Scalable Multicast Provisioning in IP Differentiated Services Networks

Ning Wang

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

UniS

Centre for Communication Systems Research
School of Electronics and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, UK

December 2004

© Ning Wang 2004
Summary

The emergence of point-to-multipoint applications with Quality of Service (QoS) requirements in the Internet has prompted research towards the deployment of multicast communications in Differentiated Services (DiffServ) environments. However, despite many past research efforts, global availability of IP multicast is still a pie in the sky for Internet users, let alone applications with QoS guarantees. One of the key factors that hamper associated progress is scalability, in terms of various types of states associated with routing and signaling in both multicast and QoS.

In this thesis we aim at a scalable architectural design of multicast service provisioning for end users with heterogeneous QoS requirements, targeted to the DiffServ environment.

Our architecture consists of three planes: management, control and data plane. First of all, we design and evaluate the Offline Multicast Traffic Engineering (OMTE) building block in the management plane for QoS aware multicast service dimensioning. The main novelty of this scheme is that we shift away from the commonly used Multi-Protocol Label Switching (MPLS) based traffic engineering, and address the bandwidth constrained IP multicast TE directly based on link state routing protocols. With this approach, end-to-end performance can be achieved without MPLS explicit routing that potentially suffers from scalability problems in terms of Label Switching Path (LSP) maintenance and is relatively expensive to deploy.

In the control plane, we propose two different paradigms. QoS aware Source Specific Multicast (QSSM) is designed for dedicated multicast delivery tree construction in different QoS classes, while another overlay scheme, known as Differentiated QoS Multicast (DQM), attempts to build a single hybrid tree that exhibits heterogeneous QoS channels within the network. In both approaches, multicast group addresses are used to encode QoS class information, and the associated benefit is reflected in scalability and backwards compatibility: neither underlying multicast protocols nor existing routers need any extension for carrying and maintaining QoS states within the network.

Finally, envisaging the importance of protecting dimensioned resources from Denial-of-Service (DoS) attacks from malicious hosts, we propose the Multicast Sender Access Control (MSAC) mechanism, which is indispensable in multicast security, but still lacks significant attention from the research community. We focus on bi-directional multicast trees, which is the most vulnerable routing paradigm to DoS attacks. Both intra- and inter-domain control mechanisms are addressed with scalability considerations in mind.
Key words: IP multicast, Source Specific Multicast (SSM), Quality of Service (QoS), Differentiated Services (DiffServ), Traffic Engineering (TE), sender access control, scalability.
Acknowledgments

First of all, I would like to express my sincere gratitude to my supervisor, Professor George Pavlou, without whom I would have not accomplished this PhD research project, for his guidance with great passion, his priceless efforts in focusing my research capacity, as well as his enthusiasm and patience during his supervision ever since I joined the University of Surrey. I particularly would like to thank Professor Pavlou for his help and guidance for improving significantly my English writing skills.

I would also like to specially thank my dear parents and Lingyu, for their everlasting encouragement and solid support during this period.

In addition, my appreciation also goes to my colleagues in the networks group, CCSR, and in particular to Panos Trimintzios, Paris Flegkas, Michael Howarth, Kin-Hon Ho and Stelios Georgoulas, for their valuable help in my research project; I have really enjoyed working with them. Besides, I would like to thank the computing support people at CCSR for their hard work in providing an excellent environment for research.

I would also like to acknowledge the contribution from the external partners in the EU IST MESCAL project, for their comments and suggestions on the design of QoS-aware multicast framework in my research work.

