SDN for Dynamic NFV Deployment

Franco Callegati, Walter Cerroni, Chiara Contoli
Department of Electrical, Electronic and Information Engineering "G. Marconi", University of Bologna, Italy

Rossella Cardone, Matteo Nocentini
Ericsson Telecomunicazioni S.p.A., Italy

Antonio Manzalini
Telecom Italia – Strategy and Innovation, Italy

***Abstract*—. This manuscript reports a Network Function Virtualization proof of concept demonstrating the added value that a dynamic Software Defined Networking control, coordinated with a flexible cloud management approach, brings to Telco operators and service providers.**

***Index Terms*— Software Defined Networking, Network Function Virtualization, Cloud Computing, Quality of Service Management**

I. INTRODUCTION

Techno-economic drivers are radically changing Telecommunications and ICT scenarios. In fact, the diffusion of fixed and mobile ultra-broadband connectivity, the increasing performance of general-purpose hardware architectures, and the growing availability of open-source software solutions are creating the conditions for a potential paradigm shift in designing, operating, and capitalizing on future network infrastructures [1].

Today we are witnessing a growing attention to Software-Defined Networking (SDN) [2] and Network Functions Virtualization (NFV) [3], which are based on two game-changing paradigms: on the one hand, SDN impose a clear separation between hardware (data plane) and software (control and management planes) functionality, resulting in a more flexible communication resource programmability; on the other hand, NFV fosters the adoption of virtualization and a "cloud-like" utility-based approach applied to networking, resulting in more elastic and efficient resource management. The rediscovered interest in said paradigms by Telco industry and service providers is most probably motivated by the novelty of the overall context, specifically concerning techno-economic sustainability.

This article reports some of the experimental achievements obtained in the framework of Activity "SDN at the Edges," funded in 2015 by the EIT-Digital[1] initiative under the "Future Networking Solutions" Action Line. The main goal of the Activity was to investigate how to create the technical conditions for accelerating the practical deployment of SDN and NFV solutions, in order to produce concrete socio-economic impacts. Specifically, one of the main arguments has been that a federation of interconnected test-beds/field trials on SDN and NFV would allow developing a distributed experimental environment where it is possible to test (and, possibly in the future, to certify) open-source software solutions in specific networking application use cases, at the same time attracting service providers, Small to Medium-sized Enterprises (SMEs), and early adopters/users of newly enabled ICT services.

The Proof-of-Concept (PoC) use case described in this manuscript leverages on Ericsson's cloud management and orchestration platform [4] and a *stateful* SDN framework developed at the University of Bologna [5]. The innovative aspect of the use case considered in the PoC is *the added value that a dynamic SDN control, coordinated with a flexible cloud management approach, brings to Telco operators and*

[1] http://www.eitdigital.eu

service providers deploying NFV solutions. In particular the PoC demonstrates how *Telco services based on NFV can be made self-adaptive to the network conditions and/or to the user changing requirements*.

The remainder of this article is structured as follows. Section II introduces the use case considered, Section III describes the Ericsson Cloud Lab environment, and Section IV discusses the scenario implemented in the PoC. Section V reports the results of the experiments, whereas conclusions are drawn in Section VI.

II. The Use Case

The general use case scenario refers to a NFV deployment in a cloud-based, Telco edge environment similar to the one discussed in [6]. The full lifecycle of a Telco service to be deployed in a virtualized environment is considered, from the set-up of a Service Level Agreement (SLA) with the operator, to the deployment of the Virtualized Network Functions (VNFs) required to implement the service, up to the operations performed to assure SLA compliance. While for the scope of this work a generic SLA could be considered, in the PoC presented later a SLA related to bandwidth availability will be assumed.

The goal and the original contribution of this work is to demonstrate full automation both in the cloud deployment of a given set of VNFs and in the capability to react to changes in the overall network conditions, safeguarding the SLA.

Typically, a given VNF *forwarding graph* [7] is implemented by properly chaining a number of VNFs, required to deploy a given service, and by steering the relevant traffic flows accordingly. Following the cloud paradigm, the VNFs are hosted in Virtual Machines (VMs) and are shared among a set of users for scalability and efficiency reasons. A VM may host from one to several VNFs, depending on the associated workload. The two extreme cases are (i) when a VM hosts all the VNFs required for the service implementation, and (ii) when each VNF is hosted by a separate VM. The former case is the least challenging from the connectivity point of view, as shown in Figure 1(a): since the VNF forwarding graph is mostly implemented inside a single VM, traffic steering is limited to sending flows to the VM and does not need complex forwarding rules in the cloud virtual network infrastructure. However, this choice provides limited elasticity in resource management. The latter case is more suitable for a Telco cloud-based NFV deployment, where the placement of VMs (and consequently of VNFs) is adaptively distributed across a number of computing nodes with the purpose to optimize performance and scalability. However, even in an intermediate case, as the one shown in Figure 1(b), the distributed approach requires to steer traffic along a data path that traverses multiple VMs in the proper order and/or combination, according to the forwarding graph. Therefore, the implementation of the forwarding graph must rely on more complex forwarding rules to be applied in the cloud virtual network infrastructure by means of a proper reconfiguration mechanism, possibly automated. This is where a dynamic SDN approach plays its crucial role.

