Virtual Network Functions Deployment between Business Expectations and Technical Challenges: The T-NOVA Approach

Y. Rebahi/M. S. Ghamsi
Fraunhofer Fokus
Berlin, Germany
{yacine.rebahi;
majid.salehi.ghamsi}@fokus.fraunhofer.de

N. Herbaut, D. Negru
Viotech Communication SARL
Versaille, France
{nherbaut, dnegru}@viotech.net

P. M. Comi, P. S. Crosta
Italtel
Settimo Milanese, Italy
{paolomaria.comi, paolosecondo.crosta}@italtel.com

P. Lorenz
University of Haute Alsace
IUT - 34 rue du Grillenbreit
68008 Colmar - France
lorenz@ieee.org

E. Pallis, E. Markakis
Informatics Engineering
Technological Educational Institute of Crete
Crete, Greece
{Pallis, Markakis}@pasiphae.teicrete.gr

***Abstract* — This paper belongs to the series of research documents describing the progress in the specification and development of the T-NOVA framework offering a marketplace for virtual network functions. T-NOVA is an international research project co-funded by the European Commission. Although, the idea of having a marketplace enabling buying, composing, and deploying "virtual" services on the fly is promising, its implementation or prototyping remains far from realization. This is mainly due to the limitations in the existing cloud computing platforms on top of which the services should be built. In this paper, we discuss the T-NOVA approach and in particular some of the Virtual Network Functions (VNFs) that have been developed in this context. Special attention is paid to the design and specification of the VNFs as well as the related technical challenges that were faced when deployed within the marketplace. Some experiments and test results are also provided.**

Keywords -- NFV; T-NOVA; Marketplace; Security; OpenStack; Content Delivery

I. INTRODUCTION

The main goal of Network Function Virtualization (NFV) is to offer network functions onto industry standard high volume servers, switches and storage systems [1]. The use of NFV will enable telecom operators and service providers to go beyond the limitations dictated by hardware based appliances, and meet their objectives in terms of revenue increase and growth plan.

In [1], a sample of Network Functions that can be virtualized was given. This includes, IPSec, VPN Gateway, DPI, AAA servers, and SBCs. In fact, any data plane packet processing and control plane function in mobile and fixed networks can be virtualized [1].

It is worth mentioning that in spite of the progress we have seen during the last years, especially in the context of standardizing NFVs, the related technology is still not mature enough to be deployed. What is also missing is a business model where various stakeholders collaborate in order to offer Virtual Network Functions (NFVs) and build added value services on top of that. In fact, this is the idea behind the T-NOVA project [2] where a marketplace for purchasing VNFs and deploying them on the fly has been developed. In the current paper, we will discuss the T-NOVA approach, in particular the progress in the specification and the development of the related framework.

This paper is organized as follows: Section II overviews the T-NOVA architecture in which the marketplace and the VNFs are being specified and developed. Section III provides a short description of a sample of VNFs that have been developed as well as potential integration scenarios. In section IV, we discuss some experiments for validating the VNFs under consideration, and section V is the conclusion.

II. THE T-NOVA ARCHITECTURE

Taking into account the required functionalities of the T-NOVA project and building upon the ETSI NFV standard, we present a four-layered architecture. The figure below delineates the design.

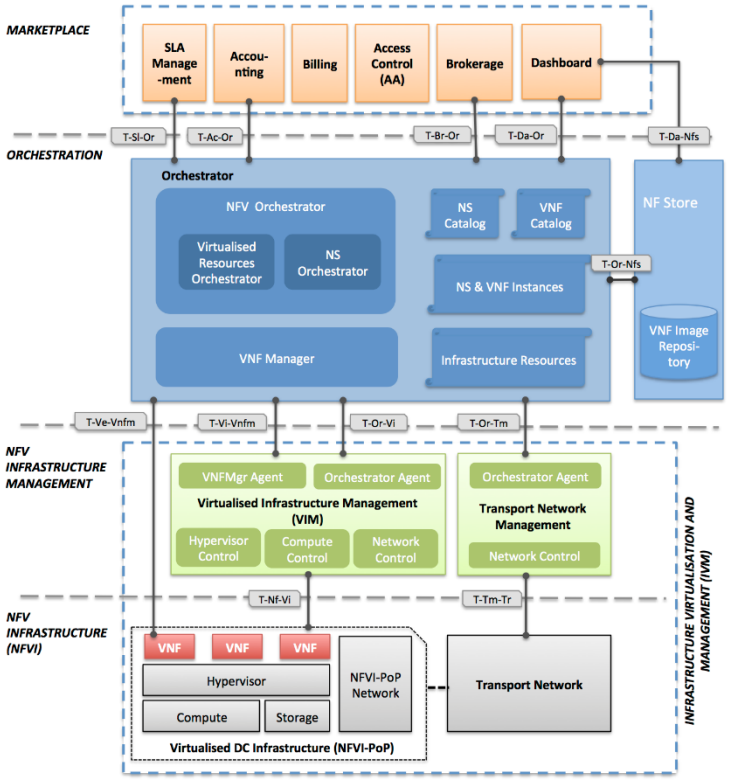

Figure 1. T-NOVA high-level architecture approach

The T-NOVA architecture can be progressively structured into four design layers each of which is populated by an arrangement of useful parts. (i) The NFV Infrastructure (NFVi) layer incorporates the physical and virtual machines (i.e. item servers, virtual machines, stockpiling frameworks, or switches) where all are conveyed to a common infrastructure; (ii) the NFVi Management layer understands the diverse administration segments to be more specific, the Virtualized Infrastructure Management (VIM) and the Transport Network Management (TNM). Both segments might be alluded to as Infrastructure Virtualisation and Management (IVM); (iii) the Orchestration layer depends on the Orchestrator segment, and it additionally incorporates the NF Store; lastly (iv) the Marketplace layer contains all the client confronting modules, which encourage multi-inhabitant contribution and execute business-related functionalities.

A. The Marketplace

The benefits of NFV compared to dedicated infrastructure are multifarious: cost-efficient, time-to-business sector decrease, adaptable and so on. However, it also brings various difficulties that must be settled to empower its huge reception in the business sector.

Specifically, the key point can be the advancement of development by opening a part of the systems administration

and changing it to a novel virtual apparatus market, encouraging the association of programming participants, including SMEs and even academia. Besides the fast presentation of novel system capacities (counting redesigning of existing ones) at a much lower cost and reduced hazards, prompting huge abatement of Time-To-Market (TTM) for new arrangements remain key issues in the NFV Market.

Keeping in mind the end goal to encourage the association of assorted performing artists in the NFV scene, a creative "Virtual Network Function Marketplace" that can take after the worldview of officially effective OS-particular "Application Stores" is required. The VNF Marketplace, which can be kept up by a Service supplier, can contain VNFs made and provided by a few outsider engineers, distributed as autonomous elements and accompanied with the fundamental metadata (counting exchanging data as a major aspect of the Virtual Network Function Descriptor). The Marketplace will permit clients to choose the virtual machines which best match their requirements, "plug" them into their current network benefits and arrange/adjust them as indicated by their necessities.

So, as to encourage competition and backing diverse value chain setups, a Brokerage Platform can likewise be set up, permitting clients to execute with the Service Provider and different outsider Function Developers for selecting the best Service that suits their requirements. After getting what the Service Provider asks for, the Brokering module will look at i) the accessible Network and IT assets and ii) the accessible capacities at the Function Store and concoct particular financial/specialized offerings and related charging models.

Via the Marketplace and the Brokerage platform, we promote a novel Marketplace for NFV, introducing new business cases and considerably expanding market opportunities by attracting new entrants into the networking market.

New architectural elements that will allow the validation and verification of the admitted VNFs will be required in order to streamline and accelerate the adoption of new VNFs in the Marketplace.

Contrasted with the ETSI reference architecture below and as indicated by Figure 1, the commercial center is totally novel concerning ETSI. This proposition presents the commercial center idea going for opening the NFV business sector to outsider designers for the procurement of VNFs, an idea that right now falls outside the specialized perspective of ETSI NFV.

