant@Client1:~$ sudo ip route add 10.3.90.0/24 via 10.90.1.55 dev enp0s8
vagrant@Client1:~$
vagrant@Client1:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG    100    0        0 enp0s3
0.0.0.0         10.90.1.1       0.0.0.0         UG    100    0        0 enp0s8
10.0.2.0        0.0.0.0         255.255.255.0   U     0      0        0 enp0s3
10.0.2.2        0.0.0.0         255.255.255.255 UH    100    0        0 enp0s3
10.3.90.0       10.90.1.55      255.255.255.0   UG    0      0        0 enp0s8
10.90.1.0       0.0.0.0         255.255.255.0   U     0      0        0 enp0s8
10.90.1.1       0.0.0.0         255.255.255.255 UH    100    0        0 enp0s8
```

Client2:

```
[vagrant@Client2 ~]$ sudo ip route add 10.90.1.0/24 via 10.3.90.110 dev eth1
[vagrant@Client2 ~]$
[vagrant@Client2 ~]$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG    100    0        0 eth0
0.0.0.0         10.3.90.1       0.0.0.0         UG    101    0        0 eth1
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0
10.3.90.0       0.0.0.0         255.255.255.0   U     101    0        0 eth1
10.90.1.0       10.3.90.110     255.255.255.0   UG    0      0        0 eth1
```

Ping and traceroute from Client1 (Int1-10.90.1.5) to Client2 (Int1 - 10.3.90.5):

```
vagrant@Client1:~$ ping 10.3.90.5 -c 4
PING 10.3.90.5 (10.3.90.5) 56(84) bytes of data.
64 bytes from 10.3.90.5: icmp_seq=1 ttl=63 time=1.69 ms
64 bytes from 10.3.90.5: icmp_seq=2 ttl=63 time=1.95 ms
64 bytes from 10.3.90.5: icmp_seq=3 ttl=63 time=1.33 ms
64 bytes from 10.3.90.5: icmp_seq=4 ttl=63 time=1.23 ms

--- 10.3.90.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.226/1.548/1.946/0.288 ms
vagrant@Client1:~$
vagrant@Client1:~$ traceroute 10.3.90.5
traceroute to 10.3.90.5 (10.3.90.5), 64 hops max
  1   10.90.1.55  0.800ms  0.572ms  0.825ms
  2   10.90.1.55  0.637ms !*  0.544ms !*  0.540ms !*
```

Ping and traceroute from Client2 (Int1 - 10.3.90.5) to Client1 (Int1 - 10.90.1.5):

```
[vagrant@Client2 ~]$ ping 10.90.1.5 -c 3
PING 10.90.1.5 (10.90.1.5) 56(84) bytes of data.
64 bytes from 10.90.1.5: icmp_seq=1 ttl=63 time=2.12 ms
64 bytes from 10.90.1.5: icmp_seq=2 ttl=63 time=5.71 ms
64 bytes from 10.90.1.5: icmp_seq=3 ttl=63 time=2.23 ms

--- 10.90.1.5 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 2.128/3.357/5.711/1.665 ms
[vagrant@Client2 ~]$
[vagrant@Client2 ~]$ traceroute 10.90.1.5
traceroute to 10.90.1.5 (10.90.1.5), 30 hops max, 60 byte packets
 1  10.3.90.110 (10.3.90.110)  0.957 ms  0.731 ms  0.786 ms
 2  10.3.90.110 (10.3.90.110)  1.585 ms !X  1.089 ms !X  0.739 ms !X
```

Awesome, we have successfully added a route from one Linux computer to another!

Permanent route adding

For Ubuntu 18.04 and later using Netplan:

```
  GNU nano 4.8                                    50-vagrant.yaml
---
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s8:
      dhcp4: true
      routes:
        - to: 10.3.90.0/24
          via: 10.90.1.55
    enp0s9:
      addresses: [172.16.1.1/24]
```

