5G E2E Network Slicing Management with ONAP

Veronica Quintuna Rodriguez*, Fabrice Guillemin* and Amina Boubendir†

*Orange Labs, 2 Avenue Pierre Marzin, 22300 Lannion, France
†Orange Labs, 44 Avenue de la République, 92320 Châtillon, France
{veronica.quintunarodriguez, fabrice.guillemin, amina.boubendir}@orange.com

***Abstract*—Network slicing is one of the central features of 5G networks and beyond. Slicing enables network operators to provide differentiated services to customers by dynamically allocating dedicated parts of their networks. To take full benefit of Slicing, we focus on its life-cycle management by paying special attention to automation for more overall network efficiency, OPEX savings and time-to-market acceleration. We address the modeling, orchestration and enforcement of an end-to-end (E2E) network slice, which involves the RAN, transport and Core networks. The slice management is performed while using a carrier-grade automation platform, namely the so-called Open Network Automation Platform (ONAP). As an illustrating use-case, we consider a private mobile network deployed over a virtualized infrastructure.**

Keywords: Network Slicing, Lifecycle management, Automation, ONAP, NFV, On-demand, SLA-compliance.

I. Introduction

Network slicing was introduced a few years ago in the framework of 5G networks to offer enriched services beyond standard connectivity. The main motivation behind introducing network slicing is the need for more accurately meeting the requirements of vertical markets (entertainment, health, automotive, etc.).

In this context, 3GPP has introduced network slices according to 3 major classes, defined as: enhanced Mobile BroadBand (eMBB), Ultra-Reliable Low-Latency Communications (uRLLC), and massive Machine Type Communications (mMTC). They roughly correspond to Quality of Service (QoS) classes with specific tailored objectives in terms of bandwidth, latency, reliability, etc. Beyond the QoS, network slicing is intended to cover a much broader scope by allowing the creation of multiple logical networks over shared infrastructures.

However, initial implementations of Slicing are based on classical interconnection services as those offered by standard Virtual Private Networks (VPNs) complemented by specific IT resources. Yet, the innovation of network slicing resides in that such logical networks can also embed functional customization through composition or chaining of Virtualized Network Functions (VNFs). This allows the deployment of logical networks, each running own protocols and network services tailored to customers' needs. In this respect, a network slice can be seen as an overlay network but with a footprint in the network.

Unlike classical overlay networks, a network slice is established via a negotiation between the customer (in charge of describing and specifying the network slice in accordance with the targeted usage) and the network operator (in charge of designing, deploying and managing the network slice). The customer specifies the needs in terms of connectivity, interconnection, cloud resources and network functions while the network operator is in charge of implementing those functions and network resources so as to offer the desired network service according to a given Service Level Agreement (SLA).

The active participation of the network operator and the negotiation of SLAs are the two main differences between network slices and classical overlay networks. These latter are purely over the top of the network infrastructure without any network footprint and hence offer best effort quality, even if in practice this might be sufficient for end users. The negotiation of a SLA is usual in VPN services, where VPNs offered by an operator are continually monitored in order to verify the conformance with regard to the SLA. The negotiation of a SLA for a complete service function chain based on VNFs is much more challenging.

Furthermore, the emergence of network slicing is strongly related with that of Network Function Virtualization (NFV) and Software Defined Network (SDN), which notably bring flexibility and agility to legacy networks. In fact, the network programmability has opened the door to new business models towards customized offers. This dynamicity involves new challenges for network operators, that mainly consist to orchestrate and manage VNFs (in particular their lifecycle with adequate assurance). Various orchestration platforms are under development, such Open Source MANO (OSM) based on ETSI-NFV Management and Orchestration (MANO) [1], Open Baton, Open Network Automation Platform (ONAP), among others. In this paper, we specifically consider ONAP, which is a carrier-grade open orchestration platform.

The objective of this paper is to investigate how network slices can be modeled and deployed in the framework of ONAP, and how this platform can facilitate the slice management. We notably consider the entire life-cycle of a network slice, which involves , modeling, onboarding, instantiation and operation.

978-1-7281-5127-4/20/$31.00 ©2020 IEEE

As a driving use case of this study, we consider the deployment of a tailored and private mobile network for a given customer or market. We address both the design- and run-time of the proposed approach while dealing with the various VNFs composing the slice as well as the policies involved in its behavior. During the operation time, we specially focus on the monitoring and assessment of the slice SLA. All these stages are detailed in light of the different ONAP components. The main contribution of this paper is to propose an architecture for deploying network slices in the framework of ONAP and to enforce negotiated SLAs by means of monitoring and policy enforcement compatible with ONAP features.

This paper is organized as follows: in Section II, we review some definitions and current studies on network slicing. In Section III, we investigate how network slicing can be implemented in the framework of ONAP. The proposed approach is illustrated in Section IV by considering a practical use case (namely, a private mobile network). Some concluding remarks are presented in Section V.

