SFA: Stateful Forwarding Abstraction in SDN Data Plane

Shuyong Zhu, Jun Bi*, Chen Sun
Institute for Network Sciences and Cyberspace, Tsinghua University
Department of Computer Science, Tsinghua University

Abstract

Software Defined Networking (SDN) is a new network architecture where network control is decoupled from forwarding and is directly programmable. However, existing techniques provide limited support for stateful forwarding in SDN data plane. Relying on the controller for all state maintaining gives rise to scalability and performance issues. In this paper, we present Stateful Forwarding Abstraction (SFA) in SDN data plane. And we design a co-processing unit in SDN switches named Forwarding Processor (FP). It can deal with state information in data plane and its instructions can be flexibly extended to meet application requirements. Through SFA, we implement stateful network processing on the datapath which covers a full range of Layer 4 to Layer 7 services. We validate its performance based on IPsec. The experiment result proves that the forwarding efficiency is greatly improved.

1. Introduction

In the SDN architecture, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from applications. OpenFlow [1] is a representative technology of SDN. It was first proposed in 2008 as a way to externally control existing switches and routers using a match+action paradigm. OpenFlow may impliedly support stateful forwarding in data plane through OpenFlow instructions. However, today's SDN data plane still provides limited support for stateful forwarding. It limits the range of applications that can be implemented. For example, it is difficult to implement layer 4 to layer 7 traffic steering and layer 4 to layer 7 service functions such as IPsec, State Firewall, etc.

To date, some good research has been done to evolve SDN data plane. Minlan Yu proposed DIFANE [2], a scalable and efficient solution that keeps all traffic in the data plane by selectively directing packets through intermediate switches that store the necessary rules. DIFANE relegates the controller to the simpler task of partitioning these rules over the switches. Pat Bosshart described how to write programs using their abstract forwarding model and their P4 programming language in order to configure switches and populate their forwarding tables [3]. They propose a step towards more flexible switches whose functionality is specified in the field. In this paper we propose a different technique named SFA. Its architecture is shown in Fig.1. Through adding intelligence to the data plane, we can explicitly support stateful forwarding in SDN data plane.

* Jun Bi is the corresponding author.

2. SFA: Stateful Forwarding Abstraction

Concretely, we design a co-processing unit in SDN Switches named FP, which is implemented by CPU. Through extended OpenFlow instructions, flows or packets are forwarded from flow table to FP. FP realizes more complex processing of flows or packets. It maintains the associated state of them and executes stateful forwarding in data plane.

2.1. FP

FP can process the incoming flows or packets through instructions. And it can store and process state information of flows or packets. Its instructions can be flexibly extended, which is not easy to implement through hardware. As a result, it can further enrich functions in data plane to meet application requirements.

FP has one or more input and output ports. It may consult or update its internal state according to incoming packets. It can modify the metadata of the packets and it can generate and terminate some data packets asynchronously. FP also reacts to incoming events from the controller such as configuration change or state change. The controller and FP communicate with each other through asynchronous messages. The controller has full control of FP.

We design and implement the instructions of FP which are used to process the incoming flows or packets. They can be divided into the following categories.

- Control instructions: they are used to control the interaction between controller, flow table and FP, including GOTO, UPDATE_CONFIG, etc.
- Processing instructions: they are used for FP to process flows or packets, including MATCH, ADD, DELETE, MODIFY, DROP, OUTPUT, PARSE, QUEUE, etc.
- Operating instructions: they are used for FP to operate the state table, including INIT, CLEAR, FIND, INSERT, UPDATE, ERASE, ISEMPTY, SIZE, etc.

These instructions can be flexibly extended to support

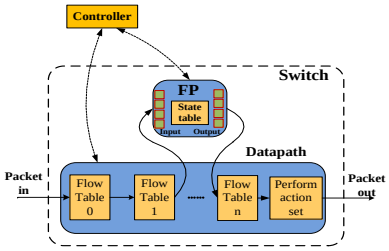

Fig. 1. the architecture of SFA

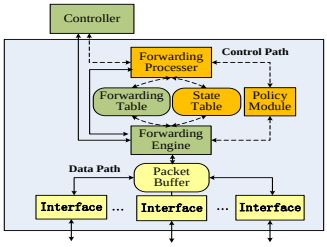

Fig. 3. SDN switch architecture

Fig. 2. Abstract view of state table

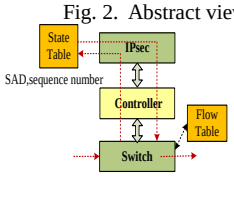

Fig. 4. OpenFlow-based

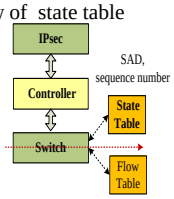

Fig. 5. SFA-based

different functions in data plane.

