***Disclaimer***

The present document has been produced and approved by the <long ISGname> (<short ISGname>) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.  
It does not necessarily represent the views of the entire ETSI membership.

ETSI GR NFV-TST 012 V0.0.5 (2020-11)

**Group REPORT**

Network Functions Virtualisation (NFV);

Testing;

VIM & NFVI Control and Management Performance Evaluation

<

Reference

DGS/NFV-xxx

Keywords

NFV, NFVI, VIM, Performance

***ETSI***

650 Route des Lucioles

F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C

Association à but non lucratif enregistrée à la

Sous-préfecture de Grasse (06) N° 7803/88

***Important notice***

The present document can be downloaded from:  
[http://www.etsi.org/standards-search](http://www.etsi.org/standards-search#Pre-defined Collections)

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at <https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:  
<https://portal.etsi.org/People/CommiteeSupportStaff.aspx>

***Copyright Notification***

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI yyyy.

All rights reserved.

**DECT**TM, **PLUGTESTS**TM, **UMTS**TM and the ETSI logo are trademarks of ETSI registered for the benefit of its Members.  
**3GPP**TM and **LTE**TM are trademarks of ETSI registered for the benefit of its Members and  
of the 3GPP Organizational Partners.  
**oneM2M** logo is protected for the benefit of its Members.  
**GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Contents

Intellectual Property Rights 4

Foreword 4

Modal verbs terminology 4

1 Scope 5

2 References 6

2.1 Normative references 6

2.2 Informative references 6

3 Definitions, symbols and abbreviations 7

4 Overview 8

4.1 Introduction 8

4.2 Background 8

4.3 Use cases 10

4.3.1 Use cases related to implementation selection 10

5 Framework for Metric and Measurement Definition 12

5.1 Performance Metric Definition Template 12

5.2 Performance Measurement Definition Template 12

6 Test Set-ups and Configuration 13

6.1 Test Setups 13

6.2 Configurations 13

7 Control and Management Performance Metric 15

7.1 Performance metrics related to compute resource control and management 15

7.1.1 Duration of compute resource allocation 15

7.1.2 Duration of compute resource termination 16

7.1.3 Duration of compute resource scaling 16

7.1.4 Duration of compute resource migration 17

7.2 Success rate of resource instances migration 17

7.3 Success rate of NFVI node evacuation 18

8 Performance Measurement 19

8.1 Performance measurement for the duration of compute resource allocation and termination. 19

8.2 Performance measurement for compute resource scaling 20

8.3 Performance measurement for compute resource migration 21

8.4 Performance measurement to measure the success rate of resource instances migration 22

8.5 Performance measurement to measure the success rate of NFVI node evacuation 23

Annex A (informative): Authors & contributors 26

History 26

# Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server ([https://ipr.etsi.org](https://ipr.etsi.org/)).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

# Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group Network Function Vistualiztion (NFV).

# Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](https://portal.etsi.org/Services/editHelp!/Howtostart/ETSIDraftingRules.aspx) (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

# 1 Scope

The present document is a group report on methods and metrics for the evaluation of VIM and NFVI control and management performance. The evaluated NFV components in the present document include the NFV Infrastructure (NFVI) and the Virtualised Infrastructure Manager (VIM). The evaluating area considered by the present document is the capability of the operation and management of virtual resources which are performed by VIM and executed by NFVI, e.g. the time for virtualization container instantiation, scaling, migration.

Based on the performance evaluation results, the present document provides guidelines for relative comparison of different implementations of VIM and NFVI. The present document also provides advises about how to choose the most suitable implementation in different scenarios.

# 2 References

## 2.1 Normative references

Normative references are not applicable in the present document.

## 2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non‑specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GS NFV-IFA 027: ”Network Functions Virtualization (NFV); Management and Orchestration; Performance Measurements Specification”.

[i.2] ETSI GS NFV-TST 001: ”Pre-deployment Testing; Report on Validation of NFV Environments and Services”.

[i.3] ETSI GR NFV-TST 004: ”Guidelines for Test Plan on Path Implementation through NFVI”.

[i.4] RFC 8172: ”Considerations for Benchmarking Virtual Network Functions and Their Infrastructure”.

[i.5] OVP test specification:” https://docs.opnfv.org/en/stable-fraser/submodules/dovetail/docs/testing/ user/testspecification/”.

[i.6] Dovetail home page:” https://wiki.opnfv.org/display/dovetail/Dovetail+Home”.

[i.7] ETSI GS NFV 003: "Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV".

[i.8] ETSI GR NFV-IFA 010: "Network Function Virtualisation(NFV); Management and Orchestration; Functional requirements specification”

# 3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ETSI GS NFV 003 [i.8] the following apply.

**Infrastructure domain:** Infrastructure domain in this document represents the combination of VIM and NFVI.

3.2 Abbreviations

For the purposes of the present document, the terms and definitions given in ETSI GS NFV 003 [i.8] the following apply.

HZ Hertz of the system clock, an operating system parameter

GB Gigabyte

MB Megabyte

# 4 Overview

## 4.1 Introduction

As described in ETSI GR NFV 003 [i.7], NFV Infrastructure (NFVI) is the key component of the NFV architecture that encompasses the hardware and software components on which virtual network functions (VNFs) are deployed. The Virtualized Infrastructure Manager (VIM) is a key component of the NFV-MANO architectural framework which is responsible for controlling and managing the NFV infrastructure (NFVI) compute, storage, and network virtual resources. The detailed functional requirements applicable to the VIM have been defined in ETSI GS NFV- IFA 010 [i.8]..

The main job of infrastructure domain performance evaluation is to measure how well those control and management functional requirements are fulfilled. In another words, it is aimed to evaluate the control and management capability of VIM. The primary assumption of performance evaluation is that those functional requirements have been realized correctly according to the corresponding functional requirements specified in ETSI GS NFV-IFA 010 [i.8]. NFV standards definition. And the interoperability requirements between VIM and NFVI have been met.

Under the current logical NFV framework, the operations for controlling and managing the NFVI resource are initiated by VIM and then executed by NFVI. So, infrastructure domain control and management performance is affected by two main parts:

* The ability of VIM to control and manage the resources
* The capability of NFVI to respond to the requests from VIM

With regard to industrial implementation, VIM and NFVI are often delivered as one products bundle. As a consequence, the above two parts are closely related and therefore it is difficult to separate them from each other during the evaluation process. Based on those considerations, VIM and NFVI are considered as a unified system under test/evaluation unity in the present document.