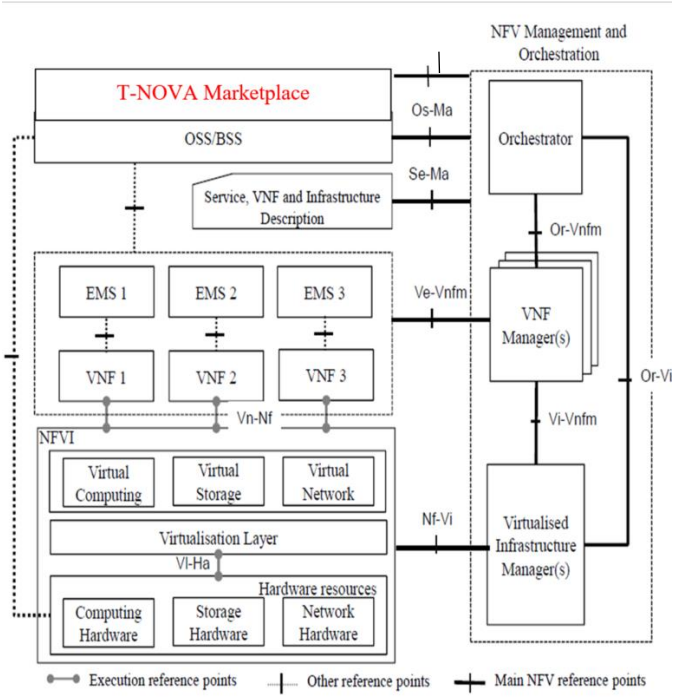

Figure 2. ETSI NFV architecture with the proposed marketplace

B. VNFs between T-NOVA and ETSI

As the Virtual Network Function Descriptor (VNFD) was introduced by ETSI, the T-NOVA project has used this descriptor that contains a variety of information (e.g. CPU, Memory, Virtual Deployment Units etc.) and extended it in order to take into account new requirements, such as (a) Node definition, (b) Link connectivity definition, and (c) the Service Level Agreement (SLA) based on the VNF performance. The above requirements were introduced in new fields inside the ETSI VNFD [21] descriptor and in a second stage uploaded to a component of the T-NOVA architecture that is able to understand and react to the new parameters.

In order to consider the relationship between the Service Provider (SP) and the Function Provider (FP), a Network Service Descriptor (NSD) was defined, compliant to the ETSI NSD, and stretched out by the T-NOVA project keeping in mind the end goal to address the confirmation parameters for the purchaser of the Service. These affirmation parameters are characterized by the available kind of administration and are utilized as a part of requesting the orchestrator to store them inside and together with the VNF descriptor, put away in the Function store to enable the TeNOR orchestrator [22] for a choice on the Service Mapping Problem [23].

The list of services is then presented inside the T-NOVA Dashboard where the info provided inside the VNFD is used in order to create the said list.

Identify applicable sponsor/s here. If no sponsors, delete this text box

The descriptor provides information about the available Connection Points, the physical resources, etc. achieving in this way (see Figure *3*) a Visual representation of the described VNFD.

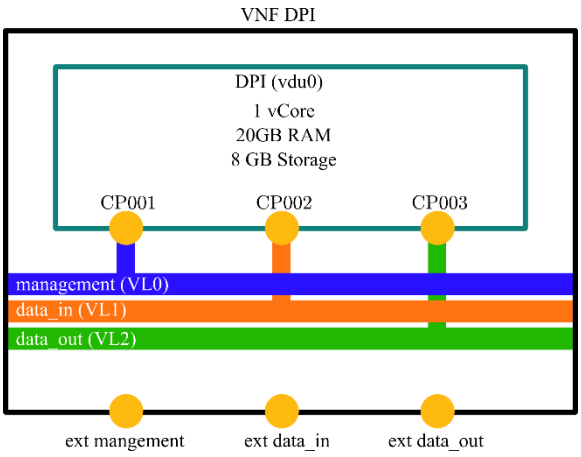

Figure 3. Representation of VNF with defined VNFD

In the visual representation of the VNFD, we can see in orange the various connection points defined inside the VNFD. These connection points are attached to networks which later will be used for inter and intra communication of the various functionalities that a VNF can provide. In the figure above, we also have internal orange dots and external ones. The main difference is that the internal ones define inter-communication inside the various internal services of the VNF while the external dots are used as connection points among the external networks. Finally, we can see the CPU, Memory and storage that this VNF has attached.

C. The VNF High level description

In T-NOVA, a VNF is defined as a group of Virtual Network Function Components (VNFC), each of them consisting of a single Virtual Machine. Each VNF shall support the T-NOVA VNF lifecycle (i.e. start, stop, pause, scaling, etc.) under the control of the VNF Manager (VNFM) entity in the orchestrator layer. VNF developers who aim to develop new
VNFs shall follow the common practices introduced in the T-NOVA framework. With the exception of some mandatory blocks (described in the following), the internal implementations of a VNF is left to the VNF developer, nonetheless, the adoption of a common structure for VNF components as depicted in Figure *4* is recommended.

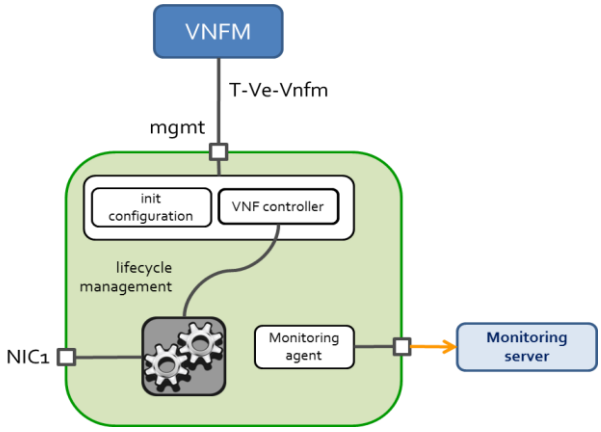

Figure 4. VNF internal components

The **VNF Controller** is the internal component devoted to the support of the VNF lifecycle. The **Init Configuration** component is responsible for the initialization of the VNF that happens at the beginning of the VNF execution. The **Monitoring Agent** component transmits both system and application-level monitoring data towards the Monitoring System [24].

All the VNF internal components are optional, except for the VNF Controller. The VNF Controller often acts as a VNF Master Function and is responsible for the internal organization of the VNFCs into a single VNF entity [25]. It must be present in each VNF because it is in charge of supporting the *T-Ve-Vnfm* interface (defined in [26]) towards the VNFM. VNF developers are free to develop VNF internal components in any way they prefer as long as they comply with the VNF lifecycle management interface *T-Ve-Vnfm*.

In the case of a composed VNF, the internal VNF components required by T-NOVA can be allocated in many different ways. The minimal mandatory requirement is that each VNF must have only one VNF controller. The other components, i.e. Init Configuration and Monitoring Agent, can be optionally allocated in the different VNFCs. Figure *5* provides an example of a VNF composed by two VNFCs. In this case, the Init Configuration component is allocated in all the VNFCs, while there is only one Monitoring Agent. Of course, other configurations are possible depending on the particularities of the VNF.

