OpenSCaaS: An Open Service Chain as a Service Platform Toward the Integration of SDN and NFV

Wanfu Ding, Wen Qi, Jianping Wang, and Biao Chen

Abstract

Service providers today rely on service-chaining policy enforced by omnipresent middleboxes (MBoxes) to achieve crucial performance and security guarantees in their networks. On the other hand, they are currently struggling to satisfy growing traffic demands on their traditional services. Given the promise of emerging technologies, such as SDN and NFV, we argue that the enforcement of service-chaining policy can greatly benefit from the integration of them. However, it is challenging to synthesize these two technologies due to the need to meet the requirements of flexibility, scalability, and auto-provisioning without modifying SDN standards (e.g. OpenFlow) and MBoxes. In this article we present the design of OpenSCaaS, an open platform for service chain as a service, by using the tangible capabilities of SDN together with NFV. OpenSCaaS allows service-providers to design a general architecture based on SDN to achieve flexibility, encapsulate Service Chain Identifier (SC-ID) based on Source MAC to guarantee scalability, and realize auto-provisioning based on NFV to offer adaptability. In doing so, we take a significant step to address industry and academic concerns about the integration of SDN and NFV.

Service-chaining policy, which dominates fixed broadband (FBB) and mobile broadband (MBB) for network operators and data centers (DC) for enterprises, has taken on new importance with the rise of SDN and NFV. The so called service-chaining policy refers to the term that describes executing multiple service functions in an ordered list to guarantee performance and security requirements. The enforcement of service-chaining policy relies on two categories of network devices: forwarding devices (L2-L3, e.g. switch, router) and MBoxes (L4-L7, e.g. NAT, Firewall). The traffic flows are steered to the predefined sequence via forwarding devices and processed via MBoxes, respectively.

Unfortunately, even though the service providers benefit a lot from the service-chaining policy, they are struggling to survive due to two intolerable burdens over a long period of time [1]: high operating expense (OPEX) due to manual configuration of network devices, and high capital expense (CAPEX) due to expenditure of network devices. SDN and NFV are two emerging notions that bring new solutions for these two burdens. SDN aims to offer the ability to control and program the network through centralized network management, decoupling the control plane and the data plane. NFV aims to offer the ability to reduce the device costs through leveraging hardware with high generality and IT virtualization technologies. SDN and NFV create new opportunities as well as challenges for traditional service-chaining policy (which will be discussed later).

This article presents the design of OpenSCaaS, a solution integrating SDN with NFV for enforcing service-chaining policy. OpenSCaaS provides a flexible, scalable, and adaptable solution to tackle the introduced challenges.

The rest of the article is organized as follows. We introduce the motivation and related work in the following section. Then we present an overview of OpenSCaaS, and we explore the optional candidate of SC-ID and fast auto-provisioning. We then evaluate the benefits of OpenSCaaS, before highlighting our conclusions and future work.

Motivation and Related Work

In this section we aim to induce the design principles for achieving a holonomic solution of service-chaining provision. First, we exemplify a common service-chaining scenario and present the cumbersome solution of a traditional service chain. Then we conclude the design requirements which cannot be satisfied by the current solutions.

Table 1 illustrates the various needs of three service-chaining policies. In this case, the video service requires its traffic to pass through cache, firewall, and NAT; the HTTP service requires its traffic to pass through firewall and NAT; the encrypted service requires its traffic to pass through NAT. As to this scenario, there are two typical service-chaining solutions today (shown in Fig. 1).

Wanfu Ding, Wen Qi, and Jianping Wang are with City University of Hong Kong.

Biao Chen is with University of Macau.

0890-8044/15/$25.00 © 2015 IEEE

Cascading Solution: This solution cascades all MBoxes in the service provider networks. As shown in Fig. 1a, the MBoxes are chained in a serial manner. Despite its simplification, it results in many deficiencies, many of which arise from the fact that all the traffic flows have to pass through all MBoxes, rather than the desired ones. First, it requires considerable manual effort and operator expertise to insert MBoxes or upgrade existing ones. Second, a failure in any MBox may interrupt the entire service-chaining network. Third, processing and forwarding all the traffic flows for every MBox increases cost.

