A Cloud Native Management and Orchestration Framework for 5G End-to-End Network Slicing

Yi-Sung Chiu*, Li-Hsing Yen*, Tse-Han Wang*†, and Chien-Chao Tseng*
*Department of Computer Science, College of Computer Science, National Yang Ming Chiao Tung University, Hsinchu, Taiwan.
Email: steven30801@gmail.com, {lhyen, wangth, cctseng}@cs.nctu.edu.tw
†Network Management Laboratory, Chunghwa Telecom Laboratories, Taoyuan, Taiwan.

***Abstract*—The 3rd Generation Partnership Project (3GPP) introduces network slicing for the provisioning of diverse network services in 5G. Additionally, boosted by cloud-based virtualization technologies, Network Function Virtualization (NFV) is also adopted in 5G to enhance scalability and elasticity. 5G network elements are evolving to cloud-native deployment. However, existing end-to-end (E2E) network slicing frameworks do not embrace cloud native yet. In this paper, we present a MANagement and Orchestration (MANO) framework for the automation of end-to-end (E2E) network slicing with the implementations of the core network (CN) and transport network (TN) slices. The framework is a showcase that follows 3GPP network slice management and 5G core network slicing mechanism, integrates novel bandwidth management techniques, and exploits all open-source approaches with state-of-the-art cloud native technologies. We evaluate the resource overhead of the framework and service throughput under bandwidth policies.**

I. Introduction

The fifth-generation (5G) network aims to provide diversified network services to various applications such as high-quality video streaming, smart vehicles, and Internet of Things (IoT). To this end, the 5G network requires elasticity to allow service customization and programmability, which are different from one-size-fits-all architecture targeting mobiles only in the 4G network. A key enabling technology to meet these requirements is *network slicing*. Network slicing enables the sharing of a common 5G network infrastructure by one or more independent logical networks called *network slices* (NSs). Each NS is provided with customized network resources [1] to support a specific type of service with diverse service demands in terms of bandwidth, latency, and availability.

Network function virtualization (NFV) is a network architecture concept that leverages virtualization technology to decouple software-based network functions (NFs) from supportive hardware. The decoupling enables virtualized network functions (VNFs) to run on top of general-purpose hardware and commercial off-the-shelf (COTS) equipment. With NFV, network operators can get rid of proprietary equipment and rigid network architecture without affecting functionality. Additionally, softwarization on network functions enables agile deployment, structure evolution, and platform innovation. Consequently, network operators can enhance flexibility and scalability as well as reduce capital expenditures (CAPEX) and operating expenditures (OPEX) investments.

NFV has been identified as a key enabler for 5G infrastructure deployment [2]. With NFV, the evolving 5G network can be deployed and executed in an agile way, which provides high scalability and reliability on network services. Similarly, network slicing in 5G can leverage NFV features to operate in a dynamic manner. In particular, NFV enables managing network slices in an automated manner because it is simple to schedule, partition, and distribute resources of decoupled hardware in the NFV paradigm.

5G network elements are evolving to *cloud-native* deployment [3], where applications and services are designed for modern cloud infrastructure. Cloud-native network functions are fine-grained and decomposed into service-oriented components. Compared with the traditional monolithic architecture, the component-based style provides efficient use of resources, fast restoration under failure without interruption to other services, and interoperability across heterogeneous systems. Cloud native components are mostly implemented with lightweight virtualization technologies such as containers (instead of virtual machines) to realize rapid instantiation and migration.

There have been some end-to-end (E2E) network slicing frameworks proposed for 5G [4], [5], [6], [7], [8]. However, these frameworks are not cloud native. Specifically, one 4G evolved packet core (EPC) or virtual EPC (vEPC) is created for each network slice, which is not compliant with the service-based architecture in 5G [1]. Furthermore, most existing proposals or testbeds either do not address transport network aspects at all [4] or consider only connectivity (routing) services [5], [7], [8]. Important transport network issues such as Quality of Service (QoS) and bandwidth management have not been well addressed.

In this paper, we present a MANagement and Orchestration (MANO) framework for the automation of E2E network slicing and showcase a proof of concept (PoC) of network slicing selection and orchestration that follows the 3rd Generation Partnership Project (3GPP) specifications [9], [10]. The framework adopts state-of-the-art cloud-native technologies and all open-source approaches. We particularly address how to enforce QoS for different network slices coexisting in a transport network. We also demonstrate the efficiency of this framework under the containerized environment.

We summarize the contributions of this work as follows.

- We have implemented a service-based 5G architecture by containerizing both MANO components and network functions.
- We have demonstrated the slice selection and configuration mechanism with the 3GPP-compliant core networks.
- We have integrated bandwidth control to exercise E2E network orchestration and to enable QoS-aware network services.

The rest of this paper is organized in the following fashion. The next section briefs network slicing in 5G. We then present our design for the slicing of core and transport networks. Sec. IV briefs our E2E NS MANO and the detailed orchestration of slices. The last session concludes this paper.

II. Network Slicing in 5G

A. Network Slice Instance (NSI)

5G introduces three standard slice categories for different application scenarios: enhanced mobile broadband (eMBB), ultra-reliable and low-latency communication (URLLC), and massive Internet of things (mIoT). A network slice instance (NSI) is the realization of a certain category of NS. It consists of a set of network functions and the required compute, storage, and networking resources.