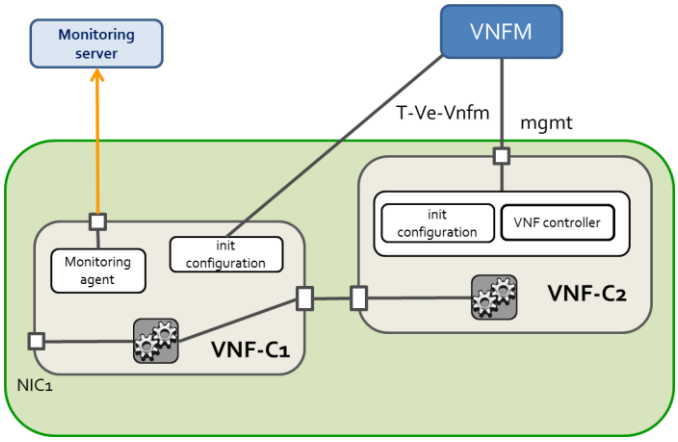

Figure 5. The T-NOVA architecture – VNF internal components in multi VNFCs VNF

D. The VNFs life Cycle

Each VNF shall have a management interface, named *T-Ve-Vnfm* interface that is used to support the VNF lifecycle. The VNF lifecycle is implemented by events generated by the VNFM towards the VNF Controller in each VNF.
The VNF lifecycle is shown in Figure 6.

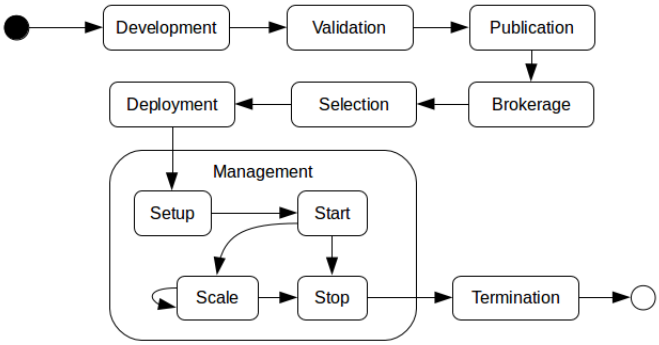

Figure 6. VNF lifecycle

In the T-NOVA project, we have defined the following states:

- **Development**: Software implementation of Network Functions (Nfs). NFs are published and aggregated in the T-NOVA Function Store.
- **Validation**: The validation procedure aims at providing a certification that the developed NFs will work as expected.
- **Publication**: NF publication is performed at the NF Store, whose repository hosts both the function Virtual Deployment Unit (VDU) images and the associated description/metadata (VNFD)
- **Brokerage**: undertaken by the brokerage module in the marketplace that matches users' service requirements with the technical capabilities provided by the NFs, thus ensuring that the resources required for NFs deployment are available.
- **Selection**: Finally, the customer selects the most suitable NFs according to their needs.
- **Deployment**: The VNF VM image and its metadata are transferred from the NF Store to the virtualized infrastructure.
- **Management**: This is the running phase of the NF. Through the *T-Ve-Vnfm* interface, the VNF Manager controls the VNF active states. They are: **Setup** of the VNF, **Start** and **Stop** of the service provided by the VNF, **Scale** of the VNF resources (scale-out and in).
- **Termination**: Involves the removal of the NF instance from the virtualized infrastructure, including network re-configuration, if needed.

An extended description of the VNF lifecycle can be found in [24].

In accordance to ETSI MANO [21], the interaction between VNFM and VNF, implementing the VNF lifecycle, is thoroughly described in the VNF Descriptor (VNFD) that contains a section called "*lifecycle_event*", providing all the details to allow the VNFM to interact with the VNF in a fully automated way.

To this aim, each VNF needs to be able to declare various lifecycle events (e.g. start, stop, scale out...). For each of those, the information needed to configure the VNF can be very different. Moreover, the command to trigger the re-configuration of the VNF can change between events.

The information related to the VNF life-cycle is inserted in the "*lifecycle_event*" section of the VNFD. In particular, in such a section the following information is available:

- **Driver**: the protocol used for the connection between the VNF Manager and the controlling VNFC. Currently T-NOVA supports two protocols: SSH and HTTP.
- **Authentication**: such fields specify the type of authentication that must be used for the connection and some specific data required by the authentication process (e.g. a link to the private key injected by the VNFM at initialization time or HTTP credentials).
- **Template File Format**: specifies the format of the file that contains the information about the specific lifecycle event, and that must be transferred once the command is run.
- **Template File**: includes the name and the location of the Template File.
- **VNF Container**: specifies the location of the Template File.
- **Command**: defines the command to be executed

E. VNFC Networking

According to the T-NOVA architecture, each VNFC should have four separate network interfaces, each one bound to a separate isolated network segment. The networks related to any VNFC are: management, datapath, monitoring and storage. The figure (Figure **7**) illustrates the above statement. In various cases, the above rule might not be followed, especially in the case where there is no real requirement for a particular network/interface e.g. a VNFC that is not using persistent storage on the storage array of the NFVI.

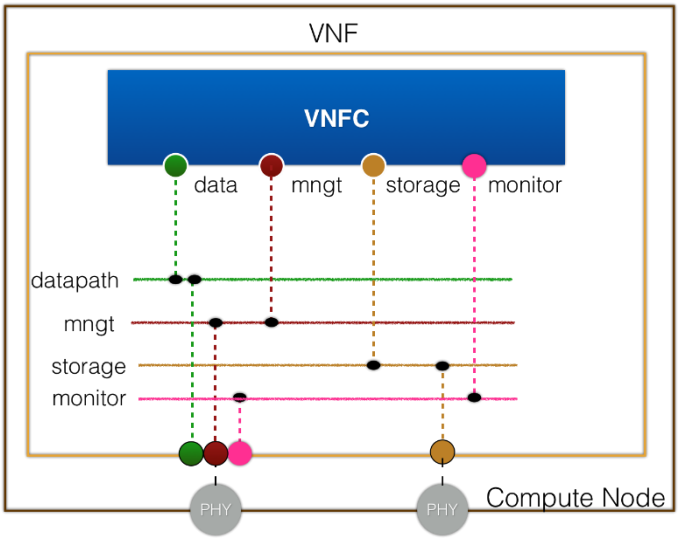

Figure 7. VNF/VNFC Virtual Links and Connection Points

The **management network** provides the connection and communication with the VNFM and passes lifecycle related events seamlessly to the VNF. The management network conveys the information passed over from VNFM to the VNF Controller (*T-Vnfm-Vnf* interface). The VNFM can control each particular VNFC, through the management interface, and provision the overall stable functionality of the VNF.

The **datapath network** provides the networking for the VNF to accept and send data traffic related to the network function it supports. For example, a virtual Traffic Classifier would receive the traffic under classification from this interface. The datapath network can consist of more than one network interface that can receive and send data traffic. For example, in the case where the function of the VNFC is a L2 function (e.g. a bridge), the anticipated interfaces are two, one for the ingress and another one for the egress. In some cases, those interfaces might be mapped on different physical interfaces too. The number of interfaces for the datapath and their particular use is decided by the VNF provider. Additionally, the data traffic that needs to be shared among different VNFCs uses the datapath network.

The **monitoring network** provides the communication with the monitoring framework. Each VNF has a monitoring agent installed on each VNFC, which collects VNF specific monitoring data and signals them to the Monitoring Framework (see [27]) or to the VNFM/EM depending on the situation. The flow of the monitoring data is from the VNF service to the monitoring agent to the framework, which collects data from all VNFs. A separate Monitoring network has been introduced in T-NOVA to cope with the amount of traffic generated in large-scale deployments. Though, the monitoring traffic can be easily aggregated with the management traffic into a single network instance, if this solution results adequate to the specific application.

The **storage network** is intended for supporting communication of the VNFCs with the storage infrastructure provided at each NFVI. This applies to the case where a VNFC will utilize persistent storage. In many NFVI deployment scenarios, the physical interface that handles the storage signaling (e.g. iSCSI) on each compute node is separated from the other network segments. This network segment is considered optional and only applicable to the above use cases.

