Towards a Telco Cloud Environment for Service Functions

João Soares[1], Carlos Gonçalves[2], Bruno Parreira[3,4], Paulo Tavares[3,4] Jorge Carapinha[3], João Paulo Barraca[4], Rui L. Aguiar[4], Susana Sargento[4]

[1]: Ericsson, Sweden, joao.monteiro.soares@ericsson.com
[2]: NEC Laboratories Europe, Germany, Carlos.Goncalves@neclab.eu
[3]: Portugal Telecom Inovação e Sistemas, Portugal, {bruno-m-parreira, paulo-c-tavares, jorgec}@telecom.pt
[4]: Instituto de Telecomunicações, Universidade de Aveiro, Portugal, {cgoncalves, jpbarraca}@av.it.pt {ruilaa, susana}@ua.pt

Abstract— **Deploying Service Functions (SFs) is an essential action for a network provider. However, the action of creating, modifying and removing network SFs is traditionally very costly in time and effort, requiring the acquisition and placement of specialized hardware devices and its interconnection. Fortunately, the emergence of concepts like Cloud Computing, Software Defined Networking (SDN), and ultimately, Network Functions Virtualization (NFV) is expected to raise new possibilities about the management of SFs with a positive impact in terms of agility and cost. From a telecom operator (Telco) viewpoint these concepts can help to reduce both Operational Expenditure (OPEX) and open the door to new business opportunities. In this article, we identify how Telcos can benefit with the abovementioned paradigms, and explore some of the aspects that still need to be addressed in the NFV domain. We focus on two major aspects: enabling Telco infrastructures to adopt this new paradigm; and orchestrating and managing SFs towards Telco-ready cloud infrastructures. The technologies we describe enable a Telco to deploy and manage SFs in a distributed cloud infrastructure. In this sense, the platform Cloud4NFV is presented. Special attention is given to the way SFs are modeled towards cloud infrastructure resources. In addition, we explore the ability to perform Service Function Chaining (SFC) as one of the fundamental features in the composition of SFs. Finally, we describe a Proof of Concept (POC) that demonstrates how a Telco can benefit from the described technologies.**

Keywords—Service Function, Service Function Chaining, Network Function Virtualization, Software-Defined Networking, Cloud Computing.

I. INTRODUCTION

The emergence of the cloud concept, its ongoing evolution and the opportunities that it brings, has led many businesses to adapt in order to get the most utility out of it. One can say that the Telco sector is today one of the most active business sectors exploring the opportunities offered by the cloud. The relationship and inter-dependency between clouds and telecommunications can be analyzed from two distinct perspectives:

- **Telcos *supporting* the cloud**: In a cloud environment, communication end points are user devices and Virtual Machines (VMs) that can be hosted in different physical locations, according to varying conditions. Compared to traditional networking environments, network capacity requirements are no longer static, but are likely to change as the associated computing and storage resources expand and reduce. This poses a whole new set of challenges to the network, now jointly including the Data Center (DC) and the Wide Area Network (WAN) segments. To provide assured levels of performance to cloud services, cloud and Telco services need to be provisioned, managed, controlled and monitored in an integrated way.
- **Telcos *using* the cloud**: Today, the establishment, management and composition of SFs (e.g. router, firewall) follow a rigid, static and time consuming process – e.g. resource overprovisioning is usually necessary to cope with estimated peak demand; a fault in a single function can disrupt an entire network, imposing the need for faster disaster recovery methods. As virtualization technologies reach maturity and are able to provide carrier-grade performance and reliability, it becomes feasible to consolidate multiple network equipment types, traditionally running on specialized hardware platforms, onto industry standard hardware, which minimizes costs, reduces time-to-market and facilitates open innovation. Cloud Computing, coupled with SDN [1] and NFV [2], promises to make SF management processes much more agile. Cloud Computing represents a paradigm for Information Technology (IT) services, which can now be delivered in an on-demand and self-service manner. SDN brings new capabilities in terms of network automation, programmability and agility that facilitate the integration with the cloud. On the other hand, NFV, from a high-level perspective, accelerates the innovation of networks and services, allowing new operational approaches, novel services, faster service deployment (shorter time to market), increased service assurance and stronger security.

Conceptually, a SF is a functional block responsible for a specific treatment of received packets and has well-defined external interfaces [3]. A SF can be embedded in a virtual instance or directly in a physical element (the usual situation until recently). Virtual SFs offer the opportunity to compose and organize virtual SFs dynamically, opening a new set of business opportunities – and technical challenges. One of the topics that arise from the combination of SFs is service function chaining (SFC). SFC is loosely defined as "*an ordered set of service functions that must be applied to packets and/or frames selected as a result of classification*"

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.

[3]. It can be considered as a particular case of service composition. It requires the placement of SFs and the adaptation of traffic forwarding policies of the underlying network to steer packets through an ordered chain of service components. However, the lack of automatic configuration and customization capabilities increases the operational complexity.

