SDNFV FINAL PROJECT

SDN Network as Virtual Router

助教：蔣汶儒

sdnta@win.cs.nycu.edu.tw

Deadline: 2025/6/4

OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Project Information and Installation
- Scoring Criteria
- Reference

OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Project Information and Installation
- Scoring Criteria
- Reference

Review of Labs

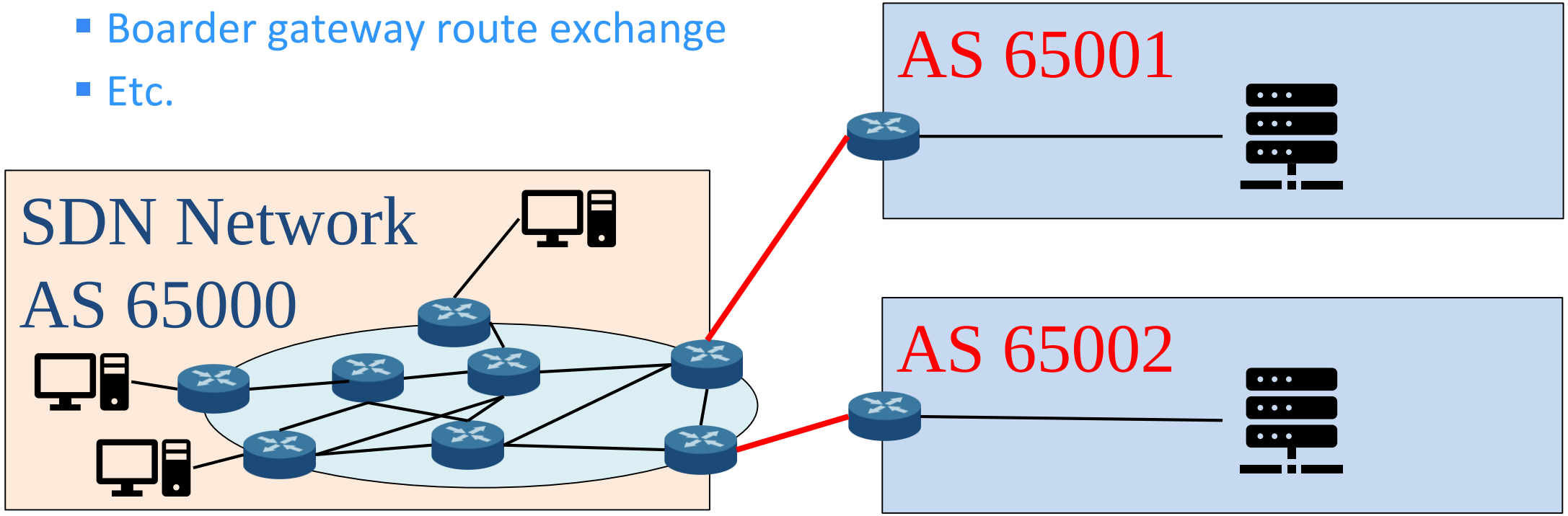
- Lab2
  - ONOS API
  - Flow rules
- Lab3 Network Function Virtualization
  - Simulate Autonomous Systems (AS)

OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Project Information and Installation
- Scoring Criteria
- Reference

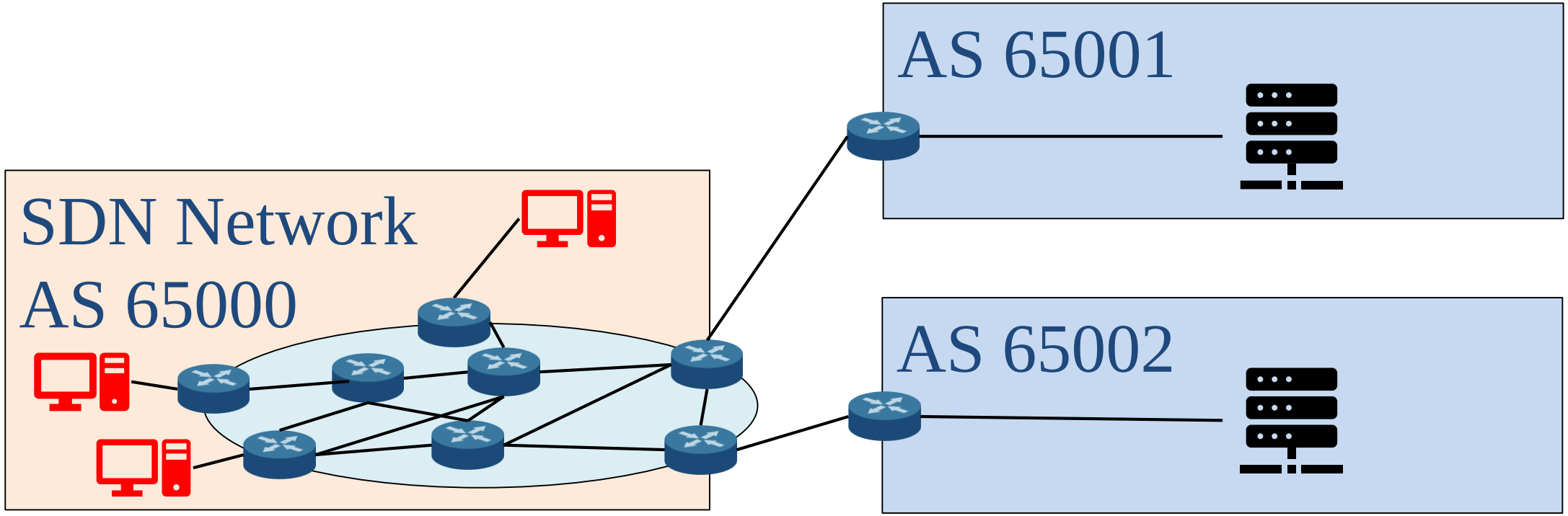
SDN-enabled Virtual Router

- SDN Network with virtual router
  - Use openflow switches and flowrules to simulate router behavior
  - For instance:
    - Layer2 forwarding for next hop communication
    - Boarder gateway route exchange
    - Etc.



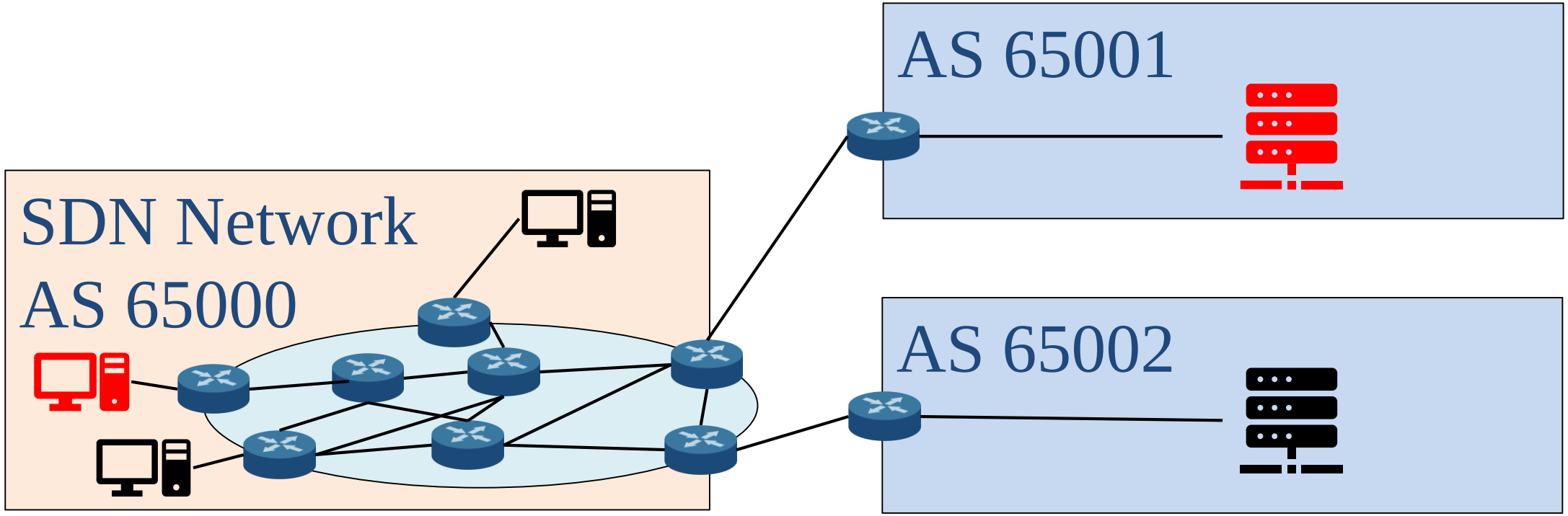
Traffic Types

- Intra-domain Traffics
  - Where **hosts** within the same AS communicates with each other.
  - SDN handles the traffic.



