Service Forwarding Label for Network Function Virtualization and Application-centric Traffic Steering

Changcheng Huang[1] and Jiafeng Zhu[2]
[1]Department of Systems and Computer Engineering, Carleyton University, Ottawa, Ontario, Canada
[2]R&D Centre, Huawei Technologies Inc., Santa Clara, U.S.A.

Keywords: Software Defined Network, Network Function Virtualization, Traffic Steering, Service Chaining, VXLAN.

Abstract: New services such as network virtualization, service chaining, and application-centric traffic steering bring new opportunities for network providers and service providers. Existing Internet architecture is suffering from the so-called ossification problem, which makes it difficult to support new applications. Software Defined Network (SDN) is becoming a new paradigm for both WAN and enterprise networks. By separating control and data planes, SDN allows network providers to sell new services without changing their physical switches. While SDN is poised to support new services, it is still in development stage. Mechanisms supporting new services are still missing. In this paper, we propose a new application driven mechanism called Service Forwarding Label (SFL) which can be used as a uniform label to differentiate various services and forward packets based on different service requirements. Compared with existing solutions such as VXLAN, our SFL approach introduces less overhead and supports more new services.

1 INTRODUCTION

Research on Software Defined Network (SDN) started nearly ten years ago. Most of earlier SDN networks were designed for datacenter infrastructure (ONF, 2015, Banikazemi et al., 2013) where flexibility and scalability are critical. With tens of thousands commodity switches and servers, datacenter presents a serious challenge to network architecture. SDN solves the problem by separating control plan from data plane. With relatively an independent control plane, SDN does not need to be run on proprietary equipment. Instead, the control plane can be realized with a large number of regular servers. Through adopting a centralized structure, these servers can be efficiently facilitated by datacenter with on-demand service capacity.

In recent years, SDN has found applications in other areas. Google, for example, built a large scale SDN-based WAN network (Jain et al., 2013) that has attracted attentions worldwide. Large carriers are looking at the possibility of upgrading their network infrastructures with SDN architecture. Issues such as scalability have been widely studied. Early benchmarks on NOX (Tavakkoli et al., 2009) (the first SDN controller) showed it could only handle 30,000 flow initiations per second while maintaining a sub-10 ms flow install time. Recent works have shown that SDN scalability can be extended by using multicore systems (Tootoonchian et al., 2012) or deploying multiple OpenFlow controllers (OFCs) (Tootoonchian and Ganjali, 2010, Yeganeh et al., 2012).

With a separated control plane, SDN has the potential to enable new services. One of the primary new services that have been envisioned is the network virtualization service, which allows physical network provider to sell different virtual networks to different service network providers. Each service network provider can use its virtual network just like the way it uses its own private network while sharing underlying physical network resources with other service network providers. The physical network provider, on the other hand, can enjoy new revenue growth through selling virtual networks with different granularities.

Service chaining is another area that SDN can play an important role. Traditionally a service chain consists of a set of dedicated network service boxes such as firewall, load balancers, and application delivery controllers that are concatenated together to support a specific application (Jacobs, 2015). With a new service, new devices must be installed and interconnected in certain order. This can be a very

Huang C. and Zhu J..
Service Forwarding Label for Network Function Virtualization and Application-centric Traffic Steering.
DOI: 10.5220/0005528800270034
In *Proceedings of the 6th International Conference on Data Communication Networking* (DCNET-2015), pages 27-34
ISBN: 978-989-758-112-0

complex, time-consuming, and error-prone process, requiring careful planning of topology changes and network outages and incurring high OPEX. This situation is exacerbated when a tenant requires different service sequences for different traffic flows or when multiple tenants share the same datacenter network. Through network function virtualization (NFV) service, SDN can dynamically create a virtual environment for a specific service chain and eliminate the complex hardware and labor work.