We may view the management of NSI from time and space perspectives. From the time perspective, the management of an NSI consists of four phases: preparation, commissioning, operation, and decommissioning [11]. From the space perspective, an E2E network slice in 5G mobile networks spans over three domains: the radio access network (RAN), the core network (CN), and the transport network (TN). Accordingly, an NS is divided into several *network slice subnets* (NSSs), one for each domain [10]. *Network slice subnet instances* (NSSIs) are instantiations of NSSs and can be interconnected to form a complete E2E NSI.

B. Radio Access Network (RAN) Domain

Network slicing on the RAN domain has been widely studied in academics and industries. According to the level of RAN resource isolation, RAN slicing may be roughly classified into three possible architectures [4]. In *slice-aware shared RAN* (e.g., [4]), the complete RAN is shared among slices with relatively little functional and performance isolation. In *slice-specific radio bearer*, network slices share only Physical (PHY) and Medium Access Control (MAC) layers in the user plane and the radio resource control (RRC) in the control plane. Some researchers proposed a two-level MAC scheduler for RAN slicing [12], [8]. Many other works also applied resource sharing polices at the MAC layer [5], [13]. In *slice-specific RAN*, only the air interface is shared among network slices. Either the spectrum is shared among slices or each slice is assigned dedicated frequency bands [14]. In this case, each slice can have a customized PHY and MAC.

Another aspect is the control and management of virtualized RAN resource. 5G-EmPOWER [13] provides a hypervisor for the isolation and sharing of virtulalizes RAN resource. It separates the control plane from user plane and supports multiple radio access technologies. 5G-EmPOWER is integrated in [7] as a RAN slicing solution.

C. Core Network (CN) Domain

The mobile core network serves as an operation and business system for network operators. It provides service connection, device authentication, and accounting. For the user equipment (UE) requesting network service in CN, the main task on the CN domain is to designate a network slice for directing, isolating, and shaping the UE traffic. For this, the MANO should create and configure an NSSI and allocate associated resources. A main consideration for the slice management is how to strike a balance between resource utilization and performance.

The creation of a CN NSSI entails the instantiation and configuration of all supporting network functions. As a cloud native design, 3GPP restructures the mobile core network from 4G EPC to 5G *service-based architecture* (SBA) [1] (Fig. 1). SBA decomposes control-plane network functions into microservices, which interact with each other through stateless HTTP. To facilitate the exploration of active network services, 3GPP uses a dedicated function, Network Repository Function (NRF), to maintain service profiles of network functions on the 5G control plane. Before a control-plane function (e.g., SMF in Fig. 1) starts operating on SBA, the function will register its service profiles to NRF (Step 0 in Fig. 1). Afterward, any other control-plane function (e.g., AMF in Fig. 1) pursuing such services will ask NRF for the service location with desired profiles (Step 3 in Fig. 1) and then send its request to the target function (Step 4 in Fig. 1).

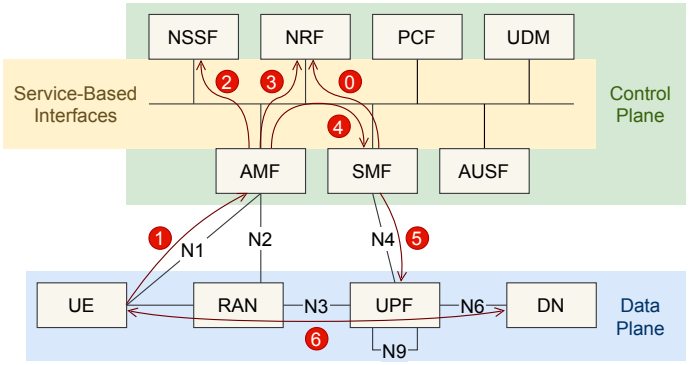

Fig. 1. Simplified UE Requested PDU Session Establishment Procedures in 3GPP Core Network Architecture

The procedure for CN to handle a network service request from a UE can be outlined as follows [9]. First, the UE initiates a Packet Data Unit (PDU) session establishment request to the core network. The request is handled by Access and Mobility Management Function (AMF) (Step 1 in Fig. 1). It contacts Network Slice Selection Function (NSSF) to identify the network slice of the UE (Step 2). AMF then selects an active Session Management Function (SMF) (Step 4 in Fig. 1) and delegates the management of the UE's PDU session to the SMF. The SMF then establishes a PDU session on an

appropriate User Plane Function (UPF) (Step 5 in Fig. 1). After completing the PDU session establishment, the UE can access network services through the given UPF (Step 6 in Fig. 1).

SBA provides a finer granularity of function sharing among multiple network slices: instead of sharing an entire core network or not, we can share some but not all microservices among network slices. Sharing a microservice among network slices increases resource utilization in the infrastructure but may increase signaling processing latency and thus cause performance interference among slices. On the other hand, creating a microservice instance for each slice may consume considerable computing resources but provides functional isolation. We shall present our design choices in the next section.

D. Transport Network (TN) Domain

The transport network domain includes inter-RAN, RAN-CN, and inter-CN networks. Network slicing on the TN domain often aims at RAN-CN networks related to the fronthaul and backhaul. The main tasks in TN slicing are to differentiate and isolate traffic flows among network slices and apply desired processing rules accordingly to meet slice-specific QoS requirements. A key technology to this goal is software-defined networking (SDN).

