A DINloop-based Inter-domain Multicast Using MPLS

H. Q. Guo
Institute for Infocomm Research, A*STAR,
21 Heng Mui Keng Terrace,
Singapore 119613
guohq@i2r.a-star.edu.sg

L. H. Ngoh
Institute for Infocomm Research, A*STAR,
21 Heng Mui Keng Terrace,
Singapore 119613
lhn@i2r.a-star.edu.sg

W. C. Wong
Institute for Infocomm Research, A*STAR,
21 Heng Mui Keng Terrace,
Singapore 119613
lwong@i2r.a-star.edu.sg

Abstract

To overcome scalability and control overhead problems existing in inter-domain multicast, DINloop (Data-In-Network loop) based multicast with MPLS is proposed. DINloop is a special logical path formed using MPLS Label Switched Paths and it consists of multiple DIN Nodes which are core routers that connect to each intra-domain. We use DINloop to manage inter-domain multicast group membership. Facilitated by the DINloop, multiple DIN Nodes in the core network easily form a Steiner tree for multicast traffic. Furthermore, traffic for different multicast groups sharing the same path are aggregated through a label stack. Simulations demonstrate that DINloop-based multicast results in less message load needed to form the multicast structure. In addition, the routing table size in other core routers does not increase as the number of multicast group increases, and therefore routing scalability is improved. Finally, the inter-domain multicast delay in DINloop-based multicast is lower than that of a unidirectional tree.

1. Introduction

IP Multicast is an extremely useful and efficient mechanism for multi-point distribution of information [1], such as, audio and video distribution, audio and video conferencing, distance learning, and so on. However, the current infrastructure for inter-domain multicast faces some problems.

Multicast Source Discovery Protocol (MSDP) [2] and Border Gateway Multicast Protocol (BGMP) [3, 4] were developed for inter-domain multicast. However, MSDP requires each multicast router to maintain forwarding state for every multicast tree passing through it and the number of forwarding states grow with the number of groups [5, 6, 7]. In addition, MSDP floods source information periodically to all other RPs (Rendezvous Points) on the Internet using TCP links between RPs [8]. Thus, the MSDP inter-domain multicast protocol suffers from scalability and control overhead problems.

BGMP scales better to large numbers of groups by allowing (*, G-prefix) and (S-prefix, G-prefix) states to be stored at the routers where the list of targets are the same. In order to achieve this, the Multicast Address-Set Claim (MASC) protocol must form the basis for a hierarchical address allocation architecture [3]. MASC uses a listen and claim with collision detection approach. This approach has two drawbacks. First, this approach is not supported by the present structure of non-hierarchical address allocation architecture. Second, the claimers have to wait for a suitably long period to detect any collision, i.e., 48 hours [3]. So it is not suitable for dynamic setup.

The multiprotocol label switching (MPLS) [9] has emerged as an elegant solution to meet the bandwidth-management and service requirements for next generation Internet protocol (IP) based backbone networks. Multicast and MPLS are two complementary technologies and merging these two technologies where multicast trees are constructed in MPLS networks will enhance performance and present an efficient solution for multicast scalability and control overhead problems [7].

To overcome the existing problems, a DINloop-based multicast with MPLS is proposed to optimize inter-domain multicast. DINloop is a special logical path formed using MPLS LSPs (Label Switched Paths) for the purpose of circulating data continuously in the network. In this paper, we use DINloop to manage inter-domain multicast group membership. A DINloop consists of multiple DIN Nodes which are core routers that connect to each intra-domain and
function as the RP for that domain respectively. Then multiple DIN Nodes form the Steiner tree using Steiner tree-based heuristic routing algorithm for the multicast traffic. The remainder of this paper is organized as follows. The related works are described in Section 2. Our solution overview is outlined in Section 3. Section 4 presents the details of DINloop-based multicast. We present experimental results in Section 5 followed by the conclusion in Section 6.

2. Related works

Significant research efforts have focused on multicast scalability and control overhead problems. In aggregated multicast [5], multiple multicast groups share one aggregated tree to reduce forwarding state, and a centralized tree manager is introduced to handle aggregated tree management and matching between multicast groups and aggregated trees. Aggregated multicast is targeted for intra-domain multicast and the centralized tree manager is a weakness. MPLS Multicast Tree (MMT) [6] utilizes MPLS LSPs between multicast tree branching node routers in order to reduce forwarding states and enhance scalability. In MMT, each domain should contain a network information manager system (NIMS) to collect join messages from all group members and have a complete overview about the multicast network. For inter-domain, the border router contacts border routers in other domains with a normal (S, G) join message. Therefore, the centralized NIMS is a weakness too, and the number of forwarding states in inter-domain routers grows with the number of groups.

