

nmcli - NetworkManager

nmcli manages NetworkManager in Linux. Work with two core concepts: **Device** (physical hardware like eth0 or wlan0) and **Connection** (a saved configuration profile).

Overview & Diagnostics

* View overall network status:

```
$ nmcli general status
```

* List physical network interfaces and their status:

```
$ nmcli device status
-----
DEVICE   TYPE        STATE                CONNECTION
enp2s0   ethernet    connected            Wired connection 1
lo       loopback    connected (externally) lo
virbr0   bridge      connected (externally) virbr0
```

* Show detailed interface info (IP, MAC, DNS, Gateway):

```
$ nmcli device show enp2s0
-----
GENERAL.DEVICE:                enp2s0
GENERAL.TYPE:                  ethernet
GENERAL.HWADDR:                50:E5:49:ED:A9:BF
GENERAL.MTU:                   1500
GENERAL.STATE:                 100 (connected)
GENERAL.CONNECTION:            Wired connection 1
GENERAL.CON-PATH:              /org/freedesktop/NetworkManager/ActiveConnection/2
WIRED-PROPERTIES.CARRIER:     on
IP4.ADDRESS[1]:                192.168.178.18/24
IP4.GATEWAY:                   192.168.178.1
IP4.ROUTE[1]:                  dst = 192.168.178.0/24, nh =
0.0.0.0, mt = 100
IP4.ROUTE[2]:                  dst = 0.0.0.0/0, nh = 192.168.178.1,
mt = 100
IP4.DNS[1]:                   192.168.178.1
IP4.DOMAIN[1]:                 lan
IP6.ADDRESS[1]:                2001:1c00:2e07:fa0a:c363:14e8:16bd:6436/64
IP6.ADDRESS[2]:                fd4a:66:67:a:4b23:15d3:3424:d1ab/64
IP6.ADDRESS[3]:                fd4a:66:67:a::e78/128
IP6.ADDRESS[4]:                2001:1c00:2e07:fa0a::e78/128
```

```

IP6.ADDRESS[5]:
2001:1c00:2e07:fa0a:52e5:49ff:feed:a9bf/64
IP6.ADDRESS[6]:          fdaa:66:67:a:52e5:49ff:feed:a9bf/64
IP6.ADDRESS[7]:          fe80::52e5:49ff:feed:a9bf/64
IP6.GATEWAY:             fe80::ee08:6bff:fe84:2043
IP6.ROUTE[1]:            dst = fe80::/64, nh = ::, mt = 1024
IP6.ROUTE[2]:            dst = fdaa:66:67:a::/64, nh = ::, mt
= 100
IP6.ROUTE[3]:            dst = 2001:1c00:2e07:fa0a::/64, nh =
::, mt = 100
IP6.ROUTE[4]:            dst = ::/0, nh =
fe80::ee08:6bff:fe84:2043, mt = 100
IP6.ROUTE[5]:            dst = fdaa:66:67:a::e78/128, nh =
::, mt = 100
IP6.ROUTE[6]:            dst = 2001:1c00:2e07:fa0a::e78/128,
nh = ::, mt = 100
IP6.DNS[1]:              fdaa:66:67:a::1

```

*** List all saved connection profiles:**

```
$ nmcli connection show
```

*** Show only active connections:**

```
$ nmcli connection show --active
-----
```

NAME	UUID	TYPE	DEVICE
Wired connection 1	483fb914-06e5-4b66-b00f-e2b31025eaff	ethernet	enp2s0
lo	ce29234b-7809-4b88-b743-79bf79ac55bf	loopback	lo
virbr0	4778a23b-a938-412c-8c64-e78ecb98571c	bridge	virbr0

Wi-Fi Management

*** Scan for nearby Wi-Fi networks:**

```
$ nmcli device wifi list
```

*** Connect to a Wi-Fi network:**

```
# nmcli device wifi connect "SSID_NAME" password "WIFI_PASSWORD"
```

*** Prompt securely for Wi-Fi password (prevents saving password in shell history):**

```
# nmcli --ask device wifi connect "SSID_NAME"
```

