ETSI Plugtests Report V1.0.0 (2018-02)

2nd ETSI NFV Plugtests
Sophia Antipolis, France
15th – 19th January 2018

ETSI

Keywords
Testing, Interoperability, NFV, MANO, VNF, VIM

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

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

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-préfecture de Grasse (06) N° 7803/88

Important notice

Copyright Notification

Contents

Executive summary

The 2nd NFV Plugtests was organised by ETSI Centre for Testing and Interoperability, and hosted by ETSI in Sophia Antipolis, France from 15th to 19th January 2018.

The main goal of the event was to run multi-vendor interoperability test sessions among different Functions Under Test (FUTs) from different participants, mainly Virtualized Network Functions (VNFs), Management and Orchestration (MANO) solutions and NFV platforms.

To prepare for the event, a remote integration phase was launched in October 2017 leveraging the NFV Plugtests Programme HIVE network to interconnect participants' labs. During the remote integration phase, participants validated connectivity and compatibility among the different FUTs participating in the Plugtests.

Following remote integration, a pre-testing phase allowed participants to run remote test sessions based on a subset of test cases extracted from the 1st NFV Plugtests test plan. These sessions focused on validating basic NFV capabilities such as management of descriptors and software images, as well as life cycle management of simple Network Services and Virtual Network Functions.

Finally, Plugtests participants gathered at ETSI for one intense week of face to face interoperability testing. During these multi-vendor test sessions the scope and complexity of multi-vendor interoperability test sessions was increased to address additional aspects such as multi-vendor Network Services, Fault and Performance Management, Enhanced Platform Awareness, and Multi-Site deployments.

The test plan was developed by the Plugtests team, led by ETSI's Centre for Testing and Interoperability and continuously reviewed to incorporate feedback from Plugtests stakeholders. It builds on the methodology and guidelines defined by the ETSI NFV TST working group and the test plan of the 1st NFV Plugtests.

In addition to the interoperability test sessions, an experimental track on NFV API compliance testing run by ETSI allowed to validate participants' API implementations against a subset of NFV SOL OpenAPIs. This track was very useful to gather experience and common understanding that will be leveraged to define a methodology for NFV Conformance Testing.

Key open source communities working in the NFV space, such as ETSI OSM, Open Baton, Open Stack and OPNFV were also actively involved in the Plugtests and participated to the test sessions through their membership with several implementations. Moreover, the co-located 1st OSM Hackfest offered an additional opportunity to stimulate synergy and alignment across the NFV ecosystem.

Overall, 189 interoperability test sessions were run, each of them on a different System Under Test featuring a different combination of the 41 participating implementations (FUTs): 19 VNFs, 10 MANO solutions and 12 NFV platforms. Over 48 organisations and 200 engineers were involved in the preparation of this one week event, forming an engaged and diverse community of NFV implementers.

The results of this second NFV Plugtests show high rates of test execution and interoperability for multi-vendor Network Services and very positive initial results in the other new areas of the test plan.

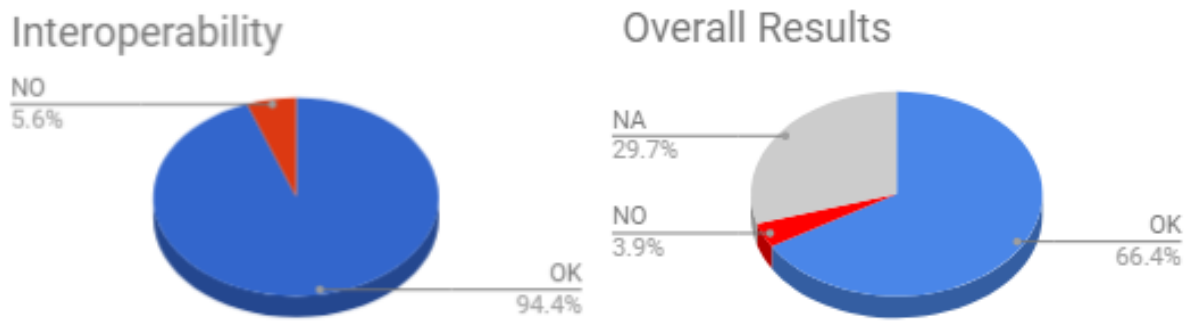

Figure 1. Summary results of 2nd NFV Plugtests

The figure above shows that interoperability was achieved in almost 95% of the tests that were executed. The overall results show an execution rate of over 65%, which reflects a significant increase of feature support across participating implementations compared to the 1st NFV Plugtests. (Note: results are reported as Not Applicable (NA) when one of the involved FUTs does not support the targeted feature and the test cannot be executed)

The above figures are even more remarkable if we take into account that the test plan included 50% more test cases addressing more advanced features and that, despite the Plugtests on-site duration being reduced by half, the total number of test sessions run was higher than during the 1st NFV Plugtests (189 vs 160). The detailed interoperability test results are found in clause 9.

The test plan, overall results and lessons learnt during NFV Plugtests are fed back to ETSI NFV Industry Specification Group.

1 Introduction

This Plugtests aimed at verifying interoperability among different implementations of the main components of the NFV Architectural Framework, which included:

- Virtual Network Functions (VNF), eventually providing additional Element Manager (EM) and specific VNF Manager (VNFM) functionalities
- Management and Orchestration (MANO) solutions, providing integrated NFV Orchestrator (NFVO) and VNFM functionalities
- NFV Platforms, including hardware, providing integrated NFV infrastructure (NFVI) and Virtual Infrastructure Manager (VIM) functionalities

Test and support VNFs were used to build the reference Network Services (NS) required to validate the proper behaviour of the Systems Under Test.

In order to enable remote integration and pre-testing among participants and allow them to get prepared for Plugtests, a dedicated VPN based network was built to interconnect local and remote implementations in a reliable and secure way: the NFV Plugtests HIVE: Hub for Interoperability and Validation at ETSI.

All of the participating implementations, Functions Under Test or Test/support Functions were connected and/or accessible through the HIVE network: most of the NFV platforms and MANO solutions run remotely on participants' labs, a subset of them were deployed locally at ETSI. VNF and NS Packages and images were made available in a local repository hosted at ETSI, and uploaded to the different NFV platforms during the pre-testing phase. Some test and support VNFs were deployed locally on local infrastructure at ETSI, and also accessible through HIVE.

All the information required to organise, coordinate and manage the 2nd NFV Plugtests was compiled and shared with participants in a dedicated private WIKI put in place by ETSI. Participants were provided with credentials that allowed them to access and update their details. Most of the information presented in this document has been extracted from the 2nd NFV Plugtests wiki: https://wiki.plugtests.net/NFV-PLUGTESTS (login required).

.

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.

[NFV002] ETSI GS NFV 002: "Network Functions Virtualisation (NFV); Architectural Framework".

[NFV003] ETSI GS NFV 003: "Network Functions Virtualisation (NFV); Terminology for main concepts in NFV".

[IFA005] ETSI GS NFV-IFA 005: "Network Functions Virtualisation (NFV); Management and Orchestration; Or-Vi reference point - Interface and Information Model Specification".

[IFA006] ETSI GS NFV-IFA 006: "Network Functions Virtualisation (NFV); Management and Orchestration; Vi-Vnfm reference point - Interface and Information Model Specification".

[IFA007] ETSI GS NFV-IFA 007: "Network Functions Virtualisation (NFV); Management and Orchestration; Or-Vnfm reference point - Interface and Information Model Specification".

[IFA008] ETSI GS NFV-IFA 008: "Network Functions Virtualisation (NFV); Management and Orchestration; Ve-Vnfm reference point - Interface and Information Model Specification".

[IFA010] ETSI GS NFV-IFA 010: "Network Functions Virtualisation (NFV); Management and Orchestration; Functional requirements specification".

[IFA013] ETSI GS NFV-IFA 013: "Network Functions Virtualisation (NFV); Management and Orchestration; Os-Ma-Nfvo reference point - Interface and Information Model Specification".

[TST002] ETSI GS NFV-TST 002: “Network Functions Virtualisation (NFV); Testing Methodology; Report on NFV Interoperability Testing Methodology”

[TST007] ETSI GR NFV-TST 007: “Network Functions Virtualisation (NFV); Testing; Guidelines on Interoperability Testing for MANO”

[SOL002] ETSI GS NFV-SOL 002: “Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Ve-Vnfm Reference Point”

[SOL003] ETSI GS NFV-SOL 002: “Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Or-Vnfm Reference Point”

[1NFVPLU-TP] 1st ETSI NFV Plugtests Test Plan: https://portal.etsi.org/Portals/0/TBpages/CTI/Docs/1st_ETSI_NFV_Plugtests_Test_Plan_v1.0.0.pdf

[1NFVPLU-R] 1st ETSI NFV Plugtests Report: https://portal.etsi.org/Portals/0/TBpages/CTI/Docs/1st_ETSI_NFV_Plugtests_Report_v1.0.0.pdf

[FORGE] ETSI Forge https://forge.etsi.org

[OSM-HACK] 1st OSM Hackfest Wiki https://osm.etsi.org/wikipub/index.php/1st_OSM_Hackfest

[2NFVPLU-TP] 2nd ETSI NFV Plugtests Test Plan: https://portal.etsi.org/Portals/0/TBpages/CTI/Docs/2nd_ETSI_NFV_Plugtests_Test_Plan_v1.0.0.pdf

3 Abbreviations

For the purposes of the present document, the terms and definitions given in [NFV003] and [TST002] apply.

4 Technical and Project Management

4.1 Scope

The main goal of the first NFV Plugtests was to run multi-vendor interoperability test sessions allowing to validate ETSI NFV Release 2 end-to-end capabilities including management of descriptors and software images, as well as life cycle management of network services, virtual network functions as well as virtual resources.

The main goal of this 2nd NFV Plugtests was to expand the scope to cover additional aspects and configurations, such as Multi VNF Network Services, Fault and Performance Management, Enhanced Platform Awareness and Multi-Site deployments.

During the interoperability Test Sessions, the Systems Under Test (SUTs) were made of different combinations of the following Functions Under Test (FUTs):

- One or several NFV Platforms, including hardware and providing pre-integrated VIM and NFVI functionality
- One MANO solution, providing pre-integrated NFVO and generic VNFM functionality
- Several VNFs eventually including EM and specific VNFM management functionality

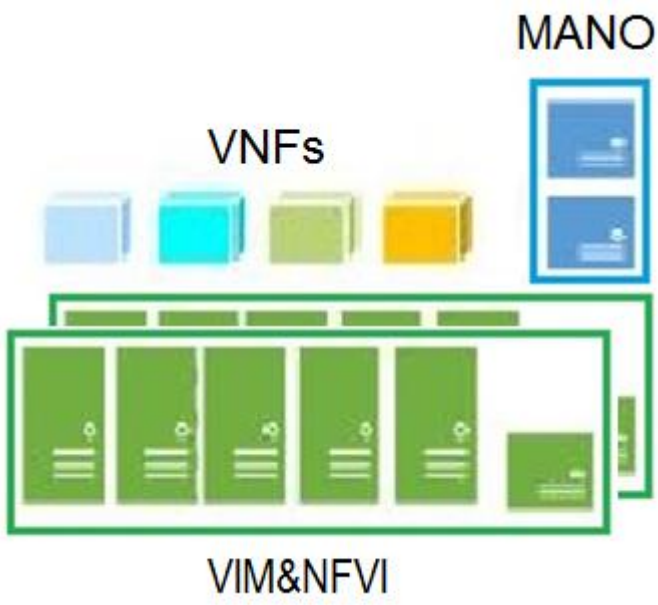

Figure 2. System Under Test

In addition to the interoperability test sessions, an experimental track on NFV API compliance testing was run by ETSI allowing participants to validate their API implementations against a subset of NFV SOL OpenAPIs.

4.2 Timeline

The 2nd NFV Plugtests preparation run through different phases as described in the figure below.

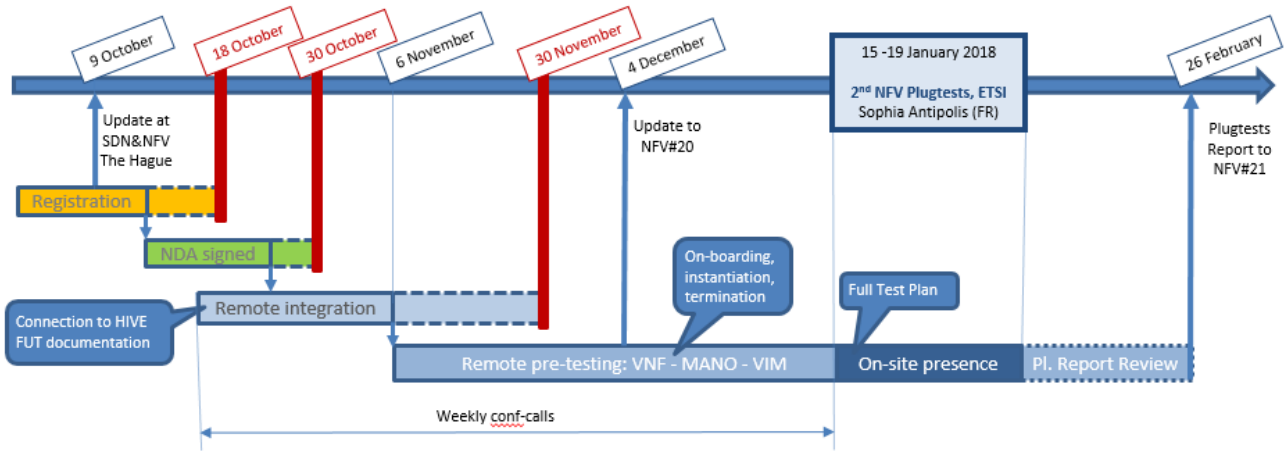

Figure 3. 2nd NFV Plugtests timeline

Registration to Plugtests was open from September to mid-October 2017 to any organisation willing to participate with a Function Under Test, Test or Support Function. Participating companies were requested to restrict on-site participation to a maximum of 2 people per implementation at a time. Additional remote participation (i.e. back office support) was possible and supported with electronic tools, see clause 4.3. A number of observers (network operators, research, and academia) and Open Source communities (ETSI OSM, Open Baton, Open Stack, OPNFV) participated to the test plan development and review. Overall, over 200 people were involved in the Plugtests either locally or remotely.

The following clauses describe the different phases of the Plugtests preparation. It is worth noting that since the start of the Remote Integration phase until the on-site week of face to face Plugtests, weekly conf-calls were run among organisers and participants to discuss and track the progress, anticipate and solve technical issues, review the test plan, and prepare for the face to face test sessions.

4.2.1 Remote Integration

During the Remote Integration phase, the following activities were run in parallel:

1) FUT Documentation

Participants documented their FUTs, by filling in a form compiling the Interoperability Features Statement (IFS) and Technical Questions (TQ) concerning their implementations. The final IFS Templates for each type of FUT can be found in the annex of the 2nd NFV Plugtests Test Plan [2NFVPLU-TP]

Participants providing VNFs complemented their documentation with diagrams and resource requirements. As for the 1st NFV Plugtests, some example VNFs and NSs were made available by the Plugtests team to facilitate the documentation of participating VNFs. These examples are documented in the Annex A of the 1st NFV Plugtests Report [1NFVPLU-R].

Participants providing MANO solutions developed and made available descriptor samples for the VNF and NS examples and supported the VNF providers in the creation of their own VNF and NS Descriptors.

Participants providing NFV Platforms created and documented tenants/projects and credentials for each participating MANO solution, and exposed and documented the North Bound Interfaces (NBI) of their VIM.

All the information described above was made available in the Plugtests WIKI, so that it could be easily maintained and consumed by participants.

2) Test Plan Development

The Test Plan development was led by ETSI Centre for Testing and Interoperability following the methodology and guidelines defined by ETSI NFV TST WG in TST002 and TST007, and building on the learnings and achievements of the 1st NFV Plugtests. The Test Plan scope was expanded to cover additional ETSI NFV Release 2 capabilities and features supported by the implementations attending the Plugtests. The supported features were compiled thanks to the IFS filled in by participants.

The Test Plan was developed and consolidated in an iterative way, taking into account input and feedback received from different stakeholders: ETSI NFV TST WG, supporting Open Source Communities and Plugtests participants. See details in Clause 8.

3) Connection to HIVE

Starting in November 2017, participants connected their implementations remotely to the NFV Plugtests Programme infrastructure, known as HIVE: Hub for Interoperability and Validation at ETSI.

During this phase, up to 38 remote labs connected to HIVE and each of them was allocated a dedicated network. The interconnection of remote labs allowed to run integration and pre-testing tasks remotely among any combination of participating FUTs, in order to ensure an efficient use of the face to face Plugtests time and a smoother run of Interoperability Test Sessions.

A site-to-site connection to HIVE was mandatory for participants providing NFV Platforms and MANO Solutions, and highly recommended for participants providing VNFs. The latest could also rely on client-to-site connection to HIVE, as long as they had no software (i.e. support function) running locally in their labs and only required access to remote labs for trouble shooting and infrastructure access purposes

Additional details on the remote test infrastructure are provided in Clause 6.

4) Once the above steps were completed, FUTs could start cross-FUT remote integration, see 7.1 for details on the procedures.

4.2.2 Pre-testing

Once remote integration was completed, participants had the opportunity to run remote pre-testing among different combinations of VNF, MANO and NFV Platforms.

The pre-testing test plan was a subset of the tests run during the 1st NFV Plugtests, in order to allow participants to concentrate on advanced features and configurations during the on-site phase.

Additional details on the pre-testing plan and procedures are provided in Clause 7.

4.2.3 On-site

From 15th to 19th of January, participants sent representatives to ETSI to collaboratively run Interoperability Test Sessions. The on-site Plugtests week (4,5 days) was organised as follows:

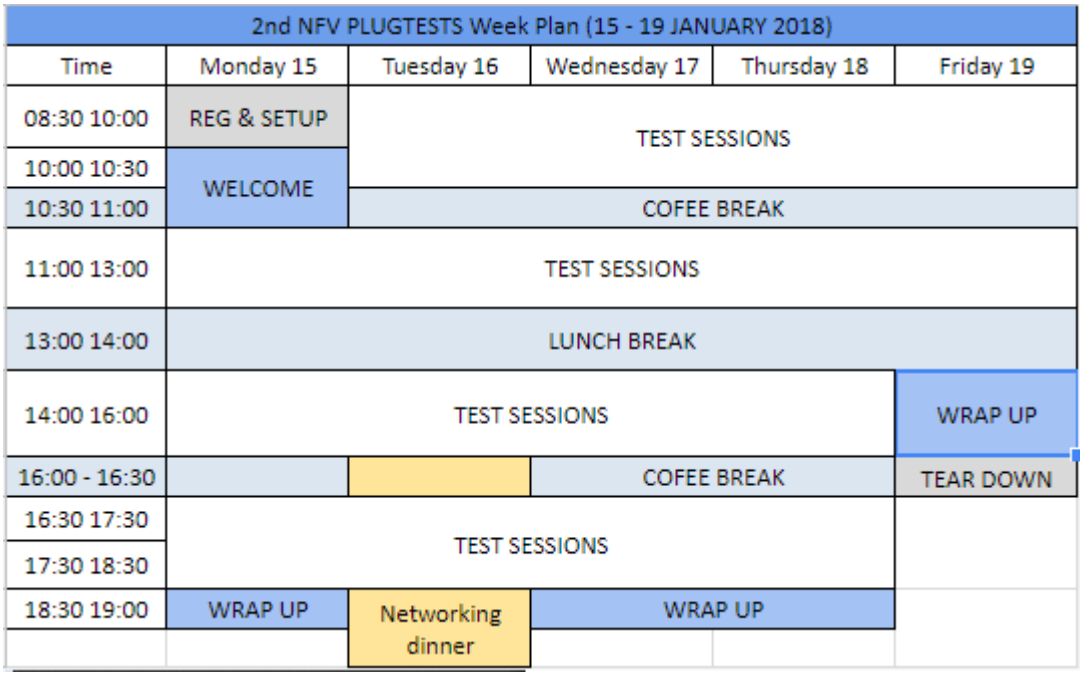

Figure 4. Plugtests on-site HL week plan

The interoperability test sessions involving all the participating FUTs were organised in several parallel tracks, see details in Clause 4.3.2.

