ENG. WORKSHOP: PRES.
Linux Networking Greatness (part II).
Roopa Prabhu/Director Engineering Linux Software/Cumulus Networks.
Goals

- Summarize Latest in Linux Network Operating systems
- Latest updates from Linux Networking Communities
  - For better collaboration
- Linux networking developments with potential to help:
  - NOS hardware offload
  - and/or NOS applications
Agenda

- Brief recap part-I
- NOS architectures overview
- Linux networking communities
- Linux networking updates and collaboration examples
- Resources
Disaggregation

Native Linux networking (server/host networking)

- Native Linux hardware acceleration model
- Linux network operating system stacks in userspace
- Hardware acceleration from userspace

Closed Networking

Network OS

Open Hardware

App

App

App
NOS architectures

Native Linux kernel HW acceleration model

- Driver
- App
- Kernel forwarding and routing database
- HW

Linux userspace NOS software stack

- Driver
- App
- App
- HW
**Disaggregated software stack**

- Using Linux natively enables you to run your networking apps everywhere
- Fosters collaboration
- Uniform networking models across Linux deployments
- Minimize operational problems
- Uniform software package management
Linux Networking Communities ...

Existing Linux Networking Communities (netdev)

+ New NOS Communities

= Larger Linux Networking Community
  Uniform Linux Networking Models
  Disaggregated Software Stack
Notable Linux networking updates...

Next few slides cover recent updates from Linux networking communities.
Extended BPF (EBPF)

- Efficient, generic in-kernel bytecode engine
- Gives Linux kernel and application superpowers
- Allows userspace to attach dynamic programs at various points in the kernel
- Users:
  - Socket filters
  - Linux traffic classifier
  - XDP [3]
  - Offload to programmable ASICS, and switch ASICS, NPUS [1]
  - eBPF hooks for cgroups [2]
XDP - eXpress Data Path

- Programmable, High performance, Packet processor in Linux networking datapath
- XDP hooks with BPF programs for packet processing
- Target use cases:
  - Pre-stack processing like filtering to do DOS mitigation
  - Forwarding and Load-balancing
  - Flow sampling, monitoring
tc (Linux traffic classifier) updates

- tc flower: flow based classifier [10]
- tc cls_bpf for a programmable classifier [11]
- tc sample for sampling packets
- tc hardware offload API for:
  - switch ASICs,
  - NPU [1]
  - NICs
eBPF hooks for control groups (cgroups)

- cgroups: mainly used for
  - resource limiting, prioritization, accounting, control
- cgroups networking subsystems: net_cls, net_prio, namespaces
- eBPF hooks for cgroups:
  - Allows for attaching eBPF programs to cgroups for
    - network socket filtering and accounting
  - Users:
    - Containers
    - NOS applications
    - VRF
Virtual Routing and Forwarding updates ..

- Linux kernel is vrf ready [4]
- Systemd and vrfs
  - Starting network services in specific vrfs
- ip vrf exec
  - Start network program in a specific vrf [6]
  - Uses cgroup eBPF hook [5]
- Deploying vrf with Linux made easier
  - iproute2 updates
  - ifupdown2 support
- Linux VRF on Hosts/Servers:
  - Micro-service networking can leverage Linux vrf implementation for traffic segmentation [7]
Light Weight Tunnels

- Replace per tunnel netdevice with attaching tunnel attributes to routes
- Helps with scaling tunnel endpoints
- More users:
  - VxLAN
  - ILA (identifier locator addressing)
  - MPLS
  - Segment routing with IPv6
Segment Routing

- **IPV6 segment routing support** [9]
  - New kernel API, data-path
  - Userspace tools to configure SR
  - Uses Light Weight tunnels to encapsulate SR header

- **MPLS segment routing** [8]
  - MPLS kernel data-path is SR ready
  - Uses Light Weight tunnels to encapsulate MPLS header
  - SR control in user space in the works
Scaling VxLAN

- Attach tunnel parameters (tunnel-id, tunnel-src, tunnel-dst) to routes using LWT
- On receive, extract tunnel parameters and attach to packet
- Per -vlan tunnel parameters for Vxlan bridging gateway

Vxlan gateway

```
ip route add <prefix> encap vxlan vni 1000 src <s> dst <d>
```

Vxlan bridging gateway

- Bridge
- Swp1: vlan: 100, 200
- Vxlan0: vni: 1000, 2000
Systemd the new Linux init system

- A modern Linux init system for your applications
- A single place to manage and monitor your services
- Easily writable, extensible, parseable service files
  - suitable for manipulation with enterprise management tools
- Service files are compatible between OS/NOS distributions
- Make your app systemd aware soon!
Quagga updates

- Un-numbered BGP and OSPF
- VRF support
- Multicast Routing
- Static MPLS/LDP support
- EVPN (In progress)
- Segment Routing (In progress)
- Routing on the host with Quagga
  - validates network software stack disaggregation model
sFLOW

- Linux API for sFLOW monitoring:
  - In pure software
  - Hardware offload to switch ASICs and NICs
- tc sampling API for sflow
  - works across NOS’s and servers
Resources

1. eBPF HW offload: https://netdevconf.org/1.2/papers/eBPF_HW_OFFLOAD.pdf
2. eBPF for cgroups: https://lwn.net/Articles/697462/
5. VRF cgroup integration: https://lwn.net/Articles/708019/
6. iproute2 vrf enhancements: https://www.spinics.net/lists/netdev/msg409852.html
7. vrf on the host: http://netdevconf.org/1.2/slides/oct7/01_ahern_microservice_net_vrf_on_host.pdf
11. tc bpf