While possessing great potential, SDN is still in development stage. The actual mechanisms for supporting new services are still not defined. Meanwhile competitive technologies are emerging. A good example is VXLAN (Mahalingam, 2015), which is designed with two major objectives: a) to extend Virtual LAN (VLAN) service from a local area network to a network that can span an IP network; b) to address the shortage of VLAN ID. It advocates the architecture that overlays virtualized Layer 2 networks over Layer 3 networks. A 24-bit VXLAN network identifier is added to allow identifications of more than 16M different VLANs. While VXLAN can be implemented relatively easily with existing technologies, it has several major disadvantages that make it difficult to meet the long term challenges. First, VXLAN is an extension of Layer 2 VLAN. Its main objective is to extend VLAN service rather than to support arbitrary new services. Therefore, it inherits characteristics of Layer 2 technologies which make it far away from application-centric concept. It is hard to generalize the VLANs supported by VXLAN to other services such as service chaining. Second, it is hard to support resource allocation and optimization for virtual networks due to its distributed nature. Third, the overhead for overlaying Layer 2 networks over Layer 3 networks is overwhelming. Fourth, It is hard to imagine how to support recursive services (discussed more in the next section) with VXLAN.

OpenADN (Paul et al., 2013) is another option that has been proposed recently. With OpenADN, two new labels are added, one at Layer 3.5 and another at Layer 4.5. The label at Layer 3.5 is used to forward packets and the label at Layer 4.5 is used to interconnect two Layer 3.5 labels after passing a middle-box. Although OpenADN can solve some of the issues with VXLAN, it is still too complex to implement. Furthermore, issues such as recursive services still cannot be supported.

In this paper, we will propose a new Service Forwarding Label (SFL) mechanism at Layer 5 that can be used to identify a service instance for packet forwarding so that NFV and application-centric traffic steering can be realized with very little overhead. Our label creation process is application driven allowing close integration of applications and forwarding functions. Combining with the power of SDN controller, SFL can support resource allocation and optimization on a per service instance basis.

The paper is organized as follows. In Section 2, we will use NFV as a new service example to discuss more detail about the issues to be resolved. In Section 3, we will propose our SFL scheme. This will be followed by several use case scenarios in Section 4. We will finish the paper with some concluding remarks in Section 5.

2 CHARACTERISTICS OF NEW SERVICES

In this section, we will discuss in detail the characteristics and challenges of new services under SDN context. We will use NFV as a key example for the new services SDN will provide. In specific, we will investigate the issues related to recursive network virtualization, which illustrates the characteristics of recursive services in more general cases. The reason we choose NFV as an example is because it can represent a significant portion of new services currently being envisioned by network providers and service providers.

The initial SDN adopters, both vendors and network providers, have focused on some fundamental issues related to separating control and data planes. The benefits of creating new services have not been fully explored. A good example is the Google SDN WAN project mentioned earlier (Jain et al., 2013). Before the SDN project, the WAN Google had been using for interconnecting their datacenters had been managed using traditional approach which was slow due to manual provisioning process. On average, the utilization of Google's WAN at that time was 20 to 30%. Through multiple years of efforts, Google has upgraded its WAN with SDN technology. The initial results are very encouraging. Utilization has been increased to around 70 to 90%. In some cases, 100% utilization has been achieved. The key enablers of this improvement are the SDN dynamic flow creation capability and a more sophisticated optimization approach based on Max-min fairness. SDN allows Google to do centralized traffic engineering that can balance load distribution across their entire WAN with sophisticated optimization algorithms. Large amount of elastic traffic also helps increase the

utilization to 100%.

However, it should be noted that Google's WAN is a special case where Google is the user, service provider, as well as network provider, i.e., Google provides service to itself on its own network. This nature allows Google to do global optimization easily. For example, because Google can control the traffic carried by the WAN, operators can decide when and how long the elastic traffic will be buffered. Also because Google owns both service network and underlying WAN network, operators can have a global view of the network and therefore optimize network usage globally.

In a real world, it is quite common that users, service providers, and network providers are separate entities. They may all have their own objectives which may conflict with each other. Take the example of enterprise network. With the fast growth of datacenters, enterprises are becoming increasingly interested in outsourcing their enterprise networks to datacenters. Under this scenario, the owner of the physical network now is the owner of the datacenter, such as Amazon. The service network providers are the enterprises who outsource their enterprise networks to the datacenters. Therefore a service provider is independent from the network provider as well as independent from other service network providers who share the same physical network. Recognizing this need, the pioneers of SDN advocate a layer called FlowVisor (Sherwood et al., 2009) which plays the same role as the hypervisor for virtual machines. FlowVisor allows a physical network provider to partition its physical network into slices for various service network providers. Each service network provider can virtually own one slice which has its own virtual network topology and related resources generated through a topology abstraction process. The service network provider can then optimize its usage of the slice which it owns. A physical network provider can sell network virtualization service to service providers with different granularities selling at different prices (Drutskoy et al., 2013). This will change the situation that physical network providers today can only sell pipes and equip physical network providers a new venue for revenue growth.

