

High Availability Network Using Linux Bonding and Failover Routing

➤ **Problem statement :** Normally, if a server has a single NIC and it fails, the server loses network connectivity.

Goal : 1) Combine two NICs into one logical interface (bond).

2) Provide **fault tolerance** — if one NIC fails, traffic continues without interruption.

3) Demonstrate **zero downtime** in network connectivity.

➤ **Introduction :**

The High Availability Network Using Linux Bonding and Failover Routing project demonstrates how to configure multiple network interfaces in a Linux system to ensure continuous network connectivity. By combining two NICs into a single bonded interface (bond0) in active-backup mode, the system automatically redirects network traffic to the backup NIC if the primary NIC fails. This setup provides fault tolerance, redundancy, and high availability, ensuring zero downtime during network interface failure. The project utilizes Linux bonding drivers, NetworkManager (nmcli), and kernel modules to implement a resilient network configuration suitable for critical environments.

➤ **Tools and Technologies Used :**

- 1) **modprobe bonding :** The modprobe bonding command is used in Linux to load the bonding kernel module, which allows multiple network interfaces to be combined into a single bonded interface for redundancy and failover.
- 2) **nmcli:** nmcli is a command-line tool for NetworkManager in Linux, used to create, modify, and manage network connections, including configuring IP addresses, creating bond interfaces, and activating or deactivating NICs.
- 3) **ifenslave:** ifenslave is a Linux utility used to attach individual network interfaces as slaves to a bonded interface, enabling them to participate in link aggregation and failover.

➤ Implementation :

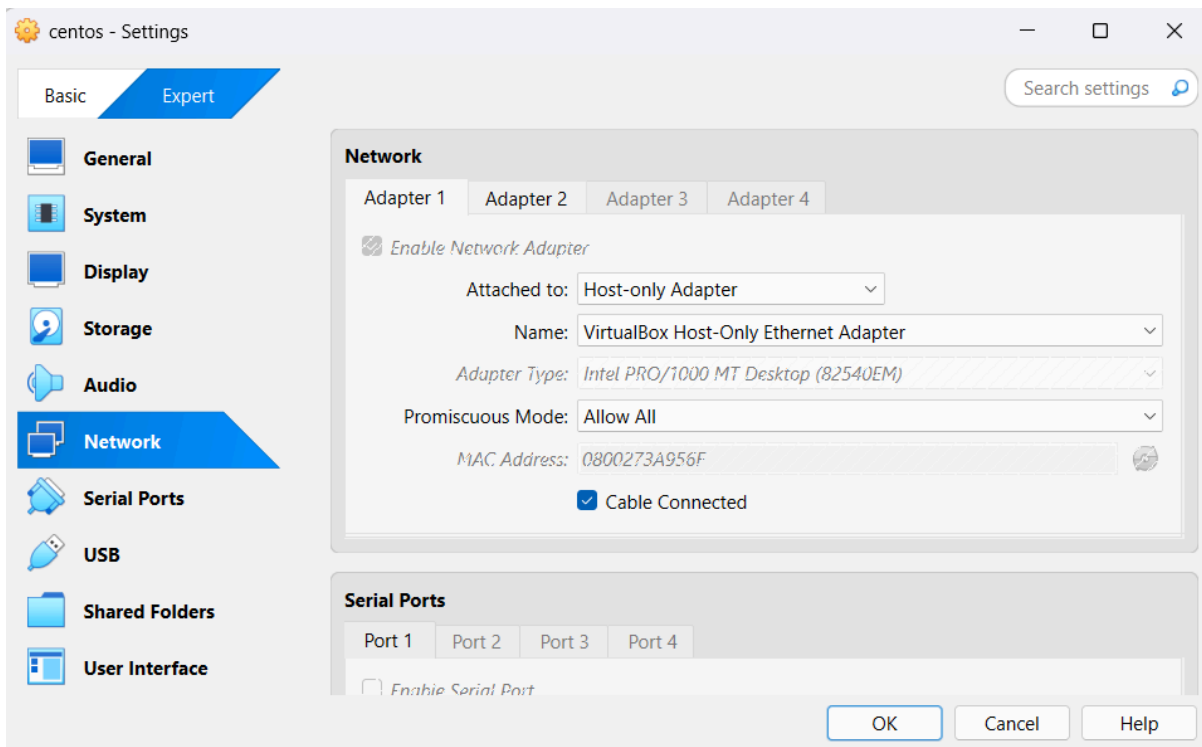
STEP 1 : Set up VirtualBox VM with 2 NICs :

