Red Hat Enterprise Linux Release 9.2 Manual Pages on 'nmcli.1' command

$ man nmcli.1

NMCLI(1)   General Commands Manual   NMCLI(1)

NAME

    nmcli - command-line tool for controlling NetworkManager

SYNOPSIS

    nmcli [OPTIONS...] {help | general | networking | radio | connection |

        device | agent | monitor} [COMMAND] [ARGUMENTS...]

DESCRIPTION

    nmcli is a command-line tool for controlling NetworkManager and

    reporting network status. It can be utilized as a replacement for

    nm-applet or other graphical clients. nmcli is used to create,

    display, edit, delete, activate, and deactivate network connections, as

    well as control and display network device status. See nmcli-

    examples(7) for ready to run nmcli examples.

    Typical uses include:

    ?   Scripts: Utilize NetworkManager via nmcli instead of managing

        network connections manually. nmcli supports a terse output format

        which is better suited for script processing. Note that

        NetworkManager can also execute scripts, called "dispatcher

        scripts", in response to network events. See NetworkManager(8) for

        details about these dispatcher scripts.

    ?   Servers, headless machines, and terminals: nmcli can be used to

        control NetworkManager without a GUI, including creating, editing,

        starting and stopping network connections and viewing network

status.

OPTIONS

-a | --ask

When using this option nmcli will stop and ask for any missing

required arguments, so do not use this option for non-interactive

purposes like scripts. This option controls, for example, whether

you will be prompted for a password if it is required for

connecting to a network.

-c | --colors {yes | no | auto}

This option controls color output (using terminal escape

sequences). yes enables colors, no disables them, auto only

produces colors when standard output is directed to a terminal. The

default value is auto.

The actual colors used are configured as described in terminal-

colors.d(5). Please refer to the COLORS section for a list of color

names supported by nmcli.

If the environment variable NO_COLOR is set (to any value), then

coloring is disabled with mode "auto". Explicitly enabling coloring

overrides the environment variable.

--complete-args

Instead of conducting the desired action, nmcli will list possible

completions for the last argument. This is useful to implement

argument completion in shell.

The exit status will indicate success or return a code 65 to

indicate the last argument is a file name.

NetworkManager ships with command completion support for GNU Bash.

-e | --escape {yes | no}

Whether to escape : and \ characters in terse tabular mode. The

escape character is \.

If omitted, default is yes.

-f | --fields {field1,field2... | all | common}

This option is used to specify what fields (column names) should be

printed. Valid field names differ for specific commands. List

available fields by providing an invalid value to the --fields

option. all is used to print all valid field values of the

command. common is used to print common field values of the

command.

If omitted, default is common.

-g | --get-values {field1,field2... | all | common}

This option is used to print values from specific fields. It is

basically a shortcut for --mode tabular --terse --fields and is a

convenient way to retrieve values for particular fields. The values

are printed one per line without headers.

If a section is specified instead of a field, the section name will

be printed followed by colon separated values of the fields

belonging to that section, all on the same line.

-h | --help

Print help information.

-m | --mode {tabular | multiline}

Switch between tabular and multiline output:

tabular

Output is a table where each line describes a single entry.

Columns define particular properties of the entry.

multiline

Each entry comprises multiple lines, each property on its own

line. The values are prefixed with the property name.

If omitted, default is tabular for most commands. For the commands

producing more structured information, that cannot be displayed on

a single line, default is multiline. Currently, they are:

?   nmcli connection show ID

?   nmcli device show

-p | --pretty

Output is pretty. This causes nmcli to produce easily readable

outputs for humans, i.e. values are aligned, headers are printed,

etc.

-s | --show-secrets

When using this option nmcli will display passwords and secrets

that might be present in an output of an operation. This option

also influences echoing passwords typed by user as an input.

-t | --terse

Output is terse. This mode is designed and suitable for computer

(script) processing.

--offline

Work without a daemon. Makes connection add and connection modify

commands accept and produce connection data via standard

input/output. Ordinarily, nmcli would communicate with the

NetworkManager service.

The connection data format (keyfile) is described in nm-settings-

keyfile(5) manual.

-v | --version

Show nmcli version.

-w | --wait seconds

This option sets a timeout period for which nmcli will wait for

NetworkManager to finish operations. It is especially useful for

commands that may take a longer time to complete, e.g. connection

activation.

Specifying a value of 0 instructs nmcli not to wait but to exit

immediately with a status of success. The default value depends on

the executed command.

GENERAL COMMANDS

nmcli general {status | hostname | permissions | logging | reload}

[ARGUMENTS...]