*** Toggle Wi-Fi on or off:**

```
$ nmcli radio wifi off
```

```
$ nmcli radio wifi on
```

Connection Controls (Up / Down / Delete)

* **Activate a connection profile:**

```
# nmcli connection up "MyConnection"
```

* **Deactivate a connection profile:**

```
# nmcli connection down "MyConnection"
```

* **Disconnect a hardware device directly:**

```
# nmcli device disconnect eth0
```

* **Delete a saved connection profile:**

```
sudo nmcli connection delete "MyConnection"
```

Network Configuration (IP, Gateway, DNS)

* **Set a static IPv4 address and gateway:**

```
# nmcli connection modify "Wired connection 1" ipv4.addresses  
192.168.1.50/24 ipv4.gateway 192.168.1.1 ipv4.method manual  
# nmcli connection up "Wired connection 1"
```

* **Switch back to dynamic IP (DHCP):**

```
# nmcli connection modify "Wired connection 1" ipv4.method auto  
# nmcli connection up "Wired connection 1"
```

* **Set custom DNS servers:**

```
# nmcli connection modify "Wired connection 1" ipv4.dns "1.1.1.1 8.8.8.8"  
# nmcli connection up "Wired connection 1"
```

Adding New Profiles

* **Add a standard Ethernet connection profile (DHCP):**

```
# nmcli connection add type ethernet con-name "Office-Eth" ifname eth0
```

* **Add a static Ethernet connection profile directly:**

```
# nmcli connection add type ethernet con-name "Static-Eth" ifname eth0 ip4  
10.0.0.10/24 gw4 10.0.0.1
```

Backing Up and Restoring NetworkManager Configuration

When testing network changes such as bridges, VLANs, or bonds with `nmcli`, back up the NetworkManager connection profiles first. Also capture the current runtime state as a readable reference.

1. Back up NetworkManager connection profiles

On Debian Bookworm, persistent NetworkManager connection profiles are normally stored in:

```
/etc/NetworkManager/system-connections/
```

Create a compressed backup:

```
sudo tar --create --gzip --preserve-permissions \  
--file="$HOME/networkmanager-backup-$(date +%F-%H%M%S).tar.gz" \  
/etc/NetworkManager/system-connections
```

Alternatively, create a directory copy under `/root`:

```
sudo mkdir -p /root/networkmanager-backup  
sudo cp -a /etc/NetworkManager/system-connections/. \  
/root/networkmanager-backup/
```

The `-a` option preserves ownership, permissions, timestamps, and symbolic links.

Security note: NetworkManager profiles can contain Wi-Fi credentials or other secrets. Keep the backup readable only by root, especially if it is stored outside `/root`.

For a tar archive stored in your home directory, tighten its permissions:

```
chmod 600 "$HOME"/networkmanager-backup-*.tar.gz
```

2. Save a readable snapshot of the current state

These files are not used for automatic restoration, but they are useful for comparing the working setup with the changed configuration:

```
mkdir -p "$HOME/network-state-before-change"

nmcli connection show \
> "$HOME/network-state-before-change/nm-connections.txt"

nmcli device show \
> "$HOME/network-state-before-change/nm-devices.txt"

ip address show \
> "$HOME/network-state-before-change/ip-address.txt"

ip route show table all \
> "$HOME/network-state-before-change/ip-routes.txt"

ip rule show \
> "$HOME/network-state-before-change/ip-rules.txt"
```

For more detail, export every NetworkManager profile in a human-readable form:

```
while IFS= read -r uuid; do
    nmcli --show-secrets connection show uuid "$uuid"
    printf '\n%s\n\n' '-----'
done <<(nmcli -g UUID connection show) \
> "$HOME/network-state-before-change/nm-connections-full.txt"

chmod 600 "$HOME/network-state-before-change/nm-connections-full.txt"
```

Because `--show-secrets` may reveal credentials, protect this file carefully. Omit `--show-secrets` if the secrets are not needed in the reference copy.

3. Restore the saved profiles

A local console, hypervisor console, IPMI/iDRAC/iLO session, or other out-of-band access is strongly recommended before restarting NetworkManager.