4.3 Tools

4.3.1 Plugtests Wiki

The NFV Plugtests Wiki was the main entry point for all the information concerning this event, from logistics aspects to testing procedures. Access to this Wiki is restricted to companies participating to the NFV Plugtests Programme.

The main technical information provided in the wiki is organised as follows:

- **2nd Plugtests: Welcome** – Logistics and technical information to help participants upon their arrival on-site
- **2nd Plugtests: Wrap-up** - Agenda and minutes of the daily wrap up meetings run during the on-site phase.
- **OSM Hackfest** – Technical information on the co-located OSM Hackfest
- **HIVE** - VPN connection request tool, and remote connections status overview
- **FUTs** – Functions Under Test (FUT) overview, Interoperability Feature Statements and Technical Questions (IFS&TQ) forms, IFS&TQ responses overview, FUT dedicated pages.
- **Ref VNF&NS** - Diagram and Requirements documentation examples for sample VNFs / NS (end-point, middle point...)
- **Pre-testing** - Remote integration and pre-testing progress tracking matrix, remote integration and pre-testing procedures
- **Test Plan** – Test Plan information, including Test Suite Structure, SUT Configurations and Test Descriptions
- **Test Sessions** – Instruction for preparing and running interoperability test session
- **Test Reporting Tool** - instructions for reporting results with the Test Reporting Tool
- **API Track** – Technical information on the experimental API Compliance validation track
- **Conf-Calls** - Calendar, logistics, agendas and minutes of the weekly conf-calls run during the remote integration and pre-testing phase

In addition, an embedded IRC (chat) allowed participants to communicate during the pre-testing phase and Test Sessions, and include their remote colleagues (back-office support) in the discussions.

4.3.2 Test Session Scheduler

The Test Session Scheduler allowed the Plugtests organisers to produce a daily schedule during the on-site phase. This tool has the following objectives:

- maximise the number of test sessions
- balance the amount of test sessions among participants
- take into account supported features of the participating FUTs
- minimise the number of participants not involved in a test session anytime.

The picture below shows a partial view a daily schedule. Each yellow box corresponds to a specific Test Session addressing a Multi-VNF configuration including 2 VNFs, 1 MANO solution and 1 VIM&NFVI. For each of these sessions a Test Session Report was recorded (see next clause). In addition to the pre-scheduled test sessions, participants were invited to request, run and report results for additional test sessions targeting either pre-testing or advanced configurations addressing more complex combinations of FUTs and features.

An average of 35 Test Sessions were run every day during the Plugtests week by different combinations of participants.

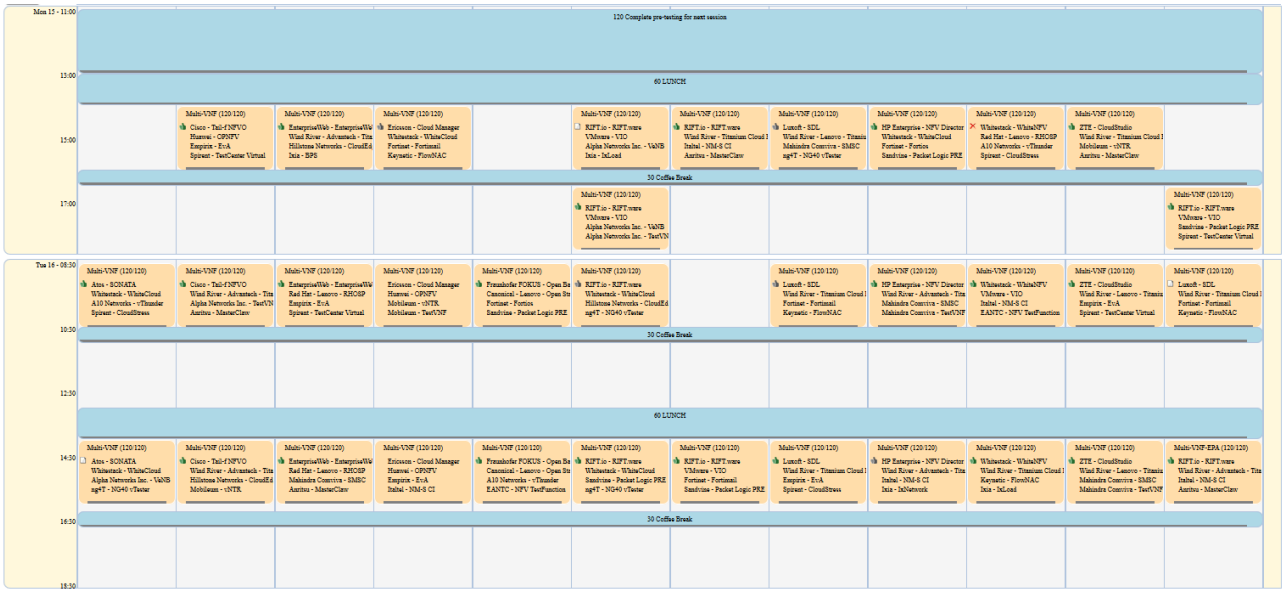

Figure 5. Daily Schedule example

4.3.3 Test Reporting Tool

The Test Reporting Tool guides participants through the Test Plan during the Test Sessions, and allows them to create Test Session Reports compiling detailed results for the Test Sessions. It allows reporting on pre-scheduled Test Sessions, but also on Test Sessions organised on the fly among participants to prepare, complete or complement the scheduled testing (freestyle sessions).

Only the companies providing the FUTs and Test Functions (when applicable) for each specific Test Session have access to the Test Session Reports (TSR) contents and specific results. All the companies providing the FUTs for a Test Session, i.e. VNF provider(s), MANO provider and NFV Platform provider(s) are required to verify and approve the reported results at the end of the session.

3016	Freestyle			PreTesting	RIFT.io - RIFT.ware VMware - VIO Keynetic - FlowNAC	
3017	2018-01-18 11:00	120	Test Track 9	Multi-VNF-EPA	Whitestack - WhiteNFV Wind River - Titanium Cloud R4 Keynetic - FlowNAC Ixia - IxNetwork	
3018	2018-01-18 11:00	120	Test Track 6b	Multi-VNF	RIFT.io - RIFT.ware VMware - VIO Keynetic - FlowNAC Ixia - IxNetwork	
3020	Freestyle			Multi-VNF	Whitestack - WhiteNFV Red Hat - Lenovo - RHOSP A10 Networks - vThunder Spirent - TestCenter Virtual	
3022	2018-01-18 11:00	120	Ad-hoc Testing	Multi-VNF-EPA	Luxoft - SDL Wind River - Titanium Cloud R4 Fortinet - Fortios Spirent - TestCenter Virtual	
3023	Freestyle			Multi-Site	Cisco - Tail-f NFVO Wind River - Advantech - Titanium Edge Wind River - Lenovo - Titanium Cloud R4 Empirix - EvA Spirent - TestCenter Virtual	
3024	Freestyle			PreTesting	HP Enterprise - NFV Director Huawei - OPNFV A10 Networks - vThunder	
3025	2018-01-18 14:00	120	Test Track 8	Multi-VNF	HP Enterprise - NFV Director Huawei - OPNFV Hillstone Networks - CloudEdge Alpha Networks Inc. - VeNB	
3026	Freestyle			PreTesting	Whitestack - WhiteNFV Wind River - Titanium Cloud R4 Empirix - EvA	
3027	2018-01-19 08:30	120	Test Track 9	Multi-VNF	Whitestack - WhiteNFV Red Hat - Lenovo - RHOSP Mobileum - vNTR	

Figure 6. Test Reporting Tool (extract of the TSR list)

Another interesting feature of this tool is the ability to generate real-time statistics (aggregated data) of the reported results, per test case, test group, test session or overall results. These stats are available in real time for all participants and organisers and allow tracking the progress of the testing with different levels of granularity, which is extremely useful to analyse the results.

5 Participation

5.1 Functions Under Test

The tables below summarise the different Functions Under Test provided by the Plugtests participants, and the location from where they were connecting to the HIVE network:

5.1.1 VNFs

Organisation	Solution(s)	Location	Short Description
A10 Networks	vThunder	USA/Germany	CGN, ADC, FW, Security
Alpha Networks	VeNB	Taiwan	Virtualised Small Cell, VeNB
Anritsu	Master Claw vProbe	Romania/Czech Republic	Virtual Probe for Customer Service Assurance
EANTC	NFV Test Function	Germany	Traffic Generator
Empirix	EvA	USA/Italy	Service Assurance
Fortinet	FortiGate	France	FW, Security
Fortinet	FortiMail	France	SMTP
Hillstone Networks	CloudEdge	China/USA	NGFW, Security
Italtel	NM-S CI	Italy	Session Border Controller
Ixia	IxLoad	USA/Romania	Application traffic simulator
Ixia	IxNetwork	USA/Romania	Traffic generator
Ixia	BPS	USA/Romania	Application traffic simulator
Keynetic	FlowNAC	Spain	Network Access Control
Mahindra Comviva	SMSC	India	SMSC
Mobileum	vNTR	India	Virtual NTR, Roaming
ng4T	NG40 vTester	Germany	Simulator, functional, capacity and load vTester
Sandvine	Packet Logic PRE	Sweden	DPI and Policy Enforcement
Spirent	CloudStress	California, USA	Workload generator
Spirent	TestCenter Virtual	California, USA	Traffic generator

Table 1. VNFs Under Test

5.1.2 MANOs

Organisation	Solution	Location	Type
Atos	SONATA	Spain	Open Source NFVO + Generic VNFM
Cisco	Tail-f NFVO	Sweden	NFVO and Elastic Services Controller
EnterpriseWeb	EnterpriseWeb	USA/Canada	Microservice-based NFVO and Generic VNFM
Ericsson	Cloud Manager	New Jersey, USA	NFVO and Generic VNFM
Fraunhofer FOKUS	Open Baton	Germany	Open Baton NFVO and Generic VNFM
HPE	NFV Director	France/Spain	HPE NFV Orchestration solution
Luxoft	SDL	Romania	NFVO and Generic VNFM
RIFT.io	RIFT.ware	USA	OSM based MANO Orchestration (NFVO and Generic VNFM)
Whitestack	WhiteNFV	USA	OSM Rel THREE distribution (NFVO and Generic VNFM)
ZTE	CloudStudio	China	NFVO and Generic VNFM

Table 2. MANOs Under Test

5.1.3 VIM&NFVIs

Organisations	VIM&NFVI	POD	Location	Type
Canonical QCT	OpenStack Cloud	QCT @ETSI	France	OpenStack Pike
Canonical Lenovo	OpenStack Cloud	OCP 'OP@L' @Lenovo	NC, USA	OpenStack Pike
Huawei	Compass OPNFV	OPNFV POD @Huawei	Singapore	OPNFV - OpenStack Ocata + Pike ODL Nitrogen
Red Hat Lenovo	Red Hat OpenStack Platform	OCP 'OP@L' @Lenovo	NC, USA	OpenStack Newton
VMware	vCloud Director	OSM Remote Lab @VMware	Canada	vCloud + NSX
VMware	VIO	OSM Remote Lab @VMware	Canada	VMware Integrated OpenStack Ocata
Whitestack	WhiteCloud	OSM Remote Lab @Whitestack	USA	OpenStack Pike ODL Carbon
Wind River	Titanium Cloud R4	OSM Remote Lab @Wind River	Hudson, USA	OpenStack Newton OF 1.3
Wind River Advantech	Titanium Cloud R4	Advantech @ETSI	France	OpenStack Newton OF 1.3
Wind River Advantech	Titanium Cloud R4	Remote Lab @Advantech	Taipei Taiwan	OpenStack Newton OF 1.3
Wind River Lenovo	Titanium Cloud R4	OCP 'OP@L' @Lenovo	NC, USA	OpenStack Newton OF 1.3
ZTE	TECS	TECS @ZTE	China	OpenStack Mitaka

Table 3. VIM&NFVIs Under Test

5.2 Test Functions

In addition to the Test VNFs included in the VNF section, the following Test Functions were available during the Plugtests to support the Test Sessions. Their use was optional.

Organisation	Solution	Type
Spirent	Life Cycle Validator	Test System for automated Life Cycle Management validation
Ubiwhere	NSD&VNFD Validator	NS & VNF Descriptor Validation (Sonata, OSM)

Table 4. Test Functions

5.3 Technical Support

The organisations below provided technical support and expertise to the Plugtests Team and contributed actively to the test plan development and technical arrangements to prepare and run the Plugtests.

Organisation	Role
Keynetic	Technical Support
Nextworks	Technical Support

Table 5. Technical Support

5.4 Observers

The following organisations joined the NFV Plugtests as observers and contributed with technical advice and test plan review:

Organisation	Role
British Telecom	Observer
Deutsche Telekom	Observer
Orange	Observer
Telefonica	Observer

Table 6. Observers

5.5 Open Source Communities

The Open Source communities listed below were actively involved in the Plugtests preparation and contributed to the Test Plan review. Their solutions were widely present in the Test Sessions through multiple distributions:

Organisation	Role	Details
Open Baton	MANO	https://openbaton.github.io
Open Source MANO	MANO	https://osm.etsi.org
Open Stack	VIM&NFVI	https://www.openstack.org
OPNFV	VIM&NFVI	https://www.opnfv.org

Table 7. Supporting Open Source communities

6 Test Infrastructure

The remote integration, pre-testing and on-site phases were enabled by the NFV Plugtests Programme's HIVE network

Figure 7. NFV Plugtests HIVE network

The NFV HIVE network interconnects securely participants' remote labs and Functions under Test and allows for remote multi-party interoperability testing and validation activities. A total of 38 remote locations including several OSM Remote Labs participating to the Plugtests leveraged the HIVE network to make their Functions Under Test available for the test sessions.

The figure below describes how the NFV and OSM HIVE networks where interconnected to support in parallel the Plugtests / Hackfest activities.

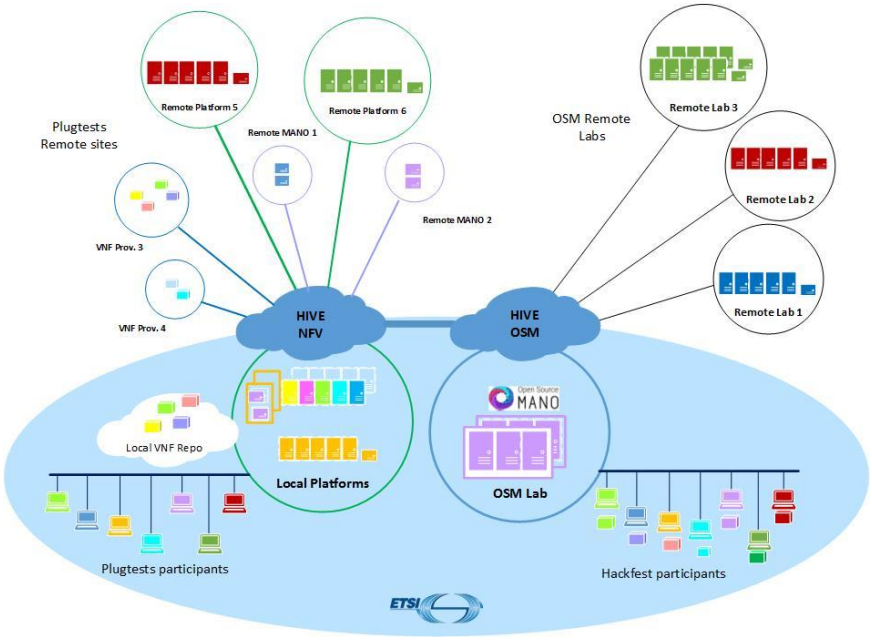

Figure 8. Remote Test Infrastructure

During the on-site phase, a local network was deployed in the Plugtests room to allow participants to access the remote labs, the local VNF Repository and some support functions deployed locally on ETSI servers to support the testing. Two participating NFV Platforms (VIM&NFVI) were also deployed locally in the ETSI Lab and connected to HIVE.

A second local network was deployed in the OSM Hackfest room, which was seen as an additional OSM Remote Lab in the OSM HIVE Network. With this setup, participants in the OSM Hackfest could choose between installing OSM in their own laptops, or in one of the OSM servers in the OSM Demo Lab. With both options they were able to interact and deploy VNFs in OSM Remote VIMs provided by the community for the Hackfest.

7 IOP Test Procedures

7.1 Remote Integration Procedures

Once the FUT documentation and HIVE connection had been successfully completed (see 4.2.1), a number of remote integration activities were run depending on the FUT type and integration target as described in the procedures hereafter:

1) MANO and VIM&NFVI:

a. Test connectivity from MANO to VIM&NFVI and from VIM&NFVI to MANO

b. Connect MANO to VIM (get NBI IP, tenant and credentials on VIM&NFVI wiki page)

c. Verify VIM resources can be accessed from MANO

d. Specify the Reference VNF in the pre-testing table

e. Upload Reference VNF to VIM

f. On-board, instantiate and terminate Reference VNF from MANO

2) VNF and MANO

a. Identify the Reference VIM&NFVI(s) that will be used for pre-testing

b. Create VNFD for MANO, and upload to VNF Repository

c. Create NSDs for the targeted SUT configurations and upload to VNF Repository

d. On-board VNFD/NSD to MANO

e. Upload VNF image(s) to Reference VIM

f. Instantiate VNF/NS from MANO

g. Scale in/out

h. Terminate VNF/NS

3) VNF and VIM&NFVI

a. Upload VNF image to VIM

b. Verify physical network connectivity in NFVI will allow for VNF/NS deployment

c. Manual creation of VM's and network infrastructure required for this VNF (to prepare MANO On-board automatic execution)

d. Manual execution Instantiate VNF steps (in preparation of MANO automatic execution)

e. Manual execution Scale in/out VNF steps (in preparation of MANO automatic execution)

f. Manual execution Terminate VNF steps (in preparation of MANO automatic execution)

The progress of documentation and remote integration procedures for the different combinations of FUTs (VNFs, MANOs, VIM&NFVIs) was captured in a multi-dimensional tracking matrix, and reviewed during the weekly calls. The tracking matrix was similar to the one shown below.

