OpenStack Neutron

Introduction and project status

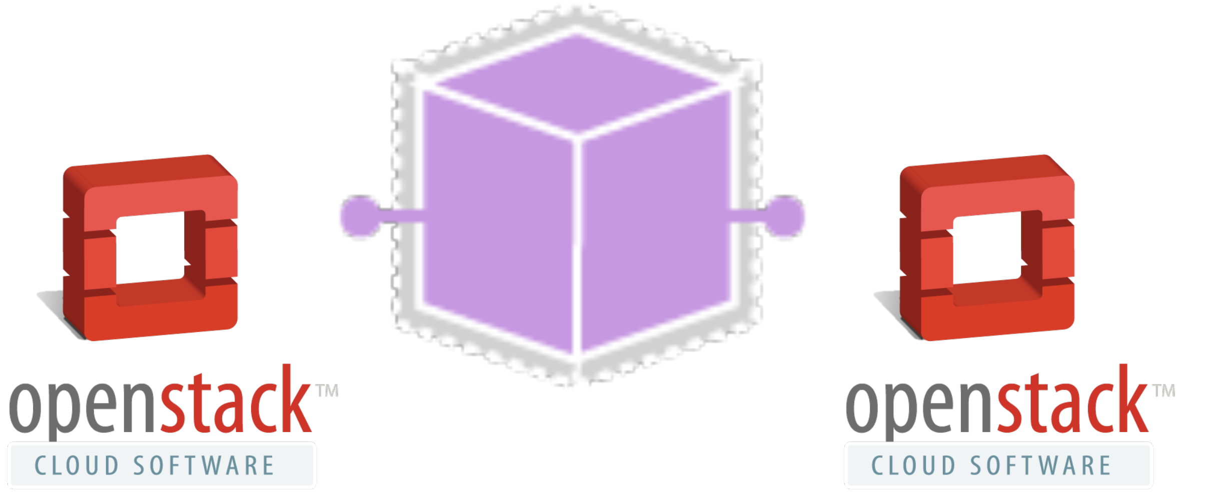
&

Use case ML2 plugin with l2 population

Summary

1. OpenStack Neutron
   - Why Neutron?
   - What's Neutron?
2. 2014.1 release
   - Please, stabilize it!
   - Features
3. ML2 plugin with l2 population mechanism driver
   - ML2 plugin
   - L2 population
   - VXLAN into Linux
   - What's next?
4. Routing HA

OpenStack Neutron

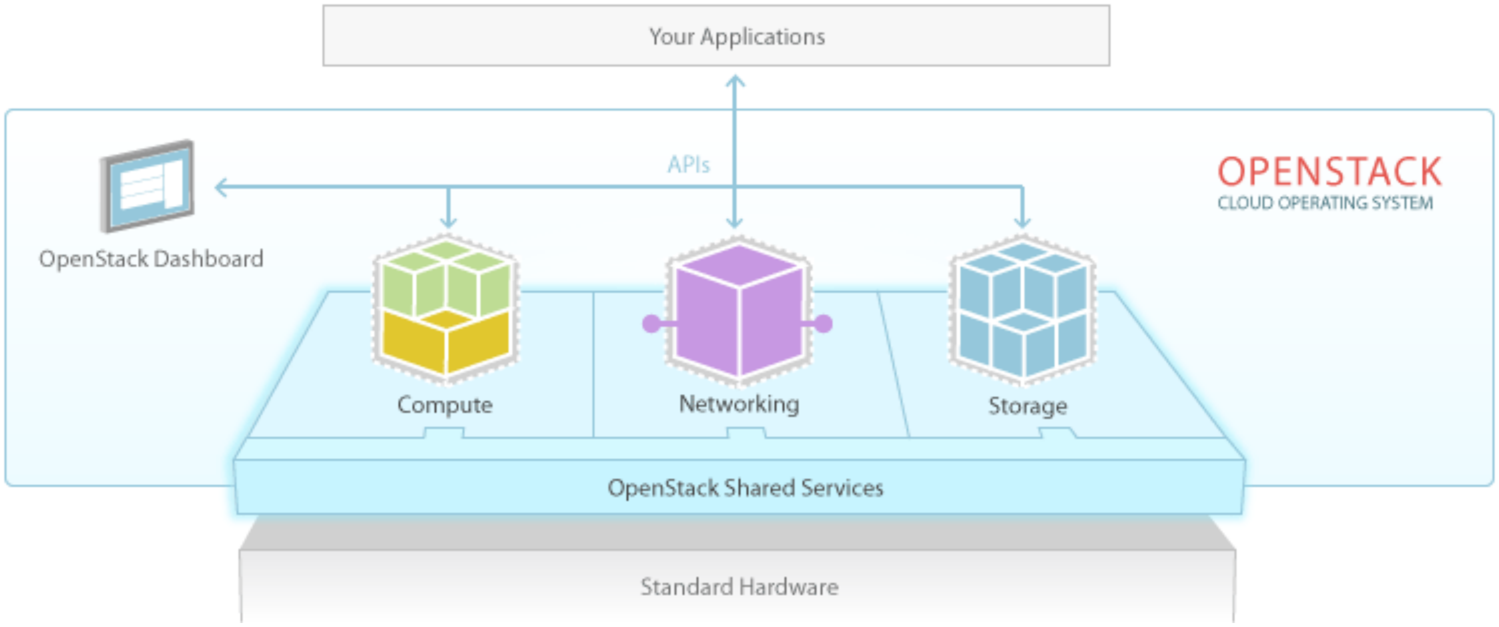


Why Neutron?

What's OpenStack:

- Open Source cloud software
- A collection of "cloud services"

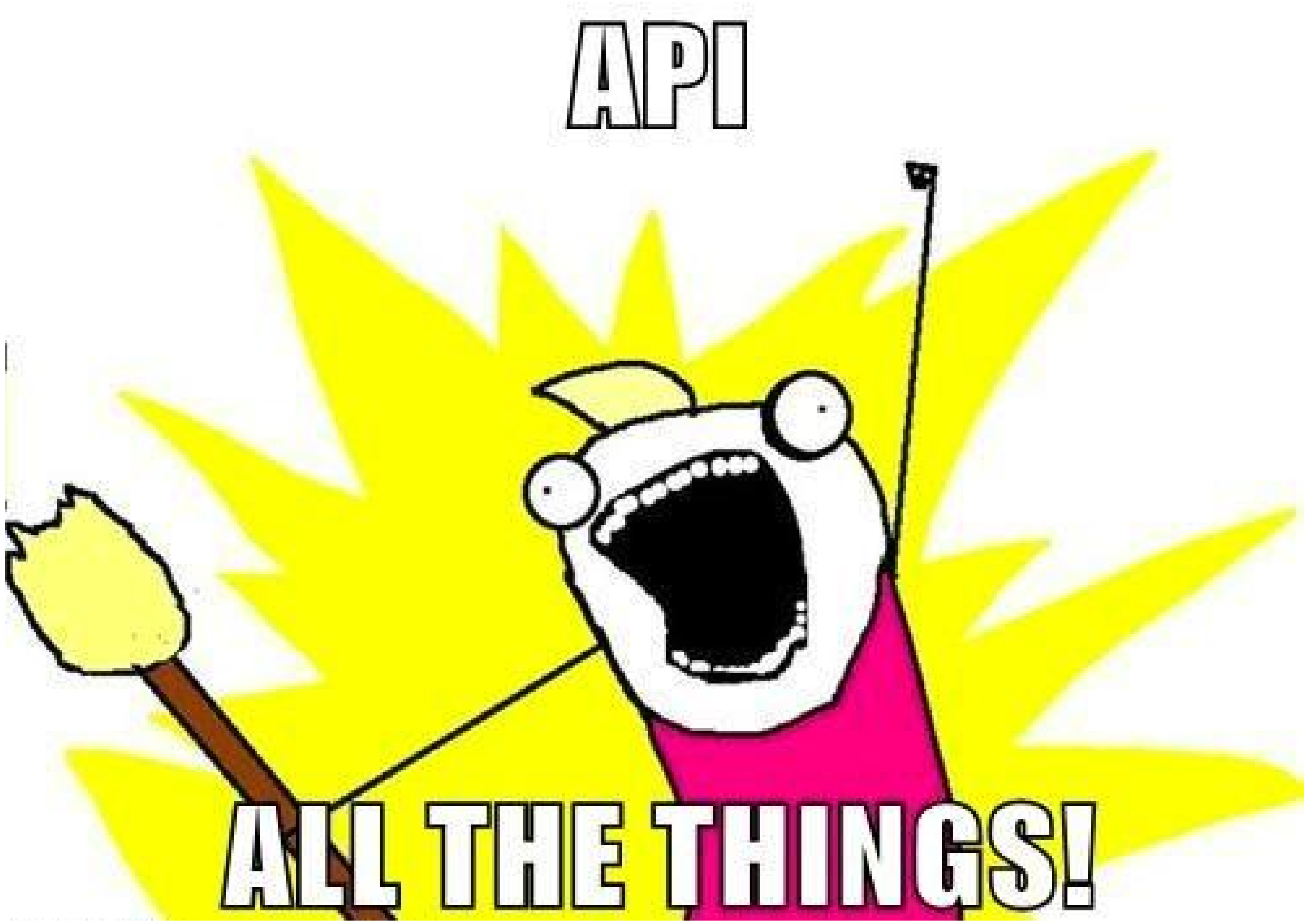
- Each service includes:
  - A tenant-facing API that exposes logical abstractions for consuming the service.
  - One or more backend implementations of that API



Why Neutron?

Compute	→	Nova
Imaging	→	Glance
Object Storage	→	Swift
Identity	→	Keystone
Networking	→	???

Why Neutron?



What’s Neutron

Neutron is an OpenStack project to provide "**Networking as a Service**" between interface devices (e.g., vNICs) managed by other Openstack services (e.g., Nova).

from OpenStack wiki

- Basic API abstraction (port, L2 network, subnet) with an eco-system of tools (CLI, GUI, API code)
- Operator selects backend to implement that core API (ML2, Open vSwitch, Linux Bridge, Nicira…)
- Extendable API to provider advanced services

2014.1 release

Please, stabilize it!

- IceHouse release was focused on stabilization of code and Neutron gate
  - tenant isolation
  - pass full tempest test suite
  - parallelized tests

  => Code Sprint in Montréal

- All third party plugin/driver need to be associated to a gate test and designate a point of contact

Features

- OVS/LB deprecated => migration script
- IPv6 improvement
- Nova ⇔ Neutron: event base
- Neutron region aware (first step)
- L3: less router scheduler
- Floating IP status
- Multiple RPC workers
- Improve SR-IOV PCI passthrough support

Plugin:

- ML2 mechanism driver:
  - Mellanox
  - Big Switch
  - Brocade
  - Open Flow (Ryu)
  - OpenDaylight
- IBM SDN-VE
- Nuage Networks

Drivers:

- LBaaS Radware

And of course, lot of bugs

316 corrected bugs during that release

But certainly not all

ML2 plugin with l2 population mechanism driver

What is Modular Layer 2?

Plugin framework allowing simultaneously utilize the variety of layer 2 networking technologies.

