Flexible Traffic Engineering: When OpenFlow Meets Multi-Protocol IP-Forwarding

Suoheng Li, Yan Shao, Shoujiang Ma, Nana Xue, Shengru Li, Daoyun Hu, and Zuqing Zhu, *Senior Member, IEEE*

***Abstract*—In this letter, we present flexible traffic engineering (F-TE), a solution to achieve highly efficient traffic engineering in a hybrid network where IPv4 and IPv6 protocols co-exist. By leveraging the programmability of OpenFlow (OF), we design the OF system to facilitate IP forwarding interchanging (i.e., switching the packets between IPv4 and IPv6 at each hop according to the network status). The OF system is implemented in a real network testbed. We conduct experiments with video streaming to verify that the proposed F-TE can improve network throughput effectively with adaptive IP interchanging.**

***Index Terms*—OpenFlow, IPv4/IPv6 transition, traffic engineering.**

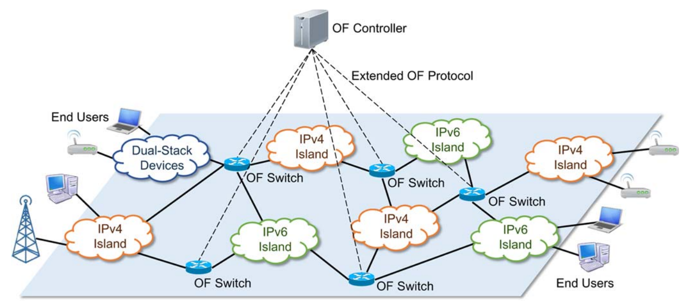

Fig. 1. Overall network architecture.

I. Introduction

THE booming of the Internet makes the volume of IP network devices grow really fast, and their number would be three times of global population in 2017 [1]. As a consequence, we have already exhausted all the new IPv4 addresses. To resolve this addressing issue, people has deployed IPv6, which is not backward-compatible to IPv4. Hence, IPv4- and IPv6-devices will coexist and create numbers of IPv4- and IPv6-islands [2]. The emerging of these islands fragments network topology and brings new difficulties to traffic engineering (TE). For instance, IPv4 traffic would only be carried through IPv4-islands, even if the adjacent IPv6-islands could provide alternative paths that are less congested. To solve this island problem, one needs the flexible traffic engineering (F-TE) that can support agile inter-operation of IPv4- and IPv6-devices. Note that an IP transition technique that only makes IPv4- and IPv6-devices inter-operable [3] is not enough for realizing F-TE, since it only pre-configures IP tunnels statically and cannot establish/disassemble them on-the-fly according to the TE's requirement. Specifically, F-TE would require the network devices to facilitate online and adaptive IP-forwarding interchanging (i.e., switching the packets between IPv4 and IPv6 at each hop according to the network status).

Previously, OpenFlow (OF) has been developed to make the networks programmable and flexible, using flow-based switching and centralized management [4]. With its centralized control plane and flexible data plane, OF can support flow-level TE in a network-wide manner [5]. In this letter, we propose to realize online and adaptive IP-forwarding interchanging with OF and leverage the technique to achieve F-TE in a network that consists of multiple IPv4- and IPv6-islands. Specifically, we have the islands interconnected by OF switches that are controlled by a centralized controller, and design the OF system to enable flow-level IP-forwarding interchanging for F-TE. We implement the system in a real network testbed, and use real-time video streaming to experimentally demonstrate the effectiveness of F-TE.

The rest of the paper is arranged as follows. Section II presents the overall system architecture and design considerations. We discuss the implementation details in Section III. The experimental results are shown in Section IV. Finally, Section V summarizes the paper.

II. Operation Principle

A. System Architecture

Fig. 1 shows the overall network architecture for F-TE with OF. The IPv4- or IPv6-islands consist of network devices that are only IPv4- or IPv6-capable. Unlike the traditional IP transition approaches that set up gateways among these islands, we deploy OF switches to connect them. The OF switches are controlled by a centralized controller, which monitors the network status and makes routing paths that across the islands available by using IP-forwarding interchanging. In this way, F-TE is realized to improve the overall routing efficiency. Note that dual-stack devices may also exist in the network, and according to their working principle, they can forward both IPv4 and IPv6 traffic but cannot make adjacent IPv4 and IPv6 islands communicate.

