***Network Interface Card (NIC)***

A network interface card (NIC) is a hardware component that connects computers over a network. This is a circuit board attached to your computer that provides a dedicated network connection for your computer. Also called a network interface controller, network adapter, or LAN adapter. A NIC provides a computer with a dedicated, permanent connection to a network.

***NIC Bonding***

In the Linux operating system, NIC bonding refers to the process of combining multiple network interfaces into a single logical “connected” interface. That is, two or more NICs are combined and connected to function as one. Note that one of the requirements for configuring bonding is to have a network switch that supports EtherChannel (almost all switches do).

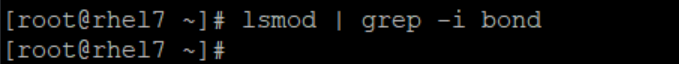
In a bond, the kernel handles all operations exclusively. You can create bonds on different types of devices, such as Ethernet devices or VLANs.

***NICs Bonding Modes***

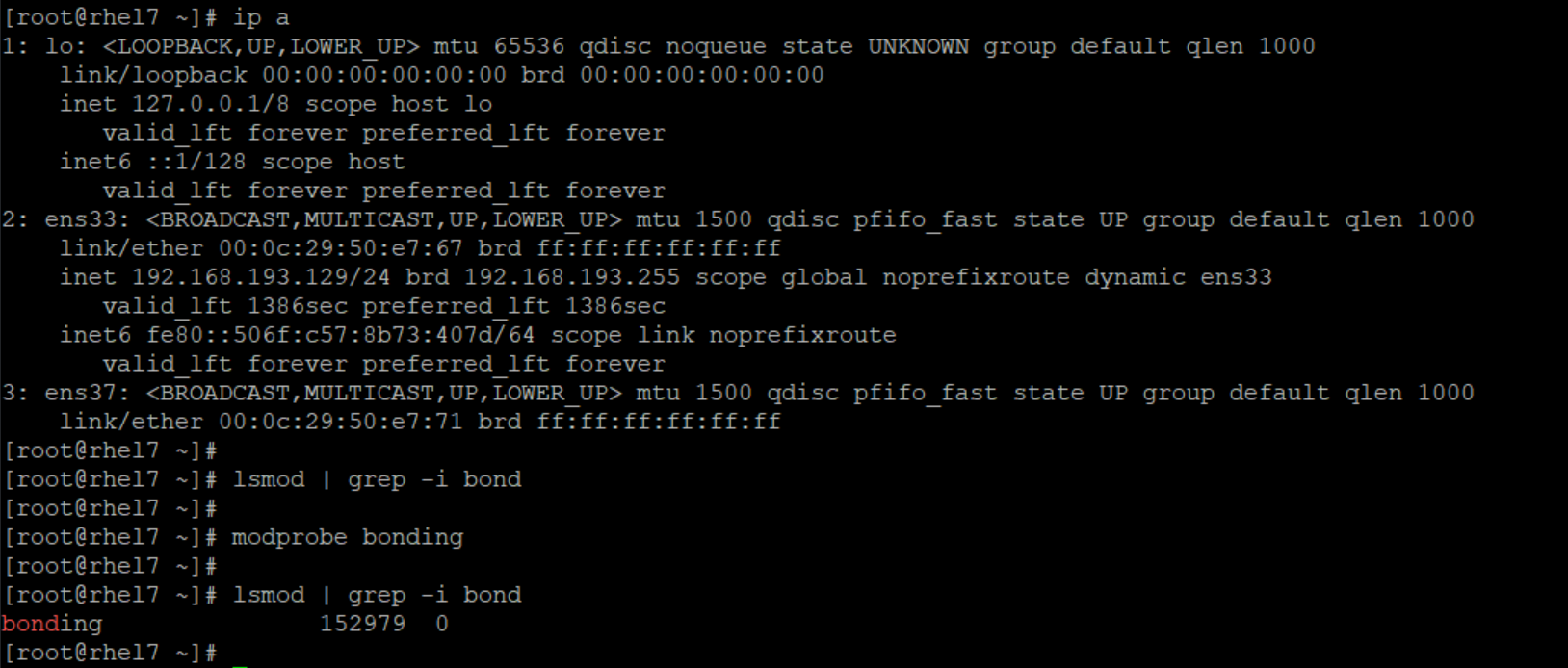
1. Mode=0 (Round-robin) – It’s a kind of default mode. In round-robin mode, packets are sent and received in order from the first available slave to his last slave.
2. Mode=1 (Active-backup) – So that Only one slave is active while another is sleeping. This standby NIC becomes active when the active NIC fails.
3. Mode=2 XOR(exclusive-or) – By this mode, XOR the source and destination MAC addresses are multiplied by the modulo port interface number.
4. Mode=3 (Broadcast) – So in this mode, all packets are sent to all slave interfaces at the expense of resource usage. Usually used for a specific purpose. B. The financial industry requires a highly reliable network etc.
5. Mode=4 (802.3ad) - Requires LACP-negotiated EtherChannel enabled. This mode is known as a Dynamic Link Aggregation mode that has it created aggregation groups having same speed. It requires a switch that supports IEEE 802.3ad dynamic link. The slave selection for outgoing traffic is done based on a transmit hashing method. This may be changed from the XOR method via the xmit\_hash\_policy option.
6. Mode=5 (balance-tlb) - No configuration required on the switch. This mode is called Adaptive transmit load balancing. The outgoing traffic is distributed based on the current load on each slave and the incoming traffic is received by the current slave. If the incoming traffic fails, the failed receiving slave is replaced by the MAC address of another slave. This mode does not require any special switch support
7. Mode=6 (balance-alb) - No configuration required on the switch. This mode is called adaptive load balancing