F. The VNF Forwarding Graph Descriptor

The creation of an end-to-end network service needs to define inter-communication among the various Virtual Network functions as displayed in figure 3 where the external part of the VNFD is ready to be used for Inter-communication with other VNFs. This inter-communication among the VNF's is defined by the VNF Forwarding graph (VNFG) providing a description of how and which connection point is linked among the various NSD of the VNFs. The forwarding graph must take into account various aspects like how we can reduce the configuration complexity in order to achieve a simpler and straightforward forward graph defined as efficient and resilient in order to achieve the best Agility. Taking all this into account, we have implemented a visual tool inside the T-NOVA Marketplace that provides all necessary means for specifying the attribute of each VNF to the desired VNFG (see VNFG configuration below).

```
"vnffgd":{
    "vnffgs":[
     { "vnffg_id":"vnffg0",
       "number_of_endpoints":1,
       "number_of_virtual_links":1,
       "dependent_virtual_links":[
         "vld0"          ],
       "network_forwarding_path":[
        {"nfp_id":"nfp0",
         "graph":[
          "vld0"      ],
         "connection_points":[
          "ns_ext_vl0-truck",
          "VNF#1625:ext_mgnt",
          "VNF#1625:ext_data"       ],
         "constituent_vnfs":[
          {"vnf_ref_id":"1625",
           "vnf_flavor_key_ref":"gold"
}]}]}]}
```


Figure 8. VNFG configuration generation

The output configuration is done by enabling connection links in a visual manner inside the T-NOVA Dashboard and creating an output of how the various VNFs are inter-connected. The output displayed above provides the ability to configure and generate the necessary inter-connection of the external networks of the various VNFs. Furthermore, the various networks are clearly defined in order to allow the Migration and coexistence of virtualized and non-virtualized NFs. In figure 10 we see with the red line how a forwarding graph is created in order to utilize various Points of Presents and connecting the end user with the vProxy Appliance.

III. Examples of Developed VNFs

Several VNFs are being developed within the T-NOVA Project. On top of implementing and fine-tuning the software to achieve design goals, these functions have been integrated within the platform following the design decisions presented in previous sections. Even if they leverage the T-NOVA framework like specific marketplace integration and monitoring, the core functionalities can be ported easily to compliant NFV platforms.

A. The Virtual Security Appliance

A Security Appliance (SA) is simply a “device” designed to protect computer networks from unwanted traffic. This device can be active and block unwanted traffic. This is the case for instance of firewalls and content filters. A Security Appliance can also be passive. Here, its role is simply detection and reporting. Intrusion Detection Systems are a good example. A virtual Security Appliance (vSA) is a SA that runs in a virtual environment. In the context of T-NOVA, we have suggested a vSA composed of a firewall, an Intrusion Detection System (IDS) and a controller that links the activities of the firewall and the IDS. The vSA high level architecture was discussed in detail in [10]. The idea behind the vSA is to let the IDS analyze the traffic targeting the service and if some traffic looks suspicious, the controller takes a decision by, for instance, revising the rules in the firewall and blocking this traffic. The architecture of this appliance is depicted in figure 9 and includes the following main components.

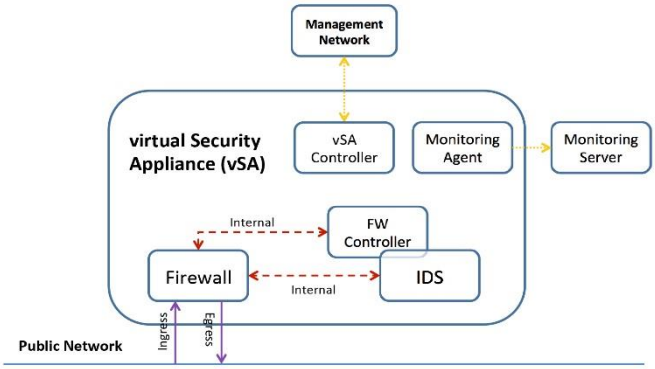

Figure 9. vSA high level architecture

The firewall: this component is in charge of filtering the traffic towards the service.

The Intrusion Detection System: In order to improve attack detection, a combination of a packet filtering firewall and an intrusion detection system using both signatures and anomaly detection is considered. In fact, Anomaly detection IDS have the advantage over signature based IDS in detecting novel attacks for which signatures do not exist. Unfortunately, anomaly detection IDS suffer from high false-positive detection rates. It is expected that combining both arts of detection will improve detection and reduce the number of false alarms. In T-NOVA, the open source signature based IDS ([3]) is used and will be extended to support anomaly detection as well. The mode of operation of the IDS component was also discussed in [10].

The FW Controller: this application looks into the IDS "alerts repository" and based on the related information the rules of the firewall are revised.

The Monitoring Agent: this is a script that reports to the monitoring server the status of the VNF through some metrics such as (Number of errors coming in/ going out of the WAN/LAN interface of pfsense, Number of bytes coming in/ going out of the wan/lan interface of pfsense, CPU usage of snort, Percent of the dropped packets, generated by snort, etc)

The vSA controller: this is the application in charge of the vSA lifecycle (for more details, we refer to section II-D).

B. *The Virtual Proxy*

vPXaaS (virtual Proxy as a Service) provides proxy services on demand to both Internet Service providers' (ISP) subscribers:

- Home users e.g. DSL subscribers.
- Corporate users e.g. company subscribers.

The idea behind using the proxy as a VNF is to move it from the LAN to the cloud to be used "as a service". Nevertheless, a subscriber (LAN administrator) will be able to configure the proxy from a web-based user friendly dashboard.

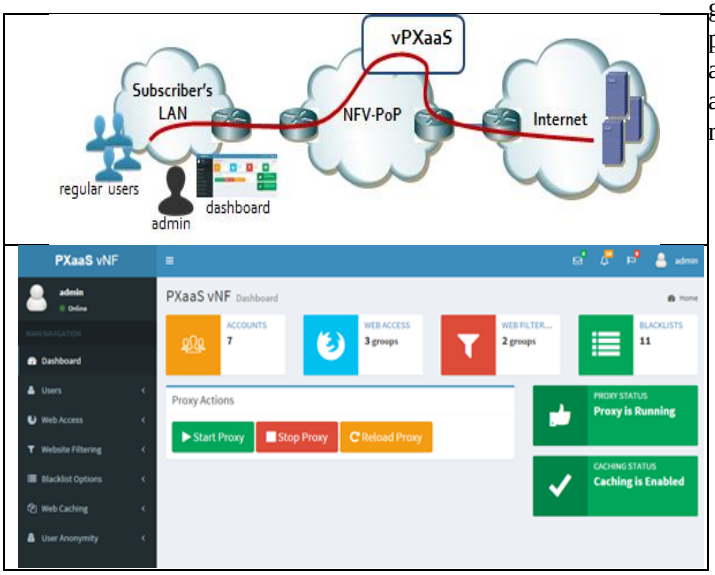

Figure 10. vProxy high level architecture

The PXaaS vNF was developed in the context of T-NOVA and offers the following features,

- Web Access control
- Bandwidth control (Web traffic)
- Web site filtering
- Web caching
- User anonymity

C. *The Virtual Session Border Controller*

A Session Border Controller (SBC) provides network interconnection and security services between two IP networks. It operates at the edge of these networks and is used whenever a multimedia session involves two different IP domains. It performs:

- the session control on the "control" plane, adopting SIP as a signaling protocol [28];
- several functions on the "media" plane (i.e : transcoding, trans-rating, NAT, etc), adopting Real time Transport Protocol (RTP) for multimedia content delivery.

The vSBC is the VNF implementing the SBC service in the T-NOVA virtualized environment. It is a prototype version of the commercial SBC that Italtel is developing for the NFV market. General requirements for vSBCs comprise both essential features (such as: IP to IP network interconnection, SIP signaling proxy, Media flow NAT, RTP media support) and also advanced requirements (such as: SIP signaling manipulation, real-time audio and/or video transcoding, Topology hiding, Security gateway, IPv4-IPv6 gateway, generation of metrics, etc.). For the objectives of the T-NOVA project, we focus on all essential features and a subset of advanced requirements (i.e. IPv4-IPv6 gateway; real-time audio and/or video transcoding for mobile and fixed network; metrics generation; etc).

