

Open vSwitch on z Systems

Open vSwitch support for OSA Express card

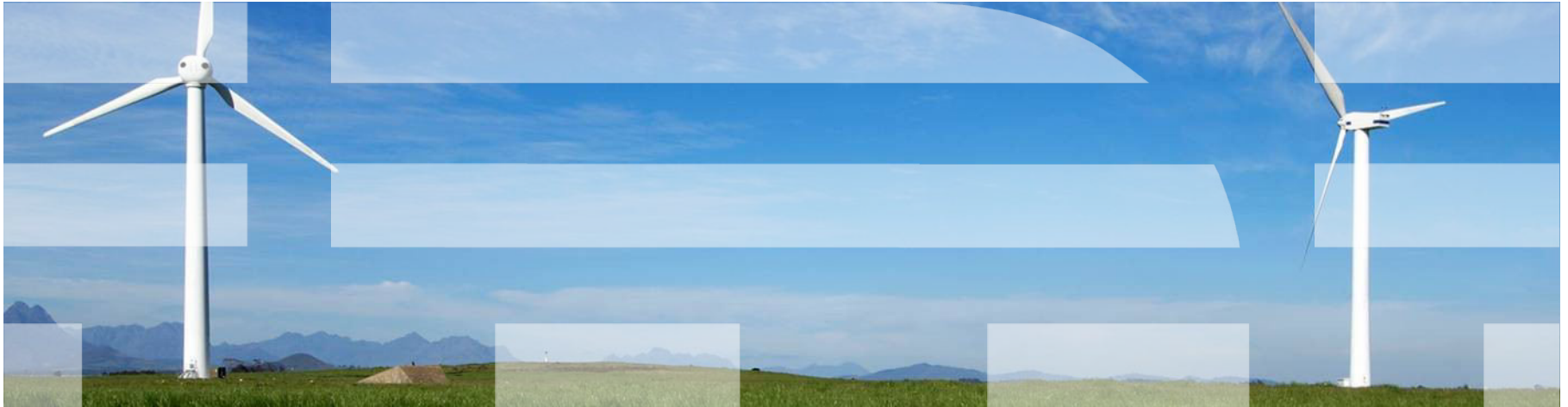


Table of Contents

- What is Open vSwitch?
- Why it's a good idea to use it?
- Open vSwitch components
- Where is OVS in a system?
- Bridgeport functionality: overall description
- Bridgeport vs. Promiscuous
- How to configure OVS
- Use cases

What is Open vSwitch?



Open vSwitch -

Production quality

Multilayer

Virtual Switch.

© openvswitch.org

What is Open vSwitch?

Open vSwitch is a production quality, multilayer virtual switch.

What does it mean?

What is Open vSwitch?

Open vSwitch is a production quality, multilayer **virtual switch**.

What does it mean?

Virtual switch: A full-sized virtual alternative to a real switch.

Virtual network environment can be equal to a real one.

What is Open vSwitch?

Open vSwitch is a production quality, **multilayer** virtual switch.

What does it mean?

Virtual switch: A full-sized virtual alternative to a real switch.

Virtual network environment can be equal to a real one.

Multilayer: OVS operates L2 as well as L3 and L4.

Can use IP address along with MAC address to direct packets.

What is Open vSwitch?

Open vSwitch is a **production quality**, multilayer virtual switch.

What does it mean?

Virtual switch: A full-sized virtual alternative to a real switch.

Virtual network environment can be equal to a real one.

Multilayer: OVS operates L2 as well as L3 and L4.

Can use IP address along with MAC address to direct packets.

Production quality: OVS is used in multiple products and runs in many large production environments. E.g. OVS is a popular plugin, used by OpenStack for its' Neutron networking service.

Why it's a good idea to use it?

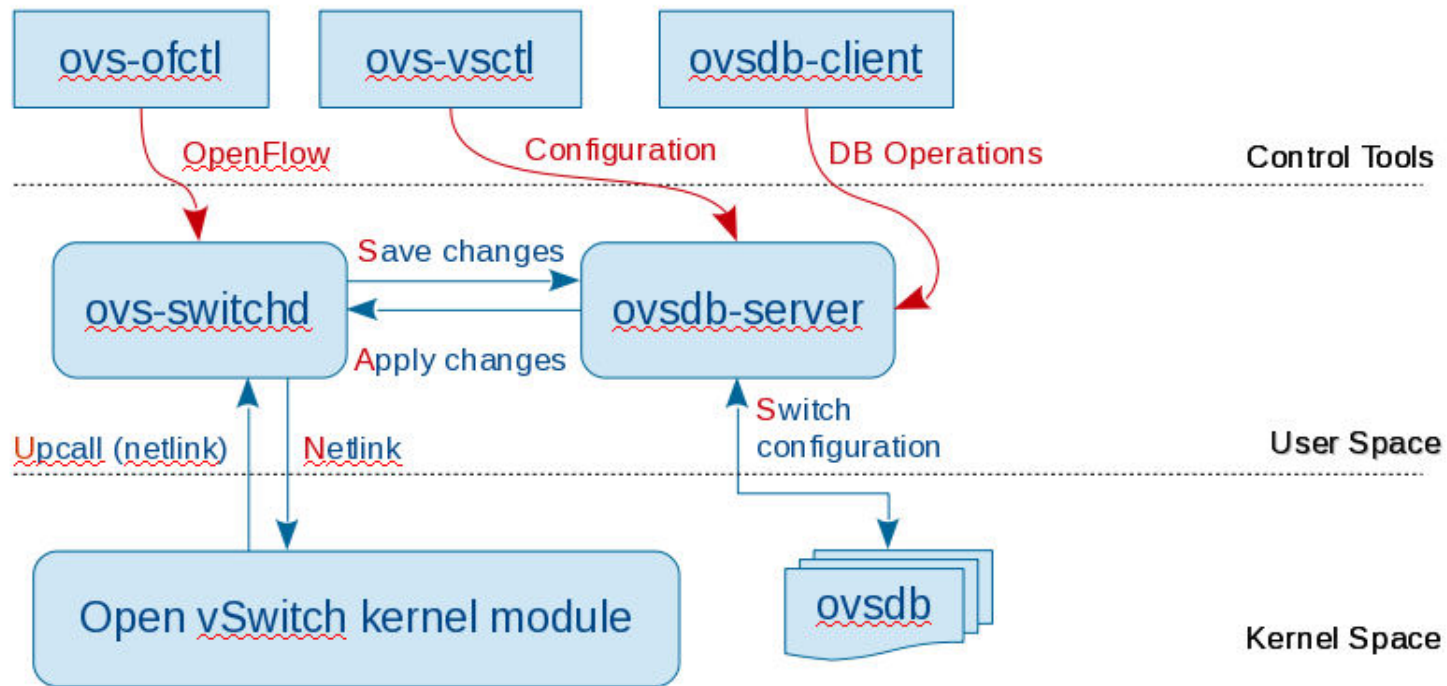
To build a virtual network environment a virtual switch should provide the same options as a real one.

With OVS you may:



- Divide existing network into several VLANs, applying individual rules to each of them.
- Create GRE or VXLAN tunnels between different networks.
- Bind several real interfaces into one to improve performance and fault-tolerance.
- Create OpenFlow rules to control the traffic.
- Support VM migration.

