Placement Optimization
Lars-Göran Magnusson (Arctos Labs)
Introduction to Placement Optimization
The purpose with Placement Optimization

- There are many VIMs (NFV nodes) that can host the NFs
- How to make best use of the available NFV infrastructure for a service instance?
  - Deploy a VNF as close to consumer as it has to be
  - Deploy a VNF as far away that it can be
  - Deploy a VNF to reduce transport load
- Placement in context of OSM is the process of deciding **which VNF goes into which VIM**
- Optimization is subject to:
  - Cost of compute in VIMs
  - Cost of links for NS interworking
  - Constraints in NS interworking (Latency, Jitter) – if there are any
- Placement feature makes this process **Automatic & Optimal**
The optimization process

Computation of optimal placement of VNFs over VIMs by matching NS specific requirements to infrastructure availability and run-time metrics, while considering cost of compute/network.

Placement function
- Will consider all VIM’s available to the user
- Will make sure constraints are met – if there are any
- Will optimize Cost (the Criteria)

I.e. select the distribution of VNFs that fulfills constraints at the lowest possible cost
- Modeled as a constraints optimization problem
Pin a VNF to a VIM

- Sometimes we have absolute constraints for which VIM a VNF must be hosted on
  - the VIM with a specific VNF (e.g. P-GW)
  - the VIM with connectivity to a PNF
  - a CPE (customer location)
- It is therefore possible to pin the VNFs to a specific VIM

In this example VNF#3 must go to VIM#3 to satisfy the purpose with the service instance.

VNF#1, VNF#2 are subject to placement optimization => this is what Placement is all about – finding out where VNFs should (or must) be deployed in a multi-VIM NFVI.
Some different scenarios

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Pinning</th>
<th>Constraints</th>
<th>Pinning</th>
<th>Constraints</th>
<th>Pinning</th>
</tr>
</thead>
<tbody>
<tr>
<td>a:</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>VIM#3</td>
<td></td>
</tr>
<tr>
<td>b:</td>
<td>-</td>
<td>latency=15</td>
<td>-</td>
<td>VIM#4</td>
<td></td>
</tr>
<tr>
<td>c:</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>VIM#3</td>
<td></td>
</tr>
</tbody>
</table>

Topology & Cost

- Global DC VIM#1
  - Cost: 10
  - Latency: 30
  - Cost: 30
  - Latency: 20
- Local DC VIM#2
  - Cost: 20
- CP VIM#3
  - Cost: 50
- CP VIM#4
  - Cost: 50
Optimization criteria

1. **Cost** optimization only

2. **Cost** optimization with Latency constraint

3. **Utilization** optimization with Latency constraint

4. **Cost** optimization with Capability constraint
Install, configure and invoke PLA in OSM
Install and Configure PLA

- PLA is optional
  - Install with --pla
  - New in rel EIGHT, basic functionality initially
- You need two configuration files
  - vnf_price_list.yaml
  - pil_price_list.yaml
- The configuration files are copied to the PLA container using the following commands:
  ```bash
  $ docker cp vnf_price_list.yaml $(docker ps -qf name=osm_pla):/placement/.
  $ docker cp pil_price_list.yaml $(docker ps -qf name=osm_pla):/placement/.
  
  Note: Don’t copy the files as hackfest participant, it’s already done!
  ```

Note: In current OSM release the link characteristics are hard coded into this file, in future releases this data should be retrieved from the infrastructure by monitoring mechanisms.

```yaml
vnfd: hackfest_vnfd
  hackfest:
    prices:
      vim_name: etsi-openstack
      price: 5
    - vim_url: http://172.21.7.5:5000/v3
      vim_name: etsi-openstack
      price: 1
  openstack:
    prices:
      vim_name: etsi-openstack
      price: 5
    - vim_url: http://172.21.7.5:5000/v3
      vim_name: etsi-openstack
      price: 1
  lowcost:
    prices:
      vim_name: etsi-openstack
      price: 5
    - vim_url: http://172.21.7.5:5000/v3
      vim_name: etsi-openstack
      price: 1

admin:
  prices:
    vim_name: etsi-openstack
    price: 5
  - vim_url: http://172.21.7.5:5000/v3
    vim_name: etsi-openstack
    price: 1

pil:
  - pil_description: Link between vim1 and vim2
    pil_price: 5
    pil_latency: 10
    pil_jitter: 2
    pil_endpoints:
      - etsi-openstack
      - etsi-openstack-lowcost
```

Note: In current OSM release the link characteristics are hard coded into this file, in future releases this data should be retrieved from the infrastructure by monitoring mechanisms.
Invoke PLA

- Automatic placement is optional, invoked by the user at instantiate of Network Service

| Request Placement Cost Optimization | --config '{ placement-engine: PLA }' |
| Request Placement Cost Optimization with pinning of specified VNF | --config '{placement-engine: PLA, vnf: [{member-vnf-index: "1", vim_account: OpenStack3}]}' |
| Request Placement Cost Optimization with VLD Constraints | --config '{placement-engine: PLA, placement-constraints: {vld-constraints: [[id: vld_1, link-constraints: {latency: 120, jitter: 20}], {id: vld_2, link-constraints: {jitter: 20}}]}}' |
Hands-on: Placement of the Magma AGW + emulator VNF
Objective: create a new slice, sharing the Magma orchestrator, automatically deployed to another DC.
Launch a 2nd slice

- Create another VIM

  osm vim-create --name osm_hackfest_1_lowcost --user osm_hackfest_\x --password osm_hackfest_\x --auth_url http://172.21.247.4:5000/v3 --tenant osm_hackfest_\x --account_type openstack --config '{management_network_name: osm-ext}'

  The vim name is important, it must match content of the vnf_price_list.yaml file

  --user, --password and --tenant follows your personal settings for the hackfest

  Don’t forget the additional configuration

  Another VIM url

- List the vims, and collect the new vim uuid, we need it in the next step

  osm vim-list
Launch a 2nd slice

- Register the PDU to the 2\textsuperscript{nd} vim account

1) Edit pdu.yaml

   - Edit pdu.yaml
   - Launch the pdu-create command

   ```
   osm pdu-create --descriptor_file pdu.yaml
   ```

2) Launch the pdu-create command

   - Note: You may also use the GUI (Instances $\rightarrow$ PDU Instances) to register the PDU
Launch a 2nd slice

- Prepare for PLA support – modify the configuration file
  - make a copy of params_slices.yaml

Uncomment placement-engine: PLA
Uncomment wimAccountId: False

Need another agw_id, agw_name e.g. 101

- Create the slice

```
oms nsi-create --nsi_name magma_slice_2 --nst_name magma_slice_hackfest_nst \  --config_file params_slices2.yaml --ssh_keys ~/_.ssh/id_rsa.pub --vim_account hackfest
```

Use your ordinary vim_account

---

© ETSI
Launch a 2nd slice

- Check where the vnf ended up
  
  osm vnf-list

- `vim_account_id` should correspond to `hackfest-lowcost` for the new slice

- same Magma `orc8r` as before

- You may configure and send traffic over the new slice

- Clean up: delete the slice
  
  osm nsi-delete `<nsi_name>` or `<nsi_id>`