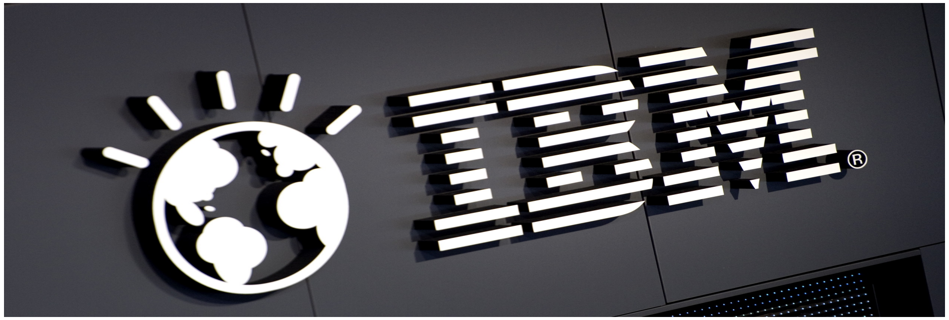
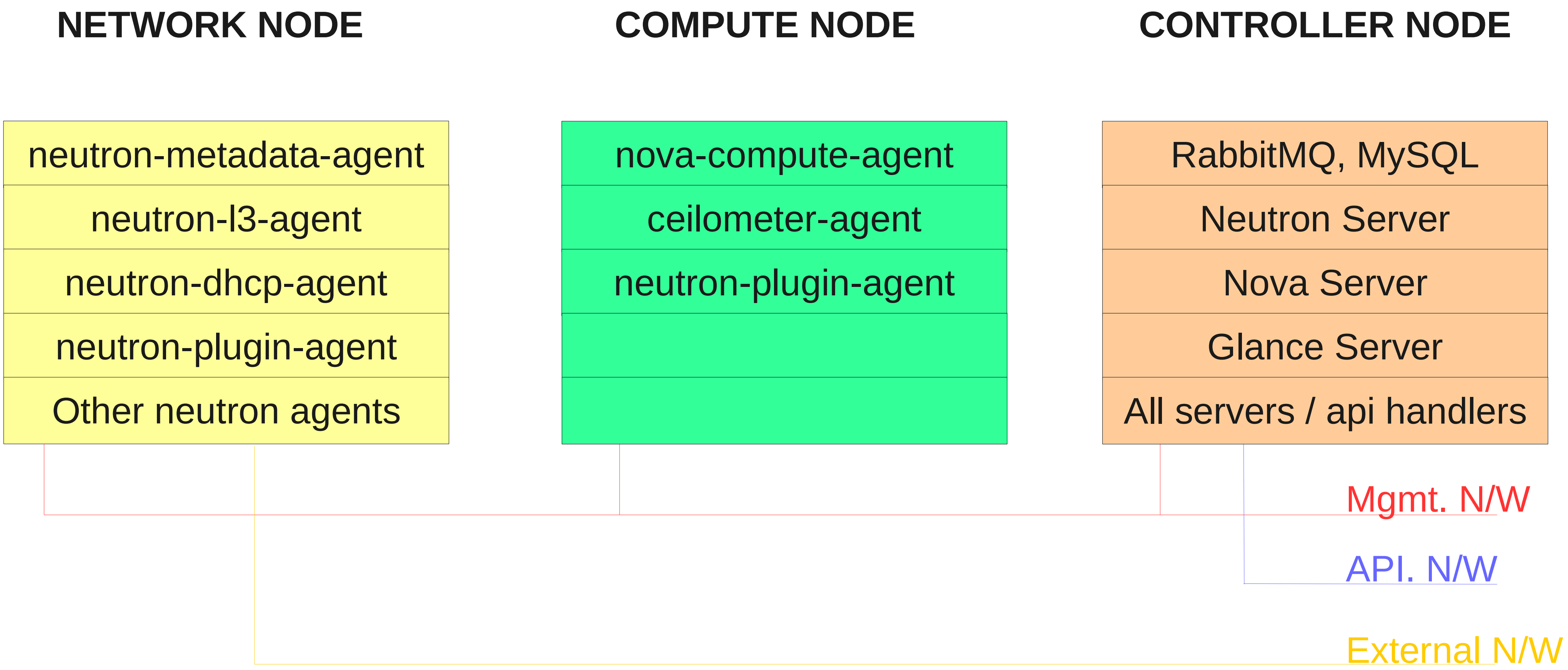
Proof-of-Concept for OpenStack Neutron Agent

-Piyush Raman
Intern,
Cloud Networking, GTS, IBM India Pvt. Ltd

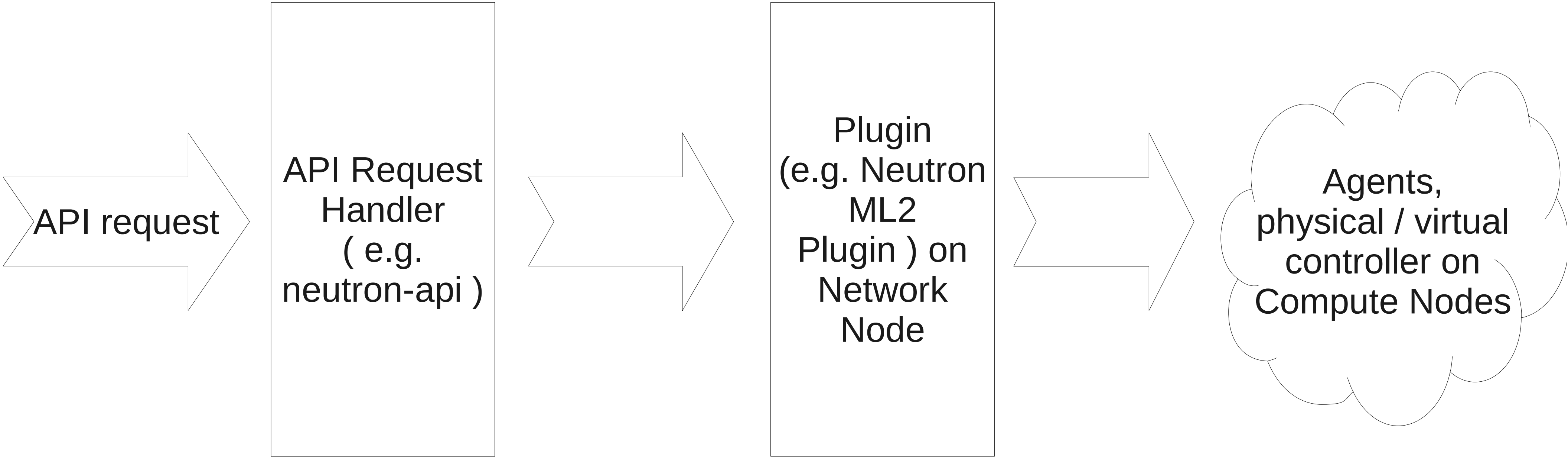


OpenStack Nodes and Data Center Networks



NETWORK NODE	COMPUTE NODE	CONTROLLER NODE
neutron-metadata-agent	nova-compute-agent	RabbitMQ, MySQL
neutron-l3-agent	ceilometer-agent	Neutron Server
neutron-dhcp-agent	neutron-plugin-agent	Nova Server
neutron-plugin-agent		Glance Server
Other neutron agents		All servers / api handlers

Plugin Architecture- OpenStack Neutron



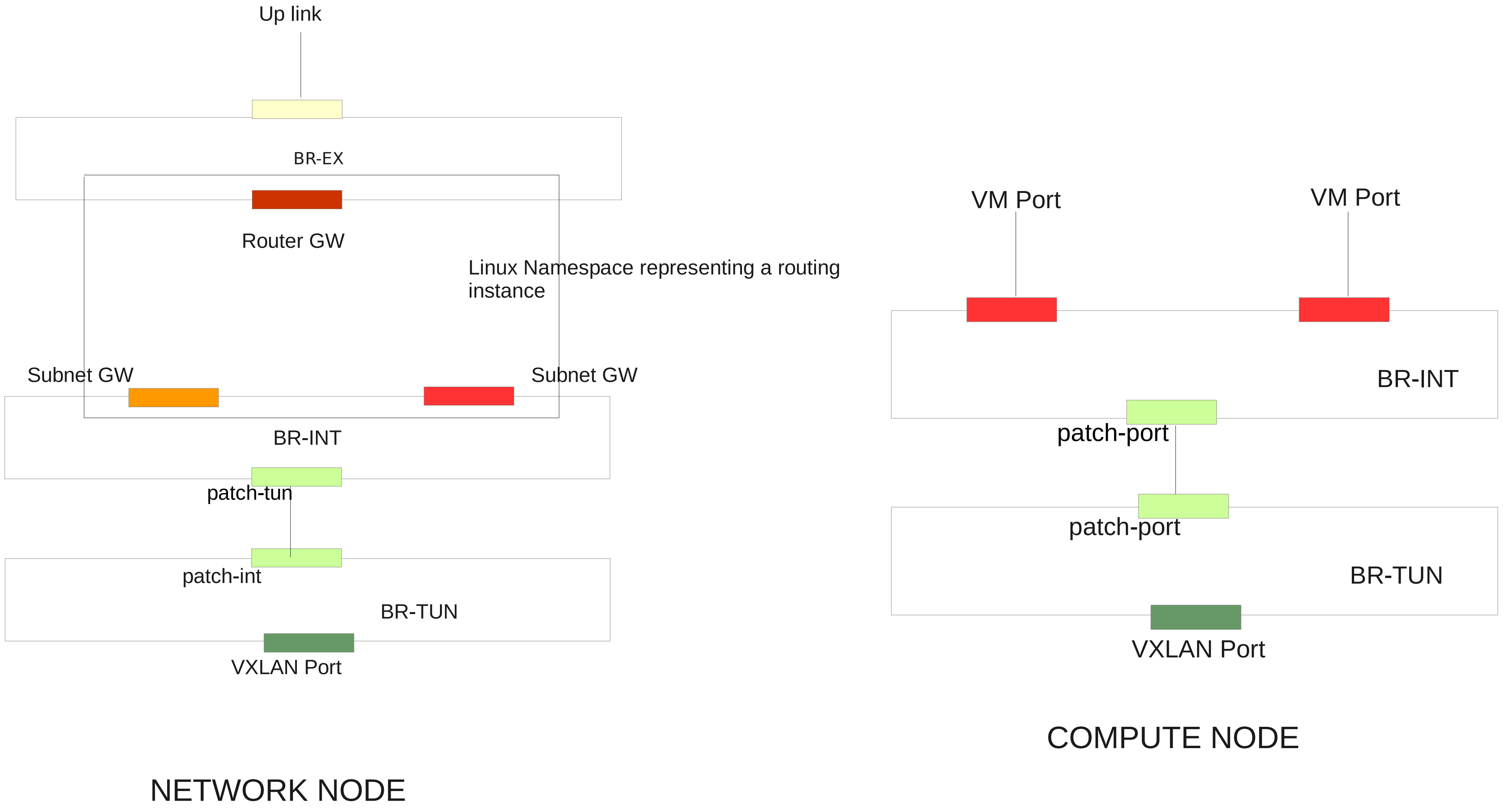
API request

API Request Handler (e.g. neutron-api)

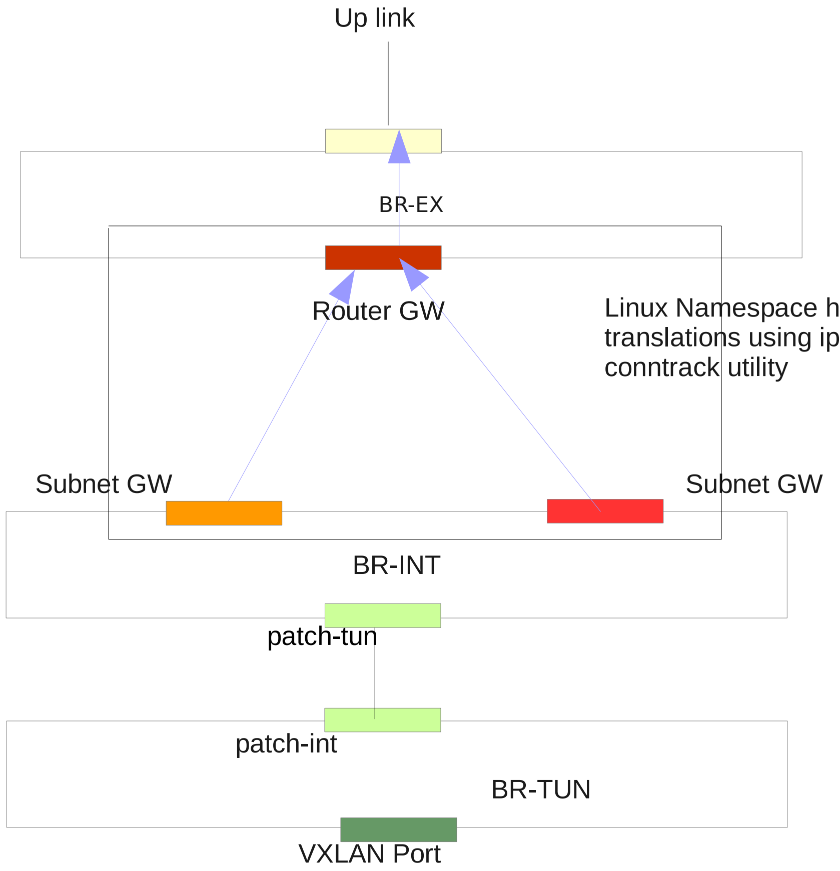
Plugin (e.g. Neutron ML2 Plugin) on Network Node

Agents, physical / virtual controller on Compute Nodes

OpenStack Bridge Setup

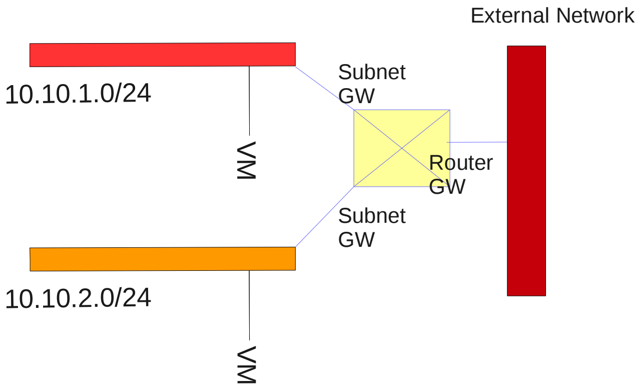


OpenStack Neutron Router



NETWORK NODE

Linux Namespace handles SNAT/DNAT translations using iptables rules / conntrack utility



