

# Cloud e Datacenter Networking

Università degli Studi di Napoli Federico II

Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione DIETI

Laurea Magistrale in Ingegneria Informatica

Prof. Roberto Canonico

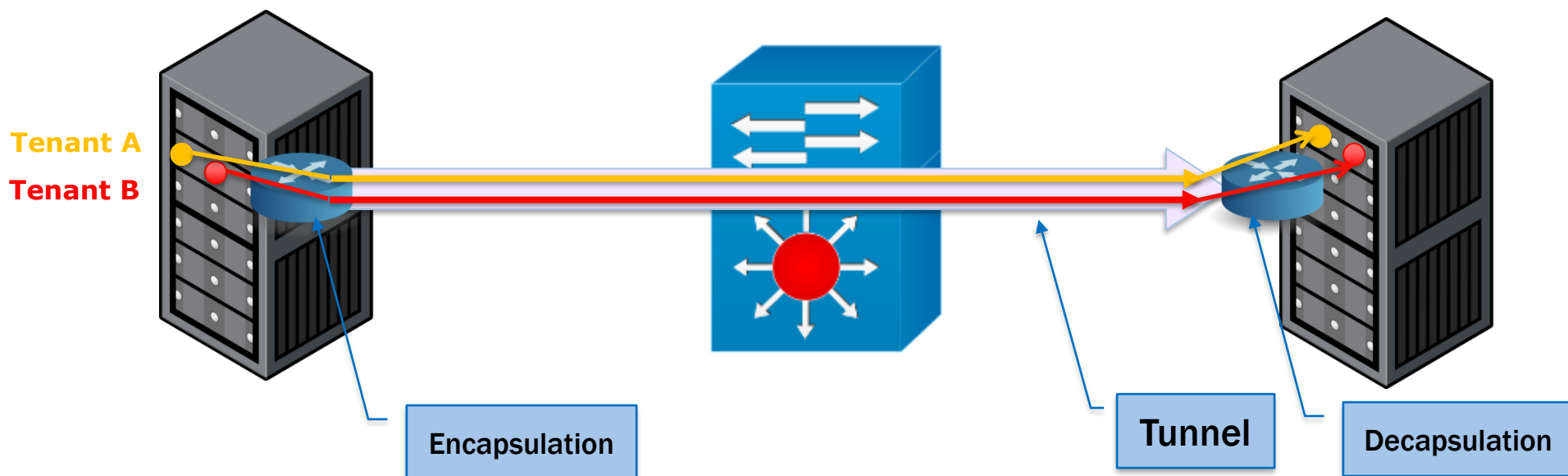
## VXLAN demo with GNS3



# Network Virtualization using encapsulation



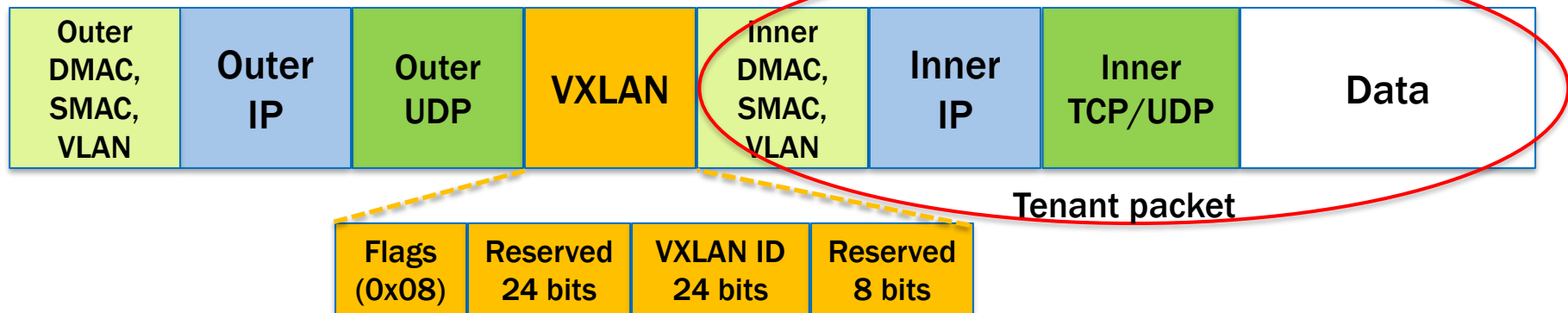
- ▶ VXLAN and NVGRE are two different network virtualization methods that use encapsulation and tunneling to create large numbers of virtual LANs for subnets that can extend across layer 2 and 3
- ▶ Encapsulation/decapsulation is performed by entities that could reside either in End Devices or in ToR edge switches (or in both)
- ▶ VXLAN is supported by Cisco and VMware
- ▶ NVGRE was proposed by Microsoft, Intel, HP and Dell



