Research Article

An Efficient SDN Multicast Architecture for Dynamic Industrial IoT Environments

Hyeong-Su Kim, Seongjin Yun, Hanjin Kim, Heonyeop Shin, and Won-Tae Kim

Smart CPS Lab, Department of Computer Science and Engineering, KOREATECH University, Cheonan, Republic of Korea

Correspondence should be addressed to Won-Tae Kim; wtkim@koreatech.ac.kr

Received 16 December 2017; Accepted 13 March 2018; Published 5 April 2018

Academic Editor: Jeongyeup Paek

Copyright © 2018 Hyeong-Su Kim et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Large-scale industrial IoT services appear in smart factory domains such as factory clouds which integrate distributed small factories into a large virtual factory with dynamic combination based on orders of consumers. A smart factory has so many industrial elements including various sensors/actuators, gateways, controllers, application servers, and IoT clouds. Since there are complex connections and relations, it is hard to handle them in point-to-point manner. In addition, many duplicated traffics are exchanged between them through the Internet. Multicast is believed as an effective many-to-many communication mechanism by establishing multicast trees between sources and receivers. There are, however, some issues for adopting multicast to large-scale industrial IoT services in terms of QoS. In this paper, we propose a novel software-defined network multicast based on group shared tree which includes near-receiver rendezvous point selection algorithm and group shared tree switching mechanism. As a result, the proposed multicast mechanism can reduce the packet loss by 90% compared to the legacy methods under severe congestion condition. GST switching method obtains to decreased packet delay effect, respectively, 2%, 20% better than the previously studied multicast and the legacy SDN multicast.

1. Introduction

Internet of Things (IoT) makes things exchange data with other physical/virtual things through the Internet. Most of IoT services monitor specific objects or place by means of suitable sensors, and make decisions based on the sensing data to optimize some situations, such as air conditioning for home. Many manufacturers try to collect the data from customers in order to accumulate user and device information generated by their products. The data made by things have the great importance in business value chain and future usage. Therefore, many components of IoT ecosystem need the raw data from sensors or processed data by some machines. That is to say, IoT application servers and IoT storage clouds may simultaneously require the data for analyzing the status of a thing and some other objectives.

In industrial IoT (IIoT), such as smart factory, so many sensors are more and more deployed in production lines in order to deeply watch into the factory environments and enhance the productivity as a result [1]. Comparing with

general IoT, IIoT requires much more dependable network services with low transfer delay, high reliability, and dynamic reconfigurability in terms of a network service because there are real-time cooperative machines with self-optimizing functions based on the machine or production process states detected by monitoring sensors [2–4]. Especially, factory cloud service or manufacturing as a service (MaaS) can be one of new manufacturing paradigms as shown in Figure 1 [5]. Manufacturing is evolving toward personalized production system by the order of user requirements. In factory cloud service, small production lines are integrated into a large-scale virtual factory where a subfactory sector should share the entire virtual factory information from the other subfactory sectors because the condition of other factories seriously affect the entire production schedule. All the production activities including automatic logistics are instantly scheduled by the order from users on line. This virtual factory concept can be also applied to a large factory system such as Samsung Electronics in the same way. Big data from each subfactory sectors are

FIGURE 1: Factory cloud service concept based on Industrial IoT.

multicast to the other sectors or computing systems, for example, IoT clouds with artificial intelligence (AI) or factory application servers including manufacturing execution system (MES), product lifecycle management system (PLM), enterprise resource planning system (ERP), supervisory control and data acquisition (SCADA), and so on. Therefore, dynamic network reconfiguration and high-level QoS will be strongly necessary to this kind of industrial services [6, 7].

As mentioned above, a large-scale IIoT faces the requirements of scalability and complex QoS [6, 7]. Actually, real-time factory control requires deterministic network operations supporting guaranteed time delay in which packets must be delivered from sensors to actuators via IIoT control systems in a few hundred milliseconds [8]. Reliability of packet delivery is also necessary to ensure very low packet loss for serious data such as control messages in terms of QoS [9, 10]. On the other hand, IP-based Internet has difficulties for supporting dynamic optimal path routing or network reconfiguration based on real network conditions because of its distributed routing characteristics. The future IIoT services need multiple network services including dynamic multipoint communications, quality of service (QoS)-based path rerouting and unpredictable IIoT device changes. Therefore, software-defined networking (SDN) is suitable for IIoT network services where it can efficiently perform the network services by means of global network view [11].

In this paper, we propose a novel SDN multicast based on group shared tree (GST) for large-scale IIoT environments. It can solve the critical issues including end-to-end delay time, packet loss, and network scalability. This paper is composed of as follows. In Section 2, we will summarize the SDN, IP multicast, and the legacy SDN multicast mechanisms. In Section 3, we describe the proposed SDN multicast

architecture. Also, our paper explains near-receiver dynamic rendezvous point (RP) selection algorithm, a state machine of GST operation, basic GST flow entries, and GST switching mechanism. Section 4 proves the performance of the proposed architecture in terms of packet loss and end-to-end delay time. Finally, the conclusion and future works of SDN multicast will be given.