OpenStack Neutron Virtualization

Proof-Of-Concept Overview

The POC implements a novel OpenStack Neutron Agent which supports L3 routing for overlay network to external (underlay / virtual) network having following functionalities-

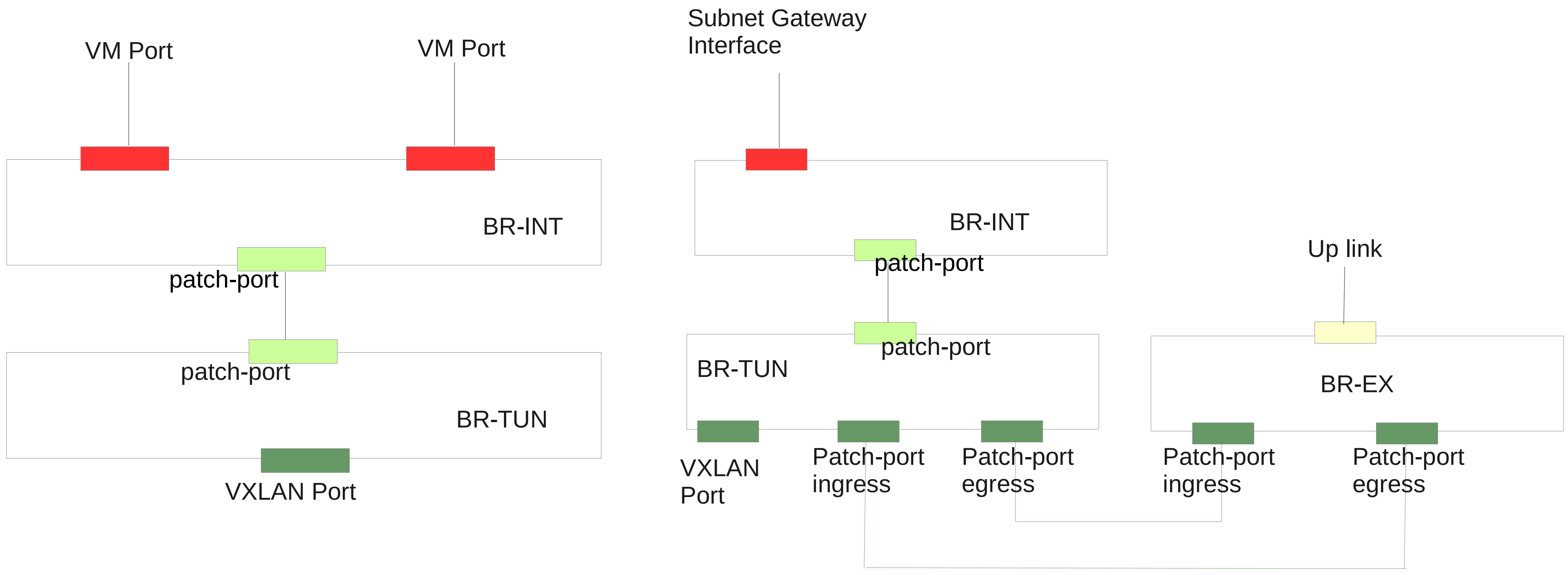
- **Overlay within / across subnet** routing and **NATing** (SNAT/DNAT) via flows
- **Virtualize Neutron Router** using **flows** defined by OpenFlow Protocol
- **Multiple external networks** across / within tenants using **single instance** of the agent
- **De-centralize the overlay across subnet** decision on each compute node
- **ARP Responder / L2 Population** Mechanism on each node to minimize ARP request broadcasts

Openstack Setup Description For POC

Following is the setup description for POC-

- OpenStack Havana
- ML2 Plugin for OpenStack Neutron
- OVS 2.1
- L2 Population / ARP Responder enabled (introduced in OpenStack Icehouse)
- VXLAN Tunneling (For ‘N’ nodes setup, N-1 VXLAN Tunnel Ports on each node)

OpenStack Neutron Bridge Setup For POC

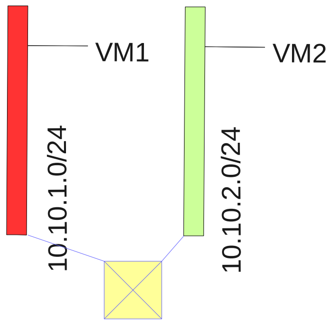


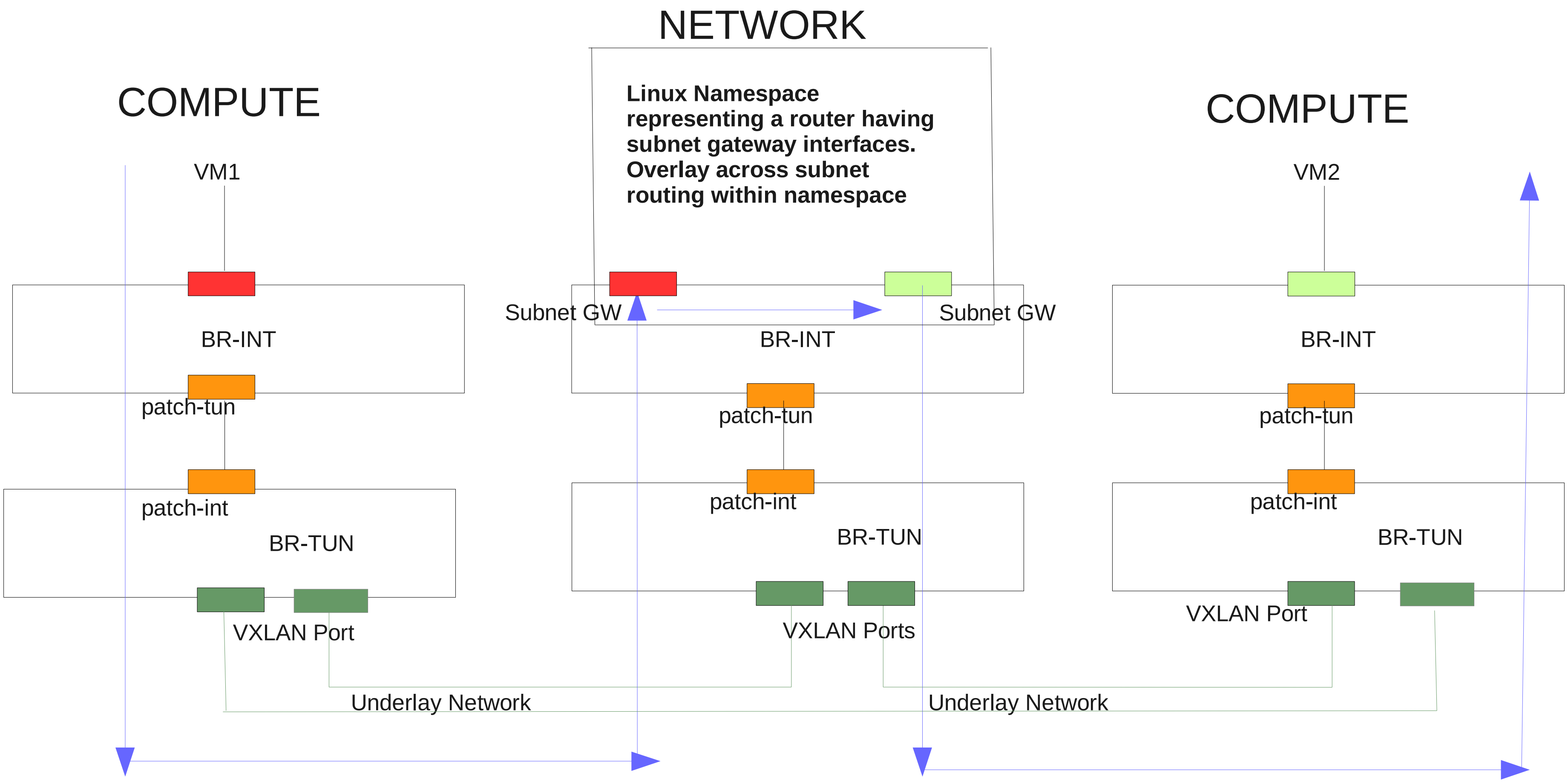
COMPUTE NODE

NETWORK NODE

Advantages

- Decentralize overlay across subnet routing decision on each compute node

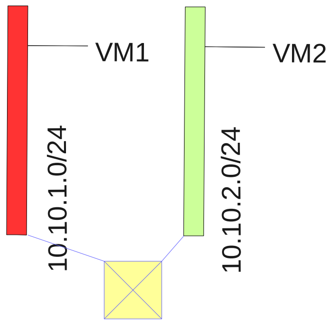


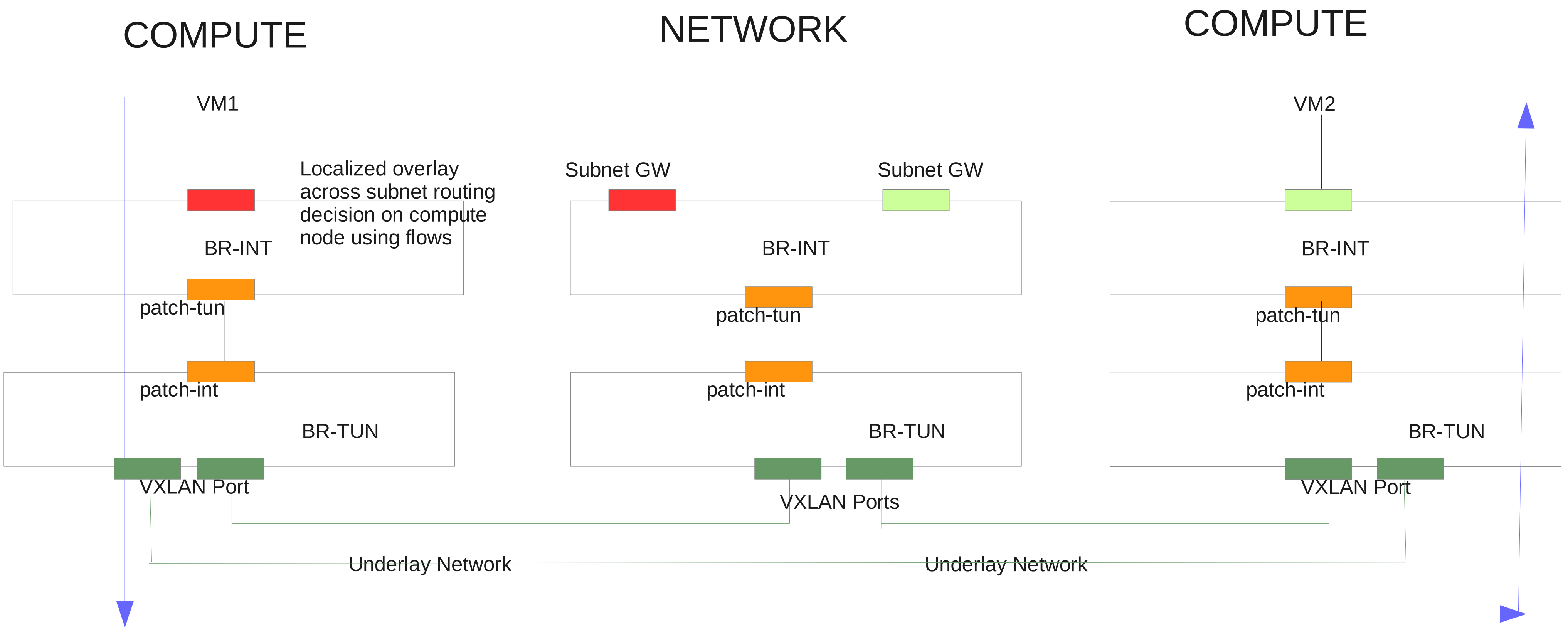

Linux Namespace representing a router having subnet gateway interfaces. Overlay across subnet routing within namespace

ORIGINAL MECHANISM

Advantages

- Decentralize overlay across subnet routing decision on each compute node (functionality introduced in Openstack Juno)



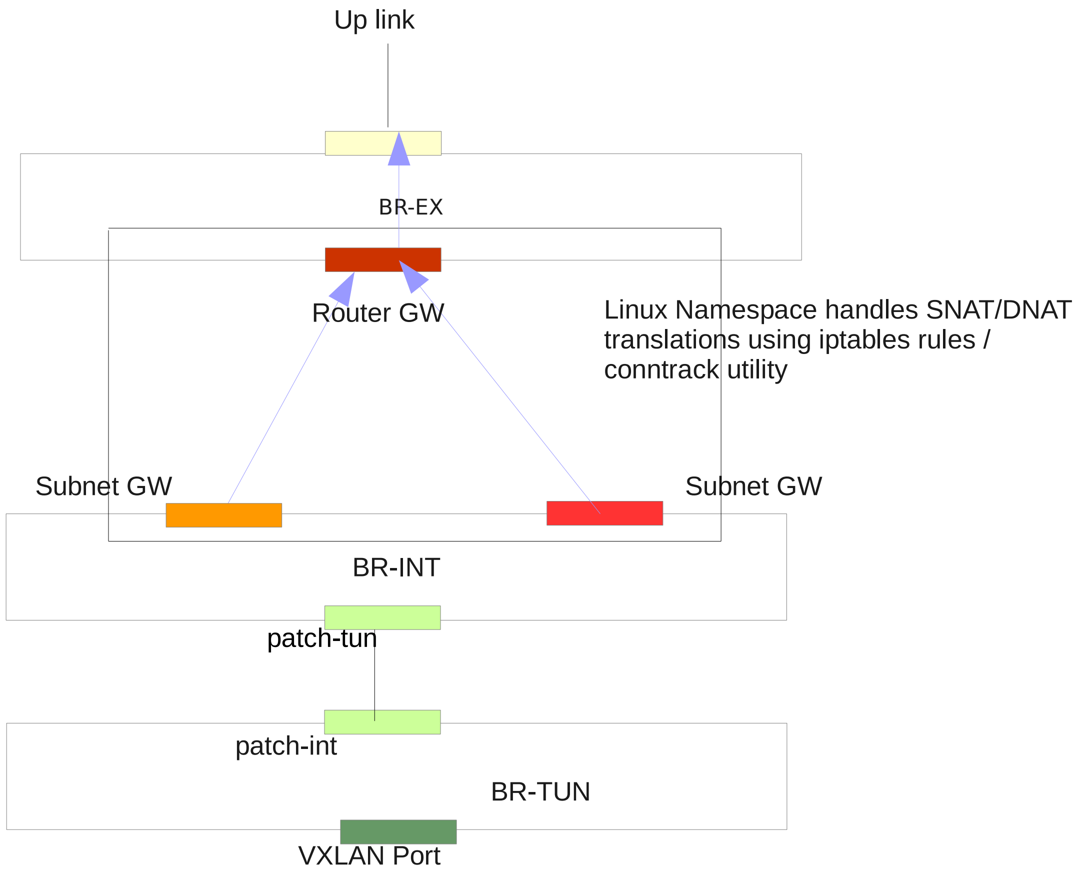


NEW MECHANISM

Advantages

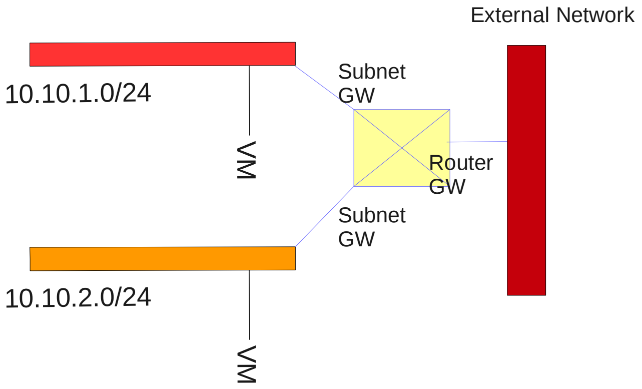
- Virtualize Neutron router using flows instead of Linux Namespace, iptables, Host Stack