The performance evaluation in the present document can serve the following purposes:

1. It can be used for the relative comparison of different infrastructure domain implementations. In reality, different NFVIs can be implemented based on different virtualization technologiesand different VIMs can have different controlling and managing strategies and algorithms. Even for the same cloud management platform (e.g. Openstack), there exist different enterprise editions. There is a lack of unified and comprehensive performance metrics and methods to find out which implementation performs better. The present document provides the corresponding metrics and methods to measure the performance of different infrastructure domain implementation. It can also provide some guide for infrastructure domain selection in different scenarios. The type and size of VNFs vary in different deployment scenarios. Therefore, operators can have different preferences about the capability of infrastructure domain. The performance evaluation in the present document can help operators have a comprehensive and detailed understanding about the capability of infrastructure domain, in order to guide operators to choose the most suitable implementation for a specific deployment scenario.
2. It can be used to describe the performance requirements for both operators and vendors. The present document provides a set of metrics for performance evaluation. On the one hand, those metrics can be used by operators to quantitatively describe their own performance requirements. On the other hand, vendors can offer the benchmarking results of these metrics to demonstrate whether the infrastructure domain implementation meets performance requirements from operators.
3. It can help to position resource-related problems. When failures happen to the whole system (VNF + NFVI + NFV-MANO), it can be caused by failed resource management, wrong VNF configurations or even a breakdown of NFV-MANO. With the help of the proposed benchmarking methods in the presentis document, the operators can easily find out whether the infrastructure domain is responsible for the failure and thus improve the efficiency of troubleshooting.

## 4.2 Background

For infrastructure domain implementation, there are a lot of different choices provided by companies and communities. OpenStack is an open source virtualisation management platform which can support operators to deploy VNFs by using virtual machines (VMs) on COTS hardwareand the OPNFV community has implemented theVIM component using OpenStack.

In addition to OpenStack, there are some other commercial implementations which also use VMs for VNF deployment provided by different companies. With the development of virtualization technologies, more and more companies are considering to build a container-based platform for NFV.

In some scenarios, service providers can have strict requirements for VNF isolation based on security considerations. In order to achieve both isolation and agility, some companies and communities start looking for possible solutions to combine Kubernetes with OpenStack. For now, there are two main options for combining OpenStack with Kubernetes:

1. The first option is to deploy Kubernetes in VMs. The advantage of this method is that the Kubernetes can be quickly deployed and flexibly expanded, and the multi-tenancy of the container is indirectly realized by the multi-tenant of the virtual machine with good isolation. The disadvantages is that deploying container on VM can cause little loss to computing performance and multi-layer nesting of network may increase the delay.
2. The second option is to integrate Kubernetes with various components of OpenStack. The two communities, OpenStack and Kubernetes, are collaborating with each other to provide a complete suite of integration drives and plugins.

The present document provides metrics and methods which can be used to evaluate the performance of different infrastructure domain implementations. There are some projects in communities and industries which can be as the reference to the present document.

Project Rally in OpenStack designs the test tool to check whether OpenStack works well under high load. The following table lists the test cases provided by Rally which are related to infrastructure domain control and management performance evaluation:

Table 4-1 Reference Use Cases from Project Rally

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OpenStack Component | | Test cases | | Corresponding requirement | |
| Cinder | | Create-and-attach-volume  Create-and-delete-volume  Create-and-list-volume  Create-and-update-volume | | Related to performance evaluation for VIM’s storage resource management | |
| Create-and-delete-snapshot  Create-and-list-snapshot | | Related to performance evaluation for VIM’s virtualized resource snapshot management | |
| Swift | | Create-container-and-object-then-delete-all  Create-container-and-object-then-list-object | | Related to performance evaluation for VIM’s storage resource management | |
| Glance | | Create-and-deactivate-image  Create-and-delete-image  Create-and-get-image  Create-and-list-image  Create-and-update-image | | Related to performance evaluation for VIM’s software image management | |
| Neutron | | Create-and-delete-network  Create-and-delete-ports  Create-and-delete-routers  Create-and-delete-subnets  Create-and-show-subnets  Create-and-update-subnets | | Related to performance evaluation for VIM’s network resource management | |
| Nova | | Boot-and-block-migrate  Boot-and-delete  Boot-and-list  Boot-and-live-migrate  Boot-and-migrate | | Related to performance evaluation for VIM’s resource instance (VM for OpenStack ) management | |

The OPNFV Verified Program (OVP) [i.6] provides a series of test areas aimed to evaluate the operation of an NFV system in accordance with carrier networking needs. OPNFV implements OVP in the Dovetail project [i.7]. Each test area contains a number of associated test cases which are described in detail in the associated test specification. The following table lists the test specifications for image test, VM resource scheduling on multiple nodes test and common virtual machine life cycle events test provided by OVP:

**Table 4-2 Reference Test Specification from Project OVP**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Specification | | Test Cases | | Corresponding requirement | |
| Tempest Image test specification | | Register Image  Upload Image  Get Image | | Related to performance evaluation for VIM’s image management | |
| VM Resource Scheduling on Multiple Nodes test specification | | Schedule VM to compute nodes  Create and delete multiple server groups with same name and policy  Create and delete server group with affinity policy  Create and delete server group with anti-affinity policy  List server groups  Show server group details | | Related to performance evaluation for VIM’s resource instance (VM) management | |
| Common virtual machine life cycle events test specification | | Resize a server  Resizing a volume-backed server  Cold migrate a server  Live migrate a server | | Related to performance evaluation for VIM’s resource instance (VM) management | |