2. Related Works

2.1. Software-Defined Networking. Applications with various characteristics have different network requirements [12]. The legacy network has limitation to meet these requirements [13]. SDN was proposed to make networks flexible to overcome the limitation of a legacy network [14]. SDN is separated by three layers, which are data layer, control layer, and application layer. The lowest layer is the infrastructure layer, which includes network devices such as switches. This layer is also called “Data Plane.” Network elements are forwarded through the infrastructure layer, and the status information of each local network device is also monitored. The above layer is the control layer. It is also called “Control Plane.” The control layer consists of SDN controllers. A SDN controller is an essential component which provides functions of global network monitoring, network service definition, and configuration.

The control layer receives and uses every local network information from the infrastructure layer through the southbound interface. The openflow protocol is the dominant protocol for the southbound interface [15]. Based on the local network information, the SDN controller monitors the network status and performs network control such as flow and path configuration. The commonly used SDN controllers are

OpenDaylight, Floodlight, and Ryu controllers [16, 17]. The application layer above the control layer has network applications to conduct various network functions which have various network features. The interface between the application layer and the control layer is called northbound interface. Through the northbound interface, the application layer receives the global view of the network and sends the guideline for defining network service. In the northbound interface, RESTful API is mainly used for exchanging network resource information. Through these three layers of SDN, the network can be flexibly controlled to satisfy various communication requirements of each network application.

2.2. IP Multicast. The legacy multicast mechanisms can be categorized into two types: source based tree (SBT) and GST [18, 19]. SBT multicast mechanism is a multicast method of constructing a multicast tree for each source as a tree root, and there are multicast OSPF (MOSPF), distance vector multicast routing protocol (DVMRP), and protocol independent multicast-dense mode (PIM-DM) [20, 21]. GST multicast mechanism is a multicast method of making a single shared multicast tree for all the sources in a multicast session, and there are core based tree (CBT) and protocol independent multicast-sparse mode (PIM-SM) [22]. GST creates a single tree path structure consisting of a group of senders and receivers. GST has a core node or an RP which works as a central point of multicast packet transmission for all the sources and the receivers in a multicast group. The SBT and GST have each different required resources of the switch. The SBT does not consider created a number of flow by receiver for each source, so a state explosion is happened caused by a large number of multicast sources, such as sensors and sensor gateways [23]. SBT does not have any multicast core which multicast traffic should be passed, so the SBT can communicate in real time. However, SBT multicast mechanism does not guarantee real time in all cases. Creating a multicast tree does not consider receiver for each source, so a state explosion is happened caused by a large number of multicast sources, such as sensors and sensor gateways [23]. On the contrary, the GST only needs a resource by the number of the multicast group. The multicast traffic of GST must pass through the RP when the source transmits a packet to all receivers. The weak points of GST are, however, RP overload and nonguaranteed real-time communication. Although GST multicast reduces the number of multicast flows, it must ensure real-time service and solve RP overload problems in order to achieve a critical QoS of the IIoT service [24, 25]. Therefore, GST applied to IIoT should reduce packet loss by RP overload and minimize other packet loss reasons.

2.3. SDN Multicast. As mentioned above, the multicast mechanisms such as GST and SBT are difficult to construct an adaptive multicast tree. Some research has been performed on setting the path by applying the SDN to the legacy multicast mechanism [26]. Tim suggested the SDN multicast SBT, early branching shortest path tree (EBSPT) [27]. EBSPT find branches, which are split by paths between sources and

receivers. The multicast packet sends to branches through the IP tunneling. The packet that arrives at the branch converts to a unicast packet and goes to the receivers. Tim proved the network scalability of EBSPT by using an installed number of flow states and transmitted multicast packet count as evaluation metrics of an experiment. Lin and Cui proposed an optimal path by applying SDN to the GST scheme [28, 29]. Locality-aware multicast approach (LAMA) proposed by Lin separates multicast groups by bandwidth threshold of sources when multicast groups form [28]. The next process selects an RP by the smallest hop counts between sources and candidate RP switches. Our proposed RP selection method chooses an RP that has short distance among candidate RP switches and receivers unlike the RP selection method by Lin. The proposed multicast method by Lin remains difficult in resolving path switch and adaptive RP selection in dynamically changing networks. Lin predicted to degrade the switch performance due to a large number of flows per switch in network with multiple sources. A large number of sources leads RP overload, and it causes packet loss. Cui implemented GST to different characteristics network, data center network (DCN) [29]. DCN has a few core switches which can be selected as an RP. Cui constructs shared trees to some core switches. Split multicast packets are sent in order to receivers through each shared tree. The proposed method by Cui is difficult to manage dynamic multicast groups, so it had limitation to apply in large-scale networks.