Finally, I have appreciated the effort of all the anonymous reviewers of my submitted journal and conference papers for their useful suggestions towards the achievements of my research work.
# Contents

Summary ...........................................................................................................  ii
Acknowledgments ...........................................................................................................iv
Contents ..............................................................................................................................................v
List of Figures ................................................................................................................................viii
List of Tables ....................................................................................................................................x
Glossary of Terms ...........................................................................................................................xi
Thesis Related Publications ..................................................................................................... xv

1 Introduction...................................................................................................................  16
  1.1 Background and Motivation ............................................................................................. 16
  1.2 Framework Overview ...................................................................................................... 18
    1.2.1 Functional Architecture ............................................................................................ 19
    1.2.2 Building Block Functionality .................................................................................... 20
  1.3 Novelty ............................................................................................................................. 21
  1.4 Thesis Structure ................................................................................................................. 22

2 Literature Review........................................................................................................................24
  2.1 Multicast Service Models ................................................................................................. 24
    2.1.1 IP Multicast ............................................................................................................ 24
    2.1.2 Source Specific Multicast (SSM). ............................................................................ 24
  2.2 Multicast Group Management ............................................................................................ 25
  2.3 Multicast Routing .............................................................................................................. 26
    2.3.1 PIM-SM Routing ................................................................................................... 26
    2.3.2 Inter-domain Multicast Routing ............................................................................ 27
    2.3.3 QoS-aware Multicast Routing ............................................................................... 28
  2.4 Differentiated Services ....................................................................................................... 30
    2.4.1 DiffServ Architecture ............................................................................................ 30
    2.4.2 Multicast in DiffServ Networks ............................................................................. 31
  2.5 Traffic Engineering (TE) ................................................................................................. 34
    2.5.1 MPLS Based TE .................................................................................................... 34
    2.5.2 IP Based TE ........................................................................................................... 36
3 Offline Multicast Traffic Engineering (OMTE) ................................................................. 38
  3.1 Introduction ...................................................................................................................... 38
  3.2 Multicast Service Level (mSLS) Specification ................................................................. 39
  3.3 OMTE Overview .............................................................................................................. 40
  3.4 M-ISIS/PIM-SM based OMTE Framework ...................................................................... 43
  3.5 Problem Formulation ........................................................................................................ 45
  3.6 A Genetic Algorithm Based Solution ............................................................................. 47
    3.6.1 Encoding and Initial Population ................................................................................ 48
    3.6.2 Fitness Evaluation ...................................................................................................... 48
    3.6.3 Crossover and Mutation ............................................................................................ 50
  3.7 GA Processing Analysis ................................................................................................... 51
    3.7.1 Optimisation Scenarios ............................................................................................. 51
    3.7.2 Time Complexity Analysis ......................................................................................... 53
  3.8 Simulation Results ........................................................................................................... 54
    3.8.1 Simulation Configuration ............................................................................................ 54
    3.8.2 Performance Evaluation ............................................................................................. 55
  3.9 Summary .......................................................................................................................... 58

4 DiffServ-aware Multicast using QSSM ................................................................................ 60
  4.1 Introduction ...................................................................................................................... 60
  4.2 Router Extension Requirements with Conventional Approaches .................................... 61
  4.3 QSSM Semantics ............................................................................................................. 63
    4.3.1 QoS Mapping ............................................................................................................ 63
    4.3.2 QSSM Tree Management ....................................................................................... 65
  4.4 Inter-domain QSSM Operations ...................................................................................... 67
  4.5 Dynamic Group Management in DiffServ Networks ..................................................... 70
  4.6 Simulation Results ........................................................................................................... 71
    4.6.1 Simulation Scenario .................................................................................................. 71
    4.6.2 Performance Comparison within One Class ............................................................. 72
    4.6.3 OMTE-Driven Performance in DiffServ Networks ................................................. 77
  4.7 Summary .......................................................................................................................... 79

5 DQM: An Alternative QoS Approach ................................................................................ 80
  5.1 Introduction ...................................................................................................................... 80
  5.2 Review of the MQ Approach ........................................................................................... 81
  5.3 The proposed DQM solution ........................................................................................... 82
    5.3.1 Overview .................................................................................................................. 82
List of Figures

