NAIST-IS-DT9961033

**Doctoral Thesis**

# Sender-Initated Multicast for small group communications

Vasaka Visoottiviseth


Department of Information System
Graduate School of Information Science
Nara Institute of Science and Technology

February 7, 2003

Doctoral Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Vasaka Visoottiviseth

Thesis Committee: Suguru Yamaguchi, Professor
Heiichi Yamamoto, Professor
Hideki Sunahara, Professor
Youki Kadobayashi, Associate Professor

# Abstract:

## Sender-Initiated Multicast for small group communications

Vasaka Visoottiviseth

Current IP multicast offers efficient multipoint-to-multipoint data delivery for large group communications. Nevertheless, it suffers from deployment issues such as configuration complexity, lack of sufficient group management, and global address allocation. IP multicast also suffers from a scalability problem because the router has to maintain forwarding states for all multicast distribution trees passing through it. Thus, the number of forwarding entries increases with the number of groups. While IP multicast was designed for large group communications, we can learn from various statistical studies on how multicast communications are under way, which are publicly available on many web pages, that most multicast sessions presently are relatively small and consist less than 50 receivers. Therefore, supporting small multicast group should be appropriate for the present applications.

In this dissertation, we propose Sender-Initiated Multicast (SIM) as an alternative multicast forwarding scheme for small group communications such as teleconferencing and file distribution. SIM eliminates the cost of allocating global multicast address, by routing packets according to receiver unicast addresses attached to packet headers. The key feature of SIM is in its Preset mode, which can lessen the costs of route lookups and provides cost-efficient packet forwarding by using a SIM Forwarding Information Base (FIB) maintained on routers. Moreover, a SIM tunnel will be automatically created between two routers that act as multicast branching points. Thus, SIM can gain scalability by maintaining FIB entries only on the branching routers.

Furthermore, all multicast applications today use the connectionless and unreliable protocol such as the User Datagram Protocol (UDP) to achieve multicast communications. However, developing a reliable protocol over UDP is as complicated as developing TCP. Moreover, the existing reliable multicast protocols have been designed for general proposes, while different levels of reliability are required by multicast applications.

In order to achieve reliability, we also propose the Multicast extension to Transmission Control Protocol (M/TCP), a TCP extension that enables multicast transmission to the existing TCP applications. M/TCP is designed for small multicast groups based on SIM. M/TCP has the following features. First, M/TCP can be applied to those applications that a sender triggers data transfer (sender-initiated), for example, FTP and SMTP. Second, it requires implementation only on the sender. Third, it provides multiple transmission rates by re-classifying receivers in multiple multicast groups.

In this dissertation, I describe the SIM mechanism in detail and present evaluated results through implementation and simulation results. I show how SIM can achieve low cost of maintaining forwarding information, cost-efficient packet forwarding, and incrementally deployment. Moreover, I also implement M/TCP on FreeBSD kernel and evaluate the performance from the implementation. I show through the evaluation that M/TCP can gain better performance than unicast in proportion to the number of receivers. Further, M/TCP is not limited to SIM, but also can be used with other network protocols, such as XCAST, which the receivers are explicitly known by the sender.

**keywords:** Multicast routing, small group communication, explicit multicast, processing overhead reduction, reliable multicast

# 論文概要

## 少人数グループを対象としたマルチキャスト通信機構の開発

IP マルチキャストは、インターネット上のグループ通信を実現し、かつ、効率の良いデータ配送を行う機構として開発されてきた。しかし、長年の研究開発にもかかわらず、IP マルチキャストの普及は進んではいない。この理由として、IP マルチキャスト機構で使われている経路制御におけるプロトコルの複雑さ、運用コストの高さ、さらに、ルータに対する高負荷などを挙げることができる。また、現在の IP マルチキャストは、単一のグループに属するメンバー数についての良好なスケーラビリティを確保することについては研究開発が進められてきたが、グループ数そのものについてのスケーラビリティの検討は少なく、結果として、多数の少人数マルチキャストグループが存在する場合、現在の IP マルチキャスト機構ではうまく機能しないことがわかっている。ところが、これまでに公開された幾つかの統計結果から分かるように、現在のマルチキャストの利用状況では、50 人を超えない程度の少人数間通信が大部分を占める。このことから、IP マルチキャストにおいて、サポート可能なグループ数という面でのスケーラビリティ確保は必須である。

本研究では，少人数におけるグループ間通信に注目し，Sender-InitiatedMulticast (SIM)　という新たなマルチキャスト経路制御プロトコルを提案する．SIM は，Explicit Multicast (XCAST) と呼ばれるパケット転送方式を利用し，受信者アドレスリストをパケットに添付し，ユニキャスト経路情報に基づくパケット転送を行なう．また，XCAST の問題点であるルータ負荷を軽減するために，ルータ上に SIM Forwarding Information Base (FIB) として転送情報を保持する Preset モードを備えている．そのため，全パケットにアドレスリストを添付する必要がない．Preset モードでは，送信者アドレスとグループのマルチキャストアドレスの組を転送情報検索のハッシュキーとして利用することで，効率のよい経路表検索を実現している．また，自動的に分岐点ルータ間を結ぶ SIM トンネル機構により，転送情報の保持と全受信者に対する経路表検索を行うルータ数を減らし，パケット転送処理の高速化を図る．

また，現在の IP マルチキャストではグループに対する配送のため，TCP のような End-to-End のコネクションを確保するような配送ができず，信頼性を保証できない UDP のみで配送する可能である．そこで，本研究では多く使われている信頼性を保証

するプロトコルである TCP に注目し，SIM を用いた信頼性を保証するトランスポート層プロトコルである Multicast-extension to Transmission Control Protocol (M/TCP) を提案する．SIM は送信者が受信者アドレスリストを管理するという特徴があるため，TCP のように個々の受信者にコネクションを確立し，個々の通信状態に応じた通信を実現することが可能である．M/TCP は既存の TCP の拡張であるため，TCP と親和性の高く既存のネットワークに悪影響を与えない通信を実現でき，受信側では既存の TCP アプリケーションをそのまま利用できるという利点がある．

本論文では，SIM の有効性を証明するために，シミュレーションと FreeBSD 上の実装を用いて，SIM の総合的な評価を行なった．その結果として導入コストの高さが問題とされている小規模グループでのマルチキャスト配送においても XCAST 方式によるパケット処理効率の向上を確認した．また，M/TCP においては，UNIX カーネルに実装し，評価を行った．その結果，ユニキャスト通信に比べて受信者数が増加にするつれて，本提案方式が有効なことを確認できた．また，実装ではマルチキャスト経路制御プロトコルとして SIM を利用したが，同様に送信者が受信者アドレスを把握しているという特性を持っている XCAST を利用することが可能である．

**キーワード:** 小人数マルチキャスト、ルーティング、スケラビリティ、信頼性マルチキャスト、送信者指定型マルチキャスト、明示型マルチキャスト

# Acknowledgements

I wish to express my gratitude to professor Suguru Yamaguchi, my advisor and committee chairman, for his support, advice and encouragement. Without his kind cooperation, I could not complete this dissertation.

I would also like to thank professor Heiichi Yamamoto for his helpful support on my committee. His useful comments helped me to finish my work.

I wish to thank professor Hideki Sunahara for his long sustained support to my work. His kind support encouraged me to make my research a fruitful one.

My special thanks go to associate professor Youki Kadobayashi for his continuous support and assistance with his great patience. He read my research papers again and again, and gave me uncountable invaluable comments.

I also thank assistant professor Katsuyoshi Iida and assistant professor Takashi Okuda for their friendly assistant. Their continuous stimulation pushes me to the goal.

I also would like to thank all members of IPLAB for their friendships, supports and comments. Graduate school and life in general would not be the same without them.

I gratefully acknowledge Noritoshi Demizu for his kindness and his support. This work would not begin without him. Also, Yosuke Takahashi and Takuya Mogami for their great helps to the initial development. I further thank Hiroyuki Kido for his contribution to the simulator, also his constant caring, support, and encouragement.

I am grateful to my Thai friends and my old Japanese friends in Tokyo. They brush away the loneliness and always push me to the goal. I really appreciate their long-lasting friendships.

Finally, I wish to thank my family who always be there for me and cheer me up every time I am exhausted and want to give up. Their sincere encouragement, their loves and support enable me to go through my degree.

# Contents


**1 Introduction** **1**
- 1.1 Motivation . . . 1
- 1.2 Contributions . . . 3
- 1.3 Dissertation Overview . . . 4

**2 Related Work** **5**
- 2.1 IP Multicast . . . 5
  - 2.1.1 IP Multicast Components . . . 5
  - 2.1.2 Current IP Multicast . . . 11
  - 2.1.3 IP Multicast Deployment Issues . . . 12
- 2.2 Single Source Multicast . . . 13
- 2.3 Explicit Multicast . . . 14
- 2.4 Recursive Multicast . . . 16

**3 Sender-Initiated Multicast** **19**
- 3.1 Introduction . . . 19
- 3.2 Overview of SIM . . . 21
  - 3.2.1 Design Goal . . . 21
  - 3.2.2 Target Applications . . . 21
  - 3.2.3 Overview of SIM Forwarding Techniques . . . 22
- 3.3 SIM Packet Header . . . 23
- 3.4 SIM Membership Management and Forwarding Process . . . 24
  - 3.4.1 SIM Membership Management . . . 24
  - 3.4.2 Allocating a multicast address . . . 27
  - 3.4.3 Forwarding Algorithm . . . 27
  - 3.4.4 Conversion to Unicast . . . 29
- 3.5 SIM Tree Maintenance . . . 29
  - 3.5.1 Management of SIM FIB . . . 30
  - 3.5.2 SIM Tunnel and SIM Redirect Message . . . 31
- 3.6 Implementation . . . 34
  - 3.6.1 SIM Header Field . . . 34
  - 3.6.2 The SIM Forwarding Information Base . . . 37
  - 3.6.3 User level - Kernel Interfaces . . . 38

3.6.4 Kernel Support . . . 41
3.7 Preliminary Evaluation . . . 44
3.8 Performance Evaluation Methods: simulation and experiment . . . 46
3.9 Performance Evaluation . . . 48
3.9.1 Application Level Evaluation . . . 48
3.9.2 Network Protocol Level Evaluation . . . 48
3.10 SIM Deployment . . . 57
3.10.1 Multiple Receivers in LAN . . . 57
3.10.2 SIM Gradual Deployment . . . 58
3.11 Security Considerations . . . 62
3.11.1 SIM FIB attacks . . . 62
3.11.2 Generating a DoS attack . . . 62
3.11.3 Using IPsec with SIM . . . 65
3.12 Summary . . . 66

**4 Multicast extension to Transmission Control Protocol 69**
4.1 Introduction . . . 69
4.2 Reliable Multicast Transport Protocols . . . 70
4.2.1 Building Blocks for Reliable Multicast Transport . . . 70
4.2.2 Multicast Congestion Control . . . 72
4.3 Overview of M/TCP . . . 73
4.3.1 Design Goals . . . 73
4.3.2 Target Applications . . . 73
4.3.3 Advantages and Disadvantage of M/TCP . . . 74
4.4 Transmission Procedure in M/TCP . . . 75
4.5 Congestion Control in M/TCP . . . 76
4.5.1 Retransmission . . . 77
4.6 Implementation . . . 78
4.6.1 M/TCP Header Format . . . 78
4.6.2 Multicast Group Membership . . . 78
4.6.3 Co-operation with SIM . . . 79
4.6.4 Extension to Protocol Control Blocks . . . 80
4.6.5 Transmission by M/TCP . . . 81
4.6.6 Kernel Support for M/TCP senders . . . 83
4.7 Performance Evaluation . . . 84
4.8 Discussions . . . 85
4.8.1 Packet Overhead . . . 85
4.8.2 ACK implosion . . . 86
4.9 Summary . . . 87

**5 Conclusions and Future Work 89**
5.1 Conclusions . . . 89
5.2 SIM Future Work . . . 90
5.3 M/TCP Future Work . . . 90

5.4 Future of Multicast . . . . . 91

**A List of Publications** **101**
A.1 Journal . . . . . 101
A.2 International Conference . . . . . 101
A.3 Internet Draft . . . . . 101
A.3.1 Domestic Conference . . . . . 102

# List of Figures


1.1 Comparison of protocol components for Unicast, IP multicast, and SIM 3

2.1 Overview of MALLOC architecture . . . . . 8
2.2 GLOP address . . . . . 8
2.3 Three dimensions of data delivery costs . . . . . 15

3.1 Three dimensions of data delivery costs . . . . . 21
3.2 SIM forwarding process and the bitmap . . . . . 23
3.3 SIM packet header and option header fields . . . . . 24
3.4 SIM Packet Header . . . . . 25
3.5 Group Management Server for many-to-many communication . . . . . 26
3.6 Communication messages between GMS and a member . . . . . 27
3.7 Flowchart of SIM packet forwarding algorithm . . . . . 28
3.8 Multicast Tree Construction . . . . . 29
3.9 SIM FIB management . . . . . 31
3.10 New branching node by newly join receiver . . . . . 32
3.11 SIM Redirect message and SIM Tunnel for capable new branching node 33
3.12 SIM Redirect message for overloaded node . . . . . 34
3.13 Jump flag processing . . . . . 35
3.14 SIM redirect header fields . . . . . 36
3.15 SIM FIB entry overall structure at kernel level . . . . . 38
3.16 Sysctl . . . . . 39
3.17 Overoview of SIM data transferring process . . . . . 41
3.18 A sample of sending program . . . . . 42
3.19 Forwarding process on a router . . . . . 43
3.20 Topology used in validation of the simulator and the implementation . . 47
3.21 End-to-end delay for SIM, and Xcast4 comparing to Unicast . . . . . 49
3.22 Number of receivers and sending packets for sending rate 512 Kbps . . . 50
3.23 Number of receivers and sending packets for sending rate 2048 Kbps . . 51
3.24 Routing table search and total number of receivers . . . . . 52
3.25 Routing table search and number of groups . . . . . 53
3.26 Routing table size on a backbone router with the growth in number of groups . . . . . 56
3.27 Deployment situation of SIM . . . . . 58

3.28 Bus topology . . . . . 59
3.29 Tree topology . . . . . 60
3.30 Total routing table lookups in Bus topology, while only one SIM router exists . . . . . 61
3.31 Total routing table lookups in Tree topology, while only one SIM router exists . . . . . 62
3.32 Total routing table lookups in Bus topology, while number of SIM routers increases . . . . . 63
3.33 Total routing table lookups in Tree topology, while number of SIM routers increases . . . . . 64
3.34 Star topology . . . . . 65

4.1 Transmisison in M/TCP . . . . . 76
4.2 M/TCP Header Format . . . . . 79
4.3 SIM Options for M/TCP . . . . . 79
4.4 M/TCP TCP_ADD . . . . . 80
4.5 Relationship of SIM and M/TCP Packet Header . . . . . 80
4.6 Extension to Protocol Control Blocks . . . . . 81
4.7 An example of M/TCP sender program . . . . . 82
4.8 Topology of Experiment 1 . . . . . 85
4.9 Number of receivers and average forwarding time . . . . . 86
4.10 Topology of Experiment 2 . . . . . 87

5.1 Future of multicast technology . . . . . 92

# List of Tables


2.1 Categories of Multicast Applications . . . . . 6
2.2 Multicast Addresses . . . . . 6
2.3 Working groups at IETF and IRTF . . . . . 11
2.4 Current multicast-enabled Autonomous Systems . . . . . 12

3.1 Multicast, Current State . . . . . 19
3.2 Groups With Most Receivers (Last 1 Hour) . . . . . 20
3.3 Comparison of IP multicast, Basic Xcast, and SIM . . . . . 45
3.4 Total number of routing table lookup observed from the simulations and experiments . . . . . 47
3.5 Average forwarding time of SIM and Xcast on branching routers ($\mu$s) . . 51
3.6 Parameters observed from the simulation . . . . . 54
3.7 Performance Evaluation results of SIM, IP multicast, and Xcast4 . . . . 57
3.8 Average forwarding time on a branching router in Star topology ($\mu$s) . . 66

4.1 Protocol Families proposed in the RMT building block . . . . . 71
4.2 Relationship of delay and forwarding time . . . . . 85

# Chapter 1

# Introduction

## 1.1 Motivation

IP multicast [1] was first proposed in 1988 to provide efficient data delivery to a large number of receivers. Multicast can reduce network bandwidth consumption than multiple point-to-point links. Applications that can benefit from multicast include the applications with multiple receivers, for example, Internet TV, teleconferencing, network games, content delivery, software distribution, and distance learning. In spite of decades of research, IP multicast is still not widely deployed throughout the Internet.

In order to understand the prerequisite for deploying IP multicast, we must consider the characteristics of multicast applications and network service costs. As mentioned in [2], the current set of applications driving multicast deployment falls into four categories: (1) audio and video distributions including webcasting; (2) push applications; (3) audio and video conferencing and group collaboration applications; and (4) file transfer, which involves sending data from one location to one or more locations. The first two categories are applications for large group communications, and the third and fourth categories are applications for small group communications. While IP multicast was designed for large group communications, most multicast sessions presently are relatively small, and consist less than 50 receivers [3]. Thus, supporting small multicast group is appropriate for present applications.

Regarding the cost of a network service, Diot, et al. noted that the cost of a network service can be defined as the sum of network-related costs and management costs [2]. Network-related costs consist of router state processing, signaling, and inter-domain routing scalability. Management costs include ease of deployment and maintenance in terms of human resources and infrastructure. In terms of these costs, multicast services are currently more expensive than unicast services. Cost per user for unicast increases linearly as new receivers are added to network costs. On the other hand, IP multicast has a high initial cost, but the cost per user should ideally decline as the number of new receivers increases. The *sweet spot* of IP multicast deployment suggested by Cain and described in [2] is where the additional cost of providing the service is out-weighed by the gain in performance.

Because of this *sweet spot*, despite many drawbacks, we still believe that IP multicast is appropriate for large group communications. However, for small multicast communications, cost per user for IP multicast is relatively high. In particular, when IP multicast has to deal with a large number of small group communications, and receivers are widely distributed on the Internet, non-branching point routers must also maintain forwarding states in order to construct the multicast distribution tree. Consequently, the network-related costs, such as state-and-signal processing on routers, will increase, and introduce scalability problems. It would also consume network resources, management costs and maintenance costs for routers.

Concerning scalability, many approach on aggregation multicast states on routers have been proposed [4] [5] [6]. Thaler and Handley propose an interface-centric data structure model which allows aggregation of ranges of multicast addresses in the forwarding table [4]. Rodoslavov *et al.* proposed algorithms to aggregate forwarding state, and studied the bandwidth-memory tradeoff through simulations in [5]. However, it was argued in [6] that both these works attempted to aggregate routing state after the distribution trees have been established, and this implementation is very complex. Cui, *et al.* propose a protocol called *Aggregate Source Specific Multicast (ASSM)* to improve the state scalability of Source Specific Multicast in [6]. The tradeoff of their approach would be the extra bandwidth in delivering multicast data to non-group-member nodes.

While there is no *one-size-fits-all* solution, in this dissertation, we focused on supporting small group communications in order to satisfy the present multicast usage and to promote a scalable deployment of multicast on the Internet. We proposed *Sender-Initiated Multicast (SIM)* as a protocol for small group communications. Small groups are groups consisting of about 10 to 100 people, the typical size for video conferencing, collaborative applications, and distance learning. SIM is a one-to-many multicast protocol, but it can also support many-to-many applications by using multiple multicast sessions. It implements multicast distribution through receiver unicast addresses attached to packet headers, the same approach used in Explicit Multicast (Xcast) [7]. Xcast eliminates state-and-signaling costs by requiring Xcast senders to attach a receiver address list to every packet, and by performing unicast route look-ups on all routers on the multicast distribution tree. SIM aims to reduce state-and-signaling costs, which occur on IP multicast routers, by adopting the packet forwarding concept from Xcast. At the same time, it reduces overheads of packet processed occurring on Xcast routers.

To achieve reliability, we also propose the Multicast extension to Transmission Control Protocol (M/TCP), a TCP extension that enables multicast transmission on existing TCP applications. The intention of developing M/TCP is the current lack of efficient reliable multicast protocol. Today, all multicast applications use the connectionless and unreliable protocols, such as User Datagram Protocol (UDP), to achieve multicast communications. However, developing reliable UDP is as complicated as developing TCP. Further, since most traffic on the Internet is TCP traffic, supporting multicast to TCP would be more challenging. M/TCP is designed based on SIM, our multicast routing protocol. M/TCP has the following features. First, it can be applied

to those applications that a sender triggers data transfer (sender-initiated), for example, FTP and SMTP. Second, it requires implementation only by the sender. Third, it provides multiple transmission rates by re-classifying receivers into multiple multicast groups.

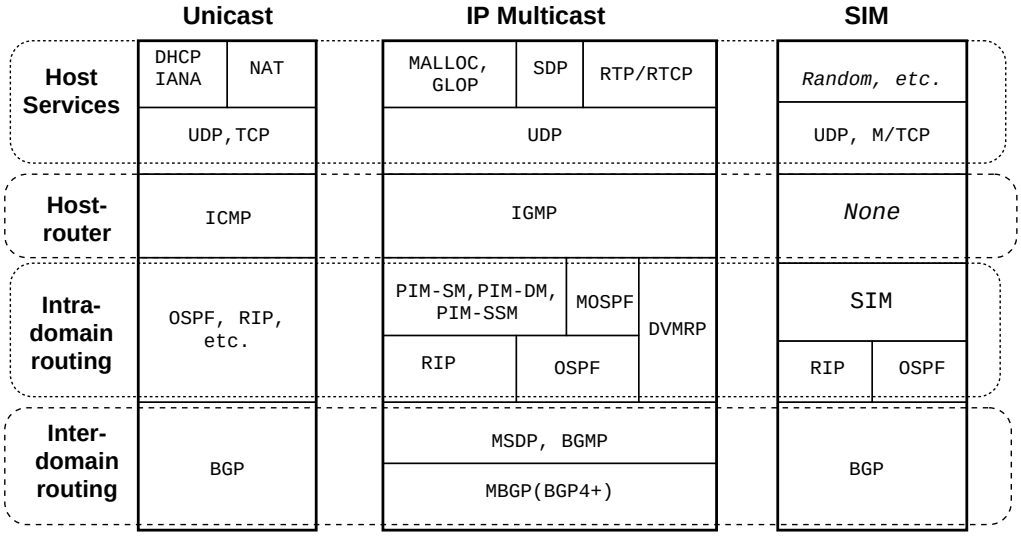



Figure 1.1: Comparison of protocol components for Unicast, IP multicast, and SIM

Figure 1.1 shows a comparison of protocol components for unicast, IP multicast, and SIM architecture. Note that this figure is the extension of *Figure 1. A comparison of protocol components for IP unicast and IP multicast architectures* in [2].

In this dissertation, I described the SIM mechanism in detail and presented evaluated results through implementation and simulation results. I show how SIM can achieve low cost of maintaining forwarding information, cost-efficient packet forwarding, and incremental deployment. Moreover, we also implement M/TCP on FreeBSD kernel and evaluate the performance of the implementation. I show through the evaluation that M/TCP has better performance than unicast in proportion to the number of receivers. Further, the usage of M/TCP is not limited to SIM but can also apply to other network protocols, such as XCAST, which the receivers are explicitly knows by the sender.

## 1.2 Contributions

Supporting a wide-scale deployment of multicast technology is the subject of this dissertation. Design and implementation of SIM and M/TCP is one step toward this goal. The primary contributions of this dissertation are as follows:

1. A scalable multicast routing protocol supporting small group communications

As we will see in Chapter 2, many deployment issues are left for the multicast routing, for example, multicast access control, multicast address allocation, security vulnerability in inter-domain routing. A problem we specially focus on in this dissertation is scalability in the number of groups. We designed, simulated, and developed a prototype of Sender-Initiated Multicast (SIM) to support small group communications. SIM does not require global multicast address allocation and does not require special multicast inter-domain routing protocol. Access control can be supported because the sender host totally controls the receiver list in our model.

2. A reliable multicast by supporting the existing TCP applications

   Many reliable multicast protocols were proposed and simulated, but a few of them that have been implemented. Almost all of them are designed based on an unreliable protocol such as UDP. To support a large number of applications and to provide complete reliability to multicast, we extend the multicast function to the existing TCP model. By using TCP, we can guarantee data delivery to all receivers, guarantee the ordering in received data, and discard duplicated data. The current TCP applications that have no needs to support real-time data transmission can also have the multicast benefit by using our protocol.

3. Promotion of multicast deployment by providing the prototypes of SIM and M/TCP

   To support a wide-scale deployment of multicast, we also designed, implemented, simulated, and evaluated SIM and M/TCP. With the current prototype, we only need to modify routers and the sender node. Therefore, users can gain the benefit of multicast without any software modification on receiver nodes.

## 1.3 Dissertation Overview

This dissertation is organized as follows. In the next chapter, I survey related work in fields of multicast routing protocol and multicast reliable transport protocol.

Chapter 3 describes the SIM mechanism in detail and presents evaluated results through implementation and simulation results. The SIM design goal is not to replace the conventional multicast protocol, but to provide SIM as an alternative subset of an entire multicast protocol suite. I show how SIM can achieve low cost of maintaining forwarding information, cost-efficient packet forwarding, and incremental deployment.

Chapter 4 presents the design of M/TCP. We also implemented M/TCP on FreeBSD kernel and evaluated the performance of the implementation. I show through the evaluation that M/TCP has better performance than unicast in proportion to the number of receivers. Furthermore, M/TCP is not limited to SIM. It also can be used with other network protocols, such as XCAST, when the receivers are explicitly known by the sender. Finally, Chapter 5 suggests the future direction of research and concludes this dissertation.

# Chapter 2

# Related Work

Depending on how packets are routed, the present multicast scheme can be divided into four categories: IP multicast, the single source multicast, the explicit multicast, and the recursive multicast. As relates SIM, our multicast routing protocol, this chapter introduces these multicast protocols, summarizes their advantages, and explores the issues that need to be resolved.

## 2.1 IP Multicast

IP multicast or *Any-Source Multicast (ASM)* [1] is a traditional multicast technology proposed by Steve Deering in 1988. It provides efficient delivery of data to a large communication group. Applications that can use delivery through multicast technology can be categorized into two categories: (1) one-to-many applications, and (2) many-to-many applications. One-to-many applications are the applications that a single host sends to two or more $n$ receivers. Many-to-many applications are the applications that any number of hosts sends to the same multicast group address, as well as receiving from it. Table 2.1 shows the example of multicast applications and their categories.

### 2.1.1 IP Multicast Components

The core concept of IP multicast is that of the *host-group* model. A packet destined for a multicast group address will be delivered to all hosts joined to the same group. Delivering multicast data requires the support of the sender and receiver hosts and also the support of network components. In order to delivery multicast data, there are four network components: (1) multicast addressing, (2) multicast group membership, (3) intra-domain multicast routing protocol, and (4) inter-domain multicast routing protocol. More research on multicast technologies can be found in [8] and [9].

Table 2.1: Categories of Multicast Applications

| **Categories** | **Applications** |
|---|---|
| One-to-Many | Live Video distribution<br>Periodic Data Delivery or "Push" technology<br>stock quotes, sports scores<br>Server/Web-site replication<br>Resource Discovery<br>Live Video distribution |
| Many-to-Many | Teleconferencing<br>Network games<br>Collaborative groupware<br>Distributed Interactive Simulation (DIS) |

Table 2.2: Multicast Addresses

| **Address** | **Description** |
|---|---|
| 224.0.0.0 - 224.0.0.255 | Local Network Control Block<br>Ex. RIP, DVMRP, OSPF |
| 224.0.1.0 - 224.0.1.255 | Internetwork Control Block,<br>Ex. NTP, rwho, RPC, SIP, DHCP-servers,<br>6 addresses for IETF conference broadcasting |
| 224.0.2.0 - 224.0.255.0 | AD-HOC Block<br>assigned to specific protocols |
| 224.1.0.0 - 224.1.255.255 | ST Multicast Groups |
| 224.2.0.0 - 224.2.255.255 | SDP/SAP Block |
| 224.252.0.0 - 224.255.255.255 | DIS Transient Block |
| 225.0.0.0 - 231.255.255.255 | RESERVED |
| 232.000.000.000-232.255.255.255 | Source Specific Multicast Block |
| 233.000.000.000-233.255.255.255 | GLOP Block |
| 234.0.0.0 - 238.255.255.255 | RESERVED |
| 239.000.000.000-239.255.255.255 | Administratively Scoped Block |

1. Multicast addressing

   Multicast data transferring will not begin without multicast address. The multicast address allocation architecture (MALLOC) [10] is under standardization of IETF malloc working group.

   According to RFC 2908, the multicast addresses for IPv4 are in the range of 224.0.0.0 to 239.255.255.255. The multicast address for IPv6 is defined by RFC 2375 [11]. Current multicast addresses for IPv4 are listed in Table 2.2. Note that more details of Internet multicast address range is specified in [12].

   Multicast addresses may be allocated in any of these three ways: static, scope-relative, or dynamic. Besides the multicast address, the usage of any given multicast address, as with unicast addresses, is limited in two dimensions: lifetime and scope. Lifetime is a duration time the address is valid, while scope is the

area of network that the address is valid.

- Static-allocated Address

  Statically allocated addresses are allocated by IANA for specific protocols that require well-known addresses to work. The range of addresses between 224.0.0.0 and 224.0.0.255 is reserved for the use of routing protocols and maintenance protocols, such as Routing Information Protocol (RIP) [13], DVMRP, and OSPF. Multicast routers should not forward any multicast datagram with destination addresses in this range, regardless of its TTL [12]. The range of addresses between 224.0.1.0 and 224.0.1.255 is reserved for the internetwork control protocol, such as Network Time Protocol (NTP) [14], RPC and SIP. Six multicast addresses are also allocated for broadcasting IETF conferences.

- Scope-relative Addresses

  Administrative scope-relative addresses (239.0.0.0 to 239.255.255.255) are reserved by RFC 2365 for use only within a local group or organization. These addresses can be reused with other local groups. Border routers are typically configured to filter out out-going multicast traffic in this address range. The highest 256 addresses in every administrative scope range is also reserved for relative assignments assigned by IANA. Generally, the relative addresses are assigned to the infrastructure protocols which require an address in every scope, such as a dynamic address assignment service. This offset is valid in every scope, and may be profiled into applications, devices and embedded systems. Such devices must have a way, e.g. via MZAP [15] or via MADCAP [16], to obtain the list of scopes in which they reside.

- Dynamic Addresses

  For a long time the multicast address has been dynamically allocated with the help of applications like SDR [17] that is implemented by using the Session Announcement Protocol (SAP) [18], the Session Description Protocol (SDP) [19], and the Session Initiation Protocol(SIP) [20]. The SDP is for defining the timing and capabilities of a session, while the SAP is for announcing those session descriptions using IP multicast. The SIP is for inviting individual users to participate in sessions. These dynamic-allocated addresses are provided on demand and have a specific lifetime and scope. The address range is between 224.2.0.0 and 224.2.255.255 is allocated for SDP and SAP protocol, while address range between 224.2.128.0 and 224.2.255.255 is for SAP Dynamic Assignments.