```
vagrant@Client1:/etc/netplan$ sudo netplan apply
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG    100    0        0 enp0s3
0.0.0.0         10.90.1.1       0.0.0.0         UG    100    0        0 enp0s8
10.0.2.0        0.0.0.0         255.255.255.0   U     0      0        0 enp0s3
10.0.2.2        0.0.0.0         255.255.255.255 UH    100    0        0 enp0s3
10.3.90.0       10.90.1.55      255.255.255.0   UG    0      0        0 enp0s8
10.90.1.0       0.0.0.0         255.255.255.0   U     0      0        0 enp0s8
10.90.1.1       0.0.0.0         255.255.255.255 UH    100    0        0 enp0s8
172.16.1.0      0.0.0.0         255.255.255.0   U     0      0        0 enp0s9
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
172.18.0.0      0.0.0.0         255.255.0.0     U     0      0        0 br-21c069d61709
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ traceroute 10.3.90.5
traceroute to 10.3.90.5 (10.3.90.5), 64 hops max
  1   10.90.1.55  1.618ms  0.537ms  0.591ms
  2   10.90.1.55  0.501ms !*  0.653ms !*  0.569ms !*
```

For CentOS 7:

On **RHEL** and **CentOS** distributions, you need to create a file named "**route-**" in the **"/etc/sysconfig/network-scripts"** folder:

```
[vagrant@Client2 ~]$ cd /etc/sysconfig/network-scripts/
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ sudo nano route-eth1
You have new mail in /var/spool/mail/vagrant
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ cat route-eth1
ADDRESS0=10.90.1.0
NETMASK0=255.255.255.0
GATEWAY0=10.3.90.110
```

```
[vagrant@Client2 network-scripts]$ sudo systemctl restart network
```

```
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ traceroute 10.90.1.5
traceroute to 10.90.1.5 (10.90.1.5), 30 hops max, 60 byte packets
 1  10.3.90.110 (10.3.90.110)  0.598 ms  0.319 ms  0.203 ms
 2  10.3.90.110 (10.3.90.110)  0.923 ms !X  0.815 ms !X  0.295 ms !X
```

4. На віртуальному інтерфейсу lo Client1 призначити дві IP адреси за таким правилом: 172.19.D+10.1/24 та 172.19.D+20.1/24. Налаштувати маршрутизацію таким чином, щоб трафік з Client2 до 172.19.D+10.1 проходив через Server1, а до 172.19.D+20.1 через Net4. Для перевірки використати traceroute.

Client1 (lo) – 172.19.11.1/24
Client1 (lo) – 172.19.21.1/24

Output addresses of the lo interface:

```
vagrant@Client1:/etc/netplan$ hostname
Client1
vagrant@Client1:/etc/netplan$ ip a sh dev lo
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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
```

Add addresses to the lo interface:

```
vagrant@Client1:/etc/netplan$ sudo ip addr add 172.19.11.1/24 dev lo
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ sudo ip addr add 172.19.21.1/24 dev lo
vagrant@Client1:/etc/netplan$
vagrant@Client1:/etc/netplan$ ip a sh dev lo
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    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
    inet 172.19.11.1/24 scope global lo
       valid_lft forever preferred_lft forever
    inet 172.19.21.1/24 scope global lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
```

Add route from Client2 to 172.19.21.1 via Net4(172.16.1.0/24) and check:

```
[vagrant@Client2 network-scripts]$ sudo ip route add 172.19.21.0/24 via 172.16.1.1
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ ip r
default via 10.0.2.2 dev eth0 proto dhcp metric 100
default via 10.3.90.1 dev eth1 proto dhcp metric 101
default via 172.16.1.1 dev eth2 proto static metric 102
10.0.2.0/24 dev eth0 proto kernel scope link src 10.0.2.15 metric 100
10.3.90.0/24 dev eth1 proto kernel scope link src 10.3.90.5 metric 101
10.90.1.0/24 via 10.3.90.110 dev eth1 proto static metric 101
172.16.1.0/24 dev eth2 proto kernel scope link src 172.16.1.2 metric 102
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
172.19.21.0/24 via 172.16.1.1 dev eth2
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ ping 172.19.21.1 -c2
PING 172.19.21.1 (172.19.21.1) 56(84) bytes of data.
64 bytes from 172.19.21.1: icmp_seq=1 ttl=64 time=0.732 ms
64 bytes from 172.19.21.1: icmp_seq=2 ttl=64 time=0.621 ms