The use case here presented tackles the most challenging situation of distributed VNFs, without lack of generality. The VM Images (VMIs) preconfigured with the installed VNFs are the basic *bricks* of the service implementation, while traffic steering is the *glue* to implement the forwarding graph. The most interesting feature is that the forwarding graph can be pruned, enhanced, and/or modified by starting/stopping VMs and/or by modifying the traffic steering rules, with a degree of flexibility and within a time frame that are absolutely impossible to achieve using legacy deployments based on physical middle-boxes and vendor-locked equipment. *The demonstration of these capabilities in a production-like environment is the core objective of this work, and this goal is achieved by implementing automated traffic steering and VNF forwarding graph management*. The basic concepts behind the use case presented here are in line with some of the PoCs proposed within the ETSI NFV framework[2]. In particular PoC #28 and PoC #33 address similar issues. These PoCs where developed approximately at the same time as the

[2] http://www.etsi.org/technologies-clusters/technologies/nfv/nfv-poc

activity reported in this manuscript. Moreover, the findings of the PoCs are not yet publicly available. Therefore this work can be considered complementary to the ETSI NFV use case analysis.

The non-trivial functional elements adopted in the considered use case include the following.

- A production-level cloud computing infrastructure, i.e., the Ericsson Cloud Lab described in Section III, equipped with a management and orchestration platform and an inventory of pre-defined VMIs to be picked up for deployment.
- An SDN controller, used to program the forwarding rules in the cloud network infrastructure in order to achieve proper traffic steering. The controller is designed according to an original stateful model developed at the University of Bologna, as described in Section IV.
- A separate entity capable of monitoring the network conditions and interacting with the SDN controller through a northbound interface. This monitor notifies the controller with any change in network conditions that is meaningful for the traffic flow maintenance. We assume the presence of such monitoring tools since they are typical components of network management solutions.

III. The Ericsson Cloud Lab environment

The Ericsson Cloud Lab [8] is an open and multi-vendor testing environment based on NFV and SDN technologies, where Telco operators can join Ericsson to perform PoC experiments and field trials for new use cases and cloud services. The Ericsson Cloud Lab is a strategic infrastructure conceived and implemented with the scopes to:

- foster in-company competence build-up;
- show specific and concrete “proof” points related to the cloud benefits;
- implement customer demo for specific products;
- demonstrate how issues and concerns can be managed mitigating the risks.

To support the fulfillment of such scopes, the Cloud Lab implements activities ranging from:

- Validation and Certification on Customer specific stack/solution
- Fully Customized PoC on Customer premises
- Deep Dive on Customer specific request
- Standard Customer Demo

Figure 2 shows the Ericsson Cloud Lab which follows the ETSI NFV architecture. It exploits a heterogeneous environment mixing hardware from the major vendors of data center servers. The virtualization hypervisors are also mixed, with KVM running aside VMware, and different Virtualized Infrastructure Managers (VIMs) are available. The whole infrastructure is managed by the proprietary Ericsson Cloud Manager (ECM) [4] on top of the VIM platforms. In the PoC presented here the VIM platform adopted is OpenStack [9].

The Cloud Lab allows the implementation of a workflow managing the full lifecycle of a specific service and of the set of VNFs used to implement it. In general terms the operational workflow implemented in the use case here presented can be summarized as follows:

1. *Service Creation*: create a new VNF and make it available in the correct format for future utilization (pack a VMI with the NFV properly installed and configured).
2. *Site and VIM Onboarding*: let ECM be aware of the geographical site and VIMs to be considered as targets for the management activity.
3. *VNF/Network Service Onboarding*: deploy a new VNF (or a new release of a VNF) into ECM, including creation of an offer in the service catalog, import of VMIs, etc.
4. *Instantiation and Configuration*: creation or provisioning of a new instance of the VNF including initial configuration (not customer-related).
5. *Runtime Management*: handling of fault, configuration, accounting, performance, and security (FCAPS) operations and in general of any change to the VNF.

6. *Decommissioning*: disposal or retirement of a VNF instance, including all ancillary activities (e.g., secure destruction of VNF-related data).

The specific setup of the Ericsson Cloud Lab for the use case considered here is shown in Figure 3. The virtualization infrastructure is running OpenStack Juno release and the KVM Hypervisor. The ECM hosts the inventory of VMIs with pre-installed VNFs and is responsible for the deployment of the logical building blocks necessary for the use case. The VNFs available into the inventory are:

- *DPI* (Deep Packet Inspection), a packet inspection utility (based on the nDPI library[3]) that is used to classify incoming traffic flows;
- *WANA* (WAN Accelerator), a compression utility (based on Traffic Squeezer[4]) that reduces the size of data packets to reduce transmission and waiting time along the WAN path, with an effect similar to an overall increase of bandwidth on the link;
- *TC* (Traffic Conditioner), a traffic shaping function (implemented with tools available in the Linux kernel[5]) that limits the bit rate of a given set of flows by enforcing the incoming traffic SLA profile.

An additional VMI (*SDNc*) is deployed to host the SDN controller (based on Ryu OpenFlow controller[6]) in charge of reconfiguring the underlying network infrastructure according to the different VNF forwarding graphs required by the users. This choice allows to virtualize the network infrastructure and enables multiple potential service providers to co-exist and separately control their slice of network resources. It is worth to mention that each VNF listed above is not an Ericsson application, which emphasizes the multi-vendor capabilities of the ECM and the Ericsson Cloud Lab.