3. The Proposed SDN Multicast Architecture

3.1. The Architecture Overview. The proposed architecture of the SDN multicast for the large-scale IIoT environment is shown in Figure 2. In the network architecture data plane of the network is connected by using openflow switches [30]. The source sends multicast packet, and the receiver waits for receiving multicast traffic. To receive this packet, the receivers send Internet group management protocol (IGMP) report message to the switch which connected at first. The switch which received IGMP message sends Packet-In message to the controller so that controller knows the group join and the location of receiver. The receiver's group leave is performed in the same way. After that, the controller creates a tree structure from the RP to the receiver. The source sends the multicast packet to the first switch, and the switch sends the multicast packet to the controller. In other words, the GST is created by combining the path between the sources and the receivers through IGMP report message. Finally, when the receiver joins or leaves the group, it is caused for the RP location to move, and the sources send packets to new GST which is created by the GST switching.

3.2. The Basic Modules of SDN Controller. The control plane of the network is managed by an SDN controller. There are three basic modules including Topology Discovery, Flow Installation, and Traffic Monitor. The function of each module is as follows:

FIGURE 2: The proposed large-scale industrial IoT SDN multicast architecture.

- (i) **Topology Discovery:** This module can identify the status of the network configuration. This can obtain link status or location information of the near switch through link layer discovery protocol (LLDP), and based on this, the user can configure desired global view.
- (ii) **Flow Installation:** This module is responsible for installing flow entries in openflow switches from the controller. This module creates the appropriate shortest path using the topology discovery module by looking at a flow entry.
- (iii) **Traffic Monitor:** This module checks the flow status of openflow switches. It monitors the packet which is forwarded to the switch by checking the packet counter value in an flow stats reply message. Using this module, the user can observe periodically how much the GST affects the RP switch and other switches and compares the bandwidth of the old GST and the new GST.
- (ii) **RP Relocation:** This module works in three cases when the RP position of the multicast GST is changed, when the summation of bandwidth usage and all multicast packet size at the new RP is smaller than the bandwidth usage at the old RP, and when the delay of the new GST is smaller than the old GST. It updates the location information of the RP and the path information of GST of Group Manager.
- (iii) **GST Switching:** This module changes the position of the RP and creates a new GST. To reduce packet loss caused by GST switching, it helps to clearly control the path setup between an old GST and a new GST by using a temporary GST.
- (iv) **RP Selection:** This module uses the group manager and topology discovery to obtain receivers and a shortest path information, and then it selects the RP. If the RP cannot be selected, it selects the default switch as the static RP. This module supports packet forwarding and GST configuration with help of flow installation module.
- (v) **Tree Management:** This module performs GST establishment on the event of RP relocate and member join/leave of a group. GST information is generated based on RP location assigned by RP selection module. This has information of GST topology. The information is sent to group manager.
- (vi) **GST Delay Measure:** This module measures the delay of the old GST and the new GST. The controller sends a Packet-Out message to the RP switch and measures a return time which arrives packet to the controller. Using this module, the controller decides whether it connects to the new GST or not [31].

3.3. The Extended Modules for SDN Multicast. We designed six extended modules including Group Manager, RP Selection, RP relocation, GST Switching, Tree Management, and GST Delay Measure. The function of each module is as follows:

- (i) **Group Manager:** This module manages information of sources, receivers, and multiple groups for multicast communication. This module controls dynamic source and receiver by parsing IGMP group join/leave message and stores the information which is RP position and path of each group.

3.4. Near-Receiver Dynamic RP Selection Algorithm. We proposed the near-receiver dynamic selection algorithm that determines the RP location when a multicast group member changes. The RP is selected by the sum of the hop count between the candidate switch and the receivers of each group. The join/leave operation of sources does not affect RP selection. After selecting the RP, the flow installation module configures the shortest tree by combining the shortest path between the source and the RP and the shortest path between the RP and the receiver. If there are multiple switches with the same distance, the switch with the lowest data path id (DPID) is created as the RP. DPID is the serial number of the switch. As a result, the location of the RP may be changed to another SDN switch that is selected by the algorithm on the multicast session.

The pseudo code of Near-Receiver Dynamic RP selection algorithm is shown in Figure 3. N is a graph of the current network. S is a one-dimensional array of the RP candidate switches and has the DPID of the switch. R is a one-dimensional array of receivers belonging to a multicast group and has the value of the receiver of IP address. D is a one-dimensional array of the sum of the distance values between the switch and the receivers. S_l and R_l are the lengths of each RP candidate switch and receiver switch array. $ShortD$ is a shortest distance value in the D . RP variable is the index of the shortest distance value in D . *shortest_path_length* is a function to calculate the path distance between S and the R in N graph [32].