- Modular
  - Drivers for layer 2 networks and mechanism -- interface with agent, hardware, controllers…
- Use existing L2 agents
  - Open vSwitch
  - Linux bridge
  - HyperV
- Deprecating existing monolithic plugins

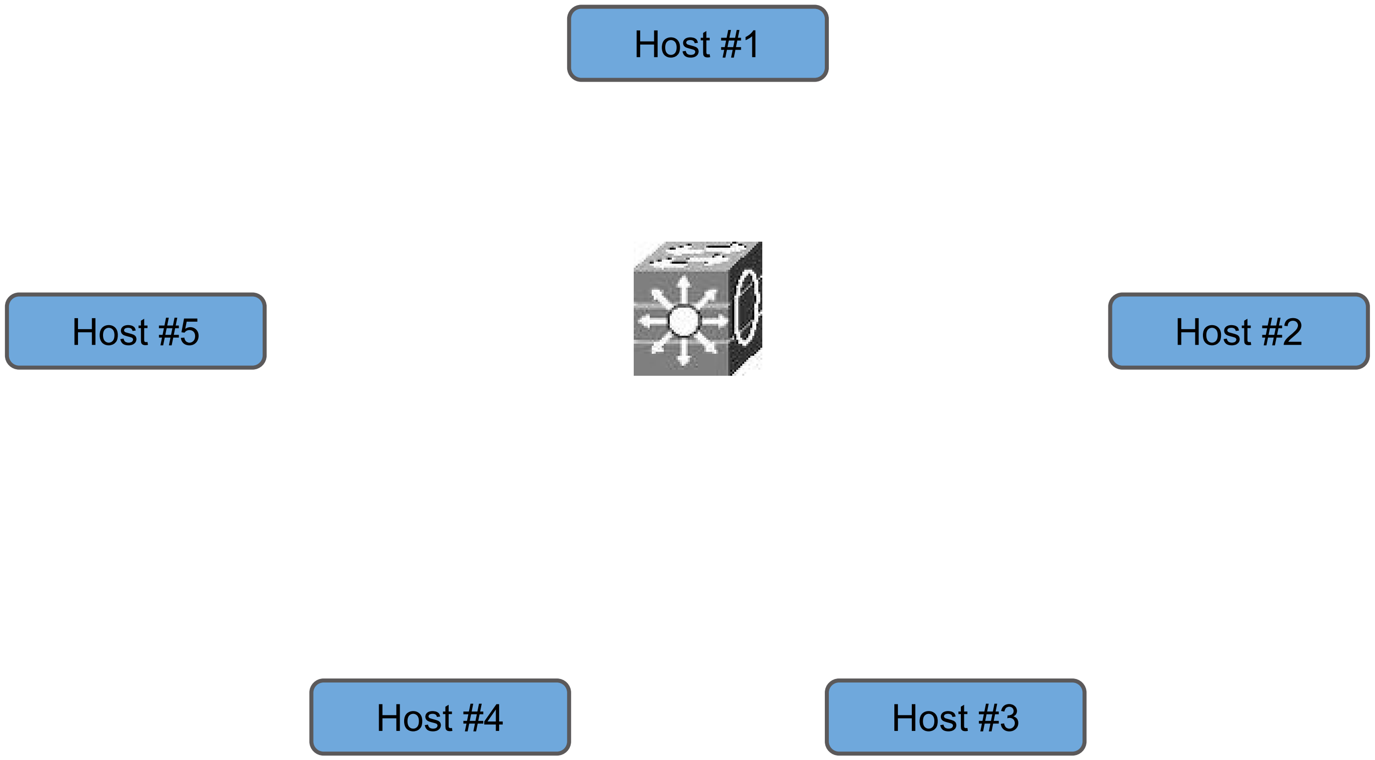
What is Modular Layer 2?

- Replace monolithic plugins
  - eliminate redundant code
  - reduce development & maintenance effort

- Ability to deploy multiple L2 technologies in a time

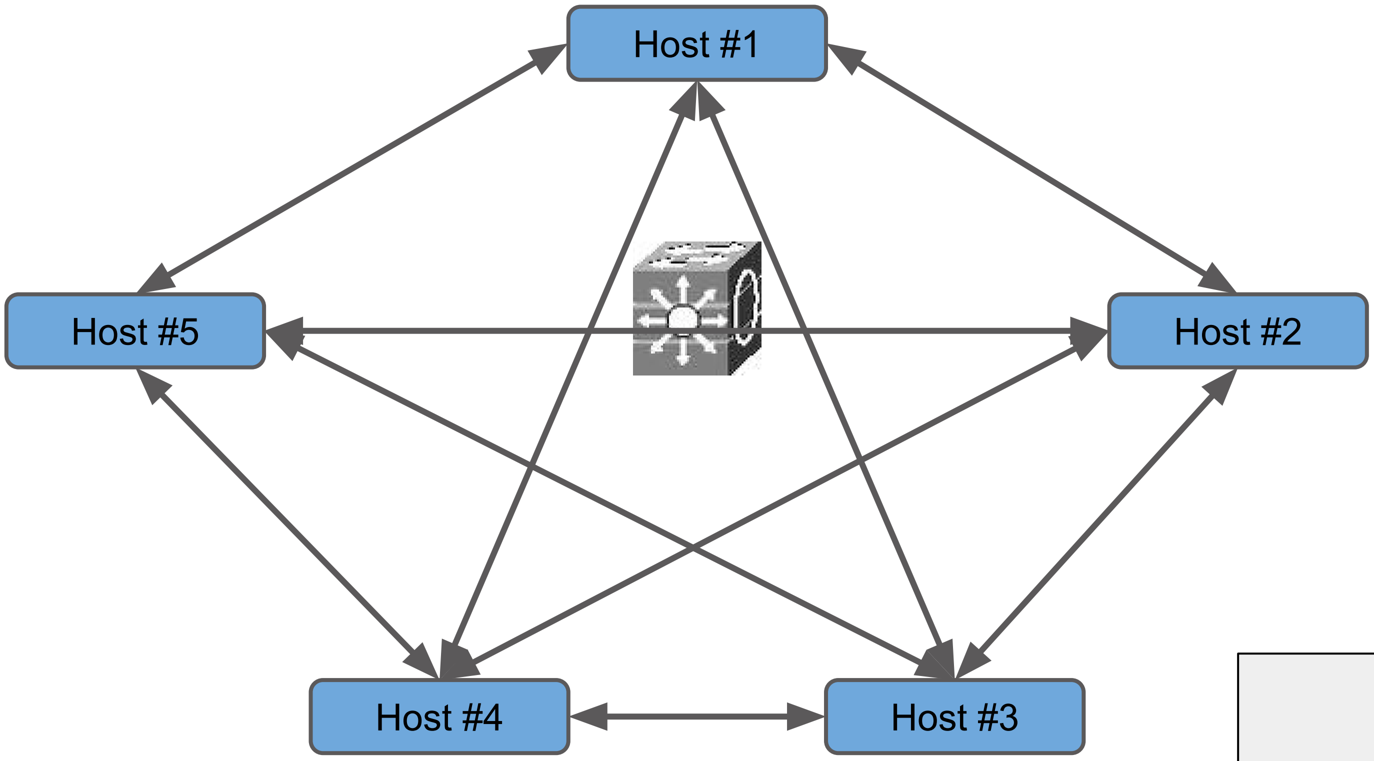
- Some new feature arrive with that plugin:
  - Top-of-Rack switch control (Arista, Cisco, Big Switch)
  - L2 population (see next)