The QoS requirements under consideration include bandwidth, latency, and packet loss. SDN supports QoS in two primary ways. The first one exploits the global view to plan QoS-aware network resources allocations. The study in [15] calculates forwarding paths according to bandwidth and latency requirements of different network slices. However, the scheduling process requires significant computing resources on the control plane as the network topology becomes complicated and may not utilize networking resources efficiently with path allocation. The other way adopts the data-plane programmability and applies the precedence on outputting data. The work in [6] and [16] classifies and schedules traffic to several levels of priority queues. Ref. [16] further disaggregates packets from the same traffic flow into specific categories based on QoS demands and thus can achieve strict bandwidth limitation and guarantee.

E. MANagement and Orchestration (MANO) for E2E NS

MANO for E2E network slicing essentially involves the design of NSIs in accordance with service demands, orchestration of constituent NSSIs, and physical and logical resource isolation between network slices. The work in [6] proposes MANO with 5G Public-Private Partnership (5G PPP) SliceNet project. It also realizes QoS-aware network slicing on the prototyping programmable data plane. The work isolates resources by running multiple vEPC instances on CN but does not further discuss slice selection and orchestration. Besides, the virtualization management is not in line with the 3GPP and ETSI framework. The work in [5] implements E2E network slicing MANO in compliance with the 3GPP management systems, which operates virtualized functions on the cloud infrastructure. However, the use of EPC needs to realize the network slice selection function on the MANO framework, which is not compliant with the 3GPP specifications. The work in [8] mainly focuses on the implementation of RAN slicing. Although it also designs CN and TN slices to demonstrate eMBB, URLLC, and mIoT scenarios, it hardly discusses MANO and slice selection mechanisms.

III. Slicing of Core and Transport Networks

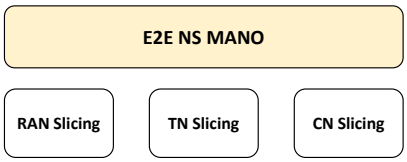

Fig. 2. MANO Framework for E2E Network Slicing

The following two sections present our design and implementation of a cloud-native MANO framework for E2E network slicing. The design consists of two major parts: E2E NS MANO and slicing of each domain (Fig. 2). This section first discusses the slicing of CN and TN networks. In the next section, we shall brief our E2E NS MANO and the detailed orchestration of CN and TN slices.

A. Core Network Slicing

We classify core network functions into two categories based on whether the function is shared among E2E NSs. Shared network functions, such as AMF and NSSF, in a common NSSI, are instantiated only once and are common to and serve requests from all UEs. These functions should have low processing loads, which are invoked only upon a new UE attachment, handover or other signaling procedures. Shared network functions may have independent scaling mechanisms based on the frequency of UE requests. On the other hand, dedicated network functions, or session NSSI, are slice-specific and instantiated once for each CN slice. Because different slices may have specific session management policies for their service types, we take SMF as a dedicated control-plane network function in our design. Unlike control-plane functions, data-plane functions (e.g., UPF) are supposed to handle every user PDU and are thus expected to be heavy-loaded. Therefore, we also take UPF as a dedicated network function.

When network functions are slice-specific, targeting the right instance for session setup becomes crucial. In Fig. 1, after AMF acquires the slice identifier associated with the UE from NSSF (Step 2), it queries the NRF for the SMF instance that corresponds to the identified CN slice (Step 3). After discovering an appropriate SMF, the AMF then requests the SMF to create a PDU session on the associated UPF, which is dedicated to the CN slice as well (Steps 4 and 5 in Fig. 1).

B. Transport Network Slicing

1) Traffic Types Classification: Two main tasks in TN slicing are classifying TN traffic into different types and enforcing respective management policies. The first task is essential

TABLE I
TRAFFIC ON INTERFACES ACROSS ACCESS AND CORE NETWORKS

Interface	N1/N2	N3
Crossing NFs	UE/RAN, AMF	RAN, UPF
Traffic Type	Control	Data
Protocol Stack	NGAP/SCTP	GTP-U/UDP
Characteristics	Mice but should not be starved	Large volume but tolerant to congestion and even loss

because traffic from all the interfaces across access and core network functions (N1, N2, and N3 in Fig. 1) is intermixed in TN. Table I compares traffic on those interfaces. Traffic on the N1/N2 interfaces between UE/RAN and the AMF is identified as control messages related to UE connection, registration, and mobility management, such as PDU session establishment requests. The control messages, in the NGAP/SCTP protocol, are mice traffic but should not be starved under congestion. By contrast, traffic on the N3 interface between RAN and the UPF is UE's data traffic in the GTP-U/UDP protocol, which is a large volume but more tolerant to network congestion in most scenarios. Additionally, the data traffic is expected to be sliced under the allocation of network resources based on service requirements.