  However, many current multicast deployment models are not amenable to dynamic allocation [18]. For example, many content aggregators require group addresses that are fixed on a time scale that is not amenable to allocation by a mechanism such as described in RFC 2974 [18]. For such reasons, IETF malloc working group proposed a multicast address allocation

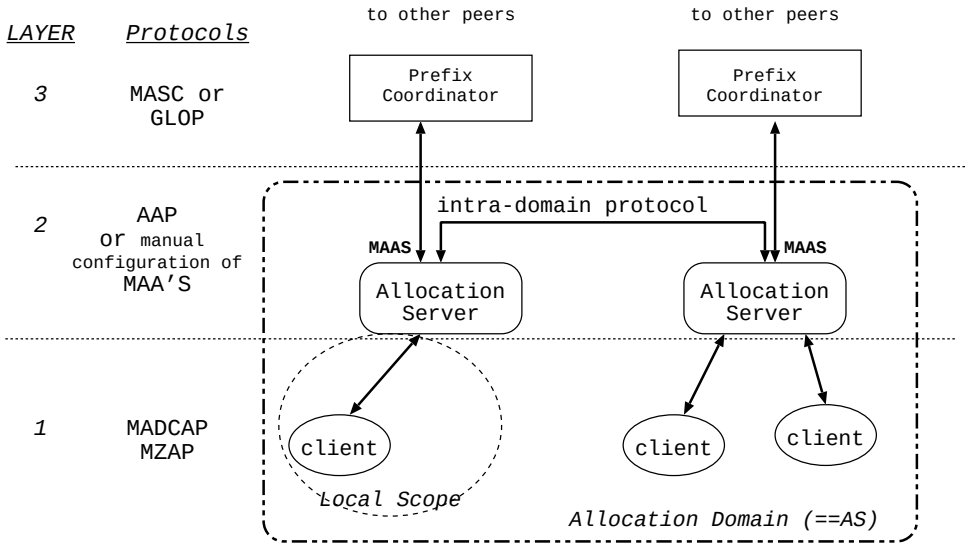



Figure 2.1: Overview of MALLOC architecture

architecture (MALLOC) [10]. The overview of MALLOC architecture [10] is depicted in Figure 2.1.

There are three components in MALLOC architecture: the client, the allocation server, and prefix coordinator. The MALLOC architecture can also be divided into three layers. In Layer 3, the prefix coordinator allocates an address range. It may use the Multicast Address-Set Claim (MASC) [21] to dynamically allocate the range, or statically allocate the range by AS number as described in GLOP [22].

The primary use of GLOP block space is many-to-many applications. GLOP aims to allocate the 233/8 range of multicast addresses amongst different ASes such that each AS is statically allocated /24 block of multicast addresses. For example, the multicast addresses for AS 5662 will be 01011000011110. Figure 2.2 is an illustration of the IP Address structure (as given in RFC 3180 [22]):

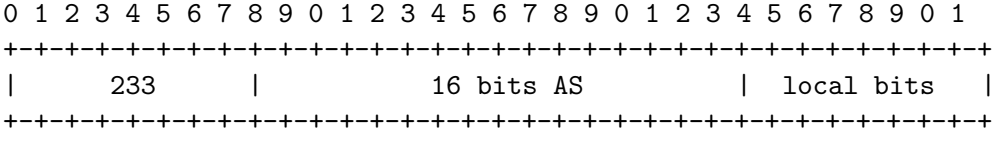



Figure 2.2: GLOP address

Once the prefix coordinator allocate an address range, it will pass the range to the Multicast Address Allocation Servers (MAAS) [10]. There may be multiple MAAS in the same allocation domain or the same autonomous

system (AS). In Layer 2, the intra-domain protocol, such as the Multicast Address Allocation Protocol (AAP) [23], will be use to ensure that the allocated address of each MAAS is not duplicated. In Layer 1, when the client wants a multicast address, it has to listen to the Multicast-Scope Zone Announcement Protocol (MZAP) [15] messages or send a Multicast Address Dynamic Client Allocation Protocol (MADCAP) [16] message to the allocation server. Then, it will choose a scope and lifetime, and send a request of address to the MAAS.

As seen from this MALLOC architecture, allocating a multicast address requires a complex configuration. Network administrators have to configure MAAS, and also have to support many protocols such as MADP and AAP. This is one of the problems that hindering IP multicast from wide deployment.

2. Multicast Group Membership Protocol

   Internet Group Management Protocol (IGMP) [24] is used by hosts and routers to inform each other about group membership. For IPv6, Multicast Listener Discovery (MLD) [25] is used by a router to discover the presence of multicast listeners on its directly attached links, and to discover specifically which multicast addresses are of interest to those neighboring nodes. MLD is derived from IGMP. However, one important different to note is that MLD uses ICMPv6 message types, rather than IGMP message type.

3. Intra-domain Multicast Routing Protocol

   Intra-domain multicast routing protocol enables routers to build a delivery tree between the sender(s) and receivers of a multicast group. An example of intra-domain multicast routing protocol is Distance Vector Multicast Routing Protocol (DVMRP) [26], Protocol Independent Multicast-Dense Mode (PIM-DM) [27], and PIM-SM [28]. Depending on how they construct multicast distribution tree, multicast routing protocols can be categorized into two groups: shortest-path tree and core-based tree.

   DVMRP is the first multicast routing protocol deployed on the Internet. Similar to RIP [13], DVMRP is distance vector routing protocol using reverse-path forwarding algorithm to compute the shortest paths. If the packet was received from an interface that is not on the shortest path from the source, the packet will be dropped. Otherwise, it is forwarded to all multicast-capable interfaces, except from incoming interface.

   Another example of shortest-path multicast protocol is PIM-DM. As in its name, PIM-DM is designed for the dense multicast group consisting of a large number of receivers. It performs similarly to DVMRP, but is different in the way that it does not depend on a specific unicast routing protocol.

   Contrary to PIM-DM, PIM-SM is designed for multicast groups with receivers sparsely distributed throughout the Internet. Each group has at least a Ren-

dezvous Point (RP), which is a router acts as a center of the multicast distribution. Each receiver sends join the group by sending a join message to the RP, while the multicast data is forwarded from the sender through the RP. The advantage of core-based-tree protocols, such as PIM-SM, is that they require less memory on routers than the shortest-path-tree, such as PIM-DM and DVMRP. However, the optimal path from the sender to each receiver depends on the network topology and the location of the RP itself.

4. Inter-domain Multicast Routing Protocol

   Currently, the combination of Multicast Source Discovery Protocol (MSDP)[29] and Multiprotocol Border Gateway Protocol (MBGP)[30] is used in order to support sources in different domains. MBGP is the BGP protocol version 4 extended to support the advertisement of reachability information and routing information for inter-domain multicast. MSDP runs over the MBGP and operate distributes the ASM active source information between the peer RPs via TCP connections. Similar to DVMRP and PIM-DM, MSDP flood and prune source active messages to all RPs in other MSDP domains by using a *peer-based reverse-path-forwarding check*. Unfortunately, MSDP does not scale and is unstable because of this flood-and-prune mechanism.

   To provide a more stable and scalable solution to the inter-domain routing, another protocol called Border Gateway Multicast Protocol (BGMP) [31] [32] is proposed and currently standardized by the IETF BGMP working group. The specification requires the use of Multicast Address Set Claim (MASC) protocol [21] to allow domains to claim the ownership of multicast address range. Consequently, different ranges of multicast address are associated with different domains; and each multicast group is associated with a single root domain. The border routers with hosts interested in a particular multicast group send the join messages toward the root domain where the sender host is located and then construct bidirectionaly shared trees between domains for active multicast groups.

Currently, one of the most common multicast protocol suites for supporting the ASM is composed of IGMP version 2, PIM-SM, MSDP and MBGP protocols. IGMP version 2 is for group membership management, while PIM-SM is for building multicast trees among receivers and sources in the same administrative domain. In the rest of this dissertation, IP multicast is mentioned for the combination of IGMPv2/PIM-SM/MSDP/MBGP. The advantages of IP multicast or Any Source Multicast (ASM) model are as follows:

- Dynamic membership Any nodes that know the multicast group address can join or leave the group at any time by the support of IGMP.
- On-demand data distribution Any group members can send data to other members at any time. At the same time, any nodes can start receiving data anytime without disturbing the sending process.

### 2.1.2 Current IP Multicast

Historically, MBONE or the Multicast Backbone [33] is started in 1992 to support experimental research. It is consisted of the multicast network tunneling through the general IP network. DVMRP is the first multicast routing protocol deployed on the MBONE. However, the inter-domain routing using DVMRP tackles problems like the explosion of router memory that very large number of groups could utilize [34]. Therefore, most multicast routers using MBONE today are replaced with PIM-SM.

At the time of writing this dissertation, many multicast routing protocols and reliable multicast protocols have been proposed. Many protocols are under standardization by IETF. Currently, there are eight IETF working groups related to multicast deployment. Further, two working groups at IRTF (the Internet Research Task Force) are concerned with multicast. Table 2.3 shows the current working group.

Table 2.3: Working groups at IETF and IRTF

| IETF | IRTF |
|---|---|
| Multicast-Address Allocation (malloc) | Reliable Multicast |
| Multicast & Anycast Group Membership (magma) | Group Security |
| Border Gateway Multicast Protocol (bgmp) | |
| Inter-Domain Multicast Routing (idmr) | |
| Protocol Independent Muticast (pim) | |
| Source-Specific Multicast (ssm) | |
| Reliable Multicast Transport (rmt) | |
| Multicast Security (msec) | |

Most of these working groups cover multicast routing protocols, while others are working on multicast address, group membership, reliable multicast, and multicast security. At present, Source-Specific Multicast, which is implemented as PIM-SSM, is the center of attention at the IETF. This technology can resolve the inter-domain routing problem and the third-party dependence problem when the RP of the group is in the other administrative domain.

Despite of a decade of efforts, IP multicast is used only for research and experiments and is not widely deployed in the Internet. This fact is referenced in Table 2.4, data from the current multicast status summary by Multicast Technologies [35]. The status shows that a multicast enabled Internet has grown from April 2001 to February 2003. The entire Internet has also grown during this time. However, the relative size of the multicast enabled Internet for Autonomous Systems (AS), i.e., the ratio of the number of multicast enabled ASes to the total number active, remains the same with not more than 3% of the entire ASes. More information about current multicast status can be found in [35] and [3].

Table 2.4: Current multicast-enabled Autonomous Systems

| Observation Date | Total multicast enable ASes | Total ASes | Percentage (%) |
|---|---|---|---|
| 2 Apr 2001 | 335 | 10427 | 3.2128 |
| 2 May 2001 | 352 | 10675 | 3.2974 |
| 2 Jul 2001 | 351 | 11150 | 3.1480 |
| 2 Aug 2001 | 349 | 11360 | 3.0722 |
| 2 Sep 2001 | 341 | 11523 | 2.9593 |
| 2 Oct 2001 | 340 | 11743 | 2.8953 |
| 2 Nov 2001 | 355 | 11857 | 2.9940 |
| 2 Dec 2001 | 364 | 12083 | 3.0125 |
| 2 Jan 2002 | 362 | 12200 | 2.9672 |
| 2 Feb 2002 | 376 | 12352 | 3.0440 |
| 2 Mar 2002 | 383 | 12555 | 3.0506 |
| 2 Apr 2002 | 389 | 12735 | 3.0546 |
| 2 May 2002 | 398 | 12941 | 3.0755 |
| 2 Jun 2002 | 392 | 13088 | 2.9951 |
| 2 Jul 2002 | 378 | 13246 | 2.8537 |
| 2 Aug 2002 | 378 | 13410 | 2.8188 |
| 2 Sep 2002 | 380 | 13551 | 2.8042 |
| 2 Oct 2002 | 377 | 13742 | 2.7434 |
| 2 Nov 2002 | 397 | 13919 | 2.8522 |
| 2 Dec 2002 | 408 | 14073 | 2.8992 |
| 2 Jan 2003 | 410 | 14221 | 2.8831 |
| 2 Feb 2003 | 413 | 14457 | 2.8567 |

### 2.1.3 IP Multicast Deployment Issues

To understand the problems, many researchers began to study multicast deployment issues, such as in [2]. Here, we summarize the IP multicast issues that currently prevent IP multicast from wide-scale deployment. These issues are indicated in [36] and [2]. Further, scalability problem is also mentioned in [37].

1. Complicate multicast address allocation

   First, as routers forward packets according to multicast address, IP multicast requires global multicast address allocation and management. There are many ways to allocate a multicast address, as describe earlier. However, these methods are complicated and require the deployment of other protocols or allocation authorities, such as MAAS.

2. Security vulnerability

   Second, to support inter-domain, it has to use Multicast Source Discovery Protocol (MSDP)[29] and MBGP [30] to announce active sources between domains. However, MSDP is vulnerable to denial-of-service attacks by domains sending out a flood of source announcement. Moreover, using a rendezvous point (RP) as in PIM-SM produces single point of failure problem.

3. Lack of access control

Third, the traditional IP multicast is *Any Source Multicast (ASM)*. Every node joining a multicast group can send packets to all group members through multicast address without any restriction. Even though on-demand data distribution is one of the advantages of this host-group model, it may be a security hole for group communication. Anyone who knows the group multicast address can join in the group and can send traffic to the group, because there is no mechanism to restrict hosts to join a specific multicast group. With this lack of a vital mechanism, an unauthorized sender can easily disrupt the multicast traffic. Therefore, the member management mechanism is also required to deploy IP multicast service in the actual commercial Internet environment.

4. Not scale in the number of groups

   Finally, IP multicast suffers from a scalability problem because a router has to maintain forwarding states for all multicast distribution tree passing through it. Thus, the number of forwarding entries increases with the number of groups.

## 2.2 Single Source Multicast

To gain wide-scale deployments, a new type of multicast protocols *the single source multicast* was proposed. Protocols that fall into this single source multicast category are EXPRESS, Simple Multicast, and SSM. EXPRESS [38] is a one-to-many multicast protocol proposed by Holbrook and Cheriton in 1999. The model was extended from IP multicast to support explicit subscription, and to provide membership information to the source. A simple solution for global multicast address allocation and management was resolved by identifying the tuple of (S,G) as a multicast channel, where S is the unicast address of the source and G is a class D multicast address. SIM also adopts this concept of group identification to eliminate multicast address allocation problem.

Another protocol that also extended the existing group identification is Simple Multicast proposed by Perlman. Simple Multicast is core-base tree multicast. Contrary to SSM, group identification in Simple Multicast is (C,G) where C is core router address, and G is multicast group address. This means that every core router in the Internet can assign the full 28 bits worth of multicast addresses. The needs for global allocation multicast addresses and coordinating multicast address allocation across the Internet is eliminated. Moreover, the core can be configured with the set of allowed or disallowed senders in order to control who can send the packets. Simple Multicast is efficient and scalable for large group communication. However, for small group communication, creating and maintaining a tree as in Simple Multicast is expensive. Furthermore, it cannot guarantee that the packets will pass through the most optimal path.

Source Specific Multicast (SSM) [39] or PIM-SSM is the successor of EXPRESS, and is implemented by extending the conventional IP multicast (PIM-SM). For IPv4, the address range of 232/8 has been assigned to SSM by IANA. This means that every source can have $2^{24}$ multicast groups. For IPv6, the range FF3x::/96 is defined for

SSM services [40].

Access control in PIM-SSM can be achieved by source filtering at the end routers of each receiver. A packet is transmitted by a source S to an SSM destination address G, and receivers can receive this packet by subscribing to channel (S,G). SSM provides host applications with a *channel* abstraction, in which each channel has exactly one source and any number of receivers [39]. Receivers must subscribe or unsubscribe to (S, G) channels to receive or to filter out traffic from specific sources. This source filtering function is done in support of IGMPv3[41] for IPv4 and MLDv2[25] for IPv6. Inter-domain routing issue is also resolved by a sending join message directly to the source. Therefore, unwanted multicast traffic will not be transported across the network unless it was requested by receivers, and can thwart denial of service (DoS) attacks from unwanted sources.

However, as Costa *et al.* argued in [42], SSM does not implement group management support. SSM resolved deployment issues summarized earlier, except the scalability issues in terms of number of groups. Both conventional IP multicast and SSM require all on-path multicast routers to exchange control messages and maintain forwarding states in order to maintain multicast distribution tree. Again, the cost of exchanging control messages and maintaining forwarding states can be taken place for efficient packet forwarding to thousands of receivers, but it is expensive for small group such as ten to hundred receivers. This cost arises a scalability issue, deployment and management problem, when the protocol has to deal with a large number of small group communications.

## 2.3 Explicit Multicast

To simplify packet forwarding and to avoid issues such as multicast address allocation, inter-domain multicast routing, and source advertisement as required in IP multicast deployment, a new multicast model called "Explicit Multicast" (Xcast) has been proposed [43][44][45][46][47]. In this dissertation, we will use the term "Xcast" as described in "Xcast Basic Specification" [7].

Contrary to IP multicast, which is targeted towards large group communication, Xcast is designed to support small group communication. Instead of using a multicast address, Xcast uses (Source Address, Channel Identifier) specified in Xcast header as channel identifications. Therefore, there is no need to allocate a global multicast address as in the traditional IP multicast model. The sender attaches a receiver address list to the packet header with the fixed *All_Xcast_Routers* address as the destination address, and sends the packet to the routers. Xcast routers look up unicast routing table, then forward and duplicate packets according to the unicast path to each receiver in the list. Basically, routers on multicast trees do not maintain any forwarding information and do not exchange any control message with peer routers. Note here that, carrying receiver addresses in a packet header was originally proposed in RFC 1770 [48]. However, RFC 1770's scheme uses an IP option to carry the addresses, which means that the maximum number of receivers that can be carried in one packet

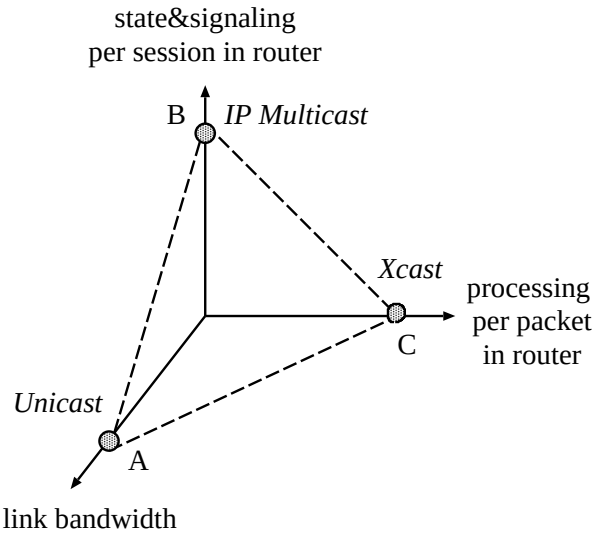



Figure 2.3: Three dimensions of data delivery costs

is restricted by the limited length of the IP option. Therefore, Xcast for IPv4 defined a new packet header, carried between the IPv4 header and the transport layer header.

As discussed in the basic specification of Xcast, there are three dimension spaces: consuming bandwidth, state-and-signaling per session, and packet processing overhead on routers (Figure 2.3). The costs of protocols move on these three dimensions. Point A, B, C depict the cost of unicast, IP multicast, and Xcast respectively. Unicast, which is delivering the same data to multiple receivers, costs most in link bandwidth dimension. However, it does not require any additional packet processing, signaling and maintaining states on routers. In contrast, IP multicast adds signaling and states on routers but sends the information only once to a multicast address. This cuts down on the excess usage of link bandwidth. For Xcast, it aims to cut both states and signaling on routers. However, the sender has to attach the receiver list to every packet, and each on-path Xcast router has to perform route looking up for each receiver in the list. That is, it introduces more packet processing overhead than IP multicast does.

Summarize, the advantages of Xcast are as follows:

- No states on routers: Routers on multicast trees basically do not need to maintain states for each multicast group.
- No control message : Basically, no control message is exchanged among routers.

However, Xcast has following drawbacks:

- High cost in forwarding packets: Xcast needs an additional header on every packet and has to process this header when the packets pass through each router.

Therefore, it is relatively expensive to forward Xcast packets, compared to ordinary multicast packets. Xcast packets may go through slow paths in high-speed routers.

- Less data space: Since a receiver list is attached to the packets, the maximum data area in each packet becomes slightly smaller, especially in the IPv6 case.

## 2.4 Recursive Multicast

In terms of using unicast addresses for routing packets, here we would like to discuss and compare SIM with REUNITE and HBH. Both REUNITE and HBH use recursive unicast to implement multicast service.

REcursive UNIcast TreE (REUNITE) [49] is based on unicast routing infrastructure. REUNITE maintains two tables on routers: Multicast Control Table (MCT) and Multicast Forwarding Table (MFT). Non-branching routers simply keep group information in their MCT, while branching routers keep MFT entries. Branching routers maintain group member addresses in their MFT entries. REUNITE has three control messages; join message, tree message, and stale tree message. Join message from destination $R_j$ is sent periodically to source, and will refresh $R_j$ entry in MFT. There is no leave message, but a receiver can leave the multicast group by stopping sending join message. Tree message is periodically sent from source to each destination to refresh $R_j$ MCT entries and $R_j$ MFT entries down the tree. Stale tree message will travel from source to $R_j$, cause routers on path to $R_j$ set timer, and will delete their $R_j$ MCT entries after this timer expired.

REUNITE constructs a SPT tree by using *tree* message that travels from the source to the destination nodes. However, it does not consider asymmetric routing. REUNITE may thereby fail to construct shortest paths, and produce unneeded packet duplications on certain links. Another problem of REUNITE is that the route to one receiver has to re-construct after the departure of another receiver. Cost *et al.* argue in [42] that this is undesirable if some QoS mechanisms are to be implemented.

The asymmetric routing problem in REUNITE has been resolved by HBH (Hop-By-Hop multicast routing protocol) [42]. HBH uses join, tree, and fusion messages to construct the multicast distribution tree, and maintains MCT and MFT tables. A join message in HBH is also sent periodically from a receiver toward the sender. However, only the first join will reach the sender, the others will be intercepted by the branching nodes, and refresh the nodes' MFT entries. The sender periodically multicast a tree message that refreshes the rest of the tree structure, i.e. the MFT and MCT. Fusion message is sent by potential branching routers and construct the distribution tree together with tree message.

As with SIM, HBH maintains addresses of the next hop branching nodes in MFT entries of branching routers, while REUNITE maintains receiver addresses instead. All MCT, MFT, and SIM FIB table entries are soft state, and are deleted after timers expire. MFT in HBH and REUNITE perform the same role as FIB entries in SIM. However, we have nothing like an MCT table. In contrast to REUNITE, SIM does not

maintain any control table on non-branching routers. This is done by a SIM Tunnel between two branching points. Non-branching routers will perform as ordinary unicast routers.

Concerning the asymmetric routing problem, SIM uses a data flow trigger from source to each receiver to construct the multicast tree. Therefore, asymmetric routing will not be a problem in SIM. The difference in group management between SIM model and HBH and REUNITE models is that we do not support join messages through routers. Join messages from receivers will be sent directly to the sender, and will not be processed on any routers. Therefore, as receivers are explicitly known by the sender in SIM, it is easy to provide access control.

A drawback of HBH is that it does not consider the situation when the router is overloaded. This is an important issue, because it also occurs today in IP multicast model, which has to maintain forwarding states on routers. REUNITE resolves this problem by using a join message. The join message in REUNITE is usually forwarded from the receiver towards the sender through REUNITE routers. In normal case, this message will not reach and be processed on the sender, but will be caught by a REUNITE router that is not overloaded and is on the distribution path. In contrast, in HBH, the first join message of a receiver will reach and be processed on the sender node. HBH has no ability to know whether a router on path is overloaded, and cannot maintain more MCT or MFT entries. SIM has considered this situation, and uses SIM redirect message to resolve this problem. We will describe the solution in detail later.

# Chapter 3

# Sender-Initiated Multicast

## 3.1 Introduction

As learnt from the failure in wide-scale deployment of IP multicast, I have concluded in chapter 2 that the problems need to be resolved are: (1) complicate multicast address allocation, (2) security vulnerability in inter-domain routing, (3) lack of access control, and (4) unscalable in the number of groups.

We have seen the deployment issues of IP multicast. Besides these technical issues, the current use of multicast is also an important issue to encourage multicast deployment. In order to understand the trend in multicast use, we referred to the multicast statistics disclosed on the web page, such as one by CAIDA. Table 3.1 and 3.2 are examples of multicast statistics, 3 April 2002 [3]. The tables show current state of multicast and groups with most receivers, respectively. On 3 April 2002, there were 1,016 groups observed on the Internet, while only three of them contained more than 50 receivers. In other words, we can say that most of multicast sessions are relatively small, and currently contain not more than 50 receivers per group. From this fact, we conclude that supporting small groups of multicast can satisfy the current use, while here we define "small multicast groups" as groups contain not more than 50 receivers.

Table 3.1: Multicast, Current State

| Entity | Value |
|---|---|
| Groups | 1016 |
| Participants | 4560 |
| Uniqe Participants | 1183 |
| ASes | 141 |
| RPs | 196 |

In this chapter, we propose Sender-Initiated Multicast (SIM) [50] [51] [52] for small group communications. It introduces the basic idea of packet forwarding from Xcast, that is attaching receiver list to packet header and making the routers to forward

Table 3.2: Groups With Most Receivers (Last 1 Hour)

| Address | Participants |
|---|---|
| 225.1.2.3 | 407 |
| 233.2.171.1 | 111 |
| 224.2.127.254 | 84 |
| 224.0.1.1 | 45 |
| 224.2.177.155 | 36 |
| 224.0.1.84 | 25 |
| 224.0.1.32 | 22 |
| 224.2.172.238 | 21 |
| 233.2.168.1 | 20 |
| 233.81.229.1 | 16 |

and copy the packets for each receiver by looking up unicast routing table. SIM can eliminate deployment issues occurring in IP multicast model for the following reasons.

- No complicated multicast address allocation

  SIM routers forward packets and maintain the forwarding information by using the combination of sender and group address. The group address has no need to be unique to the Internet. Therefore, the multicast address allocation issue is eliminated.

- No security vulnerability in inter-domain multicast routing

  In the forwarding process for both intra-domain and inter-domain, the unicast routing table is used. There is no mechanism to flood the sender advertising messages to peer domains as by MSDP.

- Support data transferring with access control

  Contrary to IP multicast, access control for private or commercial use can be easily provided, because the sender knows which hosts are in the group. The sender can send a group key to each receiver. Then, each receiver uses the key to decrypt the data.

- Scalable in number of groups

  As with Xcast, SIM can support a large number of small multicast groups. Contrary to IP multicast, SIM maintains the forwarding state only on routers that act as branching points of the distribution tree. Therefore, the memory requiring on each router is lessened. As the result, SIM can supports more number of small multicast groups.

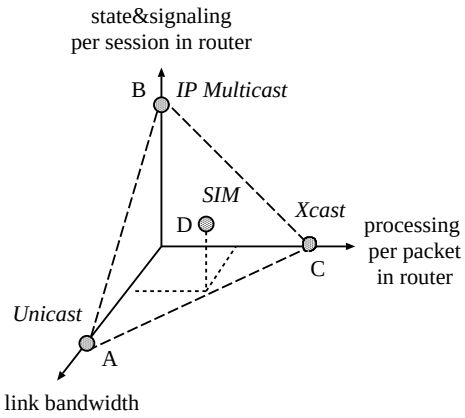



Figure 3.1: Three dimensions of data delivery costs

## 3.2 Overview of SIM

### 3.2.1 Design Goal

As shown in Figure 3.1, the network related costs can be considered as three overheads: link bandwidth, state-and-signaling per session on router, and processing per packet on router. We can conclude that: there is a trade-off between protocol performance and the three dimension costs. There is no ideal protocol with zero costs. Moreover, there is no one-size-fits-all solution for multicast routing. IP multicast, regardless of its cost in signal-and-state, may be appropriate for large multicast group, such as a group contains about 1,000 receivers. Xcast may be appropriate for relatively small group, not more than 10 receivers because it has to attach receiver list to all packets. However, the groups with about 50 receivers cannot be supported by IP multicast or Xcast with reasonable costs. In order to provide scalability in number of groups, we need a protocol that requires less cost in state-signaling and processing overhead.

Our goal is to attain Point D in Figure 3.1. Contrary to Xcast, SIM maintain some state on routers, only on branching routers, in order to reduce the packet processing overhead. While at the same time, SIM still consumes less bandwidth than the multiple unicast transmission.

The objectives of SIM are the following:

- Lessen processing overhead per packet as occurs on Xcast routers.
- Reduce state-and-signaling cost as occurs on IP multicast routers.

### 3.2.2 Target Applications

SIM is designed to support the applications that have the following characteristics:

1. Small group communications

Video conferencing is the example of small group communications. In most cases, video conferencing groups are relatively small, containing approximately four locations, i.e., four group members. A larger size of small groups may be a distance learning system with about 50 students attending the class from their houses. Another example is a mirroring server, while the origin server transfers updating data to each mirroring server. Note that, at the current implementation, SIM supports data transferring to a group with not more than 64 receivers. In order transfer data to a larger group, the sender has to divide a large group into multiple small groups and transfers data to these small groups.

2. Sender-initiated data transferring

   The prerequisite of data transferring in SIM model is that all receivers are well known to the sender. This kind of sender-initiated data transferring is also called *push applications* [53], while the receiver-initiated data transferring, such as a web browser, is called *pull applications*. E-mail is probably the oldest and most widely used push technology. The sender pushes the messages to the receivers, whether the receivers ask for it or not. Another example of push applications is the slide viewer in the distance learning. The instructor pushes or distributes class materials to each student. When the instructor turns on the slide, the application on the instructor side notifies the actions to the applications on the student side.

### 3.2.3 Overview of SIM Forwarding Techniques

SIM has two forwarding modes: *List mode* and *Preset mode*. The difference of these two modes is that the sender will attach the receiver list to every packet in List mode, but it will periodically attach the receiver list to only some of the packets in Preset mode. List mode is good for a short data transmission, and when the receiver list is supposed to be modified frequently. Preset mode is appropriate for a long data transmission, and when the receiver list is seldom modified. In List mode, SIM routers do not maintain any forwarding information. They forward SIM packets and process as same as the ordinary Xcast routers do. To make clear the significant characteristics of SIM, therefore in this dissertation I will discuss only SIM in Preset mode.

The techniques to achieve efficient packet forwarding are the followings:

- **Maintains SIM FIB with group identification**: As we explained, in Xcast the routers will suffer from packet processing overhead because they have to look up in the unicast routing table for each receiver every time the packets arrive. In Preset mode, SIM resolves this problem by caching the forwarding information to SIM Forwarding Information Base (FIB) maintained on each router. To accelerate SIM FIB entry lookup, SIM introduces the concept of using a combination of a sender unicast address and multicast group address *(S,G)* as a group identification. As the multicast group address here has no need to be unique, therefore it does not need any central address allocation and management entity.