L2 population mechanism driver



L2 population mechanism driver

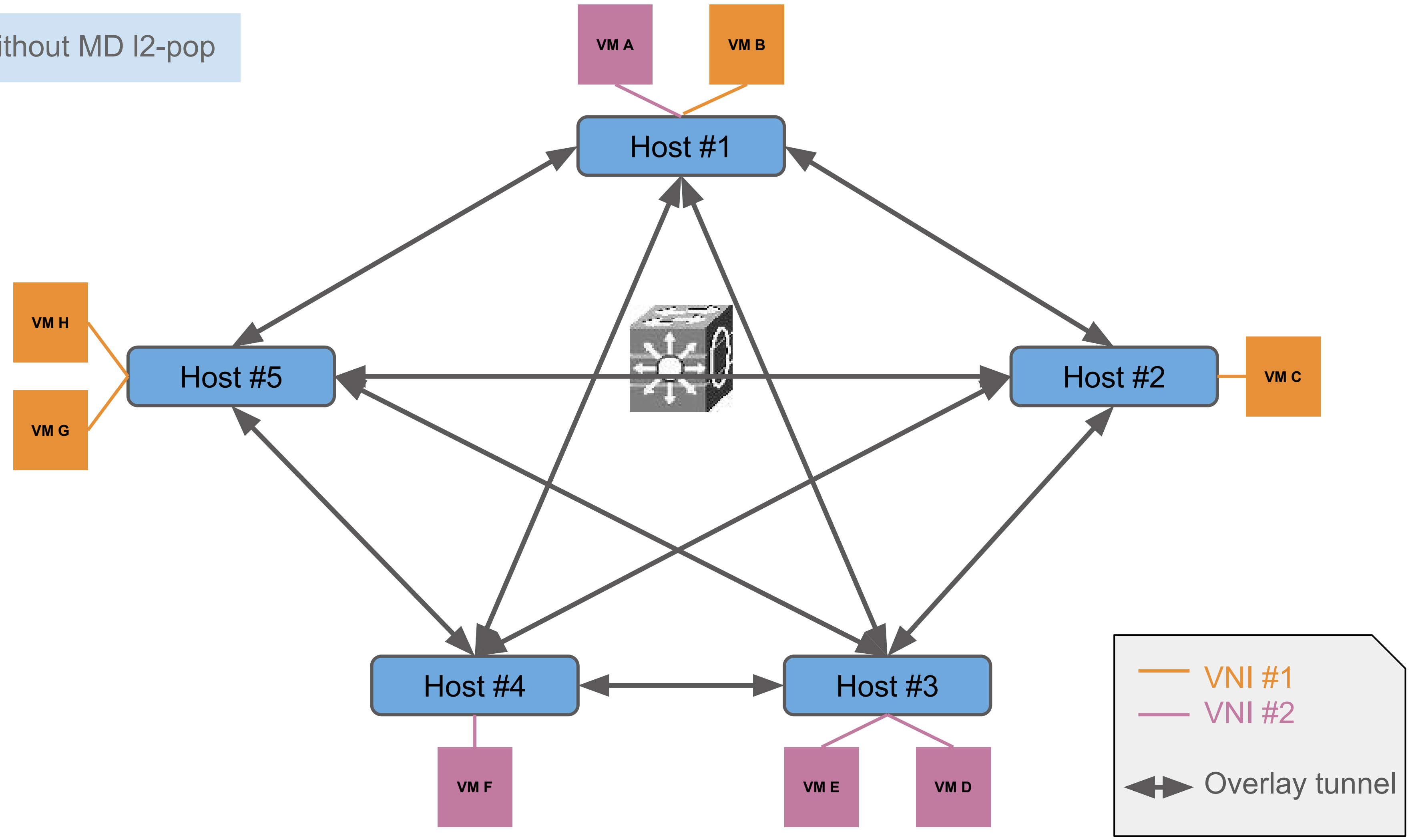
Without MD l2-pop



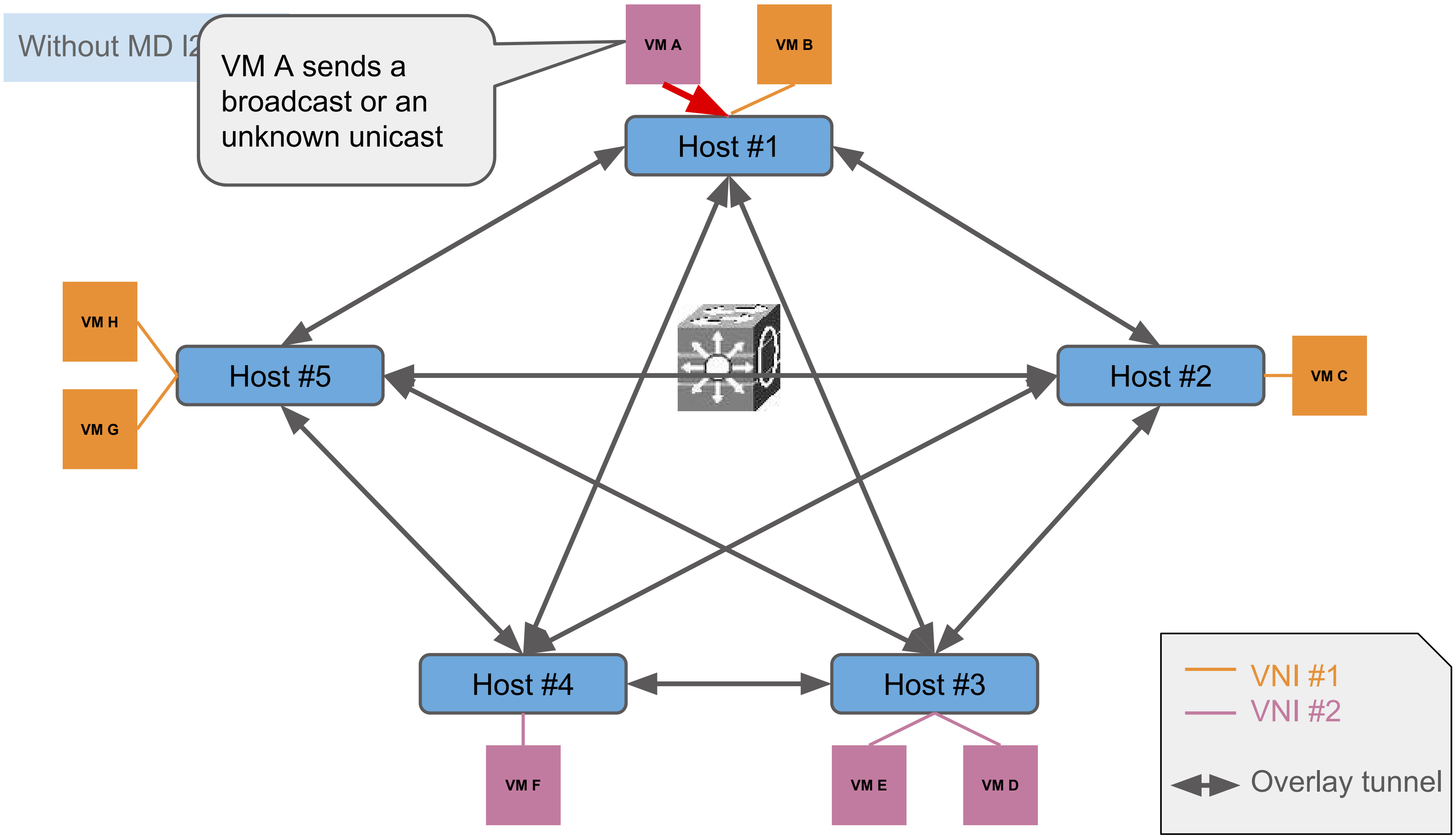
↔ Overlay tunnel

L2 population mechanism driver

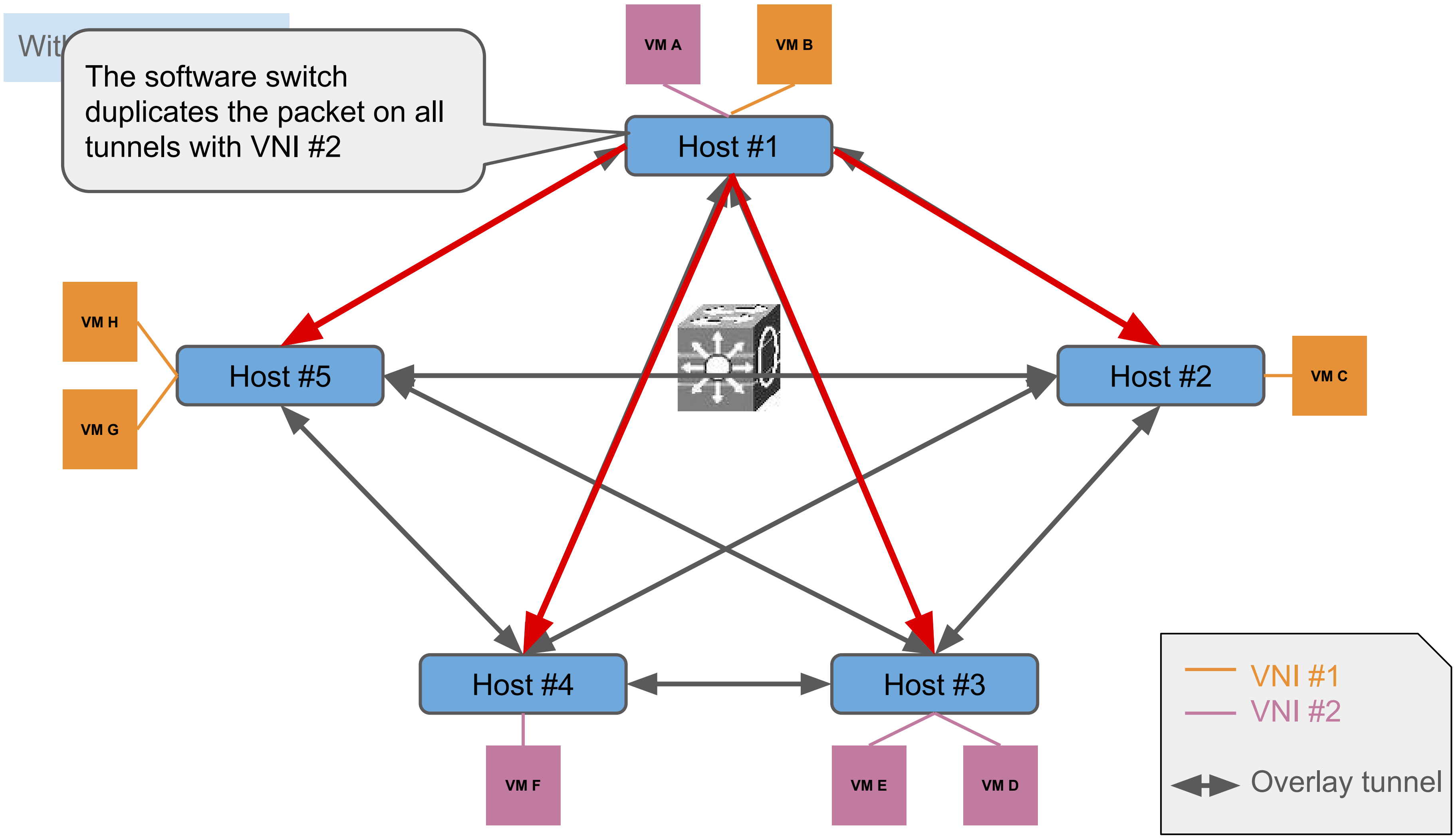
Without MD l2-pop



L2 population mechanism driver

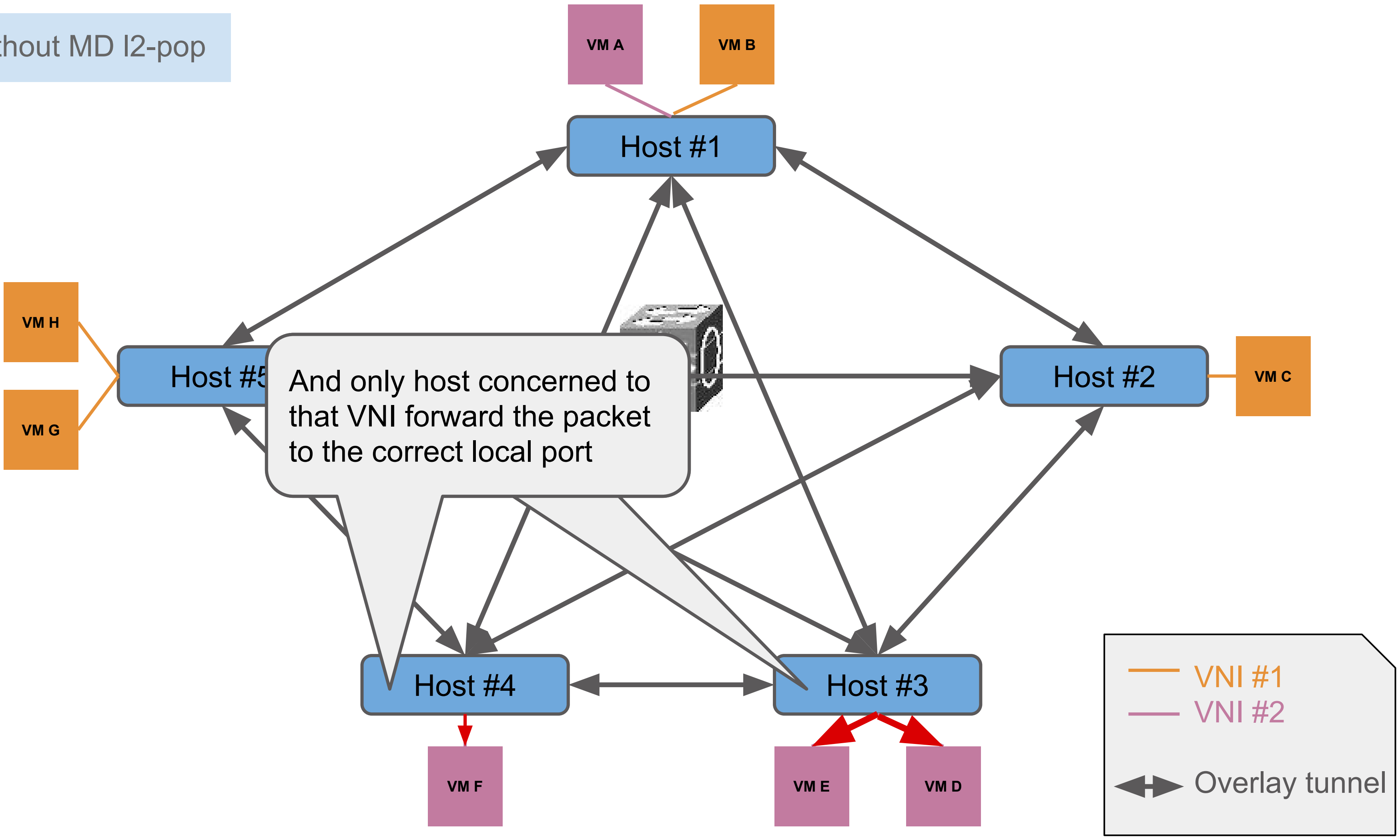


L2 population mechanism driver



L2 population mechanism driver