IV. VNF Chaining for QoS Management and Dynamic Traffic Steering

The PoC presented here focuses on VNF chaining for QoS enforcement, and assumes two classes of users sharing an access network segment with different SLAs concerning bandwidth availability:

- *BU* is a *business (priority) user*: in case of adequate bandwidth availability the BU traffic follows traditional forwarding rules but, when the available bandwidth drops below a given threshold, the BU traffic is forwarded through a WAN acceleration service;
- *RU* is a *residential* (*best effort) user*: the RU traffic follows traditional forwarding rules until it starts competing with BU on network resources; in this case, traffic from the RU is policed by a shaping function, thus limiting its bandwidth usage to a given value.

The access network is based on an SDN control plane designed to:

- dynamically handle multiple simultaneous flows according to current network conditions;
- provide that some flows follow different forwarding graphs at different times;
- steer specific data flows toward the required VNF locations, thus achieving full dynamic service chaining.

To this purpose, for each traffic flow, the SDN controller configures a set of forwarding rules that determine the specific behavior to be implemented in the forwarding graph.

In order to devise a general approach to the dynamic traffic steering configuration according to current network conditions, the behavior of the SDN control plane is modeled by means of a *Finite State Machine (FSM)* where a pre-defined set of forwarding rules is associated to each state [5]. The SDN controller runs a thread for each flow f that maintains the related FSM and installs in the network the forwarding rules associated to its current state. The FSM state transitions may be triggered either by a change in the network conditions, detected and notified by a network monitor, or by an external command directly sent by the operator or user. This kind of communication reaches the controller through its northbound interface by means of an API written to the purpose.

[3] http://www.ntop.org/products/deep-packet-inspection/ndpi Accessed on July 1, 2016.
[4] http://www.trafficsqueezer.org Accessed on July 1, 2016.
[5] http://tldp.org/HOWTO/Traffic-Control-HOWTO/intro.html Accessed on July 1, 2016.
[6] https://osrg.github.io/ryu Accessed on July 1, 2016.

In the PoC presented here, the set of states is defined as follows:

- *Init* is the *initialization* state, in which the controller can install general, flow-independent forwarding rules; this is the state from which a new thread departs whenever a new flow *f* arrives;
- *C* is the *classification* state, in which the new flow *f* is analyzed and classified to determine its SLA;
- *E* is the *enforcement* state, in which QoS is strictly enforced on flow *f*, according to its SLA;
- *N* is the *non-enforcement* state, in which QoS is not strictly enforced on flow *f*, because the network is not in a congested state and the resources available guarantee full compliance with the SLA;
- *D* is the *drop* state, in which flow *f* is blocked (e.g., packets are dropped or marked as non-compliant) because it is not admissible in the network (SLA not satisfied).

When a new flow *f* arrives, a *PacketIn* message is received at the controller that spawns a new thread and enters state *C*, steering *f* to the VNF in charge of flow classification. Depending on classification result and current network conditions, the FSM state changes and flow *f* can be:

- not accepted in the network (*D* state);
- forwarded with no actions (*N* state);
- treated according to what required by the SLA to enforce the correct QoS (*E* state).

If the new flow is classified as SLA-compliant, a conservative approach would require to move it from state *C* to state *E*, thus enforcing the SLA for any new flow admitted in the network. Then, in case network conditions are favorable, the new flow can move to state *N*, where packets can be directly forwarded to the network gateway without further processing by VNFs enforcing the negotiated SLA. Otherwise, in case of critical conditions, all previously admitted flows move to the *E* state. As soon as the potential risk of congestion is solved (e.g., some flows terminate), active flows can progressively move back to state *N*. This is how the dynamic VNF chaining is accomplished, with proper traffic steering actions governed by the cooperation among the SDN controller, the network monitoring system and the VNF itself implementing the flow classification function.

The data plane virtual network topology implemented in the Ericsson Cloud Lab for the PoC described above is shown in Figure 4(a). Two Virtual Data Centers (VDC_1 and VDC_2) are assumed to be in place. VDC_1 hosts BU and RU instances and thus represents their access network, whereas VDC_2 hosts the destination to which all flows generated by BU and RU are directed (DEST acts as a sink for the PoC purposes). It is assumed that each VNF is implemented as a layer-3 virtual appliance connected to two separate broadcast domains, i.e., Virtual Networks $VN_{i,1}$ and $VN_{i,2}$ for VDC_i. Then a layer-3 Virtual Router (VR_i) is used to interconnect VDC_i to the external network. The provider maintains layer-3 connectivity to the user while deploying the required VNFs. This approach is typical of access/edge network operator deployments. So, the dynamic traffic steering and the SDN controller must be designed taking into account these transparency requirements.

In Figure 4(b) traffic steering in the *C* state is shown. When BU or RU initiate a new flow directed to DEST, the flow is forwarded from VDC_1 to VDC_2 through the gateways VR_1 and VR_2 but only after the SDN controller configures the VDC_1 internal network to force the flow through the DPI. This VNF is in charge of classifying the flow and detect its QoS class and compliance to the respective SLA. Following the classification, the DPI communicates to the controller via the northbound interface (not shown in the figure) the set of information required to identify the flow (e.g., MAC and IP addresses, protocol, TCP/UDP ports, etc.) and its classification (whether BU or RU). No manual intervention is required, as long as the DPI is made aware of any new user and related SLA at service setup. As a result of the classification, or in case network conditions change, the SDN controller changes the flow state.