Apparently, the advantage of F-TE is due to the fact that IP-forwarding interchanging creates more feasible forwarding paths in the network and makes the network more connected

Manuscript received June 11, 2014; accepted August 11, 2014. Date of publication August 21, 2014; date of current version October 8, 2014. This work was supported in part by the Project NCET-11-0884, the NSFC Project 61371117, the Fund WK2100060010, and the SPR Program of CAS (XDA06010302). The associate editor coordinating the review of this paper and approving it for publication was I.-R. Chen.

The authors are with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: zqzhu@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LCOMM.2014.2349991

regardless of the existence of IPv4- and IPv6-islands. Or in other words, we can easily prove the following proposition.

Proposition 1: After upgrading any non-OF node to an OF one, we will have more or at least the same available paths, for any source-destination pair in the network.

B. Design Considerations

To implement the OF system for F-TE, we still need to face a few challenges and in this subsection, we discuss the design considerations to address them.

1) Address Translation: To realize F-TE, OF-enabled nodes need to perform IP forwarding interchanging and let the related packets enter a different IP-island, which requires IP address translation. Thanks to the centralized management of OF, we only need to reserve a small number of IPv4/IPv6 addresses on each OF switch, translate a flow's addresses to temporary ones when necessary, and recycle them when the flow expires.

2) Routing Protocol: To make the IP-islands know the IP interchanging capability of their adjacent OF switches, we need to disseminate the related routing information. Therefore, on each OF switch, we run the same routing protocol as that used by the IP-islands. For instance, if the open shortest path first (OSPF) protocol is used in the network, each OF switch will behave as an ordinary IP router to receive OSPF messages, learn network topology, and advertise routing paths. The only difference is that all the OF switches share only one OSPF instance, and this "singleton" characteristic enables us to run a centralized path calculation algorithm. Another important thing to consider when using non-shortest paths for TE is the routing loop problem. This problem is not an issue for our OF system, since the OF controller knows the whole network topology through routing protocol and runs centralized path computation to eliminate routing loops.

3) Link State Collection: To perform the centralized path computation, the OF controller needs to collect real-time link state information, and it uses the simple network management protocol (SNMP) to do so proactively.

4) Data Plane Support: To the best of our knowledge, there are two possible ways to realize IPv4/IPv6 interchanging in OF. One is to define new flow-matching actions for supporting IP-in-IP encapsulation, and the other is to use the out-of-band management protocol for setting up IP tunnels, e.g., the OF-CONFIG protocol [6]. In this work, we choose to implement the system with the former because it provides us more freedom on controlling the behavior of each flow and can be implemented more easily. Hence, we define new flow-matching actions to encapsulate IPv4 packets in IPv6 payloads for being delivered through an IPv6-island, or *vice versa*. Note that we use the path MTU discovery in [7] to handle the situation where the size of encapsulated packets exceeds the Maximum Transmission Unit (MTU) of the subsequent hops.

III. System Implementation

In this section, we discuss the details of system implementation for realizing F-TE with IP interchanging. Fig. 2 shows the OF controller architecture and operation procedure.

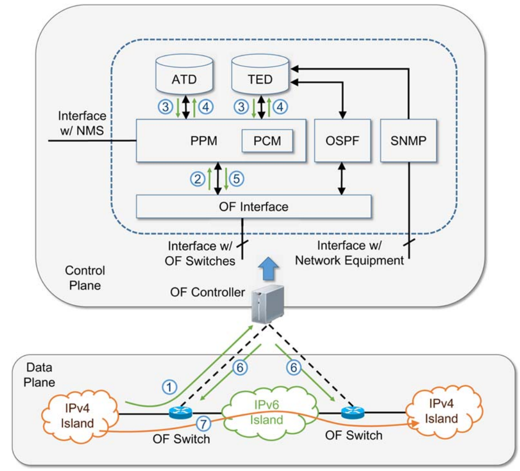

Fig. 2. OF controller architecture and operation procedure.

