Dynamic Flow Steering for IoT Monitoring Data in SDN-coordinated IoT-Cloud Services

Heebum Yoon, Seungryong Kim, Taekho Nam, and JongWon Kim
School of Electrical Engineering & Computer Science
Gwangju Institute of Science & Technology
Republic of Korea
{hbyoon, srkim, thnam, jongwon}@nm.gist.ac.kr

***Abstract*—By effectively coupling IoT and Cloud together, we need to enable diversified IoT-Cloud services. To effectively support IoT-Cloud services, it is essential to maintain persistent data transport between IoT and Cloud. Also the emerging Software-Defined Networking (SDN) paradigm can assist flexible flow-centric networking of persistent data between IoT and Cloud. Thus, in this paper, we take an example of IoT-Cloud service realized over miniaturized IoT-SDN-Cloud environment (named as SmartX-mini Playground) and propose the application of SDN-based flow steering to dynamically adjust the overlay data paths for IoT-Cloud services. More specifically, we attempt to combine the overlay data transport of Apache Kafka messages and the underlay flow-based networking coordinated by an ONOS (Open Networking Operating System) SDN controller.**

Keywords— Software-Defined Networking (SDN), flow steering, Internet of Things (IoT), and Apache Kafka messaging.

I. INTRODUCTION

To process and store huge data gathered from distributed IoT (Internet of Things) devices, emerging IoT services are being tightly coupled with the cloud-based shared infrastructure [1]. To exploit the value of dispersed IoT data, we need to securely collect and quickly analyze a large amount of data [2] without any serious interrupt. That is, a persistent monitoring (including the collection and analysis) of IoT data is required between IoT devices and the supporting cloud infrastructure. Based on this IoT and cloud linkage, so-called IoT-Cloud services can be efficiently created. Then, in order to maintain the linkage reliably for IoT-Cloud service, we can utilize the flexible flow-centric tagging, steering, and mapping capability of SDN (Software-Defined Networking). From logically-centralized SDN controllers, the operators can flexibly prepare multiple paths to support the inter-connection needs of targeted IoT-Cloud services.

Thus, in this paper, we demonstrate the support capability of SDN-based flow coordination for IoT-(SDN)-Cloud services. As an example case, the proposed SDN-based flow coordination (mostly for flow steering) assists the resource monitoring functionality for a miniaturized IoT-Cloud testbed environment, named as SmartX-mini Playground. In this example, persistent monitoring of SmartX-mini Playground is assisted to enable fast and timely delivery of IoT device status data. As depicted in the upper side of Fig. 1, IoT monitoring data is delivered by Apache Kafka [1] messaging functions located in both IoT and cloud sides [3]. In the bottom side of Fig. 1, ONOS SDN controller is adopted to assist flexible flow-based steering for IoT monitoring data [4].

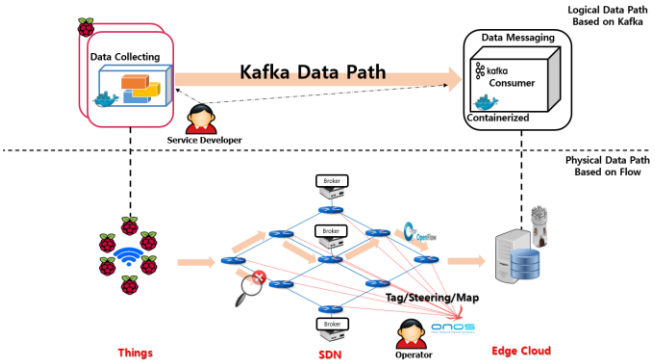

Fig. 1: SDN-coordinated delivery of IoT data for IoT-Cloud services.