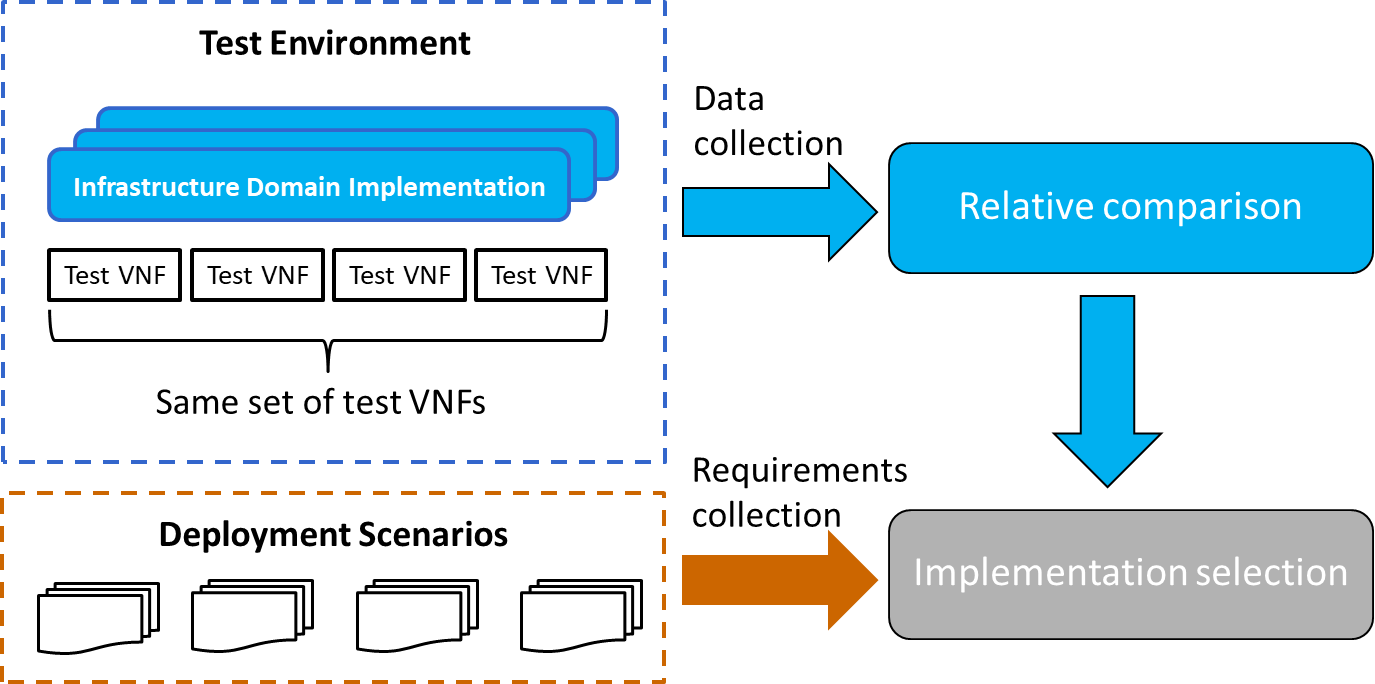
The test cases provided by OVP are designed for function test of VIM and NFVI only. They can be used as the reference for the design of test method in this document.

## 4.3 Use cases

### 4.3.1 Use cases related to implementation selection

The use case is limited to the case that multiple infrastructure domain candidates are provided for operators to select. They are installed in the same test environment with same set of testing VNFs. The set of testing VNFs is designed with simple function and different resource requirements. They will be operated (instantiated, scaled, destroyed …) in order to test the control and management plane of infrastructure domain. Continuous monitoring allows the testers (operators) to get the performance data about the infrastructure domain control and management functional behaviours in the same operator environment.

With those performance data, operators can run relative comparison between different infrastructure domains. Those performance data can also help operator to have comprehensive understanding about their capabilities from different dimensions. Combining with the requirements from different VNF deployment scenarios, operator can choose the most suitable infrastructure domain implementation. For example, in the scenarios which has a lot of micro-VNFs with short life cycle, operator could prefer the implementation with better flexibility in resource control and management. The performance data collected from the measurement can help operators to find the most suitable one.



**Figure 4-1: Use Case Related to Implementation Selection**

# 5 Framework for Metric and Measurement Definition

## 5.1 Performance Metric Definition Template

The present document provides a set of performance metrics definitions according to the following template, where each performance metric is defined in a separate clause with a separate subclause per element::

1. Background Introduction

This sub-clause contains the background information of the performance metric.

1. Name

This sub-clause contains the name of performance metric. For the metric which may already exist in industry, it may be renamed in this sub-clause.

1. Parameters

This sub-clause contains the parameters (input factors) which need to be specified in order to collect the performance metric.

1. Unit(s)

This sub-clause specifies the unit(s) of performance metric.

1. Definition

This sub-clause contains the definition of performance metric. It explains which control and management function of VIM the metric is defined to measure.

1. Method of Measurement

This sub-clause contain the method to measure the performance metric,

1. Discussion

This sub-clause contain the additional information which isn’t covered in the above sub-clause. This sub-clause is not mandatory.

## 5.2 Performance Measurement Definition Template

The present document provides a set of performance measurement definitions according to the following template, where each performance measurement is defined in a separate clause with a separate subclause per element:

1. Description

This sub-clause contains the description of the performance measurement.

1. Measurement Name

This sub-clause contains the name which is used to identify performance measurement.

1. Measurement Method

This sub-clause contains the methods in which the measurement is obtained.

1. Measured Metric(s)

This sub-clause lists the metrics which can be measured through the performance measurement

1. Trigger

This sub-clause contains the trigger which start the measurement.

1. Sources of Error

This sub-clause lists the factors which may cause failure to the performance measurement,

