Energy-efficient SDN Control and Visualization

Tao Li[1], Yuanjun Sun[1], Marek Sobe[1], Thorsten Strufe[1], and Silvia Santini[2]

[1]TU Dresden, Germany
[2]Universitá della Svizzera Italiana (USI), Switzerland
{*tao.li,thorsten.strufe*}@tu-dresden.de,{*yuanjun.sun, marek.sobe*}@mailbox.tu-dresden.de, silvia.santini@usi.ch

***Abstract*—This demo paper presents EConVi, a framework to support the implementation and visualization of the energy efficient software defined networking (SDN). This framework monitors the network workload, and dynamically changes the operational states of the switches and routing paths of flows, based on the controlling algorithms. The network topology, traffic workloads on each network link and switch, and the operational states of the switches are visualized. EConVi is able to manage the large-scaled networks simulated in the SDN emulators such as Mininet, or interact with DVFS-enabled hardware nodes installed with software switches, as shown in our demo.**

I. Introduction

Networking devices used in the large-scale data center networks or the backbones of Internet consume a large amount of energy and lead to the severe issue of carbon emissions. The approaches to reduce the energy resource consumption of the networks have attracted lots of research attentions. The major approach to reduce the energy consumption of the network is to identify those devices under low utilization and put them into the sleeping mode or power them off completely. In addition to changing the operational states of the network devices, the routing paths of data flows also need to be carefully planed, so that the data flows are aggregated to pass through only the activated switches and links.

SDN provides programmable data planes that can be configured by a remote controller. It also provides the interfaces to monitor the working states, like the flow statistics, of the networks. Several theoretical models and algorithms to achieve the energy efficiency in SDN have been proposed but they are rarely experimented with the real hardware [1], [2]. In order to support the deployment of the energy efficient algorithms, we present EConVi, a **E**nergy-efficient SDN **Con**trol and **Vi**sualization framework. EConVi provides the modules and APIs to implement a energy-efficient control loop for SDN, supporting dynamic collection of network workload data, integration of energy-aware algorithms, configuration of power-state of the switches and routing paths of flows, and interactive visualization of multiple relevant information.

II. Architecture

Figure 1 shows the overall architecture of EConVi, consisting of three layers: the visualization and administration layer, the control layer and the network layer.

978-1-5386-3416-5/18/$31.00 ©2018 IEEE

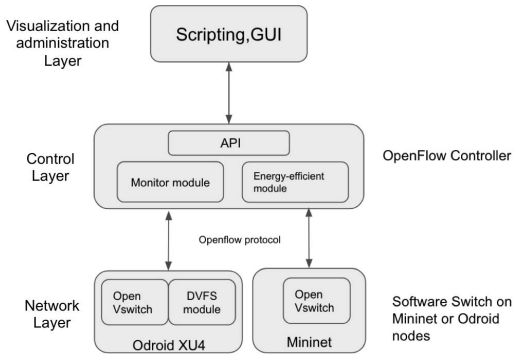

Fig. 1: System architecture of EConVi

The network layer consists of multiple software switches that are either managed by Mininet for the large-scaled networks or installed on hardware nodes for the small-scaled demo. These switches connect with each other to form a tree-liked topology with redundant paths. For the small-scaled hardware demo, as shown in Figure 2, we choose the commonly used Open vSwitch [3] as the software switch and Odroid [4] computing nodes as the hardware platform. Odroid is equipped with a CPU following the ARM Big.LITTLE architecture design, containing four Cortex-A15(”Big”) cores and four Cortex-A7(”LITTLE”) cores. This CPU has the functionality of Dynamic Voltage and Frequency Scaling (DVFS). Thus it is possible to change its operation frequency from 200MHz to the maximum clock frequencies of the big and little cores, at the interval of 200MHz. In this demo, EConVi provides APIs to select the operation mode of the Odroid nodes from two options: the high frequency mode and the low frequency mode, which lead to different energy consumption and packet throughput performance.

