Declare Your Linux Network State!

with nmstate

Edward Haas, Red Hat <edwardh@redhat.com>
Till Maas, Red Hat <till@redhat.com>
Configure a Bond

- Bond (mybond0)
  - NIC (eth1)
  - NIC (eth2)
Configure a Bond (nmcli)

$ nmcli con add type bond ifname mybond0 mode active-backup
$ nmcli con add type ethernet ifname eth1 master mybond0
$ nmcli con add type ethernet ifname eth2 master mybond0
$ nmcli con mod mybond0 ipv4.method manual ipv4.address "1.2.3.4/24"
Configure a Bond (iproute2)

# ip link add mybond0 type bond
# ip link set eth1 master mybond0
# ip link set eth2 master mybond0
# ip addr add 1.2.3.4/24 dev mybond0
Configure a Bond (ifcfg)

```
DEVICE=mybond0
mode=active-backup
TYPE=Bond
BONDING_MASTER=yes
IPV6INIT=no
NAME=bond00
ONBOOT=yes
BOOTPROTO=None
IPADDR=1.2.3.4
PREFIX=24
DEFROUTE=yes
```

```
TYPE=Ethernet
NAME=eth1
DEVICE=eth1
ONBOOT=yes
MASTER=mybond0
SLAVE=yes
```

```
TYPE=Ethernet
NAME=eth2
DEVICE=eth2
ONBOOT=yes
MASTER=mybond0
SLAVE=yes
```
Configure a Bond (nmstate)

- interfaces:
  - name: mybond0
    type: bond
    state: up
  link-aggregation:
    mode: active-backup
  slaves:
    - eth1
    - eth2
- ipv4:
  enabled: true
  address:
    - ip: 1.2.3.4
      prefix-length: 24
Design

Complete Linux host network state
Design

Configuration and reporting
Design

Declarative

Describe WHAT you want and don’t bother with the HOW.
Design

Inspired by IETF Network Modeling Working Group (NETCONF/YANG)
Design

- Based on NetworkManager
- Open for provider extensions
Design

Atomic changes

ALL \{ VOID \} NOTHING
Design

Allow partial states

Nmstate Design

- simple API:
  - state = show()
  - apply(state)
- allow partial states to configure only subset of all settings
- verification of the configuration by comparing the runtime state
- atomic configuration changes: Rollback to previous state on failure by default
- Use power of NetworkManager but allow enhancements missing in NetworkManager
Currently support devices

- Ethernet
- IPv4 & IPv6, static & dynamic
- Bonding
- Linux bridges
- OVS bridges (basic)
Command-line interface

# nmstatectl show eth0
---

interfaces:
- name: eth0
type: ethernet
state: up
mtu: 1500
ipv4:
  enabled: true
dhcp: true
  address:
  - ip: 192.168.122.197
    prefix-length: 24
Simple (Python) API

```python
state = netinfo.show()
state['interfaces'][0]['mtu'] = 9000
netapplier.apply(state)
```
Verification with rollback
Ansible network modules

- net_interface
- net_linkagg
- net_vlan
- net_l3_interface
Ansible network modules for Linux

# Ansible

tasks:
- net_linkagg:
  name: web-bond
  state: up
  members:
    - eth1
    - eth2

# nmstate

---

interfaces:
- name: web-bond
  type: bond
  state: up
  link-aggregation:
    mode: 802.3ad
    options: {}
  slaves:
    - eth1
    - eth2
**kubernetes-nmstate (PoC)**

- Manage host/node network through Kubernetes.

- Implements the suggested [Kubernetes Node Network Configuration CRD](https://github.com/nmstate/kubernetes-nmstate)

PoC: [https://github.com/nmstate/kubernetes-nmstate](https://github.com/nmstate/kubernetes-nmstate)
**kubernetes-nmstate (PoC)**

On every Node,
- On every SRIO interface,
- Define 8 VF.

NodeNetConfPolicy

Match on Nodes currentState
& Apply desireState snippet

NodeNetworkState

Define 8 VF on eth0 & eth1.

NodeNetworkState

Define 8 VF on eth0.

NodeNetworkState

Define 8 VF on eth2 & eth4.
apiVersion: "k8s.cni.cncf.io/v1"
kind: NodeNetworkState
metadata:
  name: my-node-netstate
spec:
  managed: true
  nodeName: my-node
  desiredState:
    interfaces:
    - name: bond0
      type: bond
      state: up
      link-aggregation:
        mode: balance-rr
        slaves:
        - eth0
        - eth1
    ipv4:
      enabled: true
      address:
      - ip: 10.10.10.2
        prefix-length: 24
    ipv6:
      enabled: false

status:
currentState:
  capabilities: []
  interfaces:
  - if-index: 10
    name: bond0
    type: bond
    state: up
    phys-address: aa:bb:cc:dd:ee:ff
    link-aggregation:
      mode: balance-rr
      slaves:
      - eth0
      - eth1
    ipv4:
      enabled: true
      address:
      - ip: 10.10.10.2
        prefix-length: 24
    ipv6:
      enabled: false
Challenges

# desired state

---

interfaces:
- name: eth0
type: ethernet
state: up
ipv4:
  enabled: true
dhcp: true

# actual state

---

interfaces:
- name: eth0
type: ethernet
state: up
ipv4:
  enabled: true
dhcp: true
  address:
    - ip: 192.168.122.197
      prefix-length: 24
How to participate

Development: https://github.com/nmstate/nmstate

Planning: https://nmstate.atlassian.net

Discussions:
- NetworkManager mailing list
- #nmstate on Freenode IRC
Outlook

- Different state for configuration/persistence and runtime
- Support commit and confirm commands
- More interface types
- Add read-only report values
- Proprietary vendor interfaces
- NETCONF/YANG
- Routing (under review)
- Firewall
- Integration with oVirt, OpenStack, KubeVirt
Thank you!