When network conditions are such that the SLAs are not at risk, the controller configures the active flows into state *N*, resulting in a normal forwarding through the access LAN and through the gateway as shown in Figure 4(c). Otherwise, if a potential congestion risk is detected by the network monitor (not shown), the controller moves the flows to state *E* and injects new forwarding rules to the network. With the aim to enforce SLA compliance for both BU and RU, their flows are respectively forwarded to $WANA_1$ for improved QoS and to TC for bandwidth usage control, as depicted in Figure 4(d). The BU flow is also steered to $WANA_2$ in VDC_2 because compressed traffic should be restored to the original format before reaching DEST. This peculiar behavior has been included in the PoC to show how the stateful controller design is able to support interdependence between multiple VNF instances.

V. Experimental Results

The data plane virtual topology shown in Figure 4 was deployed on the Ericsson Cloud Lab environment, following the procedure and instantiating the VMIs as described in Section III. The logic to enforce the SLAs described in Section IV was programmed within the SDN controller. However, to simplify network reconfiguration in the PoC, but without any loss of generality, the service orchestration phase was implemented in a simpler and less conservative way, i.e.:

- as long as a new flow is classified as RU, and if network conditions are not critical, the flow enters state *N*;
- as soon as a BU flow is detected, network conditions are considered critical and all flows enter state *E*.

As a proof of the correct behavior of the dynamic traffic steering configured by the SDN controller according to the aforementioned orchestration logic, the throughput of four user flows (three generated by RU and one by BU) was measured on relevant ports of the OpenStack's virtual switch that implements the virtual networks instantiated in VDC_1. Each flow consisted in a distinct TCP stream generated using the *iperf* tool[7] with the following time pattern[8]:

- flow 1 from RU started at time T_1;
- flow 2 from RU started at time $T_2 = T_1 + 30$s;
- flow 3 from RU started at time $T_3 = T_1 + 60$s;
- flow 4 from BU started at time $T_4 = T_1 + 90$s;
- all flows were terminated at time $T_5 = T_1 + 200$s.

Figure 5 shows the total throughput measured on the virtual switch ports connected to BU and RU when (a) a static network configuration is used, or (b) the SDN-based dynamic traffic steering is used to enforce QoS. It is apparent how, without dynamic traffic steering, the available bitrate is completely used by RU flows until flow 4 from BU starts at T_4 ~100s; then the four flows equally share the available bitrate, with BU using only about one fourth of the capacity. However, when dynamic traffic steering is enabled, after a small interval when flow 4 is in state *C* (between ~100s and ~110s), the controller moves all flows to state *E* and the QoS enforcement takes place: the total throughput of the three RU flows is limited to 100 Mbps, whereas the BU flow throughput is not penalized.

Figure 6 highlights how the flows traverse the different VNFs when the dynamic traffic steering is applied. At the beginning, the three RU flows and the BU flow are processed by the DPI (state *C*). For each flow, the traffic steering to the DPI stops as soon as the classification is complete, which happens before the next flow starts. This results in four spikes interleaved by sudden drops of the throughput curve measured at the input of the DPI. After SLA compliance is determined for the three RU flows, since no BU flow is active yet, the RU flows are sent directly to VR_1 and the QoS is not enforced[9] (state *N*). However, after the BU flow is classified and its compliance to the priority SLA is detected (approximately at time T_1 + 110s), the QoS is enforced by properly steering all flows to the relevant VNFs (state *E*): the three RU flows are then sent to the shaping function TC, which limits their total bitrate to 100 Mbps, whereas the BU flow is processed by the $WANA_1$ (and $WANA_2$) functions, as proved by the throughput measured at the output of the respective VNFs.

[7] https://iperf.fr Accessed on July 1, 2016.

[8] This time pattern was chosen to provide maximum readability to the results presented in the following (see comments to Figure 6 for instance). However the system discussed can cope with general arrival patterns.

[9] For readability reasons, we do not show in Figure 5 the throughput measured at VR_1.

It is worth to note that, despite the bandwidth limitation applied to RU flows, the BU flow is not able to fully take advantage of the remaining capacity: this is due to the limited amount of computing resources in the OpenStack compute nodes used for the experiments, and the increased processing burden on the OpenStack virtual network infrastructure caused by the dynamic traffic steering [10]. The detailed evaluation of the trade-off between the performance improvement gained using dynamic traffic steering and the possible performance degradation due to cloud processing overload is beyond the scopes of this work and is currently under investigation. Indeed, it is highly dependent on the hardware configuration and, at least up to a certain degree, can be overcome by hardware improvement.

VI. Conclusions

SDN and NFV, together with cloud and edge computing, can be seen as facets of a broader systemic innovation wave (*softwarization*) which is transforming the overall telecommunications and ICT ecosystems. This innovation wave will create the conditions for a potential paradigm shift in designing, operating, and capitalizing on future infrastructures. This article reported some experimental achievements obtained as part of the EIT-Digital Activity “SDN at the Edges”, whose main goal was investigating how to create the technical conditions for accelerating the adoption of SDN and NFV in order to produce concrete socio-economic impacts. In general, it is argued that the ongoing “softwarization” of telecommunications (which goes beyond the adoption of SDN and NFV in core networks, reaching also terminals and clouds) will allow virtualizing all network and service functions and executing them in horizontal software platforms fully decoupled from the physical infrastructure.