Use this command to show NetworkManager status and permissions. You can

also get and change system hostname, as well as NetworkManager logging

level and domains.

status

Show overall status of NetworkManager. This is the default action,

when no additional command is provided for nmcli general.

hostname [hostname]

Get and change system hostname. With no arguments, this prints

currently configured hostname. When you pass a hostname, it will be

handed over to NetworkManager to be set as a new system hostname.

Note that the term "system" hostname may also be referred to as

"persistent" or "static" by other programs or tools. The hostname

is stored in /etc/hostname file in most distributions. For example,

systemd-hostnamed service uses the term "static" hostname and it

only reads the /etc/hostname file when it starts.

permissions

Show the permissions a caller has for various authenticated

operations that NetworkManager provides, like enable and disable

networking, changing Wi-Fi and WWAN state, modifying connections,

etc.

logging [level level] [domains domains...]

Get and change NetworkManager logging level and domains. Without

any argument current logging level and domains are shown. In order

to change logging state, provide level and, or, domain parameters.

See NetworkManager.conf(5) for available level and domain values.

reload [flags...]

Reload NetworkManager's configuration and perform certain updates,

like flushing caches or rewriting external state to disk. This is

similar to sending SIGHUP to NetworkManager but it allows for more

fine-grained control over what to reload through the flags

argument. It also allows non-root access via PolicyKit and contrary

to signals it is synchronous. Available flags are:

conf

Reload the NetworkManager.conf configuration from disk. Note

that this does not include connections, which can be reloaded

through nmcli connection reload instead.

dns-rc

Update DNS configuration, which usually involves writing

/etc/resolv.conf anew. This is equivalent to sending the

SIGUSR1 signal to the NetworkManager process.

dns-full

Restart the DNS plugin. This is for example useful when using

dnsmasq plugin, which uses additional configuration in

/etc/NetworkManager/dnsmasq.d. If you edit those files, you can

restart the DNS plugin. This action shortly interrupts name

resolution.

With no flags, everything that is supported is reloaded, which is

identical to sending a SIGHUP. See NetworkManager(8) for more

details about signals.

NETWORKING CONTROL COMMANDS

nmcli networking {on | off | connectivity} [ARGUMENTS...]

Query NetworkManager networking status, enable and disable networking.

on, off

Enable or disable networking control by NetworkManager. All

interfaces managed by NetworkManager are deactivated when

networking is disabled.

connectivity [check]

Get network connectivity state. The optional check argument tells

NetworkManager to re-check the connectivity, else the most recent

known connectivity state is displayed without re-checking.

Possible states are:

the host is not connected to any network.

portal

the host is behind a captive portal and cannot reach the full

Internet.

limited

the host is connected to a network, but it has no access to the

Internet.

full

the host is connected to a network and has full access to the

Internet.

unknown

the connectivity status cannot be found out.

RADIO TRANSMISSION CONTROL COMMANDS

nmcli radio {all | wifi | wwan} [ARGUMENTS...]

Show radio switches status, or enable and disable the switches.

wifi [on | off]

Show or set status of Wi-Fi in NetworkManager. If no arguments are

supplied, Wi-Fi status is printed; on enables Wi-Fi; off disables

Wi-Fi.

wwan [on | off]

Show or set status of WWAN (mobile broadband) in NetworkManager. If

no arguments are supplied, mobile broadband status is printed; on

enables mobile broadband, off disables it.

all [on | off]

Show or set all previously mentioned radio switches at the same

time.

ACTIVITY MONITOR

nmcli monitor

Observe NetworkManager activity. Watches for changes in connectivity

state, devices or connection profiles.

See also nmcli connection monitor and nmcli device monitor to watch for

changes in certain devices or connections.

CONNECTION MANAGEMENT COMMANDS

nmcli connection {show | up | down | modify | add | edit | clone |

delete | monitor | reload | load | import | export |

migrate} [ARGUMENTS...]

NetworkManager stores all network configuration as "connections", which

are collections of data (Layer2 details, IP addressing, etc.) that

describe how to create or connect to a network. A connection is

"active" when a device uses that connection's configuration to create

or connect to a network. There may be multiple connections that apply

to a device, but only one of them can be active on that device at any

given time. The additional connections can be used to allow quick

switching between different networks and configurations.

Consider a machine which is usually connected to a DHCP-enabled network, but sometimes connected to a testing network which uses static IP addressing. Instead of manually reconfiguring eth0 each time the network is changed, the settings can be saved as two connections which both apply to eth0, one for DHCP (called default) and one with the static addressing details (called testing). When connected to the DHCP-enabled network the user would run nmcli con up default , and when connected to the static network the user would run nmcli con up testing.

show [--active] [--order [+-]category:...]

    List in-memory and on-disk connection profiles, some of which may also be active if a device is using that connection profile. Without a parameter, all profiles are listed. When --active option is specified, only the active profiles are shown.