# VXLAN (RFC 7348)



- ▶ Virtual eXtensible LAN (VXLAN) was originally proposed by Cisco and VMware to tunnel virtual layer 2 networks on a substrate layer 3 physical network

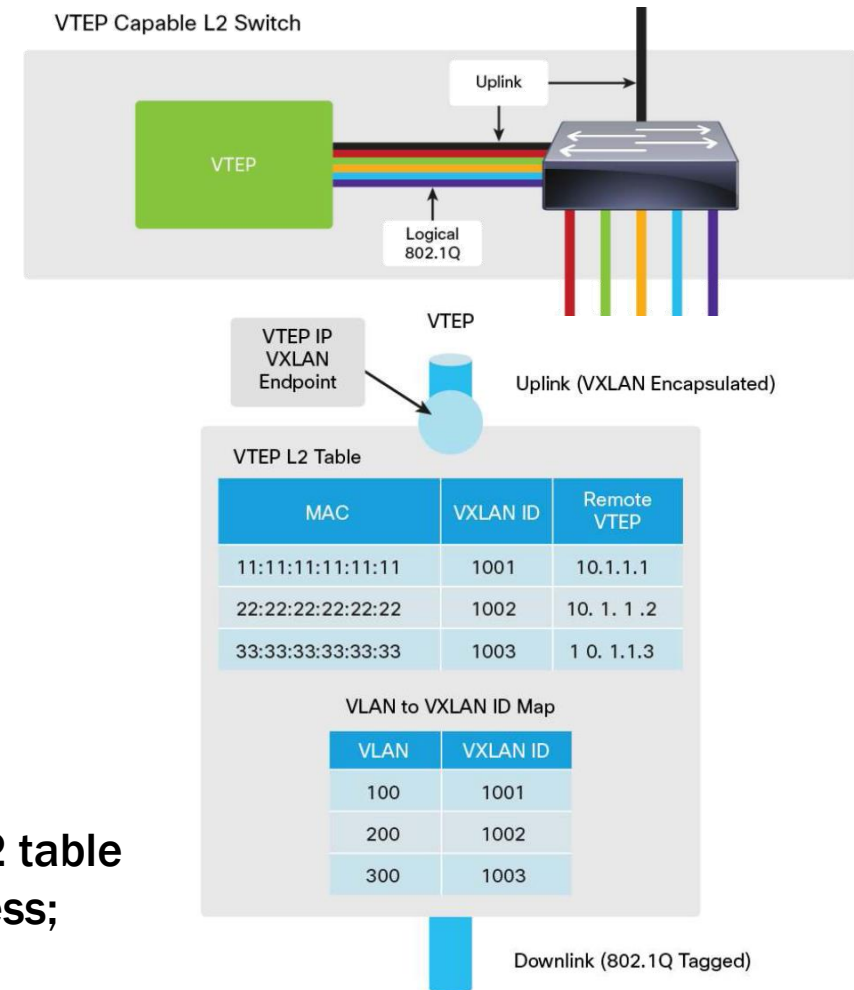


- ▶ VXLAN encapsulate packets in UDP tunnels with destination port number 4789
- ▶ In the shared L3 infrastructure, packets are identified by outer MAC addresses imposed by the infrastructure provider
- ▶ Tenants free to choose their own MAC addresses and VLAN IDs with no conflicts
- ▶ To avoid packet fragmentation in the shared infrastructure, it must support larger MTU values
- ▶ Encapsulation/decapsulation is performed at *VXLAN Tunnel End Points (VTEPs)*
- ▶ VXLAN ID allows to identify up to  $2^{24}$  distinct virtual networks