The SDN architecture for virtual network service is shown in Fig. 1. As shown in the figure, each entity has its own OpenFlow Controller (OFC). The OFC of a service provider is in charge of routing user flows and optimizing resource usage within its own virtual topology. The OFC of the network provider will execute the topology abstraction process through a topology virtualizer based on a global topology map of the underlying physical network.

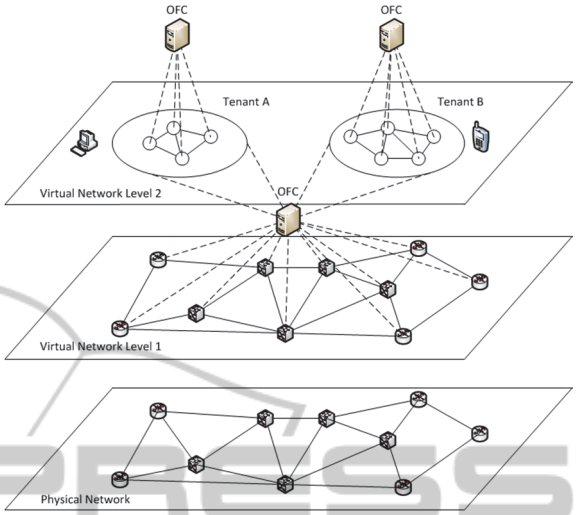

Figure 1: The architecture of virtual network service.

It is envisioned in SDN architecture that a service provider can further partition its virtual network and sell virtual network service to other service providers. This kind of recursive network virtualization is illustrated in Fig. 2. As shown in the figure, a network provider sells a virtual network to a tier-1 service provider which in turn sells virtual network to a tier-2 service provider. This process can continue several iterations making service providers become network providers in a recursive manner. This kind of recursive services is becoming popular today. For example, Amazon sells virtual server service to Netflix and Netflix in turn sells video streaming service to its clients.

While NFV is a powerful concept, it also faces several challenges that need to be addressed. These include overlapping address space, middle-box traversal, SDN network migration, multiple entities, etc. We will briefly discuss these problems in the following paragraphs.

As we mentioned earlier, each service provider treats its virtual network as its private property. They decide how to assign VLAN IDs and IP addresses within their own virtual networks without consulting each other. Most likely they will use private IP address blocks to avoid the shortage of address space. This creates a high possibility of overlapped address spaces being used by multiple virtual networks sharing the same physical network. Using Fig.1 as an example, if the two virtual networks owned by Tenant A and Tenant B respectively use the same address space, switches in the physical

network will not be able to differentiate packets for the two virtual networks. Clearly some mechanism is required to identify packets belonging to different virtual networks.

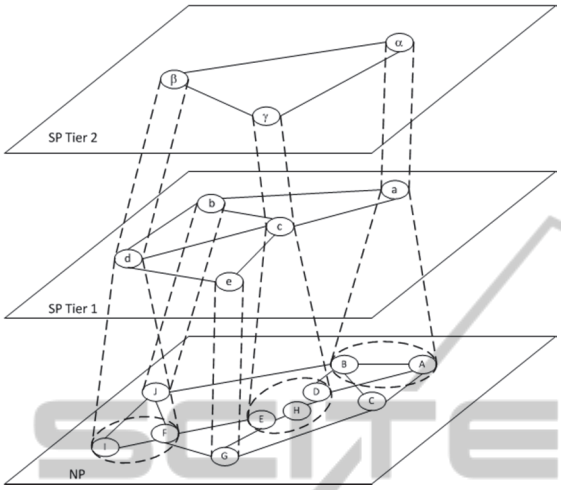

Figure 2: Illustration of recursive network virtualization.