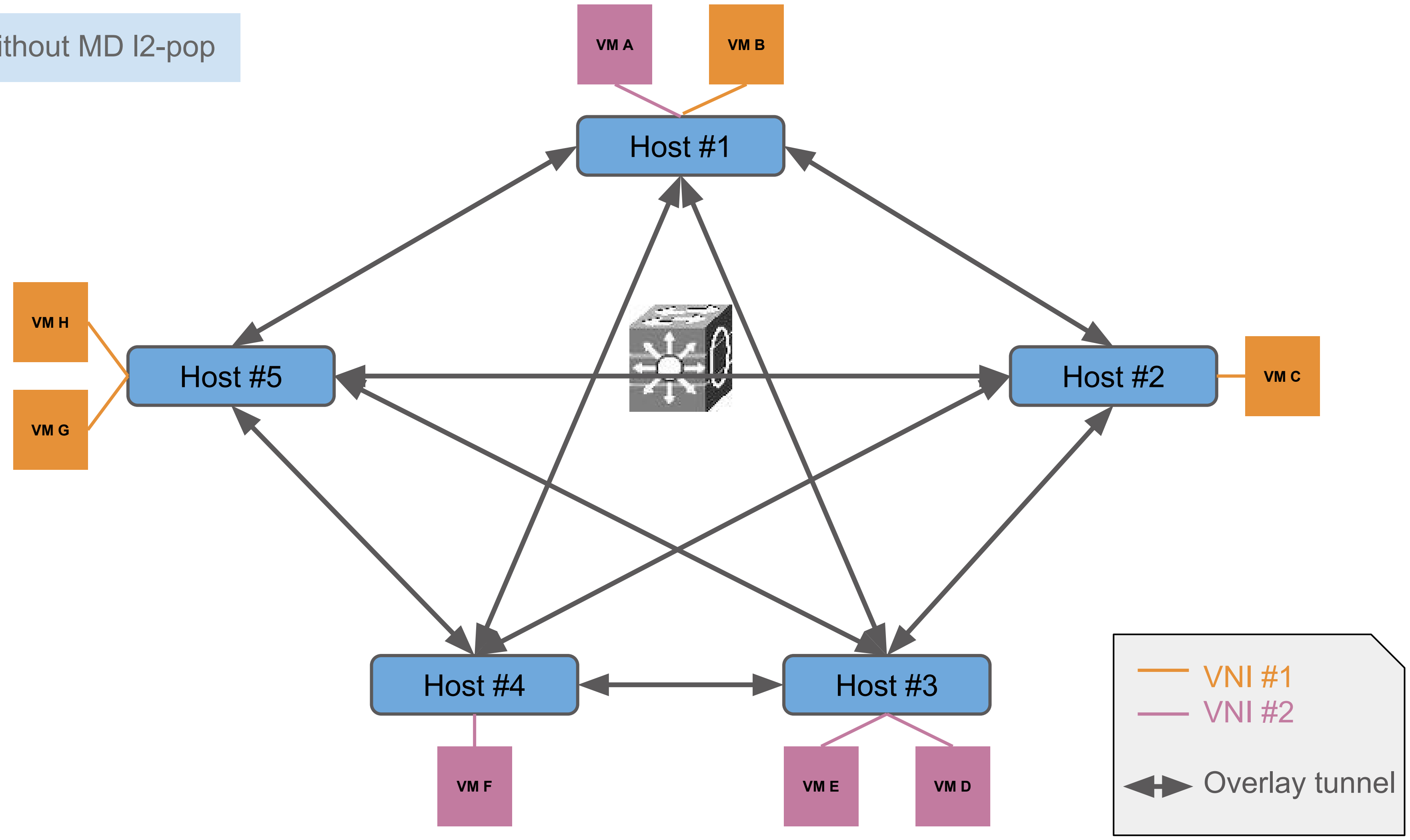
Without MD l2-pop



And only host concerned to that VNI forward the packet to the correct local port

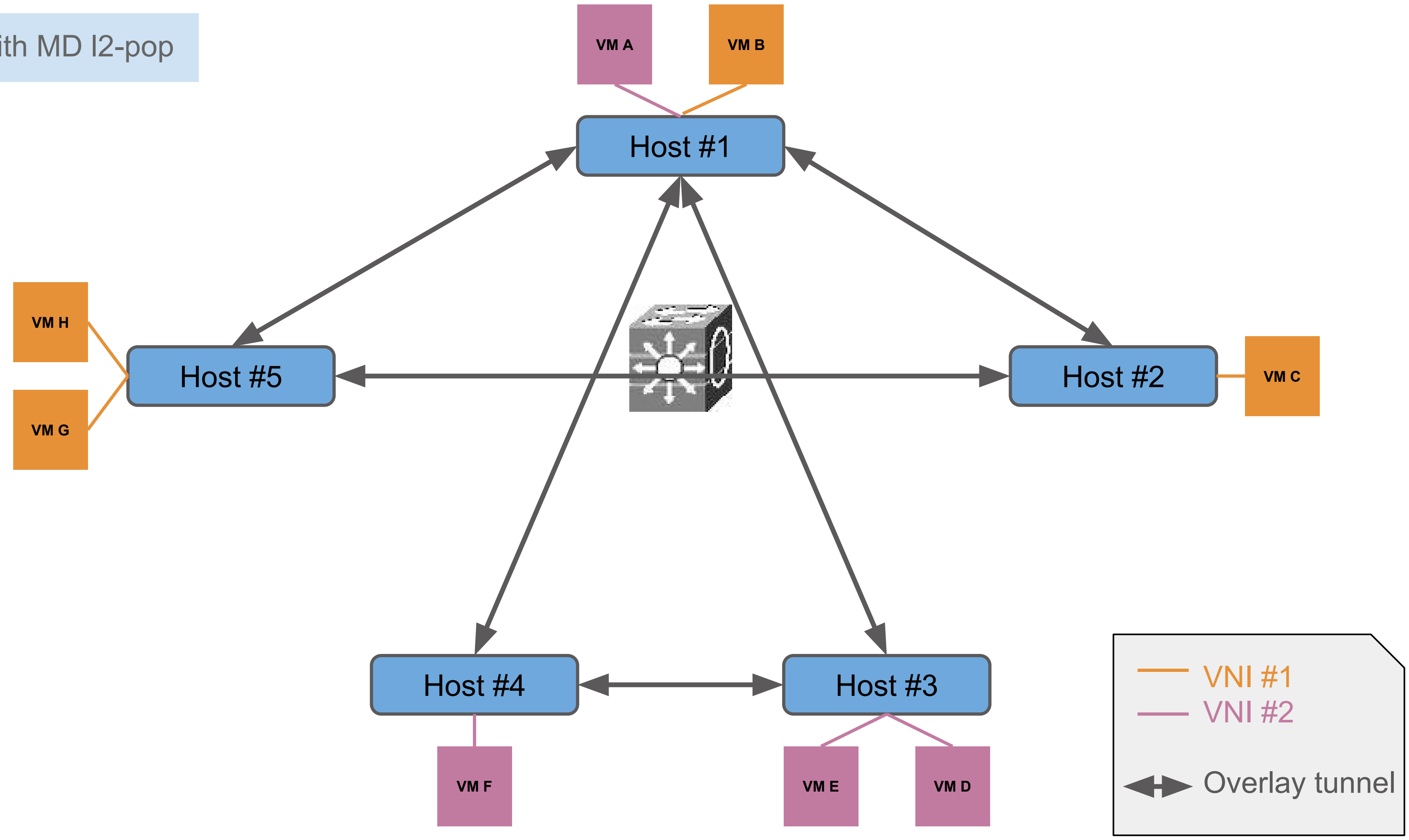
L2 population mechanism driver

Without MD l2-pop



L2 population mechanism driver

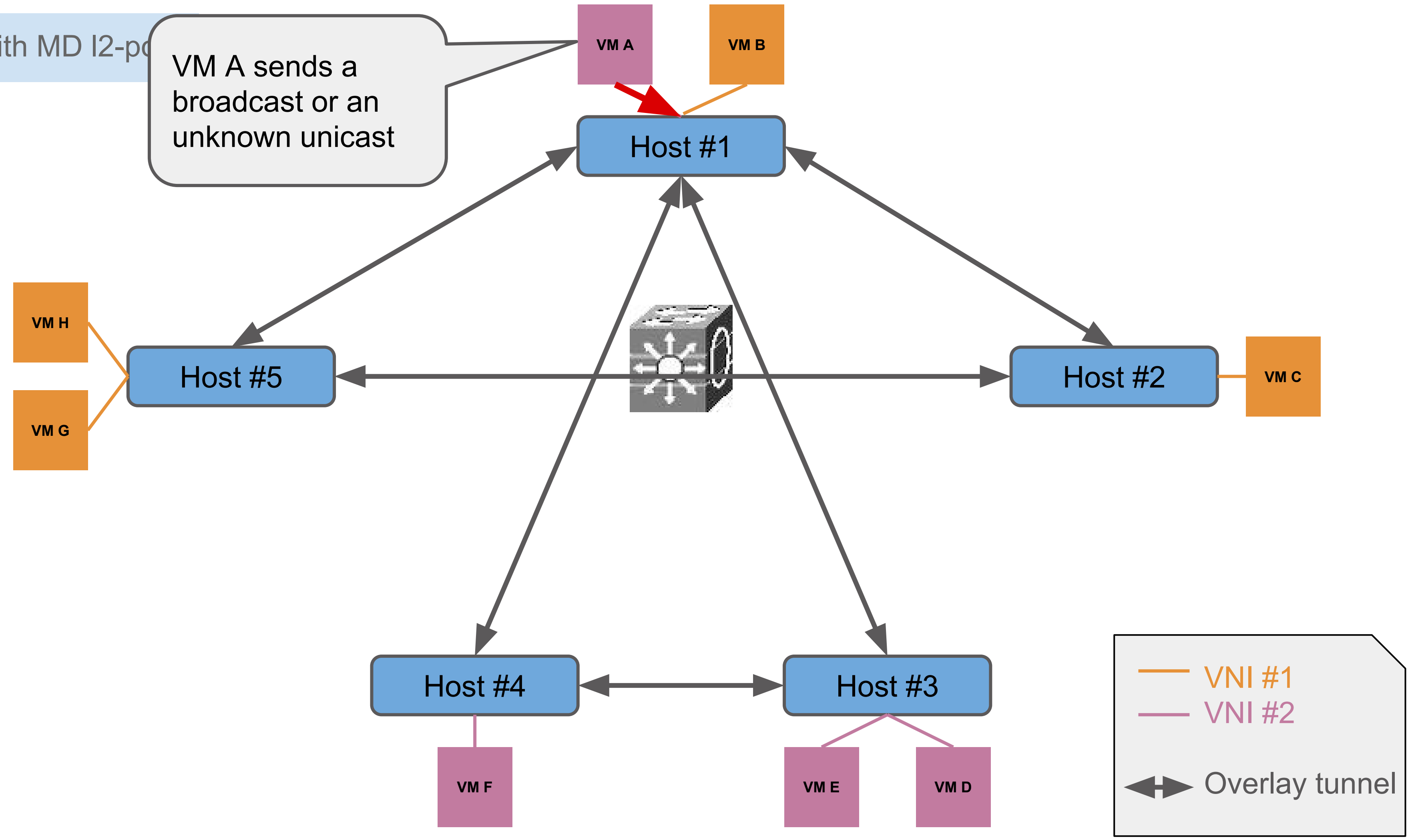
With MD l2-pop



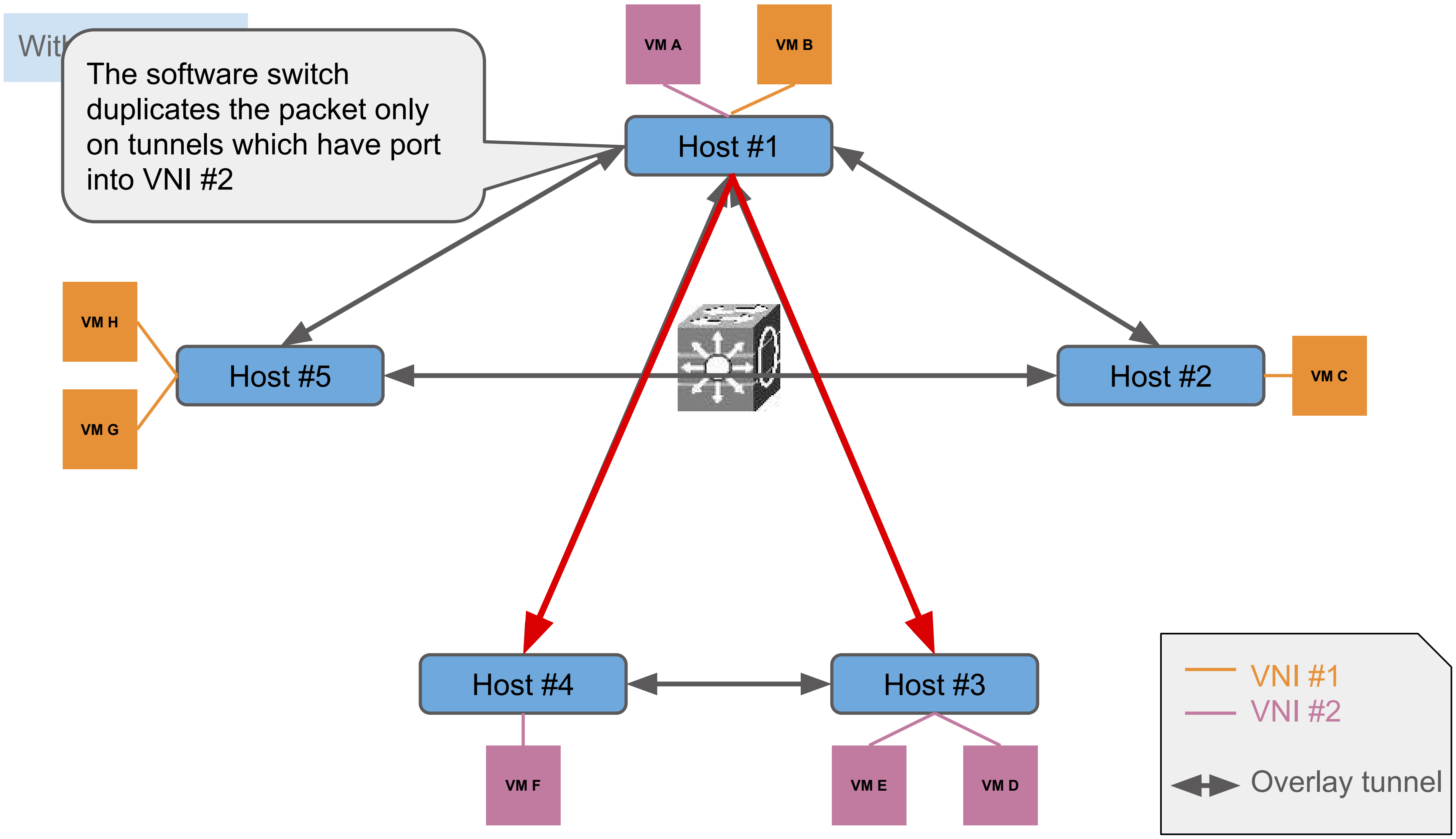
L2 population mechanism driver

