On Exploiting SDN to Facilitate IPv4/IPv6 Coexistence and Transition

Jia-Jhun Lin, Kai-Ching Wang, Shin-Ming Cheng and Yen-Chun Liu
Department of Electronic and Computer Engineering
National Taiwan University of Sciecne and Technology, Taipei, Taiwan
Email: {b10215004, m10415007, smcheng, b10232038}@mail.ntust.edu.tw

***Abstract*—Today, the booming of the IoT (Internet of things) makes the volume of IP network devices grow rapidly, and IPv4 address pool was exhausted. While IPv6 is considered to be a feasible solution due to its sufficient address space, the slow deployment of IPv6 makes the necessity of the transition from IPv4 to IPv6. This paper proposes a framework for IPv4/IPv6 co-existence by leveraging Software-Defined Network (SDN) to facilitate the IPv6-to-IPv4 address conversion. In particular, a virtual machine with 6to4NAT functionality is generated dynamically when IPv6 traffic appears in IPv4 island. With the aid of SDN and virtualization on NAT64 nodes, the load balancing of IPv6-to-IPv4 traffic is facilitated. We implement this platform by using OpenStack in a real network testbed, and show that the proposed solution is a more flexible deployment architecture.**

***Index Terms*—IPv6 Transition, NAT64, Openstack, SDN**

I. Introduction

As the explosive growing of Internet, the number of networked devices owned by each person is estimated to 3.4 in 2020 [1] and the requirement of IP address becomes much larger when the demands for IoT applications are included. IPv6 is considered as a feasible solution to afford the increasing number of requirements on IP addresses. However, due to the slow deployment of IPv6 network and noncompatible structure of IPv6 and IPv4, how to make a smooth transition from IPv4 to IPv6 is a challenging issue [2].

One possible solution for IPv6 migration is to upgrade the IPv4-only devices as dual stack and make them compatible with the IPv6-only device. Instead of upgrading on device side which incurs a significant cost, this paper tries to resolve the problem of IPv4/IPv6 co-existence from the perspective of the network side. In particular, we apply Software-Defined Network (SDN) and virtualized NAT64 node [3] to dynamically handle the IPv6 traffic for the compatibility of IPv4 networks. The programmable OpenFlow is leveraged to facilitate IP forwarding interchanging [4]. Depending on the amount of IPv6 traffic, different numbers of NAT64 nodes are created in a virtualized way for the IPv6-to-IPv4 transition. Obviously, an SDN-based approach can offer a flexible routing for IPv6 traffic under the dynamically constructed IPv4/IPv6 co-exist network, thereby introducing the advantages of load balancing.

II. Related Works

The issue of the IPv4-to-IPv6 transition has been deeply discussed in [2]. A lot of methods are proposed to assure IPv4 and IPv6 interoperability, especially for connecting IPv6 nodes across IPv4 island networks which cannot be easily upgraded. In this case, configured IPv6-in-IPv4 tunnels [5] or Teredo [6] are considered as feasible solutions. However, such tunnels usually suffer from lack of optimality [7]. Another approach is to perform address translation by the combination of NAT64 [8]. However, with limited capacity, an NAT64 node can only handle a fixed amount of traffic [3], [7]. With flow-based switching and centralized management, an SDN enables a programmable and flexible network, which could be exploited to facilitate address translation and IP-forwarding [4], [9].

978-1-5090-5569-2/17/$31.00 © 2017 IEEE

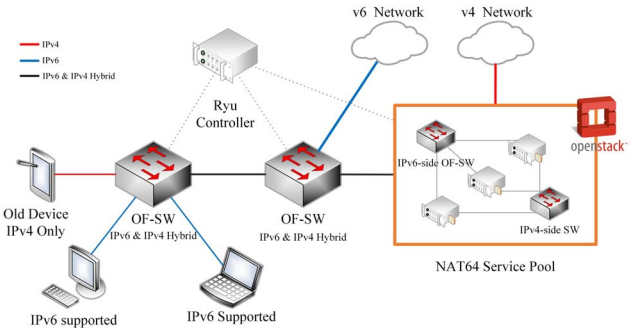

Fig. 1. The network architecture of IPv6 migration system over SDN

III. IPv6 Migration System Over SDN

A. Network Architecture

As shown in Fig. 1, we have IPv4 only network and IPv6 only network interconnected by three OpenFlow Switched (OF-SWs) that are controlled by a centralized controller. In this architecture, two OF-SWs supports IPv4/IPv6 hybrid network while a specific OF-SW is responsible for managing the NAT64 Pool and only offer IPv6 connectivity. In particular, IPv6-side OF-SW communicates with SDN controller to enable the load balancing for NAT64 nodes in the pool. The IPv4-side switch is just a general switch and OpenStack is exploited to deploy NAT64 nodes for address translation.

1) The Concept of NAT64 Service Pool: NAT64 service pool is built on top of OpenStadck to dynamically create and delete NAT64 service when it is necessary. Specifically, we increase the number of NAT64 services when utilization of IPv4 network is higher. As a result, computing resource

