The Effect of Proactive Flow on Network Downtime in Software Defined Network Using Opendaylight Controller

Rendy Munadi[1], Danu Dwi Sanjoyo[2] and Hana Rizky Herdika[3]
[123]School of Electrical Engineering
Telkom University
Bandung, Indonesia, 40257
E-mail: {rendymunadi, danudwj}@telkomuniversity.ac.id; hannrey@student.telkomuniversity.ac.id

Abstract—Computer vision is one of the favorite research topics recently, This paper will analyze the effect of proactive flow on network downtime when there is a link failure. The network is designed using a data center network model and OpenDaylight as SDN controller. Proactive flow has global view of the network before the first packet arrives. Refers to that, the changes of the network when sending packets will have an impact on the continuity of packets transmission. The result shows that downtime due to the link failure during packets transmission is 3.5 seconds.

***Keywords— Software Defined Network, OpenFlow, OpenDaylight, Proactive Flow*.**

I. INTRODUCTION

Software Defined Network (SDN) is a new paradigm to manage networks by separating the control plane and data plane. The separation offers efficiency on maintaining the network because it centralized on SDN controllers. The SDN considered as the answer for telecommunications industry problems where the networks currently very complex and growing time by time. However, the separation can refer to another problem where the communication between data plane and control plane can result delay on the network.

Forwarding decision on data plane are flow based[1]. The SDN controller select which path the packets must pass by installing flow entries to the flow table in each forwarding devices[2]. It means there are a set of rules that has to matched by the packets and instructions how the suitable packets will be processed. OpenFlow protocol commonly used in SDN for communication between the control plane and data plane[1]. With OpenFlow the SDN controllers can manage, add, update, and delete the flow entries on forwarding devices[3]. The flow entries can be installed statically or dynamically by proactive, reactive, or hybrid flow.

Currently, SDN controller applications using proactive flow as main logic to install the flow entries[1]. With proactive flow the controller already has the global view and infomation of the networks. The flow entries have been defined before the first packet arrives[4]. When the first packet arrived in the networks then forwarding devices send packet_in to the controller and the controller will install the flow entries to every forwarding device. The proactive flow algorithm presents on Fig. 1.

```
Input: G,O
for (s, d) in O do
  path ← shortest_path(s, d)
  inst ← create_openflow_instructions(path)
  for v in path do
    if FT(v, t) < C_v then
      if d is connected to v then
        out_port ← inst[v, d]
      else
        out_port ← inst[v, v+1] {v+1 is the next hop}
      end if
      action ← forward_to(d, out_port)
      match ← MT(s, d)
      opo ← create_openflow(out_port, match,
      action)
      send(opo, v)
    end if
  end for
end for
```

Fig. 1. Proactive Flow Algorithm [4].

Refers to that, the changes on the network such as link failure or switch failure can lead a downtime because a new flow entry must be updated. The goal of this paper is to calculate how much time the controller takes to update the new flow entries during packets transmission.

II. SYSTEM MODEL AND SCENARIO

This research will be simulated virtually, Fig. 2. The emulator that we used to create the data plane is mininet because it supports OpenVSwitch. For the SDN controller we used Opendaylight framework. Opendaylight framework widely used on data center model network. The flow entries in OpenVSwitch will be installed by proactive flow as we want to measure the network downtime with this flow approach. The entire system will be virtualized using the VirtualBox hypervisor.

We conduct a series of test by terminate a link when packets transmission occurs. By varying the number of packets, 1000, 5000, 10000, 15000, and 20000. Five attemps are made on each number of packets and the delay between each packet sent is 1ms. In the end, we calculate the average downtime. Downtime calculated by:

$$Downtime(s) = RTT\ total\ before\ link\ failure(s) - RTT\ total\ after\ link\ failure\ (s), \qquad (1)$$

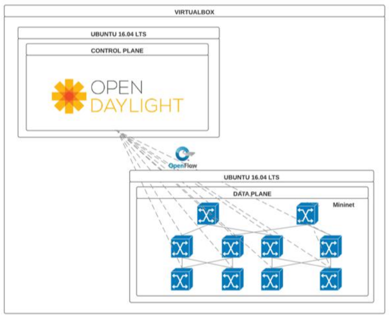

Fig. 2. System model of the test.

III. PERFORMANCE ANALYSIS

From the test we got result, Fig. 3., downtime average 2.2 seconds on 1000 packets, 3.3 seconds on 5000 packets, 3.6 seconds on 10000 packets, 3.4 seconds on 15000 packets, and 4.8 seconds on 20000 packets. Then we calculate by average the result. So the downtime network is 3.5 seconds.

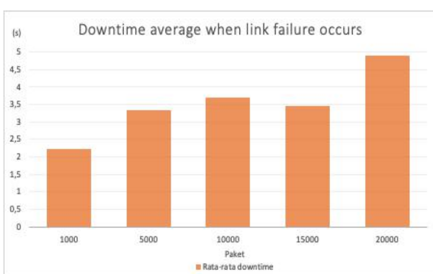

Fig. 3. Downtime average when link failure occurs.

IV. CONCLUSION

The network changes during packets transmission indeed results an extra delay. With proactive flow as the approach to install the flow entries, it takes 3.5 seconds downtime to update the flow table and continue the packets transmission.

REFERENCES

[1] D. Kreutz, F. M. V Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey - Annotated,” *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[2] M. Dusi, R. Bifulco, F. Gringoli, and F. Schneider, “Reactive logic in software-defined networking: Measuring flow-table requirements,” *IWCMC 2014 - 10th Int. Wirel. Commun. Mob. Comput. Conf.*, pp. 340–345, 2014.

[3] Open Networking Foundation, “OpenFlow Switch Specification”, version 1.3.0.” 2012.

[4] W. J. A. Silva, “Performance Evaluation of Flow Creation Inside an OpenFlow Network,” *XXXV TELECOMUNICAC, OES˜ E Process. SINAIS*, 2017.