Open vSwitch components



Open vSwitch components (2)

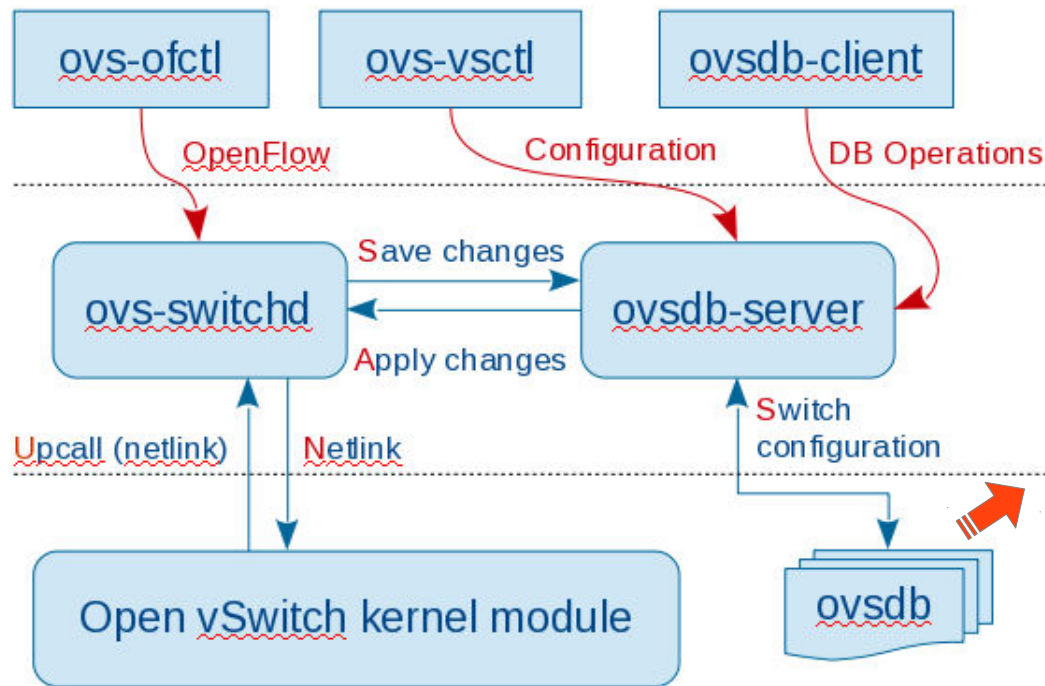
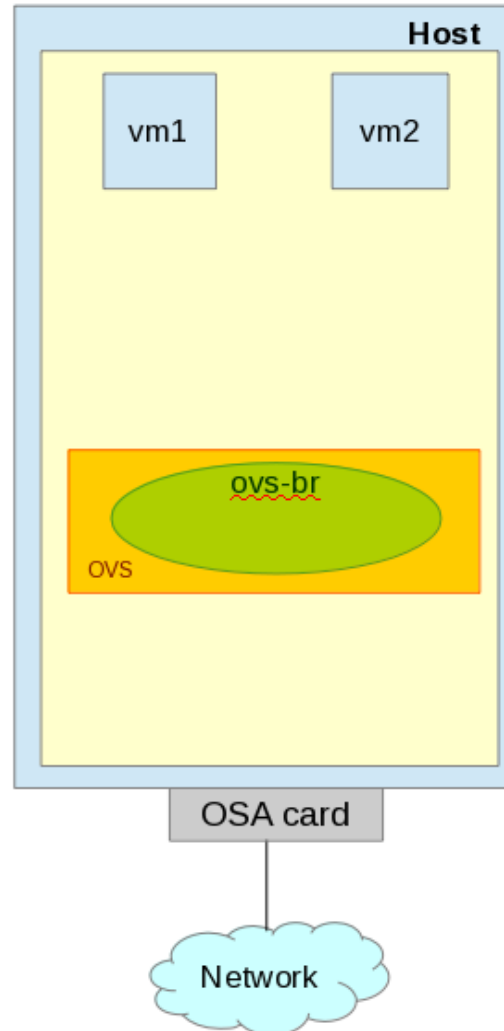


TABLE SUMMARY

Open_vSwitch - Switch configuration.
Bridge - Bridge configuration.
Port - Port configuration.
Interface - One network device in a Port.
QoS - Quality of Service configuration
Queue - QoS output queue.
Mirror - Port mirroring.
Controller - OpenFlow controller configuration.
Manager OVSDb management connection.
NetFlow NetFlow configuration.
SSL SSL configuration.
sFlow sFlow configuration.
Capability Capability configuration.

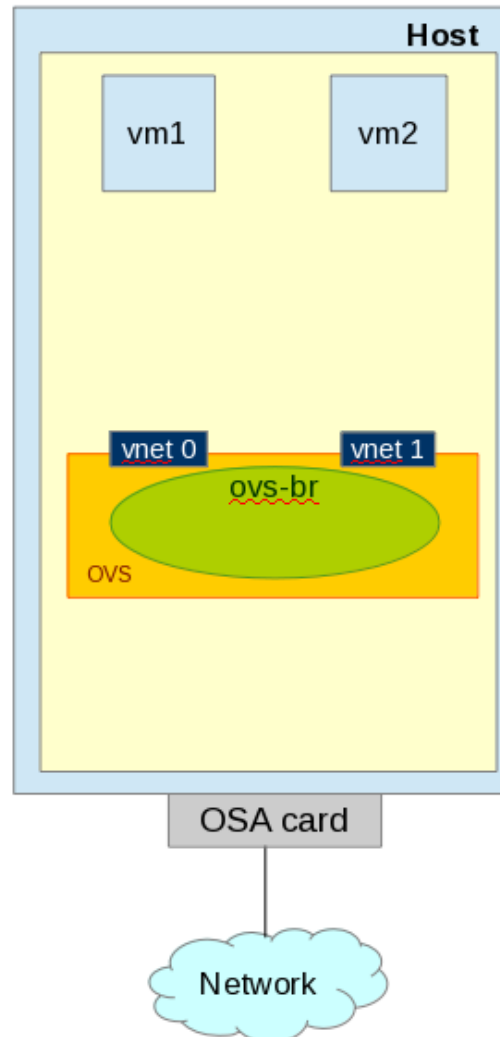
Where is OVS in a system?

An OVS bridge, created on a host, serves as a focal point for inter-guest communication and guest outside connection.



Where is OVS in a system?

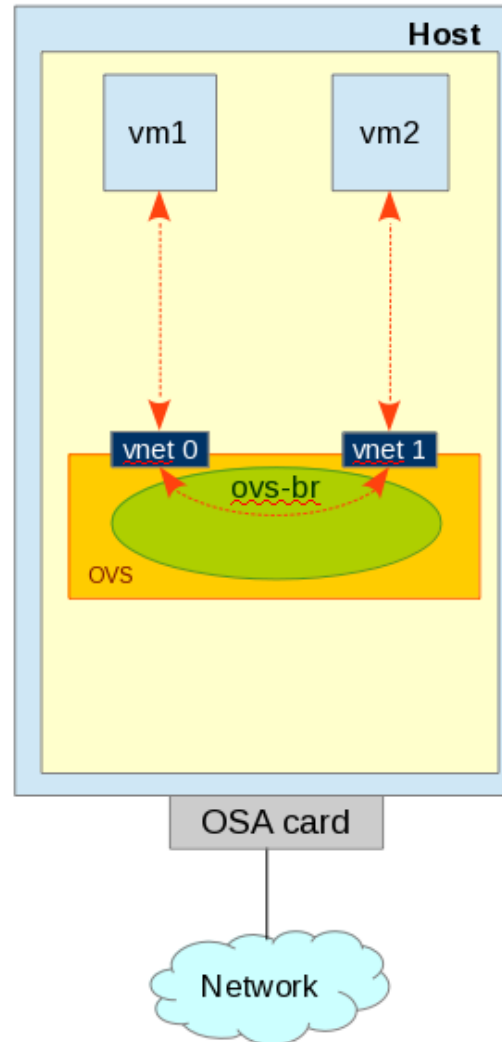
A VM can be attached to a bridge directly or via a libvirt virtual network.



A virtual port (vnet 0) is automatically created for each VM attached to the bridge.