--- 172.19.21.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.621/0.676/0.732/0.061 ms
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ traceroute 172.19.21.1
traceroute to 172.19.21.1 (172.19.21.1), 30 hops max, 60 byte packets
 1  172.19.21.1 (172.19.21.1)  0.887 ms  0.748 ms  0.282 ms
[vagrant@Client2 network-scripts]$
```

Add routes from Client2 to 172.19.11.1 via Server1 and check:

On Server1

```
→  network-scripts sudo ip route add 172.19.11.0/24 via 10.90.1.5
[sudo] password for yevhen_yakymov:
→  network-scripts ip r
default via 192.168.1.1 dev enp0s3
10.3.90.0/24 dev enp0s9 proto kernel scope link src 10.3.90.110
10.90.1.0/24 dev enp0s8 proto kernel scope link src 10.90.1.55
169.254.0.0/16 dev enp0s3 scope link metric 1002
169.254.0.0/16 dev enp0s8 scope link metric 1003
169.254.0.0/16 dev enp0s9 scope link metric 1004
172.19.11.0/24 via 10.90.1.5 dev enp0s8
192.168.1.0/24 dev enp0s3 proto kernel scope link src 192.168.1.213
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
```

On Client2

```
[vagrant@Client2 network-scripts]$ sudo ip route add 172.19.11.0/24 via 10.3.90.110
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ ip r
172.19.11.0/24 via 10.3.90.110 dev eth1
172.19.21.0/24 via 172.16.1.1 dev eth2
[vagrant@Client2 network-scripts]$ ping 172.19.11.1 -c2
PING 172.19.11.1 (172.19.11.1) 56(84) bytes of data.
64 bytes from 172.19.11.1: icmp_seq=1 ttl=63 time=1.38 ms
64 bytes from 172.19.11.1: icmp_seq=2 ttl=63 time=1.44 ms

--- 172.19.11.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.387/1.415/1.443/0.028 ms
[vagrant@Client2 network-scripts]$
[vagrant@Client2 network-scripts]$ traceroute 172.19.11.1
traceroute to 172.19.11.1 (172.19.11.1), 30 hops max, 60 byte packets
 1  10.3.90.110 (10.3.90.110)  0.571 ms  0.606 ms  0.492 ms
 2  10.3.90.110 (10.3.90.110)  0.909 ms !X  0.792 ms !X  0.592 ms !X
```

5. Розрахувати спільну адресу та маску (summarizing) адрес 172.19.D+10.1 та 172.19.D+20.1, при чому префікс має бути максимально можливим. Видалити маршрути, встановлені на попередньому кроці та замінити їх об'єднаним маршрутом, який має проходити через Server1.

Run script ipv4_calculator.py of my development to known hosts addresses in binary:

```
[vagrant@Client2 IPv4_Calculator]$ python3 ipv4_calculator.py
Enter IP address of host: 172.19.11.1
Enter IP prefix: 24

IP address: 172.19.11.1/24

1. The network class - "B"
2. The address category - "Private Internet"
3. Subnetting attributes:

Number of octets:                  1st octet         2nd octet         3rd octet         4th octet
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Host Address (decimal):               172               19                11                1
Mask (decimal):                       255               255               255               0
Network Address (decimal):            172               19                11                0
First available host (decimal):       172               19                11                1
Last available host (decimal):        172               19                11                254
Broadcast address (decimal):          172               19                11                255
Host Address (binary):             10101100          00010011          00001011          00000001
Mask (binary):                     11111111          11111111          11111111          00000000
Network Address (binary):          10101100          00010011          00001011          00000000
First available host (binary):     10101100          00010011          00001011          00000001
Last available host (binary):      10101100          00010011          00001011          11111110
Broadcast address (binary):        10101100          00010011          00001011          11111111
Available number of addresses:                                32 - 24 = 8 , 2**8 - 2 = 254
[vagrant@Client2 IPv4_Calculator]$
[vagrant@Client2 IPv4_Calculator]$ python3 ipv4_calculator.py
Enter IP address of host: 172.19.21.1
Enter IP prefix: 24