In summary, to support IoT-Cloud services by persistently collecting IoT monitoring data, we combine Kafka-based overlay data delivery with SDN-coordinated underlay flow-steering. More specifically, we prepare flow-based multiple paths coordinated by the SDN controller. Note that a prior knowledge about the underlay networking topology is assumed. Next, we maintain the continuity of streaming-style delivery of Kafka-based IoT monitoring data messages. Here mix-and-matching both Kafka data transport with SDN-coordinated flow-steering is the key factor in maintaining the persistent delivery of IoT monitoring data.

The remainder of the paper is organized as follows. In Section II, we discuss the design and implementation details about ONOS-coordinated IoT data monitoring over SmartX-mini Playground. Next, in Section III, we discuss the verification result and then conclude the paper in Section IV.

[1] Apache Kafka is an open-source distributed messaging package originated from LinkedIn, which aims to provide a unified, high-throughput, low-latency solution for handling real-time data feeds. It is, in its essence, a "massively scalable pub/sub message queue architected as a distributed transaction log," making it highly valuable in processing streaming data.

II. SDN-BASED FLOW STEERING FOR IOT-CLOUD SERVICES

A. Overview

In order to assist IoT monitoring for targeted IoT-Cloud services, reliable data transport is essential between IoT and cloud as depicted in Fig. 1. We deliver IoT monitoring data through Kafka-based overlay data streams. They are not directly coupled with SDN-coordinated flow-based multiple paths. Kafka, communicating via TCP/IP protocol, creates replications of messages and stores them in multiple (and partially redundant) Kafka brokers. Therefore, if a particular broker cannot operate normally, messages can be delivered through other brokers. In addition, they have disk-based buffering (i.e., smoothing) capability to cope with latency and loss problems, caused by networking device and link failures from underlay SDN-coordinated networking. Also SDN-coordinated flow-based multiple paths are enabled by OpenFlow-enabled switches controlled by the ONOS SDN controller. The SDN controller should identify faulty paths and dynamically adjust the associated paths (for each Kafka data stream) to resolve the faults. Thus, if a particular Kafka broker becomes inaccessible, we attempt to resolve it by modifying the associated flow-based steering.

Thus, with the improved SDN-coordinated coupling between underlay and overlay transports, it would be possible to resolve problems in a responsive manner with the proposed approach. To verify its feasibility, we first utilize SmartX-mini Playground, to be explained soon. We then coordinate the coupling by implementing an ONOS SDN application.

B. SmartX-mini Playground

Table 1. SmartX-mini Playground: Physical hardware specifications.

Name	Device	Aim	OS
µ-Box	Raspberry Pi2	IoT device	Hypriot OS
IoT Actuator	Intel NUC	IoT gateway	Ubuntu 14.04
DevOps Tower	Intel NUC	DevOps Tower	Ubuntu 14.04
µCloud Box	Intel ONP Server	µCloud Server	Ubuntu 14.04
SDN Controller	Intel NUC	SDN Controller	Ubuntu 14.04
SDN Switch	MikroTik CRS125	OpenFlow Switch	Router OS MIPSBE 6.34
SDN Switch	MikroTik RB750Gl	OpenFlow Switch	Router OS MIPSBE 6.34

The adopted development environment, SmartX-mini Playground, is depicted in Fig. 2. It is designed as a low-cost open-source leveraged environment to support diverse experiments for IoT-(SDN)-Cloud services [5]. It is composd of multiple Raspberry Pi2 µ-Boxes, Intel NUC IoT Actuators, Mikrotik OpenFlow switches, and Intel ONP µCloud Box, as summarized in Table 1. Also, typical software configuration for SmartX-mini Playground is depicted in Fig. 3.

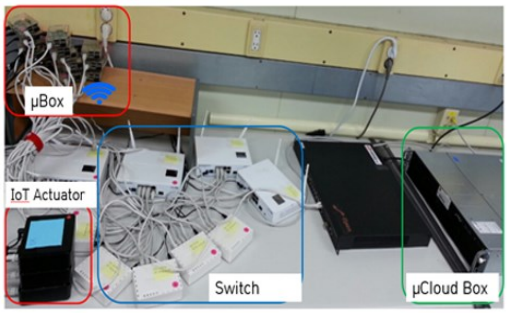

Fig. 2: SmartX-mini Playground: Physical resources.