A. OpenFlow Controller

We implement the OF controller based on the POX [8] platform. As shown in Fig. 2, it has 7 functional modules. The path provision module (PPM) interacts with all the OF switches through the OF interface, instructs the path computation module (PCM) to calculate routing paths, and manages the two databases (i.e., address translation database (ATD) and traffic engineering database (TED)) in the OF controller. PCM calculates the routing paths for F-TE by checking the information in ATD and TED, and it can be instructed to do so by either PPM or an external network management system (NMS). The OF interface runs an extended OF protocol to communicate with the OF switches, and demultiplexes the OF messages to corresponding functional modules. ATD and TED are realized with MySQL in the OF controller, and they store the special IP addresses for IP interchanging and network status, respectively. For TED, the network topology is updated through the OSPF protocol and the link state information is obtained with SNMP. The OSPF module acts as a centralized OSPF agent to communicate with the IP routers in the islands. The SNMP module collects link state information proactively from the IP routers and keeps TED updated. Currently, PCM is implemented in the OF controller for simplicity. But note that when the path computation becomes more resource-consuming, we can move PCM out of the OF controller, and replace it with an external path computation element (PCE) [9] for modular design and scalable implementation.

B. OpenFlow Switches

The OF switches are implemented with Open vSwitch [10]. We design five new OF flow-matching actions, namely *PUSH_IPv6*, *POP_IPv6*, *MOD_IPv6*, *PUSH_IPv4*, and *POP_IPv4*, to accomplish IP forwarding interchanging.

C. Operation Procedure

Fig. 2 shows how a new flow is processed to realize F-TE.

Step 1: The OF switch nearest to the source node detects a new flow and sends a *Packet-In* message to the OF controller.

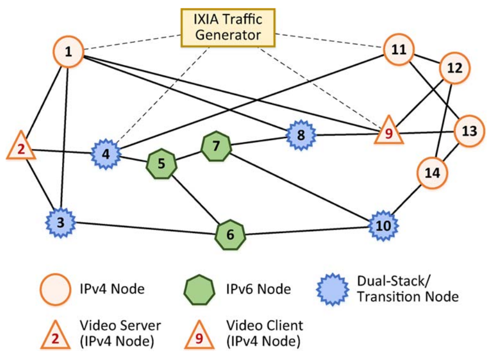

Fig. 3. Network topology of the experimental setup.

Step 2: In the OF controller, the OF Interface examines the *Packet-In* message and forwards it to PPM.

Step 3: PPM instructs PCM to calculate a routing path with the network status in TED, and checks ATD to allocate temporal IP addresses if IP interchanging is necessary at certain node(s).

Step 4: After obtaining the path from PCM, PPM updates TED and ATD accordingly.

Step 5: PPM encodes *Flow-Mod* messages for each OF switch that is on the path. The messages contain forwarding actions and address mapping information for necessary IP interchanging, with which the OF switches can encapsulate and forward the packets correctly.

Step 6: The OF interface distributes the *Flow-Mod* messages.

Step 7: The OF switches parse the *Flow-Mod* messages and set up the forwarding path for the flow accordingly.

In the example in Fig. 2, the OF controller instructs the ingress OF switch to encapsulate the IPv4 flow's packets with IPv6 header, and to deliver them through the IPv6-island. After going through the IPv6-island, the egress OF switch removes the IPv6 headers and forwards the packets for subsequent processing. In addition to establishing the path for new flows, the OF controller can also trigger a path switching when it observes network congestion. Under such circumstance, PPM performs F-TE with *Steps* 3–7. Note that in *Steps* 5 and 6, PPM needs to encode *Flow-Mod* messages that let the OF switches replace old flow-entries with new ones.

IV. Experimental Demonstrations

A. Testbed Setup

We implement the OF system on Linux servers (Lenovo ThinkServer RD530 and RD540), and set up the experiments with the topology in Fig. 3. The IXIA traffic generator is used to inject background traffic and create network congestion. In the experiments, we compare four scenarios, as 1) there is no TE and the video is sent with the shortest feasible path (*Scenario* 1, Without TE), 2) there is conventional TE with dual-stack routers, i.e., path switching for TE is possible but IP interchanging is not allowed (*Scenario* 2, Conventional TE), 3) there is TE with IP interchanging, but using pre-configured static IP tunnels (*Scenario* 3, TE with ST), and 4) there is F-TE with IP interchanging using the OF system (*Scenario* 4, F-TE). Note that for *Scenarios* 2 and 3, we set *Nodes* 3, 4, 8 and 10 in Fig. 3 as dual-stack routers, while in *Scenario* 4 these nodes are configured as OF switches.

