Istituto Nazionale di Fisica Nucleare

II Corso RedHat per sistemisti INFN

RHEL/SL/CentOS 7

Networking

Table of contents

- *IPv6 – a fast'n'furious introduction*
- *network device naming*
- *configuring network settings: NetworkManager, nmcli*
- *ip, ss: what about ifconfig, arp, route, netstat?*
- *link aggregation*
- *bridging*

IPv6 addressing

IPv6 address space folklore

- the *IPv6 address space* uses 128-bit address, which means we have a theoretical limit of 2^{128} available addresses
- *"that is 340,282,366,920,938,463,463,374,607,431,768,211,456, or 340 undecillion, 282 decillion, 366 nonillion, 920 octillion, 938 septillion, 463 sextillion, 463 quintillion, 374 quadrillion, 607 trillion, 431 billion, 768 million, 211 thousand and 456 addresses. Which should be just about enough for the* ***Internet of (Insecure Intrusive Gratuitously Connected) Things****."*
- *or ~3,40x10^{38} available addresses,which means ~6,67x10^{23} addresses/m^2*

IPv6 address representation

- Every IPv6 address is divided into 8 16-bit hexadecimal blocks separated by colons:
  - *2001:0d8b:0000:0000:0202:b3ff:fe1e:8329*
- Abbreviations are possible (see **RFC5952**: *A Recommendation for IPv6 Address Text Representation)* : 1) leading 0s in a 16-bit block can be skipped; 2) a double colon can replace consecutive 0s or leading o trailing 0s (but the double colon can appear only once in an address):
  - *2001:d8b:0:0:202:b3ff:fe1e:8329*
  - *2001:d8b::202:b3ff:fe2f:8329*
- 2 special addresses:
  - *loopback*: 0:0:0:0:0:0:0:1 => ::1/128
  - *unspecified*: 0:0:0:0:0:0:0:0 => ::0/128 *(should not be assigned to any host and it should olny be used as the source address by initializing host before it has learned his own address)*

IPv6 addressing model

- IPv6 addresses are *assigned to interfaces*, and are characterized by:
  - a topological scope:
    - interface-local
    - link-local
    - global
    - …
  - a target scope:
    - **unicast**
    - **multicast**
    - **anycast**
  - a lifetime span
    - **static**
    - **temporary**: *valid*, *preferred*, *deprecated*, ...

Topological scope

- **RFC4007**: *IPv6 Scoped Address Architecture*
  - *Every IPv6 address other than the unspecified address has a specific scope; that is, a topological span within which the address may be used as a unique identifier for an interface or set of interfaces. The scope of an address is encoded as part of the address, (…)*
  - *unicast addresses* can have:
    - *interface-local scope, for intra-node (loopback) communication*
    - *link-local scope, for uniquely identifying interfaces within (i.e., attached to) a single link only (**~LAN scope**).*
    - ....
    - *global scope, for uniquely identifying interfaces anywhere in the Internet*

Target scope

- **RFC4291**: *IP Version 6 Addressing Architecture*
  - *3 types of addresses (the type of an address is encoded as part of the address):*
    - ***unicast**: an identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.*
    - ***anycast**: an identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is delivered to one of the interfaces identified by that address (the "nearest" one, according to the routing protocols' measure of distance).*
    - ***multicast**: an identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.*
- *There are no broadcast addresses in IPv6*, their function being superseded by *multicast addresses.*

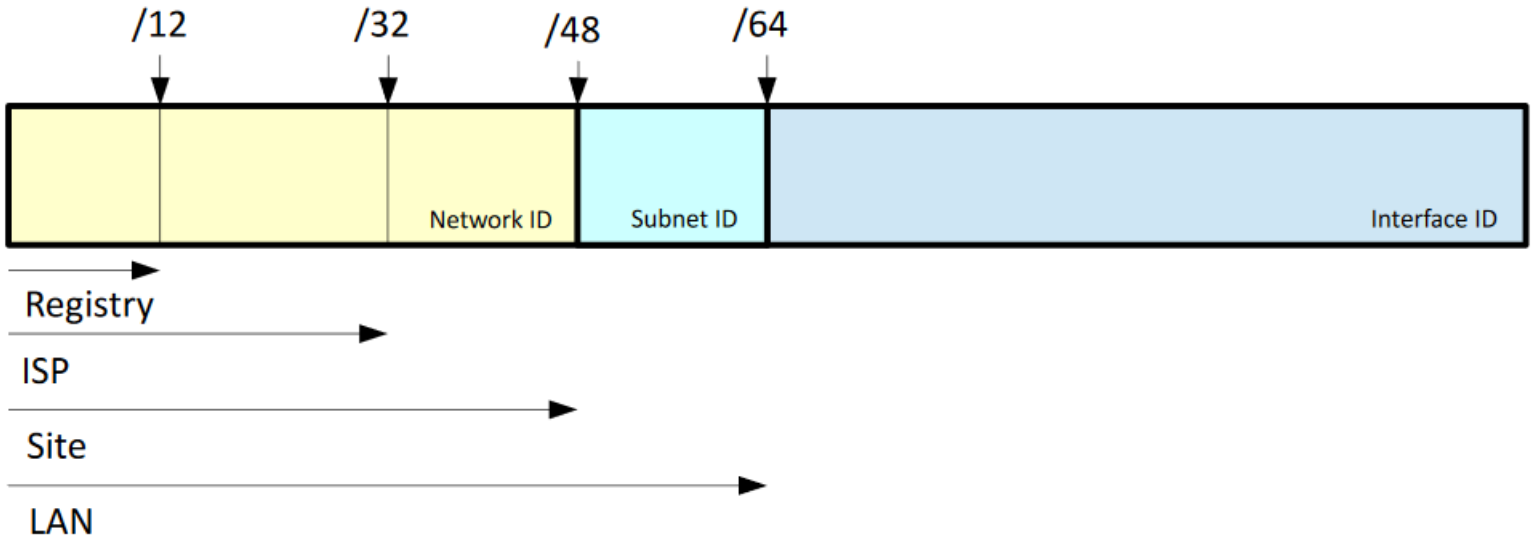
IPv6 addresses/prefixes examples

	binary	hex
unspecified	0000 … 0000	::/128
loopback	0000 … 0001	::1/128
global unicast	0010 …	2000::/3
link-local unicast	1111 1110 10...	FE80::/10
unique local unicast	1111 1100 1111 1101 ...	FC00::/7
multicast	1111 1111…	FF00::/8

IPv6 multicast addresses examples

	interface-local scope	link-local scope	generic form
all-nodes	ff01::1	ff02::1	ff0X::1
all-routers	ff01::2	ff02::2	ff0X::2
…	…	…	…
all-NTP-servers	ff01::101	ff02::101	ff0X::101
…	…	…	…

IPv6 address allocation



AS137 Consortium GARR: 2001:760::/32
2001:760:4000::/40 INFN
2001:760:4200::/48 INFN Aquila
.....
2001:760:4211::/48 INFN Milano-Bicocca
....
2001:760:4225::.48 INFN TIFPA Trento

every INFN end site LAN has room for
- 2^{16} networks with a capacity of
- 2^{64} addresses each...

IPv6 *host-required* addresses

Each IPv6 enabled host must assign the following addresses:

- the loopback address
- link-local address for each active interface
- any assigned global unicast or anycast addresses
- the *all-node* multicast address (*ff02::1*)
- the *solicited-node* multicast address for each interface (*ff02::1:ffxx:yyzz* where *xx*, *yy* and *zz* are taken from interface ID)

=> *all-node* & *solicited-node multicast addresses* are involved, for example, in link-layer address resolution via multicast

IPv6 address configuration options

- stateful
  - manual
  - via DHCPv6
- stateless
  - SLAAC: stateless address autoconfiguration
    - unique interface ID:
      - **static** via modified EUI64
      - **dynamic/temporary**, randomly generated to avoid tracking and enforce privacy
        - » RFC4941, *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*
        - » *RFC7217, A Method for Generating Semantically Opaque Interface Identifiers with IPv6 SLAAC (ie subnet-stable)*
    - network ID (routing prefix): via ***Neighbour Discovery Protocol***

Neighbour Discovery Protocol

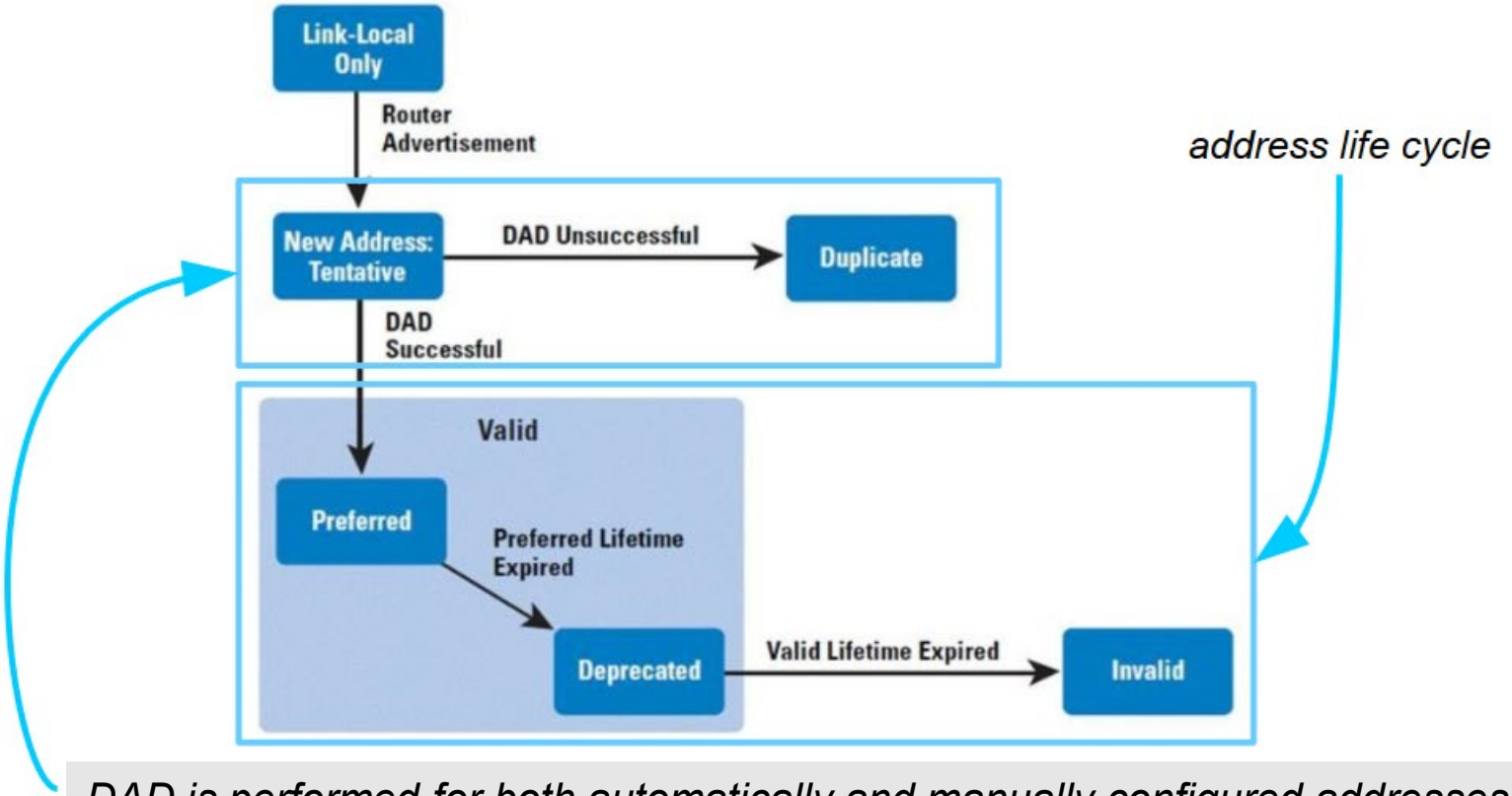
- prefix discovery/notification
- router discovery
- parameter configuration
- address autoconfiguration
- Duplicate Address Detection (DAD)
- address resolution (~ARP)
- Neighbor Unreachability Detection (NUD)

NDP messages

- *Router Solicitation* message (***RS***): sent by a host in order to discover any routers on the link – destination is all-router multicast: ff02::2
- *Router Advertisement* message (***RA***): sent on a regular basis by a router or in response to a RS - carries global prefix information, router preference, router LLA and a few flags to politely suggest address configuration method (stateful via DHCPv6, stateless, stateless with additional parameters via DHCPv6
- *Neighbor Solicitation* message (***NS***): sent by a host to perform LLA resolution (~ARP), DAD during auto-configuration (both for link-local and global addresses) or NUD
- *Neighbor Advertisement* message (***NA***): sent in response to a NS message (solicited NA) or spontaneously (unsolicited NA).

Each node mantains a Neighbor Cache in which all IPv6 and LL addresses of its neighbors are listed in one out of five possible states: INCOMPLETE, REACHABLE, STALE, DELAY, PROBE – see **RFC4861**: *Neighbor Discovery for IPv6*.

SLAAC flowchart



DAD is performed for both automatically and manually configured addresses

IPv6 addresses in linux

carbone@ssire.mib.infn.it:22 - Bitvise xterm - carbone@briseide:~

```
[carbone@briseide ~]$ ip -6 link show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
qlen 1000
    link/ether 52:54:00:a2:0d:0a brd ff:ff:ff:ff:ff:ff
[carbone@briseide ~]$ ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 state UP qlen 1000
    inet6 2001:760:4211:0:d479:6654:5c0a:a003/64 scope global temporary dynamic
       valid_lft 598441sec preferred_lft 79441sec
    inet6 2001:760:4211:0:6d97:b7a4:1686:ab44/64 scope global temporary deprecated dynamic
       valid_lft 512644sec preferred_lft 0sec
    inet6 2001:760:4211::111/64 scope global
       valid_lft forever preferred_lft forever
    inet6 2001:760:4211:0:5054:ff:fea2:d0a/64 scope global mngtmpaddr dynamic
       valid_lft 2591921sec preferred_lft 604721sec
    inet6 fe80::5054:ff:fea2:d0a/64 scope link
       valid_lft forever preferred_lft forever
[carbone@briseide ~]$
```

global (manual)

link-local

global (modified EUI64)

global (random)

DAD @ work

carbone@ssire.mib.infn.it:22 - Bitvise xterm - carbone@ssire:~

```
[carbone@ssire ~]$ ip -6 addr show eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
    inet6 2001:760:4211::100/64 scope global
       valid_lft forever preferred_lft forever
    inet6 fe80::215:e9ff:fe44:7250/64 scope link
       valid_lft forever preferred_lft forever
[carbone@ssire ~]$ host morella
morella.mib.infn.it has address 193.206.157.86
morella.mib.infn.it has IPv6 address 2001:760:4211::110
[carbone@ssire ~]$ sudo ip -6 addr add 2001:760:4211::110/64 dev eth1
[sudo] password for carbone:
[carbone@ssire ~]$ ip -6 addr show eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
    inet6 2001:760:4211::110/64 scope global tentative dadfailed
       valid_lft forever preferred_lft forever
    inet6 2001:760:4211::100/64 scope global
       valid_lft forever preferred_lft forever
    inet6 fe80::215:e9ff:fe44:7250/64 scope link
       valid_lft forever preferred_lft forever
[carbone@ssire ~]$ sudo ip -6 addr del 2001:760:4211::110/64 dev eth1
[carbone@ssire ~]$ ip -6 addr show eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
```

```
::                          ff02::1:ff00:110     78 Neighbor Solicitation for 2001:760:4211::110
2001:760:4211::110          ff02::1              86 Neighbor Advertisement 2001:760:4211::110 (ovr) is at 08:00:27:b8:de:ba
```

```
    inet6 fe80::215:e9ff:fe44:7250/64 scope link
       valid_lft forever preferred_lft forever
[carbone@ssire ~]$
```

real world IPv6 addressing schemes

Managing IPv6 DNS records *and* manually configuring hosts can be a real nightmare. You can register and configure only nodes that must be reached from outside your network (ie nodes exposing *well known services*: DNS, mail, web, … servers) and leave other nodes:

- configure itself via DHCPv6, or a slightly modified version by F. Prelz implementing the so called DA-DA mechanism (*DNS driven Allocation of DHCPv6 Addresses*);
- configure address via SLAAC and learn other network configuration parameters via DHCPv6 (default prefix, DNS servers, domain search list) and RA (default router) – privacy (and entropy ☺) enforced, but an address monitoring tool is mandatory.

