# EVN: An Elastic Virtual Network supporting NFV Customized and Rapid Migration

Jian Zou, Jingyu Wang, Qi Qi, Haifeng Sun, Zhenguang Yu, Jun Xu
State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications, Beijing, China

***Abstract*—The virtual network functions (VNF) enables to reduce reliance on expensive proprietary networking gear and increase network elasticity. In this paper, we propose an Elastic Virtual Networks (EVN) system to realize a complete virtual network layout, which supports resource customization, automated deployment, and flexible migration. Through the port and MAC address mapping, a new physical node can access the Virtual Networks (VNs), and the related VNFs are deployed through Docker. Moreover, we set up a white list function to reserve the capacity of the hosts that will be accessed in the future to facilitate a fast access. The experiment results verify that the VNF migration time in EVN by Docker improves three to four times compared with the traditional VM solution.**

***Keywords*—Software-Defined Network (SDN), Network Function Virtualization (NFV), Service Function Chains (SFCs), Container**

## I. Introduction

Software-defined network (SDN) [1] breaks the architecture of the traditional network and proposes a new network model with separate data and forwarding. In OpenDaylight [2], we complete the Virtual Linking Networks (VLNs) which is based on Virtual Tenant Network (VTN), and the physical network can be mapped to the virtual network through VLNs. However, this model still does not solve the problem of migration among physical nodes. Most research has utilized the Docker [3] for web applications or databases due to its lightweight. For the networking perspective, the NFV[4] can also try to be deployed by Docker in the next level. Typically, VNFs are deployed in virtual machines that sit on top of a hypervisor, but Linux containers use a different structure that puts isolated Linux systems (containers) on a single Linux control host and allows more granular resource isolation and greater elasticity.

In this paper, we integrate VLNs of OpenDaylight and Service Function Chains (SFCs) of NFV to complete a complete virtual network layout system. The overall of the system is shown in Figure 1. Using Docker to encapsulate NFV-related VNFs such as Firewall and Load Balance. We can control the generation of multiple VLNs through OpenDaylight, and, each VLN is a standalone virtual network and isolated from each other. After the virtual network is created, we design two access methods i.e. port access and MAC address access. At the same time, all the hosts in the SDN network can be autonomously discovered, and the white list for the reserved hosts which will be accessed in the future is also set.

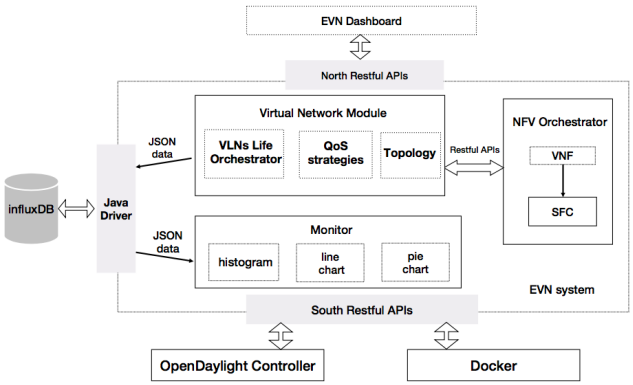


Fig. 1. EVN system architecture.

## II. EVN System

### A. Virtual Network

Virtual network service is the core component of the system and the main functions include the following parts:

- **VLNs Life Orchestrator**: OpenDaylight issues commands to generate virtual networks. An external host can access a virtual network through port access or MAC address access, at the same time, the two access modes can seamlessly switch. We have set up a white list mechanism and through this mechanism, we can add the IP address or MAC address of the host that will be accessing the virtual network to the white list.
- **Quality of Service (QoS) strategies**: We designed two general QoS strategies. One of them is Traffic Priority Strategy (TPS) and another is Rate Limit Strategy (RLS). In order to achieve traffic prioritization strategy, we designed 8 queues which named from Q1 to Q8. The priority relationship is highest for Q1, followed by Q2 lowest for Q8. Meanwhile, the speed of the network is limited through the MeterTable, the speed is less than or equal to the maximum bandwidth of the network and is greater than or equal to 0.
- **Topology**: The SDN controller controls the entire SDN network. For the overall SDN network, we can obtain the information of the physical network topology through the controller and construct the JSON data that we need.

978-1-5386-5156-8/18/$31.00 ©2018 IEEE

Through JSON data, we can draw related graphs or charts on the dashboard.