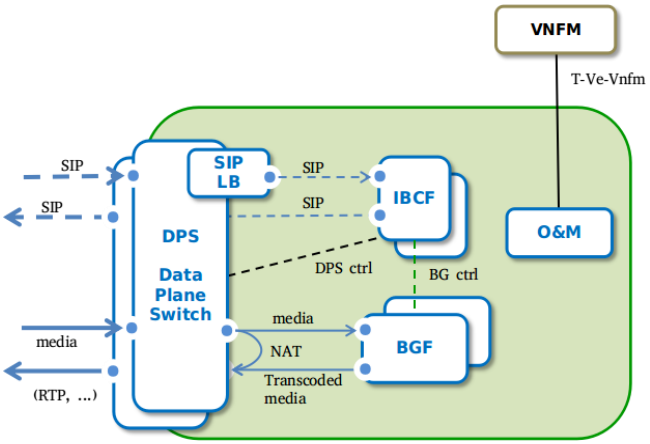

Figure 11. Enhanced vSBC internal architecture (with DPS component).

- **SIP Load Balancer (LB):** balances the incoming SIP messages, forwarding them to the appropriate IBCF instance.
- **Interconnection Border Control Function (IBCF):** implements the control function of the SBC. It analyzes the incoming SIP messages, and handles the communication between disparate SIP endpoint applications. The IBCF extracts from incoming SIP messages the information about media streams associated to the SIP dialog, and instructs media plane components (DPS and/or BGF) to process them.
- **Border Gateway Function (BGF):** processes media streams, applying transcoding and transrating algorithms when needed (transcoding transforms the algorithm used for coding the media stream, while

transrating changes the sending rate of IP packets carrying media content). This feature is used whenever the endpoints of the media connection support different codecs, and it is an ancillary function for an SBC because, in common network deployments, only a limited subset of media streams processed by the SBC need to be transcoded. The BGF is controlled by the IBCF using the internal BG ctrl interface. If the transcoding /transrating function is implemented by a pure software transcoder its performances dramatically decrease, unless GPU (Graphical Processing Units) hardware acceleration is available in the virtual environment. Otherwise this issue could be mitigated by running more BGF instances. The BGF component can also provide metrics to the T-NOVA monitoring agent**;**

- **Operating and Maintenance (O&M):** it supervises the operating and maintenance functions of the VNF. In a cloud environment, the O&M module extends the traditional management operations handled by the Orchestrator (i.e. scaling). The O&M component interacts (via HTTP) with the VNF manager, using the *T-Ve-Vnfm* interface, for applying the T-NOVA lifecycle**;**
- **Data Plane Switch (DPS):** it is the (optional) front-end component of the vSBC based on DPDK acceleration technology available in Intel x86 architectures to reach a performance level comparable to the hardware-based version adopting HW acceleration technologies. This component can use the same IP address, as ingress or egress point, both for signaling and media flows. Its goal is to provide high speed in processing the addressing information in the header of the IP packet, leaving the payload untouched. The DPS is instructed how to manage the IP packets by the IBCF component, acting as an external controller using an internal dedicated DPS ctrl interface (see Figure **11**). The DPS component can:
  - either provide the packet forwarding to the BGF (in case of transcoding)
  - or apply a local Network Address Translation (NAT)/port translation

D. The virtual Content Delivery Network.

Content Delivery networks (CDN) have been created to cope with the challenges encountered by Content Providers (CP) to deliver huge amounts of static data through best effort Internet. ISPs are interested in building these solutions as they represent an interesting growth driver.

Described by ETSI as a virtualization use case [14], vCDN fits well in the T-NOVA project due to the specific challenges of its implementation in an operator network, technical breakthroughs made possible by the NFV approach and business model innovation for ISPs.

Using their substrate to deploy vCDN architecture [15] allows the ISPs to benefit even more from the creation of Next Generation Point of Presence ([16], [17]), designed to host VNF at the edge of their network and to leverage their end-to-end network management. Furthermore, the increased flexibility offered by placing caches in POPs coupled with sophisticated service mapping models, enable them to provide solutions for Content Providers Quality of Experience requirements. Besides, using NFV enables the deployment of better alternatives to handle Service-User assignment tasks, like the Virtual Media Gateway we have introduced in this work. Finally, Network Function scaling is used to adapt to fluctuating content delivery demand and content ingestion requirements.
Our proposal is developed around 4 main modules:

- **The Virtual Media Gateway (VMG)**: is a transit Network Function inspecting high-level HTTP traffic that can influence the IP routing decisions, based on the presence of the content in a nearby POP. Its configuration is provided by the caching orchestrator which has a complete vision of the system.

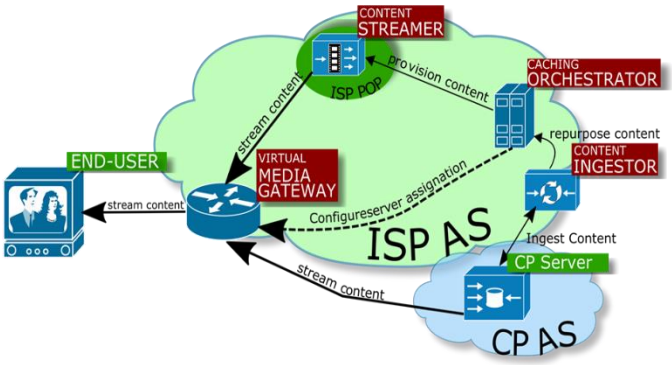

Figure 12. VCDN High Level Architecture

- **The Content Streamers**: are built around a distributed object storage engine that provides resiliency, horizontal scalability and geographical redundancy amongst POPs.
- **Content Ingestors**: are scalable workers that perform software transcoding to H264 and H265 video compression standards as well as re-segmentation of videos using both DASH [19] and HLS technology to provide adaptive HTTP Streaming capability. Ingestors receive content from the CP Servers (push model) or can be automatically provisioned from the most popular contents (pull model). Ingestors have been shown to be able to rely on hardware accelerators (Virtual Transcoding Units, also part of the T-NOVA Project [20]) for computer intensive tasks when available.
- **Caching Orchestrator**: is the module in charge of controlling the ingestion (by scheduling the job of the workers), the provisioning (by selecting which content

is cached in which streamer) and the delivery (by deploying configuration on the VMG) of content

- **Technical considerations (a case for using containers for the T-NOVA architecture)**: The only implementation available for what ETSI calls Virtualization Deployment Unit (VDU) are Virtual Machine Images. It means that whenever any software component requires an update (bug fix, or a new feature), a new VM Image needs to be generated, tested and deployed in the system. Even if this cycle is fully automated and carried out without human intervention, from a software engineering standpoint, streamlining this process is time consuming. It can take several hours to test a simple modification due to VM image processing, VM image transfer to the remote testbed and finally VNF embedding. To circumvent this issue, we packaged all our software using Docker containers technology. Without any drawback in terms of performance, we saved hours of tedious manipulation by injecting frequently updated Docker images into the same vanilla VM Image at VNF embedding time (saving the time or VM Image generation and transfer). Containers technology allowed us to increase software agility without sacrificing runtime performances.

IV. Integration of VNFs and Service Chaining

In this section, we describe a couple of scenarios showing how different VNFs can be integrated with each other. To achieve this, some real life scenarios were imitated.

A. The vSA-vProxy scenario

One way to integrate the vSA and the vProxy is to put them together to reflect a proxy scan attack detection. The latter means the attacker tries to find potential proxies to use them utilizing a port scan attack. Port scanning is a general technique used to survey one or more network connected hosts for availability. Port scanning is often called network scanning. We may scan a host for more specific services. Typically, we may check that one server responds on TCP port 80 (HTTP) to ensure that our Web service is up and running. The vSA-vProxy works as follows,

- The IDS (Snort) in the vSA is configured to detect port scan attacks)
- Nmap [11] sends packets (supposed to be the attacker)
- The packets are intercepted by the Firewall
- The packets are also analyzed by the IDS
- Port scan attack detected, IDS instructs the Firewall to block traffic coming from the IP address of Nmap

B. The vSA-vSBC scenario

The integration of the vSA and the vSBC also makes sense if we look at it as a VoIP service enhanced with security at layers three and four. The combination of vSA and vSBC allows implementing a multi-layer approach to the security of the signaling layer. The role of the vSA is to secure the IP, TCP and UDP protocols, while the vSBC provides security at SIP level. This integration could also be seen as a Border Control Function (BCF) of the IP based Emergency Services Network (ESInet) where the BCF has to be a combination of a firewall and a SBC in order to protect the ESInet [12] and the Public Answering Points (PSAPs) from malicious traffic.

