VIAVI Solutions

White Paper

NFV Enabling Automation and Performance Validation for Network Services and Labs

NFV enabling automation and performance validation for Network Services and Labs

The Impact of NFV

Network function virtualization (NFV) is revolutionizing both how carriers enable network services and how network solution vendors approach and deliver network products. With carriers seeking to deliver new network services to their customers in weeks rather than months, carriers are aggressively pushing solution vendors to deliver more in shorter time frames.

To meet these aggressive time-lines lab automation is now a critical component to success for NFV. The goal of this application note is to enable both carriers and solution vendors to make an informed decision on what automation means in terms of delivery of a NFV enabled network service. But it also shows how automation for NFV differs from the traditional tools that may exist in labs today.

The ETSI ISG NFV (http://www.etsi.org/technologies-clusters/technologies/nfv), is one of the driving forces behind recommending how carriers and solution vendors approach the future of networking. A key take-away from their published documentation is that the old ways of delivering product and the automation processes used need to change. This disruptive transformation means that for the test and measurement industry that a new paradigm of delivering test solutions is now upon us.

NFV driving transformation

Progressive carriers are at an inflection point, in which they no longer see or accept the need for proprietary hardware and/or vendor solution lock-in. The challenge for many, is how now to adopt and be commercially successful in this new technology shift, one in which agility and flexibility are core.

It's clear that no two carriers are the same and will take a different approach to NFV, for example this will include the core technology choices for the management and orchestration (MANO) stack. For network solution vendors and test solution vendors, the challenge is to be able to provide solutions which are open and agile, which are agnostic to their surroundings i.e. can easily fit into these new platforms. This will include the NFV automation process wheel of: onboarding, cataloging, orchestration, deployment and service chaining.

Demystifying MANO

The ETSI ISG NFV has delivered multiple recommendations covering the key topics of: architecture design, management flows, orchestration principles, performance management, test and security best practices. Of particular interest for lab automation is the management and orchestration (MANO) recommendations.

The vision from the carriers is that the next generation of networks will be open, in that the chosen components can easily be swapped out. This principle is a driving force behind a number of open-source communities e.g. OpenStack, OpenMANO, OpenNFV, etc.

The NFV recommendation for the MANO provides for a modular structure of three key layers, each module providing a level of abstraction enabling simplification of use for the carrier network operations, essentially removing the need for highly skilled personnel to deliver a custom network service. The modules outlined in the recommendations address:

- NFV Orchestration (NFVO) – Onboarding and catalogue management of virtual network functions (VNF)
- VNF Management (VNFM) – Management of the lifecycle of the VNF i.e. update/upgrade/etc
- Virtual Infrastructure Management (VIM) – Resource manager for compute, storage and network

To enable the abstraction between modules requires an open application programming interface (API) between the modules. With many communities and vendors now electing to use Representational State Transfer (REST), which is simply a client – server communication using the HTTP protocol to communicate instruction sets.

MANO and the proposed capabilities are a critical piece of the carriers NFV vision. This means that for network solution vendors to prove that their solutions are indeed NFV ready, they must deliver or push the VNF through the same information or objective modeling process and automation process flow that the carrier have deployed. The advantage of doing so provides automation to validate packaging, onboarding capabilities, catalogue presentation of metadata, accuracy of deployment scenarios and of course performance in the custom network service.

NFV the changing lab scene

For many to prove NFV readiness and to automate labs for a NFV process flow; is challenging and potentially cost prohibitive. The fact is - it does not have to be and in this paper we look at using open-source components to deliver parts of the MANO architecture. For example a simple and easy to use virtual infrastructure resource manager is available in the OpenMANO open-source project supported by the global carrier Telefonica. [https://github.com/nfvlabs/openmano].

The OpenVIM component of OpenMANO provides a number of advantages which are highlighted later, but fundamentally is a good foundation for understanding on how products behave in NFV platforms. A further observation is that the software being promoted in the NFV community provides a similar functionality as that of proprietary lab automation solutions, especially in terms of cataloging, management of deployments and resource management.

NFV Architecture and Uses in Lab Automation

As a starting point for this application note, it's worth defining a sample platform architecture. The network function virtual infrastructure (NFVI) i.e. where the network service will run, utilizes standard hardware blade servers (fundamental to the NFV vision) and also uses commodity network interface cards, hence minimizing the costs of building out scaled labs.

An open and standard operating system such as RedHat Ubuntu is used to deliver the core operating system, on which the hypervisor is installed. In this instance the Kernel based Virtual Machine (KVM) with Quick Emulator (QEMU) provides the virtualization layer. The popular virtual machine management library (Libvirt) provides a standard interface to assist in managing the virtual machine (VM) instances.

For networking, the open libraries for open vswitch are installed along with the OpenFlow controller enabling centralized management for networking. This approach enables ease of configuring and managing the associated network service (NS) VNF networking topologies, which can be defined as an information model such as a VNF forwarding graph (VNF-FG).

As part of the initial investigations into creating the application note, a decision was taken to prove the flexibility of the MANO stack architecture approach and its APIs. In doing so the logical separation was to implement a proprietary solution to act as the NFVO/VNFM and to use the OpenVIM from the OpenMANO project. The combined VNFM and NFVO is provided by Luxoft which is known as SuperCloud.

The decision to use a proprietary solution for the upper layers of MANO provide a number of advantages:

- API stability – fixed API implementations e.g. orchestrator and VIM (Or-Vi)
- Integration capabilities – enabling performance validation results from TeraVM to be exposed alongside key performance indicators for host, VNF and NS SLA.
- Catalogue Management – begin standardizing on the descriptor files necessary
- VIM agnostic: ability to swap between VIMs e.g. OpenStack/OpenVIM without losing functionality
- Fault, Configuration, Accounting, Performance, Security (FCAPS): A single pane of glass view for the NFV platform.

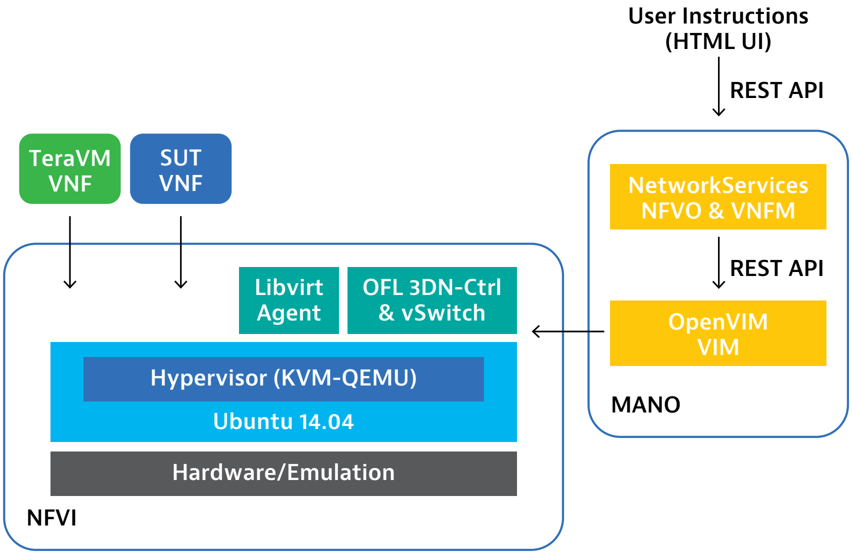


Figure 1: Abstracted NFV platform architecture

The architecture above demonstrates and proves the ability to mix both open-source and proprietary solutions in the MANO stack. The architecture design enables users to support multiple configuration options of VIM, for example OpenVIM or OpenStack whilst maintaining the integrity and functionality of the NFV flow process.

