Enterprise Linux 7 Update

NetworkManager Command Line Tools

- If you want to leave NetworkManager "in control" of the network, but cannot use the GUI, then use *nmcli* and the (easier) *nmtui* which we'll look at first.

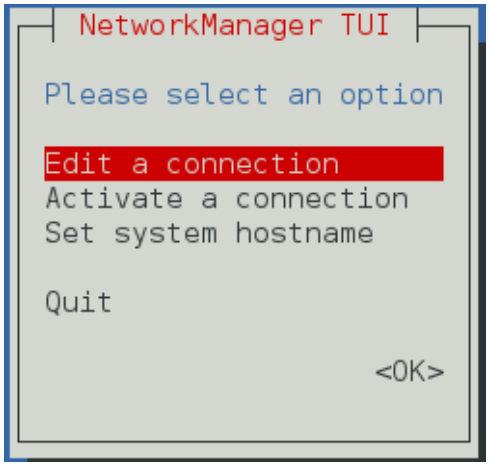
nmtui

NetworkManager is not running.

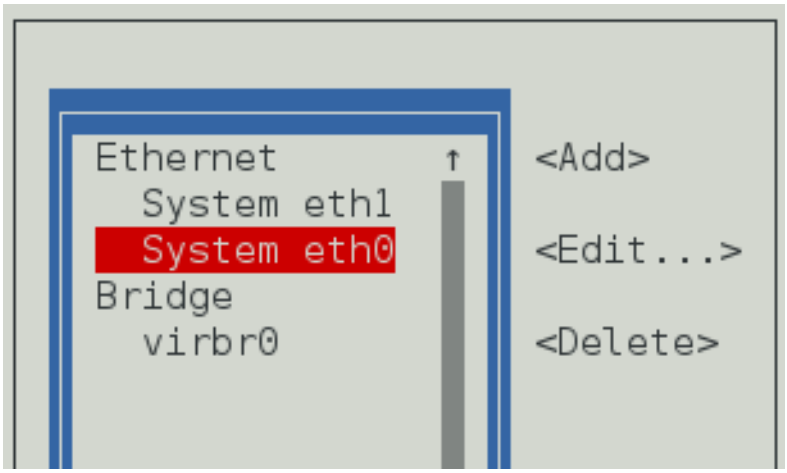
* Oh yes I switched it off!

- On a system with NetworkManager enabled:-

nmtui



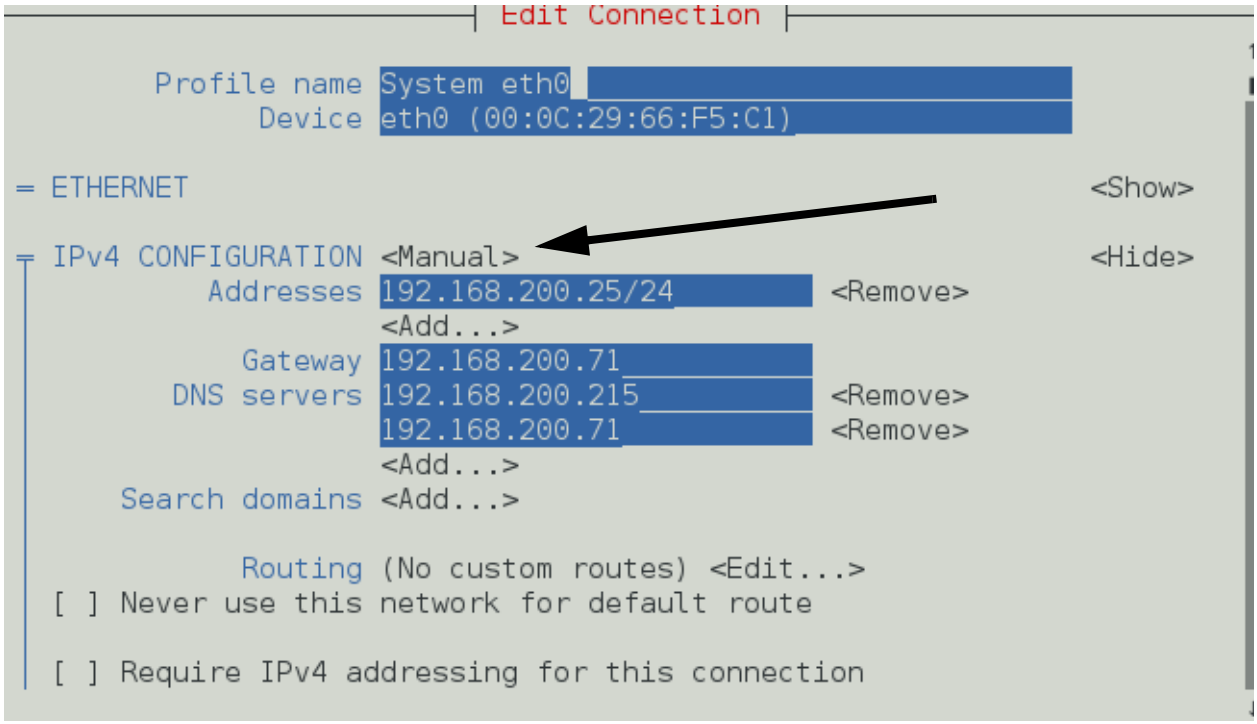
* Select *Edit a Connection,* then the interface to edit:-