With MD l2-po

VM A sends a broadcast or an unknown unicast

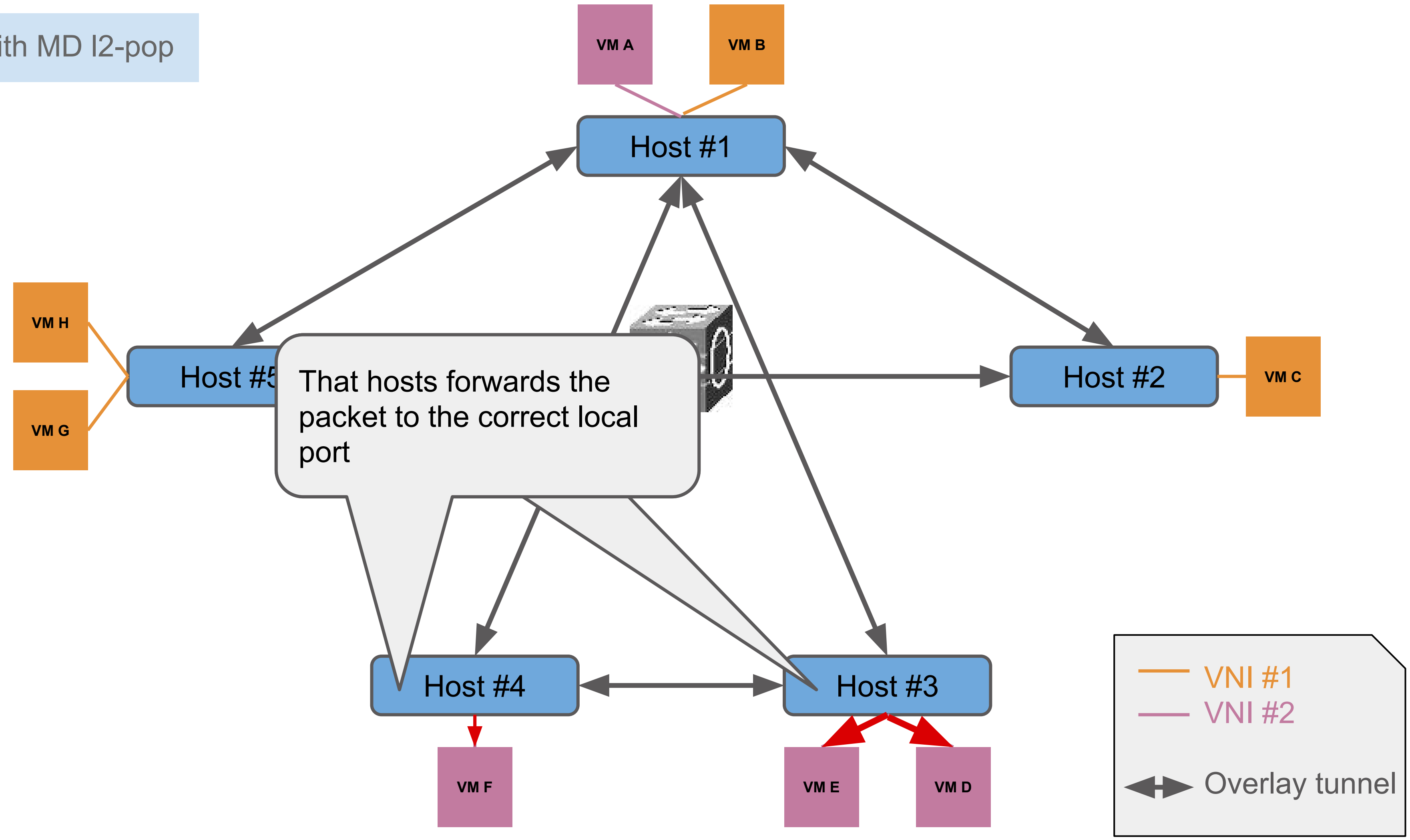


L2 population mechanism driver



L2 population mechanism driver

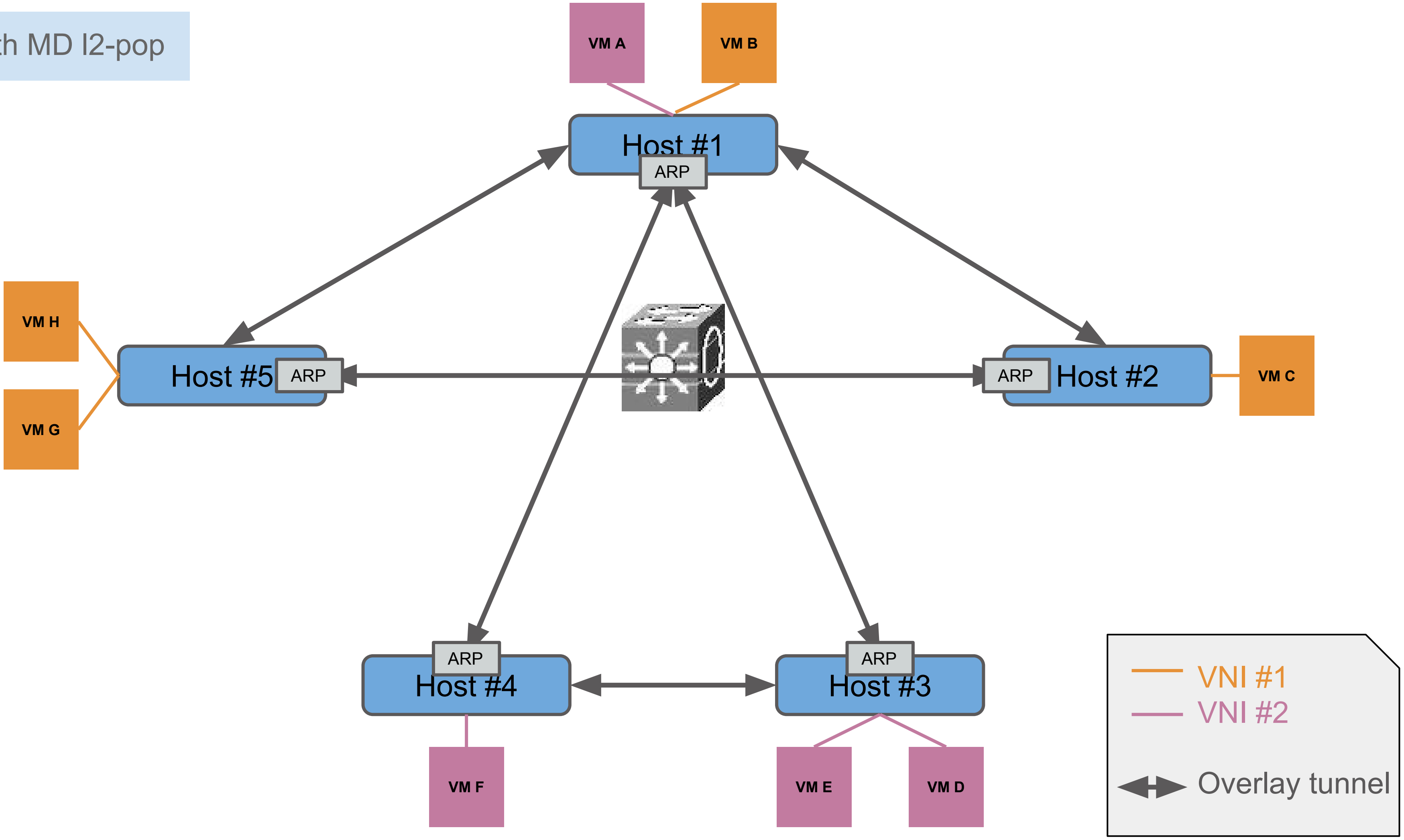
With MD l2-pop



That hosts forwards the packet to the correct local port

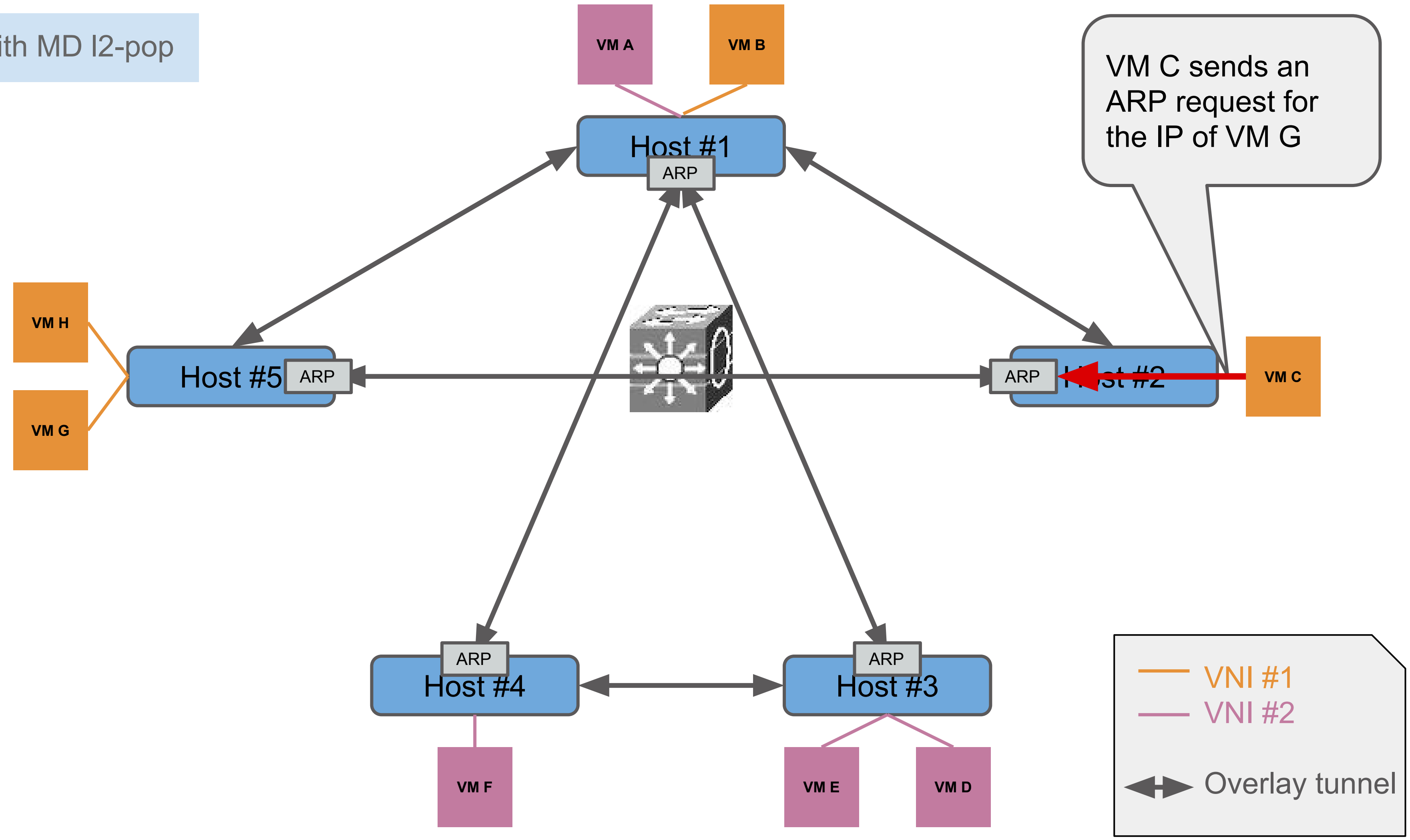
L2 population mechanism driver

With MD l2-pop



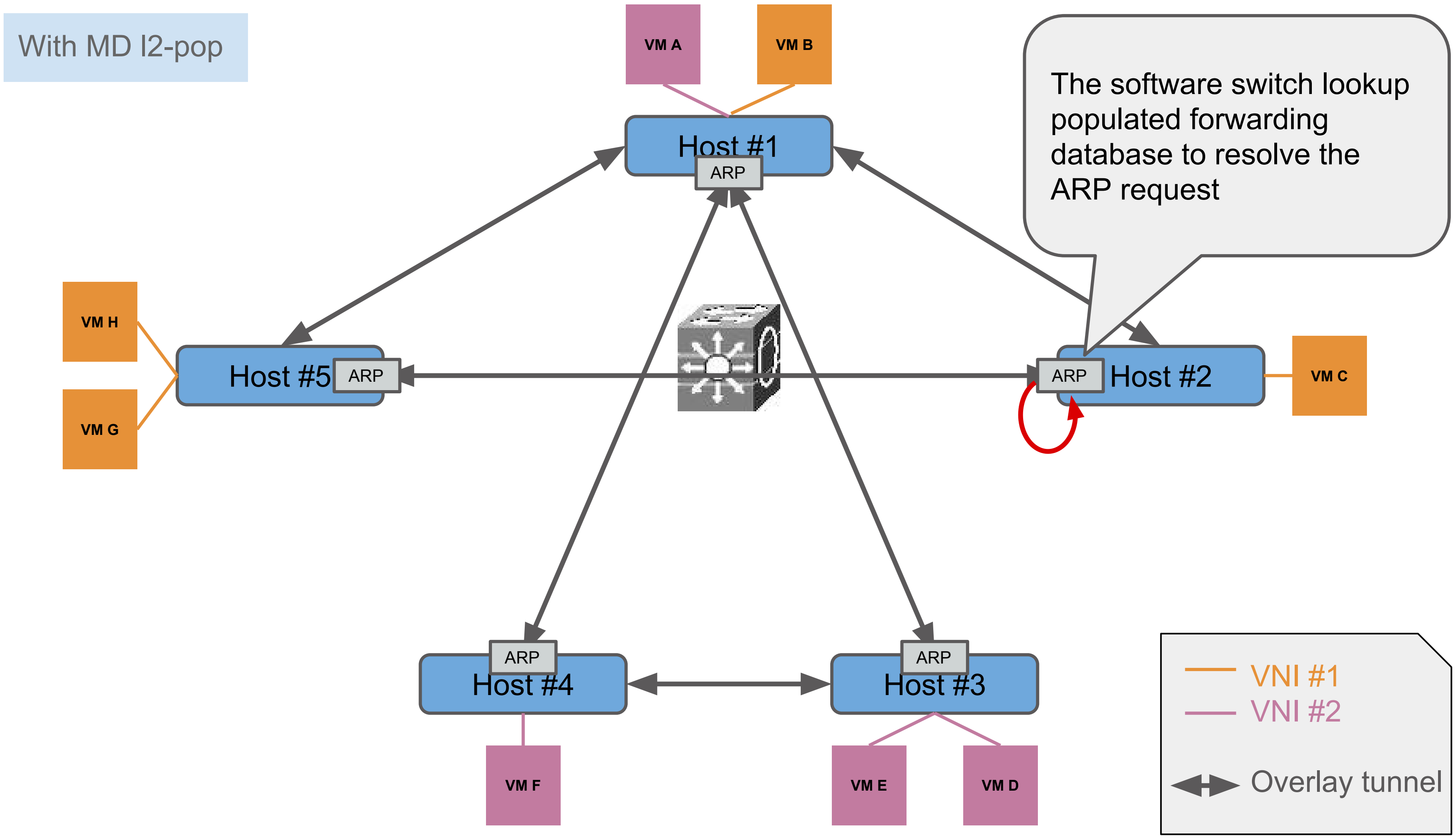