II. Network Slicing: background and definitions

A. Network slicing

Network slicing has being widely studied in the past few years in Academia as well as in Industry. There are actually various approaches to network slicing. A network slice is fundamentally defined as a bundle of services, a logical network, a type of virtual networking architecture, a chain of network functions or a substrate network [2], [3], [4], [5]. Network slicing is thus capable of providing logical dedicated networks upon a common infrastructure. It requires the coordination of the various network segments (core, access and transport networks) and aims at meeting specific requirements of vertical markets. Similar definitions can be found in the framework of European projects [6], [7].

Standardization bodies such as ETSI [8] defines a slice as a service aware logical network composed of physical or virtual network elements, resources and functions. Similarly, for 3GPP [9], [10], a slice is a paradigm, where logical partitions are created within a Public Land Mobile Network (PLMN) with appropriate network isolation, resources, optimized topology and specific configuration to serve specific service requirements.

B. Network Slicing Architecture

3GPP has defined in various 5G Technical Specifications (namely, TS 23.501 [10], TS 23.502 [11], TS 23.503 [12], TS 28.530 [9], TR 28.531 [13], TR 28.801 [14]) the main principles for supporting network slicing. Among the most relevant entities introduced by 3GPP, we can cite:

- Network Service (NS): A logical network composed of a chain of network functions.
- Network Slice Instance (NSI): A set of instances of network functions and the required cloud resources to execute them.
- Network Slice Subnet (NSS): A slice subnet is a network segments or sub-slice within a broader slice. For instance the access network within an end-to-end mobile network.
- Network Slice Subnet Instance (NSSI): A set of instances of network functions belonging to a network segment.

According to 3GPP, a single device (UE) shall support up to eight simultaneous connections to different slices. A slice shall be in fact identified by a Slice Differentiator (SD). The UE attachment to a given slice is then performed by the Network Slice Selection Function (NSSF), while using the Network Slice Selection Assistance Information (NSSAI), that refers to the expected slice performance (latency, bandwidth). During the registration procedure, the Access and Mobility Management Function (AMF) selects the adequate NSI among the enabled slices according to the user subscriptions.

The NF Repository Function (NRF) is responsible of discovering the functions involved in the selected slice. The establishment of a session and data transmission is then given after the selection of the User Plane Function (UPF) function by the Session Management Function (SMF). The slice behavior is supervised by the Policy Control Function (PCF).

Figure 1 illustrates an example of two network slices that share the same access network subnet and implement two data tunnels each carrying a specific QoS. Similarly, 3GPP has introduced various entities for dealing with the network slicing management and orchestration, as follows:

- Communication Service Management Function (CSMF): It enables translating the performance requirements of a service to slice technical features (service level);
- Network Slice Management Function (NSMF): It performs the management and orchestration of NSI (slice level);
- Network Slice Subnet Management Function (NSSMF): Responsible of the management of NSSI (subnet level).

C. Network slicing in ONAP

Network slicing is a major challenge for the ONAP community [15], particularly interested in upgrading the various ONAP components in order to support Network Slicing. The ONAP approach is in particular based on 3GPP specifications [9], [10]. A network slice is thus considered by ONAP as a logical network composed of three segments: Radio, Core and Transport sub-networks. The functional components to be implemented by ONAP for supporting slice management are still under discussion. Various scenarios are proposed for including or delegating some management functions, i.e. NSMF, NSSMF, CSMF.

Even if legacy ONAP releases as Casablanca or Dublin do not directly support network slicing, ONAP network services can be seen as network slices or subnets.

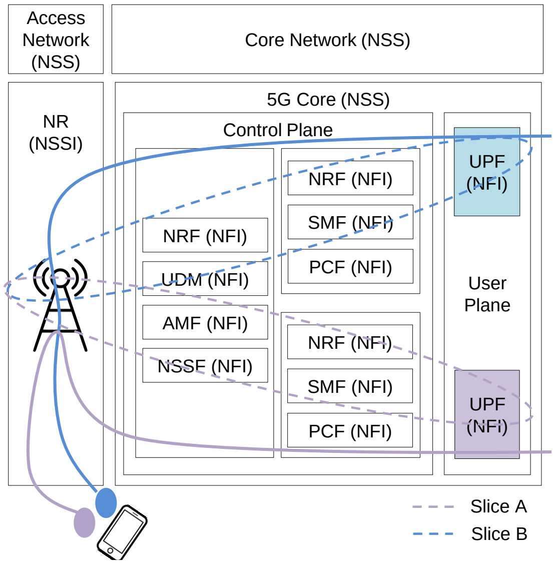

Fig. 1. Network slicing according to 3GPP.

The drawback of this approach is however the absence of hierarchical lifecycle management, i.e., there is no way to manage internal services as microservices with their own lifecycle. The ability to define top down properties on a service is currently for future ONAP releases. A slice instantiation will be able to fullfil the communication service requirements by aggregating the performance properties of the different slice segments (subnets).

III. Automating Network Slices Management

A. *Proposed architecture*

The network slice management can be done by an external entity, which directly interacts with the various service orchestration platforms (e.g. ONAP) involved in an end-to-end network slice. The architecture that we introduce in this paper is illustrated in Figure 2. We specifically propose to introduce a slice manager placed on the top of the service orchestration architecture, which in turns manages the various virtualization infrastructures and the lifecycle of network services.

