SNS Information-based Network Control System developed on FLARE Experiment Environment

Haruka Yanagida, Ochanomizu University, Bunkyo, Tokyo, Japan, haruka@ogl.is.ocha.ac.jp
Akihiro Nakao, University of Tokyo, Bunkyo, Tokyo, Japan, nakao@iii.u-tokyo.ac.jp
Shu Yamamoto, University of Tokyo, Bunkyo, Tokyo, Japan, shu@iii.u-tokyo.ac.jp
Saneyasu Yamaguchi, Kogakuin Univirsity, Shinjuku, Tokyo, Japan, sane@cc.kogakuin.ac.jp
Masato Oguchi, Ochanomizu University, Bunkyo, Tokyo, Japan, oguchi@is.ocha.ac.jp

***Abstract*—To achieve network availability in disaster situations, we propose the traffic control system based on SNS information. In order to utilize the data of packet payload obtained from SNS, Deeply Programmable Network (DPN) is needed. In this paper, we implement and evaluate the proposed method with FLARE switch and achieve more flexible control of the network.**

***Index Terms*—SDN; OpenFlow; DPN; FLARE; SNS;**

I. Introduction

When the Great East Japan Earthquake occurred in 2011, network disconnection occurred in some areas. This event has increased the necessity of a system that can grasp all network conditions immediately and is entirely automatic in Japan. Several researchers have applied SDN and OpenFlow to wide area networks to centralize the control of network devices using a software controller. However, the following two challenges must be considered. First is the difficulty of detecting failure by only monitoring using sensors inside of network when the target area is wider. Second is the limit of programmability of SDN. SDN enables control plane (C-plane) being programmable, but the data plane (D-plane) is NOT programmable and still has a hardware component.

In order to address these challenges, we propose the traffic control system based on Social Networking Service (SNS) information in a Deeply Programmable Network (DPN). In addition to the monitoring, using real-time SNS information to detect network failures from outside of network devices [1] is one of the solution for the first issue. It can specify the area of network failure and grasp all network conditions immediately. The solution for the second challenge is the DPN. DPN is the concept of a software-defined D-plane, meaning the full programmability of a network. DPN enables to access the data of packet payload obtained from SNS. Therefore for example QoS control for every application becomes possible. In disaster situations, it is assumed that prioritizing important traffic, such as mail, phone, or SNS, rather than mobile video traffic is effective. In this paper, we use the FLARE switch [2] which was developed to achieve the DPN environment. In FLARE, the D-plane is implemented with Click software module router which is a language that defines the operation of network device. This switch creates virtual networks which is called Slice composed of virtual switches written in Click. Figure 1 shows network environment including the proposed system. As shown in this figure, SNS information is collected and analyzed. Based on those, the Network Controller makes a decision for routing and bandwidths control for each slices where are applied each applications, so that it sends a direction to switches on the network.

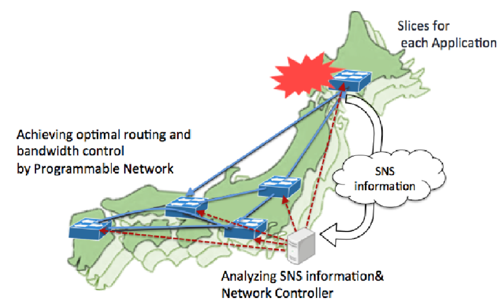

Fig. 1. Network environment achieved by the proposed system

II. Overview Of The Proposed Method

The process flow of the proposed method is as follows. All processes are executed automatically and autonomically.

(1) Receive network failure information by analyzing SNS
By the system discussed in [1], we analyze Twitter in real-time and detect network failure with high accuracy.

(2) Update the costs of links
For example, the default costs of all links are 1. If there are more than 20 tweets including the mapped area name in extracted tweets, increment the costs by 1. Updating is performed at 60-seconds intervals.

(3) Optimal route search
The minimum cost of a route is set as the optimal route by Dijkstra's algorithm.

(4) Route resetting
The optimal route is reset by applying flow entry to the switch through REST-API.

(5) Application QoS control
We classify applications and assign one application to one slice [3]. For each slice, set D-plane as optimal bandwidth programmed by Click module router.

III. Experiments and Evaluation

A. FLARE Environment

Figure 2 shows experiments environment. 1G indicates 1Gbps connection, 10G indicates 10Gbps connection. The controller is implemented on this FLARE Central server. Table I shows the specifications of the machines.

