OSM-MR#9 Hackfest
Building a Multi-VDU VNF with Day-0
Gianpietro Lavado (Whitestack)
Building a Multi-VDU VNF
Let’s start with the VNF
VNF diagram

VNF: hackfest_magma-agw-enb_vnfd

VDU: mgmtVM
- Image name: magma101_hfmr9
- VM Flavor: 1 CPU, 4 GB RAM, 50 GB disk
- Interfaces:
  - eth0: VIRTIO / SR-IOV
  - eth1: VIRTIO
  - eth2: VIRTIO

VL: internalS1

VDU: dataVM
- Image name: srsLTEzmqRF_hf9
- VM Flavor: 4 CPU, 6 GB RAM, 100 GB disk
- Interfaces:
  - eth0: VIRTIO
  - eth1: VIRTIO / SR-IOV

ICP: agw-s1
ICP: srsLTE-s1

External CP: agw-mgmt
External CP: agw-sgi
External CP: srsLTE-mgmt

192.168.100.254
192.168.100.10
Building a Multi-VDU VNF Package from scratch

- We can use the graphical composer for the VNFD, then download the package to add other artifacts, but it will be faster through the CLI.
  - Compose a new VNF
  - Create new Package
Creating a new VNF Package from CLI

- Use the command line to create the complete structure of the package, modify as desired with an editor.

```bash
osm package-create --base-directory ~/magma --image magma101_hfmr9 --vcpu 1 --memory 4096 --storage 50 --interfaces 2 --vendor OSM vnf hackfest_magma-agw-enb
```

- The final contents we need for this section are place in the following folder:
  
  `/home/ubuntu/examples/01-multivdu/hackfest_magma-agw-enb_vnfd/hackfest_magma-agw-enb_vnfd.yaml`
Creating a new VNF Package from CLI

Two options:

1. View the desired contents and replace your `hackfest_magma-agw-enb_vnfd.yaml` file, section by section.

   ```
   cat /home/ubuntu/examples/01-multivdu/hackfest_magma-agw-enb_vnfd/hackfest_magma-agw-enb_vnfd.yaml
   ```

2. [Faster] Copy all the contents from the `base_packages` directory into your VNF folder

   ```
   cp -a /home/ubuntu/examples/01-multivdu/hackfest_magma-agw-enb_vnfd/*
   ~/magma/hackfest_magma-agw-enb_vnf/
   ```
In our first VDU, interfaces section, we will make sure we have our internal “s1” interface first.

```
vdu:
  - id: magma-agw-vdu
    ...
  interface:
    - name: eth0
      type: INTERNAL
      position: 1
      virtual-interface:
        type: PARAVIRT
        internal-connection-point-ref: agw-s1
    - name: eth1
      type: EXTERNAL
      position: 2
      virtual-interface:
        type: PARAVIRT
        external-connection-point-ref: agw-sgi

  - name: eth2
    type: EXTERNAL
    position: 3
    virtual-interface:
      type: PARAVIRT
      external-connection-point-ref: agw-mgmt
    internal-connection-point:
      - id: agw-s1
        name: agw-s1
        short-name: agw-s1
        port-security-enabled: false
```
Creating a new VNF Package from CLI

- The management interface for our VNF will be the agw-mgmt CP.

```
mgmt-interface:
  cp: agw-mgmt
```

- Our Magma AGW VDU needs some information to be passed via a cloud-init file, which we will review later.

```
vdu:
  - id: magma-agw-vdu
    ... 
    cloud-init-file: magmaagw_init
```
Creating a new VNF Package from CLI