To demonstrate the effectiveness of DINloop, we have used video distribution as an application example for DINloop [10]. In addition, the DINloop has also been implemented within a peer-to-peer (P2P) overlay network and results in a hop efficient dynamic multicast infrastructure [11]. The DINloop in [11] is suitable to optimize a single application level multicast tree. This paper extends the DINloop concept to inter-domain network layer multicast using MPLS.

3. Solution overview

In DINloop-based inter-domain multicast, we separate the control plane for multicast group membership management and the data plane for the inter-domain multicast traffic. First, we use MPLS to set up an optimal DINloop in the core network, and then multiple DIN Nodes exchange multicast group membership via DINloop. Second, with the facilitation of DINloop, multiple DIN Nodes which are involved in a particular multicast group can easily form a Steiner tree for multicast traffic. Furthermore, traffic for different multicast groups which share a same path are aggregated together.

Compared with other inter-domain multicast protocols, our solution has below distinct advantages:
- DINloop for membership management removes the burden on a single multicast manager and avoids a single point of failure.
- DINloop also simplifies the setting up procedure of Steiner tree.
- Only DIN Nodes have multicast forwarding state and other core routers do not have it, and this correspondingly reduces tree maintenance overhead.
- DINloop-based multicast consumes fewer labels with label stacks so as to improve routing scalability, as well as reduce the routing look-up time in the core network.
- The inter-domain multicast delay in DINloop-based multicast is small.

4. DINloop-based multicast

The control modules in DIN Nodes are shown in Fig. 1. Intermediate System-to-Intermediate System (IS-IS) and Open Shortest Path First (OSPF) are the routing protocols and Resource Reservation Protocol - Traffic Extension (RSVP-TE) and Constraint-based Label Distribution Protocol (CR-LDP) are the signaling protocols for the establishment of LSPs. We use CR-LDP for label distribution, and extend it to support explicit routing and resource reservations.

![Figure 1. Control modules in DIN Node](image)

In the following sub-sections, we describe the details of setting up DINloop for membership control plane, setting up Steiner Tree for multicast traffic, and assigning labels to multicast packets.

4.1. Setting up DINloop for membership control plane

To elaborate, one example of DINloop is shown in Fig. 2. DIN Nodes A, B, C, D and E are core routers and function as RPs for the associated intra-domains (e.g., circle areas) respectively. DIN Nodes A, B, C, D...
and E form the DINloop (thick arrow line in Fig. 2) in the core network.

![DINloop Diagram](image)

**Figure 2.** One example of DINloop

#### 4.1.1. Finding an optimal DINloop.

Initially, DIN Node 0 and DIN Node 1 form a ring. Then DIN Node 2 is added to the ring. From DIN Node 3 onwards, the two nearest DIN Nodes will break the link and the new DIN Node is added.

Based on the steps described above, further optimization can be done. Assume DIN Node \(a\) is the new DIN Node (Fig. 3a). The value on the arc in Fig. 3 is the relative RTT value.

![Optimization Diagram](image)

**Figure 3.** Optimization of DINloop

DIN Node \(b\) has minimum RTT \((1)\) to DIN Node \(a\). DIN Node \(c\), which is adjacent to DIN Node \(b\) in the loop is selected because it has lower RTT \((4)\) to DIN Node \(a\). We use the above algorithm to form a loop (Fig. 3b). The loop goes through \(e\rightarrow b\rightarrow a\rightarrow c\). Assume \(xy\) denotes the RTT from DIN Node \(x\) to DIN Node \(y\). However, after summing the RTT \((eb+ba+ac)\) and \((ea+ab+bc)\), we find that \((eb+ba+ac)\) is larger than \((ea+ab+bc)\).

So we further optimize the DINloop. If \((eb+ba+ac) > (ea+ab+bc)\), then the optimal loop goes through \(e\rightarrow a\rightarrow b\rightarrow c\) (Fig. 3c). In this way, we obtain the minimum DINloop delay so that DINloop not only facilitates the fast membership exchanging, but also simplifies the setting up procedure of Steiner tree.

#### 4.1.2. Creating DINloop with LSP.

An explicitly routed LSP is used, that each DIN Node chooses the adjacent DIN Node in the DINloop as the next hop and puts the information in the NHLFE (Next Hop Label Forwarding Entry).

Fig. 4 shows the LSPs for DINloop. We assign the same label to all membership control packets. When forwarding the labelled control packet, the DIN Node examines the label and maps this label to the NHLFE. Using the information in the NHLFE, it forwards the membership control packet to the neighbor DIN Node in the DINloop. If two adjacent DIN Nodes in the DINloop are not the adjacent LSRs (Label Switching Routers), then a LSP tunnel is built up between two adjacent DIN Nodes.