Table of contents

- *IPv6 – a fast'n'furious introduction*
- *network device naming*
- *configuring network settings: NetworkManager, nmcli*
- *ip, ss: what about ifconfig, arp, route, netstat?*
- *link aggregation*
- *bridging*

Consistent network device naming

Modern x86-based servers support an increasing number of network interface ports on the motherboard in addition to add-in network adapters. Linux-based OSes name these interfaces as *ethN*. *The naming of network interfaces is currently non-deterministic and not governed by any standard in terms of their relationship to the way the ports are wired on the system.* Common user expectations such as 'eth0' representing the first network port on the motherboard as labeled on the server chassis cannot be fulfilled in many cases.

Ensuring that the Ethernet interface names follow the order of the devices intended by the system designer might not be sufficient. The «ethN» names currently in use do not suggest the Ethernet interface's physical location, whether it is on the system's motehrboard or if it is on an add-in card; or if it is on an add-in card with multiple ports, which porton the card it is on.

Consequently, a naming mechanism that can impart meaning to the network interface's name based on the physical location of a network port in concordance to the intended system design is necessary.

From *«Consistent Network Device Naming in Linux»* by Narendra K –DELL, 2012

Naming schemes hierarchy

By default **systemd** will name network interfaces applying following naming schemes:

- 1 - F/W or BIOS information returned from onboard devices: **`enoxxx`**
- 2 - F/W or BIOS information returned from PCI Express slot card: **`ensxxx`**
- 3 - physical location of the connector (slot address) on the MB: **`enpxxx`**
- 4 – MAC address of the NIC: **`enx00028a`**…
- 5 – if everything fails, fall back to the traditional unpredictable naming scheme: **`ethN`**

Names format

prefix	network type
en	ethernet
wl	wireless LAN

type	hardware type
o<index>	on-board device index number
s<slot>[f<function>][d<dev_id>]	hotplug slot index number
x<mac>	MAC address
[P<domain>]p<bus>s<slot> [f<function>][d<dev_id>]	PCI location – P<domain> mentioned only if not null

biosdevname scheme (DELL ws)

Note that unless the system is a DELL system, or **biosdevname** is explicitly enabled, the **systemd** naming scheme will take precedence.

device	old name	new name
embedded NIC	`eth[012…]`	`em[123…]`
PCI NIC	`eth[012…]`	`p<slot>p<eth_port>`

To disable this feature, pass the option

```
biosdevname=0
```

on the boot command line; to re-enable ths feature pass the option

```
biosedevname=1
```

on the boot command line.

Device naming can be controlled…

- **by identifying and renaming the network device**

  Setting the MAC address of a device in an *ifcfg* file using the **HWADDR** directive enables it to be identified by *udev*; the device name will be taken from the string given by the **DEVICE** directive (left as an exercise).

- **by turning off or on `biosedvname`**

  Turning off biosdevname disable *Consistent Network Device Naming* process – back to good ol' **`ethX`** names.

- **by turning off or on systemd/udev naming scheme**

  You can also supply your own manual naming scheme.

biosdevname=0

- in `/etc/default/grub` add `net.ifnames=0` and `biosdevname=0` to the `GRUB_CMDLINE_LINUX` variable.
- rebuild the /boot/grub2/grub.cfg running the command

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

- modify accordingly the device name by editing the appropriate ifcfg- file, or by running the command

```
# nmcli connection modify <connName> \
  connection.interface-name eth0
```

- reboot and enjoy…!

consistent naming scheme

carbone@seven:/home/carbone

File Edit View Search Terminal Help

```
[root@seven carbone]# udevadm info --query=all /sys/class/net/enp0s3
P: /devices/pci0000:00/0000:00:03.0/net/enp0s3
E: DEVPATH=/devices/pci0000:00/0000:00:03.0/net/enp0s3
E: ID_BUS=pci
E: ID_MM_CANDIDATE=1
E: ID_MODEL_FROM_DATABASE=82540EM Gigabit Ethernet Controller (PRO/1000 MT Desktop Adapter)
E: ID_MODEL_ID=0x100e
E: ID_NET_DRIVER=e1000
E: ID_NET_NAME_MAC=enx080027b82487
E: ID_NET_NAME_PATH=enp0s3
E: ID_OUI_FROM_DATABASE=PCS Systemtechnik GmbH
E: ID_PATH=pci-0000:00:03.0
E: ID_PATH_TAG=pci-0000_00_03_0
E: ID_PCI_CLASS_FROM_DATABASE=Network controller
E: ID_PCI_SUBCLASS_FROM_DATABASE=Ethernet controller
E: ID_VENDOR_FROM_DATABASE=Intel Corporation
E: ID_VENDOR_ID=0x8086
E: IFINDEX=2
E: INTERFACE=enp0s3
E: SUBSYSTEM=net
E: SYSTEMD_ALIAS=/sys/subsystem/net/devices/enp0s3
E: TAGS=:systemd:
E: USEC_INITIALIZED=17742

[root@seven carbone]#
```

old naming scheme

carbone@seven:/home/carbone

File Edit View Search Terminal Help

```
[root@seven carbone]# udevadm info --query=all /sys/class/net/eth0
P: /devices/pci0000:00/0000:00:03.0/net/eth0
E: DEVPATH=/devices/pci0000:00/0000:00:03.0/net/eth0
E: ID_BUS=pci
E: ID_MM_CANDIDATE=1
E: ID_MODEL_FROM_DATABASE=82540EM Gigabit Ethernet Controller (PRO/1000 MT Desktop Adapter)
E: ID_MODEL_ID=0x100e
E: ID_NET_DRIVER=e1000
E: ID_NET_NAME_MAC=enx080027b82487
E: ID_NET_NAME_PATH=enp0s3
E: ID_OUI_FROM_DATABASE=PCS Systemtechnik GmbH
E: ID_PATH=pci-0000:00:03.0
E: ID_PATH_TAG=pci-0000_00_03_0
E: ID_PCI_CLASS_FROM_DATABASE=Network controller
E: ID_PCI_SUBCLASS_FROM_DATABASE=Ethernet controller
E: ID_VENDOR_FROM_DATABASE=Intel Corporation
E: ID_VENDOR_ID=0x8086
E: IFINDEX=2
E: INTERFACE=eth0
E: SUBSYSTEM=net
E: SYSTEMD_ALIAS=/sys/subsystem/net/devices/eth0
E: TAGS=:systemd:
E: UDEV_BIOSDEVNAME=0
E: USEC_INITIALIZED=15076
E: biosdevname=0
E: net.ifnames=0

[root@seven carbone]#
```

Table of contents

the NetworkManager (nm)

- dynamic network control and (automatic) configuration daemon providing methods via *D-Bus* (a standard, generic inter process communication - IPC - framework) for querying status information, changing configuration and dealing with specific trigger events;
- support for traditional `ifcfg-<ifname>` and `network` scripts is maintained - extended functionality (VPN, bridging, …) via *connection profiles.*
- <u>it seems you have no longer to disable NetworkManager (or making interfaces *unmanaged*) as a first administration step in order to have linux networks working consistently ☺</u>

nm and the *network scripts*

network configuration in previous RH/CentOS/… releases used to be carried out by *network scripts* (ns), i.e. the script `/etc/rc.d/init.d/network` and any other installed scripts it calls/refers to (`/etc/sysconfig/network`, `/etc/sysconfig/network-scripts/*`, …). nm is nowadays intended to provide/manage the default networking service/configuration, but it can indeed coexist (and even cooperate, if needed) with *network scripts* – you may have both nm and ns enabled at the same time, or disable nm and enable ns once the network is properly configured and the network configuration is not going to change (actually not recommended, imho).

During boot process `/etc/init.d/network` reads through all the **ifcfg** files and for each one that has `ONBOOT=yes`, it checks whether nm is already starting the **DEVICE** from that **ifcfg** file. If nm is starting that device or has already started it, nothing more is done for that file, and the next `ONBOOT=yes` file is checked. If nm is not yet starting that device, the **initscripts** will continue with their traditional behavior and call **ifup** for that **ifcfg** file.

nm-settings-ifcfg-rh(5)

NetworkManager is based on the concept of connection profiles that contain network configuration (…) The profiles can be stored in various formats. NetworkManager uses plugins for reading and writing the data.

The **ifcfg-rh** plugin is used on the Fedora and RHEL distributions to read/write configuration from/to the traditional **/etc/sysconfig/network-scripts/ifcfg-*** files. Each NetworkManager connection maps to one **ifcfg-*** file, with possible usage of keys-* for passwords, route-* for static IPv4 routes and route6-* for static IPv6 routes. The plugin currently supports reading and writing Ethernet, Wi-Fi, InfiniBand, VLAN, Bond, Bridge, and Team connections. Unsupported connection types (such as WWAN, PPPoE, VPN, or ADSL) are handled by keyfile plugin (nm-settings-keyfile(5)). The main reason for using ifcfg-rh plugin is the compatibility with legacy configurations for ifup and ifdown (initscripts).

nm *auto-default*

By default, nm creates a temporary wired connection for any Ethernet device that is managed and doesn't have a connection configured – adding a NIC to a VM tipically ends up with a temporary DHCP active *'Wired Connection 1'* for which there is no configuration file whatsoever. This behavior is controlled by the parameter `no-auto-default` in the `[main]` section of the NetworkManager configuration file (`NetworkManager.conf` in `/etc/NetworkManager/`). The parameter specify devices for which nm shouldn't create default wired connection - setting it to '*' (`no-auto-default=*`) inhibits the creation of the default connection for every newly created (or activated) network device (see package **`NetworkManager-config-server.noarch`**, which is intended to be installed by default for server deployments).

nm key concepts

nm is based on the concept of *connection profiles*, or *connections*: a connection profile (**cp**) basically contain network configuration *settings* for a specific network device. When **nm** activates a **cp** on a network device the configuration will be applied and an *active* network connection will be established – a network device can have multiple **cp** referring to it, but only one active **cp** at a time.

The connection profiles are handled by **nm** via *settings service* and are exported on D-Bus (*/org/freedesktop/NetworkManager/Settings/<num>* objects).

connection (profile)	A specific, encapsulated, independent group of settings describing all the configuration required to connect to a specific network. It is referred to by a unique identifier called the UUID. A connection is tied to a one specific device type, but not necessarily a specific hardware device. It is composed of one or more Settings objects.
setting	A group of related key/value pairs describing a specific piece of a *connection (profile)*. Keys are also referred to as properties.

nm settings

- connection
- 802-1X
- ADSL
- bluetooth
- bond
- bridge
- bridge-port
- CDMA
- DCB
- dummy
- generic
- GSM
- Infiniband
- IPv4
- IPv6
- ip-tunnel
- ppp
- proxy
- serial
- team
- tun
- vlan
- vpn
- …

nm settings: connection

connection settings – *general cp settings*		
autoconnect	boolean	Whether or not the connection should be automatically connected by NetworkManager when the resources for the connection are available. TRUE to automatically activate the connection, FALSE to require manual intervention to activate the connection.
interface-name	string	The name of the network interface this connection is bound to. If not set, then the connection can be attached to any interface of the appropriate type (subject to restrictions imposed by other settings).
type	string	Base type of the connection. For hardware-dependent connections, should contain the setting name of the hardware-type specific setting (ie, "802-3-ethernet" or "802-11-wireless" or "bluetooth", etc), and for non-hardware dependent connections like VPN or otherwise, should contain the setting name of that setting type (ie, "vpn" or "bridge", etc).
…	…	…

nm settings: ipv6

connection settings – *general cp settings*		
method	string	IP configuration method. NMSettingIP4Config and NMSettingIP6Config both support "auto", "manual", and "link-local".
ip6-privacy	int32	Configure IPv6 Privacy Extensions for SLAAC, described in RFC4941. If enabled, it makes the kernel generate a temporary IPv6 address in addition to the public one generated from MAC address via modified EUI-64.
addr-gen-mode	int32	Configure method for creating the address for use with RFC4862 IPv6 Stateless Address Autoconfiguration. The permitted values are: NM_SETTING_IP6_CONFIG_ADDR_GEN_MODE_EUI64 (0) or NM_SETTING_IP6_CONFIG_ADDR_GEN_MODE_STABLE_PRIVACY (1).
address-data		Array of IPv6 addresses. Each address dictionary contains at least 'address' and 'prefix' entries, containing the IP address as a string, and the prefix length as a uint32.
…	…	…

network configuration

you can configure IPv4/v6 networking by:

- interacting with nm:
  - **`nmtui`**: nm's text user interface tool
  - GNOME GUI: *control-center*, **`nm-connection-editor, nm-applet(s)`**
  - **`nmcli`**: nm's command-line tool
- using **`ip`** command (only volatile, i.e. non-persistent, configuration);
- directly editing `ifcfg-*` network configuration files;

nmtui

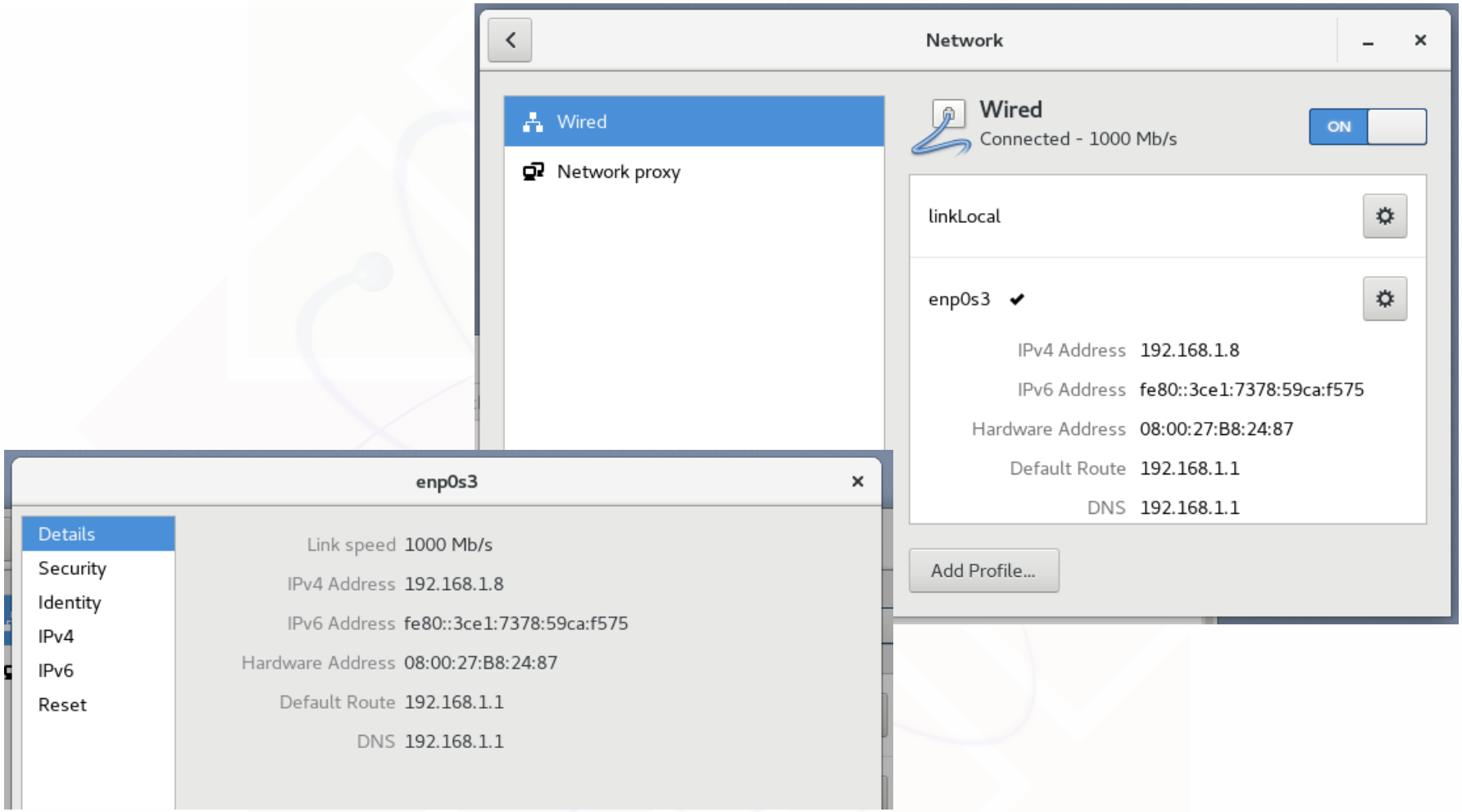
```
NetworkManager TUI

Please select an option

Edit a connection
Activate a connection
Set system hostname

Quit

<OK>
```

```
Ethernet
  enp0s3
  linkLocal
Bridge
  virbr0

<Add>
<Edit...>
<Delete>
```

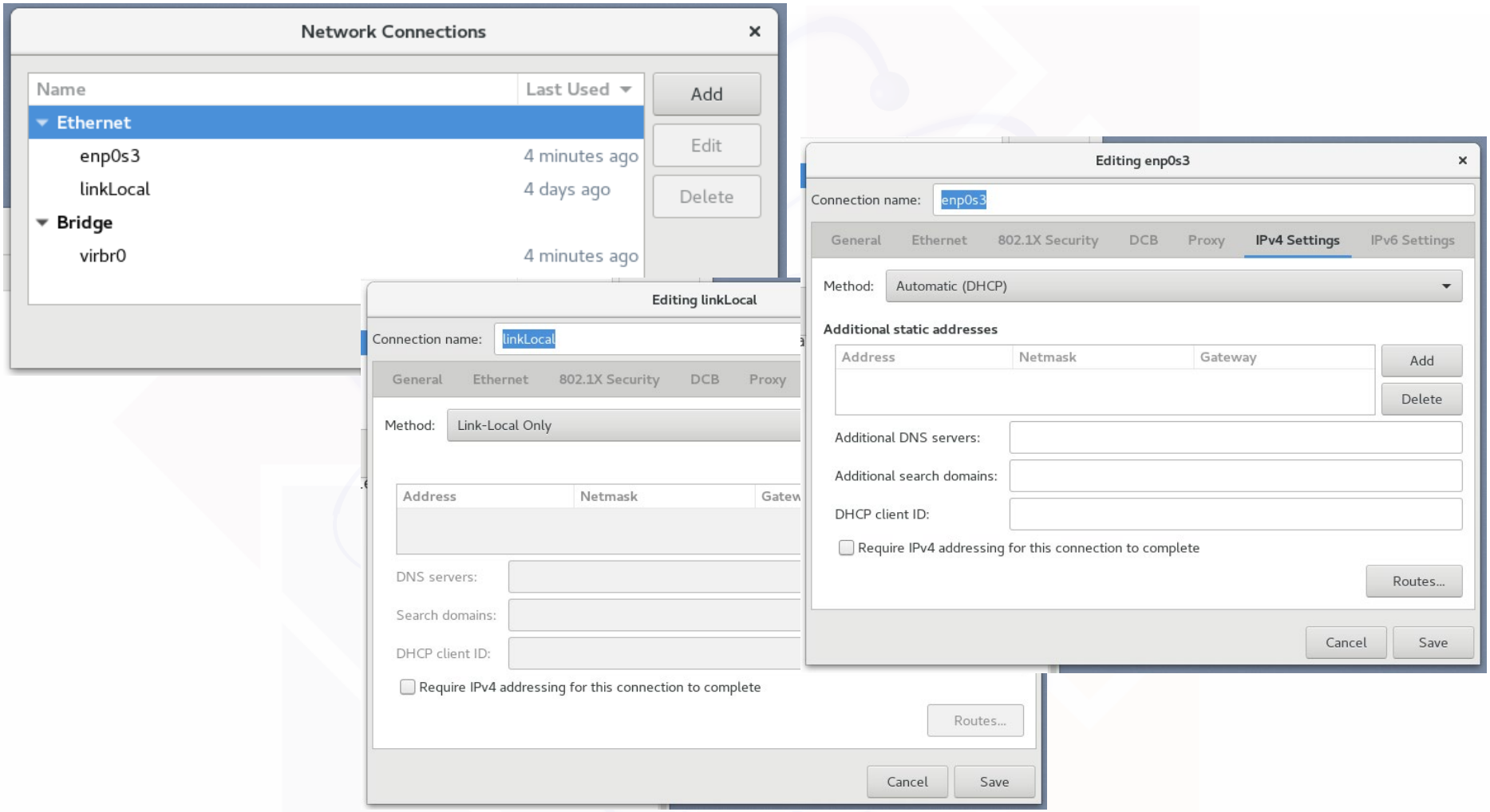
```
New Connection

Select the type of connection you wish to

DSL
Ethernet
InfiniBand
Wi-Fi
Bond
```

```
Edit Connection

Profile name enp0s3
      Device enp0s3 (08:00:27:B8:24:87)

= ETHERNET                                  <Show>

= IPv4 CONFIGURATION                        <Show>
= IPv6 CONFIGURATION                        <Show>

[X] Automatically co
[X] Available to all

Ignore
Automatic
Automatic (DHCP-only)
Link-Local
Manual

<Cancel> <OK>
```

control-center



nm-connection-editor



nmcli

- command-line tool for controlling NetworkManager - can be used both interactively and in batch-mode (scripts);
- works both on in-memory (temporary) and on-disk (persistent) configuration;
- supports
  - command completion (via **`[TAB]`** key)
  - context-sensitive help
  - abbreviations (not recommended in scripts)
- nmcli connection editor has a built-in *describe* command that can display description of particular settings and properties.

nmcli main help

carbone@seven:/home/carbone

File Edit View Search Terminal Help

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

OPTIONS
  -t[erse]                                          terse output
  -p[retty]                                         pretty output
  -m[ode] tabular|multiline                         output mode
  -c[olors] auto|yes|no                             whether to use colors in output
  -f[ields] <field1,field2,...>|all|common          specify fields to output
  -g[et-values] <field1,field2,...>|all|common      shortcut for -m tabular -t -f
  -e[scape] yes|no                                  escape columns separators in values
  -a[sk]                                            ask for missing parameters
  -s[how-secrets]                                   allow displaying passwords
  -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
  a[gent]         NetworkManager secret agent or polkit agent
  m[onitor]       monitor NetworkManager changes

[root@seven carbone]#
```

nmcli context-sensitive help

carbone@seven:/home/carbone

File Edit View Search Terminal Help

```
[root@seven carbone]# nmcli general help
Usage: nmcli general { COMMAND | help }

COMMAND := { status | hostname | permissions | logging }

  status

  hostname [<hostname>]

  permissions

  logging [level <log level>] [domains <log domains>]

[root@seven carbone]# nmcli general status
STATE      CONNECTIVITY  WIFI-HW  WIFI     WWAN-HW  WWAN
connected  full          enabled  enabled  enabled  enabled
[root@seven carbone]# nmcli networking help
Usage: nmcli networking { COMMAND | help }

COMMAND := { [ on | off | connectivity ] }

  on

  off

  connectivity [check]

[root@seven carbone]# nmcli networking connectivity
full
[root@seven carbone]#
```

permissions

carbone@seven:/home/carbone

```
[root@seven carbone]# nmcli general permissions
PERMISSION                                                VALUE
org.freedesktop.NetworkManager.enable-disable-network     yes
org.freedesktop.NetworkManager.enable-disable-wifi        yes
org.freedesktop.NetworkManager.enable-disable-wwan        yes
org.freedesktop.NetworkManager.enable-disable-wimax       yes
org.freedesktop.NetworkManager.sleep-wake                 yes
k-control                                                 yes
hare.protected                                            yes
hare.open                                                 yes
gs.modify.system                                          yes
gs.modify.own                                             yes
gs.modify.hostname                                        yes
gs.modify.global-dns                                      yes
                                                          yes
oint-rollback                                             yes
-disable-statistics                                       yes
```

carbone@seven:~

```
[carbone@seven ~]$ nmcli general permissions
PERMISSION                                                VALUE
org.freedesktop.NetworkManager.enable-disable-network     yes
org.freedesktop.NetworkManager.enable-disable-wifi        yes
org.freedesktop.NetworkManager.enable-disable-wwan        yes
org.freedesktop.NetworkManager.enable-disable-wimax       yes
org.freedesktop.NetworkManager.sleep-wake                 no
org.freedesktop.NetworkManager.network-control            yes
org.freedesktop.NetworkManager.wifi.share.protected       yes
org.freedesktop.NetworkManager.wifi.share.open            yes
org.freedesktop.NetworkManager.settings.modify.system     yes
org.freedesktop.NetworkManager.settings.modify.own        yes
org.freedesktop.NetworkManager.settings.modify.hostname   auth
org.freedesktop.NetworkManager.settings.modify.global-dns auth
org.freedesktop.NetworkManager.reload                     auth
org.freedesktop.NetworkManager.checkpoint-rollback        auth
org.freedesktop.NetworkManager.enable-disable-statistics  yes
[carbone@seven ~]$
```

Console login!!!

Try a remote login and recheck permissions…

permissions

192.168.1.8 - tamigi@seven:~ VT

File Edit Setup Control Window Help

```
[tamigi@seven ~]$ nmcli general permissions
PERMISSION                                                VALUE
org.freedesktop.NetworkManager.enable-disable-network     no
org.freedesktop.NetworkManager.enable-disable-wifi        no
org.freedesktop.NetworkManager.enable-disable-wwan        no
org.freedesktop.NetworkManager.enable-disable-wimax       no
org.freedesktop.NetworkManager.sleep-wake                 no
org.freedesktop.NetworkManager.network-control            auth
org.freedesktop.NetworkManager.wifi.share.protected       no
org.freedesktop.NetworkManager.wifi.share.open            no
org.freedesktop.NetworkManager.settings.modify.system     auth
org.freedesktop.NetworkManager.settings.modify.own        auth
org.freedesktop.NetworkManager.settings.modify.hostname   auth
org.freedesktop.NetworkManager.settings.modify.global-dns auth
org.freedesktop.NetworkManager.reload                     auth
org.freedesktop.NetworkManager.checkpoint-rollback        auth
org.freedesktop.NetworkManager.enable-disable-statistics  no
[tamigi@seven ~]$
```

devices & connections

carbone@seven:/etc/sysconfig/network-scripts

File Edit View Search Terminal Help

```
[root@seven network-scripts]# nmcli device
DEVICE      TYPE      STATE      CONNECTION
virbr0      bridge    connected  virbr0
enp0s3      ethernet  connected  enp0s3
lo          loopback  unmanaged  --
virbr0-nic  tun       unmanaged  --
[root@seven network-scripts]# nmcli connection show
NAME        UUID                                  TYPE            DEVICE
enp0s3      578ae379-03be-44b8-905d-6c336a2b9793  802-3-ethernet  enp0s3
virbr0      ed48d69f-5dbf-4a6d-94d6-07206228061e  bridge          virbr0
linkLocal   e13541c3-f557-41c0-908a-b48ce6979286  802-3-ethernet  --
[root@seven network-scripts]# ls -l ifcfg-*
-rw-r--r--. 1 root root 282 Mar 31 23:43 ifcfg-enp0s3
-rw-r--r--. 1 root root 322 Mar 29 22:33 ifcfg-linkLocal
-rw-r--r--. 1 root root 254 May  3  2017 ifcfg-lo
[root@seven network-scripts]# cat ifcfg-linkLocal
HWADDR=08:00:27:B8:24:87
MACADDR=08:00:27:B8:24:87
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=autoip
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=no
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=linkLocal
UUID=e13541c3-f557-41c0-908a-b48ce6979286
ONBOOT=no
[root@seven network-scripts]#
```

connections

carbone@seven:~

```
[carbone@seven ~]$ nmcli -f NAME,TYPE,DEVICE,AUTOCONNECT,STATE,SLAVE connection show
NAME        TYPE            DEVICE  AUTOCONNECT  STATE      SLAVE
enp0s3      802-3-ethernet  enp0s3  yes          activated  --
virbr0      bridge          virbr0  no           activated  --
linkLoc
[carbon
```

carbone@seven:~

```
[carbone@seven ~]$ nmcli -f ipv4,IP4 connection show enp0s3
ipv4.method:                            auto
ipv4.dns:                               --
ipv4.dns-search:                        --
ipv4.dns-options:                       (default)
ipv4.dns-priority:                      0
ipv4.addresses:                         --
ipv4.gateway:                           --
ipv4.routes:                            --
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-timeout:                      0
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.dhcp-fqdn:                         --
ipv4.never-default:                     no
ipv4.may-fail:                          yes
ipv4.dad-timeout:                       -1 (default)
IP4.ADDRESS[1]:                         192.168.1.8/24
IP4.GATEWAY:                            192.168.1.1
IP4.DNS[1]:                             192.168.1.1
IP4.DOMAIN[1]:                          station
[carbone@seven ~]$
```

tamigi@seven:~