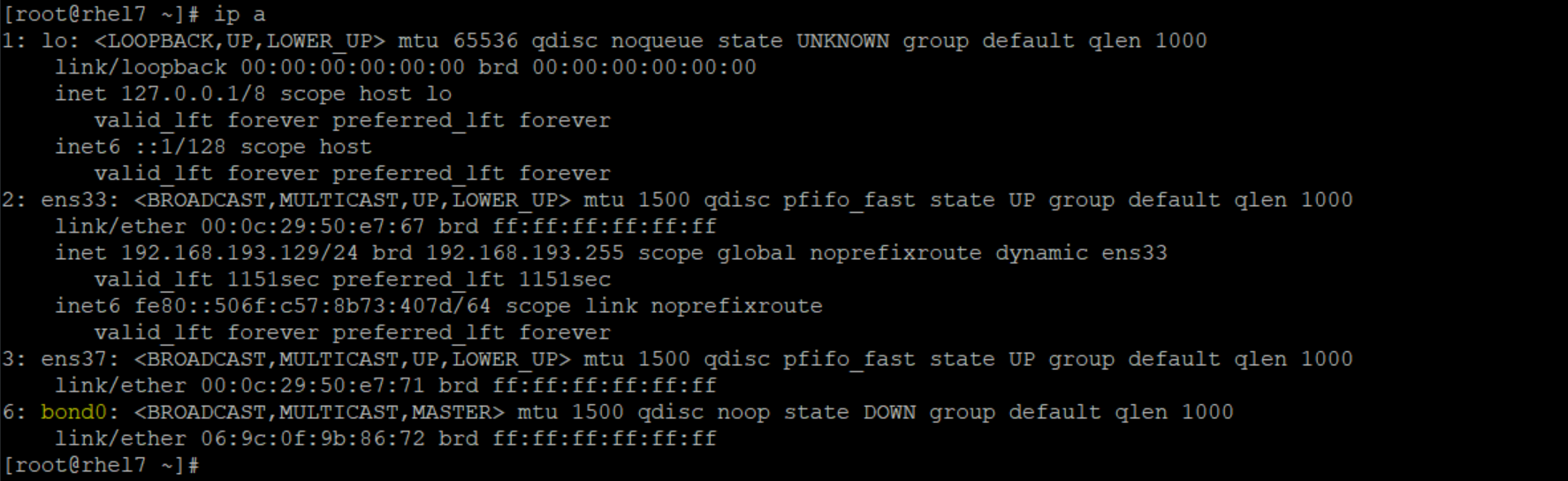
***Note:*** One NICs Bonding interface can only specify one mode at a time.

We need the Linux kernel to manage network bonding. So, let’s check if the bonding kernel module is available with ***lsmod*** command:

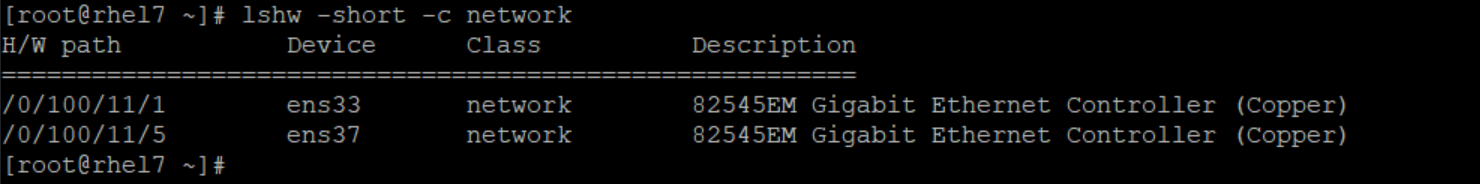


If bonding module not loaded then we need to load it with ***modprobe*** command and verify:

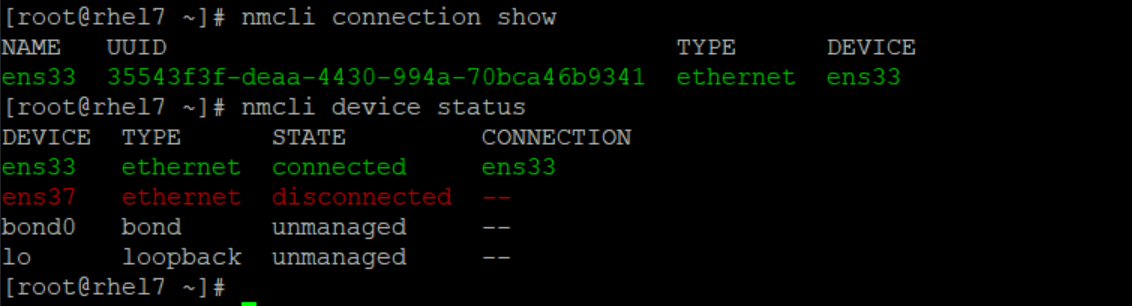




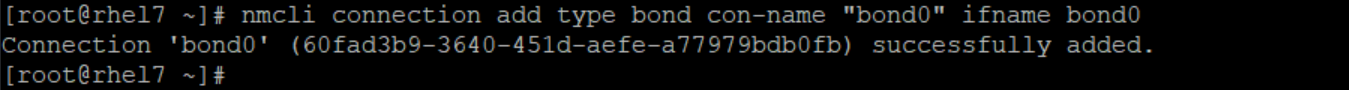
Now, let’s check network interfaces with ***lshw*** command:



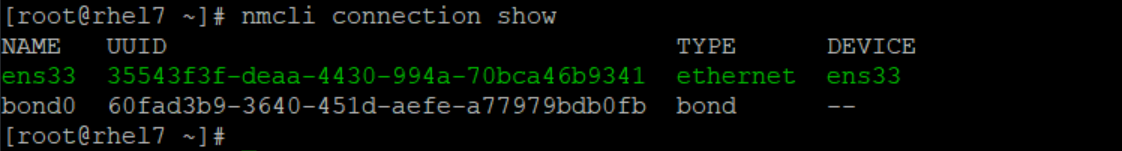
Now check all the active and inactive connections & device status using ***nmcli*** command



**Now add the bonding connection using nmcli utility**

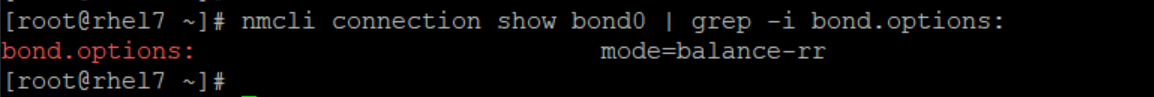


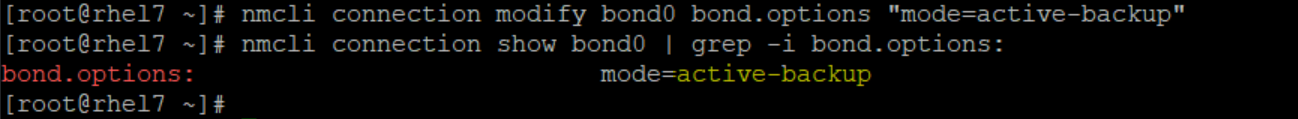
**Now verify bonding connection**



**Verify & Change the Bond Mode**

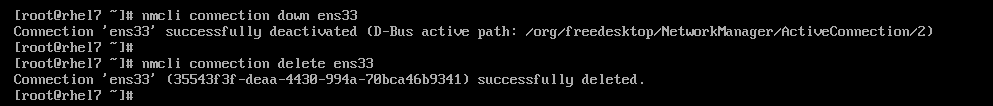
Change the bond mode used by the bond interface from Round Robin to Active Backup using the bond.options "mode=active-backup" command option



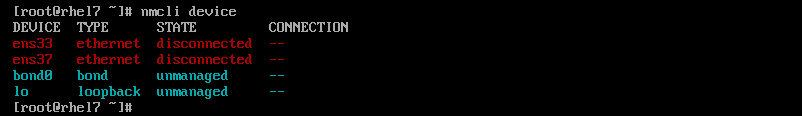


**Add Interface Links to the Bond Interface**

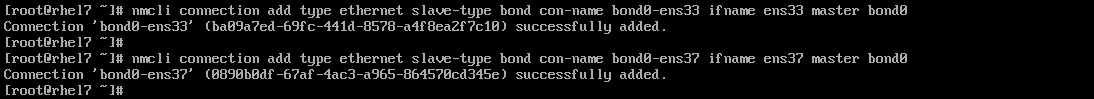
1. **First check the active connections and bring them down and delete the connections**



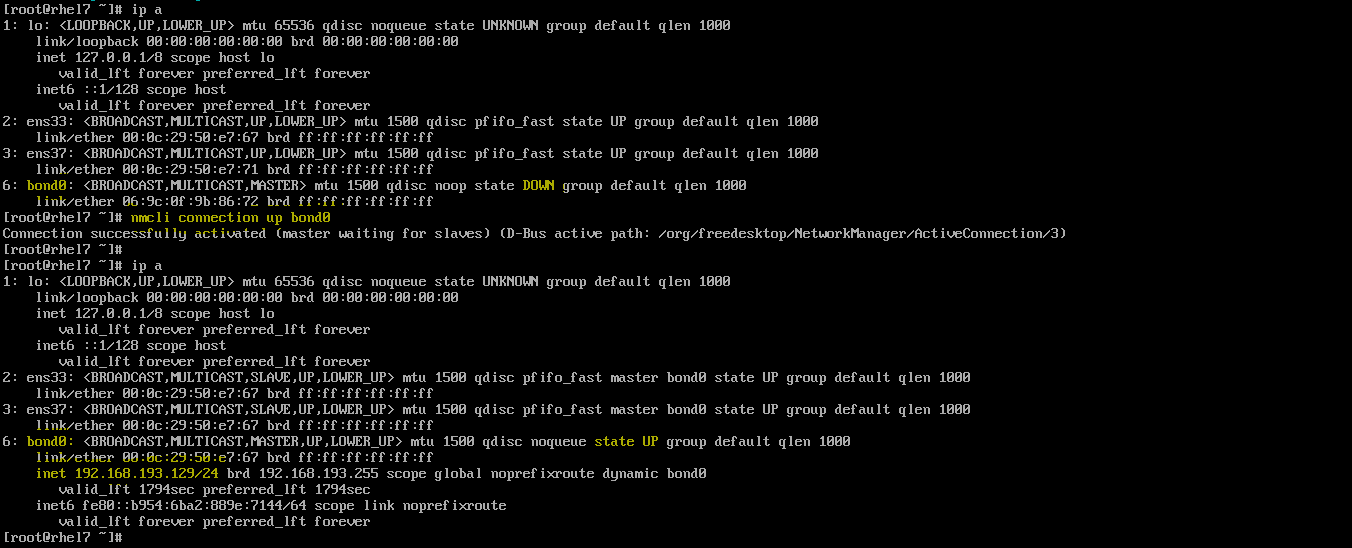
1. **View the network device interface information on the system.**