Figure 1-1 Proposed multicast functional architecture ..........................................................20
Figure 2-1 Overview of DiffServ network ..................................................................................30
Figure 2-2 DiffServ edge router functionality ...........................................................................31
Figure 2-3 The NRS Problem ...................................................................................................32
Figure 2-4 QUASIMODO multicast forwarding table ...............................................................33
Figure 2-5 Fortz and Thorup's link weight optimisation algorithm ........................................36
Figure 3-1 Multicast business relationship ..............................................................................40
Figure 3-2 Multicast routing using different approaches .........................................................42
Figure 3-3 M-ISIS based multicast traffic engineering ............................................................45
Figure 3-4 Fitness calculation ..................................................................................................50
Figure 3-5 Crossover and mutation .........................................................................................51
Figure 3-6 GA optimisation process (Scenario I) ......................................................................52
Figure 3-7 GA optimisation process (Scenario II) .................................................................53
Figure 3-8 GA Success rate vs. $MLOR_{shm}$ ........................................................................56
Figure 3-9 Total bandwidth consumption vs. Max $D_g$ .........................................................57
Figure 3-10 Overloaded link rate vs. Max $D_g$ .....................................................................58
Figure 3-11 MLOR vs. Max $D_g$ ..........................................................................................58
Figure 4-1 QSM vs. HQM .......................................................................................................63
Figure 4-2 QSSM mapping table ............................................................................................64
Figure 4-3 QSSM framework overview ..................................................................................65
Figure 4-4 QSSM group join procedure .................................................................................67
Figure 4-5 Inter-domain group address conversion ...............................................................68
Figure 4-6 Inter-domain QSSM group join and data transmission .......................................69
Figure 4-7 IGMP operation in QSM and QSSM ....................................................................71
Figure 4-8 Real-time performance in average network load (Max $D_g=3000$, $\omega=1$) ....73
Figure 4-9 Real-time performance in maximum link utilisation (Max $D_g=3000$, $\omega=1$) ...73
Figure 4-10 Real-time performance in average network load (Max $D_g=6000$, $\omega=1$) ....74
Figure 4-11 Real-time performance in maximum link utilisation (Max $D_g=6000$, $\omega=1$) ..75
Figure 4-12 Join block rate vs. invocation ratio $\omega$ ...............................................................76
Figure 4-13 Network load vs. invocation ratio $\omega$ .................................................................77
Figure 4-14 Real-time performance in network load ($\omega=1$)............................................................78
Figure 4-15 Real-time performance in max. link utilisation. ($\omega=1$)..............................................79
Figure 5-1 MQ group join and tree reshaping...................................................................................82
Figure 5-2 DQM group state maintenance .......................................................................................83
Figure 5-3 Packet replication in DQM .............................................................................................84
Figure 5-4 Packet aggregation in DQM...........................................................................................85
Figure 5-5 Flowchart for Handling DQM Group Join .................................................................87
Figure 5-6 Dynamic QoS channel subscription ..............................................................................88
Figure 5-7 Dynamic QoS channel unsubscription ...........................................................................89
Figure 5-8 Flowchart for Handling DQM Group Leaving .............................................................91
Figure 5-9 Overhead of bandwidth conservation............................................................................93
Figure 5-10 Overall link utilisation comparison with MQ.............................................................94
Figure 5-11 Overall channel states maintenance ..........................................................................95
Figure 5-12 Individual channel states maintenance .....................................................................96
Figure 6-1 Uni-directional routing vs. Bi-directional routing.......................................................99
Figure 6-2 Sender access control policy comparison ....................................................................102
Figure 6-3 Flowchart for Router Actions in SACL Management ..............................................104
Figure 6-4 Flowchart for Router Actions in SACL Management ..............................................105
Figure 6-5 Intra-domain SACL construction and activation .......................................................108
Figure 6-6 Inter-domain join and activation ..............................................................................110
Figure 6-7 Example for Inter-domain sender access control ....................................................112
Figure 6-8 Access control operation on LANs ............................................................................113
Figure 6-9 Inter-domain simulation model ..................................................................................115
Figure 6-10 SACL size vs. number of senders ...........................................................................116
Figure 6-11 SACL size vs. number of send-only hosts ...............................................................117
Figure 6-12 Average SACL Size vs. group size ..........................................................................118
Figure 6-13 SACL size distribution ..............................................................................................119
Figure 6-14 SACL size vs. number of senders (inter-domain) .......................................................121
Figure 6-15 SACL size vs. number of send-only hosts (inter-domain) .........................................122
Figure 6-16 Average SACL Size vs. group size (Inter-domain) .................................................123
List of Tables