The algorithm process is as follows: first, call the *shortest_path_length* function to calculate the distance of the shortest path between R and S ; second, store the value which is calculated at first process to D ; return the index of D with the shortest distance in D , stored in RP; and lastly, select RP switch with matching RP and DPID of S .

3.5. State Machine Design of GST Management. Figure 4 shows the state machine of the proposed GST. The old GST defines a multicast tree before receivers invoked multicast member query. The new GST is a result of the RP relocation when the receiver invoked a multicast member query. The state machine has four states: Waiting GST, Compared GST, Temporary GST, and Changing GST.

- (i) **Waiting GST:** This is the waiting state before comparing the state change of the old GST. The waiting state does not change a state until multicast member sends query message. During the waiting GST must manage the join and leave of members. This member management is the work for modifying the flow of the GST. When the multicast member state changes, the GST gets permission to choose the RP. For the optimal changing GST, the RP position is obtained using the RP selection algorithm. After that, the controller sends a packet to measure the GST delay from the RP to the switches which first connects to the receivers.
- (ii) **Compared GST:** This state compares the old GST with new GST. If a location of the new RP and old

Algorithm 1 Near-Receiver Dynamic RP Selection Algorithm

```

Input:  $N$  : Network Graph
       $S$  - RP Candidate Switch Array
       $R$  - Receiver Switch Array
Output: RP - RP Switch DataPath ID
1: function NEARRECEIVERDYNAMICRP( $N, S, R$ )
2:   for  $i = 1$  to  $S_l$  do
3:     for  $j = 1$  to  $R_l$  do
4:        $D[i] \leftarrow \text{shortest\_path\_length}(N, S[i], R[j])$ 
5:     end for
6:   end for
7:    $ShortD \leftarrow \infty$ 
8:   for  $i = 1$  to  $S_l$  do
9:     if  $D[i] < ShortD$  then
10:       $RP \leftarrow i$ 
11:       $ShortD \leftarrow D[i]$ 
12:    end if
13:  end for
14:  return RP
15: end function

```

FIGURE 3: Pseudo code of the near-receiver dynamic RP selection algorithm.

RP is different, amount of the old GST delay minus the new GST delay is bigger than ϵ , and bandwidth of the old RP is greater than the sum of new RP bandwidth and size of the multicast packet, then the RP relocation is performed. ϵ is a threshold parameter of the total delay difference between the new GST and the old GST during the RP relocation. The RP relocation is the action of changing the RP location, which is the key node of the GST, by using the near-receiver dynamic RP selection algorithm. If the three above conditions are not satisfied, the current state must return to the waiting GST state and select other GST to measure the RP relocate. As long as the status of the receiver does not change, the GST that has undergone the comparison once does not enter the state machine again.

- (iii) **Temporary GST:** This state is executed to switch the tree of the new GST which satisfies the criteria of the previous state machine. The new temporary GST is created by RP relocation in the previous step. The new temporary GST coexists with the old GST. During this state, the flow of the old GST is deleted and connected to the temporary path, so the multicast packets flow to the new temporary GST.
- (iv) **Changing GST:** This state appears when the temporary new GST finish installing a path. The path moves the temporary new GST to the actual new GST. After that, it deletes the new temporary GST and returns to the waiting GST state. Finally, completing the action, the controller selects the other multicast groups to measure the RP relocate.

3.6. Multicast SDN Switch Operation. A flow entry is placed in a switch and controls a packet. There are two kinds of flow of multicast used in the proposed SDN multicast

FIGURE 4: The proposed GST Management State Machine.

architecture. It is divided before and after passing a packet to RP. Before passing through an RP, a flow is “(*, G),” and after passing an RP, a flow is made with a VLAN Tag. G means the address of a multicast group and * is every source IP address. These flows have three kinds of actions to process packets. The first action is forward action. This action is used most frequently for passing the packet to the destination. It is used in all flows. The second action happens when a packet reaches to an RP. It takes action to check a multicast group address of a packet and attaches header of VLAN. After arriving, this action matches a tag number of the VLAN to the group address of the multicast and attaches the mapped VLAN tag to a packet header. This action occurs only on an RP switch. The third action appears on switch which is connected most closest to the receiver. The switch has flow tables that can store flow entry, and the N th flow table is called as flow table N . The flow table 0 is applied to process incoming packets. The flow table 1 to N would be applied if the incoming packets need additional processing. One of the flow tables can send packets sequentially to other tables, but not vice versa. The first and second actions start at flow table 0, but the third action starts at flow table 1. The third action sends a packet from flow table 0 to the flow table 1. After that, the packet is matched with the VLAN header of flow entry under the flow table 1. If packet find a same VLAN header, the action removes the VLAN header and forwards to receivers.

