OSM Hackfest – Session 3
Modeling multi-VDU VNF
Gianpietro Lavado (Whitestack)
VNF diagram

VNF: hackfest2-vnf

VL: internal

ICP: mgmtVM-internal
ICP: dataVM-internal

External Connection point: vnf-mgmt
External Connection point: vnf-data

VDU: mgmtVM
- Image name: US1604
- VM Flavor: 1 CPU, 1GB RAM, 10 GB disk
- Interfaces:
  - mgmtVM-eth0: VIRTIO
  - mgmtVM-eth1: VIRTIO

mgmtVM-eth0
mgmtVM-eth1
dataVM-eth0
dataVM-xe0
dataVM
- Image name: US1604
- VM Flavor: 1 CPU, 1GB RAM, 10 GB disk
- Interfaces:
  - dataVM-eth0: VIRTIO
  - dataVM-xe0: VIRTIO
Creating the new multi-VDU VNF (1/3)

Use the tool to create a new VNFD for a VNF called: "hackfest2-vnf"

```
/usr/share/osm-devops(descriptor-packages/tools/generate_descriptor_pkg.sh -t vnfd --image US1604 -c hackfest2-vnf
```

- Add 2 Connection Points (external):
  - CONNECTION POINT 1:
    - name: vnf-mgmt
  - CONNECTION POINT 2:
    - name: vnf-data

- Add new VLD ‘internal’ to the VNF:
  - Name: internal
  - TYPE: ELAN
  - Refer to internal CPs we will define later

---

**Internal VLD example**

```
...  
mgmt-interface:  
  cp: vnf-mgmt
internal-vld:
  - id: internal
    name: internal
    short-name: internal
    type: ELAN
  internal-connection-point:
    - id-ref: mgmtVM-internal
    - id-ref: dataVM-internal
...
```
Creating the new multi-VDU VNF (2/4)

- Add VDU1 in the VNF
  - Name: mgmtVM
  - Image: US1604
  - VM Flavor:
    - VCPU COUNT: 1
    - MEMORY MB: 1024
    - STORAGE GB: 10
- Add 1 internal connection point:
  - ID: mgmtVM-internal
  - Name: mgmtVM-internal
  - Type: VPORT
- Add 2 interfaces to the VDU:
  - Interface 1:
    - Name: mgmtVM-eth0
    - Position: 1
    - Connection-point-type: EXTERNAL
    - EXTERNAL-CONNECTION-POINT-REF: vnf-mgmt
    - Virtual-interface:
      - Type: VIRTIO
  - Interface 2:
    - Name: mgmtVM-eth1
    - Position: 2
    - Connection-point-type: INTERNAL
    - INTERNAL-CONNECTION-POINT-REF: mgmtVM-internal
    - Virtual-interface:
      - Type: VIRTIO
• Add VDU2 in the VNF
  • Name: dataVM
  • Image: US1604
  • VM Flavor:
    • VCPU COUNT: 1
    • MEMORY MB: 1024
    • STORAGE GB: 10
• Add 1 internal connection point:
  • ID: dataVM-internal
  • Name: dataVM-internal
  • Type: VPORT
• Add 2 interfaces to the VDU:
  • Interface 1:
    • Name: dataVM-eth0
    • Position: 1
    • Connection-point-type: INTERNAL
    • INTERNAL-CONNECTION-POINT-REF: dataVM-internal
    • Virtual-interface:
      • Type: VIRTIO
  • Interface 2:
    • Name: dataVM-xe0
    • Position: 2
    • Connection-point-type: EXTERNAL
    • EXTERNAL-CONNECTION-POINT-REF: vnf-data
    • Virtual-interface:
      • Type: VIRTIO
• Validate your descriptor using the tool:

    /usr/share/osm-devops/descriptor-packages/tools/validate_descriptor.py [yaml file]

• And finally, against the sample file:

    Hackfest 2 VNF Descriptor - https://osm-download.etsi.org/ftp/osm-4.0-four/3rd-hackfest/packages/hackfest_2_vnfd.tar.gz
NS diagram

NS: hackfest2-ns

VL: mgmtnet
CP: vnf-mgmt
VNF: hackfest2-vnf
CP: vnf-data
VL: datanet
CP: vnf-data
VNF: hackfest2-vnf
Creating the NSD (1/3)

Use the tool to create a new NSD for a NS called: "hackfest2-ns"

Add NSD

- Name: hackfest2-ns

- Specify constituent VNFs (hackfest2-vnf)

- Add first VLD:
  - VLD1:
    - name (optional): mgmtnet
    - TYPE: ELAN
    - MGMT NETWORK: True
    - INIT PARAMS
      - vim-network-ref
        - VIM NETWORK NAME: mgmt

<- This is to have a default mapped VIM network
Creating the NSD (2/3)