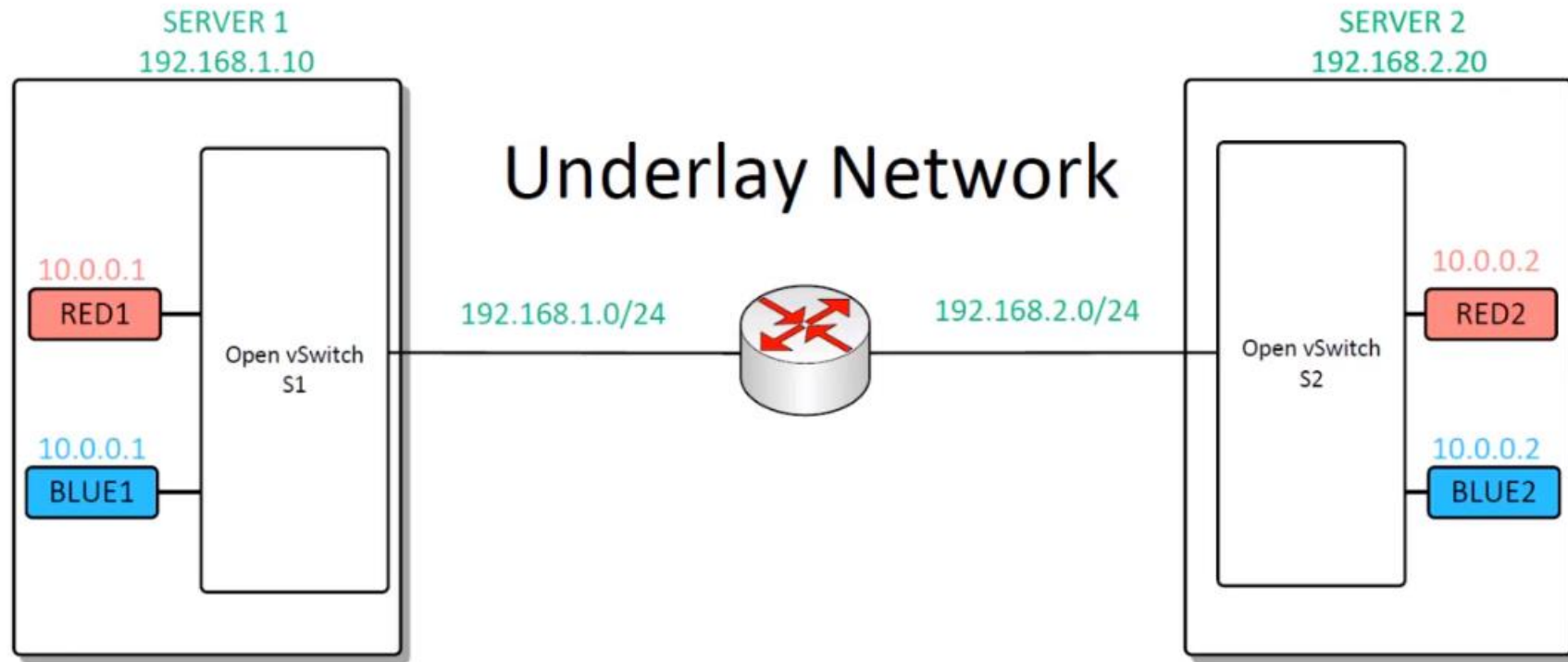
# VXLAN: VTEP encapsulation & decapsulation



- ▶ A VTEP has two logical interfaces: an uplink and a downlink
  - ▶ Uplink to encapsulate
  - ▶ Downlink to decapsulate
- ▶ The VTEP can be located either on a physical switch (e.g. a ToR) or within the hypervisor's virtual switch
- ▶ The *outer IP destination* address is that assigned to the destination VTEP
- ▶ The *outer IP source* address is that assigned to the VTEP sending the frame
- ▶ Packets received from a tenant's VM on the downlink are mapped to a VXLAN ID
  - ▶ A lookup is then performed in the VTEP Layer 2 table using the VXLAN ID and destination MAC address; this lookup provides the IP address of the destination VTEP
- ▶ Packets received from a VTEP on the uplink are mapped from the VXLAN ID to an IEEE 802.1Q VLAN ID and sent as Ethernet frames on the downlink to the VM