    The --order option can be used to get custom ordering of connections. The connections can be ordered by active status (active), name (name), type (type) or D-Bus path (path). If connections are equal according to a sort order category, an additional category can be specified. The default sorting order is equivalent to --order active:name:path. + or no prefix means sorting in ascending order (alphabetically or in numbers), - means reverse (descending) order. The category names can be abbreviated (e.g. --order -a:na).

show [--active] [id | uuid | path | apath] ID...

    Show details for specified connections. By default, both static configuration and active connection data are displayed. When --active option is specified, only the active profiles are taken into account. Use global --show-secrets option to display secrets associated with the profile.

    id, uuid, path and apath keywords can be used if ID is ambiguous.

    Optional ID-specifying keywords are:

    id

        the ID denotes a connection name.

uuid

    the ID denotes a connection UUID.

path

    the ID denotes a D-Bus static connection path in the format of

    /org/freedesktop/NetworkManager/Settings/num or just num.

apath

    the ID denotes a D-Bus active connection path in the format of

    /org/freedesktop/NetworkManager/ActiveConnection/num or just

    num.

It is possible to filter the output using the global --fields

option. Use the following values:

profile

    only shows static profile configuration.

active

    only shows active connection data (when the profile is active).

You can also specify particular fields. For static configuration,

use setting and property names as described in nm-settings-nmcli(5)

manual page. For active data use GENERAL, IP4, DHCP4, IP6, DHCP6,

VPN.

When no command is given to the nmcli connection, the default

action is nmcli connection show.

up [id | uuid | path] ID [ifname ifname] [ap BSSID] [passwd-file file]

Activate a connection. The connection is identified by its name,

UUID or D-Bus path. If ID is ambiguous, a keyword id, uuid or path

can be used. When requiring a particular device to activate the

connection on, the ifname option with interface name should be

given. If the ID is not given an ifname is required, and

NetworkManager will activate the best available connection for the

given ifname. In case of a VPN connection, the ifname option

specifies the device of the base connection. The ap option specify

what particular AP should be used in case of a Wi-Fi connection.

If --wait option is not specified, the default timeout will be 90

seconds.

See connection show above for the description of the ID-specifying keywords.

Available options are:

ifname

interface that will be used for activation.

ap

BSSID of the AP which the command should connect to (for Wi-Fi connections).

passwd-file

some networks may require credentials during activation. You can give these credentials using this option. Each line of the file should contain one password in the form:

setting_name.property_name:the password

For example, for WPA Wi-Fi with PSK, the line would be

802-11-wireless-security.psk:secret12345

For 802.1X password, the line would be

802-1x.password:my 1X password

nmcli also accepts wifi-sec and wifi strings instead of 802-11-wireless-security. When NetworkManager requires a password and it is not given, nmcli will ask for it when run with --ask. If --ask was not passed, NetworkManager can ask another secret agent that may be running (typically a GUI secret agent, such as nm-applet or gnome-shell).

down [id | uuid | path | apath] ID...

Deactivate a connection from a device without preventing the device from further auto-activation. Multiple connections can be passed to the command.

Be aware that this command deactivates the specified active connection, but the device on which the connection was active, is still ready to connect and will perform auto-activation by looking for a suitable connection that has the 'autoconnect' flag set. Note that the deactivating connection profile is internally blocked from autoconnecting again. Hence it will not autoconnect until reboot or

until the user performs an action that unblocks autoconnect, like

modifying the profile or explicitly activating it.

In most cases you may want to use device down command instead.

The connection is identified by its name, UUID or D-Bus path. If ID

is ambiguous, a keyword id, uuid, path or apath can be used.

See connection show above for the description of the ID-specifying

keywords.

If --wait option is not specified, the default timeout will be 10

seconds.

modify [--temporary] [id | uuid | path] [ID]

{option value | [+|-]setting.property value}...

Add, modify or remove properties in the connection profile.

To set the property just specify the property name followed by the

value. An empty value ("") resets the property value to the

default.

See nm-settings-nmcli(5) for complete reference of setting and

property names, their descriptions and default values. The setting

and property can be abbreviated provided they are unique.

If you want to append an item or a flag to the existing value, use

+ prefix for the property name or alias. If you want to remove

items from a container-type or flag property, use - prefix. For

certain properties you can also remove elements by specifying the

zero-based index(es). The + and - modifiers only have a real effect

for properties that support them. These are for example multi-value

(container) properties or flags like ipv4.dns, ip4, ipv4.addresses,

bond.options, 802-1x.phase1-auth-flags etc.

The connection is identified by its name, UUID or D-Bus path. If ID

is ambiguous, a keyword id, uuid or path can be used. The ID is not

used with the global --offline option.