After setting up the DINloop, multiple DIN Nodes exchange the multicast group membership through the DINloop so that all DIN Nodes know the source and multicast group members.

![LSPs Diagram](image)

**Figure 4.** LSPs for DINloop

#### 4.2. Setting up Steiner tree for multicast traffic

##### 4.2.1. Steiner tree.

After exchanging the group membership through the DINloop, multiple DIN Nodes, which are involved in the particular multicast group, form a Steiner tree using Steiner tree-based heuristic routing algorithm. This Steiner tree is formed using an explicit join method. The formation method of Steiner tree can be illustrated in Fig. 5.

In Fig. 5, DIN Node \(E\) is in the source domain, called as the source DIN Node, and DIN Nodes \(A, B, C,\) and \(D\) participate in the multicast group since they have receivers in their associated intra-domains respectively. Through the DINloop control plane, DIN Nodes \(A, B, C, D\) and \(E\) know of each other. The join procedure is enumerated as follows.

1. DIN Node \(A\) and DIN Node \(D\) (two neighbor DIN Nodes of the source DIN Node \(E\) in the DINloop) send explicit Join messages to the source DIN Node \(E\) respectively. DIN Node \(E\) creates a routing entry for DIN Node \(A\) and DIN Node \(D\) respectively.

2. DIN Node \(A\) informs neighbor DIN Node \(B\) about its RTT to the source DIN Node \(E\) (assigned as RTT\(_{AE}\)). DIN Node \(B\) then only needs to check the RTT to its neighbor DIN Node \(A\) in the DINloop (assigned as RTT\(_{BA}\)) since the delay between two neighbor DIN Nodes is minimum and the source DIN Node \(E\) (assigned as RTT\(_{BE}\)) respectively. Then DIN Node \(B\) gets the smaller RTT to DIN Node \(E\) since
RTT_{BE} < (RTT_{BA} + RTT_{AE}). Therefore, DIN Node B sends an explicit Join message to DIN Node E, and DIN Node E creates a routing entry for DIN Node B.

Similarly, DIN Node D informs DIN Node C about its RTT to the source DIN Node E (RTT_{DE}), and DIN Node C checks the RTT to its neighbor DIN Node D in the DINloop (assigned as RTT_{CD}) and the source DIN Node E (assigned as RTT_{CE}) respectively. Then DIN Node C sends an explicit Join message to DIN Node D since (RTT_{CD} + RTT_{DE}) \approx RTT_{CE}. DIN Node D creates a routing entry for DIN Node C.

(3) Step 2 is repeated until all participating DIN Nodes are in the Steiner tree.

Figure 5. Steiner tree formation method

4.2.2. Traffic aggregation. After each Steiner tree is created for each group, if the DIN Nodes in the two ends of paths in the different Steiner trees are the same, the multicast traffic for different multicast groups are aggregated.

For example, in Fig. 6, there are two Steiner trees for Group 1 and Group 2. There are two paths with the same two end DIN Nodes, i.e., from DIN Node E to DIN Node B and from DIN Node E to DIN Node D. So the traffic in these two paths are aggregated as one respectively.

Traffic aggregation is useful to reduce the number of labels to be handled in the other core LSRs so as to avoid a large size for the routing table and gain the speed in table look-ups, and thus improve the routing scalability.

Figure 6. Traffic aggregation

4.3 Assigning label to multicast packets

When a multicast packet arrives at the ingress DIN Node, the Packet Processing Module (referred to Fig. 1) is activated. The Packet Processing Module looks the IP header of the multicast packet and identifies the source address and destination address (D class for multicast). Then the Packet Processing Module reports the information to the MPLS Manager. The MPLS Manager determines the packet’s FEC and looks up the Label Table to assign a label stack with 2-level labels. The top label corresponds to the shared path with the same two end DIN Nodes and the bottom label corresponds to the destination address to differentiate the multicast packets. In order to transmit the label stack along with the packet, the label stack is encoded as a “shim” between the data link layer and network layer headers. Then the multicast packet is fast forwarded in the core network using the top label rather than address matching to determine next hop.

When DIN Node receives the packets, it pops the top label off the label stack and uses the bottom label to examine the multicast packet.

5. Experimental results

This section comprises three parts to compare DINloop-based multicast with the conventional inter-domain multicast protocols. First, we compare the message load needed to form the multicast structure. Second, we compare the forwarding state size kept in the core routers. Finally, we examine the performance in the inter-domain multicast delay.

5.1. Message load

We use the message load to evaluate the performance of DINloop-based multicast versus
conventional inter-domain multicast protocol, i.e., MSDP. The message load denotes the amount of signaling messages that is needed to form the multicast structure. The message load metric is incremented by one whenever a signaling message passes a link.