In this article, we explore how telecom operators can take advantage of the referred concepts to improve the management of SFs and potentially build new business models. First, we highlight the Telcos privileged position in this area compared to traditional cloud providers. We then present Cloud4NFV, a platform for managing SFs in a Telco cloud environment. Later, we focus on SF modeling towards cloud infrastructure resources. Special attention is given to the ability to perform SFC. To emphasize possible application scenarios of the solution presented in this paper, a POC is then detailed. Finally, we point out future work directions and conclusions.

II. The Carrier Cloud Opportunity

Traditional cloud infrastructures are far from being suitable for all types of businesses, especially when referring to network SFs. Most network SFs have carrier grade requirements, from guaranteed Quality of Service (QoS) in terms of IT resources and network connectivity, to high-availability (e.g. perform detection and forecast of operational anomalies, support fault mitigation procedures such as VM migration and network re-planning) and fast fault recovery through redundancy.

Telcos, with their already established distributed network infrastructure and hosting centers, are ideally positioned to take the lead in this area, as they can easily create a compelling end-to-end cloud proposition that integrates their network management capabilities, adapted to a more agile and cloud service-oriented operation model (on-demand, self-service, elastic).

We envision a near-future Telco cloud infrastructure that comprises not only the traditional centralized DC domains, but also the WAN domain. In such scenario, the Telco can take advantage of its already established distributed facilities (sometimes referred as Points of Presence, PoPs) to host small cloud environments. It is also possible for this distributed cloud infrastructure to extend itself into the customer site. Figure 1 depicts this scenario.

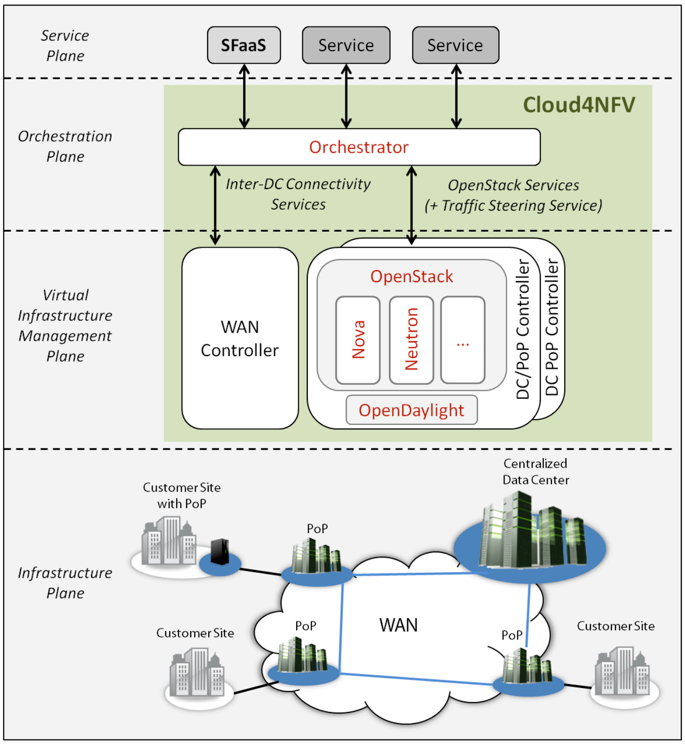

Figure 1: Cloud4NFV platform – overview

Although there are important contributions ongoing in this area, work is still required, namely when it comes to the definition of a true Telco cloud platform and to the details on how to model and actually realize SFCs. Table 1 presents a summary of the features supported by some existing solutions that more closely relate to the scope of this work. The information presented reflects the publically available information at the time of writing, and maybe meanwhile superseded. The recent UNIFY [4] and T-NOVA [5] projects seem to share a similar vision; however, these projects have recently started and have only provided conceptual approaches to some extent (the features analysis was done based on

	Cloud4NFV	UNIFY [4]	T-NOVA [5]	CloNe [6]	StEERING [7]	CloudBand[1]
Distributed Infrastructure	Yes	Yes (conceptually)	Yes (conceptually)	Yes	No	Yes
End-to-End Service Management	Yes	Yes (conceptually)	Yes (conceptually)	Yes	No	Yes
Multi-domain Architecture	Yes	Yes (conceptually)	Yes (conceptually)	Yes	No	Yes
Network QoS support	Yes	Yes (conceptually)	Yes (conceptually)	Yes	No	Yes (partially)
SF Management	Yes	Yes (conceptually)	Yes (conceptually)	No	Yes (partially)	Yes
Traffic Steering support	Yes	Yes (conceptually)	No	No	Yes	No
SFC support	Yes	Yes (conceptually)	No	No	Yes (partially)	No

Table 1: Summary of existing approaches

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.

documents publicly available). CloNe [6] has support for the infrastructure features; however, it lacks SFs management, traffic steering, and performing SFC. Moreover, StEERING [7] supports traffic steering and partially supports SFC and SF management ("partially" because the SFC service model and SF management features do not seem fully mature). Finally, we also consider the Alcatel Lucent CloudBand[1] solution, which supports some of the envisioned features.

