Software Defined Unicast/Multicast Jointed Routing for Real-Time Data Distribution

Shimin Sun[1], Wentian Huang[1], Xinchao Zhang[1], and Li Han[2(✉)]

[1] Tiangong University, Tianjin 300387, China
[2] Tianjin University of Technology, Tianjin 300384, China
hanli@tjut.edu.cn

Abstract. The explosive increasing of high bandwidth-consumption traffic puts great pressure on the Internet. Many popular applications springing up work in a manner of one-to-many or many-to-many communication, such as TikTok, Instagram, Tencent conference, and numbers of interactive games. Due to the scalability and applicability problems, existing multicast schemes, e.g. IP multicast, are not widely implemented. Instead, most of those traffic is transmitted through the Internet in unicast, which results in vast redundant traffic in backbone networks. In this paper, we propose a unicast/multicast jointed routing mechanism in software defined networks, SDUM. We devote to achieve unicast data distribution following a dynamic multicast tree, which is managed by centralized control and application plane. Other than OpenFlow protocol, this mechanism doesn't require any specific multicast protocols or software. The network can be a virtualized network with distributed OpenFlow devices interconnected by legacy routers. The evaluation results confirm the efficiency of the proposal in the number of control messages, signaling overhead, occupation of flow table entries, and qualitative comparison.

Keywords: Multicast algorithm · Data manipulation · Group management · Software defined networking

1 Introduction

In recent years, social media applications have made great success; especially those support live webcasting, Facebook, Twitter, Tik-Tok, etc. Online multimedia services are also getting much more pervasive, e.g. video conference, live broadcast, multiplayer gaming, and distributed data caching. Generally, those services distribute data in multicast-liked one-to-many or many-to-many pattern. Although, IP multicast [1] is a desirable approach in a functional viewpoint to avoid parallel delivery of identical packets on the same link. However, in reality, data packets are delivered in unicast in most cases. Large and redundant traffic utilizes significant proportions of network bandwidth resource. Under today's Internet architecture, unicast is still almost the only option for IP data packet transmission. With rapid development of multimedia technology and hardware, high quality streaming traverses the Internet to a large amount of end hosts. It is urgent to design an applicable multicast routing mechanism for those kind of services to reduce the waste of network resource.

H. Gao et al. (Eds.): ChinaCom 2020, LNICST 352, pp. 226–243, 2021.
https://doi.org/10.1007/978-3-030-67720-6_16

The limitation of scalability and rigid network architecture results in the deployment problem of existing multicast mechanisms. IP multicast, as one of the most admirable approach, faces various obstacles, such as global routing, group management, security consideration, QoS guarantee, address allocation, and inter-domain coordination issues [1]. Application Layer Multicast (ALM) [2], also known as overlay multicast, is the alternative approach of IP multicast. It is also not widely deployed due to the requirement of specific software and physical agents.

Software Defined Networking (SDN) [3] is a promising paradigm for elastic network control and resource management. Many internet enterprises, such as Google and Facebook, deployed their Data Center Networks (DCNs) based on SDN architecture. Most of the multicast approaches in SDN still focus on the implementation of IP multicast or group management protocols [4]. Therefore, the problems faced by multicasting in SDN are similar to IP multicast. Since the data is delivered using multicast address, it still requires routers on the multicast tree to support specific protocols, such as IGMP. Besides, a flow table entry of a OpenFlow Switch (OFS) is occupied by one multicast group. However, the storage of flow table, Ternary Content Addressable Memory (TCAM), is capacity constraint and expensive. The number of flow table entries is limited by 10^5 in current stage. For a network with n nodes, the maximum number of multicast group is $O(2^n)$. It is still inadequate to support universal multicast service for ordinary users depending on existing multicast mechanisms.

The motivation of this paper can be unfolded in three aspects. Firstly, instead of urging to design out-of-the-box solutions for IP multicast, we abandon all those basics and devote to realize multicast mechanism depending on pure SDN elements. Secondly, avoiding introducing any specific protocol or mechanism to network entities, we tend to use only OpenFlow protocol and primordial control plane. Finally, we design several modules in application plane to collect, analyze information and make decisions for multicast traffic, which is portable and programmable.

The main contribution of this paper is described as follows.

a. We present a practical multicast scheme for common applications. It is unnecessary for applications to be aware of multicasting. Instead, they can receive data based on unicast sessions.
b. We present a series of modules in application plane. Those modules are responsible for network information collection, resource scheduling, and decision making.
c. We present the detailed operation of packet manipulation and distribution, as well as dynamic group management. Required operations are implemented on OpenFlow-enabled routers without the involvement of end hosts and legacy routers.
d. We simulated the proposal on top of two classic network topologies using Mininet, proved more efficient than other approaches.

The rest of the paper is structured as follows. In Sect. 2, we introduce state-of-the-art literatures about multicast routing algorithms. Section 3 elaborates the proposed architecture, data processing algorithm, and group management mechanisms. Basic Operation of Data Distribution and Tree Algorithm are described in Sect. 4. Section 5 discusses the implementation and analysis, following with conclusion and future work in Sect. 6.