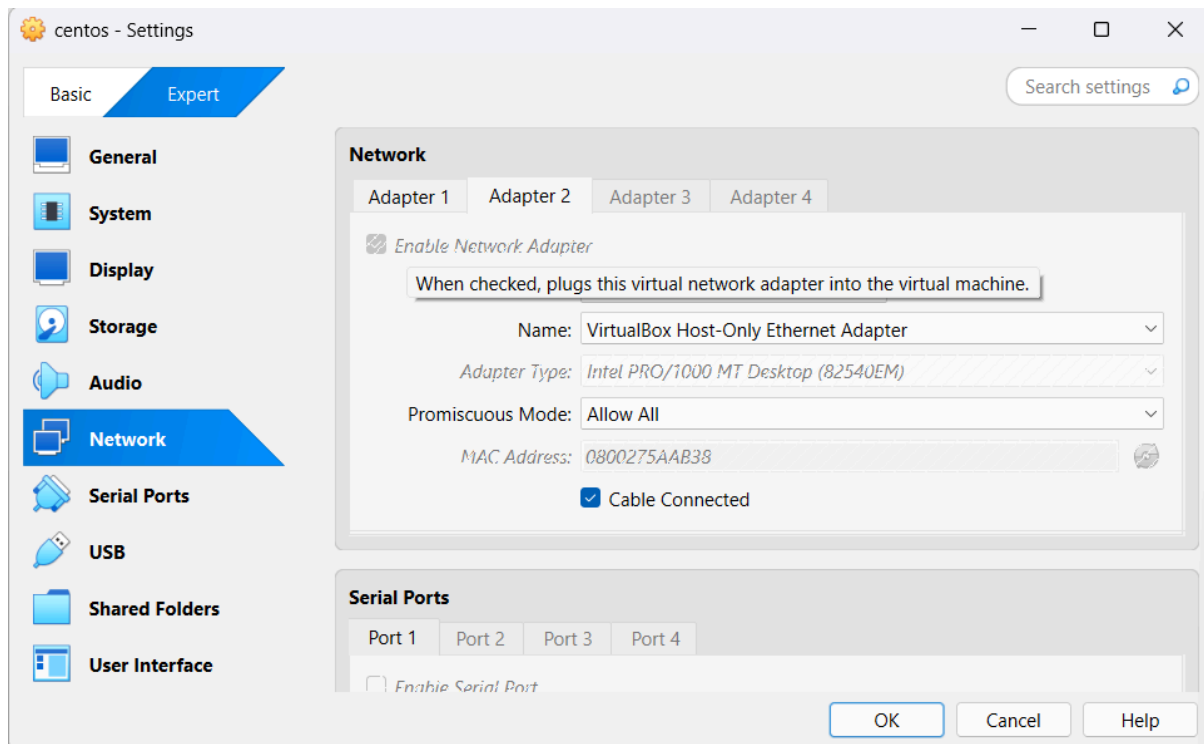
Open VirtualBox and select your CentOS 9 VM. → Click Settings → Network. →

1. Adapter 1 :
 - Check Enable Network Adapter
 - Attached to: Host-only Adapter
 - Promiscuous Mode : Allow all

2. Adapter 2 :
 - Check Enable Network Adapter
 - Attached to: Host-only Adapter
 - Promiscuous Mode : Allow all

Save And Start Virtual Machine





STEP 2 : Check Network Interfaces in Virtual Machine :

Open Terminal & Run

→ ip a :

enp0s3 → NIC 1

enp0s8 → NIC 2

STEP 3 : Assign Static IPs :

→ For Host-only network :

→ Gateway = 192.168.56.1

→ NIC1 = 192.168.56.101

→ NIC2 = 192.168.56.102

→ Bond IP = 192.168.56.150

→ **Assign IP to NICs Using Following Commands :**

1) For NIC 1 :

