SAPS: Software Defined Network Aware Pub/Sub

A Design of the Hybrid Architecture utilizing Distributed and Centralized Multicast

Toyokazu Akiyama,
Yukiko Kawai
Kyoto Sangyo University
Kyoto, Japan
akiyama@cc.kyoto-su.ac.jp,
kawai@cc.kyoto-su.ac.jp

Yuuichi Teranishi
NICT /
Osaka University
Tokyo, Japan /
Osaka, Japan
teranisi@nict.go.jp

Ryohei Banno
Tokyo Institute of
Technology
Tokyo, Japan
banno.r.aa@m.titech.ac.jp

Katsuyoshi Iida
Tokyo Institute of
Technology
Tokyo, Japan
iida@gsic.titech.ac.jp

Abstract— **Pub/Sub communication model becomes a basis of various applications, e.g. IoT/M2M, SNS. These application domains require new properties of the Pub/Sub infrastructure, for example, supporting a large number of devices with widely distributed manner, handling emergency messaging with priority control and so on. In order to meet the demands, we proposed Software Defined Network Aware Pub/Sub (SAPS) which utilize the both Application Layer Multicast (ALM) and SDN, especially OpenFlow based multicast (OFM). A simulation was done for evaluating the hybrid architecture in traffic and transmission delay reduction, and then the issues to be solved in the current design were discussed.**

Keywords- Topic-based Pub/Sub; Application Layer Multicast; OpenFlow Multicast; Software Defined Networking; Introduction

I. INTRODUCTION

Publish/Subscribe (Pub/Sub) communication model becomes a basis of various applications, for example, IoT/M2M, SNS and so on. Recently, several new properties are required for the Pub/Sub infrastructure. The existing Pub/Sub infrastructures are basically constructed locally in ISPs or Data Centers. However, emerging applications like "Smart City" requires interaction among huge amounts of widely distributed sensors and actuators [1]. It requires Pub/Sub infrastructure to be more scalable from the both geographical and load aspects.

As one of the other Pub/Sub applications, SNS also has a new requirement. SNS already advanced as a daily messaging infrastructure. Several case studies are reported to express important roles of SNS in serious disasters [2]. It implies that social media is expected to be emergency information distribution infrastructure. It is already possible to give a priority to emergency information in SNS applications. However, it is still difficult for network layer devices to recognize which packets are related to the topics with priority. It requires SNS application developers to construct another mechanism which interacts with network layer. Such an application demand should be interpreted by Pub/Sub middleware.

In many Pub/Sub systems, publishers post messages to an intermediary broker and subscribers register subscriptions to the broker. Supporting geographical and load scalability requires multiple brokers to be deployed in a distributed manner. When the number of broker increases, the *multicast* becomes desirable for inter-broker communications instead of the broadcast. Application Layer Multicast (ALM) plays an active part for inter-domain services on behalf of IP multicast [3]. In the same time, several new multicast methods are proposed [4][5]. They are implemented with Software Defined Networking (SDN) technology and SDN provides application programming interface to enable more interactive control with higher layer mechanisms. It can be a candidate solution for Pub/Sub layer to interpret application layer demands. In order to satisfy the requirements of emerging applications, those technologies must be cooperated each other.

In this paper, we propose a new Pub/Sub infrastructure which uses application layer multicast in conjunction with lower layer multicast for establishing geographical and load scalability as well as recognition of application layer demands in a lower layer. The rest of paper is organized as follows. In section II, we will discuss related work to describe the target issue of this paper and then show you the details of the proposed Pub/Sub infrastructure in section III. In section IV, we will show evaluation results and discuss the issues in the current design of the proposed hybrid architecture. Lastly, we will conclude this paper in section V.

II. BACKGROUND

In this section, we will discuss related work to clarify the target issues to be solved by our proposal.

A. Application Layer Multicast (ALM)

The issues in wide area deployment of IP multicast had already been recognized in 1990s and ALM development had been actively advanced [3]. In ALM, application layer end-systems exchange messages via overlay network constructed among them. The most important characteristic of ALM is that it does not require any new interaction among the operators of under layer infrastructures. It is easy to add new nodes to deploy ALM as a wide area service. Furthermore, ALM based on the structured overlay approach also has an autonomic resilience property in a fault situation because it does not require any centralized server. On the other hand, since ALM constructs overlay network independently of lower layer structure, unnecessary traffic generated by the overlay links go back and forth on the same physical links becomes a problem.