Branching Solution: In this solution the service chains are designed in advance and then traffic is branched to the corresponding service chain. As shown in Fig. 1b, the traffic flows are classified by deep packet inspection (DPI). Compared with the cascading deployment described above, the branching solution has been improved to solve the aforementioned problems. Despite its usefulness, it still has some shortcomings. First, the DPI device may be the bottleneck when too many traffic flows go through service-provider networks. Second, it is hard to make service-chaining adjustments, such as creating or removing MBoxes from an existing service chain. Third, it may require enormous costs since the MBoxes cannot be multiplexed.

Current service-chaining solutions have some problems [2], such as poor scalability and high operating cost. Thus, the academy and industry are looking for a new service-chaining solution with strong flexibility and scalability. Here, we conclude that a new solution must simultaneously meet the following requirements:

- **Flexibility:** Service providers can design the layout of service chains without considering the physical network.
- **Scalability:** Service providers can augment the amounts of service chains without worrying about constraining flow tables.
- **Adaptability:** Service-providers can do fast enforcement of service chains without operating configuration manually.

In this article we propose to build an open and systematic service-chaining platform which:

- Designs a general architecture based on SDN to achieve flexibility.
- Encapsulates SC-ID based on Src-MAC to guarantee scalability.
- Realizes auto-provisioning based on NFV to offer adaptability.

Next we describe how OpenSCaaS fulfills the above requirements.s

Flexibility of Service Chain based on SDN

Traditional solutions are inflexible because service-chaining policy is executed in a manual mode. Thus, the service chain should be more flexible owing to the growth of traffic flows. Through decoupling the control plane and the data plane, SDN maintains a global view of the network and enables the network control to become flexibly programmable. Thus, service-chaining solutions combined with SDN have emerged to provide flexible adjustments. The related solutions can be divided into two groups:

- **Single-controller solutions** manage and control the devices of the data plane with only one controller. OpenNF [3] presents a unified control plane architecture to manage both the network forwarding state and the internal network function (NF) state, which enables MBoxes to migrate and recover.
- **Multiple-controller solutions** manage and control network devices and MBoxes with more than one controller. Stratos [4] focuses on a network-aware orchestration layer composed of a forwarding controller, which forwards traffic flows according to the service-chaining specification, and an MBox controller, which monitors application performance and receives resource statistics from MBox.

Service chain	Cache	Firewall	NAT
Video service	✓	✓	✓
HTTP service		✓	✓
Encrypted service			✓

Table 1. Different services with different service-chaining requirements.

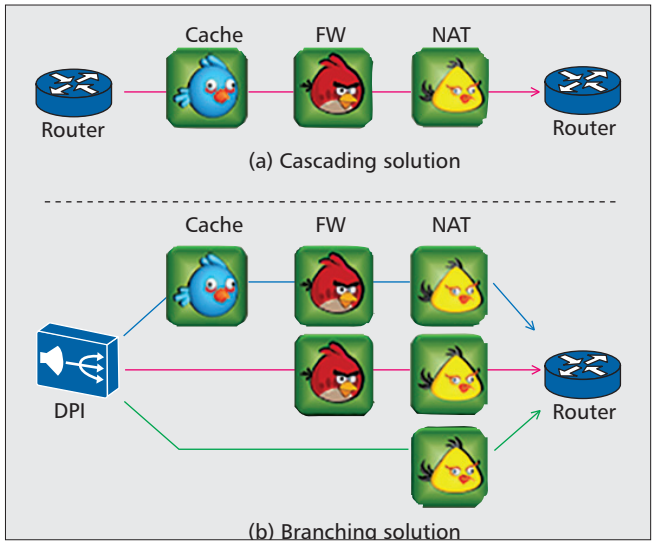

Figure 1. Traditional solutions for service chain.