In a service chain application, packets need to traverse multiple middle-boxes before reaching their destinations. Each middle-box such as firewall or load balancer provides functions that may require processing packets up to the transport layer. Therefore the header information including all layers up to the transport layer will not be maintained end-to-end. This will make it difficult for physical switches to steer traffic to the correct next middle-box on the path to the destination. This issue suggests that an end-to-end ID above the transport layer is required to identify traffic belonging to the same service chain.

SDN as a new architecture is attracting many vendors and network providers. Given the size of the Internet today, the evolution towards SDN networks will likely be a long and slow process. It is easy to see that SDN networks will be deployed initially as islands within Internet. Datacenters, for example, are the first place where SDN will be adopted early. When datacenters provide virtual network services to clients, most of the clients are unlikely to be attached directly to the datacenters. Therefore, traffic from these clients has to travel through legacy IP networks before they can reach SDN based datacenters. Any techniques used to identify virtual networks in the lower three layers may be lost when traffic reaches the datacenters. Inter-datacenter traffic may also span multiple legacy IP networks making it difficult to support virtual networks across multiple datacenters.

Recursive network virtualization raises another new challenge. As we mentioned earlier, each virtual network may be owned by a different entity. This leads to the situation that multiple entities may be involved in recursive network virtualization. Each entity has the right to control the resources within its own virtual network and decide how to route traffic based on its own policy. Orchestration among multiple OFCs is calling for simple mechanisms that help streamline business relationship among different entities.

In summary, the above mentioned challenges must be addressed before the benefits of SDN can be fully realized. In the next section, we will describe in detail how our SFL works.

3 SERVICE FORWARDING LABEL

As discussed in the last section, we need an ID that can be used to identify a service instance (In our context, a service chain instance is considered as a single service instance.). This ID needs to sit above Layer 4 so that it can stay intact while a packet traverses legacy IP networks and middle-boxes. This naturally points to an ID at Layer 5.

In OSI model, Layer 5 is called the session layer which is designed to establish, manage, and terminate connections between local and remote applications. A good example is a video conference session where multiple parties join and leave dynamically. This bears similarity to a service instance such as a virtual network which carries a large number of dynamic traffic flows. This similarity motivates us to define an ID at Layer 5 for the identification of a service instance (or service chain instance). We call this ID Service Forwarding Label (SFL).

SFLs will be created and maintained by a service provider and used by its clients and OpenFlow switches to identify and steer traffic belonging to different service instances. SFLs can be stacked for applications such as recursive services where each level of the stack is administered by the owner of the service level in a recursive business relationship as discussed in the last section. This allows easy scale to multiple levels of services with multiple ownerships nested in the SFL stack.

A SFL must be unique within the space of the service provider who administers the SFL. Multiple service providers at the same level will be differentiated by their SFLs at the lower level. The

combination of the SFLs across different levels in a label stack uniquely identifies a service at any layer in a physical substrate domain.

In the following paragraphs, we will discuss SFL format and the process to establish and terminate a SFL.

As shown in Fig.3, each SFL is represented by 4 octets. Starting from bit 0 of the 4 octets, the first 30 bits hold the label, bit 30 is reserved for experimental use, bit 31 is the top-of-stack bit (S). The S bit is set to one for the last entry in a label stack, and zero for all other label entries in the stack. As the header at Layer 5, SFL can run either over UDP or TCP making it applicable to all kinds of traffic belonging to the same service instance.

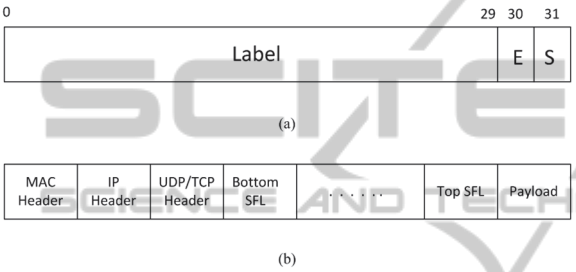

Figure 3: (a) Format of Service Forwarding Label; (b) Illustration of SFL stack in a packet header.

Each SFL is associated with a lifetime. When the lifetime expires, the SFL will be terminated or be renewed. This dynamic mechanism allows a service provider to maintain a smaller pool of SFLs.

There are various scenarios that may happen during the lifetime of a SFL. The procedures for establishing and terminating SFL depend on the actual scenario encountered. A typical scenario is shown in Fig. 4. We describe the procedures step by step in the following part.