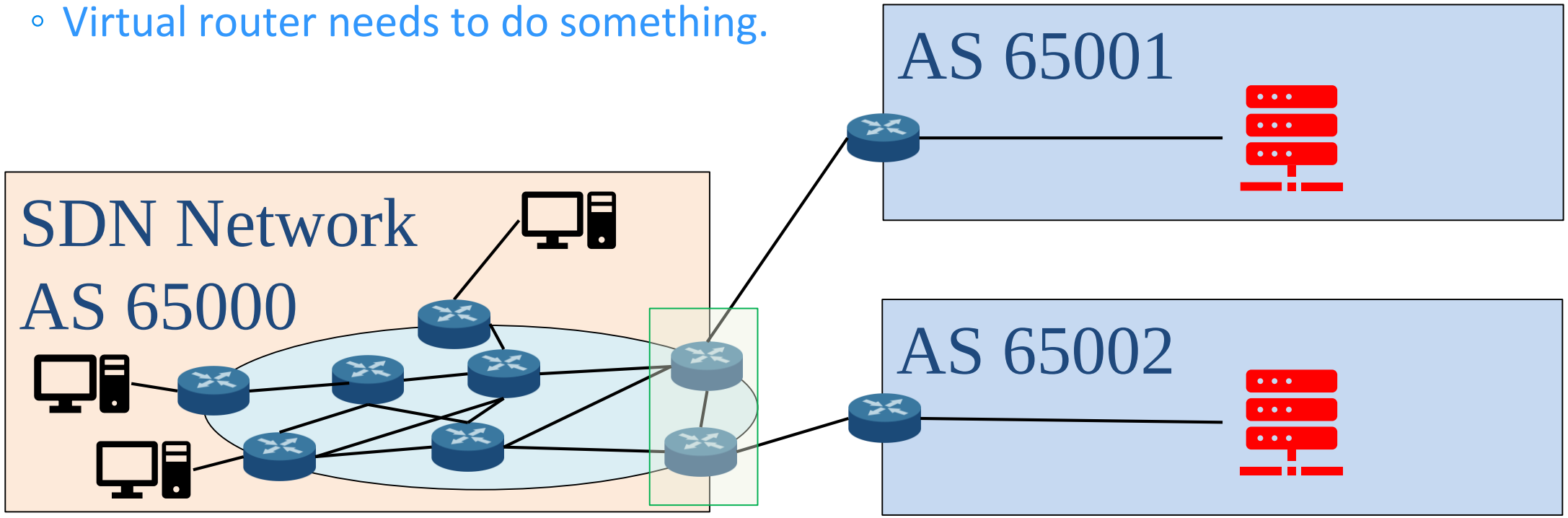
Traffic Types (cont.)

- Inter-domain Traffics
  - Where **an external host** from other domain communicates with **an internal host**.
  - The traffic pass through **gateways**.
  - Virtual router needs to do something.



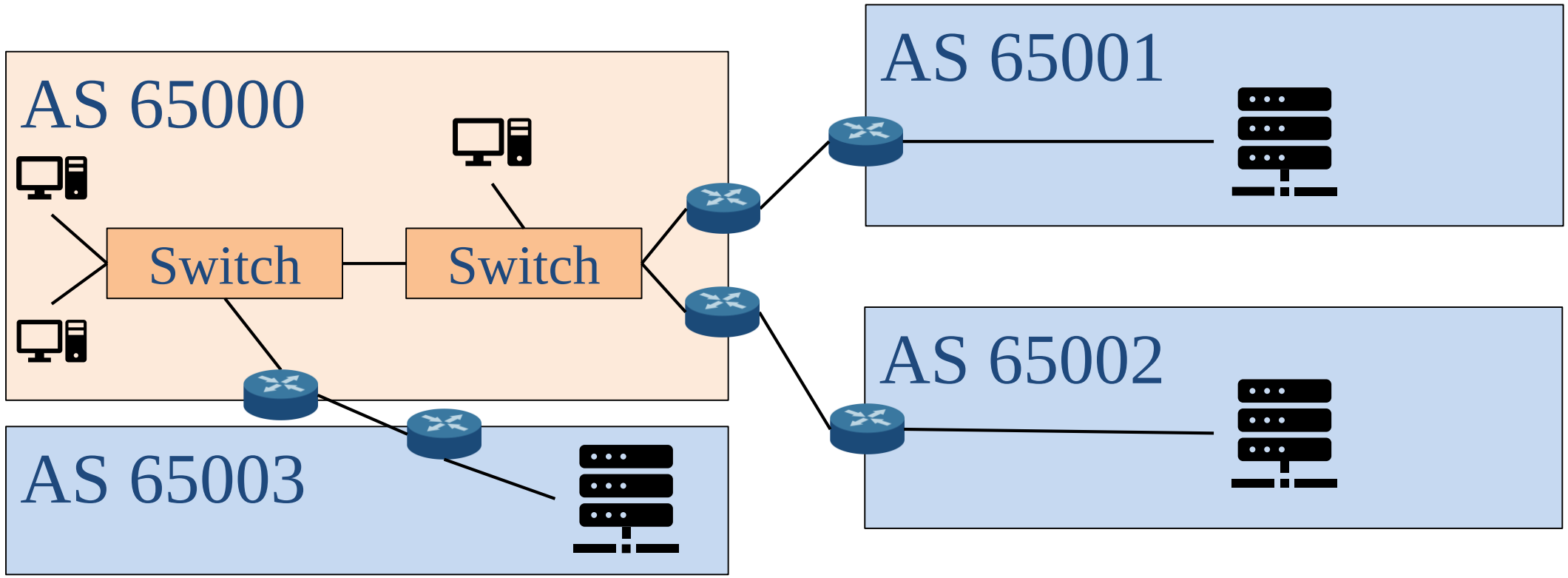
Traffic Types (cont.)

- Transit Traffics
  - Where **hosts** from different domains communicates with one another bypass the SDN network.
  - The traffic pass through virtual router.
  - Virtual router needs to do something.



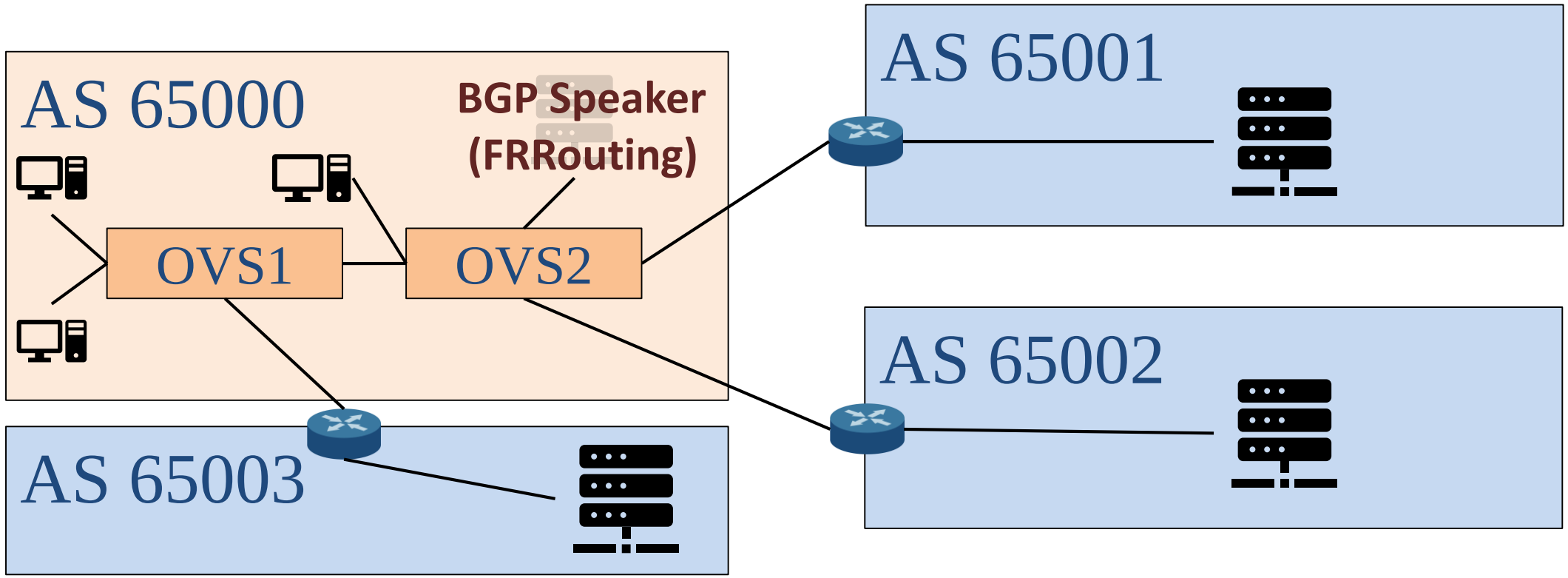
Networks with Physical Routers

- Physical routers
  - 1. Deal with routing decision.
  - 2. Deal with gateway exchange.
- Every edge requires a router, running eBGP and iBGP protocols.



SDN Networks with Virtual Routers

- SDN-enabled Virtual Routers
  - Doesn’t requires router connection to edge.
  - Only one BGP speaker is enough.
  - Doesn’t need a **real gateway.**



OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Project Information and Installation
- Scoring Criteria
- Reference

Goal

- Intra-domain host communication
  - Flow rule
- Inter-domain host communication
  - SDN domain <-> Other domain
- Transit host communication
  - Other domain <-> SDN domain <-> Other domain

vRouter Specification

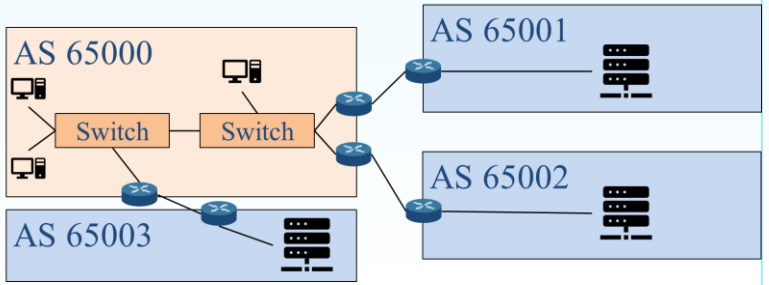
- Intra AS packet forwarding and packet-in request
- Arp Reply (gateway) for devices in AS
- Inter-domain eBGP traffic topology
- Routing table maintenance
- Flowrules for intra/inter/transit domain traffic