Advantages associated with OpenVIM

OpenVIM has been designed to enable ease of use, it links to open vswitch and Openflow controllers to deliver networking and management. The networking functionality is similar to that of the OpenStack Neutron module. But that's where the similarity ends.OpenStack Neutron can be complex to configure and operate.

An advantage of OpenVIM is the ability to quickly tune the VNF deployment for performance, tuning techniques such as non-uniform memory access (NUMA) can be defined as part of an information model scheme. In OpenStack this cannot be easily achieved, requiring multiple adoptions on the host and tenant setup e.g. (configure cpu allocation ratios, modify scheduler configurations, etc).

Frustratingly for carriers, they are inundated with vendors proposing solutions that support NFV, when all they merely offer is a software package and instruction set on how to run the software on a VM with prescribed additional package installs with defined needs. The whole purpose of NFV is to deliver to the carriers packaged solutions which can use information or objective models in order to create a NS.

The VNF package consists of the network functions machine images (in KVM known as QEMU Copy On Write or qcow) and descriptor files for deployment. The descriptor files are further segment into a generic VNF descriptor (VNFD) and VNF deployment unit descriptor (VDU). OpenVIM supports ingest of this metadata in its north-bound API. For solution vendors, this enables a viable mechanism for their development teams to learn about the VNF onboarding process, to create their first VNFD and VDU and validate the accuracy of the VNF deployment.

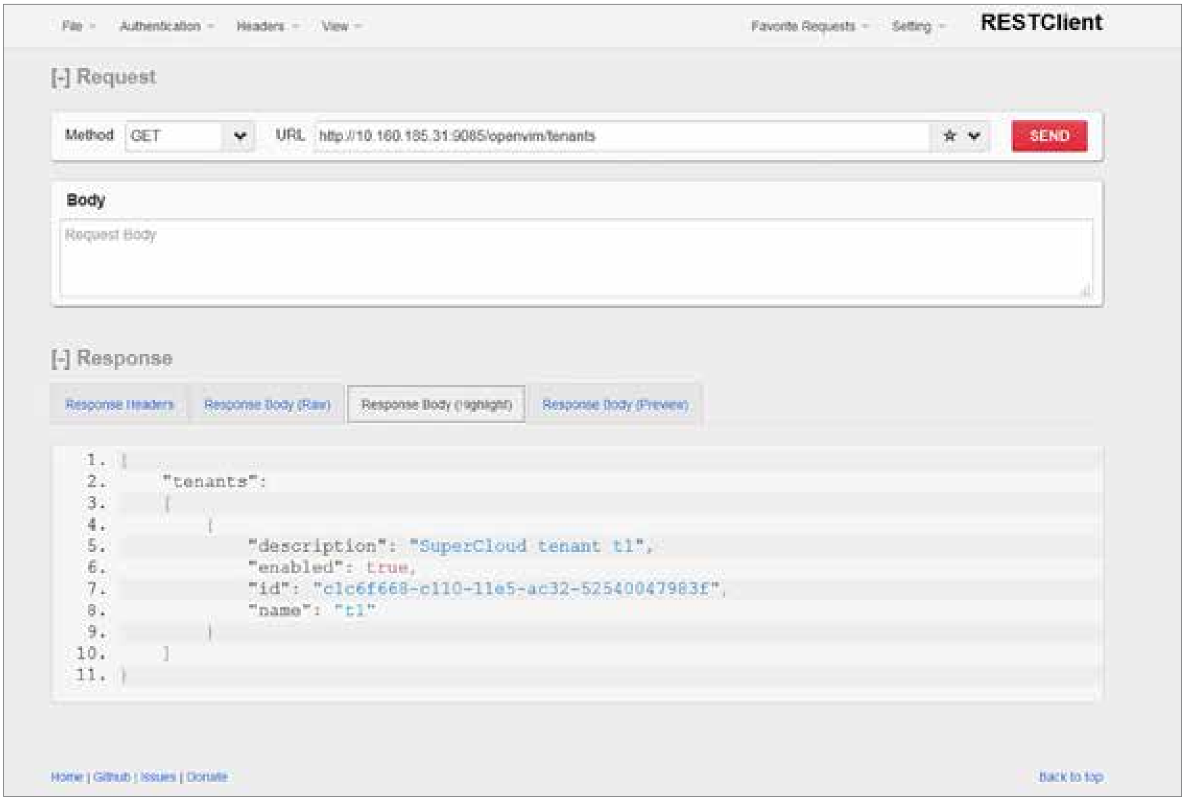
Another feature of the OpenVIM which is very relevant when it comes to delivering performance in the lab is the ability to deploy components to use pass through techniques (Direct-Path, SR-IOV, etc) ensuring the solution vendor can achieve maximum throughput in the physical networking.

The OpenMANO project is about simplicity and ease of deployment but by focusing on the key performance tuning attributes enables solution vendors to quickly assess multiple deployment options i.e. orchestrate VNF with/without NUMA.

Furthermore, for the development and operations team - OpenVIM's RESTFul API makes it ideal to run deployment exercises without the need of any sophisticated user interfaces.

An example onboarding exercise could be to use a browser e.g. Firefox with RESTClient (http://restclient.net/) plugin, in which users can quickly start interaction with the infrastructure manager, a full list of API calls is available on the OpenMANO repository. Ideal for debug scenarios.

As we touch on REST API, from a carrier perspective and as part of the recommendations in the ETSI ISG test documentation, there is the need to validate performance on these APIs. In the next chapter, we will discuss test methodology for validation and understanding the performance of the MANO stack and the associated REST API, looking at the associate latency of implementing a command.



Method: GET
URL: http://10.160.185.31:9085/openvim/tenants

```
{
    "tenants":
    [
        {
            "description": "SuperCloud tenant t1",
            "enabled": true,
            "id": "c1c6f668-c110-11e5-ac32-52540047983f",
            "name": "t1"
        }
    ]
}
```

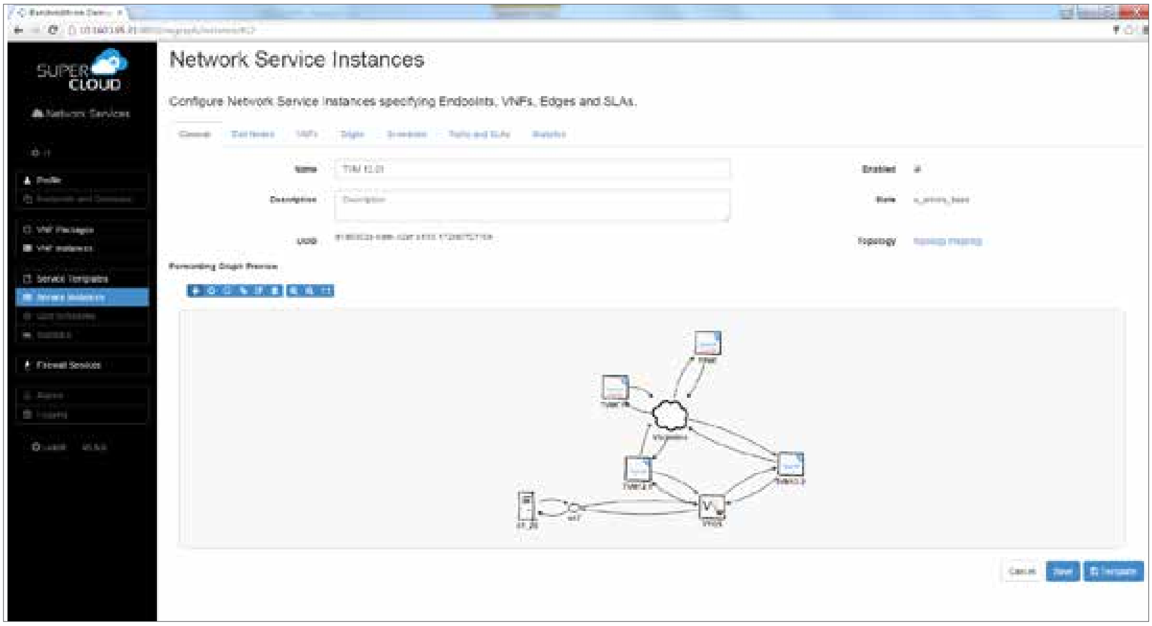
Figure 2: RESTClient a Mozilla Firefox plugin

SuperCloud NFVO/VNFM

SuperCloud provides an integrated solution for NFVO/VNFM layer, as the abstraction layer above the OpenVIM it provides an easy to use framework which enables deployment of network services.

