ipinfusion™

DANOS-Vyatta edition®

Disaggregated Network Operating System
Version 2009a

SDE On-Boarding and VNF Service Chaining

October 2020

IP Infusion Inc.
3965 Freedom Circle, Suite 200
Santa Clara, CA 95054
+1 408-400-1900
http://www.ipinfusion.com/

For support, questions, or comments via E-mail, contact:
support@ipinfusion.com

Contents

Preface

This guide shows how to set up DANOS-Vyatta edition Software-Defined Edge (SDE) devices.

Audience

This guide is intended for network administrators and other engineering professionals who install DANOS-Vyatta edition.

At a minimum, you need to be familiar with:

- Linux operating systems and their command line interfaces
- Internet routing and switching software protocols

Support

For support-related questions, contact support@ipinfusion.com.

Comments

If you have comments, or need to report a problem with the content, contact techpubs@ipinfusion.com.

Assumptions

The Silicom uCPE devices are loaded with DANOS-Vyatta edition compatible BIOS before shipping to customers.

The Silicom uCPE devices are loaded with DANOS-Vyatta edition compatible firmware before shipping to customers.

CHAPTER 1

SDE On-Boarding

Download the Virtual Machine Image

1. Log in to the SDE box.

2. Change to the directory `/var/lib/libvirt/images`:

```
tmpuser@node:~$ cd /var/lib/libvirt/images
tmpuser@node:/var/lib/libvirt/images$
```

3. Download the virtual machine image.

```
tmpuser@node:/var/lib/libvirt/images$ download virtualization image from <uri> disable-
verification authentication username <xxxxx>
Enter password for 'xxxxx':
```

   This is an example showing downloading `vyatta-2005-appliance-amd64.iso` using `scp`:

```
tmpuser@node:/var/lib/libvirt/images$ download virtualization image from scp://
10.12.65.10/root/uCPE/image/vyatta-2005-appliance-amd64.iso disable-verification
authentication username root
Enter password for 'root':
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
 61  520M   61  320M    0     0  10.9M      0  0:00:47  0:00:29  0:00:18 11.1M
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  520M  100  520M    0     0  11.0M      0  0:00:47  0:00:47 --:--:-- 11.1M
100  520M  100  520M    0     0  11.0M      0  0:00:47  0:00:47 --:--:-- 11.0M

tmpuser@node:/var/lib/libvirt/images$ ls -al
total 1871204
drwxrwxr-x+ 3 root   adm        4096 Aug 23 09:04 .
drwxr-xr-x  1 root   root       4096 Aug 18 07:32 ..
drwx------  2 root   root      16384 Aug 10 11:06 lost+found
-rw-rw-r--+ 1 root   root  545259520 Aug 23 09:05 vyatta-2005-appliance-amd64.iso
-rw-rw-r--+ 1 vyatta users 689307648 Aug 23 09:05 vyatta-2005-appliance-amd64_kvm-
1.qcow2
-rw-rw-r--+ 1 vyatta users 681508864 Aug 23 09:05 vyatta-2005-appliance-amd64_kvm.qcow2
tmpuser@node:/var/lib/libvirt/images$
```

4. Verify the virtual machine image:

```
tmpuser@node:~$ show virtualization images
vyatta-2005-appliance-amd64.iso
vyatta-2005-appliance-amd64_kvm-1.qcow2
vyatta-2005-appliance-amd64_kvm.qcow2
tmpuser@node:~$
```

Create Copies of the Virtual Machine Images

1. Log in to the SDE box.
2. Create a copy of virtual machine image.

```
tmpuser@node:~$ copy virtualization image <existing-image-name> to <newly-creating-image-name>
```

This is an example showing copying the virtual machine images:

```
tmpuser@node:~$ show virtualization images
0-vyatta-2005-appliance-amd64_kvm-2.qcow2
1-vyatta-2005-appliance_20200807T1243-amd64-Build14.64.hybrid.iso
vyatta-2005-appliance-amd64.iso
vyatta-2005-appliance-amd64_kvm-1.qcow2
tmpuser@node:~$
tmpuser@node:~$ copy virtualization image vyatta-2005-appliance-amd64_kvm-1.qcow2 to vyatta-2005-appliance-amd64_kvm-2.qcow2
tmpuser@node:~$
tmpuser@node:~$ show virtualization images
0-vyatta-2005-appliance-amd64_kvm-2.qcow2
1-vyatta-2005-appliance_20200807T1243-amd64-Build14.64.hybrid.iso
vyatta-2005-appliance-amd64.iso
vyatta-2005-appliance-amd64_kvm-1.qcow2
vyatta-2005-appliance-amd64_kvm-2.qcow2
tmpuser@node:~$
```

Delete Virtual Machine Images

1. Log in to the SDE box.
2. Delete virtual machine image.

```
tmpuser@node:~$ delete virtualization image <image-name>
```

This is an example showing deleting the virtual machine images:

```
tmpuser@node:~$ show virtualization images
0-vyatta-2005-appliance-amd64_kvm-2.qcow2
1-vyatta-2005-appliance_20200807T1243-amd64-Build14.64.hybrid.iso
vyatta-2005-appliance-amd64.iso
vyatta-2005-appliance-amd64_kvm-1.qcow2
vyatta-2005-appliance-amd64_kvm-2.qcow2
tmpuser@node:~$
tmpuser@node:~$ delete virtualization image vyatta-2005-appliance-amd64_kvm-2.qcow2
tmpuser@node:~$
tmpuser@node:~$ show virtualization images
0-vyatta-2005-appliance-amd64_kvm-2.qcow2
1-vyatta-2005-appliance_20200807T1243-amd64-Build14.64.hybrid.iso
vyatta-2005-appliance-amd64.iso
vyatta-2005-appliance-amd64_kvm-1.qcow2
tmpuser@node:~$
```

Allocate CPUs

Table 1-1: Resource Distribution

Model	Host System		Dataplane		VNF		
vCPU/RAM/Disk	vCPU	RAM GB	vCPU	RAM GB	vCPU	RAM GB	Storage GB
XSmall (4/8/64)	1	2.5	1	0.5	2	5	45
Small (8/16/256)	1	3	1	2	6	11	209
Medium (16/32/512)	2	4	2	4	12	24	416