2) Bandwidth Slicing Model: We classify TN traffic into control and slice-specific data traffic and enforce specific bandwidth policies with the help of SDN and programmable data plane. SDN switch distinguishes control traffic from data traffic by the following matching fields:

- Source-destination IP address pair: The pair is (AMF, gNodeB) or (gNodeB, AMF) in case of control traffic and (UPF, gNodeB) or (gNodeB, UPF) in case of data traffic. For bandwidth slicing on an inter-CN network with intermediate UPFs (I-UPFs), SDN switches need to identify the IP addresses of I-UPFs besides those of gNodeB and the local UPF.
- IP protocol: It has the value of 132 (SCTP) for control traffic and the value of 17 (UDP) for data traffic.

All control traffic is assigned to the TN slice with the highest priority. On the other hand, data traffic is scheduled to a bandwidth slice based on service demands.

Source-destination IP address pair also allows SDN switches to distinguish uplink from downlink data flows and apply separated bandwidth management rules. For instance, mIoT use cases may have a significant amount of uplink traffic, while eMBB use cases may occupy large downlink bandwidth.

3) QoS Enforcement: We integrated [16] into our transport network to enforce the QoS requirements on the programmable data plane. Our work extends the ONOS built-in Basic.p4 pipeline [17] by appending a Meter.p4 pipeline to it for slice identification and bandwidth control. The Meter.p4 pipeline consists of three pipeline stages (Fig. 3). In the first stage, traffic is classified with matching fields and differentiated into TN slices. Afterward, the next stage categorizes packets in the same TN slice into different priority levels (each with a

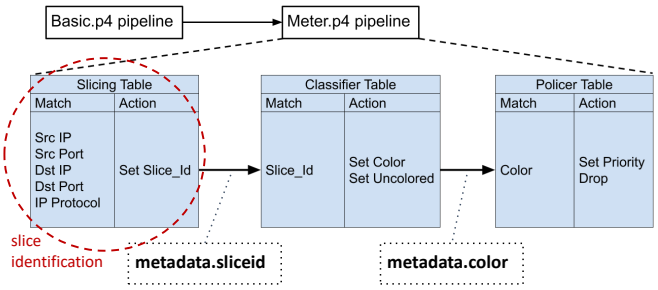

Fig. 3. Three pipeline stages to enforce bandwidth requirement

distinct "color") based on the rate-based policies. It is noted that all control traffic is assigned to the highest priority level in this stage. Finally, the last stage schedules the packets into output queues according to the previous priority levels and hence achieves differential bandwidth requirements (Fig. 4).

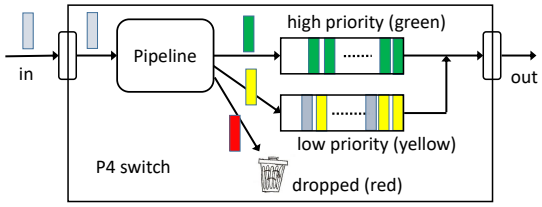

Fig. 4. Priority forwarding using two-level queues

4) Bandwidth Management: We implemented a Bandwidth Management application on the SDN controller to manage available bandwidth and apply TN slicing (Fig. 5). When the application receives the bandwidth slicing request, it first verifies that network resources along the routing path are available. If the requested bandwidth is granted, the application then installs the corresponding flow rules on programmable switches.

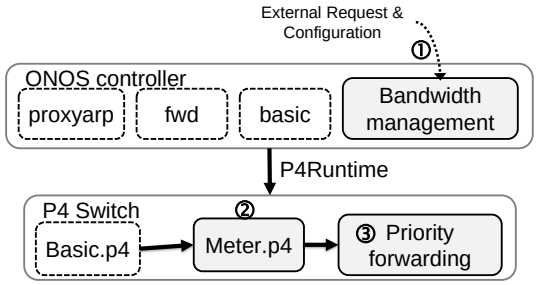

Fig. 5. The Bandwidth Management application

IV. CLOUD-NATIVE E2E NETWORK SLICING MANO

A. Infrastructure and Architecture

European Telecommunications Standards Institute (ETSI) proposes NFV-MANO framework [18] for mobile network

management. The framework consists of the following components:

- NFV Orchestrator (NFVO), which has two responsibilities; one is the orchestration of NFV infrastructure (NFVI) resources in an abstracted manner through Virtualized Infrastructure Manager (VIM), and the other is the lifecycle management of network services.
- VNF Manager (VNFM), which is responsible for the lifecycle management of VNF instances, including VNF instantiation, modification, scale in/out, and termination.
- Virtualized Infrastructure Manager (VIM), which is responsible for controlling and managing NFVI, including virtualized compute, storage, and network resources.

For network slicing management, 3GPP proposes three network slice management functions [10]:

- Communication Service Management Function (CSMF), which translates communication service requirements into network slice requirements.
- Network Slice Management Function (NSMF), which manages the lifecycle of NSIs and derives constituent network slice subnets from network slices.
- Network Slice Subnet Management Function (NSSMF), which manages the lifecycle of NSSIs.

Fig. 6 shows our infrastructure and MANO architecture, which follows the design principles of 3GPP management and ETSI NFV-MANO framework. In the infrastructure part, we containerize free5GC [19], an open-source core network in line with 3GPP Release 15 specifications, and run with Docker virtualization on the top of COTS servers. Kubernetes [20], a production-grade orchestrator for containerized applications, controls our virtualization resources and provides plentiful cloud management features, such as self-healing, service discovery, etc. To enforce bandwidth-aware flows with QoS requirements on TN, we deploy programmable switches and interconnect NSSIs with the programmable forwarding planes in P4 language. ONOS [21] is used as the SDN controller to forward traffic with centralized logic and expose bandwidth management interfaces to other MANO components for TN slicing. The placement of our MANO components within the 3GPP and ETSI framework is based on the major functionality, but in some cases would be vague. For instance, Kubernetes serves as VIM for orchestration of computing resources and however, it also plays the role of container orchestrator which can be treated as VNFM due to the lifecycle management of VNFs.