When the global --offline is used, the command reads the connection

from the standard input and prints the modified connection to

standard output instead of making the the NetworkManager daemon act

upon specified connection.

modify [--temporary] [id | uuid | path] ID remove setting

    Removes a setting from the connection profile.

add [save {yes | no}] {option value | [+|-]setting.property value}...

    Create a new connection using specified properties.

    You need to describe the newly created connections with the

    property and value pairs. See nm-settings-nmcli(5) for the complete

    reference. The syntax is the same as of the nmcli connection modify

    command.

    To construct a meaningful connection you at the very least need to

    set the connection.type property (or use the type alias) to one of

    known NetworkManager connection types:

- ? 6lowpan
- ? 802-11-olpc-mesh (alias olpc-mesh)
- ? 802-11-wireless (alias wifi)
- ? 802-3-ethernet (alias ethernet)
- ? adsl
- ? bluetooth
- ? bond
- ? bond-slave (deprecated for ethernet with master)
- ? bridge
- ? bridge-slave (deprecated for ethernet with master)
- ? cdma
- ? dummy
- ? generic
- ? gsm
- ? infiniband
- ? ip-tunnel
- ? macsec
- ? macvlan
- ? olpc-mesh
- ? ovs-bridge
- ? ovs-dpdk
- ? ovs-interface

- ? ovs-patch

- ? ovs-port

- ? pppoe

- ? team

- ? team-slave (deprecated for ethernet with master)

- ? tun

- ? veth

- ? vlan

- ? vpn

- ? vrf

- ? vxlan

- ? wifi-p2p

- ? wimax

- ? wireguard

- ? wpan

The most typical uses are described in the EXAMPLES section.

Aside from the properties and values two special options are

accepted:

save

Controls whether the connection should be persistent, i.e.

NetworkManager should store it on disk (default: yes).

--

If a single -- argument is encountered it is ignored. This is

for compatibility with older versions on nmcli.

When the global --offline is used, the command prints the resulting

connection to standard output instead of actually adding the

connection via the NetworkManager daemon.

edit {[id | uuid | path] ID | [type type] [con-name name] }

Edit an existing connection or add a new one, using an interactive

editor.

The existing connection is identified by its name, UUID or D-Bus

path. If ID is ambiguous, a keyword id, uuid, or path can be used.

See connection show above for the description of the ID-specifying

keywords. Not providing an ID means that a new connection will be added.

The interactive editor will guide you through the connection editing and allow you to change connection parameters according to your needs by means of a simple menu-driven interface. The editor indicates what settings and properties can be modified and provides in-line help.

Available options:

type

    type of the new connection; valid types are the same as for connection add command.

con-name

    name for the new connection. It can be changed later in the editor.

See also nm-settings-nmcli(5) for all NetworkManager settings and property names, and their descriptions; and nmcli-examples(7) for sample editor sessions.

clone [--temporary] [id | uuid | path] ID new_name

Clone a connection. The connection to be cloned is identified by its name, UUID or D-Bus path. If ID is ambiguous, a keyword id, uuid or path can be used. See connection show above for the description of the ID-specifying keywords. new_name is the name of the new cloned connection. The new connection will be the exact copy except the connection.id (new_name) and connection.uuid (generated) properties.

The new connection profile will be saved as persistent unless --temporary option is specified, in which case the new profile won't exist after NetworkManager restart.

delete [id | uuid | path] ID...

Delete a configured connection. The connection to be deleted is identified by its name, UUID or D-Bus path. If ID is ambiguous, a keyword id, uuid or path can be used. See connection show above for the description of the ID-specifying keywords.

If --wait option is not specified, the default timeout will be 10 seconds.

monitor [id | uuid | path] ID...

Monitor connection profile activity. This command prints a line whenever the specified connection changes. The connection to be monitored is identified by its name, UUID or D-Bus path. If ID is ambiguous, a keyword id, uuid or path can be used. See connection show above for the description of the ID-specifying keywords. Monitors all connection profiles in case none is specified. The command terminates when all monitored connections disappear. If you want to monitor connection creation consider using the global monitor with nmcli monitor command.

reload

Reload all connection files from disk. NetworkManager does not monitor changes to connection. So you need to use this command in order to tell NetworkManager to re-read the connection profiles from disk when a change was made to them.

load filename...

Load/reload one or more connection files from disk. Use this after manually editing a connection file to ensure that NetworkManager is aware of its latest state.

import [--temporary] type type file file

Import an external/foreign configuration as a NetworkManager connection profile. The type of the input file is specified by type option.

Only VPN configurations are supported at the moment. The configuration is imported by NetworkManager VPN plugins. type values are the same as for vpn-type option in nmcli connection add. VPN configurations are imported by VPN plugins. Therefore the proper VPN plugin has to be installed so that nmcli could import the data.