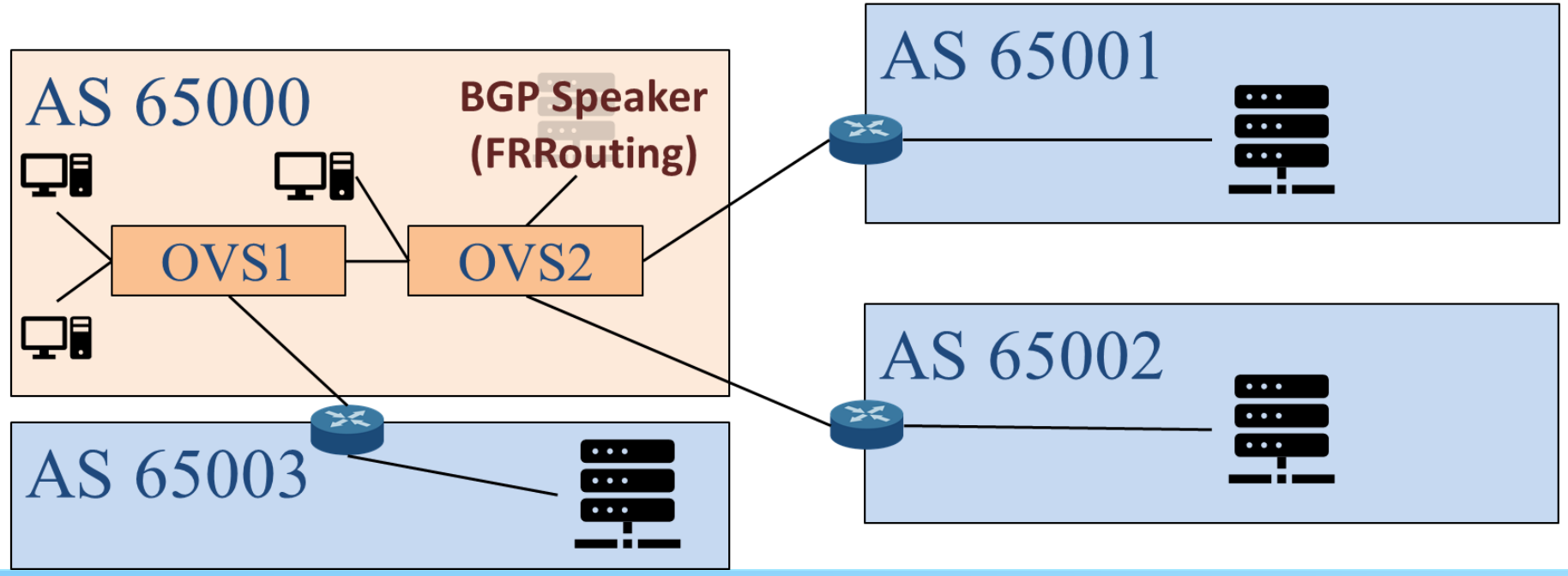
OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Project Information and Installation
- Scoring Criteria
- Reference

Virtual Router BGP Connection

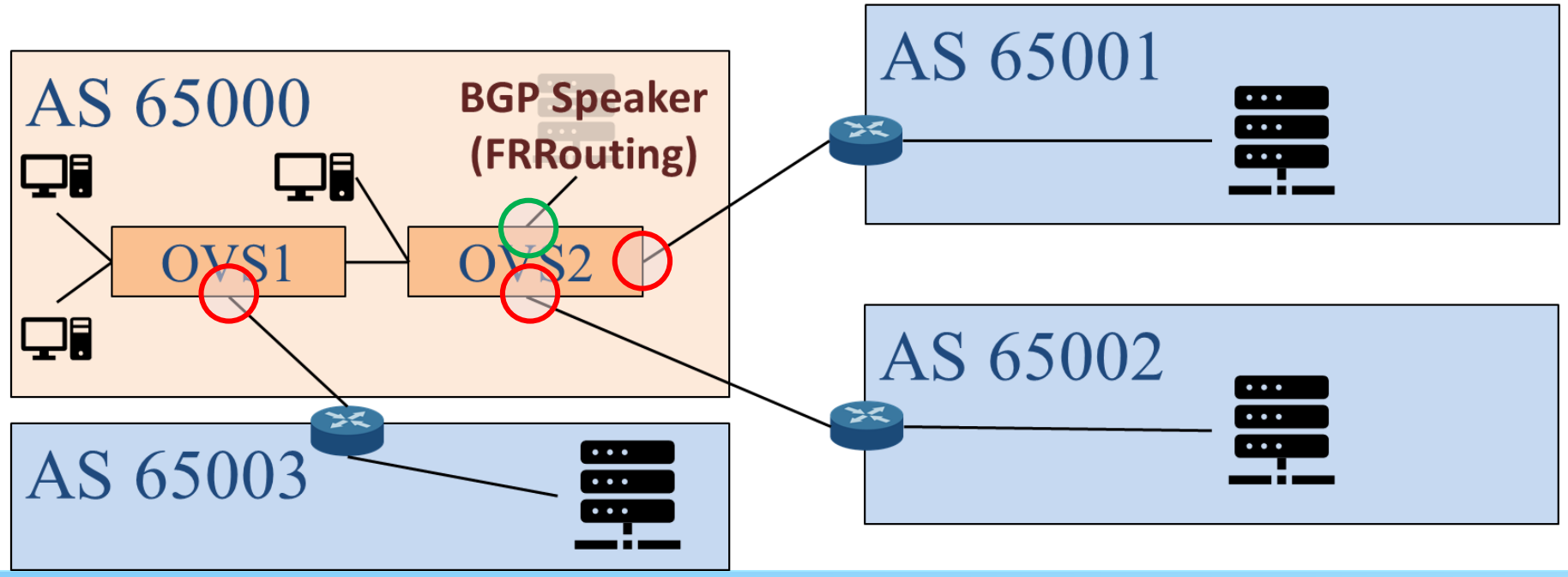
- Physical router:
  - External routers connect with the boarder gateway.
- Virtual router:
  - External routers connect with BGP Speaker.
    - ➢Simulate edge switch as router port (Looks like IP on edge switch).





BGP Speaker IP Delegation and Routing

1. Delegate BGP speaker IP to the WAN Connect Point on edge switch.
   1) Determine WAN Connect Point.
2. Route packet between BGP speaker Connect Point and WAN Connect Point.
   1. Determine BGP speaker connect point.
   2. Install rule to route packets.



Zebra FIB Pushing

- Zebra supports a Forwarding Information Base (FIB) Push Interface (FPI)
  - FPI allows an external component to learn the forwarding information.
- Forwarding Plane Manager (FPM)
  - Receives FIB
  - Decode FIB into routes
- FIB pushing:
  - FPM establishes a TCP connection with Zebra
  - Zebra pushes FIB to FPM
- In this project, we use ONOS built-in FPM to collect FIB from zebra.

```
karaf@root > app activate org.onosproject.fpm
```

BGP Route Retrieval with Route Service

- Route Service will collect route information via **FPM APP**.
- Routes provided by Route Service contains next hop info for target subnet.

```
karaf@root > routes
                            01:57:40
B: Best route, R: Resolved route

Table: ipv4
B R  Network            Next Hop         Source (Node)
> *  172.17.1.0/24      192.168.63.2     FPM (192.168.70.1)
   Total: 1

Table: ipv6
B R  Network                                  Next Hop                                Source (Node)
> *  2400:6180::/48                           fe80::42:c0ff:fea8:46fd                 FPM (192.168.70.1)
> *  2400:6180:100::/40                       fe80::42:c0ff:fea8:46fd                 FPM (192.168.70.1)
```

OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- Zebra and FRRouting Configurations
- Virtual Router Workflow
- Project Information and Installation
- Supplement
- Scoring Criteria
- Reference

Enabling FPM Module

- To enable FPM, you have to set –M fpm in zebra_options at /etc/frr/daemons

```
15  # The watchfrr, zebra and staticd daemons are always started.
16  #
17  bgpd=yes
18  ospfd=no
19  ospf6d=no
20  ripd=no
21  ripngd=no
22  isisd=no
23  pimd=no
24  pim6d=no
25  ldpd=no
26  nhrpd=no
27  eigrpd=no
28  babeld=no
29  sharpd=no
30  pbrd=no
31  bfdd=no
32  fabricd=no
33  vrrpd=no
34  pathd=no
35
36  #
37  # If this option is set the /etc/init.d/frr script automatically loads
38  # the config via "vtysh -b" when the servers are started.
39  # Check /etc/pam.d/frr if you intend to use "vtysh"!
40  #
41  vtysh_enable=yes
42  zebra_options="  -A 127.0.0.1 -s 90000000 -M fpm"
```

FRRouting Configuration

- Configurations in /etc/frr/frr.conf

```
1  ! BGP configuration for frr
2  !
3  frr defaults datacenter
4  !
5  fpm connection ip 192.168.100.1 port 2620
6  !
7  router bgp 65010
8  bgp router-id 192.168.70.1
9  timers bgp 3 9
10 neighbor PEER peer-group
11 neighbor PEER ebgp-multihop
12 neighbor PEER timers connect 5
13 neighbor PEER advertisement-interval 5
14 neighbor 192.168.63.2 remote-as 65011
15 neighbor 192.168.63.2 peer-group PEER
16 neighbor 192.168.70.253 remote-as 65000
17 neighbor 192.168.70.253 password winlab.nycu
18 neighbor 192.168.70.253 peer-group PEER
19 neighbor 192.168.70.253 solo
20 neighbor fd63::2 remote-as 65011
21 neighbor fd63::2 peer-group PEER
22 neighbor fd70::fe remote-as 65000
23 neighbor fd70::fe password winlab.nycu
24 neighbor fd70::fe peer-group PEER
25 neighbor fd70::fe solo
```

FPM connection

Peer Group (template) for neighbors

Use the template

BGP Passwords

Don’t advertise the prefix that you received

Announce IPv4 prefix on IPv4 Interface

```
27 address-family ipv4 unicast
28  network 172.16.1.0/24
29  neighbor 192.168.63.2 activate
30  neighbor 192.168.70.253 activate
31  no neighbor fd63::2 activate
32  no neighbor fd70::fe activate
33 exit-address-family
```