B. MANO Implementation

We implemented the slice management components in the proposed framework using the Kubernetes software extension, *Operator* pattern [22]. An Operator is to automate the deployment and management of a set of resources and applications beyond what Kubernetes originally provides. We use Operators to customize slice resources on Kubernetes and undertake automation tasks.

We did not implement CSMF, while the implemented *E2E Network Slice Operator* serves as NSMF and NSSMF

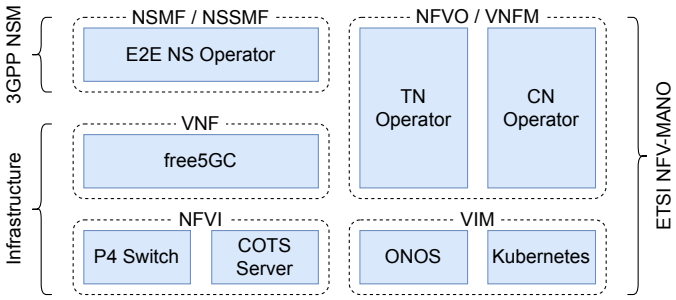

Fig. 6. Infrastructure and MANO Architecture

components. The E2E NS Operator controls the lifecycle of NSIs and NSSIs via the management services provided by NFVOs. It also handles network slice requests and slice-related information as well as activates and deactivates E2E network slices. As for the ETSI NFV-MANO framework, we implemented two special Operators, namely *CN Operator* and *TN Operator*, for the management of NSSIs in the CN and TN domains, respectively. The CN and the TN Operators play the roles of NFVO/VNFM for the CN and TN domains, respectively. Both Operators orchestrate the corresponding NSSIs when receiving management requests from the E2E NS Operator.

Fig. 7 shows the interaction between the E2E NS Operator and the CN and TN Operators. The E2E NS Operator receives and validates E2E slice requests with associated specifications submitted by an administrator or service consumer. If an E2E NS request is approved, the E2E NS Operator sends a request with CN slice specification to the CN Operator. After the CN Operator creates a CN slice, it returns the address information of AMF and UPF to the E2E NS Operator. With this information and other parameters such as bandwidth requirement, the E2E NS Operator then requests the TN Operator to create a TN slice.

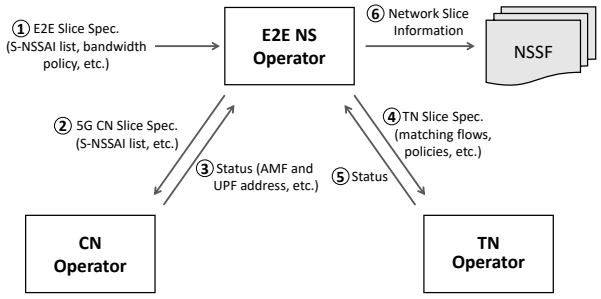

Fig. 7. Interaction between the E2E NS Operator and the CN and TN Operators

For CN slicing, the CN Operator deploys shared NFs and dedicated NFs of free5GC on the Kubernetes platform. For TN slicing, ONOS controller is the VIM that manages the virtualized infrastructure (i.e., virtual networks) in TN. The TN Operator configures the ONOS controller with matching rules and bandwidth requirements to create a virtual network.

All Operators and the ONOS controller are running in containers on Kubernetes to benefit from the elastic and reliable environment.

We implemented the proposed MANO framework with all open-source approaches to leverage the support of the collaborative public community. In addition, the separation of Operators in different slice subnets enables the MANO framework to easily support the orchestration for other NSSIs. Namely, for industrial and commercial demands, we can extend the E2E network slices to RAN and MEC domains by implementing the RAN and MEC Operators, respectively, and applying the proposed orchestrating fashions. For instance, the work in [23] demonstrated the automation of RAN slicing with the implementation of a top-level RAN Slicing Management Function (RSMF). RSMF has exactly the same functionality as the RAN Operator and partial E2E NS Operator in our design and can complement the proposed E2E slicing solution.

C. Network Slice Instance (NSI) Allocation

In this subsection, we detail the procedures of allocating an E2E NSI and compare the difference between on-boarding of the first and subsequent network slices.

1) Allocating the First NS: Fig. 8 shows the deployment details when administrators or other slicing management components request the first E2E network slice. The E2E NS Operator validates the received allocation request and checks if it needs to create a new E2E NSI for the request. Since there is no NSI initially, the E2E NS Operator will query the CN Operator for the creation of the common NSSI and a session NSSI. The CN Operator will build the free5GC shared NFs, e.g., AMF and NSSF, for the common NSSI. After that, the CN Operator creates dedicated NFs, namely SMF and UPF, for the session NSSI. We provide the common NSSI with a high-priority TN slice and the session NSSI with an on-demand TN service. The E2E NS Operator then queries the TN Operator for configuring the ONOS controller to create two TN slices, one for each NSSI. ONOS further installs bandwidth-aware flows on the P4 fabric. So far both the shared E2E NSI and the first E2E NSI for UE are ready to provide network services. Finally, the E2E NS Operator will activate the E2E NSI by updating network slice selection information on the NSSF through the management interface. The control traffic now can reach the AMF through the TN slice with the highest priority and establish PDU sessions for UEs, whereas the data traffic is forwarded to the dedicated UPF with specified QoS requirements.