In addition to enabling the core MANO aspects, a key advantage of the solution is to provide a single plane of glass view on the performance of the running NFV platform and on each of the unique network services running, with key performance indicators covering compute, networking, and tenant SLAs. SuperCloud delivers a number of key capabilities, which include:

- Administration functions (complete view of infrastructure and tenants)
- Tenant Management (incl. security, bandwidth allocation)
- Abstraction of network complexity (network topology and network creation)
- VNF Catalogue Management (complete view of available VNF packages)
- VNF onboarding/deployment (ingest VDU & VNF descriptors along with VM images)
- VNF Manager (lifecycle upgrade/update management)
- Performance monitoring (performance metrics covering – host, VNF and NS SLA)



Network Service Instances

Configure Network Service Instances specifying Endpoints, VNFs, Edges and SLAs.

Figure 3: SuperCloud UI – Topology implementation screen

MANO architecture validation

As discussed in the previous chapter, the use of an API based on Rest requires careful consideration. As each layer of the MANO stack has its own instruction set, meaning there is a translation layer between each which clearly consumes compute cycles.

For example – "What are the differences in performance between OpenStack and OpenVIM when it comes to NUMA assignment? Both have their own unique instruction sets in order to achieve the correct positioning of the VM.

As an observation, from a carrier perspective there is a need for a dynamic API which will sit on top of both VIM types, that can translate the same instruction e.g. schedule NUMA pinning. Clearly both have VIMs has an instruction set to follow in order to achieve this.

A first step in the validation methodology to determine performance of the API, will include:

- Instruction timing performance
- Ability to handle hundreds of concurrent instructions
- Responsiveness under duress

However key to delivering performance validation is the need for a feedback loop. One which is intelligible enough to translate the business logic being returned, as part of the HTTP responses - so as to deliver the next set of commands to send.

Again in terms of API selection, simply saying that it is RESTful is one thing, consideration must be given to the format of the information being communicated. Both OpenStack and OpenVIM have elected to use JavaScript Object Notation (JSON), enabling ease of presentation of parameters and values.

This means the test solution must be layer 7 aware, can parse data and send the next request. This is a stateful awareness is available in TeraVM.

Comparing VIM query structures

For carriers to validate performance of the MANO stack requires that the test and measurement solution must be able to handle numerous query types. Highlighted in the following example are the HTTP request constructs for OpenStack and OpenVIM

OpenStack uses security toked-ids, the token-id is created as part of an earlier HTTP GET request. This token-id is specific to a tenant username/password login post. The token is used to validate user requests, but also helps the VIM to deliver data specific to that tenant.

OpenVIM at the time of writing has no security token-id implementation, however the query structure uses a tenant-id, which is extracted from a separate HTTP GET request.

OpenStack

HTTP Header:
    X-Auth-Token = {security.token.id}
    Content-type: application/JSON
Command: GET
Link: http://<IP_address_OpenStack_Image_API>/**v2.0/images**

OpenVIM

HTTP Header:
    X-Auth-Token = {not implemented at time of writing}
    Content-type: application/JSON
Command: GET
Link: http://<IP_address_OpenVIM>/**openvim/<tenant_ID>/images**

Note the highlighted text in the requests urls for both VIMs, the query structure are different. Clearly this is where an agnostic VNFM adds value as it can translate "Show me images for my tenant".

It's also worth noting that the JSON response from both VIM are different. Again highlighting the difficult of truly adhering to an open framework.

Requirement for an application aware validation solution

Test and measurement must be agnostic to the framework, in that it should be capable of delivering an instruction set or multiple commands on the API irrespective of VIM type. However and more importantly it must be application aware. As highlighted above in the case of both VIMs, it is not as simple as a one query will get you the <parameter>:<value> pair you seek.

When choosing a solution to validate MANO ensure the test and measurement solution is application aware i.e. must be able to build multiple queries based on previous responses received. For example of request configuration:

OpenStack

- Enable security token-id via a HTTP POST, body containing user details:

```
"auth": {
     "tenantName": "demo",
     "passwordCredentials":{
     "username": "demo",
     "password": "DEMO_PASS"}
}
```

- Parse response to extract security token-id

```
"token": {
     "issued_at": "2016-01-26T11:54:54.906705",
     "expires": "2016-01-26T12:54:54Z",
     "id": "3afdd03757774071bd14b28046ba95e6",
     "tenant":
     {
        "description": "Demo Project",
        "enabled": true,
        "id": "3150b91202b94e0cb6e6b7300a38c834",
        "name": "demo"
     },
     "audit_ids":
     [
        "IksRwnKGTyOPCpWkFg0XZA"
     ]
}
```

- Use token-id to create the next requests, for example as seen in 3.1.

NOTE: A key advantage of TeraVM is it's per flow architecture which means that it can deliver the granularity necessary to facilitate validation with multiple request instruction sets, but more importantly offers the scale necessary to enable validation as if hundreds of tenants are active on the NFV framework.

VIM API performance validation methodology

The following is an example test methodology which is used in validating VIM responsiveness to an instruction set. The example uses a simple request to list available images per tenant. The instruction set requires a number of HTTP requests. As each HTTP request is unique carriers must assess each step to understand where potential compute cycles are intensive and slow.

