Network Functions Virtualization (NFV); Assurance; Report on Active Monitoring and Failure Detection

This DRAFT is a working document of ETSI ISG NFV. It is provided for information only and is for future development work within ETSI ISG NFV. DRAFTS may be updated, deleted, replaced, or obsoleted by other documents at any time.

ETSI and its Members accept no liability for any further use/implementation of the present DRAFT.

Do not use as reference material.

Do not cite this document other than as "work in progress."

Approved and PUBLISHED deliverables shall be obtained exclusively via the ETSI Standards search page at: http://www.etsi.org/standards-search
Reference
UNASSIGNED/PENDING

Keywords
NFV, Benchmarking, Testing

ETS

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

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the 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
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute yyyy.
All rights reserved.

DECT™, PLUGTESTS™, UMTS™ and the ETSI logo are Trade Marks of ETSI registered for the benefit of its Members. 3GPP™ and LTE™ are Trade Marks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.
GSM® and the GSM logo are Trade Marks registered and owned by the GSM Association.
Title*: Active Monitoring and Failure Detection in NFV Environments

from Source*: Spirent Communications
Contact: Gurpreet Singh

input for Committee*: NFV REL

Contribution For*: Decision
Discussion X
Information

Submission date*: 2015-9-29

Meeting & Allocation:
Relevant WI(s), or deliverable(s):

ABSTRACT: Draft for defining active monitoring framework and test methodologies for reliability, failure isolation, proactive testing to mitigate network or service failures/degradations in an NFV environment

Please note that the document structure and content may change as the group members deliver contributions to this draft.

Contents

Reproduction is only permitted for the purpose of standardization work undertaken within ETSI. The copyright and the foregoing restriction extend to reproduction in all media. ........................................3

Intellectual Property Rights ....................................................................................................................7
Foreword ....................................................................................................................................................7

1 Scope ...................................................................................................................................................8

2 References ...........................................................................................................................................8
2.1 Normative references .......................................................................................................................8
2.2 Informative references .....................................................................................................................9
2.3 Definitions .........................................................................................................................................10
2.4 Abbreviations ....................................................................................................................................10

3 Active Monitoring in traditional networks .........................................................................................11
4 Impact of NFV on active monitoring ................................................................. 12
  5.1 Roles and responsibilities for a virtual test agent ........................................... 12
  5.2 Roles and responsibilities for a Test Controller ............................................. 13
  5.3 Requirements for Test Results Analysis Module ........................................... 14
  5.4 Proposed Active Monitoring Framework for NFV ........................................... 15
6 Fault Notification Quality Indicators .................................................................. 17
  6.1 Purpose ........................................................................................................ 17
  6.2 Canonical Failure Notification Model .......................................................... 17
  6.3 Quantitative Failure Notification Indicators .................................................. 20
  6.4 Failure Notification Quality Indicators in NFV ............................................. 20
7 Methods of Measurement ................................................................................... 21
  7.1 Service Activation ........................................................................................ 21
  7.2 Fault Isolation and Troubleshooting ............................................................ 23
  7.1 Framework for End to End In Situ Monitoring ............................................. 26
  7.4 Capacity Planning ....................................................................................... 28
  7.4.1 Capacity validation ................................................................................ 29
  7.4.2 Capacity planning forecast ................................................................... 29
  7.4.3 Optimize service endpoint location ....................................................... 31
  7.5 Performance monitoring ............................................................................. 31
  7.5.2 Overload Detection ................................................................................ 34
8 Evaluating NFV Resiliency ................................................................................ 34
  8.1 Network Resiliency Evaluation .................................................................... 35
  8.2 NFV Resiliency Evaluation using active fault injection ................................... 36
  8.2.1 Fault Injection framework for evaluating NFV resiliency ....................... 36
  8.2.2. Multilevel Challenge/Fault modelling .................................................. 38
  8.2.3. NFVI faults & failures ....................................................................... 39
  8.3. Traffic Tolerance ....................................................................................... 41
  8.4. Failure Impact .......................................................................................... 41
9 Deployment Scenarios ....................................................................................... 42
9 Security Considerations ..................................................................................... 42
10 Recommendations ............................................................................................ 43
Annex A (informative): ......................................................................................... 44
  A.1 Why Active Monitoring ............................................................................. 44
  A.2 Virtual Test Agent Implementation Considerations ...................................... 44
  A.3 Test VNF .................................................................................................... 44
  A.3.1 Test VNF Descriptor ........................................................................... 44
  A.3.2 Test VNF Record (VNFR) .................................................................... 45
  A.3.3 Test Instruction Set ................................................................................ 46
  A.3.4 KeepAlive messages ............................................................................ 47
  A.4 Test Measurement Methods ....................................................................... 47
  A.4.1 Fault Localization ................................................................................ 47
  Annex A.4.2 NFVI Metrics for fault co-relation ............................................. 49
  A.5 Synchronization protocol definition for Test Controller HA ................. 49
This section defines the synchronization mechanism between the primary and backup Test Controllers
to maintain in-sync information on the VTAs assigned to the controller, test instructions for the
tests under execution and test instructions for the periodically scheduled tests ................... 49
  A.6 Result Analysis .......................................................................................... 49
Annex B ................................................................................................................... 49
  B.1 Service QoS & QoE measurements methods .............................................. 49
  B.2 User Traffic Workloads and Distributions .................................................... 51
  B.2.1 Workload Mix ...................................................................................... 51
  B.2.2 Workload Distributions ..................................................................... 51
Annex C Example Measurement Methods and Metrics .......................................... 52
  C.1 Example SLAs ........................................................................................... 52
  C.2 Application Test Methodologies for QoE measurements ......................... 53
  C.2.1 NetFLIX™ Adaptive Streaming Load Generator with Quality Detection .... 53
5/62
**Intellectual Property Rights**

IPRs essential or potentially essential to the present document 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 (http://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.

**Foreword**

This Group Specification (GS) has been produced by ETSI Industry Specification Group (ISG) Network Functions Virtualisation (NFV).
1 Scope

This GS develops a specification detailing methods for active monitoring of VNFs, NFVI and E2E network services and detection of failures. It addresses the following two aspects of service activation:

1) Periodic testing of VNFs and service chains in a live environment to ensure proper functionality and performance adherence to SLAs.
2) Failure prevention, detection and recovery – Active monitoring methods for failure prevention (proactive) or timely detection and recovery from failures. Failures include loss or degradation of network connectivity, loss or degradation of session capacity, loss of services, VM failures, VM stalls, etc.

This WI envisions that the monitoring agents will be part of the NFV architecture and be on boarded into the NFV environment, just like other VNFs. The WI will also highlight any enhancements needed in the ETSI NFV architecture to support service activation testing.

The document is intended to be informative in nature but may lead to normative specifications for active monitoring in the NFV environment.

2 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.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

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

2.1 Normative references

The following referenced documents are necessary for the application of the present document.

[2] IETF RFC 5357 A two-way active measurement protocol
[3] IETF RFC 4656 A one-way active measurement protocol
[7] ETSI GS NFV-SEC 001: "Network Functions Virtualisation (NFV); NFV Security; Problem Statement”.
[8] ETSI GS NFV-SEC 003: "Network Functions Virtualisation (NFV); NFV Security; Security and Trust Guidance".
2.2 Informative references

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 Collaborative Portal: NFV.
[i.2] ETSI GS NFV-MAN001: “Management and Orchestration”
[i.3] ETSI GS NFV-REL 001 version 1.1.1 – NFV Resiliency Requirements
[i.6] ETSI GS NFV-INF 010 Version 1.1.1 – Service Quality Metrics
[i.7] TM Forum Cloud SLA Application Note Version 1.2 - GB963
[i.8] TM Forum SLA Management Handbook Release 3.1 – GB 917
[i.9] TMF TR 178 “E2E Cloud SLA Management”
[i.13] “NFV Quality Management Framework”, SNAPSHOT Draft, April 23, 2015, The NFV white paper is posted on the NFV team portal on the QuEST Forum member web site / Executive Board / NFV Strategic Initiative / Files & Documents.
[i.15] CSMIC defined measures, http://csmic.org
[i.16] “NIST Cloud Computing Cloud Services Description”, Rev. 2.3d9
[i.20] ETSI GS NFV-REL 001 (V1.0.0): “NFV Resiliency Requirements”
[i.21] CSMIC SMI Measure for Assurance Category, http://csmic.org
[i.22] ETSI GS NFV-REL 003: “Network Functions Virtualization (NFV); Reliability; Report on Models and Features for E2E Reliability”.
[i.23] ETSI GS NFV-SEC 008: ”Security Monitoring for NFV Deployment ”.
Definitions, symbols and abbreviations

2.3 Definitions

For the purposes of the present document, the terms and definitions given in GS NFV 003 and the following apply:

**Virtual Test Agent (VTA):** VNF for active monitoring probe capable of sending and analysing control plane and data plane testing

**Test Controller:** It’s a management module (primarily software) responsible for management of the test agents/probes
- Provides test instructions to the test probes
- Co-ordinates the test scheduling when multiple tests with large number of test probes are executed
- Retrieves results from the results analysis engine to provide actionable information to the network operator via NFVO

**Test Results Analysis Module (TRAM):** Integral part of the active monitoring framework that collects or receives test results from the VTAs, NFVI resource statistics and alarms from VIM and analyses test results and presents it to Test Controller or NFVO in an actionable format.

**Frame loss ratio** [4]
Frame loss ratio is defined as a ratio, expressed as a percentage, of the number of service frames not delivered divided by the total number of service frames during a time interval T, where the number of service frames not delivered is the difference between the number of service frames arriving at the ingress ETH flow point and the number of service frames delivered at the egress ETH flow point in a point-to-point ETH connection.

**Frame delay** [4]
Frame delay can be specified as round-trip delay for a frame, where frame delay is defined as the time elapsed since the start of transmission of the first bit of the frame by a source node until the reception of the last bit of the loop backed frame by the same source node, when the loopback is performed at the frame's destination node.

**Frame delay variation** [4]
Frame delay variation is a measure of the variations in the frame delay between a pair of service frames, where the service frames belong to the same CoS instance on a point-to-point ETH connection.

**Throughput** [5]
Throughput is defined as the maximum rate at which no frame is dropped. This is typically measured under test conditions.

2.4 Abbreviations

For the purposes of the present document, the abbreviations given in GS NFV 003 and the following apply:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>Authentication, Authorization and Accounting</td>
</tr>
<tr>
<td>WG</td>
<td>Working Group</td>
</tr>
<tr>
<td>VTA</td>
<td>Virtual Test Agent</td>
</tr>
<tr>
<td>CBS</td>
<td>Constant Bit Rate</td>
</tr>
<tr>
<td>CCDF</td>
<td>Complementary Cumulative Distribution Function</td>
</tr>
<tr>
<td>CiGoodput</td>
<td>Cloud Infrastructure Goodput</td>
</tr>
<tr>
<td>CiQoE</td>
<td>Cloud Infrastructure Quality of Experience</td>
</tr>
<tr>
<td>CiR</td>
<td>Cloud Infrastructure Reliability</td>
</tr>
<tr>
<td>CoS</td>
<td>Class of Service</td>
</tr>
<tr>
<td>DPI</td>
<td>Deep packet Inspection</td>
</tr>
<tr>
<td>DUT</td>
<td>Device under Test</td>
</tr>
<tr>
<td>EBS</td>
<td>Excess Burst Size</td>
</tr>
<tr>
<td>EIR</td>
<td>Excess Information Rate</td>
</tr>
</tbody>
</table>
3 Active Monitoring in traditional networks

Live testing typically involves end-to-end testing of services versus single node testing where the testing may be performed at the pre-activation, or post-activation, of services. Three key components of a test system in live networks are: 1) Test Controller, 2) results analysis module and 3) test agent. In non-NFV network deployments, the testing agents are typically deployed in the long-term as long as the testing or monitoring PoP does not change. Test Controller and results analysis module may be part of the OSS/BSS system or may be a standalone application in the same administration domain as the OSS/BSS system.

Figure 1 Live network testing in non-NFV networks

Based on the type of test agents, testing in live networks can be categorized into active, passive or hybrid. Active testing involves sending test traffic into the network to validate the services and applications performance, SLAs and to perform fault isolation. Passive testing involves mainly observing the user traffic, providing an analysis based on test traffic and raising alarms if pre-set thresholds are crossed. Hybrid approach, as the name suggests, uses the information obtained from both active and passive approaches.
4 Impact of NFV on active monitoring

NFV increases the variability in the network topology imposing additional requirements on the active network monitoring framework. The active monitoring solution must be able to deal with NFV aspects such as VNF migration, auto-scaling and multi-tenancy of VNFs in order to be effective in a NFV environment.

An active monitoring solution for NFV networks may consist of three modules

- Test Controller
- Virtual Test Agent (VTA)
- Test Result Analysis Module (TRAM)

The following subsections list the requirements for active network monitoring modules as a result of NFV.

5.1 Roles and responsibilities for a virtual test agent

Network visibility and fault diagnostic capability in an NFV network is limited when using physical active test agents. With physical test agents/probes, it may not be possible to maintain the monitoring PoP. In order to provide increased visibility and better fault diagnostic in an NFV based network, the test agent needs to be a virtual entity. A virtual test agent provides the advantage of ability to reside between the VNFs in a service chain and automatically re-provision to maintain the monitoring PoP in a VNF migration scenario.

