TUNOS: A Novel SDN-oriented Networking Operating System

Tao Feng, Jun Bi, and Hongyu Hu
Network Research Center, Tsinghua University
Department of Computer Science, Tsinghua University
Tsinghua National Laboratory for Information Science and Technology (TNList)
fengt09@mails.tsinghua.edu.cn, junbi@tsinghua.edu.cn, and huhongyu@cernet.edu.cn

***Abstract*—Software defined networking (SDN) has been a promising network architecture to improve the openness of network and the diversity of protocols. Network operating system (NOS) in SDN is a key component for the abstraction of infrastructure and feature-rich protocols, which provide a general control plane and a unified protocol operating view. SDN-oriented NOS design requires not only the control shift from the specific network functions and vendor-dependent implementation in a traditional control plane to a general control functions, but also the extension of abstraction from computing process in a computer operating system to forwarding operation. To address this, we present a novel network operating system-TUNOS from the view of device control capacities and network control capacities. For the purpose of scalability, robustness, flexibility and high-performance, TUNOS provides open device management, cognitive network status, global network view, virtual forwarding space, and APP context management. General network control APIs are designed for user-friendly network programming.**

I. Introduction

FROM the view of SDN [1], SDN can enable control plane and data plane evolvable independently by decoupling control plane from a network device so that data plane can make progress in openness and high-performance while control plane can move forward to improve flexibility and scalability. The control plane in SDN as a middleware provides control interfaces to open network devices based on forwarding abstractions [2] and provides a global network view and general control APIs for network protocols.

The design and functions of traditional control plane have not satisfied with requirements of the control plane of SDN:

Control Structure. The control plane of SDN is deployed in a server or a cluster outside rather than embedded in a network device. The protocols can be deployed and operated in a standalone way to pursue flexibility and scalability.

Control Performance. The traditional control plane cannot improve process capacity due to sharing hardware with the data plane. The control plane of SDN can improve process capacity by means of a server or a cluster thanks to decoupling from the data plane.

This work is supported by National Science Foundation of China under Grant 61073172, Program for New Century Excellent Talents in University, Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant 200800030034

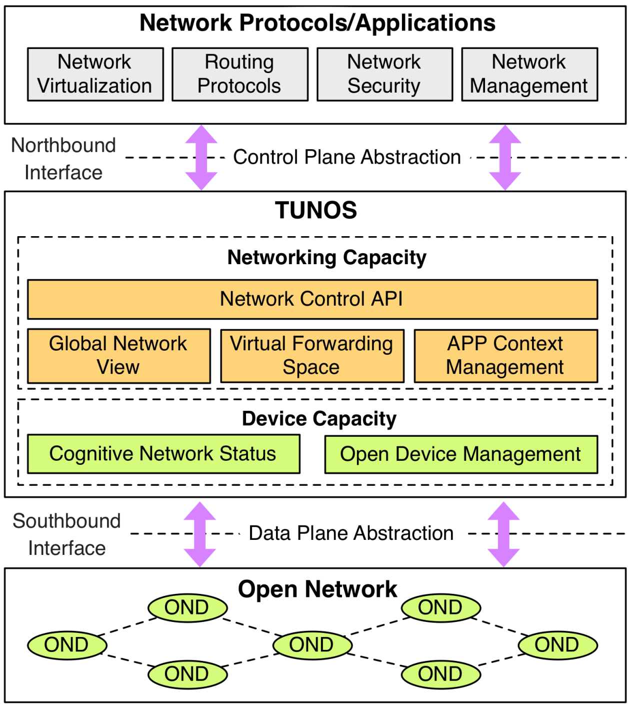

Fig. 1. TUNOS Architecture

Control Objects. The traditional control plane is for a single network device. The control plane of SDN is for a network, which is in charge of the discovery of interconnection and the handling of a mount of network status.

Control Mode. The traditional control plane uses distributed control mode generally, but the control plane of SDN uses centralized control mode (logically) with a global network view.

Therefore, the control plane of SDN needs to be re-designed for the isolation with hardware and software by analogy to computer operating system. We design a novel SDN-oriented networking operating system, named TUNOS, to abstract device resources, networking status and packet process for user-friendly network programming.

II. TUNOS Architecture

TUNOS architecture is shown as Figure 1. TUNOS provides the control capacities of open network device (OND) through southbound interface which is data plane abstraction. The functions of device control include open device management and cognitive network status. TUNOS provides the control capacities of network for network protocols through northbound interface which is control plane abstraction. The functions of network control include global network view, virtual forwarding space and app context management.