IP address: 172.19.21.1/24

1. The network class - "B"
2. The address category - "Private Internet"
3. Subnetting attributes:

Number of octets:                  1st octet         2nd octet         3rd octet         4th octet
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Host Address (decimal):               172               19                21                1
Mask (decimal):                       255               255               255               0
Network Address (decimal):            172               19                21                0
First available host (decimal):       172               19                21                1
Last available host (decimal):        172               19                21                254
Broadcast address (decimal):          172               19                21                255
Host Address (binary):             10101100          00010011          00010101          00000001
Mask (binary):                     11111111          11111111          11111111          00000000
Network Address (binary):          10101100          00010011          00010101          00000000
First available host (binary):     10101100          00010011          00010101          00000001
Last available host (binary):      10101100          00010011          00010101          11111110
Broadcast address (binary):        10101100          00010011          00010101          11111111
Available number of addresses:                                32 - 24 = 8 , 2**8 - 2 = 254
```

Routes that can be summarized:

172.19.11.1 – 10101100.00010011.00001011.00000001 - first 19 bits are the same
172.19.21.1 – 10101100.00010011.00010101.00000001

Summarized route into one route:

172.19.0.0 – 10101100.00010011.00000000.00000000
255.255.224.0 – 11111111.11111111.11100000.00000000
/19

Delete routes dedicated in previous step:

On Server1

```
→  network-scripts sudo ip route del 172.19.11.0/24 via 10.90.1.5
[sudo] password for yevhen_yakymov:
→  network-scripts
→  network-scripts
→  network-scripts ip r
default via 192.168.1.1 dev enp0s3
10.3.90.0/24 dev enp0s9 proto kernel scope link src 10.3.90.110
10.90.1.0/24 dev enp0s8 proto kernel scope link src 10.90.1.55
169.254.0.0/16 dev enp0s3 scope link metric 1002
169.254.0.0/16 dev enp0s8 scope link metric 1003
169.254.0.0/16 dev enp0s9 scope link metric 1004
192.168.1.0/24 dev enp0s3 proto kernel scope link src 192.168.1.213
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
```

On Client2

```
[vagrant@Client2 IPv4_Calculator]$ sudo ip route del 172.19.11.0/24 via 10.3.90.110
[vagrant@Client2 IPv4_Calculator]$ ip r
default via 10.0.2.2 dev eth0 proto dhcp metric 100
default via 10.3.90.1 dev eth1 proto dhcp metric 101
default via 10.3.90.1 dev eth2 proto static metric 102
10.0.2.0/24 dev eth0 proto kernel scope link src 10.0.2.15 metric 100
10.3.90.0/24 dev eth1 proto kernel scope link src 10.3.90.5 metric 101
10.3.90.1 dev eth2 proto static scope link metric 102
10.90.1.0/24 via 10.3.90.110 dev eth1 proto static metric 101
172.16.1.0/24 dev eth2 proto kernel scope link src 172.16.1.2 metric 102
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
172.19.21.0/24 via 172.16.1.1 dev eth2
```

Add summary route from Client2 to 172.19.11.1 and .21.1 via Server1 and check:

On Server1

```
→  network-scripts sudo ip route add 172.19.0.0/19 via 10.90.1.5
→  network-scripts ip r
default via 192.168.1.1 dev enp0s3
10.3.90.0/24 dev enp0s9 proto kernel scope link src 10.3.90.110
10.90.1.0/24 dev enp0s8 proto kernel scope link src 10.90.1.55
169.254.0.0/16 dev enp0s3 scope link metric 1002
169.254.0.0/16 dev enp0s8 scope link metric 1003
169.254.0.0/16 dev enp0s9 scope link metric 1004
172.19.0.0/19 via 10.90.1.5 dev enp0s8
192.168.1.0/24 dev enp0s3 proto kernel scope link src 192.168.1.213
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
```

On Client2 and ping 172.19.11.1