2.2. State Manipulating

We design state table for FP. It is used to store and process state information in data plane. The abstract view of state table is as shown in Fig. 2. Among them, the "Match field" field refers to the match field of packets. It is flexible and extensible and it can store a connection possibly represented by both the source and destination IP addresses. The "State" field is used to record the state information of flows or packets. The "Instructions" field is used to record the processing instructions as above mentioned. And we implement the operating instructions which are used to operate state table. These actions are as follows: INIT, CLEAR, FIND, INSERT, UPDATE, ERASE, ISEMPTY, SIZE.

2.3. SDN Switch Architecture We Propose

We design SDN switch architecture as shown in Fig. 3. We add FP and state table to store and process the state information in data plane. The Policy Module is used to adjust and control the forwarding policy of the switch.

3. Implementation

We validated the technology based on the use case of IPsec. We did a comparison test. First, we used controller to process the state information such as sequence number and SA in IPsec, as Fig. 4 shows. Then we implemented IPsec based on SFA, as Fig. 5 shows. In this architecture, we used FP to process sequence number and SA in data plane. Not every packet needed to be sent to controller to update state information.

We implement IPsec based on Huawei POF open source code [4]. We run the POF switch in Oracle VM VirtualBox on DELL OPTIPLEX 780 computer. The CPU of this computer is Intel® Core™ 2 Duo Processor E7500 (3M Cache, 2.93 GHz, 1066 MHz FSB). The network card is Intel(R) 82567LM-3 Gigabit Network Connection. We run Floodlight open source software as controller on this computer.

We each send 50000 packets to the switch in the two different implementations. We test the forwarding latency and the packet loss rate respectively. As can be seen from the experiment result shown in Fig.6, packet forwarding latency is an order of magnitude reduction in SFA architecture than processing state on the controller in traditional SDN architecture. And the packet loss rate is greatly reduced in SFA architecture as shows in Fig.7. Forwarding efficiency has been greatly improved.

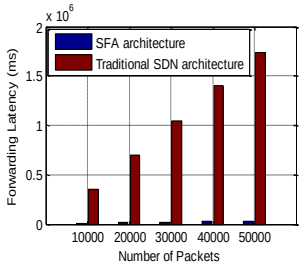

Fig. 6. Forwarding Latency

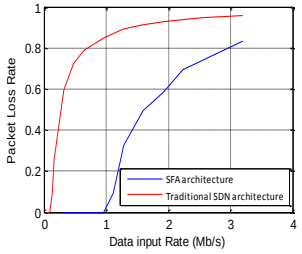

Fig. 7. Packet Loss Rate

4. Conclusion

In this paper, we propose SFA, through which we can explicitly support stateful forwarding in SDN data plane. And we design a co-processing unit in SDN Switch named FP, which can store and process state information in data plane. Its instructions can be flexibly extended and it can further enrich functions to meet application requirements. Experimental result demonstrates that forwarding efficiency is greatly improved. It can greatly increase the data plane programmability and it is conducive to network innovation. We believe it is a promising technique in the SDN data plane.

Acknowledgements

Supported by the National High-tech R&D Program ("863" Program) of China (No.2013AA010605).

References

[1] Nick.McKeown, T. Anderson, "OpenFlow: enabling innovation in campus networks", ACM SIGCOMM Computer Communication Review, Vol.38, 2008.

[2] Minlan Yu，Jennifer Rexford，Michael J. Freedman, "Scalable Flow-Based Networking with DIFANE"，SIGCOMM'10, September 3, 2010.

[3] Pat Bossharty,Dan Daly, Martin Izzardy, Nick McKeownz, Jennifer Rexford, Dan Talaycoy, Amin Vahdat , "Programming Protocol-Independent Packet Processors", arXiv:1312.1719v1 [cs.NI] 5 Dec 2013.

[4] Haoyu Song, "Portocol-Oblivious Forwarding: Unleash the Power of SDN through a Future-Proof Forwarding Plane. HotSDN13, August, 2013, Hong Kong.