For an effective active monitoring solution in a NFV environment following are the requirements for a test agent:

- Test agent must be virtual and it must be possible to instantiate test agent as a Test VNF using the VNF instantiation workflow recommended in [1,2].
- Test agent must be able to re-provision or move if the monitoring PoP is moved as a result of VNF migration.
- Test agent should have minimal impact on the performance of the VNFs that reside on the same server as the test agent. The ideal state of 100% performance isolation should be the goal when implementation of a virtual test agent.
- A repository of test agents with specific test features and performance capabilities may exist. Targeted test agents will make the test agents lightweight and help with achieving higher performance and minimizing the performance impact on other VNFs that reside on the same physical server.
- Test VNFD may be defined for specifying test capability parameters in addition to the standard VNFD parameters. None of the ETSI NFV architecture components need to use the Test VNFD and the scope of Test VNFD is pertinent to the service assurance application only.
- Periodic KeepAlive messages between VTA and Test Controller may be implemented for fast failure detection and high availability.
- Test agent may provide failure and logging messages if the measurement task was not run to completion.
  - Although Test Controller tracks the resource utilization of the VTAs, performance isolation issues or changes in the resource provisioning may result in the test agent’s inability to run the desired test. In such a scenario VTA, should send a failure or error message to the Test Controller.
If, for some reason, VTA is not able to communicate or report the results to TRAM, then it should send a failure message to the Test Controller indicating the reason.

Logging messages should be provided for various events at the start of test execution, during the test execution, at the end of test execution and during the results reporting.

- A VTA should perform the following pre-checks before it starts sending test traffic.
  - A test would need to send high throughput traffic on the same path as the service under test. In this scenario, VTA must ensure that there is no user traffic on the path before it begins transmitting. It is partly the network operator’s responsibility to schedule such tests at a time so that end user service experience is not impacted. Additionally, there should be a mechanism to differentiate between test and end user data such that test traffic does not use the service quota allocated to the user.
  - VTA is able to communicate with TRAM.

- Primary Test Controller failure:
  - Additional Test Controller may be configured as a back up to provide high availability.
  - If a backup Test Controller exists, VTA’s VNFR (VNF Record) must contain the backup Test Controller’s ID.
  - If the Test Controller timeout timer expires, VTA must establish a session with the backup Test Controller using the backup controller ID in the VNFR.
  - Primary and back up Test Controllers must be synchronized periodically in terms of information on supported VTAs, test instructions for tests under execution and the information on periodically scheduled tests.
  - Once the backup Test Controller takes over, it must also establish communication with the NFVO.

5.2 Roles and responsibilities for a Test Controller

- Maintain test agent VNFR catalogue.
- Track active tests, resource utilization taking into account the tests that are scheduled to run periodically.
- To support higher scale requirement for test agents, Test Controller may be implemented in a distributed manner where multiple instances of Test Controller work in collaboration.

Note: NFV based networks will require more number of test agents for effective monitoring and greater network visibility. The active monitoring solution for NFV is expected to reduce the cost per test agent. Considering that cost of the test agent/probe is one of the major factor that influences the SPs decision on the number of test agents that are deployed in the network, service providers and network operators may be inclined to deploy larger number of VTAs as compared to physical test probes to achieve higher network visibility and fault isolation capability

- Considering the catastrophic implications of a compromised VTA or a Test Controller, a secure channel must be established for communication between Test Controller and VTAs.
- If the VTA does not have sufficient NFVI resources or feature capabilities to support the test instructions, then the Test Controller may deploy new VTAs or increase the resource allocation for the existing VTA.
- A unique test instruction identifier may be defined to compare multiple result instances for the same set of test instructions.
- Ability to suppress/un-suppress measurement tasks of specific VTAs, suppress specific tasks, or suppress specific schedules, or suppress specific schedules of specific tasks.
- Collaboration between Test Controllers is required for HA implementations where multiple Test Controllers exist and a subset of test agents implement different communication protocol with Test Controller and TRAM.

5.3 Requirements for Test Results Analysis Module

Result report may contain the following information:
- TRAM may be implemented as a distributed topology with multiple smaller entities collecting subset of results for achieving higher scalability when large number of VTAs are deployed in the network.
- It is assumed that any service SLA parameters or subscriber contract information is part of the network service descriptor or network service record. The results analysis module will use this information to compare it against the test results for SLA validation.
- TRAM may use push or pull model to get the results from the virtual test agent and subsequently provide the processed results to the presentation module in the OSS/BSS via the NFVO.
- Test results:
  - Start/stop timestamp for test run and the timestamp when the test results were reported.
  - Test instructions, input parameters, list of VNFD ID’s for test VMs that were part of the test may be included as part of results reported.
  - NSR ID (Network Service Record ID) may be included.
  - Alarms information for any threshold violations.
5.4 Proposed Active Monitoring Framework for NFV

Figure 2: Active Monitoring Framework

Figure 3 Message Sequence Flow for Active Monitoring
The main steps for test execution in active monitoring framework are as follows:

1. **Test Instruction Set**: This step involves the operator’s management interface sending a test set instructions via NFVO to the Test Controller in consideration. Note that an active monitoring network may have multiple Test Controllers working in collaboration or independently. The test instruction may consist of NSR (Network Service Record) for the service that needs to be validated or test details such as type of test that needs to be run, PoP for active monitoring and TRAM ID.

2. **Provision VTAs (Test Controller)**: Once the Test Controller gets the instruction sets, it needs two types of information to provision the VTAs – PoP where the VTAs need to be provisioned and the capabilities of the VTAs based on the type of test that needs to be run. These two pieces of information may be present directly in the test instruction set, or it may be retrieved from the network service record of the service that needs to be validated. Once the Test Controller knows where the VTAs need to be provisioned and the type of VTAs that need to be provisioned, it may provision new VTAs via NFVO, or if the required VTAs already exist and are not being used, they may be moved to the desired PoP. It is also possible that some of the desired VTAs may be in dormant state and need to be activated or licensed to be part of the test under consideration.

3. **Provision VTAs (VNFM)**: This is same instruction as defined in previous step except that this message is sent by NFVO to VNFM for provisioning of VTA.

4. **Allocate Resources**: VNF Manager sends an “Allocate Resources” message to the VIM for the desired VTAs. Here again, the “Allocate Resources” action may be replaced by moving existing VTAs if there are VTAs available for the test in the domain and can be moved to the desired PoP or if there are any dormant VTAs with desired test and performance capabilities already present at the monitoring PoPs.

5. **Allocate Resources (VIM)**: VIM performs a check to verify if the requested resources are available and reserves and allocates the requested NFVI resources for use by the VTAs.

6. **Test Instructions (VTA)**: Subsequently, the Test Controller sends the test instructions to each individual VTA that is part of the test. Note that here, it is the responsibility of the Test Controller to ensure that the VTA is available and has the desired test and performance capabilities to run the test. Optionally, a VTA may be required to send an acknowledgement back to Test Controller to indicate that it can participate in the test.

7. **Run/Schedule Test**: The VTAs run the test or schedule it for later execution based on the test instructions from VNF Manager. The tests may be run one time or executed periodically at regular intervals.

8. **Report Results**: Once the test run is completed, the test results may be reported to the desired TRAM using a push model, or the results may be polled by TRAM using the pull model. A hybrid push/pull approach may be used.

9. **Report NFVI Utilization Stats, Result Analysis**: In an NFV environment where NFVI resources are shared across VNFs and ideal performance isolation may not be achieved, just the test results from VTAs may not be enough to provide accurate and actionable information to the network operators. Thus the result analysis has to incorporate the results from VTA and NFVI utilization stats to get a better picture. The implementation of result analysis based on these two sources of results/stats is out of the scope of this document.
Please refer to B.4.4.2 in [i.4] for more details on each of the steps involved in expansion of resources. Note that the same process may be used for modifying (increase/decrease) the NFVI resource allocation or auto-scale (scale in, scale out) operations for VTA.

6 Fault Notification Quality Indicators

6.1 Purpose

This section defines objective and quantitative measurements of the fault notification timeliness, reliability and accuracy provided by NFV management and orchestration elements.

6.2 Canonical Failure Notification Model

Figure 5 illustrates a canonical service model: an entity (e.g., NFV infrastructure), called a service provider, offers a service product (e.g., virtual machine and virtual network services) across a service boundary reference point (e.g., Vn-Nf) to a service customer (e.g., a VNF instance).
ISO-IEC-IEEE 24765:2010 “Systems and software engineering — Vocabulary” defines failure as:

1. termination of the ability of a product to perform a required function or its inability to perform within previously specified limits
2. an event in which a system or system component does not perform a required function within specified limits

Thus in the context of Figure 5, a service failure event begins when the service product functionality is no longer delivered within previously specified (quality/performance) limits by the service provider across the reference point to the service customer. For example, inability of NFV infrastructure to deliver service quality, performance or throughput to previously specified levels for one or more virtual resources (e.g., virtual machine, virtual network) is a failure of the impacted resource(s).

At the highest level, Figure 6 visualizes a canonical failure notification model:

1. $T_{\text{Activate}}$ – instant an error is activated or a fault is manifest; likely to be slightly before service delivered to service consumer across reference point is impacted.

2. $T_{\text{Failure}}$ – instant service delivered by the service provider to a service customer across the reference point is no longer within previously specified (quality/performance) limits.

3. $T_{\text{Notification}}$ – instant when failure notification associated with $T_{\text{Failure}}$ from some management component crosses applicable management notification reference point.
applies the canonical failure model to the NFV architecture for a simple hardware failure:

1. $T_{\text{Activate}}$ – instant an error is activated or a fault is manifest; likely to be somewhat before service delivered to service consumer is impacted.

2. $T_{\text{Failure}}$ – moments after the board mounted power module failure, the NFV infrastructure will no longer be able to deliver all required functions or perform within previously specified service limits for some, or all, of the VNFs that have virtual resources assigned to the impacted hardware component. Note that multiple VNFs might have virtual resources assigned to the impacted hardware component, and each of their pattern and timing of resource usage will vary, so different VNFs may experience the virtual resource failure at slightly different times, meaning that different VNFs might have slightly different $T_{\text{Failure}}$ events.

3. $T_{\text{Notification}}$ – instant when the virtual resource failure notification event crosses each applicable measurement point. Per section 7.3.5 ‘Virtual Resource Fault Management’ of NFV Management and Orchestration (ETSI GS NFV-MAN 001 v1.1.1 2014-12), the following virtual resource fault ‘notify’ interfaces are stipulated:
   a. VIM notifies NFVO via Or-Vi
   b. VIM notifies VNFM via Vnfm-Vi
   c. VNFM notifies EM via Ve-Vnfm-em

The $T_{\text{Notification}}$ for a particular $T_{\text{Activate}}$ event will likely vary across each of the applicable reference points.

Figure 6 – Canonical Failure Notification Model
6.3 Quantitative Failure Notification Indicators

The canonical failure notification model enables three quantitative performance indicators:

Failure notification latency is the elapsed time between the moment when resource failure impacts a service consumer ($T_{\text{Failure}}$) and the associated fault notification event ($T_{\text{Notification}}$). Ideally, the failure is mitigated so the service consumer never experiences a resource failure, so failure notification latency is moot. The second best scenario is that the failure notification is received before the consumer experiences a failure ($T_{\text{Notification}} < T_{\text{Failure}}$); such anticipatory notification is technically a negative latency value.

Failure notification reliability is the portion of service failure events that are alerted within the maximum acceptable failure notification time. Mathematically, this is the number of failure events where $(T_{\text{Notification}} - T_{\text{Failure}}) < T_{\text{MaxFailureNotificationLatency}}$ (numerator) divided by the total number of failure events (denominator). Service failures that prompt no notification nominally have an infinite $T_{\text{Notification}}$ and thus would exceed any finite $T_{\text{MaxFailureNotificationLatency}}$.

Note:

1. that cases where the failure notification is received before service fails are counted in the numerator, and
2. that events that do not manifest as a failure to the service consumer are not counted in either numerator or denominator.

Failure notification accuracy is the portion of events that manifest as failures to service consumers that correctly identified the underlying component that failed.

If no applicable failure events occur during the measurement window, then failure notification latency, reliability and accuracy are designated as not applicable in that period.

6.4 Failure Notification Quality Indicators in NFV

NFV Management and Orchestration (ETSI GS NFV-MAN 001 v1.1.1 2014-12) specifies the following ‘notify’ interfaces for Fault Management.

1. Virtualized Resource Fault Management (section 7.3.5):
VIM notifies NFVO via Or-Vi,
VIM notifies VNFM via Vnfm-Vi,
VNFM notifies EM via Ve-Vnfm-em.

2. VNF Fault Management (section 7.2.8):
   VNF notifies VNFM via Ve-Vnfm-vnf,
   VNFM notifies NFVO via Or-Vnfm.

3. Network Service Fault Management (section 7.1.5):
   NFVO notifies OSS via Os-Ma-nfvo.

It is likely that fault notification reliability and accuracy will be identical for a particular fault type (e.g., virtual resource failures) across all supported reference points (e.g., Or-Vi, Vnfm-Vi, Ve-Vnfm-em); however, the fault notification latency will likely vary across each of the reference points. Thus, the maximum acceptable fault notification time ($T_{\text{MaxFailureNotificationLatency}}$) may vary both by fault type and by notification reference point. $T_{\text{MaxFailureNotificationLatency}}$ values are agreed by service providers and their suppliers for particular deployments rather than being subject to standardization.

