Internet Research Task Force (IRTF)   R. Krishnan
Internet Draft   N. Figueira
Category: Experimental   Brocade
                                              Dilip Krishnaswamy
                                                     IBM Research
                                                     D. R. Lopez
                                                 Telefonica I+D
                                                   Steven Wright
                                                            AT&T

Expires: April 2015   October 20, 2014

       **NFVIaaS Architectural Framework for Policy Based Resource Placement
                              and Scheduling**

              draft-krishnan-nfvrg-policy-based-rm-nfviaas-01

Abstract

   One of the goals of Network Functions Virtualization (NFV) is to
   offer the NFV infrastructure as a service to other SP customers -
   this is called NFVIaaS. Virtual Network Function (VNF) deployment in
   this paradigm will drive support for unique placement policies,
   given VNF's stringent service level specifications (SLS) required by
   customer SPs. Additionally, NFV DCs often have capacity, energy and
   other constraints - thus, optimizing the overall resource usage
   based on policy is an important part of the overall solution. The
   purpose of this document is to depict an architectural framework for
   policy based resource placement and scheduling in the context of
   NFVIaaS.

Status of this Memo

   The list of current Internet-Drafts can be accessed at
   http://www.ietf.org/ietf/1id-abstracts.txt.

   The list of Internet-Draft Shadow Directories can be accessed at
   http://www.ietf.org/shadow.html.

   This Internet-Draft will expire in April 2015.

Copyright Notice

Conventions used in this document

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in RFC 2119.

Table of Contents

1. **Introduction**

   One of the goals of NFV [ETSI-NFV-WHITE] is to offer the NFV
   infrastructure as a service to other SP customers - this is called
   NFVIaaS [ETSI-NFV-USE-CASES]. In this context, it may be desirable
   for a Service Provider to run virtual network elements (e.g.,
   virtual routers, virtual firewalls, and etc. - these are called
   Virtual Network Functions - VNF) as virtual machine instances inside
   the infrastructure of another Service Provider. In this document, we
   call the former a customer SP and the latter an NFVIaaS SP.

   There are many reasons for a customer SP to require the services of
   an NFVIaaS SP, including: to meet performance requirements (e.g.,
   latency or throughput) in locations where the customer SP does not
   have physical data center presence, to allow for expanded customer
   reach, regulatory requirements, and etc.

   As VNFs are virtual machines, their deployment in such NFVIaaS SPs
   would share some of the same placement restrictions (i.e., placement
   policies) as those intended for Cloud Services. However, VNF
   deployment will drive support for unique placement policies, given
   VNF's stringent service level specifications (SLS) required/imposed
   by customer SPs. Additionally, NFV DCs often have capacity, energy
   and other constraints - thus, optimizing the overall resource usage
   based on policy is an important part of the overall solution.

   The purpose of this document is to depict an architectural framework
   for policy based resource placement and scheduling in the context of
   NFVIaaS.

2. **NFVIaaS Architectural Framework for Policy Based Resource Placement
   and Scheduling**

   The policy engine performs policy-based resource placement and
   scheduling of Virtual Machines (VMs) in support for NFVIaaS. It
   determines optimized placement and scheduling choices based on the
   constraints specified in the policy. Figure 1 depicts the NFVIaaS
   Architectural Framework for Policy Based resource placement and
   scheduling.

   In one instantiation of this architecture, the policy engine would
   interface with the Measurement Collector to periodically retrieve
   instantaneous per-server CPU utilization, which it would then use to
   compute a table of per-server average CPU utilization. In an
   alternative instantiation of this architecture, the measurement

collector could itself compute per-server average CPU utilization.
The latter approach reduces overhead, since it avoids too frequent
pulling of stats from Ceilometer. The policy engine evaluates such
policies based on an event trigger or a based on a programmable
timer.