In order to solve the problem, overlay optimization methods are proposed [6][7]. The issues in ISPs related to the traffic increase by overlay network applications can be suppressed to a certain degree. However, since it does not provide an interaction function with network layer, the application developers are still required to construct some interaction mechanism to control message priority.

B. OpenFlow Multicast (OFM)

Recently, SDN technology including OpenFlow is the focus of network engineering field attention. It is possible to realize flexible interaction between application layer and lower layer by controlling lower layer function with modern style programming languages. Several methods have been proposed to establish flexible control over multicast network using OpenFlow functions [4][5]. While they are the same as the IP multicast in the point that they realize multicast by lower layer functions, the difference is the centralized logic for targeting more flexible control.

Furthermore, there is a new proposal of Pub/Sub architecture based on OpenFlow network, named PLEROMA [8]. It has a possibility to solve the priority control problem assumed in this paper. However, it still has a scalability problem to support full functionality of the Pub/Sub system. OpenFlow Switch (OFS) looks up flow entries recorded at flow tables and forward packets based on the rules described in flow entries. Since flow entries must be processed in a short period, look up and processing logics are usually implemented using high cost devices like TCAM and thus the number of flow entries has an upper limit. While the number increased by technology progress year by year, it still has a limitation about 100 thousand entries per OFS even in the commercial products. Here, we consider subscribing “send a message” event in SNS. In this case, for example, the number of accounts can be considered as an event in Twitter or Facebook. It is reported as 271 million accounts in Twitter [9] and 1,281 million in Facebook [10]. It cannot be simply covered by the current OFS implementations. To establish Pub/Sub infrastructure based on SDN technology, further investigations must be required.

As one of the other approaches, there is a proposal of new generation Pub/Sub infrastructure based on the totally reconstructed lower layer functions [11]. It has a possibility to achieve even higher performance than OpenFlow based approaches. However, as discussed in the next section, it has a bigger problem than OpenFlow in the migration from the current network. In this paper, we focus on the OpenFlow as the more practical approach.

C. Hybrid mode in OpenFlow

As a matter of course, the upper limit problem of flow entries is recognized in OpenFlow community. The solution to take advantage of OpenFlow in conjunction with the limitation is one of the important issues. Wackerly [12] proposes a method to save the number of flow entries by utilizing “Hybrid mode”. Hybrid mode uses Pipeline Processing function standardized as OpenFlow switch specification [13]. If we specify NORMAL action in a flow entry, the conventional lower layer functions like IP or Ethernet, which are implemented in the OFS, forward packets instead of the flow base forwarding functions. It means that we can use flow entries only for more advanced and flexible requirements which are not supported by the conventional technologies. As a result, we can save the number of flow entries. The most of production OFS supports Hybrid mode, and it is expected as a way to take advantage of OpenFlow.

When we assume to use the Hybrid mode, the target network will be a mixed network to support seamless migration from the conventional network. If we consider applying OFM to such a mixed environment, the same problem existing in the traditional IP multicast will occur again [3]. When we construct wide area OFM network, it spreads over multiple OpenFlow administrative domain. Furthermore, if OpenFlow networks are connected via conventional IP network, tunneling function is required for OFM. When OFSs become widespread and all of the networks are OpenFlow ready, the tunneling function becomes unnecessary. However, the problem caused by multiple administrative domains still exists.

Here, since the conventional IP network is constructed as distributed manner, even if it is divided into multiple portions by a fault, it can still tolerate the situation. However, if we implement multicast only by the current OpenFlow functions, we cannot fully exploit the fault tolerance. For example, as Tootoonchian [14] proposed, deploying multiple OFC is an approach to support fault tolerance in OpenFlow. However, it is less tolerable than IP network unless all the OFS have their own controller. It requires further researches. Therefore, we currently focus on the fault tolerance exploiting OpenFlow-hybrid functions. Akiyama et al. [15] proposed a new Pub/Sub system concept which exploits the both merits of conventional IP network and OpenFlow. In this paper, we describe a design of a new Pub/Sub architecture based on the concept.