7 Methods of Measurement

7.1 Service Activation

RFC 2544 is a well-established benchmark for measuring performance of standalone network devices, but is not good enough for measuring performance of services. Activation or deployment of a service not only involves multiple network devices but also adds more complexity with associated SLAs for the end-to-end service. NFV brings additional dependencies on virtualization technologies and NFVI resource performance. NFV is marked by variability of performance due to multi-vendor environment and performance impact of interaction between the network functions that form the service chain. This requires periodic testing of end-to-end services to account for time varying impairments in the network.

In such an environment, it becomes imperative to test the service in an end-to-end manner and validate the ability of the network to satisfy the SLAs associated for to-be-deployed services as well as already existing services.

ITU-T Y.1564 standard defines methodologies for service activation. It tests the service configuration and performance of Committed Information Rate (CIR), Excess Information Rate (EIR), Committed Burst Size (CBS) and Excess Burst Size (EBS). Timing characteristics such as latency and delay variation are assessed not only at the maximum no loss transmit rate, but also at a rate that exceeds the CIR. Exceeding the CIR and CBS is expected in a real network and the Service Provider should be able to monitor and measure these parameters. Please refer to [i.5] for detailed methodologies on service activation in live networks.
Figure 8 High level test methodology for service activation (as defined in ITU Y.1564)

Figure 9 Bandwidth thresholds for testing service performance based on SLA definition

For NFV environments, we need to understand the impact of supporting the SLAs associated with service together with the understanding of the NFVI resource utilization when these services are exposed to the extremes of the SLA limits. Thus, result analysis for live testing should take into account both the results obtained from test traffic and the NFVI utilization statistics when the service is exposed to various test workloads during the test.
In addition, the sharing of NFVI resources, across the services deployed, warrants verifying that the SLAs for existing services still holds true when new services are provisioned and activated.

Traditionally, service activation has focused on testing using L2-L3 traffic. As we move to NFV based networks, performance testing based on L2-L3 test traffic is not sufficient. The variability in the network due to shared NFVI resources raises the importance of testing that focuses on QoE for the E2E services versus QoS. End users are primarily concerned about the QoE of the services and may not care about the QoS based SLAs provided by service providers. Thus, there is a need for service activation testing using stateful L4-L7 traffic or application traffic that emulates the end user service. RFC 6349 provides a well-established standard for benchmarking TCP throughput performance and is useful for establishing the performance benchmark for most of the stateful E2E services and applications. In addition, application testing by emulation of high scale real application traffic is an important feature in a test tool to validate the QoE for the end user.

Annex B.1 lists some of the test methodologies for application level testing that measures the QoE for specific types of user workloads.

Elasticity or capacity on demand may have an impact on the end user’s experience. As part of the service activation, it is important to test the ability of the network to rapidly auto-scale based on the user traffic, such that the end user’s QoE is not impacted. This testing may be done by exposing the service to various user workloads with varying slew rate and distribution. Annex B.2 lists some of the common workloads that exercise the network’s capacity to dynamically expand, or contract, capacity. Note that the max rate or max burst size of these workloads should be calibrated such that it triggers auto-scale mechanisms.

### 7.2 Fault Isolation and Troubleshooting

![Figure 10 Root Cause Analysis](image)

Faults or service issues may be categorized as connectivity or performance issues. This WI intends to focus on service availability, performance isolation and SLA validation issues for network services. Service availability issues do not need line rate test traffic, and periodic tests for service availability may be performed without interrupting end user traffic.
Passive fault localization is less intrusive as compared to active fault localization; however, it may take long time to discover the root faults, especially if loss ratio is high and there are multiple faults causing the problem. Integrated approach using both passive and active fault localization may be more effective with minimum intrusion.

On the other hand, intermittent issues are difficult to localize if only active fault isolation mechanisms are used. The static nature of the physical test agents/probes forces the user to rely on the effectiveness of probe selection and optimization algorithms to localize the fault faster and in a cost-effective manner. Virtualization of the test agents will help in the deployment and diffusion of this method for fault localization in a cost-effective way and reduce the reliance on effectiveness of probe selection algorithms. Nonetheless, efficient probe selection algorithms are required to minimize the fault isolation time duration. The combination of virtual test agents and efficient probe selection algorithms provide a cost effective solution for fault localization in shorter duration.

It is well understood that when a critical failure occurs, it is important to restore the service or have redundancy built into the network design to protect against failures so that the end user service impact is minimal. The first step to restore the service is to isolate the fault. This process is well understood in traditional networks, but not so much in the NFV environment. A single E2E service failure may have multiple causes and, at the same time, single root cause may be responsible for multiple service failures. Additionally, service failures may not be just service availability failure, but it may be service quality degradation as well.

Note that although the general procedure outlined in the following steps for fault localization is applicable, there is plenty of room to optimize the fault localization time through proprietary probe selection algorithms, and procedures to isolate faults affecting multiple services at the same time. In addition, the fault localization procedures have to take into account any dependency on dynamic routing for calculation of service VNFFG. There is a possibility that VNFFG may be modified when the fault localization procedure is not completed. In addition, due to the presence of load balancers and high availability architectures, the test packets need to test the alternative paths through explicit routes. Policies may be defined at the NFVO level to restrict any changes to the components of the service till the fault localization procedure on the service is completed. It is not the intent of this document to provide optimized algorithms for fault localization, but to highlight the need and possibility of such mechanisms. The implementation of such optimizations is expected to be proprietary and out of scope of this document.

A generic procedure for fault localization is as follows.

1. Sectionalize the service.
2. Test each of the sections independently to localize the server and then, subsequently, specific the VNF/VNFs causing the service degradation or failure.
3. The Test Controller should have the ability to parse through the VNFs within the sections and determine how the payload may be modified by the intermediate VNFs and configure the test gents to take the payload modification into account for making various types of SLA, QoS or QoE measurements, e.g., MPLS labels, VXLAN encapsulation or any other type of encapsulation or de-capsulation.
4. Once the causal VNF, or set of VNFs, is localized, narrow down the cause for the fault to VNF software, hypervisor or other NVFI resources based on looking at the alarms, triggers and NFVI statistics obtained when the test iterations were executed. The detailed procedure of identifying the individual components and type of
faults/failures is explained in [i.3] Section 7 and Section 8. Subsequently, the results may be reported to the results analysis engine.

5. The network operator may go through similar iterations to verify the service reliability once the cause of the fault has been fixed, since there may be multiple reasons for the service degradation.

The lower slew rate can cause issues in remedial actions such as scale up/down or in/out in a timely fashion to address the issue observed. That’s why it becomes important to continuously validate the thresholds set for monitoring service quality by using active monitoring techniques in a proactive manner.

The scenarios when active monitoring is required in tandem with monitoring of NFVI resource utilization include:

- Elasticity thresholds or policies are incorrect or slew rate is slower than expected and cannot adapt to the changing workload pattern.
- NFVI performance measurements are not accurate or obtained timely for the corrective action.
- Elasticity actions taken in response to threshold violation or alarms generation need to be verified as the actions may not have fixed the issue or the actions may not have been completed successfully for variety of reasons.

Once the fault isolation is narrowed down to a particular VNF, further analysis is required to identify if the application representing the VNF is causing the service fault or is it due to NFVI impairment or is it the guest OS used by the VNF. This may be done using simple analysis such as [i.5]:

1. Probing availability of the VM instance’s operating system (e.g., via ping).
2. Checking status of the VM instance hosting the nonresponse application component instance via a cloud service provider mechanism (e.g., API) to see if the VM instance is reported to be operational.
3. Heartbeat mechanism to check the liveness of the application.

Reference [i.5] Chapter 14 further illustrates how measurement of aspects such as non-delivery of VNF/VNFC capacity, measurement of service latency, clock event jitter and clock drift can help in drilling down deeper as part of the fault isolation process.

[i.5] “Measurement of non-delivery of VM capacity
Measurement of non-delivery of configured VM CPU capacity can be measured by comparing the timestamps of high frequency regularly scheduled events to isolate intervals when the VM did not run. Non-delivery of network capacity can be measured by comparing output queues with transmit data statistics. An increase in queue depth without a corresponding increase in transmitted bits may indicate a network non-delivery condition.

Measurement of delivery of degraded VM capacity
Growing work queues when the volume of offered work remains stable may indicate that the virtualized infrastructure is delivering less resource capacity. Likewise, an increase in IP packet retransmissions or lost packets suggests that cloud networking infrastructure may be congested and thus is discarding packets. Analysis of performance counters from the guest OS or the hypervisor can offer insights into the quality of infrastructure service delivered by the cloud service provider.

Measurement of service latency
The traditional way to monitor and characterize that service latency would be to build latency performance complimentary cumulative distribution functions (CCDF) or histograms. As we know that only limited number of buckets for histogram measurements
are available and it is cumbersome to determine the size the limits of the histograms to monitor the latency. Additionally the variance in case of NFV environment is high and the latency measurements need to be done more frequently. This may yield large amount of data which is difficult to store and analyze. Average latency and variance in latency may be used instead of actual latency measurement to deal with the large data challenge in order to characterize the latency of a service.

**Measurement of clock event jitter**
Measure the mean and variance latency between when each clock event was requested to trigger (e.g., 1000 μs from now) and when the timer service routine was actually executed (e.g., 2345 μs later).

**Measurement of clock drift**
Time synchronization programs, such as NTP and PTP daemon, can be configured to log the clock adjustments they make, and analysis of these adjustment logs enables one to characterize the nature and magnitude of clock drift experienced by each VM instance.

**Measurement of failed or slow allocation and startup of VNF**
If the application's monitoring and control component explicitly initiates VM instance allocation for application startup and growth actions, then that monitoring and control component can measure the response latency and status of the allocation and startup of VM instances. It is useful to record at least the following details for each VM allocation request:
1. Time of allocation request
2. Characteristics of VM being requested (e.g., number of CPU cores and RAM allocation)
3. Time of allocation response
4. Final status of allocation request (e.g., success or error code).

### 7.1 Framework for End to End In Situ Monitoring

Historically, non-NFV networks have traditionally monitored using standalone active and passive monitoring methodologies that are reactive in nature. The traditional monitoring techniques by themselves do not provide the capability to detect errors proactively and there is a need for a different level of monitoring in NFV based networks, in addition to the traditional monitoring techniques. Also, having network components from a variety of vendors has made it even more difficult to find patterns in the data because this data is usually not in a standard format. This has ultimately contributed to relatively high sustained No Fault Found (NFF) rates under certain circumstances. Anticipating how dynamic and decentralized NFV networks will likely become in the future, we need to utilize the opportunity of a new type of network to take a much more ubiquitous and proactive monitoring approach that makes use of the best known practices that have matured in the IT industry (Agile, Big Data, ubiquitous in situ monitoring, etc.) to ensure a much more flexible and adaptable network to better address the current and future needs of the end user. This monitoring would occur in a live network rather than a lab environment and, subsequently, get scaled up as appropriate while still implementing concepts from the Toyota production methods (e.g., Lean, SPC, continuous improvement, etc.). To make this approach much more effective than what has been done in non-NFV networks, monitoring the VNF, NFVI, and MANO components would be essential to identify the root cause in timely and effective manner. The in-situ monitoring in combination with active monitoring and passive monitoring of user traffic presents a comprehensive monitoring solution for NFV based networks.
Assuming a Fault→Error→Failure model \([i.12]\) in conjunction with utilizing a modular network element approach, we can monitor, from an end to end perspective, a basic set of metrics (e.g., error rate, junction temperature, memory utilization, etc.) in situ on an ongoing basis to determine, with a minimal number of resources, the basic health of the network. If a monitored value is determined to be outside of a tolerance or specification range, then additional metrics from a larger standard set of metrics list (active/passive/hybrid metrics) can be captured and automatically reported to an automatically updated network health dashboard and technical resources that can evaluate, as well as permanently resolve, the issue(s). This would be a two-step process. The first step would be a short term solution to identify the root cause of the problem by catching it early enough to clearly see the initiating issue and fix the symptom. For example, for a software coding fault, the erroneous value can then be replaced with a known safe value as part of a fault tolerance mechanism to help the overall network be more fault-tolerant. If there is a hardware fault, then the appropriate action can be prompted (e.g., sharing the resource load if the processor temperature gets too high, using an alternative hardware resource if the voltage is unstable or within specification, but not within the defined tolerance range, correlating reliability data from chipsets or other components based on service time or use cases). The longer term fix would be to remove it from the network and/or change the hardware or software design to prevent the problem from ever occurring again. Taking this approach would be analogous to monitoring the heart rate, temperature, blood pressure, and other basic health parameters of a person. If there is a problem, then the symptoms (e.g., headache) can be addressed and the root cause (e.g., a brain tumor) can be addressed using a longer term method (e.g., surgery, radiation treatment, chemotherapy, as well as, preventative treatments like eating healthy foods, maintaining a healthy weight, exercising, etc.).