```
te DOWN qlen 1000
    link/ether 52:54:00:aa:73:00 brd ff:ff:ff:ff:ff:ff
[tamigi@seven ~]$ nmcli -f ipv6,IP6 connection show enp0s3
ipv6.method:                            auto
ipv6.dns:                               --
ipv6.dns-search:                        --
ipv6.dns-options:                       (default)
ipv6.dns-priority:                      0
ipv6.addresses:                         --
ipv6.gateway:                           --
ipv6.routes:                            --
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          yes
ipv6.ip6-privacy:                       -1 (unknown)
ipv6.addr-gen-mode:                     stable-privacy
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
ipv6.token:                             --
IP6.ADDRESS[1]:                         fe80::3ce1:7378:59ca:f575/64
IP6.GATEWAY:                            --
[tamigi@seven ~]$
```

connections

```
[carbone@seven ~]$ nmcli -f ipv4,IP4 connection show enp0s3
ipv4.method:                            manual
ipv4.dns:                               193.206.157.1,193.206.157.2
ipv4.dns-search:                        mib.infn.it
ipv4.dns-options:                       (default)
ipv4.dns-priority:                      0
ipv4.addresses:                         193.206.157.70/23
ipv4.gateway:                           193.206.157.254
ipv4.routes:                            --
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-timeout:                      0
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.dhcp-fqdn:                         --
ipv4.never-default:                     no
ipv4.may-fail:                          no
ipv4.dad-timeout:                       -1 (default)
IP4.ADDRESS[1]:                         193.206.157.70/23
IP4.GATEWAY:                            193.206.157.254
IP4.DNS[1]:                             193.206.157.1
IP4.DNS[2]:                             193.206.157.2
[carbone@seven ~]$
```

ipv4, ipv6: ***connection*** properties
IP4, IP6: ***active connection*** properties

```
[carbone@seven ~]$ nmcli -f ipv6,IP6 connection show enp0s3
ipv6.method:                            manual
ipv6.dns:                               2001:760:4211::1
ipv6.dns-search:                        --
ipv6.dns-options:                       (default)
ipv6.dns-priority:                      0
ipv6.addresses:                         2001:760:4211::112/64
ipv6.gateway:                           2001:760:4211::254
ipv6.routes:                            --
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          no
ipv6.ip6-privacy:                       0 (disabled)
ipv6.addr-gen-mode:                     stable-privacy
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
ipv6.token:                             --
IP6.ADDRESS[1]:                         2001:760:4211::112/64
IP6.ADDRESS[2]:                         fe80::af64:9eea:802c:dc92/64
IP6.GATEWAY:                            2001:760:4211::254
IP6.DNS[1]:                             2001:760:4211::1
[carbone@seven ~]$
```

creating a connection

Mate Terminal

File Edit View Search Terminal Help

```
[root@seven carbone]# nmcli device
DEVICE      TYPE      STATE      CONNECTION
virbr0      bridge    connected  virbr0
enp0s3      ethernet  connected  enp0s3
enp0s8      ethernet  connected  enp0s8
lo          loopback  unmanaged  --
virbr0-nic  tun       unmanaged  --
[root@seven carbone]# nmcli connection
NAME    UUID                                  TYPE            DEVICE
enp0s3  9f7dd5a1-faa7-44ee-9b66-099b45e02500  802-3-ethernet  enp0s3
enp0s8  672d7329-fbea-4ca8-9384-80ad46a2d07f  802-3-ethernet  enp0s8
virbr0  9b026261-61af-4c47-9eeb-fa8364da657d  bridge          virbr0
[root@seven carbone]# nmcli -f IP4,IP6 connection show enp0s8
IP4.ADDRESS[1]:                         169.254.189.240/16
IP4.GATEWAY:                            --
IP4.ROUTE[1]:                           dst = 224.0.0.0/4, nh = 0.0.0.0, mt = 100
IP6.ADDRESS[1]:                         fe80::bb77:24:43ff:e88c/64
IP6.GATEWAY:                            --
[root@seven carbone]#
```

creating a connection

```
# nmcli connection down enp0s8
Connection 'enp0s8' successfully deactivated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/5)
# nmcli connection add con-name enp0s8-dyn \
        ifname enp0s8 type ethernet \
        ipv4.method auto
Connection 'enp0s8-dyn' (57c95cb8-3c4a-4c7d-a18f-9d5ca05c3db0)
successfully added.
```

The new connection becomes active as soon as the **`nmcli c add`** command completes.

Note that the **`nmcli c down`** command deactivates a connection without preventing the connection itself from further auto-activation; use the **`nmcli device disconnect`** to prevent the device from automatically reactivating further connections without manual intervention.

creating a connection

Mate Terminal

File Edit View Search Terminal Help

```
[root@seven carbone]# nmcli connection
NAME        UUID                                  TYPE            DEVICE
enp0s3      9f7dd5a1-faa7-44ee-9b66-099b45e02500  802-3-ethernet  enp0s3
enp0s8-dyn  57c95cb8-3c4a-4c7d-a18f-9d5ca05c3db0  802-3-ethernet  enp0s8
virbr0      9b026261-61af-4c47-9eeb-fa8364da657d  bridge          virbr0
enp0s8      672d7329-fbea-4ca8-9384-80ad46a2d07f  802-3-ethernet  --
[root@seven carbone]# nmcli -f IP4,IP6 connection show enp0s8-dyn
IP4.ADDRESS[1]:                         172.18.12.40/16
IP4.GATEWAY:                            172.18.1.1
IP4.DNS[1]:                             193.206.157.1
IP4.DNS[2]:                             193.206.157.2
IP4.DOMAIN[1]:                          mib.infn.it
IP4.DOMAIN[2]:                          mi.infn.it
IP6.ADDRESS[1]:                         2001:760:4211:0:18b6:13f6:3ac:47cc/64
IP6.ADDRESS[2]:                         fe80::eda5:5507:a967:d308/64
IP6.GATEWAY:                            fe80::226:bff:fe35:ae40
IP6.ROUTE[1]:                           dst = 2001:760:4211::/64, nh = ::, mt = 100
IP6.DNS[1]:                             2001:760:4211::1
[root@seven carbone]#
```

configuring static addresses

```
# nmcli connection add type ethernet \
        con-name enp0s3 ifname enp0s3 \
        ip4 192.168.100.1 gw4 192.168.200.254
Connection 'enp0s3' (20d7e3be-db21-4317-a013-03b2ba51f1bc)
successfully added.
```

nm will 1) automatically set its internal parameters `ipv4.method` to `manual` and `connection.autoconnect` to `yes`; 2) write out the settings to the corresponding `ifcfg-` file.

The same for IPv6 networks:

```
# nmcli connection add type ethernet \
        con-name enp0s3 ifname enp0s3 \
        ip6 <ipv6Addr> gw6 <ipv6gwAddr>
```

modifying a connection

useven.mib.infn.it - carbone@useven:~ VT

File Edit Setup Control Window Help

```
[carbone@useven ~]$ nmcli -f ipv4,IP4 connection show enp0s3
ipv4.method:                            manual
ipv4.dns:                               193.206.157.1,193.206.157.2
ipv4.dns-search:                        mib.infn.it
ipv4.dns-options:                       (default)
ipv4.dns-priority:                      0
ipv4.addresses:                         193.206.157.158/23
ipv4.gateway:                           193.206.157.254
ipv4.routes:                            --
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-timeout:                      0
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.dhcp-fqdn:                         --
ipv4.never-default:                     no
ipv4.may-fail:                          no
ipv4.dad-timeout:                       -1 (default)
IP4.ADDRESS[1]:                         193.206.157.158/23
IP4.GATEWAY:                            193.206.157.254
IP4.DNS[1]:                             193.206.157.1
IP4.DNS[2]:                             193.206.157.2
[carbone@useven ~]$
```

useven.mib.infn.it - carbone@useven:~ VT

File Edit Setup Control Window Help

```
[carbone@useven ~]$ nmcli device show enp0s3
GENERAL.DEVICE:                         enp0s3
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         08:00:27:85:A9:B4
GENERAL.MTU:                            1500
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     enp0s3
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveConnection/3
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         193.206.157.158/23
IP4.GATEWAY:                            193.206.157.254
IP4.DNS[1]:                             193.206.157.1
IP4.DNS[2]:                             193.206.157.2
IP6.ADDRESS[1]:                         2001:760:4211::113/64
IP6.ADDRESS[2]:                         fe80::8796:f6cf:8cdc:daa4/64
IP6.GATEWAY:                            2001:760:4211::254
IP6.DNS[1]:                             2001:760:4211::1
[carbone@useven ~]$
```

modifying a connection

```
# nmcli conn mod enp0s3 +ipv4.addresses 192.168.100.80/24
# nmcli -f ipv4.addresses,IP4.ADDRESS con show enp0s3
ipv4.addresses:   193.206.157.158/23, 192.168.100.80/24
IP4.ADDRESS[1]:   193.206.157.158/23
# nmcli con up enp0s3
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/4)
# nmcli -f ipv4.addresses,IP4.ADDRESS con show enp0s3
ipv4.addresses:   193.206.157.158/23, 192.168.100.80/24
IP4.ADDRESS[1]:   192.168.100.80/24
IP4.ADDRESS[2]:   193.206.157.158/23
```

Adding a static route:

```
# nmcli con mod enp0s8 +ipv4.routes "<net/mask> <gw>"
```

*** *static routes in* `/etc/sysconfig/network-scripts/routes-enp0s8`

the connection editor

useven.mib.infn.it - carbone@useven:/etc/sysconfig/network-scripts VT

File Edit Setup Control Window Help

```
[carbone@useven network-scripts]$ sudo nmcli connection edit enp0s3

===| nmcli interactive connection editor |===

Editing existing '802-3-ethernet' connection: 'enp0s3'

Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-3-ethernet (ethernet), 802-1x, dcb, ipv4, ipv6,
 proxy
nmcli> help
------------------------------------------------------------------------------
---[ Main menu ]---
goto     [<setting> | <prop>]         :: go to a setting or property
remove   <setting>[.<prop>] | <prop>  :: remove setting or reset property value
set      [<setting>.<prop> <value>]   :: set property value
describe [<setting>.<prop>]           :: describe property
print    [all | <setting>[.<prop>]]   :: print the connection
verify   [all | fix]                  :: verify the connection
save     [persistent|temporary]       :: save the connection
activate [<ifname>] [/<ap>|<nsp>]     :: activate the connection
back                                  :: go one level up (back)
help/?   [<command>]                  :: print this help
nmcli    <conf-option> <value>        :: nmcli configuration
quit                                  :: exit nmcli
------------------------------------------------------------------------------
nmcli>
```

the connection editor

useven.mib.infn.it - carbone@useven:/etc/sysconfig/network-scripts VT

File Edit Setup Control Window Help

```
nmcli> describe ipv4.dad-timeout

=== [dad-timeout] ===
[NM property description]
Timeout in milliseconds used to check for the presence of duplicate IP addresses on the network.  If
 an address conflict is detected, the activation will fail.  A zero value means that no duplicate ad
dress detection is performed, -1 means the default value (either configuration ipvx.dad-timeout over
ride or 3 seconds).  A value greater than zero is a timeout in milliseconds.

nmcli> describe ipv6.ip6-privacy

=== [ip6-privacy] ===
[NM property description]
Configure IPv6 Privacy Extensions for SLAAC, described in RFC4941.  If enabled, it makes the kernel
generate a temporary IPv6 address in addition to the public one generated from MAC address via modif
ied EUI-64.  This enhances privacy, but could cause problems in some applications, on the other hand
.  The permitted values are: -1: unknown, 0: disabled, 1: enabled (prefer public address), 2: enable
d (prefer temporary addresses). Having a per-connection setting set to "-1" (unknown) means fallback
 to global configuration "ipv6.ip6-privacy". If also global configuration is unspecified or set to "
-1", fallback to read "/proc/sys/net/ipv6/conf/default/use_tempaddr". Note that this setting is dist
inct from the Stable Privacy addresses that can be enabled with the "addr-gen-mode" property's "stab
le-privacy" setting as another way of avoiding host tracking with IPv6 addresses.

nmcli>
```

the connection editor – IPv6

seven.mib.infn.it - root@seven:/home/carbone VT

File Edit Setup Control Window Help

```
[root@seven carbone]# nmcli -f ipv6,IP6 connection show enp0s3
ipv6.method:                            link-local
ipv6.dns:                               --
ipv6.dns-search:                        --
ipv6.dns-options:                       (default)
ipv6.dns-priority:                      0
ipv6.addresses:                         --
ipv6.gateway:                           --
ipv6.routes:                            --
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          no
ipv6.ip6-privacy:                       0 (disabled)
ipv6.addr-gen-mode:                     stable-privacy
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
ipv6.token:                             --
IP6.ADDRESS[1]:                         fe80::af64:9eea:802c:dc92/64
IP6.GATEWAY:                            --
[root@seven carbone]#
```

the connection editor – IPv6

seven.mib.infn.it - root@seven:/home/carbone VT

File Edit Setup Control Window Help

```
[root@seven carbone]# nmcli connection edit enp0s3

===| nmcli interactive connection editor |===

Editing existing '802-3-ethernet' connection: 'enp0s3'

Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-3-ethernet (ethernet), 802-1x, dcb, ipv4, ipv6,
 proxy
nmcli> set ipv6.method auto
nmcli> set ipv6.addresses 2001:760:4211::112
Do you also want to set 'ipv6.method' to 'manual'? [yes]: no
nmcli> set ipv6.ip6-privacy 2
nmcli> save
Connection 'enp0s3' (9f7dd5a1-faa7-44ee-9b66-099b45e02500) successfully updated.
nmcli> [root@seven carbone]#
[root@seven carbone]# nmcli connection up enp0s3
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnecti
on/15)
[root@seven carbone]#
```

note: exit with [CTRL]-D
DO NOT USE 'quit'

the connection editor – IPv6

seven.mib.infn.it - root@seven:/home/carbone VT

File Edit Setup Control Window Help

```
[root@seven carbone]# nmcli -f ipv6,IP6 connection show enp0s3
ipv6.method:                            auto
ipv6.dns:                               --
ipv6.dns-search:                        --
ipv6.dns-options:                       (default)
ipv6.dns-priority:                      0
ipv6.addresses:                         2001:760:4211::112/128
ipv6.gateway:                           --
ipv6.routes:                            --
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          no
ipv6.ip6-privacy:                       2 (enabled, prefer temporary IP)
ipv6.addr-gen-mode:                     stable-privacy
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
ipv6.token:                             --
IP6.ADDRESS[1]:                         2001:760:4211::112/128
IP6.ADDRESS[2]:                         2001:760:4211:0:21a2:2545:879c:70d2/64
IP6.ADDRESS[3]:                         2001:760:4211:0:e38a:3bb1:e0e:4c62/64
IP6.ADDRESS[4]:                         fe80::af64:9eea:802c:dc92/64
IP6.GATEWAY:                            fe80::226:bff:fe35:ae40
IP6.ROUTE[1]:                           dst = 2001:760:4211::/64, nh = ::, mt = 100
IP6.DNS[1]:                             2001:760:4211::1
[root@seven carbone]#
```

Server Port: **80 / HTTP**

Your browser / system (or proxy) supports:

IPv4	IPv6	Precedence
✓	✓	IPv6
193.206.157.70	2001:760:4211:0:21a2:2545:879c:70d2	

'?' means missing support or no automatic load of images

ip address show

seven.mib.infn.it - root@seven:/home/carbone VT

File Edit Setup Control Window Help

```
[root@seven carbone]# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:ff:3e:d5 brd ff:ff:ff:ff:ff:ff
    inet 193.206.157.70/23 brd 193.206.157.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 2001:760:4211:0:21a2:2545:879c:70d2/64 scope global temporary dynamic
       valid_lft 604216sec preferred_lft 85216sec
    inet6 2001:760:4211:0:e38a:3bb1:e0e:4c62/64 scope global mngtmpaddr noprefixroute dynamic
       valid_lft 2591933sec preferred_lft 604733sec
    inet6 2001:760:4211::112/128 scope global
       valid_lft forever preferred_lft forever
    inet6 fe80::af64:9eea:802c:dc92/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:ff:f7:36 brd ff:ff:ff:ff:ff:ff
    inet 169.254.189.240/16 brd 169.254.255.255 scope link enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::bb77:24:43ff:e88c/64 scope link
       valid_lft forever preferred_lft forever
4: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN qlen 1000
    link/ether 52:54:00:7d:1d:14 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
5: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN qlen 1000
    link/ether 52:54:00:7d:1d:14 brd ff:ff:ff:ff:ff:ff
[root@seven carbone]#
```

IPv6 address configuration

```
ipv6 unicast-routing
ipv6 dhcp pool MIB_IPV6_POOL
 dns-server 2001:760:4211::1
 domain-name mib.infn.it

interface Vlan1
 description Rete Locale
 ipv6 address 2001:760:4211::254/64
 ipv6 nd other-config-flag
 ipv6 nd router-preference High
 ipv6 dhcp server MIB_IPV6_POOL
```

ciscocca conf snippet

Dalla RFC4861 - Neighbor Discovery for IP version 6 (IPv6):

```
M              1-bit "Managed address configuration" flag.  When
               set, it indicates that addresses are available via
               Dynamic Host Configuration Protocol [DHCPv6].