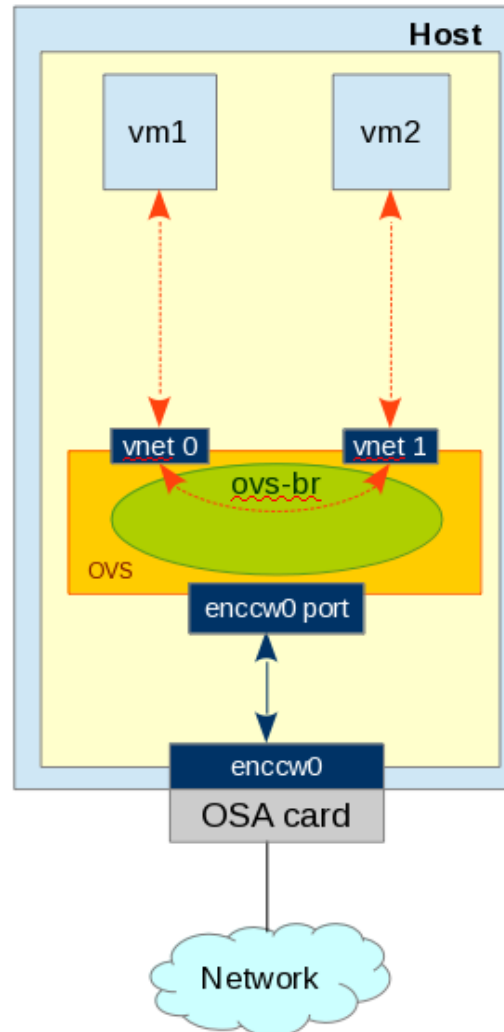
Where is OVS in a system?

The bridge can be used for inter-guest communication...



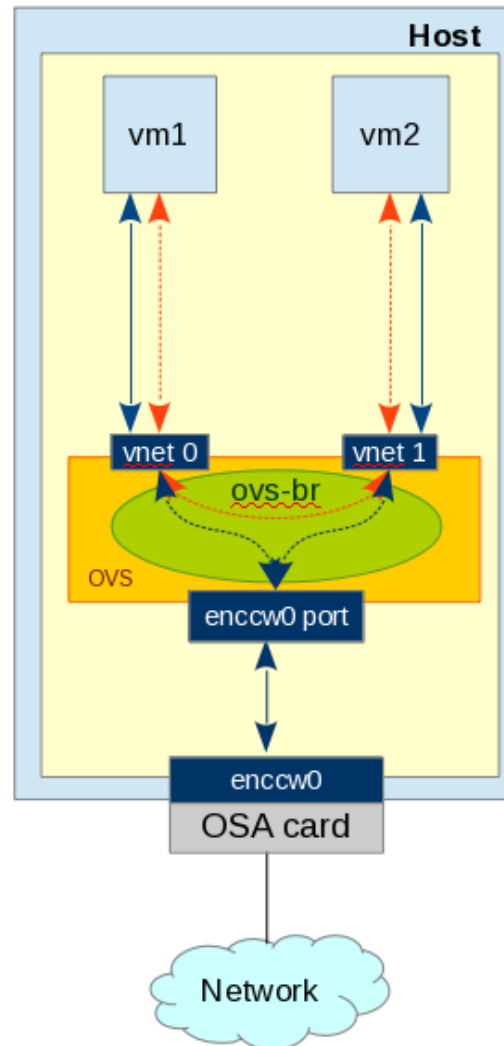
Where is OVS in a system?

...or an OSA card interface can be attached as a port to the bridge...



Where is OVS in a system?

... to provide an outside connection for the guests.



Note: An OSA card, attached to a bridge can no longer be accessed directly.

Make sure you use a separate management network.

Note2: To provide outside connection for guests OSA card interface should be in BRIDGEPORT mode. (check the next slide)

Bridgeport functionality: overall description



BRIDGEPORT mode of an OSA card is a z Systems alternative to promiscuous mode on other Ethernet adapters.



OSA interface in BRIDGEPORT mode is able to intercept and forward traffic with unknown destination (e.g. traffic, destined to a guest)

USAGE:

- BRIDGEPORT mode is required to provide outside connection for the guests on host, configured in FLAT or VLAN mode.
- If you use interface bonding you should have BRIDGEPORT mode enabled on all bonded interfaces.
- Tunneling does **NOT** require BRIDGEPORT mode for the interface as the traffic, destined to a remote guest is encapsulated.

Bridgeport functionality: overall description

Bridgeport-capable OSA device operating in Layer 2 mode gets the following key sysfs attributes:

bridge_reflect_promisc

bridge_role

bridge_state

The minimum OSA firmware level supporting BRIDGEPORT is:

z Systems	Bundle	OSA Firmware level
EC12/BC12	D15F45a	C9a
Z13	D22HS19a	715

Bridgeport functionality: overall description

bridge_reflect_promisc (read-write)

Description:

The attribute determines bridge role change on setting and resetting of IFF_PROMISC flag (when the interface enters\leaves promiscuous mode).

Possible values:

none (default): no impact on bridge_role on setting/unsetting the IFF_PROMISC flag

primary: sets bridge_role=primary on setting the IFF_PROMISC flag

secondary: sets bridge_role=secondary on setting the IFF_PROMISC flag

Check value:

```
cat /sys/devices/qeth/0.0.XXXX/bridge_reflect_promisc
```

where "0.0.XXXX" is a device number for an OSA card used

Bridgeport functionality: overall description

bridge_role (read-write)

Description:

The attribute reflects current bridge_role. It can be changed dynamically according to bridge_reflect_promisc value or by setting the value directly.

Possible values:

none (default): no role assigned

primary: bridgeport interface can intercept and forward traffic with unknown destination

secondary: one of secondary bridgeport interfaces become active when “primary” goes OFF

There can be up to 4 secondary OSA interfaces connected to different hosts.

Check value:

```
cat /sys/devices/qeth/0.0.XXXX/bridge_role
```

Bridgeport functionality: overall description

bridge_state (read-only)

Description:

The attribute reflects current bridgeport state for the interface. It is changed automatically.

Possible values:

inactive (default): bridgeport mode is not active on the interface.

active: bridgeport mode is active and OSA interface is able to intercept and forward traffic for unknown destinations. Is used with bridge_role = primary/secondary

standby: bridgeport interface is waiting to become active. Is used for bridge_role=secondary when there is some other active port.

Check value:

```
cat /sys/devices/qeth/0.0.XXXX/bridge_state
```

Bridgeport vs Promiscuous

Turning an interface connected to a port of a real Ethernet switch to promiscuous mode is similar to turning an OSA port to BRIDGEPORT mode.

However, there are a couple of important differences:

Real switch with interface in promiscuous mode	OSA with active Bridgeport interface
Any number of interfaces can be turned to promiscuous mode and receive traffic with unknown destination	But only one interface can have "active" status and receive traffic to unknown destination. For OSA with several Ports each port can have one "active" interface
A real switch allows a corresponding host with promiscuous interface to receive all traffic passing through the switch, with known and unknown destinations alike	Interface cannot intercept traffic destined to systems connected to other interfaces of the same OSA device.