The Fault→Error→Failure model \([i.12]\), along with this larger standard set of metrics list, will enable operators to proactively find patterns in the data which will help to identify the root causes of errors before they cascade into such large and complex issues that it becomes difficult to identify the root cause of the problem. Since the hardware, and likely the software, will be common among operators, this information can potentially be aggregated globally while still maintaining privacy to help reduce the uncertainty in any reliability models, as well as, finding subtle problems in either the hardware or software that may only be seen at a Parts Per Million (PPM) or Parts Per Billion (PPB) level rather than just at a percent level. This approach will help catch problems like last year’s industry wide memory design issue much earlier in the cycle so that it will not take years to find it, as well as reducing the number of resources necessary to eliminate the problem.

This proposed model will also enable leading operational productivity and efficiency metrics to be utilized in addition to the traditional lagging indicators/metrics \([i.13]\) to help manage the network. These same metrics can also be utilized to develop robust network reliability models, as well as be utilized for product development and potentially marketing opportunities.

Just like in mobile phones, the details of the physical hardware along with its physical location should be self-reporting to the resource management part of the network so that any loss could be measured, as well as correlations can be drawn between chipsets, memory modules, build dates, etc. (i.e., hardware reliability) and the reliability of the total network...
solution. Because processing, memory, and other hardware resources can be very decentralized across the globe, timing issues and other subtle nuances can be better tracked. This approach will be useful in finding and eliminating issues with the physical hardware, virtual hardware, software, and enable innovation within the network element over time.

Figure 11 End To End Metrics (Including VNF, NFVI, and MANO Components)

![Diagram of End to End Metrics and Framework](image)

Figure 12 Process Workflow

7.4 Capacity Planning

Some of the options used for capacity planning of services and applications in traditional networks are:

- Reliance on theoretical values,
- Trial and error,
- Simulation tools.

Simulation tools such as CloudSim have been traditionally used for capacity planning. Multiple scheduling algorithms for optimizing the usage of NFVI resources and scheduling of VMs and workloads in datacentres have been discussed, researched and analysed. With the widespread deployment of NFV, the current branch offices, access networks and edge networks may take the form of mini-datacentres where the traditional network devices exist as VNFs. In such scenarios, optimization of resource allocation for VNFs while maintaining SLAs will be a problem similar to the one that has been faced by traditional datacentres and some of the techniques such as admission control and scheduling, while satisfying end user SLAs may be re-used as defined in [i.4].

7.4.1 Capacity validation

Although simulation techniques may provide insight into scheduling and capacity planning, simulation models by itself are not sufficient in the world where 100% performance isolation of multiple VNFs is not there. Due to performance isolation issues, shared NFVI resources and variability introduced by multiple components from different vendors, it is important to validate the capacity planning models with real test traffic that can emulate the real-world scenarios and provide room for experimentation for future scenarios.

Additionally, the resource requirement specifications provided by application vendors may not reflect the actual resource allocation required for optimum application operation in the SP or network operator’s environment. NFV introduces portability of applications, virtualization technologies and NFVI from various vendors and makes it difficult to achieve similar performance in an environment where various NFV components may be from different vendors. These additional constraints necessitate the use of active monitoring solutions to validate the capacity bounds in the SP or network operator’s environment.

A NFV network is bound to be dynamic, where new VNFs may be added, existing ones deleted or moved to a different host machine. For optimum use of the resources, capacity validation should be performed periodically in a live NFV network to ensure that the resource allocation is optimized and also satisfies the associated SLAs at various levels.

7.4.2 Capacity planning forecast

Forecasting capacity requirements in an NFV environment is a difficult task that requires ability to run experiments for hypothetical workload scenarios. Consider a situation where an application or a service becomes popular in a very short duration of time leading to exponential increase in workload for the service. This warrants some type of modelling where the SP can anticipate the resource requirement needs, chart out a roadmap for such events and assigning signposts for such a roadmap. The SP or network operator can subsequently monitor the signals that indicate towards the signposts on the roadmap and get a head start on resource procurement and planning before the increased workload hits the end-user QoE. This technique of road mapping and signposts has been traditionally used in predicting technology futures, but it should be equally effective in forecasting demand and the type of resource required in the near future.

Here again, a combination of active and passive monitoring can be used to achieve the forecasting objectives for resource planning.

An example workflow for capacity validation is as follows:
Figure 13 Example workflow for capacity validation

- Increase the number of emulated users for a specified service till the SLA bounds for the specified service are reached.
- Measure the impact of increased users on the following NFVI utilization stats:
  - vCPU utilization, vCPU interrupts or interrupts/sec (key metric for primary use case),
  - Memory utilization (allocated and utilized, size of block and how many blocks are used, page swaps),
  - Hypervisor priority level for VNF,
  - Number of cores used by core type and utilization (% and number of no-ops/sec),
  - Acceleration technology in use (DPDK in and out frames/sec, SR-IOV, Direct Path I/O, bare metal),
  - Network I/O utilization (IP/UDP/TCP in/out stats),
  - Congestion sense counts and delay time:
    - CPU,
    - Memory,
    - Storage I/O,
    - Network I/O.
- Verify that the SLAs are satisfied at the outer boundary limit from the test traffic results such as throughput, latency, jitter, etc. Depending on the type of service used, associated SLA metrics will vary.
- Use VTA to emulate additional services that are expected to be supported on the network. Repeat the above process to verify the SLA limits for the additional services.
- Note that each time additional services are added, the iteration must make sure that SLAs for existing services are still satisfied. If there is a performance impact on existing services, then this performance impact measurement must be recorded for trend and correlation analysis.
- Additional experiments for testing the capacity of the network for E2E services beyond the SLA limits may be performed to obtain results for “what-if” scenarios.
This testing may be performed periodically to assess the capacity of the network as the NFV network is expected to be dynamic and addition, deletion and migration of VNFs are expected.

7.4.3 Optimize service endpoint location

It is always a challenge for SPs and network operators to balance the optimization of end customer QoE and TCO for providing the service. One of the key aspects involved is the location of service delivery endpoint. Service endpoint located closer to end customer in case of services such as video do improve the QoE for the end user, as it helps with lower delays and jitter in the video stream. But at the same time, maintaining multiple caching PoP closer to end consumer does not scale well for the SP and increases the TCO. In this case, NFV does bring advantage of dynamic provisioning and helps the SP to decide an optimum PoP for the caching location. Increasingly dynamic bandwidth demand/usage and dynamic nature of NFV networks may impact the decision for PoP and the resource allocation for the service delivery PoP. Active monitoring can help validate these decisions at the provisioning time and periodic post deployment validation.

Another important aspect as part of the capacity planning and network design for NFV networks is the PoP selection for VNF provisioning. The PoP or NFVI resource location should be such that the SLA constraints for the services served by the VNF are satisfied and, at the same time, achieve optimum resource utilization. At the initial provisioning time, it may not be evident which and how many services will be traversing the VNF. Additionally, the number of services and the type of services traversing the VNF can change during the lifecycle of the VNF. This again warrants periodic verification of the SLA validation to ensure that appropriate NFVI resources are available and if there is a need for NFVI resource consolidation or VNF migration.

In addition to the NFVI resource allocated to the VNFs that form the E2E service, underlay and overlay network design impact the performance and SLAs for the E2E services as well. This introduces another variable which needs to be validated to ensure that the underlay and overlay network design and resource provisioning provides enough capacity for the E2E services. Active monitoring can help validate the appropriate design by measuring the performance and SLA validation for the design options of the underlay and overlay networks.

7.5 Performance monitoring

7.5.1 SLA Monitoring for E2E services
The above service lifecycle figure from TM Forum highlights some of the important components of a monitoring system. Components such as Service Metrics Monitor, Service Resource Monitor and Service Usage Monitor re-iterate the need to develop and understand the correlation between service behavior, NFVI resource usage and provisioning. Service dashboard becomes an integral part of the periodic monitoring scenario which provides information related to the metrics and resource usage for the services that are monitored.

Tools such as ping and traceroute serve a good purpose to verify availability and map the topology path for the service, but have limitations in providing these services or accurate latency measurements for the following scenarios:

- When using ping for measuring RTT, due to path asymmetry, the intermediate hop may not be part of the reverse path from the destination, thus the RTT calculation may not represent the correct value.
- Relying on multiple ping packets for RTT measurements can result in inaccurate measurement as the multiple ping packets may experience different network conditions.
- When using ping, the forward path up to the intermediate hop may not represent a sub-path of the forward path toward the destination, since forwarding is destination-based.

**TWAMP**

TWAMP is a two way active measurement protocol that uses a TCP client/server model and primarily focuses on round trip performance measurement. The details of TWAMP protocol and measurement mechanisms can be found in [2]. Two way measurements are useful and desired in certain situations such as when measurement tools are not available at destination, or reference clock timing synchronization between source and destination is not achievable. Getting high accuracy in timing synchronization in virtual environments is a difficult problem to solve and is particularly applicable to NFV. Additionally, round-trip measurements provide a better estimate of processing time at the destination.
On the other hand, there are scenarios where round-trip measurements are not preferred and may introduce inaccuracies in measurement.

The issues with round-trip measurements [6] include:
- “The Internet path from a source to a destination may differ from the path from the destination back to the source (“asymmetric paths”), such that different sequences of routers are used for the forward and reverse paths. Therefore, round-trip measurements actually measure the performance of two distinct paths together.
- Even when the two paths are symmetric, they may have radically different performance characteristics due to asymmetric queueing.
- Performance of an application may depend mostly on the performance in one direction.
- In QoS enabled networks, provisioning in one direction may be radically different than provisioning in the reverse direction, and thus the QoS guarantees differ.”

Y.1731

ITU-T Y.1731 specification defines standard Ethernet performance monitoring functionality that includes the following performance monitoring parameters:
- Connectivity,
- Frame delay and frame delay variation,
- Frame loss ratio and availability,
- Throughput.

Y.1731 defines control plane Ethernet messages for the above listed OAM and performance monitoring functionality. Since Y.1731 uses specific control plane PDUs, the nodes participating in the performance monitoring mechanisms must implement Y.1731. This restriction may limit the use of Y.1731 to certain parts of the network that support and implement Y.1731. The details for functionality and messages used for performance monitoring are described in [4].

Application or Service Specific Performance Monitoring

As the network elements are virtualized in the access, edge and core networks, there is a greater need for E2E performance monitoring focusing on the QoE for the end users. One of the key motivations for service providers to move to NFV is improved service agility and ability to provide new E2E services on their networks based on the rapidly changing service/application landscape. Thus, the service providers may find themselves in a situation where just guaranteeing edge to edge QoS may not be enough, but E2E QoE may become prominent. This requires the service providers to test E2E QoE for the services. Annex C.2 lists some methodologies for measuring E2E QoE for popular services and applications.

Although passive monitoring techniques may be used for measuring E2E QoE, advanced DPI capabilities would be required, and performing real-time QoE using DPI becomes a challenge because of the high processing needs associated with it.

The following exhibit from [i.10] shows the various stages of QoE measurement and management.
7.5.2 Overload Detection

NFVI resources may be overloaded as new VNFs and services are provisioned. Auto-scaling and VNF migration techniques driven by policy management is one way of dealing with resource overload situations. [REL 002] describes a migration avoidance mechanism to deal with overload situations.

Before corrective actions can be taken, the first step is to detect the NFVI overload condition. This may be done by a combination of active and passive monitoring techniques. Some of the NFVI analytics techniques have been proposed as part of OPNFV Doctor Project [i.11] which recommends a monitor, inspector and notification module for detecting and notification of alarms and error conditions. Similar techniques may be used to set thresholds for the NFVI resources that trigger the appropriate notifications when the overload condition is detected. In the overload detection scenario, we need to make a distinction between overload of the NFVI node as a whole, and overload condition that may result in performance degradation of certain services only. In the second scenario, setting thresholds at the NFVI level is a challenge as the thresholds pertaining to individual services may not be consistent during the lifetime of the service. As more services and VNFs are provisioned and traverse the same intermediate nodes, the NFVI thresholds do not accurately correlate with the performance of the E2E service.

This presents a need to periodically verify if the service specific NFVI thresholds accurately correlate with service performance SLAs. Doing this allows the operators to adjust the service specific thresholds and detect the overload condition timely and accurately.

8 Evaluating NFV Resiliency

Service providers have much more stringent requirements as compared to Enterprise and DC when it comes to reliability and resiliency. Sub-30 milliseconds of convergence time, 99.999% reliability and a fault OAM management capability are some of the top requirements for traditional SP networks. These requirements in turn result in requirements for NFV based networks such as “low packet processing overhead, low latency and efficient virtual switching, along with automatic recovery from faults and extremely high availability.” [i.14].
8.1. Network Resiliency Evaluation

Traditionally network resiliency or fault-tolerance is built into the network by design using point of failure analysis (POF). The resiliency design of the network may be validated by simulation techniques as described in [i.19]. The simulation techniques involve simulating the network components and subjecting the network to various challenges that imitate faults and failures. These challenges have been described in detail as part of [i.20] Clause 6 and Clause 7.

Resilinets principles as explained in [i.19] form a good foundation and context for understanding the evaluation of resiliency for NFV based networks.