2 Related Work

2.1 Classic Multicast Protocols and Approaches

Various of IP multicast routing protocols exists for diverse application scenarios. Some studies exploit to design branch-ware forwarding technique [4]. DVMRP [5] and MOSPF [6] suit for the network with dense concentration of receivers, while PIM-SM [7] and BIDIR-PIM [8] designed for multicast tree construction in sparse networks. These protocols suffer from several uniformed issues [1], such as group dynamics, network scalability, member security, and multicast capable islands. Mbone project [9] brings multicast closer to reality, which applies IP tunneling to connect each multicast capable islands. However, static IP tunneling stymies the natural growth of multicast service requirement. ALM realizes data dissemination depending on software relay. It is easier to deploy ALM than IP multicast. This is because application owners rather than network providers manage group members and realize multicast functions with unicast relay at specific agent or end hosts. Deploying application in specific servers or end hosts is much more practical than installing network protocols to every network entity. ALM allows dynamic routing relying on network conditions and QoS requirements of applications. For example, Scattercast [10] calculates shortest path tree with minimum latency cost, while Overcast [11] constructs a multicast tree for maximum bandwidth utilization from source to receivers. Some approaches adopt comprehensive consideration of network conditions, such as PeerCast [12] and Vcast [13]. Consistent problems of those solutions are relay agent placement, network discovery, as well as group labeling.

2.2 Software Defined Multicast Approaches

SDN has been verified in practice as a promising network architecture to tackle the problems of IP multicast. SDN does not natively support multicast addressing and multicast routing protocols. Many literatures have been carried out on the efficient delivery of multicast traffic and multicast routing algorithms. However, most of them are still focus on the implementation or amendment of IP multicast in SDN environment. In other words, data distribution is still based on multicast address and suffers from majority of IP multicast problems.

To support IP multicast, MultiFlow [14] presented an approach to enable SDN to parse IGMP messages. OFSs exchange IGMP messages to accomplish group management. Evaluation results showed lower time-consuming for multicast tree creation than DVMRP. PVMC-SDN [15] proposed a virtual platform to interpret MPLS and ARP protocols, in order to forward data along the multicast tree. OFM [16] developed multicast services based on SDN programmable functionalities from clean-slate perspective. The limitation is that OFS need to be aware of joining or leaving message, so that it is able to deliver the message directly to SDN-C. Multicast address was allocated for data distribution through the network, nevertheless lack of detailed process description.

To enhance scalability of IP multicast, Locality-aware Multicast Approach (LAMA) [17] was proposed to reduce the computation for multicast tree creation.

To build a multi-group shared tree, a Rendezvous Point (RP) election algorithm was designed to pick out the proper RP with minimum distance to all sources. However, message exchanging for RP election causes high signaling overhead and challenges RP's flow table size. BAERA [18] presented a branch-aware Steiner tree construction algorithm to minimize the number of branch nodes and edges of the multicast tree. However, authors focus on tree construction algorithm, without presenting data distribution method and flow table capacity of branch nodes. SDM [19] attempted to slice multicast tree to enhance live streaming distribution. It designed SDM domain consisting of pure OFSs to distribute multicast traffic. Similar to IP multicast, a flow table entry was required for each group. Specific messages were elaborated to realize the functions of network entities, such as NL-SDM and Virtual Peer Instance. Outside the domain, packets were transmitted in unicast. A comprehensive survey on SDN based multicast approaches were presented in [20], where most of them still focused on multicast tree construction algorithm as IP multicast.

According to above analysis, SDN based multicast approaches are better than IP multicast and ALM in applicability. However, any introduced message exchanges could increase the load of flow table, which is expensive and volume-limited. For example, mainstream products of OFSs usually have hundreds of thousands of entries using Ternary Content-Addressable Memory (TCAM) [21]. The size of TCAM is considerably scarce comparing with huge amount of traffic flows. This is the great obstacle of SDN based multicast approaches and results in scalability problem.

3 Software Defined Unicast/Multicast Jointed Routing (SDUM)

3.1 Multicast Issues in Current Stage

In current Internet service architecture, one-to-many and many-to-many data distribution schemes are mostly deployed in Client-Server (C/S) or Peer-to-Peer (P2P) approaches. That is, end host fetches data based on unicast session from a single server. Access network of the server bears the maximum flow stress with massive amount of redundant data. Comparing to multicast, the primary advantage of unicast is route aggregation. Route aggregation summarizes route by network address instead of every specific IP address, which can significantly save routing table resource.

To alleviate the traffic load of server-side and backbone networks, multicast is a promising approach to improve the efficiency of data dissemination. The problem is that it is arduous to conduct route aggregation due to the random allocation of multicast addresses and distributed receivers. Every router needs to keep a routing entry for each group, in order to resolve multicast address when processing multicast traffic.

The key idea of this paper is to design agile unicast algorithms to realize data distribution along calculated multicast tree without deploying specific multicast routing protocol. Therefore, route aggregation is executed for those data traverse the network.

3.2 System Architecture

On top of SDN architecture, the system architecture is illustrated in Fig. 1, consisting of data plane, control plane, and application plane. Data plane and control plane follow the basic SDN architecture, while all the designed modules are located at application plane.

