Linux networking

For Ganeti clusters
explained

Alexandros Afentoulis
alexaf@noc.grnet.gr

grnet
Networking Research and Education

November 14th 2017

Why?

- Networking on Linux hosts is the bridge* between netops and sysops
- We often need to track down packets fleeing from our ~~hands~~ hosts
- I wish I had someone to briefly explain this to me when I got here.
- Linux networking is awesome ❤

* see what I did there?

Contents and goal

This session's contents:

- Bonding physical interfaces
- vlans
- Bridged networks
- Routed networks

Focus on principles rather than specific to ganeti implementation or GRNET automation.

Live demo's goal:

- connect kvm virtual machines to the internet

Interface bonding

- Aggregate two or more physical interfaces into a logical interface.
- Why? Maximize available bandwidth, availability, load balance traffic.
- Needs 'bonding' kernel module
- ‘ifenslave’ package loads the module and brings helpful management scripts, ip-link may be used too
- Various bonging modes, mostly interested in: active-backup and active-active
- Bond interface inherits a member’s mac address (the “lowest)
- Bonding is not the only way to aggregate interfaces. There is “team” too, with userspace controller

Active-Backup Bonding

- Simple
- Offers fault tolerance, not maximum bandwidth usage
- No configuration needed on the switch side
- Host’s responsibility to pick the outgoing interface
- Only a single interface active at any moment
- Beware, do not bridge the physical ifaces
- Problems? Can’t recall really.

Active-Active Bonding

- 802.3ad or LACP.
- Link aggregation offers maximum bandwidth capacity and load balancing.
- Needs configuration from the switch side.
- Various hashing policies, example layer3+4: /* Input: src_IP, dst_IP, src_port, dst_port */ (outgoin traffic)
- Network devices may use different hashing policy for incoming traffic
- Problems?
  - LACP failing to negotiate aggregator ID with Cisco Nexus
  - Intel X540 NIC + Linux 3.16 + “Speed Unknown” for member, opened Debian bug #851952
  - QFX5100 was flooding packets to LACP hosts, leading to mac learning mayhem on Linux bridge

Bonding configuration examples

Active backup:

```
iface bond0 inet static
    address   83.212.4.210
    netmask   255.255.255.224
    broadcast 83.212.4.223
    gateway   83.212.4.193
    mtu       9000
    bond-mode active-backup
    primary   eno1
    slaves    eno1 eno2
```

LACP:

```
iface bond0 inet static
    address   83.212.4.210
    netmask   255.255.255.224
    broadcast 83.212.4.223
    gateway   83.212.4.193
    mtu       9000
    bond-mode 802.3ad
    slaves    eno1 eno2
    bond_xmit_hash_policy layer3+4
```

Live Demo

Setup bond interface on hardware node

```
ip link add bond0 type bond
echo 1 > /sys/class/net/bond0/bonding/mode
ip link set eth0 master bond0
ip link set eth1 master bond0
ip link set bond0 up
```

Vlans

- Virtual lans
- Give the ability to create multiple separate layer 2 broadcast domains over the same physical link

Why use vlans?

- network segmentation and management
- security (broadcast domain, ARP poisoning, mac address spoofing)
- QoS, traffic manipulation

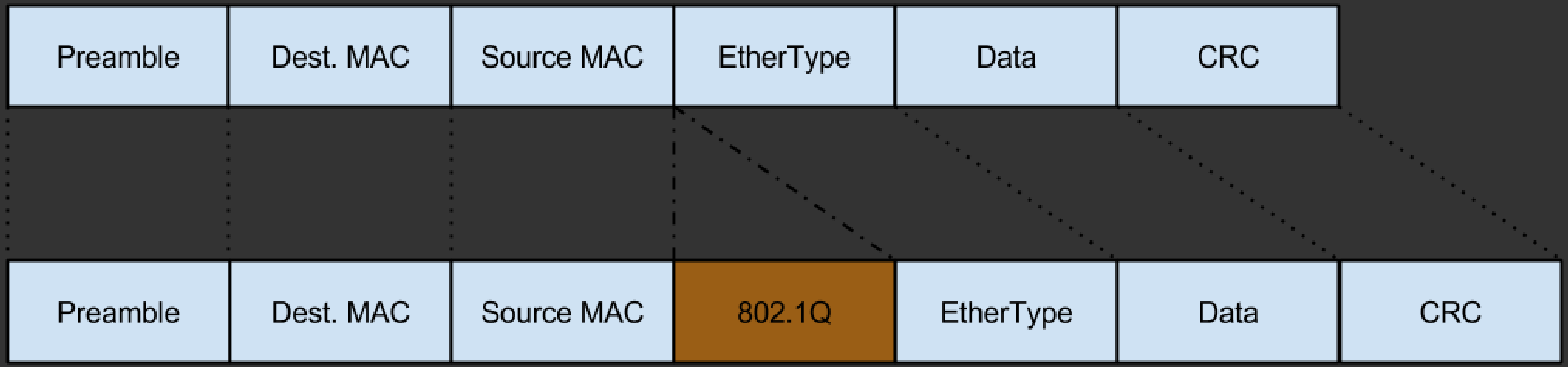
Vlans, how?