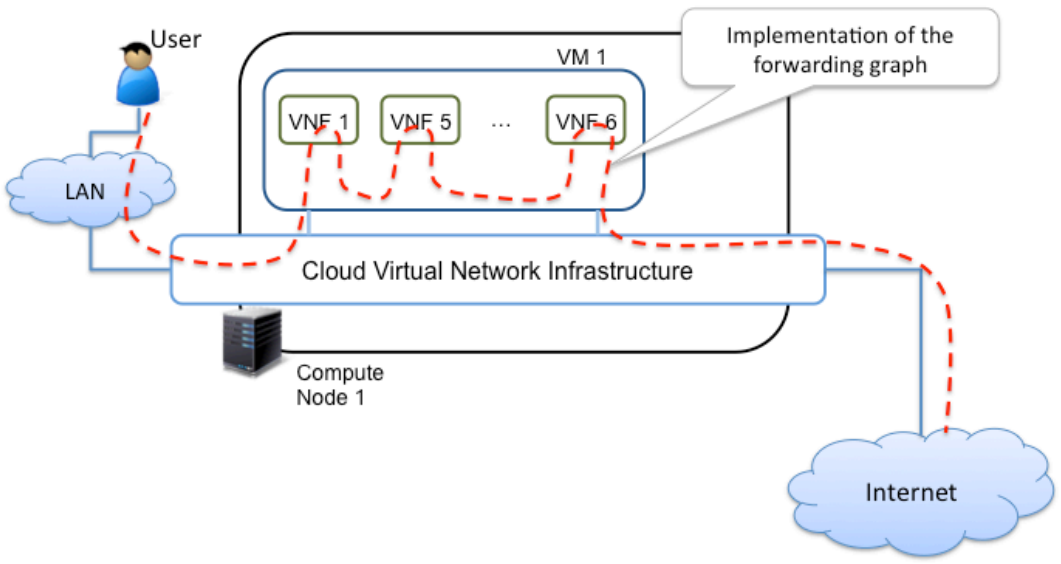
In this direction, this article reported the outcome of an experimental PoC aiming at demonstrating the capability of a software-controlled network to self-adapt by dynamically reconfiguring the VNF forwarding graph in order to guarantee the agreed SLAs to the user. The PoC was developed by leveraging on Ericsson’s cloud management and orchestration platform and a *stateful* SDN framework developed at the University of Bologna.

Acknowledgment

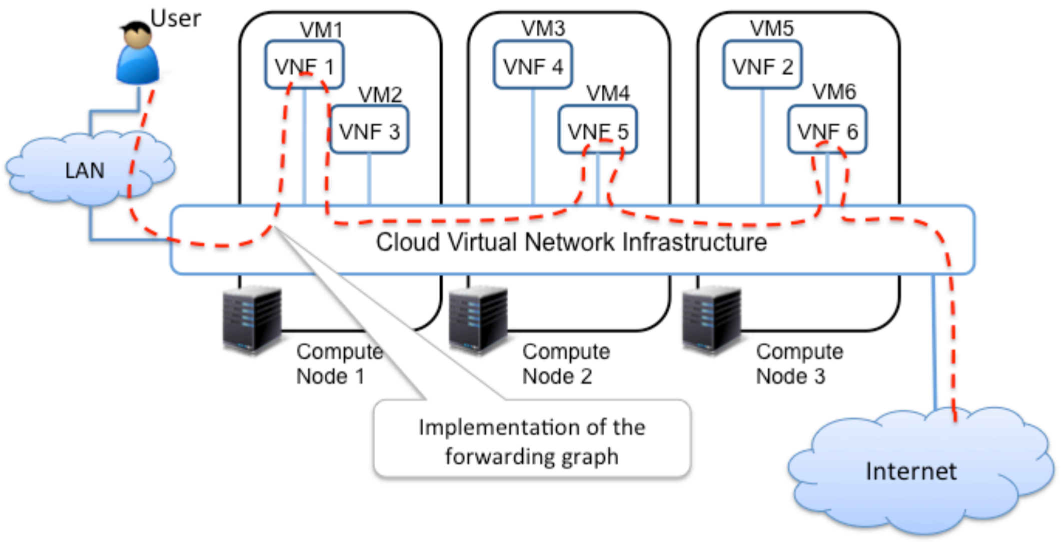
This work was partially funded by EIT-Digital, Action Line on Future Networking Solutions, Activity no. 15270/2015: “SDN at the Edges.”