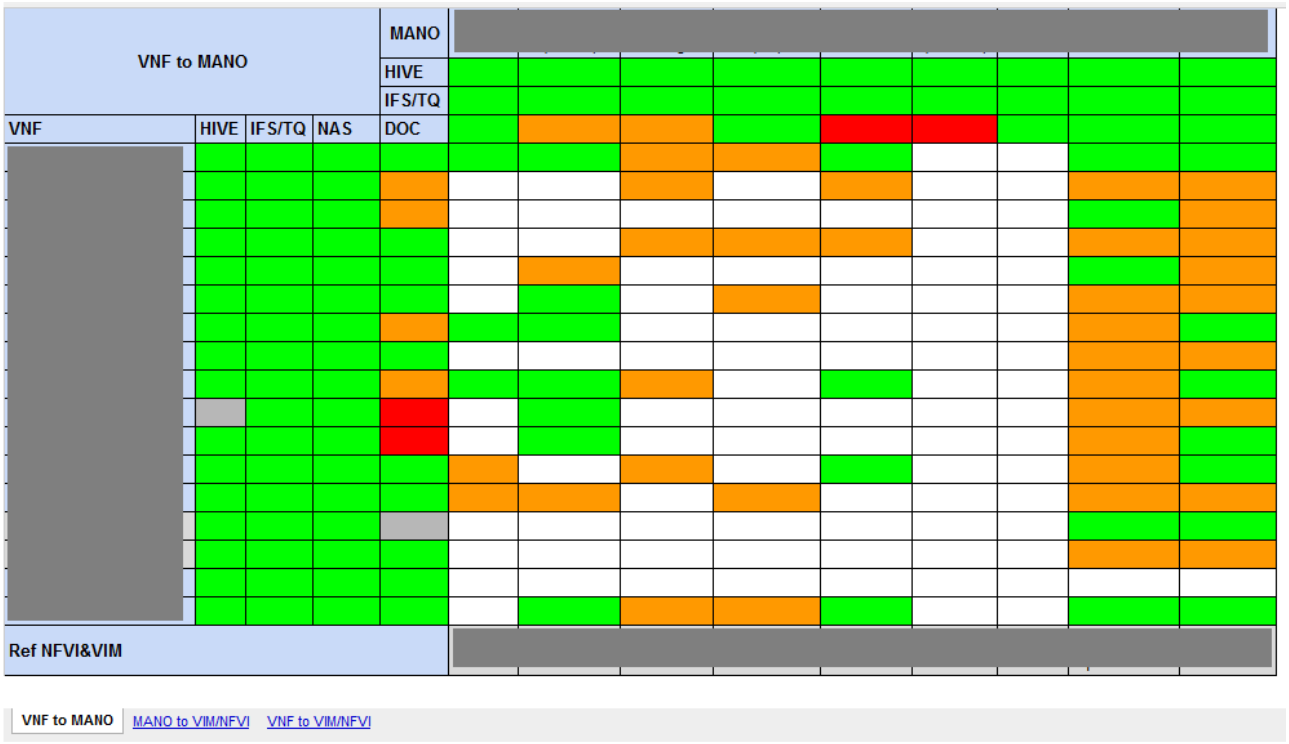

Figure 9. Pre-testing tracking matrix

The progress on each of the dimensions of the remote integration and pre-testing activities (VNF to MANO, MANO to VIM&NFVI and VNF to VIM&NFVI) was tracked following the colour code described below:

Pre-testing Matrix Colour Code	
	Not Started / Not Reported
	Not Applicable
	Pending
	Ongoing
	Completed

Table 8. Pre-testing matrix colour code

7.2 Pre-testing Procedure

Following remote integration, participants were invited to run pre-testing sessions and formally capture the results in the Test Reporting Tool. The pre-testing configuration involved 1 VNF, 1 MANO solution and 1 VIM&NFVI. The test cases included in this group were a subset of the tests run during the 1st NFV Plugtests and addressed basic capabilities such as on-boarding, instantiation, stop/start VNF, manual scaling and termination, see details in 8.2.

Running and reporting results for a pre-testing session required all the 3 parties (VNF, MANO, VIM&NFVI) to have successfully achieved remote integration (connection to HIVE, descriptors created and uploaded, images uploaded, credentials shared, VIM ready, etc...).

Participants were encouraged to find suitable testing partners in the pre-testing matrix (those that had completed remote integration) and arrange a convenient timeframe for pre-testing, leveraging the WIKI chat to facilitate discussions among the different teams.

Once the arrangements made the pre-testing procedure was as follows:

1) Open the Test Reporting Tool, go to the “Reports” TAB

2) Create a “Freestyle” report, select configuration = Pre-testing

3) Select the MANO, VIM&NFVI and VNF involved in the pre-testing session

Once the above completed, a Test Session Report would open, and participants would follow the Interoperability Testing Procedure remotely (see next clause)

7.3 Interoperability Testing Procedure

During the on-site part of the Plugtests, a daily Test Session Schedule was produced with the Plugtests Scheduler. Test Sessions were organised in several parallel tracks, ensuring that all participants had at least one Test Session scheduled any time.

Every day, on each testing track 2 Test Sessions where pre-scheduled for each MANO solution: one in the morning and one in the afternoon. Each test session involved a different combination of MANO, VIM&NFVI and 2 VNFs and targeted to run a Multi-VNF Test Group (See 8.3 for details). Each Test Session was expected to be completed in 2 hours. Participants could choose to allocate the remaining time to:

- complete pending test sessions,
- run additional test groups (EPA, Multi-Site, ...), or
- start pre-testing for the next sessions

Participants could choose to run the above mentioned additional testing as “freestyle” test sessions, and create a test report on the fly, or to ask the Plugtests team to schedule the additional sessions for them,

During each test session the Interoperability testing procedure was as follows:

1) The MANO, VIM&NFVI, and 2 VNFs representative would sit together. Each VNF was expected to have previously run a pre-testing test session with the MANO and VIM

2) One representative of the team opened the Test Session Report and the Test Plan.

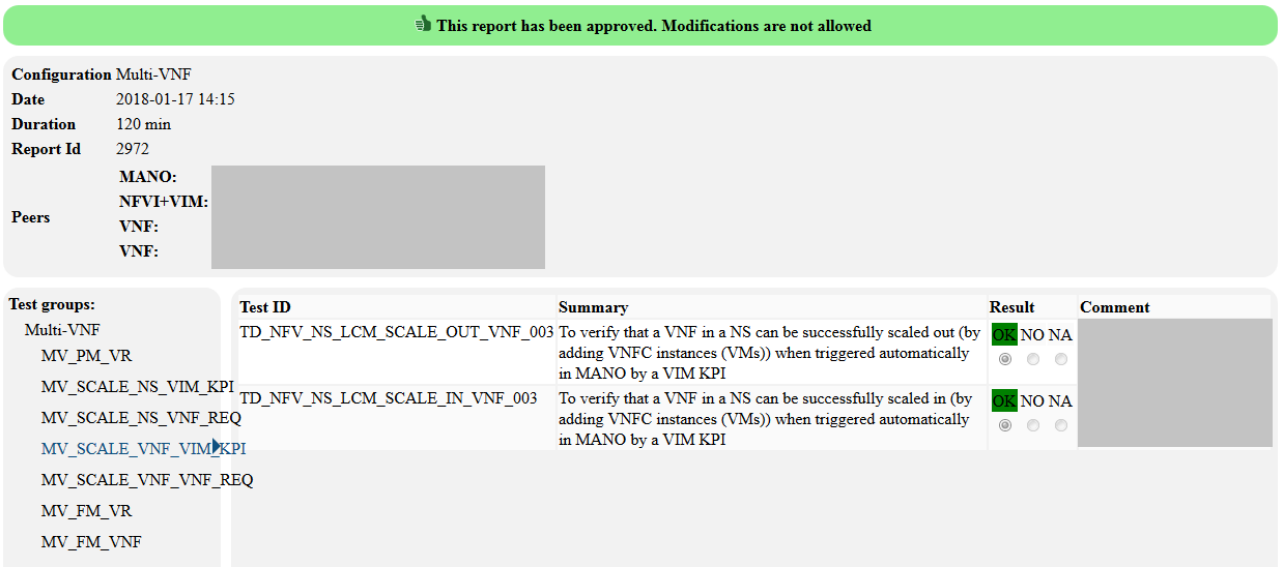

Figure 10. Test Session Report

3) For each Test in each group of the Test Plan:

a. The corresponding Test Description and SUT Configuration were followed.

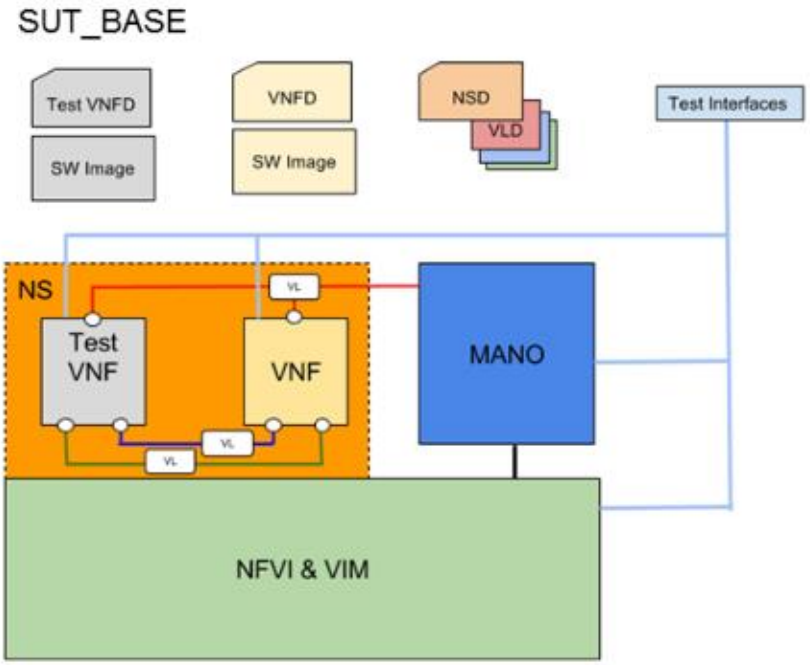

Figure 11. SUT Configuration example.

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_003			
Test Purpose	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered automatically in MANO by a VIM KPI			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2, 7.7) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2, 7.7) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4)			
Applicability	* [IFS_NFV_MANO_19] MANO supports receiving VM/VNFC KPIs from VIM * [IFS_NFV_MANO_20] MANO supports automatic scaling out/in triggered by KPIs from VIM * [IFS_NFV_MANO_15] MANO supports scaling by adding/removing VNFC instances * [IFS_NFV_VNF_5] VNF can scale out/in by adding/removing VNFC instances * [IFS_NFV_VIM_NFVI_3] NFVI/VIM exposes VM/VNFC virtual compute resource KPIs to MANO/VNFM * [IFS_NFV_VIM_NFVI_4] NFVI/VIM exposes VM/VNFC virtual network resource KPIs to MANO/VNFM * [IFS_NFV_VIM_NFVI_5] NFVI/VIM exposes VM/VNFC virtual storage resource KPIs to MANO/VNFM			
Pre-test conditions	* NS is instantiated (TD_NFV_BASE_NS_LCM_INSTANTIATE_001) * MANO is configured to trigger SCALE OUT (by adding VM(s)) when a given VIM KPI value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger NS scale out (by adding VMs to a VNF inside the NS) in MANO with a VIM KPI	
	2	IOP Check	Verify that the scale out (by adding VNFC instances (VMs)) procedure has been started in MANO	
	3	IOP Check	Verify that the requested resources have been allocated by the VIM according to the descriptors	
	4	IOP Check	Verify that the additional VM(s) have been deployed (i.e by querying the VIM)	
	5	IOP Check	Verify that the additional VM(s) are running and are reachable through the management network	
	6	IOP Check	Verify that the additional VM(s) are connected to the VL(s) according to the descriptors	
	6	IOP Check	Verify that NS has been scaled out by running the end-to-end functional test	
IOP Verdict				

Figure 12 Test Description example

b. VNFs, MANO and VIM&NFVI providers jointly executed the different steps specified in the Test Description and evaluated interoperability through the different IOP Checks prescribed in it.

c. The Test Result was reported to the Test Session Report, as follows:

   i. OK: all IOP Checks were successful

   ii. NOK: at least one IOP Check failed. A comment was requested.

   iii. NA: the feature was not supported by at least 1 of the involved FUTs. A comment was requested to clarify the missing feature.

4) Once all the tests in the Test Session Report were executed and results recorded, the VNFs, MANO and VIM&NFVI providers reviewed the Report and approved it.

8 IOP Test Plan Overview

8.1 Introduction

This 2nd NFV Plugtests Test Plan was developed by the NFV Plugtests team lead by ETSI Centre for Testing and Interoperability and following the interoperability testing methodology defined by ETSI NFV in [TST002].

The Test Plan was reviewed and discussed with participants, ETSI NFV TST working group and supporting Open Source communities during the Plugtests preparation and pre-testing phases.

The following clauses summarise the 47 test cases in scope for this Plugtests, and how they were grouped to optimise test session scheduling and duration. The following clauses provide an overview of the six test groups identified for the 2nd NFV Plugtests: Pre-Testing, Multi-VNF, Multi-VNF-EPA, Multi-Site, Specific VNFM, Automatic LCM validation.

8.2 Pre-Testing

8.2.1 Test Configuration

The Pre-testing group leverages the SUT_BASE configuration as described in the Test Plan [2NFVPLU-TP].

It involves one MANO solution, one VIM&NFVI and one VNF. The Test VNF used to verify proper NS deployment and functionality was expected to be provided by the VNF provider. Participants were encouraged to arrange and run this test group remotely as soon as remote integration was achieved to prepare for more complex sessions on-site. This group was mandatory for Plugtests participants.

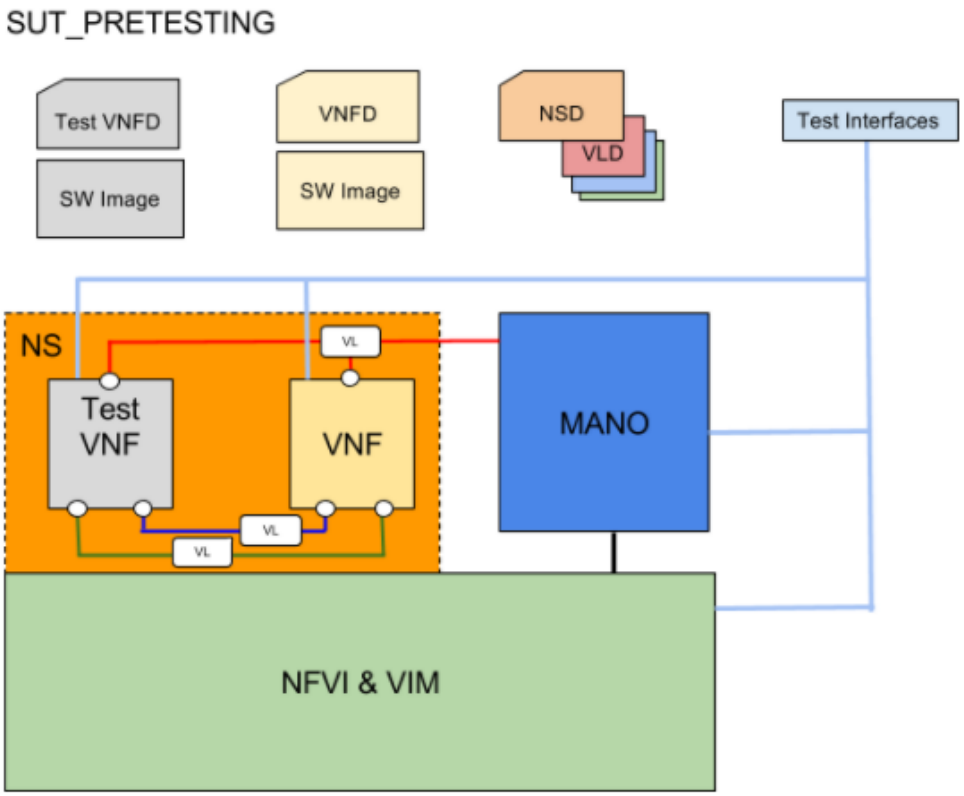

Figure 13. Pre-testing Configuration: SUT_BASE

8.2.2 Test Cases

Test Id	Test Purpose
TD_NFV_BASE_ONBOARD_VNF_PKG_001	To on-board a VNF Package
TD_NFV_BASE_ONBOARD_NSD_001	To on-board a NSD
TD_NFV_BASE_NS_LCM_INSTANTIATE_001	To verify that an NS can be successfully instantiated
TD_NFV_BASE_NS_LCM_SCALE_OUT_001	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_IN_001	To verify that a NS can be successfully scaled in (by removing VNF instances) if triggered by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_001	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_001	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered by a MANO operator
TD_NFV_BASE_NS_LCM_UPDATE_STOP_VNF_001	To verify that a VNF running in a NS can be successfully stopped by MANO
TD_NFV_BASE_NS_LCM_UPDATE_START_VNF_001	To verify that a stopped VNF in a NS can be successfully re-started by MANO
TD_NFV_BASE_NS_LCM_TERMINATE_001	To verify that a NS can be successfully terminated
TD_NFV_BASE_TEARDOWN_DELETE_NSD_001	To delete a NSD
TD_NFV_BASE_TEARDOWN_DELETE_VNF_PKG_001	To delete a VNF Package

Table 9. Test Group Pre-Testing

8.3 Multi-VNF

8.2.1 Test Configuration

The Multi-VNF group leverages the SUT_BASE configuration as described in the Test Plan [2NFVPLU-TP]

It involves one MANO solution, one VIM&NFV and at least two VNF from 2 different providers. This group was mandatory for Plugtests participants and aimed at testing multi-vendor Network Services.