Table 1-2: Dataplane and VNF CPU resource ID allocation

# CPU Cores	Host CPU ID	Dataplane	Guest VNFs
XSmall (4 no HT)	0	0,1	2,3
Small (4 HT)	0	0,4	(1,5), (2,6), (3,7)
Medium (8 HT)	0,8	0,1,9	(2,10), (3,11), (4,12), (5,13), (6,14), (7,15)

Install a Virtual Machine using a qcow2 Image

1. Log in to the SDE box.
2. Copy the `*.qcow2` image file into the path `/var/lib/libvirt/images/`.
3. Verify the image using the `show virtualization images` command.
4. Configure the number of CPUs.

```
# set virtualization guest <Guest-Name> cpus <no. of CPUs>
```

5. Configure the boot order.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> boot-order <BIOS
boot order>
```

6. Configure the bus type.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> bus-type < virtio>
```

7. Configure the device type.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> device-type <disk>
```

8. Configure the image location.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> source file /var/
lib/libvirt/images/xxxxx.qcow2
```

9. Configure memory (RAM).

```
# set virtualization guest <Guest-Name> memory <xxxx in MB>
```

10. Generate a UUID for the virtual machine.

```
$ generate virtualization uuid
```

11. Configure the UUID for the virtual machine.

```
# set virtualization guest <Guest-Name> uuid <xxxxxx>
```

12. Map the virtual machine VCPU with CPUSET.

```
# set virtualization guest <Guest-Name> vcpupin vcpu <vcpu ID> cpuset <Numbered range of
CPUs>
```

13. Generate a MAC address the virtual machine.

```
$ generate virtualization random-mac
```

14. Configure the vhost interface and assign the MAC address to the interface.

```
# set interfaces vhost dp0vhost00
# set virtualization guest <Guest-Name> devices network dp0vhost00 mac-address
<xx:xx:xx:xx:xx:xx>
```

Note: Creating a vhost interface means creating an interface in the virtual machine

15. This is a complete example showing installing the virtual machine:

```
tmpuser@node:~$ generate virtualization uuid
d4ac8681-60da-43b5-ad49-10ae649301c9
[edit]
tmpuser@node:~$

tmpuser@node:~$ generate virtualization random-mac
52:54:00:3b:29:e3
tmpuser@node:~$ generate virtualization random-mac
52:54:00:36:6b:c2
tmpuser@node:~$ generate virtualization random-mac
52:54:00:59:ea:8a
tmpuser@node:~$

tmpuser@node# set interfaces vhost dp0vhost00
tmpuser@node# set interfaces vhost dp0vhost01
tmpuser@node# set interfaces vhost dp0vhost02

tmpuser@node# set virtualization guest VM-1 cpus 2
tmpuser@node# set virtualization guest VM-1 devices disk 0 boot-order 1
tmpuser@node# set virtualization guest VM-1 devices disk 0 bus-type virtio
tmpuser@node# set virtualization guest VM-1 devices disk 0 device-type disk
tmpuser@node# set virtualization guest VM-1 devices disk 0 source file /var/lib/libvirt/
images/vyatta-2005-appliance-amd64_kvm.qcow2
tmpuser@node# set virtualization guest VM-1 devices network dp0vhost00 mac-address
'52:54:00:3b:29:e3'
tmpuser@node# set virtualization guest VM-1 devices network dp0vhost01 mac-address
'52:54:00:36:6b:c2'
tmpuser@node# set virtualization guest VM-1 devices network dp0vhost02 mac-address
'52:54:00:59:ea:8a'
tmpuser@node# set virtualization guest VM-1 memory 2048
tmpuser@node# set virtualization guest VM-1 uuid d4ac8681-60da-43b5-ad49-10ae649301c9
tmpuser@node# set virtualization guest VM-1 vcpupin vcpu 0 cpuset 1
tmpuser@node# set virtualization guest VM-1 vcpupin vcpu 1 cpuset 5
```

16. Verify Virtual Machine status:

```
tmpuser@node:~$ show virtualization status
Name            State
----            -----
VM-1            running
tmpuser@node:~$
```

Install a Virtual Machine using an iso Image

1. Log in to the SDE box.
2. Copy the `*.iso` image file into the path `/var/lib/libvirt/images/`.
3. Verify the image using the `show virtualization images` command.
4. Configure the number of CPUs.

```
# set virtualization guest <Guest-Name> cpus <no. of CPUs>
```

5. Configure the boot order.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> boot-order <BIOS
boot order>
```

6. Configure the bus type.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> bus-type < virtio>
```

7. Configure the device type.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> device-type <cdrom>
```

8. Configure the image location.

```
# set virtualization guest <Guest-Name> devices disk <Virtual disks> source file /var/
lib/libvirt/images/xxxxx.iso
```

9. Configure memory (RAM).

```
# set virtualization guest <Guest-Name> memory <xxxx in MB>
```

10. Generate a UUID for the virtual machine.

```
$ generate virtualization uuid
```

11. Configure the UUID for the virtual machine.

```
# set virtualization guest <Guest-Name> uuid <xxxxxx>
```

12. Map the virtual machine VCPU with CPUSET.

```
# set virtualization guest <Guest-Name> vcpupin vcpu <vcpu ID> cpuset <Numbered range of
CPUs>
```

13. Generate a MAC address the virtual machine.

```
$ generate virtualization random-mac
```

14. Configure the vhost interface and assign the MAC address to the interface.