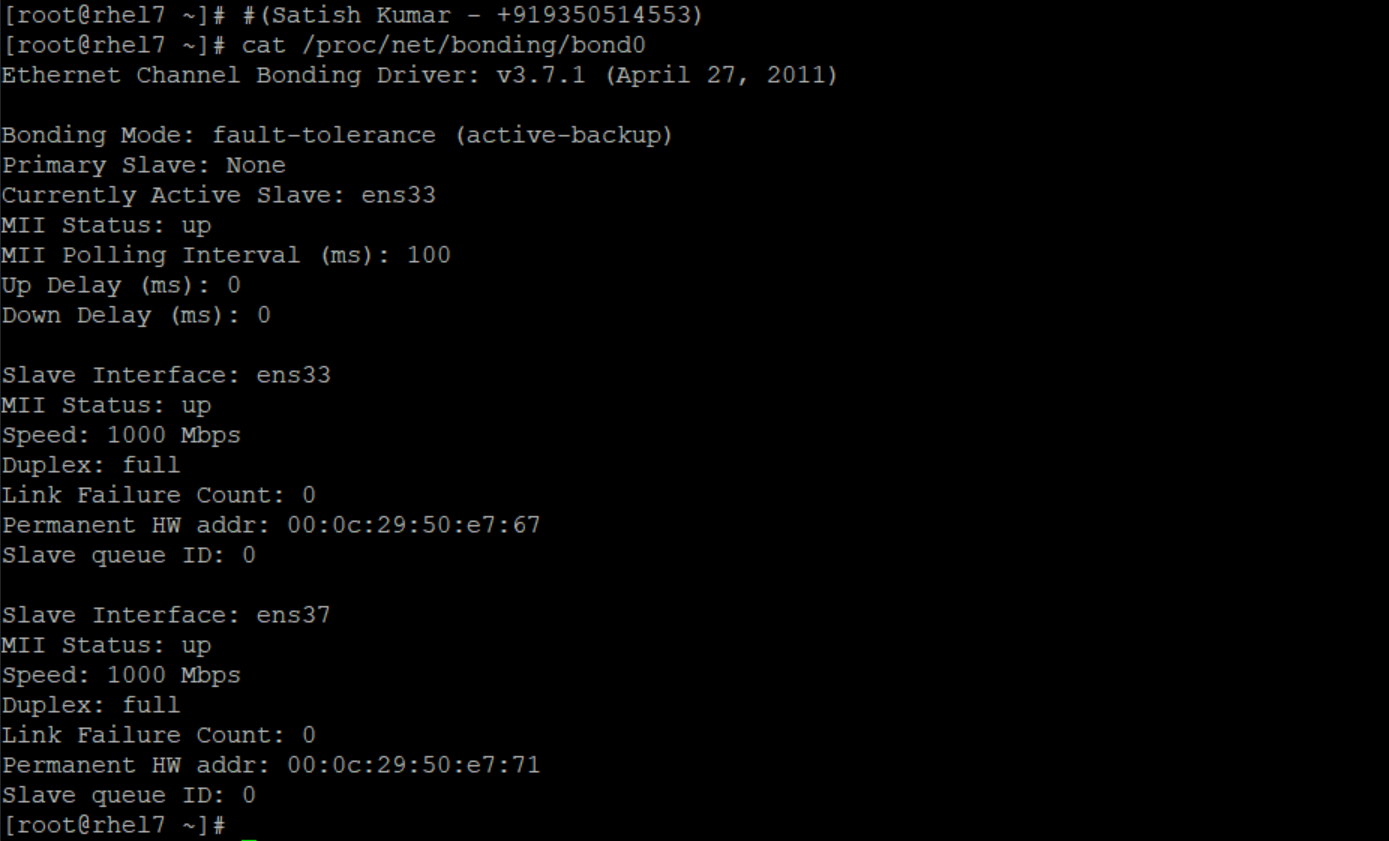
1. **Add the interface ens33 and ens37 as a links for the bond interface group**



1. **Configure IP Address to the Bond interface**
2. By default, the bond is configured to use DHCP. IP address assignment, gateway and network resolution information can be configured to the logical bond interface.
3. Use the command ***ip addr*** to verify that bond0 reports its status as UP and if is not up then bring up the connection using nmcli utility.