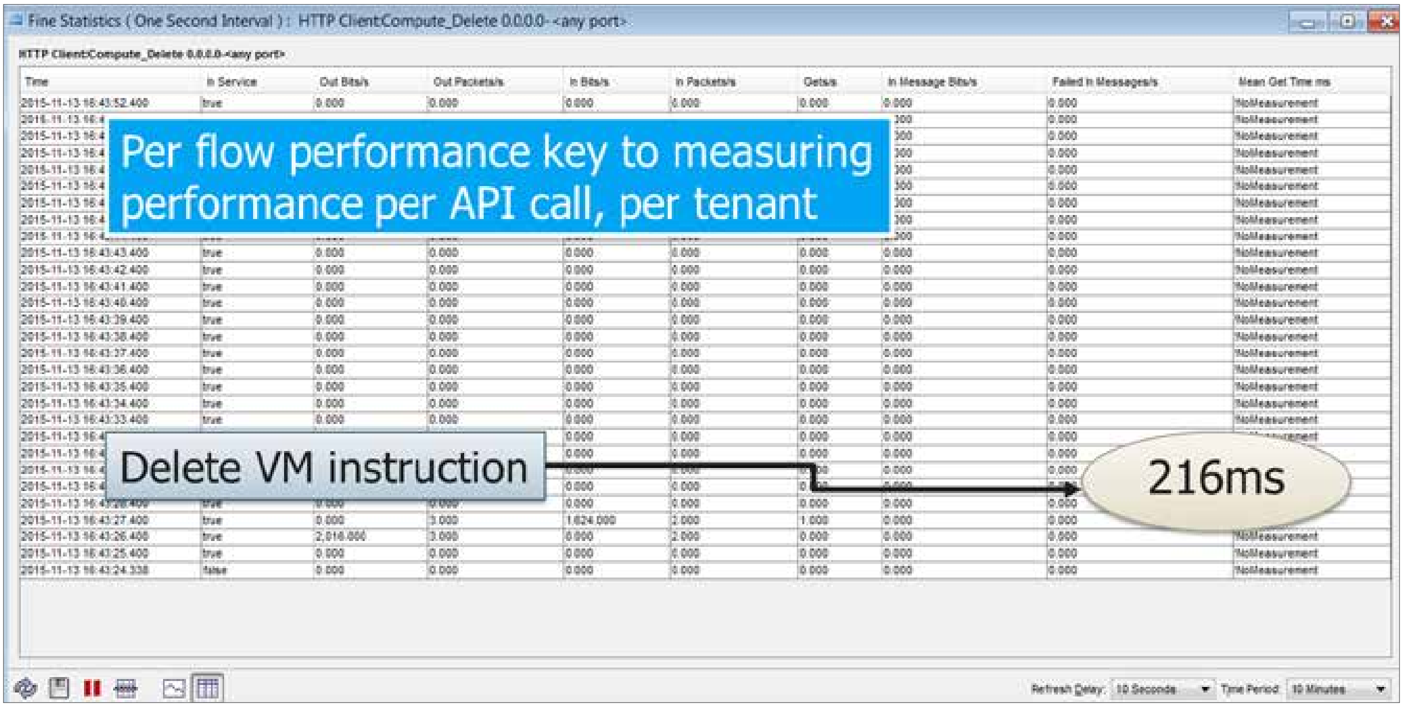
A similar methodology can be used for validating the NFVO/VNFM layers, emphasizing the need that validation requires performance measurements per request. A key advantage of using VIAVI Wireless TeraVM, is the unique ability to create multiple requests and to measure performance per request.

Uniquely to TeraVM is the ability to parse responses from the incoming JSON file enabling a series of stateful requests, enabling carriers/solution vendors to quickly pin-point bottlenecks.

The following is a sample test methodology which can be used in MANO for performance validation.

- Connect to each REST API: e.g. Identity, Compute, Image
  - Maximize memory and CPU usage
- Assess timing between different VIM API layers
  - Assess latency for Identity/Compute/Image
- Security – service multiple tenant logins/UUIDs
  - Access/Authentication flood of UUID database
- VM launch – how many VMs can the VIM instantiate
  - Concurrent instructions vs1 after the other
- 5. Host management – will the VIM scale quickly
  - Extend NFVI will the VIM assign new VMs
- 6. Robustness – hundreds of tenant instructions
  - Identify memory leaks, caching issues

Below is sample performance validation based on the command for deletion, in this simple case a single HTTP DELETE request is presented on the API, the results from the server is that it takes up to 200ms for the server to respond as instruction completed (HTTP code: 200).



Fine Statistics (One Second Interval) : HTTP Client:Compute_Delete 0.0.0.0-<any port>

HTTP Client:Compute_Delete 0.0.0.0-<any port>

Time	In Service	Out Bits/s	Out Packets/s	In Bits/s	In Packets/s	Gets/s	In Message Bits/s	Failed In Messages/s	Mean Get Time ms
2015-11-13 16:43:52.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2016-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:4							000	0.000	NoMeasurement
2015-11-13 16:43:43.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:42.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:41.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:40.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:39.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:38.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:37.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:36.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:35.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:34.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:33.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:4				0.000	0.000	0.000	0.000	0.000	urement
2015-11-13 16:4				0.000	0.000	0.000	0.000	0.000	
2015-11-13 16:4							0.000	0.000	
2015-11-13 16:4				0.000	0.000				
2015-11-13 16:4	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	
2015-11-13 16:43:27.400	true	0.000	3.000	1,624.000	2.000	1.000	0.000	0.000	
2015-11-13 16:43:26.400	true	2,016.000	3.000	0.000	2.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:25.400	true	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement
2015-11-13 16:43:24.338	false	0.000	0.000	0.000	0.000	0.000	0.000	0.000	NoMeasurement

Refresh Delay: 10 Seconds ▾ Time Period: 10 Minutes ▾

Per flow performance key to measuring performance per API call, per tenant

Delete VM instruction

216ms

Figure 4: Performance validation of a delete instruction in OpenStack

Delivering NFV Network Services

This section is aimed at understanding the NFV process to enable the network service and more importantly how it drives a pre-production network deployment validation methodology.

As part of the understanding of how to deliver a comprehensive validation methodology, this section also provides some recommendations on requirements in creating a NFV validation lab: enabling efficient management and testing of available VNFs for solution vendors.

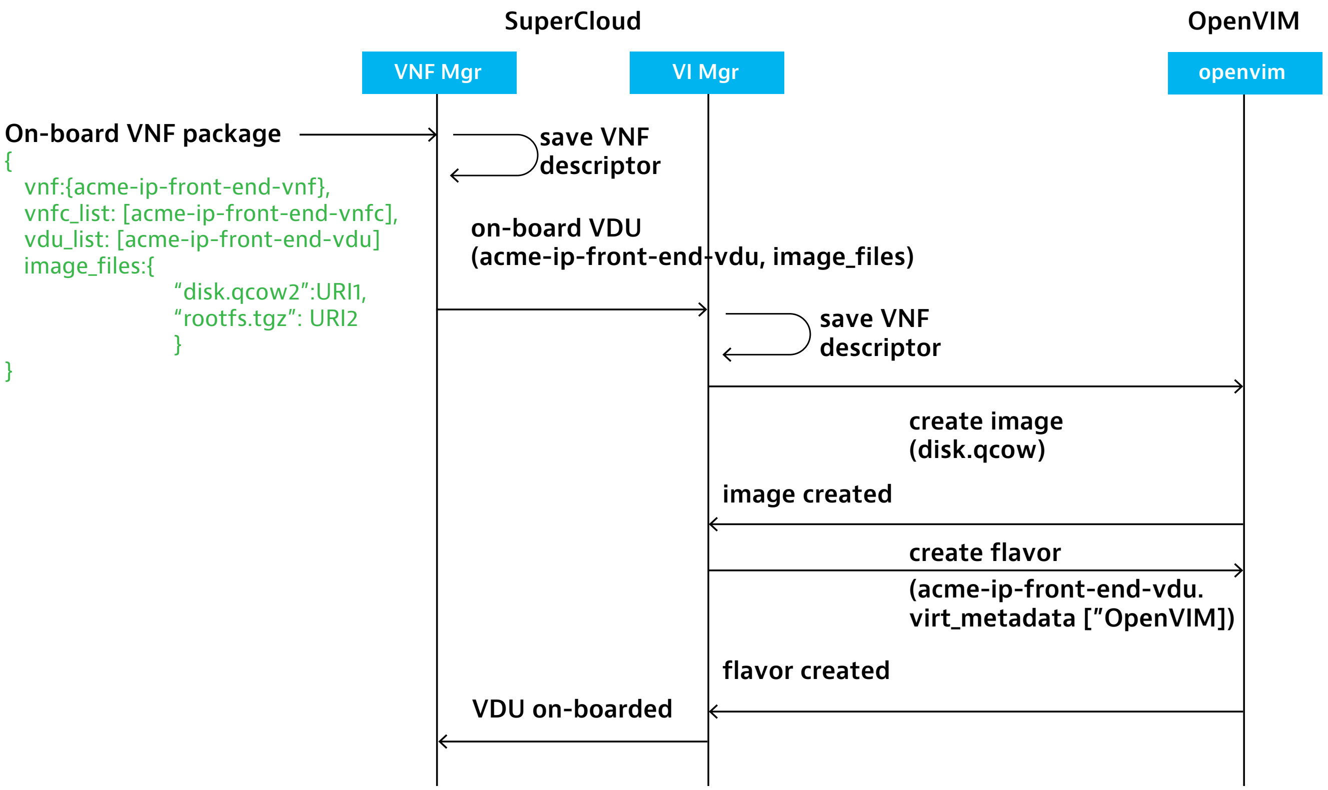
Onboarding validation

Onboarding is the process of taking the external VNFs info the NFV framework and managing them in a catalogue. The key for all VNFs - is to ensure that the package to be onboarded includes the VNF images and the necessary descriptor files, which are to be ingested by the orchestrator and used by the VIM for the successful deployment of the VMs to the compute node and networking required for an operational VNF.