```
[vagrant@Client2 IPv4_Calculator]$ sudo ip route add 172.19.0.0/19 via 10.3.90.110
[vagrant@Client2 IPv4_Calculator]$ ip r
172.19.0.0/19 via 10.3.90.110 dev eth1
172.19.21.0/24 via 172.16.1.1 dev eth2
[vagrant@Client2 IPv4_Calculator]$ ping 172.19.11.1 -c2
PING 172.19.11.1 (172.19.11.1) 56(84) bytes of data.
64 bytes from 172.19.11.1: icmp_seq=1 ttl=63 time=1.21 ms
64 bytes from 172.19.11.1: icmp_seq=2 ttl=63 time=1.58 ms

--- 172.19.11.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 1.218/1.401/1.584/0.183 ms
```

6. Налаштувати SSH сервіс таким чином, щоб Client1 та Client2 могли підключатись до Server1 та один до одного.

Verify sshd service installation and active ports:

On Server1

```
→  network-scripts netstat -at -n
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.122.1:53        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp        0      0 192.168.1.213:3306      0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:631                 :::*                    LISTEN
tcp6       0      0 ::1:25                  :::*                    LISTEN
tcp6       0      0 :::33060                :::*                    LISTEN
tcp6       0      0 :::111                  :::*                    LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
→  network-scripts systemctl is-active sshd
active
```

On Client1

```
vagrant@Client1:~/programming/python$ netstat -at -n
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 10.0.2.15:22            10.0.2.2:54198          ESTABLISHED
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::8080                 :::*                    LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
vagrant@Client1:~/programming/python$ systemctl is-active ssh
active
```

On Client2

```
[vagrant@Client2 IPv4_Calculator]$ netstat -at -n
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp        0      0 10.0.2.15:22            10.0.2.2:54223          ESTABLISHED
tcp6       0      0 :::111                  :::*                    LISTEN
tcp6       0      0 :::8080                 :::*                    LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:25                  :::*                    LISTEN
[vagrant@Client2 IPv4_Calculator]$
[vagrant@Client2 IPv4_Calculator]$ systemctl is-active sshd
active
```

Generates a pair of keys, of which one key is secret and the other public:

On Client1

```
vagrant@Client1:~/.ssh$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vagrant/.ssh/id_rsa): client1_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in client1_key
Your public key has been saved in client1_key.pub
The key fingerprint is:
SHA256:l8OIQTi6pbZ7adCqD0M6/nc88W3xt0ljqkGgi2qRRfE vagrant@Client1
The key's randomart image is:
+---[RSA 3072]----+
|     oo          |
|    oo.          |
|   ....E .       |
|  . .. o.o..     |
| . =o ..S =.     |
|o =o. .....o     |
|=. +.o..o ..o  + |
|.+o.=. + . o..+.o|
|.+=*. . . .....o.|
+----[SHA256]-----+
vagrant@Client1:~/.ssh$ ll
total 28
drwx------  2 vagrant vagrant 4096 Jan 24 21:03 ./
drwxr-xr-x 10 vagrant vagrant 4096 Jan  4 20:40 ../
-rw-------  1 vagrant vagrant 1184 Nov 11 15:06 authorized_keys
-rw-------  1 vagrant vagrant 2602 Jan 24 21:03 client1_key
-rw-r--r--  1 vagrant vagrant  569 Jan 24 21:03 client1_key.pub
-rw-r--r--  1 root    root    1679 Nov 11 15:19 id_rsa
-rw-r--r--  1 vagrant vagrant  444 Nov 11 15:19 known_hosts
vagrant@Client1:~/.ssh$
```

On Client2