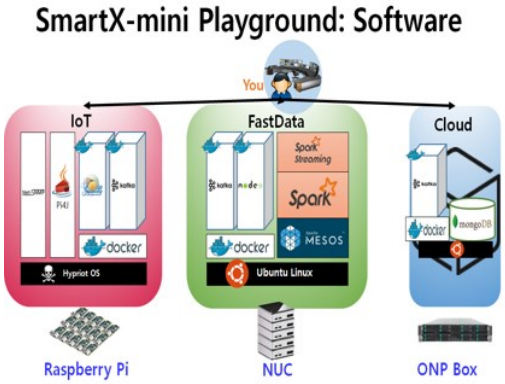

Fig. 3: SmartX-mini Playground: Software.

C. Resource Monitoring for IoT-Cloud Services

As a testing purpose, IoT monitoring functionality about the resource status of µ-Boxes, IoT Actuators, and a µCloud Box is implemented [6]. As shown in Fig. 3, most software components run as Docker container-based functions because of easy and fast deployment. µ-Boxes use Apache Flume to collect the resource status of each µ-Boxes via a SNMP daemon and sends monitoring data out to Kafka brokers (located in IoT Actuators). In detail, they collect IoT monitoring data from µ-Boxes through Kafka Producer API. Monitored IoT data is composed of 16 kinds of information including CPU, memory usage, and I/O usage. Kafka consumer located in the µCloud Box receives and then consumes it.

D. ONOS SDN Application for Dynamic Flow Steering

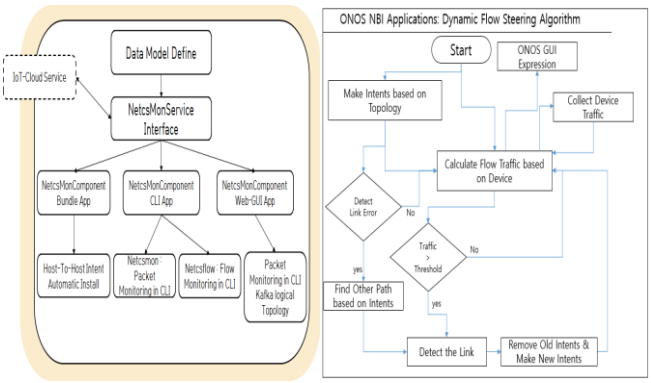

Fig. 4: Implementation and processing algorithm for 'Steering Function'.

To support the persistent data delivery of IoT monitoring data, we implement ONOS northbound application called as 'Steering Function', whose implementation detail is depicted in Fig. 4. It covers the role of delivering monitoring packets by manipulating host-to-host forwarding rules according to the single-to-multiple and multiple-to-single intents of ONOS SDN controller. After installing intents, it keeps tracking of the link traffic at all switches in order to guide the flow steering. When a physical link gets down, it enables adaptive transfers to other paths based on installed intents. When a path gets congested beyond preset threshold (e.g., 4MB for 21,000 packets/link for every 10 min), it replaces old intents with new ones.