Other average utilization parameters such as VM CPU utilization, VM
Memory utilization, VM disk read IOPS etc. could be used by the
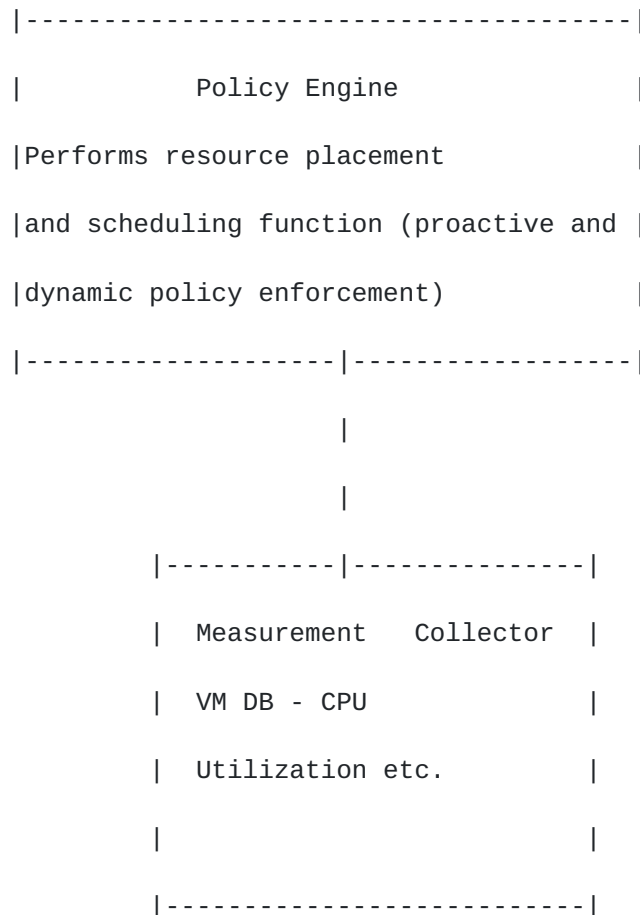policy engine to enforce other types of placement policies.

```
               |----------------------------------------|

               |             Policy Engine              |

               |Performs resource placement             |

               |and scheduling function (proactive and  |

               |dynamic policy enforcement)             |

               |-------------------|------------------|


                                   |

                                   |

                   |-----------|---------------|

                   |  Measurement   Collector  |

                   |  VM DB - CPU               |

                   |  Utilization etc.         |

                   |                           |

                   |---------------------------|
```

Figure 1: NFVIaaS Architecture for Policy Based Resource Placement
and Scheduling

In an ETSI NFV Architectural Framework [ETSI-NFV-ARCH][NFV-MANO-
SPEC], Policy Engine is part of the Orchestrator and the Measurement
Collector is part of the Virtual Infrastructure Manager (VIM).

[3](#). System Analysis in a OpenStack Framework

Consider an NFVIaaS SP that owns a multitude of mini NFV data
centers managed by OpenStack [OPENSTACK] where,

- The Policy Engine function is performed by OpenStack Congress
   [OPENSTACK-CONGRESS-POLICY-ENGINE]

- The Measurement Collector function is performed by OpenStack
   Celiometer [OPENSTACK-CELIOMETER-MEASUREMENT]

Exemplary Mini NFV DC configuration:

An exemplary mini NFV DC configuration is depicted below.

- There are 210 physical servers in 2U rack server configuration
spread over 10 racks.

- There are 4 types of physical servers each with a different
system configuration and from a particular manufacturer. It is
possible that the servers are from the same or different
manufacturer. For the purpose of this example, server type 1 is
described further. Server type 1 has 32 virtual CPUs and 128GB DRAM
from manufacturer x. Assume 55 physical servers of type 1 per mini
NFV DC.

- There are 2 types of instances large.2 and large.3, which are
described in the table below. Each parameter has a minimum guarantee
and a maximum usage limit.