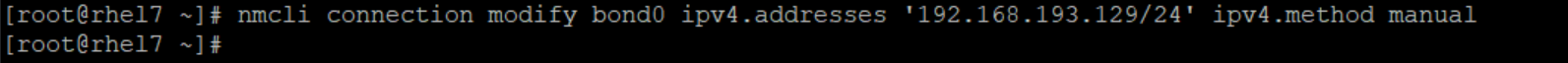


1. View the bond configuration information.

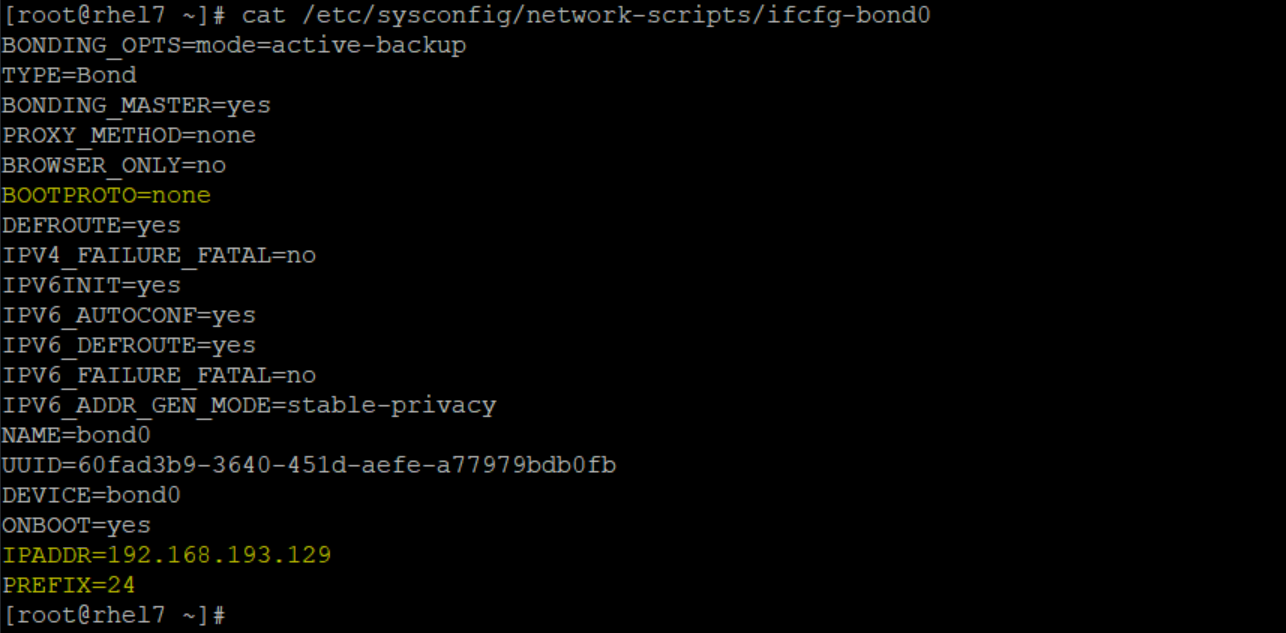


Note: Instead of dynamic IP assignment (via DHCP) IP Addresses can be assigned manually using below command.

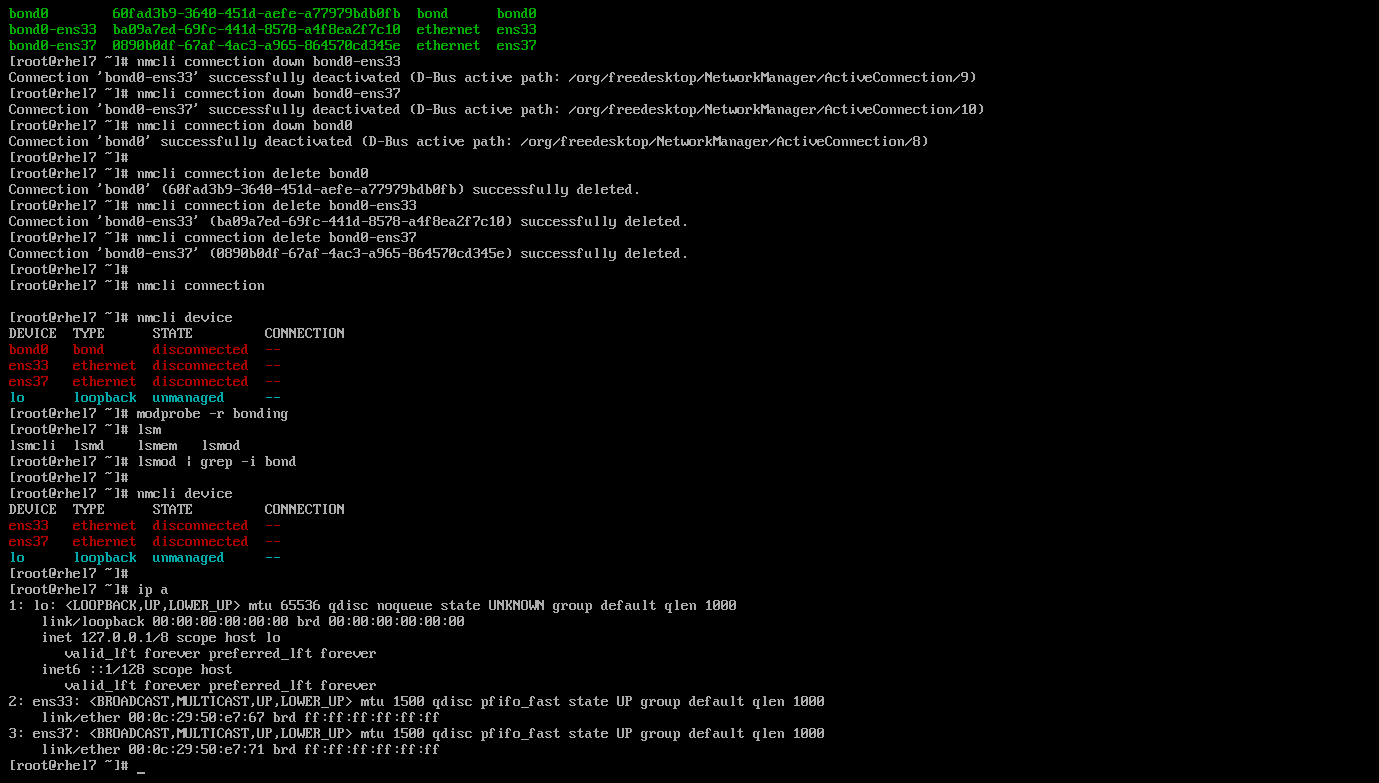
# nmcli connection modify “<connection-name>” ipv4.addresses '<IP-Address>/<subnet>' ipv4.method manual



**IP assignment via file method**



**Rollback Procedure:**



***NIC Teaming***

NIC teaming is the aggregation or bonding of two or more network links into a single logical link to provide redundancy and high availability. The logical interface/link is known as a team interface.