→ ip addr add 192.168.56.101/24 dev enp0s3

→ ip link set enp0s3 up

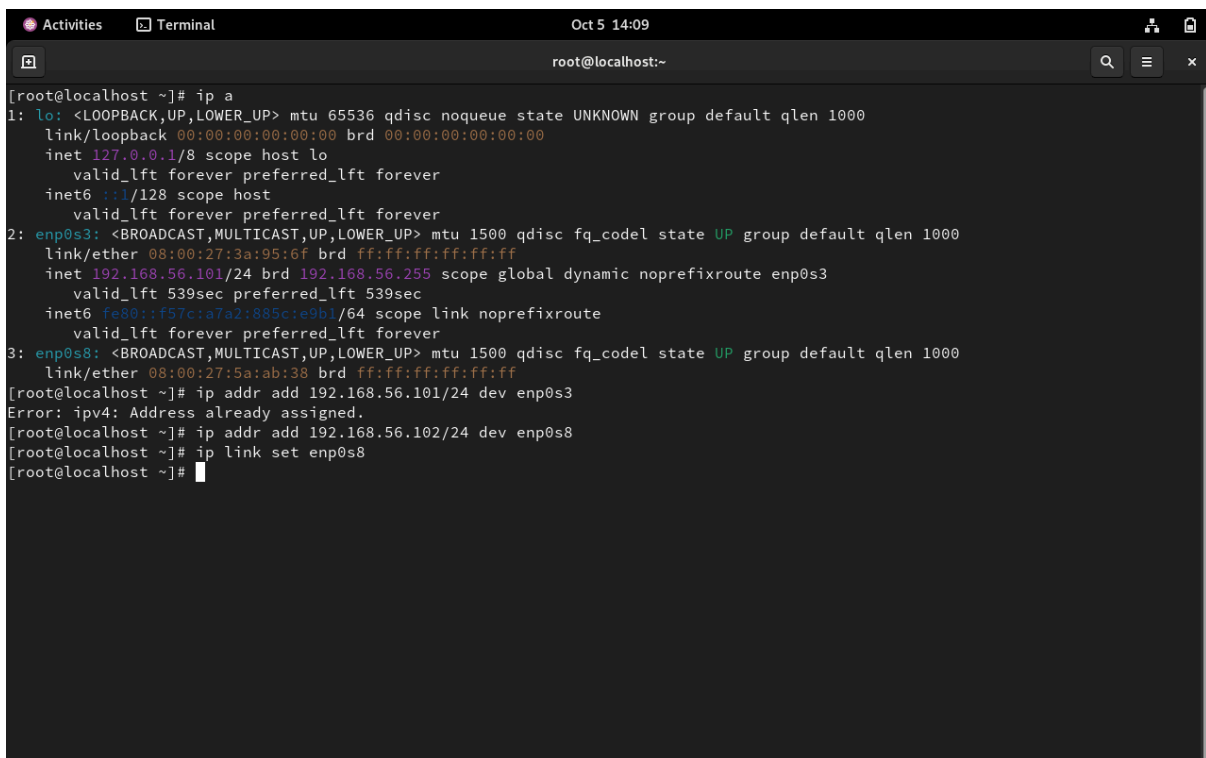
2) For NIC 2 :

→ ip addr add 192.168.56.102/24 dev enp0s8

→ ip link set enp0s8 up

→ **Test Connectivity using Following command :**

→ ping 192.168.56.1 -c 4



```
[root@localhost ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:3a:95:6f brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.101/24 brd 192.168.56.255 scope global dynamic noprefixroute enp0s3
        valid_lft 539sec preferred_lft 539sec
    inet6 fe80::f57c:a7a2:885c:e9b1/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:5a:ab:38 brd ff:ff:ff:ff:ff:ff
[root@localhost ~]# ip addr add 192.168.56.101/24 dev enp0s3
Error: ipv4: Address already assigned.
[root@localhost ~]# ip addr add 192.168.56.102/24 dev enp0s8
[root@localhost ~]# ip link set enp0s8
[root@localhost ~]#
```

STEP 4 : Install Bonding Tools :

CentOS 9 uses NetworkManager, so no need for ifenslave. Just load bonding module. Run following commands :