III. Cloud4NFV Platform

The Cloud4NFV platform builds upon Cloud, SDN and WAN technologies to allow SFs to be managed on an *as-a-Service* basis. The platform is targeted for Telcos to improve the management of SFs within their environment, but can also be used to build new services based on the concept of *Service Function as-a-Service* (SFaaS), in which case SFs or bundles containing a combination of SFs can be offered as a service to customers.

A. Functionalities

The most relevant functionalities of Cloud4NFV are:

- Automated deployment, configuration and lifecycle management (e.g. instantiation, configuration, update, scale up/down, termination, etc) of SFs.
- Exposure of functionalities such as: service deployment and provisioning; service monitoring and reconfiguration; and service teardown.
- Federated management and optimization of WAN and cloud resources for accommodating SFs.
- Support of SF composition through SFC.

All the above mentioned functionalities are essential in the scope of an NFV platform; however, we highlight the last two due to their novelty. These two functionalities are seen as key differentiation factors from other available solutions, taking this platform closer to being fully carrier-grade compliant. The federated management and optimization of WAN and cloud resources, gives the platform a broad and distributed scope. It allows the establishment of end-to-end services over a distributed physical infrastructure. The ability to perform SFC gives the platform an unprecedented flexibility with respect to the SF management and composition, allowing the definition and establishment of advanced services in a much more efficient and flexible way.

B. Architecture

Figure 1 provides an overview of the system, organized in four major planes: Infrastructure Plane, Virtual Infrastructure Management (VIM) Plane, Orchestration Plane, and Service Plane. The Service Plane handles the services that are built on Cloud4NFV, and the Infrastructure Plane comprises all physical resources. Special attention should be given to the VIM and Orchestration Plane, since we consider them to be the major lever for enabling SFC. It is important to note that this architecture is aligned with the ETSI NFV architectural guidelines [8]. This fact is highlighted along the description of the platform.

C. Orchestrator

The Orchestrator is responsible for the automated provision, management and monitoring of SFs over the virtual infrastructure. It exposes the ability to create and delete SFs, as well as the ability to chain SFs. It relies on the VIM Plane to provision the infrastructure resources where SFs run (VMs, virtual networks, etc). Looking to the ETSI NFV reference architectural framework [8], this component considers the *Orchestrator* and the *VNF Manager(s)* entities. The orchestrator has an interface (REST) that exposes the ability to create and delete SFs as well as to chain SFs.

D. Virtual Infrastructure Management Plane

The VIM Plane includes the components for management of infrastructure resources. It includes cloud DC controllers (one per DC) and a WAN controller that is able to establish inter-DC connectivity services. The VIM Plane can be seen as the *Virtual Infrastructure Manager(s)* in the ETSI NFV reference architectural framework [8]. However, the current ETSI specification does not have into consideration the WAN component. This is considered by ETSI to be subject of future analysis.

1) Data Center Controller(s)

Although the cloud model may require, to a large extent, the redefinition of SFs and the way they are managed, SFs also require adaptation from today's cloud solutions to cope with their requirements, especially in terms of networking features. A clear evidence of this fact is the OpenStack[2] project, a reference open-source cloud management platform, which has been witnessing a tremendous evolution of its networking features - in its networking project mostly known by the codename *Neutron*. It is also important to note that *Neutron* provides network service logics, and relies on different backends called "drivers" to interact with different networking technologies. Among these drivers is the recent OpenDaylight[3] SDN controller. OpenDaylight is today seen as an initiative equivalent to OpenStack in the SDN domain. With this in mind, our DC Controller is based on OpenStack and OpenDaylight.

i) OpenStack: from a networking perspective, OpenStack allows the creation and management of *networks* (L2 network segments) and *ports* (attachment points for devices connecting to networks, e.g. virtual Network Interface Cards (vNICs) in VMs). The OpenStack community has been doing a considerable effort keeping up with users' demand on introducing new *Neutron* network service types - L3 routing, firewall as a service (FWaaS), load balancer as a service (LBaaS) and VPN as a Service (VPNaaS); however, it is unfeasible (and probably unwise) in the long run to keep up with demands at this pace in a timely manner. Therefore, we argue that OpenStack should focus in offering the basic tools

[1] CloudBand, http://www.alcatel-lucent.com/solutions/cloudband

[2] OpenStack, http://www.openstack.org/

[3] OpenDaylight, http://www.opendaylight.org/

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.

for network services to be orchestrated at a higher level and be deployed as VMs.

With the orchestration and composition of SFs in mind, it is easy to identify the need to fill a gap in OpenStack, the one of steering traffic between OpenStack elements (e.g. VMs, routers). We envision a new OpenStack service abstraction that extends and relies on current OpenStack networking features, allowing traffic steering between *Neutron ports* according to classification criteria. New entities are introduced into the OpenStack *Neutron* data model: *Port Steering*, and *Classifier*. Both entities have a set of common OpenStack data model attributes, i.e. *id*, *name*, *description*, and *tenant_id*. The *Port Steering* adds to this common set a list of ports (*ports* attribute), and a list of classifiers (*classifiers* attribute). The former lists the sets of ports that must be targeted of classification and then steered. The *Classifier* entity adds the following attributes: *type*, *protocol*, *port_min*, *port_max*, *src_ip* and *dst_ip*.