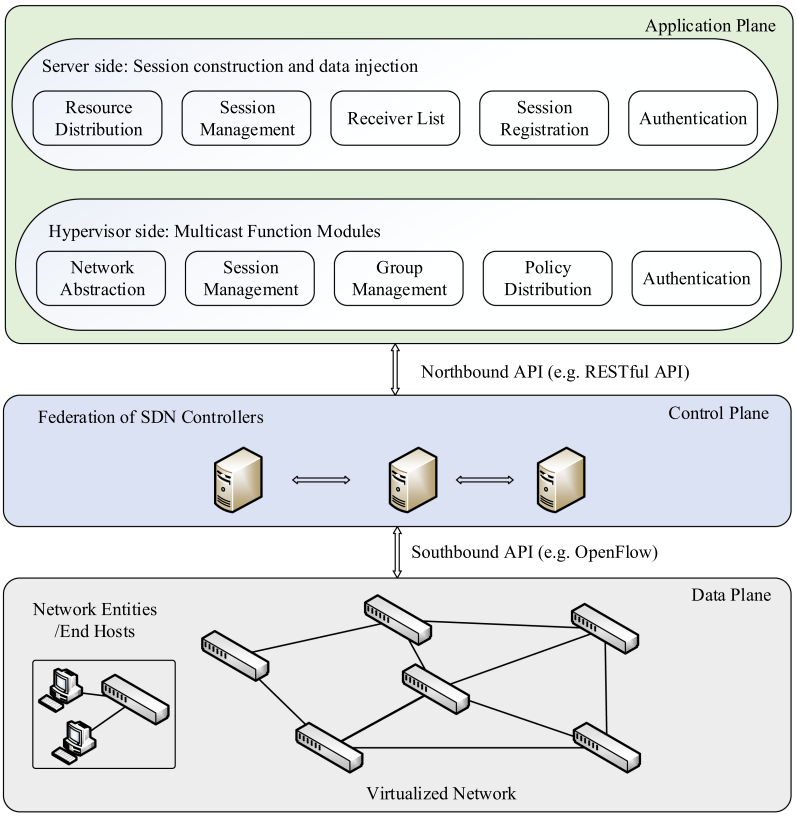


Fig. 1. System architecture to enable formulation of multicast

Data plane is composed of OpenFlow Switches (OFSs) interconnected by legacy routers, which forms a virtualized network. OFSs process data packets following the rules defined in flow table. Besides, they gather link/port status and statistical information of the network, and send to controller using OpenFlow messages through a secure channel.

Control plane is in charge of network management, which is composed of multiple controllers federated to share network information. Controller has the capability to acquire a global overview of network topology and status. Various controllers have already been developed, such as Floodlight, POX, Beacon, and Ryu [22]. They are collaborated by hypervisor to conduct network level traffic engineering. For multi-controller integration,

several hypervisor systems were developed, HyperFlow, FlowVisor, Kandoo, Onix, etc. [23].

At application plane, various applications can be developed to realize specific functionalities, such as routing algorithm, load balancing, security concern, network management, and other new features. Application plane interacts with control plane through northbound API (e.g. Restful API). Meanwhile, control plane communicates with data plane through southbound API (e.g. OpenFlow protocol).

We designed two series of modules, server-side modules and hypervisor-side modules. Server-side modules are responsible for data distribution, recording of session information, and authentication. Hypervisor-side modules are responsible for network abstraction, group management, policy distribution, and authentication.

3.3 Functional Description

The roles and responsibilities of the related network entities are described as follows.

a. Receivers. End hosts that send queries and achieve authentication to the server. They receive data in common unicast session.
b. Source. The server that provides streaming service. It performs authentication with applicants as well as its corresponding controller. Besides, it needs to upload group and session information to the controller.
c. Controller. Controller collects network status from OFSs, records session information from Source. It obtains decision from controller-side modules, and issues rules to OFSs.
d. OFSs. Network entities serve as the actors of data forwarding. It also executes data manipulation policy received from Controller.
e. Legacy routers. Current dominant network core entities, which connect OFSs distributed in the network. Legacy routers are ignored in the virtualized network when implementing the proposed mechanisms.

The interactions of entities are illustrated in Fig. 2. For simplicity, we assumed that source (*S*) only keeps a single multicast group and access to the network through its gateway OFS. The authentication between source and controller is achieved before initiating the data distribution service. Controller issues a multicast address to *S* as group ID.

Initially, *S* is ready for service request. When *S* receives the first request message from host *R1*, it replies with a response message and an authentication message. The authentication could be completed by the common ID/password login identification according to the default design of utilized applications. After the success of authentication, *S* inserts *R1*'s information to a list, Receiver List (*RL*). *RL* list records the session information of each receiver, including source IP address, source port number, destination IP address, destination IP address, destination port number, and an expiration timer. Meanwhile, *S* sends the requested source to *R1* based on the unicast session. Since data could be simply transmitted along default route, it is unnecessary to issue rules to OFSs.

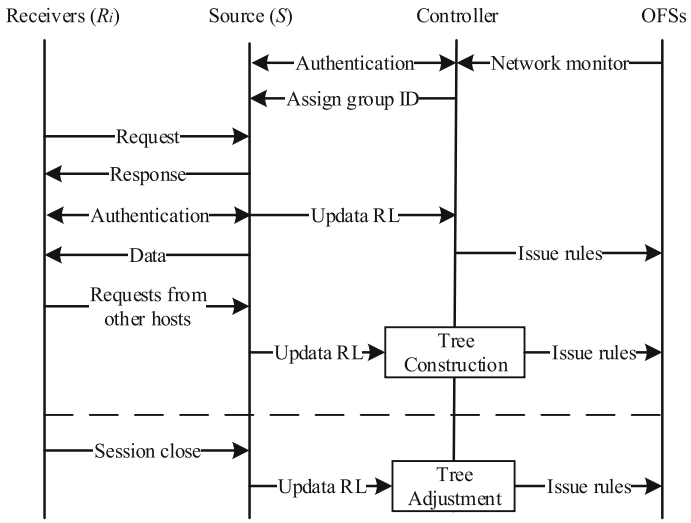


Fig. 2. Brief illustration of functional flowchart

Upon receiving other requests, *S* updates *RL* and keeps update with the controller. The controller make decision for multicast tree construction and data manipulation, which will be described in Sect. 4. Afterwards, it issues the rules to corresponding OFSs to enable data packets transmitted along the constructed multicast tree.