```
# set interfaces vhost dp0vhost00
# set virtualization guest <Guest-Name> devices network dp0vhost00 mac-address
<xx:xx:xx:xx:xx:xx>
```

Note: Creating a vhost interface means creating an interface in the virtual machine

15. This is a complete example showing installing the virtual machine:

```
tmpuser@node:~$ generate virtualization uuid
6f08e977-ce72-45d2-a76b-a2679f4528b5
[edit]
tmpuser@node:~$

tmpuser@node:~$ generate virtualization random-mac
52:54:00:31:f6:46
tmpuser@node:~$ generate virtualization random-mac
52:54:00:71:ab:11
tmpuser@node:~$ generate virtualization random-mac
52:54:00:70:88:af
tmpuser@node:~$

tmpuser@node# set interfaces vhost dp0vhost30
tmpuser@node# set interfaces vhost dp0vhost31
tmpuser@node# set interfaces vhost dp0vhost32

tmpuser@node# set virtualization guest VM-3 cpus 2
tmpuser@node# set virtualization guest VM-3 devices disk 1 boot-order 2
tmpuser@node# set virtualization guest VM-3 devices disk 1 bus-type ide
tmpuser@node# set virtualization guest VM-3 devices disk 1 device-type cdrom
tmpuser@node# set virtualization guest VM-3 devices disk 1 source file /var/lib/libvirt/
images/vyatta-2005-appliance-amd64.iso
tmpuser@node# set virtualization guest VM-3 devices network dp0vhost30 mac-address
'52:54:00:31:f6:46'
tmpuser@node# set virtualization guest VM-3 devices network dp0vhost31 mac-address
'52:54:00:71:ab:11'
tmpuser@node# set virtualization guest VM-3 devices network dp0vhost32 mac-address
'52:54:00:70:88:af'
tmpuser@node# set virtualization guest VM-3 memory 2048
tmpuser@node# set virtualization guest VM-3 uuid 6f08e977-ce72-45d2-a76b-a2679f4528b5
tmpuser@node# set virtualization guest VM-3 vcpupin vcpu 0 cpuset 3
tmpuser@node# set virtualization guest VM-3 vcpupin vcpu 1 cpuset 7

e. verify Virtual Machine status:
tmpuser@node:~$ show virtualization status
Name            State
----            -----
VM-3            running
tmpuser@node:~$
```

Log into the Virtual Machine

```
tmpuser@node:~$ monitor virtualization guest DVE1 console
Connected to domain DVE1
Escape character is ^]

node login: tmpuser
Password:
Welcome to DANOS - Vyatta NOS Edition
```

```
Version:      2005
Description:  DANOS - Vyatta NOS Edition 2005 APPLIANCE (Vyatta:Shipping:2005:GA)
Linux node 4.19.0-trunk-vyatta-amd64 #1 SMP PREEMPT Debian 4.19.114-0vyatta2 (2020-04-
29) x86_64
tmpuser@node:~$
```

Disable Autostart

Note: Note: By default, autostart is enabled.

```
tmpuser@node# set virtualization guest <Guest-Name> autostart false
```

Restart the Virtual Machine

```
$ restart virtualization guest <Guest-Name>
```

Shutdown the Virtual Machine

```
$ poweroff virtualization guest <Guest-Name>
```

Start the Virtual Machine

```
$ start virtualization guest <Guest-Name>
```

Delete the Virtual Machine

1. Shut down the Virtual Machine.

```
$ poweroff virtualization guest <Guest-Name>
```

2. Delete the virtual guest.

```
# delete virtualization guest <Guest-Name>
```

3. Delete the vhost interface.

```
# delete interface vhost dp0vhost00
```

Upgrading a Virtual Machine Image

1. Log in to the SDE box.

2. Copy the `*.qcow2` or `*.iso` image file into the path `/var/lib/libvirt/images/`.

3. Verify the image using the `show virtualization images` command.

4. Upgrade the virtual machine with the new image, keeping the old image.

```
tmpuser@node:~$ upgrade virtualization guest <Guest-Name> images device <no.> source
location <uri> keep-old-images
```

5. This is an example showing upgrading a virtual machine `*.iso` image.

```
tmpuser@node:~$ upgrade virtualization guest VM-3 images device 1 source location /var/
lib/libvirt/images/vyatta-2005-appliance_20200807T1243-amd64-Build14.64.hybrid.iso
keep-old-images
[INFO] - Disk 1 configured with source "/var/lib/libvirt/images/1-vyatta-2005-
appliance_20200807T1243-amd64-Build14.64.hybrid.iso"
[INFO] - Powering off existing guest "VM-3"
[INFO] - Starting guest "VM-3"
[INFO] - Disk image upgrade complete
tmpuser@node:~$
```

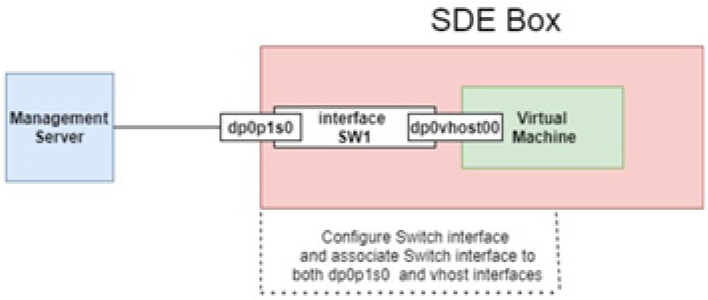
6. Upgrade the virtual machine with the new image, deleting the old image.

```
tmpuser@node:~$ upgrade virtualization guest <Guest-Name> images device <no.> source
location <uri>
```

7. This is an example showing upgrading a virtual machine `*.qcow2` image.

```
tmpuser@node:~$ upgrade virtualization guest VM-1 images device 0 source location /var/
lib/libvirt/images/vyatta-2005-appliance-amd64_kvm-2.qcow2
.[INFO] - Disk 0 configured with source "/var/lib/libvirt/images/0-vyatta-2005-
appliance-amd64_kvm-2.qcow2"
[INFO] - Powering off existing guest "VM-1"
[INFO] - Starting guest "VM-1"
[INFO] - Deleted old image "/var/lib/libvirt/images/vyatta-2005-appliance-
amd64_kvm.qcow2"
[INFO] - Disk image upgrade complete
tmpuser@node:~$
```

Assigning a Management IP Address


SDE Box
Management Server
dp0p1s0
interface SW1
dp0vhost00
Virtual Machine
Configure Switch interface and associate Switch interface to both dp0p1s0 and vhost interfaces

1. Configure the switch interface.

```
tmpuser@node:~$ set interfaces switch <swN> default-port-parameters mode access
tmpuser@node:~$ set interfaces switch <swN> default-port-parameters vlan-parameters
primary-vlan-id <xxxx>
```

2. Associate the SDE device management interface (which is connected to management server) to the switch (swN) interface

```
tmpuser@node:~$ set interfaces dataplane dp0p1s0 switch-group switch <swN>
```

3. Associate the Virtual Machine vhost interface (which is using as management interface of the VM) to the switch interface.