Restore from the directory copy

```
sudo systemctl stop NetworkManager

sudo mkdir -p /etc/NetworkManager/system-connectionsntables
sudo rm -f /etc/NetworkManager/system-connections/*
```

```
sudo cp -a /root/networkmanager-backup/. \
/etc/NetworkManager/system-connections/

sudo chown -R root:root /etc/NetworkManager/system-connections
sudo find /etc/NetworkManager/system-connections \
-type f -exec chmod 600 {} +

sudo systemctl start NetworkManager
sudo nmcli connection reload
```

Restore from the tar archive

Replace the archive name with the actual backup file:

```
sudo systemctl stop NetworkManager
sudo rm -rf /etc/NetworkManager/system-connections
sudo tar --extract --gzip --preserve-permissions \
--file="$HOME/networkmanager-backup-YYYY-MM-DD-HHMMSS.tar.gz" \
--directory=/
sudo chown -R root:root /etc/NetworkManager/system-connections
sudo systemctl start NetworkManager
sudo nmcli connection reload
```

After restoring, verify the profiles and device state:

```
nmcli connection show
nmcli device status
ip address show
ip route show
```

If the expected connection does not activate automatically, bring it up explicitly:

```
sudo nmcli connection up "CONNECTION_NAME"
```

4. Safer testing over SSH

Network changes involving a bridge can disconnect the interface carrying the SSH session. Prefer a local or out-of-band console.

If the `at` command is installed and its service is running, schedule a rollback **before** applying the test configuration.

First create a rollback script:

```
sudo tee /root/restore-networkmanager.sh >/dev/null <<'EOF'
#!/bin/sh
set -eu
```

```
systemctl stop NetworkManager
rm -f /etc/NetworkManager/system-connections/*
cp -a /root/networkmanager-backup/. /etc/NetworkManager/system-connections/
chown -R root:root /etc/NetworkManager/system-connections
find /etc/NetworkManager/system-connections -type f -exec chmod 600 {} +
systemctl start NetworkManager
nmcli connection reload
EOF
```

```
sudo chmod 700 /root/restore-networkmanager.sh
```

Schedule it, for example, ten minutes from now:

```
echo /root/restore-networkmanager.sh | sudo at now + 10 minutes
```

List the scheduled job:

```
sudo atq
```

After confirming that the new configuration works, cancel the rollback job:

```
sudo atrm JOB_ID
```

A simple scheduled `systemctl restart NetworkManager` is not a full rollback. It only restarts NetworkManager with whatever configuration is currently present. The rollback job must restore the saved profile files first.

5. Recommended quick workflow

Before the experiment:

```
sudo rm -rf /root/networkmanager-backup
sudo mkdir -p /root/networkmanager-backup
sudo cp -a /etc/NetworkManager/system-connections/. \
  /root/networkmanager-backup/

nmcli connection show
nmcli device status
```

Make the changes using `nmcli`. If they work, keep them and remove the temporary rollback job. If they fail, restore the profile directory from a local or out-of-band console.

6. Additional protection for a virtual machine

If the Debian host is a virtual machine, take a hypervisor snapshot before changing the network configuration. This provides a complete rollback point, not only a backup of NetworkManager profiles.

Important limitations

- The profile backup covers persistent NetworkManager connection profiles. - It does not necessarily capture temporary runtime-only changes made directly with commands such as `ip address add` or `ip route add`. - It does not back up unrelated network configuration, firewall rules, DNS server configuration outside NetworkManager, or custom dispatcher scripts. - For a broader system backup, also consider `/etc/NetworkManager/NetworkManager.conf`, `/etc/NetworkManager/conf.d/`, `/etc/NetworkManager/dispatcher.d/`, and any locally maintained firewall configuration.

From:

<https://wiki.oscardegroot.nl/> - HomeWiki

Permanent link:

<https://wiki.oscardegroot.nl/doku.php?id=networking:networkmanager-nmcli&rev=1789398203>

Last update: **2026/09/14 15:03**