3.7. RP Relocation and GST Switching Mechanism. The multicast path switching for changed GST must be done. Packet loss happens on the time between eliminating the old GST and configuring the new GST. The GST switching takes three steps to minimize forwarding paths on packet loss events. This mechanism must require the openflow

switches that support multiple flow tables. The first step is to configure the temporary tree path and change existing tree path to this path. The first step sets the temporary GST path based on the changed RP. A general GST has a path installed in flow table 0 or 1, but a temporary GST sets a path in the flow table 2 or 3. Therefore, the temporary GST and old GST will never overlap paths. The second step is to modify the flow of packets, which can forward into the temporary new GST and delete the path to the old GST. The last step copies the temporary new GST and pastes it in flow table with the value of 0 or 1. The controller sends a message to switches to delete the installed flow table with the value of 2 or 3.

4. Experiments and Evaluation

4.1. Experimental Configuration. There are three causes of packet loss in the proposed SDN multicast operation. The first one is an RP overload. We suggested changing the structure of GST and the algorithm for changing the location of an RP to solve this problem. The second reason is deleting the path due to the join/leave operation of receivers. We minimize the number of elimination of GST path without changing the location of an RP whenever receivers participated. The third is changing the GST by changing the location of an RP. Packet loss can be minimized by forwarding packets through a temporary GST before an actual GST path is created. We performed some experiments to verify if the proposed SDN multicast can efficiently reduce the effect of three types of packet loss. In addition, we measure the packet delay time of the proposed mechanism comparing with the legacy SDN multicast mechanisms.

The network simulation is performed in Mininet [33]. The SDN controller and switches are configured using the Ryu controller and openflow 1.3 version switch. We assumed that sources and receivers in the multicast session are

FIGURE 5: Mininet network topology for large-scale industrial IoT service.

sparingly distributed in wide-area. In addition, when comparing the number of sources and the number of receivers in multicast, we assumed that there are many sources rather than receivers which is a feature of the IIoT environment. The topology consists of a complex topology as shown in Figure 5. The topology is configured with 23 switches and 25 hosts. Each host has the different network IP address and the group of the source hosts is h10–h25, the group of the receivers is set to h1–h9. The SDN multicast packet used in the simulation was generated by Iperf [34].

The experimental environment for RP overload of the near-receiver dynamic RP selection algorithm is as follows. The total number of multicast groups is 15, consisting of an average of 5 sources and 3 receivers. The receivers send the corresponding multicast group join message. The sources send a multicast message to the receivers, where the other sources send the message after four seconds have elapsed since one source delivered the message. The multicast messages were transmitted for 200 seconds by 512 kbps bandwidth. The RP overload packet loss experiment is the same except that the packet transmission rate is 2 Mbps and the bandwidth of the entire topology link is 30 Mbps.

The previous work of Kim has a packet loss problem in a network environment where multicast groups often

participate in receivers [35]. In order to overcome packet loss, we conducted a comparative experiment with the proposed network overcoming packet loss. The simulation was set up with five sources in the same multicast group. These sources send multicast messages to receivers. The number of increments and decrements for the receiver has been set to 1 or 3. We observed the amount of lost packets during multicast communication.

The setting of the delay reduction experiment with GST change fixed the link delay of the entire network to 5 ms. In this experiment, four sources and three receivers joined the multicast group. We started the multicast transmission from the source and observed the time difference before and after applying the RP relocation. This experiment confirms that the delay of the GST is controlled by the RP relocation.

The environment setting for estimating the appropriate ϵ is set to 5 ms for the link delay. The members of the multicast group in the experiment are configured four sources and eight receivers. We tested the suitability of the GST switching by setting the GST delay difference to 100 ms, 50 ms, 10 ms, 1 ms, and 0.1 ms. The conversion conformance criterion of GST is the network overhead. Frequent GST switching increases the network overhead by the network control messages.

4.2. Analyzing Simulation Results. We compare the three multicast mechanism: LAMA, SDN multicast without GST switching, and SDN multicast with GST switching. LAMA by Lin is set based on the proximity of all sources. Once it is selected, Lin's RP does not change [28]. In our experiment, it is set on switch 7 that is the nearest switch to the sources. LAMA is the closest RP to the receivers. It also does not change the position of the RP like the SDN multicast without GST switching mechanism. This RP is located to switch 19 in our network topology. The SDN multicast without GST switching dynamically changes the RP location during a multicast session.