V. Experimentation

In this section, we discuss some of the experiments we performed in order to validate our work. Although different aspects have been tested, our focus in this paper is more on the performance issue. In fact, in the past, the performance of the appliances was achieved through dedicated hardware. In virtualized environments, this is not possible because different applications might run on the same operating system and compete for the same hardware computing resources.

A. vSA testing

To study the performance of the security appliances (in particular the firewalls), appropriate metrics are needed. Although the activities in this area are very scarce, we described in [10] potential metrics that could be used. This includes throughput, latency, jitter, and goodput. For more detail please refer to [10].

For simplicity reasons, we have used Iperf [5] for generating IP traffic in our tests. In fact, other IP traffic generators such as D-ITG [6], ostinato [7], and IPTraf [8] could have also been utilized. Iperf mainly generates TCP and UDP traffic at different rates. Diverse loads (light, medium, heavy) and different packet sizes are also considered. For analyzing IP traffic, we used "tcpdump" for capturing it and "tcptrace" to analyze it and generate statistics. The main difference with respect to the tests performed in [10] is the fact that in this paper, the tests are performed on a cloud computing platform (not simply in VirtualBox [9]) namely, Openstack [13]. This also enables the testing of some networking functionalities of OpenStack as the latter does not offer much freedom and flexibility on arbitrary traffic steering. Similarly to [10], the undertaken tests are based on three main scenarios,

- Scenario one (No firewall): Here, we configure and check the connectivity between the Iperf client and the virtual proxy without a firewall in between. This enables us to test the capacity of the communication channel
- Scenario two (TCP traffic with firewall and no rules): Here, we check whether the introduction of the vSA (in particular, the firewall in between) generates extra delay. We also test the capacity of the vSA in this context
- Scenario three (with firewall and increasing number of rules): the objective of this scenario is to study the effect of introducing rules into the firewall of the vSA. To achieve this scenario, a script for the firewall is implemented in

order to generate rules in an automatic way. The script is a shell script using specific API commands and generates blocking rules for random source IP addresses (excluding those used in the test setup) and the WAN interface. Here, the easyrule function of pfsense is extended. In this scenario, some tests are also performed using UDP instead of TCP

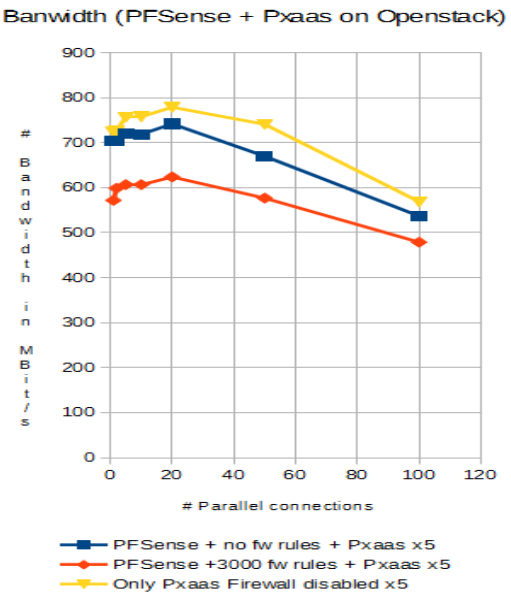

Figure 13. vSA throughput

When no firewall is used between the Iperf client and the virtual proxy, one can note that the throughput of the communication remains good (between 700 and 800 Mbit/s) as long as the number of 60 parallel connections is not exceeded. When the vSA (in particular the firewall) is in between, the throughput varies between 700 and 750 Mbit/s as long as the number of parallel connections does not exceed 20 connections. When the number of connections goes beyond the value 60, the throughput for the vSA without firewall rules decreases slowly to reach 580 Mbit/s when 100 connections are opened (Figure 13). This situation becomes worse when rules are configured on the firewall. Indeed, the throughput decreases to 480 Mbit/s when 3000 rules are configured and 100 connections are opened (Figure 13).

B. vSBC testing

The vSBC was tested in a laboratory for studying the load curve characterization under traffic conditions. As introduced in the previous sections, the SBC is logically composed by signaling and media planes. The signaling plane processes SIP messages while the media plane works on RTP packets, under the control of the signaling plane. This intrinsic architecture suggests a deployment scenario composed by separate virtual machines for the two planes.

The configuration under test was composed by two VMs, namely VDU1 performing the SIP load balancer and the IBCF control function, and VDU2 dedicated to the media plane peforming the BGF.

VDU1 was configured with 8 virtual CPUs and 16 Gb of memory.

VDU2 was configured with 8 virtual CPUs and 4 Gb of memory.

The testbed configuration was as follows:

- Server HP ProLiant DL380 G5, single server equipped with 2 Processors Quad-Core Intel® Xeon® Processor E5335 (2.00 GHz), Memory 32 GB.
- CentOS (version 7.2)
- Openstack (Liberty)

The traffic conditions were simulated by different traffic generators emulating SIP and Media flows towards the vSBC. The open-source SIPP [29] protocol generator was used to test the control plane. More sophisticated tests were performed by using commercial traffic generators such as Catapult and NeTracker. The latter was able to test both the signaling and media planes.

The goal of the test was to measure key performance indicators under different traffic conditions. These indicators were CPU load, memory usage, and network throughput.

The following diagrams report the test results in case of 0, 10, 60, 120, 240, 500 parallel sessions for a NAT service for G.711 codec with 20ms of packetization time. The sessions refer to calls of 120 seconds in duration.

The CPU load of the signaling layer smoothly increases with the number of sessions, while the media layer is heavily impacted (see Figure *14*).

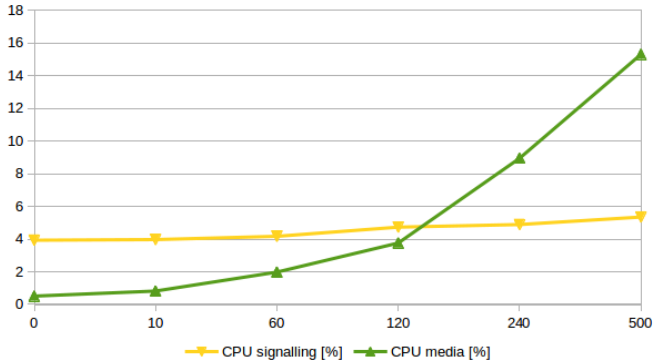

Figure 14. vSBC CPU load

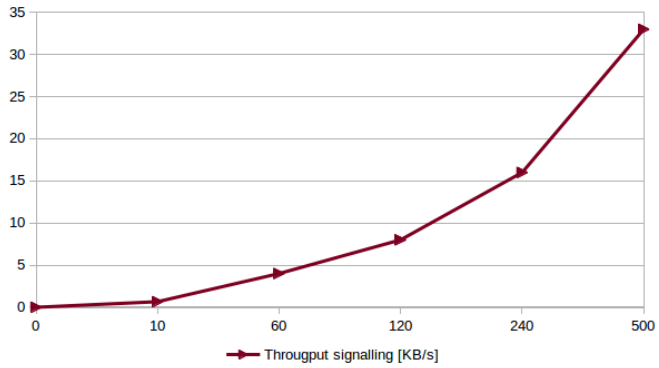

Figure 15. vSBC signaling layer throughput

The throughput of the signaling layer is represented in Figure *15*. The throughput of the media layer, represented in Figure *16*, has a similar behavior. Of course there is a difference of several order of magnitude due to the large amount of media packets with respect to the associates signaling messages.