NOTE* Older versions of FRRouting might not work, this is just an example

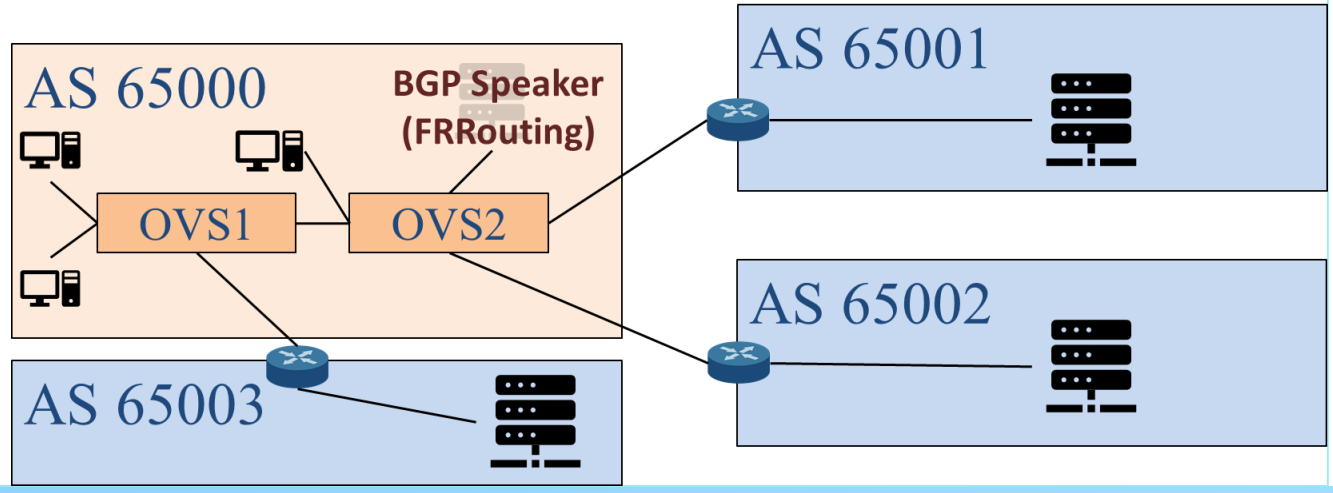
OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Project Information and Installation
- Supplement
- Scoring Criteria
- Reference

BGP Message Exchange

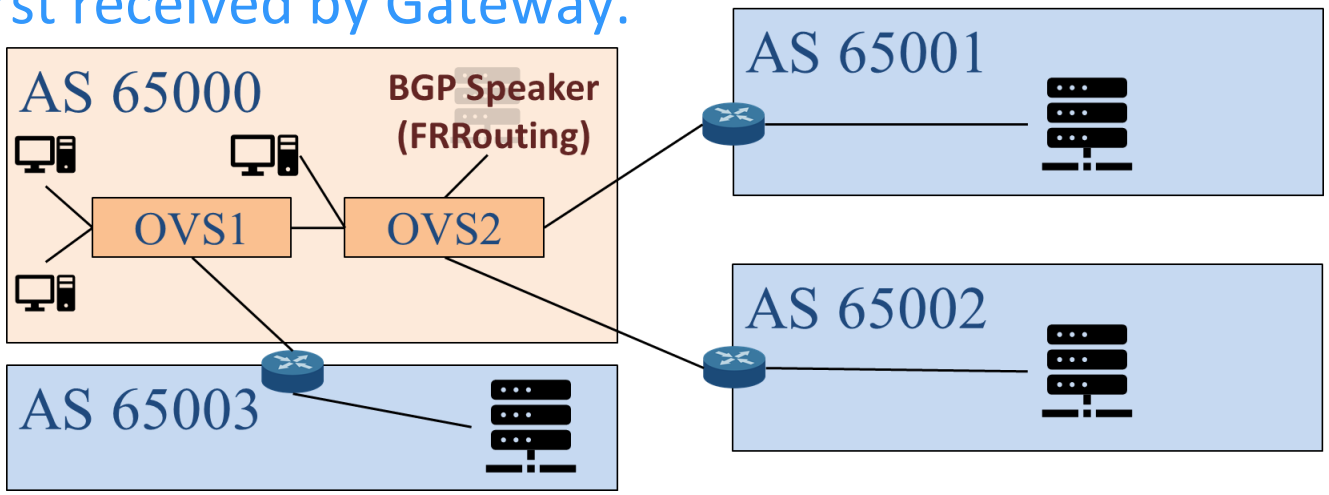
- In order to exchange BGP message with neighbor router
  - Neighbor discovery for L2 connectivity
    - Connection between BGP Speaker and edge routers.
  - L3 forwarding for BGP Messages

- **L3 forwarding for BGP Messages?**
  - **Incoming**
    - ????
  - **Outgoing**
    - ????



Virtual Gateway and Inter-domain Routing

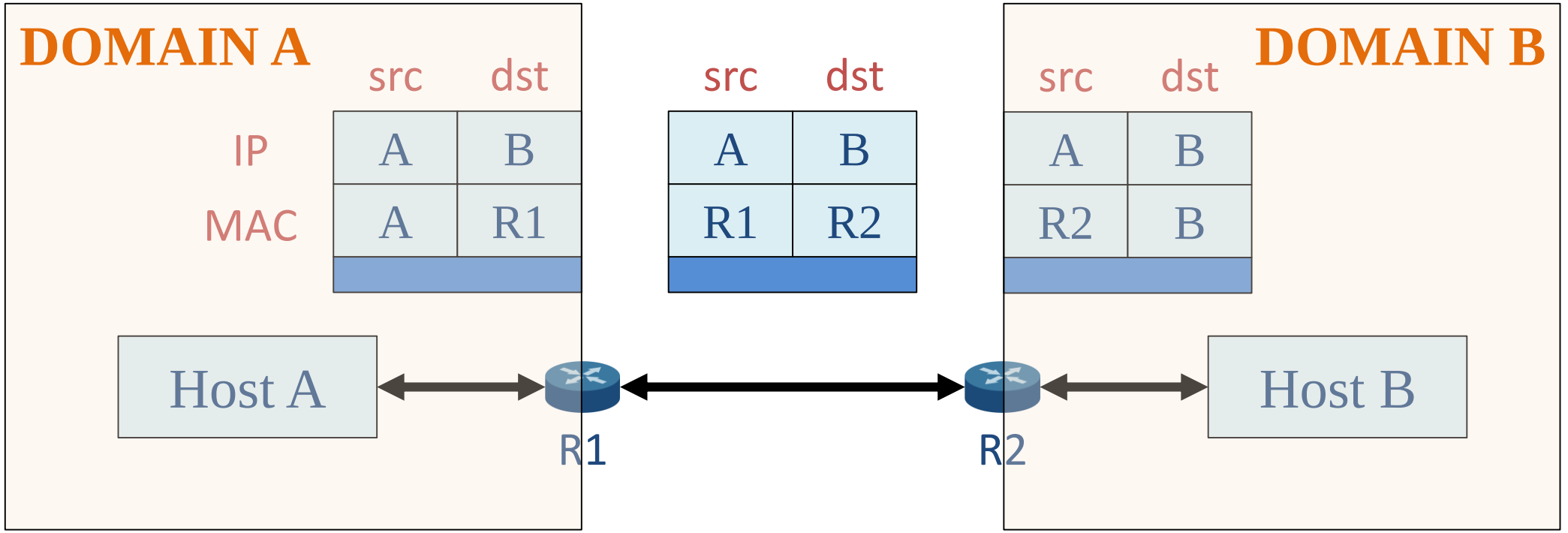
- Gateway and Routing
  - Assume Gateway IP: 192.168.1.254/24
    - Packets originated from 192.168.1.0/24 towards other networks
      - Packet first sent to Gateway.
    - Packet coming from other networks destined 192.168.1.0/24
      - Packet first received by Gateway.



- IP is the logical address of ultimate destination.
  - But, MAC is the physical address of the next hop.