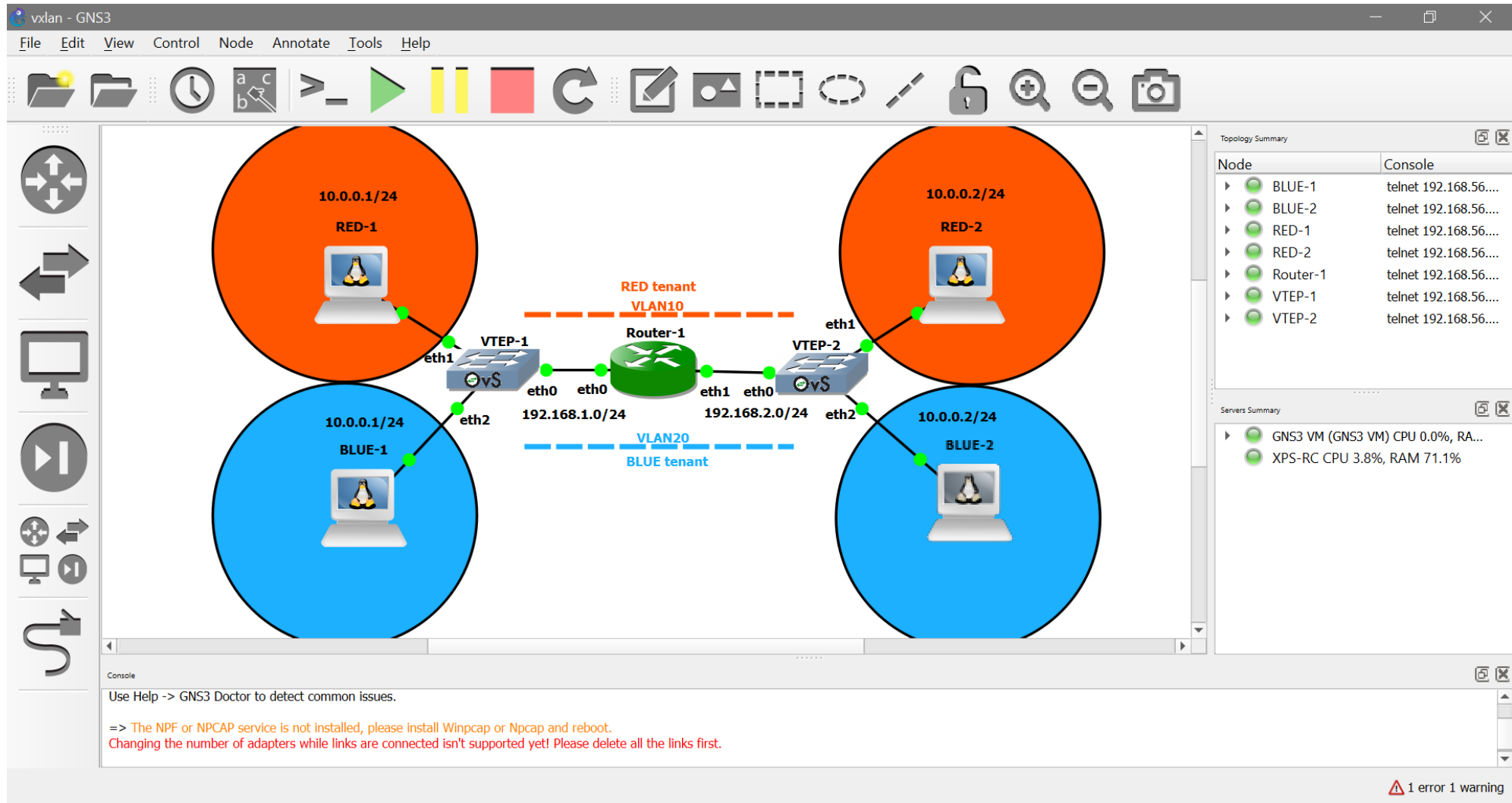
# Demo: logical setup



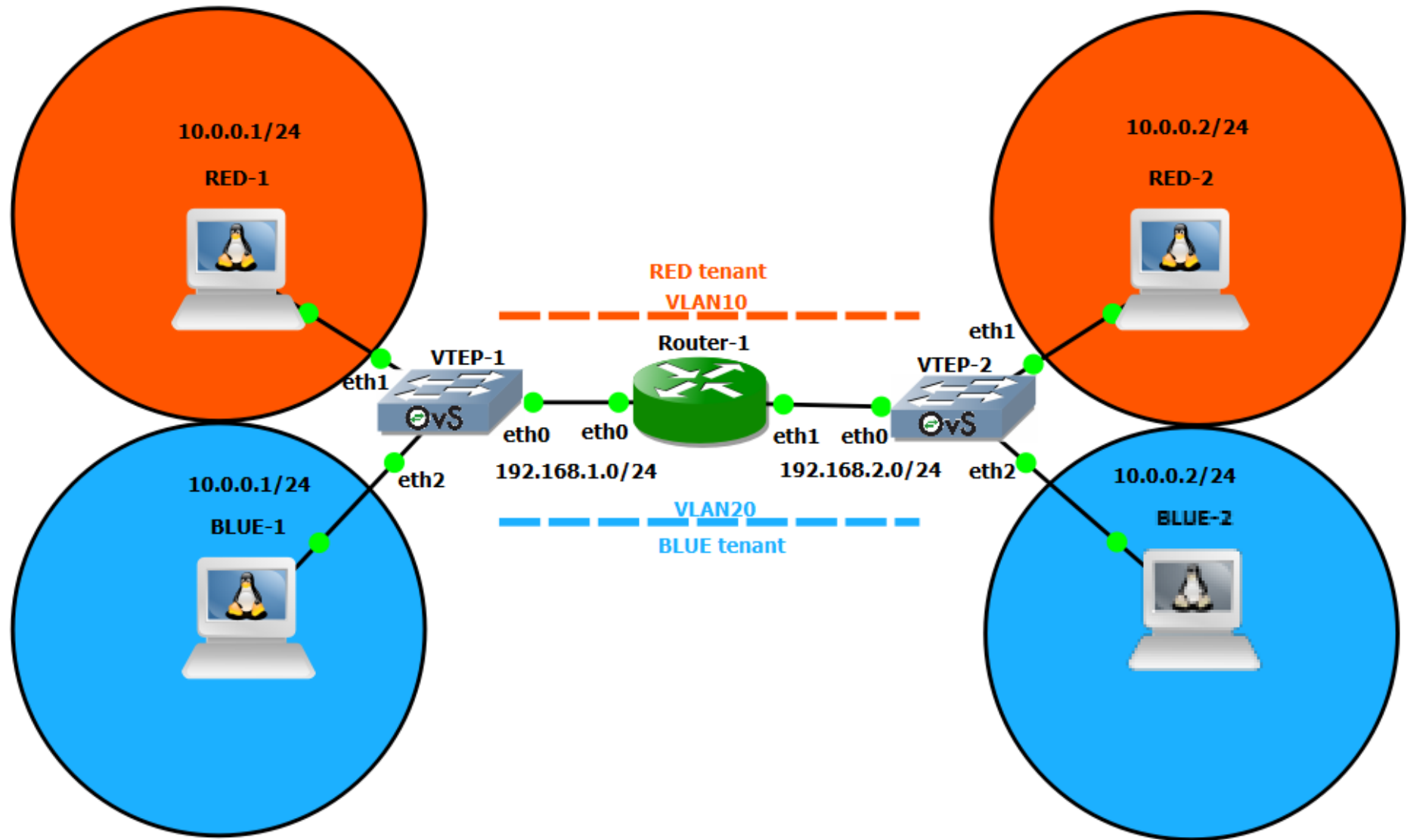
A similar demo based on mininet is presented here:

<https://www.youtube.com/watch?v=QUmRUSqaAzc>

# Demo setup in GNS3 (1)



# Demo setup in GNS3 (2)



- ▶ 4 end-systems implemented as Docker containers from the: **gns3/ubuntu:xenial** Docker image
- ▶ 1 Linux-based IP router implemented as a Docker container with an image derived from the: **kathara/base:debian10** Docker image
- ▶ 2 Open-vSwitch based VTEPs implemented as Docker containers from the: **gns3/openvswitch:latest** Docker image