- SNAT / DNATing using flows



Linux Namespace handles SNAT/DNAT translations using iptables rules / conntrack utility

NETWORK NODE

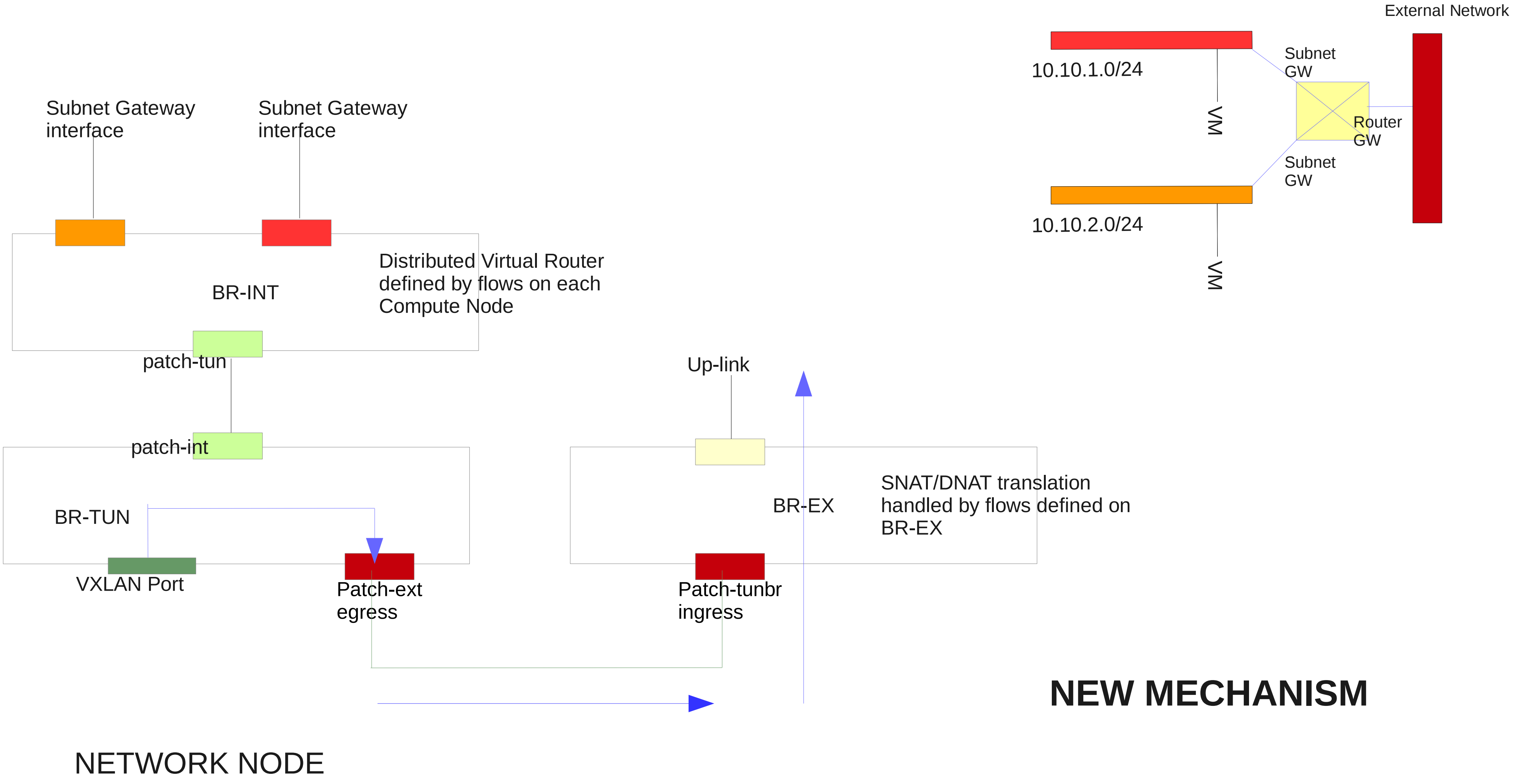


ORIGINAL MECHANISM

Advantages

- Virtualize Neutron router using flows instead of Linux Namespace, iptables, Host Stack

- SNAT / DNATing using flows



External Network

10.10.1.0/24

VM

Subnet GW

Router GW

Subnet GW

10.10.2.0/24

VM

Subnet Gateway interface

Subnet Gateway interface

BR-INT

Distributed Virtual Router defined by flows on each Compute Node

patch-tun

patch-int

BR-TUN

VXLAN Port

Patch-ext egress

Up-link

BR-EX

SNAT/DNAT translation handled by flows defined on BR-EX

Patch-tunbr ingress

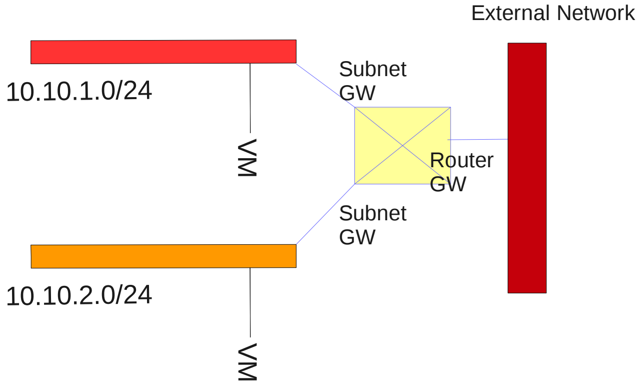
NEW MECHANISM

NETWORK NODE

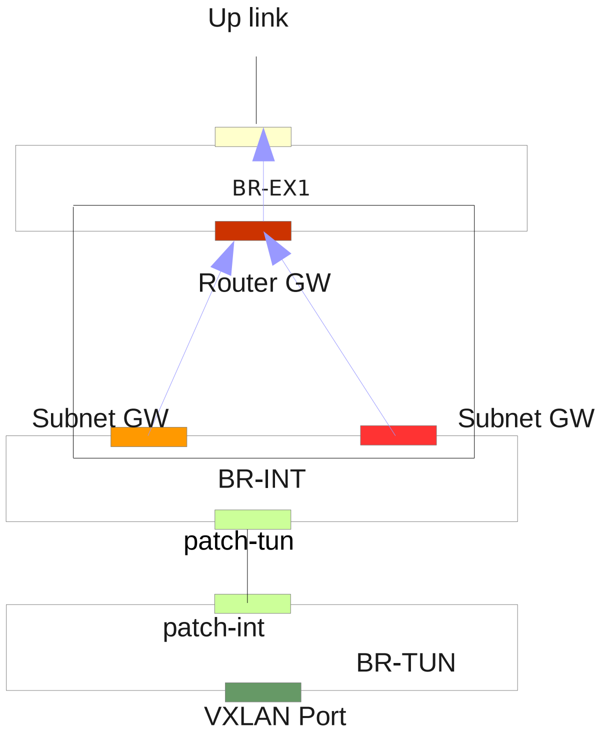
Advantages

- Manage all external networks using single instance of agent

- Multiple external networks can use same bridge

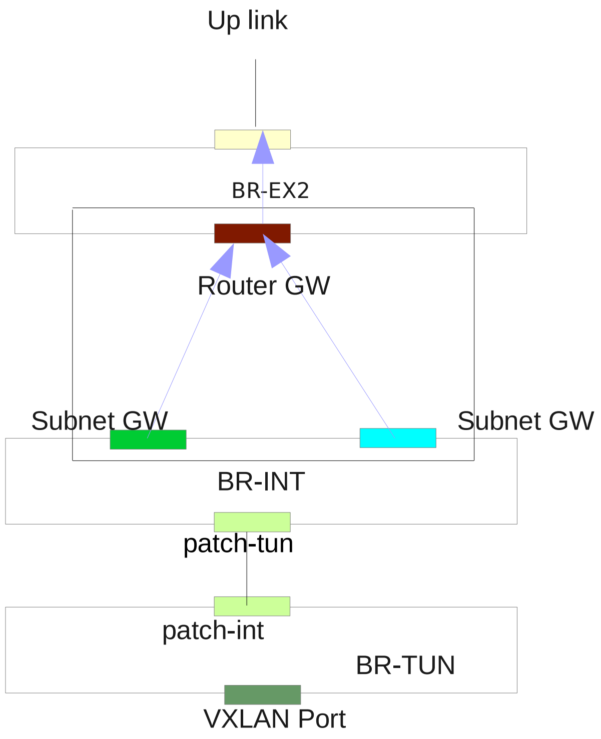


neutron-l3-agent (1st instance)

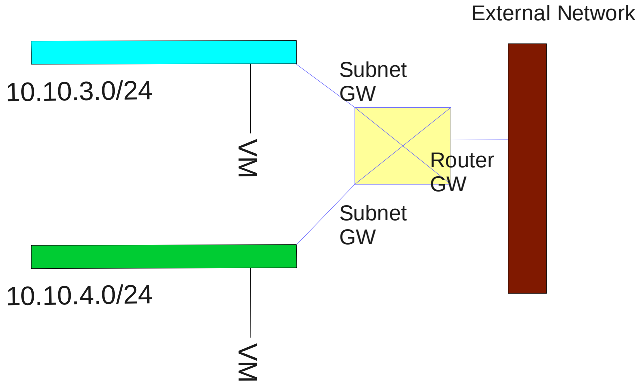


NETWORK NODE

neutron-l3-agent (2nd instance)