1. Load Kernel Module :

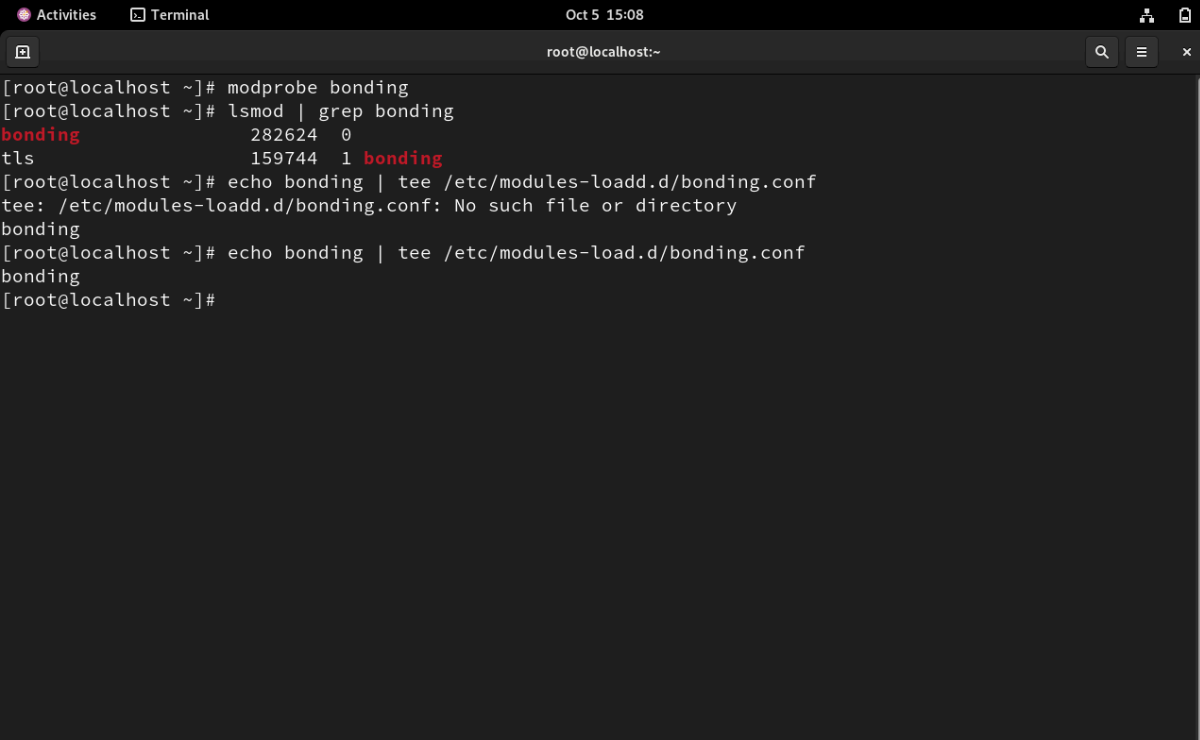
→ modprobe bonding

2. Check module loaded:

→ `lsmod | grep bonding`

3. Make bonding module load at boot:

→ `echo bonding | sudo tee /etc/modules-load.d/bonding.conf`




```
Oct 5 15:08
root@localhost:~
[root@localhost ~]# modprobe bonding
[root@localhost ~]# lsmod | grep bonding
bonding                282624  0
tls                    159744  1 bonding
[root@localhost ~]# echo bonding | tee /etc/modules-load.d/bonding.conf
tee: /etc/modules-load.d/bonding.conf: No such file or directory
bonding
[root@localhost ~]# echo bonding | tee /etc/modules-load.d/bonding.conf
bonding
[root@localhost ~]#
```

