**CONFIGURING NETWORK TEAMING**

This section describes the basics of network teaming, the differences between bonding and teaming, and how to configure a network team on Red Hat Enterprise Linux 8.

You can create network teams on different types of devices, such as:

* Physical and virtual Ethernet devices
* Network bonds
* Network bridges
* VLAN devices
* **MPORTANT**
  + Certain network teaming features, such as the fail-over mechanism, do not support direct cable connections without a network switch.

**Understanding the teamd service, runners, and link-watchers**

The team service, teamd, controls one instance of the team driver. This instance of the driver adds instances of a hardware device driver to form a team of network interfaces. The team driver presents a network interface, for example team0, to the kernel.

The teamd service implements the common logic to all methods of teaming. Those functions are unique to the different load sharing and backup methods, such as round-robin, and implemented by separate units of code referred to as runners. Administrators specify runners in JavaScript Object Notation (JSON) format, and the JSON code is compiled into an instance of teamd when the instance is created. Alternatively, when using NetworkManager, you can set the runner in the team.runner parameter, and NetworkManager auto-creates the corresponding JSON code.

The following runners are available:

* broadcast: Transmits data over all ports.
* roundrobin: Transmits data over all ports in turn.
* activebackup: Transmits data over one port while the others are kept as a backup.
* loadbalance: Transmits data over all ports with active Tx load balancing and Berkeley Packet Filter (BPF)-based Tx port selectors.
* random: Transmits data on a randomly selected port.
* lacp: Implements the 802.3ad Link Aggregation Control Protocol (LACP).

The teamd services uses a link watcher to monitor the state of subordinate devices. The following link-watchers are available:

* ethtool: The libteam library uses the ethtool utility to watch for link state changes. This is the default link-watcher.
* arp\_ping: The libteam library uses the arp\_ping utility to monitor the presence of a far-end hardware address using Address Resolution Protocol (ARP).
* nsna\_ping: On IPv6 connections, the libteam library uses the Neighbor Advertisement and Neighbor Solicitation features from the IPv6 Neighbor Discovery protocol to monitor the presence of a neighbor’s interface.

**Each runner can use any link watcher, with the exception of lacp. This runner can only use the ethtool link watcher.**

**Installing the teamd service**

To configure a network team in NetworkManager, you require the teamd service and the team plug-in for NetworkManager. Both are installed on Red Hat Enterprise Linux 8 by default. This section describes how you install the required packages in case that you remove them.

**Prerequisites**

* An active Red Hat subscription is assigned to the host.

**Procedure**

Install the teamd and NetworkManager-team packages:

# yum install teamd NetworkManager-team

**Configuring a network team using nmcli commands**

This section describes how to configure a network team using nmcli utility.

**Prerequisites**

* Two or more physical or virtual network devices are installed on the server.
* To use Ethernet devices as ports of the team, the physical or virtual Ethernet devices must be installed on the server and connected to a switch.

**Procedure**

1. Create a team interface:

# nmcli connection add type team con-name *team0* ifname *team0* team.runner *lacp*

This command creates a network team named team0 that uses the lacp runner.

1. Optionally, set a link watcher. For example, to set the ethtool link watcher in the team0 connection profile:

# nmcli connection modify team0 team.link-watchers "name=*ethtool*"

Link watchers support different parameters. To set parameters for a link watcher, specify them space-separated in the name property. Note that the name property must be surrounded by quotes. For example, to use the ethtool link watcher and set its delay-up parameter to 2500 milliseconds (2.5 seconds):

# nmcli connection modify team0 team.link-watchers "name=*ethtool* delay-up=*2500*"

To set multiple link watchers and each of them with specific parameters, the link watchers must be separated by a comma. The following example sets the ethtool link watcher with the delay-up parameter and the arp\_ping link watcher with the source-host and target-host parameter:

# nmcli connection modify team0 team.link-watchers "name=*ethtool* delay-up=*2*, name=*arp\_ping* source-host=*192.0.2.1* target-host=*192.0.2.2*"

1. Display the network interfaces, and note the names of the interfaces you want to add to the team:
2. # nmcli device status
3. DEVICE TYPE STATE CONNECTION
4. enp7s0 ethernet disconnected --
5. enp8s0 ethernet disconnected --
6. bond0 bond connected bond0
7. bond1 bond connected bond1

...

In this example:

* + enp7s0 and enp8s0 are not configured. To use these devices as ports, add connection profiles in the next step. Note that you can only use Ethernet interfaces in a team that are not assigned to any connection.
  + bond0 and bond1 have existing connection profiles. To use these devices as ports, modify their profiles in the next step.

1. Assign the port interfaces to the team:
   * If the interfaces you want to assign to the team are not configured, create new connection profiles for them:
   * # nmcli connection add type ethernet slave-type team con-name team0-port1 ifname enp7s0 master team0

# nmcli connection add type ethernet slave-type team con-name team0-port2 ifname enp8s0 master team0

. These commands create profiles for enp7s0 and enp8s0, and add them to the team0 connection.

* + To assign an existing connection profile to the team, set the master parameter of these connections to team0:
  + # nmcli connection modify enp7s0  master team0

# nmcli connection modify enp8s0 master team0

These commands assign the existing connection profiles named enp7s0  and enp8s0 to the team0 connection.

1. Configure the IP settings of the team. Skip this step if you want to use this team as a ports of other devices.
   * Configure the IPv4 settings. For example, to set a static IPv4 address, network mask, default gateway, DNS server, and DNS search domain the team0 connection, enter:
   * # nmcli connection modify team0 ipv4.addresses '*192.0.2.1/24*'
   * # nmcli connection modify team0 ipv4.gateway '*192.0.2.254*'
   * # nmcli connection modify team0 ipv4.dns '*192.0.2.253*'
   * # nmcli connection modify team0 ipv4.dns-search '*example.com*'

# nmcli connection modify team0 ipv4.method manual

* + Configure the IPv6 settings. For example, to set a static IPv6 address, network mask, default gateway, DNS server, and DNS search domain of the team0 connection, enter:
  + # nmcli connection modify team0 ipv6.addresses '*2001:db8:1::1/64*'
  + # nmcli connection modify team0 ipv6.gateway '*2001:db8:1::fffe*'
  + # nmcli connection modify team0 ipv6.dns '*2001:db8:1::fffd*'
  + # nmcli connection modify team0 ipv6.dns-search '*example.com*'

# nmcli connection modify team0 ipv6.method manual

1. Activate the connection:

# nmcli connection up team0

**Verification steps**

* Display the status of the team:
* # teamdctl team0 state
* setup:
* runner: activebackup
* ports:
* enp7s0
* link watches:
* link summary: up
* instance[link\_watch\_0]:
* name: ethtool
* link: up
* down count: 0
* enp8s0
* link watches:
* link summary: up
* instance[link\_watch\_0]:
* name: ethtool
* link: up
* down count: 0
* runner:

active port: enp7s0

In this example, both ports are up.