![Resilinets principles](image)

**Figure 16 Resilinets principles [i.19]**

Following are the pertinent Resilinets principles for evaluating network resiliency

- **Prerequisites**
  - Service Requirements: Determines the level of resiliency the system needs to provide for the service. A set of parameters define if the resiliency level is acceptable, impaired or unacceptable.
  - Normal Behaviour: Understanding normal behaviour for the service in the evaluating resiliency.
  - Threat and Challenge Models: [i.20] defines in detail the categories for various challenges and also defines a threat and challenge catalogue which may be used for evaluating the resiliency of the network.
  - Metrics: Quantification of the service requirements (acceptable, impaired, unacceptable) and operational state (normal, partially degraded and severely degraded).

- **Enablers**
  - Multilevel resilience: [i.19] defines the protocol layers, planes and network architecture dimensions for multilevel resiliency. NFV adds more variables as the network node implementations are distributed across VNF and NFVI components. In addition shared infrastructure in NFVI adds more complexity and makes it difficult to design the network for the desired resiliency levels.

Protocol level resiliency has been explored in many standardized protocols such as MPLS protection switching, unicast routing protocol convergence, RSVP-TE fast reroute or Loop-free alternate (LFA) using segment routing. Similarly, there has been work done by SDOs such as IETF on the lines of SRLG and geographic level redundancy. Diversity is an
important measure in ensuring that multiple alternate network components or network paths do not share the same fate and thus add to the resiliency of the network. [i.19] proposes methods for evaluating path diversity, geographic diversity and graph diversity. This document is not intended to address diversity measures, but it is important to note that such work as illustrated in [i.19] has been defined to improve network resiliency. Additionally, the concept of diversity has been dealt in detail as part of ETSI NFV WI [i.22] as well.

Faults and failures are further explored in [i.19] as challenge modelling where challenges are considered as events that affect normal operation and threats are considered as potential challenges. [i.20] takes the challenge modelling to explore further into the virtualization and NFV domain. This document suggests to leverage the work done in [i.19] and [i.20] for challenge modelling in the NFV environment.

8.2. NFV Resiliency Evaluation using active fault injection

NFV is characterized by multi-vendor shared infrastructure environment with increased variability in performance. Evaluating network resiliency in such an environment becomes a challenge.

Simulation techniques are helpful in evaluating resiliency of the non-NVF networks where behaviour of network nodes is deterministic and well-understood. For NFV based networks the performance of a given VNF or a network service is dependent on the shared NFVI resource allocation and the type and number of other services and VNFs provisioned on the same resources. In such multi-vendor shared infrastructure environment it is not possible to simulate the network behaviour or the response of the network to threats and challenges that affect resiliency of the network. This necessitates the resiliency evaluation in the actual network before turn up and using test traffic that emulates the real-world workload scenarios.

It is insightful to look at how CSMIC [i.15] defines resiliency for services in cloud environments and how NIST [i.16] has defined a tentative procedure based on CSMIC resiliency metrics to calculate a resiliency score. The resiliency score defined in [i.16] proposes a measure to compare the resiliency of the various cloud platforms provided that the same underlying measures and measurement rules are used for the compared scenarios.

This document proposes a methodology using active test traffic and fault injection at turn up time on the same lines as the methods described in [i.19]

[i.19] defines challenge categories and REL001 describes the applicable challenges and threats that can impact NFV resiliency.

Such challenges may be generated by the test controller that interfaces with NFVO via Os-Ma-Nfvo interface and has access to VIM and NFVI. In the scenario where certain challenges or faults cannot be generated via the Os-Ma-Nfvo interface, the test controller may interface with the NFVI components directly as the NFVI and test controller reside under the same administrative domain.

It is understood that challenges that cause failures and service degradation cannot be applied to the live network, hence the resiliency evaluation using challenges and threats are limited to turn up scenarios, where a new section of a network such as a data center or central office is provisioned.

8.2.1. Fault Injection framework for evaluating NFV resiliency
The framework utilizes the Test Controller component to provision virtual impairment generators to run experiments that enable injecting faults for various types of workloads and at various component levels such as NFVI layer, virtualization layer, and VNF layer or at NFV MANO components. It should be noted that when varying the number of faults, type of faults and location of faults, the workload generated for the services under test must be kept constant. Additionally, the performance metrics collected to determine if the service level is acceptable, degraded or unacceptable as defined in [i.19] must be the same for a given workload type. Please note that it is important that the workloads used to represent the provisioned services must closely imitate the real-world scenarios.

The Test Controller is used to run challenge/fault campaigns, co-ordinate the generation of service workloads and collection of test results. The faults may be injected in transient, intermittent and permanent ways for varying amount of time (short duration, long duration or permanent) to emulate different scenarios. Note that in some scenarios, exposure to faults surface as service degradation or failure only when the faults are sustained for longer period of times. Some of the examples for these faults are failed read/write due to bad disk sectors, emulate I/O errors that represent errors due to worn-out connectors or partially damaged h/w interfaces.

The objective of running the fault injection campaigns is to isolate single points of failures or determine the fault-tolerance of the NFV network. Thus selecting the fault injection location is a decision that may be left to the service provider. Some of the locations for fault injection that may be considered are:

- Load balancers which provide VNF redundancy for failover scenarios
- Ingress and Egress nodes for a service path. The purpose here is to validate the multi-homing operation that may have been configured for the ingress and egress nodes.
- Intermediate nodes
- VNFC
8.2.2. Multilevel Challenge/Fault modelling

![Service Interface at architecture layers](image)

[Fig. 18 Service Interface at architecture layers [i.19]]

[i.19] defines multi-level approach in terms of the OSI layers. In the example shown in Figure 18 the service interface under consideration is the network layer.

As described in [i.19] “The ability of a network layer entity, such as a routing protocol, to find the end-to-end paths is based on the mechanisms it employs and the connectivity of the topology provided by the lower layers. Resilience is evaluated as the ability to find quality paths (service parameter) under the presence of challenges that affect the link connectivity and routing mechanisms (operational metrics)”

On the same lines, the NFV architecture can be divided into VNF layer, virtualization layer and NFVI layer as shown in Figure 19. NFVI layer consists of compute, storage and network nodes that enable VNF operation via the virtualization layer. A fault/challenge at NFVI or virtualization layer may degrade the operation of VNF affecting the E2E service performance below acceptable level. In this case acceptable service level may be defined based on the SLA performance metrics for the service.
In NFV environments, a VNF may provide equivalent of a L3, L4 or L7 service, e.g., router VNF would operate at L3, whereas firewall VNF would operate at L3, L4.

Depending on the layer at which VNF operates the corresponding faults/challenges may be applied to the VNF.

8.2.3. NFVI faults & failures

[REL001] defines a fault/challenge catalogue and describes the challenges for the NIST virtualization categories. This section focuses on the NFVI related faults/failures that impact the VNF operation and thus affecting the E2E service.
The above exhibit from [i.14] lists the fault categories. Some of these faults as explained in [i.14] manifest themselves as disruption of I/O traffic. These faults may be generated using some of the techniques specified in [i.14]. Tools listed in [i.14] such as “CloudVal”, “FATE” and “PreFail” inject faults by intercepting method calls or library calls for disk failures, network partitions or crash of remote processes. These tools are mainly focused on testing cloud software against multiple faults and may be used for injecting faults in NFV based networks while running experiments and measuring the reliability and resiliency.

Some of these faults such as CPU or memory hogs may be reproduced by provisioning additional workloads via the Test Controller on the shared NFVI nodes. CPU faults and memory faults may result in reduced number of CPU cores available to VNFs or reduced memory available to VNFs. This can be used to validate the ability of the NFV framework to automatically detect the CPU core failure or memory failure and re-adjust the NFVI resource allocation to minimize the performance impact to E2E service.

Additionally, commercial tools may be used to generate packet level errors, delay, jitter for E2E services to assess the impact of impairments and faults on the resiliency of the E2E service. Injecting faults such as packet drops, delay and jitter not only helps in evaluating the NFV network resiliency at NFVI and VNF level, but it also tests the combined resiliency built by the control plane protocols and NFV network.
VNF such that the Test Controller can provision them via NFVO using mechanisms similar to that defined in MAN001 GS document [i.2].

8.3. Traffic Tolerance

![Figure 23 Resiliency: Ability to process incoming service requests [i.18]](image)

Another aspect of NFV network resiliency is the ability of the network to handle incoming requests for new service or VNF provisioning. This can be measured in terms of

1. Maximum rate at which incoming requests are accepted with no drop
2. Provisioning delay after the service request has been received.

The provisioning delay in case of NFV is a combination of delays across various MANO entities and NFVI layer whereas the accepted request rate is a function of the size of the input buffers at entities traversed by the service request and the NFVI capacity.

For applications such as mobility, the service request rate can be relatively much higher than other services. In such scenarios, ability of the NFV framework to handle the incoming service requests under varying mix and level of background workloads is an important measure to evaluate resiliency.

8.4. Failure Impact

Injecting faults using impairment generators helps in providing the trigger events to evaluate the resiliency, but at the same time it is important to understand the dimensions and level of impact on the E2E service due to the faults. That provides a method to evaluate if the resiliency design of the network is capable of satisfying the E2E service performance SLAs in the event of faults. Some of the dimensions to look at include:

- **Performance measures** such as throughput, goodput, delay, latency, delay variation, provisioning delay for incoming service requests, maximum rate to process incoming service requests
- **Security assessment** such as vulnerability of the network to DDoS attacks
- **Auto-scale capability assessment** such as capacity to auto-scale, auto-scale latency
10 Security Considerations

Passive and Active Security Monitoring of NFV networks to monitor any malicious activity is being covered in detail as part of [i.19], but it is important to understand the security aspects applicable to the Active Monitoring architecture proposed in this WI.

Following are the key security aspects to maintain the integrity of the NFV network so that the deployment of active monitoring framework does not render the security of NFV network any more vulnerable.

- Secure & encrypted interface is required between Test Controller, VTA and TRAM. A compromised communication between these components can expose these components to unauthorized test instructions being delivered to VTAs and can potentially compromise the performance of the NFV network. Additionally, a compromised telemetry information to and from TRAM can reveal important topology and performance information, and provide an opportunity for a malicious attach.
- Following security aspects defined in [i.19] Clause 9.3 are applicable for VTA, TRAM and Test Controller as well.
  - “Active Monitoring components shall be securely provisioned within the system, which means that these systems will be provisioned for deployments in a trusted environment. This includes root key provisioning, setting up Trusted Execution Environments, certificate provisioning, etc.”
  - “Active Monitoring components shall be booted using secured boot technologies.”

- Test Controller uses the Os-Ma-Nfvo interface to communicate with the NFVO to provision VTAs, retrieve network service information and retrieve NFVI telemetry data from VIM. Since the Os-Ma-Nfvo interface is part of the “Trusted Execution Environment” as defined in [6], additional security requirements for this interface are not required.
- In addition to the encrypted communication as defined earlier, the session between Test Controller, VTA and TRAM must be authenticated priori to any other communication between the entities.
- [i.19] Section 9.4 lists requirements for the Secure Telemetry Access. These requirements are equally applicable to for the results reported by VTA or results reported by VIM to TRAM and the instructions sent by Test Controller to VTA or TRAM. In addition security measures must be taken to secure the databases that contain the Active Monitoring results.
- Test Controller and TRAM are in the same administrative domain as OSS/BSS and are in the same “Trusted Execution Environment”. The security measures in use for OSS/BSS are applicable for Test Controller and TRAM as well.
11 Recommendations
Annex A (informative):

A.1 Why Active Monitoring

Monitoring techniques used for live networks fall into two high level categories: passive and active monitoring. Passive monitoring involves analysing real-time user traffic and active monitoring involves sending test traffic to assess the health of the network. In passive monitoring, measurements can only be analysed off-line, and not as they are collected. This not only creates problem with processing the huge data sets that are collected, but also introduces a time delay in getting any sort of actionable results. Following are some additional limitations when using only passive monitoring for network health analysis and troubleshooting.

- When E2E services cross network boundaries, collecting monitoring data from passive monitoring probes that are under different administration domains become an issue.
- Passive monitoring can only provide monitoring data for flows at each of the intermediate points. This data needs to be collected and has to go through extensive analysis later. The analysis is complex and does not necessarily provide a context for E2E services. The subsequent analysis also results in late availability of actionable results.
- Data privacy and lawful intercept further complicate the efficacy of the passive monitoring solution, whereas active monitoring solution does not depend on analysing user traffic.
- Active monitoring allows isolation of segments, VNFs or NFVI resources, through an iterative approach which passive monitoring cannot provide.
- Service providers can run controlled experiments to understand the impact of any particular application or service becoming popular exponentially, and help in future capacity planning in the wake of such events.
- Active monitoring enables proactive fault detection, performance degradation, configuration issues.