```
tmpuser@node:~$ set interfaces vhost dp0vhost00 switch-group switch <swN>
```

4. Configure the management IP address for SDE device (Create a Switch Virtual Interface(vif) and enable dhcp.)

```
tmpuser@node:~$ set interfaces switch <swN>  vif <xxxx> address dhcp
```

5. Configure management IP address for Virtual Machine (below example is for DVE VM).

   a. Log in to the DVE VM.

   b. Enable DHCP on the management interface.

6. This is an example showing assigning a management IP address for the virtual machine.

```
tmpuser@node# set interfaces switch sw1 default-port-parameters mode access
tmpuser@node# set interfaces switch sw1 default-port-parameters vlan-parameters primary-vlan-id 100
tmpuser@node# set interfaces dataplane dp0p1s0 switch-group switch sw1
tmpuser@node# set interfaces switch sw1 vif 100 address dhcp
tmpuser@node# set interfaces vhost dp0vhost00 switch-group switch sw1
```

7. Configure the Management IP address.

```
tmpuser@node:~$ monitor virtualization guest VM-1 console

Connected to domain VM-1
Escape character is ^]

tmpuser@node:~$configure
tmpuser@node# set interfaces dataplane dp0s3 address dhcp
[edit]
tmpuser@node# commit
[interfaces dataplane dp0s3 address dhcp]
Starting DHCP client on dp0s3 ...


[edit]
tmpuser@node# exit
exit
tmpuser@node:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface       IP Address                        S/L  Speed/Duplex  Description
---------       ----------                        ---  ------------  -----------
dp0s3           10.12.58.117/24                   u/u  a-10g/a-full
dp0s4           10.1.1.1/24                       u/u  a-10g/a-full
dp0s5           20.1.1.1/24                       u/u  a-10g/a-full
lo              1.1.1.1/32                        u/u  -/-
tmpuser@node:~$
```

8. Log in using the Management IP address.

```
[root@localhost ~]# ssh tmpuser@10.12.58.117
Welcome to DANOS - Vyatta NOS Edition

tmpuser@10.12.58.117's password:
Welcome to DANOS - Vyatta NOS Edition
Version:      2005
Description:  DANOS - Vyatta NOS Edition 2005 APPLIANCE (Vyatta:Shipping:2005:GA)
```

```
Linux node 4.19.0-trunk-vyatta-amd64 #1 SMP PREEMPT Debian 4.19.114-0vyatta2 (2020-04-
29) x86_64
Last login: Mon Aug 24 08:09:29 2020 from 10.12.65.10
tmpuser@node:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface       IP Address                        S/L  Speed/Duplex  Description
---------       ----------                        ---  ------------  -----------
dp0s3           10.12.58.117/24                   u/u  a-10g/a-full
dp0s4           10.1.1.1/24                       u/u  a-10g/a-full
dp0s5           20.1.1.1/24                       u/u  a-10g/a-full
lo              1.1.1.1/32                        u/u  -/-
tmpuser@node:~$
```

Monitoring the Virtual Machine

1. Verify the status of the virtual machine.

```
tmpuser@node:~$ show virtualization status
Name            State
----            -----
VM-1            running
VM-2            running
VM-3            running
tmpuser@node:~$
```

2. Verify the vCPU information.

```
$ show virtualization guest <Guest-Name> vcpuinfo
```

This is an example configuration:

```
tmpuser@node:~$ show virtualization guest VM-1 vcpuinfo
VCPU: 0
State: running
CPU Affinity: 1

VCPU: 1
State: running
CPU Affinity: 5

tmpuser@node:~$
```

3. Verify images.

```
tmpuser@node:~$ show virtualization images
0-vyatta-2005-appliance-amd64_kvm-2.qcow2
1-vyatta-2005-appliance_20200807T1243-amd64-Build14.64.hybrid.iso
vyatta-2005-appliance-amd64.iso
vyatta-2005-appliance-amd64_kvm-1.qcow2
tmpuser@node:~$
```

4. Verify the configuration of the virtual machine.

```
$ show configuration path virtualization
```

This is the complete configuration:

```
tmpuser@node:~$ show configuration path virtualization
virtualization {
        guest VM-1 {
                cpus 2
                devices {
                        disk 0 {
                                boot-order 1
                                bus-type virtio
                                device-type disk
                                source {
                                        file /var/lib/libvirt/images/0-vyatta-2005-
appliance-amd64_kvm-2.qcow2
                                }
                        }
                        network dp0vhost00 {
                                mac-address 52:54:00:3b:29:e3
                        }
                        network dp0vhost01 {
                                mac-address 52:54:00:36:6b:c2
                        }
                        network dp0vhost02 {
                                mac-address 52:54:00:59:ea:8a
                        }
                        watchdog {
                                action none
                        }
                }
                memory 2048
                uuid d4ac8681-60da-43b5-ad49-10ae649301c9
                vcpupin {
                        vcpu 0 {
                                cpuset 1
                        }
                        vcpu 1 {
                                cpuset 5
                        }
                }
        }
}
tmpuser@node:~$
```

CHAPTER 2 VNF Service Chaining

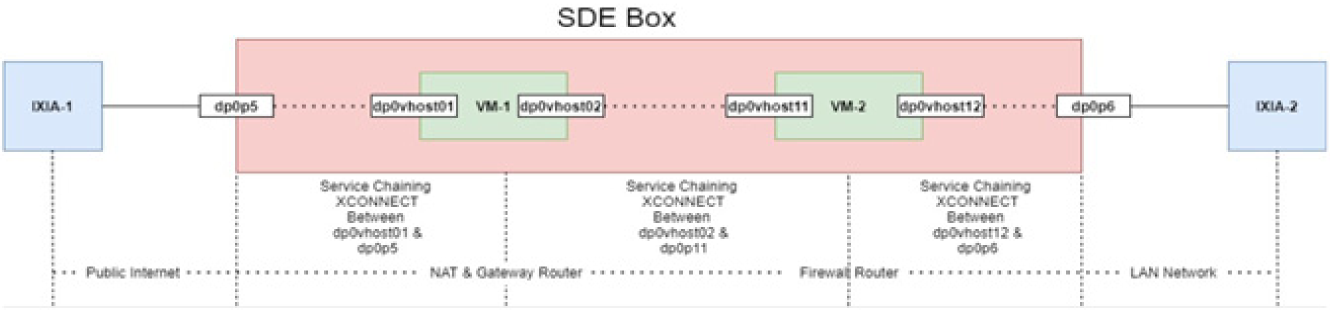
Service chaining enables delivering network and security services in a virtualized data center. With service chaining, the network or security services are extracted from the routers/appliances and created in virtual machines. Since these service virtual machines are normally not in the routing/forwarding path of traffic flows, the underlying forwarding plane (such as vCPE) uses service chaining to steer the packet flows to the corresponding service virtual machines.

These are the types of service chaining:

- Policy Based Routing (PBR)
  - Applied on Layer-3 traffic
  - Traffic could be classified, and goes through all L3 forwarding features
- Cross-Connect (L2 xconnect)
  - Applied on Layer-2 traffic
  - Traffic gets stitched from incoming interface to the outgoing interface
  - Consumes less system resource
  - Could be used if all traffic received on an interface is going through the same service-chain

Service Chaining using Cross-Connect

Topology


SDE Box
IXIA-1
dp0p5
dp0vhost01
VM-1
dp0vhost02
dp0vhost11
VM-2
dp0vhost12
dp0p6
IXIA-2
Service Chaining XCONNECT Between dp0vhost01 & dp0p5
Service Chaining XCONNECT Between dp0vhost02 & dp0p11
Service Chaining XCONNECT Between dp0vhost12 & dp0p6
Public Internet
NAT & Gateway Router
Firewall Router
LAN Network

Virtual machine details:

- VM-1 is a NAT and L3 gateway router
- VM-2 is a Firewall Router

Configure VNF VM-1

1. Generate a UUID for VM-1:

```
$ generate virtualization uuid
d127ec54-8768-450c-9b4a-f3d196af28d5
[edit]
```

```
$
```

2. Generate MAC addresses for VM-1 interfaces.

```
$ generate virtualization random-mac
52:54:00:5a:51:86
$

$ generate virtualization random-mac
52:54:00:7e:62:1f
$

$ generate virtualization random-mac
52:54:00:19:43:c5
$
```

3. Configure a vhost interface.

```
# set interfaces vhost dp0vhost00

# set interfaces vhost dp0vhost01

# set interfaces vhost dp0vhost02
```

4. Configure the number of CPUs.

```
# set virtualization guest VM-1 cpus 2
```

5. Configure the boot order.

```
# set virtualization guest VM-1 devices disk 0 boot-order 1
```

6. Configure the bus type

```
# set virtualization guest VM-1 devices disk 0 bus-type virtio
```

7. Configure the device type:

```
# set virtualization guest VM-1 devices disk 0 device-type disk
```

8. Configure the image location:

```
# set virtualization guest VM-1 devices disk 0 source file /var/lib/libvirt/images/
vyatta-2005-appliance-amd64_kvm.qcow2
```

9. Configure the memory (RAM).

```
# set virtualization guest VM-1 memory 2048
```

10. Configure the UUID for VM-1:

```
# set virtualization guest VM-1 uuid d127ec54-8768-450c-9b4a-f3d196af28d5
```

11. Associate vhost interfaces to VM-1 and assign MAC address to the interface.

```
# set virtualization guest VM-1 devices network dp0vhost00 mac-address
'52:54:00:5a:51:86'
# set virtualization guest VM-1 devices network dp0vhost01 mac-address
'52:54:00:7e:62:1f'
# set virtualization guest VM-1 devices network dp0vhost02 mac-address
'52:54:00:19:43:c5'
```

12. Map VM-1 VCPU with CPUSET:

```
# set virtualization guest VM-1 vcpupin vcpu 0 cpuset 1
# set virtualization guest VM-1 vcpupin vcpu 1 cpuset 5
```

13. Verify VNF VM-1 status mapping VM-1 VCPU with CPUSET.

```
tmpuser@node :~$ show virtualization status
Name            State
----            -----
VM-1            running
[edit]
tmpuser@node :~$
```

Configure VNF VM-2

1. Generate a UUID for VM-2:

```
$ generate virtualization uuid
4b58b1d8-b2b5-49b7-b5bc-8e4060ca4231
[edit]
$
```

2. Generate MAC addresses for VM-2 interfaces:

```
$ generate virtualization random-mac
52:54:00:34:84:8b
$
$ generate virtualization random-mac
52:54:00:34:84:8c
$
$ generate virtualization random-mac
52:54:00:60:30:ee
$
```

3. Configure vhost interfaces:

```
# set interfaces vhost dp0vhost10
# set interfaces vhost dp0vhost11
# set interfaces vhost dp0vhost12
```

4. Configure the number of CPUs.

```
# set virtualization guest VM-2 cpus 2
```

5. Configure the boot order.

```
# set virtualization guest VM-2 devices disk 0 boot-order 1
```

6. Configure the bus type.

```
# set virtualization guest VM-2 devices disk 0 bus-type virtio
```

7. Configure the device type.

```
# set virtualization guest VM-2 devices disk 0 device-type disk
```

8. Configure the image location.

```
# set virtualization guest VM-2 devices disk 0 source file /var/lib/libvirt/images/
vyatta-2005-appliance-amd64_kvm-1.qcow2
```

9. Configure the memory (RAM).

```
# set virtualization guest VM-2 memory 2048
```

10. Configure the UUID for VM-2:

```
# set virtualization guest VM-2 uuid 4b58b1d8-b2b5-49b7-b5bc-8e4060ca4231
```

11. Associate the vhost interfaces to VM-2 and assign MAC addresses to the interfaces:

```
# set virtualization guest VM-2 devices network dp0vhost10 mac-address
'52:54:00:34:84:8b'
# set virtualization guest VM-2 devices network dp0vhost11 mac-address
'52:54:00:34:84:8c'
# set virtualization guest VM-2 devices network dp0vhost12 mac-address
'52:54:00:60:30:ee'
```

12. Map VM-2 VCPU with CPUSET:

```
# set virtualization guest VM-2 vcpupin vcpu 0 cpuset 2
# set virtualization guest VM-2 vcpupin vcpu 1 cpuset 6
```

13. Verify VNF VM-1 status mapping VM-1 VCPU with CPUSET:

```
tmpuser@node :~$ show virtualization status
Name            State
----            -----
VM-1            running
VM-2            running

[edit]
tmpuser@node :~$
```

Configure Cross-Connect Service Chaining

1. Configure XCONNECT between SDE device physical interface dp0p5 and VNF VM-1 vhost interface dp0vhost01.