This functionality is very useful as it provides the means to realize, among other things, SFC, as described in Section V. Furthermore, the primitive is seen as a foundation for future (higher-level) abstractions within OpenStack.

ii) OpenDaylight: OpenDaylight has a module that integrates with OpenStack *Neutron* for the enforcement of services in the infrastructure. This module was extended in order to support and enforce the previously referred OpenStack traffic steering feature. It is important to highlight that this implementation relies on OpenFlow and Open vSwitch Database Management Protocol (OVSDB) for the management of network resources.

2) *Wide Area Network Controller*

The WAN Controller is responsible for managing the operator network, and it exposes connectivity services to the upper layers (in this case the orchestrator). In this context, WAN services are used to support SFs (SF is the client of the WAN service). Point-to-point and multi-point connections with guaranteed network QoS are provided. These are exposed through a service interface that, similar to cloud IaaS interfaces, is technology agnostic. The details and mechanisms to manage the automatic establishment of connectivity services across different locations are detailed in [6].

IV. Service Function Virtualization

This section elaborates on how SFs are modeled towards virtual infrastructure resources. Figure 2 depicts the correspondent data model and each class is detailed below.

Service Function: represents an instance of a functional block responsible for a specific treatment of received packets.

Service Function Endpoint (SFE): represents an external interface of one SF instance that is always associated to a SF. Each SFE can have associated information regarding layer 1 (e.g. physical/virtual interface), layer 2 (e.g. MAC address) and/or layer 3 (e.g. IP address), or even regarding higher layers (e.g. HTTP).

From an infrastructure perspective, the resources considered to realize a SF are: *Compute Instance* (i.e. Virtual or Physical Machines), *Image* (disk image), *Compute Flavor* (hardware specification of a compute instance, i.e. CPU, memory and root disk), *Block Storage* (additional disks), *Port* (i.e. network interface), *Network* (a network segment), and *Link* (a connection between two *Ports* from different *Compute Instances*) which has an associated *Link Flavor* (dedicated QoS in terms of bandwidth, delay and jitter). A SF can be associated to multiple *Compute Instances*, while each *Compute Instance* has a single *Image*, a single *Flavor* and can have multiple *Ports* and *Block Storages*. A *Port* can only be associated to a single *Network*; however, it can be associated to multiple *Links*. A SFE is directly associated to a *port*, but not all *ports* need to map to SFEs.

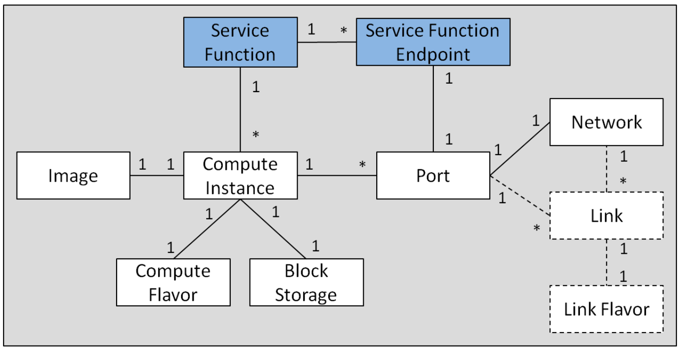

Figure 2: Service Function data model towards a cloud infrastructure

The network QoS, represented in the model by *Link* and *Link Flavor*, is not considered in today's cloud infrastructure systems. However, for a carrier grade cloud this is a must, and OpenStack already has an ongoing project to support it[4].

Figure 3 presents an example of how several SFs can be composed and organized. Furthermore, it also highlights how SFCs can be built and explored.

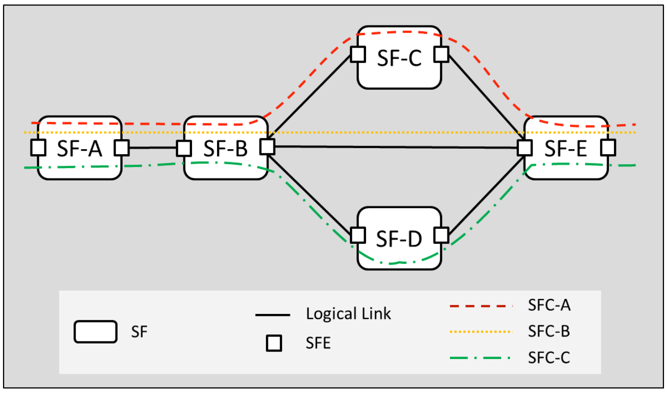

Figure 3: Service Function composition - example

V. Service Function Chaining

In this section we provide insights on the fundamentals and modeling aspects of SFC proposed in this work.