A.2 Virtual Test Agent Implementation Considerations

A.3 Test VNF

A.3.1 Test VNF Descriptor

In addition to the VNFD base information elements defined in Section 6.3.1.1 of [i.3], the following information elements may be defined for test VNFs. These information elements are specific to the communication between test VNF and Test Controller, TRAM. So these information elements should be part of the active monitoring application itself and support is not required by any of the MANO components.
<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test_Controller_ID</td>
<td>Leaf</td>
<td>1</td>
<td>ID of the Test Controller that the test VNF is bound</td>
</tr>
<tr>
<td>BackUp_Test_Controller_ID</td>
<td>Leaf</td>
<td>0..1</td>
<td>ID of the backup Test Controller</td>
</tr>
<tr>
<td>Authentication_Params</td>
<td>Element</td>
<td>1</td>
<td>Authentication parameters used by the test VNF to authenticate with the Test Controller.</td>
</tr>
<tr>
<td>Test_Group_ID</td>
<td>Leaf</td>
<td>0..N</td>
<td>Multiple test VNFs may share the same Test Group ID. This helps in keeping the results obtained from the test VNF anonymous, and thus helps in addressing the privacy concerns?</td>
</tr>
<tr>
<td>Measurement_Methods</td>
<td>Element</td>
<td>0..N</td>
<td>Measurement methods supported by the Test VNF. These are based on the feature capabilities of the test VNF.</td>
</tr>
</tbody>
</table>

**Authentication_Parameters**

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X.509 Security_Certificates</td>
<td>Element</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Measurement_Methods**

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y.1564</td>
<td>Element</td>
<td>0..1</td>
<td>Service activation testing</td>
</tr>
<tr>
<td>Y.1731</td>
<td>Element</td>
<td>0..1</td>
<td>Fault OAM</td>
</tr>
<tr>
<td>RFC_2544</td>
<td>Element</td>
<td>0..1</td>
<td>Network benchmark testing</td>
</tr>
<tr>
<td>TWAMP</td>
<td>Element</td>
<td>0..1</td>
<td>Two way active measurement protocol</td>
</tr>
<tr>
<td>RFC_6349</td>
<td>Element</td>
<td>0..1</td>
<td>TCP throughput testing</td>
</tr>
</tbody>
</table>

**Test Capabilities**

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2-L3</td>
<td>Leaf</td>
<td>0..N</td>
<td>Throughput, latency, jitter, convergence</td>
</tr>
<tr>
<td>L4-L7</td>
<td>Leaf</td>
<td>0..N</td>
<td>Goodput, transactions/sec, processing delay</td>
</tr>
<tr>
<td>Control plane</td>
<td>Leaf</td>
<td>0..N</td>
<td>Protocols, session scale, sessions/sec</td>
</tr>
</tbody>
</table>

**A.3.2 Test VNF Record (VNFR)**

These information elements are specific to the communication between test VNF and Test Controller, TRAM. So these information elements should be part of the active monitoring application itself and support is not required by any of the MANO components.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test_Controller_ID</td>
<td>Leaf</td>
<td>1</td>
<td>ID of the Test Controller that the test VNF is bound</td>
</tr>
<tr>
<td>BackUp_Test_Controller_ID</td>
<td>Leaf</td>
<td>0..1</td>
<td>ID of the backup Test Controller</td>
</tr>
<tr>
<td>Authentication_Params</td>
<td>Element</td>
<td>1</td>
<td>Authentication parameters used by test VNF to authenticate with the Test Controller.</td>
</tr>
<tr>
<td>Test_Group_ID</td>
<td>Leaf</td>
<td>0..N</td>
<td>Multiple test VNFs may share the same Test Group ID. This helps in keeping the results obtained from the test VNF anonymous, and thus helps in addressing the privacy concerns.</td>
</tr>
<tr>
<td>Performance_Params</td>
<td>Element</td>
<td>1..N</td>
<td>Performance parameters of the test VNF such as L2-L3 throughput, L4-L7 throughput</td>
</tr>
</tbody>
</table>

45/62
### Performance_Params

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2-L3 Throughput</td>
<td>Leaf</td>
<td>0..1</td>
<td>L2-L3 throughput</td>
</tr>
<tr>
<td>HTTP Throughput</td>
<td>Leaf</td>
<td>0..1</td>
<td>HTTP throughput</td>
</tr>
<tr>
<td>TCP Throughput</td>
<td>Leaf</td>
<td>0..1</td>
<td>TCP throughput</td>
</tr>
<tr>
<td>Latency Accuracy</td>
<td>Leaf</td>
<td>0..1</td>
<td>Latency accuracy (ms, ns)</td>
</tr>
<tr>
<td>Fail-over Convergence</td>
<td>Leaf</td>
<td>0..N</td>
<td>Support, type and accuracy of convergence measure</td>
</tr>
</tbody>
</table>

### A.3.3 Test Instruction Set

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test_Instruction_ID</td>
<td>Leaf</td>
<td>1</td>
<td>Unique ID to identify the test instructions. This ID may be used to compare multiple result instances of the same test.</td>
</tr>
<tr>
<td>Network_Service_Record_ID</td>
<td>Leaf</td>
<td>1..N</td>
<td>ID for the network service record under test. The same test instruction set may be used to test multiple network services. Each network service record provides the Test Controller capability to access the required information such as VNFR list, VNFFGR, VLR and other information that may be required. Test Controller may retrieve this information via the NFVO external interface to access NSR catalogue information.</td>
</tr>
<tr>
<td>Measurement_Methods</td>
<td>Element</td>
<td>1..N</td>
<td>This element defines the tests that need to be executed and the corresponding input parameters required, e.g., if service activation test defined in Y.1564 needs to be run, then the associated mandatory input parameters must be specified.</td>
</tr>
<tr>
<td>TRAM_ID</td>
<td>Leaf</td>
<td>1..N</td>
<td>ID for test result analysis module that will be used for retrieving and analysing the test results</td>
</tr>
</tbody>
</table>
A.3.4 KeepAlive messages

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTA_Status</td>
<td>Leaf</td>
<td>0..1</td>
<td>Down, operational, busy, available</td>
</tr>
<tr>
<td>KeepAlive_Period</td>
<td>Leaf</td>
<td>0..1</td>
<td>KeepAlive period in ms</td>
</tr>
</tbody>
</table>

A.4 Test Measurement Methods

A.4.1 Fault Localization

1. Are multiple services affected?
   - N: Follow the procedure "Troubleshoot single service (SOLP)"
   - Y: Find commonalities (segment(s), NFVI Resources)

2. Are commonalities found?
   - N: Follow the procedure "Troubleshoot single service fault for each service that is affected"
   - Y: Add common segments to the Fault diagnosis segment list and common NFVI resources to the fault diagnosis NFVI list

3. Follow the procedure "Troubleshoot multiple service faults with commonalities"
Figure 24 Example Fault Localization Workflow
Annex A.4.2 NFVI Metrics for fault co-relation

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%system</td>
<td>Percentage of CPU utilization that occurred while executing at the system level.</td>
</tr>
<tr>
<td>%user</td>
<td>Percentage of CPU utilization that occurred while executing at the user level.</td>
</tr>
<tr>
<td>%nice</td>
<td>Percentage of CPU utilization that occurred while executing at the user level with nice priority.</td>
</tr>
<tr>
<td>%idletime</td>
<td>Percentage of time that the CPU or CPUs were idle during which the system had an outstanding disk I/O request.</td>
</tr>
<tr>
<td>%soft</td>
<td>Percentage of time spent by the CPU or CPUs to service software interrupts.</td>
</tr>
<tr>
<td>%steal</td>
<td>Percentage of time spent in involuntary wait by the virtual CPU or CPUs while the hypervisor was serving another virtual processor.</td>
</tr>
<tr>
<td>proc/s</td>
<td>Total number of tasks created per second.</td>
</tr>
<tr>
<td>ccwch/s</td>
<td>Total number of context switches per second.</td>
</tr>
<tr>
<td>intr/s</td>
<td>Total number of interrupts received per second by the CPU.</td>
</tr>
<tr>
<td>kbnemused</td>
<td>Amount of used memory in kilobytes.</td>
</tr>
<tr>
<td>kblbuffer</td>
<td>Amount of memory used as buffers by the kernel in kilobytes.</td>
</tr>
<tr>
<td>%memused</td>
<td>Percentage of used memory.</td>
</tr>
<tr>
<td>pswpin/s</td>
<td>Total number of swap pages the system brought in per second.</td>
</tr>
<tr>
<td>pswpout/s</td>
<td>Total number of swap pages the system brought out per second.</td>
</tr>
<tr>
<td>pgpsect</td>
<td>Total number of kilobytes the system paged in from disk per second.</td>
</tr>
<tr>
<td>pgpgoutt</td>
<td>Total number of kilobytes the system paged out to disk per second.</td>
</tr>
<tr>
<td>faults</td>
<td>Number of page faults (major+minor) made by the system per second.</td>
</tr>
<tr>
<td>pgseal/s</td>
<td>Number of pages the system has reclaimed from cache (pagecache and swapcache) per second to satisfy its memory demands.</td>
</tr>
<tr>
<td>pgscans</td>
<td>Number of page scanned by the kswapd daemon per second.</td>
</tr>
<tr>
<td>%snell</td>
<td>Calculated as pgseal/pgscans, a metric of the efficiency of page reclaim.</td>
</tr>
<tr>
<td>tps</td>
<td>Total number of transfer per second that were issued to physical devices.</td>
</tr>
<tr>
<td>wr_secs</td>
<td>Number of sectors written to the device.</td>
</tr>
<tr>
<td>rd_secs</td>
<td>Number of sectors read from the device.</td>
</tr>
<tr>
<td>tcpck/s</td>
<td>Total number of packets received per second.</td>
</tr>
<tr>
<td>tcpckd/s</td>
<td>Total number of packets transmitted per second.</td>
</tr>
<tr>
<td>tcp_tw</td>
<td>Number of TCP sockets in TIME_WAIT state.</td>
</tr>
<tr>
<td>cyclres</td>
<td>Total number of CPU cycles.</td>
</tr>
<tr>
<td>ITLB-load</td>
<td>Total number of load operations to the instruction TLB.</td>
</tr>
<tr>
<td>branches</td>
<td>Number of branch operations.</td>
</tr>
<tr>
<td>branch-misses</td>
<td>Percentage of branch misses with the total number of branches.</td>
</tr>
<tr>
<td>LLC-store-misses</td>
<td>Number of last-level cache store misses operations.</td>
</tr>
<tr>
<td>LLC-prefetches</td>
<td>Number of last-level cache prefetches operations.</td>
</tr>
<tr>
<td>Major-faults</td>
<td>Number of page major faults.</td>
</tr>
<tr>
<td>DTLB-load-misses</td>
<td>Number of load misses operation to the data TLB.</td>
</tr>
<tr>
<td>DTLB-stores</td>
<td>Number of stores operation to the data TLB.</td>
</tr>
</tbody>
</table>

A.5 Synchronization protocol definition for Test Controller HA

This section defines the synchronization mechanism between the primary and backup Test Controllers to maintain in-sync information on the VTAs assigned to the controller, test instructions for the tests under execution and test instructions for the periodically scheduled tests.

A.6 Result Analysis

Annex B

B.1 Service QoS & QoE measurements methods

The following table lists various test methodologies that are applicable to traditional, as well as NFV networks, to benchmark the QoS and QoE for E2E services. This list by no means is an exhaustive list, but presents the well-known test methodologies.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Service</th>
<th>Test and Measurements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet L2 EVC</td>
<td>Ethernet OAM Test: Circuit Availability, Loss, Latency, Delay Variation</td>
<td></td>
</tr>
<tr>
<td>Carrier</td>
<td>Ethersam Test (Y.1564): Throughput, Loss, Delay, Jitter</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>IP Connectivity</td>
<td>Ethernet OAM Test: Circuit Availability, Loss, Latency, Delay Variation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ethersam Test (Y.1564): Throughput, Loss, Delay, Jitter</td>
<td></td>
</tr>
<tr>
<td>Laas</td>
<td>DNS Resolution Test: DNS Availability, Response Time, Accuracy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ping Test: Reachability, Loss, Latency</td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDP ECHO Test: Availability, Loss, Latency</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VPN Connection Test: VPN Availability</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TWAMP: Reachability, Loss, Latency, Delay Variation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDP and TCP port Test: Port availability, TCP connection delay</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TCP Port Test: Port Availability, TCP Connection Delay</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Web URL Download Test: Web Server Availability, Response Time, Download Time, Throughput</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Web URL Download Test: Web Server Availability, Response Time, Download Time, Throughput</td>
<td></td>
</tr>
<tr>
<td>Paas</td>
<td>Web Page Download Test: Website Availability, Response Time, Download Time, Throughput (for all page content)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Web Request Test: Availability, Response Time, Download Time, Throughput</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cloud Storage Test: Availability, Response Time, Upload and Download Time, Throughput</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cloud Replication Test: Replication Time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cloud Storage Test: Availability, Response Time, Upload and Download Time, Throughput</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cloud Replication Test: Replication Time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Web Application Scripted Test: Availability, Success Failure of Script, Total Script Time, Each Step Time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Email Tests for SMTP, POP3 and IMAP: Email Availability, Response Time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Email Delivery Test: Message Delivery Time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VoIP Call Tests: Availability, MOS, Post Dial Delay, Loss, Latency, Jitter, Buffer Overflow (and Underflow)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Video Download Test: Availability, Download Time, Re-buffering, Video Quality</td>
<td></td>
</tr>
</tbody>
</table>