```
# set interface dataplane dp0p5 hardware-switching disable
# set interface dataplane dp0p5 xconnect vhost dp0vhost01
# set interface vhost dp0vhost01 xconnect dataplane dp0p5
```

2. Configure XCONNECT between VNF VM-1 vhost interface dp0vhost02 and VNF VM-2 vhost interface dp0vhost11.

```
# set interface vhost dp0vhost02 xconnect vhost dp0vhost11
# set interface vhost dp0vhost11 xconnect vhost dp0vhost02
```

3. Configure XCONNECT between VNF VM-2 vhost interface dp0vhost12 and SDE device physical interface dp0p6.

```
# set interface vhost dp0vhost12 xconnect dataplane dp0p6
# set interface dataplane dp0p6 hardware-switching disable
# set interface dataplane dp0p6 xconnect vhost dp0vhost12
```

Traffic Flow

1. Traffic received from IXIA-1(public internet) to SDE interface dp0p5.

2. In SDE device xconnect is configured between interface dp0p5 and vhost interface dp0vhost01.

3. SDE device send all the packets received on interface dp0p5 to vhost interface dp0vhost01.

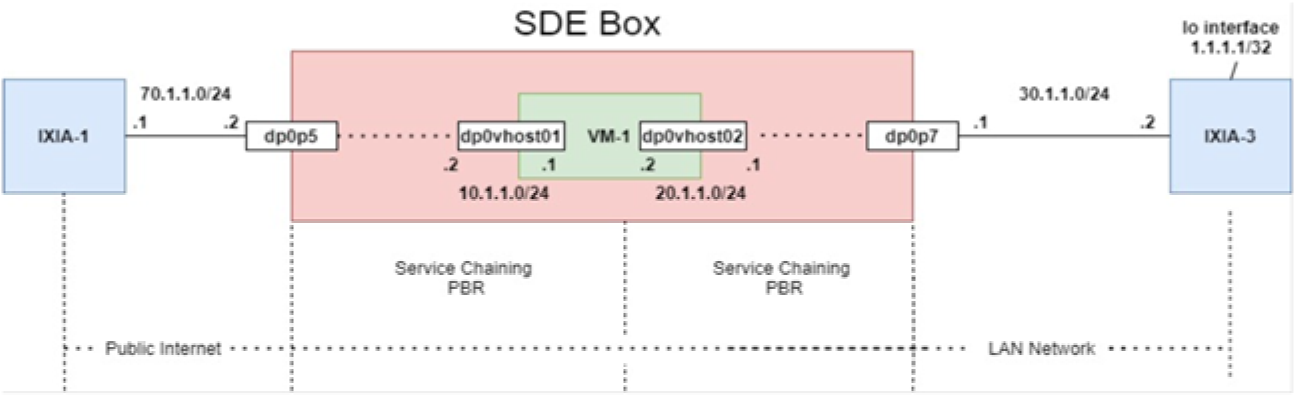
4. IN VNF VM-1 router, NAT will happen and the packets are sent on the base of routing table and the packets are egressed on corresponding interfaces.

5. SDE device will receive packets on vhost interface dp0vhost02.

6. In SDE device xconnect is configured between vhost interface `dp0vhost02` and vhost interface `dp0vhost11`.
7. So SDE device will send all the packets received on vhost interface dp0vhost02 to vhost interface `dp0vhost11`.
8. IN VNF VM-2 (firewall router), packet filtering will happen and the packets are sent out of interface `dpovhost12`.
9. In SDE device xconnect is configured between vhost interface `dp0vhost12` and interface `dp0p6`.
10. SDE device sends all the packets received on vhost interface `dp0vhost12` to interface `dp0p6`.

Service Chaining using Policy-Based Routing

Topology



Virtual machine details:

- VM-1 is a gateway router.

Configure VNF VM-1

1. Generate a UUID for VM-1.

```
$ generate virtualization uuid
d127ec54-8768-450c-9b4a-f3d196af28d5
[edit]
$
```

2. Generate MAC addresses for VM-1 interfaces.

```
$ generate virtualization random-mac
52:54:00:5a:51:86
$

$ generate virtualization random-mac
52:54:00:7e:62:1f
$

$ generate virtualization random-mac
```

```
52:54:00:19:43:c5
$
```

3. Configure the vhost interfaces:

```
# set interfaces vhost dp0vhost00
# set interfaces vhost dp0vhost01
# set interfaces vhost dp0vhost02
```

4. Configure the numbe of CPUs.

```
# set virtualization guest VM-1 cpus 2
```

5. Configure the boot order.

```
# set virtualization guest VM-1 devices disk 0 boot-order 1
```

6. Configure the bus type.

```
# set virtualization guest VM-1 devices disk 0 bus-type virtio
```

7. Configure the device type.

```
# set virtualization guest VM-1 devices disk 0 device-type disk
```

8. Configure the image location.

```
# set virtualization guest VM-1 devices disk 0 source file /var/lib/libvirt/images/
vyatta-2005-appliance-amd64_kvm.qcow2
```

9. Configure the memory (RAM).

```
# set virtualization guest VM-1 memory 2048
```

10. Configure the UUID for VM-1:

```
# set virtualization guest VM-1 uuid d127ec54-8768-450c-9b4a-f3d196af28d5
```

11. Associate vhost interfaces to VM-1 and assign MAC addresses to the interface:

```
# set virtualization guest VM-1 devices network dp0vhost00 mac-address
'52:54:00:5a:51:86'
# set virtualization guest VM-1 devices network dp0vhost01 mac-address
'52:54:00:7e:62:1f'
# set virtualization guest VM-1 devices network dp0vhost02 mac-address
'52:54:00:19:43:c5'
```

12. Map VM-1 VCPU with CPUSET:

```
# set virtualization guest VM-1 vcpupin vcpu 0 cpuset 1
# set virtualization guest VM-1 vcpupin vcpu 1 cpuset 5
```

13. Verify VNF VM-1 status mapping VM-1 VCPU with CPUSET:

```
tmpuser@node :~$ show virtualization status
Name            State
----            -----
VM-1            running
[edit]
tmpuser@node :~$
```

Configure Policy-Based Routing Service Chaining

1. Configure interfaces as per the topology.