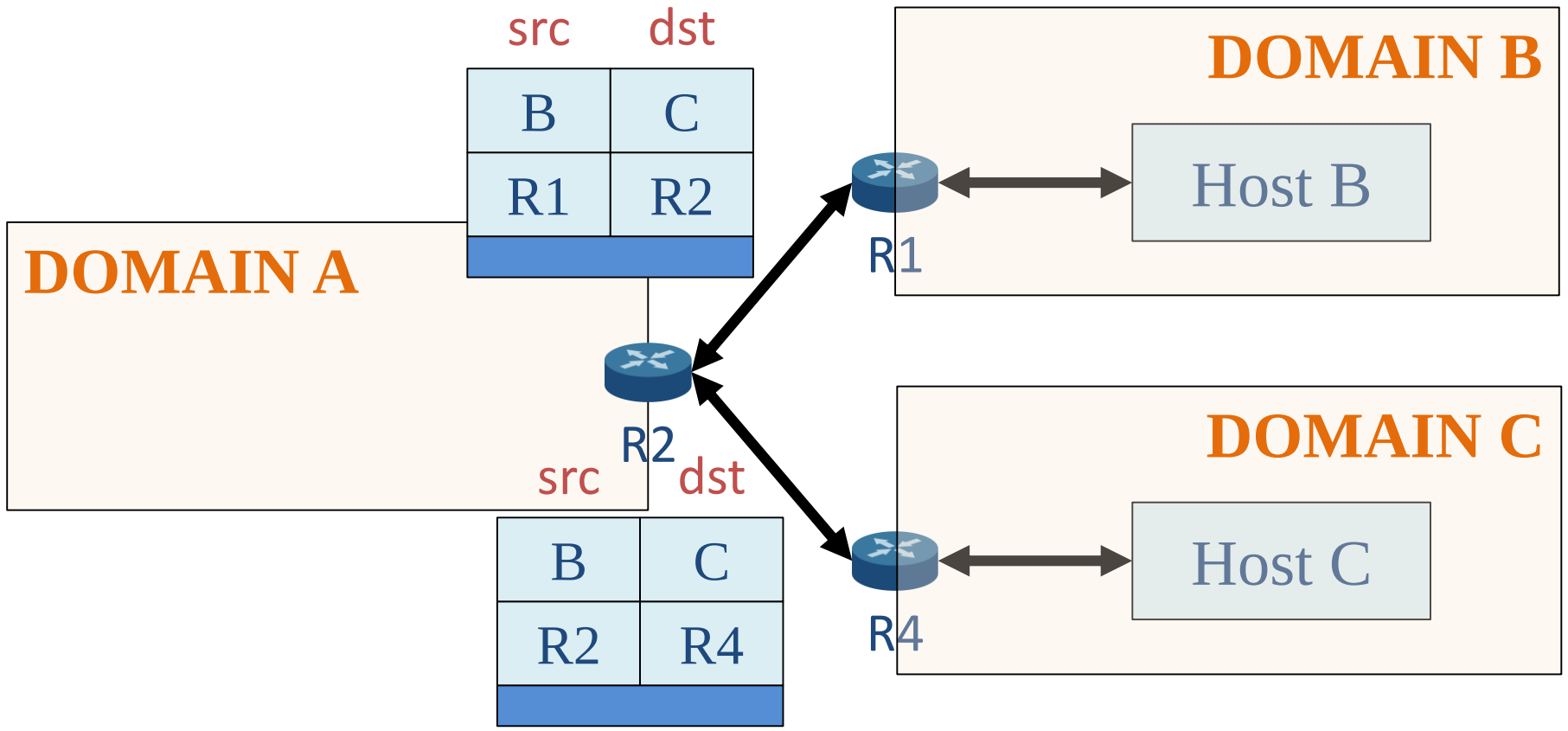
Gateway Traffic Handling Example (Inter Domain)

- Any packets within the domain only knows about the gateway's MAC.
- After analyzing the information (IP), it will change the according MAC and sends the packet out.



Transit Traffics

- Transit traffics are in fact two interdomain traffics.



OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Project Information and Installation
- Supplement
- Scoring Criteria
- Reference

Linux Network Brief Introduction

- **Network Namespaces**
  - Provide a way to create isolated network environments within a Linux system.
  - Allow processes to have their own network stack, including interfaces, routing tables, and firewall rules.

- **Each Container have it's own Network Namespace.**

- A network **Bridge** is a kernel created logical L2 switch

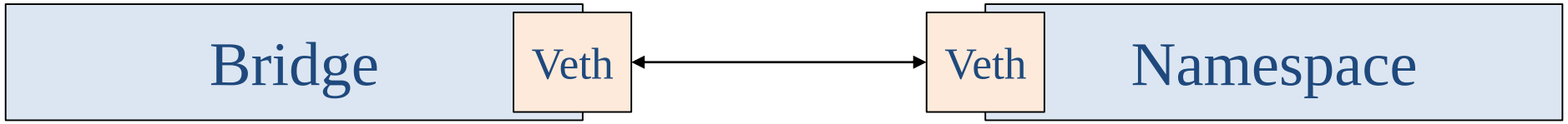
- **Veth** devices, short for virtual Ethernet devices
- Use **veth pairs** to connect Network Namespaces or Bridges together.

Linux Network Brief Introduction (cont.)

- Mapping to physical instruments.

- Namespace = node (computer/server)
- Veth pair = 2 network interface cards (NIC) that connects to each other
- Bridge = switch

- If you want to connect your computer to a switch
  - Create a veth pair (Create 2 NIC that connects to each other)
  - Connect one NIC to your namespace
  - Connect the other one to your bridge



Linux Network Brief Introduction (cont.)

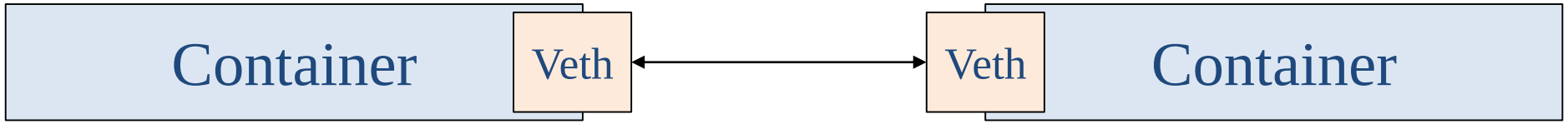
- Mapping to physical instruments.

- Namespace = node (computer/server)
- Veth pair = 2 network interface cards (NIC) that connects to each other
- Bridge = switch
- How to find a container's namespace (ns)?
  - Locate docker process id (pid)

```
n0ball@SDN-NFV:~/workspace$ docker inspect -f '{{.State.Pid}}' $(docker ps -aqf "name=sdnfv-demo")
1761815
```

  - It is at file `/proc/$pid/ns/net`

- Similarly you can connect two namespaces (containers) with the same mechanism.



Ubuntu IP Command Introduction

- Normally, we can use `ip netns exec` command to execute commands inside a namespace; however, it will only search ns for directories in `/var/run/netns`

- Two ways to run `ip netns exec` in container namespace
  - Create a soft link `ln -sfT /proc/$pid/ns/net /var/run/netns/$pid`
  - Use nsenter command `nsenter -t $pid -n <command>`

- Useful ip commands
  - `ip link add <name> type <type>`: Create a NIC by the type.
  - `ip link set <name> up`: Bring up (enable) the NIC.
  - `ip address add <ip> dev <name>`: Add an ip address to the NIC.
  - `ip route show`: Show current routes.
  - `ip route add {<ip> | default} via {ip}: Add a route.

Docker Network Namespace Introduction

```
n0ball@SDN-NFV:~/workspace$ docker run -d --rm --name sdnfv-demo alpine:3.2 sleep 10m
46cc48421aa17e80733c73cc93ff6cc3567a25edcf123336111f930553b7c27a
n0ball@SDN-NFV:~/workspace$ docker inspect -f '{{.State.Pid}}' $(docker ps -aqf "name=sdnfv-demo")
1768219
n0ball@SDN-NFV:~/workspace$ sudo ln -s /proc/1768219/ns/net /var/run/netns/1768219
n0ball@SDN-NFV:~/workspace$ sudo ip netns exec 1768219 ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
1520: eth0@if1521: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:11:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.17.0.2/16 brd 172.17.255.255 scope global eth0
       valid_lft forever preferred_lft forever
n0ball@SDN-NFV:~/workspace$ sudo ip netns exec 1768219 ip link add eth-test type dummy
n0ball@SDN-NFV:~/workspace$ docker exec sdnfv-demo ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth-test: <BROADCAST,NOARP> mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 1a:e4:0a:9a:56:c7 brd ff:ff:ff:ff:ff:ff
1520: eth0@if1521: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:11:00:02 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.2/16 brd 172.17.255.255 scope global eth0
       valid_lft forever preferred_lft forever
```

Create a container named sdnfv-demo

Find the pid of the container

Make soft link so that ip netns can find container ns

Show interface information of ns

Create a dummy NIC using netns command

Show interface information of the container

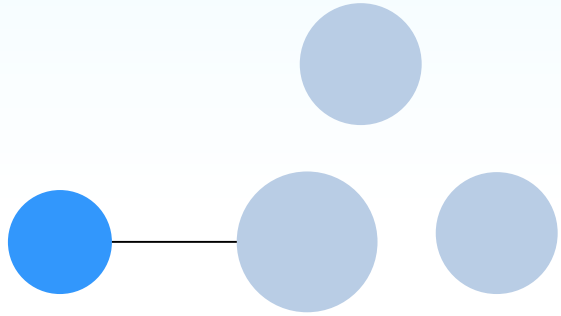
NIC is created inside the container

Ubuntu Net Tool Introduction

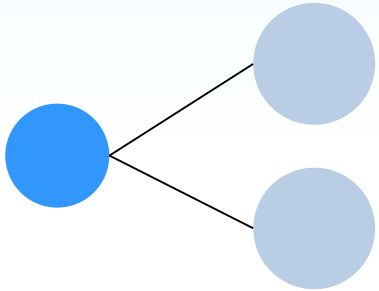
- `mtr` stands for My Traceroute (or sometimes Matt's Traceroute, after its creator).
- Traceroute is a **network diagnostic tool** used to track the path that packets take from your computer to a destination on the internet.
- Installation: `apt install mtr-tiny`
- What does mtr do?
  - `mtr` is like a live, ongoing version of traceroute that also tells you where slowdowns or dropped packets are happening along the path.
- Usage: `mtr <target ip>`

Autonomous System (AS)