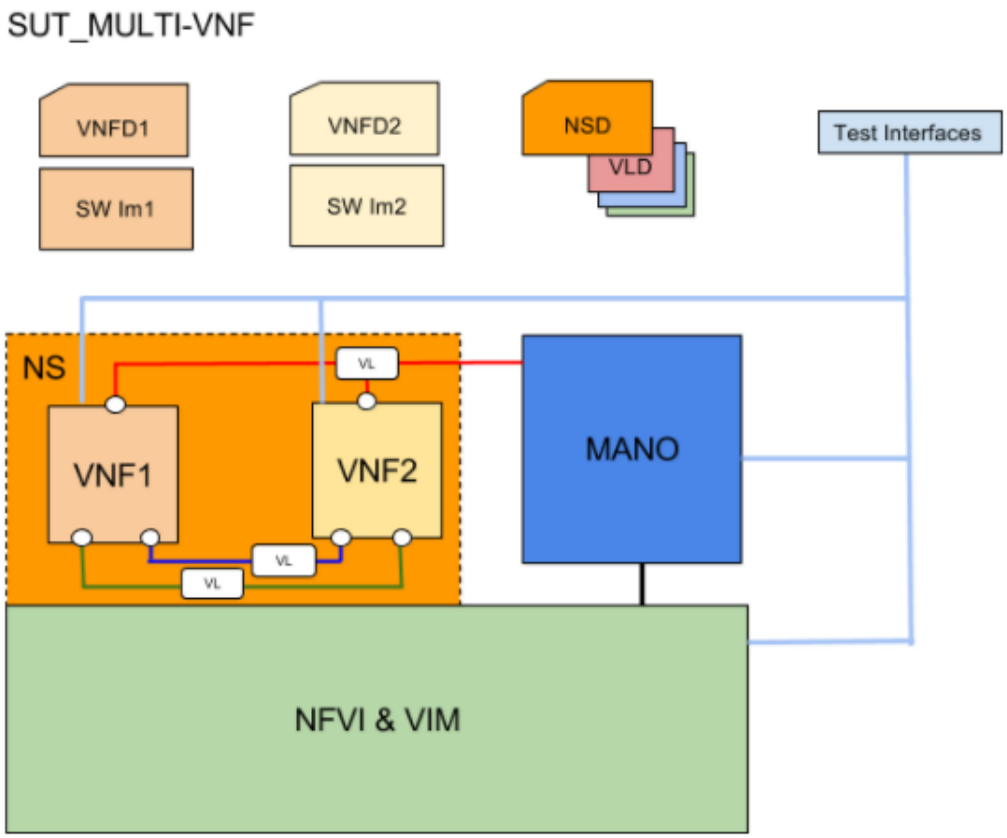

Figure 14. Multi-VNF Configuration: SUT_BASE

8.2.2 Test Cases

8.2.2.1 Basic Testing: MV_BASIC

Test Id	Test Purpose
TD_NFV_BASE_ONBOARD_NSD_001	To on-board a NSD
TD_NFV_BASE_NS_LCM_INSTANTIATE_001	To verify that an NS can be successfully instantiated
TD_NFV_BASE_NS_LCM_TERMINATE_001	To verify that a NS can be successfully terminated
TD_NFV_BASE_TEARDOWN_DELETE_NSD_001	To delete a NSD

Table 10. Test Group MV_BASIC

8.2.2.2 Performance Management - VR: MV_PM_VR

Test Id	Test Purpose
TD_NFV_BASE_PM_VR_CREATE_NOTIFY_001	To verify that the performance metrics of a virtualised resource that is required for a NS instance can be monitored using performance monitoring jobs and notifications
TD_NFV_BASE_PM_VR_CREATE_THRESHOLD_001	To verify that the performance metrics of a virtualised resource that is required for a NS instance can be monitored using performance monitoring jobs and thresholds
TD_NFV_BASE_PM_VR_DELETE_NOTIFY_001	To verify that the performance metrics of a virtualised resource that is required for a NS instance can be monitored using performance monitoring jobs and notifications
TD_NFV_BASE_PM_VR_DELETE_THRESHOLD_001	To verify that the performance metrics of a virtualised resource that is required for a NS instance can be monitored using performance monitoring jobs and thresholds

Table 11. Test Group MV_PM_VR

8.2.2.3 Performance Management - VNF: MV_PM_VNF

Test Id	Test Purpose
TD_NFV_BASE_PM_VNF_KPI_CREATE_NOTIFY_001	To verify that a VNF indicator inside a NS instance can be monitored using subscriptions and notifications
TD_NFV_BASE_PM_VNF_KPI_DELETE_NOTIFY_001	To verify that the monitoring of a VNF indicator inside a NS instance can be stopped by deleting subscriptions

Table 12. Test Group MV_PM_VNF

8.2.2.4 Scale NS from VNF Indicator: MV_SCALE_NS_VNF_IND

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_002	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered automatically in MANO by a VNF Indicator
TD_NFV_BASE_NS_LCM_SCALE_IN_002	To verify that a NS can be successfully scaled in (by removing VNF instances) if triggered automatically in MANO by a VNF Indicator

Table 13. Test Group MV_SCALE_NS_VNF_IND

8.2.2.5 Scale NS from VIM KPI: MV_SCALE_NS_VIM_KPI

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_003	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered automatically in MANO by a VIM KPI
TD_NFV_BASE_NS_LCM_SCALE_IN_003	To verify that a NS can be successfully scaled in (by removing VNF instances) if triggered automatically in MANO by a VIM KPI

Table 14. Test Group MV_SCALE_NS_VIM_KPI

8.2.2.6 Scale NS from VNF Request: MV_SCALE_NS_VNF_REQ

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_004	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered in MANO by a VNF/EM request
TD_NFV_BASE_NS_LCM_SCALE_IN_004	To verify that a NS can successfully scale in (by removing VNF instances) if triggered in MANO by a VNF/EM request

Table 15. Test Group MV_SCALE_NS_VNF_REQ

8.2.2.7 Scale VNF from VNF Indicator: MV_SCALE_VNF_VNF_IND

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_002	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered automatically in MANO by a VNF Indicator
TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_002	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered automatically in MANO by a VNF Indicator

Table 16. Test Group MV_SCALE_VNF_VNF_IND

8.2.2.8 Scale VNF from VIM KPI: MV_SCALE_VNF_VIM_KPI

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_003	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered automatically in MANO by a VIM KPI
TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_003	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered automatically in MANO by a VIM KPI

Table 17. Test Group MV_SCALE_VNF_VIM_KPI

8.2.2.9 Scale VNF from VNF Request: MV_SCALE_VNF_VNF_REQ

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_004	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered in MANO by a VNF/EM request
TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_004	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered in MANO by a VNF/EM request

Table 18. Test Group MV_SCALE_VNF_VNF_REQ

8.2.2.10 Fault Management - VR: MV_FM_VR

Test Id	Test Purpose
TD_NFV_BASE_FM_VR_NOTIFY_001	To verify that a fault alarm notification propagates when a virtualised resource that is required for the NS network connectivity fails.
TD_NFV_BASE_FM_VR_CLEAR_001	To verify that a fault clearance notification propagates when a failed virtualised resource that is required for the NS network connectivity is recovered

Table 19. Test Group MV_FM_VR

8.2.2.11 Fault Management - VNF: MV_FM_VNF

Test Id	Test Purpose
TD_NFV_BASE_FM_VNF_NOTIFY_001	To verify that a VNF fault alarm notification propagates via the VNFM when a VNF fault is triggered by a failed virtualised resource
TD_NFV_BASE_FM_VNF_CLEAR_001	Verify that a VNF fault alarm clearance notification propagates via the VNFM to the MANO when a VNF fault is cleared by resolving a failed virtualised resource

Table 20. Test Group MV_FM_VNF

8.4 Multi-VNF-EPA

8.4.1 Test Configuration

The Multi-VNF-EPA group leverages the SUT_BASE configuration as described in the Test Plan [2NFVPLU-TP].

As the Multi-VNF configuration it involves one MANO solution, one VIM&NFV and at least two VNFs from different providers. The MANO the VIM&NFVI and at least At least one of the VNFs, are required to support EPA in order to run this Test Group, which was optional for Plugtests participants.

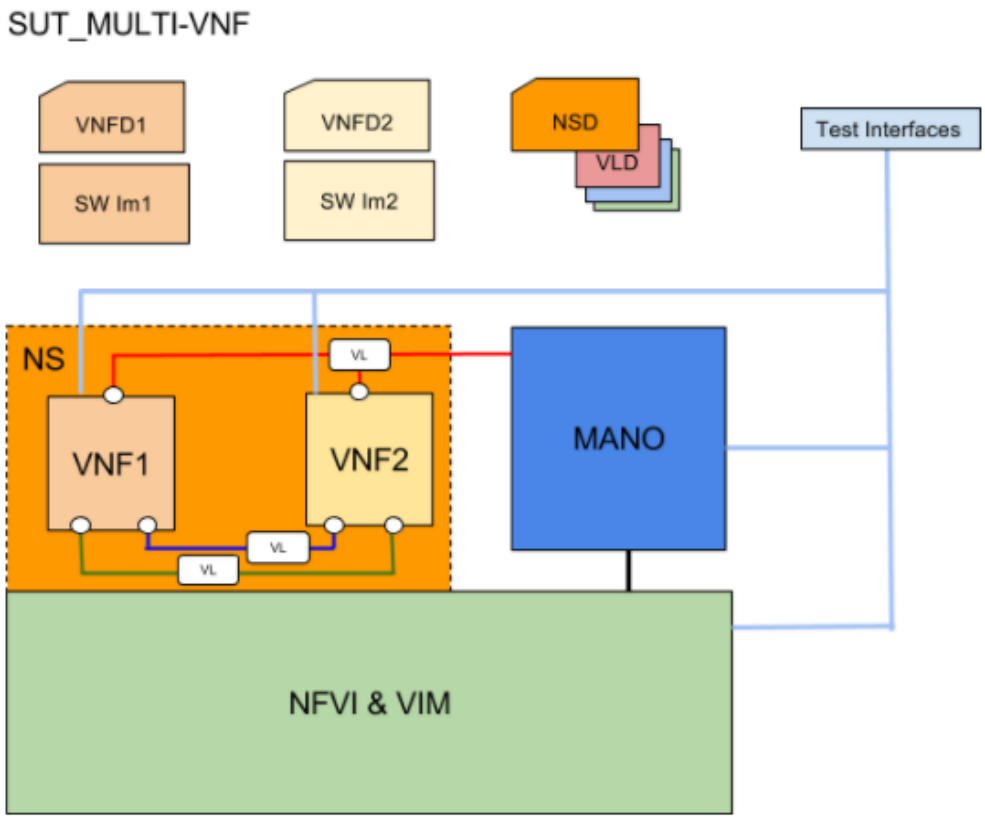

Figure 15. Multi-VNF-EPA Configuration: SUT_BASE

8.4.2 Test Cases

8.4.2.1 Basic Testing: MVE_BASIC

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_INSTANTIATE_EPA_001	To verify that an NS can be successfully instantiated with EPA requirements

Table 21. Test Group MVE_BASIC

8.4.2.2 Scale NS Manually: MVE_SCALE_NS_MANUAL

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_EPA_001	To verify that a NS can be successfully scaled out with EPA requirements (by adding VNF instances) if triggered automatically by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_IN_EPA_001	To verify that a NS can be successfully scaled in with EPA requirements (by removing VNF instances) if triggered automatically by a MANO operator

Table 22. Test Group MVE_SCALE_NS_MANUAL

8.4.2.3 Scale VNF Manually: MVE_SCALE_VNF_MANUAL

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_EPA_001	To verify that a VNF in a NS can be successfully scaled out with EPA requirements (by adding VNFC instances (VMs)) when triggered by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_EPA_001	To verify that a VNF in a NS can be successfully scaled in with EPA requirements (by removing VNFC instances (VMs)) when triggered by a MANO operator

Table 23. Test Group MVE_SCALE_VNF_MANUAL

8.5 Multi-Site

8.5.1 Test Configuration

The Multi-Site group leverages the SUT_MULI-SITE configuration as described in the Test Plan [2NFVPLU-TP].

It involves one MANO solution, two different VIM&NFV and at least two VNFs. This group was optional for Plugtests participants.

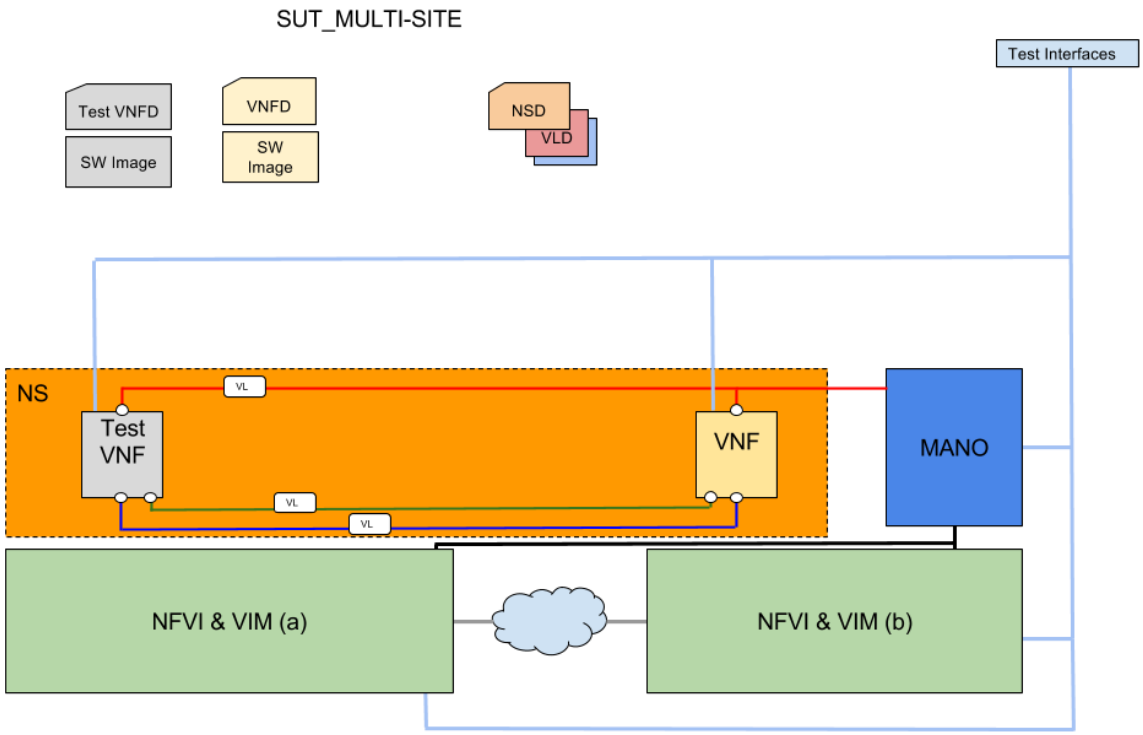

Figure 16. Multi-SITE SUT Configuration

8.5.2 Test Cases

8.5.2.1 Basic Testing: MS_BASIC

Test Id	Test Purpose
TD_NFV_MULTISITE_NS_LCM_INSTANTIATE_001	To verify that an NS can be successfully instantiated across different sites
TD_NFV_BASE_NS_LCM_TERMINATE_001	To verify that a NS can be successfully terminated

Table 24. Test Group MS_BASIC

8.5.2.2 Scale NS Manually: MS_SCALE_NS_MANUAL

Test Id	Test Purpose
TD_NFV_MULTISITE_NS_LCM _SCALE_OUT_001	To verify that a multi-site NS can be successfully scaled out (by adding VNF instances) if triggered by a MANO operator
TD_NFV_MULTISITE_NS_LCM_SCALE_IN_001	To verify that a multi-site NS can be successfully scaled in (by removing VNF instances) if triggered by a MANO operator

Table 25. Test Group MS_SCALE_NS_MANUAL

8.5.2.3 Scale VNF Manually: MS_SCALE_VNF_MANUAL

Test Id	Test Purpose
TD_NFV_MULTISITE_NS_LCM_SCALE_OUT_VNF_001	To verify that a VNF in a multi-site NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered by a MANO operator
TD_NFV_ MULTISITE_NS_LCM_SCALE_IN_VNF_001	To verify that a VNF in a multi-site NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered by a MANO operator

Table 26. Test Group MS_SCALE_VNF_MANUAL

8.5.2.4 Multi-Site: 3 Sites

This subgroup extended the SUT Configuration with a third VIM&NFVI.

Test Id	Test Purpose
TD_NFV_MULTISITE_NS_LCM_INSTANTIATE_001	To verify that an NS can be successfully instantiated across different sites
TD_NFV_BASE_NS_LCM_TERMINATE_001	To verify that a NS can be successfully terminated

Table 27. Test Group Multi-Site-3

8.6 Specific VNFM

8.6.1 Test Configurations

The Specific VNFM groups leverage the SUT_S-VNFM-D and SUT_S-VNFM-I SUT configurations as described in the Test Plan [2NFVPLU-TP] to address the Direct and Indirect Modes respectively.

These configurations involve one MANO solution, one VIM&NFV and two VNFs with at least one of them providing its own VNF Manager. The Specific VNFM and the MANO solutions are requested to support the same resource management mode depending on the targeted configuration

- S-VNFM-D, Direct Mode, VNF related resources managed by S-VNFM
- S-VNFM-I, Indirect Mode, VNF related resources managed by MANO

These groups were optional for Plugtests participants.

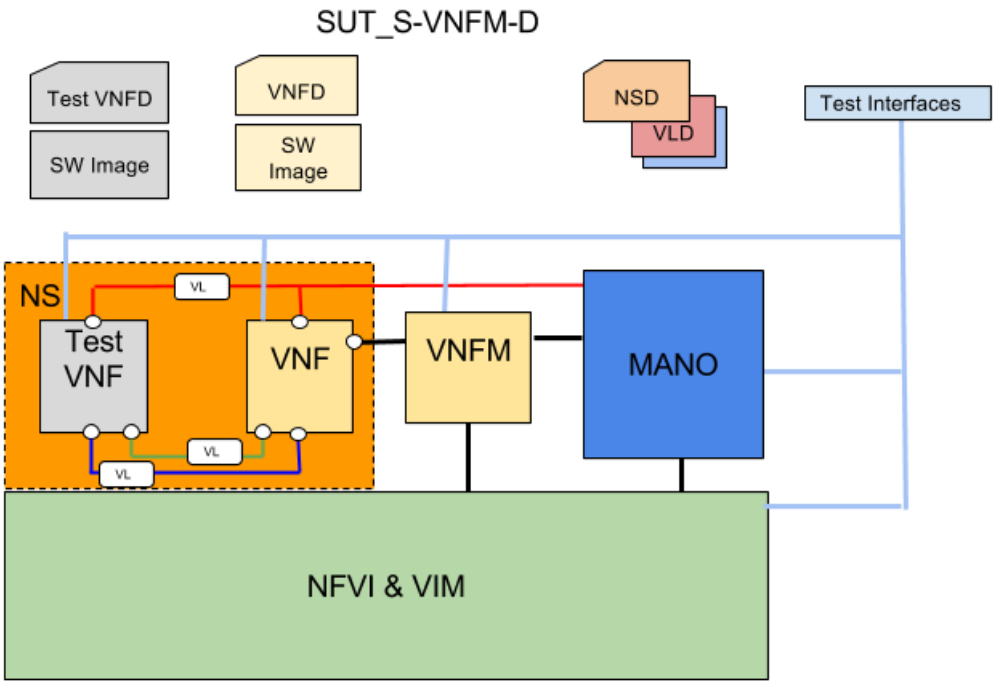

Figure 17. Specific VNFM (Direct Mode) Configuration: S-VNFM-D

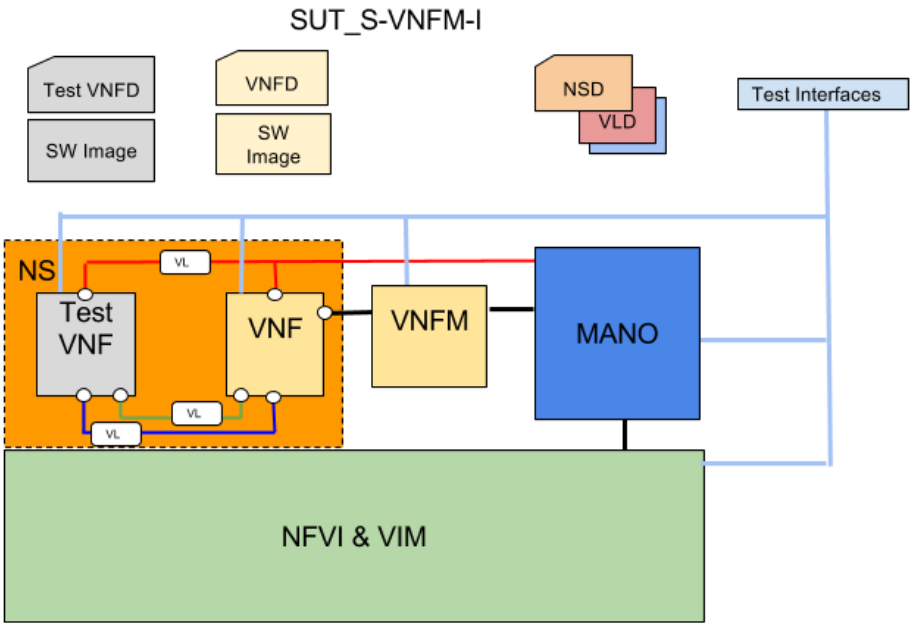

Figure 18. Specific VNFM (Indirect Mode) Configuration: S-VNFM-I

8.6.2 Test Cases

Test Id	Test Purpose
TD_NFV_BASE_ONBOARD_VNF_PKG_001	To on-board a VNF Package
TD_NFV_BASE_NS_LCM_INSTANTIATE_001	To verify that an NS can be successfully instantiated
TD_NFV_BASE_PM_VNF_VR_CREATE_NOTIFY_001	To verify that the performance metrics of a virtualised resource that is allocated to a VNF instance inside a NS instance can be monitored using VNFM performance monitoring jobs and notifications
TD_NFV_BASE_PM_VNF_VR_CREATE_THRESHOLD_001	To verify that the performance metrics of a virtualised resource that is allocated to a VNF instance inside a NS instance can be monitored using VNFM performance monitoring jobs and thresholds
TD_NFV_BASE_PM_VNF_VR_DELETE_NOTIFY_001	To verify that the monitoring of performance metrics of a virtualised resource that is allocated to a VNF instance inside a NS instance can be stopped by deleting performance monitoring jobs on the VNFM
TD_NFV_BASE_PM_VNF_VR_DELETE_THRESHOLD_001	To verify that a performance monitoring threshold created for a virtualised resource that is allocated to a VNF instance inside a NS instance can be deleted on the VNFM
TD_NFV_BASE_PM_VNF_KPI_CREATE_NOTIFY_001	To verify that a VNF indicator inside a NS instance can be monitored using subscriptions and notifications
TD_NFV_BASE_PM_VNF_KPI_DELETE_NOTIFY_001	To verify that the monitoring of a VNF indicator inside a NS instance can be stopped by deleting subscriptions
TD_NFV_BASE_NS_LCM_SCALE_OUT_001	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_IN_001	To verify that a NS can be successfully scaled in (by removing VNF instances) if triggered by a MANO operator

TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_001	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_001	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered by a MANO operator
TD_NFV_BASE_FM_VR_NOTIFY_001	To verify that a fault alarm notification propagates when a virtualised resource that is required for the NS network connectivity fails.
TD_NFV_BASE_FM_VR_CLEAR_001	To verify that a fault clearance notification propagates when a failed virtualised resource that is required for the NS network connectivity is recovered
TD_NFV_BASE_FM_VNF_NOTIFY_001	To verify that a VNF fault alarm notification propagates via the VNFM when a VNF fault is triggered by a failed virtualised resource
TD_NFV_BASE_FM_VNF_CLEAR_001	Verify that a VNF fault alarm clearance notification propagates via the VNFM to the MANO when a VNF fault is cleared by resolving a failed virtualised resource
TD_NFV_BASE_NS_LCM_TERMINATE_001	To verify that a NS can be successfully terminated
TD_NFV_BASE_TEARDOWN_DELETE_VNF_PKG_001	To delete a VNF Package

Table 28. Test Group S-VNFM

8.7 Automatic LCM Validation

8.7.1 Test Configuration

The Automatic Life Cycle Management validation group leverages the SUT_BASE configuration as described in the Test Plan [2NFVPLU-TP], with an additional element, a Test System, in charge of automating the triggers and IOP checks as described in the Test Plan. This group was optional for Plugtests participants.

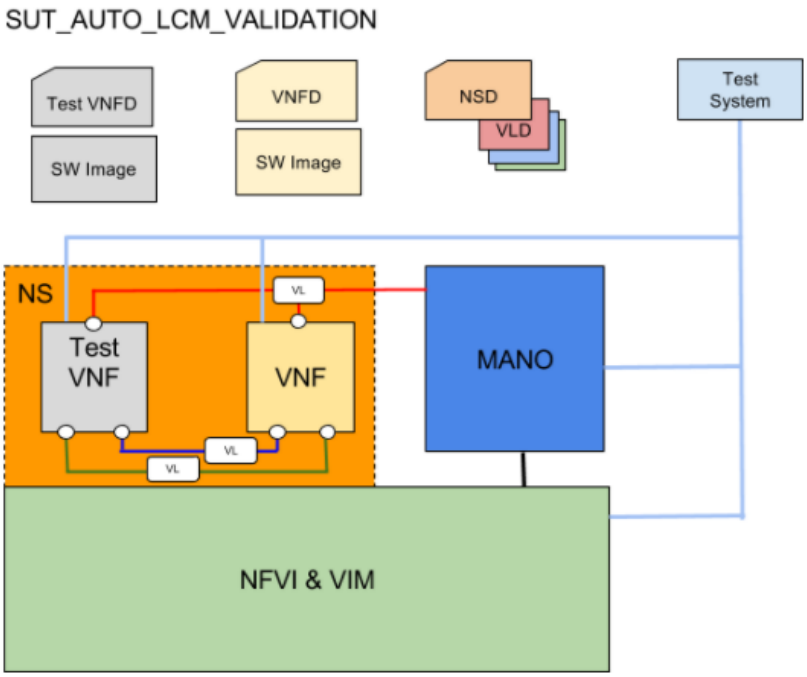

Figure 29. AUTO LCM Validation SUT Configuration: SUT_BASE

8.7.2 Test Cases

Test Id	Test Purpose
TD_NFV_BASE_NS_LCM_INSTANTIATE_001	To verify that an NS can be successfully instantiated
TD_NFV_BASE_NS_LCM_SCALE_OUT_001	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered by a MANO operator
TD_NFV_BASE_NS_LCM_SCALE_IN_001	To verify that a NS can be successfully scaled in (by removing VNF instances) if triggered by a MANO operator
TD_NFV_BASE_NS_LCM_UPDATE_STOP_VNF_001	To verify that a VNF running in a NS can be successfully stopped by MANO
TD_NFV_BASE_NS_LCM_UPDATE_START_VNF_001	To verify that a stopped VNF in a NS can be successfully re-started by MANO
TD_NFV_BASE_NS_LCM_TERMINATE_001	To verify that a NS can be successfully terminated

Table 29. Test Group Auto-LCM-Validation

9 IOP Results

9.1 Overall Results

During the Plugtests, a total of 189 Test Sessions were run: that is, 189 different combinations of the Functions Under Test in scope: VNFs, MANOs and VIM&NFVI were tested for interoperability on a number of different configurations addressing different features.

The following sections provide an overview of the reported results: overall, per test group, pert tests case. To facilitate the analysis, results are presented as follows:

Result	Meaning
OK	Test Case run. Interoperability successfully achieved.
NO	Test Case run. Interoperability not achieved.
NA	Not Applicable: Feature not supported by one or more Functions Under Test
Run	Total number of Test Cases Run = OK + NO
Total	Total number of Test Cases = OK + NO + NA = Run + Not Run

Table 30: IOP Results Interpretation

Note that the tests cases for which no result was reported (i.e. when the test session run out of time) are not taken into account in the Total Results

The table below provides the overall results (aggregated data) from all the test cases run during all the Test Sessions, from all the participating companies, including pre-testing:

Overall Results	Number of Test Sessions	Interoperability (TCs Run)		TCs Not Run	TCs Totals	
		OK	NO	NA	Run	Total
	189	1297 (94.4%)	77 (5.6%)	580 (29.7%)	1374 (70.3%)	1954 (100%)

Table 31: IOP Overall Results

During each Test Session, depending on the targeted configuration and features to be tested, a different number of test cases were offered to the involved participants. See chapter 8 for details.

Overall, the test plan included 47 test cases, organised in different groups as described in chapter 8. Through the 189 Test Sessions run, a total of 1954Test Results were reported. This figure includes both the executed and non-executed test cases. Overall, a total of 1374 individual test cases were run and interoperability results reported for them.

Among the executed test cases, the overall interoperability rate is 94,4%, which indicates a high degree of compatibility among the participating implementations (FUTs) especially considering how the Test Plan used for the 1st NFV Plugtests has been extended with +50% additional test cases to address more complex features and configurations.

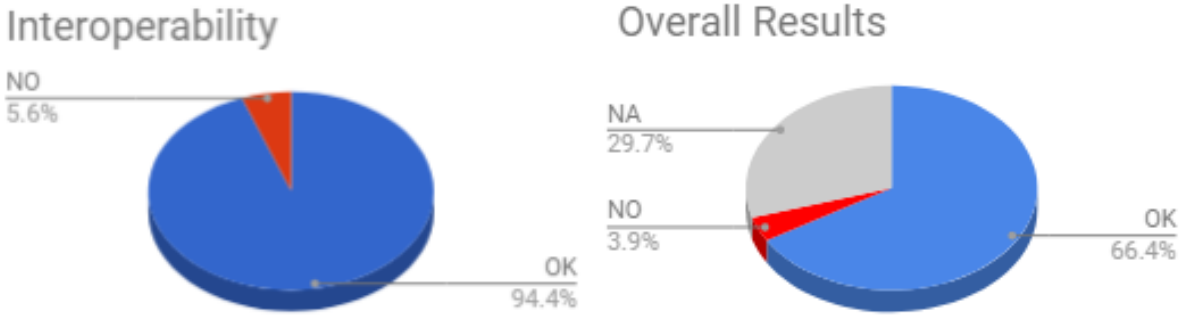

Figure 20. IOP Overall results 2nd NFV Plugtests - 2018 (%)

It is worth noting that, despite the Test Plan extension, the overall execution rate (test cases run vs total amount of test cases offered to participants) has significantly increased with regards to the 1st NFV Plugtests from less than 40% to over 65%, which reflects an increase in the number of features consistently supported and tested across participants and their engagement to go beyond basic functionality (on-boarding, instantiation, …) to address advanced capabilities (fault and performance management, EPA, multi-site, ..)

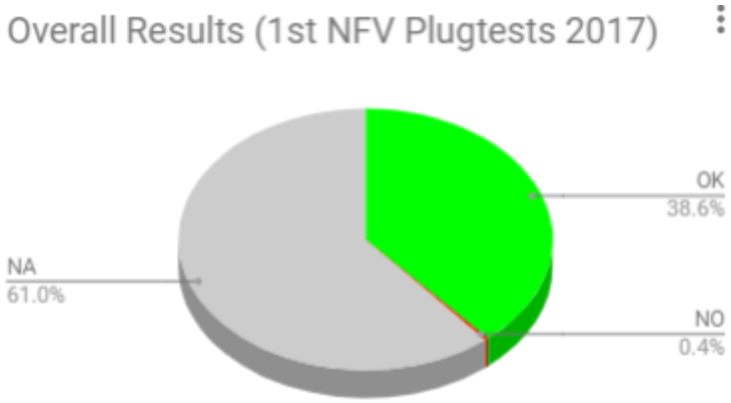

Figure 21. IOP Overall results 1st NFV Plugtests – 2017 (%)

The next clauses present more detailed results per test group and test cases and will allow to identify the areas and features with higher execution and interoperability rates.

9.2 Results per Group

9.2.1 Mandatory Sessions

9.2.1.1 Overview

The table and figure below provide an overview of the results for the test groups in the mandatory Test Sessions. The following clauses present the results on each group and subgroups.

Results per Group (mandatory sessions)	Number of Test Sessions	Interoperability (TCs Run)		TCs Not Run	TCs Totals	
		OK	NO	NA	Run	Total
Pre-testing	88	738 (98.8%)	9 (1.2%)	192 (20.4%)	747 (79.5%)	939
Multi-VNF	76	487 (88.4%)	64 (11.6%)	348 (38.7%)	551(61.3%)	899

Table 32. Results per Group (mandatory sessions)

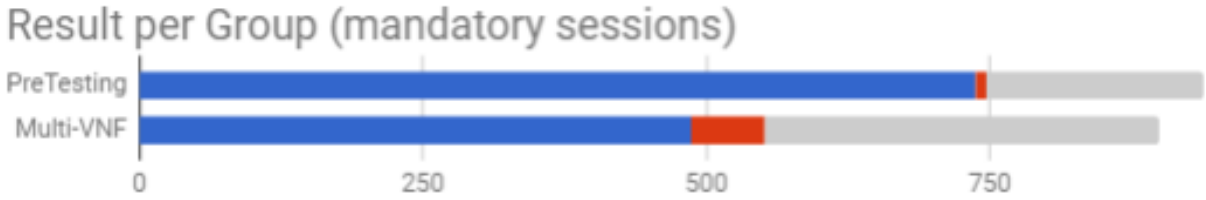

Figure 22. Results per Group (mandatory sessions)

9.2.1.2 Pre-Testing

The pre-testing group included a subset of the test cases run during the 1st NFV Plugtests, which explains the high execution and interoperability rates. For participants that did not attend the previous Plugtests, the test cases in this group were new and allowed to identity and fix a number of interoperability issues. Overall, 88 Pre-Testing test sessions were run with different combinations of VNF, MANO and NFV Platforms.

See Pre-Testing test cases in Clause 8.2, and detailed results per test case in Clause 9.3

9.2.1.3 Multi-VNF

The table and figure below provide an overview of the results per Multi-VNF sub group. Overall, 76 Multi-VNF test sessions were run.

It is worth noting that the test in these groups were run in the context of a multi-vendor Network Service, made of VNFs from different providers, and that some of the sub groups addressing features such as Fault and Performance Management were completely new for Plugtests participants.

It is also interesting to outline a significant increase in the execution rate of the automatic scaling sub groups, which was below 5% in the 1st NFV Plugtests and is now above 40% for most triggers.

See Multi-VNF sub groups' test cases in Clause 8.3, and detailed results per test case in Clause 9.3.

Multi-VNF Sub-Groups	Interoperability (TCs Run)		TCs Not Run	TCs Totals	
	OK	NO	NA	Run	Total
MV_BASIC	296 (99.3%)	2 (0.7%)	2 (0.7%)	298 (99.3%)	300
MV_PM_VR	62 (72.1%)	24 (27.9%)	90 (51.1%)	86 (48.9%)	176
MV_PM_VNF	12 (75.0%)	4 (25.0%)	24 (60.0%)	16 (40.0%)	40
MV_SCALE_NS_VNF_IND	31 (83.8%)	6 (16.2%)	0 (0.0%)	37 (100.0%)	37
MV_SCALE_NS_VIM_KPI	26 (86.7%)	4 (13.3%)	44 (59.5%)	30 (40.5%)	74
MV_SCALE_NS_VNF_REQ	14 (77.8%)	4 (22.2%)	24 (57.1%)	18 (42.9%)	42
MV_SCALE_VNF_VNF_IND	12 (100.0%)	0 (0.0%)	4 (25.0%)	12 (75.0%)	16
MV_SCALE_VNF_VIM_KPI	10 (62.5%)	6 (37.5%)	10 (38.5%)	16 (61.5%)	26
MV_SCALE_VNF_VNF_REQ	0 (0.0%)	0 (0.0%)	4 (100.0%)	0 (0.0%)	4
MV_FM_VR	8 (57.1%)	6 (42.9%)	68 (82.9%)	14 (17.1%)	82
MV_FM_VNF	16 (66.7%)	8 (33.3%)	78 (76.5%)	24 (23.5%)	102

Table 33. Results per Sub-Group (Multi-VNF)

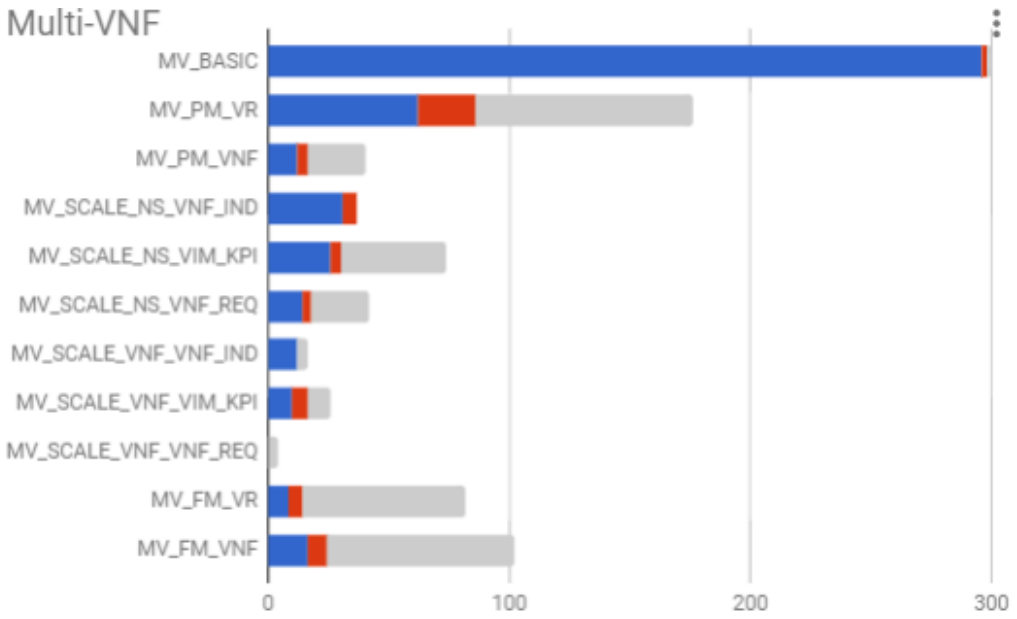

Figure 23. Results per Sub-Group (Multi-VNF)

9.2.2 Optional Sessions

9.2.2.1 Overview

The table and figure below provide an overview of the results for the test groups in the optional Test Sessions. Overall the number of optional test sessions and test cases run in these groups was lower than for the mandatory ones. The following clauses present the results on each group and subgroups.

Results per Group (optional sessions)	Number of Test Sessions	Interoperability (TCs Run)		TCs Not Run	TCs Totals	
		OK	NO	NA	Run	Total
Multi-VNF-EPA	8	9 (75%)	3 (25%)	14 (53.8%)	12 (46.2%)	26
Multi-Site	9	32 (100%)	0 (0%)	2 (5.9%)	32 (94.1%)	34
S-VNFM-D	0	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0
S-VNFM-I	0	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0
Auto-LCM-validation	8	31 (96.9%)	1 (3.1%)	16 (33.3%)	32 (66.7%)	48

Table 34. Results per Group (optional sessions)

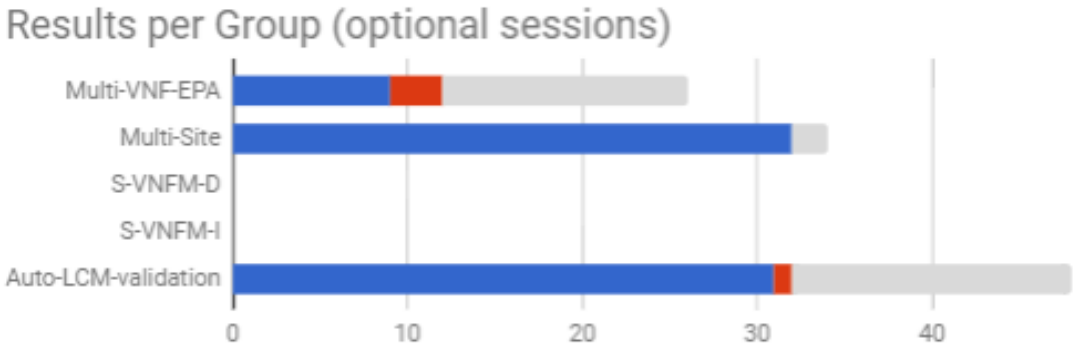

Figure 24. Results per Group (optional sessions)

9.2.2.2 Multi-VNF-EPA

The table and figure below provide an overview of the results for the sub-groups in the Multi-VNF-EPA test group. These tests were optional and could also be run when all the involved FUTs (VNFs, MANO and NFV Platform) supported EPA. Overall, 8 test sessions were run, and a number of issues identified. See Multi-VNF-EPA sub-groups and test cases in Clause 8.4, and detailed results per test case in Clause 9.3.