If a session close or expiration timer timeout, a receiver will be marked as departure. *S* and the controller update *RL*. Tree adjustment algorithm calculate the appropriate tree to guarantee data reception and tree optimization.

3.4 Data Structure

The multicast manager (Hypervisor) needs to maintain several tables for each multicast group, which are listed in Table 1 and introduced below.

a. G_i: The multicast group, which is marked by an locally unique multicast address. It stores server-side information: IP address of S (IP_{Si}), port number of S ($PORT_{Si}$), multicast address (IP_{Mi}).
b. RL_i: The table maintains host-side information: IP address of R_i (IP_{Ri}), port number of R_i ($PORT_{Ri}$), and an expiration timer (TTL_{Ri}).
c. $OFSL_i$: The table includes IP address of OFSs that on the multicast tree.
d. $FOFS_i$: The table contains OFSs that need to do specific operation for data delivery along the multicast tree. We call those OFSs, Fork OFS (*FOFS*).
e. $EOFS_i$: The table records OFSs on the multicast tree that is incapable to be a *FOFS*, such as OFSs with insufficient flow table volume.

Table 1. Data structures that deployed in application layer

G_i	$<IP_{Si}, PORT_{Si}, IP_{Mi}>$
RL_i	$<IP_{R1}, PORT_{R1}, TTL_{R1}>, ..., <IP_{Rn}, PORT_{Rn}, TTL_{Rn}>$
$OFSL_i$	$OFS_1, ..., OFS_n$
$FOFS_i$	$<IP_{R1}, OFS_1, Rule_1>, ..., <IP_{Rn}, OFS_n, Rule_n>$
$EOFS_i$	$OFS_1, ..., OFS_n$

4 Operation of Data Distribution and Tree Algorithm

4.1 Data Manipulation Mechanism

To illustrate the basic operations of data dissemination along the multicast tree, we present a brief instance in Fig. 3. Thereby, source is indicated by a triangle icon; OFSs are indicated by circular icon with sequence number inside (from OFS1 to OFS8); and receivers are gray square (R1, R2, and R3).

Source sends packets to the first receiver R1 directly. When R2 joins, according to tree construction algorithm, OFS4 is set to be the operation node for data manipulation. OFS4 duplicates the data packets, and translates the destination IP/Port of duplicated packets from $<IP_{R1}, PORT_{R1}>$ to $<IP_{R2}, PORT_{R2}>$. Then, if R3 joins, OFS2 performs the similar operation, translates duplicated packets from $<IP_{R1}, PORT_{R1}>$ to $<IP_{R3}, PORT_{R3}>$.

Notice that, it is unnecessary to perform data manipulation by a physical fork node or not. It can be any OFS on the tree, especially when the desired fork node capacity overloaded. For example, in Fig. 3, if OFS2 overloaded, OFS1, OFS3, OFS7 or even OFS4 can become the operation node for R3. The hypervisor modular decides which OFS selected as operation nodes, according to tree construction and adjustment algorithm.

4.2 Greedy-Based Multicast Tree Construction Algorithm

In this section, we elaborate multicast tree construction algorithm by host joining group and leaving group. Join group operation is realized by a greedy algorithm to snap a branch to the multicast tree. Leave group operation should take care of the impact of the prune of any branch on other group members. It is possible for the controller to obtain the entire or regional network topology and link status in SDN architecture. Therefore, we can calculate the optimal path and multicast tree in a simulated environment, without injecting any network detection message.

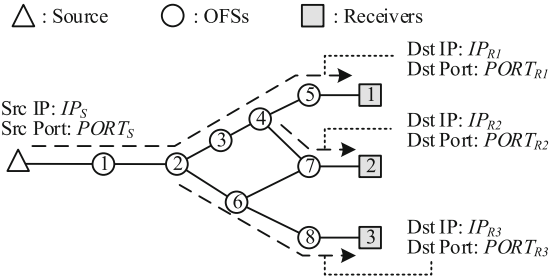


Fig. 3. Basic operations of multicast data distribution

Joining Group Operations

In tree construction stage, multicast tree grows in a greedy way. That is, when new host joins the group, the nearest OFS on the multicast tree has the top priority as the operation node.

As shown in Algorithm 1, we propose a greedy based multicast tree maintenance algorithm. When the first host requests the source, data transmitted to the host along the default route. A crowd of OFSs on the path from the source to the receiver are recorded in table *OFSL*. Then, if other hosts request the source, we need to find a path that can fetch the required data. Simply, it finds the nearest OFS on the path from the source to the first receiver (Eq. (1)). If the OFS is capable to be the operation node, then we assign it as the fork node. Otherwise, delete the OFS from *OFSL* list, and execute Eq. (1) again. If there are more than one OFS that has the same distance to the host. Equation (2) is applied to find an OFS that is closest to the source node.

$$FOFS = Min\left(Dist(\{OFS_i\}, R_j)\right) \tag{1}$$

$$FOFS = Min(Dist(\{FOFS_i\}, S)) \tag{2}$$

After the tree construction, the new receiver is inserted to *RL* list, the selected fork node is added to *FOFS* list. *OFSL* list is updated with new OFSs joining and incapable OFSs deleting. Besides, controller issue rules to the *FOFS* to execute appropriate data manipulation operation.