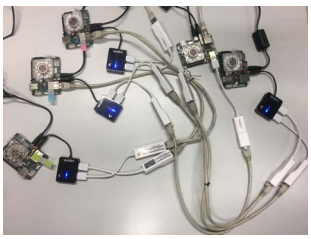

Fig. 2: Network built with Open vSwitch and Odroid nodes

The control layer consists of multiple EConVi modules

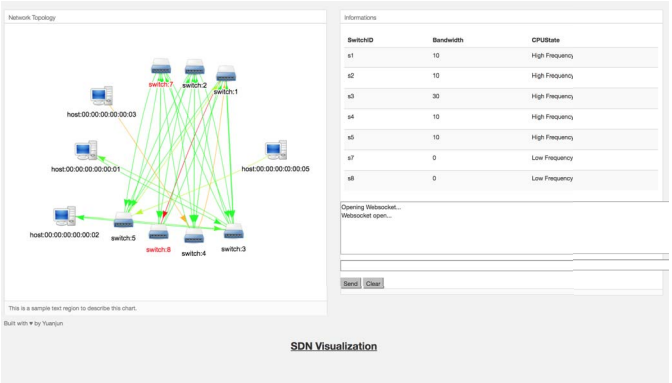

Fig. 3: Visualization and administration GUI

implemented in the Ryu controller [5], and the most important ones are the network monitor module and the energy-efficient algorithm module. The network monitor module is responsible of periodically collecting statistics about the flows and switches in the network via the standard OpenFlow APIs [6]. It collects the currently consumed bandwidth of flows and calculates the aggregated throughput of the switches. It also formats statistic data into the index-based structure that can be transmitted to the visualization and administration layer. Energy-efficient algorithm module is a template to implement those developed energy-efficient models and algorithms to dynamically decide: 1) the routing paths for the data flows, 2) the operation modes of the switches. In this demo, we implement an threshold-based algorithm, in which the current switch workload (measured in packet througput) obtained from the monitor module is compared with our predefined thresholds. Besides that, the edges switches connecting with hosts follows the on-demand principle, in which they remain in the high frequency mode as long as there are data packets transmitted from their connected hosts.

The visualization and administration layer provides the rich information of the network topology, the workload and operation state of each switch, as well as the workload of each link, as shown in Figure 3. It provides APIs like get_flow_stats(), get_all_topo() and get_node_info(), to query the relevant network state information. In addition, it also provides APIs to change the operation parameters of EConVi, for example, the frequency to measure the workload of the switches and the frequency to refresh the GUI.

III. Demo runs

In this section, we describe the process to use EConVi to control and visualize the behaviours of a emulated network to achieve energy reduction in SDN. This process also applies to our hardware setup with fewer hosts in the demo. Figure 4 shows the topology emulated in Mininet. Table I shows four flows to be added to the network. We create the flows using iperf with TCP and set their bandwidth and these flows one by one every 10 seconds.

At the beginning, there is no active flow in the network and all switches are in the low frequency mode. At the 10th second, the flow $f1 : h2 \Rightarrow h1$ is generated and only S3 is in the high frequency mode. At the 20th second, the flow $f2 : h3 \Rightarrow h1$ enters the network, thus S1 and S4 are set to the high frequency mode. At the 30th second, the flow $f3 : h5 \Rightarrow h1$ arrives. The controller chooses a path, which includes S1, for $f3$. Since the workload of S1 exceeds the threshold, $f3$ is scheduled to pass through S2, which is currently in the low frequency mode. To avoid the network performance degradation, S2 is set to the high frequency mode. At the 40th second, the flow $f4 : h7 \Rightarrow h4$ is created. The workload of those switches (S3, S1, S4, S2), that are currently in the high frequency, exceeds the threshold after adding $f4$. Thus, the controller decides to also set S7 to the high frequency and to move $f4$ to S7. Meanwhile, although $f1$ ends, S3 is still used by $f2$ and $f3$. All switches stay in the high frequency mode until the 70th second. At the 70th second, $f4$ ends, thus S8 and S7 are set to the low frequency mode. At the 80th second, $f2$ leaves, thus S4 is set to the low frequency mode. At the 90th second, $f3$ leaves, thus S1 and S5 are set to the low frequency mode. With the help of EConVi, the operation modes and workload of switches during this process can be clearly observed.

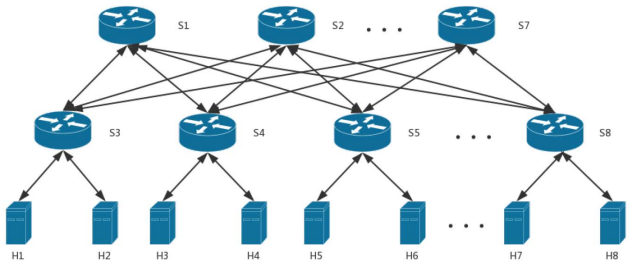

Fig. 4: Network topology emulated in Mininet

TABLE I: Details of each flow

Flow	From	To	Begin(sec)	Duration(sec)	Bandwidth(Mbits)
$f1$	h2	h1	10	30	10
$f2$	h3	h1	20	60	10
$f3$	h5	h1	30	60	10
$f4$	h7	h4	40	30	7

Acknowledgments

This work has been supported by the German Research Foundation (DFG) as part of the project A12 and A08 within the Collaborative Research Center (CRC) 912 – HAEC.

References

[1] M. Zhang, C. Yi, B. Liu, and B. Zhang, "Greente: Power-aware traffic engineering," in *Proceedings of the 18th IEEE International Conference on Network Protocols (ICNP)*, 2010.
[2] G. Lin, S. Soh, and K.-W. Chin, "Energy-aware traffic engineering with reliability constraint," *Computer Communications*, vol. 57, pp. 115–128, 2015.
[3] Open vSwitch. [Online]. Available: http://openvswitch.org
[4] Odroid-XU4. [Online]. Available: http://www.hardkernel.com
[5] Ryu controller. [Online]. Available: http://ryu.readthedocs.io/en/latest
[6] OpenFlow 1.3. [Online]. Available: https://www.opennetworking.org