The rest of this section is devoted to the description of the proposed architecture for deploying and managing network slices. Such an architecture is currently not explicitly considered in the framework of ONAP.

The proposed architecture is motivated by the need for monitoring network slices and implementing rules in order to maintain negotiated SLA. ONAP offers all the interfaces and facilities needed to implement network slices. It is worth noting that this architecture is similar to what is implemented today by operators for VPNs. ONAP however offers a much wider range of actions than simply monitoring bandwidth in IP networks. The slice manager interfaces with the service orchestrators to expose the required Key Performance Indicators (KPIs) that guarantee the slice behavior.

The end-to-end slice performance is then given by the conjunction of individual performances of the various network services involved in a slice. Network slicing goes much more beyond simply maintaining a logical chain of network functions. The role of the slice manager is to guarantee the SLA signed between a customer and a slice provider. Maintaining negotiated SLAs requires an enforcement infrastructure (implying policies and constrains), which must involve (i) a monitoring infrastructure for collecting and reporting the performance information of the various VNFs of the slice, (ii) an analytic entity capable of evaluating the reported information. The collected and computed KPIs are then used to prevent from SLA degradation. These various tasks involved in the slice management are supported by ONAP as part of both design and run-time frameworks.

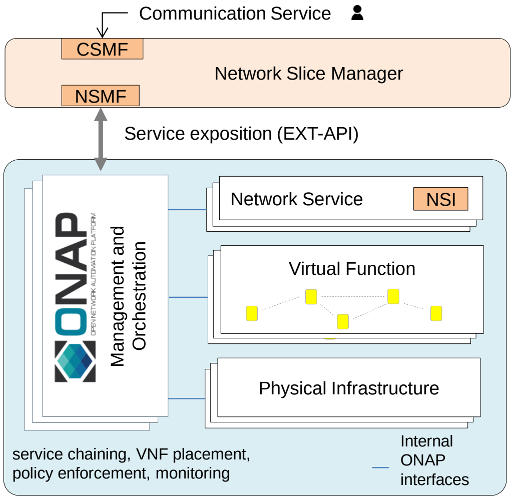
Communication Service
CSMF
Network Slice Manager
NSMF
Service exposition (EXT-API)
ONAP
OPEN NETWORK AUTOMATION PLATFORM
Management and Orchestration
Network Service
NSI
Virtual Function
Physical Infrastructure
service chaining, VNF placement, policy enforcement, monitoring
Internal ONAP interfaces

Fig. 2. Slice Management Architecture

The design time consists of specifying the various components of the slice, i.e., Cloud-Native Network Functions (CNFs), VNFs, interconnection networks, policies and rules according to SLAs, etc. VNFs (or CNFs) are software suites that need to be onboarded onto ONAP through the Service Design and Creation (SDC) module by means of Heat templates (or Helm charts). ONAP is compatible with both container- and VM-based virtualization environments supported either by Kubernetes or OpenStack. In addition, ONAP provides VNF packaging and validation tools/APIs to keep a normalized VNF catalog. Policies and constraints are defined by means of the Policy Creation Component (PCC). The runtime framework performs the creation of the slice components (namely the VNFs/CNFs and their interconnection network) and executes the rules and policies defined during the design time.

The creation and operation of network slices actually involve various ONAP components:

- The Service Orchestrator (SO), which triggers the creation, update or removal of services,