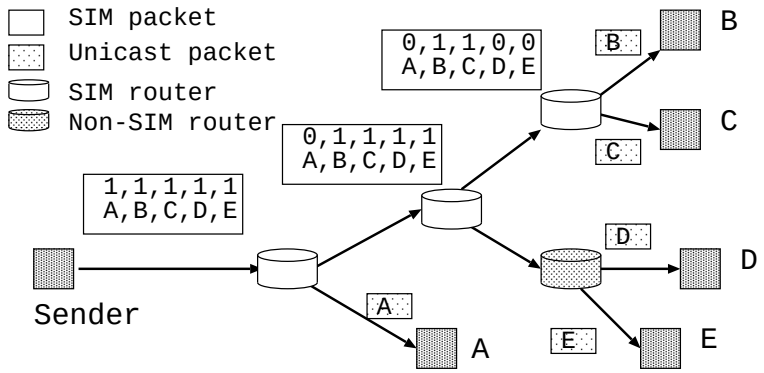



Figure 3.2: SIM forwarding process and the bitmap

Forwarding information will be hashed into SIM FIB table by using *(S,G)* value. Therefore, SIM routers can usually find SIM FIB entry by performing only one lookup, and that is the first step to accelerate SIM forwarding process.

- **Use bitmap for packet forwarding**: The second step is to use the bitmap field in packet header. The bitmap concept is originally proposed in Connectionless Multicast (CLM)[47], and is also put into consideration in the basic specification of Xcast[7]. Each bit in bitmap field has to be corresponding with the sequence of the receiver list, e.g. the first bit corresponds to the first destination in the list (Figure 3.2). Bitmap is used in order to abbreviate the receiver list, where the packet has to be forwarded to the same set of receivers. Moreover, when a router has to duplicate the packet, it can avoid complex packet-header reconstruction by only turning on or off bitmaps according to routes toward each receiver.

- **Automatically creates SIM Tunnel**: The last step to accelerate SIM forwarding process is to use SIM Tunnel, the most significant characteristics of our protocol. SIM Tunnels are created among the branching-point SIM routers. Therefore, not all on-path SIM routers but only branching-point SIM routers have to maintain SIM FIB entries and process SIM packet forwarding. We will describe this function in detail in section 3.5.2.

## 3.3 SIM Packet Header

The packet sent from a SIM sender contains a bitmap field, and in some cases a receiver list in SIM header. If necessary, port numbers and checksums for each receiver can be included in SIM options (Figure 3.3).

SIM uses two IP headers in order to perform IP-in-IP tunneling between SIM routers (Figure 3.4). The source and destination address fields in inner IP header contain the sender address and the multicast group address, respectively. It is worth

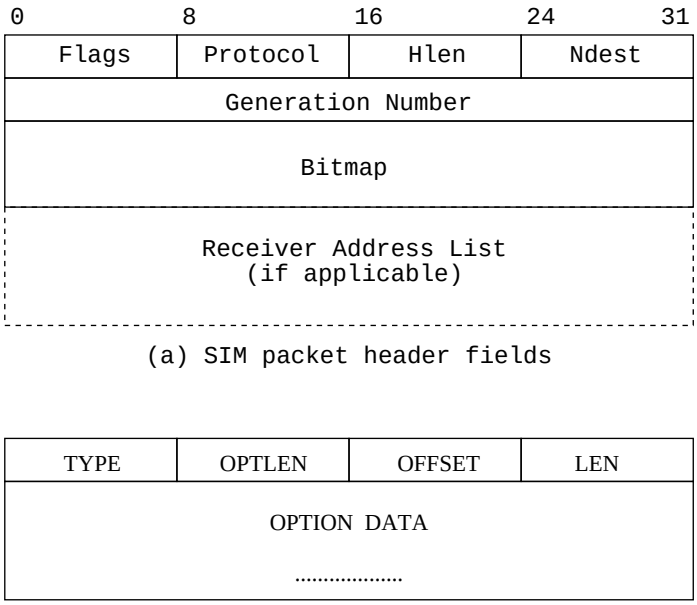



Figure 3.3: SIM packet header and option header fields

mentioning here that we do not use multicast address to route packets, as SSM does, but only for purpose of maintaining SIM FIB.

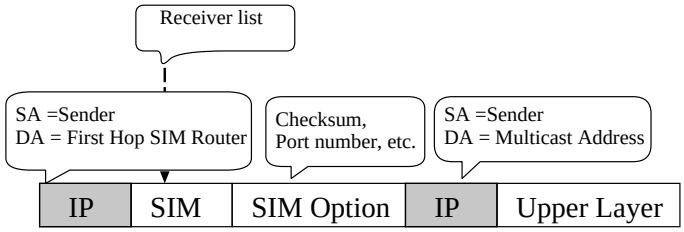
Packets are forwarded hop-by-hop through SIM routers with the next-hop SIM router as the destination address. That is, without a SIM Tunnel (section 3.5.2) the destination address field of the outer IP header will be rewritten whenever the packet passes through a SIM router. On the other hand, with SIM Tunnel, both source and destination address fields will be modified to the address of tunnel ingress and egress routers, respectively.

## 3.4 SIM Membership Management and Forwarding Process

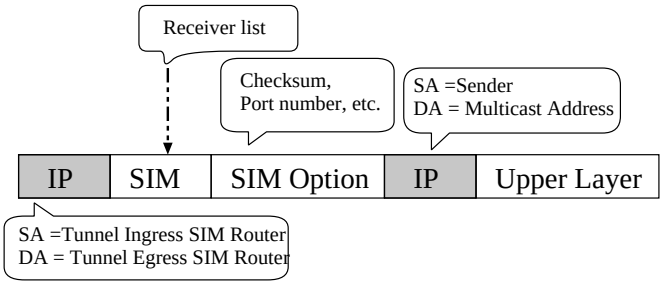
In this section, we describe SIM membership management and the overview of forwarding process performed on each router. Details of how to maintain distribution trees, e.g. maintenance of SIM FIB, will be described in the next section.

### 3.4.1 SIM Membership Management

As with Xcast, the precondition of SIM is that the sender knows all the receiver addresses in advance, before the transmission begins. There is no control message for exchanging group membership among SIM routers. The sender can find the IP ad-




(a) First sent from sender




(b) After passed SIM branching router

Figure 3.4: SIM Packet Header

dresses of receivers by many methods such as via email or a Web browser. By requiring a receiver to subscribe and unsubscribe directly to the sender in this way, we have no need to use join message or to process join message on routers. Unlike IP multicast model, there is no Internet Group Management Protocol (IGMP) communication between multicast group member and router.

In SIM model, the sender can control the access permission separately. The sender has to notify the receivers of the multicast group identifier, which is the combination of sender unicast address and multicast address. This notification mechanism can be deployed through several possible membership management systems depending on the communication models. Any kind of group management can be held in the application layer.

For one-to-many communication such as file distribution, a group membership management can also be implemented on the sender itself. It may send an Invitation message or email to each receiver. Then, after receiving Join message from all receivers, the sender can begin multicast data transferring.

However, for many-to-many communication such as video conferencing and network games, it may be preferable to use a central group management authority or server in order to synchronize the receiver list among members, and to exchange group key when data confidentiality is required. This is needed because the modification of receiver list can occur anytime when some members have left the group, or other members have joined the group. Without a central group management server, it will take a long time

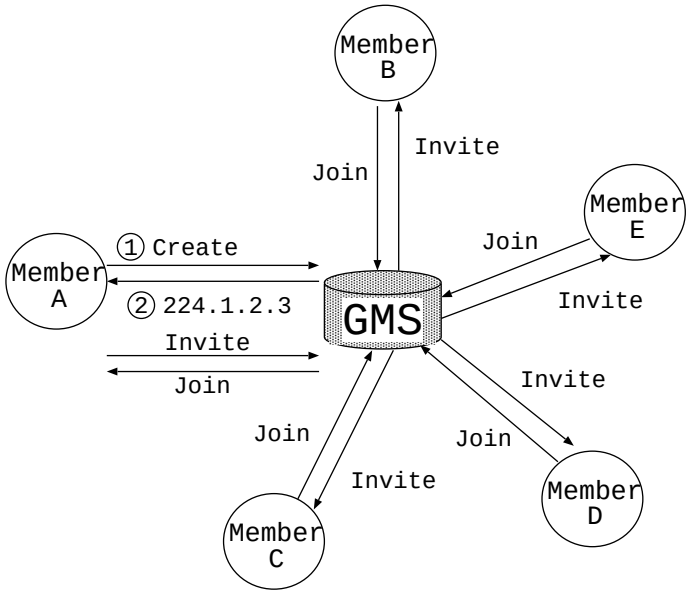



Figure 3.5: Group Management Server for many-to-many communication

to let all the group members know the current receiver list correctly. For example, suppose that there are $n$ members in a group, and $m$ newcomers have joined the same group. If each newcomer has to exchange Invitation and Join message to inform all former members, and $m$ newcomers join the group sequentially, then, to report a current receiver list to all members, it will take the following number of messages:

$$2 * n + (n + 1) + (n + 2) + \ldots + (n + m - 1)$$

To reduce the number of exchanged messages, I propose to have only one node act as Group Management Server (GMS). It will centrally maintain the receiver list [54]. This node may be one of a group member or the third party node. Each member has to register its membership to this server and share the list before the conference begins. This mechanism can also support asymmetrical group management service that may be useful in some cases, e.g., company conference and network games. Further, security functions can be provided on this server, such as distribution of group key in order to implement data encryption and to achieve traffic confidentiality.

Figure 3.5 shows how a multipoint-to-multipoint session is created by using GMS. First, one of a member (A), which may be a chairperson of the conference, has to create a group on the server. After the server created a group identification, it will inform member A. Member A can broadcast this group identification to all members by using email or web page, or it can have server send "Invite" message to other members in order to inform and invite them to the communication group. After all members have sent join message to the server, it will distribute the list of receivers to each member.

The list update can be done through the server anytime a member leaves or joins a group.

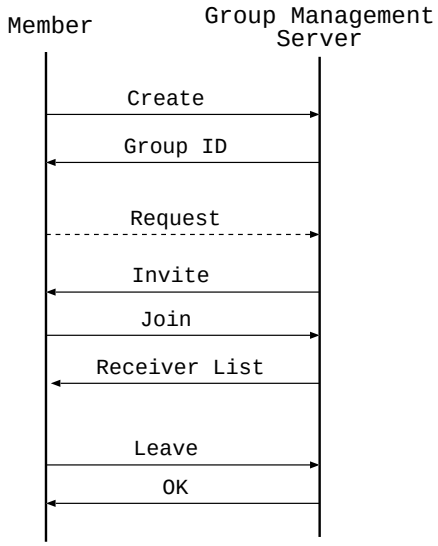



Figure 3.6: Communication messages between GMS and a member

### 3.4.2 Allocating a multicast address

SIM routers use the two-tuple of (sender address, multicast address) to identify a multicast session. Therefore, the multicast address does not need to be unique. That means every sender can assign the full 28 bits worth of multicast addresses. Each sender may assign the multicast group address randomly. A multicast session will contain one sender and multiple receivers. This scheme works well with one-to-many communication. However, for many-to-many communication, it should be better to provide a server act as GMS in order to support dynamic group membership. To identify a communication group on GMS, the multicast address for group identification has to be unique within a single GMS. Hence, allocation of a multicast address should also be done on GMS. For example, if the group identification on GMS is the multicast address G, the group identification on SIM routers for sender $S_1, S_2, S_3$ are $(S_1, G), (S_2, G)$ and $(S_3, G)$. The value of multicast address G can be obtained from the server as *ReceiverList* message as shown in Figure 3.6.

### 3.4.3 Forwarding Algorithm

Figure 3.7 shows the packet forwarding process on a SIM router.

1. A SIM packet with *IPPROTO_SIM*, identified in the next header field of IP header, will be input to SIM functions.

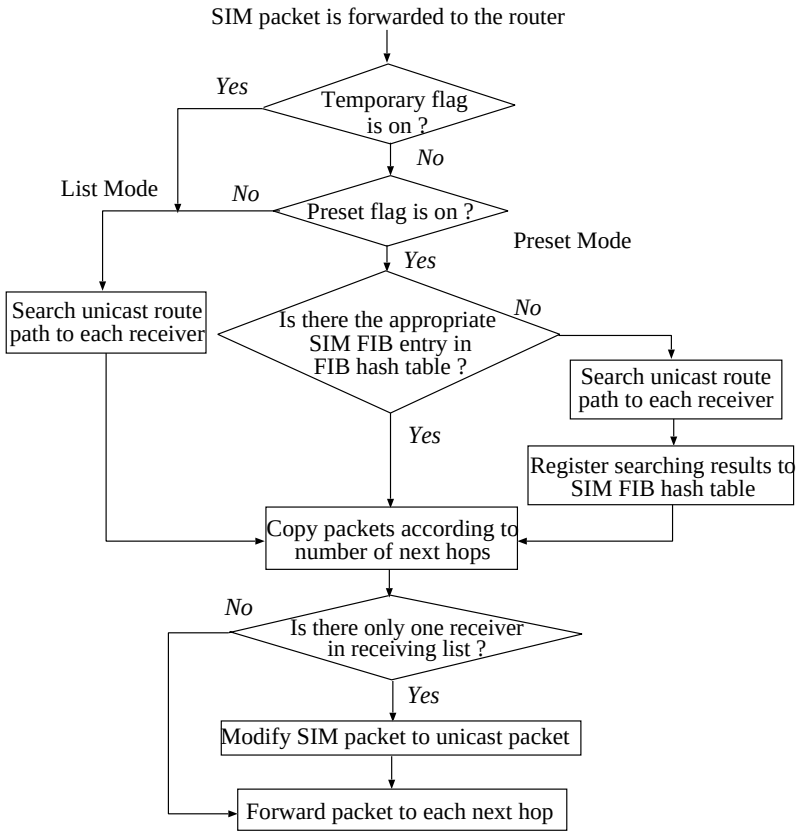



Figure 3.7: Flowchart of SIM packet forwarding algorithm

2. If the Preset flag in the SIM header is turn on, the packet needs forwarding in Preset mode. Then, go to step 3. Otherwise, go to step 4.

3. In Preset mode, the SIM router will look up the existing SIM FIB entry in the SIM hash table. If an entry for this group does not exist, go to step 4. Otherwise, check the generation counter of the FIB. If the generation counter in the packet header is the same as that in the SIM FIB entry, the SIM router will forward the packet according to the forwarding information in the SIM FIB entry. Otherwise, go to step 4.

4. The router will look up the next SIM hop routers or the appropriate forwarding interfaces for all receivers in the address list that the bitmap turned on. If the packet specifies Preset mode, the router will also register routing table lookup results as a new SIM FIB entry.

5. If this router is a branching point of the multicast distribution tree, it will duplicate the packets. Then, it forwards the packets according to the forwarding information.

### 3.4.4 Conversion to Unicast

SIM conversion to unicast is performed in two cases: first, when there is no SIM-capable router on the next hop; and second, when there is only one receiver for the next hop routers. The SIM router reconstructs the packet header so that it seems as though the packet is originally sent by an ordinary unicast application. The SIM router performs this conversion by detaching SIM header and SIM options and then rewriting the source and destination address fields of the packet. If necessary, it also rewrites some fields of the upper-layer protocol header, such as checksums and port numbers. These substitution data are created by the sender and stored in a SIM option header, in sequence of receivers in the address list.

The advantages of this conversion are:

- SIM packets can pass through some routers that cannot understand our protocol, thus we have no need to update all routers in the network.
- All receivers can receive data without any modification of the existing operating system and applications.
- It gives SIM an opportunity to support widely developments of new upper-layer protocols.

These three advantages give SIM an excellent ability to be incrementally deployed in the global Internet. All that is required to use our protocol is the sender who wants to send SIM packets, and for one router to be upgraded.

## 3.5 SIM Tree Maintenance

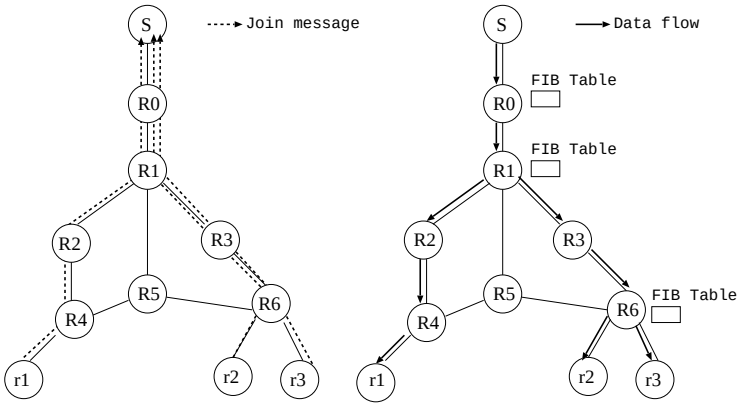



Figure 3.8: Multicast Tree Construction

As shown in Figure 3.8, SIM has no join and leave message traverse through routers. Each receiver sends a subscription or unsubscription of data directly to the sender through email or web page. Multicast distribution tree is created by the trigger of data flow sending from sender to each receiver. Therefore, we do not need to consider asymmetric routing issue and maintain MCT entries as in HBH. When routers on path received data flow in Preset mode, they maintain FIB entries for each $S, G$. However, by using SIM Redirect message to create SIM Tunnel between two branching routers, FIB entries on non-branching routers will be removed after a timer expires. This section mainly describes management of SIM FIB entries, and the message processing in the model.

### 3.5.1 Management of SIM FIB

Despite of performing routing table lookup for each receiver in all packets, SIM routers maintain the Forwarding Information Base (FIB) for each multicast session. A SIM FIB entry is registered in SIM FIB hash table as follows:

$$(S, G) \rightarrow (gen\_count, coming\_from, next\_hop\_list)$$

Here, $S$, $G$, and *gen_count* represent sender address, multicast group address, and the generation counter of SIM FIB, respectively. *Coming_from* is one of the incoming network interface or the previous branching router when SIM Tunnel is used, while *next_hop_list* points to outgoing interface or tunnel egress address.

**Generation Counter** A generation counter field in the SIM header is used to enable routers on a multicast tree to detect any change of the address list. Whenever the receiver list of a flow changes for some reason, such as a receiver joins or leaves the group, the generation counter will be incremented. Moreover, the initial value of the generation counter should be randomly created to avoid a Denial-of-Service attack. If the generation counter indicated in the packet is the same as the previous one in the SIM FIB entry, SIM routers will use the same forwarding information to forward the packets. On the opposite, if the generation counter of the packet is more than that of the SIM FIB entry, the routers will register the forwarding information as a new SIM FIB entry.

**SIM FIB Update** SIM FIB entries are soft state. In Preset mode, the sender also increases the generation counter and attaches a receiver list to the packets for each interval time. This interval time is called a *SIM_FIB_INTERVAL*. If no appropriate packet to attach a receiver list exists, when the timer expires, a SIM packet without data will be sent. SIM routers will update the SIM FIB entry every time they get a new generation counter. In contrast, if a SIM router does not receive a receiver list for a data flow for the *SIM_FIB_TIMEOUT* time, the corresponding SIM FIB entry will be removed. However, a SIM FIB entry of an old generation counter should be kept for *SIM_FIB_AGING* time, as some packets with the same generation counter may be delayed by network congestion or network failure.

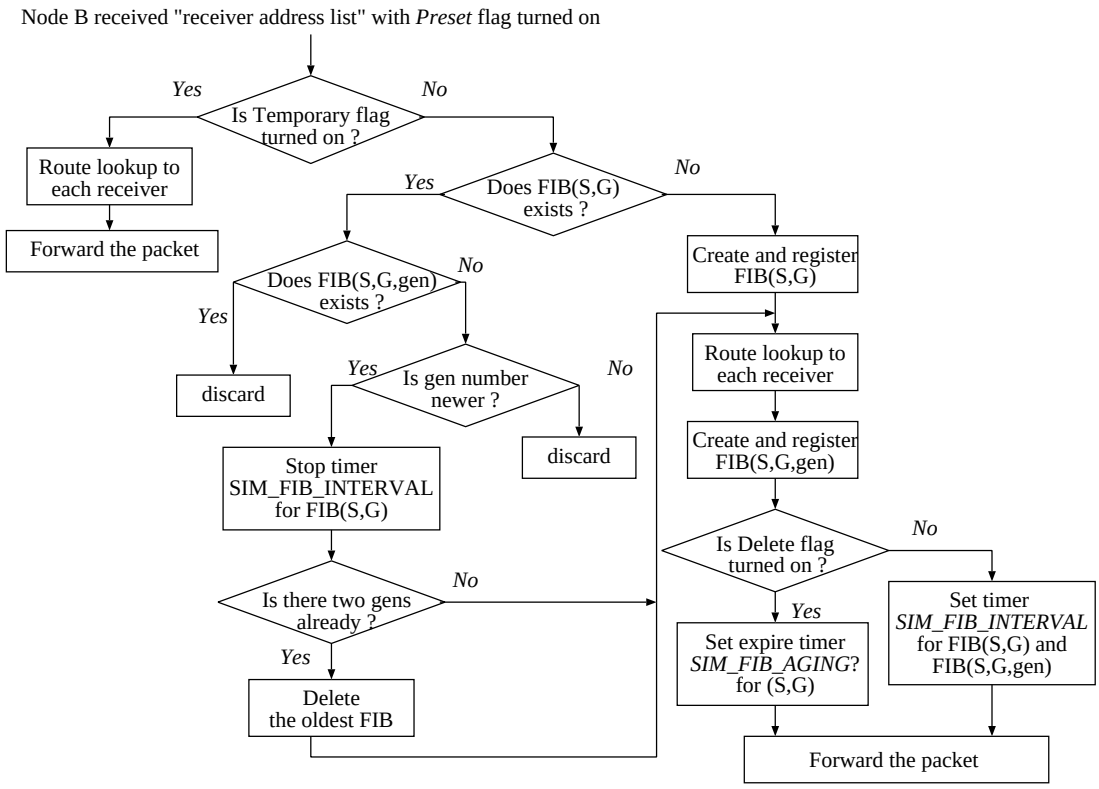



Figure 3.9: SIM FIB management

**Temporary Flag** Two flags exist on SIM header that operate SIM FIB entries. The first one is *Temporary flag*. A temporary flag is set when the members of a receiver list of a flow are temporarily modified for a few packets, for some purposes such as retransmission to a subset of receivers. If the Temporary flag is set, SIM routers must use the attached receiver list to forward the packet even if the corresponding SIM FIB entries exist.

**Delete Flag** The second flag is the *Delete flag*. When a sender sends the last packet of a flow, the sender turns on the Delete flag in the header to request SIM routers to remove the corresponding SIM FIB entries. When the router receives a SIM packet with this flag turned on, the router will not immediately eliminate the entry. In addition, the router will set a *SIM_FIB_AGING* timer and remove the entry after the timer expires. Figure 3.9 depicts SIM FIB Management on router.

### 3.5.2 SIM Tunnel and SIM Redirect Message

In Preset mode, a branching router sends *Redirect Message* to an upper router in two cases:

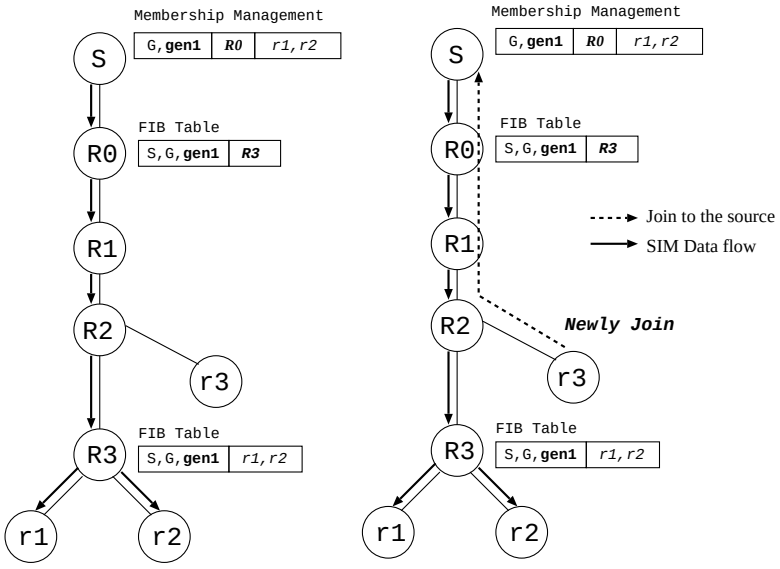



Figure 3.10: New branching node by newly join receiver

- Automatically create a SIM Tunnel between two routers that act as branching points of multicast distribution tree.

  Whenever a packets pass through a non-branching SIM router, the router will turn on *Jump flag*. The next SIM branching router will detect this flag when the packet arrives and send a Redirect Message to the previous SIM branching router to create a SIM Tunnel (Figure 3.11).

- Request an upper router to pre-duplicate packets, when the router sending Redirect Message is overloaded.

  In Figure 3.10, R1 is a new branching router is created as a receiver begin to subscribe to the group. Normally, if this router is not overloaded, it will send SIM Redirect Message to create SIM Tunnel as described. However, when the SIM FIB table of a router is full, i.e. the router cannot maintain more FIB entries but still can forward the packets, the router uses SIM Redirect Message to tell the upper router of the data flow to duplicate packets in advance (Figure 3.12).

As described in section 3.3, the source address of the outer IP header will be changed only when the packet passes through a SIM branching router; therefore, the next branching SIM router will know the previous branching SIM router from the source address field of the packet. When the previous branching router receives the SIM Redirect message, it updates the appropriate SIM FIB entry and specifies the next SIM branching router in the destination address field of the packet. Therefore, a SIM Tunnel is created during two SIM branching routers. With this tunnel, non-branching

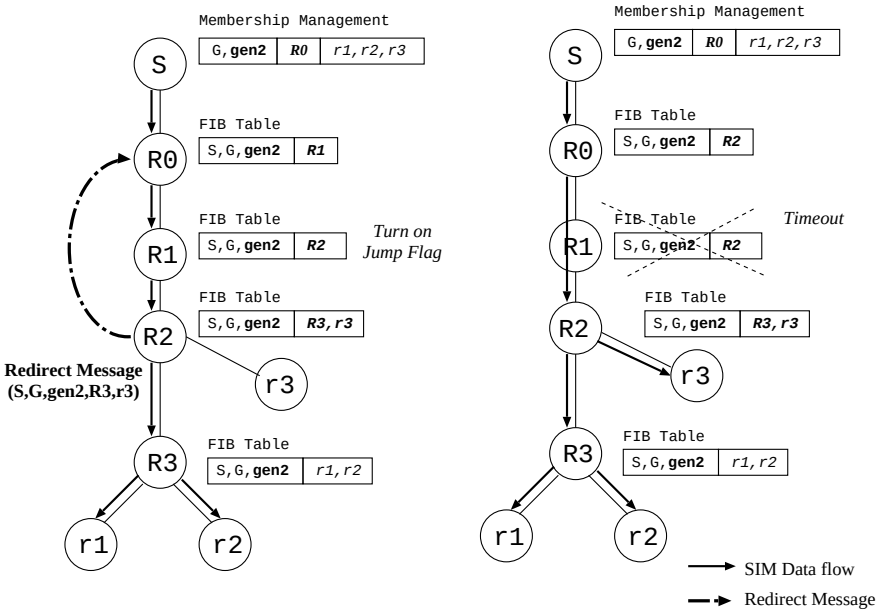



Figure 3.11: SIM Redirect message and SIM Tunnel for capable new branching node

routers will recognize SIM packets as ordinary unicast packets and just route them to the next branching router.

Figure 3.14 depicts SIM redirect header fields. Considering some kinds of DoS attack to SIM FIB on routers, we included the bitmap and generation number fileds to the header. Because the generation number is randomly generated and the bitmap is changed depending on the distribution path, it is difficult for an attacker to presume the values. The router sending SIM Redirect message has to copy the values of these two fields from the SIM packet. Everytime a router received a SIM Redirect message, it will lookup in SIM FIB hash table for a SIM FIB entry with the same bitmap and generation number. If there is no appropriate SIM FIB entry, SIM Redirect message will be abandoned.

This automatic SIM Tunnel is a significant feature of SIM. The advantage of the SIM Tunnel is that only the branching routers have to maintain the SIM FIB entries and perform routing table lookup for each receiver on the list. This advantage is very especially useful for applications such as teleconferencing, where the members of multicast groups usually are sparsely distributed in the network, and the probability that the packet has to be duplicated or branched on the same router is low. The SIM Tunnel mechanism accelerates the forwarding process, and potentially reduces the number of routers that need to maintain SIM FIB. As a result, SIM gains more scalability in the number of small multicast sessions than the current IP multicast and Xcast do.

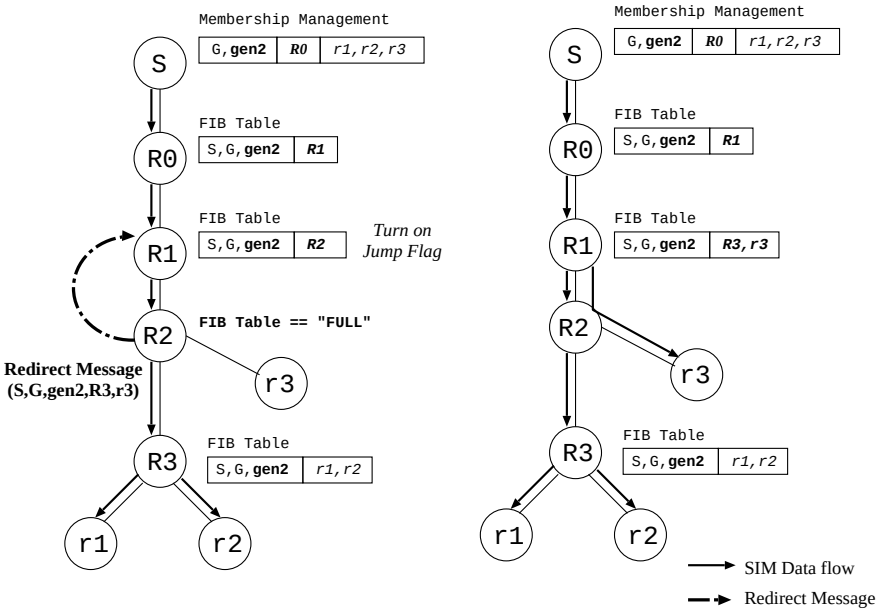



Figure 3.12: SIM Redirect message for overloaded node

## 3.6 Implementation

The first SIM prototype was developed on FreeBSD4.1.1-RELEASE by Yosuke Takahashi[51]. The problem with this prototype is that, it did not include the SIM FIB management and the SIM redirect function. The prototype is also developed for M/TCP 4 and does not support data transferring by UDP. To support data transferring by UDP and all others SIM functions described in this chapter, I have implemented a new version of SIM for IPv4 as a kernel option with kernel modifications for FreeBSD4.6-RELEASE. The new version has an added preset mode with SIM FIB management function and SIM Tunneling function. Socket interface is support by *setsockopt()* for IP level.