### B. NFV Orchestrator

The VNF images are encapsulated through Docker and these Docker images can form a number of logical functional links. In this way, all traffic that needs to pass through the function chains can be sent to Docker to complete the function of the function chains. OpenDaylight is also decoupled from the external Docker and is no longer integrated. This mode of importing traffic to Docker between OpenDaylight and Docker is designed to be highly efficient and usable, easy to deploy, and easy to migrate function chains provides the underlying support. In the case of rapid network building or disaster recovery, we can quickly establish or rebuild related networks. When it is necessary to perform service migration, we can also quickly migrate related VNFs to other servers to ensure that the logical network remains unchanged.

### C. Monitor

For the traffic data that needs to be stored, the current popular time series database influxDB is an excellent choice. By storing data in a database, EVN can generate line charts of data flow at a time, line charts of historical data, histograms of historical data and the like through visualization.

## III. Experiment Assessment

In order to evaluate our network system, in the experiment, we designed a SDN network. The topology of the SDN network is shown in Figure 2. The OpenDaylight-Carbon controller was deployed on a server equipped with 24GB of RAM and four Xeon(R) CPUs. The switches and a part of hosts in the lab environment were virtualized by the Mininet. Docker-17.09-stable is configured on another server with 24GB of RAM and four Xeon(R) CPUs.

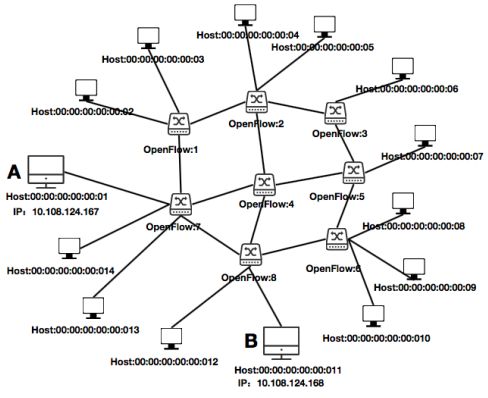


Fig. 2. Figure of the topology of the SDN network.

In the experiment, we used the other two servers connected the servers to the SDN network at the same time. The static IP of server A was 10.108.124.167, and the static IP of server B was 10.108.124.168. The test mode was that traditional VM mode and EVN mode were migrated from server A to server B. traditional VM mode is using VM and the basic migration methods. For the first time, we used traditional VM to encapsulate the relevant VNFs and then performed a static migration test on the VM. The second time, we used EVN mode to encapsulate and make an image, and then performed a static migration test on EVN mode. Figure 3 shows the difference in migration time between EVN mode.

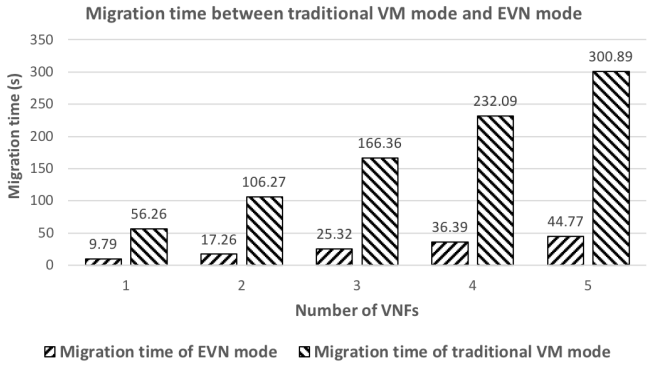


Fig. 3. Migration time between VM and Docker.

At the same time, we also tested the white list mechanism. We set the IP address of a computer to 10.108.124.20, in addition, we added the host to the whitelist, and then connected the host to the network. The EVN system can smoothly add the host to the network without any configuration. Afterward, we forcibly removed the host from the network, the EVN system can also successfully complete the deletion strategy of the host. Whether the host was joined the network or deleted the network, it will not have any impact on other hosts in the network. Our experiments verify that the EVN system has excellent elasticity.

## IV. Conclusion

The simulation results show that virtual network can be quickly created, modified and deleted, and these operations will not have any impact on other virtual networks, and the data show that using Docker is 3 to 4 times better in performance than VM.

## Acknowledgment

This work was jointly supported by: National Natural Science Foundation of China (No.61771068, 61671079, 61471063, 61372120, 61421061)

## References

[1] Foundation, Open Networking, *Software-defined Networking: The New Norm for Networks*, 2012.
[2] N. McKeown, "Software-defined networking," *INFOCOM keynote talk*, vol. 17, no. 2, pp. 30–32, 2009.
[3] D. Merkel, "Docker: Lightweight linux containers for consistent development and deployment," *Linux Journal*, vol. 2014, no. 239, p. 2, 2014.
[4] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, "Nfv: State of the art, challenges, and implementation in next generation mobile networks (vepc)," *IEEE Network*, vol. 28, no. 6, pp. 18–26, 2014.