Compared with the single-controller solution, the multiple-controller solution, which separates the forwarding controller and the MBox controller, is more dependable because it can prevent a single point of failure. OpenSCaaS adopts the triple-controller architecture to achieve flexibility (we elaborate on this later).

Scalability of Service Chain based on SC-ID

TCAM is one special type of constrained resource available for installing forwarding rules in SDN switches. Thus, we need to account for saving flow table capacity of SDN devices through aggregating micro-flow for high scalability. The related solutions can be divided into two groups.

MBox-Based SC-ID Solution: In this category of solutions, MBoxes add SC-ID to the incoming traffic flows (i.e. MBoxes are not only generators but also consumers of SC-ID, and SDN switches are only consumers). FlowTags [5] develops tag-enhanced MBoxes to guarantee two key SDN tenets (OriginBinding and PathsFollowPolicy). Those tags are generated by the first MBox and consumed by other MBoxes, and in the meantime, SDN-capable switches use the tags as part of their forwarding actions.

Switch-Based SC-ID Solution: In this category of solutions, SDN switches add SC-ID to the incoming traffic flows (i.e. SDN switches are not only generators but also consumers of SC-ID). DOA [6] and NSH [7] focus on adding a new header in traffic flows to reap the benefits of network-level MBoxes without their harmful side effects. While both of them are

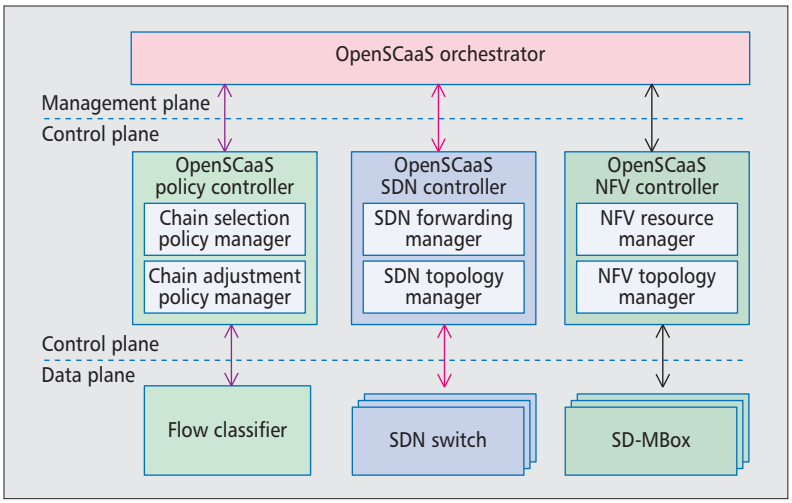

Figure 2. Overview of OpenSCaaS.

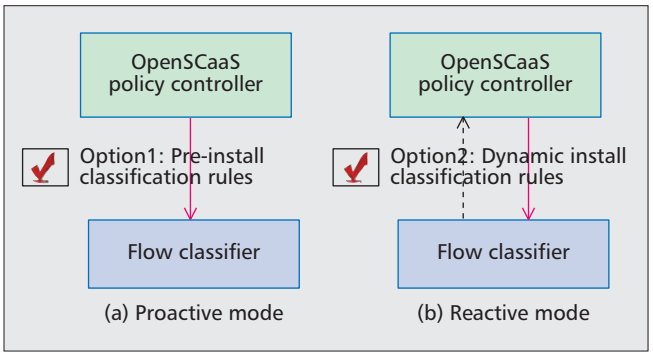

Figure 3. Two distribution modes for flow classification rules.

proposed for traditional networks, they can also be applied to SDN networks where the action of inserting a header can be easily implemented by SDN switches.