A. *Fundamentals*

In SFC two aspects are vital:

[4] OpenStack Neutron QoS support https://wiki.openstack.org/wiki/Neutron/QoS

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.

Classification: a policy for matching packets (e.g. *HTTP* traffic) used for the identification of appropriate actions (e.g. forwarding). It can be for example an explicit forwarding entry in a network device that forwards packets with a specific IP or MAC address into the SFC. (Re)Classification can also occur at each SF of the SFC independently from the previous SFs. In such cases, multiple classification policy entries should be allowed in an SFC system.

Traffic Steering: ability to manipulate the traffic route at the granularity of subscriber and traffic types [7]. The actual network topology should not be modified to accomplish this.

Moreover, the combination of classification and traffic steering can be done in two ways:

Tagged packet approach: classification can occur only at the initial redirection points to a SFC, if upon this classification packets are tagged. After that, packets are steered to the SFC and routed along it according to the embedded tags.

Non-tagged packet approach: classification occurs not only at the redirection points but also at each hop of the SFC. In this case, packets are not tagged and are subject of classification and steering at each SFC hop.

The consequences of following a tagged or non-tagged packet approach are felt at the VIM Plane level. One of the benefits is that this choice is relatively well isolated from the higher planes. We believe the non-tagged approach to be the smoothest approach to follow due to its lower impact on SFs and virtual infrastructure management systems. The advantage of the tagged approach is that the traffic only needs to be classified and tagged (e.g. with a VLAN tag, or another tag) once along the entire SFC. The drawback is the fact that the SFs need to know how to handle the tags (in the simplest case, they should at least ignore them). Although, we can add in the platform the support for a tagged approach (e.g. classify only at one point, tag, and steer traffic according to tag), it will only make sense if there is also support at the SF level. In this sense, in this work we adopt the non-tagged packet approach.

Further aspects should be taken into account when elaborating a SFC solution, such as: i) no assumption should be done on how functions are deployed, i.e. whether they are deployed on physical hardware, as one or more VMs, or any combination thereof; ii) a SF can be part of multiple SFCs; iii) a SF can be network transport independent; iv) a SFC allows chaining of SFs that are in the same layer 3 subnet and of those that are not; v) traffic must be forwarded without relying on the destination address of packets; vi) classification and steering policies should not need to be done by SFs themselves [10].

B. *Service Function Categories*

Two categories of SFs have been defined.

Active SFs: those that are in fact part of the main course of a packet, in which case two sub-types are considered: a) functions that may drop packets or forward them, such as a Firewall; b) functions that can actually change packets, e.g. an IPSec VPN server.

Passive SFs: are considered to be out of the main course of the chain. These functions mainly inspect packets, e.g. a monitoring system or a DPI. In practice one can think of a SF in a physical device connected to a hub through a single network interface configured in promiscuous mode. Traffic is considered to be duplicated when having to reach a passive function.

These two categories are important because they impose constraints on how classification and steering can be implemented. In short, passive functions can rely on packet characteristics as packets are not modified, while active functions must be integrated at a service level because ingress and egress packets can be different (e.g., VPN). If a SFC has active functions that change packets, the classification may differ when passing one of these functions.

C. *Service Function Chaining Abstraction Model*

The ability to classify and steer traffic accordingly can be enough to implement at the low level the SFC functionality. However, it is important not to forget that the traffic steering functionality is a low level functionality that does not explicitly express a SFC. Having in mind the considerations made so far, a base data model for SFC (that supports both tagged and non-tagged approaches) is now presented. Naturally, other SFC service abstraction proposals may appear in the future, but we consider that this model lays a strong foundation over which other service abstractions can easily be created by extending the model. Figure 4 depicts the model. Five main classes are considered: *Service Function Chain*; *Service Function; Service Function Endpoint; Packet Flow* and *Classifier*.

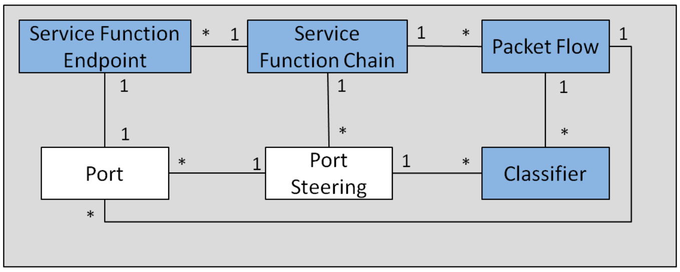

Figure 4: Service Function Chain data model towards a cloud infrastructure

All classes have the following attributes: *id*, *name*, and *description*. The *id* refers to a unique identifier able to identify the class instance within the SFC system. The remaining two, *name* and *description*, are attributes that allow a human-readable characterization of the class instance. Below it is provided further detail about each class.

Service Function Chain (SFC): a SFC has a set of *Service Functions* (SFs) associated and an attribute that defines the ordered sequence of functions (*path*). Since a function can have more than one SFE, the *path* attribute is specified by an ordered list of SFEs organized by hops. For example:

- "*path= { hop={SF-A_E2, SF-B_E1}; hop={SF-B_E2, SF-D_E1}, passive={SF-C_E1} }*"

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.

where the chain crosses SF-A, SF-B, and SF-D and has SF-C as a passive function between SF-B and SF-D.