III. SDN AWARE PUB/SUB

In this paper, we describe a design of SDN Aware Pub/Sub (SAPS) which is based on the concept proposed in [15]. The overview of SAPS is shown in Figure 1. The “topic-based Pub/Sub” is adopted in SAPS because it is adopted by major Pub/Sub products. Brokers mediate publishers and subscribers and messages are delivered by unicast or multicast among brokers. SAPS basically utilizes ALM for the multicast and if the condition meets, it also utilizes OFM

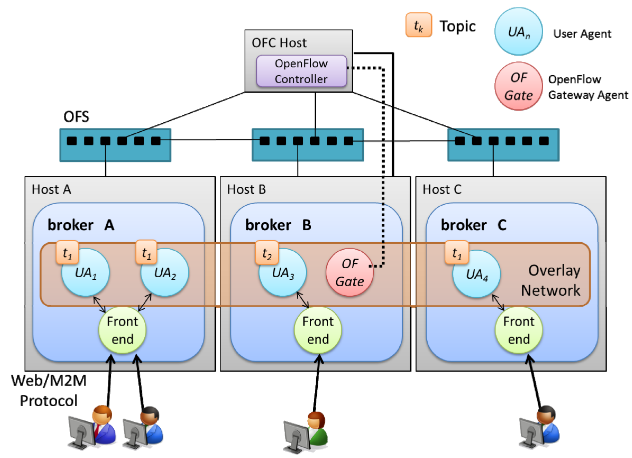

Figure 1. Overview of SDN Aware Pub/Sub (SAPS).

as described later. In order to assure the message arrival among brokers, we assume that structured overlays like DHT [16] and SkipGraph [17] are used for ALM. The multicast in overlay network is realized by registering topic t_k as a key of overlay network and then discovering the topic t_k to deliver a message.

In order to reduce multicast traffic, not only ALM but also OFM is exploited in SAPS if OFM is ready for the backbone network. To realize the interaction between ALM and OFM, application layer entities must communicate with OpenFlow Controller (OFC). A special agent *OFGate* is prepared for the purpose. *User Agents* (*UAs*) are also generated on the brokers. A *UA* is mapped to a publisher or a subscriber connected via Frontend using Web or M2M protocols. From now on, since we mainly discusses about inter-broker communications, we call the *UA* on a broker as "publisher" or "subscriber", which is mapped to the real publisher or subscriber.

The same as topics, an agent name can be registered as a key of overlay network. As a result, any agent can interact with the other agent by specifying the agent name using unicast communication. For example, publisher or subscriber can communicate with *OFGate* if required. Publishers can also receive responses from subscribers, if needed, by specifying in the message. The response is assumed to be aggregatable by tracing the multicast path in the reverse direction. Publishers collect monitoring information by using that function when they publish a message.

In the rest of this section, we will discuss how to exploit OFM in addition to ALM. In SAPS, a message published to a new topic is delivered via ALM, and then by referring the monitoring information, it is decided to exploit OFM or not. The decision is made by considering the number of delivered messages, the size of messages, delivery frequency, a priority value set to the messages and so on. Once the exploitation of OFM is decided, an OFM mapped from the ALM of the specified topic is constructed in an OpenFlow network.

In order to provide switching function between ALM and OFM, we need to consider the API of SAPS. In section III.A, we describe the design of SAPS APIs, the one is between *OFGate* and OFC and the other is between SAPS and the end system. Furthermore, when we exploit the both ALM and OFM, we need to synchronize the membership management of the both multicast network. The details are discussed in section III.B.

A. SAPS API

To realize the interaction between *OFGate* and OFC, we need to design an API of OFC, called Northbound API. And we also need Pub/Sub API as the same as the conventional Pub/Sub, e.g. publish/subscribe, which must be transparent to the switching between ALM and OFM. In section III.A.1), a design of Northbound API is described and then in section III.A.2), the Pub/Sub API for SAPS is described.