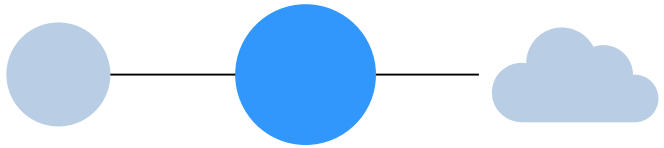
- AS Types



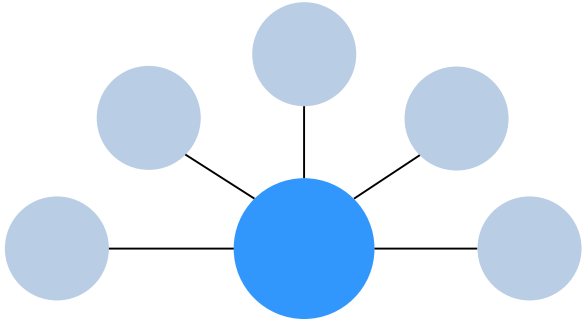
Stub



Multi Homed



Transit

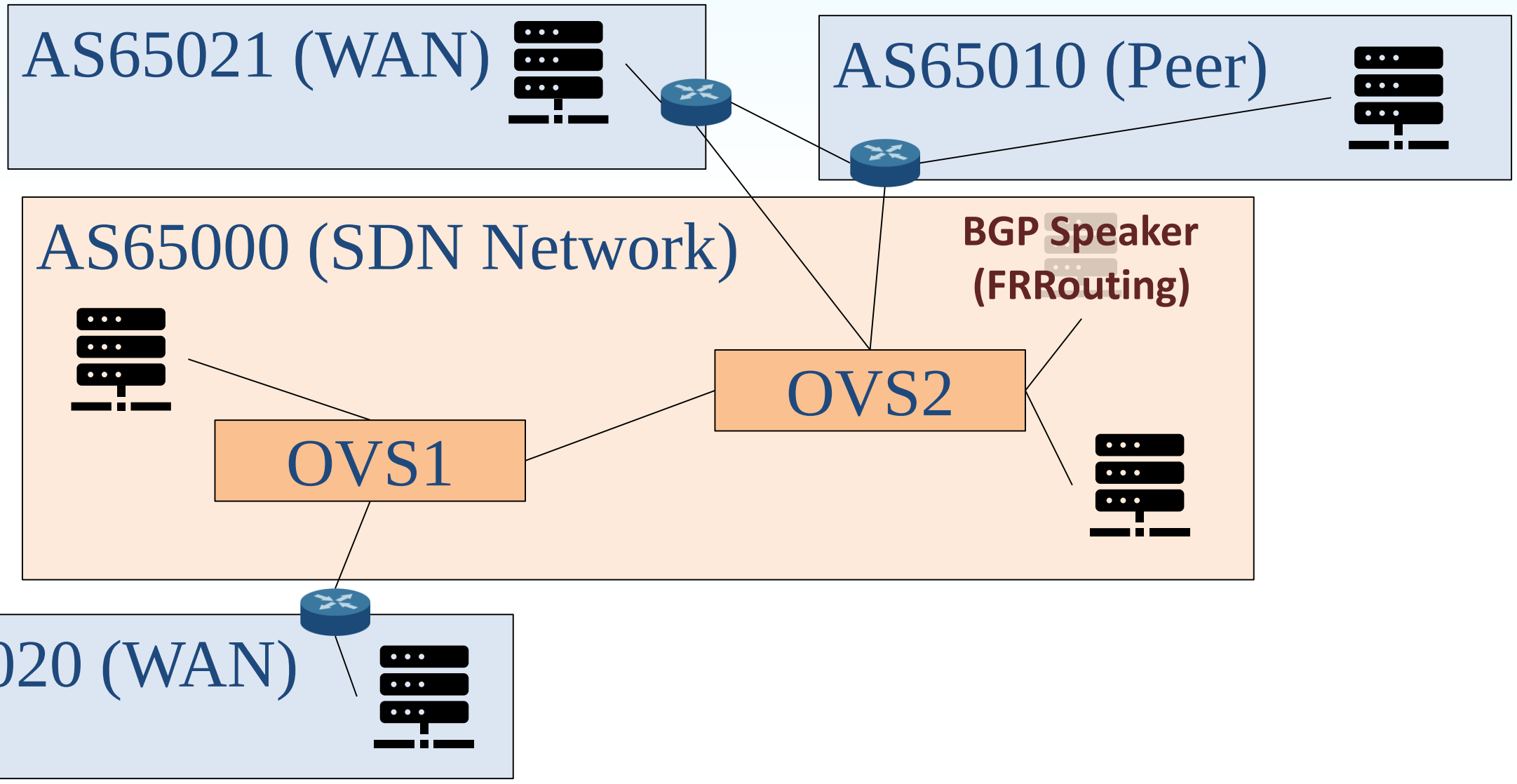


Internet Exchange Point

Topology

FRR Container

Web Container

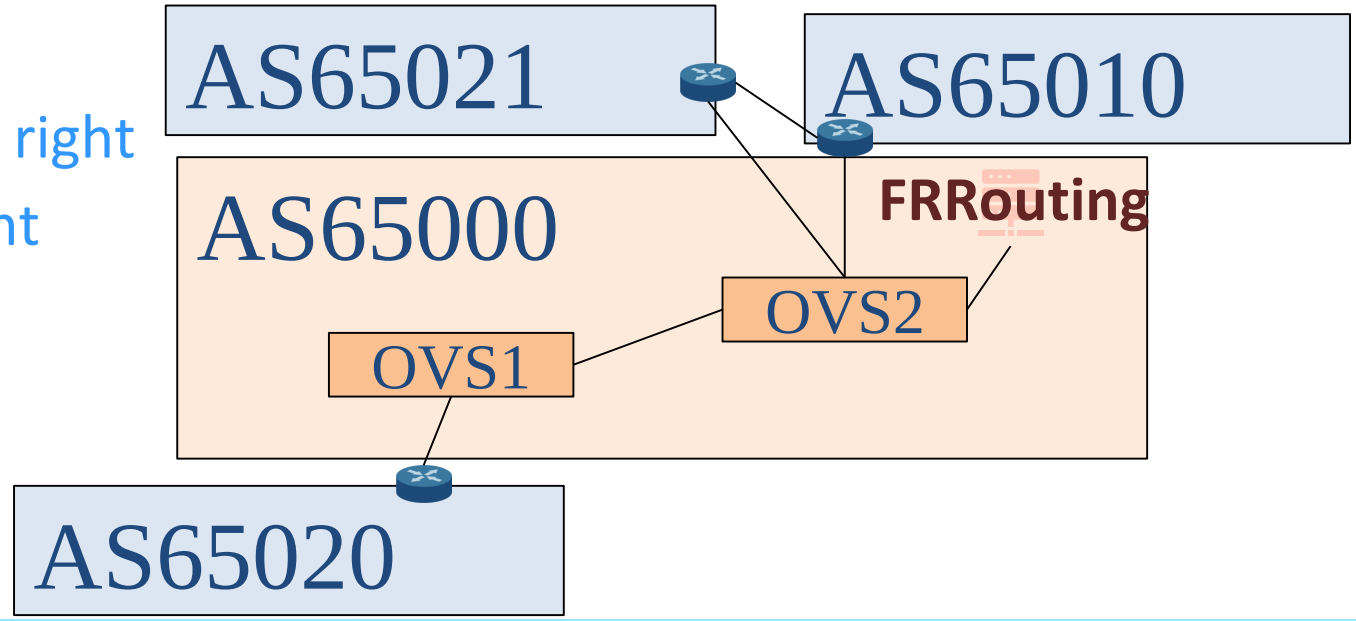


Configuration Requirements

- Networks
  - AS65000: 192.168.0.0/24
  - AS65010: 10.0.0.0/24
  - AS65020: 10.1.0.0/24
  - AS65021: 10.2.0.0/24
- AS65000 only advertise its own prefix (192.168.0.0/24) to its peers (AS65010)
- SDN Network will advertise prefixes received from WAN to WANs or peers
- BGP Speaker can set as gateway IP
- Web container IP in AS65000 must be the same
- Container images
  - FRR container: frrouting/frr-Debian
  - Web container: traefik/whoami

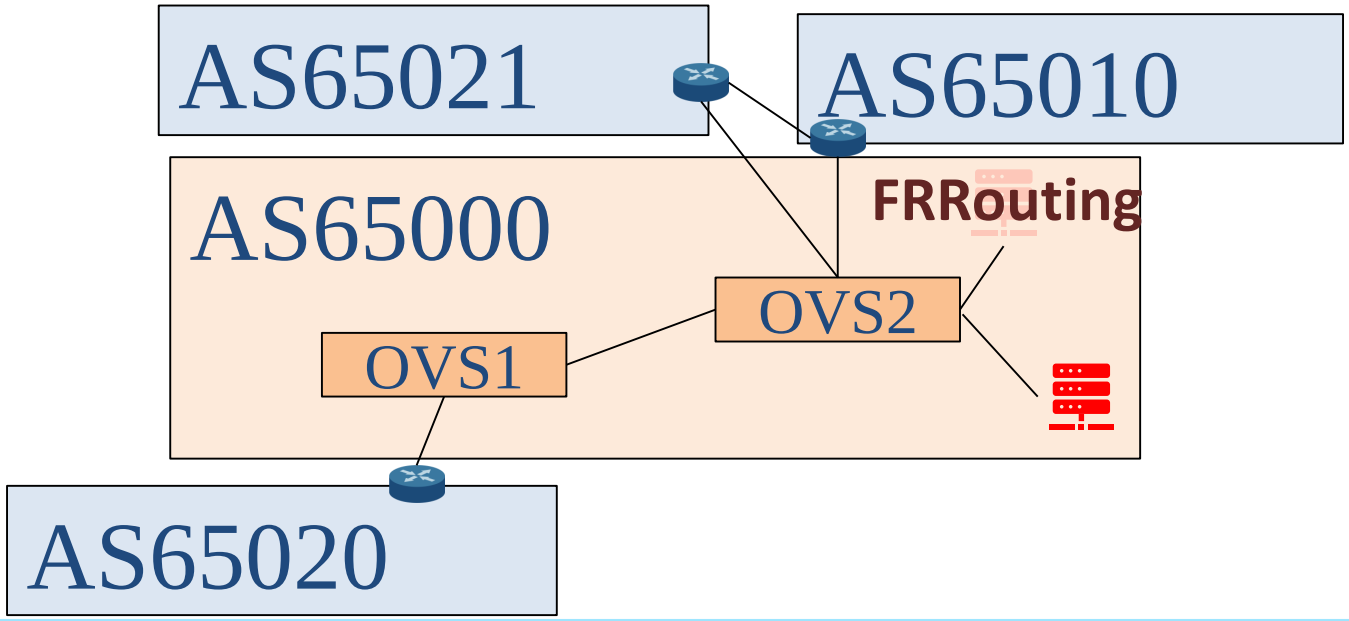
vRouter Verifications - Router Communication