Compared with the MBox-based solution, the switch-based solution, which keeps MBoxes intact, is more reasonable because it can protect the service provider's investment. OpenSCaaS adopts switch-based SC-ID solution and adds SC-ID to traffic flows with Flow Classifier. Furthermore, we adopt Src-MAC as SC-ID due to its sufficient length to achieve scalability (we elaborate on this below).

Adaptability of Service Chain based on NFV

The enforcement of traditional service-chaining policy is highly complex. This complexity originates from the need to carefully plan the network topology, and manually set up rules to steer traffic flows through the desired sequence of MBoxes. In this response, NFV offers a promising alternative to tackle the challenges based on commodity hardware and virtualization technologies. The related solutions can be divided into the following two groups.

Single-OS Solution: Provides the realization of a service chain in one single OS. xOMB [8] demonstrates the feasibility of constructing extensible MBoxes with commodity servers and OSes, similar to prior research on software MBoxes [9–11]. Through employing general programmable network processing pipelines, xOMB realizes a service chain in one single OS.

Multiple-OS Solution: Provides the realization of a service chain in multiple OSes. Jingling [12] is an architecture that adds functionality to the network via outsourcing. Resorting to placing MBox in a trusted virtual machine (VM), Jingling provides each new feature by a new dedicated virtual box.

Compared with a single-OS solution, the multiple-OS solution, which uses virtualization technology to isolate MBoxes, is more dependable because it can prevent a single point of failure. OpenSCaaS adopts the multiple-OS solution to achieve adaptability by automatically creating software-defined MBoxes (SD-MBoxes) to process traffic flows, and automatically setting up flow rules to steer traffic flows (we elaborate on this below).

OpenSCaaS Overview

Our goal in this article is to introduce a new architecture, called OpenSCaaS, building on the requirements discussed above without making any modification to SDN standards and MBoxes. Figure 2 gives an overview of the OpenSCaaS architecture, showing the various components for different planes and the interactions between them.

OpenSCaaS Management Plane

Through decoupling the logical service chain and physical network, operators can customize the desired and holonomic service chains by operating logical MBoxes without considering physical network deployment.

OpenSCaaS Control Plane

Through decoupling the control plane and the data plane with a standard networking interface, operators do not need to configure the forwarding devices manually. Furthermore, the control plane can be divided into three parts as follows.

Policy Controller: OpenSCaaS closely manages the classification of incoming traffic flows with the policy controller, which acquires policy contexts from the orchestrator, and then sends flow classification rules to the flow classifier in two modes (Fig. 3):

- Proactive mode: As shown in Fig. 3a, the policy controller proactively pre-installs flow classification rules in the flow classifier.
- Reactive mode: As shown in Fig. 3b, the policy controller installs flow classification rules reactively in the flow classifier according to the first packet of a new flow, which is sent from the flow classifier.

SDN Controller: The SDN switches are programmed by the centralized SDN controller and forward the coming traffic flows through the desired MBoxes. Through the physical topology information of SDN switches and MBoxes, the SDN controller installs flow tables in SDN switches by the southbound interface.

NFV Controller: The logic service chains are dynamically mapped to physical resources by the NFV controller. The NFV resource manager allocates resources (e.g. CPU, memory, etc.) and then sends consolidated lightweight ClickOS-based VM images to commodity servers. Besides, the NFV resource manager will synchronize the new configuration profile (e.g. characteristic of SD-MBox, address, etc.) to mainframes. (Note: the SDN controller and the NFV controller will exchange topology information with each other by the topology manager.)

OpenSCaaS Data Plane

Through decoupling the SDN switches and the SD-MBoxes, service-providers can achieve high dependability because these elements are managed by different con-

trollers to forward and process packets respectively.

Flow Classifier and SDN Switches: To save flow table space, OpenSCaaS first classifies all the incoming traffic flows with the newly added flow classifier, and then the classified traffic tagged with SC-ID will be forwarded by SDN switches. The candidates of SC-ID have multiple choices, and in this article we choose the Source-MAC (Src-MAC) field as SC-ID (we elaborate on this later).

