PNF / HNF - Definitions

- **PNF**: Physical network function. It refers to a HW box that provides a networking function. Routers, firewalls, load balancers, etc.

- **PDU**: Physical deployment unit. It refers to the “instance” of the physical appliance that will be incorporated to a Network Service instance.

- **HNF**: Hybrid network function: Network function composed of both physical and virtual elements.

In OSM, there are no fundamental differences between a VNF, a PNF or a Hybrid Network Function (HNF)

In those cases where we want to define NS packages consisting of PNF packages or HNF packages, **OSM needs to be instructed about the available PDUs**.
PNFD/HNFD vs PDU

PNFD/HNFD
...
  pdu
  - type
    - interfaces
      - name
      - mgmt
    - name
  config
...

PDU
  name
  type
  vim_accounts
  shared
  interfaces
  - ip-address
    mgmt
    name
  - ip-address
...

Deployment
Let’s orchestrate the PNF!

Both Internet exit and management

© ETSI 2020
Tell OSM about the PDU instance(s)

Via the OSM CLI

a) Create a file that describes the PDU (for example, pdu.yaml)

```yaml
name: router01
description: VyOS Router
type: gateway
vim_accounts: [ YOUR_VIM_ID_HERE ]
shared: false
interfaces:
  - name: eth0
    ip-address: 172.21.250.200
    vim-network-name: osm-ext
    mgmt: true
  - name: eth1
    ip-address: 192.168.239.7
    mgmt: false
```

b) Create the PDU in OSM:

```
osm pdu-create --descriptor_file pdu.yaml`
```
(1) Tell OSM about the PDU instance(s)

Or via the OSM GUI

<table>
<thead>
<tr>
<th>New PDU</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name *</td>
<td>router01</td>
</tr>
<tr>
<td>PDU type *</td>
<td>gateway</td>
</tr>
</tbody>
</table>

**Vim Accounts **

- x esi-openstack

**Interfaces:**

<table>
<thead>
<tr>
<th>Name</th>
<th>IP</th>
<th>Net name</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>172.21.250.200</td>
<td>osm-ext</td>
</tr>
<tr>
<td>Mgmt</td>
<td>True</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>IP</th>
<th>Net name</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth1</td>
<td>192.168.239.7</td>
<td>sgi</td>
</tr>
<tr>
<td>Mgmt</td>
<td>False</td>
<td></td>
</tr>
</tbody>
</table>
(2) Now create the PNFD

You will create it in the next section. The PNF uses the same model as a VNF and will look like this:

```
vnfd-catalog:
  vnfd:
    - connection-point:
        - name: gateway_public
          type: VPORT
          description: Gateway PNF
          id: hackfest_gateway_vnfd
          mgmt-interface:
            cp: gateway_public
            name: hackfest_gateway_vnfd
            short-name: hackfest_gateway_vnfd
    vdu:
      - description: gateway_pdu
        id: gateway_pdu
        interface:
          - external-connection-point-ref: gateway_public
            name: eth0
            type: EXTERNAL
        pdu-type: gateway
        vnf-type: gateway
        vnf-configuration:
```
(3) Next, include it in your NSD

You will need to modify your NSD to include your new PNF. You will do it in the next section and it will become an Hybrid NS!

```yaml
constituent-vnfd:
  - member-vnf-index: 'MagmaAGW+srsLTE'
    vnfd-id-ref: hackfest_magma-agw-enb_vnfd
  # - member-vnf-index: 'VYOS-PNF'
  #   vnfd-id-ref: hackfest_gateway_vnfd
 connection-point:
  - name: nsd_cp_mgmt
    vld-id-ref: mgmt
  - name: nsd_cp_sgi
    vld-id-ref: sgi
    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
  # - member-vnf-index-ref: 'VYOS-PNF'
  #   vnfd-id-ref: hackfest_gateway_vnfd
  #   vnfd-connection-point-ref: gateway_public
```
(4) Don’t forget to add automation!

PNF orchestration, since it is an element that already exists in the environment, is all about automating its configuration in the context of our Network Service.

Stay tuned for the **Automating Day 1 & 2 PNF Operations with OSM Primitives** session!