The imported connection profile will be saved as persistent unless --temporary option is specified, in which case the new profile

won't exist after NetworkManager restart.

export [id | uuid | path] ID [file]

Export a connection.

Only VPN connections are supported at the moment. A proper VPN

plugin has to be installed so that nmcli could export a connection.

If no file is provided, the VPN configuration data will be printed

to standard output.

migrate [--plugin plugin...] [id | uuid | path] [ID...]

Migrate connection profiles to a different settings plugin, such as

keyfile (default) or ifcfg-rh.

The connection to be migrated is identified by its name, UUID or

D-Bus path. If ID is ambiguous, a keyword id, uuid or path can be

used. See connection show above for the description of the

ID-specifying keywords.

If no connections are specified, the command acts on all available

connections. Therefore, with no arguments, the command migrates all

connection profiles to the keyfile plugin.

If --wait option is not specified, the default timeout will be 10

seconds.

DEVICE MANAGEMENT COMMANDS

nmcli device {status | show | set | up | connect | reapply | modify |

down | disconnect | delete | monitor | wifi | lldp |

checkpoint} [ARGUMENTS...]

Show and manage network interfaces.

status

Print status of devices.

This is the default action if no command is specified to nmcli

device.

show [ifname]

Show detailed information about devices. Without an argument, all

devices are examined. To get information for a specific device, the

interface name has to be provided.

set [ifname] ifname [autoconnect {yes | no}] [managed {yes | no}]

Set device properties.

up ifname

Connect the device. NetworkManager will try to find a suitable
connection that will be activated. It will also consider
connections that are not set to auto connect.

If no compatible connection exists, a new profile with default
settings will be created and activated. This differentiates nmcli
connection up ifname "$DEVICE" from nmcli device up "$DEVICE"

If --wait option is not specified, the default timeout will be 90
seconds.

connect ifname

Alias for command up. Before version 1.34.0 up was not supported.

reapply ifname

Attempt to update device with changes to the currently active
connection made since it was last applied.

modify ifname {option value | [+|-]setting.property value}...

Modify the settings currently active on the device.

This command lets you do temporary changes to a configuration
active on a particular device. The changes are not preserved in the
connection profile.

See nm-settings-nmcli(5) for the list of available properties.

Please note that some properties can't be changed on an already
connected device.

down ifname...

Disconnect a device and prevent the device from automatically
activating further connections without user/manual intervention.

Note that disconnecting software devices may mean that the devices
will disappear.

If --wait option is not specified, the default timeout will be 10
seconds.

disconnect ifname...

Alias for command down. Before version 1.34.0 down was not
supported.

delete ifname...

> Delete a device. The command removes the interface from the system.
>
> Note that this only works for software devices like bonds, bridges,
>
> teams, etc. Hardware devices (like Ethernet) cannot be deleted by
>
> the command.
>
> If --wait option is not specified, the default timeout will be 10
>
> seconds.

monitor [ifname...]

> Monitor device activity. This command prints a line whenever the
>
> specified devices change state.
>
> Monitors all devices in case no interface is specified. The monitor
>
> terminates when all specified devices disappear. If you want to
>
> monitor device addition consider using the global monitor with
>
> nmcli monitor command.

wifi [list [--rescan | auto | no | yes] [ifname ifname] [bssid BSSID]]

> List available Wi-Fi access points. The ifname and bssid options
>
> can be used to list APs for a particular interface or with a
>
> specific BSSID, respectively.
>
> By default, nmcli ensures that the access point list is no older
>
> than 30 seconds and triggers a network scan if necessary. The
>
> --rescan can be used to either force or disable the scan regardless
>
> of how fresh the access point list is.

wifi connect (B)SSID [password password] [wep-key-type {key | phrase}]

[ifname ifname] [bssid BSSID] [name name] [private {yes | no}]

[hidden {yes | no}]

> Connect to a Wi-Fi network specified by SSID or BSSID. The command
>
> finds a matching connection or creates one and then activates it on
>
> a device. This is a command-line counterpart of clicking an SSID in
>
> a GUI client. If a connection for the network already exists, it is
>
> possible to bring up (activate) the existing profile as follows:
>
> nmcli con up id name. Note that only open, WEP and WPA-PSK networks
>
> are supported if no previous connection exists. It is also assumed
>
> that IP configuration is obtained via DHCP.

If --wait option is not specified, the default timeout will be 90
seconds.