References

1. J. Soares, C. Gonc¸alves, B. Parreira, P. Tavares, J. Carapinha, J.P. Barraca, R.L. Aguiar, and S. Sargento, “Toward a Telco cloud environment for service functions,” *IEEE Communications Magazine*, Vol. 53, No. 2, pp. 98-106, Feb. 2015.
2. F. Hu, Q. Hao, and K. Bao, “A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation,” *IEEE Communications Surveys & Tutorials*, Vol. 16, No. 4, pp. 2181-2206, 4th quarter 2014.
3. R. Mijumbi, J. Serrat, J. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, “Network Function Virtualization: State-of-the-Art and Research Challenges,” *IEEE Communications Surveys & Tutorials*, Vol. 18, No. 1, pp. 236-262, 1st quarter 2016.
4. Ericsson Cloud Manager [online]. Available: http://www.ericsson.com/ourportfolio/products/cloud-manager?nav=productcategory008 Accessed on June 14, 2016.
5. F. Callegati, W. Cerroni, C. Contoli, and G. Santandrea, “SDN Controller Design for Dynamic Chaining of Virtual Network Functions,” *Proc. of 4th European Workshop on Software Defined Networks (EWSDN 2015)*, Bilbao, Spain, October 2015.
6. A. Manzalini, et. al., "Clouds of virtual machines in edge networks," in Communications Magazine, IEEE , vol.51, no.7, pp.63-70, July 2013
7. ETSI Industry Specification Group (ISG) NFV, “ETSI GS NFV 001 V1.1.1: Network Function Virtualization. Use Cases,” October 2013.
8. Ericsson opens a cloud lab in Italy for faster co-creation and innovation, press release, June 26, 2015 [online]. Available: http://www.ericsson.com/news/1923781 Accessed on June 14, 2016.
9. OpenStack: Open source software for creating private and public clouds [online]. Available: http://www.openstack.org Accessed on June 14, 2016.
10. F. Callegati, W. Cerroni, C. Contoli, and G. Santandrea, “Performance of Multi-tenant Virtual Networks in OpenStack-based Cloud Infrastructures,” *Proc. of 2nd IEEE Workshop on Cloud Computing Systems, Networks, and Applications (CCSNA 2014)*, in conjunction with IEEE Globecom 2014, Austin, TX, December 2014.


User
VM 1
Implementation of the forwarding graph
VNF 1
VNF 5
...
VNF 6
LAN
Cloud Virtual Network Infrastructure
Compute Node 1
Internet

(a) VNFs deployed as software entities running in a single VM hosted by a compute node in the cloud. The requested forwarding graph can be mostly implemented inside the VM.



(b) VNFs deployed as software entities running in different VMs hosted by different compute nodes. The requested forwarding graph must be implemented in the cloud virtual network infrastructure.

Figure 1. Schematic graphical example of alternative implementations of the VNF forwarding graph.

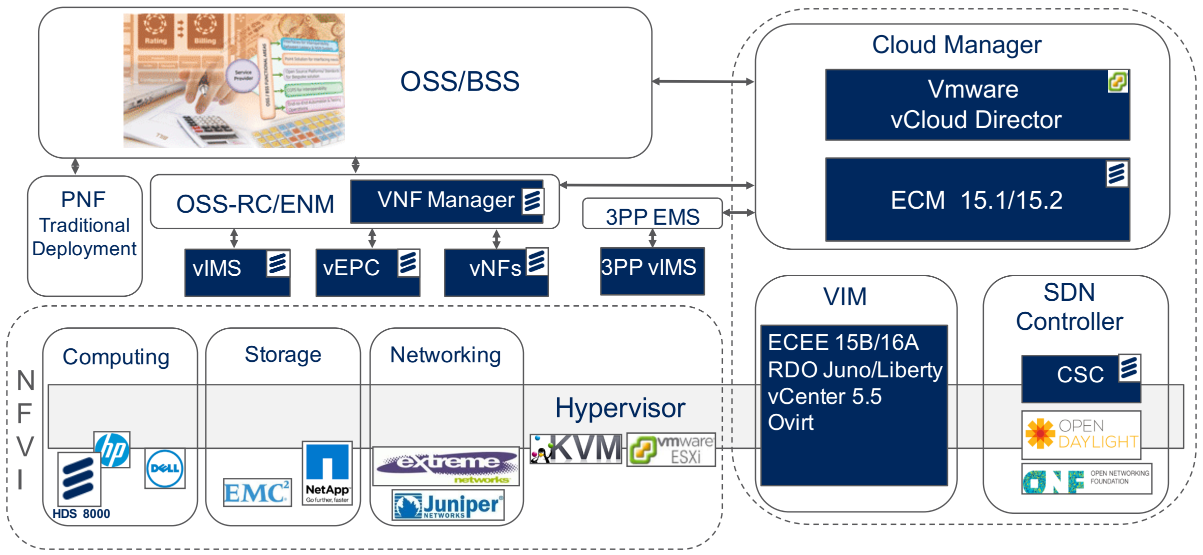


Figure 2 - The Ericsson Cloud Lab NFV architecture.