               If the M flag is set, the O flag is redundant and
               can be ignored because DHCPv6 will return all
               available configuration information.

O              1-bit "Other configuration" flag.  When set, it
               indicates that other configuration information is
               available via DHCPv6.  Examples of such information
               are DNS-related information or information on other
               servers within the network.

  Note: If neither M nor O flags are set, this indicates that no
        information is available via DHCPv6.
```

Managed flag	Other flag	auto address	default route	DNS
on	off	ok	ok	nope (dhclient: **NOADDRS-AVAIL**)
off	on	ok	ok	ok (dhclient: **srv, domain**)
off	off	ok	ok	nope (dhclient **doesn't start at all**)

Network parameter	DHCPv4	DHCPv6	SLAAC	RA
address	yes	yes	yes	
default router	yes			yes
name server	yes	yes		*RFC6106, seldom implemented*
domain search list	yes	yes		*RFC6106, seldom implemented*

MiB IPv6 wiki

IPv6 tokenized interface ID

IPv6 tokenized interface identifer support is used for assigning well-known host-part addresses to nodes whilst still obtaining a global network prefix from Router Advertisements (RA). The primary target for tokenized identifiers are server platforms where addresses are usually manually configured, rather than using DHCPv6 or SLAAC. By using tokenized identifiers, hosts can still determine their network prefix by use of SLAAC, but more readily be automatically renumbered should their network prefix change [1]. Tokenized IPv6 Identifiers are described in the draft [1]: <draft-chown-6man-tokenised-ipv6-identifiers-02>.

```
[ssire]$ host useven.mib.infn.it
useven.mib.infn.it has address 193.206.157.158
useven.mib.infn.it has IPv6 address 2001:760:4211::113
```

network prefix

token

IPv6 token @ work - nmcli

Mate Terminal

File Edit View Search Terminal Help

```
[root@useven carbone]# nmcli -f ipv6,IP6 connection show enp0s3
ipv6.method:                            auto
ipv6.dns:                               --
ipv6.dns-search:                        --
ipv6.dns-options:                       (default)
ipv6.dns-priority:                      0
ipv6.addresses:                         --
ipv6.gateway:                           --
ipv6.routes:                            --
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          yes
ipv6.ip6-privacy:                       2 (enabled, prefer temporary IP)
ipv6.addr-gen-mode:                     eui64
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
ipv6.token:                             ::113
IP6.ADDRESS[1]:                         2001:760:4211:0:ec72:e1b9:2b74:c4c7/64
IP6.ADDRESS[2]:                         2001:760:4211::113/64
IP6.ADDRESS[3]:                         fe80::a00:27ff:fe85:a9b4/64
IP6.GATEWAY:                            fe80::226:bff:fe35:ae40
IP6.ROUTE[1]:                           dst = 2001:760:4211::/64, nh = ::, mt = 100
IP6.DNS[1]:                             2001:760:4211::1
[root@useven carbone]#
```

IPv6 token @ work - ip

Mate Terminal

File Edit View Search Terminal Help

```
[root@useven carbone]# ip addr show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:85:a9:b4 brd ff:ff:ff:ff:ff:ff
    inet 193.206.157.158/23 brd 193.206.157.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet 192.168.100.80/24 brd 192.168.100.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 2001:760:4211:0:ec72:e1b9:2b74:c4c7/64 scope global temporary dynamic
       valid_lft 604487sec preferred_lft 85487sec
    inet6 2001:760:4211::113/64 scope global mngtmpaddr noprefixroute dynamic
       valid_lft 2591999sec preferred_lft 604799sec
    inet6 fe80::a00:27ff:fe85:a9b4/64 scope link
       valid_lft forever preferred_lft forever
[root@useven carbone]# ip -6 route list type unicast
2001:760:4211::/64 dev enp0s3 proto ra metric 100
fe80::226:bff:fe35:ae40 dev enp0s3 proto static metric 100
fe80::/64 dev enp0s3 proto kernel metric 256
default via fe80::226:bff:fe35:ae40 dev enp0s3 proto static metric 100
[root@useven carbone]#
```

nmcli on laptops

Mate Terminal

File Edit View Search Terminal Help

```
[carbone@ricotta ~]$ nmcli connection
NAME                    UUID                                  TYPE             DEVICE
Vodafone-34067780       9b38667c-1607-4b64-8617-1f0b768905a2  802-11-wireless  wlp1s0
virbr0                  191e4e5d-5830-45d7-9118-695d3f8c2b5a  bridge           virbr0
CNAF-dot1x              0e189265-666e-40cf-9147-c8c65a9c151d  802-11-wireless  --
FASTWEB-1-2Pe3ZGZF3FPt  673524c7-3bed-4a68-8300-3742cd9c5ad9  802-11-wireless  --
FASTWEB-1-UwmdTvszasEZ  53ed1b3f-f259-4e92-9a07-33198d869c44  802-11-wireless  --
Giobatta Network        f8dbf59e-cad5-4186-946c-614ca8d7687e  bluetooth        --
INFN-Web                ecf4e1c0-86a3-465e-936a-281aaa6ca481  802-11-wireless  --
INFN-dot1x              fd54a8a6-15e9-45b3-b289-6097a235df06  802-11-wireless  --
MobileWiFi-2319         7098e35e-1cce-4939-96c4-47b49593438d  802-11-wireless  --
bifrost LDAP            6836aa63-2c42-4f7b-b161-afe0e2333960  vpn              --
bifrost X509            121a1908-1769-4166-ba69-ffa6363ed535  vpn              --
eduroam                 99be2f34-3e36-44f8-8865-96feebf89479  802-11-wireless  --
enp2s0                  b13b0e81-2787-440c-9a77-1d45fccad8aa  802-3-ethernet   --
hermitage               5d0da762-71be-43c8-a7f1-3ee90340c0c4  802-11-wireless  --
sansone                 89a49c30-c0ea-4e41-8f64-1cac13ccd6a7  vpn              --
sansone LDAP            8dc9213e-0259-4c4c-bb6a-c10ebdee7ca7  vpn              --
[carbone@ricotta ~]$
```

nmcli on laptops – wifi support

Mate Terminal

File Edit View Search Terminal Help

```
[carbone@ricotta ~]$ nmcli device wifi list
*  SSID                    MODE   CHAN  RATE       SIGNAL  BARS  SECURITY
   Vodafone-WiFi           Infra  10    54 Mbit/s  79      ▂▄▆▇  --
   TISCALI-A20B55          Infra  6     54 Mbit/s  62      ▂▄▆▇  WPA2
*  Vodafone-34067780       Infra  10    54 Mbit/s  58      ▂▄▆▇  WPA2
   Vodafone-WiFi           Infra  52    54 Mbit/s  52      ▂▄▆_  --
   Vodafone-34067780       Infra  52    54 Mbit/s  52      ▂▄▆_  WPA2
   FASTWEB-U2XJKW          Infra  52    54 Mbit/s  47      ▂▄▆_  WPA2
   TISCALI5G-A20B5D        Infra  100   54 Mbit/s  37      ▂▄▆_  WPA2
   InfostradaWiFi-502682   Infra  9     54 Mbit/s  35      ▂▄▆_  WPA1 WPA2
   L+Perm-Mi               Infra  9     54 Mbit/s  35      ▂▄▆_  WPA1
   Edda&Sergio             Infra  11    54 Mbit/s  32      ▂▄▆_  WPA2
   Vodafone-WiFi           Infra  13    54 Mbit/s  32      ▂▄▆_  --
   FASTWEB-U2XJKW          Infra  1     54 Mbit/s  30      ▂___  WPA2
   Vodafone-34520468       Infra  13    54 Mbit/s  30      ▂___  WPA2
   NETGEAR88               Infra  1     54 Mbit/s  29      ▂___  WPA2
   Vodafone2.4GHz-zhao     Infra  1     54 Mbit/s  27      ▂___  WPA2
   FASTWEB-1-464835        Infra  1     54 Mbit/s  27      ▂___  WPA1 WPA2
   Vodafone-22233686       Infra  12    54 Mbit/s  27      ▂___  WPA2
   witourist.326           Infra  6     54 Mbit/s  25      ▂___  WPA2
   --                      Infra  9     54 Mbit/s  25      ▂___  --
   Telecom-74035541        Infra  6     54 Mbit/s  24      ▂___  WPA1 WPA2
   Telecom-73850545        Infra  6     54 Mbit/s  24      ▂___  WPA1 WPA2
   Vodafone-WiFi           Infra  13    54 Mbit/s  20      ▂___  --
   Vodafone-34777587       Infra  13    54 Mbit/s  20      ▂___  WPA2
   vodafone01              Infra  13    54 Mbit/s  19      ▂___  WPA1 WPA2
   Vodafone-WiFi           Infra  13    54 Mbit/s  19      ▂___  --
[carbone@ricotta ~]$
```

NM tips & tricks

lightweight environments (or if you really don't trust NM…): if your configuration is mainly static and you don't want your system to react to dbus or udev events do not disable NM but just use the

`configure-and-quit = true`

option - with this in place NetworkManager will carry out the configuration of the interface and then gracefully exit leaving the network up as desired and notifying systemd that networking is up but without the NM daemon left running in the background listening for, and responding to, udev, dbus or similar events.

```
$ cat /etc/NetworkManager/conf.d/conf-and-quit.conf
[main]
configure-and-quit = true
```

NM tips & tricks – cont'd

automatically respond to configuration files changes (somewhat dangerous): the default behaviour for NetworkManager is to only respond to changes to configuration files on a restart or when directed to through **`nmcli conn reload`**. However, if a more dynamic response to changes to the files is preferred it is possible to have NM monitor all the configuration files and have it immediately make changes to network state from this by using the option

`monitor-connection-files = true`

with this option as any changes to the files will be immediately applied which could quite easily break network connectivity if the wrong thing is put in place.

```
$ cat /etc/NetworkManager/conf.d/monitor-files.conf
[main]
monitor-connection-files = true
```

Table of contents

- *IPv6 – a fast'n'furious introduction*
- *network device naming*
- *configuring network settings: NetworkManager, nmcli*
- *ip, ss: what about ifconfig, arp, route, netstat?*
- *link aggregation*
- *bridging*

ip – man page

carbone@seven:~

File Edit View Search Terminal Help

```
IP(8)                                Linux                                IP(8)

NAME
       ip - show / manipulate routing, devices, policy routing and tunnels

SYNOPSIS
       ip [ OPTIONS ] OBJECT { COMMAND | help }

       ip [ -force ] -batch filename

       OBJECT := { link | address | addrlabel | route | rule | neigh | ntable
                 | tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm
                 | netns | l2tp | tcp_metrics | token }

       OPTIONS := { -V[ersion] | -h[uman-readable] | -s[tatistics] |
                 -d[etails] | -r[esolve] | -iec | -f[amily] { inet | inet6 | ipx
                 | dnet | link } | -4 | -6 | -I | -D | -B | -0 | -l[oops] { max-
                 imum-addr-flush-attempts } | -o[neline] | -rc[vbuf] [size] |
                 -t[imestamp] | -ts[hort] | -n[etns] name | -a[ll] }

OPTIONS
       -V, -Version
              Print the version of the ip utility and exit.
 Manual page ip(8) line 1 (press h for help or q to quit)
```

ip vs ifconfig/route/…

net-tools (old)	Iproute (new)
ifconfig	**ip link (show/up/down)**
	ip addr (show/add/delete)
route	**ip route**
	ip rule
arp	**ip neighbour**
	ip ntable
ipmaddr	**ip maddress**
	ip mroute
	ip mrule

```
Useful flags: -4|-6|-D => protocol family: inet, inet6, DECNet!!!
              -stats   => output more statistics information
              -batch   => read command from file
```

network configuration with ip

(non persistent!!)