1. Assure Router Communication
   - Each FRRouting can ping it's neighbors
     - For Non-SDN Network, directly connect -> easy
     - For SDN Network, not directly connect -> what to do?
   - Hint: What IP should you assign?
     - L2 traffic doesn't need routing, i.e. IP information
   - Expected results
     - `show routes` in FRR looks right
     - `routes` in ONOS looks right



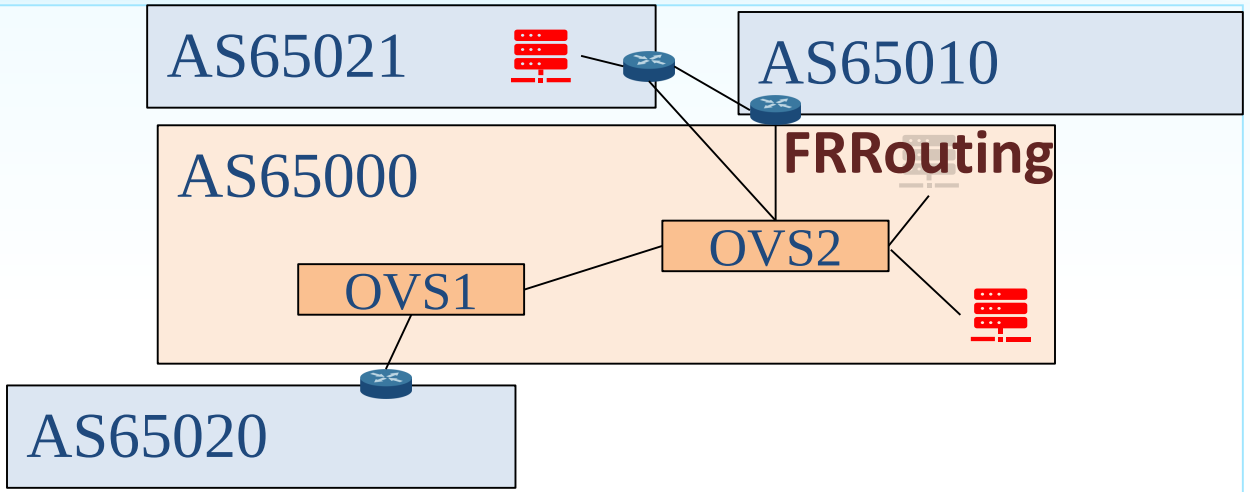
vRouter Verifications – Intra Domain Traffic

2. Assure Intra domain traffics
   - BGP speaker in AS65000 can ping Web server vice versa
     - There are multiple ways to achieve
     - IP should be within 192.168.0.0/24
   - traefik/whoami does not have a shell how to ping?
     - `ip netns exec …`



vRouter Verifications – Inter Domain Traffic

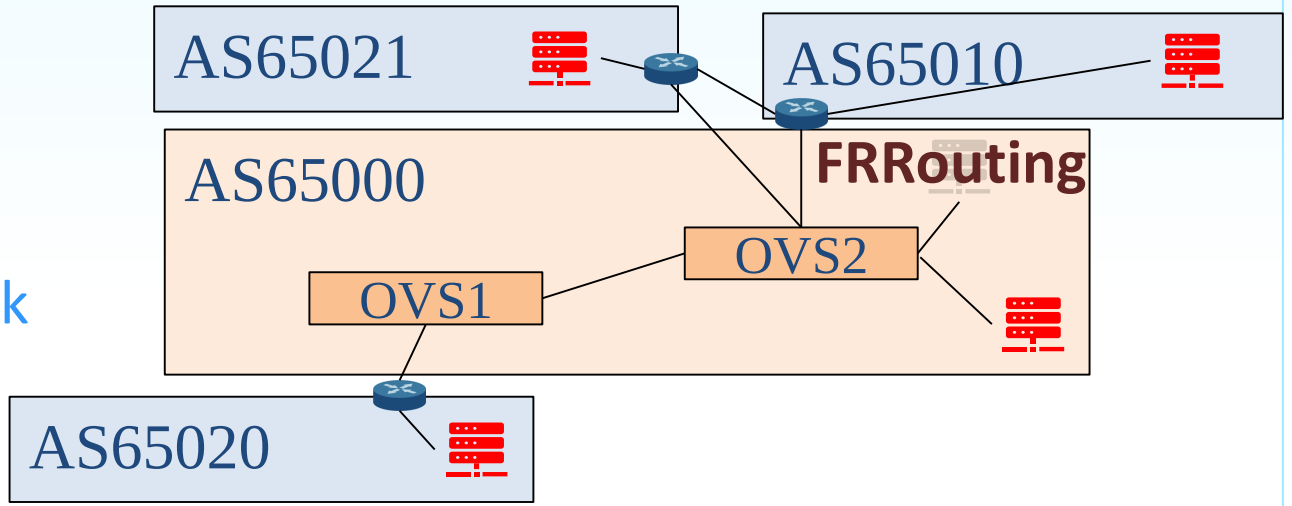
3. Assure Inter domain traffics
   - Must add flow rules to OVS
     - `routes` in ONOS
   - Hint
     - Gateway in none-SDN network
       - set via `ip route add…`
     - Gateway in SDN Network
       - any difference?
   - Expected results
     - AS65021 web container can ping AS65000 web container vice versa
     - Use mtr (traceroute tool) will see packets from
       - AS65021 FRRouting IP -> AS65000 Web Container
       - No AS65000 FRRouting IP showed in mtr



NOTE* If you are using docker compose, it will automatically assign a gateway for the container which is often not what you want, remember to `ip route delete …`

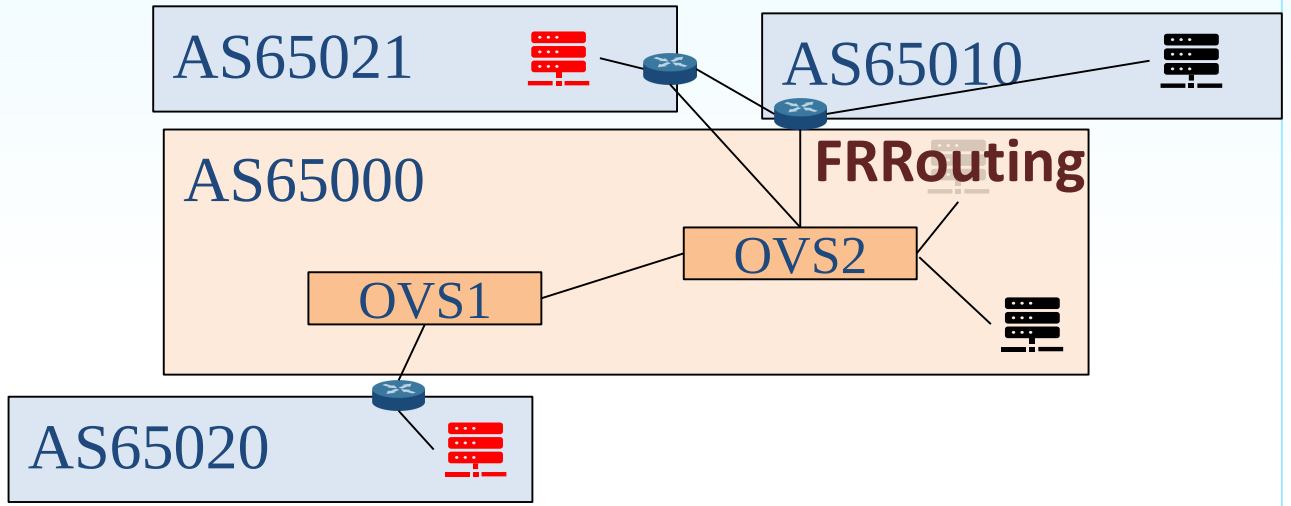
vRouter Verifications – Inter Domain Traffic (cont.)

3. Assure Inter domain traffics
   - More flow rules to OVS
   - Hint
     - Gateway in none-SDN network
       - set via `ip route add…`
     - Gateway in SDN Network
       - any difference?
   - Expected results
     - Web containers in AS650xx can ping AS65000 web container vice versa
     - Use mtr (traceroute tool) will see packets does not pass AS65000 BGP Speaker



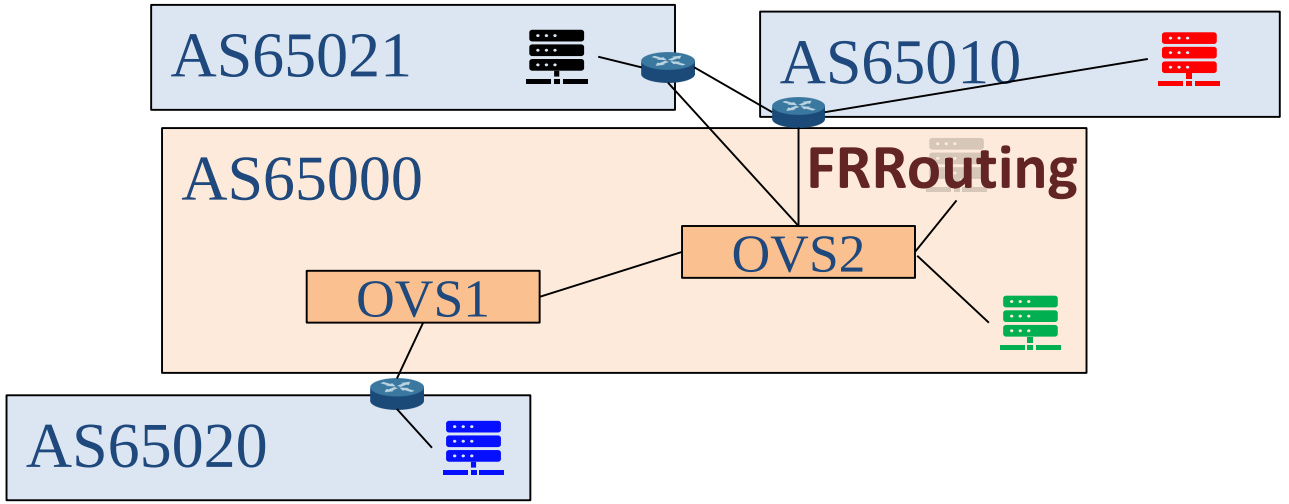
vRouter Verifications – Transit Traffic

4. Assure transit traffics
   - Flow rules to OVS
   - Hint
     - Think how routers works
     - Compare types of traffic
       - What are the same?
       - How to determine which type of traffic?
       - Can some traffics merge in to one flow rule?
   - Expected results
     - Web containers in AS65021 can ping AS65020 web container vice versa
     - Use mtr (traceroute tool) will see packets does not pass AS65000 BGP Speaker
       - AS65021 FRR <-> AS65020 FRR <-> AS65020 Web Container