Table 2-1 Summary of constrained and unconstrained Steiner tree problems ........................................... 29
Table 3-1 GA parameter configuration ........................................................................................................... 55
Table 4-1 Multi-QC block ratio performance comparison .................................................................................. 79
Table 5-1 Traffic distribution comparison with MQ ......................................................................................... 94
Table 6-1 Comparison with FPM (Average SACL size) ................................................................................. 118
# Glossary of Terms

<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS</td>
<td>Autonomous System</td>
</tr>
<tr>
<td>ASM</td>
<td>Any Source Multicast</td>
</tr>
<tr>
<td>BB</td>
<td>Bandwidth Broker</td>
</tr>
<tr>
<td>BE</td>
<td>Best Effort</td>
</tr>
<tr>
<td>BGMP</td>
<td>Border Gateway Multicast Protocol</td>
</tr>
<tr>
<td>BGP</td>
<td>Border Gateway Protocol</td>
</tr>
<tr>
<td>Bidir-PIM</td>
<td>Bi-directional Protocol Independent Multicast</td>
</tr>
<tr>
<td>BR</td>
<td>Border Router</td>
</tr>
<tr>
<td>CBT</td>
<td>Core Based Tree</td>
</tr>
<tr>
<td>CR-LDP</td>
<td>Constrained Routing – Label Distribution Protocol</td>
</tr>
<tr>
<td>DBR</td>
<td>Designated Border Router</td>
</tr>
<tr>
<td>DGM</td>
<td>Dynamic Group Management</td>
</tr>
<tr>
<td>DiffServ</td>
<td>Differentiated Services</td>
</tr>
<tr>
<td>DI</td>
<td>Downstream Interface</td>
</tr>
<tr>
<td>DIS</td>
<td>Distribution Interactive Simulation</td>
</tr>
<tr>
<td>DMR</td>
<td>Dynamic Multicast Routing</td>
</tr>
<tr>
<td>DoS</td>
<td>Denial of Service</td>
</tr>
<tr>
<td>DQM</td>
<td>Differentiated QoS Multicast</td>
</tr>
<tr>
<td>DR</td>
<td>Designated Router</td>
</tr>
<tr>
<td>DSCP</td>
<td>Differentiated Services Code Point</td>
</tr>
<tr>
<td>DVMRP</td>
<td>Distance Vector Multicast Routing Protocol</td>
</tr>
<tr>
<td>EXPRESS</td>
<td>EXPlicitly REquest Single Source</td>
</tr>
<tr>
<td>FEC</td>
<td>Forwarding Equivalence Class</td>
</tr>
<tr>
<td>FIB</td>
<td>Forwarding Information Base</td>
</tr>
<tr>
<td>FPM</td>
<td>Full Policy Maintenance</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>GA</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>G-RIB</td>
<td>Group Routing Information Base</td>
</tr>
<tr>
<td>HQM</td>
<td>Hybrid QoS Multicast</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IGMP</td>
<td>Internet Group management Protocol</td>
</tr>
<tr>
<td>IGP</td>
<td>Interior Gateway Protocol</td>
</tr>
<tr>
<td>iif</td>
<td>Incoming interface</td>
</tr>
<tr>
<td>INP</td>
<td>Internet Network Provider</td>
</tr>
<tr>
<td>IntServ</td>
<td>Integrated Services</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IR</td>
<td>Ingress Router</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>IS-IS</td>
<td>Intermediate System to Intermediate System</td>
</tr>
<tr>
<td>ISIS-TE</td>
<td>Intermediate System to Intermediate System Traffic Engineering</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LBE</td>
<td>Less than Best Effort</td>
</tr>
<tr>
<td>LSA</td>
<td>Link State Advertisement</td>
</tr>
<tr>
<td>LSP</td>
<td>Label Switching Path</td>