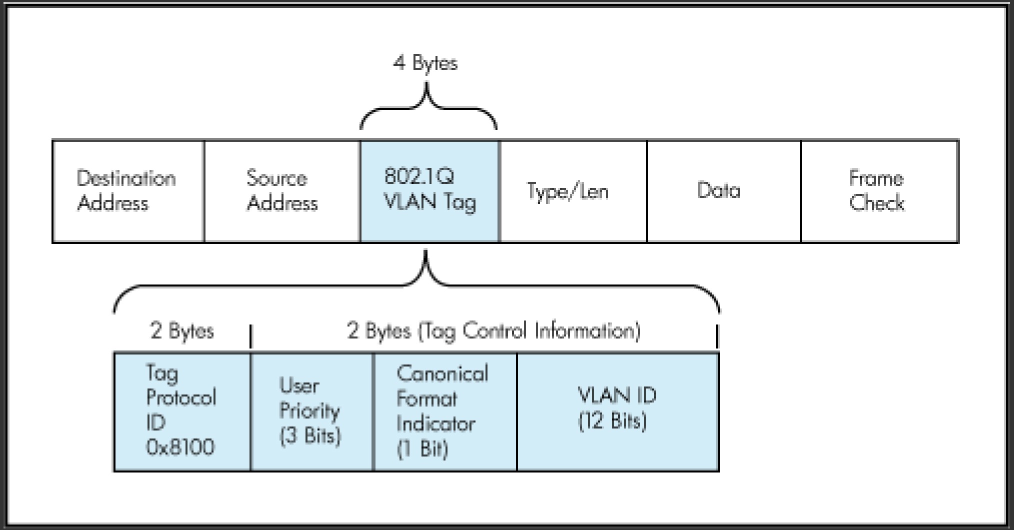
- IEEE 802.1Q or Dot1q standard.
- Use tags/ids in the ethernet frame header.
- Network devices as well as physical hosts need to be aware/configured to handle vlan tagged frames.
- Vlans are implemented in switches, but are mostly terminated in routers (vlan network gateway).
- tcpdump’s ‘-e’ will reveal the packets’ vlan id

Vlans, how?

- A vlan aware interface may also carry untagged frames, these belong to the "native" vlan.
- Forget 'vconfig' (and 'ifconfig') use 'ip' of iproute2 to create vlan interfaces.
- The convention is that bond0.XXX interfaces in Linux correspond to vlan id XXX.
- Packets arriving to bond0 will be "untagged" and be "available" in bond0.XXX interface.
- Inversely packets sent out the bond0.XXX interface will be tagged before getting out through bond0.
- Get only a specific vlan traffic with tcpdump?
  - `tcpdump -ni bond0 -Uw - | tcpdump -en -r - vlan 124`