Classifier: a *classifier* represents a classification criteria applied to a packet, which determines if the packet matches that specific criteria or not. In this sense, a *classifier* has an attribute *filter* that contains the classification criteria, e.g.:

- "*filter={protocol='6'; port='80-90'; source_IP='192.168.10.20/32'; destination_IP='192.168.10.40/32'}*" –

matches all TCP traffic using ports between 80 and 90 with source IP address 192.168.10.20 and destination IP address 192.168.10.40.

Packet Flow: One *classifier* only identifies packets with a certain criteria, while a *packet flow* identifies a broader set of packets as it can aggregate packets associated to multiple *classifiers*. In this sense, a *packet flow* can have multiple *classifiers*, and a *classifier* can be associated to multiple *packet flows*. Moreover, a *packet flow* has a *source* and *destination port*. The former identifies where the initial classification and redirection of the packet flow to the SFC takes place, while the latter identifies where packets are to be delivered after passing through the SFC. The attributes considered so far would be enough if the system realizing the SFC followed a tagged packet approach. For a non-tagged approach, an additional attribute is considered - *sfc_classifiers*. Due to the possibility of (active) SFs to modify packets, the classification initially done may not be the same along all hops of the SFC, and therefore, the *sfc_classifiers* attribute matches the classification criteria (*classifiers*) at each hop of the SFC. Furthermore, the attribute *direction* is also considered to identify the direction of the SFC which the *packet flow* must traverse – this attribute can assume one of two values: *forward*, *reverse*. We consider that multiple *packet flows* can be associated to a single SFC instance.

Port Steering: this entity refers to the functionality presented above in OpenStack. This feature allows steering traffic between *ports*. Further details about the traffic steering functionality can be found in the OpenStack proposal[5], for which we developed a prototype implementation.

In terms of operations, all classes are considered to allow CRUD (Create, Read, Update and Delete) operations.

VI. PROOF OF CONCEPT

A POC environment has been deployed to showcase how a Telco can leverage the features described in this work. We highlight one of the most attractive use-cases in the NFV scope and how it has been realized in this POC.

The testbed in place is depicted in Figure 5, focusing on the PoP setup that is detailed further ahead. At the core of the operator network (Telco Core Network) there is an IP/MPLS backbone composed of four provider (P) routers and four provider edge (PE) routers. The core network is managed by proprietary OSSs that expose connectivity services through a service interface in a technology agnostic manner (the WAN controller). The core network connects to two DC premises (managed by OpenStack *IceHouse* release with the traffic steering functionality), one of which represents a centralized DC and the other a PoP. Finally, the customer premises are represented by switching equipment, which is logically connected to the PoP over an access network (a simple switch based network).

A prototype of the Cloud4NFV orchestrator, which interacts with the WAN and DC controllers, was developed using the Python language. Details regarding the orchestrator implementation (e.g. RESTful API) can be found in [11].

A. *Customer Premises Equipment Use-Case*

Customer Premises Equipment (CPE) are often pointed out as one of the most suitable candidates SFs for virtualization [2] [12]. SFC will surely play a particularly relevant role in this case.

The CPE can be seen as a standard routing node enhanced by collection of SFs, such as Network Address Translation (NAT), Firewall (FW), Voice over IP (VoIP) servers, Virtual Private Network (VPN) servers, Network-Attached Storage (NAS), WAN Optimization Controllers (WOC), Deep Packet Inspection (DPI) or Intrusion Prevention System (IPS). These services are deployed for different scenarios, and not all traffic needs to traverse them, leaving room for optimization through SFC. It should be noted that some of the chains can even be temporary, which requires a model that enables the dynamic definition of chains.

B. *Service Function as a Service (SFaaS)*

At the service layer we implemented a prototype of the SFaaS concept. This is exposed via a web portal. CPE functions are available in the SFaaS, and the ability to perform SFC is not exposed to the end-user. The user requests CPE SFs, which already have a pre-determined relation with other SFs, and associates them to one of his sites. The instantiation and configuration of SFs is done in a timescale of seconds/minutes (depending on the SF/VM and cloud infrastructure) and the SFC enforcement in a timescale of seconds. Furthermore, the user is able to control them through a dedicated SF management portal.