SD-MBox: SD-MBox can be easily adapted to the desired MBox through the VM image and the configuration profile sent from the NFV controller (we elaborate on this later).

Field / Requirement	Tag		New Header	L2	L3		L4
	MPLS	VLAN	NSH	Src-Mac	ToS (DS)	IPv4/v6 option	TCP option
Whether no switch modification	✓	✓	×	✓	✓	×	×
Whether meet length requirement	✓	×	✓	✓	×	✓	✓
Whether no conflict with existing service	×	×	✓	✓	×	✓	✓
Whether no Mbox modification	×	✓	×	✓	×	×	×

Table 2. Candidate solutions for SC-ID.

The Optional Candidate of SC-ID

This section aims at analyzing the various selections of SC-ID, as well as providing one optional recommendation. SC-ID must simultaneously meet four requirements:

1. **No switch modification:** Since the SDN protocol has been standardized, we cannot make any modification to SDN switches.
2. **Meet length requirement:** A 32-bit field would meet the demands for deployment in a very large-scale network [13].
3. **No conflict with existing service:** Conflicting with existing services will send networks into chaos.
4. **No MBox modification:** As there are considerable existing legacy MBoxes, we cannot make any modifications to MBoxes.

As shown in Table 2, we make a detailed comparison of the existing options of SC-ID which can be divided into three groups as follows:

- **Tag Solution:** Even though the MPLS label meets the length need, MPLS is not supported by MBoxes. The VLAN-tag is supported by MBoxes, but its size is 12 bits less than the required 32 bits, and furthermore, it may conflict with existing VLAN services.
- **New Header Solution:** Even though DOA [6] and NSH [14] are compatible with existing services and meet the 32-bit length requirement, they may induce large overhead due to the modifications to SDN switches and MBoxes.
- **Existing Field Solution:** This solution can be divided into L2, L3, and L4. We argue that Src-MAC (L2) is the optional candidate for SC-ID because it satisfies all the requirements for SC-ID. Compared with Src-MAC, the L3 solution, such as 6-bit DS (part of the 8-bit ToS) and redefined new option for IPv4/v6, violates the principles of selections. ToS could not meet the length requirement and may conflict with existing services (e.g. TE or QoS). The IPv4/v6 solution needs support from vendors, and thus needs to modify SDN switches and MBoxes.

To illustrate how the Src-MAC solution works, we revisit the example from the second section in Table 1, and in the simplest case, we use hop-by-hop forwarding at every SDN switch along a physical sequence of MBoxes as shown in Fig. 4. As mentioned, there are three traffic flows that need to be processed according to three different service-chaining policies. First, the service-provider customizes three service chains through the OpenSCaaS orchestrator, and then the orchestrator automatically generates policy contexts (e.g. classification information) and sends them to corresponding controllers. Third, flow classification rules, forwarding rules, and configuration profiles will be sent to the flow classifier, SDN switches, and SD-MBoxes by the OpenSCaaS policy controller, the SDN controller, and the NFV controller separately. Here we should note that flow classification rules are sent to the flow classifier in the mixed mode. That is to say, the policy controller proactively pre-installs flow classification rules if the flow rules are known in advance, and if not, then reactively installs flow classification rules according to new incoming flows. Finally, the incoming traffic flows are steered to the MBoxes according to the pre-defined service-chain specifications.

Fast Auto-Provisioning of Service Chain

Having introduced the overview of OpenSCaaS and evaluated the candidate solution of SC-ID, we now describe how to achieve fast auto-provisioning of a service chain.

Networks today rely on MBoxes to provide critical performance and security capabilities. Despite their usefulness, these hardware-based MBoxes bring many problems, such as being expensive to purchase, complex to manage, and hard to scale up/down and out/in with shifting demand. To address these issues, the notion of NFV has been proposed recently to shift MBox processing from hardware appliances to software running on inexpensive, commodity hardware. Thus, the presence of software-based SD-MBoxes has exploded. They are essential to network operators, supporting a diverse set of functions ranging from improving performance (e.g. Squid [11] as proxy and accelerator), enhancing security (e.g. Snort [10] as NIDS/NIPS), and shaping traffic (e.g. BalanceNG [9] as Load Balancer).