### 3.6.1 SIM Header Field

As depicted in Figure 3.4, SIM header is located between the outer IP header and the inner IP header. The next protocol field in the outer IP header specifies the value of *IPPROTO_SIM*, which is defined as a new protocol number in `<netinet/in.h>` header file. Note that, this protocol number is used for both SIM data transmission and SIM Redirect message between routers. The definitions of the SIM header structure is shown below:

```
typedef struct sim_hdr {
        u_int8_t  sim_flags;          /* several flag */
        u_int8_t  sim_protocol;       /* protocol of next header */
        u_int8_t  sim_hlen;           /* length in units of 4 octets */
```

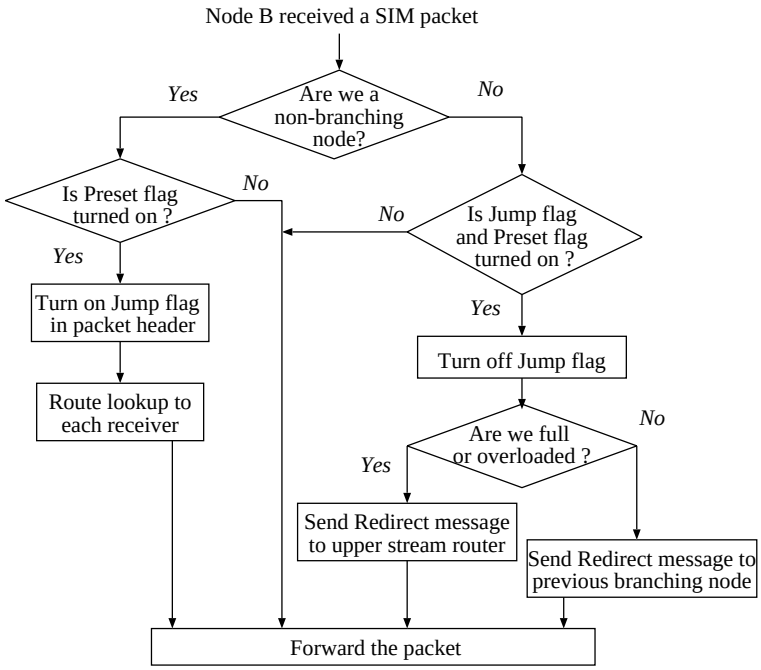



Figure 3.13: Jump flag processing

```
        u_int8_t  sim_ndest;          /* # of destinations  */
        u_int32_t sim_gen;            /* generation count */
        u_int64_t sim_bitmap;         /* variable length destination bitmap */
        struct in_addr sim_addr[1];   /* up to 32 addresses */
} sim_hdr_t;
```

The definition of *sim_flags* is as follows.

```
/* SIM header flags */
#define SIM_PRESET_FLAG     0x80   /* 10000000 */
#define SIM_TEMPORARY_FLAG  0x40   /* 01000000 */
#define SIM_DELETE_FLAG     0x20   /* 00100000 */
#define SIM_JUMP_FLAG       0x10   /* 00010000 */
#define SIM_SIM_FLAG        0x08   /* 00001000 */
#define SIM_OPTIONS_FLAG    0x04   /* 00000100 */
#define SIM_REDIRECT_FLAG   0x02   /* 00000010 */
#define SIM_ADDRLIST_FLAG   0x01   /* 00000001 */
```

SIM option header, which will be used later in Chapter 4, is defined as the following.

```
typedef struct sim_opthdr {
        u_int8_t   simopt_opttype;  /* option type */
        u_int8_t   simopt_optlen;   /* option length in units of 4 octets */
        u_int8_t   simopt_off;      /* offset from top of upper-layer header */
```

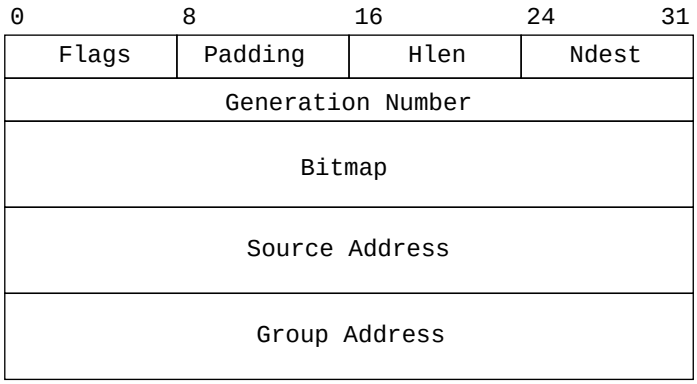



Figure 3.14: SIM redirect header fields

```
        u_int8_t   simopt_len;      /* option length in units of 4 octets */
        u_int32_t  *simopt_data;    /* variable length substitution data */
} sim_opthdr_t;

#define OPT_TYPE_MTCP   0x01  /* partial substitution option type for MTCP */
#define OPT_TYPE_UDP    0x02  /* partial substitution option type for UDP  */

/* SIM partial substitution option */
/* M/TCP */
#define PART_MTCP_LEN 6     /* partial substitution length of MTCP*/
#define PART_MTCP_ACKLEN 4  /* partial substitution length of MTCP*/
#define PART_MTCP_SUMLEN 2  /* partial substitution length of MTCP*/
#define PART_MTCP_ACKPOS 8  /* ACK field position */
#define PART_MTCP_SUMPOS 16 /* CHECKSUM field position */

/* UDP: in case port number for each receiver is different */
#define PART_UDP_LEN     4  /* partial substitution length of UDP */
#define PART_UDP_PORTLEN 2  /* partial substitution length of UDP */
#define PART_UDP_SUMLEN  2  /* partial substitution length of UDP */
#define PART_UDP_PORTPOS 2  /* PORT field position */
#define PART_UDP_SUMPOS  6  /* CHECKSUM field position */

/* SIM pre-definitions */
#define SIM_MAX_DEST        32 /* 32 is recommendation max number
* as considering of throughput */
#define SIM_PATH_MTU       576 /* Path MTU for fragmentation */
#define SIM_MAX_GEN        ((2^31) + 4957)
```

### 3.6.2 The SIM Forwarding Information Base

SIM routers in preset mode maintains the SIM Forwarding Information Base (FIB) for each *(S,G)* session. The SIM FIB is defined in *sim.h*, and consists basically of the source address, group address, and out-going route entries. The complete definition is as the following.

```
/* Structure for SIM FIB hash table. */
struct simhash {
        struct sim_fib  *head;     /* ptr to first fib entry */
};

/* SIM FIB rtlist (for each branching point). */
typedef struct fib_rtlist {
        struct fib_rtlist *fibrt_prev;
        struct fib_rtlist *fibrt_next;
        u_int64_t         fibrt_bitmap; /* variable length destination bitmap */
        struct route      fibrt_rt;     /* route (dest/if/gway) */
        struct in_addr    fibrt_dst;    /* nexthop address */
} fib_rtlist_t;


/* SIM FIB info (for each generation counter). */
typedef struct fib_entry {
        struct sim_fib     *group_fib;        /* prt to head of our group FIB */
        struct fib_entry   *fibentry_prev;    /* ptr to prev fib entry
                                                 with different gen counter */
        struct fib_entry   *fibentry_next;    /* ptr to next fib enry
                                                 with different gen counter */
        u_int32_t          fibentry_gen;      /* generation counter */
        u_int8_t           fibentry_ndest;    /* number of destination for
                                                 this generation of FIB */
        struct callout     fibentry_deltimer; /* SIM Timer:- SIM_FIB_Aging */
        u_int8_t           fibentry_rtnum;    /* number of branch routes */
        struct fib_rtlist  *fibentry_rtl;     /* fib route list */
} fib_entry_t;

/* SIM FIB for each sender-group. */
typedef struct sim_fib {
        struct sim_fib     *sim_fib_prev;     /* ptr to previous SIM FIB entry
                                                 with the same hash value */
        struct sim_fib     *sim_fib_next;     /* ptr to next SIM FIB entry
                                                 with the same hash value */
        struct in_addr     sim_fib_s_addr;    /* sender address */
        struct in_addr     sim_fib_gr_addr;   /* multicast group address */
        u_int32_t          sim_fib_last_gen;  /* generation counter */
        struct fib_entry   *sim_fib_entry;    /* ptr to sim fib route entry */
        struct callout     sim_fib_deltimer;  /* SIM Timer:
                                                 First - SIM_FIB_Timeout
                                                 Second - SIM_FIB_Aging */
```

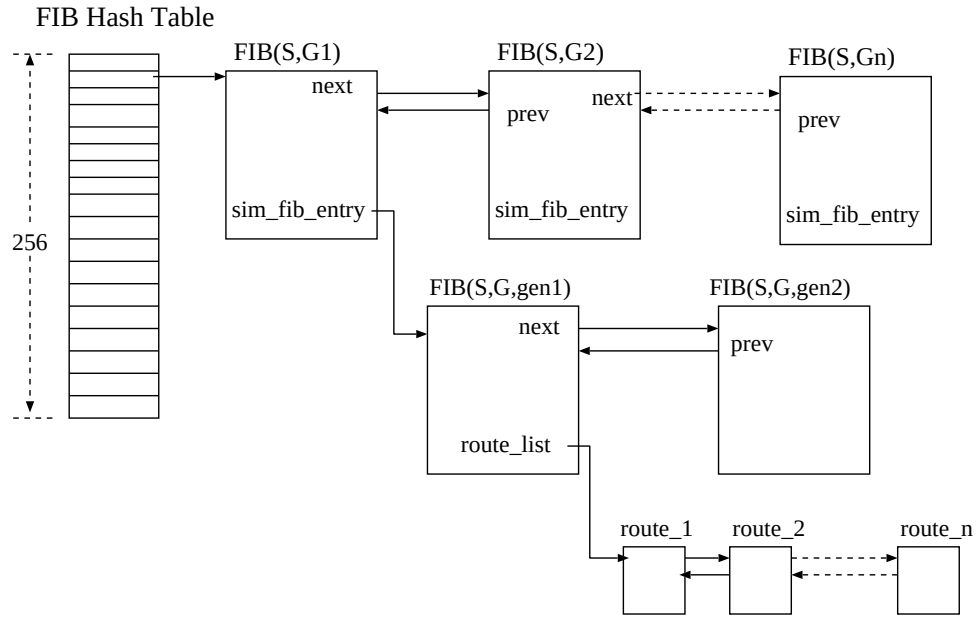



Figure 3.15: SIM FIB entry overall structure at kernel level

```
        u_int8_t            sim_fib_lost;      /* flag to check whether we got
                                                  a new FIB in last
                                                  SIM_FIB_TIMEOUT */
        u_int8_t            sim_fib_delflag;   /* SIM FIB delete flag */
} sim_fib_t;
```

The overall structure of the SIM FIB entry in the kernel is shown in Figure 3.19.

At the present implementation, the length of SIM FIB hash table is fixed to 256 bytes.

```
#define SIM_FIBTBL_SIZE    256      /* size of SIM FIB hash table */
extern struct simhash simfibtable[SIM_FIBTBL_SIZE];
```

### 3.6.3 User level - Kernel Interfaces

Communication between the user level and the kernel is established by using *setsockopt()* function. *IP_SIM_ADD* and *IP_SIM_SETFLAG* are newly defined in `<netinet/in.h>` header file as a new socket option.

- Add receiver list

  ```
  (setsockopt(s, IP_PROTO_IP, IP_SIM_ADD, &rcvlist, sizeof(receiverlist);
  ```

  Adding a receiver is done by using *IP_SIM_ADD* to pass a receiver address list (*rcvlist*) and SIM flags to the kernel. The definition of address list is shown below:

```
typedef struct addr_list{
        u_char              al_flags;
        u_char              al_ndest;
        struct in_addr      *al_addr;
} addr_list_t;
```

- Delete FIB entries or temporarily change the address list

  `(setsockopt(s, IP_PROTO_IP, IP_SIM_SETFLAG, &rcvlist, sizeof(receiverlist);`

  *IP_SIM_SETFLAG*, when used during the transmission, is for setting a *Temporary Flag* to SIM header, in case that the receiver list is temporarily changed. When the transmission is finished, *IP_SIM_SETFLAG* is also used to delete FIB entries on SIM routers. However, this is not compulsory, because FIB entries will be deleted after the *SIM_FIB_TIMEOUT* timer expires.

- Compute Data size for a packet

  In order to avoid fragmentation occurring in UDP transmission, the application has to pass a buffer not larger than *SIM_MAX_BUF_SIZE* size. The definition of *SIM_MAX_BUF_SIZE* is shown below:

```
#define ALL_HDR(ndest,options)  ((options == 1) ? (sizeof(struct udpiphdr)+ \
            SIM_HDRLEN_LIST(ndest) +sizeof(struct ip) +SIM_ OPTLEN(ndest) ): \
           (sizeof(struct udpiphdr) +SIM_HDRLEN_LIST(ndest) +sizeof(struct ip))

#define SIM_MAX_BUF_SIZE(ndest,options)  (IP_MSS - ALL_HDR(ndest,options))
```

- Show statistics of SIM routers

  Figure 3.16 shows the SIM variables that can be shown or modified by *sysctl()* system call.

```
net.inet.sim.simfibaging: 0
net.inet.sim.simfibinterval: 0
net.inet.sim.simfibtimeout: 0
net.inet.sim.simhellointerval: 0
net.inet.sim.sim_enable: 0
net.inet.sim.sim_gw_router: 255.255.255.255
```

Figure 3.16: Sysctl

The variable *net.inet.sim.simfibaging*, *net.inet.sim.simfibinterval*, and *net.inet.sim.simfibtimeout* are the value of *SIM_FIB_AGING*, *SIM_FIB_INTERVAL*, and *SIM_FIB_TIMEOUT*, respectively. *net.inet.sim.sim_enable*, when set to 1, means that the host will be

perform as a SIM-enable router. *net.inet.sim.sim_gw_router* shows the SIM first hop router as seen from the sender.

Another way to show statistics of SIM router is to use our developed *simstat* utility. *Simstat* shows the following information.

```
struct simstat {
        u_quad_t simrts_bad_sourceaddr; /* packets that source address is
                                         * multicast address (may be a DoS) */
        u_quad_t simrts_bad_groupaddr;  /* packets that group address is not
                                         * in multicast address (not SIM) */
        u_quad_t simrts_bad_bitmap;     /* number of packets with all
                                         * bitmap turn off                */
        u_quad_t simrts_bad_receivers;  /* number of receivers exceed
                                         * SIM_MAX_DEST                   */
        u_quad_t simrts_not_support;    /* not support length, etc.       */
        u_quad_t simrts_bad_gen;        /* bad generation number          */
        u_quad_t simrts_no_fib;         /* no appropriate gen of FIB      */
        u_quad_t simrts_error;          /* unknown error                  */
        u_quad_t simrts_proto_nosup;    /* no suppport upper layer protocol */
        u_quad_t simrts_fake_redirect;  /* fake redirect message           */
        u_quad_t simrts_true_redirect;  /* true redirect message           */
        u_quad_t simrts_fibcount;       /* number of FIB registered        */
        u_quad_t simrts_simfib_lookups; /* # forw. cache hash table hits   */
        u_quad_t simrts_simfib_misses; /* # forw. cache hash table misses */
        u_quad_t simrts_no_route;       /* no route for packet's origin    */
        u_quad_t simrts_bad_tunnel;     /* malformed tunnel options        */
        u_quad_t simrts_cant_tunnel;    /* no room for tunnel options      */
        u_quad_t simrts_wrong_if;       /* arrived on wrong interface      */
        u_quad_t simrts_pkt2large;      /* pkts dropped - size > BKT SIZE  */
        u_quad_t simrts_upq_sockfull;   /* upcalls dropped - socket full   */
        u_quad_t simrts_rcv_pktcount;   /* number of received packets      */
        u_quad_t simrts_rcv_bytecount;  /* byte of received packets        */
        u_quad_t simrts_snd_pktcount;   /* number of sent packets          */
        u_quad_t simrts_snd_bytecount;  /* byte of sent packets            */
};
```

- Configure the SIM first hop router

  As described above, *sysctl* system call can be used only to show the SIM first hop router. The administrator of the sending host has to configure the SIM first hop in either way: specifying in */etc/rc.conf* file or using our developed *sim1sthop* utility.

Figure 3.17 and figure 3.18 depict the overall of SIM data transferring process and a sample of SIM sending program, respectively.

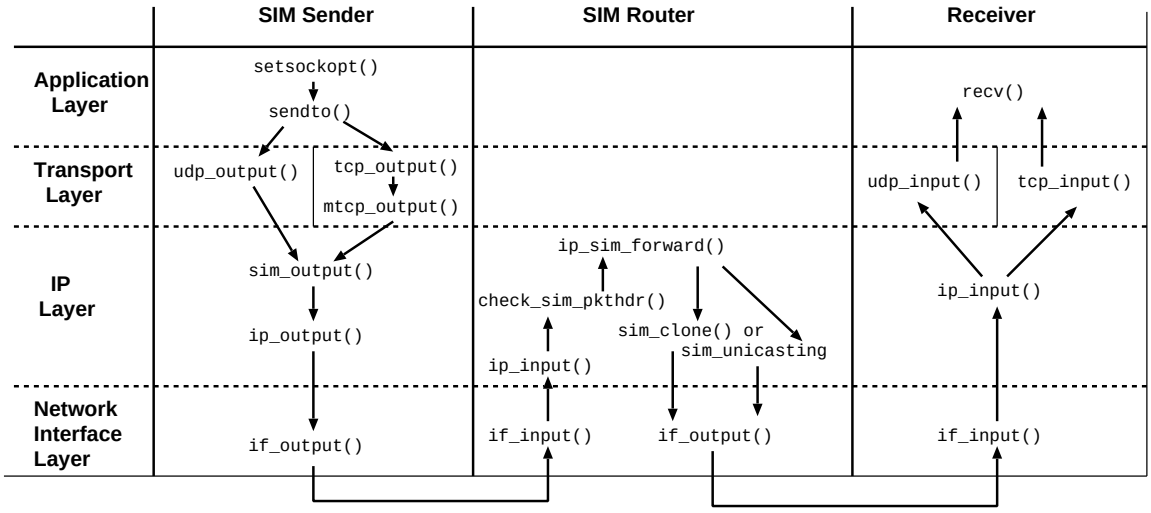



Figure 3.17: Overoview of SIM data transferring process

### 3.6.4 Kernel Support

#### Kernel Support for SIM senders

- `sim_timer_fib_interval()`
  Increase generation counter, when the FIB Interval Timer to attach receiver list to packets(active only on SIM senders).
- `sim_random_gen()`
  Randomly create 32 bits of generate a generation number for SIM FIB.
- `sim_output()`
  Allocate mbuf for outer IP header, SIM header and options.
- `sim_innerip_cksum()`
  Calculate checksum for upper layer protocol and inner IP.

#### Kernel Support for SIM routers

- `sim_timer_fib_timeout()`
  Check timeout for SIM routers to receive new FIBs. If the router did not receive an FIB entry for two *SIM_FIB_INTERVAL* periods, the router will delete the old FIB entry by passing it to *sim_fib_delete* function.
- `sim_timer_fib_aging()`
  Delete an old SIM FIB entry, after *SIM_FIB_TIMEOUT* is timeout.
- `sim_fib_delete()`
  Delete SIM FIB for the group.

```
#include <hoge.h>

int main()
{
        struct addr_list receiverlist;

        receivelist.al_flags |= SIM_PRESET_FLAG; /* set flags */
        receiverlist.al_addr[0].s_addr = xxx;
        receiverlist.al_addr[1].s_addr = yyy;
        .....                                                          10
        receiverlist.al_addr[n-1].s_addr = zzz;

        /* Add receiver list */
        if (setsockopt(sender_socket, IPPROTO_IP,IP_SIM_ADD, &receiverlist,
                        sizeof(receiverlist)) < 0) {
                perror("cannot setsockopt IP_SIM_ADD");
        }

        while(1) {
                sendto(sender_socket, ....);                           20
                receivelist.al_flags |= SIM_DELETE_FLAG;
                ....
        }

        /* Set SIM_DELETE_FLAG to the last packet of the data flow */
        if (setsockopt(sender_socket, IPPROTO_IP, IP_SIM_SETFLAG, &receiverlist,
                        sizeof(receiverlist)) < 0) {
                perror("cannot setsockopt IP_SIM_SETFLAG");
        }
                                                                       30
        close(sender_socket);
}
```

Figure 3.18: A sample of sending program

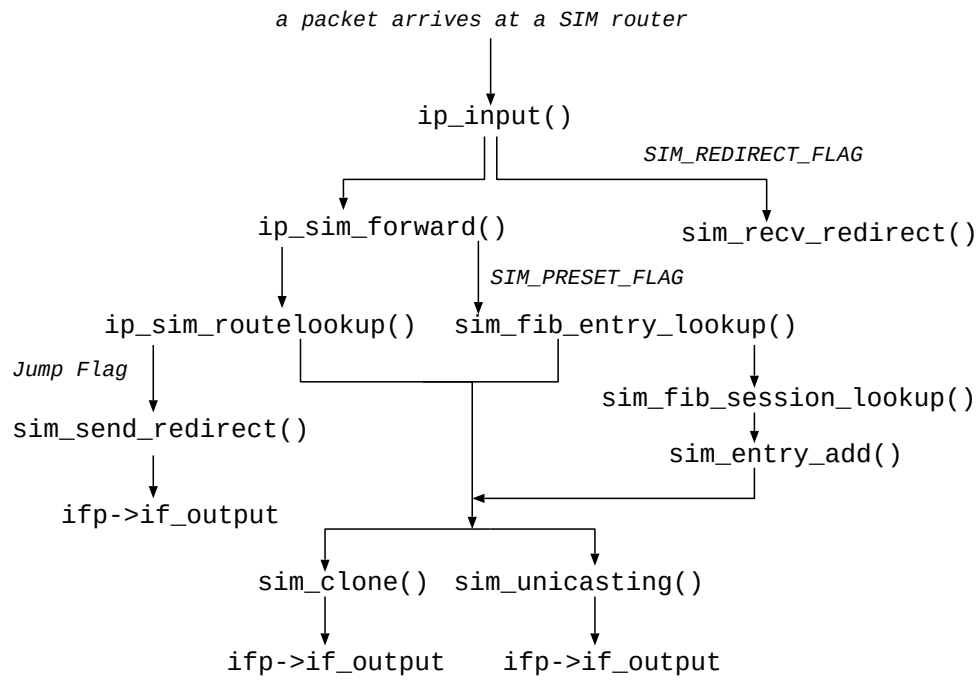



Figure 3.19: Forwarding process on a router

- `sim_fib_delete_entry()`
  Delete a SIM FIB entry (not the first entry of the group).

The followings are the definitions of each timer.

```
#define  SIM_FIB_AGING          (10*hz)
#define  SIM_FIB_INTERVAL       (10*hz)
#define  SIM_FIB_TIMEOUT        (60*hz)
#define  SIM_HELLO_INTERVAL     (30*hz)
```

- `sim_init()`
  Description: Initialize SIM FIB table
- `check_sim_pkthdr()`
  Check SIM packet coming from *ip_input()*, before forward mbuf to *ip_sim_forward()*. If any error is returned, the packet will be dropped.
- `ip_sim_forward()`
  Forward SIM packet from router to router, or from router to receiver nodes.
- `ip_sim_routelookup()`
  Lookup unicast routing table, and copy route entry to *routetab* structure.

- `sim_clone()`
  Duplicate mbuf and rewrite some headers as necessary.
- `sim_unicasting()`
  Duplicate data and rewrite packet header as unicast packet.
- `sim_fib_session_lookup()`
  Find the same group (s,g) from the old list.
- `sim_fib_entry_lookup()`
  Look up for the same sender, group, and generation number of FIB entry from the SIM FIB hash table.
- `sim_fib_entry_add()`
  Create and add new FIB entry to hash table.
- `sim_send_redirect()`
  Send SIM redirect message to previous branching router.
- `sim_recv_redirect()`
  Receive SIM redirect message from next branching router, and rewrite SIM FIB entry.

The structure of SIM redirect header is defined as follows:

```
struct sim_redirect_hdr {
        u_int8_t  sim_flags;          /* several flag */
        u_int8_t  sim_pad;            /* padding */
        u_int8_t  sim_hlen;           /* length in units of 4 octets */
        u_int8_t  sim_ndest;          /* # of destinations  */
        u_int32_t sim_gen;            /* generation count */
        u_int64_t sim_bitmap;         /* variable length destination bitmap */
        struct in_addr s_addr;        /* sender ip address */
        struct in_addr gr_addr;       /* group ip address */
};
```

## 3.7 Preliminary Evaluation

Table 3.3 compares characteristics and performance of IP multicast, Xcast and SIM. First, in IP multicast model, receivers are not specified in multicast packets and multicast routers use multicast address (class D) to identify and forward multicast traffic. However, in Xcast model, as the sender attaches receiver list to all multicast packets, routers can forward the packets to each receiver along unicast routing path. It does not need a unique multicast address. Optionally, a sender can specify a channel identifier in the packets. In this case, (source address, channel identifier) uniquely identifies the channel for several purposes, such as error and flow control. However, SIM requires (source address, multicast address) as group identification for operating

Table 3.3: Comparison of IP multicast, Basic Xcast, and SIM

| | **IP multicast** | **Xcast** | **SIM** |
|---|---|---|---|
| Group identification | Multicast address | (Source, Channel id) as an option | (Source, Multicast address) |
| Attachment of receiver list | None | Every packets | Some packets (preset mode) Every packets (list mode) |
| Overhead of packet header | None | Receiver addresses | Receiver address and outer IP header |
| Maintaining forwarding information | All on-tree routers | On sender host or or in packets | Branching routers |
| Packet routing | multicast routing | uses unicast routing | depends on unicast routing |
| Number of forwarding table lookup | 1 | Random ($\leq N$) | 1 (preset mode), Random ($\leq N$) (List Mode) |
| Header modification while packets forwards through branching routers | None | Bitmap and header checksum | Bitmap, src and dst field in outer IP header |
| Trasmission packets to a subset of receivers | Impossible | Capable by modifying bitmap | Capable by using temporary flag |
| Gradual deployment | IP tunnel | IP tunnel and premature X2U | IP tunnel conversion to unicast |

FIB on routers. In list mode, SIM acts as an Xcast protocol, and requires the sender to attach receiver list to all packets. On the other hand, the sender can periodically attach receiver list in preset mode.

Comparing number of forwarding table lookup per router, IP multicast routers and SIM routers in preset mode use group identifier to look up the forwarding table, therefore, they require only one lookup. However, Xcast routers and SIM routers in list mode have to look up the forwarding table for each receiver specified in the packets. Thus, if there are N receivers in a group, they require less or equal times of lookup depending on the number of receivers specified in the packets.

For overhead on packet header, both SIM and Xcast need to attach receiver addresses to the packets. That is, the size of data that can be inserted in Xcast and SIM packets are smaller than that of current IP multicast. Although this packet overhead will become bigger as receivers increase, SIM and Xcast can better serve small communication group than IP multicast and also alleviate IP multicast issues described in Section 2.1.3.

Contrary to IP multicast, header modification occurs every time Xcast packets pass through a branching router. As Xcast routers have to modify the bitmap field in IP options, the checksum field in IP header has to be re-calculated. On the other hand, as SIM use IP-in-IP tunnel to forward the packet, it can modify bitmap in SIM header without re-calculation of checksum. However, it needs source and destination address field modified in order to specify the next tunnel egress router.

Another advantage of SIM and Xcast is the capability to transmit packets to a

subset of receivers, which will be useful for retransmission purpose. In Xcast, this can be achieved by modifying bitmap in packet header. SIM also adopts the same concepts with the combination of temporary flag and selective retransmission with modified bitmap.

Gradual deployment for IP multicast, Xcast, and SIM is achieved by IP tunneling. Xcast also provides premature Xcast-to-unicast (X2U) when there is no Xcast router on the next hop. As with Xcast, SIM also convert packet to unicast when there is no SIM routers on the next hop. However, conversion-to-unicast in SIM can also be introduced in all last branching routers. Therefore, the receiver can receive multicast packets without any kernel and application modification.

## 3.8 Performance Evaluation Methods: simulation and experiment

In this next section, I compare SIM in preset mode with Xcast4, and unicast. I have implemented SIM for IPv4 as a kernel option with kernel modifications for FreeBSD4.6-RELEASE. The existing implementation of Xcast4 is based on Linux kernel. However, as describe earlier, the sender sending packets through SIM in list mode attaches receiver list to every packets, and the SIM routers in list mode behaves as same as Xcast4 routers does. Therefore, in this dissertation I use SIM in list mode as a substitute for Xcast4.

To evaluate our protocol, first I perform the experiments from the implementation. However, the experiments with a limited scale of network and limited number of hosts cannot answer the question: how SIM can scale with number of groups in the real Internet. Therefore, we have also developed a simulators [52] and performs simulations by using the real network topology with a large number of multicast groups.

Before we discuss the performance evaluation results from the both implementation and the simulations in the next section, here we have to validate that both the simulators and the implementations are developed correctly. Our validation is done by using topology depicted in Figure 3.20. We will call this topology *Hop-count* topology through this dissertation. This topology consists of a sender, 31 routers, and 30 receivers. All nodes are Pentium 1 GHz with 512 MB of memory and well connected with 100 Base-T Ethernet links.

The topology is considered from the average hop counts studied in [55]. The study result in [55] shows that the IP-path length, i.e., number of hop counts that most senders can reach each receivers is approximately $15 \pm 4$ hops. That is, in multicast transferring, the 11th, 15th, and 19th routers are usually the branching points of the distribution tree. The position of the branching points is important for the performance evaluation of SIM, as SIM Tunnels are created among the branching points. The longer the tunnel, i.e. the distance between the branching nodes, the better performance SIM can gain.

In both experiment and simulation, we increase number of receivers from 6 to 30. To fix the position of the branching points although number of receivers increase, in

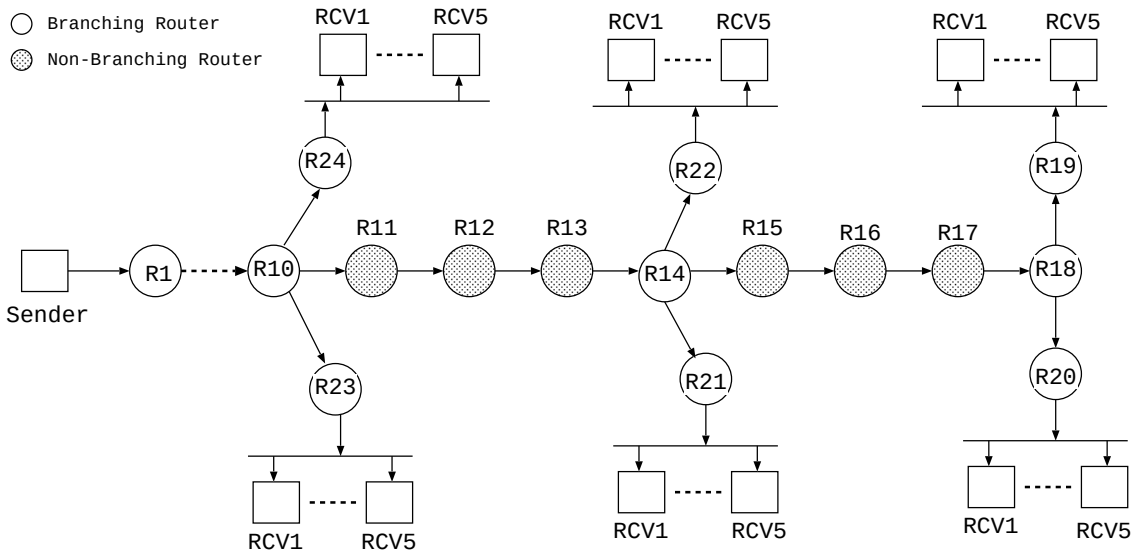



Figure 3.20: Topology used in validation of the simulator and the implementation

Table 3.4: Total number of routing table lookup observed from the simulations and experiments

| Number of receivers | Simulation | | Experiments | |
|---|---|---|---|---|
| | SIM | Xcast | SIM | Xcast |
| 6 | 288 | 1080 | 288 | 1080 |
| 12 | 288 | 2160 | 288 | 2160 |
| 18 | 288 | 3240 | 288 | 3240 |
| 24 | 288 | 4320 | 288 | 4320 |
| 30 | 288 | 5400 | 288 | 5400 |

each experiment and simulation, the sender sent 12 packets to 6, 12, 18, 24, and 30 receivers. That is, each last hop router has 1, 2, 3, 4, and 5 receivers in each experiment. The results of total routing table lookups are shown in table 3.4.