STEP 5 : Create the Bond Interface

1. Create bond0 in active-backup mode:

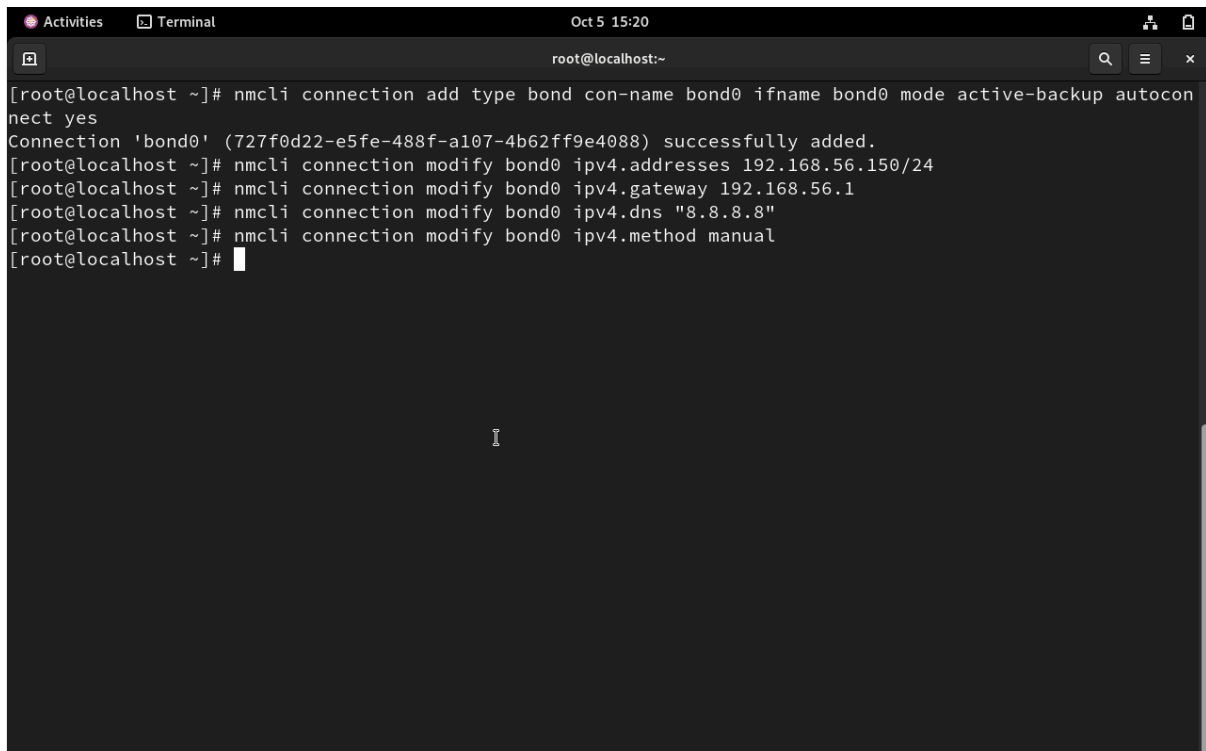
→ `nmcli connection add type bond con-name bond0 ifname bond0 mode active-backup autoconnect yes`



```
Oct 5 15:10
root@localhost:~
[root@localhost ~]# nmcli connection add type bond con-name bond0 ifname bond0 mode active-backup autoconnect yes
Connection 'bond0' (727f0d22-e5fe-488f-a107-4b62ff9e4088) successfully added.
[root@localhost ~]#
```

2. Assign IP to bond0:

- nmcli connection modify bond0 ipv4.addresses 192.168.56.150/24
- nmcli connection modify bond0 ipv4.gateway 192.168.56.1
- nmcli connection modify bond0 ipv4.dns "8.8.8.8"
- nmcli connection modify bond0 ipv4.method manual



```
[root@localhost ~]# nmcli connection add type bond con-name bond0 ifname bond0 mode active-backup autoconnect yes
Connection 'bond0' (727f0d22-e5fe-488f-a107-4b62ff9e4088) successfully added.
[root@localhost ~]# nmcli connection modify bond0 ipv4.addresses 192.168.56.150/24
[root@localhost ~]# nmcli connection modify bond0 ipv4.gateway 192.168.56.1
[root@localhost ~]# nmcli connection modify bond0 ipv4.dns "8.8.8.8"
[root@localhost ~]# nmcli connection modify bond0 ipv4.method manual
[root@localhost ~]#
```

STEP 6 : Add NICs to the Bond :

1. ADD

- nmcli connection add type ethernet con-name bond-slave-enp0s3 ifname enp0s3 master bond0 autoconnect yes
- nmcli connection add type ethernet con-name bond-slave-enp0s8 ifname enp0s8 master bond0 autoconnect yes

```
Oct 5 15:24
root@localhost:~
[root@localhost ~]# nmcli connection add type ethernet con-name bond-slave-enp0s3 ifname enp0s3 master bond0 autoconnect yes
Connection 'bond-slave-enp0s3' (fe0724ea-eb74-43c3-a301-83af67f06405) successfully added.
[root@localhost ~]# nmcli connection add type ethernet con-name bond-slave-enp0s8 ifname enp0s8 master bond0 autoconnect yes
Connection 'bond-slave-enp0s8' (233fdeb9-d493-4911-a4ba-531c4d110b51) successfully added.
[root@localhost ~]#
```

Disable IP On Slave :