1) Northbound API

Figure 2 shows the mapping between ALM and OFM during the OFM construction. The top of the Figure 2. shows the status of ALM. A subscriber is registered and managed as a member of a topic group by registering a topic key to the overlay, which we call "subscribe" in ALM. In Figure 2, subscriber S_1, S_2 and S_3 are the member of the topic group $S(t_1)$ which is a group of subscribers interested in the topic t_1. During the subscription, publishers cannot detect the member changes because the overlay network is updated locally around the new subscriber. As described later, when OFC creates a multicast tree related to the topic t_k, it requires the list of the brokers hosting the subscribers in the topic group $S(t_k)$. An element of *broker list* is described as a tuple <IP address, MAC address, port number>. How to collect *broker list* at *OFGate* is described in section III.B. A publisher makes the decision of OFM construction based on the monitoring information and sends a request to *OFGate*. After receiving the request, *OFGate* asks OFC to construct OFM via REST API. The OFC assigns *OFM address* to the *broker list* and returns it to *OFGate*. *OFM address* is composed of IP address and port number like "<IP address>:<port number>", and its range must be preregistered to the system. This approach has a merit of flexibility compared to the use of the standardized multicast address. After finishing the construction of OFM, the publisher publishes a message to the subscribers by sending packets to *OFM address*.

OFC construct multicast tree for packet delivery from a publisher to subscribers by using *broker list*. There are mainly two approaches to construct OFM tree, one is to construct source specific tree and the other is to construct shared tree [18]. While the source specific tree approach must be adopted if the performance is the principal factor, it requires to construct multicast tree for each publisher. When the number of publisher becomes large, the flow entry limitation must be a problem as described in section II.C. In

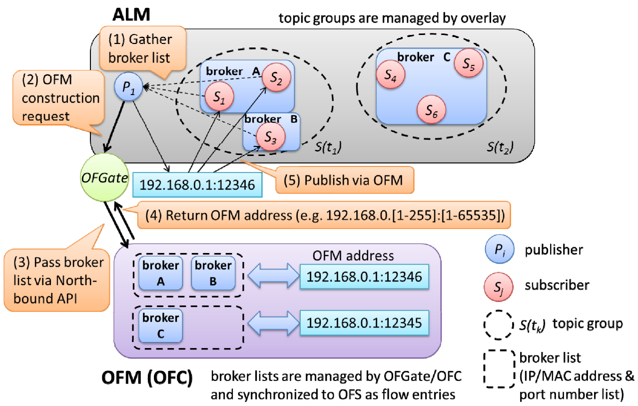

Figure 2. Overview of ALM/OFM mapping.

this paper, since we assume IoT and SNS applications where an arbitrary end entity becomes a publisher, we need to consider the shared tree approach. In order to support publish from subscribers, bidirectional tree is constructed among subscribers.

2) *Pub/Sub API*

SAPS provides basic interface, e.g. publish/subscribe, like general Pub/Sub system. While the API itself is the same as usual, the behavior inside the broker changes for each multicast method. For example, when a publisher publishes a message to ALM, it is confirmed by transport layer function, like TCP ACK. However, when the publisher uses OFM, it basically uses UDP and the publisher cannot confirm the packets are properly sent via OFM or not. In order to enable publishers to confirm the existence of OFM, the packets sent by publishers are sent back to themselves by adding flow entries for the loopback. Since in many cases, the number of subscribers is larger than that of publishers, it is desirable that subscribers can receive messages regardless of the multicast method selected by the publisher, ALM or OFM. Therefore, the broker where subscribers exist listens on the both ALM port and OFM port, and then the broker delivers the received message to the subscribers by topic name written in the message. Since the broker manages mapping between subscribers and topics, the only one listen address and port is enough for ALM and OFM respectively.

In addition to constructing multicast tree, the OFC must generate the flow entries which rewrite the destination address of the packets from *OFM address* to the listen address and port of the broker with subscribers.

In the rest of this paper, "topic" string is often used to manage ALM and OFM. However, the descriptions added by SAPS are not necessary to be shown to the end users. The end users just express the interested topics as a string. The additional descriptions are automatically added or deleted in SAPS.

B. *Membership Management and Delivery Method*

As described in III.A.1), in order to construct OFM including ALM subscribers, it is necessary to collect *broker list*, and it must be notified to the OFC. However, if a subscriber joins or leaves after the collection and before the completion of OFM preparation, an inconsistency is generated between the *broker list* on OFC and topic group on ALM. As a result, messages cannot be delivered to some of the subscribers. Such a message loss must be avoided. Furthermore, we have choices in multicast delivery methods, the one is utilizing either ALM or OFM (EITHER) and the other is utilizing the both ALM and OFM (BOTH). While *broker list* management method depends on the delivery method, on account of limited space, we describe only the former approach (EITHER) in this paper. In EITHER, (1) subscriber notifies subscription to *OFGate* after joining ALM and (2) publisher asks *OFGate* about the OFM construction status before publish. When a publisher decides to exploit OFM based on the monitoring information, it asks *OFGate* to construct OFM. In EITHER, since subscriptions are notified to *OFGate*, *broker list* already exists at *OFGate* and the process (1) in Figure 2 is not required.