NETWORK NODE

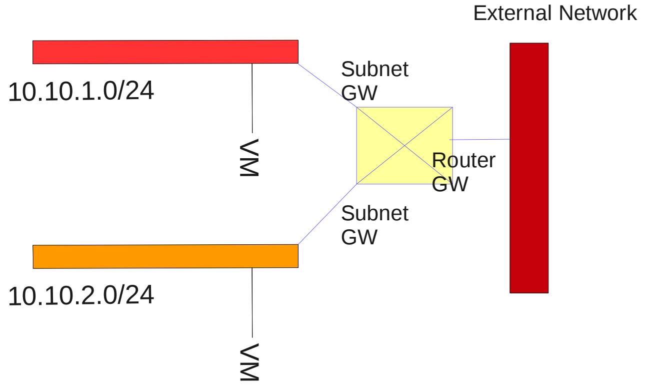


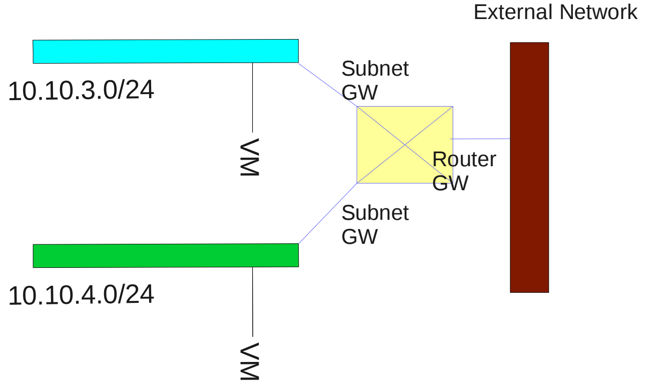
ORIGINAL MECHANISM

Advantages

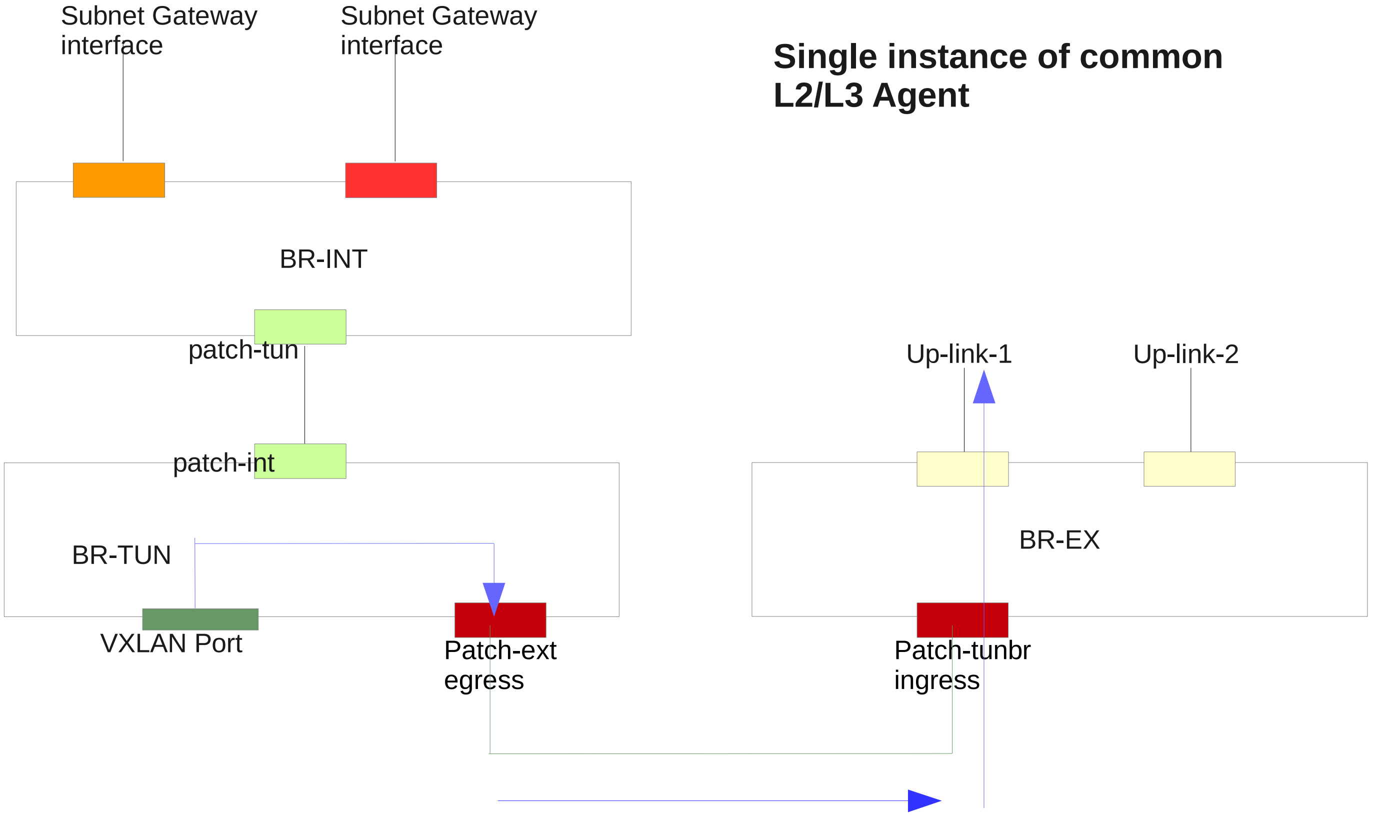
- Manage all external networks using single instance of agent (functionality introduced in Openstack Icehouse)

- Multiple external networks can use same bridge





Single instance of common L2/L3 Agent



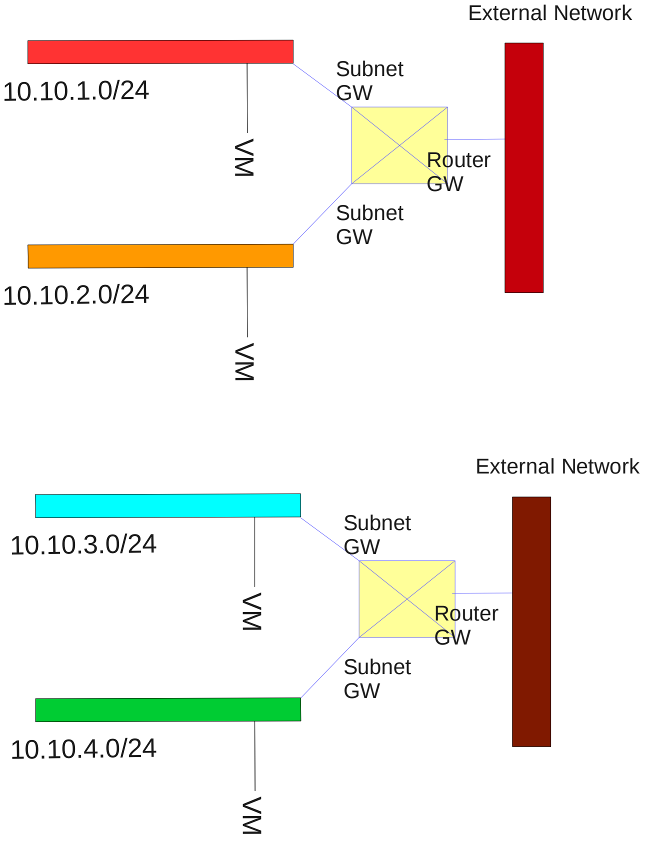
NEW MECHANISM

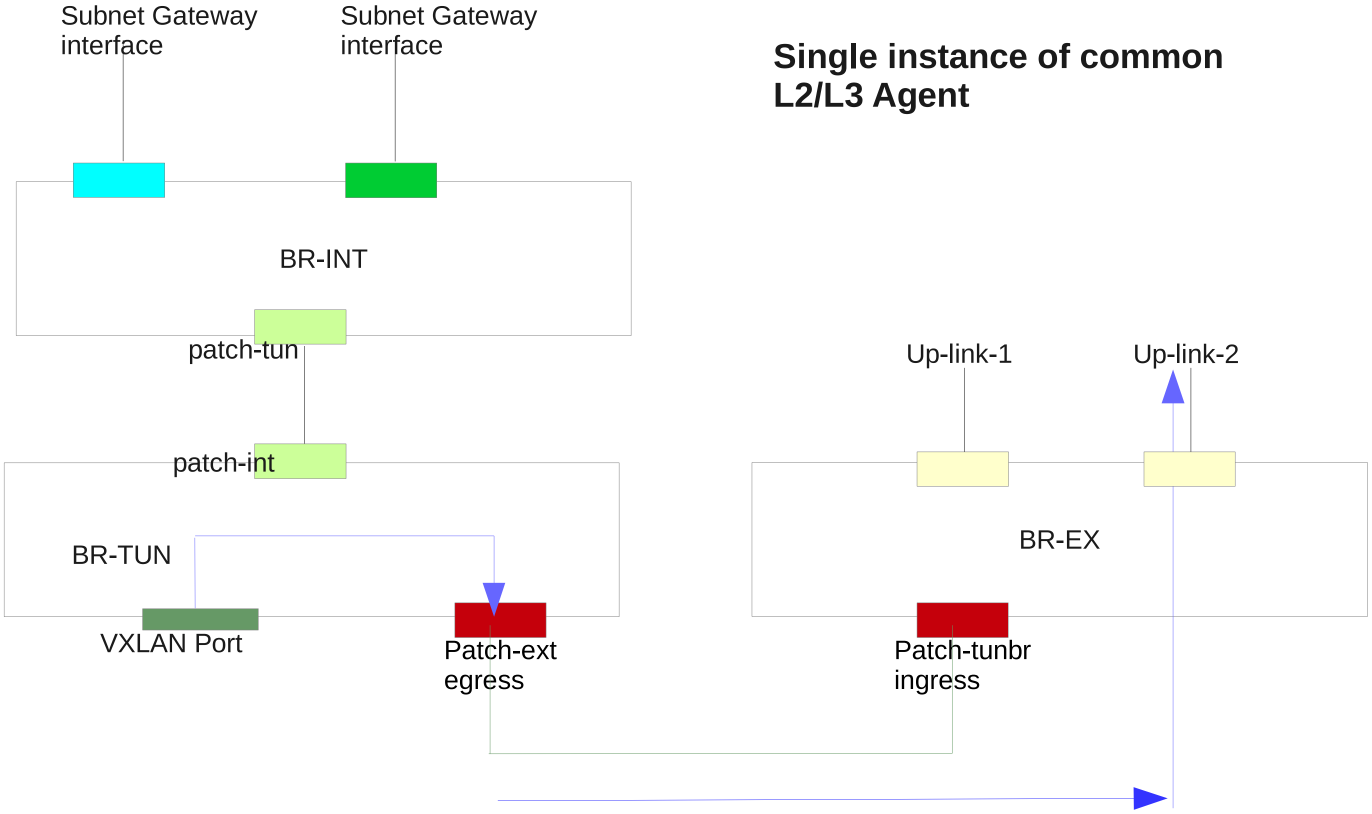
NETWORK NODE

Advantages

- Manage all external networks using single instance of agent (functionality introduced in Openstack Icehouse)

- Multiple external networks can use same bridge





Single instance of common L2/L3 Agent

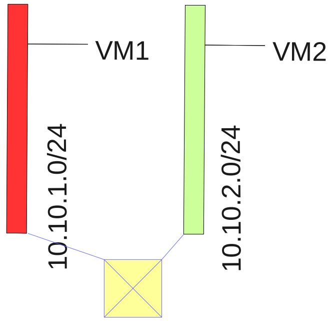
NEW MECHANISM

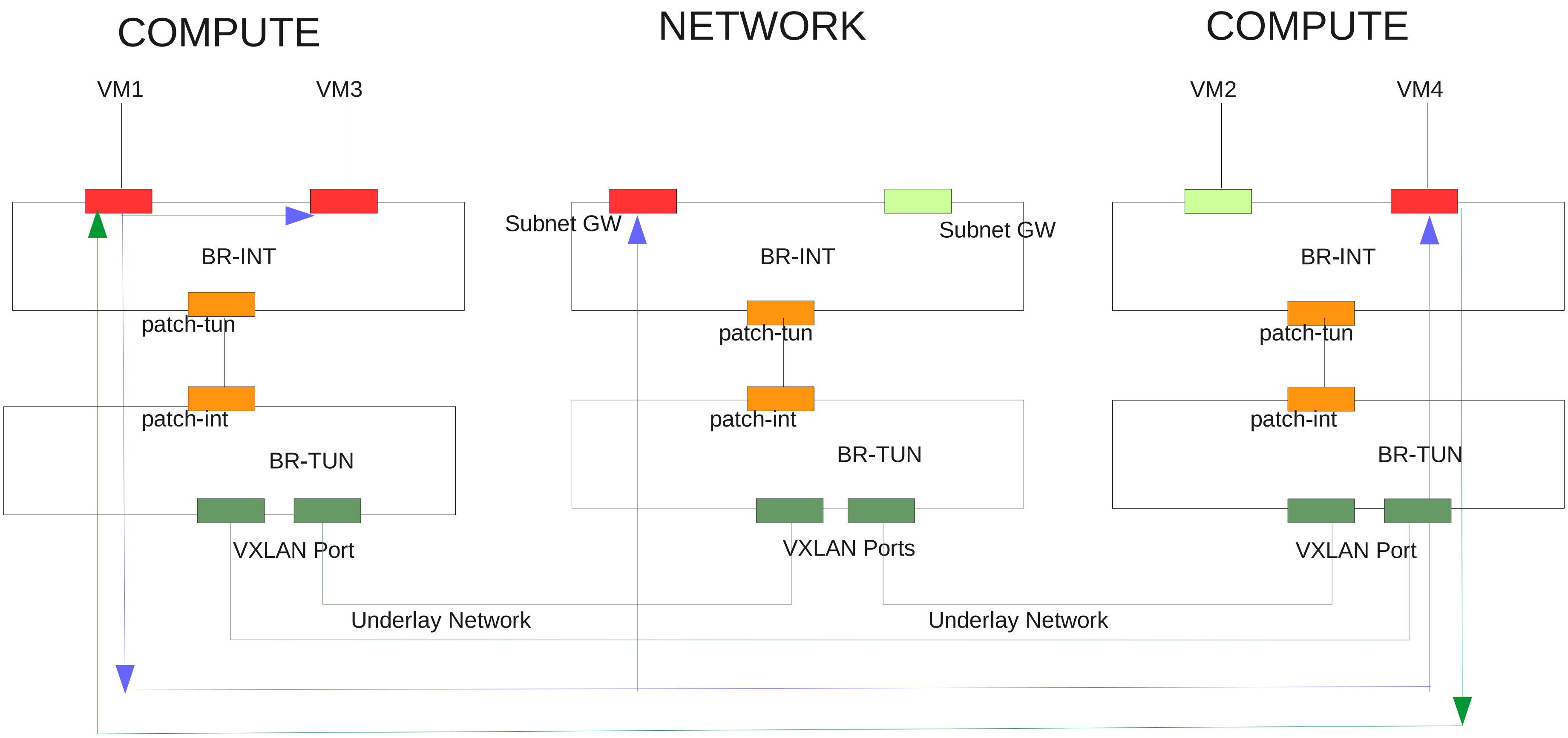
NETWORK NODE

Advantages

- Minimize ARP Request flooding using ARP Responder / L2 Population

(functionality derived from Openstack Havana)



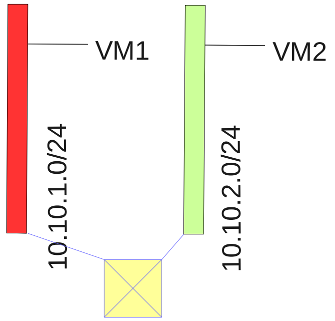


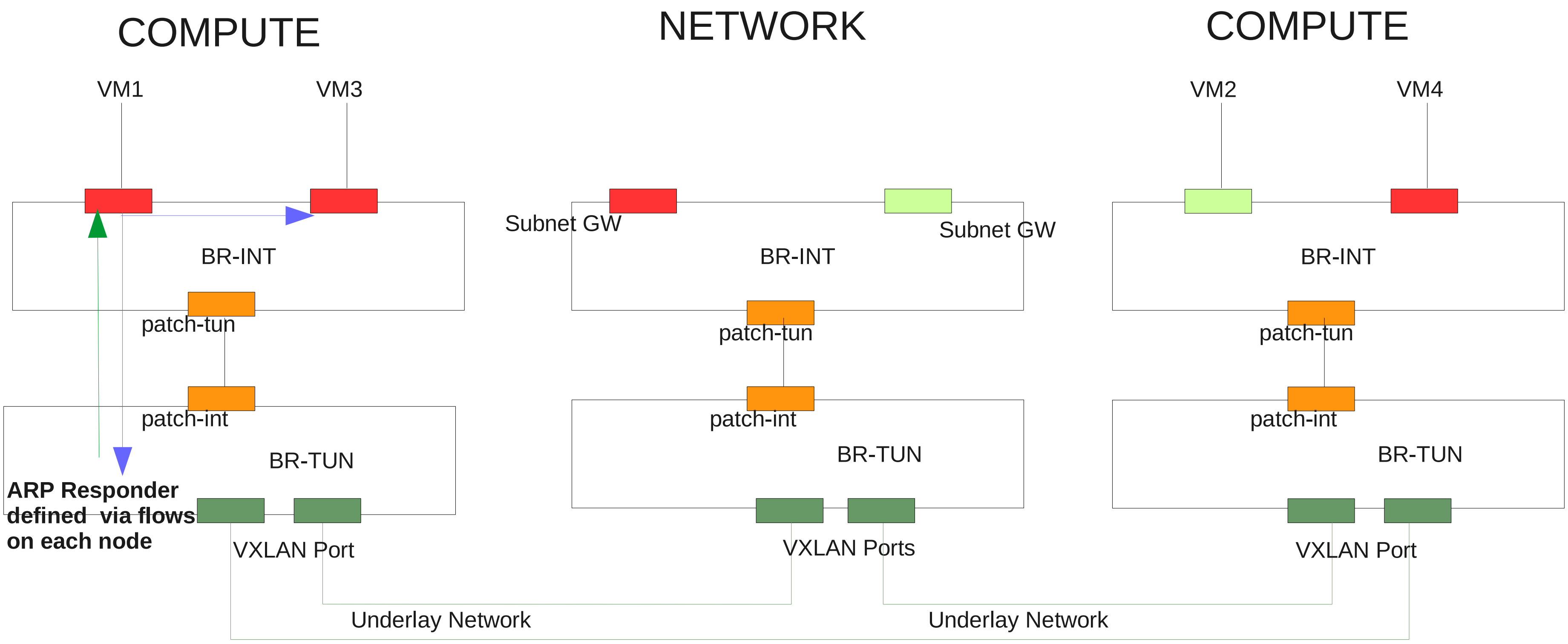
ORIGINAL MECHANISM (FLOODING OF ARP PACKET)