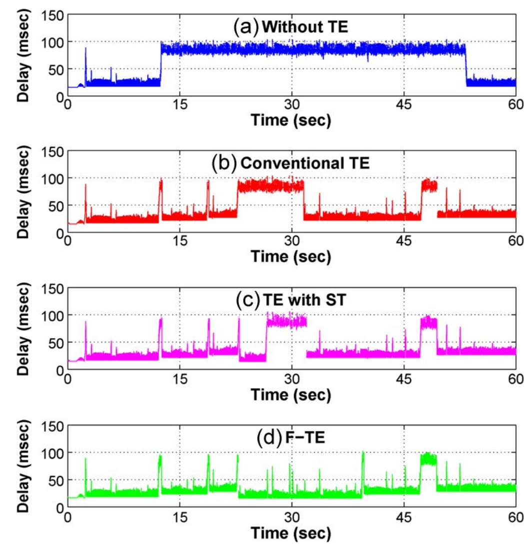

Fig. 4. Results on packet-level streaming delay.

The real-time video streaming uses a 60-second H.264 video sequence that is encoded as 1080P. The server on *Node* 2 encapsulates the video with IP/UDP/RTP headers and sends it across the network, while the client on *Node* 9 receives and decodes the video for performance evaluation. The experiments also generate background traffic flows and inject them into the IPv4- and IPv6-islands. The parameters of each background flow, i.e., the starting time, duration, source-destination pair, and average data-rate are chosen randomly.

B. Experimental Results

Fig. 4 shows the results on packet-level delay that we obtain for the four scenarios. Apparently, F-TE achieves the shortest streaming delay as it can use agile IP interchanging to avoid congested links. For *Scenario* 1 (Without TE), the delay results are the longest because when the shortest path $(2 \rightarrow 1 \rightarrow 9)$ is congested, video packets have no alternative paths and suffer from severe queuing delay increase. *Scenario* 2 (Conventional TE) can switch the video packets to $2 \rightarrow 1 \rightarrow 8 \rightarrow 9$ or $2 \rightarrow 4 \rightarrow 11 \rightarrow 13 \rightarrow 9$ when the primary path is congested. However, this is still not good enough as those three paths can be all congested when the network is crowded. e.g., *Links*$4 \rightarrow 11$, $1 \rightarrow 8$, and $1 \rightarrow 9$ are all congested in [23, 32] seconds. The limitation of *Scenario* 3 (TE with ST) is from the coverage of static tunnels. Obviously, we cannot pre-configure all the possible IP tunnels due to the high complexity. Hence, when the IPv4 paths and the pre-configured tunnels are all congested, *Scenario* 3 performs worse than F-TE. The streaming bandwidth obtained at the client is plotted in Fig. 5, which shows the similar trend and verifies our analysis above.

Fig. 6 illustrates the decodability of the received video frames, where a dark red bar indicates that a frame is not decodable due to severe packet loss. For the received videos, we also obtain their luminance component's peak signal-to-noise ratios (Y-PSNRs) and plot the results in Fig. 7. It is interesting to notice that there are four dips on the Y-PSNR curves for

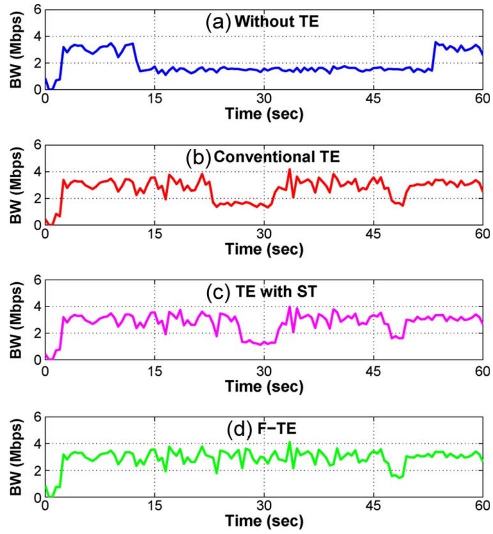

Fig. 5. Results on streaming bandwidth at the client side.

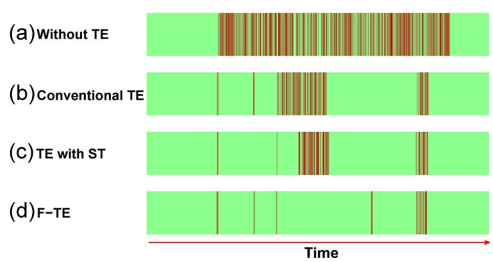

Fig. 6. Results on video frames' decodability.