→ nmcli connection modify bond-slave-enp0s3 ipv4.method disabled

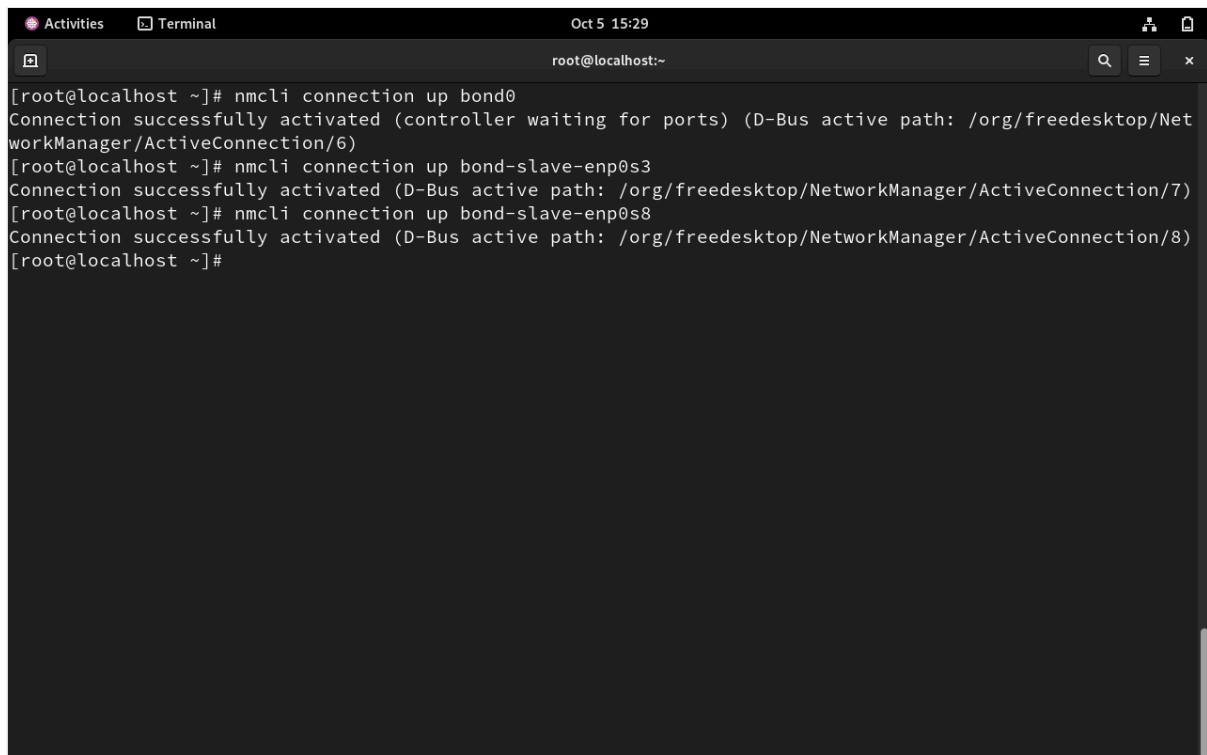
→ nmcli connection modify bond-slave-enp0s8 ipv4.method disabled

```
Oct 5 15:28
root@localhost:~
[root@localhost ~]# nmcli connection modify bond-slave-enp0s3 ipv4.method disabled
[root@localhost ~]# nmcli connection modify bond-slave-enp0s8 ipv4.method disabled
[root@localhost ~]#
```

STEP 7 : Bring Up the Bond :

1. Bring up using follwing commands :

- nmcli connection up bond0
- nmcli connection up bond-slave-enp0s3
- nmcli connection up bond-slave-enp0s



```
[root@localhost ~]# nmcli connection up bond0
Connection successfully activated (controller waiting for ports) (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/6)
[root@localhost ~]# nmcli connection up bond-slave-enp0s3
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/7)
[root@localhost ~]# nmcli connection up bond-slave-enp0s8
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/8)
[root@localhost ~]#
```

2. Check Status


```
Activities Terminal Oct 5 15:30 root@localhost:~
[root@localhost ~]# nmcli device status
DEVICE    TYPE      STATE      CONNECTION
bond0     bond      connected  bond0
enp0s3    ethernet  connected  bond-slave-enp0s3
enp0s8    ethernet  connected  bond-slave-enp0s8
lo        loopback  connected (externally)  lo
[root@localhost ~]#
```

