OSM Ecosystem Day

Automatic Placement of VNFs

Lars-Göran Magnusson (Arctos Labs)
Placement Optimization - Outline

• What do we mean by Placement?
• Introduction to Placement Optimization in OSM
• Achieving different types of Optimization
• Use case examples for Placement Optimization
• Invoking Placement
• Demonstration
What do we mean by Placement Optimization?

- Placement in context of OSM is the process of deciding **which VNF goes into which VIM**
- Optimal 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**

Business Service Basic Architecture, from OSM Deployment and Integration WP, Feb 2020
The Optimization Process

- 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 option that fulfills constraints at the lowest possible cost
  - Modeled as a constraints optimization problem

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.
The PLA component in OSM

- Basic functionality initially
- Automatic placement is optional, invoked by the user at instantiate of Network Service
  - `--config '{placement-engine: PLA, placement-constraints: {}, ...}'`
- Constraints given in the instantiation request
  - Open issue: Should NSD based constraints be supported?
  - Will consider placement over the VIMs available to the user
- Interacts with LCM, Common Services

- New component
  - Optional, install with `--pla`
Placement optimization examples

1. **Cost** optimization only

   - Constraint: Nothing
   - Criteria: Placement
   - Cost: Cost

2. **Cost** optimization with Latency constraint

   - Constraint: Latency / Jitter
   - Criteria: Placement
   - Cost: Cost

3. **Utilization** optimization with Latency constraint

   - Constraint: Latency
   - Criteria: Placement
   - Cost: Cost

4. **Cost** optimization with Capability constraint

   - Constraint: Capability (e.g. CPU / GPU etc)
   - Criteria: Placement
   - Cost: Cost

© ETSI 2020
Examples of use cases

**UPF supporting Low-latency**
- Placement of UPF close to customer to achieve latency constraint
- Deploy as close as it has to be

**Transport optimization (cost) for Application components**
- Placement of Application components close to the source of data to reduce transport cost / load

**Compute cost optimization for slicing**
- Placement of CN VNF’s at most cost effective compute
- Deploy as far away as it can be
**Invoking Placement**

1. Request Placement Cost Optimization
   ```
   --config '{placement-engine: PLA}'
   ```

2. Request Placement Cost Optimization with pinning of specified VNF
   ```
   --config '{placement-engine: PLA, 
   vnf: [{member-vnf-index: "1", vim_account: OpenStack3}]}'
   ```

3. 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}}]}}'
   ```

4. Combo of 2 and 3
   ```
   --config '{placement-engine: PLA, 
   vnf: [{member-vnf-index: "1", vim_account: OpenStack4}], 
   placement-constraints: {vld-constraints: 
   [{id: vld_1, link-constraints: {latency: 15}}]}}'
   ```

*Note: GUI is also supported, with or without YAML file*
VNF Pinning

- Ability to “pin” a VNF to e.g.:
  - the VIM with a specific VNF (e.g. P-GW)
  - the VIM with connectivity to a PNF
  - a CPE (customer location)

Example 1: Auto Auto VIM#3
Example 2: VIM#2 Auto Auto
Example 3: Auto Auto Auto

Auto implies there is no VIM specified, this placement is therefore 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
Demo introduction

### Cost
- Scenario a: VIM#3
  - Pinning: Auto
  - Constraints: Auto
- Scenario b: VIM#4
  - Pinning: Auto
  - Constraints: Auto, latency=15

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

**Demo scenarios outline**
Live Demo
We want to assist...

- We would appreciate community feedback to evolve the placement feature
- We would be happy to assist You in applying placement in your next project

mats.eriksson@arctoslabs.com
lars-goran.magnusson@arctoslabs.com
OSM Slack: @Lars-Göran Magnusson