Despite their varieties, two principles limit SD-MBoxes to realize auto-provisioning:

- **Functional Consolidation.** We need an extensible "catholicon" to cover all functions of MBox processing. Unfortunately, SD-MBox can achieve only one single function.
- **Excellent Performance.** We need a minimalistic SD-MBox equivalent to hardware MBoxes not only in function but also in performance.

Unfortunately, SD-MBoxes cannot meet the performance requirement in that SD-MBoxes, similar to Click [15], are running on commodity OSes (e.g. Linux), and hence inherit the overhead of commodity OSes simultaneously. Click allows operators to customize a suitable set of processing elements, which are "stitched" together, to act as complex MBoxes at run-time. Click is flexible enough for MBox provisioning based on commodity servers. ClickOS [14] creates a tailor-made guest operating system to turn Click into reality, and furthermore, implements a software BRAS and a carrier-

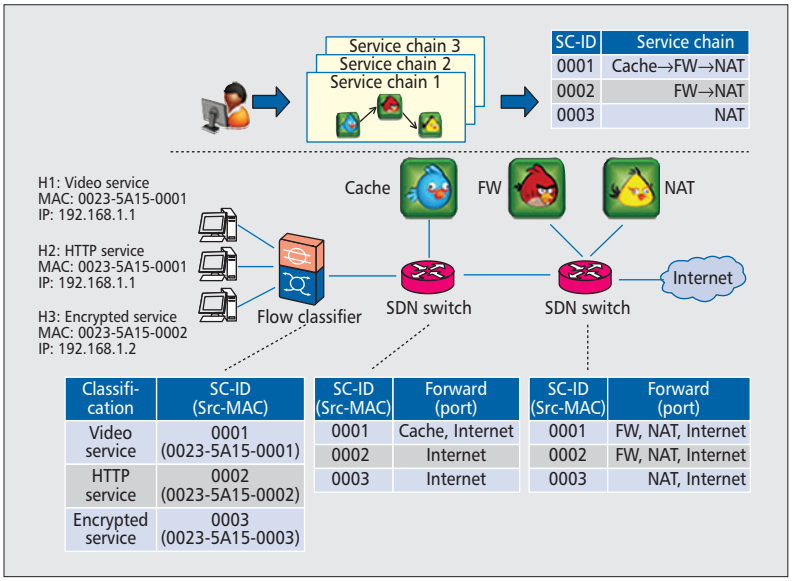

Figure 4. Example to illustrate how Src-MAC solution makes SC-ID feasible.

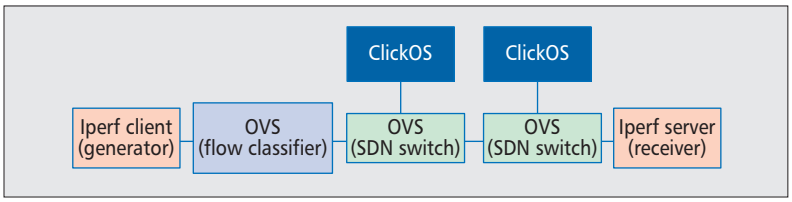

Figure 5. Experimental scenario in GiLAN.

grade NAT to show that ClickOS delivers production-level performance when running real MBox functionality. Next we illustrate our solution to realize fast auto-provisioning combined with ClickOS.

Fast Auto-Provisioning Solution: First, the service provider defines the service-chaining policy through the orchestrator, which will send a notification to the NFV controller to provide network functions. In response, the NFV resource manager allocates resources (e.g. CPU, memory, etc.) and then sends consolidated lightweight ClickOS-based VM images to commodity servers that will create and run a new VM instance. After that, the NFV controller synchronizes the new configuration profile (e.g. characteristic of SD-MBox, address, etc.) to servers, and at the same time, the NFV topology manager sends topology adjustment information to the SDN controller through interfaces. To the end, the SDN controller generates service-chaining flow tables and sends them to SDN switches, which will steer traffic flows to the new SD-MBoxes.