An example VNF onboarding scenario could be traffic as a service – TeraVM. When implementing the information model or descriptor metadata for TeraVM a number of observations were made, which are worth sharing here for other VNF solution vendors.

The descriptor files enable flexibility, for example a VM image (with the correct drivers) can be consumed in multiple ways. That is one descriptor file may define connectivity to a vswitch and the other could be used to define a choice of pass through technology. For TeraVM this provides for ease of deployment to test inside tenant and across hosts or multi-domain tenancies. Unique descriptors can also enable provisioning of different NUMA requirements.

The clever piece for the VNF vendor is that it references the same VM image, in the VNF package repository. From a carriers perspective this offers a number of efficiencies, most obvious being storage. In terms of TeraVM, it means we support both deployment options which provides for greater test coverage.



```
{
  vnf:{acme-ip-front-end-vnf},
  vnfc_list: [acme-ip-front-end-vnfc],
  vdu_list: [acme-ip-front-end-vdu]
  image_files:{
                    "disk.qcow2":URI1,
                    "rootfs.tgz": URI2
                    }
}
```

Figure 5: SuperCloud onboarding flow chart

Another important part of the VDU and VNFD is the enablement for the VNF lifecycle management process, over the life of the service a number of updates/upgrades will be made on individual running virtual machines. A flexible and well maintained VNF package repository will contribute to the success of the NS, this is one of the core featured of a good NFVO solution such as SuperCloud.

From early on in the NFV process cycle, solution vendors must understand how the VDU descriptors impact the performance of their solutions, if it is badly thought out or described it can seriously hamper the carrier experience. Clearly as a first impression solution vendors should provide VM images and descriptor files that match the carrier onboarding process, the importance of which is often under-stated.

From an automation perspective for a network service or facility for performance testing, the descriptor files provide other relevant information which can be used for accurate cataloging of the VNFs available in the NFV platform. Versioning and licensing details are also key parameters included in these files. Knowing your VNFD and VDU is key, simply being if it doesn't work, the carrier can't use the VNF.

Example information modeling of TeraVM

The following is a snaphost of the TeraVM descriptor file, which is delivered in JSON format. JSON provides a lightweight parameter:value pairing with minimal text ensuring ease of creation and consumption of the information needed for a VNF deployment.

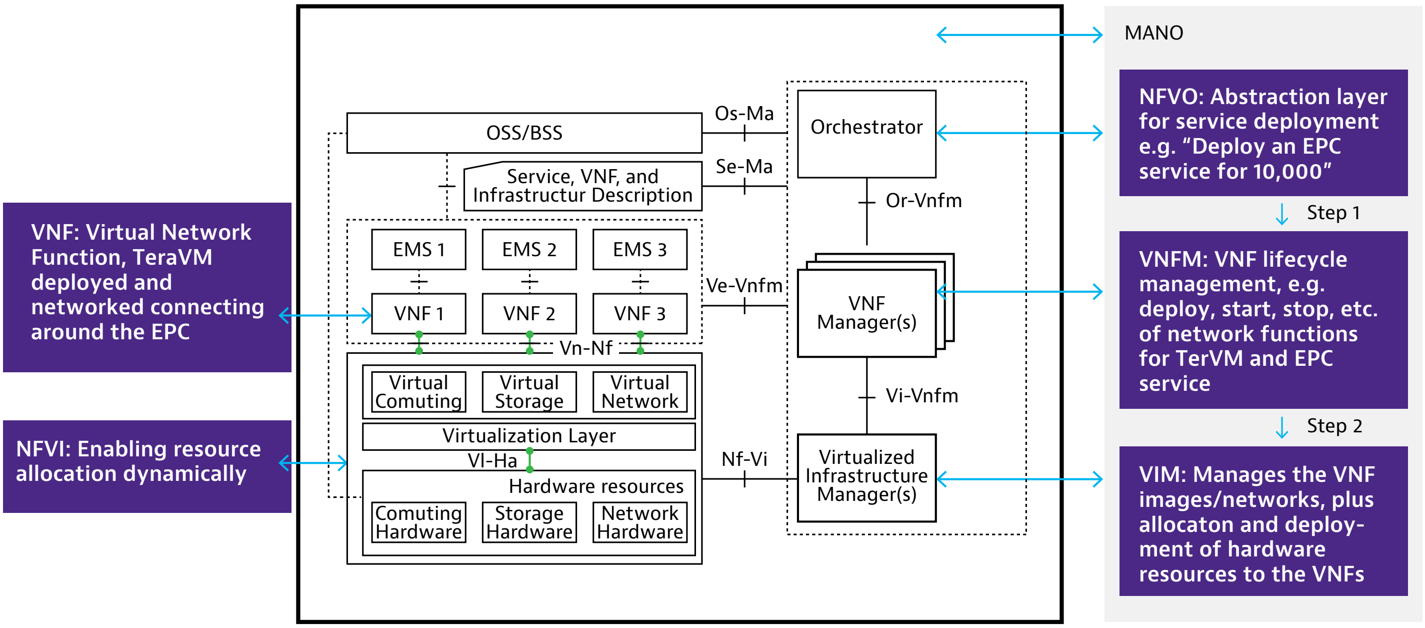
```
######
# Example VNF descriptor file in JSON format used as part of the onboarding process for #
the TeraVM traffic
as a service VNF.
######

{
“id”:”VIAVI_TeraVM_Executive-1.0-353”,
  “version”:”12.01_353”,
  “service_type”:”service_verification_executive”,
  “service_instance_type”:”tvm-e”,
  “license”:”system_specific”,
  “name”:”VIAVI TVM-E VNF”,
  “description”:”VIAVI TeraVM executive VNF”,
  “author”:”VIAVI Wireless”,
  “shared”:true,
  “tenant_id”:”t1”,
  “vnfc_list”:{
    “1”:{
      “name”:”TVM-E”,
    “author”: “VIAVI Wireless”,
    “version”: “12.01-353”,
    “license”: “licence-server”,
      “description”:”TeraVM executive VNF”,
      “min_instances”:1,
      “max_instances”:1,
      “vdu_list”:{
        “TVM-E_12.01-353”:null
      },
      “intf_list”:{
        “1”:{
        “name”:”Comm”,
        “description”:”TVM internal communication interface”,
        “vdu_intf_id”:{
          “TVM-E_12.01-353”:1
        },
        “access”:”bidirectional”,
        “intf_class”:”external”,
        “min_instances”:1,
        “max_instances”:1
      }
    }
  }
}
```

NOTE: TeraVM is a fully virtualized solution which is packaged for numerous hypervisors which include ESXi, KVM and Xen. Enabling carrier and solution vendors validate performance with consistency and ease across multiple platforms. In addition TeraVM supports public cloud platforms of AWS and Azure, with support for private cloud such as OpenStack.

VNF Lifecycle management and orchestration validation

As already highlighted an advantage of keeping accurate VNF catalogue control, is that it ensures that the process for upgrade/update can be implemented without complexity. As vendors release new patches, it will be necessary to run the update/upgrade but more importantly to easily snapshot the running VM and cataloging it as a new/unique VNF version in the repository.