</tr>
<tr>
<td>LSR</td>
<td>Label Switching Router</td>
</tr>
<tr>
<td>MASC</td>
<td>Multicast Address Set Claim</td>
</tr>
<tr>
<td>MBGP</td>
<td>Multi-Protocol Border Gateway Protocol</td>
</tr>
<tr>
<td>M-FIB</td>
<td>Multicast Forwarding Information Base</td>
</tr>
<tr>
<td>M-IGP</td>
<td>Multicast Interior Gateway Protocol</td>
</tr>
<tr>
<td>M-ISIS</td>
<td>Multi-topology ISIS</td>
</tr>
<tr>
<td>MLOR</td>
<td>Maximum Link Overload Rate</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multi-Protocol Label Switching</td>
</tr>
<tr>
<td>MQ</td>
<td>Multicast QoS</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>M-RIB</td>
<td>Multicast Routing Information Base</td>
</tr>
<tr>
<td>MSDP</td>
<td>Multicast Source Discovery Protocol</td>
</tr>
<tr>
<td>mSLS</td>
<td>Multicast Service Level Specification</td>
</tr>
<tr>
<td>MSAC</td>
<td>Multicast Sender Access Control</td>
</tr>
<tr>
<td>MT-ID</td>
<td>Multi Topology IDentifer</td>
</tr>
<tr>
<td>MT-IGP</td>
<td>Multi-Topology Interior Gateway Protocol</td>
</tr>
<tr>
<td>NLRI</td>
<td>Network Layer Reachability Information</td>
</tr>
<tr>
<td>NMS</td>
<td>Non-Member Sender</td>
</tr>
<tr>
<td>NRS</td>
<td>Neglected Reservation Sub-tree</td>
</tr>
<tr>
<td>oif</td>
<td>Outgoing interface</td>
</tr>
<tr>
<td>OMTE</td>
<td>Offline Multicast Traffic Engineering</td>
</tr>
<tr>
<td>OSPF</td>
<td>Open Shortest Path First</td>
</tr>
<tr>
<td>OSPF-TE</td>
<td>OSPF-Traffic Engineering</td>
</tr>
<tr>
<td>PHB</td>
<td>Per Hop Behaviour</td>
</tr>
<tr>
<td>PIM-DM</td>
<td>Protocol Independent Multicast – Dense Mode</td>
</tr>
<tr>
<td>PIM-SM</td>
<td>Protocol Independent Multicast – Sparse Mode</td>
</tr>
<tr>
<td>QC</td>
<td>QoS Class</td>
</tr>
<tr>
<td>QMRP</td>
<td>QoS aware Multicast Routing Protocol</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>QSM</td>
<td>QoS Specific Multicast</td>
</tr>
<tr>
<td>QSSM</td>
<td>QoS aware Source Specific Multicast</td>
</tr>
<tr>
<td>RAMA</td>
<td>Root Addressed Multicast Architecture</td>
</tr>
<tr>
<td>RIB</td>
<td>Routing Information Base</td>
</tr>
<tr>
<td>RP</td>
<td>Rendezvous Point</td>
</tr>
<tr>
<td>RPC</td>
<td>Resource Provisioning Cycle</td>
</tr>
<tr>
<td>PRF</td>
<td>Reverse Path Forwarding</td>
</tr>
<tr>
<td>RSVP</td>
<td>ReSource Reservation Protocol</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>RSVP-TE</td>
<td>RSVP Traffic Engineering</td>
</tr>
<tr>
<td>SACL</td>
<td>Sender Access Control List</td>
</tr>
<tr>
<td>SAFI</td>
<td>Subsequent Address Family Identifier</td>
</tr>
<tr>
<td>SAS</td>
<td>Source Authorisation Server</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td>SLS</td>
<td>Service Level Specification</td>
</tr>
<tr>
<td>SOM</td>
<td>Send-Only Member</td>
</tr>
<tr>
<td>SP</td>
<td>Shortest Path</td>
</tr>
<tr>
<td>SRM</td>
<td>Send-Receive capable Member</td>
</tr>
<tr>
<td>SSM</td>
<td>Source Specific Multicast</td>
</tr>
<tr>
<td>TE</td>
<td>Traffic Engineering</td>
</tr>
<tr>
<td>TEQUILA</td>
<td>Traffic Engineering for Quality of Service in the Internet, at Large Scale</td>
</tr>
<tr>
<td>TTL</td>
<td>Time To Live</td>
</tr>
<tr>
<td>UI</td>
<td>Upstream Interface</td>
</tr>
<tr>
<td>YAM</td>
<td>Yet Another Multicast</td>
</tr>
</tbody>
</table>
Thesis Related Publications