Algorithm 1: Host Joins a Group
Input: group ID: G, host: R_j, Source: S

1. Add R_j to RL
2. **if** j=1, //the first group member
3. find all the OFSs on the path from S to R_j
4. Add above OFSs to $OFSL$
5. **else**
6. **for** $\{OFS_i\}$ in $OFSL$, **do**
7. $FOFS_j = Min\left(Dist(\{OFS_i\}, R_j)\right)$
8. **if** multiple $FOFS_j$ exist
9. $FOFS_j = Min\left(Dist(\{FOFS_i\}, S)\right)$
10. **if** $FOFS_j$ overloaded
11. Add $FOFS_j$ to $EOFS$, Delete from $OFSL$
12. Go to **do**
13. **else**
14. Add $FOFS_j$ to $FOFS$ list, add OFSs on the path from $FOFS_j$ to R_j to $OFSL$, issue rules to $FOFS_j$
15. **endif**
16. **endif**
17. **Output:** RL, $OFSL$, $FOFS$, issue rules to $FOFS_j$

Leaving Group Operations

Multicast group member status is dynamic. A member leaves a group when the session closed or network interrupted. If a member is determined as leaving by the controller, it is deleted from RL list. Multicast tree is recalculated according to Algorithm 2. The controller should check whether other receivers acquire data using the branch from $FOFS_j$ to R_j or not.

If no impact on the data transmission to other group members, the branch will be pruned simply. The controller issues deletion rules to $FOFS_j$ to remove related flow entry. Besides, we remove $FOFS_j$ from $FOFS$ table.

If the prune of the branch causes the termination of data dissemination to other members, following operations should be carried out before pruning the branch.

a. According to the greedy algorithm, find an alternative $FOSF$ for the members that will be affected if $FOFS_j$ removed.
b. Issue rules to the selected operation node $FOFS_k$ for R_k.
c. Add those OFSs that connect to R_k to $OFSL$ list.
d. Delete $FOFS_j$ from $FOFS$ list as well as the corresponding flow table entry.

Finally, it outputs updated *RL*, *OFSL, FOFS* table, and issues rules to $FOFS_k$.

Algorithm 2: Host Leaves a Group
Input: group ID: G, host: R_j, Source: S

1. Get the set of $\{OFS_i\}$ in *OFSL*
2. Get *FOFS* of R_j: $FOFS_j$
3. **for** $\{OFS_i\}$, **do**
4. Delete OFSs on R_j's branch from *OFSL*
5. **if** no other branch exists on R_j's branch
6. Delete $FOFS_j$ from *FOFS*
7. **else** \\ R_k exists on R_j's branch
8. $FOFS_k = Min\left(Dist(\{OFS_i\}, R_k)\right)$
9. **if** multiple $FOFS_k$ exist
10. $FOFS_k = Min\left(Dist(\{FOFS_i\}, S)\right)$
11. Issue flow table entry to $FOFS_k$
12. Add OFSs on R_k's branch to *OFSL*
13. Delete $FOFS_j$ from *FOFS*
14. **endif**
15. **Endif**
16. Delete flow table entry of R_j from $FOFS_j$
17. **Output:** *RL*, *OFSL*, *FOFS*, issue rules to $FOFS_k$

We can conclude that the application plane achieves the most workload. What the control plane to do is issue rules to corresponding OFSs, while the data plane is responsible for data forwarding. Only the corresponding OFSs are need to execute those received policies. In this way, not all network entities need to support multicast protocol or OpenFlow protocol. AS a contrast, IP multicast requires all routers on the tree to support multicast protocol, while SDN based multicast approaches requires all OFSs on the tree to maintain a forwarding table entry for each group.

5 Methodology, Implementation and Evaluation

Most of multicast approaches in SDN are still explicitly or implicitly to realize IP multicast functions. However, the performance on data delivery efficiency is commonly no better than IP multicast. As a comparison, we chose SDM [19], an SDN based multicast approach; as well as a classical IP multicast routing protocol, PIM-SM [7]. PIM-SM is suitable routing protocol for IP multicast in sparse networks, which is in line with the cases of our proposal.

In the simulation, the default forwarding principle of OFSs was set to be same as the typical routing protocol, Routing Information Protocol (RIP). Default paths from the source to each receiver based on the shortest path. In the experiments, data delivery efficiency was evaluated by counting the number of control messages. Signaling overhead ratios were analyzed to evaluate the overall cost metrics of control packets to enable entire multicast traffic distribution. To evaluate the workload of OFSs in or routers in traditional network, we compared the number of installed flow table rules and routing table entries.

5.1 Simulation Setup

a. **Operating environment:** We implemented SDUM on top of Ryu controller [24] based on OpenFlow protocol version 1.3. The network topology was constructed using Mininet [25] version 2.2.2. Due to function limitation, it is difficult to implement PIM-SM protocol in Mininet. So that, we deployed the same network with exact same link status using NS2 which was able to simulate IP multicast protocols. The testbed was allocated in a desktop PC with Ubuntu Desktop 16.04. $S_i$mulation time was set to 60 s for each scenario.
b. **Network topology:** Two classic network topologies were implemented, DCN topology and realistic WAN topology. A fat-tree DCN topology with three tiers is shown in Fig. 4. There are 14 OFSs and 15 hosts. A realistic WAN topology for Microsoft's data centers is illustrated in Fig. 5, where every OFS connects to two hosts as exhibited in the legend, totally 15 hosts. A single source with a controller were allocated in the network. For each link, bandwidth and delay were set to 1 Gbps and 10 ms respectively in both network. BRITE [26] was used to generate network topologies.
c. **Traffic generation**: The leftmost host was the source to generate constant UDP traffic using iPerf. The multicast address was configured to 224.0.55.55. Data rate injected by the source is constantly 1 Mbps.