In selecting the correct NFVO/VNFM pair . For example,. SuperCloud, there should be the functionality to enable a roll back service, ensuring prior to any update/upgrade, the deployed environment can be restored to an operational state.



VNF: Virtual Network Function, TeraVM deployed and networked connecting around the EPC

NFVI: Enabling resource allocation dynamically

NFVO: Abstraction layer for service deployment e.g. "Deploy an EPC service for 10,000"

VNFM: VNF lifecycle management, e.g. deploy, start, stop, etc. of network functions for TerVM and EPC service

VIM: Manages the VNF images/networks, plus allocaton and deployment of hardware resources to the VNFs

Figure 6: SuperCloud onboarding flow chart

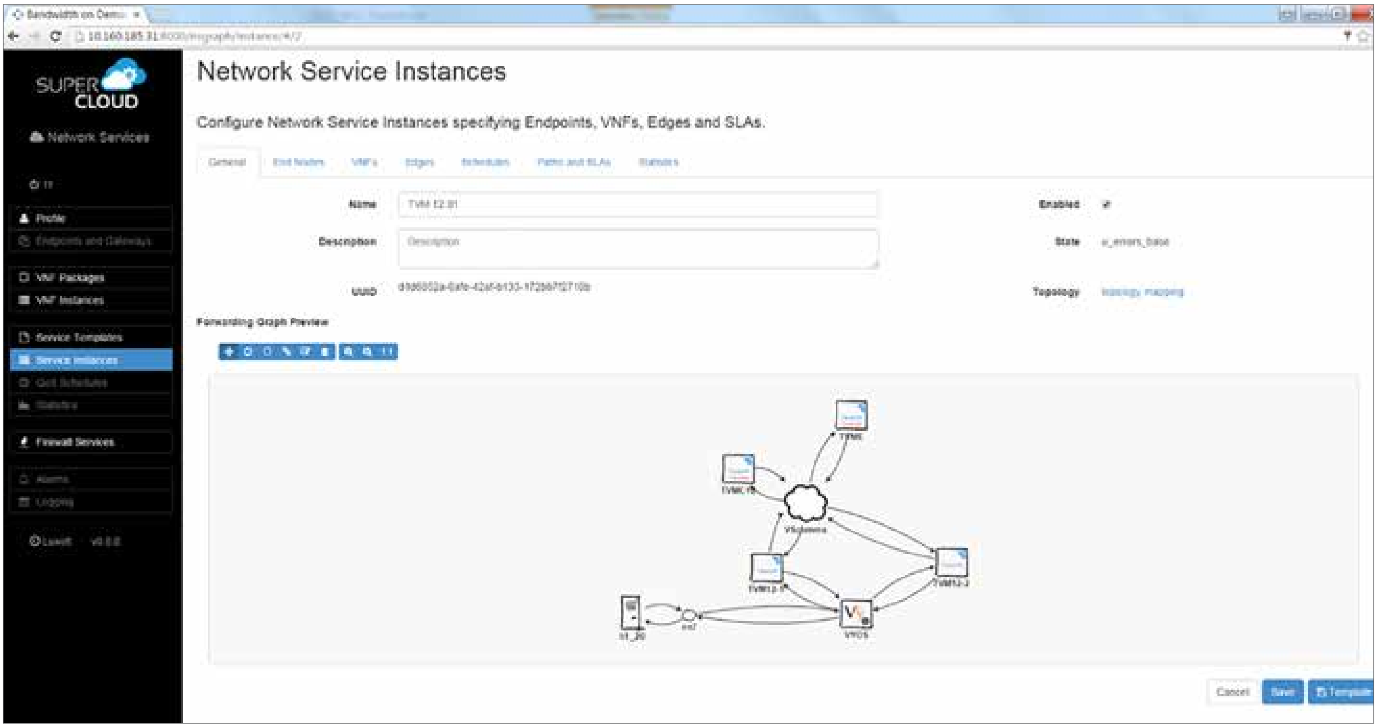
Network service topology creation

Once the new network service or lab deployment requirements have been pieced together as per the example in Figure 6, the next step is to transfer the business plan into a working topology. At this point there is a translation of the commercial requirement to a technical topology.

A number of tools are available today that implement to some degree this functionality, this is where there is some similarity to traditional automation tools in the lab i.e. a wizard enabling translation of commercial requirements to a deployable topology.

But this is where the similarity ends as the orchestrator must now present to the carrier user what the available VNFs are and enable networking deployment options. SuperCloud NFVO provides a mature interface enabling ease of use in translating business needs to technical deployments, which provides a very precise topology as quickly as possible.

The SuperCloud solution means the users do not necessarily need to be networking experts/architects



Network Service Instances

Configure Network Service Instances specifying Endpoints, VNFs, Edges and SLAs.

Name: TVM 12.81

Description: Description

UUID: d9d6002a-0afe-42af-b133-172bb7f2710b

Enabled: ✓

State: ai_errors_base

Topology: topology mapping

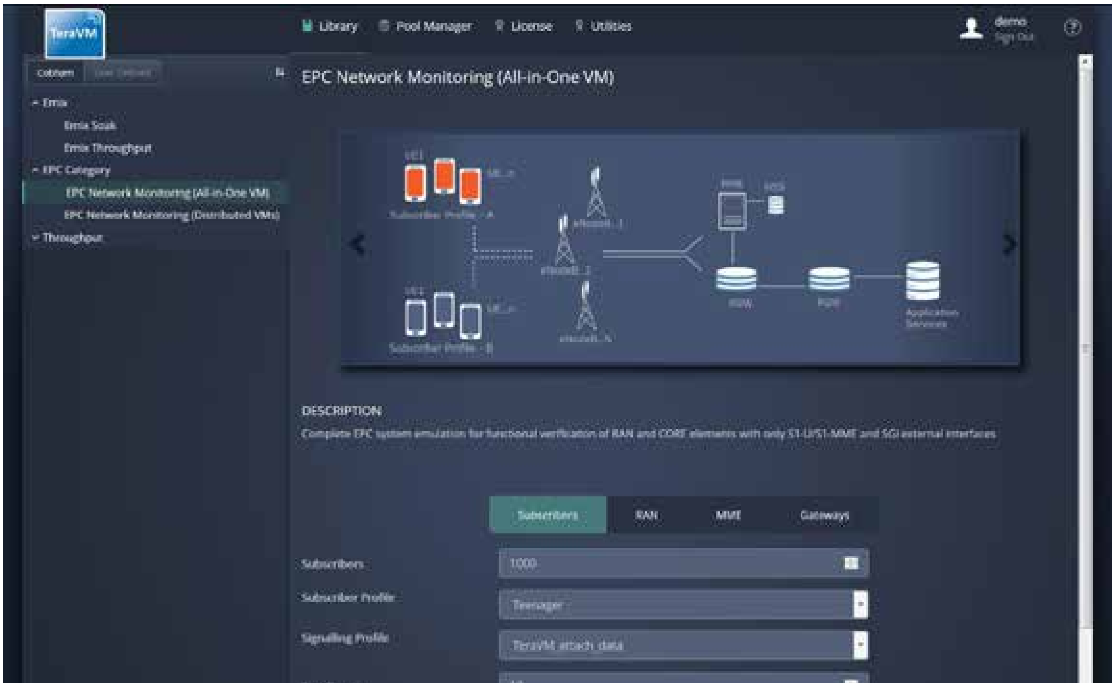
Forwarding Graph Preview

Figure 7: Supercloud convert business plans into deployable network services

TeraVM enabling performance validation in NFV

To this point we have discussed the features of NFV and its clear from the open-source we can deliver a managed deployment via a business logic UI.