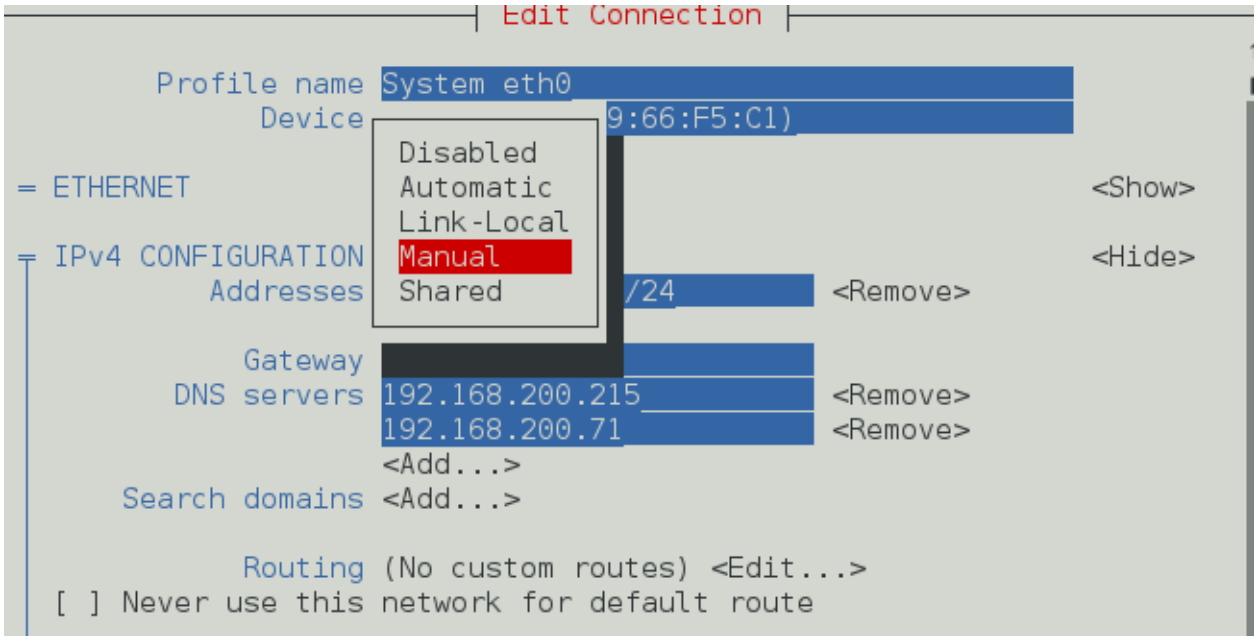
Enterprise Linux 7 Update

NetworkManager Command Line Tools - nmtui

- Having selected the interface, you can use *nmtui* to make changes:-



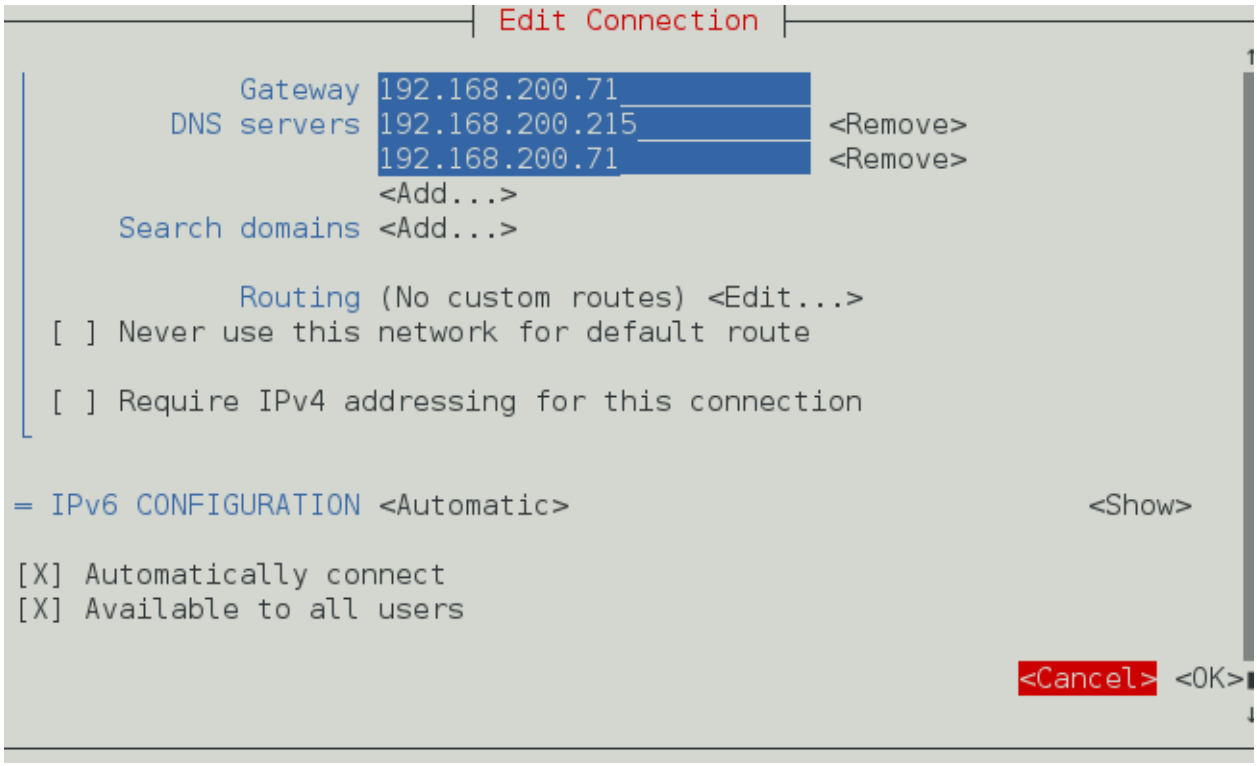
- For example, you can use the cursor keys to move to *<Manual>* then press space bar to invoke a menu of choices:-



Enterprise Linux 7 Update

NetworkManager Command Line Tools - nmtui

- Most operations work in a similar fashion; it just needs a bit of patience to get the hang of it.
- It is not obvious, but there are *Cancel* and *OK* options at the foot of the screen; use the cursor key to reach them:-



- Note the *Automatically connect* option on the left.
- We will now look at the (more complicated) *nmcli* utility.

Enterprise Linux 7 Update

NetworkManager Command Line Tools - nmcli

- *nmcli* is the NetworkManager Command Line Interface (sorry for stating the obvious) and as such works only when the NetworkManager service in running.
- You would use it because:-
    * You cannot run the Network Settings GUI.
    * You need to know it for the RHCSA certification exam.
- One point to mention is the use of the terms *devices* and *connections*; a device is a physical or virtual network interface, and a connection is the configuration used on that device.
- You could have several connections on one device.
- Here are some *nmcli* examples to get us going; first showing connections:-

```
# nmcli con show
Error: NetworkManager is not running.
```

    * OK, on a system where it is running:-

```
# nmcli con show
NAME          UUID                                   TYPE            DEVICE
System eth0   57cdc4d7-3d6b-4ba4-b6c5-83173           802-3-ethernet  eth0
System eth1   a54c021d-7fc5-4ea3-863a-746ef           802-3-ethernet  eth1
```

- Use *nmcli con show --active* to show only active connections.