```
# ip addr show
# ip link show

# ip address add 192.168.100.80/24 dev enp0s3
# ip address del 192.168.100.80/24 dev enp0s3

# ip link set [dev] em1 down
# ip link set [dev] em2 up
# ip link set [dev] em1 mtu 9000

# ip neigh show
# ip neigh add 192.168.100.81 lladdr 0:2:4:1:3:5 dev em3
# ip neigh del 192.168.100.81 dev em3

# ip route add 192.168.100.0/24 dev em1
# ip route add default via 193.206.156.254
```

ip l, ip r, ip n

virtone.mib.infn.it - carbone@virtone:~ VT

File Edit Setup Control Window Help

```
[carbone@virtone ~]$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
3: em2: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN mode DEFAULT qlen 1000
    link/ether 04:7d:7b:68:81:c6 brd ff:ff:ff:ff:ff:ff
4: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DEFAULT qlen 1000
    link/ether 52:54:00:a0:64:0e brd ff:ff:ff:ff:ff:ff
5: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN mode DEFAULT qlen 1000
    link/ether 52:54:00:a0:64:0e brd ff:ff:ff:ff:ff:ff
[carbone@virtone ~]$ ip neigh
fe80::862b:2bff:fe01:652c dev em1 lladdr 84:2b:2b:01:65:2c STALE
2001:760:4211::1 dev em1 lladdr 84:2b:2b:01:65:2c STALE
2001:760:4211::2 dev em1 lladdr 54:52:00:57:37:9b STALE
fe80::5652:ff:fe57:379b dev em1 lladdr 54:52:00:57:37:9b STALE
fe80::226:bff:fe35:ae40 dev em1 lladdr 00:26:0b:35:ae:40 router STALE
193.206.156.4 dev em1 lladdr 00:30:48:8e:67:d5 STALE
212.189.204.247 dev em1  FAILED
212.189.204.254 dev em1 lladdr 00:26:0b:35:ae:40 DELAY
212.189.204.2 dev em1 lladdr 54:52:00:57:37:9b STALE
212.189.204.242 dev em1 lladdr 06:17:65:2e:58:1b STALE
193.206.156.105 dev em1 lladdr 00:1b:63:b9:69:cc STALE
212.189.204.50 dev em1 lladdr 00:30:48:8e:67:d5 STALE
[carbone@virtone ~]$ ip -6 route list type unicast
2001:760:4211::/64 dev em1 proto ra metric 100
fe80::226:bff:fe35:ae40 dev em1 proto static metric 100
fe80::/64 dev em1 proto kernel metric 256
default via fe80::226:bff:fe35:ae40 dev em1 proto static metric 100
[carbone@virtone ~]$ ip -4 route list
default via 212.189.204.254 dev em1 proto static metric 100
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
212.189.204.0/24 dev em1 proto kernel scope link src 212.189.204.210 metric 100
[carbone@virtone ~]$
```

ss: socket statistics

Mate Terminal

File Edit View Search Terminal Help

```
[carbone@seven ~]$ ss -uta
Netid  State      Recv-Q Send-Q        Local Address:Port                       Peer Address:Port
udp    UNCONN     0      0             192.168.122.1:domain                                *:*
udp    UNCONN     0      0                  *%virbr0:bootps                                *:*
udp    UNCONN     0      0                         *:mdns                                  *:*
udp    UNCONN     0      0                         *:48405                                 *:*
udp    UNCONN     0      0                 127.0.0.1:323                                   *:*
udp    UNCONN     0      0                       ::1:323                                  :::*
tcp    LISTEN     0      128                       *:sunrpc                                *:*
tcp    LISTEN     0      5             192.168.122.1:domain                                *:*
tcp    LISTEN     0      128                       *:ssh                                   *:*
tcp    LISTEN     0      128               127.0.0.1:ipp                                   *:*
tcp    LISTEN     0      100               127.0.0.1:smtp                                  *:*
tcp    LISTEN     0      128               127.0.0.1:x11-ssh-offset                        *:*
tcp    LISTEN     0      128               127.0.0.1:41028                                 *:*
tcp    ESTAB      0      0            193.206.157.70:ssh                        193.206.156.4:57533
tcp    ESTAB      0      0                 127.0.0.1:49468                           127.0.0.1:41026
tcp    FIN-WAIT-1 0      85           193.206.157.70:ssh                          42.7.26.91:fg-sysupdate
tcp    ESTAB      0      0                 127.0.0.1:41028                           127.0.0.1:59508
tcp    ESTAB      0      0                 127.0.0.1:41026                           127.0.0.1:49468
tcp    ESTAB      0      0                 127.0.0.1:59508                           127.0.0.1:41028
tcp    TIME-WAIT  0      0            193.206.157.70:ssh                       122.226.181.165:40462
tcp    LISTEN     0      128                      :::sunrpc                                :::*
tcp    LISTEN     0      128                      :::ssh                                   :::*
tcp    LISTEN     0      128                     ::1:ipp                                   :::*
tcp    LISTEN     0      100                     ::1:smtp                                  :::*
tcp    LISTEN     0      128                     ::1:x11-ssh-offset                        :::*
tcp    LISTEN     0      128                     ::1:41028                                 :::*
tcp    ESTAB      0      0        2001:760:4211::112:ssh                   2001:760:4211::101:64812
[carbone@seven ~]$
```

ss usage examples

General syntax:

```
$ ss [options] [FILTER]
FILTER := [ state STATE-FILTER ] [ EXPRESSION ]
```

STATE-FILTER allows to construct arbitrary set of states to match (established, syn-sent, syn-recv, time-wait, listen, …)

Display all TCP IPv4/IPv6 (`-4|-6`) listening sockets:

```
$ ss -tl
$ ss -t state listening
```

Display all established ssh incoming/outgoing connections:

```
$ ss -o state established '(dport = :ssh or sport = :ssh)'
```

List all the TCP sockets in all TCP state for outgoing connections to http/https port on 192.84.138/24 network, and look at their timers:

```
$ ss -o state all '(dport = :http or dport = :https)' \
  dst 192.84.138/24
```

Table of contents

link aggregation

The purpose of the Team project is *to provide a mechanism to team multiple NICs (ports) into one logical one (**teamdev**) at L2 layer*. The process is called "channel bonding", "Ethernet bonding", "channel teaming", "link aggregation", etc., *and aims to provide a logical link with higher throughput, or to provide redundancy.* This is already implemented in the Linux kernel by the bonding driver. The main thing to realize is that the Team project is not trying to replicate or mimic the bonding driver. What it does is solve the same problem using a different approach. Therefore, for example, the way Team is configured differs dramatically from the way bonding is. Team has many advantages over Bonding. These will be described later in this text.

An example setup might look like this. Team softdev Linux driver instance is netdev called *team0*. It has two ports: *eth0* and *eth1*. *team0* has an assigned IP address X. Note that *eth0* and *eth1* do not have an IP assigned. It would not make sense because as a part of *team0*, the Team softdev Linux driver collects all received traffic and "changes" it so that it appears to be coming from *team0*. More info about this later in the text.

From libteam Team project introduction

Team components

- ***Team kernel driver***: very slim, implements all things which should be done fast, mainly transmit and receive packet (skbs – socket buffers) flows.
- ***Team lib (teamlib)***: uses libnl and its primary purpose is to do userspace wrapping of Team Netlink communication (nl: kernel <-> userspace communication).
- ***teamd***: Team daemon. It runs as a daemon and one instance of *teamd* works with one instance of Team softdev Linux driver (one team netdev, for example *team0*). The purpose is to implement various logic of Team's behavior, from the most basic ones such as round-robin, to more complex such as active-backup and load-balancing. The logic is implemented in *teamd* parts called "runners". It also initializes link-watchers and D-Bus interface; takes care of port addition and removal and relative event handling, …
- ***teamdctl***: provides a wrapper for control API using it to monitor and control *teamd* runtime.

teaming *modes*

- *broadcast* - Basic mode in which all packets are sent via all available ports.
- *roundrobin* - Basic mode with very simple transmit port-selecting algorithm based on looping around the port list. This is the only mode able to run on its own without userspace interactions.
- *random* - Basic mode similar to the previous one. Transmit port is selected randomly for each outgoing skb.
- *activebackup* - In this mode, only one port is active at a time and able to perform transmit and receive of skb. The rest of the ports are backup ports. Mode exposes *activeport* option through which userspace application can specify the active port.
- *loadbalance* - A more complex mode used for example for LACP and userspace controlled transmit and receive load balancing. LACP protocol is part of the 802.3ad standard and is very common for smart switches (required for this mode to work correctly).

link-watchers

link-watchers serve for link monitoring purposes. Depending on the particular type they use different methods to find out if a port is capable of data transfers. In other words "if the link is up".

Following types are supported:

- *ethtool* - Uses Libteam lib to get port ethtool state changes.
- *arp_ping* - ARP requests are sent through a port. If an ARP reply is received, the link is considered to be up. Target IP address, interval and other options can be setup in *teamd* config.
- *nsna_ping* - Similar to the previous, only it uses the IPv6 Neighbour Solicitation and Neighbour Advertisement mechanism. This is an alternative to *arp_ping* and becomes handy in pure-IPv6 environments.

Either one link-watch is set for all ports or each port can have its own link-watch. User can also specify multiple link-watchers used at the same time. In that case, link is up if any of the link-watchers reports the link up.

runners

Runners determine the behaviour of the Team device. They operate using the kernel Team mode they want. Runners watch for port link state changes (propagated by the selected link-watch) and react to that. They may implement other functionality as well.

The following runners can be used (Team softdev Linux driver modes are stated in parenthesis):

- *broadcast* (*broadcast*) - Does almost nothing because it only says to put teamdev into *broadcast* mode.
- *roundrobin* (*roundrobin*) - Does almost nothing because it only says to put teamdev into *roundrobin* mode.
- *random* (*random*) - Does almost nothing because it only says to put teamdev into *roundrobin* mode.
- *activebackup* (*broadcast*) - Watches for link changes and selects active port to be used for data transfers. Each port can be configured to have its priority and to be "sticky" or not. Being "sticky" here means to not be de-activated even if a port with a better priority gains its link.
- *loadbalance* (*loadbalance*) - To do passive/active load balancing.
- *lacp* (*loadbalance*) - Implements 802.3ad LACP protocol.

teaming vs bonding

- teaming has a small(er) kernel module which implements fast handling of packets flowing through your teamed interfaces
- support for IPv6 (NS/NA) link monitoring
- capable of working with D-Bus and Unix Domain Sockets (the default)
- it provides an extensible and scale-able solution for your teaming requirements
- load balancing for LACP support
- full user-space runtime control and NM integration
- multiple-link monitoring setup
- port prioritization in activebackup mode

setting up a *team* device

root@virtone:/etc/sysconfig/network-scripts

File Edit View Search Terminal Help

```
[root@virtone network-scripts]# nmcli device
DEVICE      TYPE      STATE         CONNECTION
virbr0      bridge    connected     virbr0
em1         ethernet  disconnected  --
em2         ethernet  disconnected  --
lo          loopback  unmanaged     --
virbr0-nic  tun       unmanaged     --
[root@virtone network-scripts]# nmcli connection
NAME    UUID                                  TYPE    DEVICE
virbr0  5e1ccdcf-5590-437a-ae4d-2fc76320fdbb  bridge  virbr0
[root@virtone network-scripts]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
3: em2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 04:7d:7b:68:81:c6 brd ff:ff:ff:ff:ff:ff
5: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN qlen 1000
    link/ether 52:54:00:a0:64:0e brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
6: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN qlen 1000
    link/ether 52:54:00:a0:64:0e brd ff:ff:ff:ff:ff:ff
[root@virtone network-scripts]#
```

adding master *team0* dev/conn

root@virtone:~

File Edit View Search Terminal Help

```
[root@virtone ~]# nmcli connection add type team con-name team0 ifname team0
Connection 'team0' (800f76cd-277a-465e-a5a4-f9bc23ccfefb) successfully added.
[root@virtone ~]# nmcli device
DEVICE      TYPE      STATE                                  CONNECTION
virbr0      bridge    connected                              virbr0
team0       team      connecting (getting IP configuration)  team0
em1         ethernet  disconnected                           --
em2         ethernet  disconnected                           --
lo          loopback  unmanaged                              --
virbr0-nic  tun       unmanaged                              --
[root@virtone ~]# nmcli connection
NAME    UUID                                  TYPE    DEVICE
team0   800f76cd-277a-465e-a5a4-f9bc23ccfefb  team    team0
virbr0  5e1ccdcf-5590-437a-ae4d-2fc76320fdbb  bridge  virbr0
[root@virtone ~]# ls -lt /etc/sysconfig/network-scripts/ifcfg-*
-rw-r--r--. 1 root root 282 Apr  9 14:27 /etc/sysconfig/network-scripts/ifcfg-team0
-rw-r--r--. 1 root root 254 May  3  2017 /etc/sysconfig/network-scripts/ifcfg-lo
[root@virtone ~]#
```

team0 initial status: IPv4

root@virtone:~

File Edit View Search Terminal Help

```
[root@virtone ~]# nmcli -f ipv4,IP4 connection show team0
ipv4.method:                            auto
ipv4.dns:                               --
ipv4.dns-search:                        --
ipv4.dns-options:                       (default)
ipv4.dns-priority:                      0
ipv4.addresses:                         --
ipv4.gateway:                           --
ipv4.routes:                            --
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-timeout:                      0
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.dhcp-fqdn:                         --
ipv4.never-default:                     no
ipv4.may-fail:                          yes
ipv4.dad-timeout:                       -1 (default)
[root@virtone ~]#
```

team0 initial status: IPv6

root@virtone:~

File Edit View Search Terminal Help

```
[root@virtone ~]# nmcli -f ipv6,IP6 connection show team0
ipv6.method:                            auto
ipv6.dns:                               --
ipv6.dns-search:                        --
ipv6.dns-options:                       (default)
ipv6.dns-priority:                      0
ipv6.addresses:                         --
ipv6.gateway:                           --
ipv6.routes:                            --
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          yes
ipv6.ip6-privacy:                       -1 (unknown)
ipv6.addr-gen-mode:                     stable-privacy
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
ipv6.token:                             --
[root@virtone ~]#
```

configure *team0* IPv4 addresses

root@virtone:~

File Edit View Search Terminal Help

```
[root@virtone ~]# nmcli connection modify team0 \
>                      ipv4.method manual \
>                      ipv4.addresses 212.189.204.210/24 ipv4.gateway 212.189.204.254
[root@virtone ~]# nmcli connection modify team0 \
>                      ipv4.dns 212.189.204.2 ipv4.dns-search mib.infn.it
[root@virtone ~]# nmcli -f ipv4 connection show team0
ipv4.method:                            manual
ipv4.dns:                               212.189.204.2
ipv4.dns-search:                        mib.infn.it
ipv4.dns-options:                       (default)
ipv4.dns-priority:                      0
ipv4.addresses:                         212.189.204.210/24
ipv4.gateway:                           212.189.204.254
ipv4.routes:                            --
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-timeout:                      0
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.dhcp-fqdn:                         --
ipv4.never-default:                     no
ipv4.may-fail:                          yes
ipv4.dad-timeout:                       -1 (default)
[root@virtone ~]#
```

adding *slave* devs/conns

root@virtone:~

File Edit View Search Terminal Help

```
[root@virtone ~]# nmcli connection add type team-slave ifname em1 master team0
Connection 'team-slave-em1' (43efa824-5608-4aeb-afd8-22f1b14ff674) successfully added.
[root@virtone ~]# nmcli device
DEVICE      TYPE      STATE         CONNECTION
virbr0      bridge    connected     virbr0
em1         ethernet  connected     team-slave-em1
team0       team      connected     team0
em2         ethernet  disconnected  --
lo          loopback  unmanaged     --
virbr0-nic  tun       unmanaged     --
[root@virtone ~]# nmcli connection
NAME            UUID                                   TYPE            DEVICE
team-slave-em1  43efa824-5608-4aeb-afd8-22f1b14ff674  802-3-ethernet  em1
team0           800f76cd-277a-465e-a5a4-f9bc23ccfefb  team            team0
virbr0          5e1ccdcf-5590-437a-ae4d-2fc76320fdbb  bridge          virbr0
[root@virtone ~]# nmcli connection add type team-slave ifname em2 master team0
Connection 'team-slave-em2' (9ba745ff-77bd-49cc-8cc6-dcdff81aa24c) successfully added.
[root@virtone ~]# nmcli device
DEVICE      TYPE      STATE      CONNECTION
virbr0      bridge    connected  virbr0
em1         ethernet  connected  team-slave-em1
em2         ethernet  connected  team-slave-em2
team0       team      connected  team0
lo          loopback  unmanaged  --
virbr0-nic  tun       unmanaged  --
[root@virtone ~]#
```

team0 status after init