Multi-VNF-EPA Subgroups	Interoperability		Not Run	Totals	
	OK	NO	NA	Run	Total
MVE_BASIC	5 (62.5%)	3 (37.5%)	0 (0.0%)	8 (100.0%)	8
MVE_SCALE_NS_MANUAL	4 (100.0%)	0 (0.0%)	8 (66.7%)	4 (33.3%)	12
MVE_SCALE_VNF_MANUAL	0 (0.0%)	0 (0.0%)	6 (100.0%)	0 (0.0%)	6

Table 35. Results per Sub-Group (Multi-VNF-EPA)

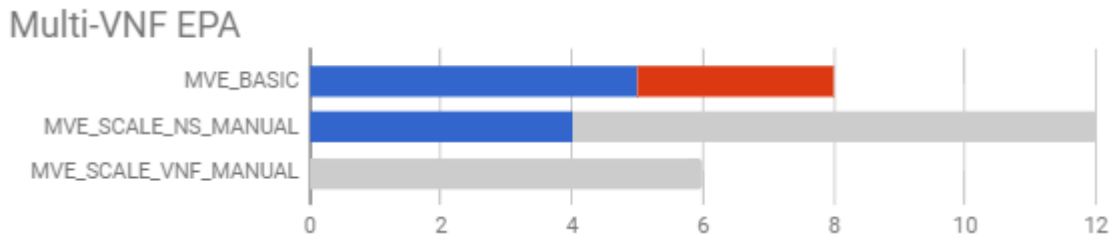

Figure 25. Results per Sub-Group (Multi-VNF-EPA)

9.2.2.3 Multi-Site

The table and figure below provide an overview of the results for the sub-groups in the Multi-VNF test group. These tests were optional and required a high number of FUTs in the Test Session (2+ VNFs, 2+ VNF Platforms, 1 MANO). During the Plugtests, nine multi-site Test Sessions were run, each with a different combination of FUTs. The results of this groups are extremely good, with full interoperability achieved in all the sessions. See Multi-Site sub-groups and test cases in Clause 8.5, and detailed results per test case in Clause 9.3.

Multi-Site Subgroups	Interoperability		Not Run	Totals	
	OK	NO	NA	Run	Total
MS_BASIC	14 (100.0%)	0 (0.0%)	0 (0.0%)	14 (100.0%)	14
MS_SCALE_NS_MANUAL	10 (100.0%)	0 (0.0%)	2 (16.7%)	10 (83.3%)	12
MS_SCALE_VNF_MANUAL	4 (100.0%)	0 (0.0%)	0 (0.0%)	4 (100.0%)	4
Multi-Site-3	4 (100.0%)	0 (0.0%)	0 (0.0%)	4 (100.0%)	4

Table 36. Results per Sub-Group (Multi-Site)

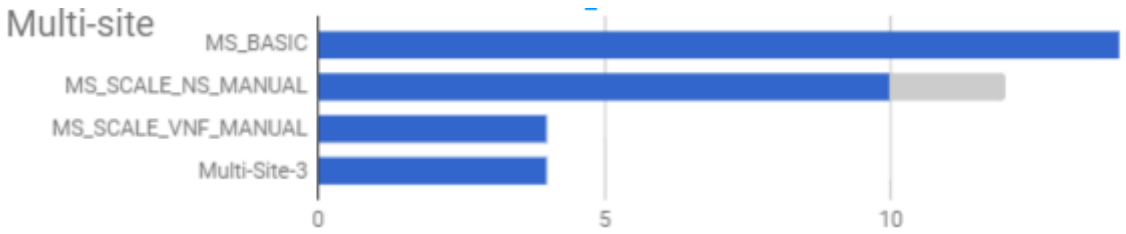

Figure 26. Results per Sub-Group (Multi-Site)

9.2.2.4 Specific VNFM

Specific VNFM test groups were optional during the Plugtests. No results were reported for them on either direct or indirect mode: S-VNFM-D and S-VNFM-I. See the list of tests cases for S-VNFM-D and S-VNFM-I test groups in 8.6.

9.2.2.5 Automatic LCM Validation

The Automatic LCM Validation group was run on 8 test session with different combinations of NFV components (FUTs). The test group included originally 6 Test Cases addressing instantiation, manual scaling, Network Service update and termination. Unfortunately, manual scaling could not be automated on time and these tests could not be run, which explains the relatively high number of N/A results (16/48) in this group. The interoperability rates for the remaining test cases in the group are consistent with the results obtained when running them manually.

See Automatic Life Cycle Management Validation test cases in Clause 8.7, and detailed results per test case in Clause 9.3

9.3 Results per Test Case

The Figure and table below provide an overview of the execution, success and failure rates for each individual test case:

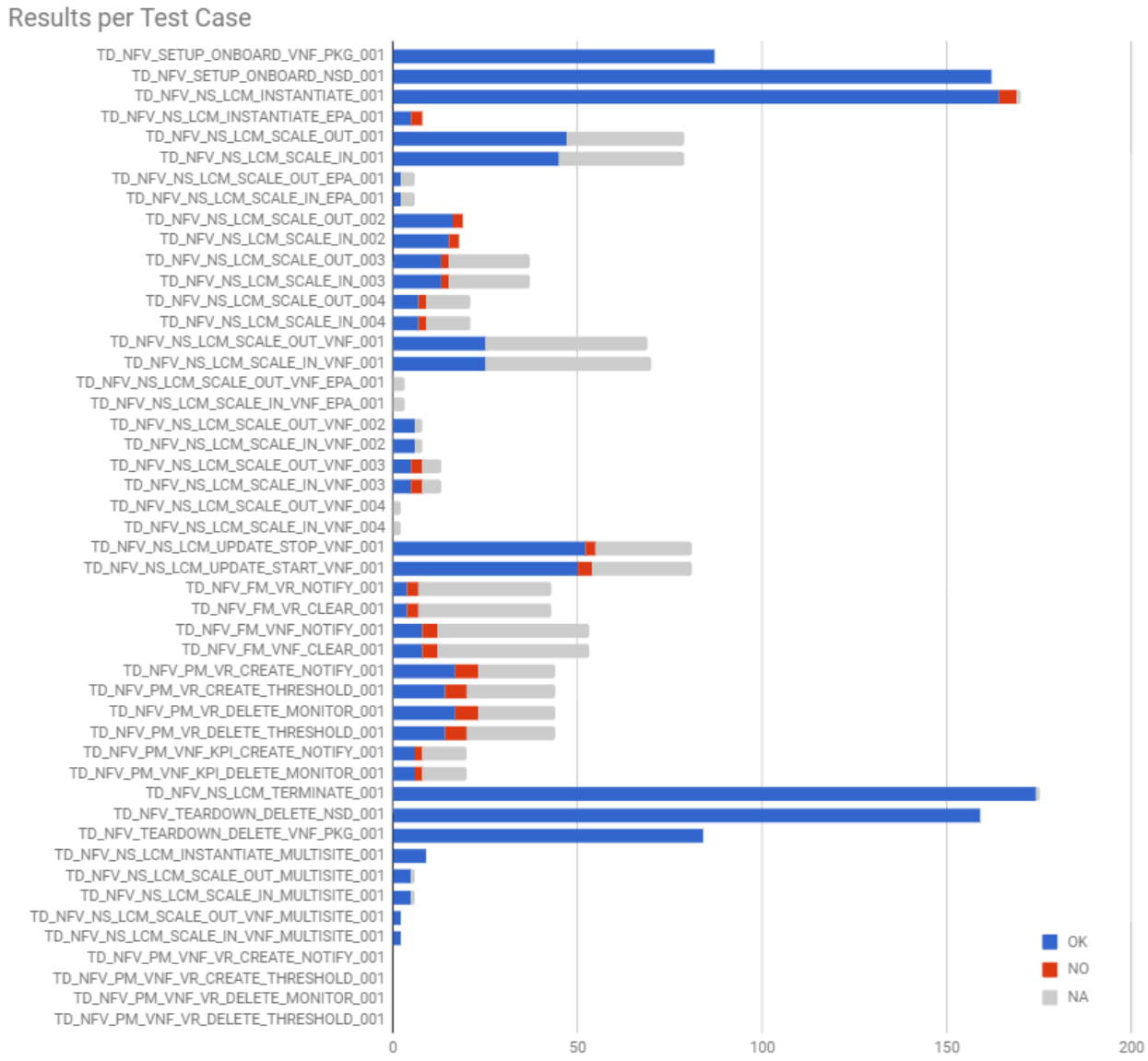

Figure 27. Results per Test Case

Results per Test Case	Interoperability		Not Run	Totals	
	OK	NO	NA	Run	Total
TD_NFV_SETUP_ONBOARD_VNF_PKG_001	87 (100.0%)	0 (0.0%)	0 (0.0%)	87 (100.0%)	87
TD_NFV_SETUP_ONBOARD_NSD_001	162 (100.0%)	0 (0.0%)	0 (0.0%)	162 (100.0%)	162
TD_NFV_NS_LCM_INSTANTIATE_001	164 (97.0%)	5 (3.0%)	1 (0.6%)	169 (99.4%)	170
TD_NFV_NS_LCM_INSTANTIATE_EPA_001	5 (62.5%)	3 (37.5%)	0 (0.0%)	8 (100.0%)	8
TD_NFV_NS_LCM_SCALE_OUT_001	47 (100.0%)	0 (0.0%)	32 (40.5%)	47 (59.5%)	79
TD_NFV_NS_LCM_SCALE_IN_001	45 (100.0%)	0 (0.0%)	34 (43.0%)	45 (57.0%)	79
TD_NFV_NS_LCM_SCALE_OUT_EPA_001	2 (100.0%)	0 (0.0%)	4 (66.7%)	2 (33.3%)	6
TD_NFV_NS_LCM_SCALE_IN_EPA_001	2 (100.0%)	0 (0.0%)	4 (66.7%)	2 (33.3%)	6
TD_NFV_NS_LCM_SCALE_OUT_002	16 (84.2%)	3 (15.8%)	0 (0.0%)	19 (100.0%)	19
TD_NFV_NS_LCM_SCALE_IN_002	15 (83.3%)	3 (16.7%)	0 (0.0%)	18 (100.0%)	18
TD_NFV_NS_LCM_SCALE_OUT_003	13 (86.7%)	2 (13.3%)	22 (59.5%)	15 (40.5%)	37
TD_NFV_NS_LCM_SCALE_IN_003	13 (86.7%)	2 (13.3%)	22 (59.5%)	15 (40.5%)	37
TD_NFV_NS_LCM_SCALE_OUT_004	7 (77.8%)	2 (22.2%)	12 (57.1%)	9 (42.9%)	21
TD_NFV_NS_LCM_SCALE_IN_004	7 (77.8%)	2 (22.2%)	12 (57.1%)	9 (42.9%)	21
TD_NFV_NS_LCM_SCALE_OUT_VNF_001	25 (100.0%)	0 (0.0%)	44 (63.8%)	25 (36.2%)	69
TD_NFV_NS_LCM_SCALE_IN_VNF_001	25 (100.0%)	0 (0.0%)	45 (64.3%)	25 (35.7%)	70
TD_NFV_NS_LCM_SCALE_OUT_VNF_EPA_001	0 (0.0%)	0 (0.0%)	3 (100.0%)	0 (0.0%)	3
TD_NFV_NS_LCM_SCALE_IN_VNF_EPA_001	0 (0.0%)	0 (0.0%)	3 (100.0%)	0 (0.0%)	3
TD_NFV_NS_LCM_SCALE_OUT_VNF_002	6 (100.0%)	0 (0.0%)	2 (25.0%)	6 (75.0%)	8
TD_NFV_NS_LCM_SCALE_IN_VNF_002	6 (100.0%)	0 (0.0%)	2 (25.0%)	6 (75.0%)	8
TD_NFV_NS_LCM_SCALE_OUT_VNF_003	5 (62.5%)	3 (37.5%)	5 (38.5%)	8 (61.5%)	13
TD_NFV_NS_LCM_SCALE_IN_VNF_003	5 (62.5%)	3 (37.5%)	5 (38.5%)	8 (61.5%)	13
TD_NFV_NS_LCM_SCALE_OUT_VNF_004	0 (0.0%)	0 (0.0%)	2 (100.0%)	0 (0.0%)	2
TD_NFV_NS_LCM_SCALE_IN_VNF_004	0 (0.0%)	0 (0.0%)	2 (100.0%)	0 (0.0%)	2
TD_NFV_NS_LCM_UPDATE_STOP_VNF_001	52 (94.5%)	3 (5.5%)	26 (32.1%)	55 (67.9%)	81
TD_NFV_NS_LCM_UPDATE_START_VNF_001	50 (92.6%)	4 (7.4%)	27 (33.3%)	54 (66.7%)	81
TD_NFV_FM_VR_NOTIFY_001	4 (57.1%)	3 (42.9%)	36 (83.7%)	7 (16.3%)	43
TD_NFV_FM_VR_CLEAR_001	4 (57.1%)	3 (42.9%)	36 (83.7%)	7 (16.3%)	43
TD_NFV_FM_VNF_NOTIFY_001	8 (66.7%)	4 (33.3%)	41 (77.4%)	12 (22.6%)	53
TD_NFV_FM_VNF_CLEAR_001	8 (66.7%)	4 (33.3%)	41 (77.4%)	12 (22.6%)	53
TD_NFV_PM_VR_CREATE_NOTIFY_001	17 (73.9%)	6 (26.1%)	21 (47.7%)	23 (52.3%)	44
TD_NFV_PM_VR_CREATE_THRESHOLD_001	14 (70.0%)	6 (30.0%)	24 (54.5%)	20 (45.5%)	44
TD_NFV_PM_VR_DELETE_MONITOR_001	17 (73.9%)	6 (26.1%)	21 (47.7%)	23 (52.3%)	44

TD_NFV_PM_VR_DELETE_THRESHOLD_001	14 (70.0%)	6 (30.0%)	24 (54.5%)	20 (45.5%)	44
TD_NFV_PM_VNF_KPI_CREATE_NOTIFY_001	6 (75.0%)	2 (25.0%)	12 (60.0%)	8 (40.0%)	20
TD_NFV_PM_VNF_KPI_DELETE_MONITOR_001	6 (75.0%)	2 (25.0%)	12 (60.0%)	8 (40.0%)	20
TD_NFV_NS_LCM_TERMINATE_001	174 (100.0%)	0 (0.0%)	1 (0.6%)	174 (99.4%)	175
TD_NFV_TEARDOWN_DELETE_NSD_001	159 (100.0%)	0 (0.0%)	0 (0.0%)	159 (100.0%)	159
TD_NFV_TEARDOWN_DELETE_VNF_PKG_001	84 (100.0%)	0 (0.0%)	0 (0.0%)	84 (100.0%)	84
TD_NFV_NS_LCM_INSTANTIATE_MULTISITE_001	9 (100.0%)	0 (0.0%)	0 (0.0%)	9 (100.0%)	9
TD_NFV_NS_LCM_SCALE_OUT_MULTISITE_001	5 (100.0%)	0 (0.0%)	1 (16.7%)	5 (83.3%)	6
TD_NFV_NS_LCM_SCALE_IN_MULTISITE_001	5 (100.0%)	0 (0.0%)	1 (16.7%)	5 (83.3%)	6
TD_NFV_NS_LCM_SCALE_OUT_VNF_MULTISITE_001	2 (100.0%)	0 (0.0%)	0 (0.0%)	2 (100.0%)	2
TD_NFV_NS_LCM_SCALE_IN_VNF_MULTISITE_001	2 (100.0%)	0 (0.0%)	0 (0.0%)	2 (100.0%)	2
TD_NFV_PM_VNF_VR_CREATE_NOTIFY_001	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0
TD_NFV_PM_VNF_VR_CREATE_THRESHOLD_001	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0
TD_NFV_PM_VNF_VR_DELETE_MONITOR_001	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0
TD_NFV_PM_VNF_VR_DELETE_THRESHOLD_001	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0

Table 37. Results per Test Case

10 API Track

10.1 Scope

The API experimental track at the 2nd ETSI NFV Plugtests aimed at testing compliance with interfaces as specified in the ETSI NFV SOL specifications, namely the target specifications [SOL002] -for the Ve-Vnfm reference point- and [SOL003] – for the Or-Vnfm reference point.

Each of the above documents specifies a set of RESTful interfaces using the HTTP protocol as transport. Each interface (also called API) specifies a set of operations described in term of one HTTP request/response exchange. The scope of the API track focused on checking the formal correctness of the exchanged HTTP payloads for one or more of the available interfaces within a reference point.

At the time of the 2nd NFV Plugtests, ETSI NFV was creating a new Work Item on Conformance Testing. At this stage, the goal of this experimental track was NOT to assess the conformance of participating implementations with NFV specifications, but to gather some common understanding on API testing, evaluate existing tools and derive some initial methodology to be fed back to ETSI NFV work on conformance testing.

Participation to the API track was optional for participants and required the FUT to implement the API(s) under test. Given the reference points in scope, the API track targeted VNF/EM, VNFM and NFVO implementations.

The preparation of the API track was shared with participants during the remote integration phase conf-calls, where thet were invited to provide recommendations and feedback.

10.2 Interfaces

The test system (described in more details in Sect. 10.4) leveraged some machine readable descriptions of the API in scope formatted with the OpenAPI language.

Out of the 17 interfaces in the targeted specifications, the participants were given the possibility to test 14 Interfaces for which OpenAPI definitions were available at that time in the ETSI [FORGE].

Depending on the FUT provided, the availability of NFV API(s) implementation and their interest, each participant was asked to register its participation in the track and the interfaces to be tested during the remote integration part of the event.

For each interface, the tests focused on the compliance of the API producer, not on the API consumers. The available interfaces are listed in the following table.

Interface	Spec	Producer	Web editor
SOL003-VirtualisedResourcesQuotaAvailableNotification-API	SOL003	NFVO	Link
SOL003-VNFFaultManagementNotification-API	SOL003	NFVO	Link
SOL003-VNFIndicatorNotification-API	SOL003	NFVO	Link
SOL003-VNFLifecycleOperationGranting-API	SOL003	NFVO	Link
SOL003-VNFPackageManagement-API	SOL003	NFVO	Link
SOL003-VNFPerformanceManagementNotification-API	SOL003	NFVO	Link
SOL002-VNFConfiguration-API	SOL002	VNF	Link

Interface	Spec	Producer	Web editor
SOL002-VNFIndicator-API	SOL002	VNF/EM	Link
SOL002-VNFIndicatorNotification-API	SOL002	VNFM	Link
SOL003-VNFFaultManagement-API	SOL003	VNFM	Link
SOL003-VNFIndicator-API	SOL003	VNFM	Link
SOL003-VNFLifecycleManagement-API	SOL003	VNFM	Link
SOL003-VNFPackageManagementNotification-API	SOL003	VNFM	Link
SOL003-VNFPerformanceManagement-API	SOL003	VNFM	Link

Table 38. Interfaces in scope

10.3 Logistics

Tests were run jointly by a participant organization (providing the FUT) and the Plugtests team (providing the experimental test system). The test sessions for the API track were scheduled upon request in parallel with the interoperability test sessions.