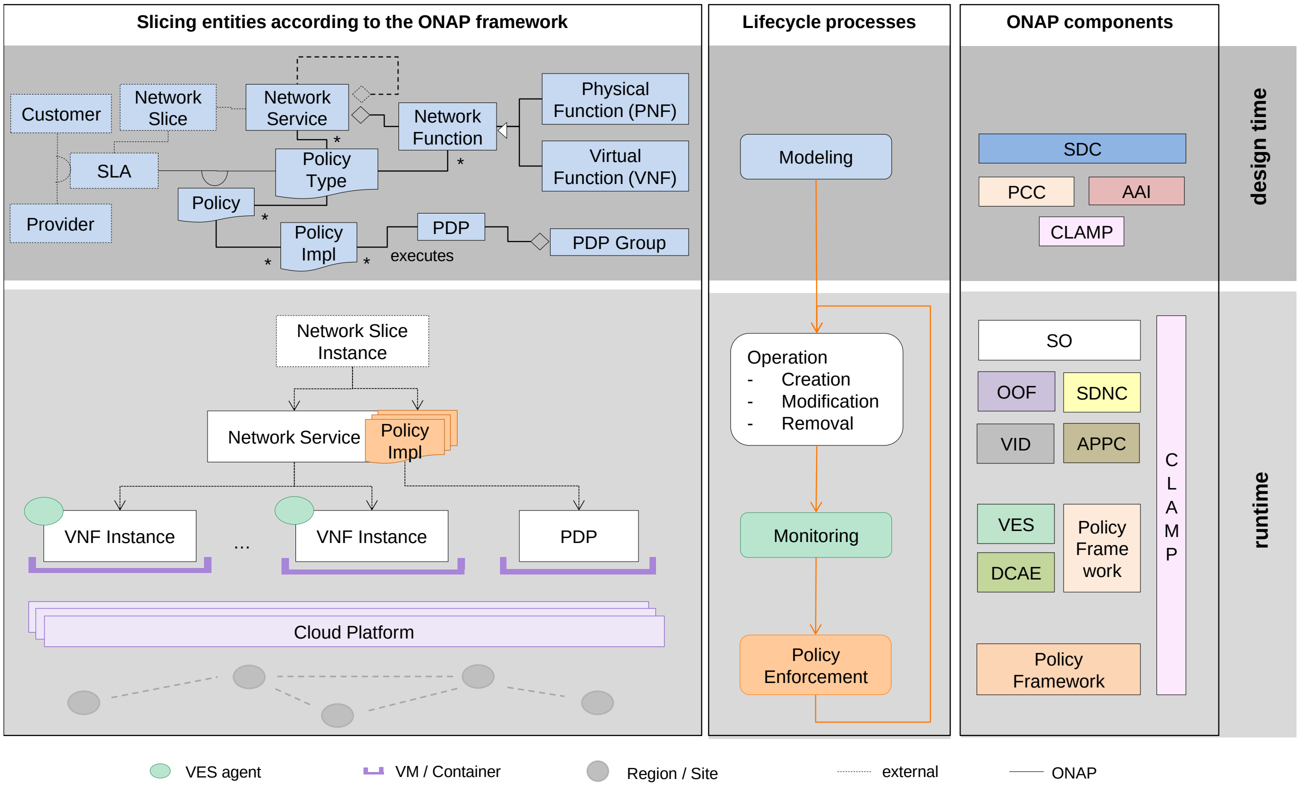

Fig. 3. ONAP-compliant network slicing lifecycle.

- The ONAP Optimization Framework (OOF), which enables policy-driven optimized placement of services and VNFs across muti-site and multi-cloud infrastructures. The placement problem is addressed in [16],
- The controllers, mainly the SDN-C, which manages, assigns and provisions network resources, and the APP-C, which provide lifecycle management of VNFs. In addition, the VF-C aims at leveraging ETSI NFV MANO [17] architecture and at implementing Fault-management Configuration Accounting Performance and Security (FCAPS) of VNFs (VF-C project is not yet available),
- The Active and Available Inventory (AAI), that enables real-time views of resources, services, products and their relationships with each other.

The modules listed above are used to create and maintain a slice in order to meet the negotiated SLA. Globally, the complete lifecycle of a slice involves four phases, as we detail in the following subsections and shown in Figure 3.

B. *Lifecycle of a network slice*

1) Modeling: A network slice can be built in ONAP by means of a single NS, which in turn is a chain of either virtual or physical Network Functions (NFs).

In ONAP, NFs are defined as part of Vendor Software Products (VSPs). The NS can then be modeled as a bundle of NFs through the SDC.

We have proposed a detailed ONAP-compatible network slicing model in [18]. In the present paper, we argue that policies and constraints should also be defined and attached to a network slice. These rules shall trigger specific tasks when conditions are met during the runtime.

At a glance, a network slice can be modeled as in Figure 3, where all the functions of the network service and the associated monitoring and enforcement are represented. A slice involves a Customer (the owner of the slice), which negotiates an SLA with a Network Slice Provider. In practice, the slice is instantiated as a Network Service holding the NFs and their instances. These ones are subsequently monitored and KPIs are computed to verify that the negotiated SLA is respected. In case of violation, policies are enforced.

At the end of this first phase, a network slice constitutes an NS enriched by policies. The artifacts, VFs, and rules defined during the design time are used in the run-time by other ONAP modules to deploy and monitor the slice.

2) Deployment: The various VNFs composing a slice give rise to VNF instances running on multi-cloud and/or multi-site infrastructures. The policies attached to the slice are also launched in this phase. In ONAP, the SO executes the defined processes for the slice creation (see Figure 3 for an illustration). In addition, ONAP makes available a user application, the Virtual Infrastructure Deployment (VID), for instantiating services from the SDC.

The deployment phase is critical for the slice performance. In particular, this phase is in charge of placing the various VNFs composing the slice in the hosting servers. This operation notably takes into account the requests in terms of networking and cloud resources but also QoS requirements such as latency and bandwidth.

3) Monitoring: The salient feature of network slices is that they have to respect an established SLA. The slice supervision begins with the monitoring of services, VNFs and volumes involved in an end-to-end slice. In ONAP, the monitoring is assured by various components notably the Data Collection Analytics and Events (DCAE) and the Virtual Event Streaming (VES). After event collection, policies are called for enforcing actions.

The set up of the monitoring infrastructure of a slice is critical for maintaining the negotiated agreements. The behavior checkup begins with the instantiation of VES agents attached to each and every one of the components that require to be controlled (e.g., VNFs, servers, network links). The KPIs for the slice are then built on the basis of the reporting information elements. If there is any discrepancy of the current performance of the slice with regard to the SLA, the enforcement procedure is triggered. Decisions are taken as detailed below.