- A second VDU is added, for the srsLTE eNodeB/UE emulator

```yaml
vdus:
  ...
  - id: srsLTE-vdu
    name: srsLTE-vdu
    description: srsLTE-vdu
    count: 1
    cloud-init-file: srslte_init
  vm-flavor:
    vcpu-count: 4
    memory-mb: 6144
    storage-gb: 100
  image: 'srsLTEzmqRF_hf9'

interfaces:
  - name: eth0
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
      external-connection-point-ref: srsLTE-mgmt
      mgmt-interface: true
  - name: eth1
    type: INTERNAL
    virtual-interface:
      type: PARAVIRT
      internal-connection-point-ref: srsLTE-s1
  internal-connection-point:
    - id: srsLTE-s1
      name: srsLTE-s1
      short-name: srsLTE-s1
```
The internal VLD, for the S1 network, must be defined in the VNFD. An IP Profile is used to force a specific IP addressing.

```yaml
internal-vld:
  - id: internalS1
    name: internalS1
    short-name: internalS1
    type: ELAN
    ip-profile-ref: internalS1
  internal-connection-point:
    - id-ref: agw-s1
      ip-address: 192.168.100.254
    - id-ref: srsLTE-s1
      ip-address: 192.168.100.10

ip-profiles:
  - name: internalS1
    description: S1 test network
    ip-profile-params:
      ip-version: ipv4
      subnet-address: 192.168.100.0/24
    dhcp-params:
      enabled: true
```
Finally, the external connection points that the VNF will expose, are defined.

```
connection-point:
- name: agw-mgmt
- name: agw-sgi
- name: srsLTE-mgmt
```

We are exposing the two management ports of both VDUs, and the SGi interface, to the Network Service.
Building a NS Package from scratch

We can use the graphical composer for the NSD, then download the package to add other artifacts, but it will be faster through the CLI.

- Compose a new NS
- Create new Package
Creating a new NS Package from CLI

• Use the command line to create the complete structure of the package, modify as desired with an editor.

```
  osm package-create --base-directory ~/magma --vendor OSM ns hackfest_magma-agw-enb
```

• The final contents we need for this section are place in the following folder:
  `/home/ubuntu/examples/01-multivdu/hackfest_magma-agw-enb_nsd/hackfest_magma-agw-enb_nsd.yaml`
Creating a new NS Package from CLI

- Two options:

  1. View the desired contents and replace your `hackfest_magma-agw-enb_nsd.yaml` file, section by section.

```
cat /home/ubuntu/examples/01-multivdu/hackfest_magma-agw-enb_nsd/hackfest_magma-agw-enb_nsd.yaml
```

  2. [Faster] Copy all the contents from the `examples` directory into your VNF folder

```
cp -a /home/ubuntu/examples/01-multivdu/hackfest_magma-agw-enb_nsd/* ~/magma/hackfest_magma-agw-enb_ns/
```
Creating a new NS Package from CLI

- The first important part is the ‘constituent-vnfd’ section, which will specify which VNFs form our NS.

```json
constituent-vnfd:
- member-vnf-index: 'MagmaAGW+srsLTE'
  vnfd-id-ref: hackfest_magma-agw-enb_vnfd
```
Creating a new NS Package from CLI

- Our management VLD will connect all the external management CPs exposed at our VNF

```yaml
vld:
  - id: mgmt
    name: mgmt
    short-name: mgmt
    type: ELAN
    mgmt-network: true
    vnfd-connection-point-ref:
      - member-vnf-index-ref: 'MagmaAGW+srsLTE'
        vnfd-id-ref: hackfest_magma-agw-enb_vnfd
        vnfd-connection-point-ref: agw-mgmt
      - member-vnf-index-ref: 'MagmaAGW+srsLTE'
        vnfd-id-ref: hackfest_magma-agw-enb_vnfd
        vnfd-connection-point-ref: srsLTE-mgmt
```
Finally, our SGi VLD will connect the Magma AGW VDU to an existing network called “sgi” in our VIM.

```yaml
vld:
  ...
  - id: sgi
    name: sgi
    short-name: sgi
    type: ELAN
    mgmt-network: false
    vim-network-name: sgi
    vnfd-connection-point-ref:
      - member-vnf-index-ref: 'MagmaAGW+srsLTE'
        vnfd-id-ref: hackfest_magma-agw-enb_vnfd
        vnfd-connection-point-ref: agw-sgi
```
Automating Day-0 configuration through cloud-init
What is cloud-init and what can it be used for?

- It is a Linux package used to automate initial configuration of a VM

- VM requirements:
  - Cloud-init package
  - Cloud-init configuration (data source) via /etc/cloud/cloud.cfg
    - Config drive
    - Openstack metadata server
    - ...