The simulations run on the network topologies which are generated using the Georgia Tech [12] random graph generator according to the transit-stub model [13]. We use the graph generator to generate different network topologies. The number of nodes in the transit domain that connect to stub domains is fixed, i.e., 25 means 25 DIN Nodes for 25 domains. The number of nodes in stub domains is varied from 500 to 6000 that spread across 25 domains. The number of sources is 25 and one domain has one source. Each DIN Node is the RP for its associated stub domain respectively.

The results are shown in Fig. 7, where it is clear that DINloop-based multicast uses less message load to form the multicast structure than MSDP. Correspondingly, DINloop-based multicast reduces tree maintenance overhead at the core network. As the node number increases, the message load increases linearly for both protocols.

5.2. Inter-domain router forwarding state size

In this sub-section, the forwarding state size kept in the core routers is compared with MSDP.

In MSDP, the core router looks up the IP header of every packet and identifies the destination address. Then, the core router looks up the routing table and forwards the packet to the correct interface (Fig. 8). Therefore, the routing table size is increased linearly with the number of multicast groups.

In DINloop-based multicast, only DIN Nodes keep the multicast forwarding state and the other core routers do not keep the multicast forwarding state. Furthermore, the forwarding state is aggregated if the traffic pass through the same path. All multicast groups in the same path are assigned to the same label (Label A in Fig. 9) in the top of the label stack. Each label has one egress interface, i.e., Label A corresponds to Interface 1 in the routing table.

![Figure 8. Multicast routing table size in MSDP](image)

![Figure 9. Routing table size in other core routers in DINloop-based multicast](image)

From Fig. 9, we obtain the results that the routing table sizes in other core routers do not increase as the number of multicast group increases, and therefore the DINloop-based multicast increases the routing scalability for inter-domain multicast, as well as reduces the routing look-up time for fast routing, at the price of adding additional MPLS header to the multicast packets.

5.3. Inter-domain multicast delay

In this sub-section, we compare the multicast delay of the different types of inter-domain multicast structures constructed by MSDP and DINloop-based multicast. MSDP constructs a unidirectional shared tree, while DINloop-based multicast constructs a Steiner tree. Inter-domain multicast delay is made up of delay from sources sending out data to receivers receiving the data.
We use the graph generator to generate different network topologies. The number of nodes in the transit domain that connect to stub domains is varied, i.e., from 1 to 25 that means domain number changes from 1 to 25. The number of nodes in stub domains is fixed at 6000, and is spread across all domains. The legend in Fig. 10 is explained as below.

\[
\text{Unidirectional Tree (max)} = \frac{\text{Maximum Delay in Unidirectional Tree}}{\text{Maximum Delay in Shortest Path Tree}}
\]

\[
\text{Unidirectional Tree (ave)} = \frac{\text{Average Delay in Unidirectional Tree}}{\text{Average Delay in Shortest Path Tree}}
\]

\[
\text{DINloop (max)} = \frac{\text{Maximum Delay in DINloop Based Multicast}}{\text{Maximum Delay in Shortest Path Tree}}
\]

\[
\text{DINloop (ave)} = \frac{\text{Average Delay in DINloop Based Multicast}}{\text{Average Delay in Shortest Path Tree}}
\]

![Multicast Delay Comparison (6000 nodes)](image)

**Figure 10.** Delay ratio comparison on the effect of domain number

From Fig. 10, the delay ratio for the unidirectional tree is more than 1.4, and the delay ratio for DINloop-based multicast is less than 1.2. Thus, the inter-domain multicast delay in DINloop-based multicast is lower than that in the unidirectional tree. The multicast delay in DINloop-based multicast is less than 20% higher than that of the shortest path tree.

### 6. Conclusion

To overcome the existing problems, DINloop-based multicast with MPLS is presented to optimize inter-domain multicast. In the DINloop-based multicast, we separate the control plane for multicast group management and the data plane for the inter-domain multicast traffic. First, we use MPLS to set up the DINloop in the core network by establishing LSPs through the DIN Nodes. Then multiple DIN Nodes exchange multicast group membership via the DINloop. The DINloop for membership management removes the burden on a single centralized multicast manager and avoids a single point of failure. Second, with the facilitation of the DINloop, multiple DIN Nodes which are involved in a particular multicast group easily form a Steiner tree using Steiner tree-based heuristic routing algorithm for multicast traffic. Furthermore, the traffic for different multicast groups sharing the same path are aggregated through assigning the same label at the top of the label stack. Compared with MSDP, simulations demonstrate that DINloop-based multicast has a number of advantages in terms of message load, the routing table size in the other core routers and the multicast delay.

### References