The results from Table 3.4 shows that number of routing table lookups by SIM is constant regardless to the number of receivers. We can explain that this is because SIM tunnels are created between SIM branching nodes, and all SIM branching nodes maintain SIM FIB. Therefore, the branching nodes require only one lookup for each packet, while non-branching nodes also forward packets as the packets are unicast packets. Thus, non-branching nodes are also required only one lookup for each packet. On the other hand, Xcast routers do not maintain the forwarding state. Their routing table lookups increase as the number of receivers increase. In addition, there is no tunnel between branching nodes. That is, non-branching nodes in Xcast models have to

perform routing table lookups. As a result, the total number of routing table lookups in Xcast model increases highly when the number of receivers increase. Finally, the results form the table show that simulations and experiments from the implementation produce the same value. Therefore, we can conclude that the simulator and the implementation perform correctly. Moreover, this simulator can be used to perform simulations with a large number of multicast groups in the real network topology, as we will see in the next section.

## 3.9 Performance Evaluation

In this section, I evaluate our protocol by comparing it with unicast, IP multicast and Xcast for IPv4 (Xcast4). The performance evaluation is considered in two perspectives [56]. From the users point of view, we are concerned with the performance metrics experienced by the applications. This application level evaluation is achieved by the experiments using the SIM implementation. From the network operator's point of view, the resource consumption in network links, the processing overhead and memory requirements on on-path routers are considered. This network protocol level performance is evaluated through the simulations, excluding the packet forwarding time, which is measured from the implementation.

### 3.9.1 Application Level Evaluation

The metrics I use to evaluate the application level performance are end-to-end delay occurred by each protocol, when comparing to one-to-one packet forwarding by unicast.

In this experiment, we use network topology in Figure 3.20. We measure the average end-to-end delay occurred on each receivers, when comparing to the one-to-one unicast transferring. Figure 3.21 shows the experimental result. This delay value is the average delay observed on all receivers. The results show that the average end-to-end delays of both SIM and Xcast increase as the number of receivers increase. One of the reasons is that the increase in number of receivers causes more overheads in routing table lookup on the on-path routers. The other reason is that more packet duplications, i.e. conversion-to-unicast, occur at the last hop router.

### 3.9.2 Network Protocol Level Evaluation

As described earlier, the performance of protocol can be evaluated by three dimension costs: bandwidth consumption, packet processing overhead, and state-and-signaling per session in router. In this section, we evaluate SIM performance comparing with those of PIM-SM and Xcast4, and demonstrate how SIM can achieve its goal, i.e. Point D in Figure 3.1.

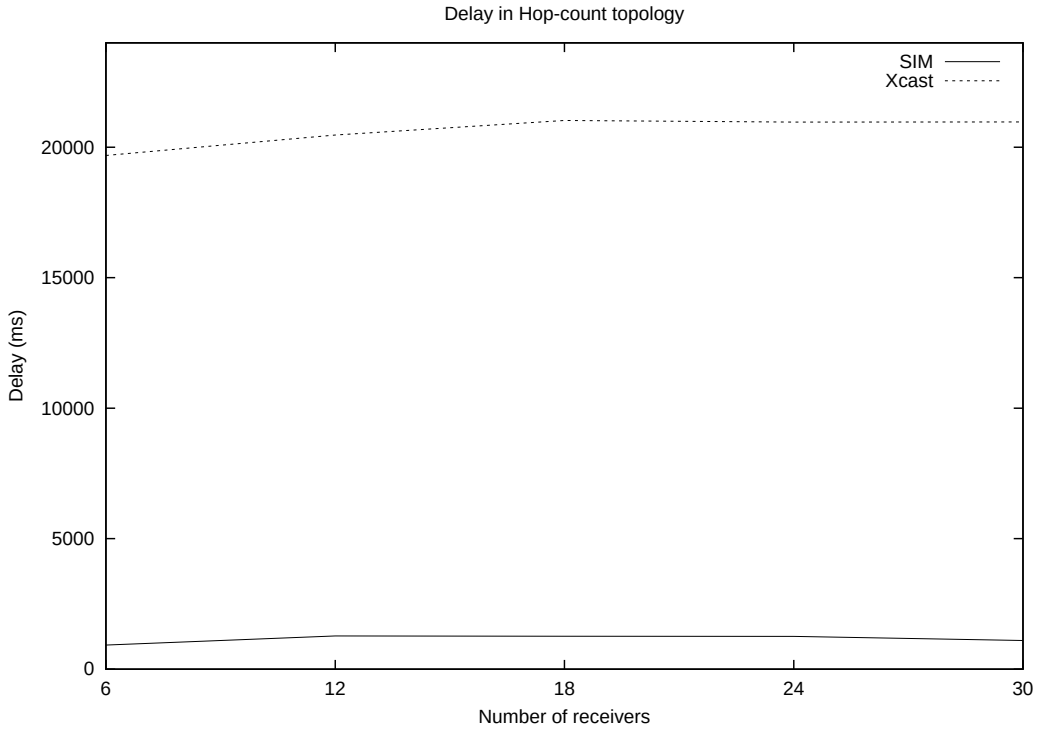



Figure 3.21: End-to-end delay for SIM, and Xcast4 comparing to Unicast

### Bandwidth Consumption

Bandwidth consumption can be considered as the overhead of a packet header in Xcast4 and SIM, as both require an additional newly defined header attached to the packets. Contrarily, the conventional IP multicast only requires the sender to specify a multicast group address as a destination. Hence, IP multicast produces no overhead in bandwidth consumption when compared to an ordinary unicast.

In the first simulation, we evaluate bandwidth consumption from the number of packets the sender has to transmit in order to gain a certain data rate. We assume that the sender transmits 512 Kbps and 2048 Kbps data flows toward 3 to 100 receivers using these three protocols. To make a distinction between SIM in the Preset mode and Xcast4, we set the total sending time to 10 seconds, which is equal to the *SIM_FIB_INTERVAL* time, and the path MTU to all receivers as 1500 bytes. The simulation results are shown in Figures 3.22 and 3.23.

We found that the Xcast4 sender has to send more packets than those of IP multicast and SIM in the Preset mode. At a sending rate of 512 Kbps for 50 receivers, Xcast4 and SIM have to send 15.5% and 2.4% respectively more packets than IP multicast. For 100 receivers at the same rate, the Xcast4 overhead becomes 40% while the SIM overhead remains the same. At a sending rate of 2048 Kbps for 50 receivers, Xcast4

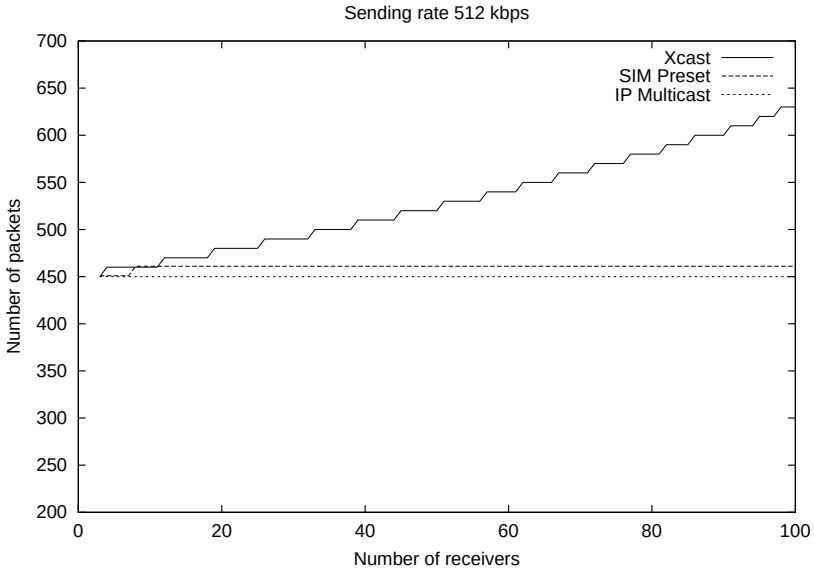



Figure 3.22: Number of receivers and sending packets for sending rate 512 Kbps

and SIM have to send 17.5% and 2.3%, respectively, more packets. For 100 receivers at the same rate, the overheads are 41.2% and 2.8%. Therefore, attaching a receiver list to every packet as in Xcast4 model introduces more bandwidth consumption than SIM in the Preset mode. This overhead becomes larger when the sending rate and number of receivers in a group increase. Note that the SIM sender always attaches a receiver list for every *SIM_FIB_INTERVAL* time, i.e. the number of packets in the Y axis will gradually increase according to the simulation duration time.

### Packet Processing Overhead

Packet processing overhead on routers can be observed from the forwarding time in the real data transferring. Moreover, this overhead is caused by packet duplication and route computation (routing table lookup). However, packet duplication occurs only on the multicast branching routers, and is the same for all three protocols: SIM, PIM-SM, and Xcast4. Therefore, I evaluate only route computation overhead in this dissertation.

1. **Packet Forwarding Time**

   We measured the average of packet forwarding time in the experiments described in section 3.8. This packet forwarding overhead is the consequence of the processing in SIM and Xcast forwarding routine. Table 3.5 shows the average packet forwarding time on branching routers for SIM and Xcast4.

   From Table 3.5, the SIM average forwarding time remains almost the same, and is not affected by the number of receivers. In contrast, for Xcast, the more the

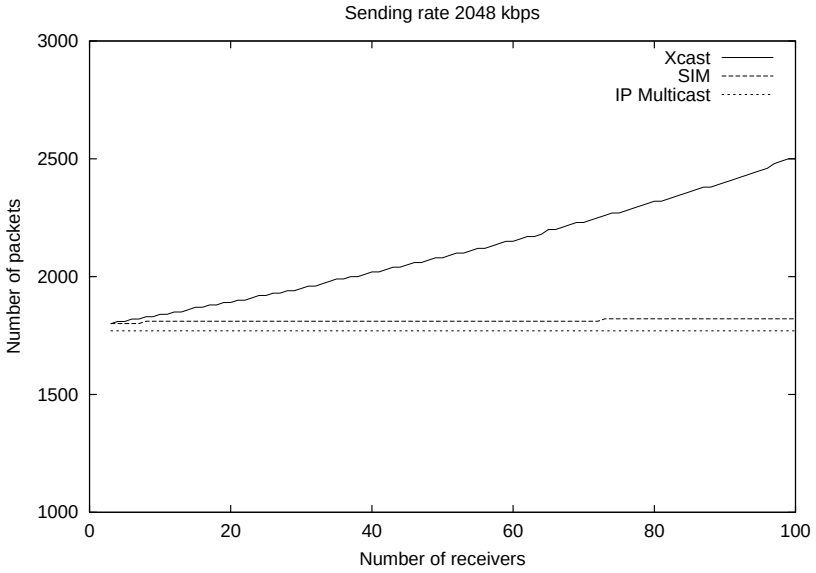



Figure 3.23: Number of receivers and sending packets for sending rate 2048 Kbps

Table 3.5: Average forwarding time of SIM and Xcast on branching routers ($\mu$s)

| Number of receiver | SIM | Xcast |
|---|---|---|
| 6 | 6,312.189 | 1,425,997.622 |
| 12 | 6,381.789 | 1,486,012.427 |
| 18 | 6,412.208 | 1,500,807.534 |
| 24 | 6,750.661 | 1,510,390.467 |
| 30 | 6,813.713 | 1,548,267.109 |

number of receivers increases, the more the average forwarding time increases. From this experimental result, we can confirm that the most important factor which causes SIM end-to-end delay, as shown in section 3.9.1, is not the overhead of routing table lookup, but the packet duplications on the last hop router. In SIM preset mode, the overhead of routing table lookup is alleviated by the SIM FIB entries.

2. **Route Computation**

   The route computation is evaluated through the two simulations. To compare route computation overhead, we measured the number of routing table lookups for each protocol. However, IP multicast uses a multicast address as group identification; thus, the number of routing table lookups should be the same as SIM in the Preset mode. Therefore, we measure only the number of routing table

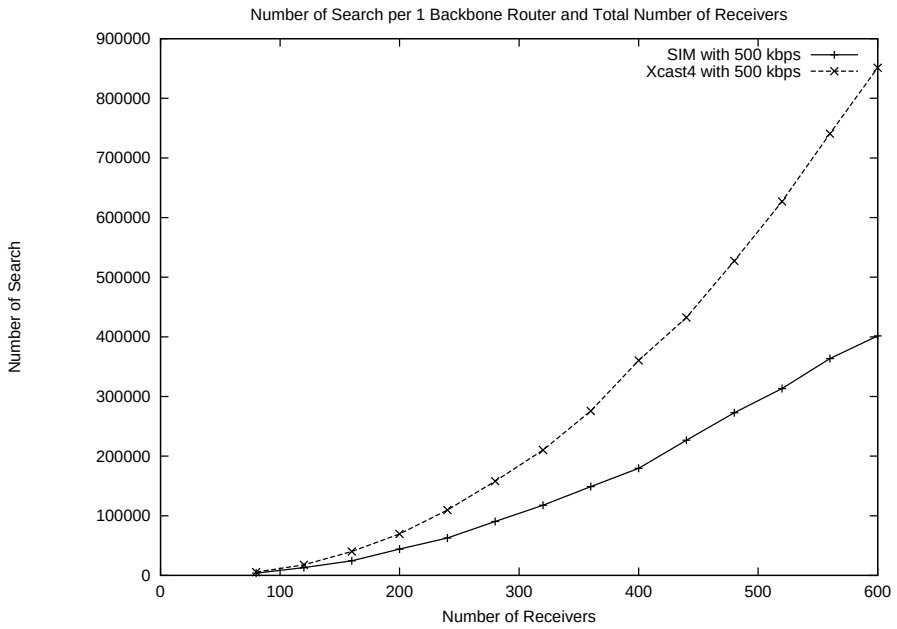



Figure 3.24: Routing table search and total number of receivers

lookups for Xcast4 and SIM.

In order to answer the question "how SIM can gain scalability in number of groups", we perform simulations with the actual topology of the Japan Gigabit Network (JGN) [57]. The JGN is a nationwide network designed for the research and development of very high-speed networking and high-performance application technologies. This network consists of 10 backbone routers connected with gigabit network links, and approximately 60 routers connected to thick links of 50 to 600 Mbps. We assume that each router, excluding backbone routers, connects with 4 end nodes; therefore, a total of 240 end nodes act as multicast senders and receivers in our simulations. Routing paths are assumed being symmetries. Again, and contrary to IP multicast, which is appropriate for large one-to-many multicast sessions, we assume small multicast sessions such as video conferencing and network games among only a few people. These kinds of sessions can be held using multiple one-to-many multicast sessions. That is, in our simulations a multicast group with $n$ members will contain $n$ multicast sessions. The sending rate for both simulations is 500 kbps.

In the first simulation, we randomly chose nodes as senders, varied the number of receivers in a multicast session from 2 to 15, and counted total routing table lookups performed on all routers. We fixed the number of the group at 40, i.e. for many-to-many applications such as video conferencing we will have 80 to 600 multicast sessions. Note that we do not perform more simulations with a further

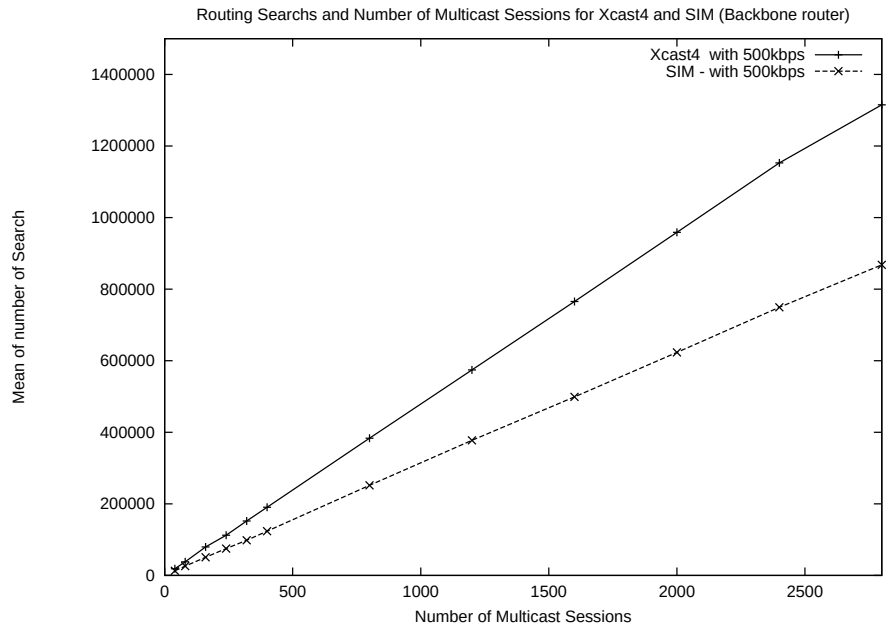



Figure 3.25: Routing table search and number of groups

number of groups, since more groups will introduce network congestion at the router next to the end nodes. This congestion will cause packet dropped on the routers and will affect number of routing table lookups.

In the second simulation, we assumed video conferencing by four people. Therefore, we fixed the number of receivers to 4, and varied the number of multicast groups from 10 to 700. Thus, we have 40 to 2800 multicast sessions. Results from both simulations are shown in Figures 3.24 and 3.25. From the results, we found that Xcast4 requires about 1.5 times as many as routing table lookups required by SIM. This is because SIM has a significant scheme as SIM FIB and SIM Tunnel, which helps in reducing the number of routing table lookups and the number of FIB entries maintaining on each router. These simulation results provide evidence that SIM can reduce packet processing overhead, and provide scalability to a large number of small groups.

### State-and-signaling per session

The state-and-signaling cost of each protocol depends on the implementations. However, the signaling cost per session is difficult for the quantitative evaluation, since it depends on the network topologies and number of nodes. On the contrary, for qualitative evaluation, the cost of signaling per session of each protocol is explicitly shown in Figure 3.1. IP multicast has to send join and leave message periodically and maintains forwarding state on all on-path routers in order to maintain distribution tree, while SIM

Table 3.6: Parameters observed from the simulation

| | |
|---|---|
| $N_{all}$ | Total number of all session *(S,G)* passing through a router |
| $N_{br}$ | Total number of session *(S,G)* that performed packet duplication or branched at the router |
| $N_d$ | Total number of packet duplication performed on the router |
| $S_{rt}$ | Size (bytes) of a PIM-SM routing entry for a *(S,G)* session |
| $S_{fe}$ | Size (bytes) of a multicast forwarding entry in the PIM-SM kernel |
| $S_{fib}$ | Size of a SIM FIB entry with a generation number for a *(S,G)* session |
| $B_{pim}$ | Size (bytes) of a PIM-SM downstream entry for a *(S,G)* session |
| $B_{sim}$ | Size of a SIM downstream entry for a *(S,G)* session |

has only Redirect Message and maintain forwarding state only on branching routers. Xcast has no state-and-signaling.

Concerning to the state cost, we evaluate the cost of by applying the performance metrics proposed in [58]. Billhartz *et al.* argued in section 4.5 of [58] that the intricacy of the protocol, operating system overhead, and routing table size become especially important issues when the number of senders and groups grow to a large size because router speed and memory requirements are influenced. Therefore, for the quantitative evaluation, we choose the routing table size and the average number of timers at each router as the performance metrics.

1. **Routing Table Size**

   PIM-SM and SIM require each router maintain a table of multicast routing information, while Xcast4 does not. Therefore, in this dissertation we compare only routing table size of PIM-SM and SIM. We do not use the equation proposed in [58] to calculate routing table size, because the different characteristics of protocols. PIM-SM maintains multicast routing information on every multicast router, while SIM maintains SIM FIB only on multicast routers that act as branching points. Moreover, as a backbone router has more chances to be a branching point, we focus on the routing table size on each backbone router. We perform a simulation with the same JGN network topology described in "Route Computation" of section 3.9.2, and examine the relationship between number of groups and routing table size. In our simulation, we fix the number of receivers to 4, and vary number of multicast groups from 10 to 10000. Therefore, in the simulations, number of multicast sessions varies from 40 to 40000. Each sender sent only one packet to the group. On 10 backbone routers, for each protocols, we observed parameters described in Table 3.6.

   Note that a PIM-SM routing entry $S_{rt}$ consists of information of source, group, and upstream router. Moreover, the number of packet duplication $N_d$ represents the number of downstream entries required on each router. As PIM-SM in our simulation constructs source-based trees for all *(S,G)* session, the average size of

routing table on per router $S_{pimsm}$ is given by the following expression.

$$S_{pimsm} = S_{rt} \cdot N_{all} + S_{fe} \cdot N_{br} + B_{pim} \cdot N_d \tag{3.1}$$

Contrary to PIM-SM, SIM maintains multicast routing information, i.e. SIM FIB entries only on if the router is a branching point. Therefore, the average size of routing table per router $T_sim$ is given by the following expression.

$$S_{sim} = S_{fib} \cdot N_{br} + B_{sim} \cdot N_d \tag{3.2}$$

To illustrate how the average size of routing table increases with the number of groups, we input values $N_{all}$, $N_{br}$, $N_d$ obtained from the simulation. The sizes of a routing entry $S_{rt}$, multicast forwarding cache entry $S_{fe}$, downstream entry $B_{pim}$, and the sizes of a SIM FIB entry $S_{fib}$, SIM destination entry $B_{sim}$ are obtained from the header file of each implementation. The values of $S_{rt}$ and $B_{pim}$ are examined from the header file *mrt.h* of PIM-SM in GateD (5.0 alpha 0) disclosed on [59]. Moreover, the value of $S_{fe}$ is examined from the header file *ip_mroute.h* in the pim-patched kernel [60]. We found that $S_{rt}$, $B_{pim}$, $S_{fe}$, $S_{fib}$, and $B_{sim}$ are respectively 148, 32, 74, 88, and 36 bytes. Figure 3.26 shows the growth of memory required on a backbone router with the growth in number of groups.

Moreover, from the graph in Figure 3.26, we can see a relationship between number of group and an approximate size of routing table for each protocol as follows:

$$S_{pimsm} \approx 505 \cdot G \tag{3.3}$$

$$S_{sim} \approx 210 \cdot G \tag{3.4}$$

To illustrate how many groups each protocol can support, we give the maximum memory size of a router to equation 3.3 and 3.3. To the best of our knowledge, the maximum memory size on a backbone router at present is 128 Mbytes. Therefore, an approximate maximum number of multicast groups a PIM-SM and a SIM router can support are 253,465 and 609,524, respectively. We can conclude that, on average, each SIM router requires approximately 41% of routing table size required by a PIM-SM router.

2. **Number of Timers**

According to the PIM-SM protocol specification [28], there are three types of timers relating to tree maintenance:

- Timer for each interface in the outgoing interface list A timer kept per outgoing interface of each route entry is *Oif-Timer* which is used to time out that outgoing interface, and update the routing information.

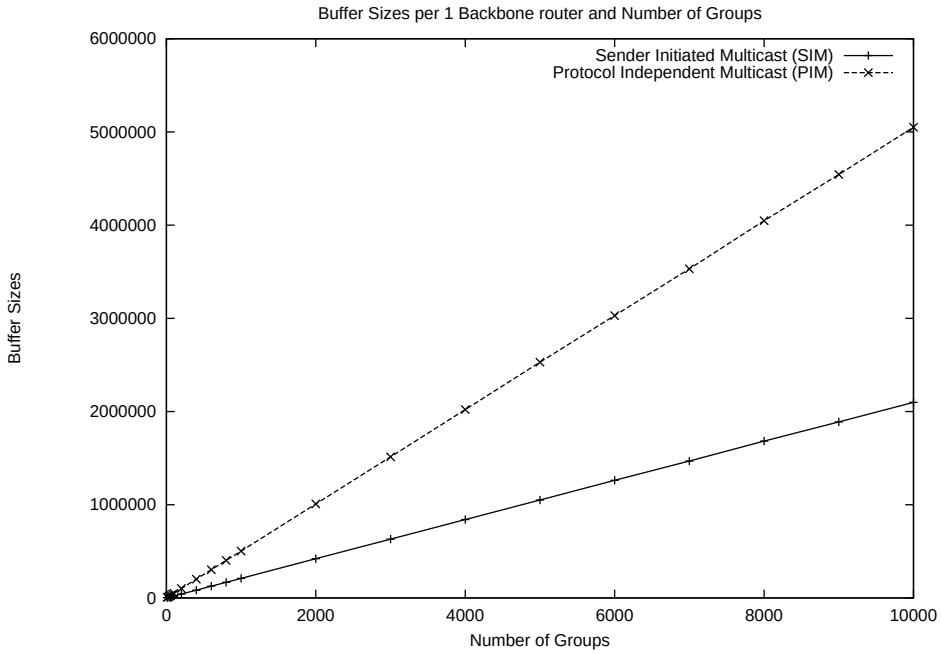



Figure 3.26: Routing table size on a backbone router with the growth in number of groups

- Timer for the multicast routing entry itself There are three timers: Entry-Timer, Assert-Timer, Random-Delay-Join-Timer are kept per *(S,G)* or *(*,G)* route entry. Entry-Timer is kept per route entry, and is used to time out that entry. Assert-Timer is used for timing out Asserts received. The Random-Delay-Join-Timer for an *(S,G)* or *(*,G)* route entry is used to prevent synchronization among downstream routers or a LAN when their RPF neighbor changes [28].
- Optional Timer For multi-access LAN, Join/Prune-Suppression-Timer, which is kept per route entry, may be used to suppress duplicate joins from multiple downstream routers. In addition, Designated Routers (DR) have a Register-Suppression-timer for each *(S,G)* entry and every router has a single Join/Prune-timer.

Note that, other PIM-SM timers are timers related to neighbor discovery, and timers relating to Rendezvous Point (RP) information.

While PIM-SM requires a router to maintain many timers, SIM requires router to maintain only *SIM_FIB_TIMEOUT* for each SIM FIB *(S,G)* entry, and *SIM_FIB_AGING* per a generation of SIM FIB *(S,G)* entry. Instead of using a timer to update routing information of the outgoing interface list, SIM routers update the routing information by the trigger of the new generation number of data packet.

The results of the performance evaluation are summarized in Table 3.7. SIM has less end-to-end delay than Xcast4. SIM bandwidth consumption is also much less than that of Xcast. Route computation is similar for IP multicast and SIM because both maintain the forwarding state, while Xcast requires much more route computation. This fact also reveals in the forwarding time of Xcast. The forwarding time required in Xcast is more than that required in SIM. The routing table size and number of timers in SIM model confirm that SIM gains more scalability in the number of groups than IP multicast.

Table 3.7: Performance Evaluation results of SIM, IP multicast, and Xcast4

| Evalution methods | Metrics | Comparison results |
|---|---|---|
| Experiments | End-to-end Delay | IP multicast $<$ SIM $<$ Xcast4 |
| | Forwarding Time | SIM $<$ Xcast4 |
| Simulations | Bandwidth Consumption | IP multicast $<$ SIM $\ll$ Xcast4 |
| | Route Computation | IP multicast $\cong$ SIM $<$ Xcast4 |
| Analysis | Timers | SIM $<$ IP multicast |
| Simulations and analysis | Routing Table Size | SIM $\ll$ IP multicast |

## 3.10 SIM Deployment

### 3.10.1 Multiple Receivers in LAN

In IP multicast model, when multiple receivers are on the same subnet (LAN), the last hop router has no need to duplicate the packet. The router just forwards the packet encapsulated by Ethernet frame with an Ethernet multicast address in the destination field [1]. The receivers receive packets addressed to the Ethernet multicast addresses that correspond to the multicast group address.

In this dissertation, we use SIM conversion-to-unicast function in order to support the incremental deployment on the Internet. The advantage of this approach is that the receivers can receive the packet through the existing unicast applications. The disadvantage of this approach is that, even when multiple receivers are on the same subnet, the last hop router duplicates packets and performs SIM conversion-to-unicast for each receiver. Therefore, the duplication overhead is be added to the forwarding process, i.e. the end-to-end delay increases.

However, this problem can be resolved by the supporting of the last hop routers. Consider Figure 3.27, where multiple receivers are in the same subnet, and the routers *R3* and *R4* are the last hop router, which are *SIM-capable*. In this case, all we need in order to support Ethernet multicast address is: (1) supporting IGMP Join message sending from the receivers to the last hop router, (2) decapsulating the packet instead of performing conversion-to-unicast. As described in section 3.3, the inner IP header,

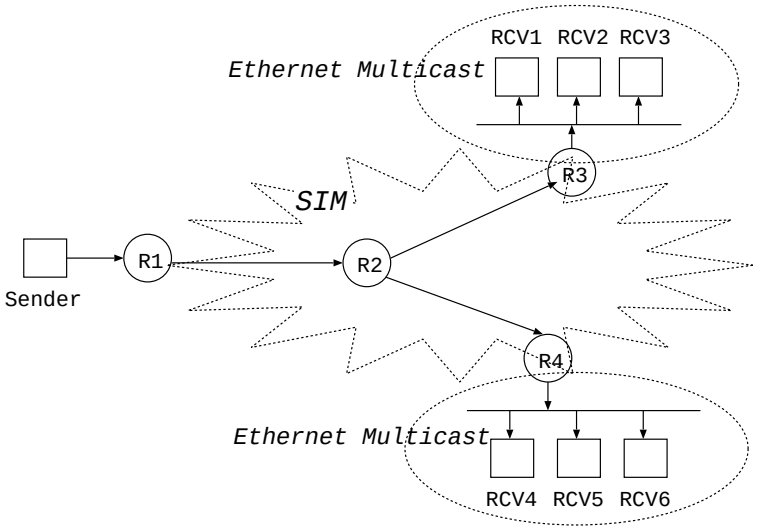



Figure 3.27: Deployment situation of SIM

with multicast address as the destination address, is encapsulated in the SIM packet header. Therefore, SIM routers can easily convert the SIM packet by decapsulting the outer IP header and the SIM header fields. The receivers send IGMP Join message to the router to make its own Ethernet interface mapping the multicast group address. Then, the receivers can receiver SIM packets through the multicast applications. Note that, in this case, the last hop SIM routers, i.e. *R3* and *R4* in Figure 3.27 do not forward any Join message to other IP multicast routers in the Internet, but forward packets only in the subnet.

### 3.10.2 SIM Gradual Deployment

Consider the SIM gradual deployment, the first question comes to our mind is *where to place a SIM router on the Internet*. Then, the second question is *how many SIM routers are required*. Finally, the last question is *how does the packet processing overhead change, when number of network branches increases*.