```
[root@virtone ~]# ip a s dev em1; ip a s dev em2; ip a s dev team0
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master team0 state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
3: em2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master team0 state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
12: team0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
    inet 172.18.12.57/16 brd 172.18.255.255 scope global dynamic team0
       valid_lft 21290sec preferred_lft 21290sec
    inet6 2001:760:4211:0:b716:8c16:f8ea:e0dc/64 scope global noprefixroute dynamic
       valid_lft 2591933sec preferred_lft 604733sec
    inet6 fe80::7127:a4ac:1cd9:97f2/64 scope link
       valid_lft forever preferred_lft forever
[root@virtone ~]# teamdctl team0 state
setup:
  runner: roundrobin
ports:
  em1
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
  em2
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
[root@virtone ~]#
```

reload *team0* connection

root@virtone:~

File Edit View Search Terminal Help

```
[root@virtone ~]# nmcli con up team0
Connection successfully activated (master waiting for slaves) (D-Bus active path: /org/freedesktop/N
etworkManager/ActiveConnection/59)
[root@virtone ~]# ip a s dev em1; ip a s dev em2; ip a s dev team0
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master team0 state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
3: em2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master team0 state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
12: team0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
    inet 212.189.204.210/24 brd 212.189.204.255 scope global team0
       valid_lft forever preferred_lft forever
    inet6 fe80::7bef:431d:56d1:1765/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
    inet6 fe80::5efe:82f2:1776:328b/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
    inet6 fe80::7127:a4ac:1cd9:97f2/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
[root@virtone ~]#
```

DAD failed…?

Why IPv6 link-local address are marked with "tentative dadfailed" over bond devices in latest RHEL versions?

SOLUTION VERIFIED - Updated May 5 2017 at 6:56 AM - English

Issue

- After an upgrade from RHEL 6.6 to 6.7 several bond interfaces shows `tentative dadfailed`
- `ip addr` command show `tentative dadfailed` for link-local address in RHEL7.2 with Active-backup bond modes.

Environment

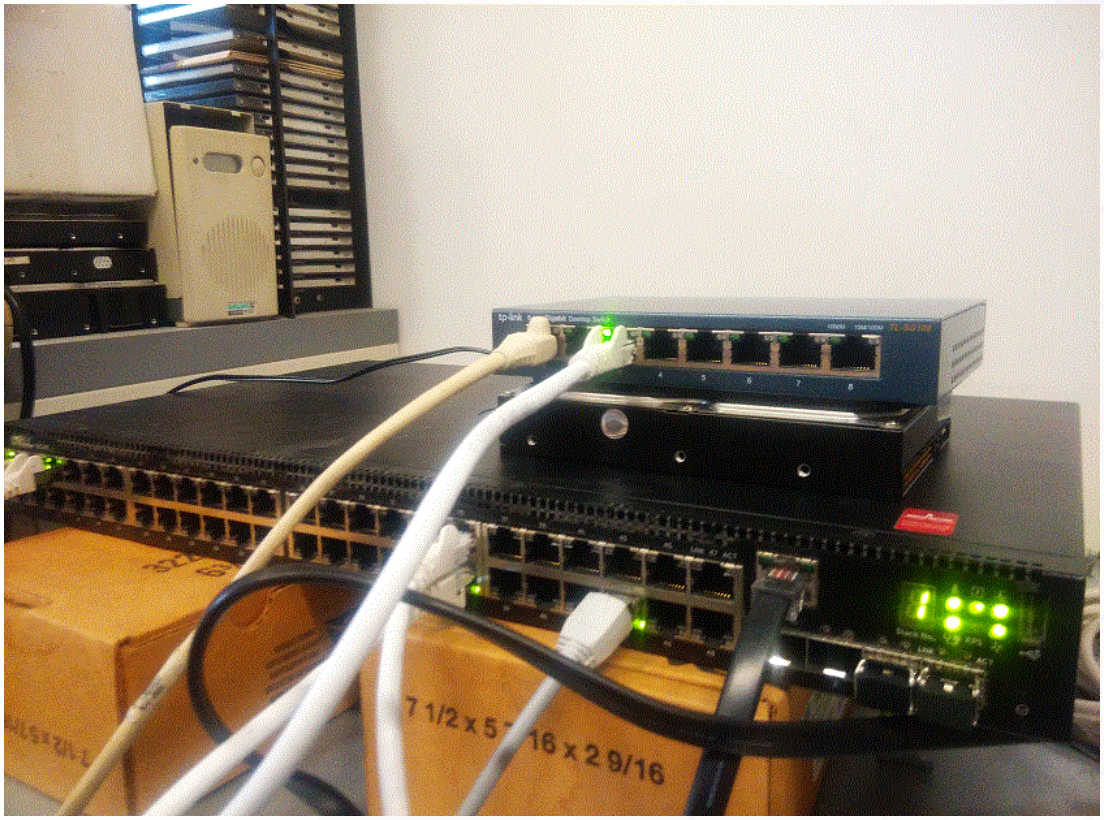
- Red Hat Enterprise Linux 6.7
- Red Hat Enterprise Linux 6.8 Beta 1
- Red Hat Enterprise Linux 7.2

SUBSCRIBER EXCLUSIVE CONTENT

A Red Hat subscription provides unlimited access to our knowledgebase of over 48,000 articles and solutions.

Unless the *exclusive content* describes a magic workaround it seems that a switch supporting static LAG (Link Aggregation) ***at least*** is required in order for DAD to work properly, and for IPv6 to work at all. As far as IPv6 is concerned a dumb switch (TP-Link TL-SG108, 8x10/100/1000) only supports smoothy *activebackup* mode; a DELL N1548 (48x10/100/1000+4) supports every teamd mode (*lacp* too). On the other hand *activebackup* mode isn't supported by LAG ports, but *seems* to work fine even between different switches (dumb, smart, dumb/smart…).

NIC teaming test setup



TP-LINK TL-SG108 (~30 €…)

- 8 RJ45 10/100/1000

DELL N1548

- 48 RJ45 10/100/1000
- 4 10 GbE SFP+ ports
- LAG (static/dynamic) supported

configure another *runner*

root@virtone:~

File Edit View Search Terminal Help

```
[root@virtone ~]# nmcli conn modify team0 team.config '{"runner": {"name":"activebackup"}}'
[root@virtone ~]# nmcli con up team0
Connection successfully activated (master waiting for slaves) (D-Bus active path: /org/freedesktop/N
etworkManager/ActiveConnection/62)
[root@virtone ~]# ip a s dev em1; ip a s dev em2; ip a s dev team0
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master team0 state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
3: em2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master team0 state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
12: team0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP qlen 1000
    link/ether 04:7d:7b:68:81:c5 brd ff:ff:ff:ff:ff:ff
    inet 212.189.204.210/24 brd 212.189.204.255 scope global team0
       valid_lft forever preferred_lft forever
    inet6 2001:760:4211:0:b716:8c16:f8ea:e0dc/64 scope global noprefixroute dynamic
       valid_lft 2591999sec preferred_lft 604799sec
    inet6 fe80::7127:a4ac:1cd9:97f2/64 scope link
       valid_lft forever preferred_lft forever
[root@virtone ~]#
```

team0 state & config dump

```
root@virtone:~
File  Edit  View  Search  Terminal  Help
[root@virtone ~]# teamdctl team0 state
setup:
  runner: activebackup
ports:
  em1
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
  em2
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
runner:
  active port: em2
[root@virtone ~]#
```

```
root@virtone:~
File  Edit  View  Search  Terminal  Help
[root@virtone ~]# teamdctl team0 config dump
{
    "device": "team0",
    "mcast_rejoin": {
        "count": 1
    },
    "notify_peers": {
        "count": 1
    },
    "ports": {
        "em1": {
            "link_watch": {
                "name": "ethtool"
            }
        },
        "em2": {
            "link_watch": {
                "name": "ethtool"
            }
        }
    },
    "runner": {
        "name": "activebackup"
    }
}
[root@virtone ~]#
```

activebackup @ work

carbone@virtone:/home/carbone

File Edit View Search Terminal Help

```
Apr 10 10:39:47 virtone kernel: bnx2 0000:03:00.1 em2: NIC Copper Link is Up, 1000 Mbps full duplex
Apr 10 10:39:47 virtone kernel: , receive & transmit flow control ON
Apr 10 10:39:47 virtone NetworkManager: em2: ethtool-link went up.
Apr 10 10:39:47 virtone NetworkManager[999]: <info>  [1523349587.7485] device (em2): link connected
Apr 10 10:40:01 virtone systemd: Created slice User Slice of root.
Apr 10 10:40:01 virtone systemd: Starting User Slice of root.
Apr 10 10:40:01 virtone systemd: Started Session 19 of user root.
Apr 10 10:40:01 virtone systemd: Starting Session 19 of user root.
Apr 10 10:40:01 virtone systemd: Removed slice User Slice of root.
Apr 10 10:40:01 virtone systemd: Stopping User Slice of root.
Apr 10 10:41:16 virtone kernel: bnx2 0000:03:00.1 em2: NIC Copper Link is Down
Apr 10 10:41:16 virtone NetworkManager: em2: ethtool-link went down.
Apr 10 10:41:18 virtone kernel: bnx2 0000:03:00.1 em2: NIC Copper Link is Up, 1000 Mbps full duplex
Apr 10 10:41:18 virtone kernel: , receive & transmit flow control ON
Apr 10 10:41:18 virtone NetworkManager: em2: ethtool-link went up.
Apr 10 10:41:18 virtone NetworkManager[999]: <info>  [1523349678.3861] device (em2): link connected
Apr 10 10:41:48 virtone kernel: bnx2 0000:03:00.0 em1: NIC Copper Link is Down
Apr 10 10:41:48 virtone NetworkManager: em1: ethtool-link went down.
Apr 10 10:41:48 virtone NetworkManager: Changed active port to "em2".
Apr 10 10:46:02 virtone kernel: bnx2 0000:03:00.0 em1: NIC Copper Link is Up, 1000 Mbps full duplex
Apr 10 10:46:02 virtone kernel: , receive & transmit flow control ON
Apr 10 10:46:02 virtone NetworkManager[999]: <info>  [1523349962.8972] device (em1): link connected
Apr 10 10:46:02 virtone NetworkManager: em1: ethtool-link went up.
```

After the ***em1*** up->down transition ***em2*** becomes *active port* and doesn't change status even when ***em1*** goes up again.

ifcfg files

```
DEVICE=team0
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=none
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=team0
UUID=800f76cd-277a-465e-a5a4-f9bc23ccfefb
ONBOOT=yes
DEVICETYPE=Team
IPADDR=212.189.204.210
PREFIX=24
GATEWAY=212.189.204.254
DNS1=212.189.204.2
DOMAIN=mib.infn.it
TEAM_CONFIG="{\"runner\": {\"name\":\"activebackup\"}}"
```

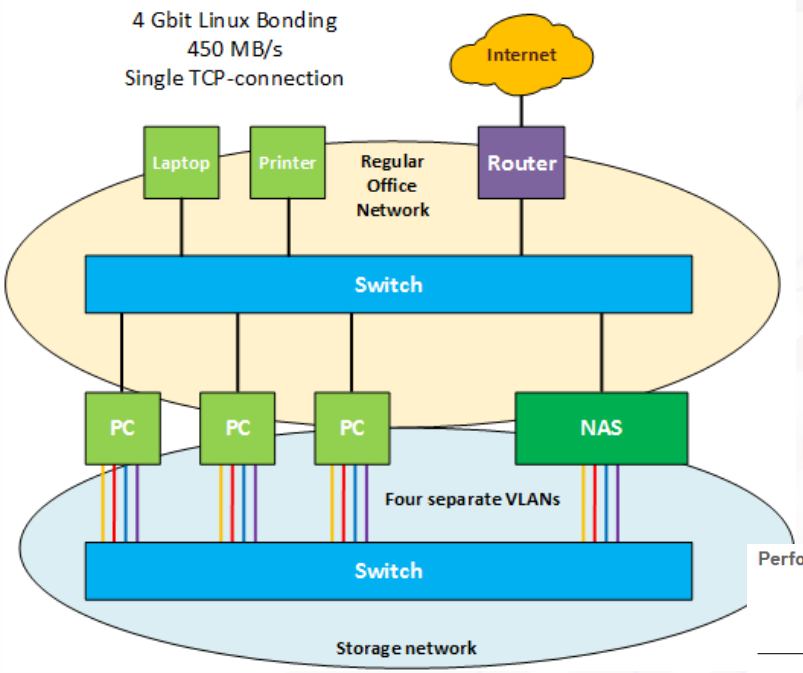
ifcfg-team0

```
NAME=team-slave-em1
UUID=43efa824-5608-4aeb-afd8-
22f1b14ff674
DEVICE=em1
ONBOOT=yes
TEAM_MASTER=team0
DEVICETYPE=TeamPort
```

ifcfg-team-slave-em1

LAG performances?

Achieving 450 MB/s Network File Transfers Using Linux Bonding (2014)



If You Like Bonding, You Will Love Teaming (2014)

Performance

Machine type: 3.3Ghz CPU (Intel), 4GB RAM
Link Type: 10GFO

Interface	Performance with 64byte packets	Performance with 1KB packets	Performance with 64KB packets	Average Latency
eth0	1664.00Mb/s (27.48%CPU)	8053.53Mb/s (30.71%CPU)	9414.99Mb/s (17.08%CPU)	54.7usec
eth1	1577.44Mb/s (26.91%CPU)	7728.04Mb/s (32.23%CPU)	9329.05Mb/s (19.38%CPU)	49.3usec
bonded (eth0+eth1)	1510.13Mb/s (27.65%CPU)	7277.48Mb/s (30.07%CPU)	9414.97Mb/s (15.62%CPU)	55.5usec
teamed (eth0+eth1)	1550.15Mb/s (26.81%CPU)	7435.76Mb/s (29.56%CPU)	9413.8Mb/s (17.63%CPU)	55.5usec

undocumented (new?) feature

```
jtamigi@castore:/home/jtamigi
File  Edit  View  Search  Terminal  Help
[root@castore jtamigi]# nmcli d
DEVICE      TYPE      STATE         CONNECTION
virbr0      bridge    connected     virbr0
enp0s3      ethernet  disconnected  --
enp0s8      ethernet  disconnected  --
lo          loopback  unmanaged     --
virbr0-nic  tun       unmanaged     --
[root@castore jtamigi]# nmcli connection add type team con-name team0 ifname team0
Connection 'team0' (b46ab772-1917-49e9-ad28-e8524f76bde9) successfully added.
[root@castore jtamigi]# nmcli connection add type ethernet ifname enp0s3 con-name team0s0 master team0 conn
ection.autoconnect yes
Connection 'team0s0' (c78fe609-cf29-44ff-954a-d366ff881ae0) successfully added.
[root@castore jtamigi]# nmcli connection add type ethernet ifname enp0s8 con-name team0s1 master team0 conn
ection.autoconnect yes
Connection 'team0s1' (2b6b9caa-1ae1-4d75-a7fd-89dcb1df51f6) successfully added.
[root@castore jtamigi]# nmcli c
NAME     UUID                                  TYPE      DEVICE
team0    b46ab772-1917-49e9-ad28-e8524f76bde9  team      team0
team0s0  c78fe609-cf29-44ff-954a-d366ff881ae0  ethernet  enp0s3
team0s1  2b6b9caa-1ae1-4d75-a7fd-89dcb1df51f6  ethernet  enp0s8
virbr0   cc65965a-4f9d-4e26-8a8d-8de6cce49a10  bridge    virbr0
[root@castore jtamigi]#
```

Note: it seems the same holds for bonds and bridges too (i.e.: no more team-slave or bond-slave or bridge-slave devices, but the old syntax is still working – linux mysteries ☺)

undocumented feature – cont'd

jtamigi@castore:/home/jtamigi

File Edit View Search Terminal Help