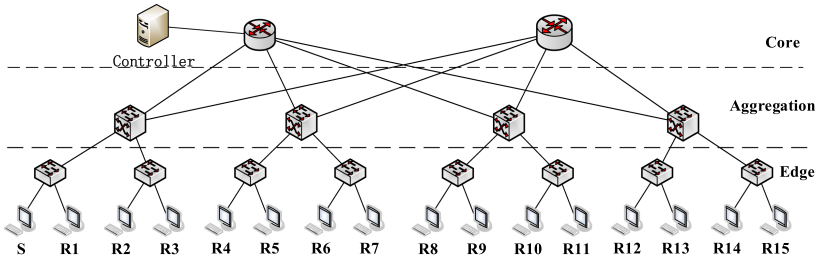


Fig. 4. Typical data center interconnected network topology

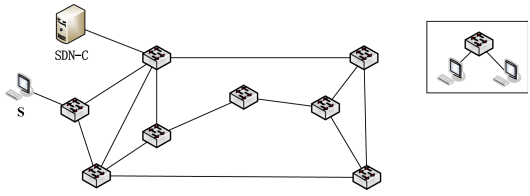


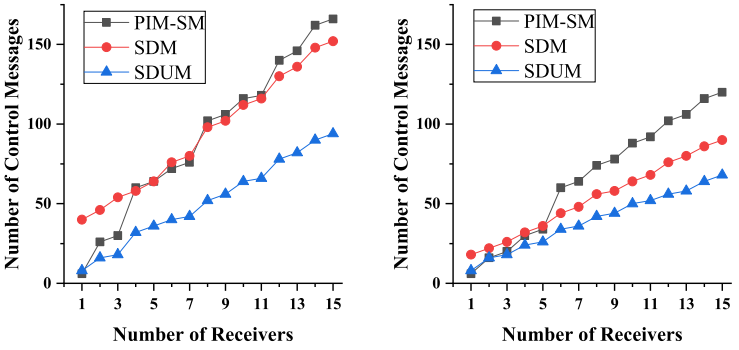
Fig. 5. WAN topology with each switch connected to two hosts

5.2 Evaluation and Analysis

Examined protocols and approaches were based on shortest path tree algorithm. Therefore, they conducted actually the same multicast tree. It is meaningless to compare tree efficiency, data delivery latency or bandwidth consumption with each other. Instead, we evaluated the number of control messages, signaling overhead ratios, and the number of rules and routing table entries installed to enable multicasting. Besides, qualitative comparison of typical multicast approaches is presented.

Control Messaging Overhead

Control messaging overhead was assessed by the number of control messages that injected to the network and its signaling overhead ratio. The number of control messages for the operation of multicast service reveals the load of protocols or approaches imposing on the network. For PIM-SM, control messages are exchanged among routers to maintain multicast tree, such as IGMP messages, Hello messages, and RP election messages. For SDM and SDUM, control messages are mainly OpenFlow messages exchanged between OFSs and controller, including PacketOut messages for rule installation, and PacketIn messages for unknown packet uploading, etc. SDM generates control messages for domain management, virtual peer operation, host joining and leaving, etc. We omitted the Beacon messages between OFSs and controller, because it is inevitable in SDN regardless of whether deploying SDUM or not.



a) fat-tree DCN topology b) WAN topology

Fig. 6. The number of control messages with incremental receivers

Figure 6 shows the number of control messages variation. It indicates that PIM-SM leads to more reliance on network topology. In Fig. 6(a), the line-symbol graph of PIM-SM in fat-tree DCN topology shows a sharp leap when R4, R8, or R12 joins the tree. This is because the multicast tree topology occurs a significant change. Routers exchanges more control messages to maintain the multicast tree with the growth of network.

The number of control messages of SDM is close to IP multicast of DCN topology, but lower in WAN topology. SDUM generates about 40% reduction of control messages comparing to PIM-SM in both topology, while around 30% reduction comparing with SDM in WAN topology on average.

Signaling Overhead Ratio

Signaling overhead ratios are depicted in Fig. 7. As background traffic, we emulated constant 1Mbps UDP traffic as the streaming. Depending on the basic format of control messages, PIM-SM control messages are around 70 bytes while control messages in SDN are around 200 bytes. As a result, the number of the control message may not reveal the real signaling overhead.

From the evaluation results in Fig. 7, we can see that SDM produces higher signaling overhead than PIM-SM and SDUM. It is on average 47% and 25% higher than SDUM for DCN and WAN topology respectively. The overhead ratio of SDUM is close to PIM-SM due to its lightweight control cost.