The comparison criteria are packet amount of RP switch, network congestion, and packet loss. First, we compare the number of packets that the switch concentrated on a RP about each mechanism, as shown in Figure 6. The RP switch is monitored at the interval of 60 seconds, and the packet transmission amount of the RP is observed for 260 seconds. LAMA and SDN multicast without GST switching receive 580,000 packets which is the total amount of experimental multicast packets. SDN multicast with GST switching does not change the total amount of packets received. However, due to the change of the position of each group RP, the total packets received by the RP were distributed and received an average of 90,000 packets. As a result, we observed that the proposed SDN multicast with GST switching mechanism can reduce load on a RP. The second comparison is network congestion. The network range of this comparison is from source to RP. The matrix for evaluating network congestion is the average number of flows that the switch has installed and the average packet forwarding rate of the switch. The formula for this matrix is shown in (1) and (2).

In (1), P_{Avg} is the result of an expression representing the number of packets forwarding per switch. P_{Total} is defined as the sum of forwarding packets of the switches included in the path from the source to the RP. S is the number of switches included in the path from the RP to the source. F_{Avg} in (2) represents the number of flow entries that make flow entries connecting the RP up to the path of the source and RP per switch. F_i is the number of at the source of the i th switch. S in (2) is the same as S as follows:

$$P_{avg} = \frac{P_{total}}{S}, \quad (1)$$

$$F_{avg} = \frac{\sum_{i=1}^S F_i}{S}. \quad (2)$$

As a result of (1) and (2), the network congestion is evaluated by the graphs of Figure 7. As shown in Figure 7, the SDN multicast with GST switching processes 29,000 less forward packets than LAMA and 8,000 fewer than SDN multicast without GST switching. In Figure 8, the proposed algorithm has shown that the number of flow states per switch is reduced by 36% rather than LAMA and by 30% rather than the SDN multicast without GST switching.

The reason for this result is that using LAMA increases the density of the switch. This causes an increase in the

FIGURE 6: Concentration of multicast packet on an RP switch.

FIGURE 7: The average number of multicast packets processed by an SDN switch.

FIGURE 8: The average number of flow states processed by an SDN switch.

number of flow states to be processed. As the role of the switch increases, the throughput of the packet forwarding also increases. The SDN multicast without GST switching is far from the source, so the throughput of the packet is reduced, but the RP and the switches around the RP still

FIGURE 9: The average multicast packet loss of all receivers.

cannot decrease the number of flow states. Due to the fixed number of flow states, the performance of SDN multicast without GST switching cannot be advanced. However, the SDN multicast with GST switching does not concentrate on low density and the RP switch packet. It has the lowest congestion of the network with the formula of (1) and (2). As a result, the SDN multicast with GST switching is the most efficient multicast mechanism in the large-scale IIoT environment with a large number of sources.

The last comparison is the result of the packet loss in the second experimental environment. Packet loss occurs when the amount of packets sent from the source to the receiver over the RP exceeds the 30Mbps capacity. In Figure 9, packet loss LAMA, SDN multicast without GST switching, and SDN multicast with GST switching are 63%, 40%, and 35%, respectively. The number of packets to be processed by the link increases when the congestion of the network is high and the forwarding processing of the concentrated packets of the RP is large. This causes link to discard the packet. Therefore, the packet loss is small in the order of SDN multicast with GST switching, SDN multicast without GST switching, and LAMA.

With these three comparisons in case of many sources and many multicast groups, the SDN multicast with GST switching overcomes single point of failure (SPOF) problem caused by RP overload, which is the essential disadvantage of GST multicast. In addition, when using our proposed algorithm, it is possible to use IIoT network more efficiently by reducing the use of ternary content-addressable memory (TCAM) [36], which is a flow memory of a switch.

Figure 10 shows the packet loss result where group join and leave operations frequently occur. In these experiments, we compared the extended SDN multicast preventing packet loss caused by GST Switching with the previous work of Kim which was not considered packet loss [34]. We performed this packet loss experiments four times. Each of the experiments increases and decreases the number of receivers in the multicast group by one or three. The extended SDN multicast average packet loss is 1%, while the multicast without GST switching has an average of 10% [34]. The extended SDN multicast can reduce packet loss by applying RP relocation and SDN multicast GST switching module. The experimental result shows that the proposed multicast

FIGURE 10: The amount of packet loss on GST switching.

FIGURE 11: The average packet delay comparison.

FIGURE 12: Network overhead by the factor of ϵ .

mechanism can reduce the packet loss, which is one of the important factors in IIoT.

Figure 11 shows the average packet delay comparison of the three SDN multicast mechanisms. The rectangle dotted line means the packet delay time of LAMA. The circular dotted line is the SDN multicast without GST switching of our previous study, and the proposed SDN

multicast is represented as the diamond dotted line. The proposed SDN architecture showed 20% average delay reduction than LAMA and 2% decreased delay compared with SDN multicast without GST switching. Since the proposed SDN multicast has GST switching mechanism based on the difference between the average delay time of old GST and that of new GST, the average delay comparison shows that the proposed mechanism has the lowest delay level as depicted in Figure 11.