2) Allocating Subsequent NS: For subsequent E2E network slices, our MANO framework deploys E2E NSIs as shown in Fig. 8, with a slight difference from the allocation of the first NSI. When the E2E NS Operator receives an NS allocation request, it will first query the CN Operator to create a CN slice as well. Since the shared NFs are already activated, the CN Operator will not create the common NSSI but proceed into the creation of a new session NSSI. After the CN slice is ready, the configuration of the TN slice is performed in the same way, and so is the following activation with the NSSF.

V. Performance Evaluation

A. Environment Setup

We modified the free5GC test framework and implemented a RAN simulation (RANSIM) to emulate all signaling messages for UE to register and set up PDU sessions on 5G core networks. RANSIM runs on a white box server and connects to P4 fabric, as shown in Fig. 8. On the other side of the P4 fabric, we deploy Kubernetes along with the MANO framework (including Operators and ONOS) on another white box server. To accelerate packet processing in the virtualization environment, we adopt SR-IOV techniques on the interfaces connecting the AMF and UPFs to the P4 fabric. In the P4 fabric, the Meter.p4 pipeline is used for slice identification and bandwidth control.

B. Results and Discussion

1) Deployment Time: Deployment time is crucial to building a resilient network that can mitigate service disruption under failure and scale out rapidly when overloading. The deployment of the MANO framework took 71.02 seconds. The instantiation of E2E network slices took 52.15 seconds for the first one but only 33.21 seconds for the subsequent one, which is relatively short. The deployment time of the first E2E network slice was longer due to the additional creation of the common NSSI.

However, the performance result should not be overemphasized because free5GC currently only implements essential functionalities, lacking some features (such as charging or QoS policies) that could be crucial in some scenarios. Also, a real RAN slicing may have a significant impact on performance metrics. Therefore, the current result only serves the purpose to see how well an all open-source approach with limit but indispensable functionalities can perform.

2) Resource Overhead and Scalability Concern: Fig. 9 depicts CPU and memory usages when deploying our framework on a white box server with 40 cores of Intel E5-2630 processor and 128 GB memory. We built the MANO framework at the fifth second, and the first and the second E2E network slices at the 85-th and the 145-th seconds, respectively.

The CPU utilization increased sharply during the deployment of the MANO components, i.e., the Operators and ONOS controller, and during instantiations of the shared and dedicated NFs. Afterward, the CPU utilization decreased and remained relatively low. However, the utilization continuously increased by about 3% more after each instantiation of the E2E NSIs. This increase of the CPU utilization is because the UPF was actively expecting traffic from UE.

As for memory usages, it highly depends on the applications we containerize. Specifically, ONOS is a production-grade SDN controller with a highly abstracted and modular networking core, which consumes considerable memory resources after startup. Besides, a short sharp peak of memory usage appeared at the 104-th second due to the simultaneous initializations of free5GC shared NFs and the release of memory resources afterward.