- Journal Papers


- Conference Papers


Chapter 1

1 Introduction

1.1 Background and Motivation

The Internet is experiencing a tremendous transition from point-to-point communications with best effort (BE) traffic delivery towards a multi-service platform supporting various multimedia applications with Quality of Service (QoS) requirements. Despite the fact that the cost of increasing physical network capacity is becoming cheaper (e.g., adding high-speed switching and routing elements, high capacity network links, etc.), the strategy of over-provisioning cannot satisfy the demand of the sharply growing customer traffic. With bandwidth-intensive applications deployed on the Internet, such as Internet TV/radio, teleconferencing, distributed network games, etc, bandwidth resources remain precious and congestion is still experienced, mostly in access networks and inter-domain links but also in the backbone.

As many multimedia applications involve point-to-multipoint communication, S. Deering et al proposed the IP Multicast service model [26] for bandwidth conservation purposes. Unfortunately, despite decades of research efforts, global availability of IP multicast services is still a pie in the sky for Internet customers and users until now. IP multicast is also known as “Any Source Multicast (ASM)” since any information source can send data traffic to a multicast group without any control mechanism. In other words, in the current ASM service model group management is not strict enough to control the behaviour of both sources and receivers. Moreover, scalability is another key issue that prevents the successful deployment of IP multicast throughout the Internet. Typically, this problem includes: (1) heavy overhead in group state maintenance, (2) shortage of multicast group addresses in IPv4, and (3) Inter-domain source discovery when the multicast tree spans multiple Autonomous Systems (ASs). Recently, with the successful development of Multicast Source Discovery Protocol (MSDP) [31], the common practice in IP multicast deployment tends to use Protocol Independent Multicast Sparse Mode (PIM-SM) [32] and Multi-protocol BGP (MBGP) [8] respectively as the intra- and inter-domain multicast routing protocols, with MSDP taking the responsibility of discovering remote sources across multiple ASs. At the network boundary, the Internet Group Management Protocol (IGMP) [17, 33] is running on Designated Routers (DRs) for dealing with dynamic group joins and leavings. In the late 1990's, realising that many multicast applications are based on a one-to-many
communication model, e.g. Internet TV/radio, content distribution, etc., H. W. Holbrook et al proposed the EXPRESS routing scheme [39], from which the Source Specific Multicast (SSM) [9] service model has subsequently evolved to gain more and more popularity. In SSM each group is identified by an address tuple (S, G) where S is the unique address of the information source and G is the destination channel address. A single multicast tree is built, rooted at the well-known source for delivering data to all subscribers. Compared to the traditional IP multicast, the distinct advantages of SSM can be summarised as follows. First, the scalability of multicast address allocation is not a problem, as each information source can use the whole range of the assigned address space 232.0.0.0/8. Second, inter-domain multicast source discovery mechanism (e.g., MSDP) is not mandatory because each DR is able to send explicit group joins towards any remote source over the Internet. Finally, centralised source authorisation and authentication can be achieved at the root of the single source through application level mechanisms. SSM is currently receiving significant attention from the research community, believed to be a promising alternative to IP multicast before a more sophisticated solution is possibly invented in the future.