The network overhead depends on the control messages generated by GST switching. We design the frequency of GST switching by means of value ϵ , which means the delay difference between old GST and new GST. If network operators set value of ϵ to small level, there will be very frequent GST switching. As shown in Figure 12, as the ϵ value increases from 0.1 ms to 100 ms, the network overhead decreases and vice versa. The decision of the value of ϵ may be a difficult problem because the ϵ value is an engineering factor. The value should be decided by considering network topology, resource, and other conditions of network.

5. Conclusion

In this paper, we proposed a novel SDN multicast with GST switching for large-scale IIoT networks. The legacy IP GST-based multicast has packet loss problems caused by the overload on RP. In addition, it is difficult to reliably transmit multicast packets under congestion conditions and to configure the optimal multicast path for dynamic join/leave operations. In order to overcome the above issues, we adopted SDN-based multicast architecture including tree switching for discovering optimal multicast paths between sources and receivers. Especially, the architecture includes the near-receiver dynamic RP selection algorithm and GST switching. In the near-receiver dynamic RP selection algorithm, we suggested RP relocation method for the overload on a RP before making a new GST. A new RP location is selected by the calculation of the minimum distance between the old RP and all multicast receivers. With RP relocating process, GST switching helps to enhance the performance of delay time. A SDN controller measures the delay difference between the old GST and the new GST, and it makes GST switching decision to achieve lower packet delay. Finally, we compared the performance of the proposed SDN multicast with that of the legacy SDN multicast in terms of packet concentration on RP, average packet loss, and average delay time. As a result, we could prove that our SDN multicast with GST switching has lower packet loss ratio and delay time than the other SDN multicast mechanisms.

Disclosure

This paper is an extended work of the paper published in 2017 8th International Conference on ICT Convergence (ICTC).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study is partially supported by the Graduate School Research Program of KOREATECH (No. 2017-0295) and by the National Research Foundation of Korea (NRF) grant (No. 2017010875) from the Korea government (MSIT).

References

- [1] T. Stock and G. Seliger, "Opportunities of sustainable manufacturing in industry 4.0," *Procedia CIRP*, vol. 40, pp. 536–541, 2016.
- [2] G. Peralta, M. I. Urkia, M. Barcelo, R. Gomez, A. Moran, and J. Bilbao, "Fog computing based efficient IoT scheme for the industry 4.0," in *Proceedings of the 2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM)*, pp. 1–6, San Sebastian, Spain, May 2017.
- [3] H. P. Breivold and K. Sandström, "Internet of Things for Industrial Automation—Challenges and Technical Solutions," *The 2015 IEEE International Conference on Data Science and Data Intensive Systems*, pp. 532–539, Sydney, Australia, 2015.
- [4] H. Derhamy, J. Eliasson, J. Delsing, and P. Priller, "A survey of commercial frameworks for the internet of things," in *Proceedings of the IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, pp. 1–8, Luxembourg, September 2015.
- [5] J. Kozłowska, "Product-service systems in a manufacturing company strategy—a review paper," *Economics and Management*, vol. 7, no. 2, pp. 48–56, 2015.
- [6] X. D. Li, W. He, and S. Li, "Internet of things in industries: a survey," *IEEE Transactions on Industrial Informatics*, vol. 10, pp. 2233–2243, 2014.
- [7] H. P. Breivold and K. Sandström, "Internet of things for industrial automation—challenges and technical solutions," in *Proceedings of the IEEE International Conference on Data Science and Data Intensive Systems*, pp. 532–539, Sydney, NSW, Australia, December 2015.
- [8] J. Q. Li, F. R. Yu, G. Deng, C. Luo, Z. Ming, and Q. Yan, "Industrial internet: a survey on the enabling technologies, applications, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1504–1526, 2017.
- [9] M. Weiner, M. Jorgovanovic, A. Sahai, and B. Nikolić, "Design of a low-latency, high-reliability wireless communication system for control applications," in *Proceedings of the IEEE International Conference on Communications (ICC)*, pp. 3829–3835, Sydney, NSW, Australia, June 2014.
- [10] V. Anitha and D. Tandur, "Wireless requirements and challenges in industry 4.0," in *Proceedings of the International Conference on Contemporary Computing and Informatics IEEE*, pp. 634–638, Mysore, Karnataka, India, November 2014.
- [11] J. Wan, S. Tang, Z. Shu et al., "Software-defined industrial internet of things in the context of industry 4.0," *IEEE Sensors Journal*, vol. 16, pp. 7373–7380, 2016.
- [12] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.
- [13] K. Murat and A. Duresi, "Quality of Service (QoS) in Software Defined Networking (SDN): A survey," *Journal of Network and Computer Applications*, vol. 80, pp. 200–218, 2017.