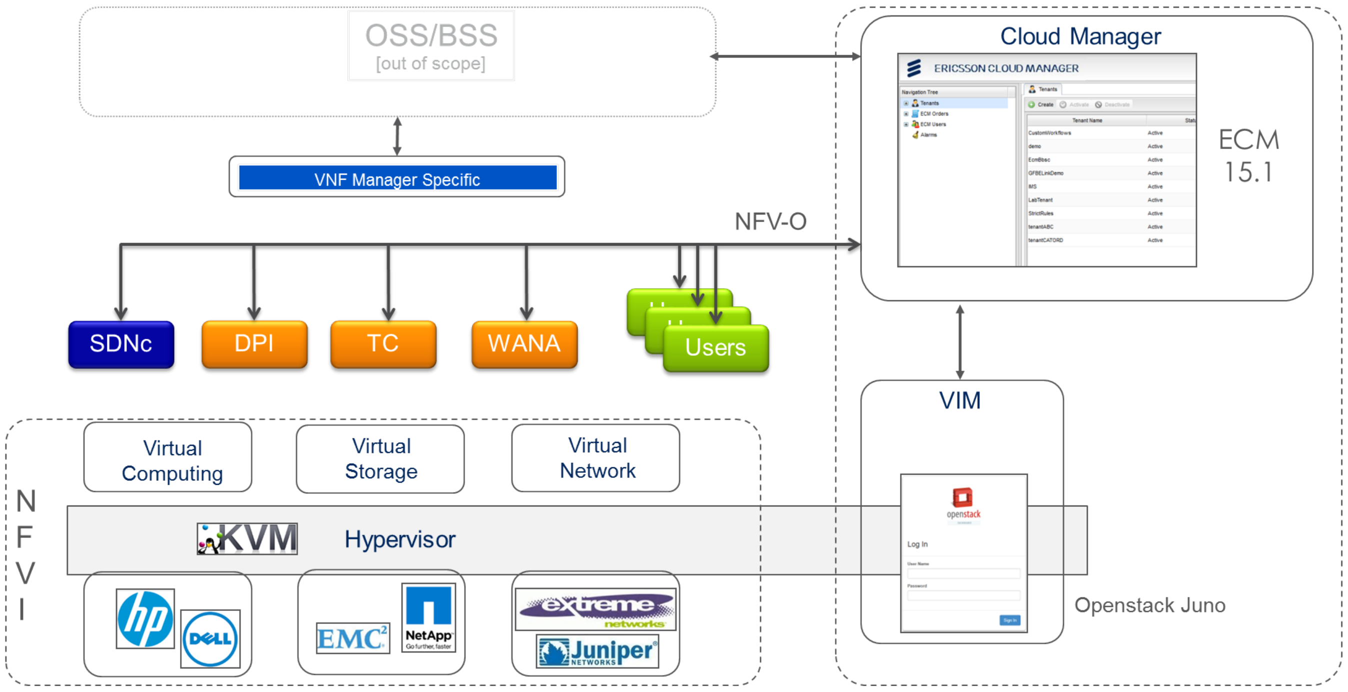
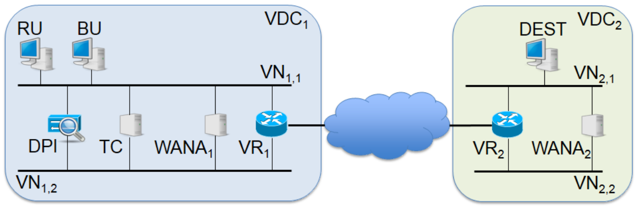
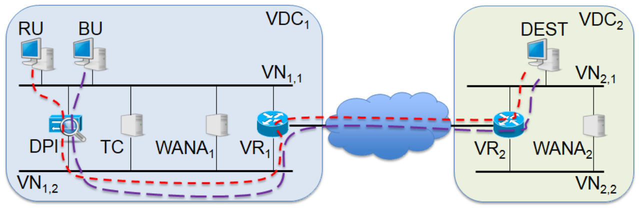


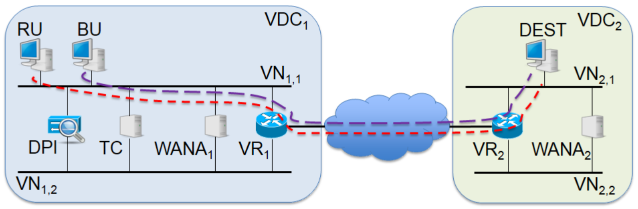
Figure 3 - Use case Ericsson Cloud Lab setup.



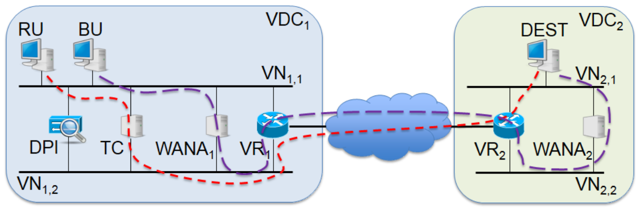
(a) PoC data plane virtual network topology.



(b) *C* state, classification of traffic flows.

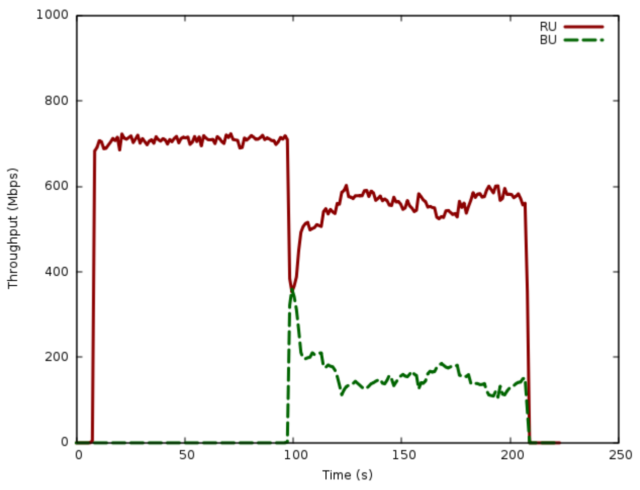


(c) *N* state, no QoS enforcement.

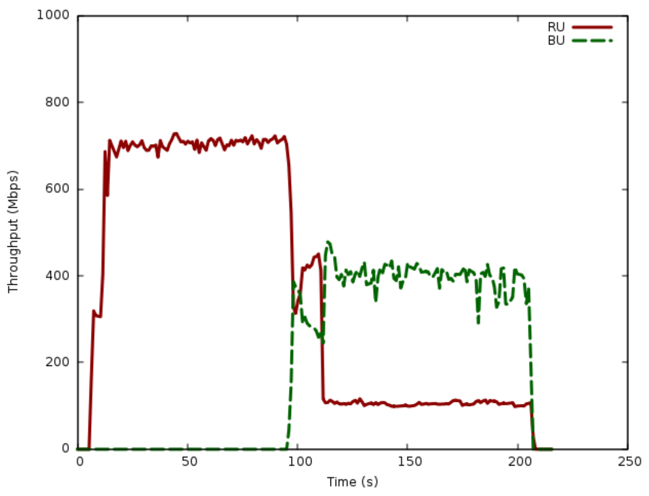


(d) *E* state, QoS enforcement with VNF chaining.