STEP 8 : Verify Bonding

```
Activities Terminal Oct 5 15:31 root@localhost:~
[root@localhost ~]# cat /proc/net/bonding/bond0
Ethernet Channel Bonding Driver: v5.14.0-612.el9.x86_64

Bonding Mode: fault-tolerance (active-backup)
Primary Slave: None
Currently Active Slave: enp0s3
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0
Peer Notification Delay (ms): 0

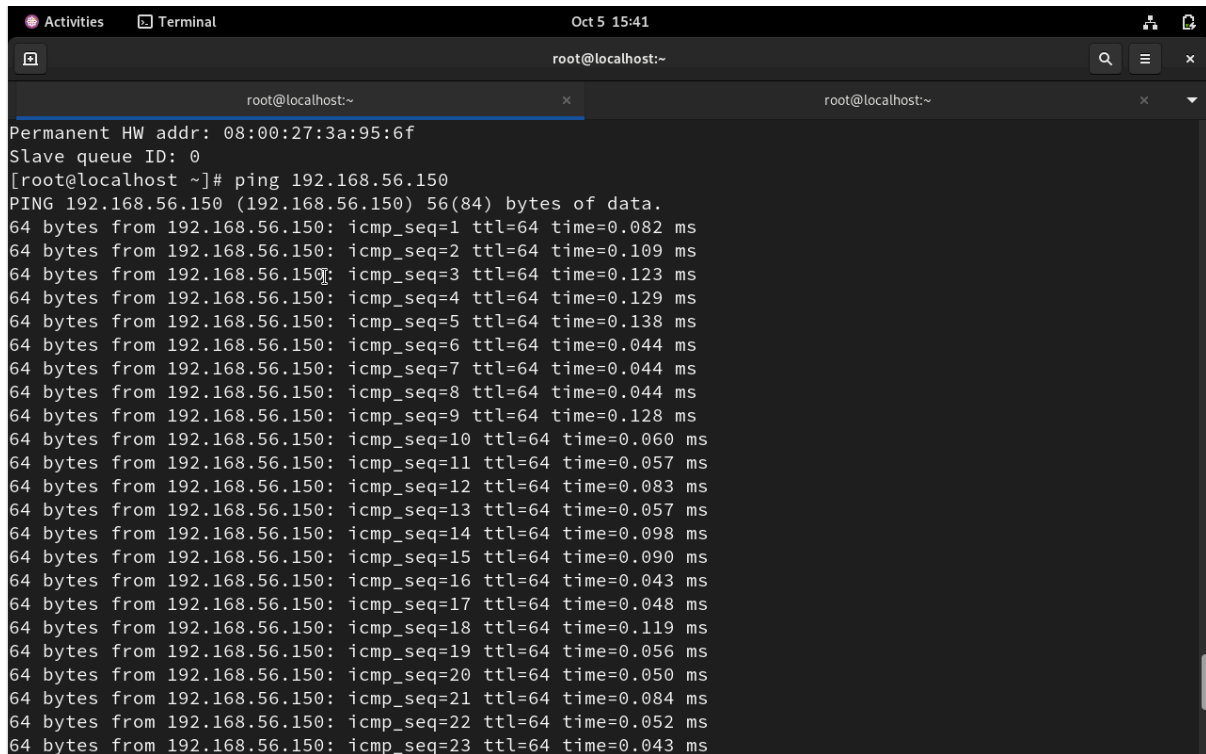
Slave Interface: enp0s3
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:3a:95:6f
Slave queue ID: 0

Slave Interface: enp0s8
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:5a:ab:38
Slave queue ID: 0
[root@localhost ~]#
```

STEP 9 : Test

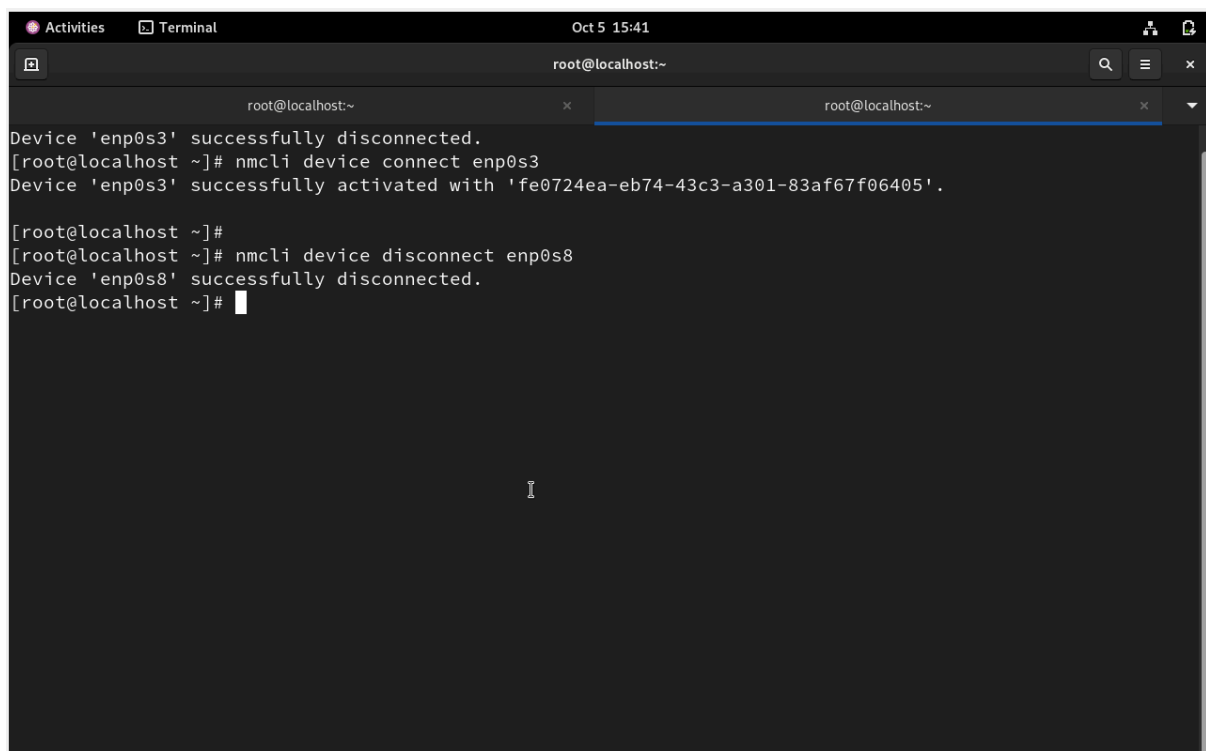
1. Start a continuous ping on one terminal :