Available options are:

password

password for secured networks (WEP or WPA).

wep-key-type

type of WEP secret, either key for ASCII/HEX key or phrase for

passphrase.

ifname

interface that will be used for activation.

bssid

if specified, the created connection will be restricted just

for the BSSID.

name

if specified, the connection will use the name (else NM creates

a name itself).

private

if set to yes, the connection will only be visible to the user

who created it. Otherwise, the connection is system-wide, which

is the default.

hidden

set to yes when connecting for the first time to an AP not

broadcasting its SSID. Otherwise, the SSID would not be found

and the connection attempt would fail.

wifi hotspot [ifname ifname] [con-name name] [ssid SSID]

[band {a | bg}] [channel channel] [password password]

Create a Wi-Fi hotspot. The command creates a hotspot connection

profile according to Wi-Fi device capabilities and activates it on

the device. The hotspot is secured with WPA if device/driver

supports that, otherwise WEP is used. Use connection down or device

down to stop the hotspot.

Parameters of the hotspot can be influenced by the optional

parameters:

ifname

 what Wi-Fi device is used.

con-name

 name of the created hotspot connection profile.

ssid

 SSID of the hotspot.

band

 Wi-Fi band to use.

channel

 Wi-Fi channel to use.

password

 password to use for the created hotspot. If not provided, nmcli

 will generate a password. The password is either WPA pre-shared

 key or WEP key.

 Note that --show-secrets global option can be used to print the

 hotspot password. It is useful especially when the password was

 generated.

wifi rescan [ifname ifname] [ssid SSID...]

 Request that NetworkManager immediately re-scan for available

 access points. NetworkManager scans Wi-Fi networks periodically,

 but in some cases it can be useful to start scanning manually (e.g.

 after resuming the computer). By using ssid, it is possible to scan

 for a specific SSID, which is useful for APs with hidden SSIDs. You

 can provide multiple ssid parameters in order to scan more SSIDs.

 This command does not show the APs, use nmcli device wifi list for

 that.

wifi show-password [ifname ifname]

 Show the details of the active Wi-Fi networks, including the

 secrets.

lldp [list [ifname ifname]]

 Display information about neighboring devices learned through the

 Link Layer Discovery Protocol (LLDP). The ifname option can be used

 to list neighbors only for a given interface. The protocol must be

enabled in the connection settings.

checkpoint [--timeout seconds] [ifname...] -- COMMAND...

> Runs the command with a configuration checkpoint taken and asks for
>
> a confirmation when finished. When the confirmation is not given,
>
> the checkpoint is automatically restored after timeout.
>
> This allows doing disruptive configuration changes over remote
>
> connections with an option of restoring the network configuration
>
> to a known good state in case of an error.
>
> If the a list of interface names is specified, the checkpoint is
>
> taken, the checkpoint is takes only on the specified devices.
>
> Otherwise a checkpoint is taken for all devices.
>
> Currently the timeout defaults to 15 seconds. This may change in a
>
> future version.

SECRET AGENT

nmcli agent {secret | polkit | all}

Run nmcli as a NetworkManager secret agent, or polkit agent.

secret

> Register nmcli as a NetworkManager secret agent and listen for
>
> secret requests. You usually do not need this command, because
>
> nmcli can handle secrets when connecting to networks. However, you
>
> may find the command useful when you use another tool for
>
> activating connections and you do not have a secret agent available
>
> (like nm-applet).

polkit

> Register nmcli as a polkit agent for the user session and listen
>
> for authorization requests. You do not usually need this command,
>
> because nmcli can handle polkit actions related to NetworkManager
>
> operations (when run with --ask). However, you may find the command
>
> useful when you want to run a simple text based polkit agent and
>
> you do not have an agent of a desktop environment. Note that
>
> running this command makes nmcli handle all polkit requests, not
>
> only NetworkManager related ones, because only one polkit agent can
>
> run for the session.

all

 Runs nmcli as both NetworkManager secret and a polkit agent.

COLORS

 Implicit coloring can be disabled by an empty file

 /etc/terminal-colors.d/nmcli.disable.

 See terminal-colors.d(5) for more details about colorization

 configuration. The logical color names supported by nmcli are:

 connection-activated

  A connection that is active.

 connection-activating

  Connection that is being activated.

 connection-disconnecting

  Connection that is being disconnected.

 connection-external

  Connection representing configuration created externally to

  NetworkManager.

 connection-invisible

  Connection whose details is the user not permitted to see.

 connection-deprecated

  Connection that uses deprecated settings. It might not be possible

  to activate it.

 connectivity-full

  Connectivity state when Internet is reachable.

 connectivity-limited

  Connectivity state when only a local network reachable.

 connectivity-none

  Connectivity state when the network is disconnected.

 connectivity-portal

  Connectivity state when a captive portal hijacked the connection.

 connectivity-unknown

  Connectivity state when a connectivity check didn't run.

 device-activated

  Device that is connected.

device-activating

   Device that is being configured.

device-disconnected

   Device that is not connected.

device-external

   Device configured externally to NetworkManager.

device-firmware-missing

   Warning of a missing device firmware.