III. TUNOS Design and Implementation

A. Open Device Management

Open device management provides a unified identification for OND and a general access point for network protocols. File management is applied to organize and management open network device. When an OND (maybe an OpenFlow switch) connects with TUNOS successfully, TUNOS will mount the OND as a file node to device directory tree using IntialONDDaemon, such as '/dev/of/ofs_1', shown as in Figure 2. TUNOS will setup a mapping between the file node and physical OND, for example, the datapath ID of OpenFlow and a file node are a mapping. Thus, a network protocol can access an OND through the file node: open ("/dev/of/ofs_1", RO). An OND file node includes basic configure information: port/interface (type, quantity), forwarding table (quantity, match field, capacity), packet process ability (forward, drop and enqueue), etc.

B. Cognitive Network Status

Network status is a trigger for network protocol to generate or adjust network control rules. Network status includes device state and flow state. The former represents topology or performance change by means of OpenFlow or SNMP trap, the latter represents network application state by means of the first unmatched packet (OpenFlow) or sampling packet (NetFlow/sFlow) [3,4]. TUNOS dispatches network status information using event-loop using NetStatusDaemon in order to elastic scalability and non-blocking communication, shown as in Figure 2.

C. Global Network View

Global network view is a collection of ONDs and links. An endpoint of a link is a physical port or a logical interface of an OND. TUNOS can discover link layer topology by LLDP and network layer topology by the openness of routing information [3]. Network protocol can query topology, shown as in Figure 2, through the semantic description defined by Thrift which is a scalable cross-language framework [5].

D. Virtual Forwarding Space

The capacity of the memory in the network device are usually not enough and not extendable to flow-level forwarding. Therefore, virtualized forwarding space (VFS) is proposed to provide extendable and application based isolated logical forwarding space. VFS consists of some virtual forwarding pages (VFP) which are a part of memory space in an OND and a TUNOS server physically, shown as in Figure 2. The VFP in a TUNOS server is a primary page to cache all of forwarding rules of a network protocols and the VFP in an OND is a secondary page to cache a device-related subset of forwarding rules. TUNOS will swap forwarding rules in a primary VFP with secondary VFPs according to a tradeoff between the hit ratio of forwarding rules and cache

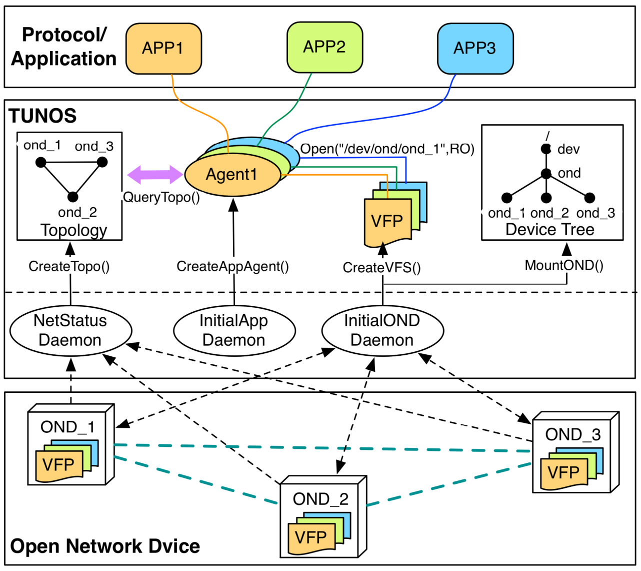

Fig. 2. TUNOS design and implementation

capacity of forwarding table in an OND. The mechanism of swapping VFPs is LRFU which is combined with LRU (Least Recently Used) and LFU (Least Frequently Used).

E. APP Context Management

APP context refers to the information and process that network protocols depend to generate network control rules. It includes OND tree, network view, VFS and network status. TUNOS takes use of process management to initial an operational context for a network protocol, shown as in Figure 2. When a network protocol registers to TUNOS, a daemon process, InialAPPDaemon, will create a process named App agent. Each App agent will be allocated and related with a VFS to cache forwarding rules. The App agent as a represent of a network protocol can subscribe the required types of network status, query topology, open OND, read or write VFP for control device.

IV. Conclusion and Future Work

In this paper, we present a novel network operating system-TUNOS from the view of device control capacities and network control capacities to achieve scalability, robustness, flexibility and high-performance. We have prototyped the functions of TUNOS. In future, we will validate TUNOS with some network protocol applications.

References

[1] Software-Defined Networking: The New Norm for Networks. Open Networking Foundation White Paper, 2012

[2] N. McKeown, T. Anderson, H. Balakrishnan,G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev. 38(2):69-74, 2008.

[3] T. Feng, J. Bi, H.Hu, OpenRouter: OpenFlow Extension and Implementation Based on a Commercial Router. In proceedings of the 19th IEEE International Conference on Network Protocols (ICNP11), Vancouver, Canada, 2011.

[4] A. Curtis, J. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee. Devoflow: Scaling flow management for high-performance networks. In ACM SIGCOMM, 254-265, 2011.

[5] M. Slee, A. Agarwal, and M. Kwiatkowski. Thrift: Scalable cross-language services implementation. Technical Report, Facebook Inc., 2007.