- A client sends a service request to the OFC of a service provider with its user ID and requested service type using HTTP request message. Metadata can be sent through HTTP Post message;
- The OFC of the service provider decides whether it can accept the request by applying optimization process which determines how to route traffic and allocate resources for the requested service;
- If the request is admissible, the OFC will create a new SFL which is unique to the service provider and send the SFL and associated lifetime to relevant OpenFlow switches or middle-boxes that need to steer or process traffic based on the SFL through an OpenFlow OFPT_FLOW_MOD message;
- Upon receiving the message from the OFC, the OpenFlow switches or middle-boxes will set the SFL and its lifetime into their flow tables as part of a rule set;
- The OFC will send HTTP response message with the SFL and associated lifetime to the client confirming the acceptance of the request;
- The client will add the label as Layer 5 header to its packets destined for the requested service and send them out;
- When the packets reach the switches or middle-boxes within the service provider network, the service provider will match the Layer 5 header (and other headers in other layers if necessary) to its rule set and decide how to forward or process the packets based on their service requirement;
- The switches or middle-boxes will then process those packets and steer them to the next switch or middle-box if necessary;
- When the lifetime of the SFL expires, the client can choose either to renew the service or leave. If it decides to renew, it will send a
- HTTP request message with the SFL to the OFC, the above procedures will be repeated except that the original SFL will be used instead of generating a new SFL.

In the next section, we will discuss some use cases that will help illustrate how the SFL can be used in real applications.

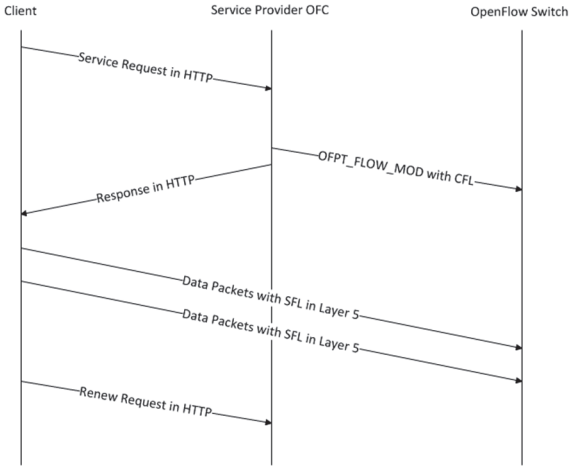

Figure 4: Flowchart of labelling process.

4 SAMPLE USE CASES

There are numerous use cases that the proposed SFL

can be applied to. We will discuss some common use cases briefly in this section.

The first use case is the virtual network service we mentioned earlier. Here a physical network provider will serve as the service provider and service network providers will serve as clients. Service network providers request virtual networks from the physical network provider. Each service network provider will have full control over its virtual network. One issue we mentioned earlier is that the address space used by service providers can be overlapped. An example is shown in Fig. 5 where Client Network 1 owns Virtual Network 1 and Client Network 2 owns Virtual Network 2. Both Virtual Network 1 and Virtual Network 2 share a physical network owned by a SDN network provider. When a packet reaches a switch in the SDN network, the switch needs to decide which virtual network the packet belongs to.

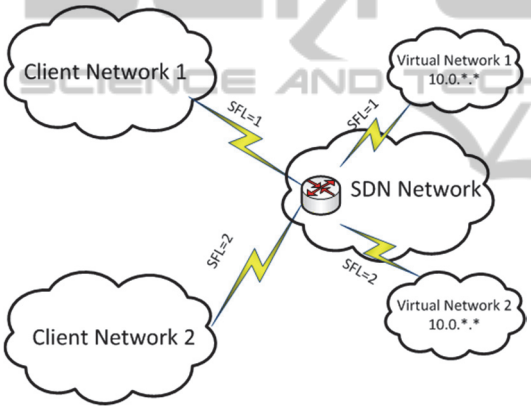

Figure 5: Virtual networks with overlapped address space.

Through the procedures discussed in the last section, each client network will receive an SFL assigned by the SDN network as an identifier of its virtual network. The client network will inform its users of adding the SFL for all packets that need to use the virtual network it owns. When packets reach the switches in the SDN network, they can be differentiated using SFL even though their IP address spaces may be overlapped. This can be done by a simple match in the flow table. Without SFL, multiple header fields may need to be matched in order to identify packets belong to a virtual network, which will likely cause flow table fragmented and bloated.