Enterprise Linux 7 Update

nmcli Display connection settings

- To get all the settings of a connection:-

 # nmcli con show "System eth0"

```
connection.id:                          System eth0
connection.uuid:                        57cdc4d7-3d6b-4ba4-b6c5-83173876df8d
connection.interface-name:              eth0
connection.type:                        802-3-ethernet
connection.autoconnect:                 yes
connection.autoconnect-priority:        0
connection.timestamp:                   1451382999
connection.read-only:                   no
etc..

GENERAL.NAME:                           System eth0
GENERAL.UUID:                           57cdc4d7-3d6b-4ba4-b6c5-83173876df8d
GENERAL.DEVICES:                        eth0
GENERAL.STATE:                          activated
GENERAL.DEFAULT:                        yes
GENERAL.DEFAULT6:                       no
GENERAL.VPN:                            no
GENERAL.ZONE:                           --
GENERAL.DBUS-PATH                       /org/freedesktop/NetworkManager/
ActiveConnection/1
GENERAL.CON-PATH                        /org/freedesktop/NetworkManager/Settings/1
GENERAL.SPEC-OBJECT:                    /
GENERAL.MASTER-PATH:                    --
IP4.ADDRESS[1]:                         192.168.200.25/24
IP4.GATEWAY:                            192.168.200.71
IP4.DNS[1]:                             192.168.200.215
IP4.DNS[2]:                             192.168.200.71
IP6.ADDRESS[1]:                         fe80::20c:29ff:fe66:f5c1/64
IP6.GATEWAY:
```

- The name given (*System eth0*) is taken from the first column of the output from the *nmcli con show* command.
- See *man nm-settings* to find out what the settings mean.

Enterprise Linux 7 Update

nmcli Display device information

- *nmcli* can also show device status:-

nmcli dev

DEVICE	TYPE	STATE	CONNECTION
eth0	ethernet	connected	System eth0
eth1	ethernet	connected	System eth1
lo	loopback	unmanaged	--

- Also for a specific device:-

nmcli dev show eth0

GENERAL.DEVICE:	eth0
GENERAL.TYPE:	ethernet
GENERAL.HWADDR:	00:0C:29:66:F5:C1
GENERAL.MTU:	1500
GENERAL.STATE:	100 (connected)
GENERAL.CONNECTION:	System eth0
GENERAL.CON-PATH:	/org/freedesktop/NetworkManager/ ActiveConnection/1
WIRED-PROPERTIES.CARRIER:	on
IP4.ADDRESS[1]:	192.168.200.25/24
IP4.GATEWAY:	192.168.200.71
IP4.DNS[1]:	192.168.200.215
IP4.DNS[2]:	192.168.200.71
IP6.ADDRESS[1]:	fe80::20c:29ff:fe66:f5c1/64
IP6.GATEWAY:	

- Now we'll see how to perform network configuration changes with *nmcli*.

Enterprise Linux 7 Update

nmcli Configuration example - new connection

- To start with we will create another connection on a device.

```
# nmcli con add type Ethernet ifname eth0 \
con-name "New eth0" ip4 130.100.4.5/16 gw4 130.100.1.1
```

Connection 'New eth0' (84e9fb77-7f0f-4c5c-9470-45f2b1927e0c) successfully added.

  * *ifname eth0* is the device name on which to create the connection; use an appropriate device name that exists on your system.

- A new *ifcfg-New-eth0* is created in the */etc/sysconfig/network-scripts* directory.
- Had we not specified *ip4* and the address, the connection, when activated, would use DHCP.
- The original configuration in *ifcfg-eth0* is still there, and is still the currently active connection.