In order to tightly link the outcome of the test to the interface or API in scope and to remove possible causes of errors in other parts of the SUT, the FUTs where considered in isolation. Any exchange between the test system and the FUT in a given session was made over the interface in scope.

Following the prerequisites defined above, three test configurations where defined and shared with participants. Each configuration involved the test system – playing the role of the API consumer – and the FUT in the role of the API producer. See details in Clause 10.5.1.

10.4 Test System

As described in the test configurations, the test system was required to act as an API consumer for a number of APIs possibly belonging to different NFV components and over different reference points. The capabilities required for the test system where:

- Sending configurable HTTP(S) requests
- Allowing custom payloads to be exchanged

The test system was implemented over the Postman HTTP testing environment, using as base input the OpenAPI descriptions provided by ETSI NFV.

Test execution and test results where operated and recorded manually by the Plugtests Team. Before and during the API test session the set of predefined tests were compiled to accommodate different participants' targets. The resulting test suites (in form of "Postman Collections") were made available in the Plugtests wiki.

10.5 Test plan

The test plan was created according to the APIs and test cases that the participants required. Out of the total 17 APIs specified by ETSI NFV (of which 14 were available in a machine readable format) 5 APIs were tested during the Plugtests.

For each API in the test plan, a subset of supported operations were used to create a total of 29 test cases. Each test case focuses on the test of one operation. The verdict of the test is "pass" if the response is complying with NFV SOL specifications, in terms of:

- Response code
- Response headers
- Response body

The notification system provided in several interfaces was not tested in correspondence of the requests and responses.

The test system provided a CallbackURI for notifications where it was capable of logging all messages and their bodies, for debugging and off-line analysis. During the execution of the tests the entire sessions were recorded in a Pcap file to enable further analysis.

10.5.1 Test Configurations

10.5.1.1 SUT_1_API_VNF

The test configuration as described in the figure below was defined to test the interfaces exposed by a VNF/EM towards the VNFM, such as VNF Configuration API and VNF Indicator API. The test system acts as the VNFM.

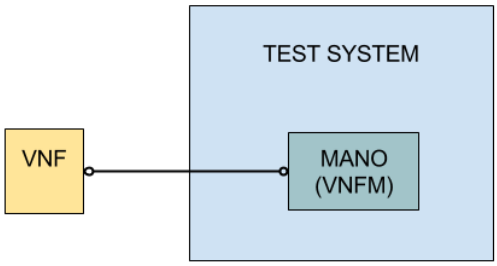

Figure 28: Test configuration SUT_1_API_VNF

10.5.1.2 SUT_1_API_VNFM_OR

The test configuration as described below was defined to test the interfaces exposed by a VNFM towards the NFVO such as VNF Fault Management API, VNF Indicator API, VNF Lifecycle Management API, and VNF Performance Management API. The test system is acting as the NFVO.

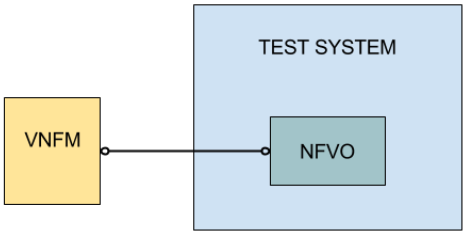

Figure 29: Test configuration SUT_1_API_VNFM_OR

10.5.1.3 SUT_1_API_NFVO

The test configuration as described below was defined to test the interfaces exposed by NFVOs towards the VNFM such as Virtualised Resources Quota Available Notification API, VNF Lifecycle Operation Granting API and VNF Package Management. The test system is acting as the VNFM.

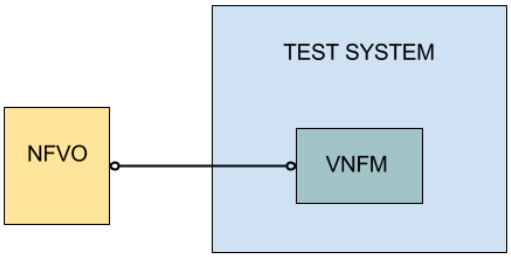

Figure 30: Test configuration SUT_1_API_NFVO

10.5.2 Test Cases

10.5.2.1 VNF Lifecycle Management API (Or-Vnfm)

TD ID	Operation	Resource	Method
TD_API_1	Create VNF Identifier	/vnf_instances	POST
TD_API_2	Query all VNF Identifiers	/vnf_instances	GET
TD_API_3	Query VNF	/vnf_instances/{Id}	GET
TD_API_4	Terminate VNF	/vnf_instances/{Id}/terminate	POST
TD_API_5	Patch VNF Identifier	/vnf_instances/{Id}	PATCH
TD_API_6	Delete VNF Identifier	/vnf_instances/{Id}	DELETE
TD_API_7	Instantiate VNF	/vnf_instances/{Id}/instantiate	GET
TD_API_8	Get Operation Status	/vnf_lcm_op_occs/{Id}	POST
TD_API_9	Subscribe	/subscriptions	POST
TD_API_10	Query all subscriptions	/subscriptions	GET
TD_API_11	Query single subscription	/subscriptions/{Id}	GET
TD_API_12	Terminate Subscription	/subscriptions/{Id}	DELETE
TD_API_13	Notify	/custom	POST

Table 39. VNF Lifecycle Management TCs

10.5.2.2 VNF Package Management API (Or-Vnfm)

TD ID	Operation	Resource	Method
TD_API_14	Query On-boarded VNF Package Information, including obtaining the VNFD	/vnf_packages	GET
TD_API_15	Fetch On-boarded VNF Package	/vnf_packages/{Id}/package_content	GET
TD_API_16	Fetch On-boarded VNF Package Artifacts	/vnf_packages/{Id}/artifacts/{Path}	GET
TD_API_17	Querying/reading on-boarded VNF package information (individual)	/vnf_packages/{Id}	GET
TD_API_18	Reading the VNFD of an on-boarded VNF package	/vnf_packages/{Id}/vnfd	GET

Table 40. VNF Package Management API TCs

10.5.2.3 VNF Lifecycle Operation Granting API (Or-Vnfm)

TD ID	Operation	Resource	Method
TD_API_19	Grant request with synchronous response	/grants	POST

Table 41. VNF Lifecycle Operation Granting API TCs

10.5.2.4 VNF Configuration API (Ve-Vnfm)

TD ID	Operation	Resource	Method
TD_API_20	Read VNF/VNFC configuration from VNF.	/configuration	GET
TD_API_21	Modify VNF/VNFC configuration	/configuration	PATCH
TD_API_22	Modify VNF/VNFC configuration with partial JSON	/configuration	PATCH

Table 42. VNF Configuration API TCs

10.5.2.5 VNF Indicator API (Ve-Vnfm)

TD ID	Operation	Resource	Method
TD_API_23	Query Multiple indicators	/vnfind/v1/indicators	GET
TD_API_24	Query multiple indicators related to a VNF instance.	/indicators/{Id}	GET
TD_API_25	Read individual indicator related to a VNF instance.	/indicators/{Id}/{Id}	GET
TD_API_26	Subscribe	/subscriptions	POST

TD_API_27	Query all subscriptions	/subscriptions	GET
TD_API_28	Read individual subscription	/subscriptions/{Id}	GET
TD_API_29	Terminate Subscription	/subscriptions/{Id}	DELETE

Table 43. VNF Indicator API TCs

10.6 Test Results

During the Plugtests, three different FUTs of different type participated to the API Experimental track, which allowed to test all the targeted reference points in scope on both ends.

Participants executed the tests several times while fixing the implementations, which in the end resulted in an increased number of successful tests.

In total, 4 test sessions were held and 48 individual tests were run (passed or failed). The next figures present the percentages of Test Cases Passed, Failed, Not Run and Not Applicable, overall all per interface. To facilitate the analysis, results are presented as follows:

Result	Meaning
P	Test Case Passed
F	Test Case Failed
NR	Test Case Not Run
NA	Test Case Not Applicable

Table 44: API Track Results Interpretation

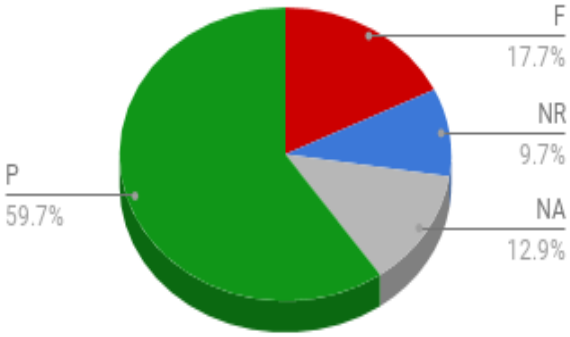

Figure 31: API Track Overall results

The figure below shows the total numbers on the same test outcomes, categorized per API.

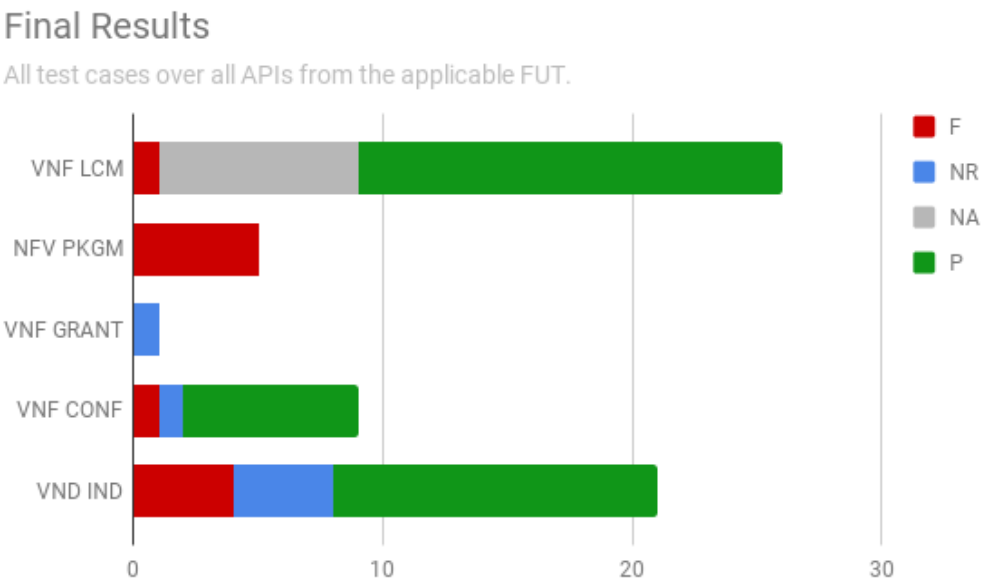

Figure 32: API Track results per interface

11 OSM Hackfest

11.1 Overview

The 1st OSM Hackfest exercised the main functionalities provided by **OSM Release THREE** from basic operations (installation, setup, common operations, etc.) to most advanced activities, with a big focus on VNF on-boarding activities, covering Day 0/1/2 operations .

The main objectives of the OSM Hackfest were to allow participants to:

- Install OSM and run some examples
- Get familiar with OSM's GUI and CLI
- Create their own VNF and NS descriptors and build packages
- Understand how to deploy NS/VNF with EPA
- Learn how to add dynamic LCM with Day-1 and Day-2 actions
- Learn the basics for troubleshooting
- Troubleshoot real cases found in the co-located NFV Plugtests.

The requirements for participation were:

As a minimum:

- Laptop / VM in laptop with Linux installed (preferred, Ubuntu 16.04)
- User-level knowledge of Linux
- Familiarity with NFV and SDN concepts

For optimal participation:

- Laptop / VM in laptop:
    - MINIMUM: 4 CPUs, 8 GB RAM, 40GB disk
    - RECOMMENDED: 8 CPUs, 16 GB RAM, 80GB disk
    - Ubuntu 16.04
- Bring own VNF and basic knowledge of:
    - Its internal structure (diagram recommended) and resource requirements, as in these examples
    - How to change/adapt its configuration
    - Known restrictions/limitations
    - Have VM images available

11.2 Participation

The OSM Hackfest was attended by a wide range of organisations, and actively supported by the OSM community members participating to the NFV Plugtests, especially during the troubleshooting sessions.

Organisation	Role
Allot Communications	VNF Provider
Alpha Networks	VNF Provider
Altran	Integrator, instructor
British Telecom	Network Operator
Canonical	Instructor
Datatronics	Integrator
Deutsche Telekom	Network Operator
FBK	Research
Fortinet	VNF Provider
Huawei	VNF Provider
I2CAT	Research
Intel	Infrastructure provider
Keynetic	VNF Provider
Mavenir	VNF Provider
NextWorks	Integrator
Oracle	VNF Provider
RIFT.io	Integrator
University of Oulu	5GTest Network
UPV/EHU	University
Telenor	Network Operator
Telefonica	Network Operator, instructor
VMware	VIM provider
Whitestack	Integrator, VIM provider

Table 45. OSM Hackfest participation

11.3 Logistics

The 1st OSM Hackfest relayed on HIVE to interconnect participants to the OSM Demo Lab running at ETSI and the OSM Remote Labs made available by the OSM community and hosting different combinations of Hardware, VIMs and SDN Controllers. See details of the setup in Clause 7 Test Infrastructure.

11.4 Outcome

During the one week event, the Hackfest participants learnt and achieved to

1) Install OSM
- Configure LXD
- Install OSM
- Update RO
- Update NAT rules for local access to the UI
- Configure env variables for OSM client

2) Configure VIM accounts
- Add VIM accounts with OSM client
- Use config dictionary for some specific options
- Add an SDN controller from the RO
- Update VIM account from the RO to use external SDN controller
- Add port mapping to a specific VIM from the RO

3) Deploy several VNFs and NSs:
- Basic VNF (single VDU) and basic NS (single VNF)
- Two-VDU VNF and two-VNF NS
- Add day-0 configuration to VNF via cloud-init
- Add EPA capabilities to VNF
- Added day-11/day-2 configuration to VNF by making use of charms

4) Create and build a charm
- layers.yaml, metadata.yaml, actions.yaml
- Implement an action (‘touch’)
- Learn how to map charm actions to VNF primitives in the VNF descriptor
- Learn the way to allow proper identification of interfaces through a metadata server

5) Get familiar with Graphical User Interface
- Catalog, VNF and NS composer, dashboard
- Onboard packages
- Instantiate NS
- Check records to get info like IP address
- Get familiar with command-line client (‘osm’)
- Learn how to generate a VNF package from a terminal

6) Build and deploy own VNF and NS descriptors

In addition, a number of bugs and opportunities for improvement in OSM were reported and fixed during the Hackfest.

11.5 Hackfest material

All the 1st OSM Hackfest material is available online in the [OSM-HACK] wiki, see details here after:

Presentations:

- Welcome
- Introduction to OSM
- Installation and first use
- Session 1. Creating a basic VNF and NS
- Session 2. Modelling multi-VDU VNF
- Session 3. Adding day-0 configuration to your VNF
- Session 4. Modelling EPA capabilities in your VNF descriptor
- Pre-session 5. How to generate VNF package from command line + Charms and VNF primitives in OSM
- Session 5. Adding day-1/day-2 configuration to your VNF. Creating a charm for your VNF
- Guidelines for VNF builders
- Closing session. Recap

VNF, NS packages and images are available in the this FTP folder

Interactive minutes were captured in this pad

12 Plugtests Outcome

12.1 Feedback on the Test Plan

12.1.1 Scaling NS/VNF from VNF Indicators

During the Plugtests it was brought up that the NS or VNF scaling operations can be triggered by VNF indicators in several ways:

- By the VNF/EM sending a VNF indicator notification to the MANO (VNFM)
- By the MANO (VNFM) querying the VNF for VNF indicators

In order to clarify the scope of the Test Descriptions, and to allow testing both approaches independently, it was decided to split the following TDs as follows:

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_SCALE_OUT_002a			
Test Purpose	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered automatically in MANO by a VNF Indicator notification			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.3)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_14] MANO supports scaling by adding/removing VNF instances * [IFS_NFV_VNF_4] VNF can scale out/in by adding/removing VNF instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_BASE_NS_LCM_INSTANTIATE_001) * MANO is configured to trigger SCALE OUT (by adding VNF instances) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger the VNF to send the targeted VNF indicator notification to MANO until the configured threshold is crossed	
	2	IOP Check	Verify that the scale out (by adding VNF instance(s)) procedure has been started in MANO	
	3	IOP Check	Verify that the requested resources have been allocated by the VIM according to the descriptors	
	4	IOP Check	Verify that the additional VNF instance(s) have been deployed	
	5	IOP Check	Verify that the additional VNF instance(s) are running and reachable through the management network	
	6	IOP Check	Verify that the additional VNF instances(s) have been configured according to VNFD (i.e by obtaining a result from the management interface)	
	7	IOP	Verify that the additional VNF instances(s), VL(s) and VNFFG(s) are connected	

		Check	according to the Descriptors	
	8	IOP Check	Verify that NS has been scaled out by running the end-to-end functional test	
IOP Verdict				

Table 46. TD_NFV_BASE_NS_LCM_SCALE_OUT_002a (new TD)

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_SCALE_IN_002a			
Test Purpose	To verify that a NS can be successfully scaled in (by removing VNF instances) if triggered automatically in MANO by a VNF Indicator notification			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.3)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_14] MANO supports scaling by adding/removing VNF instances * [IFS_NFV_VNF_4] VNF can scale out/in by adding/removing VNF instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_BASE_NS_LCM_INSTANTIATE_001) * NS has been scaled out by adding VNF instances * MANO is configured to trigger SCALE IN (by removing VNF instances) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger the VNF to send the targeted VNF indicator notification to MANO until the configured threshold is crossed	
	2	IOP Check	Verify that the scale in (by removing VNF instance(s)) procedure has been started in MANO	
	3	IOP Check	Verify that the impacted VNF instance(s) have been terminated	
	4	IOP Check	Verify that the impacted VNF related resources have been released by the VIM	
	5	IOP Check	Verify that the remaining VNF instances(s) are still running and reachable through the management network	
	6	IOP Check	Verify that the remaining VNF instances(s), VL(s) and VNFFG(s) are still connected according to the descriptors	
	7	IOP Check	Verify that NS has been scaled in by running the end-to-end functional test	
IOP Verdict				

Table 47. TD_NFV_BASE_NS_LCM_SCALE_IN_002a (new TD)

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_SCALE_OUT_002b			
Test Purpose	To verify that a NS can be successfully scaled out (by adding VNF instances) if triggered automatically in MANO by a querying VNF Indicator by querying a VNF Indicator			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.4)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_14] MANO supports scaling by adding/removing VNF instances * [IFS_NFV_VNF_4] VNF can scale out/in by adding/removing VNF instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_BASE_NS_LCM_INSTANTIATE_001) * MANO is configured to trigger SCALE OUT (by adding VNF instances) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1a	Stimulus	~~Trigger the VNF/EM to send the targeted VNF indicator to MANO until the configured threshold is crossed~~ In the VNF, trigger the target VNF indicator to cross the configured auto-scaling threshold value for scale out operation	
	1b	Stimulus	Trigger MANO to query the VNF for retrieving a new value of the VNF indicator	
	2	IOP Check	Verify that the scale out (by adding VNF instance(s)) procedure has been started in MANO	
	3	IOP Check	Verify that the requested resources have been allocated by the VIM according to the descriptors	
	4	IOP Check	Verify that the additional VNF instance(s) have been deployed	
	5	IOP Check	Verify that the additional VNF instance(s) are running and reachable through the management network	
	6	IOP Check	Verify that the additional VNF instances(s) have been configured according to VNFD (i.e by obtaining a result from the management interface)	
	7	IOP Check	Verify that the additional VNF instances(s), VL(s) and VNFFG(s) are connected according to the Descriptors	
	8	IOP Check	Verify that NS has been scaled out by running the end-to-end functional test	
IOP Verdict				