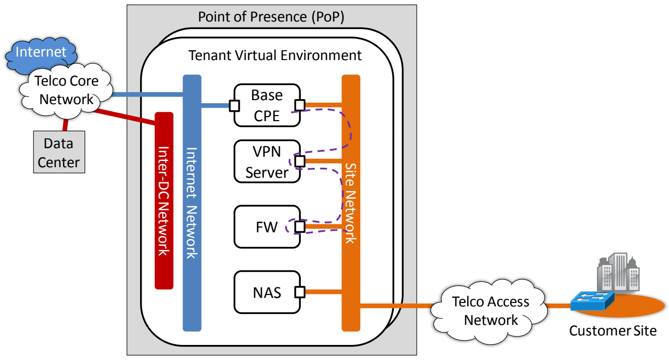

Figure 5: POC prototype setup

Note that the use of the SFaaS service requires a basic business relationship between the customer and the Telco (customer sites registered and with connectivity services). In

[5] OpenStack Traffic Steering blueprint, https://review.openstack.org/#/c/92477/

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.

other words, the user is a customer of the Telco who provides connectivity services (e.g. fiber, copper) from the client's sites (e.g. house, enterprise premises). On the site side, a L2 device (or a L3 device in bridge mode) is considered to be in place.

Currently, from a demonstration viewpoint, it is considered that the user, after having the physical connection in place, must first buy a base CPE function with routing, DHCP and NAT functionalities (the POC relies on the OpenStack L3 native device). From that moment on, the user can acquire other CPE functions and services, e.g.: Internet connection, Firewall (POC relies on *iptables*), VPN Server (POC relies on *OpenVPN*), NAS (POC relies on *Samba*) and others.

C. Prototype setup

Figure 5 depicts the POC prototype setup with four functions as an example. Special attention is given to the setup at the PoP level.

Each customer has a dedicated virtual private environment in the PoP that is serving his site. This environment allows the creation of virtual networks and VMs (in OpenStack this is known as *tenant* or *project*). There is a point to point logical connection between the customer's premises (L2) device (currently we are using VLAN encapsulation to establish this connection, but others can be used). On the PoP side this logical connection is extended to a virtual network in the tenant virtual environment – in the figure "Site Network", which has a private IP range (the OpenStack *provider network* concept is used to achieve this). Moreover, there is a virtual network that is shared among all tenants, which in the figure is the "Internet Network" (in OpenStack this is achieved using the *external network* concept). This latter network is then connected to the core network which provides the Internet access. Also depicted in the figure is the "Inter-DC Network" which provides access between the PoP and the DC over a Telco VPN service in the core network (again, on the PoP side we rely on the OpenStack *provider network* concept to connect to the VPN). The processes explained so far are considered to be in place as soon as the customer establishes the basic business relation with the Telco.

All functions, when deployed upon request, are connected to the "Site Network". When an Internet connection is requested, the base CPE is connected to the "Internet Network" and configured to perform NAT. The figure also highlights a SFC that comprises the base CPE, VPN server and firewall.

VII. Future Work

Currently, the POC does not support the enforcement of network QoS in DC domains; this is only supported in the WAN connectivity services. We expect to add this support by the time OpenStack officially releases this feature. Furthermore, runtime management operations (such as scaling and migration of SFs) are yet to be included in the platform. On the WAN domain, we are currently adding a SDN-based network. The purpose is to have both legacy and SDN network technologies in place to better evaluate the advantages and disadvantages of each approach. Finally, we are working on exposing the ability of performing SFCs to the end-user.

VIII. Conclusions

The orchestration and management of SFs is today a complex task that takes considerable time and effort. However, concepts like Cloud Computing, SDN and NFV are paving the way to handle SFs in a much more flexible and agile manner. The Telco will play a key role in this scenario, and we have given some insights on how that can be performed in the near future. Special attention has been given to the modeling of SFs towards cloud resources and to the combination of SFs through SFC. Finally, a platform for managing virtual SFs in a Telco cloud infrastructure has been presented and we described a POC that showcases how the platform and the principles here presented can be leveraged in a Telco environment.

References

[1] H. Kim, N. Feamster, "Improving network management with software defined networking," Communications Magazine, IEEE , vol.51, no.2, pp.114,119, February 2013.

[2] ETSI, Network Functions Virtualisation (NFV): Use Cases, Technical Report ETSI GS NFV 001 v1.1.1, October 2013.

[3] P. Quinn, Kumar, T. Nadeau, "Service Function Chaining Problem Statement," IETF Internet draft, Informational, December. 2013.

[4] A. Császár, W. John, M. Kind, C. Meirosu, G. Pongrácz, D. Staessens, A. Takács, and F.J. Westphal, "Unifying Cloud and Carrier Network: EU FP7 Project UNIFY", IEEE/ACM 6th International Conference on Utility and Cloud Computing (UCC). IEEE Computer Society, Washington DC, USA, 2013.

[5] G. Xilourisl, E. Trouva, F. Lobillo, J.M. Soares, J. Carapinha, M.J. McGrath, G. Gardikis, P. Paglierani, E.Pallis, L. Zuccaro, Y. Rebahi, A. Kourtis, "T-NOVA: A Marketplace for Virtualized Network Functions", European Conference on Networks and Communications (EUCNC 2014), June 2014.

[6] H. Puthalath, J. Soares, et al., "Negotiating On-demand connectivity between clouds and wide area networks," IEEE CloudNet, Paris, November 2012.