## B.2 User Traffic Workloads and Distributions

### B.2.1 Workload Mix

<table>
<thead>
<tr>
<th>Name</th>
<th>DiffServ (Real-Time)</th>
<th>DiffServ 0x31 (Critical)</th>
<th>DiffServ 0x20 (General)</th>
<th>DiffServ 0x00 (Best Effort)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enterprise Campus Apps</strong></td>
<td>VoIP 15%</td>
<td>Routing 3% OSPF Routing Updates 2%, BGP Updates 1%, Database 1.9% (Oracle SQL Net Updates), Corporate Web 2%, IMAP 5%</td>
<td>Multicast Video 13% (480i, MPEG2-5, 60/in, 5 Multicast Channels), Telnet/SSH (2%), CIFS 10% (1:1.3 Small/Medium/Large Ratio)</td>
<td>Internet Web 5% HTTP (1024 byte index.html), 30500 Byte JPEG, 51K JPEG, 1x100k (jpeg), BitTorrent 11%</td>
</tr>
<tr>
<td>Higher Education Network Administration 2% (SSH)</td>
<td>SQL 7% SQL/Net SQL Table Updates), HTTPS University Admin 3% (64 Bytes index.html), 5x 1K JPEG Images (560), VoIP 5% (0.725A COED/COM)</td>
<td>FTP 7% (Large Files), HTTPS Student Services, HTTP 3%, POP3/SMTP 9%, CIFS 8% (1:1.8 Small/Medium/Large Objects, bidirectional), Multicast Video 5% (400i)</td>
<td>IM 12% (AIM), BitTorrent 24%, HTTP 3% (1024 Byte index.html, 30500 Byte JPEG, 51K JPEG, 1x100k (jpeg), HTTPS 1% (64 Bytes index.html), 5x 1K JPEG Images), Mail 5%, FTP 1% (Large Files), Telnet/SSH 8%</td>
<td></td>
</tr>
<tr>
<td><strong>Service Providers</strong></td>
<td>Telnet/SSH 1%</td>
<td>BGP Route Updates 1%</td>
<td>N/A</td>
<td>No Payload, RAW TCP</td>
</tr>
<tr>
<td>1G max Bandwidth</td>
<td></td>
<td></td>
<td></td>
<td>No Payload, RAW TCP</td>
</tr>
<tr>
<td><strong>Small/Medium Business Apps</strong></td>
<td>POP2/SMTP 15% (5/2.1 Ratio of Small/Medium/Big ratio) HTTPS 20% (64 Bytes index.html, 5x 1K JPEG Images), CIFS 30% (1:1.3 Small/Medium/Large Objects, bidirectional), BitTorrent 10%, Internet Web 25% HTTP (1024 byte index.html), 30500 Byte JPEG, 51K JPEG, 1x100k (jpeg)</td>
<td>BitTorrent 10%</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>WAN Accelerator</strong></td>
<td>Network Control 5% (Windows Domain Controller Updates), Network Logins</td>
<td>HTTPS 10% (64 Bytes index.html, 5x 1K JPEG Images)</td>
<td>BitTorrent 10%</td>
<td></td>
</tr>
<tr>
<td><strong>Internet AppMix 2011</strong></td>
<td>CIFS 40% (1:1.3 Small/Medium/Large Files), Exchange 35% (5/2.1 Small/Medium/Large Ratio)</td>
<td>HTTP 7% (Large Files), HTTPS Student Services, HTTP 3%, POP3/SMTP 9%, CIFS 8% (1:1.8 Small/Medium/Large Objects, bidirectional)</td>
<td>BitTorrent 10%</td>
<td></td>
</tr>
<tr>
<td><strong>WAN Accelerator</strong></td>
<td>Network Control 5% (Windows Domain Controller Updates), Network Logins</td>
<td>HTTPS 10% (64 Bytes index.html, 5x 1K JPEG Images)</td>
<td>BitTorrent 10%</td>
<td></td>
</tr>
</tbody>
</table>

### B.2.2 Workload Distributions

- Staircase Load Pattern
- Flat Load Pattern with Ramp up
Annex C Example Measurement Methods and Metrics

C.1 Example SLAs

Table 1 Example Enterprise service performance monitoring SLA ([i.7])

<table>
<thead>
<tr>
<th>Component</th>
<th>Time to Provision</th>
<th>Time to Use</th>
<th>Time to Restore</th>
<th>Time to Report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface (Kbps)</td>
<td>Load time &lt; 2 s</td>
<td>≥99% of the time,</td>
<td>Refresh time will be</td>
<td>Logging will trail by</td>
</tr>
<tr>
<td></td>
<td>seconds on average</td>
<td>response time will</td>
<td>≤ reload time</td>
<td>≤ latency + 10 ms always</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≤ latency+10 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Storage (GB)</td>
<td>Time to populate</td>
<td>Time to perform</td>
<td>Time to restore from</td>
<td>Time to report after</td>
</tr>
<tr>
<td></td>
<td>data ≤ 100 mbps</td>
<td>standard agreed query will be</td>
<td>backup will be ≤</td>
<td>a change will be ≤ 1</td>
</tr>
<tr>
<td></td>
<td>internal transfer</td>
<td>&lt; 100 mbps 99% of the time</td>
<td>100mbps internal transfer rate</td>
<td>second after transfer completed 99%</td>
</tr>
<tr>
<td></td>
<td>rate always</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connectivity (MB/S)</td>
<td>Time to connect</td>
<td>Time to verify end to end</td>
<td>Time to re-establish</td>
<td>Time to electrically</td>
</tr>
<tr>
<td></td>
<td>will be ≥ 10 ms +</td>
<td>will be ≤ (latency x 2) + 10</td>
<td>connection will be ≤</td>
<td>report performance</td>
</tr>
<tr>
<td></td>
<td>latency 99% of the</td>
<td>ms 99% of the time</td>
<td>latency + 10 ms 99% of the</td>
<td>internally will be ≤ 1 ms 99% of the</td>
</tr>
<tr>
<td></td>
<td>time</td>
<td></td>
<td>time</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Processing (MIPS)</td>
<td>Clock speeds will be</td>
<td>Cache performance will be so many</td>
<td>Single/double fault recovery times will</td>
<td>Core dump will require ≤</td>
</tr>
<tr>
<td></td>
<td>as contracted in</td>
<td>MIPS as tested</td>
<td>be ≤ microseconds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GHz ≥ 99% of the time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Federated Access (Security = SAML)</td>
<td>Establish new access ≤ 5 seconds 99% of the time, never to exceed 300 seconds</td>
<td>“Alert” on access grant/use/removal; ≤ 1 second 99% of the time, never to exceed 10 seconds</td>
<td>Reconnect ≤ 500 ms 99% of the time, never to exceed 10 seconds even if new access is being refused</td>
<td>Logging will trail ≤ 30 seconds 99% of the time, never to exceed 300 seconds</td>
</tr>
</tbody>
</table>
Table 2 Example cloud transport connectivity service attributes and SLA ([i.7])

<table>
<thead>
<tr>
<th>Service</th>
<th>Priority</th>
<th>CIR</th>
<th>EIR</th>
<th>Frame Delay</th>
<th>Delay Variation</th>
<th>Loss</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>VoIP calls</td>
<td>0</td>
<td>10 mbit/s</td>
<td>0</td>
<td>5 ms</td>
<td>&lt; 1 ms</td>
<td>0.1%</td>
<td>≥ 99.99%</td>
</tr>
<tr>
<td>Telepresence</td>
<td>1</td>
<td>50 mbit/s</td>
<td>0</td>
<td>25 ms</td>
<td>&lt; 10 ms</td>
<td>0.1%</td>
<td>N/A</td>
</tr>
<tr>
<td>Mission critical data</td>
<td>2</td>
<td>25 mbit/s</td>
<td>0</td>
<td>5 ms</td>
<td>&lt; 1 ms</td>
<td>0.01%</td>
<td>≥ 99.995%</td>
</tr>
<tr>
<td>Streamed live content</td>
<td>3</td>
<td>40 mbit/s</td>
<td>0</td>
<td>5 ms</td>
<td>&lt; 1 ms</td>
<td>0.01%</td>
<td>≥ 99.99%</td>
</tr>
<tr>
<td>Non real-time content</td>
<td>4</td>
<td>15 mbit/s</td>
<td>500 mbit/s</td>
<td>25 ms</td>
<td>10 ms</td>
<td>1%</td>
<td>≥ 99%</td>
</tr>
</tbody>
</table>

C.2 Application Test Methodologies for QoE measurements

C.2.1 NetFLIX™ Adaptive Streaming Load Generator with Quality Detection

NetFLIX™ adaptive Video Streaming service can present a substantial load upon a network infrastructure. This test emulates NetFLIX™ streaming service across the Device Under Test (DUT), which is a NFV based network. Measuring the impact of NetFLIX™ streaming service on the network can be invaluable when sizing a network.

NetFLIX™ streaming video service is a popular VoD service across the Internet. The service can take up substantial bandwidth and require QoS and tunes services for proper delivery. Specifically, NetFLIX™ streams take up the following network resources:

<table>
<thead>
<tr>
<th>Stream Type</th>
<th>Data Rate</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Netflix Movies (HD)</td>
<td>3.8Mbit/s</td>
<td>3600 MB for a 2-hour HD movie</td>
</tr>
<tr>
<td>Netflix Movies (SD)</td>
<td>1.3 Mbit/s</td>
<td>500-700 MB depending on movie length</td>
</tr>
<tr>
<td>Netflix TV Shows (HD)</td>
<td>1.0 Mbit/sec</td>
<td>1500 MB for a 30-minute TV show</td>
</tr>
<tr>
<td>Netflix TV Shows (SD)</td>
<td>700 kbps</td>
<td>400 MB for a 30-minute TV show</td>
</tr>
</tbody>
</table>
Test Procedure

1. On the server side, configure an emulated server farm for each class of video. Use emulated content and a minimum of 255 servers.

2. On the client side, create a user profile per port. In the action list, randomly choose a server from the correct pool. Then stream the video.

3. On the client side, create a unique load profile per user population with a specification of simulated users. Use a ramp and sustain pattern, ramping up to the desired simulated user count. Set the duration to the user specified test duration.

4. Start traffic.

5. Measure errors, MDI Scores, and bandwidth metrics and report.

Control Variables and Relevance

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relevance</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Duration</td>
<td>Total test Duration</td>
<td>30 minutes</td>
</tr>
<tr>
<td>NetFLIX Movies (HD) User Count</td>
<td>Concurrent number of Users</td>
<td>100</td>
</tr>
<tr>
<td>NetFLIX Movies (SD) User Count</td>
<td>Concurrent number of Users</td>
<td>100</td>
</tr>
<tr>
<td>NetFLIX TV (HD) User Count</td>
<td>Concurrent number of Users</td>
<td>100</td>
</tr>
<tr>
<td>NetFLIX TV (SD) User Count</td>
<td>Concurrent number of Users</td>
<td>100</td>
</tr>
<tr>
<td>Internet Cloud Latency</td>
<td>Fixed Latency Across the Emulated Internet in ms</td>
<td>45 ms</td>
</tr>
<tr>
<td>Internet Packet Loss</td>
<td>Typical Internet packet Loss</td>
<td>3%</td>
</tr>
</tbody>
</table>

Key Measured Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Relevance</th>
<th>Metric Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth by video class</td>
<td>Bandwidth Delivered by the network</td>
<td>Mbps</td>
</tr>
<tr>
<td>Errors</td>
<td>Errors caused by the network</td>
<td>Count</td>
</tr>
<tr>
<td>MDI Score</td>
<td>Video Quality</td>
<td>Count</td>
</tr>
</tbody>
</table>

C.2.2 HTTP Adaptive Streaming Video Start Time and Underflow Scale Test

QoE is a highly sensitive attribute of HTTP adaptive streaming service. One aspect of QoE is the ability for the desired video to start within a small and predictable window and for the video to not pause in the middle of play due to bitrate underflow. This test measures video start time and underflow as network conditions change and users scale. Without testing start time and underflow, user perception of the video service within the network may be challenged.

The ability of the server and network to reliably deliver video to the client in a timely manner with quality will regulate the user’s perception of the video service as a highly desirable and reliable service.

Test Procedure
1. Setup the HTTP adaptive server with the bitrate described in the control variables and relevance section of this document. Start the server. The server should send to the clients the manifest of the deliverable bit rates. The reference video should be 300 seconds long and of a known quality.

2. On each client port, setup a user profile representing one of the four classes of users described below. Within the user ActionList, have each user request and stream the video. Each user should wait a minimum of 15 seconds and then switch to the next bit rate.

3. Setup the following load profiles per class, with current subscriber as described in the control variable and relevance session below. The total length of time per current subscriber should be 15 minutes.
   a. Smartphone
      i. Create a burst pattern bursting zero users to the current number of subscribers.
   b. Tablet
      i. Ramp up to 50% of the current subscribers, and then create a sawtooth pattern from 50% to the current number of subscribers.
   c. PC
      i. Ramp to 50% of current subscribers. Keeping at least 50% of the current subscribers, randomly generate users up to the current subscribers

**Control Variables and Relevance**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relevance</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartphone Bitrate List</td>
<td>Rates to test in kbps</td>
<td>(50, 100, 300, 500, 1M)</td>
</tr>
<tr>
<td>Tablet Bitrate List</td>
<td>Rates to test in kbps</td>
<td>(100, 300, 500, 1M, 1.5M)</td>
</tr>
<tr>
<td>PC Bitrate List</td>
<td>Rates to test in kbps</td>
<td>(300, 500, 1M, 1.5M, 3M)</td>
</tr>
<tr>
<td>HDTV Bitrate List</td>
<td>Rates to test in kbps</td>
<td>(3M, 5M, 8M)</td>
</tr>
<tr>
<td>Concurrent Subscribers</td>
<td>Users per class</td>
<td>(5, 10, 100, 1000, 10000, 50K, 100K)</td>
</tr>
<tr>
<td>MGPEG GOB Size</td>
<td>In mSec, per bitrate and class</td>
<td>~500 nSec</td>
</tr>
<tr>
<td>TIA-921 / G.1050 List</td>
<td>List of Impairments</td>
<td>1A, 1C, 1F, 1H</td>
</tr>
</tbody>
</table>

**Key Measured Metrics**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Relevance</th>
<th>Metric Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video Start time</td>
<td>Time when video starts for a subscriber</td>
<td>ms</td>
</tr>
<tr>
<td>Pause Count</td>
<td>How many PAUSE underflow events occurred</td>
<td>List</td>
</tr>
</tbody>
</table>

**C.2.3 CloudMark Virtualized Performance Benchmark**

This benchmark tests the elements and systems of a cloud and ranks performance. We can expect NFV based networks to be deployed on shared infrastructure similar to that used by cloud service
providers. This test provides a method to validate the SLAs for the already provisioned or to be provisioned services.