If a publisher asks *OFGate* about the OFM status during OFM construction, *OFGate* returns the status of "under construction". In this case, the publisher uses ALM. When the construction finished, *OFGate* returns the status of "constructed" and the publisher uses OFM. When a subscriber joins or leaves, the update request is sent to *OFGate*. During the update, *OFGate* returns "updating" and the publisher uses ALM. If the update request arrives during OFM construction, the exploitation of OFM is delayed until the update completion. When the all of the subscribers have been left from a topic, the related OFM is deleted. During the deletion, the status is managed in the same way. Publishers can start exploiting OFM just after the completion of construction because it confirms the status before publishing.

The merit of this method is that it is simple to implement. On the other hand, the demerits are (1) *OFGate* must deal with heavy query load, (2) Since the broker list is synchronized to topic groups, once the OFM is constructed, all of the updates must be reflected to the OFM.

IV. EVALUATION AND CONSIDERATION

We describe a simulation result of traffic and message transmission delay of ALM compared with OFM in section IV.A and discuss the issues remaining in the current design including exploitation approaches of SAPS architecture in section IV.B.

A. *PERFORMANCE EVALUATION*

In this section, we show a simulation result. In the simulation, we define the "cluster" as a set of the broker hosts mutually connected by switches with enough bandwidth and then connect clusters each other by switches as shown in Figure 3. We evaluated the reduction performance of inter-cluster traffic and message transmission delay. The parameters are the number of clusters (8, 16), the topology (line, tree) and the number of subscribers. The practical data center network topology is described in [19] and it has intermediate characteristic of line and tree topology. So, as a first step, we examine those two patterns. As an overlay network for ALM, SkipGraph which has relay-free characteristics is used [20]. As described in section II.A, ALM forwards messages without considering the lower layer topology, in some cases, messages go back and forth on the same physical link. However, if it is possible to obtain the cluster ID where the broker is belonging to and topics can be sorted by cluster ID, e.g. t_1*@clusterA*, t_1*@clusterB*, unnecessary inter-cluster communications can be reduced because SkipGraph creates logical links based on the sorted order of registered keys. In this case, cluster ID helps to create logical links between the brokers located at the same or neighboring cluster. As a result, unnecessary jumps between distant clusters are reduced. In the rest of paper, we call the method using cluster ID as ALM with Locality Awareness (ALM-LA) and add as a target of comparison. The evaluation results are shown in Figure.4, 5. Figure.4 shows results of line topology with 16 clusters. The left side shows the inter-cluster traffic per publish and the right side shows the maximum delay (cluster hops) per publish. In the both results, OFM shows better performance than ALM. When the cluster ID is numbered randomly, ALM-LA performance obviously degrades. Especially in the maximum delay, it is almost the same as ALM. If a pair of clusters is randomly chosen from line topology, the hops between the two clusters become extremely larger than the sorted case. That causes performance degradation. On the other hand, the physically sorted result shows better performance. From those results, it is said that if an identifier which is assigned regardless of physical order, e.g. IP address, is used as a sort key of ALM-LA in line topology, OFM efficiently improve the performance. Figure 5 shows results of height 2 tree topology with 8 and 16 clusters. Since inter-cluster distance is always 2 hops in the tree topology, the all methods show slightly better performance than in the line topology. As the number of cluster becomes larger, the performance of the all methods slightly degrades because the possibility that publishers and subscribers are located at the same cluster becomes lower.

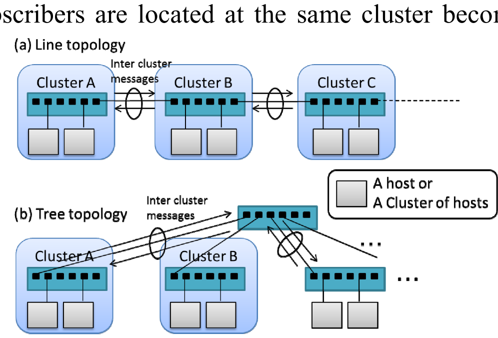

Figure 3. Simulation topology.