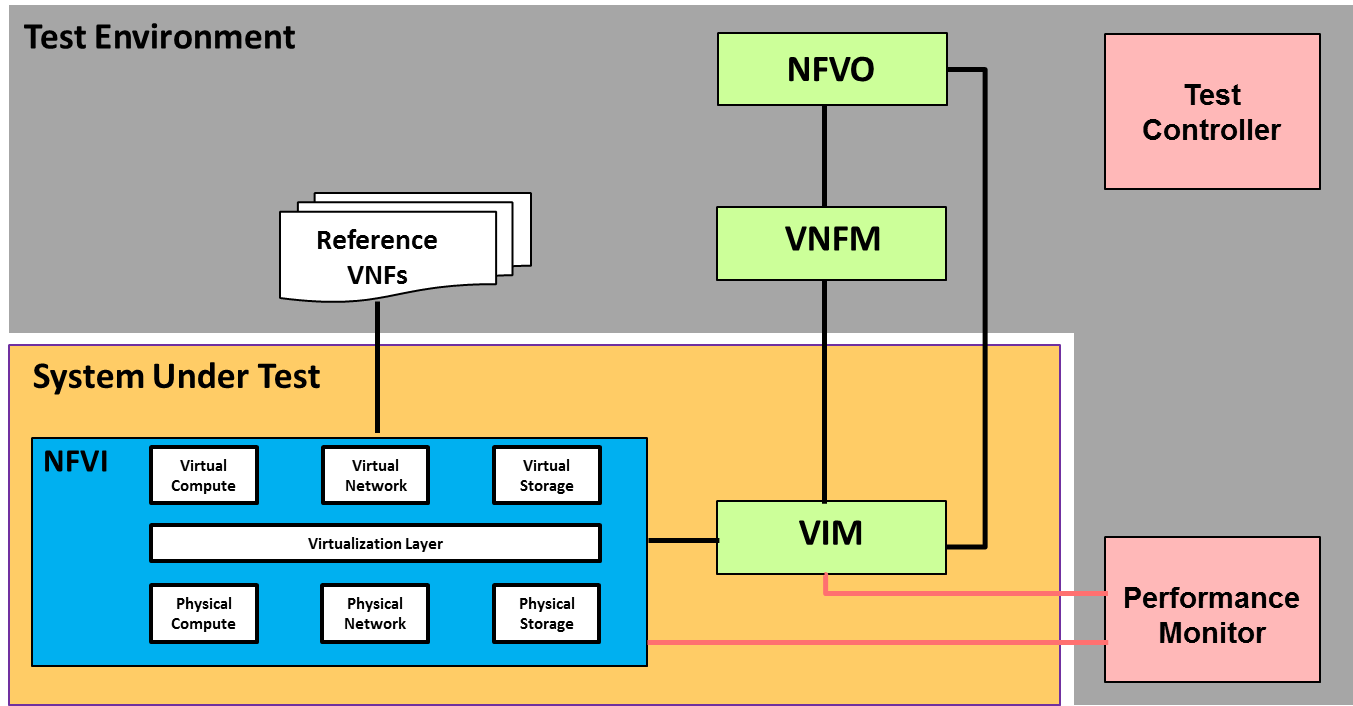
1. Measurement Procedure

This sub-clause provides the procedure manual of performance measurement.

# 6 Test Set-ups and Configuration

## 6.1 Test Setups

In the context of performance evaluation, the System Under Test (SUT) is made of infrastructure domain (VIM and NFVI) coming from different providers.



**Figure 6-1: System Under Test and Test Environment**

As illustrated in figure 6.1, the test environment consists of a reference implementation of the NFV MANO functional components (NFVO and VNFM), reference VNFs, one performance monitor and one test controller.

The testing domain is the performance of the operations and management resource instance and virtual resource. The NFV MANO functional components (NFVO and VNFM) are responsible to help VIM to trigger the control and management operations to be evaluated. The test controller in figure 6.x is used to control the whole test procedure. The performance monitor measures the performance indicators from the VIM. According to the definition of testing domain, the performance of VNFs is out of the scope. During the evaluation, a set of VNFs with simple functions and different resource requirements are used as the reference VNFs.

## 6.2 Configurations

The one purpose of performance measurement is to help operators select the most suitable implementation of the infrastructure domain. The performance measurement results are intended to be used for the relative comparison of different implementations. To ensure fairness of performance measurement, the configuration of the measured infrastructure domain all aspects of hardware, software, and reference VNF should be clearly specified..

The following hardware related configuration parameters should be specified before the measurement:

1. The number of NFVI nodes: To measure the control and management capability of the infrastructure domain, the number of NFVI nodes should be specified the same for different implementations.
2. The available resource provided by each NFVI node: The number of NFVI nodes and the available resource provided by each NFVI node are two important parameters that describe the available resource managed by VIM. For example, ten small NFVI nodes and five big NFVI nodes require different management capabilities from VIM, even their total amounts of available resources are equal. The available resources provided by each NFVI node should be specified by using the following configuration parameters:
   * The number of available CPU cores.
   * Type and size of available memory.
   * Type and size of available disk storage.
   * Type and number of physical NICs (The version of the NIC driver should also be specified. Those parameters can affect the forwarding performance of NFVI nodes, and thus measurement results).
   * Type and size of the acceleration resource (optional).

Different implementations of infrastructure domain can have the same software dependencies, e.g. OVS. The version of that software should be the same.

One flexible infrastructure domain should be able to support the deployment of different VNFs provided by different vendors. To measure the performance of the infrastructure domain independently and objectively, the reference VNFs should be used.

For different implementations of the infrastructure domain, the reference VNFs used in performance measurement should be the same. The OPNFV project provides a series of sample VNFs (DPPD-PROX, UDP\_Replay, vACL, vCGNAPT, and vFW) [i.8] which can be used as the reference VNFs in the performance measurement.

The resource control and management performance of the infrastructure domain varies in different VNF deployment scenarios because different VNFs have different resource requirements (as specified in the corresponding VDU). To measure the control and management capability of the infrastructure domain comprehensively, users should define a set of reference VNFs which have the same simple function but are based on different VDUs for the same performance measurement. This document provides one example set of VDUs:

**Table 6-1 Example Set of VDUs**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Small VDU | Medium VDU | Large VDU |
| CPU Cores | 1 cores | 2 cores | 4 cores |
| Memory | 2 G | 4G | 8G |
| Disk | 10 G | 20G | 40G |
| swImage | Software images loaded on the virtualized container should be the same for different VDUs. | | |

# 7 Control and Management Performance Metric

## 7.1 Performance metrics related to compute resource control and management

### 7.1.1 Duration of compute resource allocation

1. Background Introduction

During the instantiation of a VNF, infrastructure domains are expected to perform the operation of compute resource allocation according to the virtual resource requirements of the VNF.

Some VNFs have affinity/anti-affinity requirements. For example, for those VNFs, the infrastructure domain is expected to allocate the compute resource from the same/different physical host according to the policies.

In order to guarantee service performance, some VNFs have acceleration requirements. For VNFs that have significant memory requirements or are memory intensive, huge page memory allocation can be requested from the infrastructure domain. To ensure that all memory accesses are local to the node and thus do not consume the limited cross-node memory bandwidth, adding latency to memory accesses, NUMA support can be requested from infrastructure domain. For the VNF which have real-time constraints, the infrastructure domain is requested to assign each vCPU to one particular physical CPU during the allocation of compute resource.