Assuming the network service as operational, the next step is to begin to validate performance of the service level agreement or contract. Clearly from an automation perspective, it would be weak to expect that the carrier/VNF solution vendor would need to go through a manual process to enable validation. This is where a template driven test solution such as TeraVM adds real benefit.



EPC Network Monitoring (All-in-One VM)

DESCRIPTION
Complete EPC system emulation for functional verification of RAN and CORE elements with only S1-U/S1-MME and SGi external interfaces

Subscribers: 1000

Subscriber Profile: Teenager

Signalling Profile: TeraVM_attach_data

Figure 8: TeraVM simplifying validation through centralized and templated use cases

Highlighted in figure 8 is the TeraVM centralized library approach for validation, this provides carriers/solution vendors with an integrated approach to validate the widest range of network service use cases e.g. security, video delivery networks, vEPC, etc.

The advantage of the centralized library is the ability to deliver automated validation, this further ensures repeatable and reliability in assessing performance. Furthermore by centralizing golden use cases as templates minimizes the need for manual configuration. Critical for repeatable testing across multi-domain tenancies.

Consider a highly specialized test methodology for a NS enabling a mobile virtual network operator (MVNO), the requirement is such that the carrier can lock in place the core parameter configuration for the radio access networks, gateways, management entity and continually thereafter be able to validate key performance indicators. In this instance the test solution itself is ephemeral, turned up for a duration in the tenancy and removed once KPIs are gathered.

> *NOTE: TeraVM provides performance validation of key application services of voice, video and data. TeraVM offers validation of core mobile networks (vEPC), security validation of both access technologies (VPN) and security appliances. TeraVM's comprehensive cybersecurity threat library is used in security hardening of major network services.*

VNF validation from lab to production networks

The telecoms industry is predicting for exponential growth which will come from the likes of the Internet of Things and the insatiable demand for video. This multiplexed with the NFV principles of mobility and portability means that VNF solution vendors are facing a tough challenge ahead, most notably is to prove scale, where core networks will need to handle terabits of data.

Scaling is a key test case that both carriers and solution vendors need to perform, however this produces a number set of touch points that warrant validation:

- Robustness of the MANO stack to manage racks of compute, NS under real load
- Performance monitoring across multi-host/tenant domains
- Security assessment (keeping the bad guys out)

This further compounds the needs for truly virtualized test solutions, such as TeraVM over open-source or indeed proprietary hardware based testing. Both for carriers and solution vendors delivering a test bed to cover all may appear daunting, however by choosing the right partnership it's possible to deliver validation cost effectively.

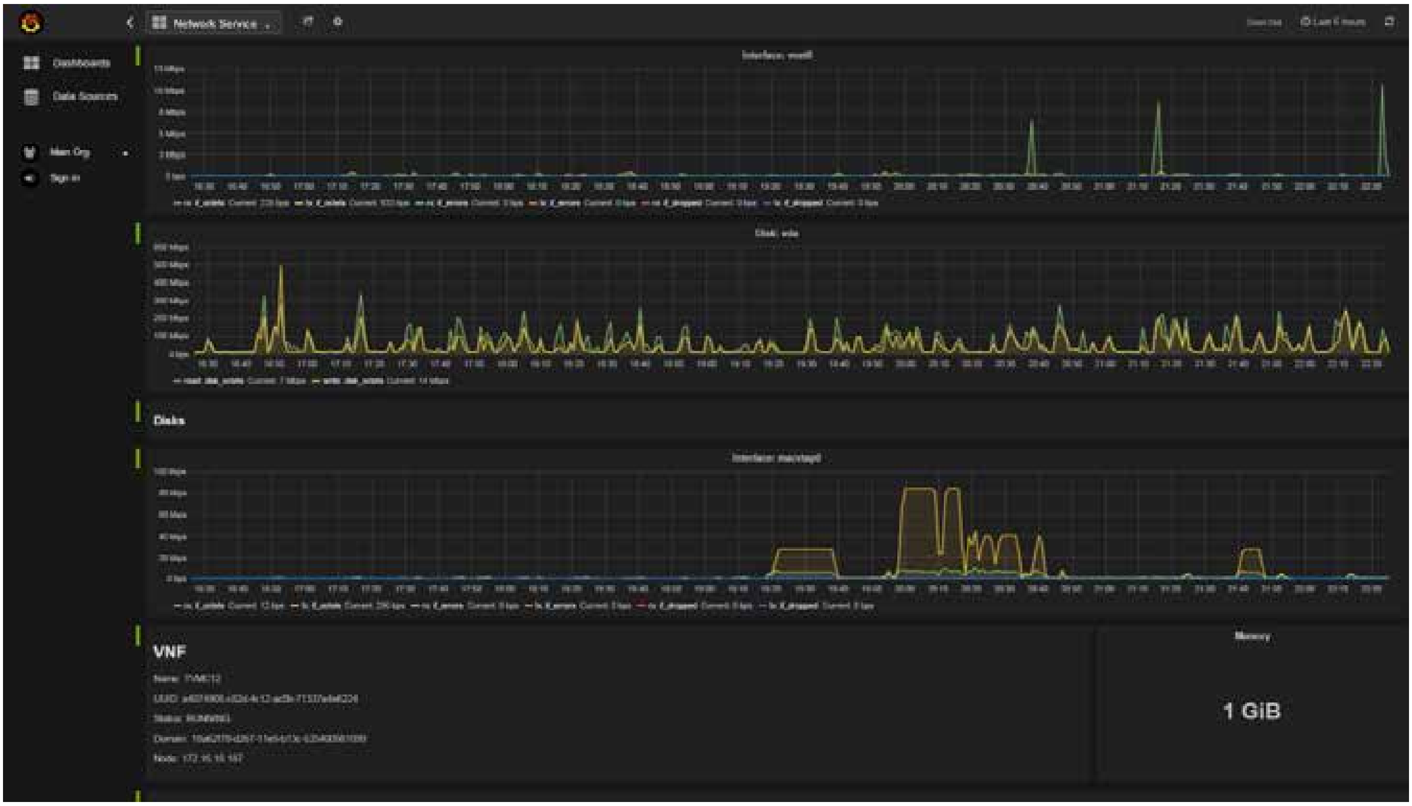
TeraVM enables an elastic test bed on standard hardware, one of the advantages is that once the initial validation at scale is completed, the underlying infrastructure can be released back into compute pool and the TeraVM used in smaller test beds for dev-ops or indeed for the carrier could be repositioned as a part of a service operation for customer validation e.g. MVNO offering.

NFV Network Service Validation and performance (FCAPS)

A comprehensive NFV test methodology must encompass how the end-to-end network service is validated, but more importantly continues to operate in real-time. Ideally this is achieved with minimal intervention from the operator.

Using TeraVM it is possible to deliver a range of test use cases, enabling the carrier/solution vendor assess performance consistently across different service types, also known as Traffic as a Service (TaaS). Once the NS is operational, it's possible through the SuperCloud UI to see performance on a per tenant basis.

The advantage of the combined TeraVM and SuperCloud solution is the ability to clearly see a number of KPIs at a number of levels on the NFVI, VNF and indeed the MANO stack. The combined solution provides for fault notification, enabling carriers/solution vendors to enable event logging when things are not right in the NFV framework.