To answer these questions, we categorized network topologies into three categories: *bus*, *tree*, and *star* topology. The Internet topology is considered be the mixed network of these topologies.

First, in order to answer the first and second question, we performed two experiments per bus and tree topologies. Figure 3.28 and 3.29 show bus and tree topologies using in our experiments, respectively. Each topology consists of a sender, 31 routers, and 32 receivers. All nodes are well connected to each other with 100 Mbps link. Note that, the worst cast of SIM is when every router becomes the branching points of the distribution tree because SIM cannot create a SIM Tunnel. Therefore, to evaluate SIM in the worst case, we place one receiver next to each router in bus and tree network.

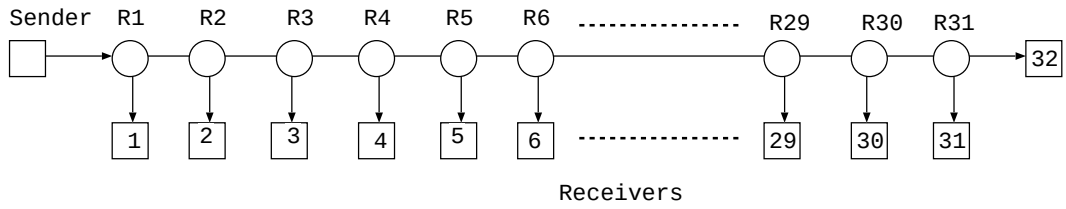



Figure 3.28: Bus topology

Moreover, number of receivers next to each router is not the real factor we have to consider.

1. Where to place SIM routers in the Internet ?

   The first experiment is to answer the first question: *where to place SIM routers in the Internet*. We assume that there is only one SIM router in each topology. The place where we locate the SIM routers is gradually changed from the first router to the last routers. In each experiment, we sent 12 packets to 32 receivers and observed total routing table lookups. Note that, as we can see from results in section 3.8 and section 3.9.1 that the end-to-end delay is proportion to number of routing table lookups, therefore measuring only number of routing table lookups is enough to evaluate our protocol. Figure 3.30 and 3.31 depicts total routing table lookups in Bus and Tree topology, respectively.

   Figure 3.30 shows that, when a SIM router is the 2nd router of the bus topology, total routing table lookups slightly increases. This is because the duplicated packets have been forwarded back to the first routers. However, when the SIM router is located near the center of the network, total routing table lookups increases. This is also the result from duplicated packets passing through each network link. At the point of router number 15th and 16th, the duplicated packets are equally forwarded to the previous and next routers. Therefore, total routing table lookups is the least in this experiment.

   Figure 3.31 shows that increase gradually. That is, the farther the SIM router from the sender, the more routing table looking up requires. Number of duplicated packets is the same for routers who have the equal hop length from the sender.

   We conclude that the best place to deploy SIM is the center of a bus network; while in a tree topology, the best place to deploy SIM is the router nearest to the sender node.

2. How many SIM routers are required ?

   In order to answer the next question, *how many SIM routers are required*, we performed the second experiment. This experiment used the same topologies described above. Contrary to the first experiment, we deploy multiple SIM routers

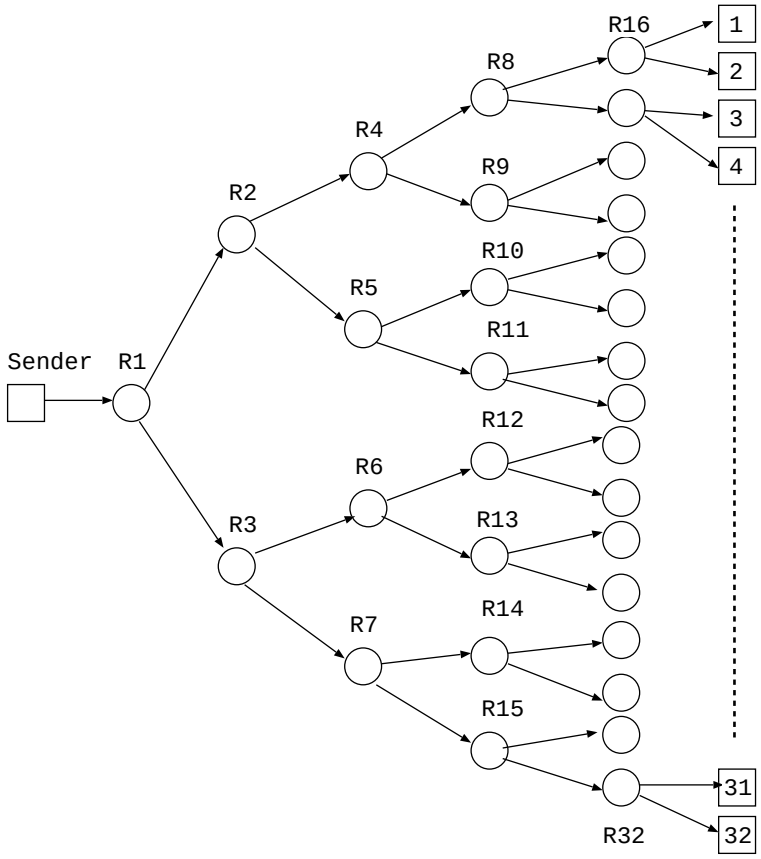



Figure 3.29: Tree topology

gradually from the nearest router to the farthest one, when looking from the sender. Figure 3.32 and 3.33 depicts total routing table lookups for a bus network and tree network, respectively.

Both Figure 3.32 and 3.33 show that when number of SIM routers increases with a bus topology, the total routing table lookups gradually decreases. This means that the more SIM routers deployed, the more packet processing overhead is reduced. Moreover, we can observe that the slopes in both graphs do not decline linearly. Total routing table lookups decreased slightly when all routers are deployed SIM. In other words, we can gain benefits from SIM, only if half of all on-path routers are deployed our protocol.

3. How does the packet processing overhead change, when number of network branches increases ?

   In order to answer the last question: *how does the packet processing overhead change, when number of network branches increases*, we performed the last experiment with Star topology (Figure 3.34). This topology consists of a sender, a

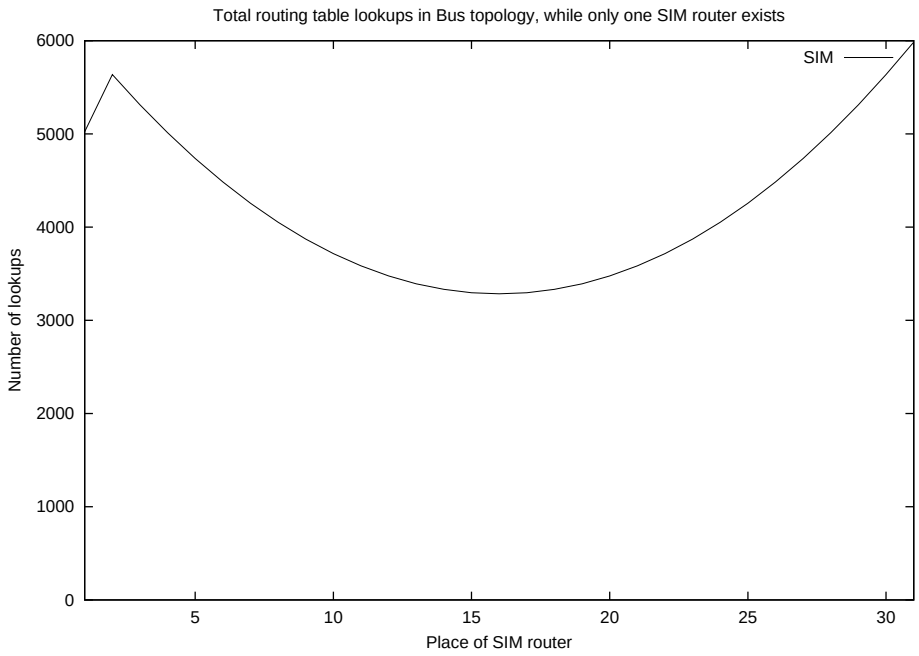



Figure 3.30: Total routing table lookups in Bus topology, while only one SIM router exists

SIM router, and 6 nodes acting as receivers. Each receiver node is alias to 10 IP addresses, therefore we have totally 60 receivers. Number of network branches is represented by number of receivers, which gradually increases from 6 to 58 during this experiment, while 58 is the maximum number of receivers supporting by the present SIM implementation. The measurement parameter is the average forwarding time on the router, which can be considered as the router overhead. We do not measure number of routing table lookups because it is obviously equal to number of receivers in the experiment. Table 3.8 shows the experimental results measured on a computer, with 450 MHz of CPU and 128 MB of memory, acting as our SIM router. Results from this figure shows that the average forwarding time increases as the number of network branches or receivers increases. However, this increment is not proportion to number of receivers, as we can see that when number of receivers is 54, the average time is only about six times of the average time of 6 receivers.

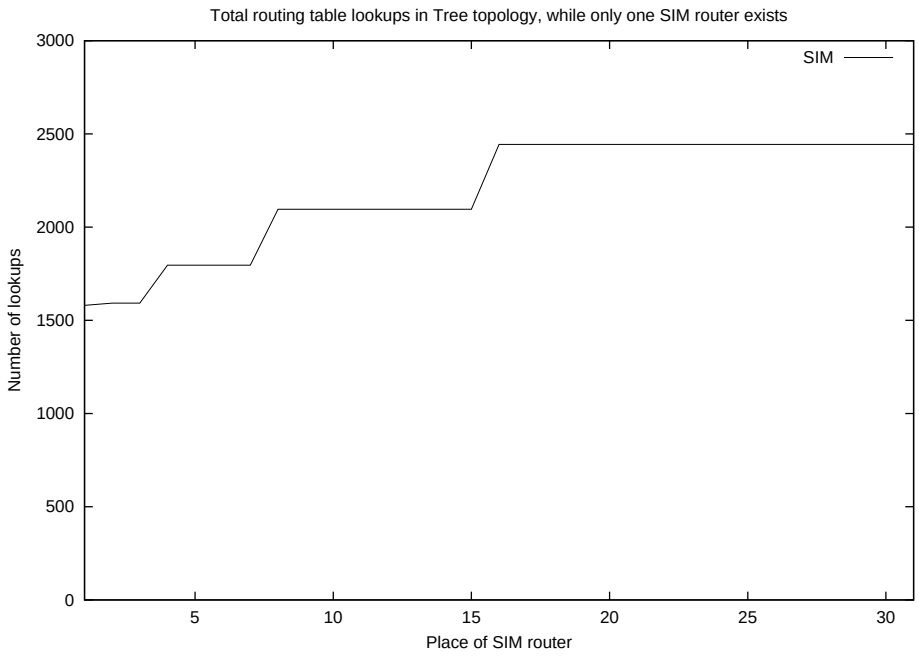



Figure 3.31: Total routing table lookups in Tree topology, while only one SIM router exists

## 3.11 Security Considerations

### 3.11.1 SIM FIB attacks

It is possible for malicious nodes to send a SIM packet with a bad receiver list in order to confuse SIM FIB management. It is possible for a malicious node to send a SIM packet with the Delete flag in order to schedule to remove a valid SIM FIB entry. Because the malicious nodes may be able to receive packets or may inhibit valid receivers from receiving packets, some protection mechanisms need to be deployed to protect valid SIM FIB entries from malicious nodes. Strict update rules for SIM FIB entry might be a possible solution. An example of such rules is to prohibit updating a SIM FIB entry except when routing change is detected. However, this rule might make things worse because malicious nodes could send illegal SIM packets with various generation counts before valid SIM packets are sent. Another solution, which we currently adopt, was to generate the initial generation counter randomly.

### 3.11.2 Generating a DoS attack

Some forms of DoS attacks can take advantage of the multicast characteristics to increase their effects. An example of such an attack is the smurf attack, which uses ICMP Echo request to a multicast or broadcast address with a source address that is

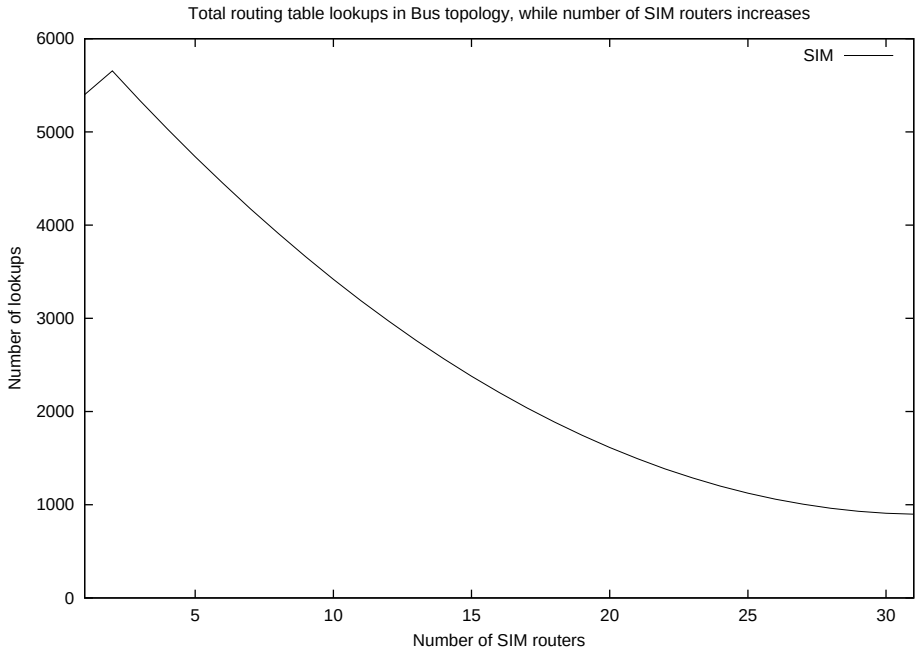



Figure 3.32: Total routing table lookups in Bus topology, while number of SIM routers increases

the target of the attack. The traditional multicast, which uses multicast address for the destination address, deals with this problem by setting up a router or a host not to reply to ICMP requests with multicast or broadcast address.

In Xcast model, the DoS effect is significantly diminished because of the small number of members in a multicast group. However, the attacker can use multiple multicast groups in order to create a DoS attack as effectively as in the traditional multicast model. It will be more difficult to deal with this problem because the destination may not recognize that it is the Xcast traffic. One strategy proposed in "Security Framework for Explicit Multicast" [61], is to use IPsec for securing explicit multicast traffic. However, it only provides confidentiality and/or authentication between sender and receivers. It cannot restrict nodes that can forward packets to Xcast routers. Thus, it may not be a good solution to stop a smurfing DoS attack.

In the case of SIM, the same smurfing DoS attacks may occur. The attacker may attack a host by using SIM packets to send ICMP Echo request. However, similar to Xcast protocols, SIM supports only small group multicast, so we also can say that the smurfing DoS attacks may not be effective. Even though, detection for this DoS attack can probably be provided.

Basically, at the last SIM branching router, the source and destination fields of the packets will be replaced with the sender unicast address and the receiver address. Therefore, a receiver cannot distinguish the receiving ICMP Echo request, whether

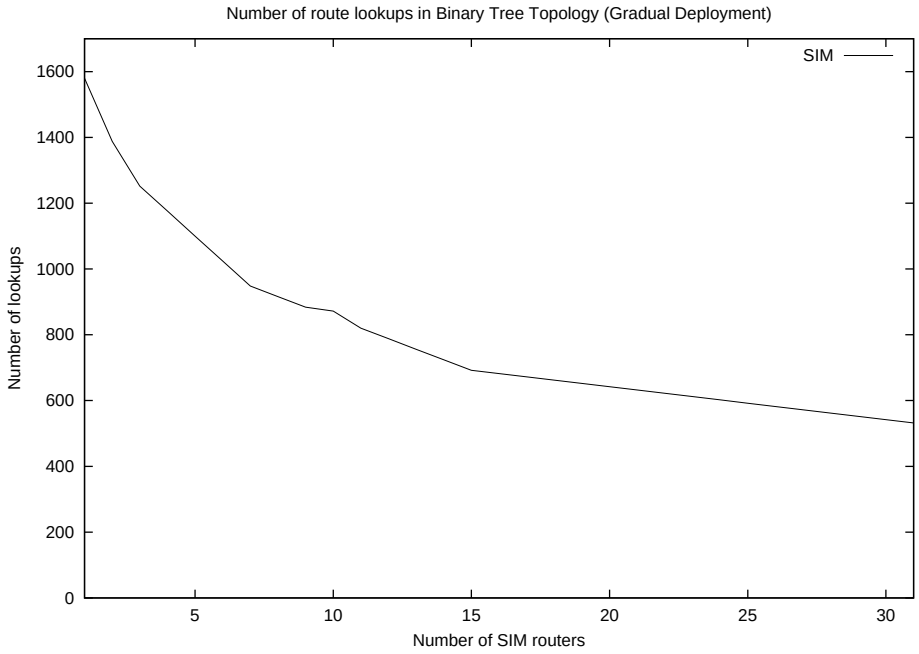



Figure 3.33: Total routing table lookups in Tree topology, while number of SIM routers increases

it was originally SIM packet or was ordinary unicast packet. However, the last SIM branching router should know the answer. Thus, the detection of this type of DoS attack should be deployed on SIM routers. Here we propose two approaches in order to detect a DoS attack:

1. Filtering receiver address
   As with IP multicast, some attackers may use SIM to generate some kinds of DoS attacks by specifying multicast address or broadcast address as a receiver. A last branching SIM router might provide a filtering of destination address and deny forwarding any packets that specify a multicast or broadcast address as the destination.

2. Filtering protocols
   When the last branching SIM router duplicates a packet for each receiver, it should check the Protocol field of IP header for IP version 4. This Protocol field indicates the next level protocol type. And as assigned in RFC1700 [62], ICMP has the protocol number 2. Thus, SIM router should not forward the packet, and it should discard if it detects that protocol field is 2. In this strategy, all SIM branching routers have to check all of the SIM packets. Thus, it may generate more overheads on SIM router. Moreover, since this strategy can only alleviate the effect of smurfing DoS attack, the strategy to deal with other kinds of DoS

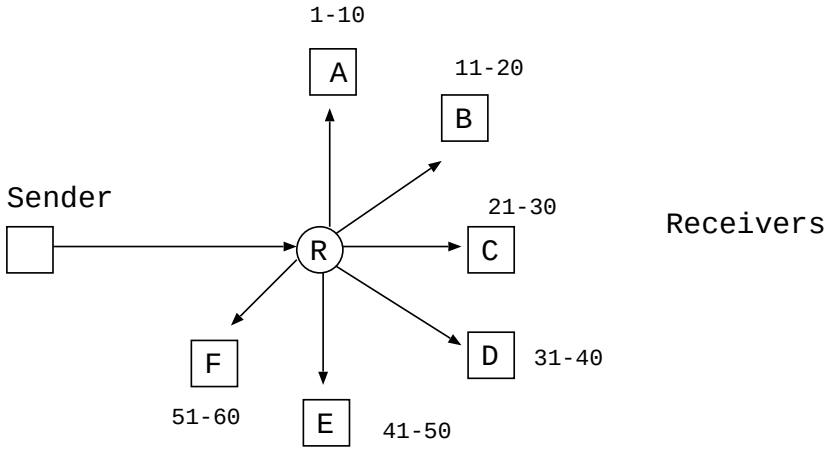



Figure 3.34: Star topology

attack needs more consideration.

In addition, in order to build more secure protocol, we can use the similar technique as proposed in SSM. The Internet Service Providers (ISPs) may limit the hosts that can send SIM packets by filtering source address on SIM routers.

### 3.11.3 Using IPsec with SIM

In this section, we focus on the possibility of providing secure multicast traffic by using IPsec with SIM. To use IPsec [63], we have to consider two issues: key distribution and packet encoding.

First, we discuss key distribution issue. Normally in case of unicast traffic, the destination side of Security Association (SA) is in charge of choosing a Security Parameter Index (SPI) for each connection. However, in traditional multicast model, there are multiple destination nodes and there probably are multiple sources per multicast group. As discussing in "Security Architecture for the Internet Protocol" [63] and in "Secure IP multicast: Problem areas, Framework, and Building Blocks" [64], some mechanisms are required to select an unique SPI for each multicast group. They may be a Group Controller (GC) and Key Controller (KC) to create key parameters, such as an SPI for a multicast group. In the case of SIM, since the size of group is small, the receivers can set their own SA, choose a SPI and send it to the sender. However, they can have a special node that will act as the GC and KC, and distribute the key to each receiver in the group.

For packet encoding, IPsec defines two protocols to provide network-layer security, which is Authentication Header (AH) [65] and Encapsulating Security Payload (ESP) [66]. Each protocol supports two modes of use: transport mode and tunnel mode. In transport mode, the protocols provide protection primarily for upper layer protocols.

Table 3.8: Average forwarding time on a branching router in Star topology ($\mu$s)

| Number of receivers | Forwarding Time ($\mu$s) |
|---|---|
| 6 | 52,955.833 |
| 12 | 104,484.242 |
| 18 | 140,095.556 |
| 24 | 154,741.778 |
| 30 | 175,502.626 |
| 36 | 203,043.555 |
| 42 | 243,524.222 |
| 48 | 261,398.889 |
| 54 | 310,562.444 |
| 58 | 339,250.889 |

While in tunnel mode, the protocols are applied to all tunneled IP packets. If we apply SIM packets in transport mode with AH or ESP or both, SIM header will be covered and encapsulated. That is, bitmap field in SIM header cannot be modified at each branching point. Therefore, IPsec in transport mode cannot be used with SIM. Contrarily, if we apply tunnel mode with SIM, the bitmap still can be modified. However, we have to deploy AH or ESP between SIM neighbors, encrypting and decrypting the packets on every SIM routers. This procedure will cause more overhead and does not work well. Moreover, even if we can use IPsec, we still cannot conceal the receiver list and the bitmap field in the SIM header. Some attacks can still be done by spoofing the receiver list and modify flags in the bitmap field for malicious use.

## 3.12 Summary

Sender Initiated Multicast (SIM) is designed to support small group communications with a cost-efficient packet forwarding mechanism. SIM will not replace the existing multicast protocol, but will be a subset of entire multicast protocol suite. Its design goals are to gain more simplicity than the traditional IP multicast and to gain less header-processing overhead than the Xcast protocol. The results obtained through simulations and experiments reveal that SIM can accelerate the forwarding process with less packet processing overhead than Xcast and can gain more scalability than IP multicast in the number of groups by using the SIM Tunnel.

The advantages and disadvantages of our protocol are summarized as follows:

1. The advantages of SIM

   - Scalable in number of multicast group
     Even though SIM is designed for small group communications, it scales well with the number of small multicast group. By using a two tuple of sender

address and multicast group address, each data sender can has as many as multicast group they want. Moreover, it eliminates cost in deploying global multicast address allocation protocol.

- Simple and cost-efficient packet forwarding mechanism

  A SIM router forwards the packets to other SIM routers and also to receivers by unicast routing. Therefore, there is no need to deploy any additional multicast inter- and intra- domain routing protocol. In addition, because a packet is duplicated only at branching routers, SIM does not excessively consume the network resources.

- Easy to deploy on the global Internet

  In order to use SIM, at least the sender and one router on the path have to implement this SIM protocol. In the SIM mechanism, at the last branching router, the SIM header will be detached from the packets. Some fields of the headers, such as the destination address field, will be replaced with receiver's unicast address. Therefore, receivers can receive SIM packets through the ordinary unicast packets processing mechanism. In other words, the receivers do not need to modify their protocol stacks. The existing applications residing on the receivers can also be used without any modifications.

2. The disadvantages of SIM

   - Constrained in the number of receivers

     As the receiver addresses are put into the packet header, the maximum number of receivers SIM can support is restricted to the minimum size of path MTU (PMTU) among the PMTUes from the sender to each receiver. Theoretically, in case PMTU is 1500 bytes, the maximum number of receivers SIM can support is 288. However, at the present implementation, the maximum number of receivers supported by SIM is 58.

   - Consume memory resource on routers

     In preset mode, SIM routers maintain SIM FIB entries as forwarding information for each *(S,G)* session. The size of an SIM FIB entry depends on the number of routes the router has to branch.

# Chapter 4

# Multicast extension to Transmission Control Protocol

## 4.1 Introduction

IP multicast is a host-group model, where the receivers are unknown to the sender. As a result, the present multicast applications cannot transfer the multicast packets by using the Transmission Control Protocol (TCP) [67], which is a connection-oriented protocol and offers a reliable data transfer. Normally, the multicast transmission currently is based on the connectionless and unreliable protocol such as the User Datagram Protocol (UDP). Many efforts have been made to provide a reliable multicast data transferring. However, developing a reliable transport protocol based on UDP is as complicated as developing TCP.

Earlier multicast transport mechanisms were designed as a general solution to the group communication problem [68]. However, there is *no one-size-fits-all* solution. Multicast applications have much wider range of requirements than the unicast applications. Some applications may require only all the data be received by all receivers, regardless of the order. Some applications may require that all data be received by all receivers, regardless of how long the transmission takes.

In the previous chapter, we have discussed SIM and Xcast. Contrary to the host-group model, both SIM and Xcast usually do not require a unique global IP multicast address. In addition, there is only one sender in a multicast session. Many-to-many communication can be achieved by using multiple multicast sessions. The concepts of packet forwarding in SIM and Xcast model are: first, the unicast routing information is used to transmit the packets; second, a session identification is defined by the sender; third, the receivers of a multicast session are explicitly specified by the sender of the group. In respect to the last characteristic, i.e. the receiver addresses are known by the sender, the connection-based protocol such as TCP can be used.

In this dissertation, to provide reliability to multicast, we chose using TCP instead of a connectionless and unreliable protocol such as UDP. We focused on the basic concept of the SIM and Xcast, which the receiver addresses are explicitly known by

the sender. By using this explicit multicast, connection-oriented transport can be provided. We use the explicit multicast as the network layer protocol, and enhance TCP protocol stacks in order to achieve a reliable multicast transport protocol. We propose the *Multicast extension to Transmission Control Protocol (M/TCP)* [69] [70], which is a TCP extension that enables multicast transmission to the existing TCP applications. The M/TCP is designed and implemented based on the Sender Initiated Multicast (SIM). However, it can also apply to other Xcast protocols. The features of M/TCP are: first, M/TCP can be applied to those applications that a sender triggers data transfer (sender-initiated), for example, server mirroring, File Transfer Protocol (FTP) [71], Simple Mail Transfer Protocol (SMTP) [72], and Network News Transfer Protocol (NNTP) [73]; second, it requires only implementation of the sender. Receivers will receive the packets as the unicast packets. Therefore, there is no need to modify both receiver-side protocol stacks and their applications. In this dissertation, we provide the details of M/TCP, and show how M/TCP can be widely deployed with the existing global Internet.

## 4.2 Reliable Multicast Transport Protocols

### 4.2.1 Building Blocks for Reliable Multicast Transport

Many efforts have been made to provide reliability with the connectionless multicast transmission with UDP. Most of these efforts are worked by the IETF Reliable Multicast Transport (RMT) working group. For unicast transport, the requirements for reliable data delivery are quite general. In contrast, different multicast applications have widely different requirements for reliability [74] [75] [76]. Some applications require packet-ordering guarantee, while others do not. Some applications contain with one data sender, while others contain with multiple senders. Some applications such as a file transfer require a high bandwidth with a reasonable delay, while others such as a stock quote require only a low bandwidth with real-time delivery.

Since different applications have different requirements of a reliable multicast protocol, these requirements constrain the design space in ways that two applications with differing requirements often cannot share a single solution [75]. The RMT working group has proposed a *Building Blocks for One-to-Many Bulk-Data Transfer* [76] as a framework for the standardization of bulk- data reliable multicast transport. The protocol families proposed in the RMT building blocks are NACK, Tree-based ACK (TRACK), Asynchronous Layered Coding (ALC), and Router assist. Table 4.1 shows the protocol families and examples of protocol in each family.

Instead of sending an ACK for every data packet received, the NACK protocols require the receivers to return negative acknowledgements (NACK) whenever they discover that they did not receive the packets. The advantage of protocols in this family is that, the sender does not need to know exactly how many receivers are in the multicast session. They can retransmit the loss packets as soon as a single NACK is received from any receiver. Thus, NACK suppression is possible. The disadvantages are that it is more difficult for the sender to know that it can free transmission buffers,

Table 4.1: Protocol Families proposed in the RMT building block

| Protocol Family | Example |
|---|---|
| Negative ACK (NACK) | SRM, MDP |
| Tree-based ACK (TRACK) | RMTP, RMTP-II, TRAM |
| Asynchronous Layered Coding (ALC) | FEC |
| Router assist | PGM |

and that additional session level mechanisms are needed if the sender really needs to know if a particular receiver actually received all the data [75].

Scalable Reliable Multicast (SRM) [77] is one of a NACK-based multicast protocols that guarantee out-of-order reliable delivery. SRM is a reliable multicast framework designed for light-weighted sessions and application level framing. The basic idea of SRM is that the lost packet is not retransmitted by the sender; instead, by the nearest receiver who has received the packet. SRM suppresses the NACKs returned from the receivers by using random timers weighted by the round trip time (RTT) between the sender and each receiver. However, this method requires computing the RTT to each receiver before suppression works properly [75].

MDP [78] is another NACK protocol that implements reliable multicasting for bulk files and other objects using application layer framing concepts. To achieve scalability, MDP uses random back-off NACK with NACK suppression mechanisms.

The Tree-based ACK (TRACK) protocols aggregate the ACKs by arranging the receivers into a tree whereby receivers generate ACKs to a parent node. The parent nodes aggregate those ACKs to its parent of the tree in turn. The examples of TRACK protocol are RMTP [79], RMTP-II [80], and TRAM [81]. These protocols use positive acknowledgements (ACKs) to provide delivery confirmation, while the receivers are grouped and organized in a tree hierarchy. ACKs from the receivers will be collected at the local representatives, which would aggregate and forward the ACKs to an upper-level representative in the tree.

Asynchronous Layered Coding (ALC) [82] protocols use sender-based Forward Error Correction (FEC) [83] [84] with no feedback from receivers or the network. The data is organized in a layered fashion, where the number of streams delivering to each receiver depends on the local bandwidth availability and network conditions. The advantage of this kind of protocol is that it allows different receivers to be able to receive the traffic at different rates according to the available capacity.

Note that the distinction in protocol families in not necessarily precise and mutually exclusive [76]. Many reliable multicast protocols are the hybrid solution of these protocol families. For instance, MDPv2 [85], which is the successor of MDP, is a hybrid approach of NACK and Asynchronous Layered Coding (ALC). MDPv2 extends MDP framework by including optional parity-based repair using forward error correction (FEC) coding techniques. RMTP-II [80] is also a hybrid NACK/ACK based protocol. Protocols such as PGM [86] and [87] take advantage of new router software to perform

constrained negative acknowledgments and retransmissions.

### 4.2.2 Multicast Congestion Control

One of the technical issues need to be considered in multicast transmission is how to fairly share scarce bandwidth among receivers when the congestion occurs. Most of the existing reliable multicast transport protocols are based on the connectionless protocol such as UDP. However, because of the lack of end-to-end congestion control mechanism in UDP, the well-known problem of UDP is that, when UDP traffic is introduced into the congested link, it forces the TCP connections to back off. Hence, running UDP is not fair to all users. This problem is more critical in the multicast transmission than in the unicast, because the traffic generally spreads through a wide area of the Internet.