OpenFlow Rules Required for Multicast

Flow table utilization is essentially to be measured since the volume of TCAM in OFSs and routing table entries in a router are limited. In the experiments, we counted the number of routing table entries required to realize PIM-SM routing and data delivery, and the flow table entries of OFSs used for multicast packet duplication, address translation, and forwarding. For SDM, the number of rules installed greatly relies on network topology and receiver location. It is necessary to install rules not only on access OFS of each receiver, but also on specific core OFSs. The results in Fig. 8 show that rules increase sharply along with the increase of receivers in both experimental scenarios.

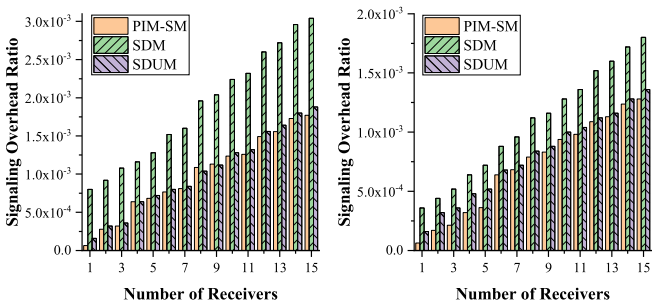


Fig. 7. Signaling overhead with incremental receivers

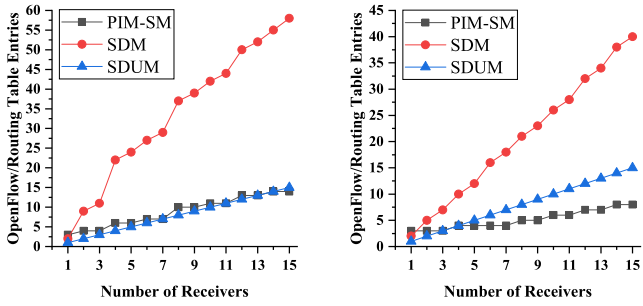


Fig. 8. The number of rules that issued to OFSs with incremental receivers

As a contrast, although PIM-SM generates more control messages as shown in Fig. 8, the occupation of routing table entries is less. This is because IP multicast aims to share multicast routing table with each other, which is one entry per group. Only routers in the multicast tree have the multicast routing table entry. Therefore, the number of routing table entries depend on the quantity of multicast groups deployed in the router. Besides, a RP-Set table exists to store RP connection information with few entries, which we skipped in the experiment.

In SDUM, only *FOFSs* maintain individual entries for each receiver. The number of rules is linearly increasing along with the number of receivers. Other OFSs on the tree only forward packets based on default routing strategies without installing extra rules.

From the experimental results, we can conclude that SDM shows a poor scalability performance with the increasing receivers. Routing table size required by PIM-SM relies on the number of groups. Flow table size required by SDUM highly depends on the number of receivers in a group. Therefore, SDM is suitable for small network with limited receivers; PIM-SM is suitable for sparse network with limited groups; while SDUM is better to be deployed for large networks with a small quantity of receivers in each group.

Qualitative Comparison

It is worthy to analyze the pros and cons of SDUM by making a qualitative comparison with typical protocols and approaches, as listed in Table 2, including PIM-SM, ALM, SDM, and SDUM.

The major difference of them is the deployment requirement. PIM-SM requires all routers to install the multicast protocol, while SDUM has no specific requirement on network entities. ALM needs to deploy specific hardware agents or terminal relay software, which is difficult to be widely deployed with controlled data relay. SDM attempts to build a SDM domain where the designated group ID can be parsed by OFSs, which is implicitly in line with IP multicast.

Those problems lead to deployment difficulty and low scalability to large networks. Distribution management method of IP multicast and ALM results in notoriously difficult to achieve ISP control of multicast traffic. Traffic latency may be high in ALM

caused by terminal relay through a longer route. For access control, application layer software development is easier than the design of network layer protocols.

Table 2. Qualitative comparison of typical multicast approaches

	PIM-SM	ALM	SDM	SDUM
Incremental deployment	All routers multicast capable	Relay entities or terminal relay	All OFSs in SDM domain	Only branch nodes are OFS required
Scalability	Low (routers disabled multicasting exist)	High (protocols and entities mature)	Low (inadequate in legacy router coexisting networks	High (deployment with partial SDN)
ISP control	Low (hard to manage distributed routers)	None	High (centralized control of SDM domain)	High (centralized control of FOFSs)
Latency	Low (optimized route)	High (due to terminal relay)	Low (optimized route)	Low (optimized route)
Access control	IGMP-AC and SIGMP required	Application layer authentication	No concern	Application layer source and controller authentication

From experimental results and qualitative evaluation, we distinctly verdict that SDUM performs better than IP multicast (PIM-SM) and SDM in terms of control message overhead, signaling overhead ratio, and flow table entry size. This is an evidence that SDUM is a promising approach for multicast support in evolutional IP networks, conquering the shortage of IP multicast, ALM, and SDM.

6 Conclusion

In this paper, we propose to improve multicast routing by manipulating unicast data transmission on-the-fly to provide a generic multicast service in network layer. All required functionalities are placed on the centralized application plane. With the assistance of the source for information collection and authentication of group members, it offers a transparent service to heterogeneous hosts and network entities. The evaluation results indicate an evident improvement in terms of traffic overhead, flow table occupation, and qualitative comparison. Future work will focus on following studies, including multiple controller coordination mechanism, multicast tree optimization and fast failure recovery mechanism, QoS guarantee, and experiments in random networks with complex topologies.

Acknowledgements. This work is supported by Science and Technology Development Fund of Tianjin Education Commission for Higher Education (No. 2017KJ090).