III. SDN-COORDINATED FLOW STEERING: VERIFICATION

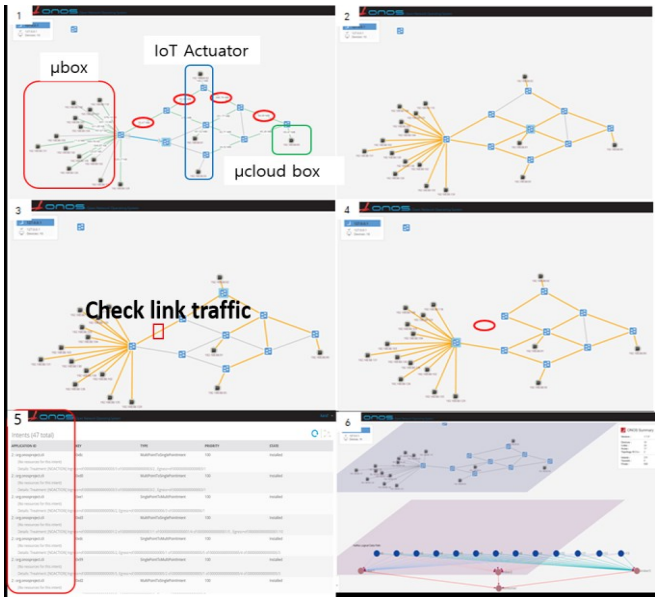

Fig. 5: SDN-coordinated flow steering example.

To verify our proposed approach, we integrate and evaluate it as shown in Fig. 5. First, it shows the overall topology and flow inter-connections over SmartX-mini Playground. It also shows that the modified inter-connections after a link failure (Step 3 and 4 of Fig. 5). Also, Step 6 of Fig. 5 shows the topology of Kafka-based overlay paths.

Table 2: µ-Box: average ping test results.

µ-box	IoT Actuator (192.168.88.92)	IoT Actuator (192.168.88.92)	µCloud box (192.168.88.92)
Fig. 5. Step 3			
192.168.88.104	1.929 ms	0.732 ms	0.917 ms
192.168.88.103	0.921 ms	0.711 ms	0.694 ms
192.168.88.135	0.913 ms	1.016 ms	0.841 ms
192.168.88.129	0.839 ms	0.901 ms	0.862 ms
Fig. 5. Step 4			
192.168.88.104	1.132 ms	0.945 ms	0.929 ms
192.168.88.103	1.077 ms	0.916 ms	1.012 ms
192.168.88.135	1.108 ms	0.972 ms	0953 ms

Table 2 represents averaged ping tests from µ-Boxes to IoT Actuators and µCloud Box when the circled link gets broken. After the link goes down, ICMP packets get delayed on average 0.156 ms. But this delay can be tolerated with the bufferings at Kafka brokers. Also, once the successful delivery of IoT monitoring data is completed, we can enjoy the Grafana-based visualization of collected data as shown in Fig. 6.

Fig. 6: The collected resource visibility for µ-Boxes.

IV. CONCLUSION

In this paper, we presented dynamic flow-based path steering for IoT monitoring data in SDN-coordinated IoT-Cloud services. By combining the overlay data transport of Apache Kafka messages and the underlay flow-based networking coordinated by an ONOS SDN controller, we can partially verify the feasibility in providing more flexible and persistent data delivery for IoT-Cloud services.

ACKNOWLEDGEMENT

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grants funded by the Korea Government (MSIP): No.B0115-16-0001, 5G CHAMPION (Communication with a Heterogeneous, Agile Mobile network in the PyeongChang wInter Olympic competioN) and No. B0190-16-2012, K-ONE (Korea – Open Networking Everywhere) Global SDN/NFV Open-Source Software Core Module/Function Development.

REFERENCES

[1] K. S. Yeo, M. C. Chain, T. C. W. Ng, and A. T. Do, "Internet of Things: Trends, challenges and applications," in *Proc. IEEE ISIS,* Mar. 2014.

[2] Y. Kim and S. Shin, "Design secure IoT environments with SDN and NFV." *KICS Information and Communications Magazine*, pp. 27-33, vol. 32, no. 7, July 2015.

[3] Kafka, http://kafka.apache.org/.

[4] ONOS, http://onosproject.org/.

[5] H. Yoon and J. Kim, "SDN-based data routing to support services utilizing SmartX-mini Playground systems," in *Proc. KICS 2015 Fall Conference,* Seoul, Korea, Nov. 2015.

[6] S. Kim and J. Kim, "Enabling container-based operational visibility for SmartX-mini Playground," in *Proc. KICS 2015 Fall Conference*, Seoul, Korea, Nov. 2015