Evaluation

In this section we first examine the correctness of the OpenSCaaS data plane to realize complex service chains in the presence of dynamics, and then highlight some of the benefits of OpenSCaaS for service-chaining policy enforcement.

Functionality Evaluation

We evaluate OpenSCaaS' ability to satisfy the requirement of functionality correctness. For this experiment, we use three servers with Intel quad-core i3 3.30GHz CPU and 4.00 GB DRAM. One server runs five VMs with Linux 3.8.0: two VMs run Iperf Client and Iperf Server, between pairs of VMs, to generate and receive background traffic flows respectively; the remaining three VMs run Open vSwitch (OVS) as the flow classifier and SDN switches. The version of OVS is 2.1.0, which supports OpenFlow1.3 protocol, as the OpenFlow1.3 specification provides an action type (OFPAT_SET_FIELD), which can replace the value of Src-MAC (OXM_OF_ETH_SRC) with SC-ID. The other two servers run ClickOS VMs, which increase the number of intermediate ClickOS VMs to process flows. Our evaluation is conducted in a small, typical GiLAN testbed (Fig. 5).

We use Iperf Client to generate packets at a speed of 50-100 requests/second for 10 minutes and vary the request sizes for each run from 10KB to 100KB; we also dynamically construct several service chains consisting of multiple MBoxes. Comparing the generated packets and corresponding received packets, the results prove the correctness of our solution, as expected.

Benefits of OpenSCaaS

Next, we prove the abilities of OpenSCaaS. Integrated with SDN with NFV, the OpenSCaaS system has the following benefits.

The Ability to User-Define Service Chain: Using traditional solutions, which require manual planning and significant configuration on MBoxes, it is hard to create a new service chain. Besides, it is hard to make an adjustment to a service chain, such as adding or removing MBoxes from a service chain, because it is extremely error-prone to make the configuration changes on the network devices and MBoxes. Through separating the logical service chain from the physical network, OpenSCaaS offers the ability to implement a user-defined service chain, which makes it possible for service-providers to customize the at will.

The Ability to Save Flow Table Space: Most flow tables, which are used to install forwarding rules, have finite capacity. Thus, the volume of a flow table is definitely a scarce resource in SDN switches. In response, the standardized OpenFlow protocol introduces an eviction mechanism enabling the SDN devices to automatically eliminate rules of lower importance to make space for newer rules. This seems to achieve smoother degradation, but this may also cause a disruption of lower important service. Through using SC-ID, which enables micro flows of the same service chain to be aggregated into one macro flow, OpenSCaaS offers the ability to significantly save flow table space.

The Ability to Realize Auto-Provisioning: Traditional solutions to build the service chain takes a great deal of time and effort since each service requires a specialized hardware device. NFV moves network functions into software and typically executes as VMs under the control of a hypervisor. Thus, there is no need to over-provision since additional server-based capacity can be added when needed. Through employing a NFV-driven service chaining solution, ClickOS-based OpenSCaaS offers the ability to realize fast service-chaining auto-provisioning.

Conclusions

This article introduces today's common practice at enforcement of service-chaining policy. After illustrating typical service-chaining solutions, we motivate the new requirements in the context of SDN and NFV, and then we go through the

related work. The key contribution of this article is an OpenSCaaS solution that provides an open, novel, and extensible approach for service-providers. OpenSCaaS demonstrates a new design and architecture by using the benefits of SDN and NFV for enforcing service-chaining policy without modifying current SDN standards or mandating any implementation constraints on MBoxes. Finally, we evaluate the benefits that OpenSCaaS brought: flexibility, scalability, and adaptability.