How to configure Open vSwitch (Create a switch)

STEP 1: Run openvswitch:

```
systemctl start openvswitch
```


STEP 2: Create a bridge:

```
ovs-vsctl add-br ovsbr0
```

STEP 3: Make sure the bridge is created:

```
ovs-vsctl show
```

```
cac08a10-d7c7-421d-bc9b-4b994ae7e0cb  
    Bridge "ovsbr0"  
        Port "ovsbr0"  
            Interface "ovsbr0"  
                type: internal  
            ovs_version: "2.3.0"
```



You get

Note: A port and an internal interface are created automatically with a new bridge

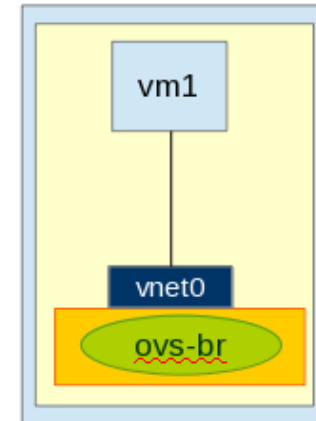
How to configure Open vSwitch (Connect a guest)

STEP 4: Connect a guest to the created ovs bridge

OPTION 1: Add bridge directly to a guest xml configuration

virsh edit VM1

```
<interface type='bridge'>
  <mac address='XX:XX:XX:XX:XX:XX'/>
  <source bridge='ovsbr0'/>
  <virtualport type='openvswitch'/>
</interface>
```



OPTION 2: Create a virtual network with a link to the ovs bridge and then add the network to a guest xml

virsh net-define /path_to_xml/ovs-net

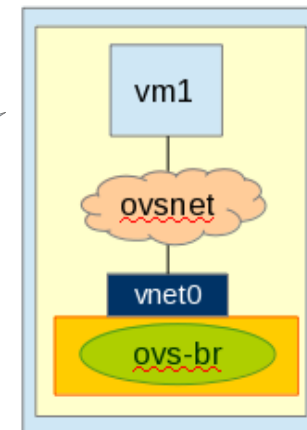
```
<network>
  <name>ovsnet</name>
  <forward mode='bridge'/>
  <bridge name='ovsbr0'/>
  <virtualport type='openvswitch'/>
</network>
```

Then attach the network to the guest:

virsh edit VM1

```
<interface type='network'>
  <mac address='XX:XX:XX:XX:XX:XX'/>
  <source network='ovsnet'/>
  <model type='virtio'/>
</interface>
```

This option gives you an advantage of an automated network configuration, e.g. NAT, DHCP, etc.



How to configure Open vSwitch (Start a guest)

STEP 5: Start (restart) guests:

```
virsh start VM1
```

STEP 6: Check that a virtual port appeared in ovs bridge configuration:

```
ovs-vsctl show
```

```
cac08a10-d7c7-421d-bc9b-4b994ae7e0cb
```

```
Bridge "ovsbr0"
```

```
Port "ovsbr0"
```

```
Interface "ovsbr0"
```

```
type: internal
```

```
Port "vnet0"
```

```
Interface "vnet0"
```

```
ovs_version: "2.3.0"
```



You get

! Now you may attach another guest to the switch assign IP address to each guest and check their connection.

How to configure Open vSwitch (Prepare an OSA interface 2)

STEP 7. To provide outside connection for guests OSA card interface should be in BRIDGEPORT mode. You have two options to do that:

OPTION 1

A static way, the role is assigned immediately:

1. Set bridge_role = primary (secondary)

- when you add an OSA interface

```
znetconf -a 0.0.XXXX -o layer2=1 -o bridge_role=primary
```

- any moment by writing directly into the attribute

```
echo primary > /sys/devices/qeth/0.0.XXXX/bridge_role
```

The attribute value can only be set if bridge_reflect_promisc=none.

2. Check bridge_role and bridge_state

```
cat /sys/devices/qeth/0.0.1200/bridge_role  
primary
```

```
cat /sys/devices/qeth/0.0.1200/bridge_state  
active
```

NOTE!

This may not work:

IF THERE IS ANOTHER PRIMARY ACTIVE
interface of the same OSA
BRIDGE ROLE WILL STAY NONE.

How to configure Open vSwitch (Prepare an OSA interface)

OPTION 2

A dynamic way, bridge role and state change automatically when OFF_PROMISC flag is set:

1. Set bridge_reflect_promisc = primary (secondary)
 - when you add an OSA interface

```
znetconf -a 0.0.XXXX -o layer2=1 -o bridge_reflect_promisc=primary
```

- any moment by writing directly into the attribute

```
echo primary > /sys/devices/qeth/0.0.XXXX/bridge_reflect_promisc
```

The attribute value can only be set if bridge_role=none.

2. Turn promiscuous mode ON

```
ip link set dev enccw0 promisc on
```

3. Check bridge_role and bridge_state

```
cat /sys/devices/qeth/0.0.1200/bridge_role  
primary
```

```
cat /sys/devices/qeth/0.0.1200/bridge_state  
active
```

NOTE!

This may not work:

IF THERE IS ANOTHER PRIMARY ACTIVE
interface of the same OSA
BRIDGE ROLE WILL NOT CHANGE
and you will get no error message.

How to configure Open vSwitch (Attach an OSA interface)


STEP 8: Attach encw0 interface to the bridge as a port:

```
ovs-vsctl add-port ovsbr0 encw0
```

STEP 9: Check that a new port appeared in the ovs bridge configuration:

```
ovs-vsctl show
```

```
cac08a10-d7c7-421d-bc9b-4b994ae7e0cb
Bridge "ovsbr0"
  Port "ovsbr0"
    Interface "ovsbr0"
      type: internal
  Port "vnet0"
    Interface "vnet0"
  Port "encw0"
    Interface "encw0"
ovs_version: "2.3.0"
```



You get

Done! Now your guests have a connection to the outside world.

How to configure Open vSwitch (Persistent configuration)

STEP 10 (optional): To create a persistent configuration or apply additional parameters you may create an ifcfg file for the ovs bridge:

```
>> /etc/sysconfig/network-scripts/ifcfg-ovsbr0
```

For static IP:

```
DEVICE="ovsbr0"  
ONBOOT="yes"  
DEVICETYPE="ovs"  
TYPE="OVSBridge"  
OVSBOOTPROTO="static"  
IPADDR=192.168.10.1  
NETMASK=255.255.255.0
```

For DHCP service:

```
DEVICE="ovsbr0"  
ONBOOT="yes"  
DEVICETYPE="ovs"  
TYPE="OVSBridge"  
OVSBOOTPROTO="dhcp"  
OVS DHCPINTERFACES=enccw0.0.1200 #only if dhcp is used.
```

You may assign an IP address to your bridge, if needed. E.g. to provide guest-to-host connection.

You may take an IP address from the OSA card interface, attached to the switch.

In this case make sure you remove the IP from the OSA interface.

How to configure Open vSwitch (Persistent configuration 2)

STEP 10 (optional): You also need to update enc cw0.0.XXXX interface configuration file to use ovsbr0:

```
>> /etc/sysconfig/network-scripts/ifcfg-enc cw0.0.XXXX
```

```
DEVICE="enc cw0.0.XXXX"  
NAME="enc cw0.0.XXXX"  
ONBOOT="yes"  
DEVICETYPE="ovs"  
TYPE="OVSPort"  
BOOTPROTO="none"  
OVS_BRIDGE="ovsbr0"  
NETTYPE="qeth"  
SUBCHANNELS="0.0.XXX0,0.0.XXX1,0.0.XXX2"  
OPTIONS="layer2=1 bridge_role=primary"  
OPTIONS="layer2=1 bridge_reflect_promisc=primary"
```

You may use either **blue** or **red** configuration. For the **red** option you'll need to set promiscuous mode for an interface.