4) Policy enforcement: The policy enforcement involves the various actions that can be executed in the case of degradation in the global performance of a slice, for instance allocating more cloud or network resources, terminating secondary tasks, scaling up or down slice components, moving some VNFs, etc. The policy enforcement of a slice can be performed by an operator, but it is highly desirable to automate this procedure. This is notably the objective of the Policy Framework, which is central for decision making in ONAP. This framework manages both policy execution and enforcement by means of Policy Decision Point (PDP) and Policy Enforcement Point (PEP), respectively.

The various service agreements of a given slice are then defined by means of technical rules implemented as policies. More concretely, a policy defines the features (values of KPIs) and the actions that shall be triggered when the requirements of a given slice are not respected. In ONAP, the policies can be modeled by means of TOSCA artifacts. ONAP uses various entities for dealing with policies, which can be used either by the Policy Framework, Closed Loop Automation Management Platform (CLAMP) or other components. Various data entities are involved in the construction of a policy, they mainly are:

- Policy Type (TOSCA): It enables designing abstract policies used by a service or its components.
- Policy (TOSCA): It is the instance of a given policy type that is attached to a service. This artifact defines the values of the properties, the entities where the policy acts on (VNFs, resources, or other network elements), and the list of actions to perform when the trigger fires.
- Policy Type Implementation: It defines the implementation of a Policy Type by means of APEX, XACML, Drools or other. The creation and/or integration of this artifact is out of the scope of ONAP (Dublin), however ONAP has pre-loaded various Policy Type Implementations that can be used for closed loop management (namely, monitoring, operational actions, guard policies, optimization policies, coordination among multiple loops at runtime, etc).
- Policy Implementation: This artifact is the runnable version of a Policy, which is built from both Policy and Policy Type Implementation.
- PDP: This module executes the policies (more precisely, Policy Implementation). PDP returns the decision either to the requester (synchronous mode) or to another entity (asynchronous mode). During asynchronous invocations the requester is not waiting for the policy execution result. The policy framework allows the allocation of PDPs to a PDP Groups and Subgroups so that they can be managed as microservices.

In ONAP, policies are executed in PDPs while the policy enforcement occurs in the component that receives the policy decision. It is shown in Figure 4. The enforcement procedure of a slice by means of policies is illustrated in Section IV.

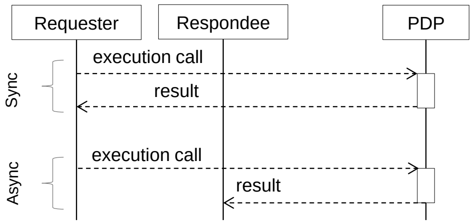

Fig. 4. Policy execution in ONAP.

C. Slice Enforcement Automation

In the perspective of automating the slice enforcement, it is worth noting that ONAP has especially defined CLAMP to automate the enforcement of services. This closed loop considers three micro-services, namely (i) event detection, (ii) policy validation, and (iii) actions implementation. These micro-services are enabled via the cooperation of various ONAP components, namely, the SDC for the closed loop design and its association with the Service and VNFs, the DCAE to collect data and to manage loops themselves (namely, the DCAE uses TOSCA blueprints for launching the loops), the Policy Framework (which is fully TOSCA compliant) for creating, executing and updating policies, and the controllers and orchestrators to implement the policy-driven actions. The goal of CLAMP is to avoid user interaction with other components for managing the lifecycle of network services and consequently of slices.

D. Slice Supervision

The network slice supervision can be performed by a specialized entity inside or outside ONAP, e.g. a function of an external network slice manager or the supervision procedure of the NSMF. Today, ONAP is able to expose the data involved in the monitoring and more generally in the whole closed loop of a slice (or service) by means of REST-based APIs to a third-party system.

Going deeper, ONAP implements various APIs which are consumed for the different components, for instance CLAMP offers a *health-check* API to verify the activity of the closed loop, SDC APIs enable the verification of the distribution of services, DCAE APIs allow an entity to check that the various microservices that are involved in monitoring are up, among others. In addition, Policy APIs provide communication channels to update parameters during runtime.

To conclude this section, we see that ONAP provides all the tools to implement a slice with a given SLA. The slice is basically a network service enriched by monitoring and policy enforcement functions. In fact, it is possible to introduce in ONAP a specific module for managing network slices or to expose via REST APIs all the information elements necessary for managing a slice by a third party, for instance a supervision center (as it is the case today for VPNs for most network operators) or directly to the slice Customer. In fact, the final choice is driven by non technical considerations, out of the scope of the present paper. In the next section, we illustrate how a slice corresponding to a dedicated mobile network could be deployed using ONAP.

IV. Network Slicing Implementation

In this section, we consider as a slicing use case, the deployment of a private mobile network for a given customer. For instance a vertical market (e-health, automotive) that requires a tailored mobile network, which basically includes two kinds of QoS: best-effort and broadband communications. The required performance is specified in a SLA between the customer and the operator. The so-called broadband channel guarantees a given data rate and has priority over the best-effort communications.