• Add second VLD:
  • VLD2:
    • name (optional): datanet
    • TYPE: ELAN
    • MGMT NETWORK: False (default)

• Refer VNF Connection Points to the VLs:
  • vnf-mgmt → VL:mgmtnet
  • vnf-data → VL:datanet
Creating the NSD (3/3)

• Validate your descriptor using the tool:

/usr/share/osm-devops(descriptor-packages/tools/validate_descriptor.py [yaml file]

• And finally, against the sample file:

Hackfest 2 NS Descriptor - https://osm-download.etsi.org/ftp/osm-4.0-four/3rd-hackfest/packages/hackfest_2_nsd.tar.gz
Deploying NS in the UI

• Using the corresponding tool, create your packages
  • /usr/share/osm-devops/descriptor-packages/tools/generate_descriptor_pkg.sh -t vnfd -N <VNF_NAME>_vnfd
  • /usr/share/osm-devops/descriptor-packages/tools/generate_descriptor_pkg.sh -t nsd -N <NS_NAME>_nsd

• Onboard VNFD and NSD to catalog using the UI

• Launch the NS from the UI
  • Depending on the VIM, specify a VIM network name to map MGMTNET
  • If you need to change the VIM, change the network name using config:
    
    {vld: [(name: mgmtnet, vim-network-name: public1)]}

• Click the info button to see the mgmt IP address of each VNF

• Connect to each VNF:
  • ssh osm@<IP> (pwd: osm4u)
Final Multi-VDU Picture

NS: hackfest2

VNF 1 / VDU: mgmtVM

VNF 2 / VDU: mgmtVM

VNF: hackfest2-vnf

Internal VLDs & CPs

VNF 1 / VDU: dataVM

VNF 2 / VDU: dataVM

External VLDs & CPs
Network modelling with IP Profiles

- Using IP Profiles, we can configure the attributes of subnets that are created by OSM. We can do it for internal or external VLDs.

- Subnet's DHCP server will not deliver a default gateway if explicitly set to 0.0.0.0

---

**External VLD with IP Profile (NSD level)**

...  
  - id: NS1-nsd  
  ...  
  ip-profiles:  
    - name: profile_external1  
      description: external network  
      ip-profile-params:  
        ip-version: ipv4  
        dns-server: 8.8.8.8  
        gateway-address:  
        subnet-address: 192.168.17.0/24  
      dhcp-params:  
        enabled: true  
  vld:  
    - id: external1  
      ip-profile-ref: profile_external1  
...  

**Internal VLD with IP Profile (VNFD level)**

...  
  - id: VNF1-vnfd  
  ...  
  ip-profiles:  
    - name: p1  
      description: p1  
      ip-profile-params:  
        ip-version: ipv4  
        dns-server:  
        gateway-address: 0.0.0.0  
        subnet-address: 192.168.100.0/24  
      dhcp-params:  
        enabled: true  
  internal-vld:  
    - id: internal  
      ip-profile-ref: p1  
...  

Note: attributes with no values are informational only, they can be removed.
Network modelling with Static IPs

• We can also set static IP addresses, having IP Profile and DHCP enabled.

**External VLD with IP Profile (NSD level)**

```json
...  
  - id: NS1-nsd

...  
  ip-profiles:
  - name: profile_external1
    description: external network
    ip-profile-params:
      ip-version: ipv4
      subnet-address: 192.168.17.0/24
    dhcp-params:
      enabled: true

vld:
  - id: external1
    ip-profile-ref: profile_external1
    vnfd-connection-point-ref:
      ...  
      ip-address: 192.168.17.100
```

**Internal VLD with IP Profile (VNFD level)**

```json
...  
  - id: VNF1-vnfd

...  
  ip-profiles:
  - name: p1
    description: p1
    ip-profile-params:
      ip-version: ipv4
      subnet-address: 192.168.100.0/24
    dhcp-params:
      enabled: true

internal-vld:
  - id: internal
    ip-profile-ref: p1
    internal-connection-point:
      ...  
      id-ref: mgmtVM-internal
      ip-address: 192.168.100.100
```
Network modelling with MACs

- We can set MAC addresses as well, just set them up at the VDU level.

```yaml
Changing MAC (VNFD level)
...
interface:
  - name: mgmtVM-eth0
    position: '1'
    type: EXTERNAL
    virtual-interface:
      type: VIRTIO
      external-connection-point-ref: vnf-mgmt
      mac-address: '01:02:03:01:02:03'
  - name: mgmtVM-eth1
    position: '2'
    type: INTERNAL
    virtual-interface:
      type: VIRTIO
      internal-connection-point-ref: mgmtVM-internal
      mac-address: '03:02:01:03:02:01'
...
```

--> Be careful about duplicated MACs!