Where "enc cw0.0.XXXX" is a full name of your OSA interface,

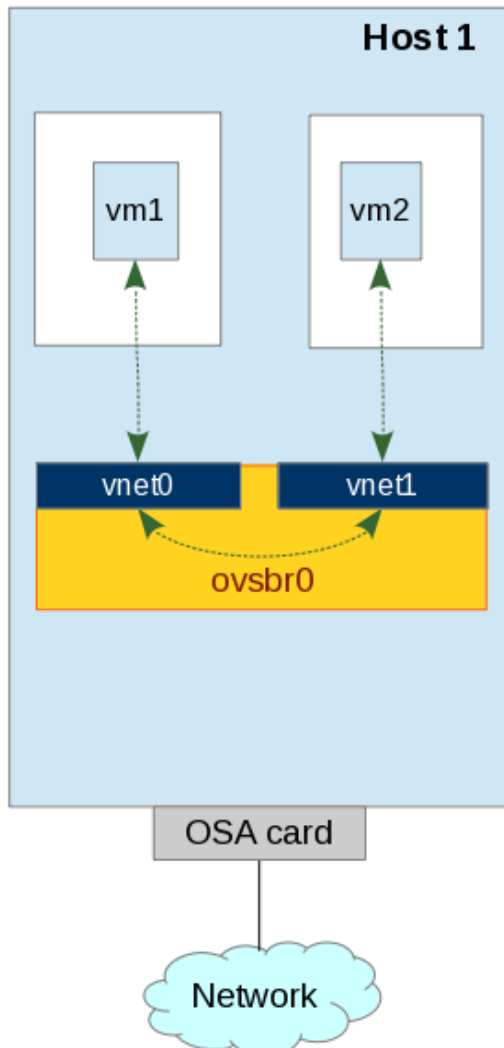
"ovsbr0" is a name of your ovs bridge

"0.0.XXX0, 0.0.XXX1, 0.0.XXX2" are OSA devices numbers attached to the LPAR

STEP 11 (optional): Restart the network service to apply the changes:

```
systemctl restart network.service
```

Use cases (VLANs)

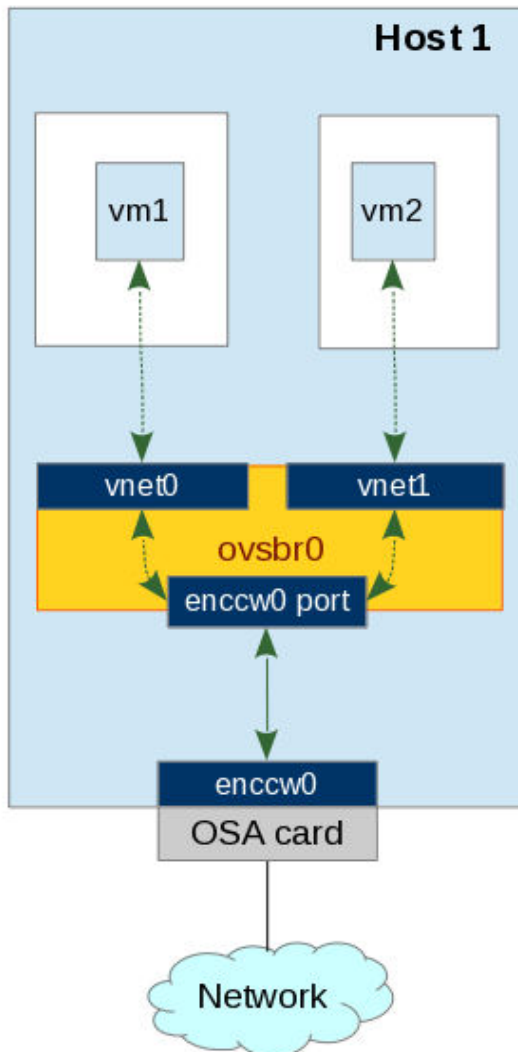


OVS supports guest traffic isolation using VLANs.

STEP 1: Setup a bridge and configure guests to be connected to the bridge:

```
ovs-vsctl add-br ovsbr0
```

Use cases (VLANs)



OVS supports guest traffic isolation using VLANs.

STEP 1: Setup a bridge and configure guests to be connected to the bridge:

```
ovs-vsctl add-br ovsbr0
```

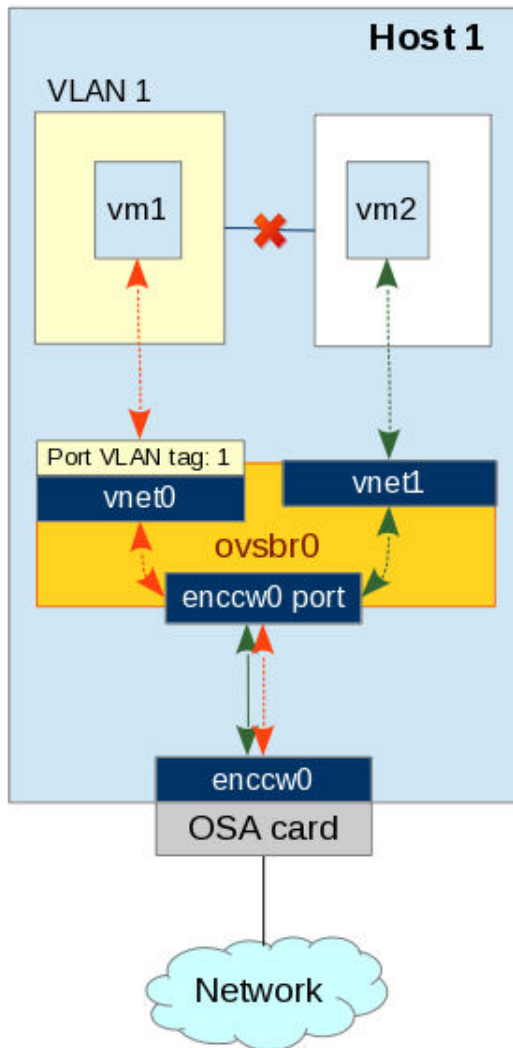
STEP 2: Plug enccw0 interface into the bridge as a VLAN trunk:

```
ovs-vsctl add-port ovsbr0 enccw0
```

Note: An OSA card, attached to a bridge can no longer be accessed directly.

Make sure you use a separate management network.

Use cases (VLANs)



OVS supports guest traffic isolation using VLANs.

STEP 1: Setup a bridge and configure guests to be connected to the bridge:

```
ovs-vsctl add-br ovsbr0
```

STEP 2: Plug enccw0 interface into the bridge as a VLAN trunk:

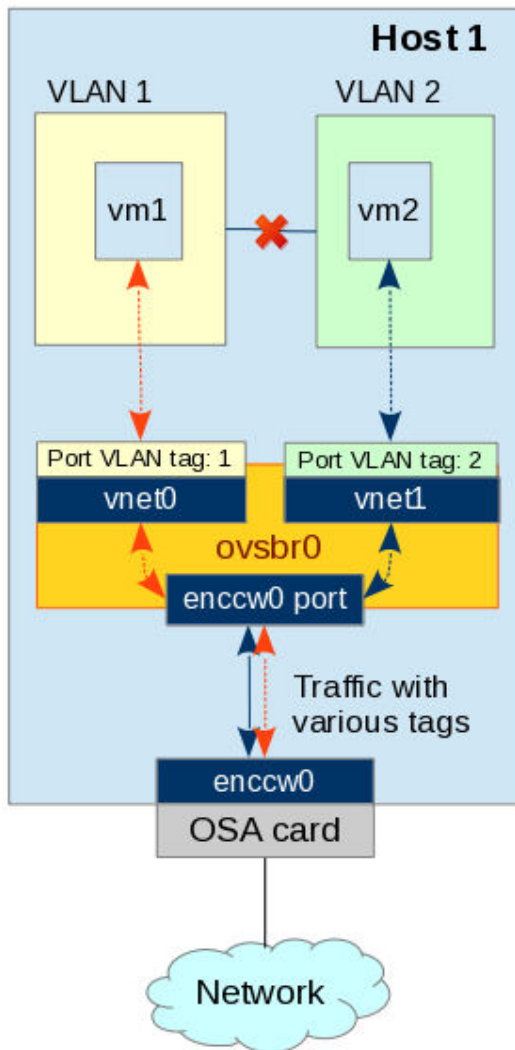
```
ovs-vsctl add-port ovsbr0 enccw0
```

STEP 3: Add vm1 to VLAN 1:

```
ovs-vsctl add-port ovsbr0 vnet0 tag=1
```

where vnet* is the port to which the guest is connected.