L2 population mechanism driver

With MD l2-pop



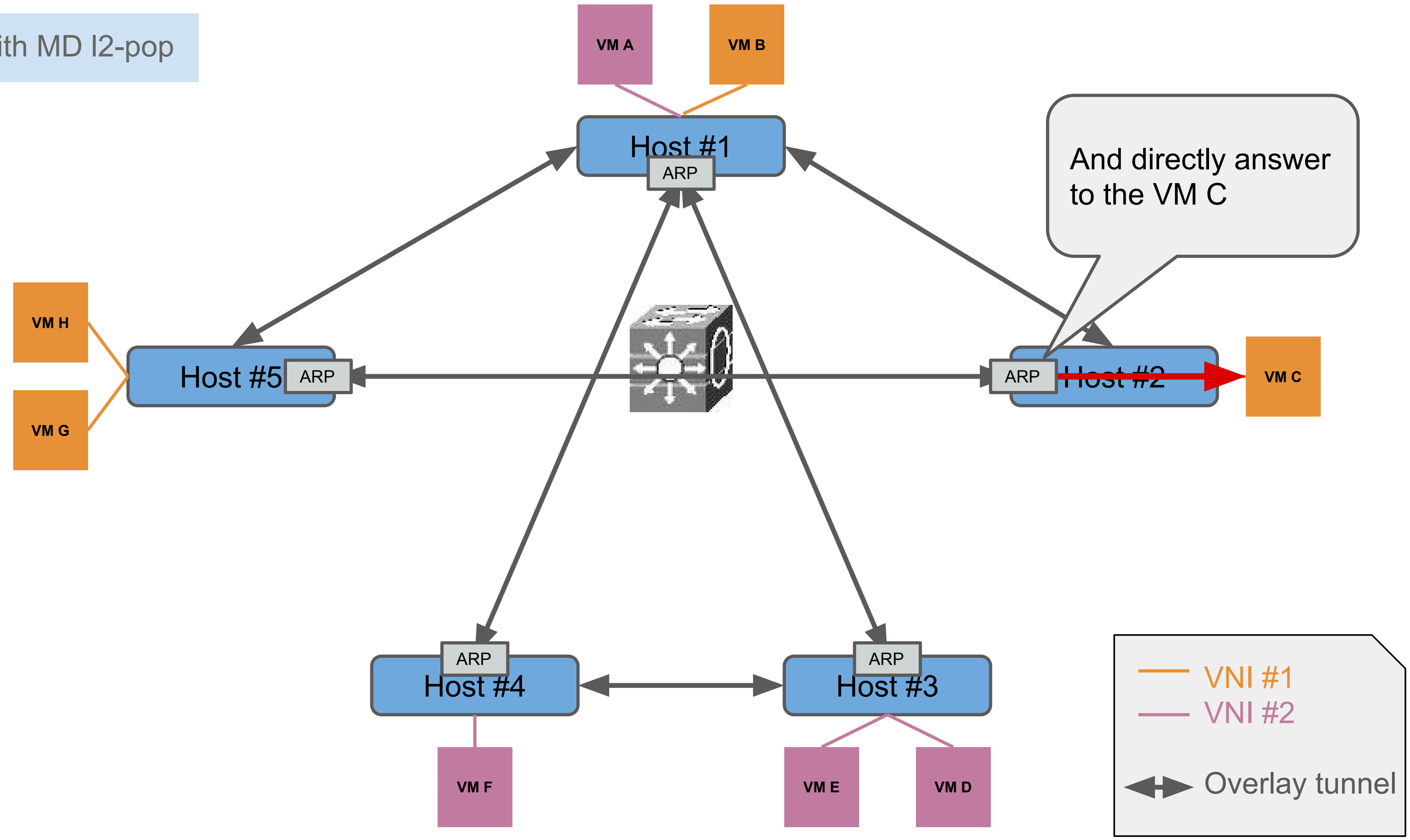
L2 population mechanism driver

With MD l2-pop

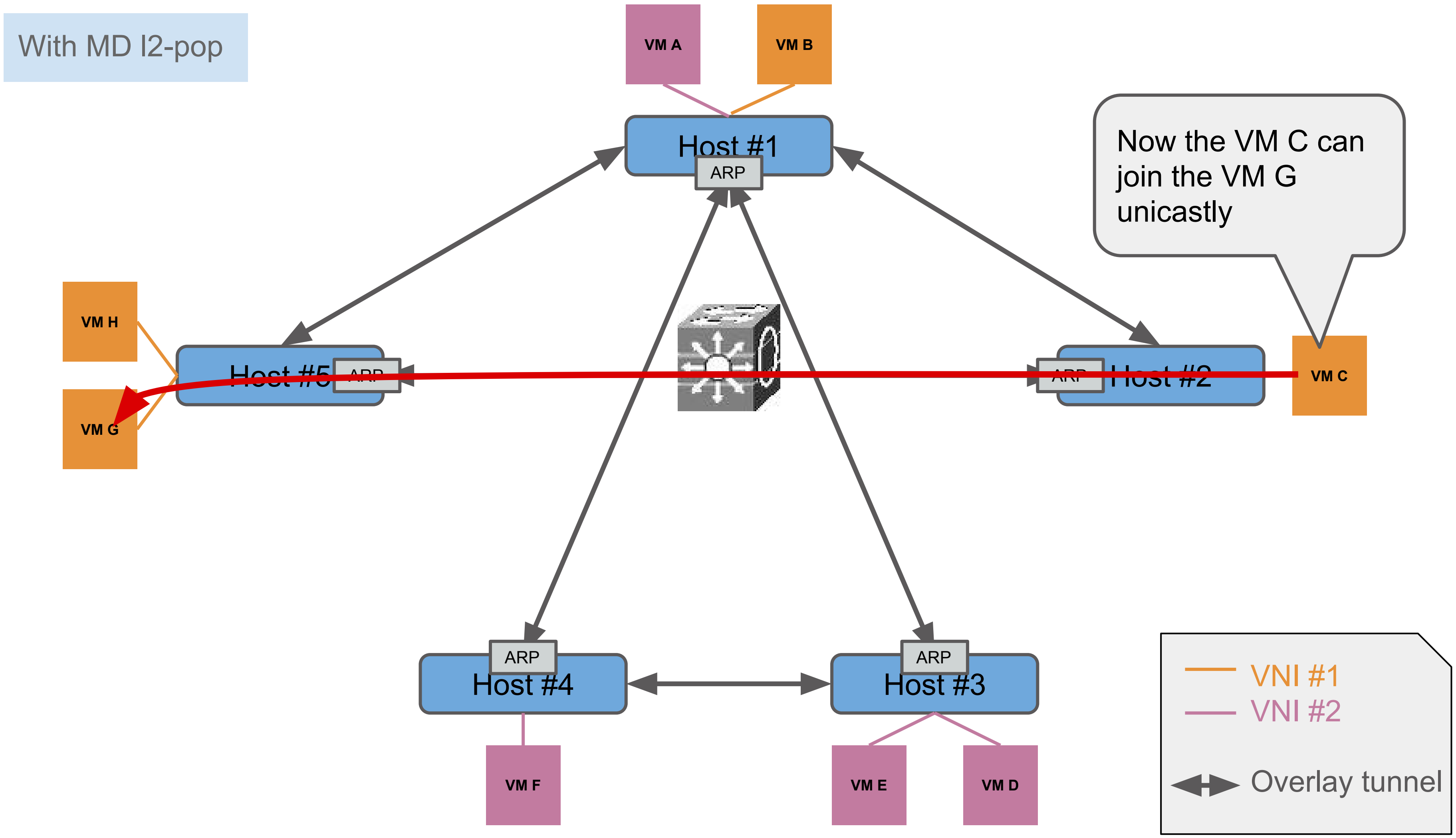


L2 population mechanism driver

With MD l2-pop



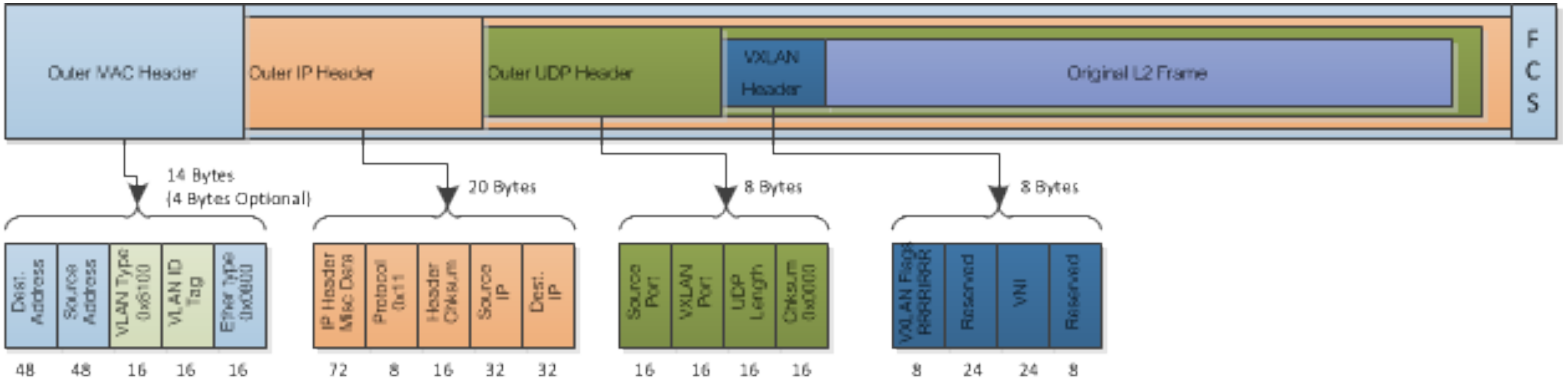
L2 population mechanism driver



VXLAN into Linux

From release 3.7 of the kernel Linux, a new module called “VXLAN” appears.