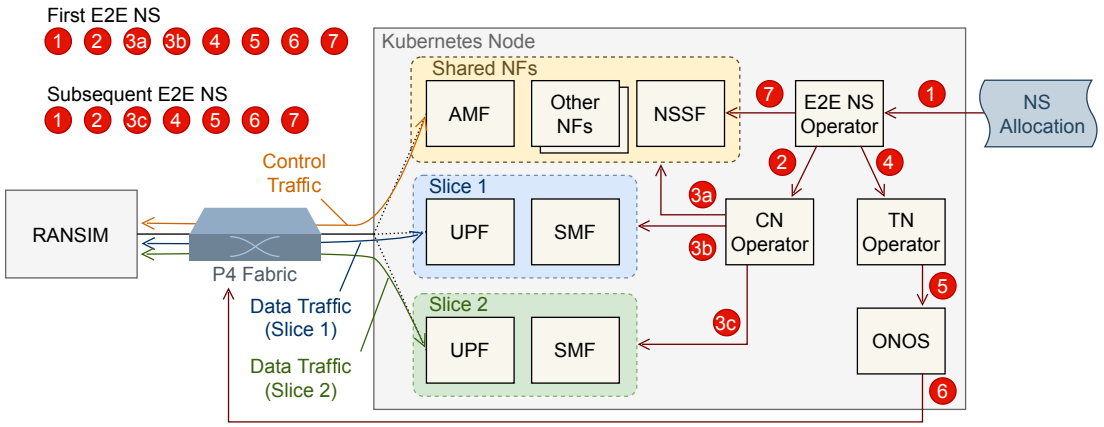

Fig. 8. Deploying Details of the First and Subsequent Network Slices

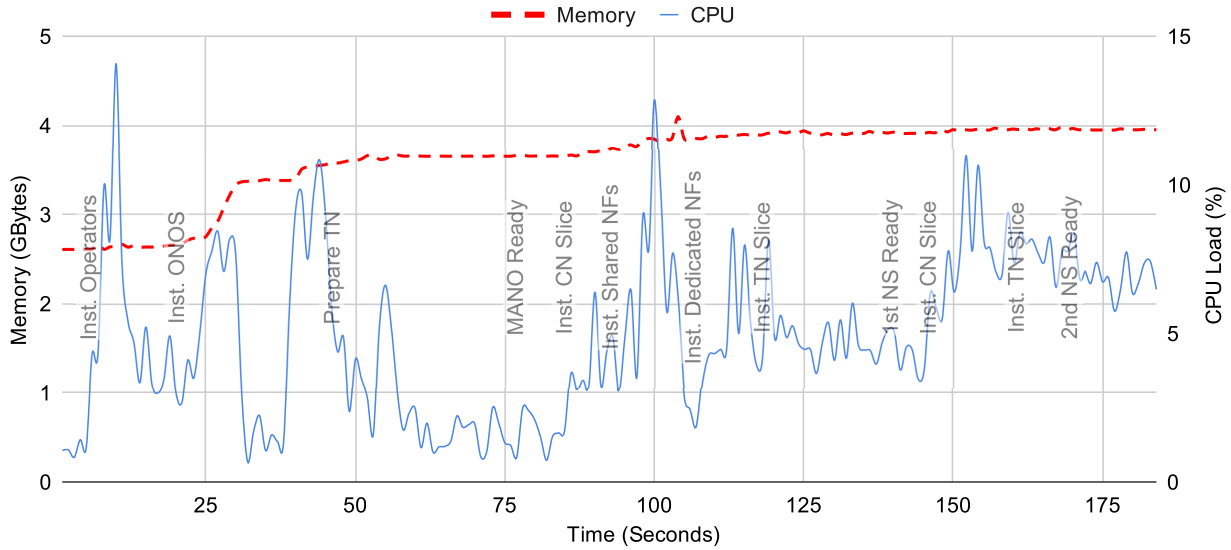

Fig. 9. CPU Load and Memory Usage of Deploying the MANO Framework and E2E Network Slices

Overall, the resource overhead of the proposed framework is satisfactory. However, the current implementation puts all virtualized CN functions and the MANO components in a single white box, which may perform unsatisfactorily when considerable slices are to be created. A feasible solution to this scalability problem is to distribute session-specific functions (UPF and SMF) to a Kubernetes cluster rather than a single server.

3) Traffic Throughput under TN Slicing: In our design, the TN slicing integrates bandwidth management to provide QoS-aware network slices. We evaluated UE traffic with specified bandwidth policies to verify slicing effects on networking resource isolation under the orchestration. We enforced bandwidth slices on downlink data of two E2E network slices, namely Slice X and Slice Y, with the maximum rates limited to 5 Mbps and 10 Mbps, respectively. Then we imposed the downlink traffic of 15 Mbps on Slice X at the sixth second and Slice Y at the tenth second. Fig. 10 shows the downlink throughputs on the core and access sides of TN. The throughput at the core side fluctuated around 15 Mbps, while the access side received only the corresponding limited rates of data traffic. The results demonstrate not only the effective bandwidth control on TN slicing but also the practical E2E network slicing through orchestration.

VI. Conclusions

This paper shows the design and implementation of a MANO framework for the management automation of 5G E2E network slicing with state-of-the-art cloud-native principles and open-source approaches. The MANO framework follows the slice management architecture and orchestration procedures introduced by 3GPP and ETSI. We containerized both MANO components and network functions, which were deployed in virtualization environments to realize high flexibility and scalability. In the CN slicing, we have demonstrated the slice selection and configuration mechanism with the 3GPP-

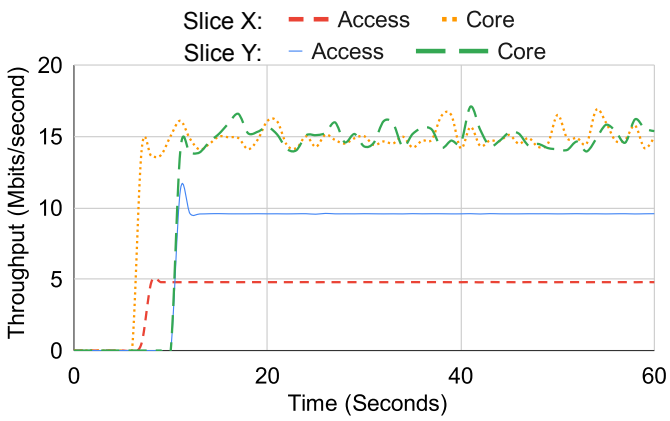

Fig. 10. Downlink Throughput of Two Network Slices with Distinct Bandwidth Policies

compliant core networks. In the TN slicing, we have integrated bandwidth control to exercise E2E network orchestration and to enable QoS-aware network services for tenants with E2E network slicing. Experimental results show that the framework performs well in terms of deployment time and resource overhead. The resulting MANO framework, together with the E2E network slicing mechanism, provides a PoC implementation of the 5G virtualization infrastructure.