The private mobile network involves both the core and access networks. We consider that the access network can be given either by sharing the RAN with the public mobile network service or by deploying a virtual RAN and dedicated antennas. The core network slice subnet is composed of various Virtual Function Components (VFCs) that belongs to the control or user plane. The access network slice subnet is emulated by using virtual eNBs and UEs. The entire mobile network functionality (i.e., the VFCs: Mobility, Subscribers DB, Serving & Packet Gateway, DHCP, SDNC, AAA, NAT, OvS, Base Band) is based on various open source solutions notably Open Air Interface (OAI) and the enhanced core developed by *b<>com*.

We use OpenStack as Cloud engine to allocate the required computing, networking and storage resources to the VMs that host the VFCs. Finally the orchestration and management functions involved in the lifecycle of the slice are performed by ONAP as illustrated in Figure 5.

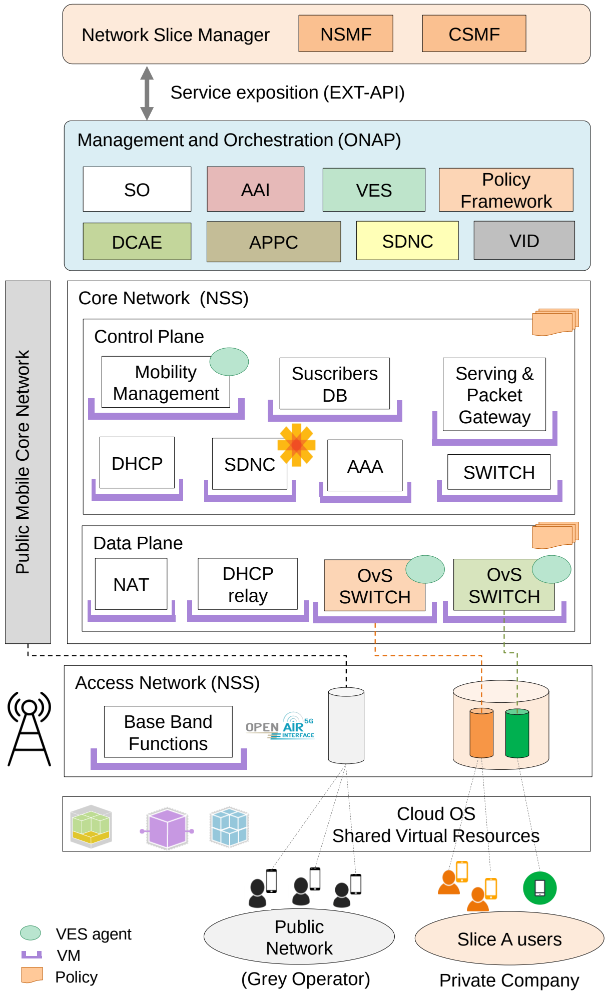

Fig. 5. Network slicing usecase. Testbed architecture.

A. Slice Modeling

The slice can be modeled either as a single ONAP network service, which in turn is composed of a forwarding graph of ONAP Virtual Functions (VFs) or as a bundle of ONAP-network services, where each of them is formed of VFs. The drawback of this latter possibility is that the slice is not seen as a unique entity, which limits the enforcement of end-to-end policies. Other modeling approaches can be found in [19]. We specify the slice by means of a single ONAP-Network-Service which in turn is defined by three VFs - Core Control Plane (CP), Core Data Plane (DP), and RAN emulator- each of them containing various components or subfunctions. We also define two policies for assuring the negotiated SLA. The slice model is illustrated in Figure 6.

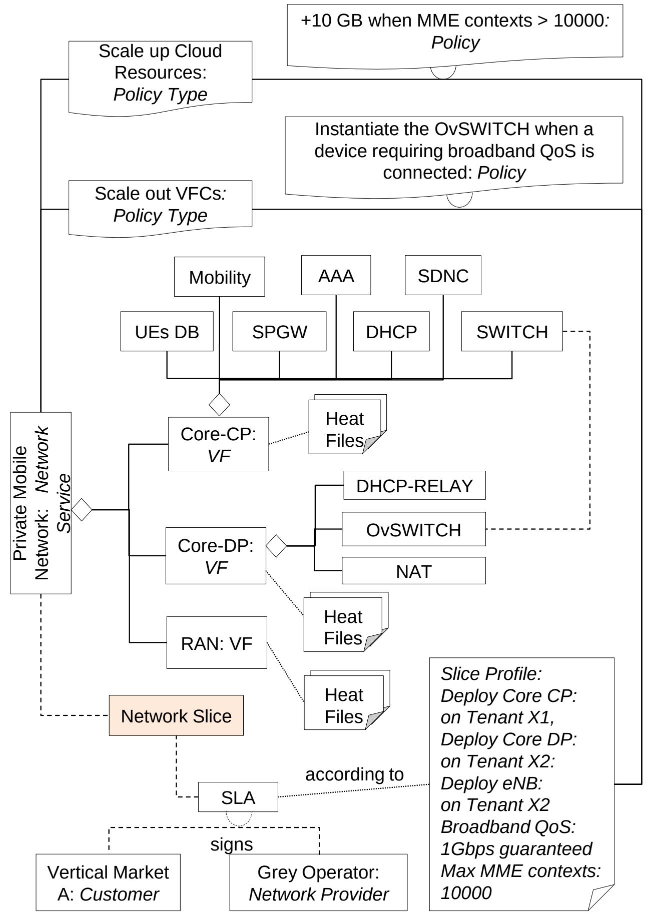
+10 GB when MME contexts > 10000: Policy
Scale up Cloud Resources: Policy Type
Instantiate the OvSWITCH when a device requiring broadband QoS is connected: Policy
Scale out VFCs: Policy Type
Mobility
AAA
SDNC
UEs DB
SPGW
DHCP
SWITCH
Private Mobile Network: Network Service
Core-CP: VF
Heat Files
DHCP-RELAY
OvSWITCH
NAT
Core-DP: VF
Heat Files
RAN: VF
Heat Files
Network Slice
Slice Profile: Deploy Core CP: on Tenant X1, Deploy Core DP: on Tenant X2: Deploy eNB: on Tenant X2 Broadband QoS: 1Gbps guaranteed Max MME contexts: 10000
according to
SLA
signs
Vertical Market A: Customer
Grey Operator: Network Provider