Advantages

- Minimize ARP Request flooding using ARP Responder / L2 Population

(functionality derived from Openstack Havana)





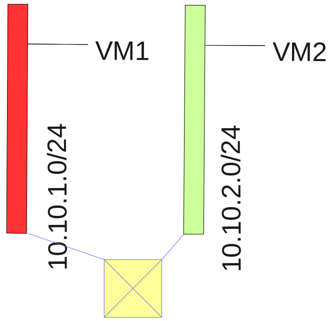
ARP Responder defined via flows on each node

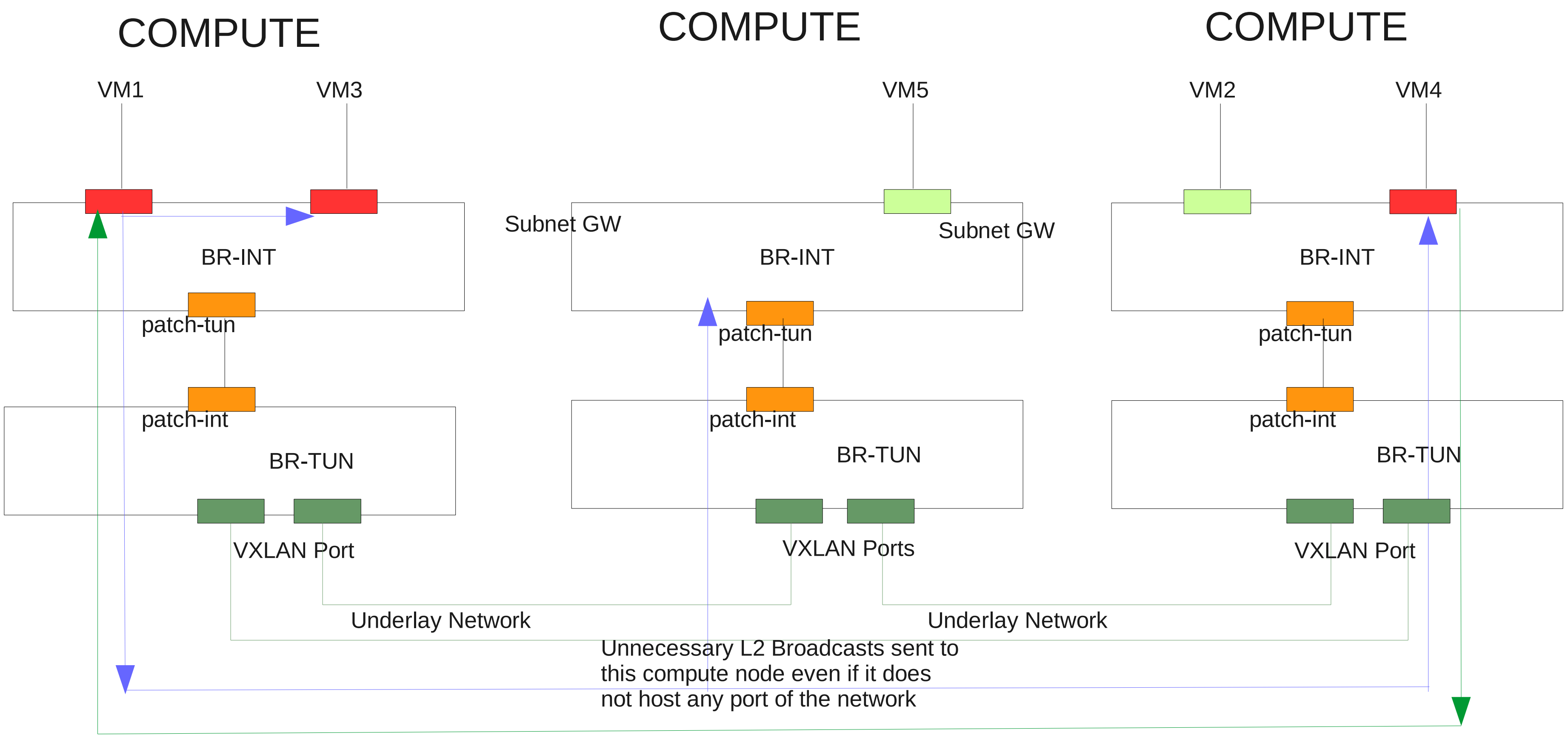
NEW MECHANISM (ARP RESPONDER)

Advantages

- Minimize ARP Request flooding using ARP Responder / L2 Population

(functionality derived from Openstack Havana)





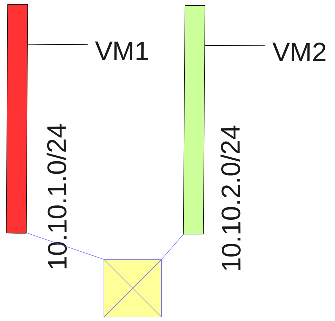
Unnecessary L2 Broadcasts sent to this compute node even if it does not host any port of the network

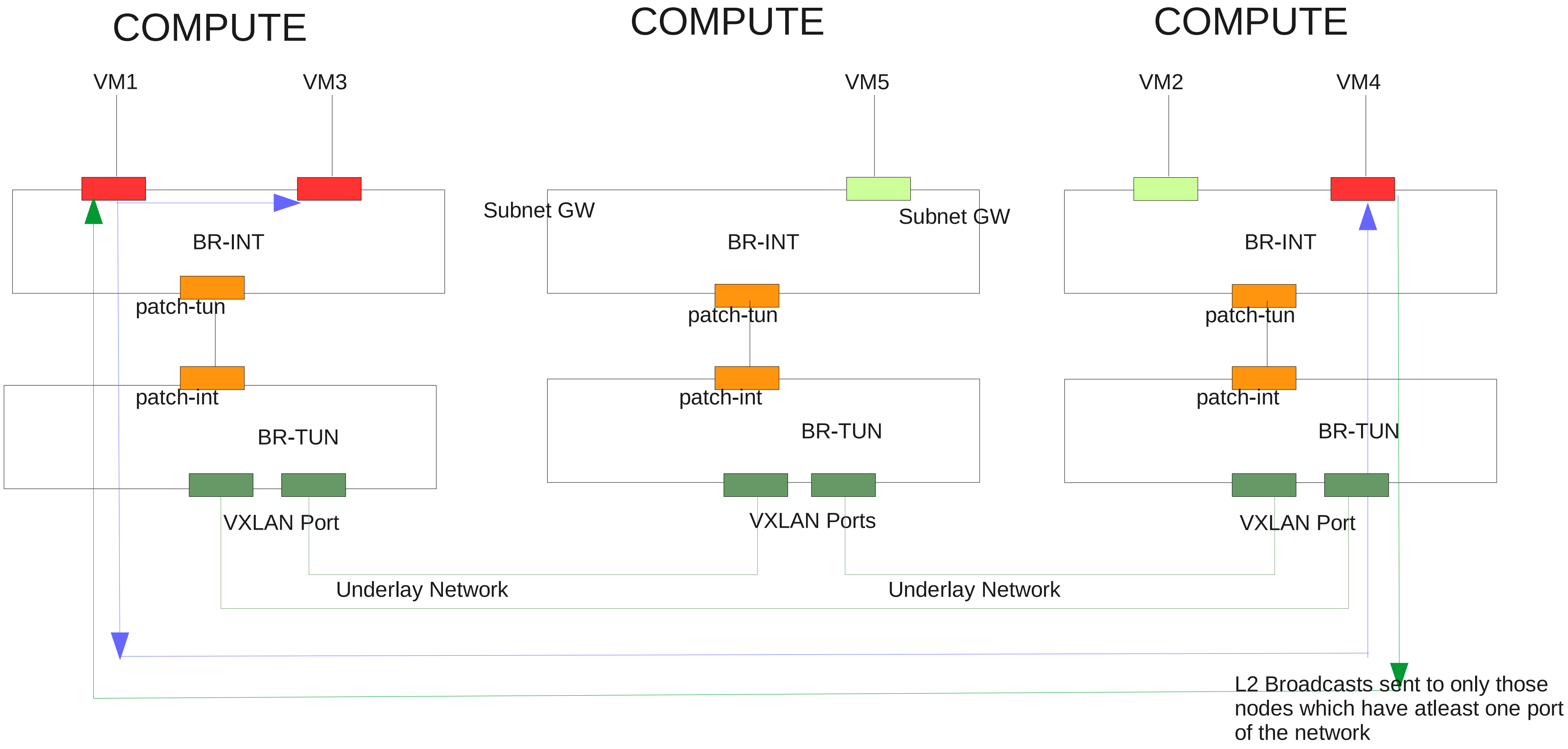
ORIGINAL MECHANISM

Advantages

- Minimize ARP Request flooding using ARP Responder / L2 Population

(functionality derived from Openstack Havana)





NEW MECHANISM (With L2 Population)

Performance Analysis

Hardware Specification-

- 4 nodes setup (1 controller / network and 3 compute nodes)

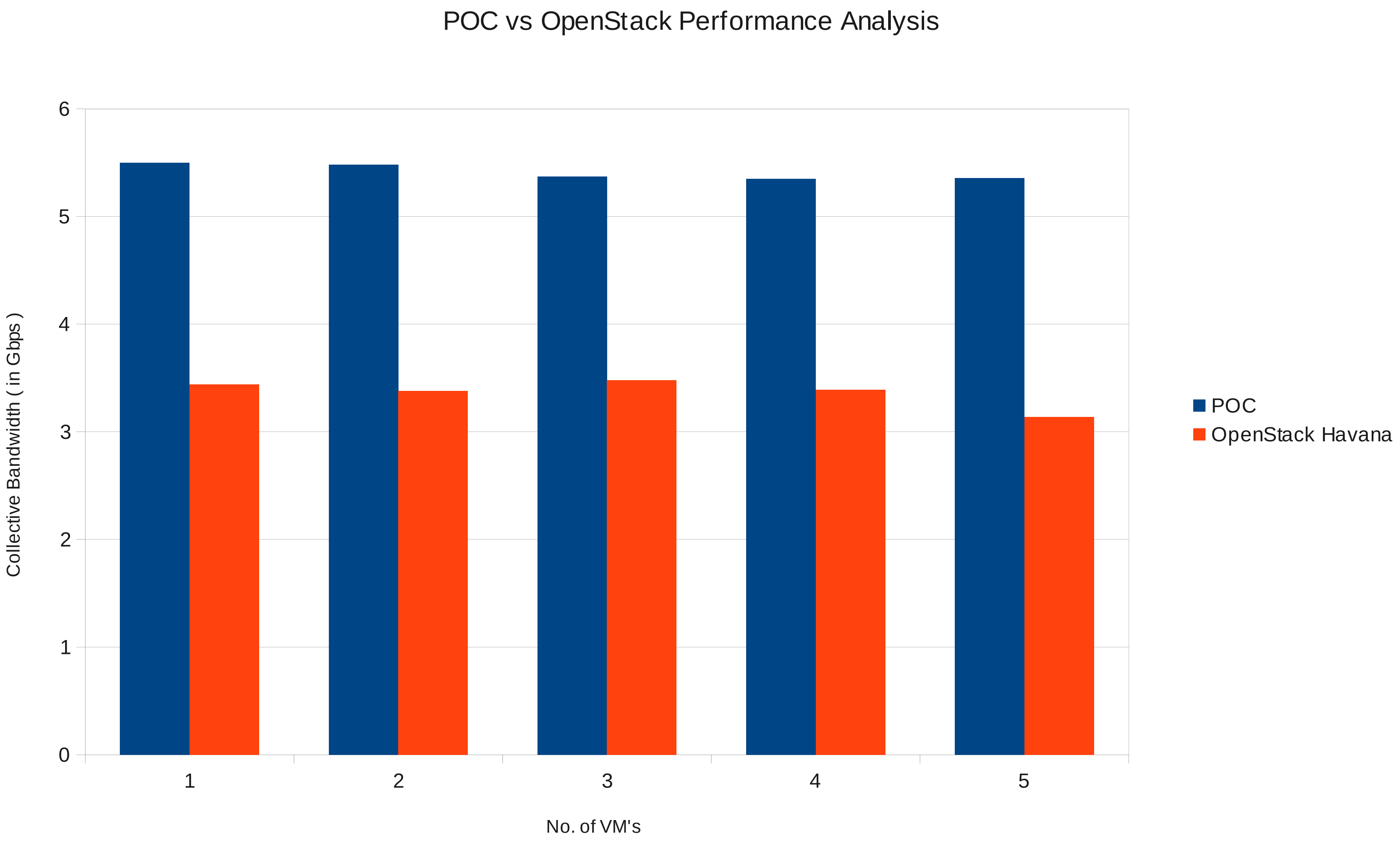
- Each node connected to Tunnel Network via 10 Gbps link