```
[vagrant@Client2 .ssh]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vagrant/.ssh/id_rsa): client2_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in client2_key.
Your public key has been saved in client2_key.pub.
The key fingerprint is:
SHA256:htAolabNDvVCNjyJRjqtVvnIli6N7RQVDIBH1NN16sI vagrant@Client2
The key's randomart image is:
+---[RSA 2048]----+
|.===+= .. .      |
|.o+.#+o  o       |
|ooo@o*. .        |
| o=.Oo.o         |
|.. O oE S        |
|. * o  o         |
| o =             |
|  +              |
|   .             |
+----[SHA256]-----+
[vagrant@Client2 .ssh]$ ll
total 44
-rw-------. 1 vagrant vagrant 1184 Nov 11 15:02 authorized_keys
-rw-------. 1 vagrant vagrant 1675 Jan 24 21:01 client2_key
-rw-r--r--. 1 vagrant vagrant  397 Jan 24 21:01 client2_key.pub
```

Configure /etc/ssh/sshd_config file:

Client1

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile     .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no
```

Client2

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
# AuthorizedKeysFile    .ssh/authorized_keys

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no
PasswordAuthentication no
```

Copy public key to Server1 and establish ssh connection:

From Client1 to Server1

```
vagrant@Client1:~/.ssh$
vagrant@Client1:~/.ssh$ ssh-copy-id -i client1_key.pub yevhen_yakymov@10.90.1.55
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "client1_key.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the
                (if you think this is a mistake, you may want to use -f option)

vagrant@Client1:~/.ssh$
vagrant@Client1:~/.ssh$ ssh -i client1_key yevhen_yakymov@10.90.1.55
Last login: Tue Jan 24 20:07:32 2023
→  ~
→  ~ whoami
yevhen_yakymov
→  ~
→  ~ hostname
Server1
```

From Client2 to Server1

```
[vagrant@Client2 .ssh]$ ssh-copy-id -i client2_key.pub yevhen_yakymov@10.3.90.110
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "client2_key.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the
                (if you think this is a mistake, you may want to use -f option)

[vagrant@Client2 .ssh]$
[vagrant@Client2 .ssh]$ ssh -i client2_key yevhen_yakymov@10.3.90.110
Last login: Tue Jan 24 23:30:16 2023 from 10.90.1.5
→  ~
→  ~ whoami
yevhen_yakymov
→  ~
→  ~ hostaname
zsh: hostaname: command not found...
zsh: command not found: hostaname
→  ~ hostname
Server1
→  ~
→  ~ hostname -I
192.168.1.213 10.90.1.55 10.3.90.110 192.168.122.1
```

From Client1 to Client2

```
vagrant@Client1:~/.ssh$ ssh-copy-id -i client1_key.pub vagrant@172.16.1.2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "client1_key.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the
vagrant@Client1:~$ ssh -i client1_key vagrant@172.16.1.2
Warning: Identity file client1_key not accessible: No such file or directory.
Last login: Wed Jan 25 21:17:29 2023 from 172.16.1.1
[vagrant@Client2 ~]$
[vagrant@Client2 ~]$ whoami
vagrant
[vagrant@Client2 ~]$ hostname
Client2
[vagrant@Client2 ~]$ hostname -I
10.0.2.15 10.3.90.5 172.16.1.2 172.17.0.1
[vagrant@Client2 ~]$
```

7. Налаштуйте на Server1 firewall таким чином:

• Дозволено підключатись через SSH з Client_1 та заборонено з Client_2:

Deny ssh connection to Server1 from Client2

```
→  ~ hostname
Server1
→  ~
```

```
→  ~ sudo iptables -I INPUT 2 -s 10.3.90.5/24 -p tcp --dport ssh -j DROP
[sudo] password for yevhen_yakymov:
→  ~
→  ~
→  ~ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
DROP       tcp  --  10.3.90.0/24         anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
```

Check ssh connection to Server1 from Client2

```
[vagrant@Client2 .ssh]$ ssh -i client2_key yevhen_yakymov@10.3.90.110
ssh: connect to host 10.3.90.110 port 22: Connection timed out
```

(more different options)