Due to the fact that most multicast applications involve multimedia stream delivery which requires Quality of Service guarantees, various schemes have been proposed to integrate QoS mechanisms with multicast services. Research works on QoS aware multicast in the literature can be classified into three categories. At an early stage (beginning and mid 1990's), centralised multicast routing with QoS constrains (e.g., bandwidth, delay, delay variation, etc.) was the major topic which attracted significant research attention. QoS-aware multicast routing is generally formulated as a constrained Steiner tree problem, which is NP-complete, and various heuristic approaches were proposed for different optimisation objectives [43, 50, 64, 83]. Since the path calculation of constrained multicast trees is fundamentally time-consuming, most of these approaches should be applied as offline schemes with manual configuration of the resulted routing repository. The second category concerns distributed algorithms/protocols for QoS-aware multicast routing with local or global search for feasible paths. Although most of these approaches can be used in real-time routing without full knowledge of the network topology, they still cannot be directly supported in the legacy protocols in either the IP multicast service model or SSM. In this sense, it is more reasonable to regard them as overlay routing protocols over the underlying IP network. In late 1990's, the Differentiated Services (DiffServ) framework [11] was proposed, and it is seen as the most promising solution to the QoS deployment because of its simplicity and scalability. In the DiffServ architecture, the core network is kept relatively simple, with most of the complexity confined at the network edge and the management plane. Specifically, admission control and traffic conditioning are performed at edge routers, while core routers simply treating traffic aggregates on a Per Hop Behaviour (PHB) basis according to the Differentiated Services Code Point (DSCP) in each packet header. Not surprisingly, various attempts have been made
towards the integration of multicast services with the DiffServ architecture ever since the birth of the latter, typical examples including DSMCast [70, 71] and QUOSIMODO [10]. The advantages are obvious. First, heterogeneous QoS requirements from individual group members can be natively supported through different QoS classes. Second, multicast traffic belonging to the same QoS class can still be treated in an aggregate fashion within the network for scalability purposes. Finally, since service differentiation is realised through DiffServ Per Hop Behaviours in an orthogonal manner to path computations (note that QoS routing is the solution for achieving multicast "QoS awareness" without DiffServ), it is possible to achieve QoS multicast without modifying the already deployed multicast routing protocols (e.g., PIM-SM). From a feasibility point of view, solutions in the last category are the most promising, since there is no need to introduce extra complexity and overhead to the network layer specifically for QoS-aware multicast routing. On the other hand, most proposed schemes address integration issues of DiffServ-aware Multicast in the control plane. However, efficient network dimensioning and resource provisioning for supporting multicast services with end-to-end QoS heterogeneity is still an unsolved issue. To the best of our knowledge, no research work has yet addressed a comprehensive and systematic solution to network provisioning for end-to-end multicast QoS guarantees in DiffServ.

Today QoS provisioning is closely linked to Internet Traffic Engineering (TE) [4]. In [4], TE is defined as large-scale network engineering for dealing with IP network performance evaluation and optimisation. Its key task is to enhance network performance while at the same time optimising resource utilisation. It has become a common belief that end-to-end QoS requirements can be supported through efficient TE mechanisms. Many research works have investigated the use of traffic engineering for guaranteeing QoS for unicast traffic in DiffServ networks [56, 74, 79]. Again, traffic engineering for QoS-demanding multicast traffic still remains a dark area till now. Provisioning network resources in order to achieve a good grade of service is a major concern for Internet Service Providers (ISPs). Motivated from this, we propose a comprehensive framework for achieving QoS multicasting in DiffServ networks through efficient multicast traffic engineering. This framework is not only concerned with techniques for deploying multicast services in DiffServ networks with minimum impact but it also proposed sophisticated off-line TE solutions for optimising network resources in the management plane according to the expected traffic demand. An overview of our framework is presented next in section 1.2.

1.2 Framework Overview

As we have mentioned, our major task is to achieve a scalable solution for multicast services with end-to-end QoS guarantees in a multi-service Internet. We first present a brief description of our
proposed functional architecture, and then describe the individual building blocks of the framework. It should be mentioned that there might exist multiple traffic engineering solutions based on the proposed architecture, and in this thesis we only present one possible solution (IP-based approach using PIM-SM routing without MPLS overlay), which will be presented in detail in chapter 3.