Vlans in Ethernet Headers





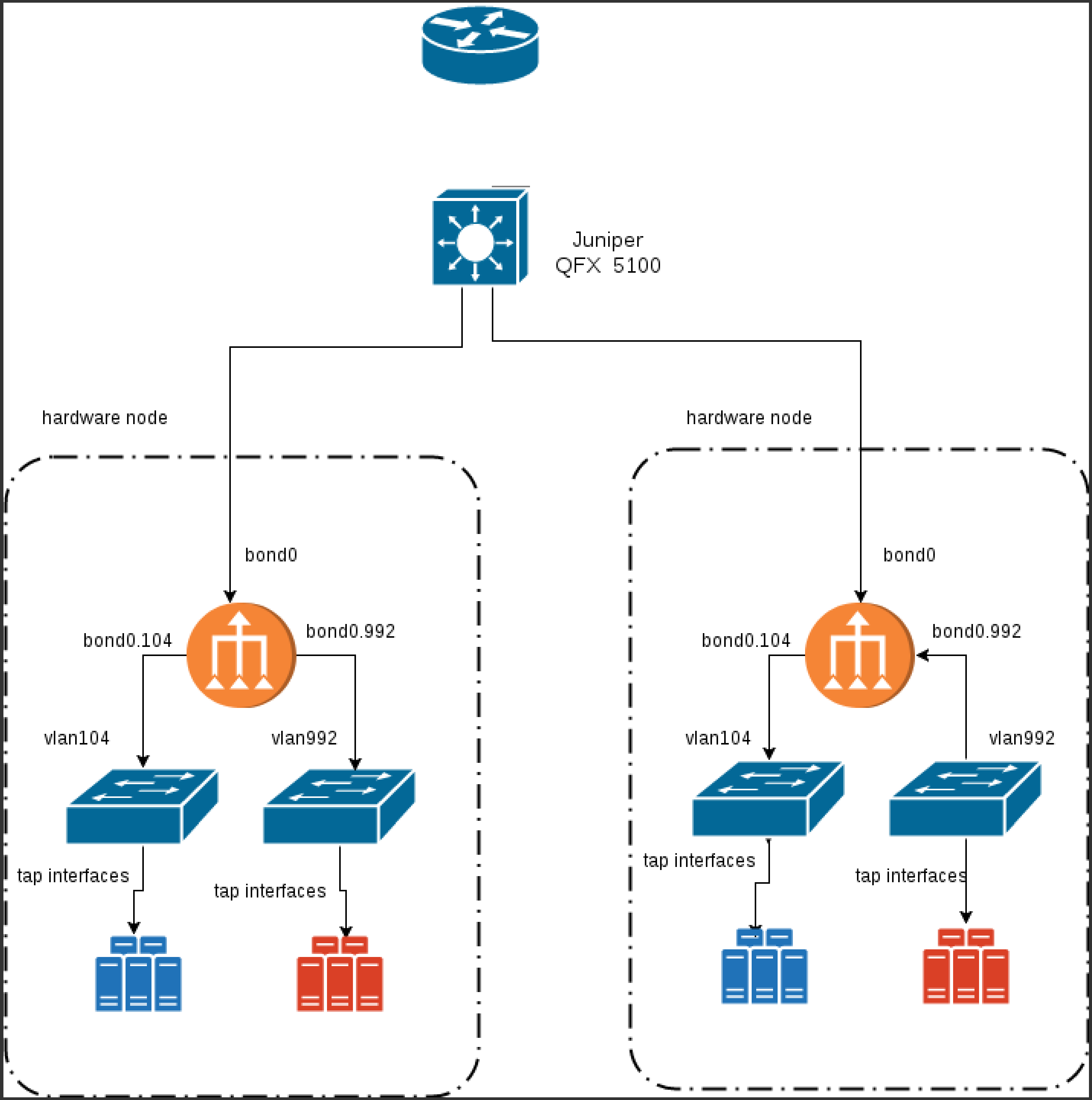
Live Demo

Create vlan interfaces on bond0

```
ip link add link bond0 name bond0.992 type vlan id 992
ip link set dev bond0.992 up
```

Bridged networks

- Bridged networks is a way to interconnect vms via one or more linux bridges.
- Linux bridges are essentially virtual switches
- What is switching? Map mac addresses to ports, forward frames accordingly
- Connect two (or more) Ethernet segments together in a protocol independent way. Packets are forwarded based on Ethernet address, rather than IP address
- On multiple hosts create a bridge for every vlan and add the vlan interface as a member => vms on the same layer 2
- Do we need STP? No.
- Can be used to interconnect containers too (bridge + veth + namespaces, hello docker)
- 'brctl' and 'bridge' commands to interact with the linux bridge.


Juniper
QFX 5100
hardware node
hardware node
bond0
bond0
bond0.104
bond0.992
bond0.104
bond0.992
vlan104
vlan992
vlan104
vlan992
tap interfaces
tap interfaces
tap interfaces
tap interfaces

Bridged networks (2)

- Bridged networks are simple and effective:
  - minimum configuration
  - nice bandwidth achieved
  - expected networking features just work
- Pros when vms reside on the same layer2:
  - Broadcast works => ARP works
  - Multicast works => VRRP works
- Cons when vms reside on the same layer2:
  - ARP poisoning, MAC address spoofing, IP address stealing
- What happens when a vm migrates?
  - MAC persists, ARP not changing, (juniper) switch sees mac on a different port
- Problems?
  - IGMP snooping enabled in 3.16 => neigbor solicitations dropped => IPv6 not working within the vlan, summer 2015
  - Once packets where flooded in juniper QFX5100 and got reflected, leading to mac learning craziness ('bridge monitor all'), early 2017

Bridged networks(3)

Config as simple as:

```
auto vlan104
    iface vlan104 inet manual
    bridge_ports   bond0.104
```

```
➜ ena.test ~  # brctl show vlan992
bridge name     bridge id               STP enabled     interfaces
vlan992         8000.00262d0062f8       no              bond0.992
                                                        tap0
```

```
➜ ena.test ~  # brctl showmacs vlan992
port no mac addr                is local?       ageing timer
  1     00:00:0c:9f:f0:01       no                1.39
  1     00:05:73:a0:00:01       no                2.69
  1     00:26:2d:00:62:f8       yes               0.00
  1     00:26:2d:00:62:f8       yes               0.00
  2     02:61:dd:8d:15:2f       yes               0.00
  2     02:61:dd:8d:15:2f       yes               0.00
  1     64:a0:e7:42:ca:c1       no                0.06
  1     64:a0:e7:42:dc:c1       no               16.35
  2     aa:00:00:94:ed:49       no                5.52
```