## ► Scripts to be executed to activate the two VTEPs

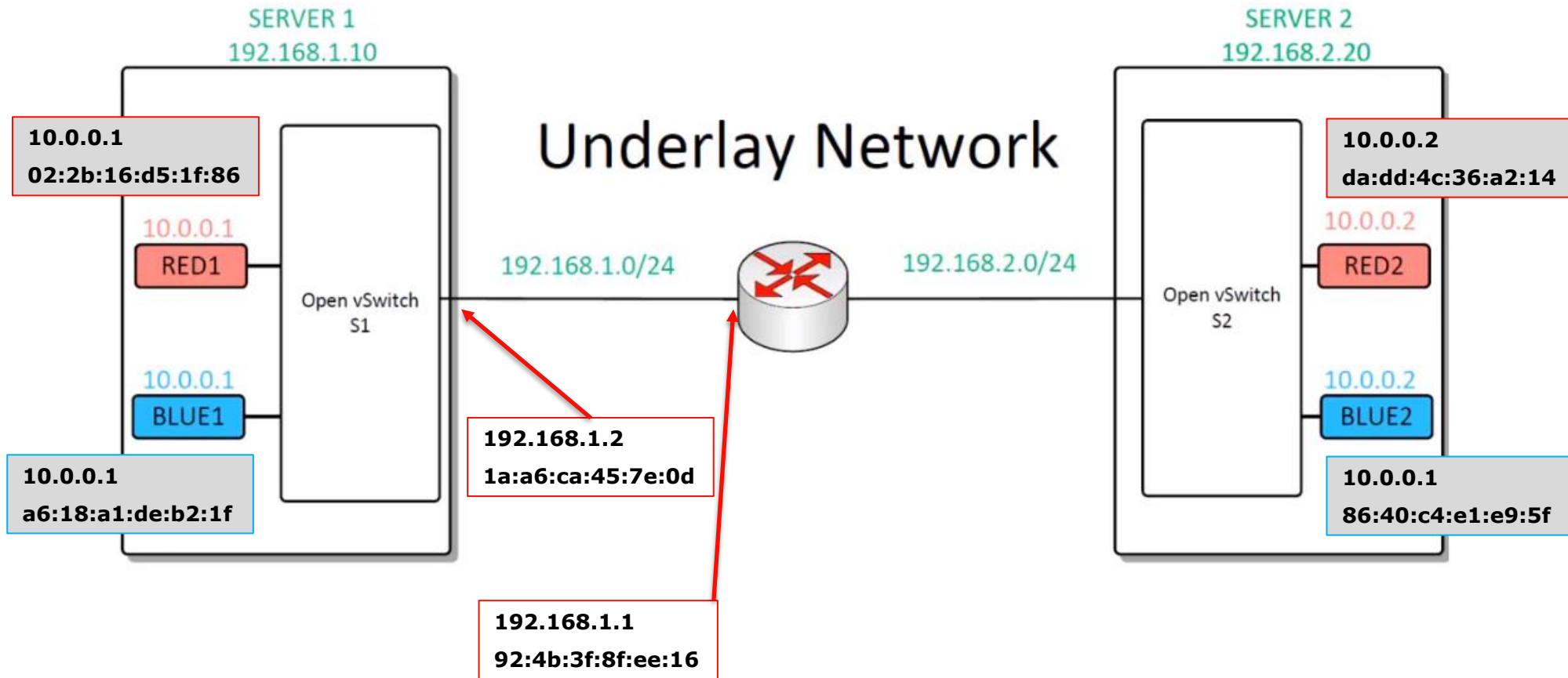
### VTEP-1: ovs1.sh

```
#!/bin/sh
for br in `ovs-vsctl list-br`; do ovs-vsctl del-br ${br}; done
ovs-vsctl add-br ovs0
ovs-vsctl add-port ovs0 eth1 tag=10
ovs-vsctl add-port ovs0 vxlan1 tag=10 -- set interface vxlan1 type=vxlan \
    options:key=10 options:remote_ip=192.168.2.2
ovs-vsctl add-port ovs0 eth2 tag=20
ovs-vsctl add-port ovs0 vxlan2 tag=20 -- set interface vxlan2 type=vxlan \
    options:key=20 options:remote_ip=192.168.2.2
```

### VTEP-2: ovs2.sh

```
#!/bin/sh
for br in `ovs-vsctl list-br`; do ovs-vsctl del-br ${br}; done
ovs-vsctl add-br ovs0
ovs-vsctl add-port ovs0 eth1 tag=10
ovs-vsctl add-port ovs0 vxlan1 tag=10 -- set interface vxlan1 type=vxlan \
    options:key=10 options:remote_ip=192.168.1.2
ovs-vsctl add-port ovs0 eth2 tag=20
ovs-vsctl add-port ovs0 vxlan2 tag=20 -- set interface vxlan2 type=vxlan \
    options:key=20 options:remote_ip=192.168.1.2
```