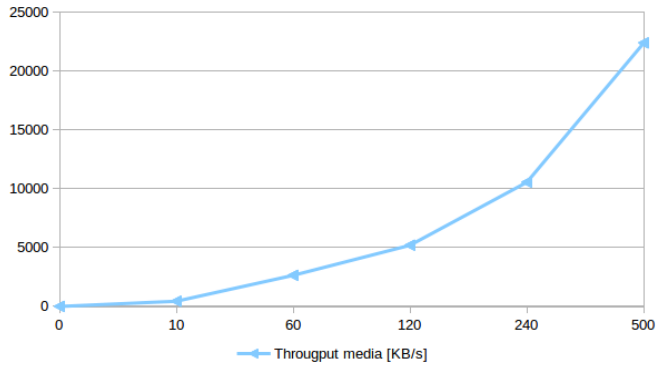

Figure 16. vSBC media layer throughput

From the laboratory tests we learned that when the traffic increases to around 900 parallel sessions, the system starts suffering and dropping packets. This is a limit situation for the described configuration that could be managed either by limiting the traffic or implementing a scale-out procedure. Currently scaling is under final development and meaningful figures are not available yet.

C. vCDN testing

For testing purposes, the vCDN can be seen as a chain of micro-services working together to implement the function. Having several components interacting together augments the complexity of the task of characterizing the bottlenecks of the solution. We also need to take into account the fact that absolute performance is not really meaningful for scalable applications since adding additional resources increase the processing capacity and the state of the cloud environment hosting the solution can vary over time, along with the performances.
We carried out our experiments in a fully-fledged NFV Infrastructure deployed within the T-NOVA project for a baseline configuration of 5 Virtual Machines (with 4 vcore and 4GB of RAM each). We only present high level performance results corresponding to the 2 end-to-end scenarios: Ingestion-Provisioning and Delivery.

1) Testing vCDN Ingestion-Provisionning

For the vCDN, ingestion means deploying the original content in the object store, analyzing this content, deciding which the optimal format for the content is and producing the adapted content. It is a very CPU and memory intensive task that can be easily scaled with the adjunction of a "worker" VM. Figure 17 shows a setting where we let the system ingest 200 videos of 20 MB at an average arrival rate of 30 videos per minute. We compare the number of "pending" video jobs that are queued by the system for several settings. We scaled our VNF out and allowed the number of ingestion VM to vary.

We can see that the configuration with only 1 VM doesn't cope with the load as it accumulates more than 120 pending videos and it depletes its video stock in more than 900 s. On the contrary the 3 VM setting manages to finish nearly on time (420s).

Thanks to this design the number of ingestion-provisioning VM can be adjusted based on the characteristics of the videos and on the tolerance to delay of the customer.

Figure 17. VCDN Ingestion-Provisionning Performances

2) Testing the vCDN Delivery

The overall performance of the delivery part of the vCDN depends to a large extent on the network performances between

the object storage nodes. Indeed each content is chunked and spread on several nodes to provide redundancy and increase performance. Furthermore, the Virtual Media Gateway is used to inspect HTTP Packets, which may also cause delay and reduced throughput.

In Figure 18 we used apache's *ab* tool to compute the 95 percentile maximum time taken to download a 10s, 6 MB video file encoded as 600 KBps. We increased the number of concurrent connections to establish the threshold above which the video cannot be streamed at its nominal bitrate for the 5VM baseline configuration.

We can see two important results from the graph. First of all, there's no significant difference between the performance of storage with or without the VMG. It means that the storage is the bottleneck in this case and the VMG does not need to be scaled-up to increase performance. Next, the video can be streamed by 250 simultaneous users. This value is strongly correlated to the underlying state of the network on our infrastructure and also on the storage technology used in the platform. For example, our object storage engine is designed to use SSD disks to boost the delivery of the most used files. This feature was not available on our infrastructure and could have dramatically increased performances, especially for internet content where only a small number of items are popular while the rest remain unknown.

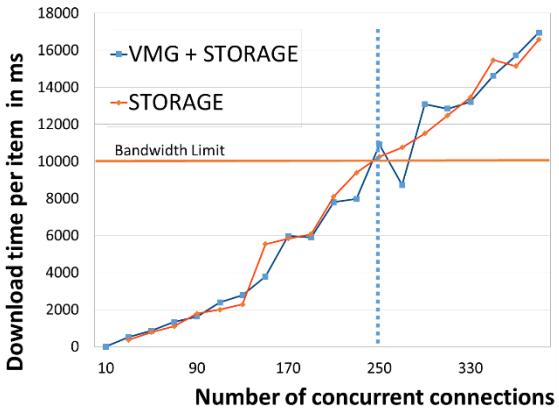

Figure 18 . vCDN Content Delivery Performances

VI. NFV Architectures Survey

This section provides a short summary of a number of NFV platforms and architectures as proposed by industry frameworks and solutions as well as efforts from Standardization Bodies related to NFV. For a more detailed description, we refer to the T-NOVA deliverable D2.22 [30].

ETSI ISG NFV: A network operator led Industry Specification Group (ISG) with open membership was setup in the last quarter of 2012 under the umbrella of ETSI to work through the technical challenges of Network Functions Virtualization. It is worth noting that ETSI ISG NFV does not provide standards but rather produces guideline documents in the form of Group Specifications. The outputs are openly published and shared with relevant standards bodies, industry Fora and Consortia to encourage a wider collaborative effort. A PoC of T-NOVA end-to-end orchestration has been recently accepted by ETSI [31]. It will verify that E2E Service Orchestration enables the VNF to run on top of the NFVI and is able to optimize the location and required resources of the VNFs.

TM Forum (TMF): TMF is a global trade association of service providers and suppliers with the overall objective of progressing and succeeding in the digital economy. In short, TM forum works in 4 key areas: Business & IT transformation, Business metrics and KPIs, Cybersecurity and Managing Virtualized Networks and Services. It is in the context of this last mentioned area where TMF has recently kicked off a major new project with the aim of creating a blueprint for a new generation of service provider support systems to achieve business agility when delivering virtual network and services; it is the zero-touch orchestration, operations and management (ZOOM) project [32].

CloudNFV: It is an open platform for implementing NFV based on cloud computing and SDN technologies in a multi-vendor environment. The companies currently involved are: 6WIND, CIMI Corporation, Dell, Enterprise Web, Overture Networks, and Qosmos. It has been recently accepted as a proof of concept (PoC) in the frame of ETSI NFV ISG. CloudNFV builds on the NFV ISG work in order to validate it within the broadest possible framework of service creation and operations and to incorporate recent critical revolutions such as 'Cloud' and SDN. The project motivation stems from the fact that it considers NFV ISG's scopes too large to progress it in time. In this concept CloudNFV proposes an implementation, an extension of ISG principles to ISG adjacent domains.

OpenNFV: It is a comprehensive project launched by HP, built around a proposed open reference architecture encompassing a service portfolio and enforced by an ecosystem of ISVs, NEPs and application developers [33].
OenNFV architecture is aligned with the ETSI model. Its main components are a NFV Infrastructure and a NFV Orchestrator module, in turn based on HP Converged Infrastructure and HP Converged Cloud propositions. It also capitalizes on the SDN role, and on HP's SDN technology assets. It is a modular architecture, basically vendor agnostic and allowing a modularized approach to NFV take -up.

Qosmos/Intel/Tieto: Intel has long been an active player in supporting the development and evolution of NFV and SDN through industry and vendor specific initiatives. The network builders program for example is an industry initiative

comprising of more than 70 companies. The goal of the program is to make it easier to build, enhance and operate SDN/NFV-based infrastructure while lowering capital and operating expenditure. The program publishes function specific architectures such as vEPC, vBRAS, and vCPE.

VII. Acknowledgement

This work has been undertaken in the context of the European project T-NOVA that is an Integrated Project co-funded by the European Commission / 7th Framework Program, Grant Agreement no. 619520.

