# nmcli命令

nmcli是个很强大的命令，后面一大堆选项和对象可以配置。看看帮助文档，对象可以是综合信息，网络，信号和连接。

**地址配置工具：nmcli**

nmcli – command-line tool for controlling NetworkManager

命令语法：

nmcli[ OPTIONS ] OBJECT { COMMAND | help }

OBJECT和COMMAND可以用全称也可以用简称，最少可以只用一个字母，建议用头三个字母。OBJECT里面我们平时用的最多的就是connection和device，这里需要简单区分一下connection和device。

**device** 网络接口，是物理设备

device -show and manage network interfaces

nmcli device help

**connection** 是连接，偏重于逻辑设置

connection -start, stop, and manage network connections

nmcli connection help

多个connection可以应用到同一个device，但同一时间只能启用其中一个connection。这样的好处是针对一个网络接口，我们可以设置多个网络连接，比如静态IP和动态IP，再根据需要up相应connection

```
[root@server1 ~]# nmcli --help
Usage: nmcli [OPTIONS] OBJECT { COMMAND | help }

OPTIONS
  -t[erse]                                terse output
  -p[retty]                               pretty output
  -m[ode] tabular|multiline               output mode
  -f[ields] <field1,field2,...>|all|common specify fields to output
  -e[scape] yes|no                        escape columns separators in values
  -n[ocheck]                              don't check nmcli and NetworkManager versions
  -a[sk]                                  ask for missing parameters
  -w[ait] <seconds>                       set timeout waiting for finishing operations
  -v[ersion]                              show program version
  -h[elp]                                 print this help

OBJECT
  g[eneral]       NetworkManager's general status and operations
  n[etworking]    overall networking control
  r[adio]         NetworkManager radio switches
  c[onnection]    NetworkManager's connections
  d[evice]        devices managed by NetworkManager
```

如何配置连接，还是先看看帮助，他后面可以跟show，up, down, add, modify, edit, delete, reload,

```
[root@server1 ~]# nmcli connection --help
Usage: nmcli connection { COMMAND | help }

COMMAND := { show | up | down | add | modify | edit | delete | reload | load }

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

  up [[id | uuid | path] <ID>] [ifname <ifname>] [ap <BSSID>]

  down [id | uuid | path | apath] <ID>

  add COMMON_OPTIONS TYPE_SPECIFIC_OPTIONS IP_OPTIONS

  modify [--temporary] [id | uuid | path] <ID> ([+|-]<setting>.<property> <value>)+

  edit [id | uuid | path] <ID>
  edit [type <new_con_type>] [con-name <new_con_name>]

  delete [id | uuid | path] <ID>

  reload

  load <filename> [ <filename>... ]
```

看看device有哪些参数

```
[root@server1 ~]# nmcli device --help
Usage: nmcli device { COMMAND | help }

COMMAND := { status | show | connect | disconnect | wifi }

  status

  show [<ifname>]

  connect <ifname>

  disconnect <ifname>

  wifi [list [ifname <ifname>] [bssid <BSSID>]]

  wifi connect <(B)SSID> [password <password>] [wep-key-type key|phrase] [ifname <ifname>]
                         [bssid <BSSID>] [name <name>] [private yes|no]

  wifi rescan [[ifname] <ifname>]
```

```
[root@server1 ~]# nmcli connection down eno16777736
[root@server1 ~]# nmcli connection show
NAME          UUID                                  TYPE            DEVICE
eno16777736   a460f7fb-9236-4b4d-b037-0382d1b382e7  802-3-ethernet  --
eno33554992   0885a5e8-269f-4306-ab63-cbd2f41392e8  802-3-ethernet  eno33554992
[root@server1 ~]# nmcli device status
DEVICE       TYPE      STATE         CONNECTION
eno33554992  ethernet  connected     eno33554992
eno16777736  ethernet  disconnected  --
lo           loopback  unmanaged     --
```

查看具体的设备信息可以通过 nmcli connection show 设备名来查看

```
[root@server1 ~]# nmcli connection show eno16777736
connection.id:                          eno16777736
connection.uuid:                        a460f7fb-9236-4b4d-b037-0382d1b382e7
connection.interface-name:              --
connection.type:                        802-3-ethernet
connection.autoconnect:                 yes
connection.timestamp:                   1430385092
connection.read-only:                   no
connection.permissions:
connection.zone:                        public
connection.master:                      --
```

添加一个新连接，先先看看帮助

```
nmcli connection add --help
```

例如：

nmcli connection add con-name test-con ipv4.addresses "10.1.1.100/24" ipv4.gateway 10.1.1.1 ipv4.dns 202.106.0.20 ipv4.method manual connection.autoconnect yes type ethernet ifname ens33

**注意：添加新连接时con-name、type、ifname这3项必须要设定**

修改现有连接，可以先先看看帮助

nmcli connection modify --help

```
[root@server1 ~]# nmcli connection modify eno16777736 ipv4.method manual
[root@server1 ~]# nmcli connection modify eno16777736 ipv4.addresses "192.168.5.20/24"
[root@server1 ~]# service  network restart
Restarting network (via systemctl):                        [  OK  ]
[root@server1 ~]# ifconfig
eno16777736: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.5.20  netmask 255.255.255.0  broadcast 192.168.5.255
        inet6 fe80::20c:29ff:fe24:ec72  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:24:ec:72  txqueuelen 1000  (Ethernet)
        RX packets 2680  bytes 219713 (214.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3173  bytes 236425 (230.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

nmcli device disconnect ens33　　不连接网卡设备ens33网卡

nmcli device connect ens33　　连接并使用网卡设备ens33

nmcli connection up ens33　　激活一个网络连接ens33

nmcli connection down ens33　　关闭一个网络连接ens33

附：修改RHEL7的网卡名称

RHEL7安装完成之后，默认的网卡名称是eno16777736

输入如下命令，进入对应目录，编辑文件：

vim  /etc/sysconfig/grub

然后，往这个文件中添加“net.ifnames=0 biosdevname=0”内容，作用是禁用该可预测命名规则，如下图所示：

```
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="rd.lvm.lv=rhel/root crashkernel=auto  rd.lvm.lv=rhel/swap vconsole.font=latarcyr
heb-sun16 vconsole.keymap=us rhgb quiet net.ifnames=0 biosdevname=0"
GRUB_DISABLE_RECOVERY="true"
```

接着执行下面的命令，效果如下：

```
[root@server1 ~]# grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-123.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-123.el7.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-60e839870acd49b68d569dcbb05142e2
Found initrd image: /boot/initramfs-0-rescue-60e839870acd49b68d569dcbb05142e2.img
[ 1634.511385] end_request: I/O error, dev fd0, sector 0
done
```

然后，重启系统后查看网卡名称

```
[root@server1 ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.100  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::20c:29ff:fe24:ec72  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:24:ec:72  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 119  bytes 10067 (9.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

**课后练习：**

1、使用 nmcli 显示所有连接

2.、使用 nmcli 显示其中一个活动链接的配置设置

3.、使用 nmcli 查看设备状态

4、 使用 nmcli 删除现有所有链接

5、使用 nmcli 创建一个新连接。该链接使用 DHCP 方式自动获取 ip 地址，网关和 DNS。

6、使用 nmcli 查看新连接获得 ip 地址，网关和 DNS。

7、使用 nmcli 将刚才的新连接改为手动设置，并将刚才通过自动获取获得的 ip 地址，网关和 DNS 改为手工配置。

8、确认配置，使用新配置的地址测试网络是否通信。