The performance metric in this clause is defined to measure the agility of compute resource allocation in the above VNFs instantiation scenarios.

1. Name

Duration of compute resource allocation

1. Parameters

The following parameters are specified for the measurement of duration of compute resource allocation:

* **Compute Resource Requirement:** This parameter provides the information which specify the CPU requirement using number of cores, memory requirement and the disk requirement using GB.
* **Number of Resource Instances:** There are two optional method to specify the number of resource instance: (1) X% of max resource instance load is used to specify the number of resource instances. Max resource instance load represents the max number of the chosen type of resource instances which can be deployed in test environment. (2) Absolute value can be used to specify the number of resource instance.
* **Affinity/Anti-affinity Constraints:** This parameter specifies whether the compute resource allocation have affinity/anti-affinity constraints and which type of constraints apply:
  + No Constraints: In this scenario, nothing else is specified.
  + Affinity Constraints: In this scenario, the scope of affinity policy is specified, possible values are "Zone" and "NFVI-node".
  + Anti-Affinity Constraints: In this scenario, the scope of affinity policy is specified, possible values are "Zone" and "NFVI-node".
* **Acceleration Requirements:** This parameter specifies whether the compute resource allocation have acceleration requirement and which types of requirements apply:
  + No requirement: In this scenario, nothing else is specified.
  + Hugepage Memory Support: In this scenario, the hugepage memory size is specified. The typical values may be “2MB” or “1GB”.
  + NUMA Support: In this scenario, the information about the allocation of vCPUs and memory from same or different host NUMA nodes are specified.
  + CPU Pinning Support: In this scenario, the CPU pinning mode need to be specified, possible values are “Share mode” and “delicate mode”.

1. Unit(s)

Duration of compute resource allocation is expressed in microseconds.

1. Definition

This metric measures the time slot from VIM receiving request of compute resource allocation to VIM sending response of compute resource allocation.

1. Method of Measurement

The method of measuring this performance metric is described in clause 8.

1. Discussion

TBD

### 7.1.2 Duration of compute resource termination

1. Background Introduction

To support the deletion of VNF, infrastructure domains are expected to perform the operation of terminating the corresponding compute resource.

1. Name

Duration of compute resource termination

1. Parameters

The following parameters are specified for the measurement of duration of compute resource termination:

* **Compute Resource Requirement:** This parameter provides the information which specify the CPU requirement using number of cores, memory requirement and the disk requirement using GB.
* **Number of Resource Instances:** There are two optional method to specify the number of resource instance: (1) X% of max resource instance load is used to specify the number of resource instances. Max resource instance load represents the max number of the chosen type of resource instances which can be deployed in test environment. (2) Absolute value can be used to specify the number of resource instance.

1. Unit(s)

Duration of compute resource termination is expressed in microseconds.

1. Definition

This metric measures the time slot from VIM receiving the request of compute resource termination to VIM sending response of termination.

1. Method of Measurement

The method of measuring this performance metric is described in clause 8.

1. Discussion

TBD

### 7.1.3 Duration of compute resource scaling

NOTE: The vertical scaling (scale up/down) of VNF is not supported in the current NFV specification.

1. Background Introduction

During the lifetime of a VNF instance, for the sake of guaranteeing the performance or saving resources, the resources of a VNF instance can be resized (Scale up/down).To support the scaling of VNF, infrastructure domains perform the operation of compute resource scaling. Operations of scaling in/out resource instance are similar to the operations of allocating/terminating compute resource for VIM. They will not be included in this sub-clause.

1. Name

Duration of compute resource scaling

1. Parameters

The following parameters are supported for the measurement of duration of compute resource scaling:

* **Size of Compute Resource Instance:** This parameter provides the information about the compute resource instance which need to be scaled. The information should specify the CPU requirement using number of cores, memory requirement and the disk requirement using GB.
* **Scaling Step:** During the measurement, different scaling steps are chosen in order to evaluate the agility of control and management of SUT. For scaling up, the scaling steps may include 120%, 150%, 200% and 300%. For scaling down, the scaling steps may include 20%, 50% and 70%.

1. Unit(s)

Duration of compute resource scaling is expressed in microseconds.

1. Definition

This metric measures the time slot from VIM receiving the request of compute resource scaling to VIM sending the response when complete the compute resource scaling.

1. Method of Measurement

The method of measuring this performance metric is described in chapter 8.

1. Discussion

TBD

### 7.1.4 Duration of compute resource migration

1. Background Introduction

Resource instance can be migrated from the original physical machine to the new physical machine when some failures happen to the original physical machine. In this scenario, infrastructure domain need to perform the operation of virtualized compute resource migration.

1. Name

Duration of compute resource migration

1. Parameters

The following parameters are specified for the measurement of duration of compute resource migration:

* **Size of Compute Resource Instance:** This parameter provides the information about the compute resource instance which need to be migrated. The information should specify the CPU requirement using number of cores, memory requirement and the disk requirement using GB.
* **Migration Constraint:** This parameter specify where to migrate the resource, possible values are “a specified resource zone” or “a specified host”.

1. Unit(s)

Duration of compute resource migration is expressed in microseconds.

1. Definition

This metric measures the time slot from VIM receiving the request of compute resource migration to VIM sending the response of completing the migration.

1. Method of Measurement

The method of measuring this performance metric is described in chapter 8.

1. Discussion

TBD

## 7.2 Success rate of resource instances migration

1. Background Introduction

When failures happen to one NFVI node, the infrastructure domain is expected to allocate the available resource from rest NFVI nodes to support the migration of all VNFs from failed NFVI nodes. If the infrastructure domain can complete the migration successfully, MANO can recover the network service deployed in the failed NFVI node.

The sudden failure of NFVI nodes is unpredictable. The success rate of resource instance migration for all NFVI nodes can be used to measure the capability of the infrastructure domain to provide reliable network services. The infrastructure domain with the higher success rate of resource instances migration can achieve better performance in the auto-healing scenarios.

1. Name

Success rate of resource instances migration

1. Parameters

The following parameters are supported for the measurement of success rate of compute resource migration:

* **The deployed set of resource instances:** The deployed set of resource instances is a group of resource instances deployed according to the resource requirement of the reference VNF. The parameter provides information about the number and size of the resource instances deployed in each NFVI node. With this specified parameter, the information about the available resource distributing in each node is also specified.
* **Number of trials:** This parameter specifies the number of trials performed in one performance measurement.