Table 48. TD_NFV_BASE_NS_LCM_SCALE_OUT_002b (new TD)

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_SCALE_IN_002b			
Test Purpose	To verify that a NS can be successfully scaled in (by removing VNF instances) if triggered automatically in MANO by querying a VNF Indicator			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.4)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_14] MANO supports scaling by adding/removing VNF instances * [IFS_NFV_VNF_4] VNF can scale out/in by adding/removing VNF instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_BASE_NS_LCM_INSTANTIATE_001) * NS has been scaled out by adding VNF instances * MANO is configured to trigger SCALE IN (by removing VNF instances) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1a	Stimulus	~~Trigger the VNF/EM to send the targeted VNF indicator to MANO until the configured threshold is crossed~~ In the VNF, trigger the target VNF indicator to cross the configured auto-scaling threshold value for scale in operation	
	1b	Stimulus	Trigger MANO to query the VNF for retrieving a new value of the VNF indicator	
	2	IOP Check	Verify that the scale in (by removing VNF instance(s)) procedure has been started in MANO	
	3	IOP Check	Verify that the impacted VNF instance(s) have been terminated	
	4	IOP Check	Verify that the impacted VNF related resources have been released by the VIM	
	5	IOP Check	Verify that the remaining VNF instances(s) are still running and reachable through the management network	
	6	IOP Check	Verify that the remaining VNF instances(s), VL(s) and VNFFG(s) are still connected according to the descriptors	
	7	IOP Check	Verify that NS has been scaled in by running the end-to-end functional test	
IOP Verdict				

Table 49. TD_NFV_BASE_NS_LCM_SCALE_IN_002b (new TD)

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_002a			
Test Purpose	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered automatically in MANO by a VNF Indicator notification			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.3)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_15] MANO supports scaling out/in by adding/removing VNFC instances * [IFS_NFV_VNF_5] VNF can scale out/in by adding/removing VNFC instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_BASE:NS_LCM_INSTANTIATE_001) * MANO is configured to trigger SCALE OUT (by adding VM(s)) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger the VNF to send the targeted VNF indicator notification to MANO until the configured threshold is crossed	
	2	IOP Check	Verify that the scale out (by adding VNFC instances (VMs)) procedure has been started in MANO	
	2	IOP Check	Verify that the requested resources have been allocated by the VIM according to the descriptors	
	3	IOP Check	Verify that the additional VM(s) have been deployed (i.e by querying the VIM)	
	4	IOP Check	Verify that the additional VM(s) are running and are reachable through the management network	
	5	IOP Check	Verify that the additional VM(s) are connected to the VL(s) according to the descriptors	
	6	IOP Check	Verify that NS has been scaled out by running the end-to-end functional test	
IOP Verdict				

Table 50. TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_002a (new TD)

Interoperability Test Description				
Identifier	TD_NFV_NS_LCM_BASE_SCALE_IN_VNF_002a			
Test Purpose	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered automatically in MANO by a VNF Indicator notification			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.3)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_15] MANO supports scaling out/in by adding/removing VNFC instances * [IFS_NFV_VNF_5] VNF can scale out/in by adding/removing VNFC instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_NS_LCM_INSTANTIATE_001) * NS has been scaled out by adding VM(s) * MANO is configured to trigger SCALE IN (by removing VM(s)) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger the VNF to send the targeted VNF indicator notification to MANO until the configured threshold is crossed	
	2	IOP Check	Verify that the scale out (by removing VNFC instances (VMs)) procedure has been started in MANO	
	3	IOP Check	Verify that the impacted VM(s) have been terminated	
	4	IOP Check	Verify that the impacted VM related resources have been released by the VIM	
	5	IOP Check	Verify that the remaining VM(s) are still running and reachable through the management network	
	6	IOP Check	Verify that the remaining VM(s) and VL(s) are still connected according to the descriptors	
	7	IOP Check	Verify that NS has been scaled in by running the end-to-end functional test	
IOP Verdict				

Table 51. TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_002a (new TD)

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_002b			
Test Purpose	To verify that a VNF in a NS can be successfully scaled out (by adding VNFC instances (VMs)) when triggered automatically in MANO by querying a VNF Indicator			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.4)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_15] MANO supports scaling out/in by adding/removing VNFC instances * [IFS_NFV_VNF_5] VNF can scale out/in by adding/removing VNFC instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_BASE:NS_LCM_INSTANTIATE_001) * MANO is configured to trigger SCALE OUT (by adding VM(s)) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1a	Stimulus	~~Trigger the VNF/EM to send the targeted VNF indicator to MANO until the configured threshold is crossed~~ In the VNF, trigger the target VNF indicator to cross the configured auto-scaling threshold value for scale out operation	
	1b	Stimulus	Trigger MANO to query the VNF for retrieving a new value of the VNF indicator	
	2	IOP Check	Verify that the scale out (by adding VNFC instances (VMs)) procedure has been started in MANO	
	2	IOP Check	Verify that the requested resources have been allocated by the VIM according to the descriptors	
	3	IOP Check	Verify that the additional VM(s) have been deployed (i.e by querying the VIM)	
	4	IOP Check	Verify that the additional VM(s) are running and are reachable through the management network	
	5	IOP Check	Verify that the additional VM(s) are connected to the VL(s) according to the descriptors	
	6	IOP Check	Verify that NS has been scaled out by running the end-to-end functional test	
IOP Verdict				

Table 52. TD_NFV_BASE_NS_LCM_SCALE_OUT_VNF_002b (new TD)

Interoperability Test Description				
Identifier	TD_NFV_NS_LCM_BASE_SCALE_IN_VNF_002b			
Test Purpose	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered automatically in MANO by querying a VNF Indicator			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.4)			
Applicability	* [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_15] MANO supports scaling out/in by adding/removing VNFC instances * [IFS_NFV_VNF_5] VNF can scale out/in by adding/removing VNFC instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_NS_LCM_INSTANTIATE_001) * NS has been scaled out by adding VM(s) * MANO is configured to trigger SCALE IN (by removing VM(s)) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1a	Stimulus	~~Trigger the VNF/EM to send the targeted VNF indicator to MANO until the configured threshold is crossed~~ In the VNF, trigger the target VNF indicator to cross the configured auto-scaling threshold value for scale in operation	
	1b	Stimulus	Trigger MANO to query the VNF for retrieving a new value of the VNF indicator	
	2	IOP Check	Verify that the scale out (by removing VNFC instances (VMs)) procedure has been started in MANO	
	3	IOP Check	Verify that the impacted VM(s) have been terminated	
	4	IOP Check	Verify that the impacted VM related resources have been released by the VIM	
	5	IOP Check	Verify that the remaining VM(s) are still running and reachable through the management network	
	6	IOP Check	Verify that the remaining VM(s) and VL(s) are still connected according to the descriptors	
	7	IOP Check	Verify that NS has been scaled in by running the end-to-end functional test	
IOP Verdict				

Table 53. TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_002b (new TD)

The above changes have been implemented in the latest revision of the NFV Plugtests Test Plan [2NFVPLU-TP]

12.1.2 NS Update – Stop VNF

Following discussions on the scope of Network Service Update – Stop VNF operations started during the 1st NFV Plugtests, the clarification hereafter was made in the TD_NFV_BASE_NS_LCM_UPDATE_STOP_VNF_001 Test Description:

Interoperability Test Description				
Identifier	TD_NFV_BASE_NS_LCM_UPDATE_STOP_VNF_001			
Test Purpose	To verify that a VNF running in a NS can be successfully stopped by MANO			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA013 V2.3.1 (clause 7.3.5) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.11)			
Applicability	* MANO can request VIM_NFVI to stop VM(s) * VIM_NFVI supports stopping VM(s)			
Pre-test conditions	* NS is instantiated (TD_NFV_NS_LCM_INSTANTIATE_001) * VNF instance(s) in the NS are running			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger the VNF(s) stop operation in MANO	
	2	IOP Check	Verify the VNF(s) state inside the NS is "Stopped" on MANO (query, display, ...)	
	3	IOP Check	Verify that individual VM(s) inside the VNF(s) are ~~stopped~~ shutdown on VIM (i.e query or display the state from VIM)	
IOP Verdict				

Table 54. TD_NFV_BASE_NS_LCM_UPDATE_STOP_VNF_001 (updated TD)

The above changes have been implemented in the latest revision of the NFV Plugtests Test Plan [2NFVPLU-TP]

12.1.3 FM: Non-VR related fault propagation

In the context of Fault Management and propagation, the question was raised on whether it could be foreseen that the VNFM propagated to the NFVO VNF faults non directly related to VR. While it was a common understanding that purely application related faults wouldn't be propagated through the NFVO, it was not clear if it would be in scope that the VNFM propagated a life cycle related fault to the NFVO, for instance, in the case it detected that the VNF was not alive any longer.

After study of NFV Specs and discussion with the NFV IFA working group, the conclusion was that this case is in scope, and explicitly covered by IFA008 clause 8.8 as "Faults detected by the VNFM":

8.8 Information elements and notifications related to VNF Fault Management

8.8.1 Introduction

This clause defines information elements and notifications related to VNF Fault Management.

8.8.2 AlarmNotification

8.8.2.1 Description

This notification informs the receiver of alarms related to the VNFs managed by the VNFM. Alarms are created in response to:

- *faults detected by the VNFM; and*
- *faults generated due to changes in the state of virtualised resources used by the VNF instances managed by the VNFM.*

The notification is mandatory.

The Plugtests test plan, as TST007, only addresses the case of propagation of VR related alarms. The recommendation is to extend the PM test group to cover non-VR related alarm propagation, that is, of faults detected by the VNFM.

This issue has been fed back to NFV TST working group for discussion during [TST007] maintenance.

12.1.4 TDs Applicability

During the Test Plan development for this 2nd NFV Plugtests, it was agreed to simplify the applicability field in the Test Descriptions to focus on the Interoperability Feature Statements covering the specific features in scope as compiled during the preparation phase. In a nutshell, basic features expected to be supported by any implementation (i.e resource allocation) are not explicitly listed any longer, and the listed features are clearly identified with an IFS ID.

See below an example of the implementation of this simplification:

Interoperability Test Description				
Identifier	TD_NFV_NS_LCM_BASE_SCALE_IN_VNF_002			
Test Purpose	To verify that a VNF in a NS can be successfully scaled in (by removing VNFC instances (VMs)) when triggered automatically in MANO by a VNF Indicator notification			
Configuration	SUT_BASE SUT_S-VNFM-D SUT_S-VNFM-I			
References	ETSI GS NFV-IFA005 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA006 V2.3.1 (clause 7.3.1.2, 7.4.1.2, 7.5.1.2) ETSI GS NFV-IFA007 V2.3.1 (clause 7.2.4) ETSI GS NFV-IFA008 V2.3.1 (clause 6.3.3)			
Applicability	~~* MANO can request VIM_NFVI to terminate virtualised resources~~ ~~* VIM_NFVI supports terminating virtualised resources~~ ~~* MANO supports receiving VNF indicators from VNF/EM~~ ~~* VNF/EM can send VNF indicator values to MANO~~ ~~* MANO supports triggering scale in when a given VNF Indicator value crosses a certain threshold~~ ~~* MANO supports scale in by removing VNFC instances (VMs)~~ ~~* NS/VNF supports scale in by removing VNFC instances (VMs)~~ * [IFS_NFV_MANO_17] MANO supports receiving VNF indicators from VNF/EM * [IFS_NFV_MANO_18] MANO supports automatic scaling triggered by VNF indicators from VNF/EM * [IFS_NFV_MANO_15] MANO supports scaling out/in by adding/removing VNFC instances * [IFS_NFV_VNF_5] VNF can scale out/in by adding/removing VNFC instances * [IFS_NFV_VNF_9] VNF can send indicators (KPIs) to MANO			
Pre-test conditions	* NS is instantiated (TD_NFV_NS_LCM_INSTANTIATE_001) * NS has been scaled out by adding VM(s) * MANO is configured to trigger SCALE IN (by removing VM(s)) when a given VNF Indicator value crosses a certain threshold			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger the VNF to send the targeted VNF indicator notification to MANO until the configured threshold is crossed	
	2	IOP Check	Verify that the scale out (by removing VNFC instances (VMs)) procedure has been started in MANO	
	3	IOP Check	Verify that the impacted VM(s) have been terminated	
	4	IOP Check	Verify that the impacted VM related resources have been released by the VIM	
	5	IOP Check	Verify that the remaining VM(s) are still running and reachable through the management network	
	6	IOP Check	Verify that the remaining VM(s) and VL(s) are still connected according to the descriptors	
	7	IOP Check	Verify that NS has been scaled in by running the end-to-end functional test	
IOP Verdict				

Table 55. TD_NFV_BASE_NS_LCM_SCALE_IN_VNF_002 (new approach)

The above changes have been implemented in the latest revision of the NFV Plugtests Test Plan [2NFVPLU-TP]

12.2 Feedback on NFV Specifications

12.2.1 PM/FM by VNFM in indirect mode

During the Plugtests there was some discussion about whether in indirect mode, where the NFVO is in charge of VNF related resource management, we could have a VNFM dealing with VR performance and/or fault management.

During the discussion a possible scenario was foreseen of a multi-site deployment where the VNFM would be physically closer to the VNF than the NVFO and allow for a closer loop and shorter response time if performance and/or faults where monitored directly by the VNFM.

No explicit information on the support of the configuration above has been found on NFV IFA specs. This point has been fed back to ETSI NFV IFA working group.

12.2.2 VNF Configuration interface for EM

During the API experimental track, questions were raised from the participants on the reasons for having the VNF Configuration interface and API specified only on the Ve-Vnmf-vnf reference point and not on the Ve-Vnfm-em.

This questions has been fed back to ETSI NFV SOL and IFA working groups and will be studied in the context of [SOL002] and [IFA008].

12.2.3 Empty collection of resources handling in APIs

During the experimental API track, some inconsistencies were detected when handling empty collections of resources in the responses sent by an API producer according to [SOL003].

The question was raised on the Subscriptions resource in the VNF Lifecycle Management Interface, but could potentially impact any other collection of resources.

During the testing, it was observed that in case of no resources for Subscriptions, the Function Under Test responded with a 404 code, following the specification Clause 4.3.5.5 in [SOL003]:

***404 Not Found**: If the API producer did not find a current representation for the resource addressed by the URI passed in the request, or is not willing to disclose that one exists, it shall respond with this response code. The "ProblemDetails" structure may be provided, including in the "detail" attribute information about the source of the problem, e.g. a wrong resource URI variable.*

The test system, instead, expected a successful response with an empty array of resource representations, following the provisions of Table 5.4.18.3.2-2 of the same specification [SOL003]:

Table 5.4.18.3.2-2: Details of the GET request/response on this resource

Request body	Data type	Cardinality	Description	
	n/a			
Response body	**Data type**	**Cardinality**	**Response Codes**	**Description**
	LccnSubscription	0..N	200 OK	The list of subscriptions was queried successfully. The response body shall contain the representations of all active subscriptions of the functional block that invokes the method.
	ProblemDetails	1	400 Bad Request	Error: Invalid attribute-based filtering parameters. The response body shall contain a ProblemDetails structure, in which the "detail" attribute should convey more information about the error.
	ProblemDetails	See clauses 4.3.5.4 / 4.3.5.5	4xx/5xx	In addition to the response codes defined above, any common error response code as defined in clause 4.3.5.4, and any other valid HTTP error response as defined in clause 4.3.5.5, may be returned.

Figure 33. Table 5.4.18.3.2-2 in [SOL003]

This issue has been fed back to ETSI NFV SOL working group.

12.3 Feedback on IOP

12.3.1 Multi-vendor Network Services

While not directly in scope of NFV testing, building multi-vendor Network Services required a significant effort of alignment and integration to prepare the inter-VNF communication and test traffic patterns. This task needs to be planned and included in the pre-testing phase to ensure efficient face to face sessions.

12.3.2 Multi-Site deployments

Multi-site deployments were successfully achieved during Plugtests, with MANO solutions deploying Network Services across several remote VIM&NFVI. This testing relied on the HIVE network to fully interconnect all the involved parties (MANO and VIMs), which required to define some guidelines in terms of inter-VIM connectivity and test traffic.

12.3.3 Automated IOP Testing

During this Plugtests, a subset of the test plan was automated and successfully run on 8 combinations of participating components. The Test Configuration was adapted depending on the VNF and Network Service under test to include the number of Test VNFs required to generate the appropriate type of traffic and validate end to end functionality.

Automating the test plan allowed to run test cases repeatedly and analyse additional parameters such as the system response time, results stability, and dependency on other factors, for instance, the system load. Overall, it was observed that networking instability was one of the most regular root causes of test cases failure.

One of the conclusions of the automation efforts is that common APIs across implementations and towards the test system simplify and reduce the integration and automation efforts.

12.4 Feedback on API testing

The experience of designing and executing the test plan for the API experimental track, and the outcomes from the test execution resulted in a set of interesting learnings, summarized below.

12.4.1 Scope of the testing

The experience in the API track showed that errors in the Request and Response formatting (URI structure, header names or content) in both the Test System and the implementations can be easily detected and fixed.

On the other hand, inconsistencies and more relevant errors are identified when combining single operations in more complex flows, which should be taken into account when aiming for conformance testing.

12.4.2 Tests configurations

The concept of testing the FUT "in isolation" for API testing is a valid principle which reduces complexity but sometimes prevents from catching erroneous behaviours in the FUT.

In the context of future conformance testing, it could be advisable to re-think the test configurations in order to enable the addition of other NFV components either in the Test System or in the System Under Test.

In the case of testing a VNFM, the main two reasons identified for extending the configurations are:

- On the Or-vnmf reference point, some operations require the VNFM to ask the API consumer (the NFVO) for resources before replying;
- On the Ve-vnfm reference point, during the creation of a VNF instance, the information on the VIM to be used is actually given by the NFVO to the VNFM through the Lifecycle Operation Granting API, thus requiring to have an NFVO in the configuration.

12.4.3 OpenAPIs

The experimental API track allowed to identify some bugs in the OpenAPIs definitions, mainly misalignments with the NFV SOL specifications. These bugs were reported on the ETSI Forge.

12.4.4 Test Suites

The test suites for the experimental API track were generated automatically from the OpenAPI definitions which reduced significantly the amount of effort required to produce them.

This being said, the Postman Collections generated automatically required some post-processing and improvements in order to fit the test purpose, and especially to allow for different parameters in the FUT.

This feedback will be contributed to ETSI NFV TST working group to guide future work on NFV conformance testing.

History

Document history		
V0.0.1	31/01/2018	1st internal draft
V0.0.2	04/02/2018	2nd draft for review
V0.0.3	12/02/2018	3rd draft with participants feedback
V0.0.4	16/02/2018	4th draft with API track
V1.0.0	19/02/2018	Publication