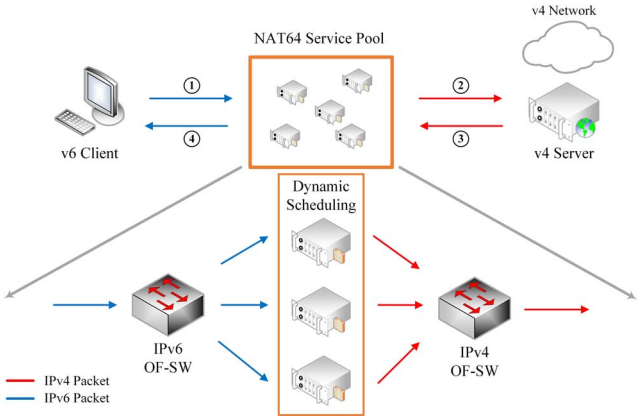

Fig. 2. The concept of NAT64 service pool

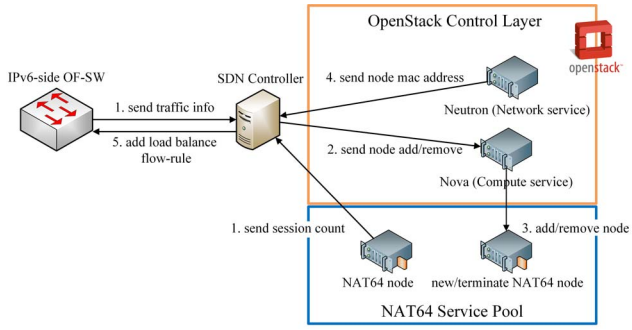

Fig. 3. Message flow of IPv6 migration system over SDN

is exploited in an efficient and flexible way. To prevent the NAT64 service from being the operational burden due to the construction overhead of NAT64, we create a certain number of NAT64 services in advance. With the control of OpenFlow control, the traffic will be redirected to the NAT64 pool.

B. System Architecture

As shown in Fig. 2, when an IPv6 client makes a connection to an IPv4 target, OF-SW will forward the traffic to the NAT64 node with minimum utilization in the service pool to achieve the load balancing. The traffic redirection to the chosen NAT64 is achieved by changing the destination MAC address on the OF-SW. The selected NAT64 node will repack IPv6 packets into IPv4 packets. Meanwhile, the SDN controller monitors each NAT64's utilization and check flow speed from OF-SWs. Whenever the SDN controller informs OpenStack to add or remove an NAT64 node, it updates the flow table to NAT64 edge OF-SW as well. As a result, the network architecture can be adopted immediately.

IV. Operations and Implementations

We use Ryu 3.4 as SDN controller and OpenvSwitch 2.3.0 to achieve good compatibility with OpenFlow 1.4. By installing OpenStack of the kilo version in x86 server, we implement components such as SDN controller, DNS64 (bind9), IPv4 router, IPv6 router, and NAT64 node (we apply Jool [10] here). Moreover, we adopt ARM-base network development board, Banana Pi R1, to simulate OF-SW.

As shown in Fig. 3, IPv6 devices can be obtained an IPv6-formatted IPv4 address by using DNS64 service. Then the IPv6-formatted IPv4 address will be routed to NAT64 service pool. As the packet reaches IPv6-side OF-SW, it will be forwarded to the SDN controller for the selection of NAT64 nodes. Then the packet will be transmitted to the chosen NAT64 node according to the source IPv6 address. After that, OF-SW will perform the same route assignment according to the flow-rule until the chosen NAT64 node is removed or the flow-rule is changed. Moreover, our system records the number of sessions of each NAT64 node from OpenStack and OF-SW. If too many IPv6-formatted IPv4 address packets exist, the SDN controller will call OpenStack nova APIs to create a new NAT64 node for the load balancing.

V. Conclusions

This paper proposed an IPv6 migration system over SDN by leveraging NAT64 service pool. The flexibility of SDN and dynamic of NAT64 service pool facilitates the IPv4-to-IPv6 address translation and achieves load balancing. Obviously, such system offers Internet service provider and enterprises an efficient solution where maintenance cost of NAT64 nodes is significantly reduced due to the flexible OpenStack and SDN. Moreover, the proposed system is compatible with the existing infrastructure, and thus shows the potential for the IPv4 to IPv6 transition.

Acknowledgement

This work is supported in part by Ministry of Science and Technology, Taiwan, under contract number MOST 104-2815-C-011-004-E.

References

[1] "Cisco visual networking index predicts near-tripling of ip traffic by 2020," Website. [Online]. Available: https://newsroom.cisco.com/press-release-content?type=press-release&articleId=1771211

[2] 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, July 2013.

[3] M. Bagnulo, A. Garcia-Martinez, and I. van Beijnum, "The NAT64/DNS64 tool suite for IPv6 transition," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 177–183, July 2012.

[4] S. Li, Y. Shao, S. Ma, N. Xue, S. Li, D. Hu, and Z. Zhu, "Flexible traffic engineering: When OpenFlow meets multi-protocol IP-forwarding," *IEEE Commun. Lett.*, vol. 18, no. 10, pp. 1699–1702, Oct. 2014.

[5] E. Nordmark and R. Gilligan, "Basic transition mechanisms for IPv6 hosts and routers," Internet Requests for Comments, RFC 4213, Oct. 2005.

[6] ——, "Teredo: tunneling IPv6 over UDP throughnet work address translators NATs," Internet Requests for Comments, RFC 4380, Feb. 2006.

[7] A. Azcorra, M. Kryczka, and A. Garcia-Martinez, "Integrated routing and addressing for improved IPv4 and IPv6 coexistence," *IEEE Commun. Lett.*, vol. 14, no. 5, pp. 477–479, May 2010.

[8] M. Bagnulo, P. Matthews, and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers," Internet Requests for Comments, RFC 6146, Apr. 2011.

[9] C.-W. Tseng, S.-J. Chen, Y.-T. Yang, L.-D. Chou, C.-K. Shieh, and S.-W. Huang, "IPv6 operations and deployment scenarios over SDN," in *Proc. IEEE APNOMS 2014*, Sept. 2014.

[10] "JOOL SIIT & NAT64," Website. [Online]. Available: https://www.jool.mx/en/index.html