→ ping 192.168.56.150



```
Permanent HW addr: 08:00:27:3a:95:6f
Slave queue ID: 0
[root@localhost ~]# ping 192.168.56.150
PING 192.168.56.150 (192.168.56.150) 56(84) bytes of data:
64 bytes from 192.168.56.150: icmp_seq=1 ttl=64 time=0.082 ms
64 bytes from 192.168.56.150: icmp_seq=2 ttl=64 time=0.109 ms
64 bytes from 192.168.56.150: icmp_seq=3 ttl=64 time=0.123 ms
64 bytes from 192.168.56.150: icmp_seq=4 ttl=64 time=0.129 ms
64 bytes from 192.168.56.150: icmp_seq=5 ttl=64 time=0.138 ms
64 bytes from 192.168.56.150: icmp_seq=6 ttl=64 time=0.044 ms
64 bytes from 192.168.56.150: icmp_seq=7 ttl=64 time=0.044 ms
64 bytes from 192.168.56.150: icmp_seq=8 ttl=64 time=0.044 ms
64 bytes from 192.168.56.150: icmp_seq=9 ttl=64 time=0.128 ms
64 bytes from 192.168.56.150: icmp_seq=10 ttl=64 time=0.060 ms
64 bytes from 192.168.56.150: icmp_seq=11 ttl=64 time=0.057 ms
64 bytes from 192.168.56.150: icmp_seq=12 ttl=64 time=0.083 ms
64 bytes from 192.168.56.150: icmp_seq=13 ttl=64 time=0.057 ms
64 bytes from 192.168.56.150: icmp_seq=14 ttl=64 time=0.098 ms
64 bytes from 192.168.56.150: icmp_seq=15 ttl=64 time=0.090 ms
64 bytes from 192.168.56.150: icmp_seq=16 ttl=64 time=0.043 ms
64 bytes from 192.168.56.150: icmp_seq=17 ttl=64 time=0.048 ms
64 bytes from 192.168.56.150: icmp_seq=18 ttl=64 time=0.119 ms
64 bytes from 192.168.56.150: icmp_seq=19 ttl=64 time=0.056 ms
64 bytes from 192.168.56.150: icmp_seq=20 ttl=64 time=0.050 ms
64 bytes from 192.168.56.150: icmp_seq=21 ttl=64 time=0.084 ms
64 bytes from 192.168.56.150: icmp_seq=22 ttl=64 time=0.052 ms
64 bytes from 192.168.56.150: icmp_seq=23 ttl=64 time=0.043 ms
```

2. In another terminal, bring down the active NIC



```
Device 'enp0s3' successfully disconnected.
[root@localhost ~]# nmcli device connect enp0s3
Device 'enp0s3' successfully activated with 'fe0724ea-eb74-43c3-a301-83af67f06405'.

[root@localhost ~]#
[root@localhost ~]# nmcli device disconnect enp0s8
Device 'enp0s8' successfully disconnected.
[root@localhost ~]#
```

OUTPUT / RESULT :

During testing, a continuous ping to the bonded interface IP (192.168.56.150) was started in one terminal. When the active NIC `enp0s8` was brought down in another terminal, the ping continued without interruption, demonstrating that network traffic automatically switched to the backup NIC (`enp0s3`), thereby verifying high availability and failover functionality of the bonded interface.