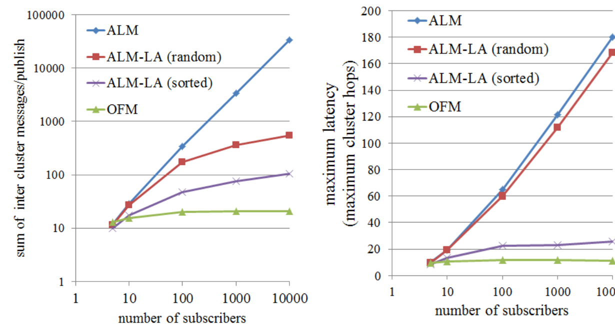

Figure 4. Line topology (16 clusters) results.

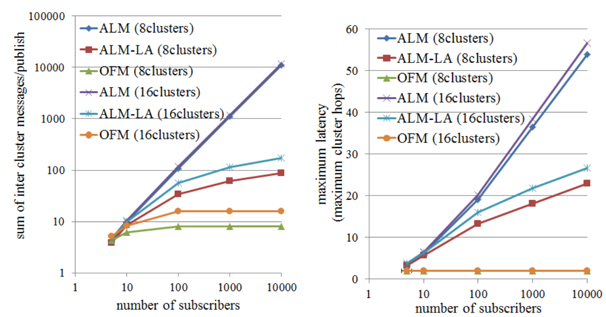

Figure 5. Tree topology (8, 16 clusters) results.

B. *CONSIDERATION OF CURRENT DESIGN*

In this section, we will discuss the issues in the current design of SAPS.

1) *Scalability*

As discussed in section III.B, membership management and delivery method has scalability problems. An approach to exploit EITHER method is to cache OFM address locally at the publisher. It can reduce the load of *OFGate*. However, if publisher uses cached OFM address during OFM updates, it may cause message loss. Join and leave frequency should be considered to set cache expiration.

From the results shown in section VI.A, OFM has an efficiency to reduce traffic and delay with only 100 subscribers. For example, in IoT applications, we can set topics for each room or building where target sensors are located. Limiting the number of subscribers and related area enable us to extract stable and closely connected sub groups. The problem of EITHER method can be reduced in such a situation.

The destination address rewriting in SAPS discussed III.A.2) provides flexibility for address usage. However, currently, the address rewriting in OFS is known to have performance issue. It should also be evaluated in the prototype system.

2) *Security*

As described in Figure 1. SAPS can interact with end user/system via Web or M2M protocols. To establish access control with end entities, authentication and authorization technologies in the Web or M2M fields can be applicable. It is also possible to control backend communication by using PKI technologies. Not only unicast but also multicast can be protected by using the secure group communication protocols [21]. However, it is difficult to defend DoS attack to the multicast address. In order to establish more secure environment, network space separation of Pub/Sub infrastructure as proposed in [22] must be required.

3) Multi-domain and Split-domain

The method proposed in this paper currently support only single administrative domain. Hybrid use of ALM and OFM which spreads over multiple domains including domain splitting as described in Figure 1. is a part of future work. However, we already provide materials to consider the extensions for multiple domain awareness. For example, as discussed in section IV.A, cluster ID can be used to map OpenFlow administrative domain and overlay network, e.g. topic1@domain1, topic1@domain2. Once the domain is mapped to the overlay network, subscribers can join the topic with local domain. At the same time, *OFGate* can also be prepared for each domain. It enables us to switch OFM utilizing local domain function. Furthermore, it is possible to elect a representative subscriber for each domain and the representative subscriber forwards messages to OFM which were received from the other domains. The detail design and implementation of multi-domain function is future work.

V. CONCLUSION

In this paper, we have picked up several new demands for Pub/Sub infrastructure and in order to meet the demands, we designed the basic functions of SDN Aware Pub/Sub (SAPS). In addition to ALM, SAPS exploits OFM to reduce traffic and message transmission delay for inter-broker communication. A simulation was done for evaluating the hybrid architecture in traffic and transmission delay reduction, and then the issues to be solved in the current design were discussed. We have already implemented a prototype system. However, its evaluation has not finished yet. As a part of future work, we will evaluate total system performance including the investigation of ALM maintenance cost, ALM and OFM switching cost, churn resiliency and so on.