Use cases (VLANs)



OVS supports guest traffic isolation using VLANs.

STEP 1: Setup a bridge and configure guests to be connected to the bridge:

```
ovs-vsctl add-br ovsbr0
```

STEP 2: Plug enccw0 interface into the bridge as a VLAN trunk:

```
ovs-vsctl add-port ovsbr0 enccw0
```

STEP 3: Add vm1 to VLAN 1:

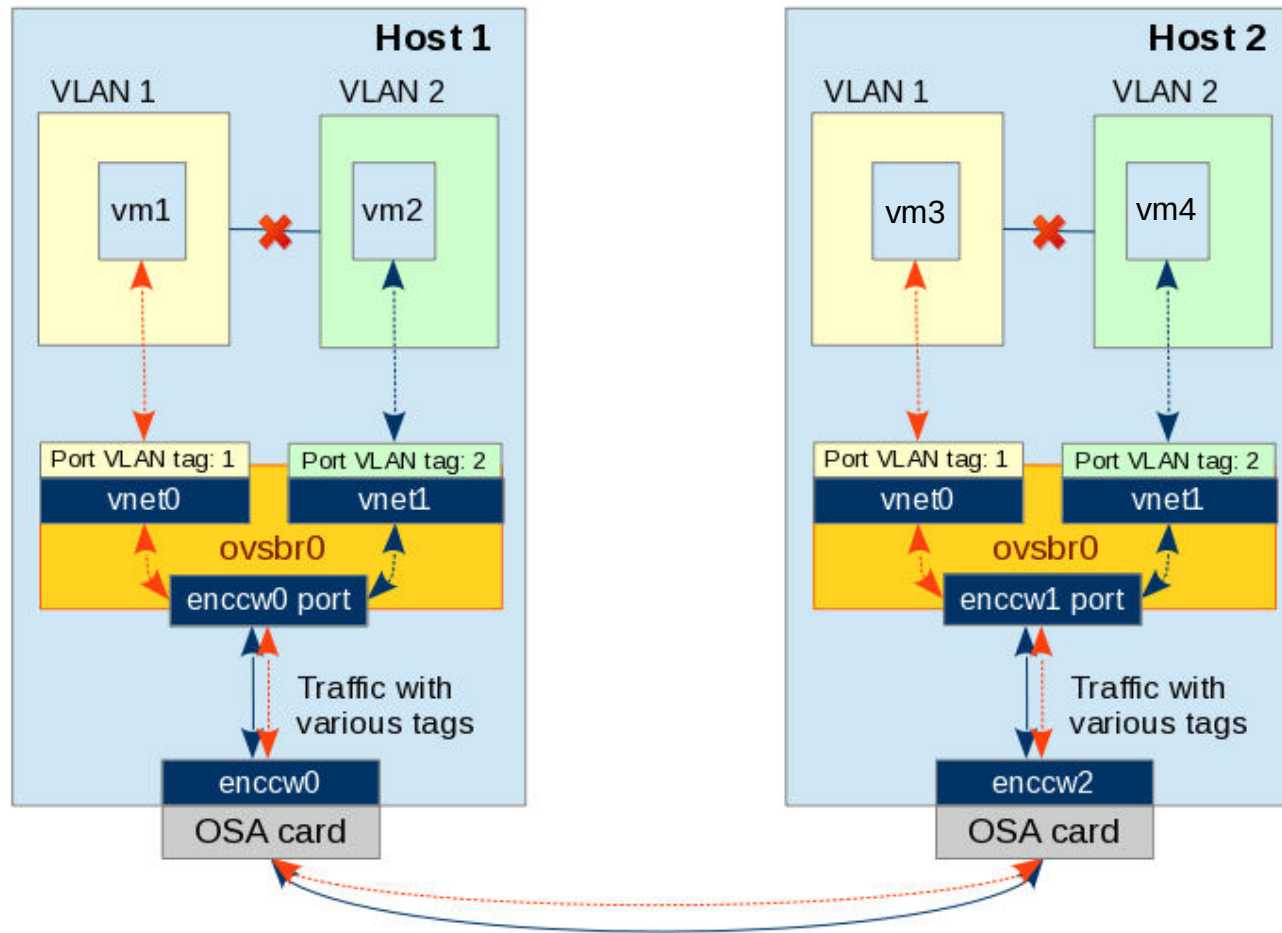
```
ovs-vsctl add-port ovsbr0 vnet0 tag=1
```

STEP 4: Add vm2 to VLAN 2:

```
ovs-vsctl add-port ovsbr0 vnet1 tag=2
```

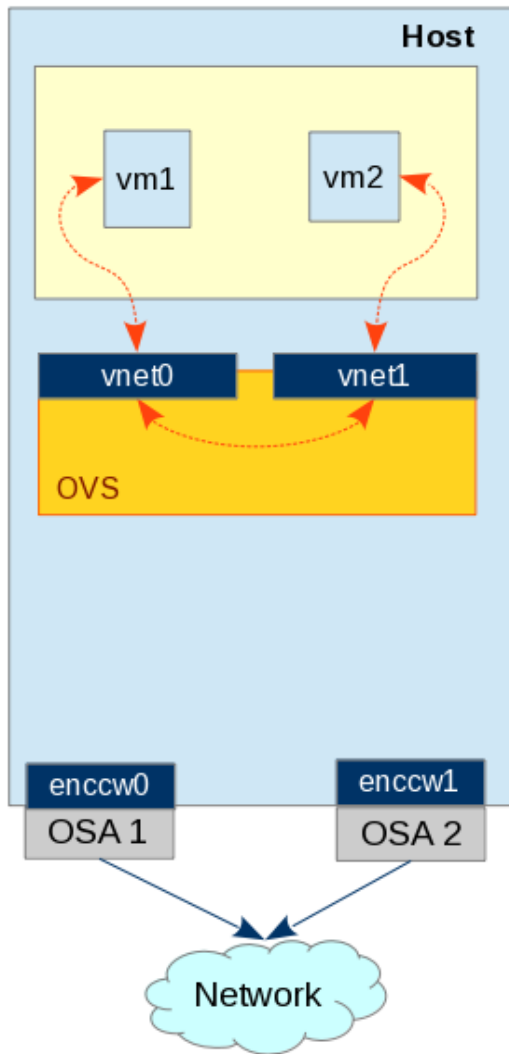
where vnet* is the port to which the guest is connected.

Use cases (VLANs)



Guests on different hosts, having the same VLAN tag may communicate with each other.

Use cases (Bonding)

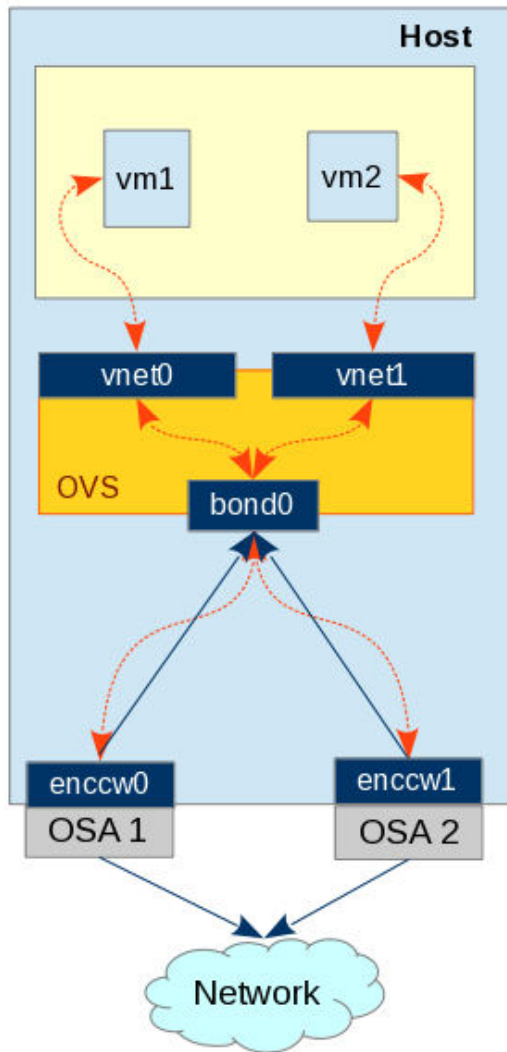


OVS supports link aggregation for several physical NICs.