Fig. 6. Network slicing model for the driving use-case.

B. Slice Deployment

The slice deployment involves the following steps:

- Slice production and packaging: at this phase network engineers implement the source code of the various VFs, write heat files and generate VMs images.
- VF validation to be compliant with the orchestrator requirements and guidelines,
- Cloud OS preparation. This stage involves creating flavors, networks, images in the hosting cloud operating system, namely, OpenStack.
- Slice onboarding: deals with creating and distributing the services in the Orchestration Platform (ONAP).
- Slice instantiation: This last phase of the deployment triggers the execution of the VFs composing the slice.

VFs can be written as a chain of VFCs hosted in individual or multiple VMs or containers and interconnected by internal networks. We define the Core CPVF, Core DPVF, and RAN by means of three text-based templates (heat templates when using OpenStack or Helm charts when using Kubernetes). Each template contains the various cloud resources (VMs, internal networks, subnetworks, ports, security groups, etc) composing the VF.

Before onboarding the various VFs in ONAP, they need to be validated to be compliant with a given orchestrator (ONAP in this case). In fact, a slice is a software product that may be provided by a third party; hence, a certain compatibility between VFs and the orchestration platform must be checked. ONAP makes available a validation environment, namely, VNF Validation Program (VVP). It notably checks the required metadata and naming conventions to facilitate the management of the cloud resources composing a VNF. We have notably adapted the VFs provided by b<>com to be compliant with ONAP.

```
---
SliceE2E:
    tosca_file_from_SDC: service-SliceE2E-template
    version: "1.0"
    vnfs:
        - vnf_name: CoreCP
          heat_files_to_upload: CoreCP/CoreCP.zip
          vnf_parameters: [
              ...
          ]
        - vnf_name: CoreDP
          heat_files_to_upload: CoreDP/CoreDP.zip
          vnf_parameters: [
              ...
          ]
        - vnf_name: RAN
          heat_files_to_upload: RAN/RAN.zip
          vnf_parameters: [
              ...
          ]
```

Fig. 7. Slice template.

Beyond compliance, all resources belonging to a given slice require to be available in the cloud operating system, i.e., in the hosting virtualization environment. Hence, images, flavors, keypairs and other cloud resources need to be *a priori* created in the cloud platform (OpenStack). The slice onboarding can then be performed. The service building begins with the creation of VFs while uploading the heat files. In ONAP, VFs are part of VSPs and need to be defined under a given software license. Once VFs have been imported to the SDC and certified by the service designer, they are ready to be used as part of a slice (network service). The slice is then built as a chain of VFs. The design stage is finished, however, before the service is ready for instantiation, it needs to be validated (by a tester), approved (by a governor) and distributed (by the operator). With regard to automation, we have specifically automated the onboarding, distribution and instantiation using "onap-tests", an open-source automation tool. It uses as inputs the various Heat files, the service model, and the cloud platform information (cloud owner, region id, tenant name -also referred to as project-, and tenant id). The service model definition is illustrated in Figure 7. The end-to-end slice deployment can then be performed on-the-fly within one to very few minutes. In addition, VNFs can be automatically instantiated in different tenants or cloud platforms according to placement policies using the OOF.

C. *Slice Monitoring and performance enforcement*

To illustrate the use of policies in ONAP for network slice management, we consider the scale-up and scale-in of cloud resources by means of automated closed loops in ONAP. We particularly address the scale-up of RAM resources of the *Mobility Management VM* when the number of existing UE contexts is greater than a given threshold (e.g., 10000). We also consider the scale-out of the *OVSwitch VFC* for enabling differentiated traffic, e.g, best-effort and dedicated channels.