Instance Type	Virtual CPU Units Minimum Guarantee /Maximum Usage	Memory (GB) Minimum Guarantee /Maximum Usage	Minimum/Maximum Physical Server Utilization (%)
large.2	0/4	0/16	0/12.5
large.3	0/8	0/32	0/25

Table 1: NFVIaaS Instance Types

For the purpose of this example, the Mini NFV DC topology is considered static -- the above topology, including the network interconnection, is available through a simple file-based interface.

Policy 1 (an exemplary NFV policy):

Policy 1 is an exemplary NFV policy. In a descriptive language, Policy 1 is as follows - "For physical servers of type 1, there can be at most only one physical server with average overall utilization less than 50%."

The goal of this policy is to address the energy efficiency requirements described in the ETSI NFV Virtualization Requirements [ETSI-NFV-REQ].

Various Module Interactions for Policy 1:

The various module interactions with respect the architectural framework in Figure 1 for Policy 1 is described below.

The policy calls for the identification of servers by type. OpenStack Congress would need to support server type, average CPU utilization, and be able to support additional performance parameters (in the future) to support additional types of placement policies. OpenStack Congress would run the policy periodically or based on events such as deleting/adding VMs etc. Initially, we could use a periodic timer based approach. In case OpenStack Congress detects a violation, it determines optimized placement and scheduling choices so that the policy is not violated.

OpenStack Congress could interface with OpenStack Celiometer to periodically retrieve instantaneous per-server CPU utilization, which it would then use to compute a table of per-server average CPU utilization. Alternatively, OpenStack Celiometer could itself compute per-server average CPU utilization which could be used by OpenStack Congress.

The proposed module interactions in this NFVIaaS placement policy example are as depicted in the architectural framework in Figure 1.

A key goal of Policy 1 above is to ensure that servers are not kept under low utilization, since servers have a non-linear power profile and exhibit relatively higher power wastage at lower utilization. For example, in the active idle state as much as 30% of peak power is consumed. At the physical server level, instantaneous energy consumption can be accurately measured through IPMI standard. At a customer instance level, instantaneous energy consumption can be

approximately measured using an overall utilization metric, which is a combination of CPU utilization, memory usage, I/O usage, and network usage. Hence, the policy is written in terms of overall utilization and not power usage.

For an exemplary maximum usage scenario, 53 physical servers could be under peak utilization (100%), 1 server (server-a) could be under partial utilization (50%) with 2 instances of type large.3, and 1 server (server-b) could be under partial utilization (37.5%) with 3 instances of type large.2.

When one of the large.3 instances mapped to server-a is deleted from physical server type 1, Policy 1 will be violated, since the overall utilization of server-a falls to 25%.

OpenStack Congress, on detecting the policy violation, uses various constraint based placement techniques to find the new placement(s) for physical server type 1 to address the policy violation. Constrained based placement will be explored in a convex optimization framework [CONVEX-OPT]; some of the algorithms which would be considered are linear programming [LINEAR-PROGRAM], branch and bound [BRANCH-AND-BOUND], interior point methods, equality constrained minimization, non-linear optimization etc.

Various new placement(s) are described below.

1) New placement 1: Move 2 instances of large.2 to server-a. Overall utilization of server-a - 50%. Overall utilization of server-b - 12.5%.

2) New placement 2: Move 1 instance of large.3 to server-b. Overall utilization of server-a - 0%. Overall utilization of server-b - 62.5%.

3) New placement 3: Move 3 instances of large.2 to server-a. Overall utilization of server-a - 62.5%. Overall utilization of server-b - 0%.

New placements 2 and 3 could be considered optimal, since they achieve maximal bin packing and open up the door for turning off server-a or server-b and maximizing energy efficiency.

To detect violations of Policy 1, an example of a classification rule is expressed below in Datalog, the policy language used by OpenStack Congress.