We believe that there are two major areas for future work: measuring the performance of SCaaS, including the control plane and the data plane, and enhancing the functionalities of SCaaS, including monitoring the load of MBoxes and rescheduling the service chain path to complete service load-balancing.

Acknowledgments

The work is supported in part by the National Science Foundation of China under project 61272462.

References

[1] W. John *et al.*, "Research Directions in Network Service Chaining," *Proc. IEEE SDN4FNS*, 2013.
[2] P. Quinn *et al.*, "Network Service Chaining Problem Statement," http://tools.ietf.org/html/draft-ietf-sfc-problem-statement-10. August, 2014.
[3] A. Gember *et al.*, "OpenNF: Enabling Innovation in Network Function Control," *Proc. SIGCOMM, 2014.*
[4] A. Gember *et al.*, "Stratos: A Network-Aware Orchestration Layer for Virtual Middleboxes in Clouds," CoRR, abs/1305.0209, 2013.
[5] S. K. Fayazbakhsh *et al.*, "Enforcing Network-Wide Policies in the Presence of Dynamic Middlebox Actions using Flowtags," NSDI, 2014.
[6] M. Walfish *et al.*, "Middleboxes no Longer Considered Harmful," *Proc. OSDI '04* (Berkeley, CA, USA, 2004), USENIX Association, pp. 215–30.
[7] P. Quinn *et al.*, "Network Service Header," https://tools.ietf.org/html/draft-quinn-nsh-02. Feb., 2014.
[8] J. W. Anderson *et al.*, "xOMB: Extensible Open Middleboxes with Commodity Servers," *Proc. ANCS*, 2012.
[9] BalanceNG. http://www.inlab.de/balanceng/index.html.
[10] Snort. http://www.snort.org/.
[11] Squid. http://www.squid-cache.org/.
[12] G. Gibb, H. Zeng, and N. McKeown, "Outsourcing Network Functionality," *Proc. HotSDN*, 2012.
[13] M. Boucadair *et al.*, "Service Function Chaining: Design Considerations, Analysis & Recommendations," https://tools.ietf.org/html/draft-boucadair-sfc-design-analysis-02. Feb., 2014.
[14] J. Martins *et al.*, "ClickOS and the Art of Network Function Virtualization," NSDI, 2014.
[15] E. Kohler *et al.*, "The Click Modular Router," *ACM Trans. Computer Systems*, Aug. 2000.

Biographies

WANFU DING (wanfding@cityu.edu.hk) received his Ph.D. degree at the Chinese Academy of Sciences in 2011. From June 2011 to August 2014 he worked as a senior engineer in IP Technology Research Department of Huawei. He is currently a senior research associate in the Department of Computer Science, City University of Hong Kong. His main research interests are software-defined networks and network function virtualization.

WEN QI (qi.wen@my.cityu.edu.hk) received his B.E. degree from the Department of Automation, Nankai University in 2009, and the M.S. degree in computer science from the City University of Hong Kong in 2013. He is currently a Ph.D student in the Department of Computer Science, City University of Hong Kong.

JIANPING WANG (jianwang@cityu.edu.hk) is an associate professor in the Department of Computer Science at the City University of Hong Kong. She received the B.S. and the M.S. degrees in computer science from Nankai University, Tianjin, China in 1996 and 1999, respectively, and the Ph.D. degree in computer science from the University of Texas at Dallas in 2003. Jianping's research interests include dependable networking, optical networks, cloud computing, service oriented networking, and data center networks.

BIAO CHEN (bchen@umac.mo) received his B.S. in computer science from Fudan University in China, and an M.S. in mathematics and a Ph.D. in computer science from Texas A& M University, respectively. After graduation he joined the Department of Computer Science at the University of Texas at Dallas as an assistant professor. Currently he is a visiting professor in the Department of Computer and Information Science at the University of Macau. His research interests include distributed systems, networking, and security. He is a member of Sigma Xi, IEEE, and ACM.