References

1. Diot, C., Levine, B.N., Lyles, B., Kassem, H., Balensiefen, D.: Deployment issues for the IP multicast service architecture. IEEE Netw. **14**(1), 78–88 (2000)
2. Park, J., Lee, J.M., Kang, S.: Design implementation of overlay multicast protocol for many-to-many multicast services, In: The 9th International Conference on Advanced Communication Technology, pp. 2144–2147, Okamoto, Kobe. IEEE (2007)
3. Nunes, B.A.A., Mendonca, M., Nguyen, X.-N., Obraczka, K., Turletti, T.: A survey of software-defined networking: past, present, future of programmable networks. IEEE Commun. Surv. Tutor. **16**(3), 1617–1634 (2014)
4. Chiang, S., Kuo, J., Shen, S., Yang, D., Chen, W.: Online multicast traffic engineering for software-defined networks, In: Proceedings of IEEE INFOCOM, Honolulu, HI, pp. 414–422 (2018)
5. Waitzman, D., Partridge, C., Deering, S.E.: Distance vector multicast routing protocol, IETF RFC 1075 (Experimental) (1988)
6. Moy, J.: Multicast extensions to OSPF, IETF RFC 1584 (1994)
7. Fenner, B., et al.: Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification (Revised), IETF RFC 7761 (2016)
8. Speakman, T., Vicisano, L., Handley, M.J., Kouvelas, I.: Bidirectional protocol independent multicast (BIDIR-PIM), IETF RFC 5015 (2007)
9. Almeroth, K.C.: The evolution of multicast: from the MBone to interdomain multicast to Internet2 deployment. IEEE Netw. **14**(1), 10–20 (2000)
10. Yatin, C.: Scattercast: an adaptable broadcast distribution framework. Multimedia Syst. **9**, 104–118 (2003)
11. Jannotti, J., Gfford, D.K., Johnson, K.L., Kaashoek, M.F., O'Toole, J.W.: Overcast: reliable multicasting with an overlay network. In: Proceedings of the 4th Conference on Symposium on Operating System Design & Implementation, San Diego, CA, pp. 197–212. Usenix (2000)
12. Zhang, J., Liu, L., Ramaswamy, L., Pu, C.: PeerCast: churn-resilient end system multicast on heterogeneous overlay networks. J. Netw. Comput. Appl. **31**(4), 821–850 (2008)
13. Hua, K.A., Tran, D.A., Villafane, R.: Overlay multicast for video on demon the internet. In: Proceedings of the 2003 ACM Symposium on Applied Computing, Melbourne, Florida, pp. 935–942. Association for Computing Machinery (2003)
14. Bondan, L., Müller, L.F., Kist, M.: Multiflow: multicast clean-slate with anticipated route calculation on OpenFlow programmable networks. J. Appl. Comput. Res. **2**(2), 68–74 (2012)
15. Liao, S., Wu, C., Hong, X., Zhu, K., Chen, S.: Virtualized platform for multicast services in software defined networks. Chin. J. Electron. **26**(3), 453–459 (2017)
16. Yu, Y., Zhen, Q., Xin, L., Shanzhi, C.: OFM: a novel multicast mechanism based on Openflow. Adv. Inf. Sci. Serv. Sci. **4**(9), 278–286 (2012)
17. Lin, Y.-D., Lai, Y.-C., Teng, H.-Y., Liao, C.-C., Kao, Y.-C.: Scalable multicasting with multiple shared trees in software defined networking. J. Netw. Comput. Appl. **78**, 125–133 (2017)
18. Huang, L.-H., Hung, H.-J., Lin, C.-C., Yang, D.-N.: Scalable bandwidth-efficient multicast for software-defined networks. In: Proceedings of 2014 IEEE GLOBECOM, Austin, TX, USA, pp. 1890–1896. IEEE (2014)
19. Rückert, J., Blendin, J., Hausheer, D.: Software-defined multicast for over-the-top overlay-based live streaming in ISP networks. J. Netw. Syst. Manag. **23**(2), 280–308 (2015)

20. Islam, S., Muslim, N., Atwood, J.W.: A survey on multicasting in software-defined networking. IEEE Commun. Surv. Tutor. **20**(1), 355–387 (2018)
21. Alsaeedi, M., Mohamad, M.M., Al-Roubaiey, A.A.: Toward adaptive scalable OpenFlow-SDN flow control: a survey. IEEE Access **7**, 107346–107379 (2019)
22. Priya, A.V., Radhika, N.: Performance comparison of SDN OpenFlow controllers. Int. J. Comput. Aided Eng. Technol. **11**(4/5), 467–479 (2019)
23. Bannour, F., Souihi, S., Mellouk, A.: Distributed SDN control: survey, taxonomy, challenges. IEEE Commun. Surv. Tutor. **20**(1), 333–354 (2018)
24. Asadollahi, S., Goswami, B., Sameer, M.: Ryu controller's scalability experiment on software defined networks. In: Proceedings of 2018 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC), Bangalore, India. IEEE (2018)
25. Mišić, M.J., Gajin, S.R.: Simulation of software defined networks in Mininet environment. In: Proceedings of 2014 22nd Telecommunications Forum Telfor (TELFOR), Belgrade, Serbia, pp. 1055–1058. IEEE (2014)
26. Medina, A., Lakhina, A., Matta, I., Byers, J.: BRITE: an approach to universal topology generation. In: Proceedings of Proceedings Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, Cincinnati, OH, USA, pp. 346–353. IEEE (2001)