Acknowledgment

This work was supported in part by The Featured Areas Research Center Program within the Framework of the Higher Education Sprout Project by the Ministry of Education, Taiwan; in part by the Ministry of Science and Technology, Taiwan, under Grants 110-2221-E-A49-044-MY3, 110-2221-E-A49-064-MY3 and 111-2218-E-011-014, and in part by the Ministry of Economic Affairs, Taiwan, under Grant 107-EC-17-A-02-S5-007.

References

[1] "System architecture for the 5G system (5GS)," 3GPP, TS 23.501, V16.4.0, Release 16, Mar. 2020.
[2] P. Rost, A. Banchs, I. Berberana, M. Breitbach, M. Doll, H. Droste, C. Mannweiler, M. A. Puente, K. Samdanis, and B. Sayadi, "Mobile network architecture evolution toward 5G," *IEEE Commun. Mag.*, pp. 84–91, May 2016.
[3] "CNCF cloud native definition v1.0," Jun. 2018.
[4] G. Garcia-Aviles, M. Gramaglia, P. Serrano, and A. Banchs, "POSENS: A practical open source solution for end-to-end network slicing," *IEEE Wireless Communications*, vol. 25, no. 5, pp. 30–37, Oct. 2018.
[5] I. Afolabi, T. Taleb, P. A. Frangoudis, M. Bagaa, and A. Ksentini, "Network slicing-based customization of 5G mobile services," *IEEE Network*, vol. 33, no. 5, pp. 134–141, 2019.
[6] Q. Wang, J. Alcaraz-Calero, R. Ricart-Sanchez, M. B. Weiss, A. Gavras, N. Nikaein, X. Vasilakos, B. Giacomo, G. Pietro, M. H. Mark Roddy, P. Walsh, T. Truong, Z. Bozakov, K. Koutsopoulos, P. Neves, C. Patachia-Sultanoiu, M. Iordache, E. Oproiu, I. G. B. Yahia, C. Angelo, C. Zotti, G. Celozzi, D. Morris, R. Figueiredo, D. Lorenz, S. Spadaro, G. Agapiou, A. Aleixo, and C. Lomba, "Enable advanced QoS-aware network slicing in 5G networks for slice-based media use cases," *IEEE Trans. on Broadcasting*, vol. 65, no. 2, pp. 444–453, Jun. 2019.
[7] A. Esmaeily, K. Kralevska, and D. Gligoroski, "A cloud-based SDN/NFV testbed for end-to-end network slicing in 4G/5G," in *Proc. 6th IEEE Conf. on Network Softwarization*, Ghent, Belgium, 2020, pp. 29–35.
[8] X. Li, R. Ni, J. Chen, Y. Lyu, Z. Rong, and R. Du, "End-to-end network slicing in radio access network, transport network and core network domains," *IEEE Access*, vol. 21, no. 8, pp. 29 525–29 537, 2020.
[9] "Procedures for the 5G system (5GS)," 3GPP, TS 23.502, V16.4.0, Release 16, Mar. 2020.
[10] "Telecommunication management; study on management and orchestration of network slicing for next generation network," 3GPP, TR 28.801, V15.1.0, Release 15, Jan. 2018.
[11] "Management and orchestration; concepts, use cases and requirements," 3GPP, TS 28.530, V16.1.0, Release 16, Dec. 2019.
[12] A. Ksentini and N. Nikaein, "Toward enforcing network slicing on RAN: Flexibility and resources abstraction," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 102–108, Jun. 2017.
[13] E. Coronado, S. N. Khan, and R. Riggio, "5G-EmPOWER: A software-defined networking platform for 5G radio access networks," *IEEE Trans. on Network and Service Management*, vol. 16, no. 2, pp. 715–728, Jun. 2019.
[14] "Network sharing; architecture and functional description," 3GPP, TR 23.251, 2018.
[15] Z. Shu and T. Taleb, "A novel QoS framework for network slicing in 5G and beyond networks based on SDN and NFV," *IEEE Network*, vol. 34, no. 3, pp. 256–263, 2020.
[16] Y.-W. Chen, L.-H. Yen, W.-C. Wang, C.-A. Chuang, Y.-S. Liu, and C.-C. Tseng, "P4-enabled bandwidth management," in *Proc. The 20th Asia-Pacific Network Operations and Management Symp.*, Matsue, Japan, Sep. 2019.
[17] "ONOS basic pipeline," https://github.com/opennetworkinglab/onos/tree/master/pipelines/basic, accessed: 2021-06-14.
[18] "Network functions virtualisation (NFV); management and orchestration," ETSI, GS NFV-MAN 001, V1.1.1, Dec. 2014.
[19] "free5gc," https://www.free5gc.org, accessed: 2021-06-14.
[20] "kubernetes: Production-grade container orchestration," https://kubernetes.io, accessed: 2021-06-14.
[21] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, and G. Parulkar, "ONOS: Towards an open, distributed SDN OS," in *Proc. 3rd Workshop on Hot Topics in Software Defined Networking*, Chicago, IL, USA, Aug. 2014.
[22] "Operator framework," https://github.com/operator-framework, accessed: 2021-06-14.
[23] R. Ferrús, O. Sallent, J. Pérez-Romero, and R. Agustí, "On the automation of RAN slicing provisioning: solution framework and applicability examples," *EURASIP Journal on Wireless Communications and Networking*, 2019.