- 3.7: first experimental release
- 3.8: first stable release, no edge replication (multicast necessary),
- 3.9: edge replication only for the broadcasted packets,
- 3.11: edge replication for broadcast, multicast and unknown packets.



VXLAN into Linux

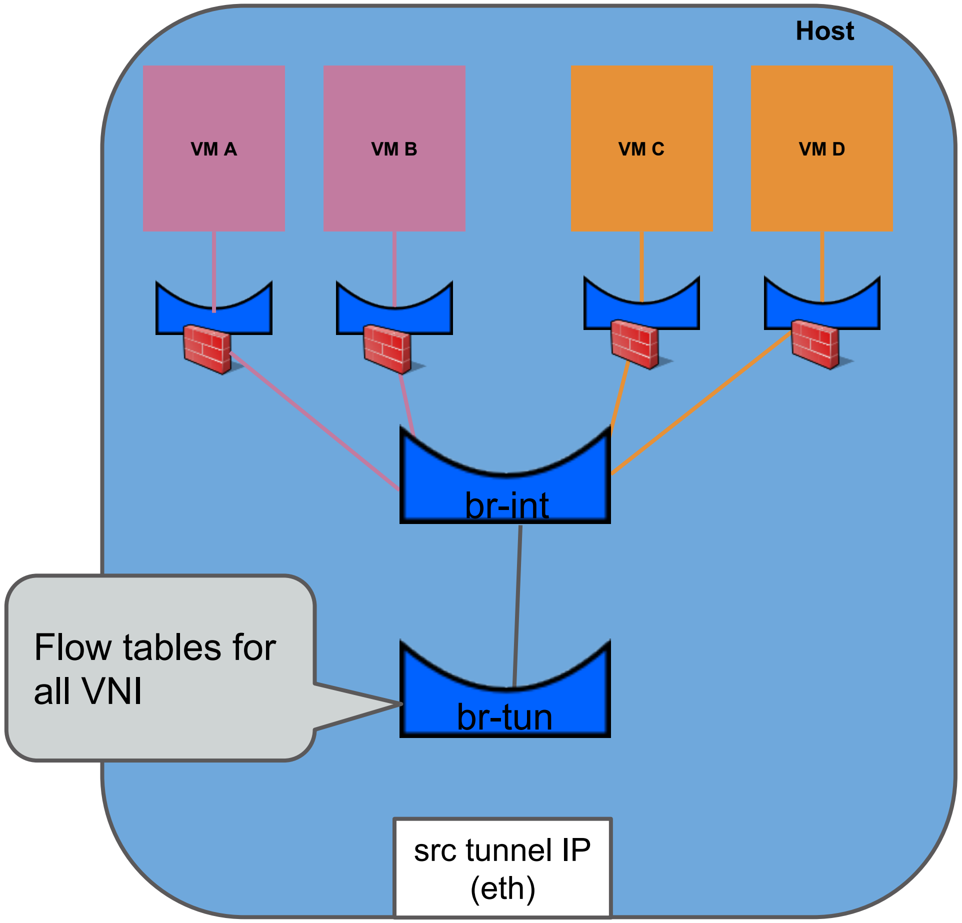
Linux bridge:

- Clearer topology
- Netfilter aware
- Integrated on recent kernel
- ARP responder aware

Open vSwitch:

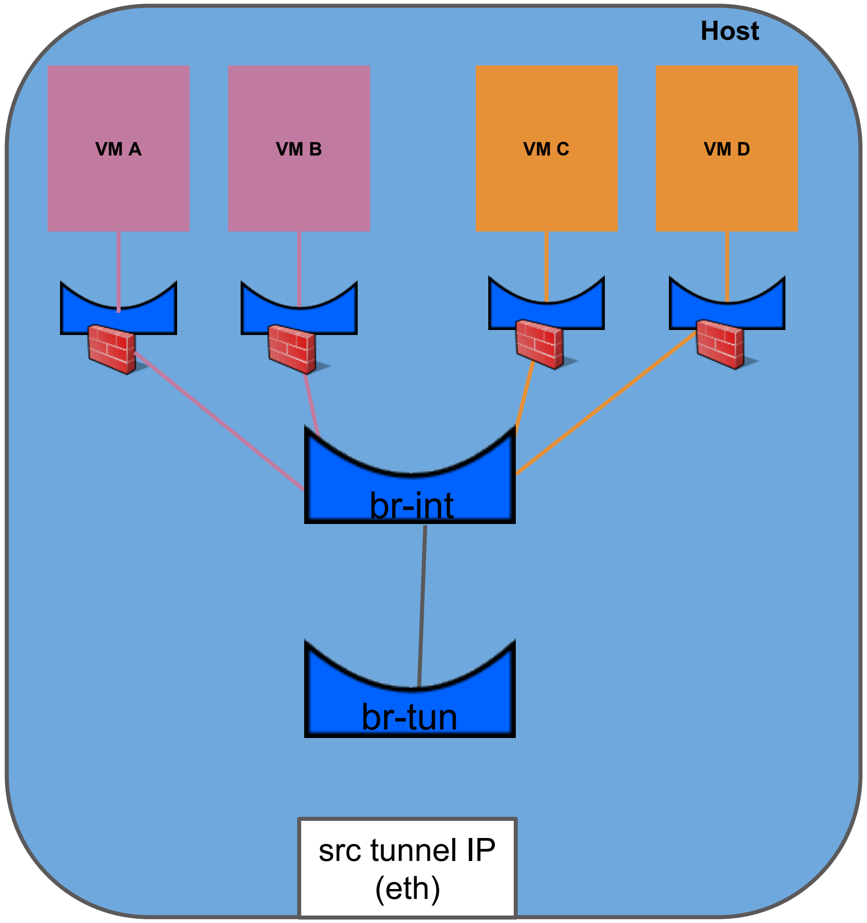
- Complex topology
- Not compatible with Netfilter
- Need to be installed
- No ARP responder

VXLAN into Linux



```
$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN…
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN…
   link/ether fe:59:d1:21:df:e9 brd ff:ff:ff:ff:ff:ff
9: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500...
   link/ether 00:ff:ff:e2:28:ff brd ff:ff:ff:ff:ff:ff
11: eth1.: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500...
   link/ether 00:ee:ee:e2:28:ff brd ff:ff:ff:ff:ff:ff
12: br-int: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc…
   link/ether 2a:ed:01:84:95:4e brd ff:ff:ff:ff:ff:ff
21: phy-br-eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500...
   link/ether a6:05:7b:85:8f:43 brd ff:ff:ff:ff:ff:ff
22: int-br-eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500...
   link/ether 62:80:22:8b:5c:db brd ff:ff:ff:ff:ff:ff
23: br-tun: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue…
   link/ether 92:91:13:ea:b6:4c brd ff:ff:ff:ff:ff:ff
28: qbr1d41986e-34: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500...
   link/ether 56:9f:e0:48:b5:5b brd ff:ff:ff:ff:ff:ff
29: qvo1d41986e-34: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether be:40:a0:65:83:88 brd ff:ff:ff:ff:ff:ff
30: qvb1d41986e-34: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 56:9f:e0:48:b5:5b brd ff:ff:ff:ff:ff:ff
31: tap1d41986e-34: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether fe:16:3e:33:38:0c brd ff:ff:ff:ff:ff:ff
32: qbrf952e707-40: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether 1a:59:e0:e5:ab:22 brd ff:ff:ff:ff:ff:ff
33: qvof952e707-40: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 3e:4e:a7:93:ee:65 brd ff:ff:ff:ff:ff:ff
34: qvbf952e707-40: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 1a:59:e0:e5:ab:22 brd ff:ff:ff:ff:ff:ff
35: tapf952e707-40: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500...
   link/ether fe:16:3e:a6:df:5e brd ff:ff:ff:ff:ff:ff
36: qbr071bfc31-4f: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether 9e:7b:b9:23:0e:de brd ff:ff:ff:ff:ff:ff
37: qvo071bfc31-4f: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether fa:6d:23:15:d6:aa brd ff:ff:ff:ff:ff:ff
38: qvb071bfc31-4f: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 9e:7b:b9:23:0e:de brd ff:ff:ff:ff:ff:ff
39: tap071bfc31-4f: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether fe:16:3e:b8:4d:a8 brd ff:ff:ff:ff:ff:ff
```