To alleviate this problem, many multicast transport protocols have been proposed with TCP-friendly congestion control mechanism. Examples of TCP-friendly multicast protocol are TFMCC, *pgmcc*, and *TCP-like*. TCP-Friendly Multicast Congestion Control (TFMCC) [88] is a single-rate congestion control scheme, where the sending rate is adapted to the receiver experiencing the worst network conditions. *pgmcc* [89], as with TFMCC, provides only a single-rate transmission. The pre-assumption of *pgmcc* is based on UDP and the host-group model multicast. It also uses window-based congestion control like in TCP. ACKs from the receivers will be collected at the group representative as in TRACK protocols. The last example is *TCP-like* [90]. *TCP-like* uses a receiver-driven congestion control scheme, instead of sender-driven congestion control scheme as in TCP. It supports multiple bandwidths by using a layered organization of data with multiple multicast groups. The number of multicast groups a receiver joins is depends on the available bandwidth from the sender to the receiver.

Even though many reliable multicast transport protocols are under standardization in IETF, and some of them introduce the TCP-friendly congestion control mechanism, most approaches are too complicated and require modifications on both the sender and the receiver hosts. In addition, it is widely known that providing reliability to UDP is as difficult as developing TCP. In other words, as with TCP, the state information such as a sequence number, retransmission counts, and round-trip time estimator have to be maintained. In principle, we need just about all the information currently in the TCP control block.

Developing TCP for multicast, other than our M/TCP approach, are proposed in [91] and *TCP-SMO* [92]. Our work differs from both works in two important ways. First, contrary to both works, our goal is not only to provide multicast capability to TCP, but also to support the existing TCP applications to use multicast without any modification. These two protocols have to be deployed to both the sender and receivers, while we require only modification on the sender. Second, the basic idea to extend TCP for multicast in both research are based on the IP multicast model, while we focus only on small group multicast. Third, these protocols cannot gain a good throughput, because they are a single-rate transmission and totally depends on the slowest receiver. When a receiver fails to return the ACK, such as in case of the hardware failure or the network failure, both the transmission will be hold

on until a time has elapsed, or until the send buffer is full. Moreover, [91] did not consider the congestion control mechanism, and the protocol was proposed but never be implemented.

## 4.3 Overview of M/TCP

### 4.3.1 Design Goals

M/TCP is designed based on the concepts of the explicit multicast. The design goals of M/TCP are: first, to provide a reliability in multicast communication by enhancing multicast functions to TCP, and second, to support the existing TCP applications on the receiver side. M/TCP takes over the reliability from TCP by doing the following:

- Pass the received data in correct order to the application.
- Discard a duplicated data
- Maintain an acknowledge timer. If an acknowledgement is not received in time, the segment/packet is retransmitted.
- Maintain a checksum on its header and data for purpose of data integrity. If the checksum is invalid, the arriving segment will be discarded.
- Provide flow control, which prevents a fast host from taking all the buffers on a slower host.

### 4.3.2 Target Applications

As discussed in RFC 2357 [74], no single reliable multicast protocol will likely meet the needs of all applications. In the same manner, M/TCP is not a general solution for all reliable multicast applications. In contrast, we hope M/TCP will be a subset of reliable multicast protocols for a special requirement of applications. M/TCP can be considered as a protocol in the router assist family, which is described in the RMT building blocks. M/TCP is aiming to support the applications that have the following characteristics:

1. Require a high reliability

   As an extension of TCP, M/TCP provides reliability by receiving packets in correct order, discarding duplicated packets, retransmit loss packets, and providing checksum to detect whether data was modified before it arrives at each receiver. In addition to the reliability provided in unicast TCP, M/TCP guarantees that all receivers receive all the packets.

2. Asymmetrical Data Transmission

   Basically, M/TCP is designed for one-to-many multicast communications, although many-to-many multicast communications can also be handled by using

multiple multicast groups. M/TCP is suitable for the asymmetrical application, i.e. the traffic from the sender is much more than the traffic from the receivers. The transmission of M/TCP can be considered as sender-initiated. Commonly, only the sender will transfer the data. Applications that have this characteristic are, for example, FTP, SMTP, and NNTP.

3. Non real-time and non-multimedia service

   Usually, multimedia data streams such as sounds and movies can be divided into multiple layers and transferred separately in multiple multicast groups. This method to transmit layered data in multiple multicast groups is proposed in [93] and [90]. The receivers can obtain data in the appropriate rate by joining multiple multicast groups. However, some data, such as text file and binary data, will be useless if the receivers receive only some parts of them. Data like this cannot be organized into multiple layers. In our approach, we do not support such layered multicast data transfer. M/TCP would be appropriate for bulk data transfer.

   Additionally, TCP uses window-based congestion control mechanism to control sender transmission rate. It will not send more packets, unless it has received ACKs of the old packets from the receivers. Contrary to UDP, which is datagram-oriented transport protocol, TCP is stream-oriented. There are no data boundary or record markers automatically inserted by TCP. When a segment is lost, it cannot know whether the left segments can be used and has to discard all the segments. Accordingly, the application that uses M/TCP should not be the real-time or multimedia data transfer services.

### 4.3.3 Advantages and Disadvantage of M/TCP

M/TCP has the following advantages.

1. Not require any modifications on the receiver side

   M/TCP provides point-to-point multicast communication. As a novel function of SIM, our fundamental explicit multicast routing protocol, all M/TCP packets will be rewritten to unicast packets before arriving at the receivers. Therefore, from the sender's point of view, the packet is transmitted in multicast. From the receiver's point of view, the packet has reached them in unicast. This advantage makes the receiver and its existing multicast applications receive the packets without any modifications. Only the sender-side kernel and its applications have to be modified in order to use M/TCP and SIM.

2. Alleviate ACK suppression on the sender

   As there is no modification on the receivers, the acknowledgement (ACK) of the packets will be sent by unicast as usual. Thus, the ACK suppression at the sender is normally a problem in the existing reliable multicast. However, Delay ACK mechanism is normally used in TCP. Therefore, ACK from the receivers will reach the sender in random. In addition, our model is based on Xcast that

supports only small multicast group. Thus, the ACK suppression will not occur to our sender.

3. Provide multiple transmission rates by re-classifing receivers group

   As mentioned in [93], the source-based rate-adaptation performs poorly in a heterogeneous multicast environment because there is no single target rate — the conflicting bandwidth requirements of all receivers cannot be simultaneously satisfied with one transmission rate. Therefore, we make an effort to provide multiple transmission rates that match to each receiver's requirements. During the transmission, we re-classify receivers in multiple groups depending on the packet loss they experienced. Receivers who can receive data packets in the same rate will be grouped in the same multicast group. A receiver will be eliminated from a group, when it becomes the bottleneck of that group, i.e. has extremely poor receiving rate. The bottleneck receiver will receive data packets by unicast instead. By performing this procedure, we believe that it can mollify the influence from a bottleneck receiver. The group's transmission rate will not be determined by one bottleneck receiver.

Besides its advantages, M/TCP has the following two disadvantages.

1. Depends on RTT and RTO

   As mentioned in [90], in congested network, the relative throughputs for competing sessions experiencing the same loss rate should not depend on the "distance" (RTT) between the sender and the receiver. However, in our approach, we will classify receivers by its packet loss experience. Normally, the congestion control of TCP is ACK-based. A receiver with a larger RTT is less responsive than one with a smaller RTT. Using ACK-based to control the transmission, we would not know whether the packet loss occurred by inappropriate group's retransmission time out (RTO) timer, or by the congestion of the network. If the RTO is not appropriate, it will always be timeout before we receive ACKs from some receivers, even though they have received the packets.

2. ACK suppression when there are many groups on the same sender

   As previously mentioned, M/TCP can avoid the ACK suppression on the sender by the delay ACK mechanism of TCP. The ACK suppression is trivial, when the sender holds a small multicast group. However, if there are multiple multicast sessions or multiple multicast groups on the same sender, ACK suppression can occur. In M/TCP, only the sender is concerned with the multicast transmission. There are no strategies to aggregate ACKs before reporting them to the sender, as proposed in [86].

## 4.4 Transmission Procedure in M/TCP

The transmission is based on the capability of each receiver. When a subset of receivers in the group lose a packet, the lost packet will be retransmitted in another TCP flow.

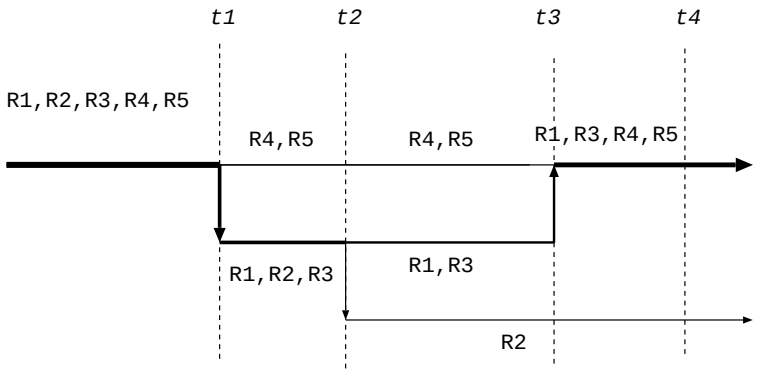



Figure 4.1: Transmisison in M/TCP

The transmission rate will not be dominated by the slowest receiver. The following procedure will be performed when a packet is lost.

- When two or more receivers lose the same packet, the packet will be retransmitted in a M/TCP packet.
- When only one receiver loses the packet, the retransmission will be performed by unicast.
- When the retransmission is finished, new packets will be transmitted separately to the group of receivers that has experienced packet loss.
- The retransmission group will be re-grouped again, in the case the receiver who has experienced packet loss reached the transmission rate with the original transmission flow.

Figure 4.1 depicts the example of our transmission. First, the receiver group contained R1, R2, R3, R4, and R5. At the beginning of the flow, the packets are transmitted to all five receivers. At time t1, R1, R2, and R3 lost the packet. At this point, we temporarily divide receivers into two groups. The transmission on the original flow will not be interrupted. We still transmit the packets to R4 and R5. The lost packet will be separately retransmitted to R1, R2, and R3. After the retransmission procedure has finished, the new packets will be transmitted to R1, R2, and R3. At time t2, R2 lost the packet again. Therefore, after time t2, we will retransmit the packet to R2 by unicast. Unfortunately, R2 cannot receive data as fast as the original group. Therefore, it will never be re-grouped again.

## 4.5 Congestion Control in M/TCP

Congestion control in multicast is more difficult than in unicast, since receivers in a group with different requirements may exist. Adapting to the needs of one set of

receivers would penalize the others with different requirements. The goal of flow control in multicast is to make the sender transmit no faster than the rate available at the slowest receiver. Congestion control in M/TCP is as follows:

**Receiver-based congestion control:** The acknowledgement will be cumulatively feed-backed to the sender, and used in adjusting the sender transmission rate. Receivers will be re-grouped according to packet loss they have experienced. When there is a bottleneck receiver in the multicast group, that receiver will be deleted from the group, and receive the data via unicast instead. Receivers who have the same receiving rate, i.e. receivers who have experienced the same lost rate will be classified in the same group. Finally, receivers with the higher receiving rate can finish the transmission faster.

**Window-sized based control:** Window size is based on acknowledgement of sent data that has to wait for RTT. In multicast, the transmission rate is finally controlled by the slowest receiver. In the worst case, TCP transmission rate is *Wmin/RTTmax*, where *Wmin* is the minimum window size, and *RTTmax* is the maximum round-trip time among the receivers. Although all receivers share some paths of the multicast link, the packets finally reach each receivers through the different paths. It is too difficult to detect whether the congestion that has occurred on a specific path affects all the receivers. With this reason, congestion control is a difficult issue of multicast. That means, we have to keep the congestion window and packet loss, separately for each receiver.

**Loss-driven additive increase/multiplicative decrease (AIMD) policy:** M/TCP uses AIMD policy to control the congestion as same as the existing TCP. At the beginning of the transmission, slow start mechanism is applied. The congestion window is increased by one packet upon the receipt of ACKs from all the receivers. When the multicast congestion window reaches the slow start threshold (*ssthresh*), the congestion avoidance mechanism will be applied. That is, all congestion windows have to be reset to half of its *ssthresh*. The smallest congestion window will be used as the multicast congestion window.

### 4.5.1 Retransmission

M/TCP will transmit the loss packet to the children multicast group. The "Children multicast group" is a new group divided from the original multicast groups, when a packet loss has occurred. It contains receivers who missed the packet. We will never retransmit the packet to the original multicast group, unless all receivers lost the packet.

**Retransmission Time Out:** Retransmission Time Out (RTO) has to be set in order to allow the slowest receiver returns ACKs. In TCP, RTO is basically calculated from the Round Trip Time (RTT). In multicast, we can guarantee that the packet will reach the receivers through the same routing path. They will have the different RTTs. Instead of calculating the RTO for a multicast group by using the biggest RTTs, RTOs should be calculated separately for each receiver. The largest RTO will be used as the group's timer.

$$RTO = max(RTOi, RTOi + 1, ....., RTOn) \tag{4.1}$$

**Fast retransmission and Fast Recovery:** In TCP, fast retransmission is accomplished by using duplicate ACK mechanism. TCP will retransmit the packet when they get an acknowledgement for the same packets three times. Therefore, we have to wait until ACKs and duplicate ACKs returned from all receivers. We cannot retransmit the packets as soon as we get duplicate ACK three times from the same receivers. In practical, when we receive the duplicate ACK more than three times from the same receiver, we will interpret it as a negative ACK (NACK). Retransmission of a lost packet will be performed when the group's RTO has expired, or when the number of ACK and NACK is equal to the number of the receivers.

```
If  ( ACK + NACK  == Number of Receivers ) {
          Process RETRANSMISSION DATA;
}
```

## 4.6 Implementation

This section describes the implementation issues of M/TCP. In the present, M/TCP is under development on FreeBSD 4.6-RELEASE.

### 4.6.1 M/TCP Header Format

As shown in Figure 4.2, some fields in the TCP header are different for each receiver. These fields are destination port, acknowledgement sequence number, header length and checksum. In addition, to reduce costs of calculating checksum on SIM routers, we compute TCP checksum for each receiver at the sender, and attach them to SIM options (Figure 4.3). Before the packets reach the receivers, the last branching SIM router is responsible for re-constructing their packet headers as if they are unicast packets sent to each receiver.

### 4.6.2 Multicast Group Membership

Group membership in M/TCP is managed by SIM, the IP-level protocol. The multicast group membership is totally controlled by the sender; and a multicast group is identified by the combination of sender address and the multicast group address. The group multicast address is handed over to M/TCP by using connect() system call. To add or delete receivers from a group, setsockopt() function will be used. Note that, besides IP_SIM_ADD socket option described in Chapter 3, we defined TCP_ADD in `<netinet/tcp.h>` header file for M/TCP purpose. TCP_ADD socket option is used to the kernel that data transmission will be done by M/TCP.

```
setsockopt( s, IPPROTO_TCP, TCP_ADD, &addr_list, IN_SIZE(n));
```

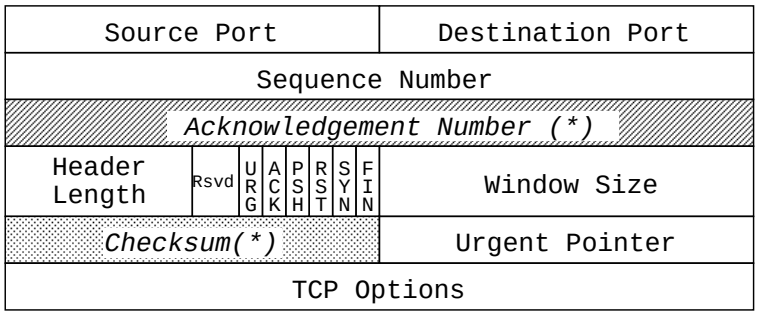



Figure 4.2: M/TCP Header Format

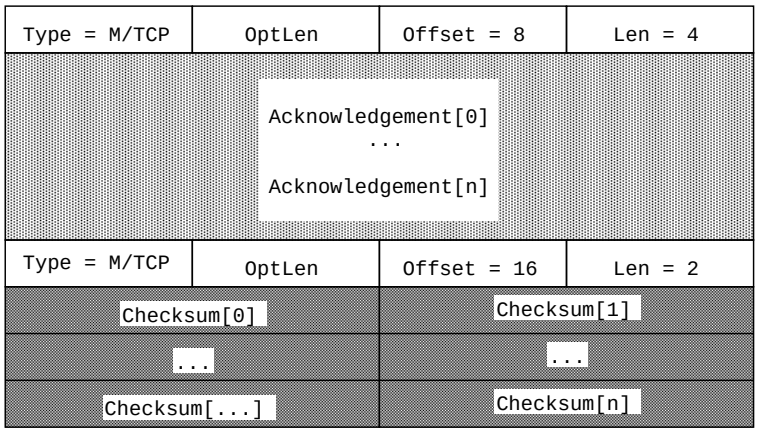



Figure 4.3: SIM Options for M/TCP

### 4.6.3 Co-operation with SIM

M/TCP uses the following mt_rcv_info structure to maintain receiver list, the acknowledgement numbers and TCP header checksums for each receiver.

```
struct mt_rcv_info {
        struct in_addr m_destination_address;
        u_int32_t m_ack;
        u_short m_checksum;
        TAILQ_ENTRY(mt_rcv_info) m_rinfo;
};
```

This information will be handed over to SIM. Receiver list for a group will be attached to the SIM header. The acknowledgement numbers and checksums will be attached to SIM option (Figure 4.5). On the last multicast branching SIM router, the

```
int
tcp_ctloutput(so, sopt)
        struct socket *so;
        struct sockopt *sopt;
{
        switch (sopt->sopt_dir) {
                .......
        case TCP_ADD:
                /* Should use with IP_SIM_ADD(...) */
                so->so_options |= SO_MTCP;
                inp->inp_sim_flags |= SIM_OPTIONS_FLAG;
                tp->mt_flags |= (MTF_MTCP | MTF_MAINCB);
                error = mt_tcp_attach(so, sopt->sopt_p);
                break;
                .......
        }
        return (error);
}
```

Figure 4.4: M/TCP TCP_ADD

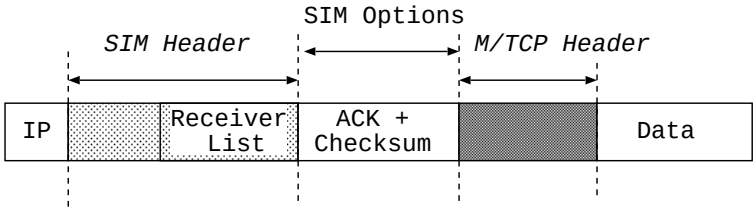



Figure 4.5: Relationship of SIM and M/TCP Packet Header

packet header will be recreated for unicast packets, before transmitting the packets to each receiver. SIM header will be detached, and the acknowledgment numbers and checksums in SIM option will be used to replace the fields in M/TCP header.

### 4.6.4 Extension to Protocol Control Blocks

To maintain each receiver's IP address, we have to create the Internet protocol control block (*inpcb*) for each receiver. Moreover, state information such as duplicate ACK and retransmission timer has to be maintained individually. Therefore, we have to create the transmission control protocol control block (*tcpcb*) for each receiver (Figure 4.6).

At present, we only provide one send socket buffer for a multicast group. Data will be deleted from the send socket buffer whenever we get ACKs from all receivers of a group. If the RTO timer is timeout before receiving ACKs from all receivers, receivers

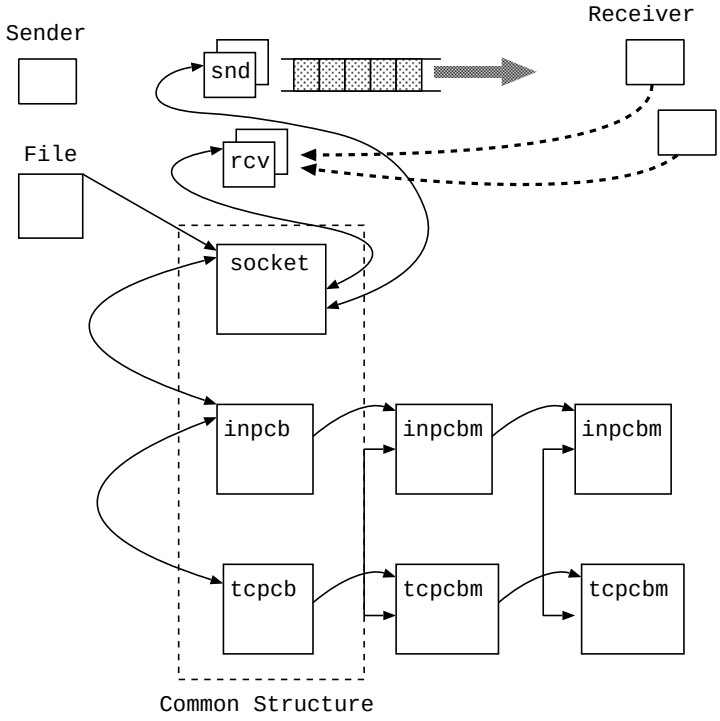



Figure 4.6: Extension to Protocol Control Blocks

who assumed to loss the packet will be classified to another multicast group. Data will be copied from the original send socket buffer to that of the new group. This data will be used in retransmission the lost packet. Then, we can delete the segment that is acknowledged from the original send socket buffer.

### 4.6.5 Transmission by M/TCP

Transmission by M/TCP use the ordinary TCP application interface: socket(), connect(), write(), read(), and close(). Figure 4.7 shows the example of M/TCP sender program. There is no modification on receiver hosts, thus the receivers can receive the packets by the ordinary TCP applications.

One problem we have to encounter is that, when a multicast group consists of receivers who have extremely conflicting requirements, they are probably re-classified in multiple multicast groups. In the worst case, all receivers are probably divided into each unicast flow. This cannot avoid in the present design of M/TCP. However, even in this case, the performance will not be worse than transferring data by unicast. M/TCP will be effective, mostly when receivers who have similar requirements are in the same multicast group. In addition, it is guarantee that a specific bottleneck receiver will not degrade the total performance.

```
#define MTCPPORT 9876
#define BUFFSIZE   1024
#define RCVNUM    2

int
main()
{
        struct addr_list allreceiver;
        struct sockaddr_in sin;
        static char *al_buf;                                                    10
        char buf[BUFFSIZE];
        int sender_socket;

        if((sender_socket = socket(PF_INET, SOCK_STREAM, 0)) < 0) {
                perror("socket error");
                exit(1);
        }

        allreceiver = (addr_list_t *)malloc((unsigned)IN_SIZE(RCVNUM));
        bzero((char*)allreceiver, IN_SIZE(RCVNUM));                             20
        allreceiver.al_len = (IN_SIZE(RCVNUM));
        allreceiver.al_flag = MTCP_SIM_ADDRLIST;
        allreceiver.al_port = MTCPPORT;
        allreceiver.al_addr[0].s_addr = inet_addr("163.221.52.142");
        allreceiver.al_addr[1].s_addr = inet_addr("163.221.52.168");

        /* Add receiver list to SIM header. */
        if (setsockopt(sender_socket, IPPROTO_IP, IP_SIM_ADD, &allreceiver,
                        sizeof(allreceiver)) < 0 ) {
                printf("errno 1 = %d\n",errno);                                 30
                perror("setsockopt error: IP_SIM_ADD");
                exit(EXIT_FAILURE);
        }

        if(setsockopt(sender_socket, IPPROTO_TCP, TCP_ADD, allreceiver, allreceiver.al_len) < 0) {
                perror("setsockopt error:TCP_ADD");
                exit(1);
        }

        bzero((char*)&sin, sizeof(sin));                                        40
        sin.sin_len = sizeof(sin);
        sin.sin_family = AF_INET;
        sin.sin_port = htons(MTCPPORT);
        sin.sin_addr.s_addr = inet_addr("224.0.0.10");

        if (connect(sender_socket, (struct sockaddr*)&sin, sizeof(sin)) < 0) {
                perror("connect error");
                exit(1);
        }
        close(sender_socket);                                                   50
        exit(0);
}
```

Figure 4.7: An example of M/TCP sender program

### 4.6.6 Kernel Support for M/TCP senders

We provide multicast-capability to TCP by appending the following kernel functions:

- `mt_tcp_attach()`

  This function is called by `tcp_ctloutput()`, when the TCP_ADD option is specified. Then, it calls `mt_in_pcballoc()` to allocate sub *inpcb* (Internet Protocol Control blocks), `tcp_newtcpcb()` to allocate *tcpcb* (TCP control block) for each receiver, and insert sub *tcpcb* to the tail queue of group *tcpcb*.

- `mt_in_pcballoc()`

  This function is called by `mt_tcp_attach()` to allocate sub *inpcb* for each receiver, and insert them to the tail queue of group *inpcb*.

- `min_snduna()`

  This function is called by `mt_sync_datain()`. It calculates the minimum send unacknowledged (*snd_una*) sequence number for the group.

- `min_sndwnd()`

  This function is called by `mt_sync_synin()` and `mt_sync_datain()`. It calculates minimum send window size for the group.

- `mt_sync_synout()`

  This funtion is called by `tcp_output()` to synchronize TCP state of the group *tcpcb* and sub *tcpcb* for each receiver.

- `mt_sync_synin()`

  This funtion is called by `tcp_input()` to synchronize TCP state of the group *tcpcb* and sub *tcpcb* for each receiver, when the *SYN+ACK* packets are returned from all receivers.

- `mt_sync_dataout()`

  This funtion is called by `tcp_output()` to synchronize TCP state of the group *tcpcb* and sub *tcpcb* for each receiver, before `ip_output()` is called. It is called when the TCP state is *TCPS_ESTABLISHED* and the TH_ACK flag is set on TCP header.

- `mt_sync_datain()`

  This funtion is called by `tcp_input()` to synchronize TCP state of the group *tcpcb* and sub *tcpcb* for each receiver, when the TCP state is *TCPS_ESTABLISHED* and the sender received ACKs from all receivers.

- `mt_sync_persist()`

  This funtion is called by `tcp_output()` to synchronize TCP state of the group *tcpcb* and sub *tcpcb* for each receiver, when FIN has been sent but not acked.

This funtion is called by `tcp_output()` to synchronize the size of sending window among the group *tcpcb* and sub *tcpcb*s.

- `mt_timer_stop()`

  This funtion is called by `tcp_close()` to stop all timers in sub *tcpcb*s.

- `mt_in_pcbdetach()`

  This funtion is also called by `tcp_close()` to free receivers' information and and sub *inpcb*s.

## 4.7 Performance Evaluation

As with SIM, M/TCP is implemented on FreeBSD. In this section, we performed two experimentations in order to evaluate performance of M/TCP. Note that, at present, there is no implementation to enabling multicast for TCP on IP multicast and Xcast. Therefore, we compare M/TCP performance only with the multiple unicast transmission.

In the first experiment, we try to answer the question: what is the relationship between the number of receivers and average forwarding time. The topology we used in the first experiment is illustrated in Figure 4.8. Our experimental network consists of one sender, one SIM router, and eight receivers. We gradually increase number of receivers, and measure the average forwarding time experienced by all receivers. Figure 4.9 shows the experimental result. Note that we do not include more SIM routers, because we would like to seperate the characteristics of the routing protocol, i.e. SIM, from the characteristics of M/TCP. Results with more SIM routers will not provide more information about performance of M/TCP, because M/TCP routine is performed only on the sender and the last hop router.

As depicted in Figure 4.9, both average forwarding time in SIM and unicast increased, as the number of receivers increase. However, the result shows that forwarding time in M/TCP does not increase much as it does in multiple unicast. Furthermore, at the point of four receivers, the average forwarding time in M/TCP is lower than multiple unicast. From this point, we can conclude that M/TCP is superior to unicast, when there are more than four receivers in the group.

In the section experiment, we tried to answer "how is the transmission by M/TCP when a receiver is in a delay network". In order to answer this question, we located an ATM delay device between the SIM router and a receiver. We gradually increased the delay and measured the changes in average forwarding time experienced by all receivers. The second topology we used is shown in Figure 4.10. Note that in this topology, we have only one receiver in delay network. This is enough to evaluate M/TCP performance, because only the worst receiver, i.e. receiver in the most delay network will affect the performance. Our *RCV6* in Figure 4.10 represents the worst-case receiver.

Table 4.2 shows the result from the second experiment. The result shows that the average forwarding time in both SIM and multiple unicast increase, as the delay

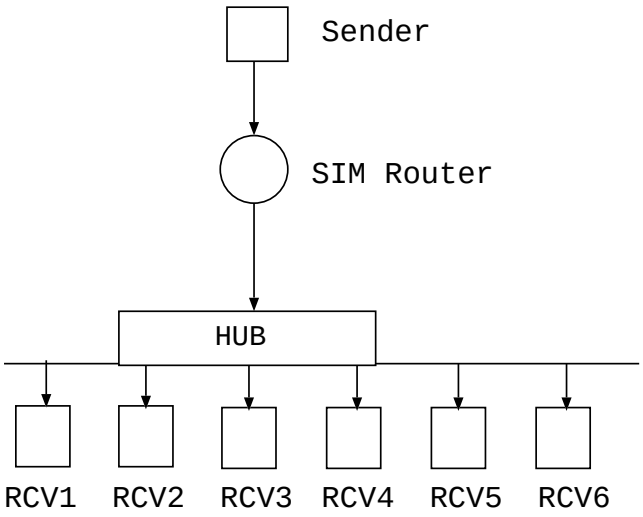



Figure 4.8: Topology of Experiment 1

increases. However, the average forwarding time in M/TCP transmission is superior to unicast in every case, and gives evidence that multicasting by M/TCP is effective.

Table 4.2: Relationship of delay and forwarding time

| *Delay (ms)* | *Forwarding Time (ms)* | |
|---|---|---|
| | SIM and M/TCP | Multiple Unicast |
| 15.8 | 33,370 | 34,631 |
| 31.6 | 65,027 | 65,956 |
| 47.41 | 96,616 | 97,803 |
| 100.08 | 20,1933 | 202,902 |
| 252.84 | 507,438 | 508,307 |

## 4.8 Discussions

In this section, we will discuss about the packet overhead, the extension to socket buffer, restriction of membership, and performance evaluation.