1. Unit(s)

Success rate of resource instances migration is expressed in percentage.

1. Definition

This metric measures the success rate of resource instances migration for all NFVI nodes. It assumes one NFVI node failed each time to test whether the infrastructure domain can complete the migration of all resource instances from the failed NFVI node. The test is performed against every NFVI node to calculate the success rate of resource instance migration for all NFVI nodes.

1. Method of Measurement

The method of measuring this performance metric is described in clause 8.

1. Discussion

TBD

## 7.3 Success rate of NFVI node evacuation

1. Background Introduction

When some serious failures happen to one NFVI node, the infrastructure domain needs to evacuate the whole NFVI node. The infrastructure domain needs to migrate all VNFs out to recover the service deployed in this NFVI node.

The success rate of NFVI node evacuation can be used to measure the capability of the infrastructure domain to provide reliable network services. The infrastructure domain with the higher success rate of NFVI node evacuation can achieve better performance in the auto-healing scenarios.

1. Name

Success rate of NFVI node evacuation

1. Parameters

The following parameters are supported for the measurement of success rate of NFVI node evacuation:

* **The deployed set of resource instances:** The deployed set of resource instances is a group of resource instances deployed according to the resource requirement of the reference VNF. The parameter provides information about the number and size of the resource instances deployed in each NFVI node. With this specified parameter, the information about the available resource distributing in each node is also specified.
* **Number of trials:** This parameter specifies the number of trials performed in one performance measurement.

1. Unit(s)

Success rate of NFVI node evacuation is expressed in percentage.

1. Definition

This metric measures the success rate of NFVI node evacuation. Assume one NFVI node failed each time to test whether the infrastructure domain can complete the migration of all resource instances from the failed NFVI node. If all resource instances are migrated successfully, the NFVI node is considered to be evacuated successfully. Perform the test against every NFVI node to calculate the success rate of NFVI node evacuation.

1. Method of Measurement

The method of measuring this performance metric is described in chapter 8.

1. Discussion

TBD

# 8 Performance Measurement

## 8.1 Performance measurement for the duration of compute resource allocation and termination.

1. Description

To allocate the compute resource, the infrastructure domain should instantiate one resource instance which contains the virtual resource to be allocated by the reference VNF. To release the compute resource, the infrastructure domain should terminate the resource instance which contain this virtual resource. The performance measurement defined in this clause describes the procedure of measuring the duration of compute resource allocation and termination.

1. Measurement Name

Performance measurement for the duration of compute resource allocation and termination.

1. Measured Metric(s)

* Duration of compute resource allocation
* Duration of compute resource termination

1. Trigger

Test Controller triggers the VIM API consumer’s (e.g. NFVO) action of sending compute resource allocation/termination request to VIM.

1. Sources of Error

The sources of error for this benchmark are listed below:

1. Loss of synchronization of clock may affect the measurement result.
2. Latency between test monitor and the infrastructure domain may affect the measurement result.
3. Measurement Method

**Table 8-1 Performance measurement for the duration of compute resource allocation and termination**

|  |  |  |
| --- | --- | --- |
| **Test Purpose** | **Measure the duration of compute resource allocation and termination.** | |
| **Pre-test Conditions** | * The Test Setups should be implemented as described in clause 6. * Test controller can trigger the VIM API consumer’s action of sending “allocate/terminate compute resource” operation request to VIM * Test monitor can monitor VIM’s operation of receiving requests from the VIM API consumer and record the event time. * Test monitor can monitor VIM’s operation of sending responses to the VIM API consumer and record the event time. * The test configuration should be specified as described in clause 6. * The parameters for the metrics—duration of compute resource allocation and duration of compute resource termination should be specified as requires in clause 7.1.1&7.1.2. | |
|  | | |
| **Test Sequence** | **Name** | **Description** |
| Sub-Test-1: measure the duration of compute resource allocation. | | |
| Step-01 | Trigger the test | Test Controller triggers the VIM API consumer’s action of sending a compute resource allocation request to NFVO to trigger the allocation procedure. |
| Step-02 | Generate and send the request | The VIM API consumer generates and sends the “allocate compute resource” request to VIM. |
| Step-03 | Receive the request | VIM receives the “allocate compute resource” request from the VIM API consumer.  Test monitor records the event time as T1 |
| Step-04 | Instantiate one resource instance | After receiving the request from the VIM API consumer, VIM should instantiate one resource instance to allocate the compute resource. |
| Step-05 | Send the response | After completing the compute resource allocation, VIM sends the response to the VIM API consumer.  Test monitor records the event time as T2. |
| Step-06 | Calculate the duration of compute resource allocation | The duration of compute resource allocation equals: T2-T1 |
| Step-07 | Repeat Step-01 to Step-06 | Repeat Step-01 to Step-06 according to the parameters specified in Pre-Test. |
| Step-08 | Calculate the average value | Calculate the average value of the duration of compute resource allocation |
| Sub-Test-2: measure the duration of compute resource termination by using the resource instances instantiated in Sub-Test-1. | | |
| Step-09 | Trigger the test | Test Controller triggers the VIM API consumer’s action of sending a compute resource termination request to the VIM API consumer to trigger the termination procedure. |
| Step-10 | Generate and send the request | The VIM API consumer generates and sends the “terminate compute resource” request to VIM. |
| Step-11 | Receive the request | VIM receives the “terminate compute resource” request from the VIM API consumer.  Test monitor records the event time as T3 |
| Step-12 | Terminate one resource instance | After receiving the request from the VIM API consumer, VIM should terminate one resource instance to release the compute resource. |
| Step-13 | Send the response | After completing the compute resource termination, VIM sends the response to the VIM API consumer.  Test monitor records the event time as T4. |
| Step-14 | Calculate the duration of compute resource termination | The duration of compute resource termination equals: T4-T3 |
| Step-15 | Repeat Step-09 to Step-14 | Repeat Step-09 to Step-14 according to the parameters specified in Pre-Test. |
| Step-16 | Calculate the average value | Calculate the average value of the duration of compute resource termination |

## 8.2 Performance measurement for compute resource scaling

1. Description

To support VNF scaling, the infrastructure domain should scale the resource instance. The performance measurement defined in this clause describes the procedure of measuring the duration of compute resource scaling.