STEP 1: Setup a bridge

```
ovs-vsctl add-br br0
```

Use cases (Bonding)



OVS supports link aggregation for several physical NICs.

STEP 1: Setup a bridge

```
ovs-vsctl add-br br0
```

STEP 2: Create an ovs bond0 for enccw0 and enccw1

```
ovs-vsctl add-bond br0 bond0 enccw0 enccw1
```

OR

STEP 2: Create an ovs bond0 for enccw0 and enccw, using LACP.

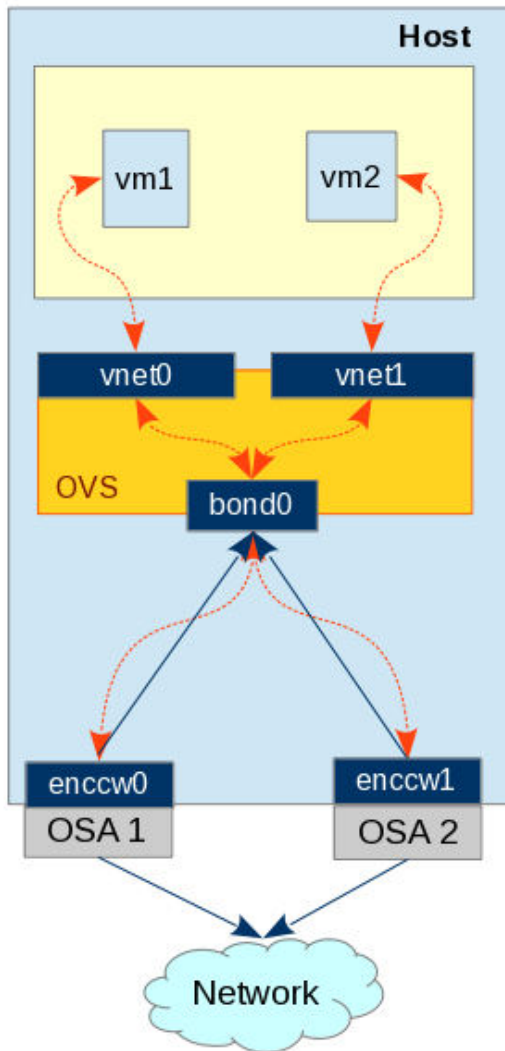
```
ovs-vsctl add-bond br0 bond0 enccw0 enccw1 lacp=active
```

STEP 3: Check you bond

```
ovs-appctl bond/show bond0
```

LACP
(Link Aggregation Control Protocol) is used to work around network limitations cause by presence of STP.

Use cases (Bonding)



To create a permanent bonding configuration:

STEP 1: Create a config file for a bonding interface:

`/etc/sysconfig/network-scripts/ifcfg-bond0`

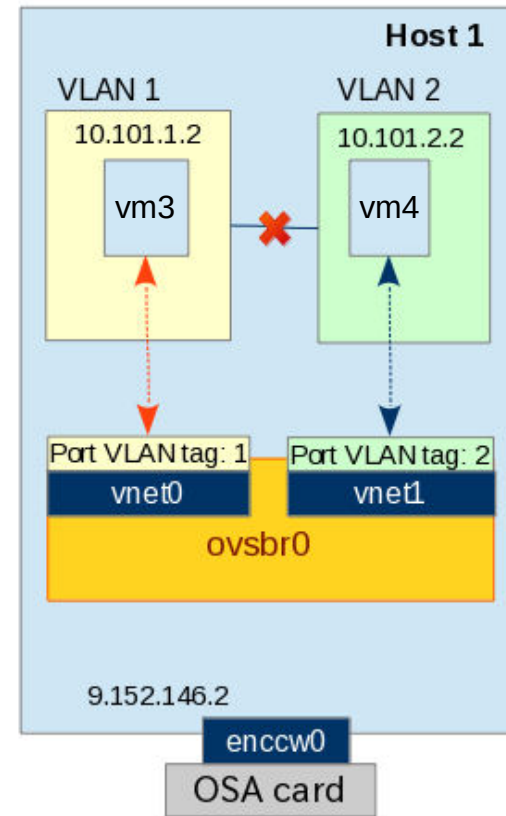
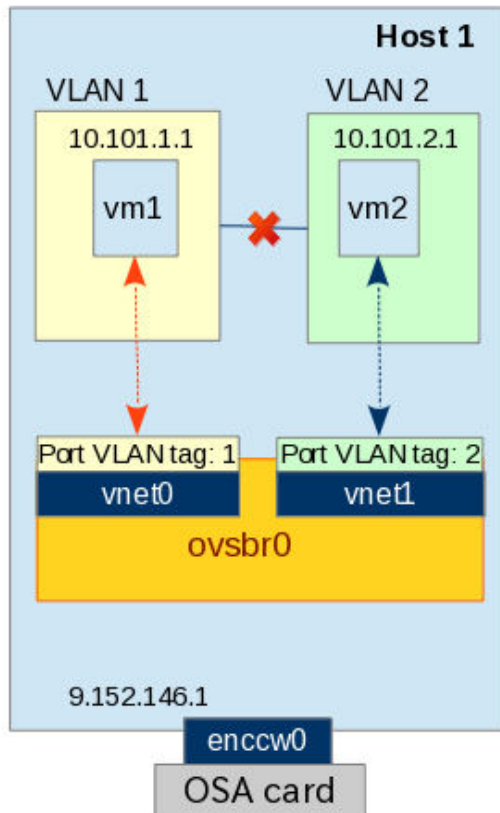
```
DEVICE=bond0
ONBOOT=yes
DEVICETYPE=ovs
TYPE=OVSBond
OVS_BRIDGE=ovsbridge0
BOOTPROTO=none
BOND_IFACES="enccw0 enccw1"
OVS_OPTIONS="lacp=active"
HOTPLUG=no
```

STEP 2: Update config files for bonded interfaces:

`/etc/sysconfig/network-scripts/ifcfg-enccw*`

```
DEVICE=enccw*
ONBOOT=yes
HOTPLUG=no
```