device-plugin-missing

   Warning of a missing device plugin.

device-unavailable

   Device that is not available for activation.

device-disabled

   Device is disabled by software or hardware kill switch.

manager-running

   Notice that the NetworkManager daemon is available.

manager-starting

   Notice that the NetworkManager daemon is being initially connected.

manager-stopped

   Notice that the NetworkManager daemon is not available.

permission-auth

   An action that requires user authentication to get permission.

permission-no

   An action that is not permitted.

permission-yes

   An action that is permitted.

prompt

   Prompt in interactive mode.

state-asleep

   Indication that NetworkManager in suspended state.

state-connected-global

   Indication that NetworkManager in connected to Internet.

state-connected-local

Indication that NetworkManager in local network.

state-connected-site

Indication that NetworkManager in connected to networks other than

Internet.

state-connecting

Indication that NetworkManager is establishing a network

connection.

state-disconnected

Indication that NetworkManager is disconnected from a network.

state-disconnecting

Indication that NetworkManager is being disconnected from a

network.

wifi-signal-excellent

Wi-Fi network with an excellent signal level.

wifi-signal-fair

Wi-Fi network with a fair signal level.

wifi-signal-good

Wi-Fi network with a good signal level.

wifi-signal-poor

Wi-Fi network with a poor signal level.

wifi-signal-unknown

Wi-Fi network that hasn't been actually seen (a hidden AP).

wifi-deprecated

Wi-Fi network that might be impossible to connect to due to use of

deprecated functionality.

disabled

A property that is turned off.

enabled

A property that is turned on.

ENVIRONMENT VARIABLES

nmcli's behavior is affected by the following environment variables.

LC_ALL

If set to a non-empty string value, it overrides the values of all

the other internationalization variables.

LC_MESSAGES

Determines the locale to be used for internationalized messages.

LANG

Provides a default value for the internationalization variables

that are unset or null.

NO_COLOR

Default to not producing colored and paged output. The --colors

option, if used, takes precedence.

PAGER

Filter to pipe the output through if it doesn't fit on a screen.

Can be a file name of an executable or a shell command. Empty

string to disable the functionality.

Note that the pager command is expected to handle wide characters

and ANSI escape sequences for changing colors (unless they're

disabled). nmcli sets up the environment variables LESS and

LESSCHARSET appropriately for the less(1) pager, other pagers may

or may not need extra configuration.

If unspecified, pager(1), less(1) and more(1) will be tried (in

that order).

TERM

Terminal type. If dumb, nmcli will not use a pager or produce ANSI

escape sequences for coloring.

Terminal types other than dumb are assumed to support ASCII escape

sequences for setting the output color.

INTERNATIONALIZATION NOTES

Be aware that nmcli is localized and that is why the output depends on

your environment. This is important to realize especially when you

parse the output.

Call nmcli as LC_ALL=C nmcli to be sure the locale is set to C while

executing in a script.

LC_ALL, LC_MESSAGES, LANG variables specify the LC_MESSAGES locale

category (in that order), which determines the language that nmcli uses

for messages. The C locale is used if none of these variables are set, and this locale uses English messages.

EXIT STATUS

nmcli exits with status 0 if it succeeds, a value greater than 0 is returned if an error occurs.

0

Success ? indicates the operation succeeded.

1

Unknown or unspecified error.

2

Invalid user input, wrong nmcli invocation.

3

Timeout expired (see --wait option).

4

Connection activation failed.

5

Connection deactivation failed.

6

Disconnecting device failed.

7

Connection deletion failed.

8

NetworkManager is not running.

10

Connection, device, or access point does not exist.

65

When used with --complete-args option, a file name is expected to follow.

EXAMPLES

This section presents various examples of nmcli usage. If you want even more, please refer to nmcli-examples(7) manual page.

nmcli -t -f RUNNING general

tells you whether NetworkManager is running or not.

nmcli -t -f STATE general

    shows the overall status of NetworkManager.

nmcli radio wifi off

    switches Wi-Fi off.

nmcli connection show

    lists all connections NetworkManager has.

nmcli -p -m multiline -f all con show

    shows all configured connections in multi-line mode.

nmcli connection show --active

    lists all currently active connections.

nmcli -f name,autoconnect c s

    shows all connection profile names and their auto-connect property.

nmcli -p connection show "My default em1"

    shows details for "My default em1" connection profile.

nmcli --show-secrets connection show "My Home Wi-Fi"

    shows details for "My Home Wi-Fi" connection profile with all

    passwords. Without --show-secrets option, secrets would not be

    displayed.

nmcli -f active connection show "My default em1"

    shows details for "My default em1" active connection, like IP, DHCP

    information, etc.