In the event that the active physical link goes down, one of the backup or reserved links automatically kicks and ensures an uninterrupted connection to the server.

***Teaming Terminologies***

Before we roll our sleeves, it’s crucial to familiarize yourself with the following terminologies:

**Teamd** – This is the NIC teaming daemon that uses the libteam library to communicate with team devices via the Linux kernel.

**Teamdctl** – This is a utility that allows users to control an instance of teamd. You can check and change the port status, as well as switch between backup and active states.

**Runner** – These are units of code written in JSON and are used for the implementation of various NIC teaming concepts.

Examples of runner modes include round robbin, load balancing, broadcast, and active backup.

For this guide, we will configure NIC teaming using the active-backup mode. This is where one link remains active while the rest are on standby and reserved as backup links in case the active link goes down.

**Step 1: Install the teamd Daemon in RHEL7**

Teamd is the daemon that is responsible for creating a network team that will act as the logical interface during runtime. By default, it comes installed with RHEL 8. But if, for whatever reason, it’s not installed, execute the following dnf/yum command to install it

#yum install teamd

**Step 2: Configure NIC Teaming**

To configure NIC teaming we will use the handy nmcli tool that can be used for the management of NetworkManager service.

In my system, I have 2 NIC cards that I’m going to bond or combine to create a logical team interface: ens33 and ens37

This may be different in your case.

To confirm the active network interfaces run:

# nmcli device status

# nmcli connection show

***Note:*** Active interfaces cannot be added to a team.

Use the ip command to bring down the component ports:

# ip link set <interface-name> down

# ip link set <interface-name> down

To create a network teaming link or interface, which will be our logical link, we are going to delete the existing network interfaces.

Thereafter we will create slave interfaces using the deleted interfaces and then associate them with the teaming link.

#nmcli connection delete <connection-name> or <connection-UUID>

#nmcli device status

Next, we are going to create a team interface called team0 in active-backup runner mode. As earlier stated, the active backup runner mode uses one active interface and reserves the others for redundancy in case the active link goes down.

# nmcli connection add type team con-name team0 ifname team0 config '{"runner": {"name": "activebackup"}}'

To view the attributes assigned to the team0 interface run the command:

#nmcli connection show team0

#nmcli connection show

Next, configure IP address for the team0 interface as shown using the nmcli command.

Be sure to assign the IP’s according to your network’s subnet & IP addressing scheme.

# nmcli con mod team0 ipv4.addresses <IP-Address>/<netmask> or # ip addr add <IP-Address>/<netmask> dev team0

# nmcli con mod team0 ipv4.gateway <Gateway-IP>

# nmcli con mod team0 ipv4.dns <DNS-IP>

# nmcli con mod team0 ipv4.method manual

# nmcli con mod team0 connection.autoconnect yes

**Thereafter, create slave links and associate the slaves to the team link:**

# nmcli con add type team-slave con-name team0-slave0 ifname <interface-name> master team0

# nmcli con add type team-slave con-name team0-slave1 ifname <interface-name> master team0

**Check the status of the links again, and you’ll notice that the slave links are now active.**

#nmcli connection show

Next, deactivate and activate the team link. This activates the connection between the slave links and the team link.

# nmcli connection down team0 && nmcli connection up team0

Next, verify the state of the team link connection as shown.

# ip addr show dev team0

To retrieve additional details about the team link, run the command:

#teamdctl team0 state

**Step 3: Testing Network Teaming Redundancy**

#nmcli device disconnect <interface-name>

#teamdctl team0 state

***Note:*** If you wish to delete the teaming interface/link and revert to default network settings, first bring down the teaming link:

#nmcli connection down team0

#nmcli connection delete team0-slave0 team0-slave1

#nmcli connection delete team0

***Removing a Team***

To terminate, or kill, an instance of the team daemon use below command:

# teamd -t team0 -k

At this point, all the interfaces are down and your server is not reachable.

To activate your network interfaces and regain connectivity, run the commands:

#ifconfig <interface-name> up

#ifconfig <interface-name> up

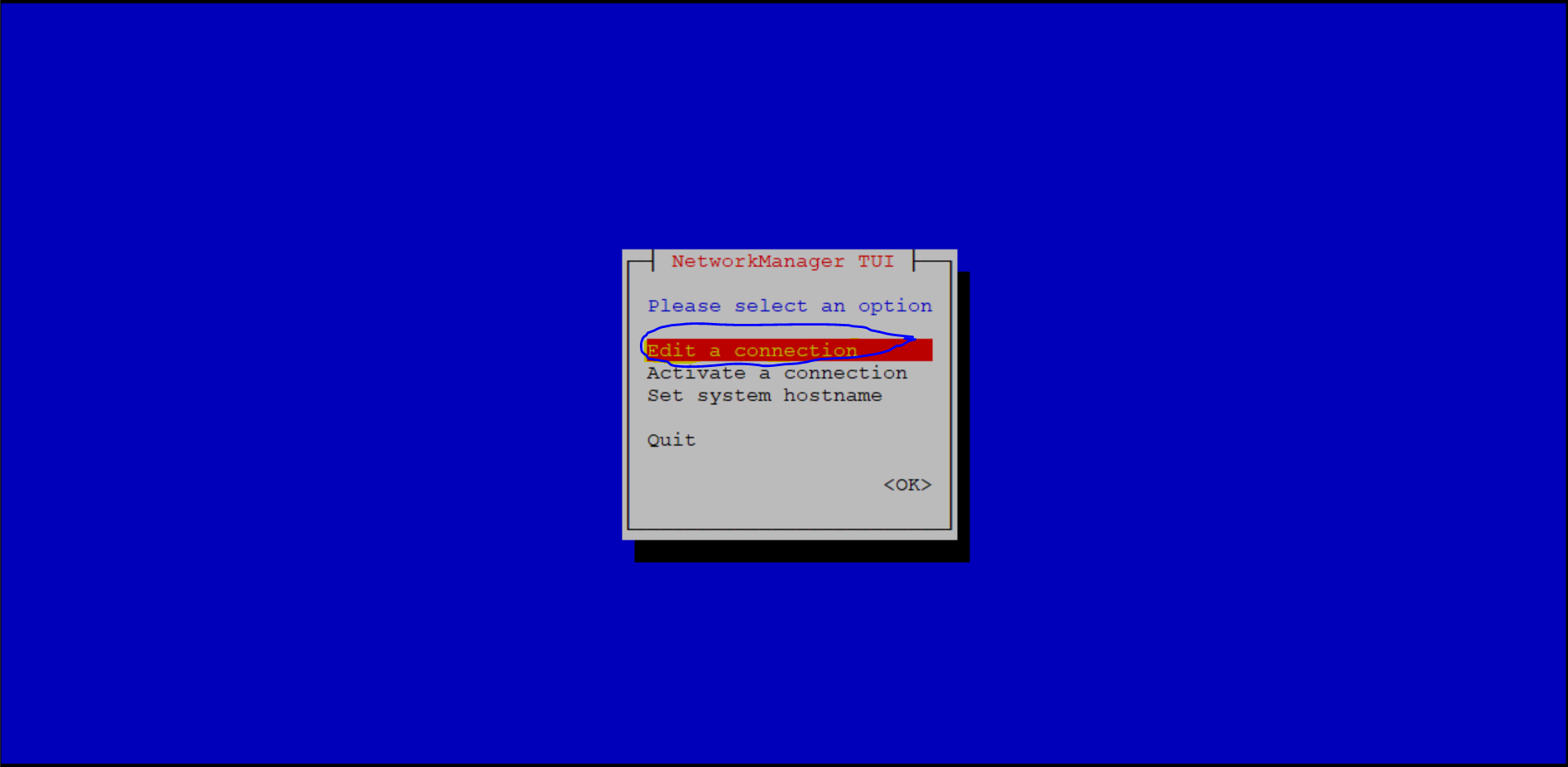
#systemctl restart NetworkManager

***NIC Teaming via NMTUI Utility***

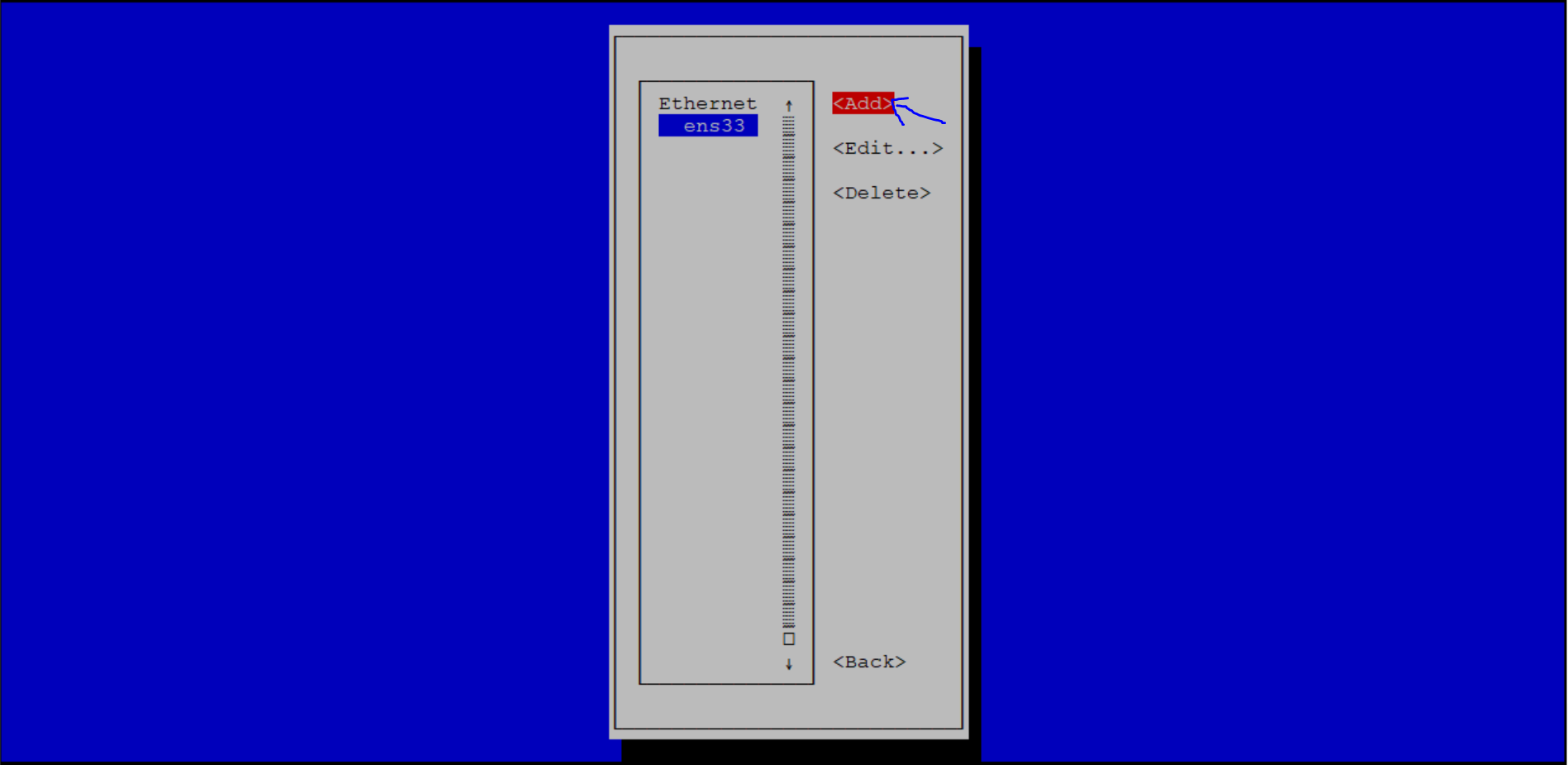
1. Execute below ***nmtui*** command from root account