Figure 4 - PoC data plane virtual network topology and traffic flow steering in the different operating states.



(a) Without dynamic traffic steering.



(b) With dynamic traffic steering.

Figure 5 - Measured total throughput for RU and BU.

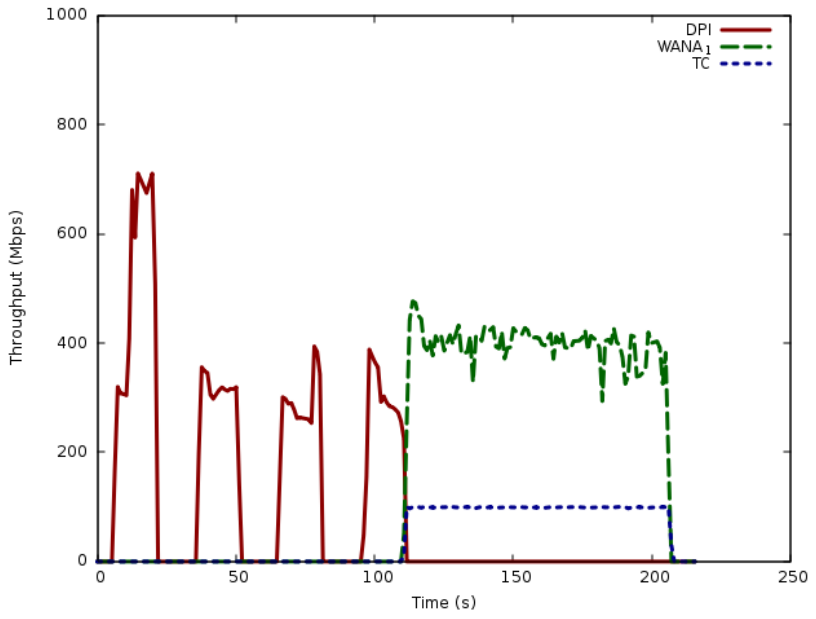


Figure 6. Measured throughput across the VNFs with dynamic traffic steering.

AUTHORS BIOGRAPHIES

FRANCO CALLEGATI [M'98, SM'11] (franco.callegati@unibo.it) is an Associate Professor of Telecommunication Networks at the University of Bologna, Italy. His research interests are in the field of teletraffic modeling and performance evaluation of telecommunication networks. He is currently working on performance evaluation and experimental validation of SDN/NFV based networking solutions. He has been active in EU-funded research projects since FP4, where he led activities and participated in various steering committees.

WALTER CERRONI [M'01, SM'16] (walter.cerroni@unibo.it) is an Assistant Professor of Communication Networks at the University of Bologna, Italy. His most recent research interests include: design, implementation and performance evaluation of virtual network function chaining in cloud computing platforms (e.g. OpenStack); modeling and design of inter- and intra-data center interconnection networks for cloud computing infrastructures; design of programmable, software-defined hybrid optical network architectures; performance evaluation of dynamic spectrum allocation techniques in flexible optical networks.

CHIARA CONTOLI (chiara.contoli@unibo.it) is a Ph.D. candidate at the University of Bologna Italy, where she graduated in Computer Engineering in 2013. She was research scientist at the Network Research Laboratory of the University of California at Los Angeles, where she worked on Content Delivery Networks for automotive applications to complete her Master thesis. Her research interests are in programmable networks, SDN, NFV and more generally on advanced networking architectures.

ROSSELLA CARDONE (rossella.cardone@ericsson.com) has Master degree in Mathematics, Master degree in Probability and Statistics, PMI certified. Over 18 years in Ericsson company, she has been product manager and total project manager for international projects in research, innovation and for product intro of large-scale telecom IP and MINI-LINK solutions. She was responsible for Ericsson R&D partnership strategy in Italy. Today she is Head of Innovation, Sustainability and Corporate Responsibility for Ericsson Region Mediterranean countries.

MATTEO NOCENTINI (matteo.nocentini@ericsson.com) received the B.Eng. degree in electronic engineering from the University of Perugia, Perugia, Italy in 2003. Since then, he has been dealing with IP Networking taking several roles at Siemens, Alcatel-Lucent and Ericsson in the pre and post-sales process. Being part of the Pre-Sales Organization, he is currently covering the roles of business development and technical sales of the Ericsson IP & SDN portfolio and solutions towards major SP's in Italy.

ANTONIO MANZALINI (antonio.manzalini@telecomitalia.it) received the M. Sc. degree in electronic engineering from the Politecnico of Turin, Italy. In 1990 he joined Telecom Italia Lab (formerly CSELT). He is currently Senior Manager at the Strategy & Innovation Dept. (Future Centre) of Telecom Italia. His current interests are mainly in the area of Software Defined Networks (SDN) and resources/functions virtualization (NFV), primarily for 5G. He is Chair of the IEEE initiative on SDN.