- Network Node connected to External Network via 10 Gbps link

- VM configuration – 2 VCPUs, 2048 MB RAM, CentOS

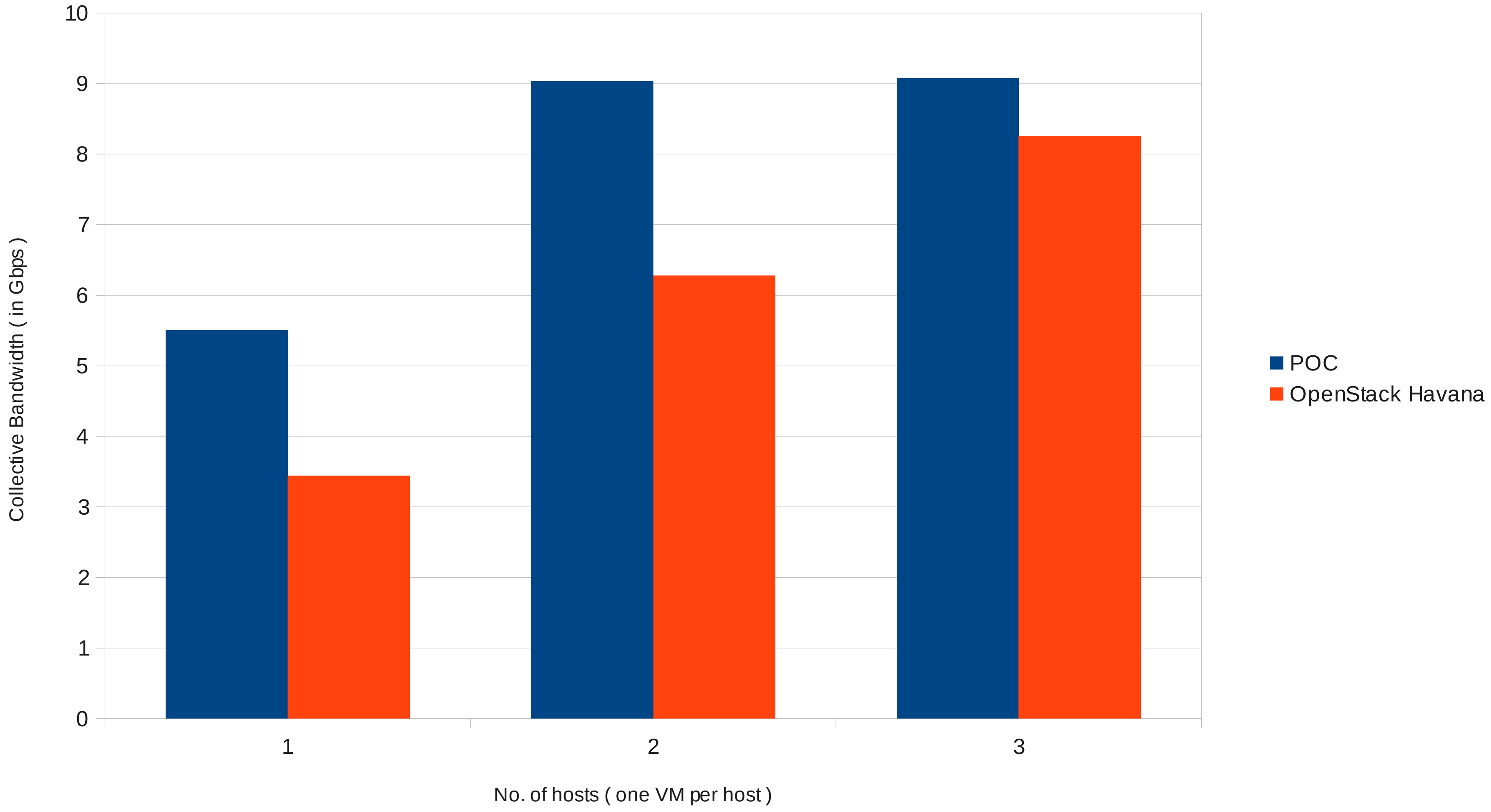
Performance Analysis

Use Case - SNAT for single host



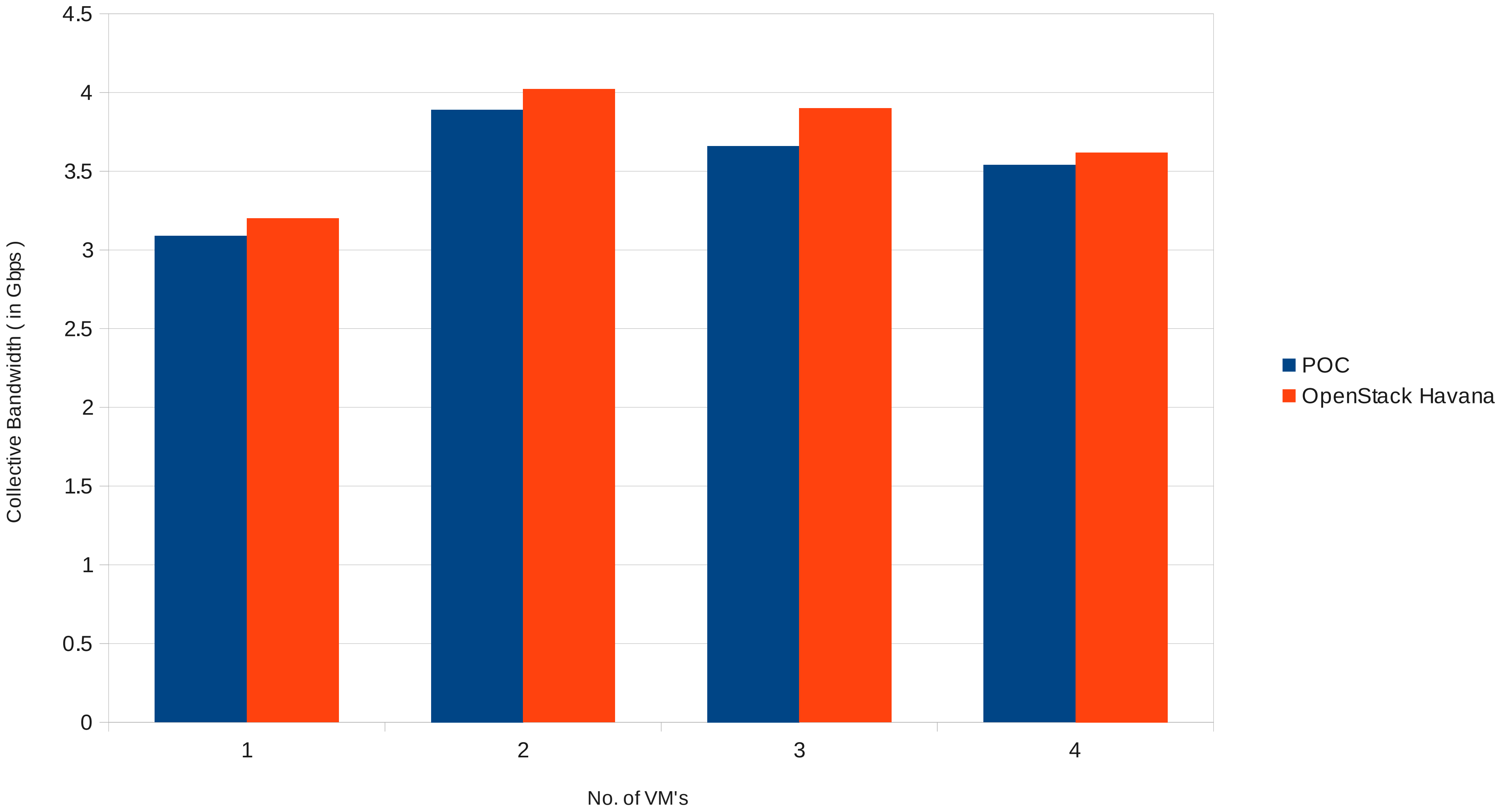
Performance Analysis

Use Case - SNAT for multiple hosts



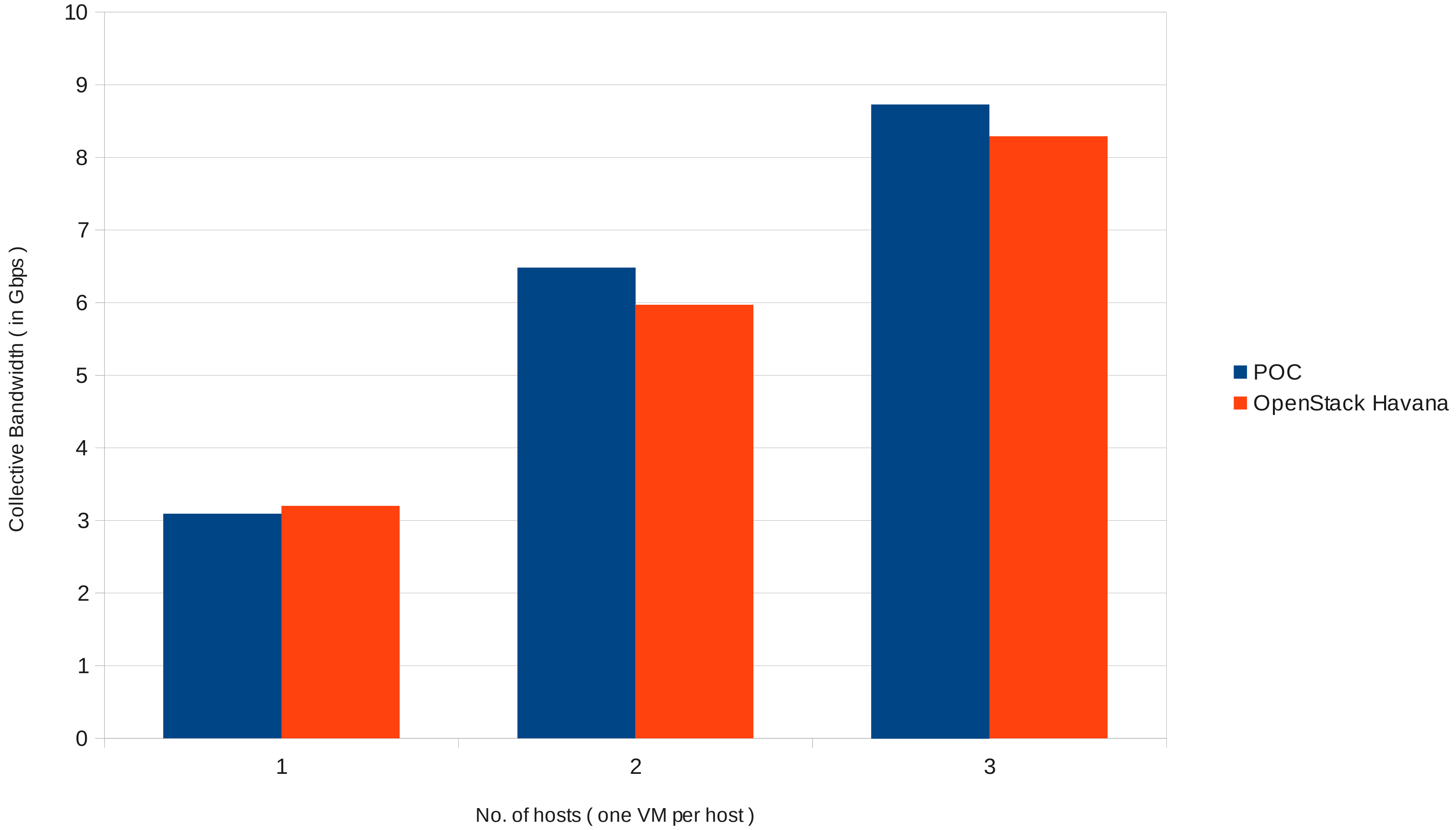
Performance Analysis

Use Case - DNAT for single host



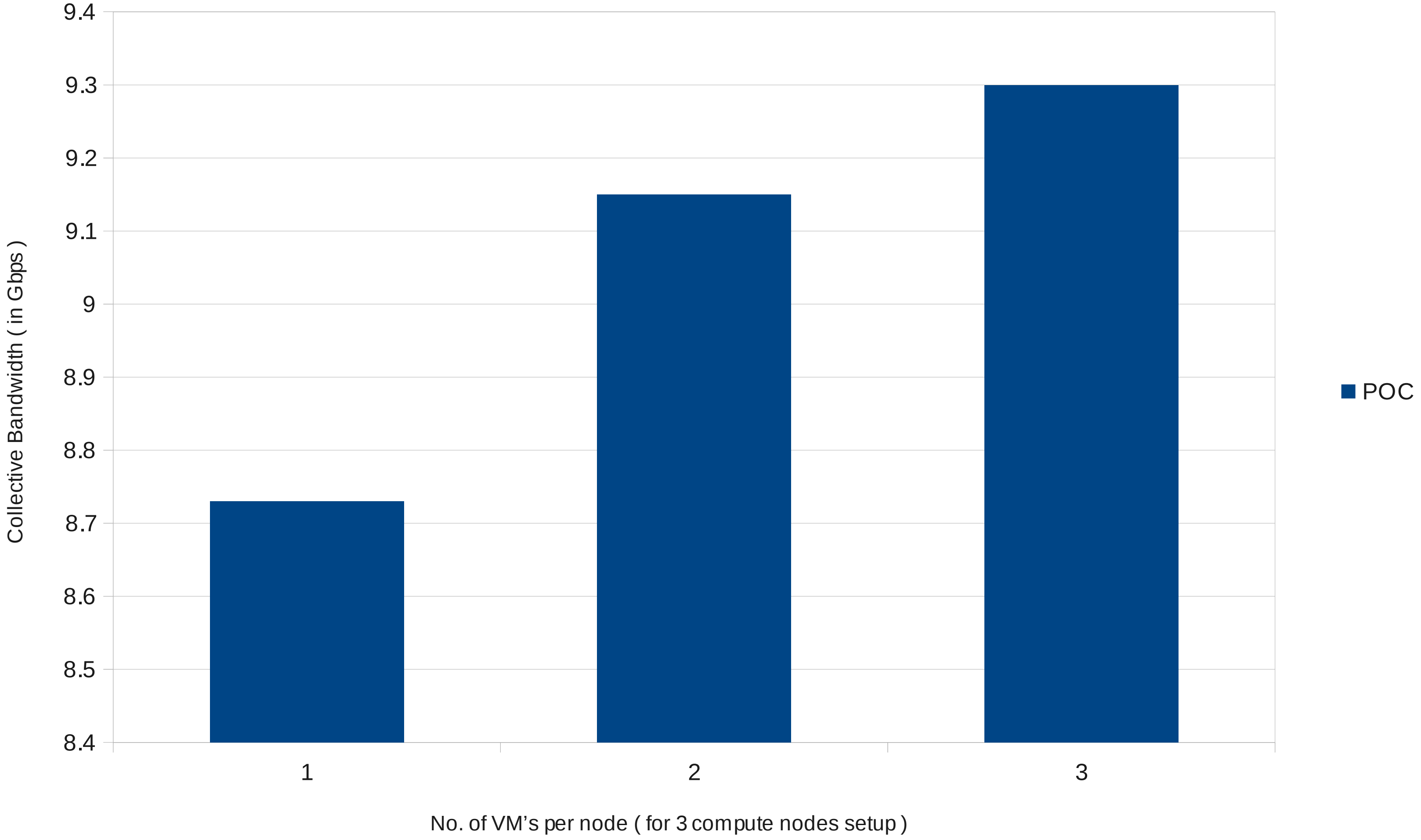
Performance Analysis

Use Case - DNAT for multiple host



Performance Analysis

Use Case - DNAT for multiple host (POC only)