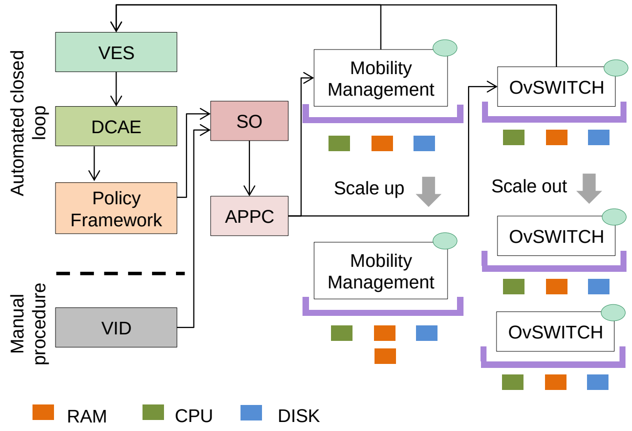

Fig. 8. Slice monitoring and enforcement.

The dedicated channel guarantees a given data rate according to the SLA. While scaling-up the *Mobility Management VFC* enables adding capacity to meet increased demand, the scale-out of the *OVSwitch VFC* instantiates an additional VM to provide a specific service.

The scale-out operation can be manually triggered from the ONAP dashboard using the VID portal, as well as, automatically by means of the defined policies (using VES and DCAE for monitoring and the policy framework for the evaluation of rules). Both manual and automated procedures call the SO to perform the scaling operation. Then, the SO sends a request to the Application Controller (APPC) to health check and to reconfigure the VNF. These procedures are shown in Figure 8. We notably observe that ONAP enables scaling VFs on the fly without interrupting the service.

V. Conclusion

It turns out that the real challenge in network slicing, beyond the specification and the deployment, is to define adequate policies to monitor and thus to work on guaranteeing the slice functional and non-functional behavior.

The technical behavior should be maintained to reflect the functional (service) and non-functional (QoS service requirements) as expressed by the slice service tenant or client in a form of SLA and SLO.

To measure such a technical behavior aggregated end-to-end KPIs should be derived from the measures exposed by ONAP coming from individual network functions and the cloud resources executing them.

We have developed in this paper an approach to network slicing in the framework of ONAP. We have introduced a slice manager on the top of ONAP for specifying an end-to-end network slice and then to deploy it via ONAP. This latter subsequently offers all the features in terms of monitoring and policy enforcement to maintain the negotiated SLA of end-to-end slices.

References

[1] ETSI, "Network Functions Virtualization. Architectural Framework," 2013, Technical Report ETSI GS NFV 002 V1.1.1.

[2] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network slicing as a service: enabling enterprises' own software-defined cellular networks," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 146–153, 2016.

[3] China Mobile, Huawei, Deutsche Telecom, Volskwagen, "5G Service-Guaranteed Network Slicing White Paper," 2017.

[4] Ericsson, "Network Slicing can be a piece of cake," Ericsson study: How network slicing pays off, 2018.

[5] China Telecom, "China Telecom 5G Technology White Paper," 2018.

[6] A. Sgambelluri et al., "Orchestration of Network Services across multiple operators: The 5G Exchange prototype," in *2017 European Conference on Networks and Communications (EuCNC)*, June 2017, pp. 1–5.

[7] Q. Wang et al., "SliceNet: End-to-End Cognitive Network Slicing and Slice Management Framework in Virtualised Multi-Domain, Multi-Tenant 5G Networks," in *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, June 2018, pp. 1–5.

[8] ETSI, "Next Generation Protocols (NGP) E2E Network Slicing Reference Framework and Information Model," 2018, Report ETSI GR NGP 011 V1.1.1.

[9] 3GPP TS 28.530 V15.01.0, "Management and orchestration; Concepts, use cases and requirements (Release 15)," 2018.

[10] 3GPP TS 23.501 V16.0.2, "System Architecture for the 5G System (Release 16)," 2019.

[11] 3GPP TS 23.502 V16.2.0, "Procedures for the 5G System (5GS), Stage 2, (Release 16)," 2019.

[12] 3GPP TS 23.503 V16.2.0 , "Policy and Charging Control Framework for the 5G System (5GS), Stage 2, (Release 16)," 2019.

[13] 3GPP TS 28.531 V15.1.0, "Management and orchestration Provisioning (Release 15)," 2018.

[14] 3GPP TR 28.801 V15.1.0, "Study on management and orchestration of network slicing for next generation network (Release 15)," 2018.

[15] S. Sundaramurthy et al., "5G Slicing," https://wiki.onap.org/display/DW/5G+-+Slicing, 2018.

[16] F. Slim, F. Guillemin, A. Gravey, and Y. Hadjadj-Aoul, "Towards a dynamic adaptive placement of virtual network functions under ONAP," in *Third International NFV-SDN'17-O4SDI-Workshop on Orchestration for Software-Defined Infrastructures)*, 2017.

[17] ETSI, "Network Functions Virtualisation (NFV). Management and Orchestration ," 2014, Technical Report ETSI GS NFV-MAN 001 V1.1.1.

[18] V. Quintuna Rodriguez, F. Guillemin, and A. Boubendir, "Automating the deployment of 5G Network Slices using ONAP," in *NoF 2019 – 10th International Conference on Networks of the Future)*, 2019.

[19] A. H. Celdrán, M. G. Pérez, F. J. García Clemente, F. Ippoliti, and G. M. Pérez, "Policy-based network slicing management for future mobile communications," in *2018 Fifth International Conference on Software Defined Systems (SDS)*, April 2018, pp. 153–159.