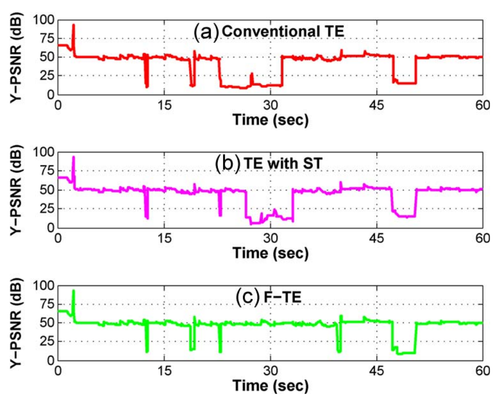

Fig. 7. Results on received videos' Y-PSNRs.

Fig. 8. Screen-shots of received videos.

F-TE within [15, 45] seconds. We believe they are caused by the path switchings in this period. We also capture two screen-shots of the received videos (as in Fig. 8) to compare the actual streaming qualities illustratively. The statistics of the experimental results are listed in Table I, in which we can clearly see that F-TE performs the best among the four scenarios, in terms of all the metrics, including the delay, bandwidth, link utilization, frame loss rate and Y-PSNR.

TABLE I
STATISTICS OF EXPERIMENTAL RESULTS

		Without TE	Conventional TE	TE with ST	F-TE
Delay (msec)	Average	58.0	36.4	33.2	30.8
	Standard Deviation	31.6	19.2	16.9	13.0
Bandwidth (Mbps)	Average	1.93	2.69	2.76	2.90
	Standard Deviation	0.81	0.82	0.82	0.70
Link Utilization (%)	Average	38.6	53.9	55.2	57.9
Frame Loss (%)	Average	28.1	7.7	6.0	2.1
Y-PSNR (dB)	Average	–	42.33	43.87	47.22

V. CONCLUSION

This letter proposed to achieve F-TE in a network that consists of multiple IPv4- and IPv6-islands with online and adaptive IP-forwarding interchanging enabled by OF. We designed the system architecture, implemented the OF system in a real network testbed, and conducted experiments for performance evaluation. The video streaming experiments verified that the proposed OF system could improve network throughput effectively with adaptive IP interchanging. Note that in addition to F-TE, the SDN-enabled IP interchanging architecture presented in this work could also facilitate seamless transition from IPv4 to IPv6. Specifically, if we first deploy a small number of OF switches in an IP network that consists of multiple IPv4- and IPv6-islands, and use them to forward IPv4 traffic into IPv6-islands purposely, the IPv4-islands would become relatively empty. Then, we could replace more IPv4 routers with the OF switches. By repeating these steps, we could make the whole network IPv6-capable, and accomplish the IPv4-to-IPv6 transition seamlessly and smoothly.

REFERENCES

[1] Cisco Visual Network Index. [Online]. Available: http://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html

[2] RFC 6219 X. Li, C. Bao, M. Chen, H. Zhang, and J. Wu, The China Education and Research Network (CERNET) IVI Translation Design and Deployment for the IPv4/IPv6 Coexistence and Transition, May 2011 RFC 6219.

[3] P. Wu, Y. Cui, J. Wu, J. Liu, and C. Metz, "Transition from IPv4 to IPv6: A state-of-the-art survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1407–1424, 2013.

[4] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Feb. 2008.

[5] S. Agarwal, M. Kodialam, and T. Lakshman, "Traffic engineering in software defined networks," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 2211–2219.

[6] Open Networking Foundation, OpenFlow Management and Configuration Protocol 1.2 (OF-Config 1.2). [Online]. Available: https://www.opennetworking.org/sdn-resources/onf-specifications/openflow-config

[7] RFC 4821 M. Mathis and J. Heffner, Packetization Layer Path MTU Discovery, Mar. 2007 RFC 4821.

[8] POX Wiki. [Online]. Available: https://openflow.stanford.edu/display/ONL/POX+Wiki

[9] RFC 4655 A. Farrel, J. Vasseur, and J. Ash, "A Path Computation Element (PCE)-Based Architecture," , Aug. 2006 RFC 4655.

[10] Open vSwitch. [Online]. Available: http://openvswitch.org/