# Demo: MAC and IP addresses in the GNS3 demo setup



# Demo: analysis of packets with Wireshark



vxlan-gns3.pcapng

File Modifica Visualizza Vai Cattura Analizza Statistiche Telefonja Wireless Strumenti Aiuto

Applica un filtro di visualizzazione ... <Ctrl-/>

| No. | Time     | Source            | Destination       | Protocol | Length | Info                                                          |
|-----|----------|-------------------|-------------------|----------|--------|---------------------------------------------------------------|
| 1   | 0.000000 | 1a:a6:ca:45:7e:0d | 92:4b:3f:8f:ee:16 | ARP      | 42     | Who has 192.168.1.1? Tell 192.168.1.2                         |
| 2   | 0.000080 | 92:4b:3f:8f:ee:16 | 1a:a6:ca:45:7e:0d | ARP      | 42     | 192.168.1.1 is at 92:4b:3f:8f:ee:16                           |
| 3   | 7.304699 | 02:2b:16:d5:1f:86 | ff:ff:ff:ff:ff:ff | ARP      | 92     | Who has 10.0.0.2? Tell 10.0.0.1                               |
| 4   | 7.305010 | da:dd:4c:36:a2:14 | 02:2b:16:d5:1f:86 | ARP      | 92     | 10.0.0.2 is at da:dd:4c:36:a2:14                              |
| 5   | 7.305156 | 10.0.0.1          | 10.0.0.2          | ICMP     | 148    | Echo (ping) request id=0x0044, seq=1/256, ttl=64 (reply in 6) |
| 6   | 7.305332 | 10.0.0.2          | 10.0.0.1          | ICMP     | 148    | Echo (ping) reply id=0x0044, seq=1/256, ttl=64 (request in 5) |
| 7   | 8.305165 | 10.0.0.1          | 10.0.0.2          | ICMP     | 148    | Echo (ping) request id=0x0044, seq=2/512, ttl=64 (reply in 8) |

> Frame 23: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits) on interface -, id 0

✓ Ethernet II, Src: 1a:a6:ca:45:7e:0d, Dst: 92:4b:3f:8f:ee:16

- > Destination: 92:4b:3f:8f:ee:16
- > Source: 1a:a6:ca:45:7e:0d
- Type: IPv4 (0x0800)

> Internet Protocol Version 4, Src: 192.168.1.2, Dst: 192.168.2.2

> User Datagram Protocol, Src Port: 51220, Dst Port: 4789

> Virtual eXtensible Local Area Network

✓ Ethernet II, Src: a6:18:a1:de:b2:1f, Dst: 86:40:c4:e1:e9:5f

- > Destination: 86:40:c4:e1:e9:5f
- > Source: a6:18:a1:de:b2:1f
- Type: IPv4 (0x0800)

> Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2

```
0000  92 4b 3f 8f ee 16 1a a6 ca 45 7e 0d 08 00 45 00  ·K?....·E~...E·
0010  00 86 b4 37 40 00 40 11 01 db c0 a8 01 02 c0 a8  ··7@·@·.....
0020  02 02 c8 14 12 b5 00 72 00 00 08 00 00 00 00 00  ····r·.....
0030  14 00 86 40 c4 e1 e9 5f a6 18 a1 de b2 1f 08 00  ··@·_·.....
0040  45 00 00 54 7b 4b 40 00 40 01 ab 5b 0a 00 00 01  E·T{K@·@·[·...
0050  0a 00 00 02 08 00 9e 1b 00 42 00 01 9f 39 a6 60  ······B··9·`
0060  00 00 00 00 47 34 0e 00 00 00 00 00 10 11 12 13  ···G4·.....
0070  14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23  ······!"#
```

vxlan-gns3.pcapng

Pacchetti: 34 · visualizzati: 34 (100.0%)

Profilo: Default

- ▶ Containers in GNS3 series: Advanced OpenVswitch switching  
<https://gns3.com/community/blog/containers-in-gns3-series-advanc>
- ▶ Connecting VMs Using Tunnels (Userspace)  
<https://docs.openvswitch.org/en/latest/howto/userspace-tunneling/>