[7] Y.Zhang, N. Beheshti, L. Beliveau, G. Lefebvre, R. Manghirmalani, R. Mishra, R. Patney, M. Shirazipour, R. Subrahamaniam, C. Truchan, Tatipamula, M., "StEERING: A Software-Defined Networking for Inline Service Chaining," Proceedings of. IEEE ICNP 2013, Goettingen, Germany, October 2013.

[8] ETSI, Network Functions Virtualisation (NFV): Architectural Framework, Technical Report ETSI GS NFV 002 v1.1.1, October 2013.

[9] ETSI, Network Functions Virtualisation (NFV): Terminology for Main Concepts, Technical Report ETSI GS NFV 003 v1.1.1, October 2013.

[10] W. John, K. Pentikousis, G. Agapiou, E. Jacob., M. Kind, A. Manzalini; F. Risso, D. Staessens, R. Steinert, C. Meirosu, "Research Directions in Network Service Chaining", *Future Networks and Services* (SDN4FNS), 2013 IEEE SDN for , vol., no., pp.1,7, 11-13 November 2013.

[11] J. Soares, B. Parreira, M. Dias, J. Carapinha, S. Sargento, "Cloud4NFV: A Platform for Virtual Network Functions", IEEE CloudNet, October 2014.

[12] Alcatel Lucent, "Network Functions Virtualization – Challenges and Solutions", White Paper, 2013.

Biographies

João Soares received his M.Sc. degree in Electronics and Telecommunications Engineering from University of Aveiro in 2009, where he is currently pursuing a Ph.D. Initiated his professional activity in the Institute of Telecommunications in 2009 and joined Portugal Telecom Inovação e Sistemas in 2010. He recently joined Ericsson Research as an Experienced

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.

Researcher in the area of Cloud Technologies. His interests cover the areas of Cloud Computing, Cloud Networking, SDN and Network Virtualization.

Carlos Gonçalves received his Master of Science degree in Computers and Telematics Engineering from University of Aveiro in 2013. Currently, he is a Research Associate at NEC Laboratories Europe working in the areas of Network Functions Virtualization and Carrier-Cloud Operation & Management. In the last years, Carlos participated in several open-source projects including OpenStack and Open Platform for NFV, serving as a member of the OpenStack Networking Advanced Services and Telco Working Groups.

Bruno Parreira received his M. Sc. Degree Electronic and Telecommunications Engineering from University of Aveiro in 2012. He started his professional activity in Institute of Telecommunications through a research scholarship in May 2012. He has participated in several European funded projects related with Cloud Networking. His main interests are: cloud computing, cloud networking, software defined networking and network functions virtualization.

Paulo Tavares received his M. Sc. Degree of Engineering of Computers and Telematics from University of Aveiro in 2011. He started his professional activity in Institute of Telecommunications through a research scholarship in Setember of 2011. Main interests includes cloud computing, cloud networking, software defined networking and network functions virtualization.

Jorge Carapinha graduated in Electronic Engineering from the University of Coimbra in 1984 and got an M.Sc. degree in Telecommunications from the University of Aveiro in 1998. He has been with Portugal Telecom Inovação e Sistemas (formerly CET) since 1985. Currently his main fields of interest are Network Virtualization, Cloud Networking and Software Defined Networking.

Dr. João Paulo Barraca received a Ph.D. degree in Informatics Engineering from the University of Aveiro, where he developed work focused in network management functions. He currently is an invited lecturer at the University of Aveiro, and researcher at Institute of Telecommunications, in areas related to programming, networking and security. He has published more than 40 papers in the areas of networking and computer systems, and acted as reviewer for tens of events and journals.

Rui L. Aguiar received a Ph.D. degree in electrical engineering in 2001 from the University of Aveiro, Portugal, where he is currently a professor. He is leading a research team at the Institute of Telecommunications, and is an invited researcher at Universidade Federal de Uberlandia, Brasil. His current research interests are centered on the implementation of advanced networks and systems with special emphasis on future Internet and 5G architectures, and he is currently involved in the 5G-PPP initiative. He is a member of ACM and a senior member of IEEE, with more than 350 published papers. He has served as technical and general chair of several conferences, such as (recently) Monami'2012, ISCC'2014, NTMS'2014 and Mobiarch'2014. He has been invited as keynote in several events, both technical and for large generalist audiences.

Susana Sargento (http://www.av.it.pt/ssargento) is with the University of Aveiro and the Institute of Telecommunications since February 2004, where she is leading the Network Architectures and Protocols (NAP) group (http://nap.av.it.pt). She is also the Veniam co-Founder (www.veniam.com), a spin-off that commercializes vehicular technology. She has been involved in several national, US and European projects, taking leaderships of several activities in the projects, such as the QoS and ad-hoc networks integration activity in the FP6 IST-Daidalos Project, and the deployment of 600 nodes in a vehicular network in the EU Future Cities. Her main research interests are in the areas of Future Networks, more specifically routing, QoS, mobility, and could integration.

* João Soares and Carlos Gonçalves were with Portugal Telecom and Instituto de Telecomunicações at the time this work was carried out.