Exercise for Lecture Software Defined Networking

Prof. Dr. David Hausheer, Julius Rückert
Christian Koch, Jeremias Blendin, Leonhard Nobach, Matthias Wichtlhuber

TECHNISCHE UNIVERSITÄT DARMSTADT

Winter Term 2016/17
Exercise No. 4
Published: 29.11.2016
Submission exclusively via Moodle, Deadline: 13.12.2016
Contact: Please use the Moodle forum to post questions and remarks on the exercise.
Web: `http://www.ps.tu-darmstadt.de/teaching/ws1617/sdn/`
Submission: `https://moodle.tu-darmstadt.de/course/view.php?id=8385`

Surname (Nachname):	
First name (Vorname):	
ID# (Matrikelnummer):	

Problem 4.1 - SDN Hardware

a) Explain in your own words the main differences between the OpenFlow hardware classes “SOFTWARE” and “HARDWARE”.

SOFTWARE: This class of openflow hardware implements additional openflow features based on the hardware of traditional network devices. This kind of openflow hardware could perfectly work with traditional devices, and the software implementation of openflow protocol doesn't cost much, so most of openflow devices are made in the class of software.
HARDWARE: This class features in ASIC. The design and optimization of ASIC hardware cost lot of money and could last a long time, so there are not so many choices of this class on the markt. The most important advantage of this class is the fast package processing speed.

b) OpenFlow v1.4.0 introduces the notion of *vacancy events*. Which inherent problem does this feature address? Shortly explain what *vacancy events* are and what would happen if a controller does not make use of them.

Vacancy events address the inherent problem that controller may sent too many entries to the openflow switch, which is more than the capacity of the switch. This may result in error and will be returned to the controller, then the controller have to deal with the errors first and cannot continue processing the incoming packets(in full speed).
Vacancy events could warn the controller when the number of entries in the switch reaches the threshold chosen by the controller, so that the controller could deal with this problem in advance to avoid it.

c) Explain and briefly discuss which parts of an OpenFlow flow entry should be stored in TCAM and which could be placed in normal DRAM to minimize the required TCAM space and still ensure constant-time packet processing. (Note: You can exclude the priority of a flow entry from the discussion.)

Hint: The answer can be derived from the material presented in the lecture. For a more detailed understanding, it might be helpful to understand how CAMs and TCAMs work on a lower level: `https://www.pagiamtzis.com/cam/camintro/` (this is optional material)

The part of matching field & priority should be stored in TCAM, while the other parts such as instructions, counters, timeouts could be placed in normal DRAMs.
To implement constant-time packet processing, first of all, the constant-time packet matching should be implemented. However, if the matching field is stored in a RAM space, the incoming packet must be matched with usage of searching algorithms, which means the matching time is not constant, but depends on the size of matching table. Therefore, matching field must be stored in TCAM to ensure the constant-time packet matching. In comparison to that, making use of other parts such as instructions is not dependent on the number of entries, so different instructions could be saved in different positions of DRAM and pointed by the matching field in TCAM.

Problem 4.2 - NEC guest lecture: OpenState

a) New Tables proposed by OpenState

In the lecture, an approach called *OpenState* was presented that extends the matching behavior of OpenFlow. For this, it proposes to expose two new table abstractions to the controller: the *state table* and the *XFSM table*. Explain the purpose of these two tables and how they are used in handling of incoming packets. For this, please have a look at the original paper on OpenState (Hint: Sections 2.1 and 2.2 are relevant for answering the question):

http://openstate-sdn.org/pub/openstate-ccr.pdf

State table is used to match packet header fields, and then add a state label to the packet as metadata. XFSM is responsible for the definition of a finite state machine with a 4-tuple (S, I, O, T) and a default state So, based on the state label from state table and the headers in the packet, XFSM table could match them to the action(s) on the packet and get a feedback to update the state table.

b) Applicability ot OpenState

Which types of use cases or applications could benefit most from OpenState?

Port Knocking

c) Using OpenState for a simple use case

At the end of Section 2.2 of the above referenced paper on OpenState, the “port knocking” example is realized using OpenState. Briefly explain how it is mapped to the two tables.

A packet from ip address 1.2.3.4, port 8456 comes to the OpenState machine;
The packet is matched in state table to the state STAGE-3, whicht becomes a label to the packet;
The labeled packet is transferred to the XFSM table and is matched again;
According to the match fieds, this packet gets the action drop & next-state OPEN;
The new state OPEN would be responsed to the state table, and the flow "IPsrc=1.2.3.4 -> STAGE-3" would be changed to "IPsrc=1.2.3.4 -> OPEN";
So if another packet from 1.2.3.4, port 22 comes to this OpenState machine, it could be forwarded to its destination, where port knocking is implemented.

Problem 4.3 - Network Virtualization and Slicing

a) Duties of an SDN Controller

Name three duties of a typical SDN controller and state at least two examples for each of them.

Monitor (e.g. physical network, VM locations & states)
Control (e.g. Tunnel setup, All packets on virtual and physical network, Virtual/physical mapping)
Tells OVS what to do (e.g. modify flow rules)

b) Network Virtualization with VLANs

Two datacenters D and E use VLANs for traffic isolation between their hosts. The datacenters are connected over a L3 connection, e.g. the Internet, and need to share their network segments.

I Propose a solution (without using SDN or VXLAN) for preserving the slices, even when communication is carried out over the L3 link.

Since VLAN of each datacenter are disjoint and packets can not be routed directly between them through a public network, all packets need to be encapsulated and transported through an ip tunnel. In order to accomplish this, two additional switches should be used as end points between two datacenters as routers to route packets among them.

II On an example of two hosts A (in Datacenter D) and B (in Datacenter E), draw a communication diagram of a packet sent from A to B over a common network segment. This diagram should include every intermediate hop (except pure L2 switches and simple routers on the Internet).

Host A ->End Point of D -> End Point of E -> Host B

III Draw the protocol stack (except Layer 1 and everything above the Transport Layer) of the packet into the diagram, between each two intermediate hops. Example: (Ethernet, IP, UDP)

(Ethernet, VLAN, IP, UDP) -> (Tunnel Header, Ethernet, VLAN, IP, UDP) -> (Ethernet, VLAN, IP, UDP)

IV Make yourself familiar with VXLAN and the VXLAN header. Name at least *two* advantages of VXLAN compared to your solution (Hint: Think about an intermediate NAT between the datacenters).

1. Using 24 bit VXLAN we can potentially create 16 million networks, compare this to only 4 thousand networks using VLAN with 12 bits ID.
2. Enables users to use the layer 3 features, which is impossible in VLAN.
3. The virtual layer 2 is invisible to the physical layer, this feature offers many benefits, such as enabling duplication of MAC addresses, as long as they don't exist in the same VXLAN segment.

V Name at least *one* advantage of OpenFlow compared to (pure) VXLAN in the context of network slicing.

A potential advantage of OpenFlow is that it uses local labels instead of encapsulation, which requires high performance of data plane. I.E. OpenFlow may reduce workload.

c) Energy saving

Imagine at least one scenario in which SDN can save energy.

As introduced in slice, SDN can save energy in a data center network. For example, control plan can be used as an effective controller instead of many other switches together.