- What can be done?
  - Setting a default locale
  - Setting an instance hostname
  - Generating instance SSH private keys
  - Adding SSH keys to a user’s .ssh/authorized_keys so they can log in
  - Setting up ephemeral mount points
  - Configuring network devices
  - Adding users and groups
  - Adding files

- Cloud-init is available in Linux VMs and might be supported in other OS

- Not all VIMs support cloud-init via a metadata server

Let's explore the Cloud-init files

```bash
~/magma/hackfest_magma-agw-enb_vnf/cloud_init/magmaagw_init
#cloud-config
runmd:
  # deleting default mgmt route to Internet
  - route delete -net 0.0.0.0/0 gw 172.21.251.254
  # adding specific ETSI HIVE mgmt segments through mgmt network
  - route add -net 10.100.0.0/16 gw 172.21.251.254
  - route add -net 10.101.0.0/16 172.21.251.254
  - route add -net 172.21.0.0/16 gw 172.21.251.254
  - route add -net 172.22.0.0/16 gw 172.21.251.254
  - route add -net 192.168.170.0/24 gw 172.21.251.254
  # adding specific ORCH_IP through mgmt network
  - route add -host {{ orch_ip }}/32 gw 172.21.251.254
  # adding new default route to VyOS PNF
  - route add -net 0.0.0.0/0 gw 192.168.239.7
  # adding new specific routes to reach MetalLB ranges (Squid and other svcs) through VyOS PNF
  - route add -net 172.21.250.0/24 gw 192.168.239.7
  - route add -net 172.21.251.0/24 gw 192.168.239.7
```

- From the AGW, we are removing the default route and pointing it towards the data plane interface (router at the SGI, at 192.168.239.7)
- We are also passing a route towards the Magma Orc8r, through the management port, the IP is parametrized!
Let's explore the Cloud-init files

```bash
~/magma/hackfest_magma-agw-enb_vnf/cloud_init/srslte_init

#cloud-config
password: osm2020
chpasswd: { expire: False }
ssh_pwauth: True
runcmd:
  - route add -net 10.0.0.0/8 gw 172.21.251.254
  - route add -net 172.21.0.0/16 gw 172.21.251.254
  - route add -net 172.22.0.0/16 gw 172.21.251.254
  - route add -net 192.168.170.0/24 gw 172.21.251.254
```

- From the srsLTE emulator, we are adding some specific management routes towards the management network, as we will remove the default route in a later stage (Day-1 primitive).
- We are also setting a fixed password, ‘osm2020’, for the default ‘ubuntu’ user.
Packaging and instantiation
Building, validating and uploading packages

• Once finished, you can build and upload the NS/VNF Package to OSM with the following commands.

  osm nfpkg-create ~/magma/hackfest_magma-agw-enb_vnf
  osm nspkg-create ~/magma/hackfest_magma-agw-enb_ns

• This single command will:
  • **Validate** the package according to the Information Model.
  • **Build** the package.
  • **Upload** the package to OSM.

  osm nfpkg-list
  osm nspkg-list
Instantiation parameters

• Prepare any parameter you want to pass during instantiation.

In this case, we will prepare a ‘params.yaml’ file that will pass some parameters we will need during the following tests.

```yaml
additionalParamsForVnf:
  - member-vnf-index: 'MagmaAGW+srsLTE'
    additionalParams:
      agw_id: 'agw_01'
      agw_name: 'AGW1'
      orch_ip: '172.21.251.x'  ## change this to your assigned address
      orch_net: 'osmnet'
```
Launch your first instance

• With your NS and VNF package ready, you can proceed to instantiation.

```
osm ns-create --ns_name magmaAGW_x --nsd_name hackfest_magma-agw-enb_nsd --vim_account etsi-openstack-x --config_file params.yaml```

**NS Instances**

<table>
<thead>
<tr>
<th>Name</th>
<th>Identifier</th>
<th>Nsd name</th>
<th>Operational Status</th>
<th>Config Status</th>
<th>Detailed Status</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>magmaAGW_x</td>
<td>89dfb9c9-e846-411f-a48e-0d0c0e5a4f97</td>
<td>hackfest_magma-agw-enb_nsd</td>
<td>running</td>
<td>configured</td>
<td>Done</td>
<td></td>
</tr>
</tbody>
</table>

Showing 1 to 1 of 1 entries