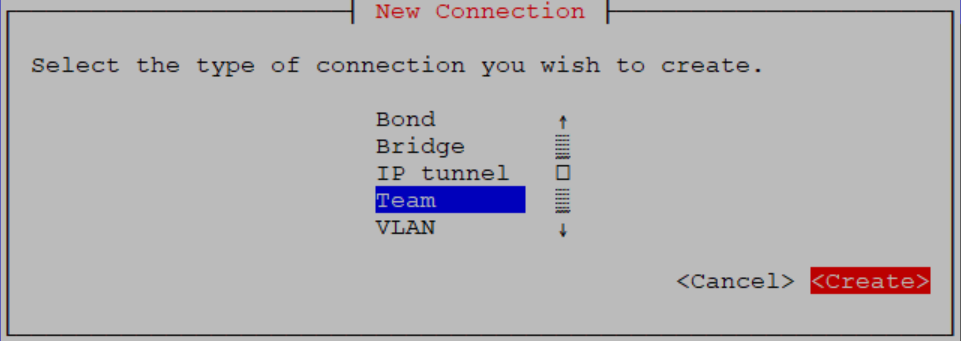
1. Below screen will be prompted after executing the ***nmtui*** command and select “Edit a connection” and press enter.



1. Select the <Add> button as shown in below screen



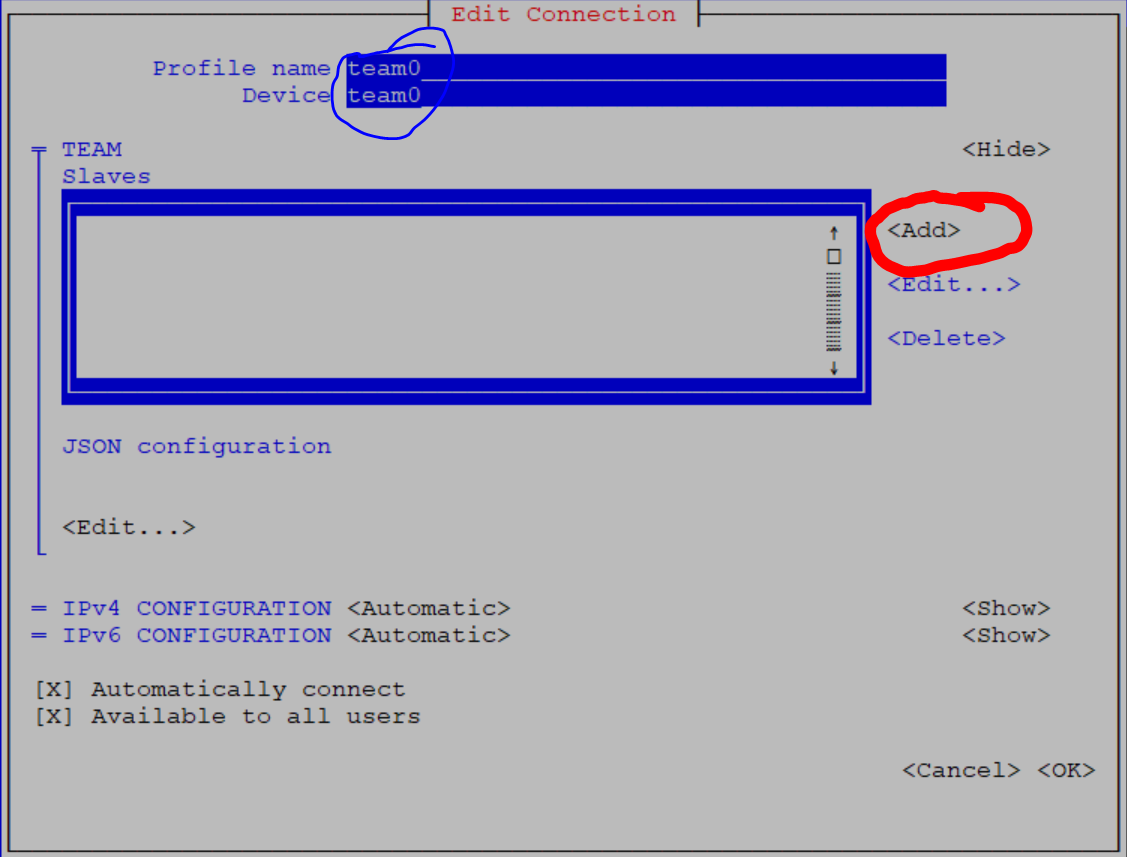
1. Select the type of connection “Team” and click on “Create” as shown in below prompt



1. Write the Profile name and Device name as per the requirement as shown in below snap

Ex: Profile name/Connection name: team0

Device name/Logical NIC card name: team0



1. Create the slaves and make them the member of logical NIC card which is team0. To create the slaves click on “Add” button