**Description**

The cloud is composed of switch fabrics, virtual switch fabrics, physical network elements, virtual networking elements, physical servers, virtual servers and client endpoints. A benchmark is required to measure the performance of the cloud infrastructure in a comparable, independent fashion. This benchmark test measures the performance of the cloud infrastructure. Cloud Infrastructure performance can be measured using the following test cases. Cloud Infrastructure Reliability (CiR) is the failure rate of the cloud infrastructure to provide the environment to allow cloud protocols to operate without infrastructure-related errors. The generalized goal of 99.999% uptime, means CiR<=0.001%.

Cloud Infrastructure Quality of Experience (CiQoE) is the ratio of QoE of the protocols flowing over the Cloud Infrastructure to a client connected to a server though a back-to-back cable. By expressing QoE as a normalized set compared to a back-to-back ratio, the local VM operating system, server implementation, etc. are normalized. The Cloud Infrastructure Quality of Experience variance (CiQoE Variance) is the variance over time of the user experience. As a general rule, a variance measurement should be from a sample of 12 hours or longer. Further, this measurement determines the reliability of the cloud to act in a deterministic fashion.

Cloud Infrastructure Goodput (CiGoodput) measures the ability of the cloud to deliver a minimum bitrate across TCP.

**Test Procedure – Cloud Infrastructure Reliability Test (CiR)**

1. Begin servers on physical and virtual endpoints.
2. Set the loading profile to a 45 degree ramp up to a value in excess of the DUT.
3. Start Table 1 Traffic on client and server.
4. Client traffic should be evenly split between physical server and virtual server endpoints.
5. Start test traffic client.
6. Stop once either a failed transaction or fail connection occurs.
7. Set a new loading profile to ramp up to the measured failure point minus one connection or transaction. In the case of multiple failures, use the SimUser count for the lowest value minus 1. In the sustaining portion of the load profile, run the duration in excess of 12 hours.

8. Start traffic.

9. Stop traffic if and when a failed connection or transaction was detected.

10. Calculate the CIR by creating the ratio of 1 failure divided by the cumulative number of SimUsers.

11. In the case of no failure, keep doubling the time until a failure is reached or until the CIR Ratio becomes less than 0.001%.

12. The CiR is reported as “X % reliability at Y concurrent user sessions.”

Control Variables and Relevance

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relevance</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of open users</td>
<td>Peak level of reliability</td>
<td>Measured</td>
</tr>
<tr>
<td>Cumulative successful SimUsers session before failure</td>
<td>Used to build ratio of reliability</td>
<td>Measured</td>
</tr>
<tr>
<td>Cumulative users at Failure minus one</td>
<td>Upper level of measurement</td>
<td>Measured</td>
</tr>
<tr>
<td>Test Duration</td>
<td>Time of Steady State Phase</td>
<td>60 minutes</td>
</tr>
</tbody>
</table>

Key Measured Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Relevance</th>
<th>Metric Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>CiR</td>
<td>Ratio of first failure to the number of open SimUser sessions of success</td>
<td>percent</td>
</tr>
</tbody>
</table>

Test Procedure – Cloud Infrastructure Quality of Experience (CiQoE)

1. Calculate the virtual server QoE baseline.
   a. Turn off all background traffic.
   b. Using the service flow describe above, setup a single virtual endpoint as a client and a single virtual endpoint as a server. The pathway should traverse the virtual switch fabric.
   c. The virtual client should run one user session to the virtual server.
   d. Measure the QoE metrics as describe above. These become the baseline for the virtual servers.
   e. Reset all virtual endpoints as virtual servers.
2. Calculate the physical server QoE baseline.
   a. Turn off all background traffic.
b. Using the service flow described above, setup a single physical endpoint as a client and a single physical endpoint as a server. The pathway should traverse the virtual switch fabric.

c. The physical client should run one user session to the physical server.

d. Measure the QoE metrics as described above. These become the baseline for the physical servers.

3. Use the loading profile in the CiR test (Error minus one). Set the load profile to send 50% of the traffic to the virtual servers and 50% to the physical servers. Ramp up to the peak value and sustain for the desired duration of the test. (Minimum of 60 minutes.)

Start traffic.

5. If any QoE metrics failed, determine the number of concurrent SimUsers at failure and adjust the ramp down to that level. Go to Step 4 until no QoE failures are detected.

6. Measure the maximum value measure (Longest page load time, slowest FTP transfer, and smallest MOS-AV and MOS-LQ scores).

7. Divide the measure QoE number by its baseline equivalent. This is a percent impact of the infrastructure on traffic.

8. CiQoE is this calculated percent impact by protocol at a peak concurrent SimUser count.

Control Variables and Relevance

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relevance</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peek Concurrent SimUsers with No QoE Errors</td>
<td>Upper limit of users</td>
<td>Measured</td>
</tr>
<tr>
<td>Baseline QoE Values</td>
<td>Perfect case value</td>
<td>Measured</td>
</tr>
<tr>
<td>Measure Infrastructure QoE Values</td>
<td>Cloud impacted QoE Metrics</td>
<td>Measured</td>
</tr>
<tr>
<td>Test Duration</td>
<td>Time of Steady State Phase</td>
<td>60 minutes</td>
</tr>
</tbody>
</table>

Key Measured Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Relevance</th>
<th>Metric Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>CiQoE</td>
<td>Quality of Experience Impact</td>
<td>percent</td>
</tr>
</tbody>
</table>

Test Procedure – Cloud Infrastructure Quality of Experience Variance (CiQoEv)

1. Calculate the virtual server QoE baseline.
   a. Turn off all background traffic.

b. Using the service flow described above, setup a single virtual endpoint as a client and a single virtual endpoint as a server. The pathway should traverse the virtual switch fabric.

c. The virtual client should run one user session to the virtual server.
d. Measure the QoE metrics as described above. These become the baseline for the virtual servers.

e. Reset all virtual endpoints as virtual servers.

2. Calculate the physical server QoE baseline.

a. Turn off all background traffic.

b. Using the service flow described above, setup a single physical endpoint as a client and a single physical endpoint as a server. The pathway should traverse the virtual switch fabric.

c. The physical client should run one user session to the physical server.

Measure the QoE metrics as described above. These become the baseline for the physical servers.

3. Use the loading profile calculated in the CiR test (Error minus one). Set the load profile to send 50% of the traffic to the virtual servers and 50% to the physical servers. Ramp up to the peek value and sustain for the desired duration of the test. (minimum of 60 minutes.)


5. If any QoE metrics failed, demine the number of concurrent SimUsers at failure and adjust the ramp down to that level. Go to Step 4 until no QoE failures are detected.

6. Measure the maximum value measure (Longest page load time, slowest FTP transfer, smallest MOS-AV and MOS-LQ scores) every 4 seconds during the duration of the test.

7. By protocol, divide the measured QoE by the Baseline for each 4 second interval. This is the instantaneous cloud infrastructure impact percent.

8. With the set calculated, determine the standard deviation and variance.

9. The CiQoEv value is presented as a variance of at a measured standard deviation.

Control Variables and Relevance

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relevance</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peek Concurrent SimUsers with No QoE Errors</td>
<td>Upper limit of users</td>
<td>Measured</td>
</tr>
<tr>
<td>Baseline QoE Values</td>
<td>Perfect case value</td>
<td>Measured</td>
</tr>
<tr>
<td>Measure Infrastructure QoE Values</td>
<td>Cloud impacted QoE Metrics</td>
<td>Measured</td>
</tr>
<tr>
<td>Test Duration</td>
<td>Time of Steady State Phase</td>
<td>60 minutes</td>
</tr>
</tbody>
</table>

Key Measured Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Relevance</th>
<th>Metric Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>CiQoEv Variance</td>
<td>Variance of change</td>
<td></td>
</tr>
<tr>
<td>CiQoEv Std. Dev.</td>
<td>Deviation of change</td>
<td></td>
</tr>
</tbody>
</table>

Test Procedure – Cloud Infrastructure Quality of Experience Variance (CiGoodput)

1. Start traffic in Table 1.
2. Setup client and server traffic in a fully partial mesh. All clients should talk evenly to both virtual and physical servers.

3. Use the load profile calculated in the CiR test case.

4. Generate traffic for the desired duration.

5. Measure the minimum goodput achieved after the ramping phase by protocol.


Control Variables and Relevance

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relevance</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peek Concurrent SimUsers with No QoE Errors</td>
<td>Upper limit of users</td>
<td>Measured</td>
</tr>
<tr>
<td>Test Duration</td>
<td>Time of Steady State Phase</td>
<td>60 minutes</td>
</tr>
</tbody>
</table>

Key Measured Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Relevance</th>
<th>Metric Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum / Average / Maximum Goodput</td>
<td>Achievable goodput by protocol</td>
<td>bandwidth</td>
</tr>
</tbody>
</table>

Annex C.3.1 Example Test Methodology for Evaluating NFV Resiliency

Objective
Evaluate the resiliency of the NFV network at service or network turn up time on a real network in event of different types of faults at various NFV component levels

Test Setup
The test setup consists of a pre-defined set of services provisioned in the NFV based network. The Test Controller has the ability to provision the VTAs to generate realistic traffic to emulate the E2E service. The test traffic may be stateful or stateless based on the type of service being tested. The Test Controller should have the ability to provision virtual impairment generators (VIG), establish control plane sessions with the VIGs and send instructions that describe the type of faults, duration of the faults and location of the faults.

Test Procedure

- The Test Controller sets up VTAs to generate real world service traffic for the services provisioned in the network.
- The Test controller specifies fault injection targets, creates fault injection load distribution.
- The Test Controller instructs the VTAs to generate test traffic that emulates the service traffic. Run test traffic for short duration tests and medium duration tests. The exact value of the duration is up to the operator as long the duration of the test takes into account the time required to stabilize the input and output rate of the test traffic.
- Measure the performance metrics applicable to the service obtained from the test results and retrieve and catalog the NFVI performance metrics collected for the duration of the test. This set of metrics will form the baseline for the resiliency evaluation.
• The Test Controller injects faults at various locations and iteratively modifies the type of faults injected.
• For each fault type injected and for each fault location, record the service performance metrics and NFVI metrics.
• Repeat fault injection by varying the time for which the fault condition is maintained resulting in intermittent, periodic or permanent faults.
• Record the service performance metrics and NFVI metrics for each variation.
• The test cycle may be repeated by injecting multiple faults at the same time and assess the performance impact on services assisting in resiliency evaluation of the framework.

Test Results

The set of service performance metrics, fault information and NFVI usage and performance metrics can help in determining the following indicators that evaluate the resiliency/fault tolerance of the network and further assist in improving the network design to enhance the resiliency.

• Types of faults that have a greater impact on a given service. Note that all faults will not have similar performance impact on all services.
• Faults that impact NFVI performance and capacity the most.
• Impact of fault location on the performance of a given E2E service depending on the type of service.
• Co-relation between faults and service degradation behaviour to identify what type of faults and the severity of the fault will result in violation of the performance SLAs for E2E services.

Authors & contributors

The following people have contributed to the present document:
Rapporteur:
Gurpreet Singh (Spirent Communications)

Contributor:
Gurpreet Singh (Spirent Communications)
Eric Bauer (Alcatel-Lucent)
Randee S Adams (Alcatel Lucent)
Cameron Shearon (AT&T)
Don Topper (Huawei)
### History

<table>
<thead>
<tr>
<th>Document history</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>V0.0.1</strong></td>
</tr>
<tr>
<td><strong>V0.0.2</strong></td>
</tr>
<tr>
<td><strong>V0.0.3</strong></td>
</tr>
<tr>
<td><strong>V0.0.4</strong></td>
</tr>
<tr>
<td><strong>V0.0.5</strong></td>
</tr>
<tr>
<td><strong>V0.0.6</strong></td>
</tr>
<tr>
<td><strong>V0.0.7</strong></td>
</tr>
</tbody>
</table>
| **V0.1.0**       | 6 August 2015 | Added approved contribution for in-situ monitoring as Section 7.3, moved rest of the sections of 7.x by 1  
Added approved contribution for “Fault Notification Metrics” as Section 6 |
| **V0.1.1**       | 29 September 2015 | Added approved Section 8 and Section 9  
Updated REL004 draft document based on editorial feedback from REL WG |