- [14] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the internet-of-things," in *Proceedings of the IEEE Network Operations and Management Symposium*, pp. 1–9, Krakow, Poland, May 2014.
- [15] W. Braun and M. Menth, "Software-defined networking using OpenFlow: protocols, applications and architectural design choices," *Future Internet*, vol. 6, no. 2, pp. 302–336, 2014.
- [16] Y. E. Okian, S. G. Lee, H. J. Lee, and J. H. Lam, "Distributed SDN controller system: a survey on design choice," *Computer Networks*, vol. 121, pp. 100–111, 2017.
- [17] M. Karakus and A. Durrresi, "A survey: control plane scalability issues and approaches in Software-Defined Networking (SDN)," *Computer Networks*, vol. 112, pp. 279–293, 2017.
- [18] J. Wang, Z. Liu, Y. Shen et al., "A distributed algorithm for inter-layer network coding-based multimedia multicast in Internet of Things," *Computers & Electrical Engineering*, vol. 52, pp. 125–137, 2016.
- [19] K. U. Adhvaryu and P. Kamboj, "Performance comparison between multicast routing protocols in MANET," in *Proceedings of the 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON)*, pp. 16–20, Mathura, India, October 2017.
- [20] C. Sule, P. Shah, K. Doddapaneni, O. Gemikonakli, and E. Ever, "On demand multicast routing in wireless sensor networks," in *International Conference on Advanced Information Networking and Applications Workshops*, pp. 233–238, Victoria, BC, Canada, May 2014.
- [21] K. Kaur, K. K. Saluja, and R. Singh, "Performance analysis of multicast routing protocols in ad-hoc networks," in *Proceedings of the International Conference on Computer and Communication Technology (ICCCCT)*, pp. 273–278, Allahabad, India, September 2014.
- [22] N. Mir, S. Musa, R. Torresand, and S. Swamy, "Evaluation of PIM and CBT multicast protocols on fault-tolerance," *International Journal of Computing and Networking Technology*, vol. 2, no. 2, pp. 59–64, 2014.
- [23] W.-K. Jia, "A scalable multicast source routing architecture for data center networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 1, pp. 116–123, 2014.
- [24] K. Sinha, B. Gupta, S. Rahimi, and A. Alyanbaawi, "Locality based core selection for multicore shared tree multicasting," arXiv preprint arXiv:1606.04928, 2016.
- [25] D. Sarddar and E. Nandi, "A new approach on optimized routing technique for handling multiple request from multiple devices for mobile cloud computing," *International Journal of Computer Science and Mobile Applications*, vol. 3, no. 8, pp. 50–61, 2015.
- [26] S. Islam, N. Muslim, and J. W. Atwood, "A survey on multicasting in software-defined networking," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 1, pp. 335–387, 2018.
- [27] T. Humernbrum, B. Hagedorn, and S. Gorchlatch, "Towards efficient multicast communication in software-defined networks," in *2016 IEEE 36th International Conference on Distributed Computing Systems Workshops*, pp. 106–113, Nara, Japan, June 2016.
- [28] Y. D. Lin, Y. C. Lai, H. Y. Teng, C. C. Liao, and Y. C. Kao, "Scalable multicasting with multiple shared trees in software defined networking," *Journal of Network and Computer Applications*, vol. 78, pp. 125–133, 2017.
- [29] W. Cui and C. Qian, "Dual-structure data center multicast using software defined networking," arXiv:1403.8065, 2014.
- [30] ONF, *OpenFlow Switch Specific Version 1.5.1*, Open Networking Foundation, Menlo Park, CA, USA, 2015.
- [31] J. M. Llopis, J. Pieczerek, and T. Janaszka, "Minimizing latency of critical traffic through SDN," in *2016 IEEE International Conference on Networking, Architecture and Storage (NAS)*, pp. 1–6, Long Beach, CA, USA, August 2016.
- [32] A. Hagberg, P. Swart, and D. Schult, *Exploring Network Structure, Dynamics, and Function using NetworkX*, Los Alamos National Lab (LANL), Los Alamos, NM, USA, 2008.
- [33] R. L. S. de Oliveira, A. A. Shinoda, C. M. Schweitzer, and L. R. Prete, "Using Mininet for emulation and prototyping Software-Defined Networks," in *Proceedings of the IEEE Colombian Conference on Communications and Computing (COLCOM)*, pp. 1–6, Bogota, Colombia, June 2014.
- [34] A. Tirumala, T. Dunigan, and L. Cottrell, "Measuring end-to-end bandwidth with Iperf using Web100," No. SLAC-PUB-9733, 2003.
- [35] H. S. Kim, S. J. Yun, H. J. Kim, and W. T. Kim, "A novel SDN multicast for large-scale IoT environments," in *Proceedings of the 8th International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju Island, Korea, October 2017.
- [36] P. T. Congdon, P. Mohapatra, M. Farrens, and V. Akella, "Simultaneously reducing latency and power consumption in openflow switches," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 3, pp. 1007–1020, 2014.