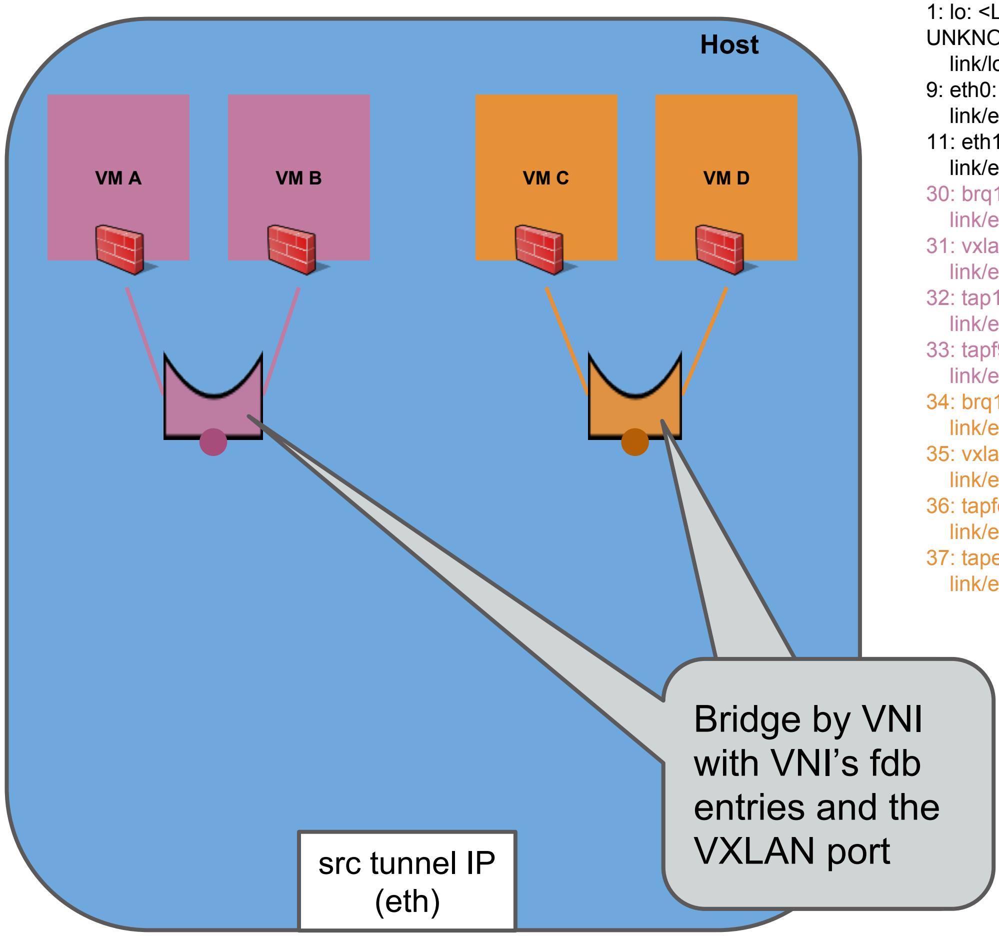
VXLAN into Linux



$ brctl show

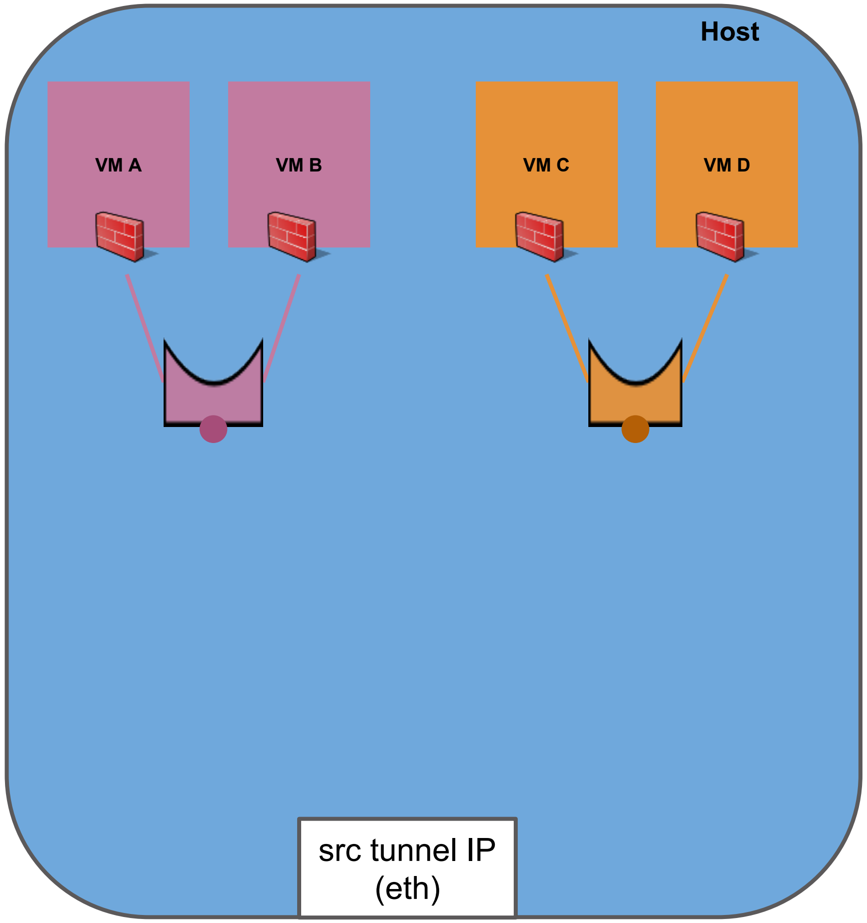
```
bridge name     bridge id               STP enabled     interfaces
qbr071bfc31-4f          8000.9e7bb9230ede       no              qvb071bfc31-4f
                                                                tap071bfc31-4f
qbr1d41986e-34          8000.569fe048b55b       no              qvb1d41986e-34
                                                                tap1d41986e-34
qbr92a4d5e1-9c          8000.b273e09878bd       no              qvb92a4d5e1-9c
                                                                tap92a4d5e1-9c
qbrf952e707-40          8000.1a59e0e5ab22       no              qvbf952e707-40
                                                                tapf952e707-40
```

VXLAN into Linux



```
$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
UNKNOWN…
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
9: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether 00:ff:ff:e2:28:ff brd ff:ff:ff:ff:ff:ff
11: eth1.: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether 00:ee:ee:e2:28:ff brd ff:ff:ff:ff:ff:ff
30: brq1d41986e-34: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 56:9f:e0:48:b5:5b brd ff:ff:ff:ff:ff:ff
31: vxlan-1000: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 56:9f:e0:48:b5:5b brd ff:ff:ff:ff:ff:ff
32: tap1d41986e-34: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether fe:16:3e:a6:df:5e brd ff:ff:ff:ff:ff:ff
33: tapf952e707-40: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether fe:16:3e:a6:df:5f brd ff:ff:ff:ff:ff:ff
34: brq123986e-ef: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 56:9f:e0:48:b5:5b brd ff:ff:ff:ff:ff:ff
35: vxlan-1001: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP>...
   link/ether 56:9f:e0:48:b5:5b brd ff:ff:ff:ff:ff:ff
36: tapfe34586e-3e: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether fe:16:3e:a6:df:5e brd ff:ff:ff:ff:ff:ff
37: tape452e347-43: <BROADCAST,MULTICAST,UP,LOWER_UP>...
   link/ether fe:16:3e:a6:df:5f brd ff:ff:ff:ff:ff:ff
```

VXLAN into Linux



```
$ brctl show
bridge name     bridge id               STP enabled     interfaces
brq1d41986e-34          8000.fe163e2fbbd1       no              vxlan-1000
                                                                 tap1d41986e-34
                                                                 tapf952e707-40
brq123986e-ef          8000.9e7bb9230ede       no              vxlan-1001
                                                                 tapfe34586e-3e
                                                                 tape452e347-43
```

What next?

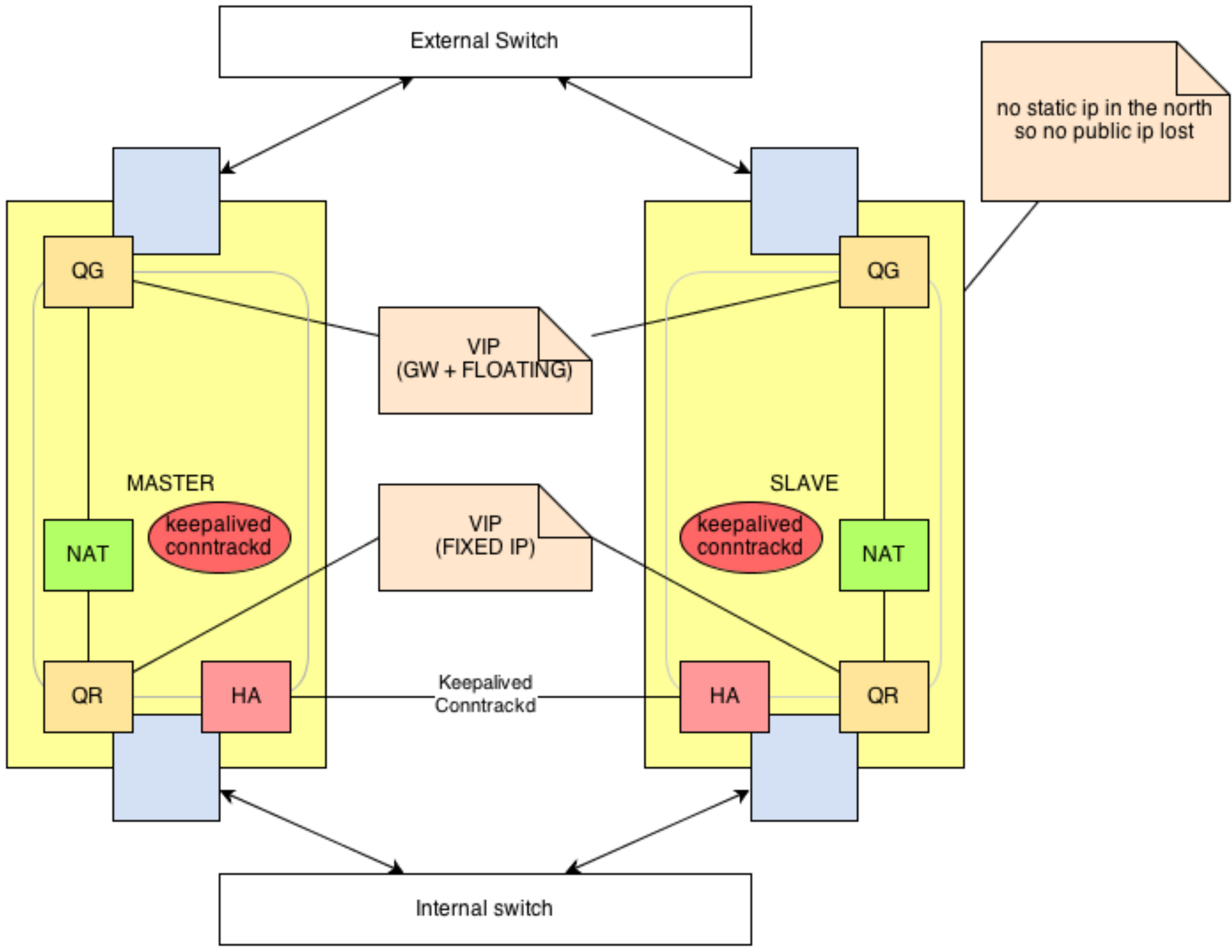
Re-think the l2-pop mechanism driver by dividing it into multiple mechanism drivers:

1. Topology MD to publish forwarding database entries
   - Try to create a topic by network
   - Agent could consume network topic according to their needs
2. Partial mesh MD to provision broadcast flows on the agent
3. ARP responder MD to populate fdb entries

Routing HA

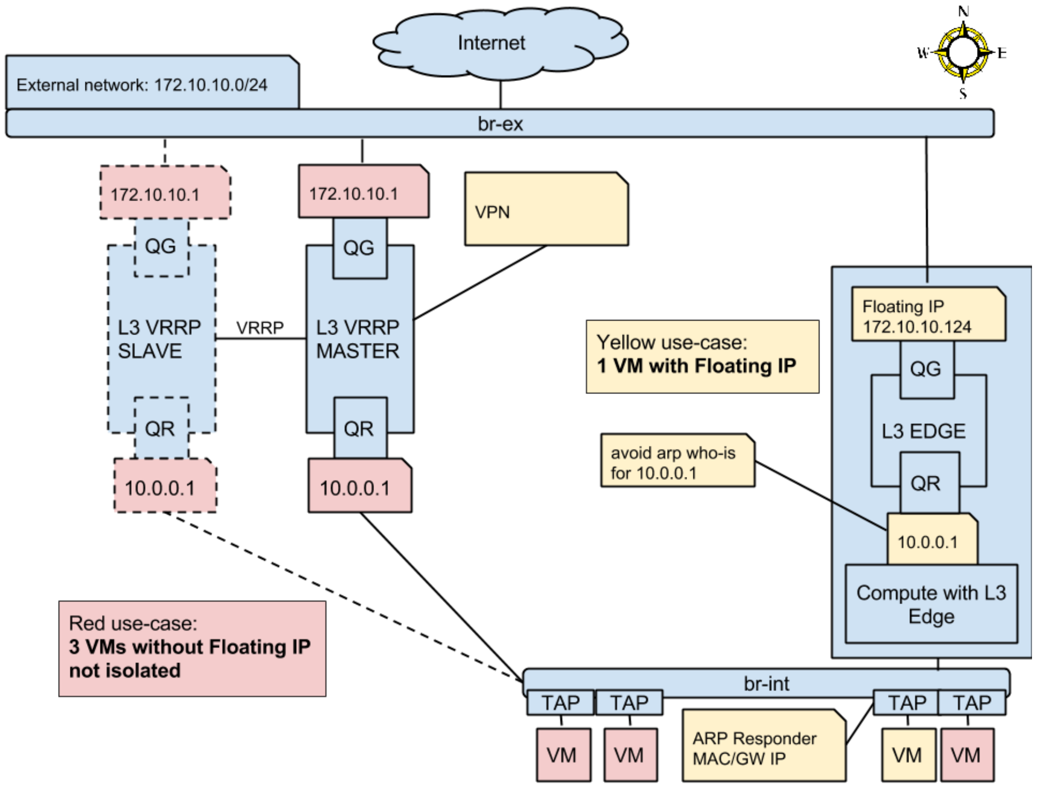
Routing HA

A first implementation based on VRRP and Conntrackd



Routing HA

Another improvement is plan for the **J** release: edge routing distribution



Questions

?

- OpenStack release status:
  http://status.openstack.org/release/
- ML2 wiki page:
  https://wiki.openstack.org/wiki/Neutron/ML2
- ML2 MD L2 population:
  https://wiki.openstack.org/wiki/L2population
- Routage HA:
  - v1: https://wiki.openstack.org/wiki/Neutron/L3_High_Availability_VRRP
  - v2: https://docs.google.com/drawings/d/1GGwbLa72n8c2T3SBApKK7uJ6WLTSRa7erTI_3QNj5Bg/edit & https://docs.google.com/document/d/1depasJSnGZPOnRLxEC_PYsVLcGVFXZLqP52RFTe21BE/edit#heading=h.5w7clq272tji