1. Measurement Name

Performance measurement for compute resource scaling.

1. Measured Metric(s)

* Duration of compute resource scaling

1. Trigger

Test Controller triggers the VIM API consumer’s (e.g. NFVO) action of sending the compute resource scaling request to VIM.

1. Sources of Error

The sources of error for this benchmark are listed below:

1. Loss of synchronization of clock may affect the measurement result.
2. Latency between test monitor and the infrastructure domain may affect the measurement result.
3. Measurement Method

**Table 8-2 Performance measurement for the duration of compute resource scaling**

|  |  |  |
| --- | --- | --- |
| **Test Purpose** | **Measure the duration of compute resource scaling.** | |
| **Pre-test Conditions** | * The Test Setups should be implemented as described in clause 6. * Test controller can trigger the VIM API consumer’s action of sending the “scale compute resource” operation request to VIM * Test monitor can monitor VIM’s operation of receiving requests from the VIM API consumer and record the event time. * Test monitor can monitor VIM’s operation of sending responses to the VIM API consumer and record the event time. * The test configuration should be specified as described in clause 6. * The parameters for the metric—duration of compute resource scaling should be specified as requires in clause 7.1.3. * The resource instance which need to be scaled has been instantiated successfully. | |
|  | | |
| **Test Sequence** | **Name** | **Description** |
| Step-01 | Trigger the test | Test Controller triggers the VIM API consumer’s action of sending a compute resource scaling request to the VIM API consumer to trigger the scaling procedure. |
| Step-02 | Generate and send the request | The VIM API consumer generates and sends the “scale compute resource” request to VIM. |
| Step-03 | Receive the request | VIM receives the “scale compute resource” request from the VIM API consumer.  Test monitor records the event time as T1 |
| Step-04 | Scale the resource instance | After receiving the request from the VIM API consumer, VIM should update the internal management objects for the compute resource instance and scale this resource instance. |
| Step-05 | Send the response | After completing the compute resource scaling, VIM sends the response to the VIM API consumer.  Tests monitor records the event time as T2. |
| Step-06 | Calculate the duration of compute resource scaling | The duration of compute resource scaling equals: T2-T1 |

## 8.3 Performance measurement for compute resource migration

1. Description

The performance measurement defined in this clause describes the procedure of measuring the duration of compute resource migration.

1. Measurement Name

Performance measurement for compute resource migration.

1. Measured Metric(s)

* Duration of compute resource migration

1. Trigger

Test Controller triggers the VIM API consumer’s (e.g. NFVO) action of sending the compute resource migration request to VIM.

1. Sources of Error

The sources of error for this benchmark are listed below:

1. Loss of synchronization of clock may affect the measurement result.
2. Latency between test monitor and the infrastructure domain may affect the measurement result.
3. Measurement Method

**Table 8-3 Performance measurement for the duration of compute resource migration**

|  |  |  |
| --- | --- | --- |
| **Test Purpose** | **Measure the duration of compute resource migration.** | |
| **Pre-test Conditions** | * The Test Setups should be implemented as described in clause 6. * Test controller can trigger the VIM API consumer’s action of sending the “migrate compute resource” operation request to VIM * Test monitor can monitor VIM’s operation of receiving requests from the VIM API consumer and record the event time. * Test monitor can monitor VIM’s operation of sending responses to the VIM API consumer and record the event time. * The test configuration should be specified as described in clause 6. * The parameters for the metric—duration of compute resource migration should be specified as requires in clause 7.1.4. * The resource instance which needs to be migrated has been instantiated successfully. | |
|  | | |
| **Test Sequence** | **Name** | **Description** |
| Step-01 | Trigger the test | Test Controller triggers the VIM API consumer’s action of sending a compute resource migration request to the VIM API consumer to trigger the migration procedure. |
| Step-02 | Generate and send the request | The VIM API consumer generates and sends the “migrate compute resource” request to VIM. |
| Step-03 | Receive the request | VIM receives the “migrate compute resource” request from the VIM API consumer.  Test monitor records the event time as T1 |
| Step-04 | Migrate the resource instance | After receiving the request from the VIM API consumer, VIM should update the internal management objects for the compute resource instance and migrate this resource instance. |
| Step-05 | Send the response | After completing the compute resource migration, VIM sends the response to the VIM API consumer.  Tests monitor records the event time as T2. |
| Step-06 | Calculate the duration of compute resource migration | The duration of compute resource migration equals: T2-T1 |

## 8.4 Performance measurement to measure the success rate of resource instances migration

1. Description

In migration scenarios, the infrastructure domain should migrate some resource instances from the one NFVI node to other NFVI nodes. The performance measurement defined in this clause describes the procedure of measuring the success rate of resource instances migration. This performance measurement method is designed to be used across all resource instances deployed in one NFVI node.

1. Measurement Name

Performance measurement to measure the success rate of resource instances migration

1. Measured Metric(s)

* Success rate of resource instances migration equals C1/ (C1+C2). C1 represents the number of resource instances that can be migrated successfully. C2 represents the number of resource instances which fail to be migrated.

1. Trigger

Test Controller triggers the VIM API consumer’s action of sending the resource migration request to VIM.

1. Sources of Error

The sources of error for this benchmark are listed below:

1. Loss of synchronization of clock may affect the measurement result.
2. Latency between test monitor and the infrastructure domain may affect the measurement result.
3. Measurement Method