nmcli -f profile con s "My wired connection"

    shows static configuration details of the connection profile with

    "My wired connection" name.

nmcli -p con up "My wired connection" ifname eth0

    activates the connection profile with name "My wired connection" on

    interface eth0. The -p option makes nmcli show progress of the

    activation.

nmcli con up 6b028a27-6dc9-4411-9886-e9ad1dd43761 ap 00:3A:98:7C:42:D3

    connects the Wi-Fi connection with UUID

    6b028a27-6dc9-4411-9886-e9ad1dd43761 to the AP with BSSID

    00:3A:98:7C:42:D3.

nmcli device status

shows the status for all devices.

nmcli dev down em2

disconnects a connection on interface em2 and marks the device as

unavailable for auto-connecting. As a result, no connection will

automatically be activated on the device until the device's

'autoconnect' is set to TRUE or the user manually activates a

connection.

nmcli -f GENERAL,WIFI-PROPERTIES dev show wlan0

shows details for wlan0 interface; only GENERAL and WIFI-PROPERTIES

sections will be shown.

nmcli -f CONNECTIONS device show wlp3s0

shows all available connection profiles for your Wi-Fi interface

wlp3s0.

nmcli dev wifi

lists available Wi-Fi access points known to NetworkManager.

nmcli dev wifi con "Cafe Hotspot 1" password caffeine name "My cafe"

creates a new connection named "My cafe" and then connects it to

"Cafe Hotspot 1" SSID using password "caffeine". This is mainly

useful when connecting to "Cafe Hotspot 1" for the first time. Next

time, it is better to use nmcli con up id "My cafe" so that the

existing connection profile can be used and no additional is

created.

nmcli -s dev wifi hotspot con-name QuickHotspot

creates a hotspot profile and connects it. Prints the hotspot

password the user should use to connect to the hotspot from other

devices.

nmcli dev modify em1 ipv4.method shared

starts IPv4 connection sharing using em1 device. The sharing will

be active until the device is disconnected.

nmcli dev modify em1 ipv6.address 2001:db8::a:bad:c0de

temporarily adds an IP address to a device. The address will be

removed when the same connection is activated again.

nmcli connection add type ethernet autoconnect no ifname eth0

non-interactively adds an Ethernet connection tied to eth0

interface with automatic IP configuration (DHCP), and disables the

connection's autoconnect flag.

nmcli c a ifname Maxipes-fik type vlan dev eth0 id 55

non-interactively adds a VLAN connection with ID 55. The connection

will use eth0 and the VLAN interface will be named Maxipes-fik.

nmcli c a ifname eth0 type ethernet ipv4.method disabled ipv6.method

link-local

non-interactively adds a connection that will use eth0 Ethernet

interface and only have an IPv6 link-local address configured.

nmcli connection edit ethernet-em1-2

edits existing "ethernet-em1-2" connection in the interactive

editor.

nmcli connection edit type ethernet con-name "yet another Ethernet

connection"

adds a new Ethernet connection in the interactive editor.

nmcli con mod ethernet-2 connection.autoconnect no

modifies 'autoconnect' property in the 'connection' setting of

'ethernet-2' connection.

nmcli con mod "Home Wi-Fi" wifi.mtu 1350

modifies 'mtu' property in the 'wifi' setting of 'Home Wi-Fi'

connection.

nmcli con mod em1-1 ipv4.method manual ipv4.addr "192.168.1.23/24

192.168.1.1, 10.10.1.5/8, 10.0.0.11"

sets manual addressing and the addresses in em1-1 profile.

nmcli con modify ABC +ipv4.dns 8.8.8.8

appends a Google public DNS server to DNS servers in ABC profile.

nmcli con modify ABC -ipv4.addresses "192.168.100.25/24 192.168.1.1"

removes the specified IP address from (static) profile ABC.

nmcli con import type openvpn file ~/Downloads/frootvpn.ovpn

imports an OpenVPN configuration to NetworkManager.

nmcli con export corp-vpnc /home/joe/corpvpn.conf

exports NetworkManager VPN profile corp-vpnc as standard Cisco

(vpnc) configuration.

NOTES

nmcli accepts abbreviations, as long as they are a unique prefix in the

set of possible options. As new options get added, these abbreviations

are not guaranteed to stay unique. For scripting and long term

compatibility it is therefore strongly advised to spell out the full

option names.

BUGS

There are probably some bugs. If you find a bug, please report it to

your distribution or upstream at

https://gitlab.freedesktop.org/NetworkManager/NetworkManager.

SEE ALSO

nmcli-examples(7), nm-settings-nmcli(5), nm-online(1),

NetworkManager(8), NetworkManager.conf(5), nm-applet(1), nm-connection-

editor(1), terminal-colors.d(5).

NetworkManager 1.42.2   NMCLI(1)