### 4.8.1 Packet Overhead

In the present design of M/TCP, we use multicast to transfer every packet without considering its size. However, in the practical, it is not appropriate if we have to transmit a small packet with a big header. The example for the data that is quite small and different for each receiver is ACK-only packet. It may be preferable to

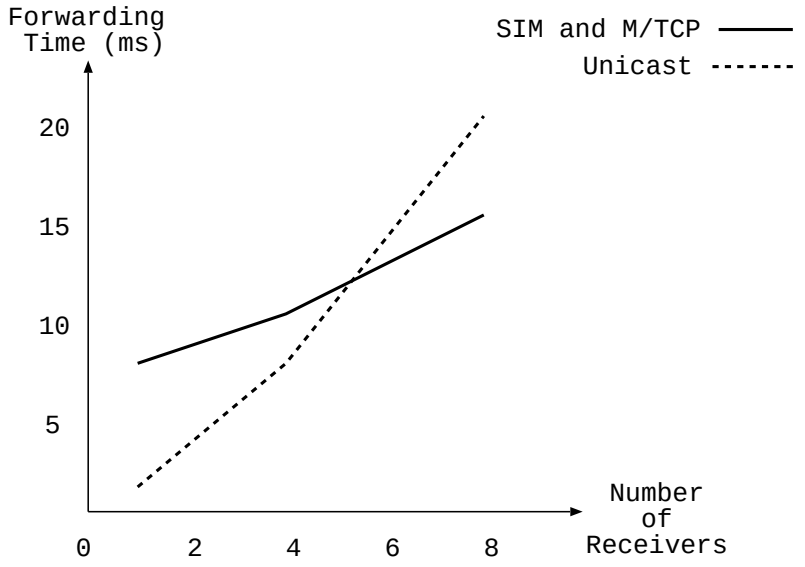



Figure 4.9: Number of receivers and average forwarding time

transmit such data by unicast. However, besides this packet overhead, our attempts to provide multicast to the existing TCP application is considered a significant strategy.

### 4.8.2 ACK implosion

In the multicast congestion control, the term *ACK implosion* [94] [95] is used to specify the condition that a large number of ACK packets, which are generated by the potentially large number of receivers, return to the sender host and cause a processing overhead at the sender host. Many techniques such as NACK and ACK aggregation are used to avoid this problem. NACK is used by the receivers who lose the packet, while the ACK aggregation is used by the routers to aggregate the ACK packets from the receivers. When the sender receives at least a NACK from a receiver, it can immediately retransmit the loss packet to all the receivers in order to gain a good throughput.

However, we do not use both NACK and ACK aggregation in M/TCP model. One reason that we do not use NACK is that, our target applications are the applications that do need to know that every receiver received the data. Using NACK cannot guarantee that the data is delivered to all receivers, because a NACK may be lost during the congestion time. Therefore, NACK is not appropriate in our model. The other reason is that, while NACK in the IP multicast model can accelerate the retransmission procedure by retransmitting the data to all receivers, we do not have a need to retransmit data to all receivers in the group. Instead, by specifying the *Temporary* flag in SIM header, the lost packet can be retransmitted only to the set of receivers who did not return ACK to the sender. Therefore, NACK is not useful in M/TCP model.

The reason why we do not need the ACK aggregation is that, the number of re-

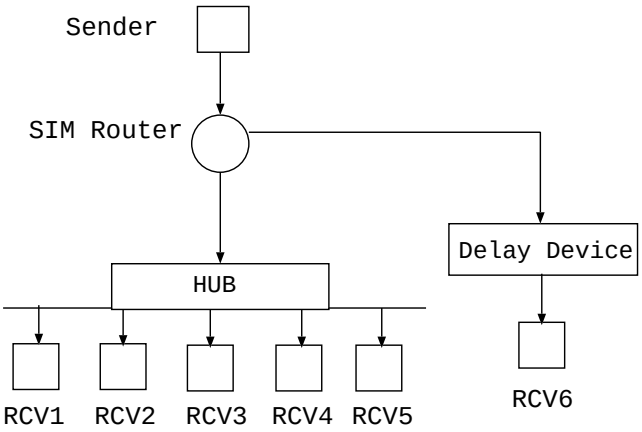



Figure 4.10: Topology of Experiment 2

ceivers supported in our model, which is based on SIM or Xcast, is relatively small with not more than 100 receivers. Besides, as previously described, delay ACK mechanism is normally used in TCP; therefore, ACKs from each receiver arrive at the sender in random manner. Thus, the probability that the ACK implosion occurs to our sender is low. Moreover, it is demonstrated by TCP-SMO [92] that, the ACK packets take only a small percentage of the available network bandwidth; and even without ACK aggregation, the ACK processing overhead at the sender host is not a serious problem for a multicast session with fewer than a thousand receivers.

## 4.9 Summary

In this chapter, we have proposed M/TCP as a new approach to provide reliability to multicast by using TCP. M/TCP is based on the concepts of the explicit multicast. At the present, it is under implementation. The design goals of M/TCP are (1) to provide reliability to multicast communication by enhancing multicast functions to TCP, and (2) to support the existing TCP applications without any modifications. We believe that the advantages of M/TCP, that do not require any modifications for the receivers and alleviates ACK suppression on the receivers, give M/TCP the possibility to be widely deployed on the Internet.

# Chapter 5

# Conclusions and Future Work

The main theme of this dissertation is how to provide scalability to a number of small multicast groups and how to achieve a reliable multicast transmission with less software modification. In order to provide scalability to a number of small multicast groups, we introduced *Sender-Initiated Multicast (SIM)* as an IP layer multicast protocol in chapter 3. In addition, we introduced *Multicast extension to TCP (M/TCP* in Chapter 4, in order to provide a reliable multicast transmission through the reliability of TCP. In this chapter, we conclude all the works for this dissertation, and then discuss future work.

## 5.1 Conclusions

In this dissertation, we described the SIM mechanism in detail and presented evaluating results through implementation and simulation. We showed how SIM can achieve low cost of maintaining forwarding information, cost-efficient packet forwarding, and incrementally deployment. The result from the simulation in the bandwidth consumption shows that in SIM List mode and in Xcast, the receiver list attached to the packets will consume about forty percent of the total sending traffic, when there are about 100 receivers in a multicast group.

Further, the results obtained through performance evaluation show that SIM can accelerate the forwarding process by maintaining SIM FIB entries, and can gain scalability in number of groups by using SIM Tunnel to reduce forwarding states on routers. To achieve incremental deployment of multicast widely in the global Internet, SIM has several advantages: explicit group management scheme, cost effective packet forwarding mechanism, and ease of deployment on the global Internet. For gradual deployment, SIM provides conversion-to-unicast function that requires at least the sender and one router on the distributing path to implement the protocol. Thus, applications at receiver side can benefit from SIM without any additional modification.

In order to provide reliability to multicast, we also proposed M/TCP as a new solution. M/TCP is based on the concepts of explicit multicast. The design goals of M/TCP are to provide reliability to multicast communication by enhancing mul-

ticast functions to TCP and to support the existing TCP applications without any modification. The result obtained from the experiments shows that M/TCP will be more effective than unicast transmission, when there are more than four receivers in the group. We believe that the advantages of M/TCP, i.e., require no modification to the receivers and alleviate ACK suppression to the receiver would allow M/TCP to be widely deployed on the current Internet.

## 5.2 SIM Future Work

Future work for SIM is as follows:

- Study of an efficient scheme for SIM Neighbor Discovery

  In this dissertation, we did not explain how SIM can discovery SIM-capable neighbor routers. The conventional method is the manual configuration by the network administrators, as done in today's MBone. However, dynamical discovering of SIM-capable neighbor node should be provided in order to support gradual deployment. One of the method is to use *SIM Hello Message*, which is broadcasted to all next hop routers with 1 TTL (Time To Live), as we proposed in [50]. However, a more efficient scheme is required for discovering other SIM-capable routers locating farther than one hop.

- Find an appropriate value for *SIM_FIB_INTERVAL* timer

  In present SIM implementation, the default time interval to attach a receiver list to packet, i.e. the *SIM_FIB_INTERVAL* time, is 10 seconds. While in a congested network, we cannot guarantee that all on-path SIM routers will receive the packets with the receiver list, because we have no kind of acknowledgement between routers and sender. In the case where packets with the receiver list loss, the routers will fail to create an FIB entry of the lost generation number. As a result, lost packets without the receiver list in the same generation number will be occurred. On the other hand, a reliable transmission, such as in M/TCP, the loss packets will be retransmitted out, cause delay in the transmission. First, in order to obtain less packet loss when network failure occurs, the appropriate definition of *SIM_FIB_INTERVAL* time should be defined. We have to study the relationship between the length of *SIM_FIB_INTERVAL* timer and the percentage of packet loss when a network failure occurs. For example, we have to find the average throughput experienced by receivers, when the packet loss is 5%. For the reliable transmission, we have to study how long is the delay in transmission, when a packet loss occurs. This work has to be performed with M/TCP.

## 5.3 M/TCP Future Work

Future work for M/TCP is as follows:

- Extension to socket buffer

  At the present M/TCP implementation, we use only one common socket buffer for a multicast group. Depending on the applications, individual parameters, such as password and user name for FTP server, have to be used in order to transmit data. Similarly, each receiver may return different data back to the sender. In order to provide more redundancy to the applications, we have to treat each receiver separately. In the future, we plan to extend both the send socket buffer (*so_snd*) and receive socket buffer (*so_rcv*).

- Implementing multiple transmission rate

  As depicted in Figure 4.1 in Chapter 4, the transmission rate is based on the capacity of each receiver. However, we have not implemented a dynamic re-grouping mechanism in the present implementation. One of the techniques to support multiple rates is to provide more sender buffers for sub-multicast groups. The other is to fork or duplicate the socket description. To avoid complicated implementation, receivers should be appropriately grouped, according to their receiving performance, before the beginning of transmission. A pre-grouping scheme may be provided at the application level. This scheme should take the path MTU and delay to each receiver in consider for an appropriate grouping.

## 5.4 Future of Multicast

For the future of multicast, there may be two types of multicast models existing in the Internet. Even though there is a trend to bring PIM-SSM as a standard multicast routing protocol, we believe that there is no one-size-fits-all solution to this problem. PIM-SSM is the successor the PIM, which have the scalability issue in supporting a large number of multicast groups; and yet no solution for PIM-SSM is proposed. Multiple multicast routing protocols will still consist in the network, as we have IPv4 and IPv6 at the present.

SIM and IP multicast will serve for a large group as hundreds to thousand receivers. IP multicast will serve for large multicast groups, while SIM will serve for small multicast groups. Xcast may not be existed because the same service can also be served by SIM List mode.

IP multicast will remain in the Internet, but many components would be replaced. First, at host service level, the reliable transmission will be served by reliable multicast. Multicast address allocation will not be required, by using random allocation on the sender host. For reliable multicast, many RM's building block has been approved as RFCs. However, all of them concern with providing reliability to UDP. Supporting multicast to TCP like in our M/TCP method may be included as a subset of the RM's building block with the support of SIM, which can be included in the router assist building block. At host-router level, IGMPv2 and IGMPv1 will be replaced with IGMP version 3. For Inter-domain routing, MSDP and MBGP solution, which have a security vulnerability, may be abandoned. They will be replaced by BGMP.

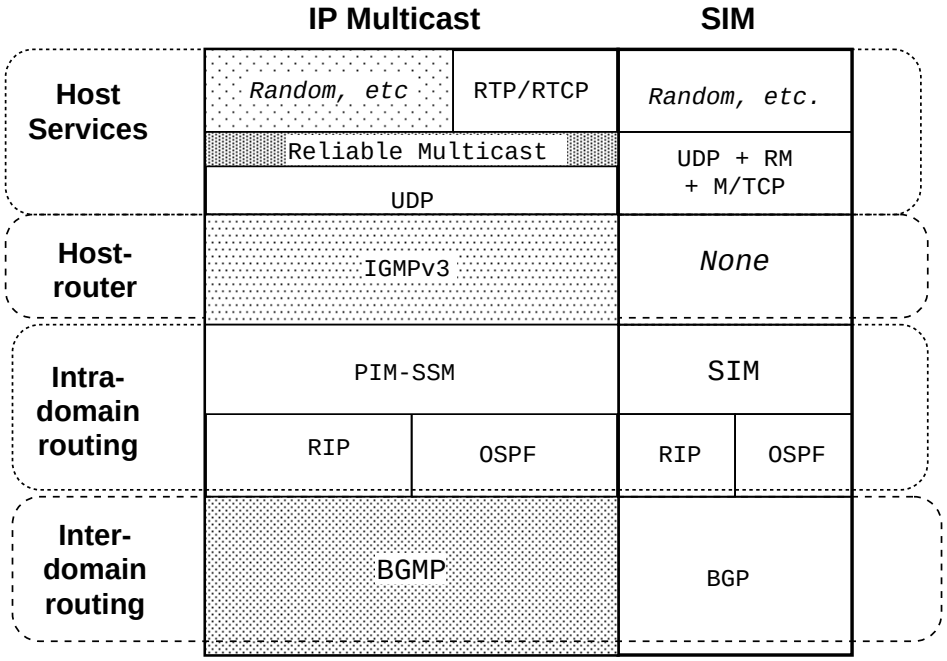



Figure 5.1: Future of multicast technology

The intra-domain routing protocols may be replaced with PIM-SSM, both DVMRP and MOSPF will not be used anymore.

In order to gain wide-scale deployment of multicast, other than multicast routing and reliability issues, access control problems, lack of killer applications, and lack of software upgrade have to be considered. The lack of software upgrade occurs because ISPs still use old routers. Replacing these routers with the new equipments require a large amount of money; therefore ISPs will not replace the equipments when the old ones still can serve the traffics and when they do not find the needs of multicast. In this case, enough knowledge about multicast is required for ISPs' administrators. Therefore, we require simpler architecture for simpler complex deployment. Access control problem should also be resolved. This problem is currently studied by IETF Multicast Security working group and IRTF Group Security working group. It also receives the attention, since everyone began to recognize the importance to provide secure multicast in order to support commercial needs.

While current multicast networks or MBone are not widely deployed and limited only to the research uses, such a few users can use multicast. Therefore, the ISPs do not find the needs, and do not support the multicast. Moreover, most multicast applications today are concerning only to the audio and video conference. Ultimately, the future work for multicast deployment should include developing the killer application, which will give the most impact on multicast deployment, as the customer needs will force the ISPs to support the multicast.

# Bibliography


[1] S. E. Deering. Host extensions for IP multicasting. RFC 1112, Internet Engineering Task Force, August 1989.

[2] Christophe Diot, Brian Neil Levine, Bryan Lyles, Hassan Kassem, and Doug Balensiefen. Deployment issues for the IP multicast service and architecture. *IEEE Network*, 14(1):78–88, January/February 2000.

[3] CAIDA.org Mantra. Session and membership monitor, monitoring characteristics of multicast sessions and participant hosts, March 2002. http://www.caida.org/tools/measurement/Mantra/session-mon/.

[4] David Thaler and Mark Handley. On the aggregatability of multicast forwarding state. In *Proceedings of INFOCOM (3)*, pages 1654–1663, 2000.

[5] P. Radoslavov, D. Estrin, and R. Govindan. Exploiting the bandwidth-memory tradeoff in multicast state aggregation, February 1999.

[6] Jun-Hong Cui, Dario Maggiorini, Jinkyu Kim, Khaled Boussetta, and Mario Gerla. A protocol to improve the state scalability of source specific multicast. In *Proceedings of IEEE Globecom2002*, 2002.

[7] Rick Boivie, Nancy Feldman, Yuji Imai, Wim Livens, Dirk Ooms, , and Olivier Paridaens. Explicit multicast (xcast) basic specification. Internet-draft, Internet Engineering Task Force, June 2002. Work in progress.

[8] Roca Vincent, Costa Luis, Vida Rolland, Dracinschi Anca, and Fdida Serge. A survey of multicast technologies. Technical Report RP-LIP6-2000-09-05, Universit Pierre et Marie Curie, September 2000.

[9] Pragyansmita Paul and S. V. Raghavan. Survey of multicast routing algorithms and protocols. In *Proceedings of the Fifteenth International Conference on Computer Communication (ICCC 2002)*, August 2002.

[10] D. Thaler, M. Handley, and D. Estrin. The internet multicast address allocation architecture. RFC 2908, Internet Engineering Task Force, September 2000.

[11] R. Hinden and S. Deering. IPv6 Multicast Address Assignments. RFC 2375, Internet Engineering Task Force, July 1998.

[12] Internet Assigned Numbers Authority(IANA). Internet multicast addresses, October 2002. http://www.iana.org/assignments/multicast-addresses.

[13] G. Malkin. RIP Version 2. RFC 2453, November 1998.

[14] D. Mills. Network Time Protocol (Version 3) Specification, Implementation. RFC 1305, March 1992.

[15] Mark Handley, David Thaler, and Roger Kermode. Multicast-scope zone announcement protocol (MZAP). RFC 2776, Internet Engineering Task Force, February 2000.

[16] S. Hanna, B. Patel, and M. Shah. Multicast Address Dynamic Client Allocation Protocol (MADCAP). RFC 2730, Internet Engineering Task Force, December 1999.

[17] Sdr multicast session directory. http://www.icir.org/sdr/.

[18] M. Handley, C. Perkins, and E. Whelan. Session Announcement Protocol. RFC 2974, Internet Engineering Task Force, October 2000.

[19] M. Handley and V. Jacobson. SDP: Session Description Protocol. RFC 2327, Internet Engineering Task Force, April 1998.

[20] A. B. Roach. *Session Initiation Protocol (SIP)-Specific Event Notification*, June 2002. RFC 3265.

[21] P. Radoslavov, D. Estrin, R. Govindan, M. Handley, S. Kumar, and D. Thaler. The Multicast Address-Set Claim (MASC) Protocol. RFC 2909, Internet Engineering Task Force, September 2000.

[22] D. Meyer and P. Lothberg. GLOP Addressing in 233/8. RFC 3180, Internet Engineering Task Force, September 2001.

[23] Mark Handley and S. Hanna. Multicast address allocation protocol (aap). Technical report, Internet Engineering Task Force, June 2000. Work in Progress.

[24] W. Fenner. Internet Group Management Protocol, version 2. RFC 2236, Internet Engineering Task Force, November 1997.

[25] S. Deering, W. Fenner, and B. Haberman. Multicast listener discovery (MLD) for IPv6. RFC 2710, Internet Engineering Task Force, October 1999.

[26] D. Waitzman, C. Partridge, and S.E. Deering. Distance vector multicast routing protocol. RFC 1075, Internet Engineering Task Force, November 1988.

[27] Andrew Adams, Jonathan Nicholas, and William Siadak. Protocol independent multicast - dense mode (pim-dm): Protocol specification (revised). Internet-draft, Internet Engineering Task Force, October 2002. Work in progress.

[28] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol independent multicast-sparse mode (PIM-SM): protocol specification. RFC 2362, Internet Engineering Task Force, June 1998.

[29] Bill Fenner. Multicast source discovery protocol (msdp). Internet-Draft Version 2, Internet Engineering Task Force, November 2001. Work in progress.

[30] T. Bates, Y. Rekhter, R. Chandra, and D. Katz. Multiprotocol extensions for BGP-4. RFC 2858, Internet Engineering Task Force, June 2000.

[31] Satish Kumar, Pavlin Radoslavov, David Thaler, Cengiz Alaettinoglu, Deborah Estrin, and Mark Handley. The MASC/BGMP architecture for inter-domain multicast routing. In *Proceedings of SIGCOMM '98*, pages 93–104, Vancouver, British Columbia, September 1998. ACM.

[32] David Thaler. Border gateway multicast protocol (bgmp): Protocol specification. Internet-draft, Internet Engineering Task Force, June 2002. Work in progress.

[33] Kevin Almeroth. The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment. *IEEE Network*, 14:10–20, January/February 2000.

[34] Christian Huitema. *Routing in the Internet*. Prentice Hall, 2 edition, 1999.

[35] Multicast Technologies Inc. Multicast status web page. http://www.multicasttech.com/status/index.html.

[36] K. C. Almeroth, S. Bhattacharyya, and C. Diot. Challenges of integrating ASM and SSM IP multicast protocol architectures. *Lecture Notes in Computer Science*, 2170, 2001.

[37] T. Wong and R. Katz. An analysis of multicast forwarding state scalability. In *Proceedings of the 8th International Conference on Network Protocols (ICNP), Japan*, nov 2000.

[38] Hugh W. Holbrook and David R. Cheriton. IP multicast channels: EXPRESS support for large-scale single-source applications. In *Proceedings of SIGCOMM '99*, pages 65–78, 1999.

[39] Supratik Bhattacharyya, Christophe Diot, Leonard Giuliano, Rob Rockell, John Meylor, David Meyer, Greg Shepherd, and Brian Haberman. An overview of source-specific multicast (ssm). Internet-draft, Internet Engineering Task Force, November 2002. Work in progress.

[40] Brian Haberman and David Thaler. RFC 3306, Internet Engineering Task Force, August 2002.

[41] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan. Internet group management protocol, version 3. RFC 3376, Internet Engineering Task Force, October 2002.

[42] Luis Henrique M.K. Costa, Serge Fdida, and Otto Carlos M.B. Duarte. Hop-by-hop multicast routing protocol. In *Proceedings of ACM SIGCOMM '2001*, pages 249–259, August 2001.

[43] Ljubica Blazevic and Jean-Yves Le Boudec. Distributed core multicast (DCM): A multicast routing protocol for many groups with few receivers. In *Proceedings of Networked Group Communication*, pages 108–125, 1999.

[44] Yuji Imai. Multiple destination option on ipv6(mdo6). Internet-draft, Internet Engineering Task Force, September 2000. Work in progress.

[45] Rick Boivie and Nancy Feldman. Small group multicast. Internet-draft, Internet Engineering Task Force, February 2001. Work in progress.

[46] Rick Bovie, Nancy Feldman, and Christopher Metz. Small group multcast: A new solution for multicasting on the internet, 2000.

[47] Dirk Ooms and Wim Livens. Connectionless multicast,. Internet-draft, Internet Engineering Task Force, April 2000. Work in progress.

[48] C. Graff. IPv4 option for sender directed multi-destination delivery. RFC 1770, Internet Engineering Task Force, March 1995.

[49] Ion Stoica, T. S. Eugene Ng, and Hui Zhang. REUNITE: A recursive unicast approach to multicast. In *Proceedings of INFOCOM (3)*, pages 1644–1653, 2000.

[50] Vasaka Visoottiviseth and Yosuke Takahashiand Noritoshi Demizu. Sender initiated multicast. Internet-draft, Internet Engineering Task Force, July 2001. Work in progress.

[51] Yosuke Takahashi. Design and implementation of small group multicast routing protocol: Sim. Technical Report NAIST-IS-MT9951063, Nara Institute of Sceince and Technology, February 2001.

[52] Hiroyuki Kido. Performance evalution of router processing load and group scalability in explicit multicast. Technical Report NAIST-IS-MT0051028, Nara Institute of Sceince and Technology, February 2002.

[53] Michael Franklin and Stan Zdonik. Data in your face: push technology in perspective. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 516–519, June 1998.

[54] Vasaka Visoottiviseth, Youki Kadobayashi, and Suguru Yamaguchi. An asymmetrical group management system for sender initiated multicast. In *IPSJ SIG Notes*, number 88, pages 31–36, September 2001.

[55] Bradley Huffaker, Marina Fomenkov, Daniel J. Plummer, David Moore, and k claffy. Distance metrics in the internet. In *Proceedings of IEEE International Telecommunications Symposium (ITS)*.

[56] Oliver Hermanns. Performance evaluation of connectionless multicast protocals for cooperative multimedia applications. In *Proceedings of Messung, Modellierung und Bewertung von*, pages 372–384, 1995. citeseer.nj.nec.com/article/hermanns95performance.html.

[57] Japan Gigabit Network. Topology of japan gigabit network. http://www.jgn.tao.go.jp/org_tec/kousei_map.html.

[58] T. Billhartz, J. B. Cain, E. Farrey-Goudreau, D. Fieg, and S. Batsell. Performance and resource cost comparison for the cbt and pim multicast rouging protocols. In *Proceedings of INFOCOM '96, vol. 1, San Francisco, USA*, pages 85–93, 1996.

[59] University of Southern California. Pim in gated (5.0 alpha). http://www.isi.edu/ eddy/pim/pim.html.

[60] University of Southern California. Pim-dm (standalone pim-sm with patch). http://www.isi.edu/ eddy/pim/pim.html.

[61] Olivier Paridaens, Dirk Ooms, and Bernad Sales. Security framework for explicit multicast. Internet-draft, Internet Engineering Task Force, June 2002. Work in progress.

[62] J. Reynolds and J. Postel. Assigned numbers. RFC 1700, Internet Engineering Task Force, October 1994.

[63] S. Kent and R. Atkinson. Security architecture for the internet protocol. RFC 2401, Internet Engineering Task Force, November 1998.

[64] Internet Research Task Force. The secure multicast research group (smug). http://www.securemulticast.org/smug-index.htm.

[65] S. Kent and R. Atkinson. IP authentication header. RFC 2402, Internet Engineering Task Force, November 1998.

[66] S. Kent and R. Atkinson. IP encapsulating security payload (ESP). RFC 2406, Internet Engineering Task Force, November 1998.

[67] Jon Postel. Transmission control protocol. RFC 793, Internet Engineering Task Force, September 1981.

[68] Katia Obraczka. Multicast transport protocols: a survey and taxonomy, January 1998.

[69] Vasaka Visoottiviseth, Takuya Mogami, and Suguru Yamaguchi. M/tcp: Multicast extension to tcp. In *Proceedings of ICACT 2001*, Muju, Korea, February 2000.

[70] Takuya Mogami. M/tcp: Tcp extension for reliable multicast communication. Technical Report NAIST-IS-MT9951113, Nara Institute of Sceince and Technology, February 2001.

[71] J. Postel and J. K. Reynolds. File transfer protocol. RFC 959, Internet Engineering Task Force, October 1985.

[72] Jon Postel. Simple mail transfer protocol. RFC 821, Internet Engineering Task Force, August 1982.

[73] B. Kantor and P. Lapsley. Network news transfer protocol. RFC 977, Internet Engineering Task Force, February 1986.

[74] A. Mankin, A. Romanow, S. Bradner, and V. Paxson. IETF criteria for evaluating reliable multicast transport and application protocols. RFC 2357, Internet Engineering Task Force, June 1998.

[75] M. Handley, S. Floyd, B. Whetten, R. Kermode, L. Vicisano, and M. Luby. The reliable multicast design space for bulk data transfer. RFC 2887, Internet Engineering Task Force, August 2000.

[76] B. Whetten, L. Vicisano, R. Kermode, M. Handley, S. Floyd, and M. Luby. Reliable multicast transport building blocks for one-to-many bulk-data transfer. RFC 3048, Internet Engineering Task Force, January 2001.

[77] Sally Floyd, Van Jacobson, Ching-Gung Liu, Steven McCanne, and Lixia Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, 5(6):784–803, December 1997.

[78] Joe Macker and Brian Adamson. Mdp history. http://manimac.itd.nrl.navy.mil/MDP/mdphist.html.

[79] K. Lin and S. Paul. A reliable multicast transport protocol. In *Proceedings of IEEE INFOCOMM 1996*, pages 1414–1424, March 1996.

[80] Brian Whetten, Murali Basavaiah, Sanjoy Paul, Todd Montgomery, Naveen Rastogi, Jim Conlan, and Thomas Yeh. The rmtp-ii protocol. Internet-draft, Internet Engineering Task Force, April 1998. Work in progress.

[81] Miriam Kadansky, Dah Ming Chiu, Joe Wesley, and Joe Provino. Tree-based reliable multicast (tram). Internet-draft, Internet Engineering Task Force, January 2000. Work in progress.

[82] M. Luby, J. Gemmell, L. Vicisano, L. Rizzo, and J. Crowcroft. Asynchronous Layered Coding (ALC) Protocol Instantiation. RFC 3450, Internet Engineering Task Force, December 2002.

[83] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft. Forward error correction building block. RFC 3452, Internet Engineering Task Force, December 2002.

[84] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft. The use of forward error correction in reliable multicast. RFC 3453, Internet Engineering Task Force, December 2002.

[85] Joe Macker and Brian Adamson. Multicast dissemination protocol version 2 (mdpv2). Work in Progress, http://manimac.itd.nrl.navy.mil/MDP.

[86] T. Speakman, J. Crowcroft, J. Gemmell, D. Farinacci, S. Lin, D. Leshchiner, M. Luby, T. Montgomery, L. Rizzo, A. Tweedly, N. Bhaskar, R. Edmonstone, R. Sumanasekera, and L. Vicisano. *PGM Reliable Transport Protocol Specification*, December 2001. RFC 3208.

[87] B.N. Levine and J.J. Garcua-Luna-Aceves. Improving internet multicast routing with routing labels. In *Proceedings of IEEE International Conference on Network Protocols (ICNP-97)*, pages 241–250, October 1997.

[88] Joerg Widmer and Mark Handley. Tcp-friendly multicast congestion control (tfmcc): Protocol specification. Internet-draft, Internet Engineering Task Force, November 2002. Work in progress.

[89] Luigi Rizzo. pgmcc: a TCP-friendly single-rate multicast. In *Proceedings of SIGCOMM '2000*, pages 17–28, 2000.

[90] Lorenzo Vicisano, Luigi Rizzo, and Jon Crowcroft. TCP-like congestion control for layered multicast data transfer. In *Proceedings of INFOCOM (3)*, pages 996–1003, 1998.

[91] Naohiko Takura and et al. An approach to realize multicast facility in tcp protocol. In *Proceedings of Multimedia communication and Distributed Computing Workshop*, 1992.

[92] Sam Liang and David Cheriton. Tcp-smo: Extending tcp to support medium-scale multicast applications. 2002.

[93] Steven McCanne, Van Jacobson, and Martin Vetterli. Receiver-driven layered multicast. In *Proceedings of ACM SIGCOMM '96*, volume 26,4, pages 117–130, New York, August 1996. ACM Press.

[94] Jon Crowcroft and Karen Paliwoda. A multicast transport protocol. In *Proceedings of ACM SIGCOMM '98*, pages 247–256, 1998.

[95] George Varghese, Christos Papadopoulos, and Guru M. Parulkar. An error control scheme for larg-scale multicast applications. In *Proceedings of IEEE INFOCOM*, 1998.

# Appendix A

# List of Publications

## A.1 Journal

1. ワサカ ヴィスーティヴィセット, 門林雄基, 山口英, “少人数を対象とした送信者指定型マルチキャストプロトコルSIM”, 電子情報通信学会論文誌 VOL.J85-B No.8, August 2002.

2. Vasaka Visoottiviseth, Hiroyuki Kido, Youki Kadobayashi, Suguru Yamaguchi, ”Design and Evaluation of Sender-Initiated Multicast”, IEEE Transaction of Networking (Under Submission).

## A.2 International Conference

1. Vasaka Visoottiviseth, Takuya Mogami, Noritoshi Demizu, Youki Kadobayashi, Suguru Yamaguchi, “M/TCP: The Multicast-extension to Transmission Control Protocol”, In Proc. of ICACT2001, Muju, Korea, Feb 2001.

2. Vasaka Visoottiviseth, Youki Kadobayashi, Suguru Yamaguchi, “SIM: Sender Initiated Multicast for small group communications”, In Proc. of INET ’2001, Stockholm.

3. Vasaka Visoottiviseth, Hiroyuki Kido, Youki Kadobayashi, Suguru Yamaguchi, “Sender-initiated Multicast Forwarding Scheme”, In Proc. of ICT2003, Tahiti, Feb 2003.

## A.3 Internet Draft

1. Vasaka Visoottiviseth, Yosuke Takahashi, Noritoshi Demizu, ”draft-vasaka-xcast-sim-00.txt”, Internet draft, July 2001.

### A.3.1 Domestic Conference

1. ワサカ・ヴィスーティヴィセット, 門林雄基, 山口英: “ 送信者指定型マルチキャストにおける非対称なメンバー管理方式の提案 ”, 研究報告 「マルチメディア通信と分散処理」No.104 - 006, 2001 年 9 月.