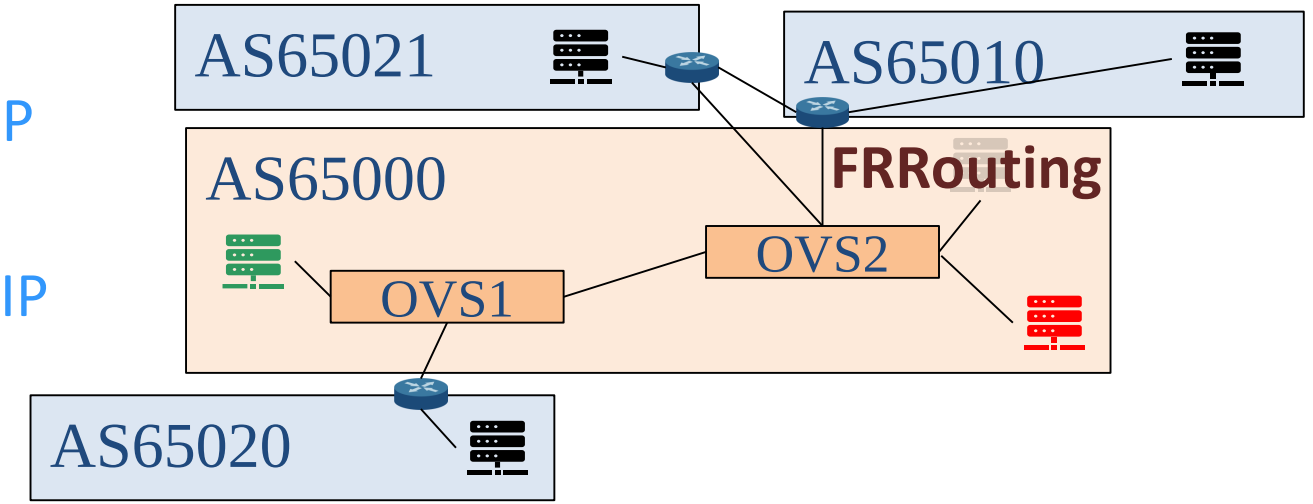
Requirement Verifications – Peers

- Requirement
  - AS65000 only advertise its own prefix (192.168.0.0/24) to its peers (AS65010)
    - AS65010 FRR will not receive AS65020's prefix from AS65000
  - Hint:
    - AS65010 FRR will receive AS65020's prefix from AS65021
    - Packet path from AS65010 Web container to AS65020 Web container?
    - Packet path from AS65010 Web container to AS65000 Web container?
  - Expect Results
    - mtr ?????



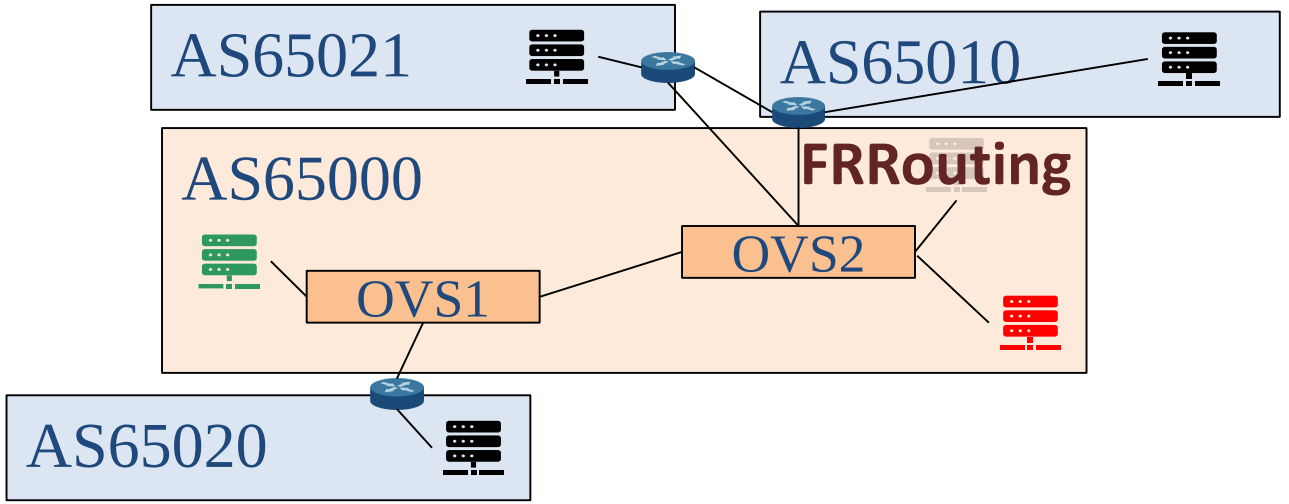
Requirement Verifications – Anycast Web Container

- Requirement
  - Web container IP in AS65000 must be the same
  - Why?
    - OVS1 and OVS2 might be in different physical zone (America vs Taiwan)
    - OVS1 and OVS2 have very high latency
    - AS65021 can get data from OVS2 Web Container
    - AS65020 can get data from OVS1 Web Container
  - Expected Result
    - AS65021 curl Web Container IP
      - Hostname: xxxxxxxx
    - AS65020 curl Web Container IP
      - Hostname: yyyyyyyy



Requirement Verifications – Anycast Hints

- Requirement
  - Web container IP in AS65000 must be the same
  - Hints:
    - After adding the new Web container, many things might be destroyed
    - Use wireshark/tcpdump to check one by one from Step 1. again
      - Why things get weird? How to fix?
      - Mostly, block packets that is not expected



TA Contacts

- If you have any problem
  - Mail to me
  - Register demo time for help

- Final Project (Start at 5/2)
  - Help Register (only offline help is available since everyone might have different environment settings)
    https://calendar.google.com/calendar/u/0/appointments/AcZssZ2sbtt-446xWK_xPxxIv22bY-FV947i-odtBV4=
  - Demo Register
    https://calendar.google.com/calendar/u/0/appointments/AcZssZ2sbtt-446xWK_xPxxIv22bY-FV947i-odtBV4=

TA Hints

- DEMO part
  - Requires network knowledge and SDN knowledge
  - Steps to solve a problem
    1. Think what is the expect result
    2. Think what shall be done to achieve the expect result
       - Separate tasks into small achievements
    3. Verify if the result met the expected result
  - Wireshark and tcpdump (if you are familiar with Linux cli) is good
  - Make sure you know MAC and IP value
    - Enters/Leaves the router
    - How it is decided
- **Flow priority is important remember to check**
- **Docker compose will auto assign a gateway, this might not be what you want!!!!**

OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Scoring Criteria
- Reference

Deployment Requirements

- **Only openflow (and route service) related apps can be use**
  - **Reactive forwarding (org.onosproject.fwd) is a no no!!!**

```
*   8 org.onosproject.drivers              2.7.1.SNAPSHOT Default Drivers
*  15 org.onosproject.fpm                  2.7.1.SNAPSHOT FIB Push Manager (FPM) Route Receiver
*  21 org.onosproject.gui2                 2.7.1.SNAPSHOT ONOS GUI2
*  36 org.onosproject.hostprovider         2.7.1.SNAPSHOT Host Location Provider
* 100 org.onosproject.lldpprovider         2.7.1.SNAPSHOT LLDP Link Provider
* 102 org.onosproject.openflow             2.7.1.SNAPSHOT OpenFlow Provider Suite
* 101 org.onosproject.openflow-base        2.7.1.SNAPSHOT OpenFlow Base Provider
*   7 org.onosproject.optical-model        2.7.1.SNAPSHOT Optical Network Model
*  14 org.onosproject.route-service        2.7.1.SNAPSHOT Route Service Server
```

- **If not, you will be scored 0**

Scores

- Code (60 points)
  - Intra-domain traffic (from both AS)
    - IPv4 **(20 points)**
  - Inter-domain traffic (from both AS)
    - IPv4 **(20 points)**
  - Transit traffic
    - IPv4 **(15 points)**
  - Routes in ONOS `routes` and FRR `show bgp routes` correct
    - IPv4 **(5 points)**
- Demo (40 points)
  - Peers traffic (**10 points** explanation + **10 points** verification)
  - Anycast traffic (**10 points** explanation + **10 points** verification)

OUTLINE

- Review of Labs
- Virtual Router Explained
- Virtual Router Specification
- ONOS App and Services in Use
- In Used App Configurations
- Virtual Router Workflow
- Scoring Criteria
- Reference

Reference

- ovs-vsctl(8) - Linux manual page (https://man7.org/linux/man-pages/man8/ovs-vsctl.8.html)
- mtr man | Linux Command Library (https://linuxcommandlibrary.com/man/mtr)
- ip(8) - Linux man page (https://linux.die.net/man/8/ip)