Future Work

- ICMP support for router interfaces by generating ICMP replies using the new Agent
- SNAT support for ICMP packets. Implement OpenFlow protocol to allow access to ICMP identifier field and use port mapping for ICMP and TCP/UDP sessions
- Implement OpenStack Nova Security Groups using flows on OVS integration bridge, thus, mapping VM’s directly on the OVS integration bridge
- Updating ARP cache for external network at OVS external bridge programatically
- Updating ARP Responder / Cache on OVS tunnel / integration bridge programatically
- Deletion of flows in a programmatic manner

Credits

- Rachappa B Goni
  Advisory Engineer
  Cloud Networking, GTS, IBM India Pvt. LTD
  grachapp@in.ibm.com

- Prashanth K Nageshappa
  Senior Engineer, Master Inventor
  Cloud Networking, GTS, IBM India Pvt. LTD
  nprashan@in.ibm.com

- Gowtham Narasimhaiah
  Development Manager
  Cloud Networking, GTS, IBM India Pvt. LTD
  ngowtham@in.ibm.com

- Vaidyanathan Gopalakrishnan
  Operations Manager
  Cloud Networking, GTS, IBM India Pvt. LTD
  vaigopal@in.ibm.com

BACKUP SLIDES

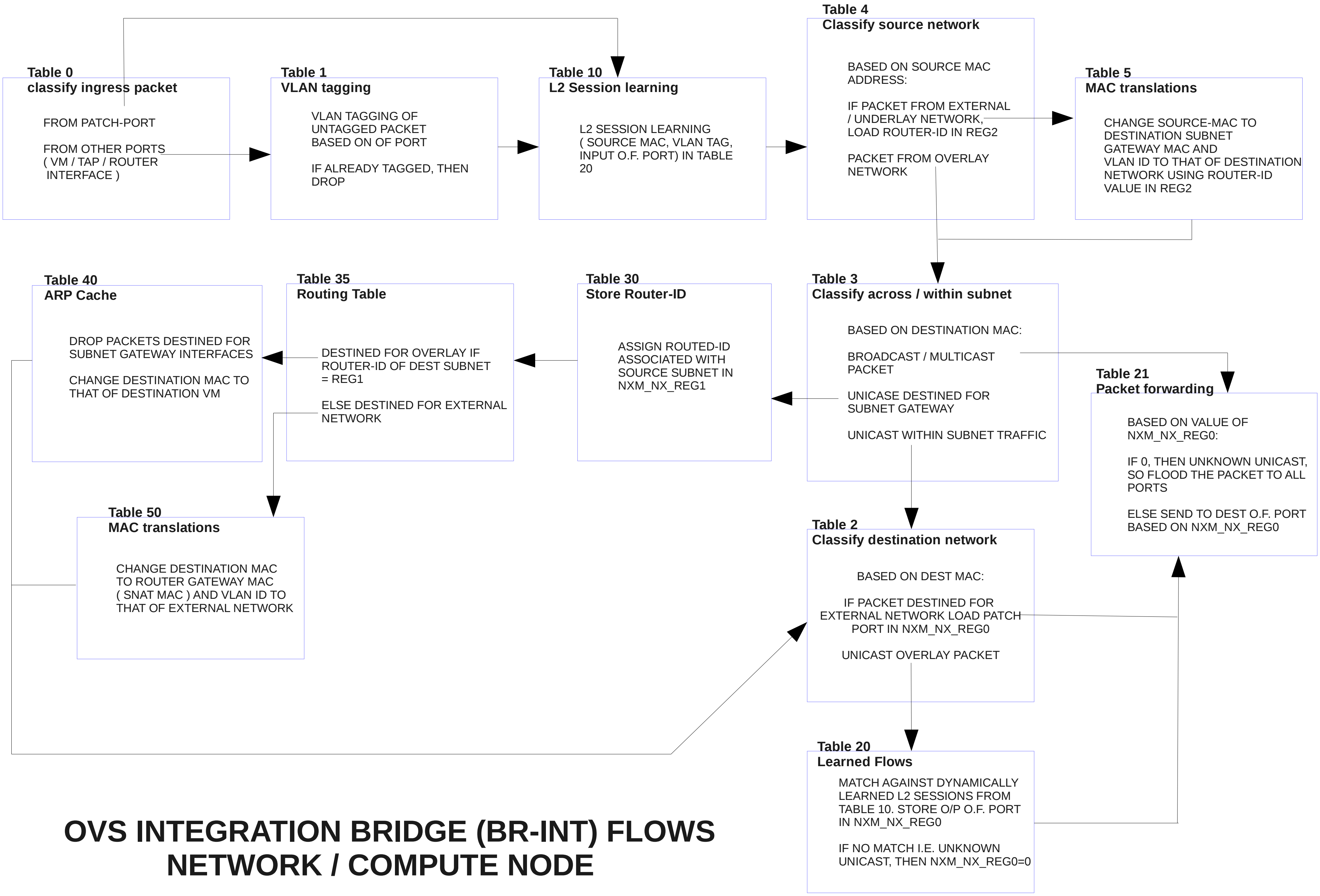


Table 0
classify ingress packet

FROM PATCH-PORT

FROM OTHER PORTS
(VM / TAP / ROUTER
INTERFACE)

Table 1
VLAN tagging

VLAN TAGGING OF
UNTAGGED PACKET
BASED ON OF PORT

IF ALREADY TAGGED, THEN
DROP

Table 10
L2 Session learning

L2 SESSION LEARNING
(SOURCE MAC, VLAN TAG,
INPUT O.F. PORT) IN TABLE
20

Table 4
Classify source network

BASED ON SOURCE MAC
ADDRESS:

IF PACKET FROM EXTERNAL
/ UNDERLAY NETWORK,
LOAD ROUTER-ID IN REG2

PACKET FROM OVERLAY
NETWORK

Table 5
MAC translations

CHANGE SOURCE-MAC TO
DESTINATION SUBNET
GATEWAY MAC AND
VLAN ID TO THAT OF DESTINATION
NETWORK USING ROUTER-ID
VALUE IN REG2

Table 40
ARP Cache

DROP PACKETS DESTINED FOR
SUBNET GATEWAY INTERFACES

CHANGE DESTINATION MAC TO
THAT OF DESTINATION VM

Table 35
Routing Table

DESTINED FOR OVERLAY IF
ROUTER-ID OF DEST SUBNET
= REG1

ELSE DESTINED FOR EXTERNAL
NETWORK

Table 30
Store Router-ID

ASSIGN ROUTED-ID
ASSOCIATED WITH
SOURCE SUBNET IN
NXM_NX_REG1

Table 3
Classify across / within subnet

BASED ON DESTINATION MAC:

BROADCAST / MULTICAST
PACKET

UNICASE DESTINED FOR
SUBNET GATEWAY

UNICAST WITHIN SUBNET TRAFFIC

Table 21
Packet forwarding

BASED ON VALUE OF
NXM_NX_REG0:

IF 0, THEN UNKNOWN UNICAST,
SO FLOOD THE PACKET TO ALL
PORTS

ELSE SEND TO DEST O.F. PORT
BASED ON NXM_NX_REG0

Table 50
MAC translations

CHANGE DESTINATION MAC
TO ROUTER GATEWAY MAC
(SNAT MAC) AND VLAN ID TO
THAT OF EXTERNAL NETWORK

Table 2
Classify destination network

BASED ON DEST MAC:

IF PACKET DESTINED FOR
EXTERNAL NETWORK LOAD PATCH
PORT IN NXM_NX_REG0

UNICAST OVERLAY PACKET

Table 20
Learned Flows

MATCH AGAINST DYNAMICALLY
LEARNED L2 SESSIONS FROM
TABLE 10. STORE O/P O.F. PORT
IN NXM_NX_REG0

IF NO MATCH I.E. UNKNOWN
UNICAST, THEN NXM_NX_REG0=0

OVS INTEGRATION BRIDGE (BR-INT) FLOWS
NETWORK / COMPUTE NODE

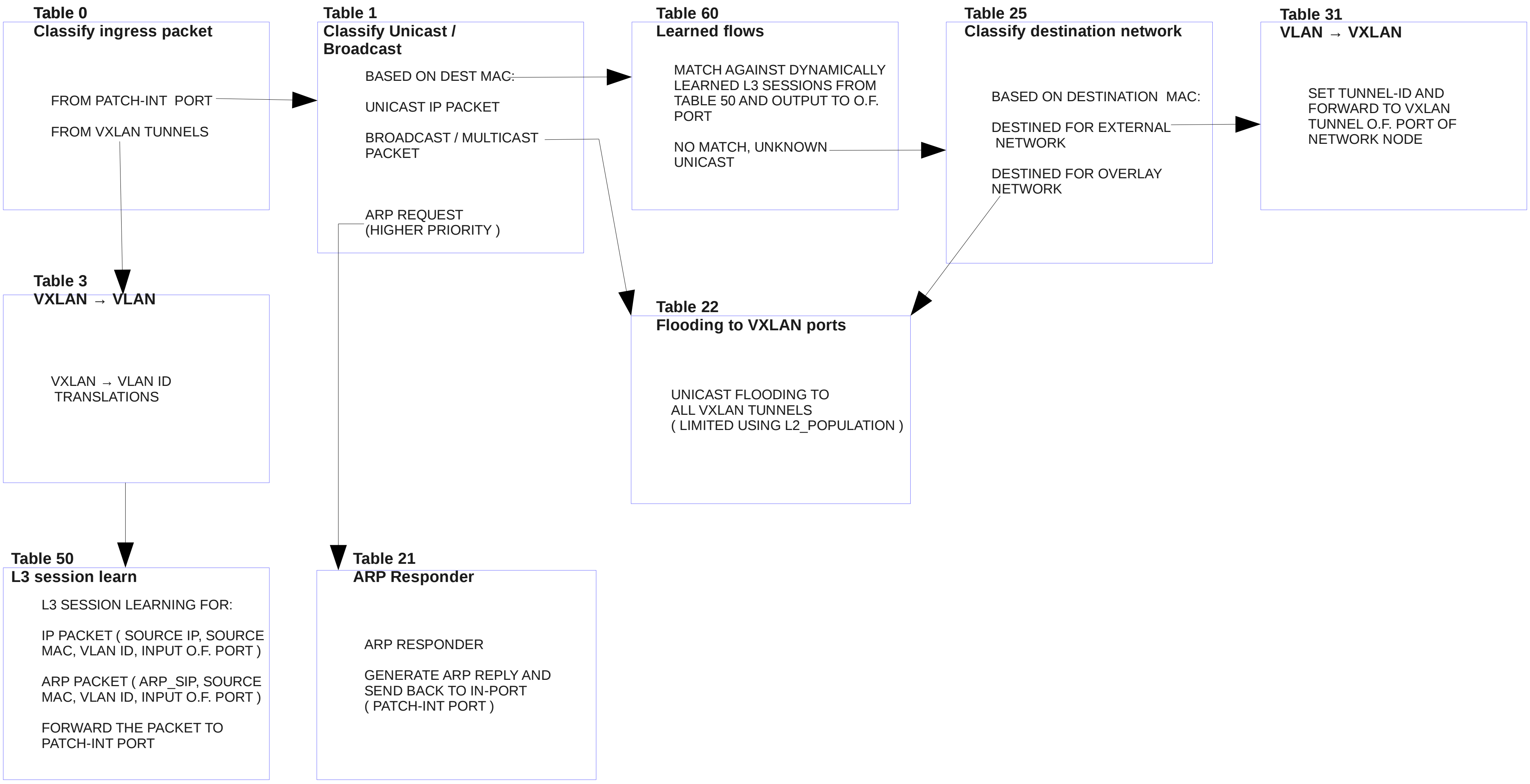


Table 0
Classify ingress packet

FROM PATCH-INT PORT

FROM VXLAN TUNNELS

Table 1
Classify Unicast / Broadcast

BASED ON DEST MAC:

UNICAST IP PACKET

BROADCAST / MULTICAST PACKET

ARP REQUEST (HIGHER PRIORITY)

Table 60
Learned flows

MATCH AGAINST DYNAMICALLY LEARNED L3 SESSIONS FROM TABLE 50 AND OUTPUT TO O.F. PORT

NO MATCH, UNKNOWN UNICAST

Table 25
Classify destination network

BASED ON DESTINATION MAC:

DESTINED FOR EXTERNAL NETWORK

DESTINED FOR OVERLAY NETWORK

Table 31
VLAN → VXLAN

SET TUNNEL-ID AND FORWARD TO VXLAN TUNNEL O.F. PORT OF NETWORK NODE

Table 3
VXLAN → VLAN

VXLAN → VLAN ID TRANSLATIONS

Table 22
Flooding to VXLAN ports

UNICAST FLOODING TO ALL VXLAN TUNNELS (LIMITED USING L2_POPULATION)