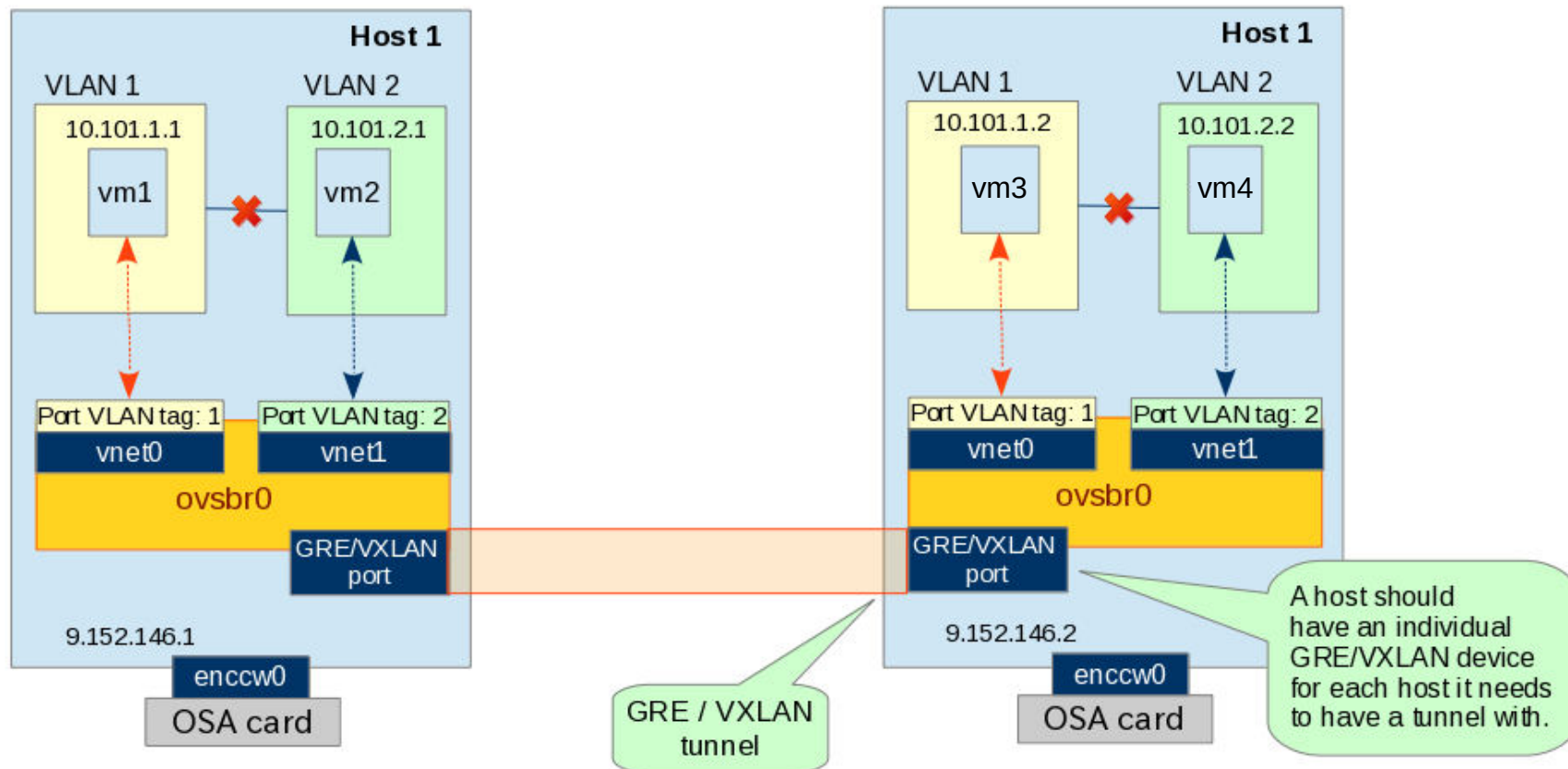
Use cases (Tunneling)



STEP 1: Setup a bridge:

```
ovs-vsctl add-br ovsbr0
```

Use cases (Tunneling)



STEP 1: Setup a bridge:

```
ovs-vsctl add-br ovsbr0
```

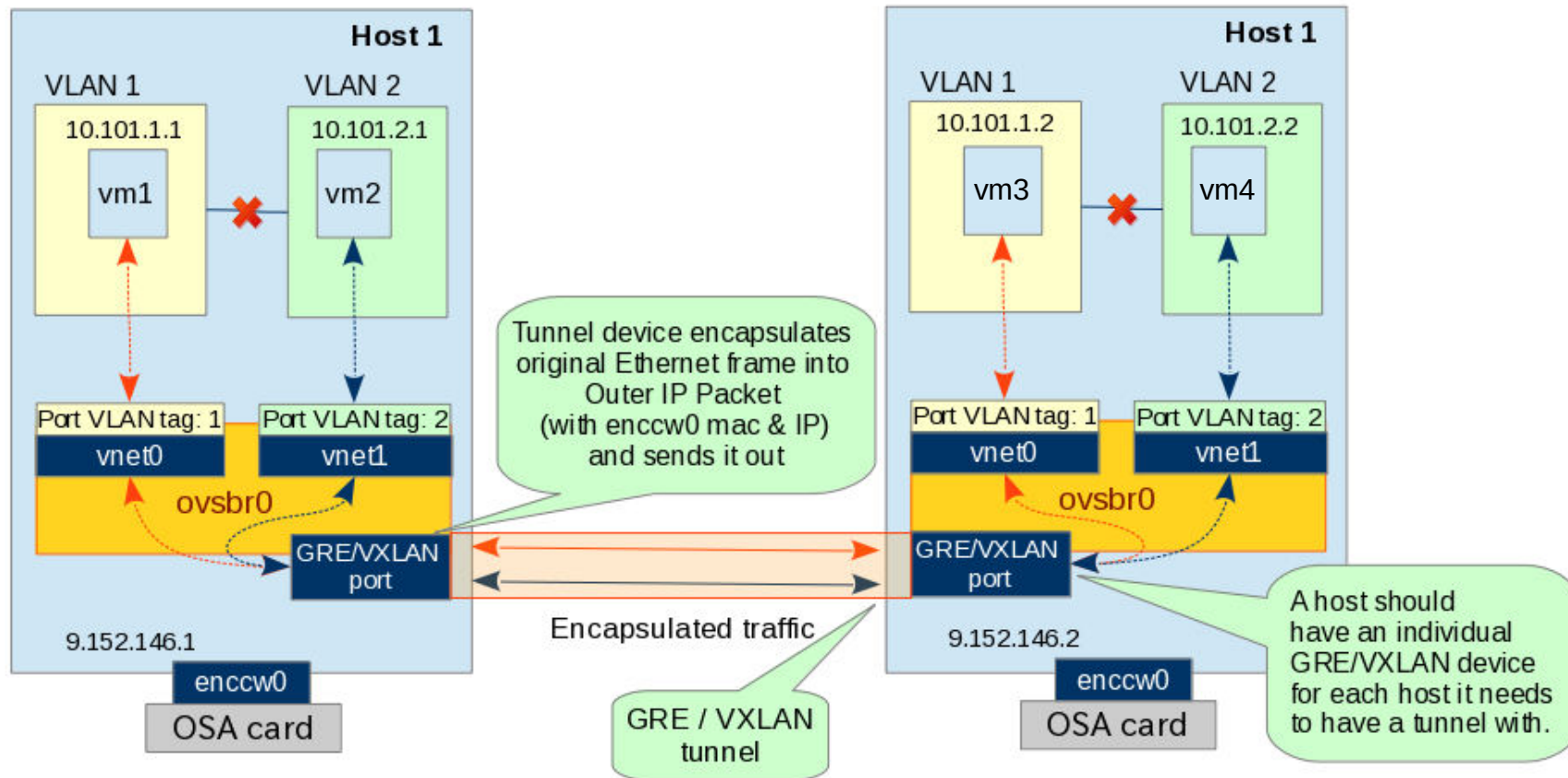
STEP 2: Create a gre/vxlan port:

```
ovs-vsctl add-port ovsbr0 vxlan0 -- set interface vxlan0 type=vxlan options:remote_ip=9.152.146.2
```

Where 'vxlan0' is a port name, 'type' can be "gre or "vxlan", 'remote_ip' is an IP of a remote host.

This configuration should be done on both hosts, that are supposed be connected.

Use cases (Tunneling)



STEP 1: Setup a bridge:

```
ovs-vsctl add-br ovsbr0
```

STEP 2: Create a gre/vxlan port:

```
ovs-vsctl add-port ovsbr0 vxlan0 -- set interface vxlan0 type=vxlan options:remote_ip=9.152.146.2
```

Where 'vxlan0' is a port name, 'type' can be "gre or "vxlan", 'remote_ip' is an IP of a remote host.

This configuration should be done on both hosts, that are supposed be connected.

Thank you for your attention!

Questions?