VNF

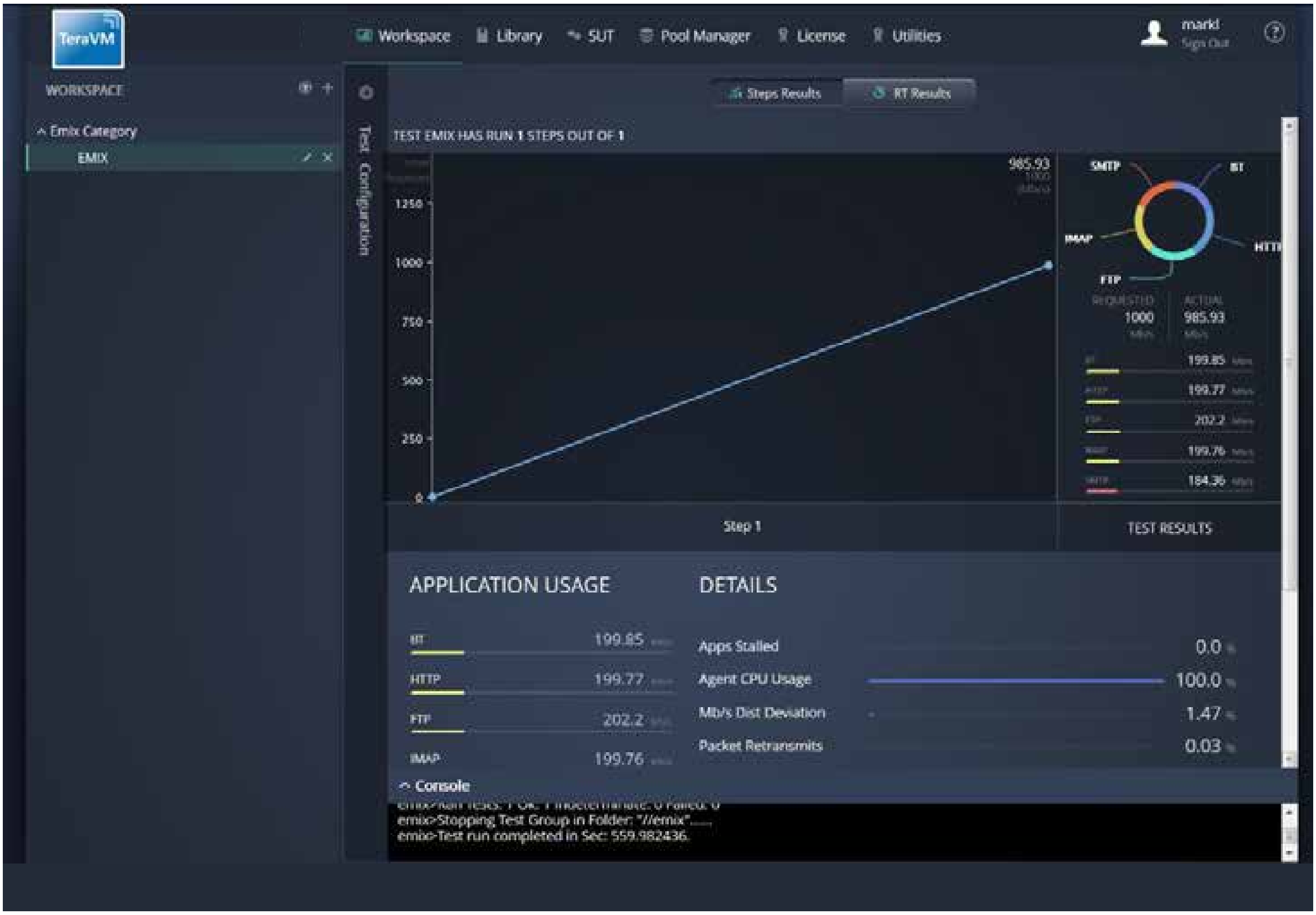
Name: TVMC12

UUID: [illegible]

Status: RUNNING

Domain: [illegible]

Node: 172.16.16.167



TEST EMIX HAS RUN 1 STEPS OUT OF 1

REQUESTED 1000 Mb/s

ACTUAL 985.93 Mb/s

BT	199.85 Mb/s
HTTP	199.77 Mb/s
FTP	202.2 Mb/s
IMAP	199.76 Mb/s
SMTP	184.36 Mb/s

APPLICATION USAGE

BT	199.85
HTTP	199.77
FTP	202.2
IMAP	199.76

DETAILS

Apps Stalled	0.0 %
Agent CPU Usage	100.0 %
Mb/s Dist Deviation	1.47 %
Packet Retransmits	0.03 %

```
emix>Ran Tests: 1 OK: 1 Indeterminate: 0 Failed: 0
emix>Stopping Test Group in Folder: "//emix"......
emix>Test run completed in Sec: 559.982436.
```

Figure 10: TeraVM business login enabled SLA performance measurement

Conclusion

NFV enabling lab automation

The premise for NFV is to enable carriers with agility and flexibility. However, taking these same principles it enables solution vendors with a very efficient means to enable test automation in labs.

The core principle of openness for management and orchestration has resulted in numerous open-source communities enabling technology to achieve automation, offering cost effective alternatives to proprietary solutions.

This enables a solution to be delivered in-house which will replicate the carrier environment. By replicating the carrier environment and using the same information modeling and automation processes, the solution vendors can ensure that there is no nasty surprises throughout the lifecycle of the virtual network function.

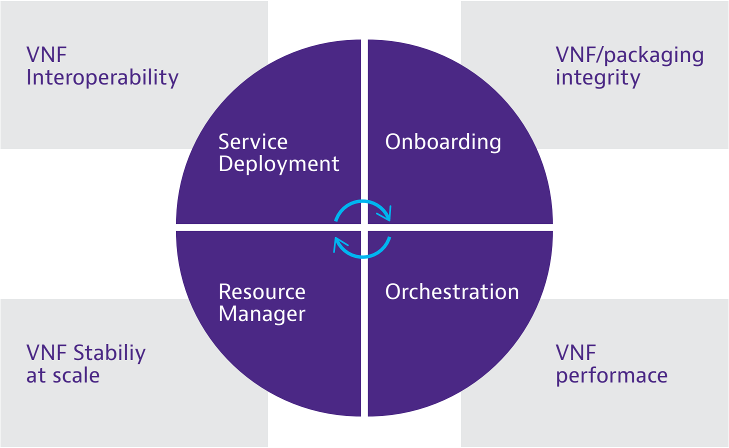


Figure 11: NFV supports VNF lifecycle validation

Delivering performance and secure NFV frameworks

Choosing the right NFV framework is critical for carriers, one which is reliable, robust and performing. Today there are many variants of frameworks available, but for the network solution vendor it means they need to be agile and flexible in the delivery of their solutions. A key strategy for many vendors is to support multiple hypervisors or deliver containerized solutions, along with enabling a flexible molding to support information or objective modeling of the network product.

The dynamic and changing landscape of NFV is resulting in new challenges, key being the ability to prove that virtual network functions are ready for live deployment, compatible with a multitude of carrier NFV frameworks and support the VNF lifecycle – onboarding, deployment and reliability at the necessary scale.

With so many unknowns on the plate, the aim for all should be for a consistent test and measurement approach using known and proven solutions. This means investment in the right solution, such as TeraVM. Of course, the open-source domain produces solutions for network validation. Use of such solutions adds yet another unknown into the mix, heightening the risk for failure.

As highlighted, NFV validation is multi-dimensional i.e. from information modeling process validation, to validation of management APIs and assuring the reliability/robustness of a tenant architecture or Network Service. More importantly once the NFV framework is live there is the need to assure that the cyber-criminals are kept off the NFV platform by running regular security checks and screening.

By selecting the right validation partner from as early as possible in the NFV project, assures that the widest range of validation scenarios are covered, ensuring reliability over the life of the tenant service and most importantly tenant security.

By selecting the right validation partner today, maximizes the overall return on investment in validation for many years to come.

VIAVI Solutions

To reach the VIAVI office nearest you,
visit viavisolutions.com/contact

nfvenabling-wp-wir-nse-ae
30187525 900 1118