Live Demo

Create bridge for vlan traffic

```
ip link add name vlan992 type bridge
ip link set bond0.992 master vlan992
ip link set dev vlan992 up
ip link set tap0 master vlan992
```

Routed networks

- Why “routed networks”?
  - Cloud is a zero-trust, hostile environment
  - How to host different clients’ vms in the same subnet?
  - Work around bridged networks weak points
- No flat layer2 and no switching here
- Host acts as a router for guest vms.
- In practice, the host isolates vm from the broadcast domain => broadcast and multicast from the vlan will never reach the guest vm
- We still need vlans for different subnets
- Need to apply a different routing policies
  - both between different vlans
  - and between a vlan and the host's management (native) vlan.

Routed networks (2)

- Multiple routing tables, one for each vlan/subnet.
- ip-rule rules to implement policy routing
  - Lookup vlan's routing table if incoming iface is tap or the vlan interface
- Host need to fool everyone in the vlan:
  - tells the vlan that it holds vms' IP address
  - tells the vm that it holds gateway's IP address
  - proxy_arp and proxy_ndp
  - arptables mangle source IP
- What happens on vm's migration?
  - MAC address changes
    - ARP needs update, GARP performed
    - Neighbor solicitation too
- Prevent IP spoofing with iptables rules in FORWARD chain
  - `-A FORWARD -i tap0 ! -s 62.217.124.52  -j DROP`

Routed networks (3)

- More complex configuration
- Stateful with state not easily restored
- No multicast, no VRRP
- Zero visibility, this is clients’ vms, no Icinga here :’(
- Problems? Lots.
  - Multiple stale nd_proxy entries => IPv6 packets hopping around the DC
  - GARP not being sent => IPv4 traffic routed with extra hop and potential downtime
  - Redundant/wrong iptables + ip6tables rules in FORWARD chain => downtime
  - hardware node ARP replies for entire routed subnet after ‘ifdown bond0 ;ifup bond0’
  - etc etc

Routed networks

Live demo: convert the bridged vm to routed

vlan993, default gateway 62.217.124.49, arp ip 62.217.124.54, vm’s IP 62.217.124.53

```
ip link add link bond0 name bond0.993 type vlan id 993
ip link set dev bond0.993 up

echo "993 public_993" >> /etc/iproute2/rt_tables
ip r add 62.217.124.48/29 dev bond0.993 table 993
ip r add default via 62.217.124.49 dev bond0.993 table 993
ip r add 62.217.124.53 dev tap0 proto static table public_993


echo 1 > /proc/sys/net/ipv4/conf/bond0.993/proxy_arp
arptables -A OUTPUT -j mangle  -o bond0.993 --opcode 1 --mangle-ip-s 62.217.124.54
echo 1 > /proc/sys/net/ipv4/conf/tap0/proxy_arp
arptables -A OUTPUT -j mangle -o tap0 --opcode 1 --mangle-ip-s 62.217.124.49

ip rule add iif bond0.993 lookup public_993
ip rule add iif tap0 lookup public_993
```

Networking at GRNET cloud

GRNET ViMA clusters:

- Routed networks for clients' vms
- Bridged networks for managed/puppetized vms running services
- Bridged networks for client dedicated vlans

GRNET ~okeanos clusters:

- Routed networks for clients' vms, different vlans/subnets/routing tables/interfaces for v4 and v6 :(
- A single vlan+bridge (prv0) for private networks via the mac-filtered networks trick

gnt-networking: unified(?) ganeti networking software

Tha future?

- Open Vswitch can be spotted on the horizon. A way to easily and scalably provide vlans over layer 3
- No VRRP for ~okeanos vms, could we fix that?
- How to provision a vlan with public addresses to a client?
- Cross DC networking ?
- Network announcements from servers?

Links

Bonding:

- https://www.kernel.org/doc/Documentation/networking/bonding.txt
- https://wiki.debian.org/Bonding

Linux bridges:

- https://developers.redhat.com/blog/2017/09/14/vlan-filter-support-on-bridge/
- https://vincent.bernat.im/en/blog/2017-linux-bridge-isolation

Networking technologies in Linux:

- https://events.linuxfoundation.org/sites/events/files/slides/2016%20-%20Linux%20Networking %20explained_0.pdf

Gnt-networking:

- https://github.com/grnet/gnt-networking

Synnefo networks documentation:

- https://www.synnefo.org/docs/synnefo/latest/networks.html#flavors