When recursive network virtualization is deployed, each service network will serve as client as well as service provider at the same time. As a client, it receives a SFL from the service provider one level below it. As the service provider, it administers the SFLs that identify the virtual networks it sells. A physical switch can use multiple levels of the label stack to steer packets for the correct virtual networks they belong to.

Now we look at the second use case that demonstrates how service chain can be supported. It is easy to see that a service chain instance can be realized using NFV where each virtual node represents a specific service such as firewall that can be dynamically mapped to a physical node in the lower level. By the virtualization of a service chain, dynamic sharing of physical resources can be achieved. Traffic flows for different service chain instances can be uniquely identified and steered by the combinations of the multiple SFLs in their label stacks. This enables great flexibility and leads to significant cost reduction in OPEX. An example is illustrated in Fig.6.

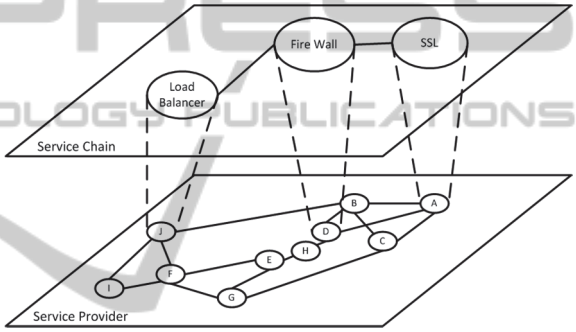

Figure 6: Service chain as network function virtualization.

Application service providers such as Google are increasingly interested in providing different treatments to different types of customers, e.g. subscribers vs. casual users. Based on the SFLs they are carrying, user traffic flows can be steered to different environments with different networking and computing resources provisioned. Under this context, SFL provides a simple and effective handle that connects applications to physical layer devices directly and enables application-centric traffic steering. There are many existing Quality of Service (QoS) schemes such as VLAN and DiffServ. But they are Layer 2 or 3 mechanisms which are hard to scale to end-to-end applications. As mentioned earlier, it is difficult to maintain any code points in headers up to Layer 4 for end-to-end services due to middle boxes and different domains a packet may traverse. By sitting at Layer 5, our SFL can travel through networks and middle boxes easily and therefore provide a very strong support for various end-to-end applications.

There are many application scenarios that can demonstrate the usage of SFL. For example, a

service provider may want some of its user traffic be protected from server or link failures while other traffic not. When a server or link failure happens, the traffic that needs protection is steered to a protection path. In OpenFlow switches, packets that require protection will be matched at a group table instead of the regular flow table. Therefore incoming packets must be de-multiplexed into regular flow table or group table based on whether they need protection or not. The proposed SFL provides an excellent option to achieve this function. Specifically, we can assign one SFL to identify traffic requiring protection and another SFL for traffic not requiring protection. As shown in Fig.7, when packets arrive at a switch, it first goes to a regular flow table. If the SFL matching indicates a packet without protection requirement, other header fields will be matched as regular case; otherwise, the packets will be forwarded to a group table for protection matching.

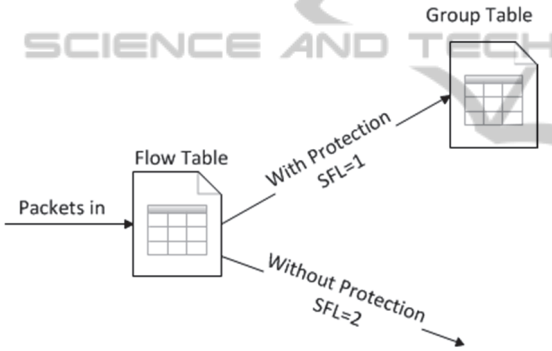

Figure 8: Forwarding packets with and without protection.

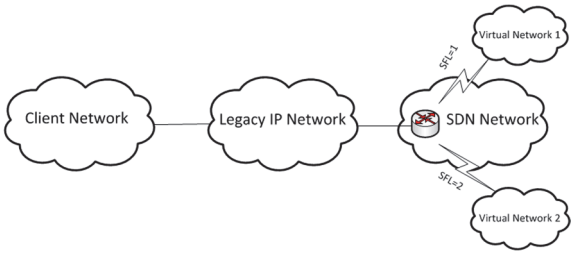

Figure 9: Virtual network service with SDN network as an island.