Database table exported by the Resource Placement and Scheduler for Policy 1:

The database table exported by the Resource Placement and Scheduler for Policy 1 is below.

server_utilization (physical_server, overall_util)

- Each database entry has the physical server and the calculated average overall utilization.

Policy 1 (in Datalog [DATALOG] policy language):

Policy 1 in a Datalog policy language is as follows.

error (physical_server) :-

nova [OPENSTACK-NOVA-COMPUTE]: node (physical_server, "type1"),

resource placement and scheduler: server_utilization (physical_server, overall_util < 50)

4. Related Work

A related proof of concept in ETSI NFV on placement and scheduling is [ETSI-NFV-POC-PLACEMENT].

5. Summary

6. Future Work

TBD

7. IANA Considerations

This draft does not have any IANA considerations.

8. Security Considerations

9. Contributors

Hesham ElBakoury

Huawei

Hesham.ElBakoury@huawei.com

[10](). Acknowledgements

   None.

[11](). References

[11.1](). Normative References

[11.2](). Informative References

   [ETSI-NFV-WHITE] "ETSI NFV White Paper,"
   [http://portal.etsi.org/NFV/NFV_White_Paper.pdf]()

   [ETSI-NFV-USE-CASES] "ETSI NFV Use Cases,"
   [http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_N]()
   [FV001v010101p.pdf]()

   [ETSI-NFV-REQ]   "ETSI NFV Virtualization Requirements,"
   [http://www.etsi.org/deliver/etsi_gs/NFV/001_099/004/01.01.01_60/gs_N]()
   [FV004v010101p.pdf]()

   [ETSI-NFV-ARCH]   "ETSI NFV Architectural Framework,"
   [http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_N]()
   [FV002v010101p.pdf]()

   [OPENSTACK] "OpenStack Open Source Software,"
   [https://www.openstack.org/]()

   [OPENSTACK-CONGRESS-POLICY-ENGINE] "A policy as a service open
   source project in OpenStack,"
   [https://wiki.openstack.org/wiki/Congress]()

   [OPENSTACK-CELIOMETER-MEASUREMENT] "OpenStack Celiometer,"
   [http://docs.openstack.org/developer/ceilometer/measurements.html]()

   [OPENSTACK-NOVA-COMPUTE] "OpenStack Nova,"
   [https://wiki.openstack.org/wiki/Nova]()

   [LINEAR-PROGRAM] Dmitris Alevras and Manfred W. Padberg, "Linear
   Optimization and Extensions: Problems and Solutions," Universitext,
   Springer-Verlag, 2001.

   [BRANCH-AND-BOUND] "Fundamentals of Algorithmics," G. Brassard and
   P. Bratley.

   [NFV-MANO-SPEC] "NFV Management and Orchestration Framework
   Specification,"

http://docbox.etsi.org/ISG/NFV/Open/Latest_Drafts/NFV-MAN001v061-%20management%20and%20orchestration.pdf

[CONVEX-OPT] "Convex Optimization,"
https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf

[ETSI-NFV-POC-PLACEMENT] "ETSI NFV Proof of Concept on Placement and Scheduling,"
http://nfvwiki.etsi.org/index.php?title=Constraint_based_Placement_and_Scheduling_for_NFV/Cloud_Systems

[DATALOG] Ceri, S. et al., "What you always wanted to know about Datalog (and never dared to ask)," Knowledge and Data Engineering, IEEE Transactions on (Volume:1 , Issue: 1)

Authors' Addresses

Ram (Ramki) Krishnan
Brocade Communications
ramk@brocade.com

Norival Figueira
Brocade Communications
nfigueir@Brocade.com

Dilip Krishnaswamy
IBM Research
dilikris@in.ibm.com

Diego Lopez
Telefonica I+D
Don Ramon de la Cruz, 82
Madrid, 28006, Spain
+34 913 129 041
diego.r.lopez@telefonica.com

Steven Wright
AT&T
sw3588@att.com