```
[root@castore jtamigi]# teamdctl team0 state
setup:
  runner: roundrobin
ports:
  enp0s3
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
  enp0s8
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
[root@castore jtamigi]#
```

jtamigi@castore:/home/jtamigi

File Edit View Search Terminal Help

```
team0: connected to team0
        "team0"
        team, 08:00:27:FB:CF:61, sw, mtu 1500
        inet4 192.168.32.4/24
        route4 192.168.32.0/24
        inet6 fe80::6dca:fd5a:7337:2e4d/64
        route6 ff00::/8
        route6 fe80::/64
        route6 fe80::/64

virbr0: connected to virbr0
        "virbr0"
        bridge, 52:54:00:B1:4C:88, sw, mtu 1500
        inet4 192.168.122.1/24
        route4 192.168.122.0/24

enp0s3: connected to team0s0
        "Intel 82540EM Gigabit Ethernet Controller (PRO/1000 MT Desktop Adapter)"
        ethernet (e1000), 08:00:27:FB:CF:61, hw, mtu 1500
        master team0
        route6 ff00::/8

enp0s8: connected to team0s1
        "Intel 82540EM Gigabit Ethernet Controller (PRO/1000 MT Desktop Adapter)"
        ethernet (e1000), 08:00:27:FB:CF:61, hw, mtu 1500
        master team0
        route6 ff00::/8

lines 1-28
```

Table of contents

- *IPv6 – a fast'n'furious introduction*
- *network device naming*
- *configuring network settings: NetworkManager, nmcli*
- *ip, ss: what about ifconfig, arp, route, netstat?*
- *link aggregation*
- *bridging*

software bridges

- bridging code in Linux has been around for quite a long time, and implements a stable, robust and fully featured level-2 S/W switch (supporting STP, FDB, …) whose performances are quite reasonable (~6 Gbps throughput on 1518 bytes long UDP frames between two dual Xeon E5-2407 – 8 cores total - equipped with 10 GbE Intel 82599 chips and connected via a IBM blade G8124 switch – *almost 3 cores used for IRQ handling, kernel and user space computations*).

bridge manipulation tools

- brctl (the old faithful)
  - `# brctl addbr BR0`
  - `# brctl addif BR0 eth0`
  - `# brctl showmacs BR0`
- iproute2
  - `# ip link add name BR0 type bridge`
  - `# ip link set BR0 up`
  - `# ip link set eth0 up`
  - `# ip link set eth0 master BR0`
  - `# bridge fdb`
- NetworkManager: nmtui, nm-connection-editor, nmcli

creating a bridge with nmcli

```
# nmcli connection add type bridge \
    con-name bridge0 ifname bridge0 \
    ipv4.method manual ipv4.addresses 212.189.204.210/24

# nmcli connection modify ip4.gateway 212.189.204.254 \
    ipv4.dns 212.189.204.2 ipv4.dns-search 'mib.infn.it'

# nmcli con add type bridge-slave con-name bridge0p2 \
    ifname em2 master bridge0

# nmcli con add type bridge-slave con-name bridge0p1 \
    ifname em1 master bridge0

# nmcli connection up bridge0
```

running bridge

carbone@virtone:/home/carbone

File Edit View Search Terminal Help

```
[root@virtone carbone]# nmcli
bridge0: connected to bridge0
        "bridge0"
        bridge, 04:7D:7B:68:81:C5, sw, mtu 1500
        ip4 default, ip6 default
        inet4 212.189.204.210/24
        inet6 2001:760:4211:0:bca8:4949:d9ca:c90d/64
        inet6 2001:760:4211:0:fa7f:3651:abb1:7463/64
        inet6 fe80::d2cc:2968:5b3a:e2be/64
        route6 2001:760:4211::/64

virbr0: connected to virbr0
        "virbr0"
        bridge, 52:54:00:A0:64:0E, sw, mtu 1500
        inet4 192.168.122.1/24

em1: connected to bridge0p1
        "Broadcom Limited NetXtreme II BCM5716 Gigabit Ethernet"
        ethernet (bnx2), 04:7D:7B:68:81:C5, hw, mtu 1500
        master bridge0

em2: connected to bridge0p2
        "Broadcom Limited NetXtreme II BCM5716 Gigabit Ethernet"
        ethernet (bnx2), 04:7D:7B:68:81:C6, hw, mtu 1500
        master bridge0

lo: unmanaged
        "lo"
        loopback (unknown), 00:00:00:00:00:00, sw, mtu 65536

virbr0-nic: unmanaged
        "virbr0-nic"
        tun, 52:54:00:A0:64:0E, sw, mtu 1500

DNS configuration:
```

bridge info

carbone@virtone:/home/carbone

File Edit View Search Terminal Help

```
[root@virtone carbone]# bridge -s -d link
2: em1 state UP : <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 master bridge0 state forwarding priority 32 cost 100
    hairpin off guard off root_block off fastleave off learning on flood on mcast_flood on
3: em2 state UP : <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 master bridge0 state forwarding priority 32 cost 100
    hairpin off guard off root_block off fastleave off learning on flood on mcast_flood on
6: virbr0-nic state DOWN : <BROADCAST,MULTICAST> mtu 1500 master virbr0 state disabled priority 32 cost 100
    hairpin off guard off root_block off fastleave off learning on flood on mcast_flood on
[root@virtone carbone]# bridge -d -s fdb | tail -20
00:21:5a:e7:3d:d7 dev em1 used 9657/3 master bridge0
d8:9e:f3:16:7f:ff dev em1 used 293/45 master bridge0
ec:9a:74:35:01:b1 dev em1 used 119/49 master bridge0
01:00:5e:00:00:01 dev em1 self permanent
33:33:00:00:00:01 dev em1 self permanent
04:7d:7b:68:81:c6 dev em2 vlan 1 used 9675/9675 master bridge0 permanent
04:7d:7b:68:81:c6 dev em2 used 9675/9675 master bridge0 permanent
70:8b:cd:26:51:25 dev em2 used 0/2 master bridge0
01:00:5e:00:00:01 dev em2 self permanent
33:33:00:00:00:01 dev em2 self permanent
01:00:5e:00:00:01 dev virbr0 self permanent
01:00:5e:00:00:fb dev virbr0 self permanent
52:54:00:a0:64:0e dev virbr0-nic used 22516/22516 master virbr0 permanent
52:54:00:a0:64:0e dev virbr0-nic vlan 1 used 22516/22516 master virbr0 permanent
01:00:5e:00:00:01 dev bridge0 self permanent
01:00:5e:00:00:fb dev bridge0 self permanent
33:33:00:00:00:01 dev bridge0 self permanent
33:33:ff:3a:e2:be dev bridge0 self permanent
33:33:ff:b1:74:63 dev bridge0 self permanent
33:33:ff:ca:c9:0d dev bridge0 self permanent
[root@virtone carbone]#
```

ifcfg files

```
DEVICE=bridge0
STP=yes
BRIDGING_OPTS=priority=32768
TYPE=Bridge
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=none
IPADDR=212.189.204.210
PREFIX=24
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=bridge0
UUID=175c951d-50d1-42f5-aced-02edb6cf42e8
ONBOOT=yes
GATEWAY=212.189.204.254
DNS1=212.189.204.2
DOMAIN=mib.infn.it
IPV6_PRIVACY=rfc3041
```

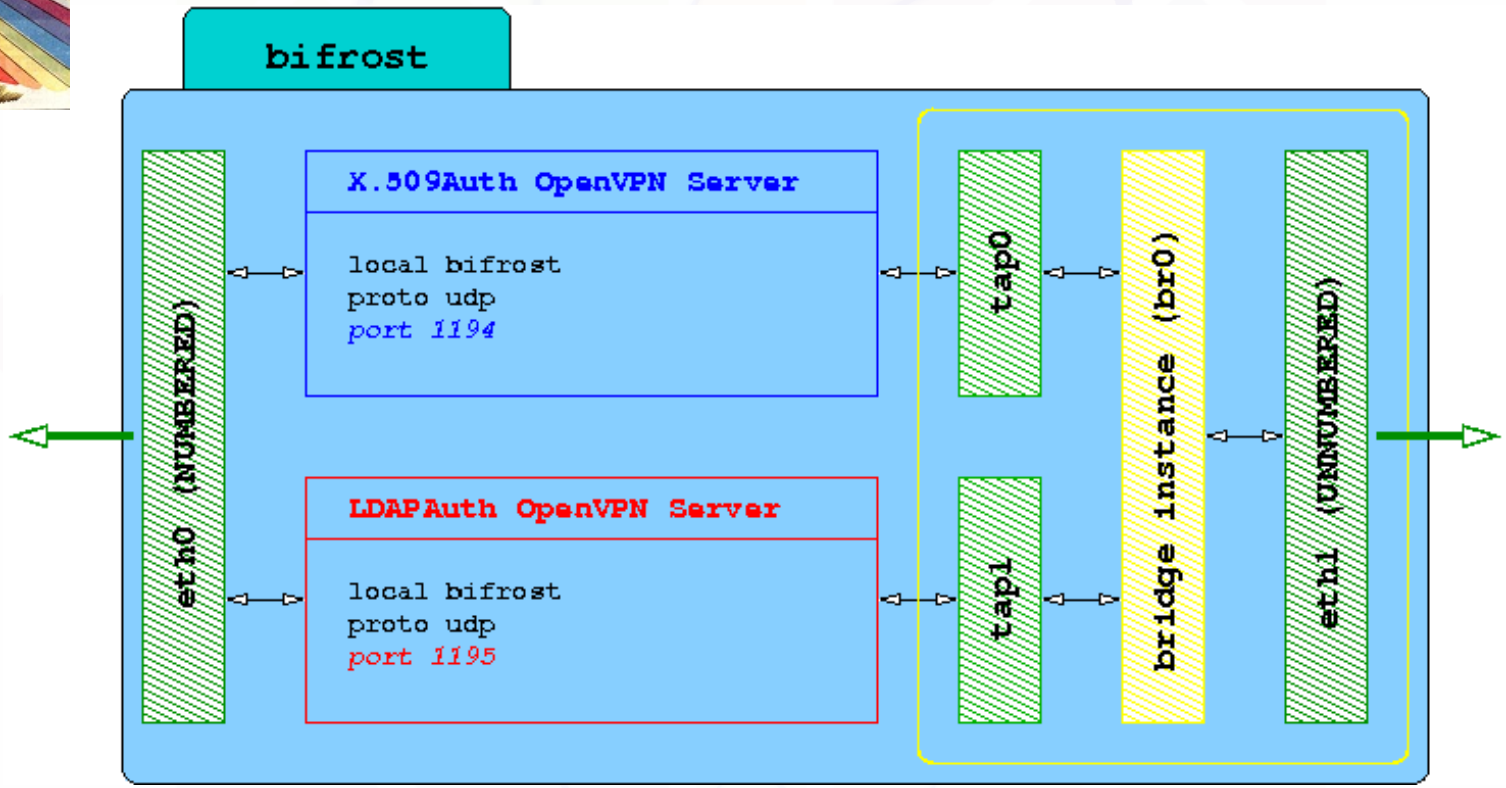
ifcfg-bridge0

```
TYPE=Ethernet
NAME=bridge0p1
UUID=d72bebf7-5c8f-48fd-8feb-
dce9853aa24b
DEVICE=em1
ONBOOT=yes
BRIDGE=bridge0
```

ifcfg-bridge0p1

a real world example

bifrost.mib.infn.it: *dual auth (X.509/LDAP) VPN server with direct access to remote LAN level-2*



old-style config file

```
start_br()
{
    # create TAP devices

    for t in $tap
    do
        $OPENVPN --mktun --dev $t >/dev/null 2>&1
        [ $? -ne 0 ] && ERR=1
    done

    # create bridge instance & add interfaces

    $BRCTL addbr $br >/dev/null 2>&1
    [ $? -ne 0 ] && ERR=1

    $BRCTL addif $br $eth >/dev/null 2>&1
    [ $? -ne 0 ] && ERR=1

    for t in $tap
    do
        $BRCTL addif $br $t >/dev/null 2>&1
        [ $? -ne 0 ] && ERR=1
    done

    # configure bridge interfaces
    for t in $tap
    do
        $IFCONFIG $t $IP_ZERO promisc up
        [ $? -ne 0 ] && ERR=1
    done

    $IFCONFIG $eth $IP_ZERO promisc up
    [ $? -ne 0 ] && ERR=1

    if [ -n "$br_ip" ]
    then
        $IFCONFIG $br $br_ip netmask $br_nm
broadcast $br_bc up
    else
        $IFCONFIG $br up
    fi
    [ $? -ne 0 ] && ERR=1

    # do * not * make iptables see bridged traffic
    echo 0 >
      /proc/sys/net/bridge/bridge-nf-call-iptables
}
```

it works…

ssire.mib.infn.it - root@bifrost:/etc/rc.d/init.d VT

File Edit Setup Control Window Help

```
tunnel status ----------------------------------------------------
openvpn (pid 17956 17954) is running...
-----------------------------------------------------------------
x509 -------------------------------------------------------------
OpenVPN CLIENT LIST
Updated,Wed Apr 11 12:00:22 2018
Common Name,Real Address,Bytes Received,Bytes Sent,Connected Since
Luca Giovanni Carbone,2.34.94.138:56013,14862602,315403761,Wed Apr 11 09:08:06 2018
ROUTING TABLE
Virtual Address,Common Name,Real Address,Last Ref
00:ff:f4:64:b4:b8,Luca Giovanni Carbone,2.34.94.138:56013,Wed Apr 11 12:00:21 2018
GLOBAL STATS
Max bcast/mcast queue length,3
END
LDAP -------------------------------------------------------------
OpenVPN CLIENT LIST
Updated,Wed Apr 11 12:00:22 2018
Common Name,Real Address,Bytes Received,Bytes Sent,Connected Since
ROUTING TABLE
Virtual Address,Common Name,Real Address,Last Ref
GLOBAL STATS
Max bcast/mcast queue length,2
END
-----------------------------------------------------------------
[root@bifrost init.d]# /etc/openvpn/brfwt| grep -v em2
mac address        port   local
00:ff:f4:64:b4:b8  tap0   no
7e:13:b0:b1:91:8c  tap0   yes
e2:c9:93:46:fc:4b  tap1   yes
[root@bifrost init.d]#
```