TABLE I
SPECIFICATIONS OF MACHINES

FLARE switch1 ∼ FLARE switch4	CPU	Core i7-3612QE Mobile 2.1GHz
	Memory	8GB
	OS	CentOS 6.4
h1∼h4	CPU	Core i5-4210 M 2.6GHz
	Memory	8GB
	HDD	SATA 500GB 5400RPM
	OS	Ubuntu14.04
h5, h6	CPU	Xeon E3-1241 v3 3.5GHz
	Memory	8GB
	HDD	SATA 1TB 7200RPM
	OS	Ubuntu14.04

B. Verifying the Proposed System Experiment

We verify our proposed system of (1)-(4). In this experiment, (5) is operating OpenFlow1.3 by the Ofswitch module of Click. Accordingly, the SDN level experiment is executed on FLARE, which has the capability of DPN. In this experiment, we use actual tweets from 14:00 to 15:00 on 11 March 2011 when the Great East Japan Earthquake occurred. We map from FLARE Switch1 to 4 to "Iwate", "Kyoto", "Tokyo", "Fukuoka", respectively. For example, the start point of communication is set to h1 near Iwate, and the goal point is set to h5 near Tokyo. Through this experiment, we verify the proposed system can switch the routes which go around the disaster area in a disaster situation based on Twitter information. Specific operations are as follows.

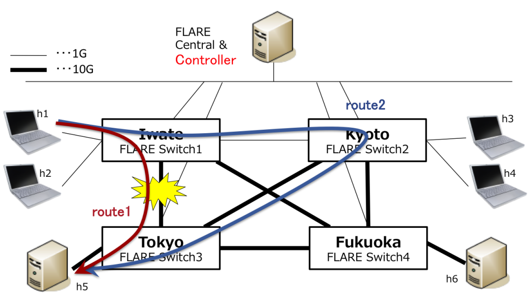

Fig. 2. Network Physical Diagram

Firstly, route1 shown in Figure 2 is used because the default cost of all links is 1, so the minimum cost of route1 is 1. Then, 120 seconds later, the link cost between Iwate and Tokyo is updated to 2 because the Twitter analyzing system detects more than 20 tweets related to network failure in Iwate and Tokyo around this time. The tweets continue increasing, so the cost of route1 is incremented. 180 seconds later, the cost between them become 3, so route2 is selected because route2 costs 2, which is the minimum. As a result, the route is switched by REST-API, which sets the route1 to route2.

C. Throughput Evaluation Experiment

To evaluate the proposed system, throughputs are measured by iPerf. Figure 3 presents a difference of performances between the case of the proposed system and the normal one. Because of almost no difference between them, we confirm that the proposed system achieves almost full performance of Hardware without overhead.

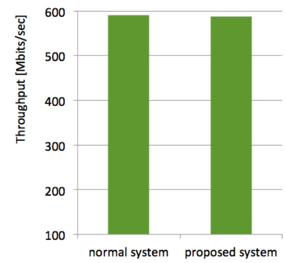

Fig. 3. Comparison Throughput

D. RTT Evaluation Experiment

To see the time of switching the routes, RTT are measured using a program sending Ping every one millisecond. The outcome is shown in Figure 4. Approximately 20-30 ms RTT to switch routing can be seen because the packets do not take a new right path while route is in the middle of switching.

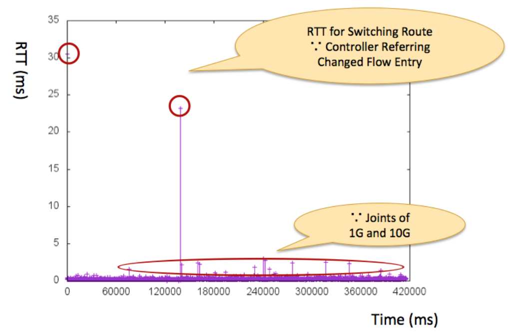

Fig. 4. RTT

IV. CONCLUSION

In this paper, we propose programmable network control based on SNS information, which optimizes traffic for each application automatically and autonomically. As a contribution, we built a system for switching routes to avoid a disaster area based on Twitter information, and achieved almost full performance of hardware without overhead. In addition, we confirmed that our system can be operated with sufficient performance on a wide area network from RTT.

As a next step, we plan to extract more detailed situations of users from Twitter and achieve unique QoS control reflecting those.

ACKNOWLEDGMENT

This work is partially supported by the Strategic Information and Communications R&D Promotion Program (SCOPE).

REFERENCES

[1] Chihiro Maru et. al., "Network Failure Detection System for Traffic Control using Social Information in Large-Scale Disasters", ITU Kaleidoscope Conference 2015: Trust in the Information Society, S5.3, Dec. 2015.

[2] Akihiro Nakao, "Software-Defined Data Plane Enhancing SDN and NFV", IEICE Transactions on Communications, vol.E98-B, No.1, pp.12-19, Jan. 2015.

[3] Akihiro Nakao et. al., "Application Specific Slicing for MVNO through Software-Defined Data Plane Enhancing SDN", IEICE Transactions on Communications, Vol.E98-B, No.11 pp.2111-2120, 2015.