ACKNOWLEDGMENT

This study is partially supported by Strategic Information and Communications R&D Promotion Programme (SCOPE) and National Institute of Informatics's joint research project. We would also like to express our sincere thanks to the late Mr. Mikio Yoshida for giving us advanced and discerning advices to design the proposed architecture.

REFERENCES

[1] J. M. Hernández-Muñoz, et al. "Smart cities at the forefront of the future internet," Springer Berlin Heidelberg, 2011.

[2] B. R. Lindsay, "Social media and disasters: Current uses, future options, and policy considerations," Congressional Research Service, vol. 41987, 2011.

[3] M. Hosseini, D.T. Ahmed, S. Shirmohammadi, N.D. Georganas, "A survey of application-layer multicast protocols," Communications Surveys & Tutorials, IEEE, vol.9, no.3, pp.58,74, 2007.

[4] D. Kotani, K. Suzuki, and H. Shimonishi, "A design and implementation of OpenFlow controller handling IP multicast with fast tree switching," Proc. of IEEE/IPSJ 12th International Symposium on Applications and the Internet (SAINT), 2012.

[5] C. AC Marcondes, et al. "CastFlow: Clean-slate multicast approach using in-advance path processing in programmable networks," Proc. of IEEE Symposium on Computers and Communications (ISCC), 2012.

[6] H. Xie, Y.R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: provider portal for applications," Proc. of ACM SIGCOMM 2008, pp. 351-362, Aug. 2008.

[7] J. Seedorf, S. Kiesel, and M. Stiemerling, "Traffic localization for P2P-applications: the ALTO approach," IEEE P2P 2009, pp. 171-177, Sep. 2009.

[8] M. A. Tariq, B. Koldehofe, S. Bhowmik, and K. Rothermel, "PLEROMA: a SDN-based high performance publish/subscribe middleware," Proc. of the 15th International Middleware Conference (Middleware '14), pp.217-228, Dec. 2014.

[9] Twitter Inc., "Who's on Twitter," https://biz.twitter.com/ja/whos-twitter (confirmed Jan. 2015)

[10] Facebook, Inc. , "Facebook Reports First Quarter 2014 Results," http://investor.fb.com/releasedetail.cfm?ReleaseID=842071 (confirmed Jan. 2015)

[11] N. Fotiou, et al. "Developing information networking further: From PSIRP to PURSUIT," Broadband Communications, Networks, and Systems, Springer Berlin Heidelberg, vol. 66, 2012.

[12] S. Wackerly, "OpenFlow Hybrid Mode", Open Daylight Developer Design Forum, Sep. 2014. https://wiki.opendaylight.org/images/1/1d/ODL_Hybrid_Mode.pdf (confirmed Jan. 2015)

[13] OPEN NETWORK FOUNDATION, "OpenFlow Switch Specification Version 1.3.3", Sep. 2013. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.3.pdf (confirmed Jan. 2015)

[14] A. Tootoonchian, and Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow," Proc. of internet network management conference on Research on enterprise networking (INM/WREN'10), USENIX Association, 2010.

[15] T. Akiyama, et al. "Proposal for a New Generation SDN-Aware Pub/Sub Environment." Proc. of The Thirteenth International Conference on Networks (ICN 2014), 2014.

[16] M. Castro, et al. "SCRIBE: A large-scale and decentralized application-level multicast infrastructure," Selected Areas in Communications, IEEE Journal on vol.20, no. 8, pp. 1489-1499, 2002.

[17] Y. Teranishi, "PIAX: Toward a framework for sensor overlay network," Proc. of 6th Consumer Communications and Networking Conference (CCNC 2009), 2009.

[18] S. Bhattacharyya, "An Overview of Source-Specific Multicast (SSM)," IETF RFC 3569 , Jul. 2003.

[19] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," ACM SIGCOMM Computer Communication Review, Vol. 38. No. 4. ACM, 2008.

[20] Ryohei Banno, et al. "Designing Overlay Networks for Handling Exhaust Data in a Distributed Topic-based Pub/Sub Architecture," Journal of Information Processing, Vol.23, No.2, pp.105-116, March 2015.

[21] M. Baugher, et al. "The Group Domain of Interpretation," IETF RFC 3547, Jul. 2003.

[22] S. Ishii, et al. "A study on designing OpenFlow controller RISE 3.0," Proc. of 19th IEEE International Conference on Networks (ICON), 2013.