```
→  ~ sudo iptables -A INPUT -p tcp -s 10.3.90.5/24 --dport ssh -j DROP
[sudo] password for yevhen_yakymov:
→  ~
→  ~ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
DROP       tcp  --  10.3.90.0/24         anywhere             tcp dpt:ssh
```

```
→  ~ sudo iptables -A INPUT -i enp0s9 -p tcp --dport 22 -j DROP
→  ~
→  ~ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
DROP       tcp  --  anywhere             anywhere             tcp dpt:ssh
```

Allow ssh connection to Server1 from Client1

```
➜  ~
➜  ~ hostname
Server1
➜  ~
➜  ~ sudo iptables -I INPUT 2 -s 10.90.1.5/24 -p tcp --dport ssh -j ACCEPT
➜  ~
➜  ~ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     tcp  --  10.90.1.0/24         anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
```

Check ssh connection to Server1 from Client1

```
vagrant@Client1:~/.ssh$
vagrant@Client1:~/.ssh$ ssh -i client1_key yevhen_yakymov@10.90.1.55
Last login: Thu Jan 26 16:53:23 2023 from 10.3.90.5
➜  ~
➜  ~ hostname
Server1
```

(more different options)

```
➜  ~ sudo iptables -A INPUT --source 10.90.1.5/24 -p tcp --dport ssh -j ACCEPT
➜  ~
➜  ~ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     tcp  --  10.90.1.0/24         anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
ACCEPT     tcp  --  10.90.1.0/24         anywhere             tcp dpt:ssh
```

```
➜  ~ sudo iptables -A INPUT -i enp0s8 -p tcp --dport 22 -j ACCEPT
➜  ~
➜  ~ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     tcp  --  10.90.1.0/24         anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
ACCEPT     tcp  --  10.90.1.0/24         anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
```

• З Client_2 на 172.19.D+10.1 ping проходив, а на 172.19.D+20.1 не проходив

Deny icmp to Client1(172.19.21.1) from Client2

```
[vagrant@Client2 .ssh]$ ping 172.19.21.1 -c3
PING 172.19.21.1 (172.19.21.1) 56(84) bytes of data.
64 bytes from 172.19.21.1: icmp_seq=1 ttl=64 time=0.592 ms
64 bytes from 172.19.21.1: icmp_seq=2 ttl=64 time=0.654 ms
64 bytes from 172.19.21.1: icmp_seq=3 ttl=64 time=0.641 ms

--- 172.19.21.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.592/0.629/0.654/0.026 ms
```

```
vagrant@Client1:~/.ssh$ sudo iptables -I INPUT 1 --source 172.16.1.0/24 -p icmp -j DROP
vagrant@Client1:~/.ssh$
vagrant@Client1:~/.ssh$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  172.16.1.0/24        anywhere
```

```
[vagrant@Client2 .ssh]$ ping 172.19.21.1 -c3
PING 172.19.21.1 (172.19.21.1) 56(84) bytes of data.

--- 172.19.21.1 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2000ms
```

Allow icmp to Client1(172.19.21.1) from Client2

```
vagrant@Client1:~/.ssh$ sudo iptables -I INPUT 2 --source 10.3.90.110/24 -p icmp -j ACCEPT
vagrant@Client1:~/.ssh$
vagrant@Client1:~/.ssh$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  172.16.1.0/24        anywhere
ACCEPT     icmp --  10.3.90.0/24         anywhere
```

```
[vagrant@Client2 .ssh]$
[vagrant@Client2 .ssh]$ ping 172.19.11.1 -c3
PING 172.19.11.1 (172.19.11.1) 56(84) bytes of data.
64 bytes from 172.19.11.1: icmp_seq=1 ttl=63 time=1.70 ms
64 bytes from 172.19.11.1: icmp_seq=2 ttl=63 time=1.27 ms
64 bytes from 172.19.11.1: icmp_seq=3 ttl=63 time=1.10 ms

--- 172.19.11.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 1.109/1.365/1.708/0.252 ms
[vagrant@Client2 .ssh]$
```