**Table 8-4 Performance measurement to measure the success rate of resource instances migration**

|  |  |  |
| --- | --- | --- |
| **Test Purpose** | **Measure the success rate of resource instances migration.** | |
| **Pre-test onditions** | * The Test Setups should be implemented as describedin clause 6. * Test Controller can trigger the VIM API consumer’s action of sending “migrate compute resource” operation request to VIM * Test Monitor can monitor VIM’s operation of receiving requests from the VIM API consumer. * Test Monitor can monitor VIM’s operation of sending responses to the VIM API consumer. * Test Monitor can verify whether the migration operation is performed successfully according to VIM’s response to the VIM API consumer and record the number. * Test Monitor can verify whether all resource instances have been migrated out from the NFVI node. * The test configuration should be specified as described in clause 6. * The parameters for the metric—the success rate of resource instances migration should be specified as requires in clause 7.2. * The deployed set of resource instances have been instantiated successfully. | |
|  | | |
| **Test Sequence** | **Name** | **Description** |
| Step-01 | Assume one NFVI node failed | Test Controller assumes one NFVI node failed. Test Controller should trigger the VIM API consumer to migrate all resource instances out one by one from the failed NFVI node. |
| Step-02 | Trigger the test | Test Controller triggers the VIM API consumer’s action of sending a compute resource migration request to the VIM API consumer in order to trigger the migration procedure. |
| Step-03 | Generate and send the request | The VIM API consumer generates and sends the “migrate compute resource” request to VIM. |
| Step-04 | Receive the request | VIM receives the “migrate compute resource” request from the VIM API consumer. |
| Step-05 | Migrate the resource instance | After receiving the request from the VIM API consumer, VIM should update the internal management objects for the resource instance and migrate this resource instance. |
| Step-06 | Send the response | After completing the compute resource migration, VIM sends the response to the VIM API consumer. |
| Step-07 | Check the response | Test Monitor verifies whether the migration operation is performed successfully according to VIM’s response to the VIM API consumer.  If successes, Counter C1 plus 1.  If fails, Counter C2 plus 1.  Repeat Step-02 to Step-07. Migrate the next resource instance.  If all resource instances have been attempted to be migrated, Continue. |
| Step-08 | Calculate the success rate of resource instances migration | The success rate of resource instances migration equals: C1/ (C1+C2). |
| Step-09 | Reset the test environment | Undo all the migration operations that have been performed successfully. Reset the test environment. |
| Step-10 | Repeat Step-01 to Step-10 | Repeat Step-01 to Step-10 according to the parameters specified in Pre-Test. |
| Step-11 | Calculate the average value | Calculate the average value of the success rate of resource instances migration. |

## 8.5 Performance measurement to measure the success rate of NFVI node evacuation

1. Description

To recover the network service, the infrastructure domain should migrate all resource instances from one NFVI node to other NFVI nodes. The performance measurement defined in this clause describes the procedure of measuring the success rate of NFVI node evacuation. This performance measurement method is designed to be used across all NFVI nodes.

1. Measurement Name

Performance measurement to measure the success rate of NFVI node evacuation

1. Measured Metric(s)

* Success rate of NFVI node evacuation equals C1/N. C1 represents the number of the NFVI nodes which can be evacuated successfully. N represents the number of all NFVI nodes.

1. Trigger

Test Controller triggers the VIM API consumer’s action of sending the resource migration request to VIM.

1. Sources of Error

The sources of error for this benchmark are listed below:

1. Loss of synchronization of clock may affect the measurement result.
2. Latency between test monitor and the infrastructure domain may affect the measurement result.
3. Measurement Method

**Table 8.5 Performance measurement to measure the success rate of NFVI node evacuation**

|  |  |  |
| --- | --- | --- |
| **Test Purpose** | **Measure the success rate of NFVI node evacuation.** | |
| **Pre-test conditions** | * The Test Setups should be implemented as described in clause 6. * Test Controller can trigger the VIM API consumer’s action of sending “migrate compute resource” operation request to VIM * Test Monitor can monitor VIM’s operation of receiving requests from the VIM API consumer. * Test Monitor can monitor VIM’s operation of sending responses to the VIM API consumer. * Test Monitor can verify whether the migration operation is performed successfully according to VIM’s response to the VIM API consumer and record the number. * Test Monitor can verify whether all resource instances have been migrated out from the NFVI node. * The test configuration should be specified as described in clause 6. * The parameters for the metric—the success rate of NFVI node evacuation should be specified as requires in clause 7.3. * The deployed set of resource instances have been instantiated successfully. | |
|  | | |
| **Test Sequence** | **Name** | **Description** |
| Step-01 | Assume one NFVI node failed | According to the configuration, there are N NFVI nodes in the infrastructure domain.  Test Controller assumes one NFVI node failed. Test Controller should trigger the VIM API consumer to migrate all resource instances out one by one from the failed NFVI node. |
| Step-02 | Trigger the test | Test Controller triggers the VIM API consumer’s action of sending a compute resource migration request to the VIM API consumer in order to trigger the migration procedure. |
| Step-03 | Generate and send the request | The VIM API consumer generates and sends the “migrate compute resource” request to VIM. |
| Step-04 | Receive the request | VIM receives the “migrate compute resource” request from the VIM API consumer. |
| Step-05 | Migrate the resource instance | After receiving the request from the VIM API consumer, VIM should update the internal management objects for the compute resource instance and migrate this resource instance. |
| Step-06 | Send the response | After completing the compute resource migration, VIM sends the response to the VIM API consumer. |
| Step-07 | Check the response | Test Monitor verifies whether the migration operation is performed successfully according to VIM’s response to the VIM API consumer.  If successes, repeat Step-02 to Step-07. Migrate the rest resource instances one by one.  If fails, Continue.  If no resource instance left, Counter C1 plus 1. Continue. |
| Step-08 | Reset the test environment | Undo all the migration operations which have been performed successfully. Reset the test environment.  Counter C2 plus 1.  If C2 < N, Repeat Step-01 to Step 08. Assumes another NFVI node failed.  If C2 = N, Continue. |
| Step-09 | Calculate the success rate of NFVI node evacuation | The success rate of NFVI node evacuation equals: C1/N. |
| Step-10 | Reset the test environment | Undo all the migration operations that have been performed successfully. Reset the test environment. |
| Step-11 | Repeat Step-01 to Step-08 | Repeat Step-01 to Step-11 according to the parameters specified in Pre-Test. |
| Step-12 | Calculate the average value | Calculate the average value of the success rate of NFVI node evacuation. |

# Annex A (informative): Authors & contributors

The following people have contributed to this report:

**Rapporteur:**  
Mr. Cheng Huang, Huawei Technologies Co. Ltd.

**Other contributors:**

Mr. Xianming Li, Huawei Technologies Co. Ltd.

Mr. Xuliang Wang, China Telecom

# History

|  |  |  |
| --- | --- | --- |
| **Document history** | | |
| <Version> | <Date> | <Milestone> |
| V0.0.1 | October 2018 | Skeleton |
| V0.0.2 | December 2018 | Early Draft based on contributions |
| V0.0.3 | February 2019 | Early Draft based on contribution |
| V0.0.4 | May 2019 | Early Draft based on contribution |
| V0.0.5 | October 2020 | Early Draft based on contribution |