```
# set interfaces dataplane dp0p5 address 70.1.1.2/24
# set interfaces dataplane dp0p5 hardware-switching disable
# set interfaces vhost dp0vhost01 address 10.1.1.2/24 #connected to VM-1
# set interfaces vhost dp0vhost02 address 20.1.1.1/24 #connected to VM-1

# set interfaces dataplane dp0p7 address 30.1.1.1/24
# set interfaces dataplane dp0p7 hardware-switching disable
```

2. Create two policy-based routing configurations.

```
# set policy route pbr sc1-h1 rule 1 action accept
# set policy route pbr sc1-h1 rule 1 address-family ipv4
# set policy route pbr sc1-h1 rule 1 destination address 1.1.1.1/32
# set policy route pbr sc1-h1 rule 1 table 1

# set policy route pbr sc1-h2 rule 1 action accept
# set policy route pbr sc1-h2 rule 1 address-family ipv4
# set policy route pbr sc1-h2 rule 1 destination address 1.1.1.1/32
# set policy route pbr sc1-h2 rule 1 table 2
```

3. Configure two static route table entries.

```
# set static table 1 route 1.1.1.1/32 next-hop 10.1.1.1
# set static table 2 route 1.1.1.1/32 next-hop 30.1.1.2
```

4. Configure policy-based routing on SDE interfaces `dp0p5` and `dp0vhost02`.

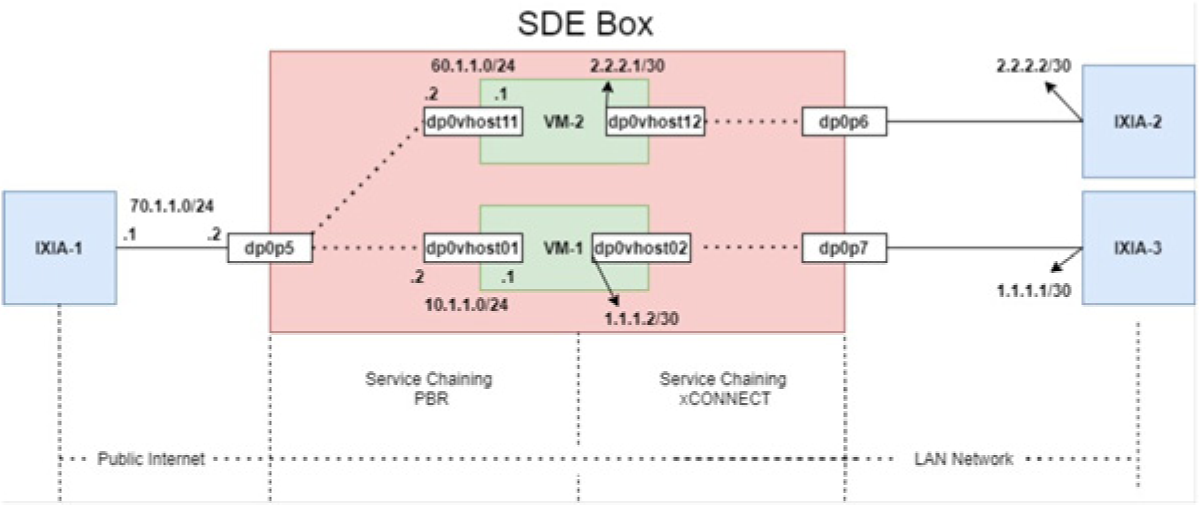
```
# set interfaces dataplane dp0p5 policy route pbr sc1-h1
# set interfaces vhost dp0vhost02 policy route pbr sc1-h2
```

Traffic Flow

1. Traffic is received from IXIA-1(public Internet) at SDE interface dp0p5.
2. The SDE device checks the destination IP address of the packet and the packet will be routed based on the policy-based routing configured on interface `dp0p5` (the packet will be sent to `dp0vhost01` [VM-1]).
3. In VM-1, the packet will be processed and sent out of `dpovhost02`.
4. The SDE device checks the destination IP address of the packet and the packet will be routed based on the policy-based routing configured on interface `dpovhost02`.
5. SDE device will send all the packets received on vhost interface `dp0vhost02` to interface `dp0p7`.

Service Chaining using both PBR and Cross-Connect

Topology



Virtual machine details:

- VM-1 and VM-2 are two different gateway routers

Configure VNF VM-1

1. Generating a UUID for VM-1:

```
$ generate virtualization uuid
d127ec54-8768-450c-9b4a-f3d196af28d5
[edit]
$
```

2. Generate MAC addresses for the VM-1 interfaces:

```
$ generate virtualization random-mac
52:54:00:5a:51:86
$
$ generate virtualization random-mac
52:54:00:7e:62:1f
$
$ generate virtualization random-mac
52:54:00:19:43:c5
$
```

3. Configure the vhost interfaces:

```
# set interfaces vhost dp0vhost00
# set interfaces vhost dp0vhost01
# set interfaces vhost dp0vhost02
```

4. Configure the number of CPUs.

```
# set virtualization guest VM-1 cpus 2
```

5. Configure the boot order.

```
# set virtualization guest VM-1 devices disk 0 boot-order 1
```

6. Configure the bus type.

```
# set virtualization guest VM-1 devices disk 0 bus-type virtio
```

7. Configure the device type.

```
# set virtualization guest VM-1 devices disk 0 device-type disk
```

8. Configure the image location.

```
# set virtualization guest VM-1 devices disk 0 source file /var/lib/libvirt/images/
vyatta-2005-appliance-amd64_kvm.qcow2
```

9. Configure the memory (RAM).

```
# set virtualization guest VM-1 memory 2048
```

10. Configure UUID for VM-1.

```
# set virtualization guest VM-1 uuid d127ec54-8768-450c-9b4a-f3d196af28d5
```

11. Associate vhost interfaces to VM-1 and assign MAC addresses to the interfaces.