VIII. Conclusion

In this paper, we have provided an overview on the T-NOVA Platform with a special focus on the deployment and performances of a wide variety of Virtual Network Functions implemented during the project. Furthermore, real-world considerations for handling VNF Lifecycles, Monitoring and Networking have also been discussed. Lastly, we have highlighted two examples of Service Chaining where several VNFs are combined through the T-NOVA Marketplace to create added-value services.

It is expected that the challenges and opportunities described in this paper will help foster innovation around NFV and will help pave the way for practitioners and researchers alike to further extend the use of Network Function virtualization in the future.

References

[1] Network Function Virtualization: An Introduction, Benefits, Enablers, Challenges, and Call for Action. ETSI White Paper, link: https://portal.etsi.org/nfv/nfv_white_paper.pdf

[2] The T-NOVA project, link: www.t-nova.eu

[3] Snort, link: https://www.snort.org/

[4] Pfsense firewall, link: https://www.pfsense.org//

[5] Iperf, link: https://iperf.fr/

[6] D-ITG, link: http://traffic.comics.unina.it/software/ITG/

[7] Ostinato, link: https://code.google.com/p/ostinato/

[8] Iptraf, link: http://iptraf.seul.org/

[9] VirtualBox, link: https://www.virtualbox.org/

[10] Y. Rebahi, et Al, Virtual Security Appliances: The Next Generation Security", In the Proc of the IEEE ComManTel 2015, December 2015, Da Nang, Vietnam

[11] Nmap, link: www.nmap.org

[12] ESInet in What is NG911, link: https://c.ymcdn.com/sites/www.nena.org/resource/resmgr/ng9-1-1_project/whatisng911.pdf

[13] Openstack, link: www.openstack.org

[14] ETSI, GSNFV. "Network Functions Virtualisation (NFV); Use Cases." V1 1 (2013): 2013-10.

[15] Pantelis A. Frangoudis, Louiza Yala, Adlen Ksentini, Tarik Taleb. "An architecture for on-demand service deployment over a telco CDN" In Communications (ICC), 2016 IEEE International Conference on. IEEE, 2016.

[16] Gosselin, Stéphane, et al. "Converged fixed and mobile broadband networks based on next generation point of presence." Future Network and Mobile Summit (FutureNetworkSummit), 2013. IEEE, 2013.

[17] Combo Project COMBO (COnvergence of fixed and Mobile BrOadband access/aggregation networks) http://www.ict-combo.eu

[18] Nicolas Herbaut, Daniel Negru, Damien Magoni, Pantelis A. Frangoudis. " Deploying a Content Delivery Service Function Chain on an SDN-NFV Operator Infrastructure" Telecommunications and Multimedia (TEMU), 2016 International Conference on. IEEE, 2016.

[19] Sodagar, Iraj. "The mpeg-dash standard for multimedia streaming over the internet." IEEE MultiMedia 4 (2011): 62-67.

[20] P. Comi, P. Secondo Crosta, M. Beccari, P. Paglierani, G. Grossi, F. Pedersini, A. Petrini "Hardware-accelerated High-resolution Video Coding in Virtual Network Functions" Networks and Communications (EuCNC), 2016 European Conference on. IEEE, 2016.

[21] ETSI GS NFV-MAN 001: "Network Functions Virtualisation (NFV); Management and Orchestration"

[22] T-NOVA D3.42: "Service Provisioning, Management and Monitoring - Final" Web link: http://www.t-nova.eu/results/

[23] T-NOVA D3.3: "Service Mapping" Web link: http://www.t-nova.eu/wp-content/uploads/2016/02/Deliverable_3.3_Service_Mapping_v1.0.pdf

[24] T-NOVA D2.42: "Specification of the Network Function framework and T-NOVA Marketplace - Final" Web link: http://www.t-nova.eu/wp-content/uploads/2016/03/TNOVA_D2.42_Specification_of_the_Network_Function_Framework_and_T-NOVA_Marketplace.pdf

[25] ETSI GS NFV-SWA 001: "Network Functions Virtualisation (NFV); Virtual Network Functions architecture"

[26] T-NOVA D2.22: "Overall System Architecture and Interfaces - Final" Web link: http://www.t-nova.eu/wp-content/uploads/2016/03/TNOVA_D2.22_Overall_System_Architecture_and_Interfaces_v1.0.pdf

[27] T-NOVA D4.41: "Monitoring and Maintenance – Interim" Web link: http://www.t-nova.eu/wp-content/uploads/2016/03/TNOVA_D4.41_Monitoring_and_Maintenance_Interim.pdf

[28] IETF RFC 3261 "SIP: Session Initiation Protocol"

[29] http://sipp.sourceforge.net

[30] T-NOVA Deliverable D2.22 "Overall System Architecture and Interfaces - Final", September 2015, Link: http://wiki.t-nova.eu/tnovawiki/images/5/52/TNOVA_D2.22_Overall_System_Architecture_and_Interfaces_v1.0.pdf

[31] PoC#40: VNFaaS with end-to-end full service orchestration link: https://docbox.etsi.org/ISG/NFV/TST/05-CONTRIBUTIONS/2016//NFVTST(16)000094r2_PoC_proposal_VNFaaS_e2e_ServOrch.docx

[32] TM Forum ZOOM link: https://www.tmforum.org/zoom/

[33] HP OpenNFV link: http://www8.hp.com/us/en/cloud/nfv-architecture.html

[34] Joel J. P. C. Rodrigues, Kai Lin, and Jaime Lloret, "Mobile Networks and Cloud Computing Convergence for Progressive Services and Applications", IGI-Global Publishers, Hershey, PA, USA, November 2013, 408 pp.s, ISBN: 978-1-4666-4781-7 (hardcover), ISBN: 978-1-4666-4782-4 (ebook), ISSN: 2327-3305, DOI: 10.4018/978-1-4666-4781-7.

[35] Xianbin Wang, Guangjie Han, Xiao-Jiang (James) Du, and Joel J. P. C. Rodrigues (Guest Editors), Special Issue on "Mobile Cloud Computing in 5G: Emerging Trends, Issues, and Challenges", in *IEEE Network Magazine*, IEEE, ISSN: 0890-8044, Volume: 29, Issue 2, March/April 2015, pp. 4-5, DOI: 10.1109/MNET.2015.7064896.

[36] Wenxiang Li, Chunsheng Zhu, Xia Wei, Joel J. P. C. Rodrigues, Kun Wang, "Characteristics Analysis and Optimization Design of Entities Collaboration for Cloud Manufacturing", in *Concurrency and Computation: Practice and Experience*, Wiley, ISSN: 1532-0634, DOI: 10.1002/cpe.3948.

[37] Samaresh Bera, Sudip Misra, Joel J. P. C. Rodrigues, "Cloud Computing Applications for Smart Grid: A Survey", *IEEE Transactions on Parallel and Distributed Systems*, IEEE Computer Society, ISSN: 1045-9219, Vol. 26, Issue 5, May 2015, pp. 1477 - 1494, DOI: 10.1109/TPDS.2014.2321378.

[38] Xianbin Wang, Guangjie Han, Xiao-Jiang (James) Du, and Joel J. P. C. Rodrigues, "Mobile Cloud Computing in 5G: Emerging Trends, Issues, and Challenges [Guest Editorial]", in *IEEE Network Magazine*, IEEE, ISSN: 0890-8044, Volume: 29, Issue 2, March/April 2015, pp. 4-5, DOI: 10.1109/MNET.2015.7064896.

[39] Yonggang Wen, Xiaoqing Zhu, Joel J.P.C. Rodrigues, and Chang Wen Chen, "Cloud Mobile Media: Reflections and Outlook", *IEEE Transactions on Multimedia*, IEEE, ISSN: 1520-9210, Vol. 16, No. 4, June 2014, pp. 885-902, DOI: 10.1109/TMM.2014.2315596.