For the case that SDN networks form some islands in an end-to-end network, the proposed SFL also provides an elegant migration solution. An example is shown in Fig. 8 where the traffic from a client network needs to traverse a legacy IP network to utilize the virtual network service provided by a SDN network. Because SFL sits at Layer 5, it will not be affected by the legacy network. Packets carrying SFL will travel through the legacy network just like a regular packet. This allows the SDN network island to provide new services even when it is surrounded by legacy networks.

5 CONCLUSIONS

By separating control plane from data plane and centralizing resource allocation, SDN has the potential to allow network and service providers to create a variety of new services. Existing SDN products have been focused on some basic functions such as flow setup and teardown. The potential to create new services has not been fully explored.

NFV is one of the most promising services SDN can provide. Through NFV process, a network provider can sell virtual slices of its physical network to different service providers. Different service providers as tenants have full control of the virtual topologies within their own slices while the network provider has the control of its physical network. However lack of an effective virtual network identifier makes the potential of SDN difficult to realize.

Application-centric traffic steering is another type of services SDN can provide. Application service providers have been looking for ways to differentiate user traffic flows so that resources can be used more effectively on generating revenues for them. An end-to-end identifier that can be utilized to differentiate user traffic flows is required so that user traffic flows can be steered to the right server and network resources for maximizing the benefits of the service providers.

In this paper, we proposed SFL as an identifier for service instance. It is controlled by service providers and used by clients and OpenFlow switches to steer traffic to different services. The format of SFL is simple enough to minimize overhead. Through SFL stacking, recursive services such as recursive network virtualization can be supported easily while allowing different entities to exercise their controls over their own resources. With SFL as a Layer 5 mechanism, it can traverse middle-boxes and legacy networks without any changes so that the relationship between clients and service providers can be maintained end-to-end.

We have demonstrated various use cases ranging from NFV, service chaining, to application-centric traffic steering. Through these use cases, we can see that the proposed mechanism is simple to implement with existing protocols and technologies and can effectively enable various new services.

REFERENCES

ONF, 2015. https://www.opennetworking.org/

Banikazemi, et al., 2013. Meridian: An SDN Platform for Cloud Network Services, IEEE Communications Magazine, Feb. 2013.

Jain, S., et al., 2013. B4: Experience with a Globally-Deployed Software Defined WAN, ACM SIGCOMM 2013, August 12-16, 2013, Hong Kong.

Tavakkoli, A., et al., 2009. Applying NOX to the Datacenter, ACM Workshop on Hot Topics in Networks (HotNets-VIII), October 22-23, 2009, New York.

Tootoonchian, A., et al., 2012. On Controller Performance in Software-Defined Networks, 2nd USENIX Hot-ICE'12, April 24, 2012, San Jose.

Tootoonchian, A. and Ganjali, Y., 2010. Hyperflow: A Distributed Control Plane for OpenFlow, USENIX NSDI INM/WREN, April 2010, San Jose.

Yeganeh, S. Hassas and Ganjali, Y., 2012. Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications, ACM SIGCOMM HotSDN'12, August 13, 2012, Helsinki.

Jacobs, D., 2015. How SDN and NFV simplify network service chain provisioning, http://searchsdn.techtarget.com/tip/How-SDN-and-NFV-simplify-network-service-chain-provisioning.

Mahalingam, M., et al., 2015. "VXLAN: A Framework for Overlaying Virtulized Layer 2 Networks over Layer 3 Networks," IETF draft, http://datatracker.ietf.org/doc/draft-mahalingam-dutt-dcops-vxlan/.

Paul, S., et al., 2013. OpenADN: A Case for Open Application Deliver Network," Proceedings of ICCCN 2013, July 2013.

Sherwood, R., et al., 2009. FlowVisor: A Network Virtualization Layer, OPENFLOW-TR-2009-1, OpenFlow Consortium, October 2009.

Drutskoy, D., et al., 2013. Scalable Network Virtualization in Software Defined Networks, IEEE Internet Computing, Vol. 17, Iss. 2, March, 2013.