Table 50
L3 session learn

L3 SESSION LEARNING FOR:

IP PACKET (SOURCE IP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

ARP PACKET (ARP_SIP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

FORWARD THE PACKET TO PATCH-INT PORT

Table 21
ARP Responder

ARP RESPONDER

GENERATE ARP REPLY AND SEND BACK TO IN-PORT (PATCH-INT PORT)

OVS TUNNEL (BR-TUN) FLOWS
COMPUTE NODE

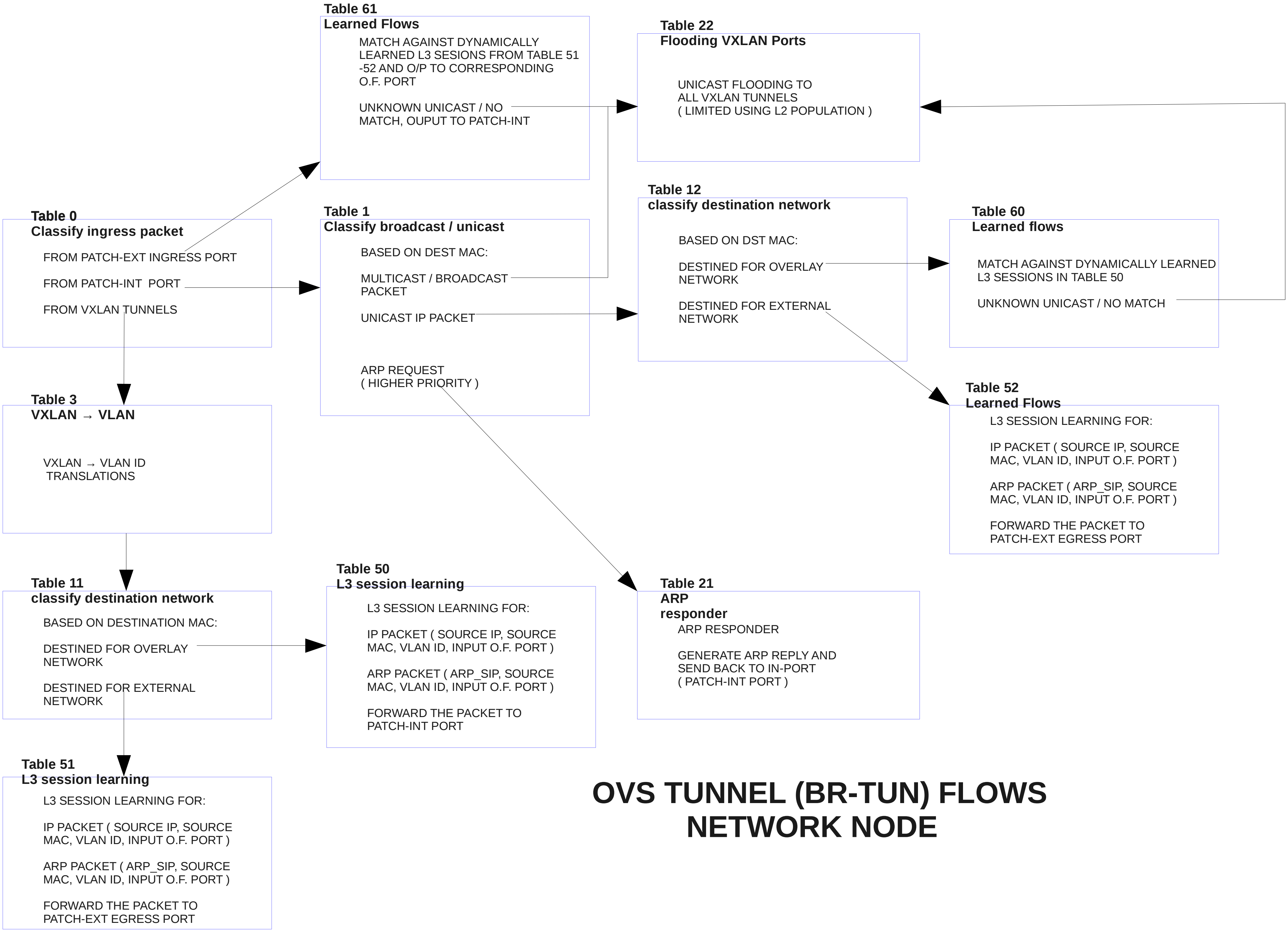


Table 0
Classify ingress packet

FROM PATCH-EXT INGRESS PORT

FROM PATCH-INT PORT

FROM VXLAN TUNNELS

Table 61
Learned Flows

MATCH AGAINST DYNAMICALLY LEARNED L3 SESIONS FROM TABLE 51 -52 AND O/P TO CORRESPONDING O.F. PORT

UNKNOWN UNICAST / NO MATCH, OUPUT TO PATCH-INT

Table 1
Classify broadcast / unicast

BASED ON DEST MAC:

MULTICAST / BROADCAST PACKET

UNICAST IP PACKET

ARP REQUEST
(HIGHER PRIORITY)

Table 22
Flooding VXLAN Ports

UNICAST FLOODING TO
ALL VXLAN TUNNELS
(LIMITED USING L2 POPULATION)

Table 12
classify destination network

BASED ON DST MAC:

DESTINED FOR OVERLAY NETWORK

DESTINED FOR EXTERNAL NETWORK

Table 60
Learned flows

MATCH AGAINST DYNAMICALLY LEARNED L3 SESSIONS IN TABLE 50

UNKNOWN UNICAST / NO MATCH

Table 52
Learned Flows

L3 SESSION LEARNING FOR:

IP PACKET (SOURCE IP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

ARP PACKET (ARP_SIP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

FORWARD THE PACKET TO PATCH-EXT EGRESS PORT

Table 3
VXLAN → VLAN

VXLAN → VLAN ID
TRANSLATIONS

Table 11
classify destination network

BASED ON DESTINATION MAC:

DESTINED FOR OVERLAY NETWORK

DESTINED FOR EXTERNAL NETWORK

Table 50
L3 session learning

L3 SESSION LEARNING FOR:

IP PACKET (SOURCE IP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

ARP PACKET (ARP_SIP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

FORWARD THE PACKET TO PATCH-INT PORT

Table 21
ARP responder

ARP RESPONDER

GENERATE ARP REPLY AND SEND BACK TO IN-PORT
(PATCH-INT PORT)

Table 51
L3 session learning

L3 SESSION LEARNING FOR:

IP PACKET (SOURCE IP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

ARP PACKET (ARP_SIP, SOURCE MAC, VLAN ID, INPUT O.F. PORT)

FORWARD THE PACKET TO PATCH-EXT EGRESS PORT

OVS TUNNEL (BR-TUN) FLOWS NETWORK NODE

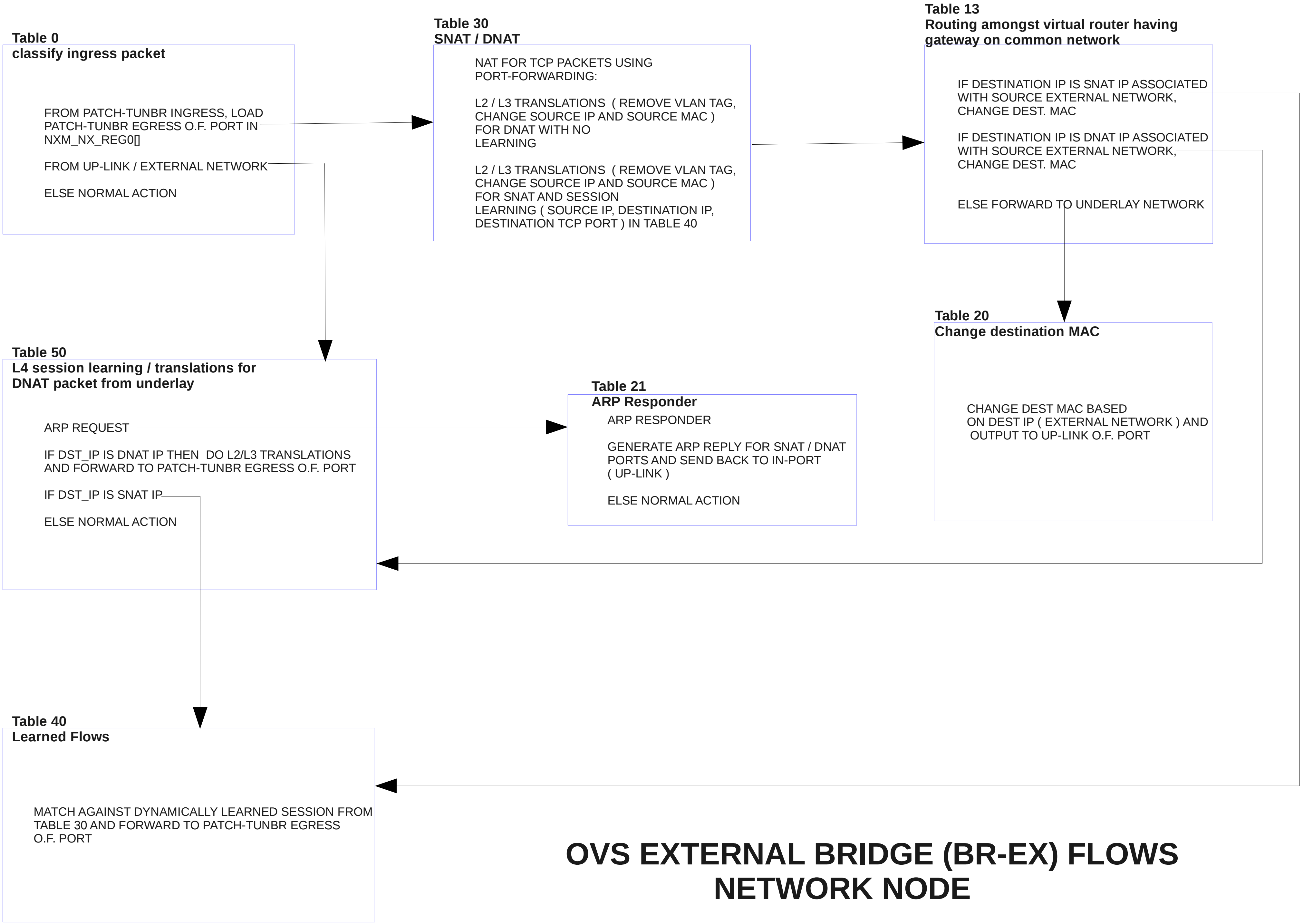


Table 0
classify ingress packet

FROM PATCH-TUNBR INGRESS, LOAD PATCH-TUNBR EGRESS O.F. PORT IN NXM_NX_REG0[]

FROM UP-LINK / EXTERNAL NETWORK

ELSE NORMAL ACTION

Table 30
SNAT / DNAT

NAT FOR TCP PACKETS USING PORT-FORWARDING:

L2 / L3 TRANSLATIONS (REMOVE VLAN TAG, CHANGE SOURCE IP AND SOURCE MAC) FOR DNAT WITH NO LEARNING

L2 / L3 TRANSLATIONS (REMOVE VLAN TAG, CHANGE SOURCE IP AND SOURCE MAC) FOR SNAT AND SESSION LEARNING (SOURCE IP, DESTINATION IP, DESTINATION TCP PORT) IN TABLE 40

Table 13
Routing amongst virtual router having gateway on common network

IF DESTINATION IP IS SNAT IP ASSOCIATED WITH SOURCE EXTERNAL NETWORK, CHANGE DEST. MAC

IF DESTINATION IP IS DNAT IP ASSOCIATED WITH SOURCE EXTERNAL NETWORK, CHANGE DEST. MAC

ELSE FORWARD TO UNDERLAY NETWORK

Table 20
Change destination MAC

CHANGE DEST MAC BASED ON DEST IP (EXTERNAL NETWORK) AND OUTPUT TO UP-LINK O.F. PORT

Table 50
L4 session learning / translations for DNAT packet from underlay

ARP REQUEST

IF DST_IP IS DNAT IP THEN DO L2/L3 TRANSLATIONS AND FORWARD TO PATCH-TUNBR EGRESS O.F. PORT

IF DST_IP IS SNAT IP

ELSE NORMAL ACTION

Table 21
ARP Responder

ARP RESPONDER

GENERATE ARP REPLY FOR SNAT / DNAT PORTS AND SEND BACK TO IN-PORT (UP-LINK)

ELSE NORMAL ACTION

Table 40
Learned Flows

MATCH AGAINST DYNAMICALLY LEARNED SESSION FROM TABLE 30 AND FORWARD TO PATCH-TUNBR EGRESS O.F. PORT

OVS EXTERNAL BRIDGE (BR-EX) FLOWS NETWORK NODE

Performance Analysis

Use Case - SNAT for single host (POC)

No. of VM’s	Bandwidth (Gbps)
1	5.5
2	5.48
3	5.37
4	5.35
5	5.36

Use Case - SNAT for single host (OpenStack)

No. of VM’s	Bandwidth (Gbps)
1	3.44
2	3.38
3	3.48
4	3.39
5	3.14

Performance Analysis

Use Case - SNAT for multiple hosts (POC)

No. of hosts	No. of VMs	Bandwidth (Gbps)
2	2 (one per host)	9.03
3	3 (one per host)	9.07

Use Case - SNAT for multiple hosts (OpenStack)

No. of hosts	No. of VMs	Bandwidth (Gbps)
2	2 (one per host)	6.28
3	3 (one per host)	8.25

Performance Analysis

Use Case - DNAT for single host (POC)

No. of VM’s	**Bandwidth (Gbps)**
1	3.09
2	3.89
3	3.66
4	3.54

Use Case - DNAT for single host (OpenStack)

No. of VM’s	**Bandwidth (Gbps)**
1	3.2
2	4.02
3	3.9
4	3.618

Performance Analysis

Use Case - DNAT for multiple hosts (POC)

No. of hosts	No. of VM’s	Bandwidth (Gbps)
2	2 (one per host)	6.48
3	3 (one per host)	8.73
3	6 (two per host)	9.15
3	9 (three per host)	9.3

Use Case 4- DNAT for multiple hosts (OpenStack)

No. of hosts	No. of VM’s	Bandwidth (Gbps)
2	2 (one per host)	5.97
3	3 (one per host)	8.29