```
# set virtualization guest VM-1 devices network dp0vhost00 mac-address
'52:54:00:5a:51:86'
# set virtualization guest VM-1 devices network dp0vhost01 mac-address
'52:54:00:7e:62:1f'
# set virtualization guest VM-1 devices network dp0vhost02 mac-address
'52:54:00:19:43:c5'
```

12. Map VM-1 VCPU with CPUSET.

```
# set virtualization guest VM-1 vcpupin vcpu 0 cpuset 1
# set virtualization guest VM-1 vcpupin vcpu 1 cpuset 5
```

13. Verify VNF VM-1 status mapping VM-1 VCPU with CPUSET.

```
tmpuser@node :~$ show virtualization status
Name            State
----            -----
VM-1            running
[edit]
tmpuser@node :~$
```

Configure VNF VM-2

1. Generate a UUID for VM-2.

```
$ generate virtualization uuid
4b58b1d8-b2b5-49b7-b5bc-8e4060ca4231
[edit]
$
```

2. Generate MAC address for VM-2 interfaces

```
$ generate virtualization random-mac
52:54:00:34:84:8b
$
```

```
$ generate virtualization random-mac
52:54:00:34:84:8c
$
$ generate virtualization random-mac
52:54:00:60:30:ee
$
```

3. Configure vhost interfaces:

```
# set interfaces vhost dp0vhost10
# set interfaces vhost dp0vhost11
# set interfaces vhost dp0vhost12
```

4. Configure the number of CPUs

```
# set virtualization guest VM-2 cpus 2
```

5. Configure the boot order.

```
# set virtualization guest VM-2 devices disk 0 boot-order 1
```

6. Configure the bus type.

```
# set virtualization guest VM-2 devices disk 0 bus-type virtio
```

7. Configure the device type.

```
# set virtualization guest VM-2 devices disk 0 device-type disk
```

8. Configure the image location.

```
# set virtualization guest VM-2 devices disk 0 source file /var/lib/libvirt/images/
vyatta-2005-appliance-amd64_kvm-1.qcow2
```

9. Configure the memory (RAM).

```
# set virtualization guest VM-2 memory 2048
```

10. Configure the UUID for VM-2.

```
# set virtualization guest VM-2 uuid 4b58b1d8-b2b5-49b7-b5bc-8e4060ca4231
```

11. Associate vhost interfaces to VM-2 and assign MAC addresses to the interfaces:

```
# set virtualization guest VM-2 devices network dp0vhost10 mac-address
'52:54:00:34:84:8b'
# set virtualization guest VM-2 devices network dp0vhost11 mac-address
'52:54:00:34:84:8c'
# set virtualization guest VM-2 devices network dp0vhost12 mac-address
'52:54:00:60:30:ee'
```

12. Map VM-2 VCPU with CPUSET:

```
# set virtualization guest VM-2 vcpupin vcpu 0 cpuset 2
# set virtualization guest VM-2 vcpupin vcpu 1 cpuset 6
```

13. Verify VNF VM-1 status mapping VM-1 VCPU with CPUSET:

```
tmpuser@node :~$ show virtualization status
Name            State
----            -----
VM-1            running
VM-2            running

[edit]
tmpuser@node :~$
```

Configure Policy-Based-Routing Service Chaining

1. Configure interfaces as per the Topology

```
# set interfaces dataplane dp0p5 address 70.1.1.2/24
# set interfaces dataplane dp0p5 hardware-switching disable

# set interfaces vhost dp0vhost01 address 10.1.1.2/24 #connected to VM-1

# set interfaces vhost dp0vhost11 address 60.1.1.2/24 #connected to VM-2
```

2. Configure Policy-Based-Routing

```
# set policy route pbr sc1-h1 rule 1 action accept
# set policy route pbr sc1-h1 rule 1 address-family ipv4
# set policy route pbr sc1-h1 rule 1 destination address 1.1.1.1/32
# set policy route pbr sc1-h1 rule 1 table 1
# set policy route pbr sc1-h1 rule 2 action accept
# set policy route pbr sc1-h1 rule 2 address-family ipv4
# set policy route pbr sc1-h1 rule 2 destination address 2.2.2.2/32
# set policy route pbr sc1-h1 rule 2 table 1
```

3. Configure Static route table

```
# set static table 1 route 1.1.1.1/32 next-hop 10.1.1.1
# set static table 1 route 2.2.2.2/32 next-hop 60.1.1.1
```

4. Configure PBR on SDE interface dp0p5

```
# set interfaces dataplane dp0p5 policy route pbr sc1-h1
```

5. Configure Cross-Connect service chaining:

   a. Configure XCONNECT between SDE-Device physical interface `dp0p7` and VNF VM-1 vhost interface `dp0vhost02`.

```
# set interface dataplane dp0p7 hardware-switching disable
# set interface dataplane dp0p7 xconnect vhost dp0vhost02

# set interface vhost dp0vhost02 xconnect dataplane dp0p7
```

6. Configure XCONNECT between VNF VM-2 vhost interface `dp0vhost12` and SDE-Device physical interface `dp0p6`.

```
# set interface vhost dp0vhost12 xconnect dataplane dp0p6

# set interface dataplane dp0p6 hardware-switching disable
# set interface dataplane dp0p6 xconnect vhost dp0vhost12
```

Traffic Flow

1. Traffic is received from IXIA-1 (public Internet) at SDE interface `dp0p5`.
2. The SDE device checks the destination IP address of the packet and routes it per the policy-based routing configuration for the interface.
3. If the destination IP address of Packet is 1.1.1.1 then the packet will be sent to `dp0vhost01` (VM-1).
4. If the destination IP address of Packet is 2.2.2.2 then the packet will be sent to `dp0vhost11` (VM-2).

5. In VM-1, the packet will be processed and sent out of `dpovhost02`.
6. In the SDE device, xconnect is configured between interface `dp0p7` and vhost interface `dp0vhost02`.
7. The SDE device will send all the packets received on vhost interface `dp0vhost02` to interface `dp0p7`.
8. In VM-2, the packet will be processed and sent out of `dpovhost12`.
9. In the SDE device, xconnect is configured between interface `dp0p6` and vhost interface `dp0vhost12`.
10. The SDE device will send all the packets received on vhost interface `dp0vhost12` to interface `dp0p6`.