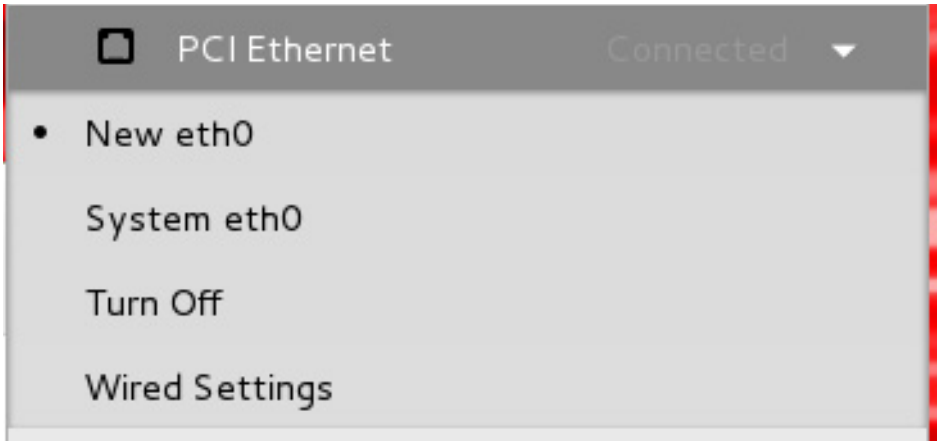
Enterprise Linux 7 Update

nmcli Configuration example - new connection

- To switch between connections, you can use:-

```
# nmcli  con  down  "System eth0"
# nmcli  con  up  "New eth0"
```

- Use *ip addr show eth0* to check the connection address.
- The desktop Network menu would also show two available connections, with one active:-



- Actually, it is probably best to use a *disable* instead of a *con down*, to make sure all the current network connections are properly disabled:-

```
# nmcli  dev  dis  eth0
Device 'eth0' successfully disconnected.
```

* Use the device name this time.

Enterprise Linux 7 Update

nmcli Configuration example - new connection

- When you add a new connection with *nmcli*, and there is already a connection active, the new connection is simply stored and not activated.
- However, under other circumstances, such as accessing wireless networks or maybe hot-plug network devices, the connection may be automatically activated.
- If you do not wish to activate the connection automatically, you can use *autoconnect no* in the command options.
- Let's delete the new connection, then add it again:-

```
# nmcli con delete "New eth0"
Connection 'New eth0' (84e9fb77-7f0f-4c5c-9470-45f2b1927e0c) successfully deleted.
# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state
UP qlen 1000
    link/ether 00:0c:29:66:f5:c1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.25/24 brd 192.168.200.255 scope global eth0
etc..
# nmcli con add type Ethernet ifname eth0 autoconnect no \
con-name "New eth0" ip4 130.100.4.5/16 gw4 130.100.1.1
Connection 'New eth0' (b231e70b-d6cb-4ec0-8d09-e3d8afb4b345) successfully added.
```

- There is no need to do this with our type of connection.

Enterprise Linux 7 Update

nmcli Configuration example - modifying connections

- To set *autoconnect no* after the connection is created:-

```
# nmcli con mod "New eth0" connection.autoconnect no
```

- The change the connection *New eth0* IP:-

```
# nmcli con mod "New eth0" ipv4.addresses 130.100.4.6/16
# nmcli con show "New eth0" | grep ipv4.address
ipv4.addresses:                 130.100.4.6/16
```

  * Note how we use *ipv4* this time, rather than *ip4*.

- To make the changes active:-

```
# nmcli con up "New eth0"
Connection successfully activated (D-Bus active path: /org/freedesktop/
NetworkManager/ActiveConnection/13)
# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state
UP qlen 1000
   link/ether 00:0c:29:66:f5:c1 brd ff:ff:ff:ff:ff:ff
   inet 130.100.4.6/16 brd 130.100.255.255 scope global eth0
etc.
```

- To add a new DNS server:-

```
# nmcli con mod "New eth0" ipv4.dns 8.8.8.8
# nmcli con show "New eth0" | grep ipv4.dns
ipv4.dns:                       8.8.8.8
ipv4.dns-search:
```

  * To activate the changes:-

```
# nmcli con up "New eth0"
```

Enterprise Linux 7 Update

nmcli Configuration example - modifying connections

- To specify multiple DNS servers:-

```
# nmcli con mod "New eth0" ipv4.dns "8.8.8.8 192.168.200.71"
# nmcli  con  show  "New eth0" | grep  ipv4.dns
ipv4.dns:                       8.8.8.8,192.168.200.71
```

- That concludes our session on the NetworkManager command line tools.
- The are a lot more examples of *nmcli* in the man pages for *nmcli* and *nmcli-examples*.
- We have covered more than enough for certification exam purposes.