1.2.1 Functional Architecture

In general multicast services include three basic business entities: the content provider (sender), the multicast receiver set (group members) and the Internet network provider (INP). Figure 1-1 presents the framework snapshot of our proposed solution for QoS aware multicast services. From the INP side, the network operation for achieving DiffServ-aware multicast is viewed as three layers or planes: the Management plane, the Control plane and the Data plane.

The Management plane is concerned with off-line functionality, having the responsibility of network planning and dimensioning in long time scales (e.g., weekly or monthly). In our proposed framework, the only element in the management plane is Offline Multicast Traffic Engineering (OMTE), whose task is to perform long-term (re-)dimensioning of network resources for optimal delivery of multicast traffic from the content provider to all group members.

Compared to the management plane, the control plane deals with relatively more dynamic behaviour of external multicast sources and group members, as well as routing semantics. As shown in the figure, this plane includes Multicast Sender Access Control (MSAC), Dynamic Multicast Routing (DMR) and Dynamic Group Management (DGM). The cooperation of these three blocks provides sophisticated control mechanisms for network resource optimisation.

The data plane is responsible for per packet treatment. Typically, in order to provide differentiated services to heterogeneous group members, priority-based forwarding queues are implemented for different QoS treatment.

The interaction between the three planes is that, functional blocks in the higher-level plane provide guidelines on the behaviour of those blocks in the lower-level plane. For example, Offline Multicast Traffic Engineering (OMTE) provides instructions on path selections by the dynamic multicast tree construction (i.e., DMR), and Multicast Sender Access Control (MSAC) enforces authentication mechanisms on Multicast Forwarding (MF) in the data plane.
1.2.2 Building Block Functionality

The content provider side contains two functional blocks that interact with the associated blocks in the INP part. The multicast Service Level Specification (mSLS) ordering is the process through which the content provider purchases network resources from the INP for the deployment of its multicast services. This mSLS is in effect a type of contract between the two parties about the usage of network resources owned by the INP. Detailed specification of mSLS will be presented in section 3.2. In order to achieve efficient resource allocation to external group sessions, the INP needs to obtain necessary parameters from mSLSs with the content provider (e.g., ingress/egress routers, QoS requirements, etc.), and feed them into the OMTE process. Once the network has been dimensioned, multicast end users (group members) start to join their subscribed groups, triggering multicast traffic to be injected into the network and delivered along the engineered multicast tree. This process indicates the activation of the signed mSLS, and we name it mSLS invocation.

At the INP side, the task of the Offline Multicast Traffic Engineering (OMTE) block is to optimally map the requested multicast flows onto the physical network resources and configure these resources in order to accommodate the forecasted multicast traffic (obtained from mSLS and monitoring data) injected by external sources. This type of network optimisation according to the signed mSLSs with external sources is generally performed in relatively long time scales e.g. every week or month. This periodicity is known as the Resource Provisioning Cycle (RPC).
Apart from the network resource optimisation, we also envisage the importance of protecting these resources from Denial-of-Service (DoS) attacks by malicious hosts, as it has been specifically pointed out in [3]. Taking this issue into consideration, we introduce the *Multicast Sender Access Control (MSAC)* block, since relevant issues have not received significant attention from the research community. This block is responsible for authentication of external sources’ behaviour, so that senders without a valid mSLS are not authorised to inject data into the network. The central component of the control plane is the *Dynamic Multicast Routing (DMR)* block, which can be implemented through any underlying multicast routing protocol (e.g., PIM-SM). The third block in the control plane is *Dynamic Group Management (DGM)*. This block extends the current Internet Group management Protocol (IGMP) for dealing with DiffServ aware group membership reports at Designated Routers (DR) where heterogeneous group members are attached.

The data plane contains three blocks. The *PHB Enforcement* block is the mechanism of implementing DiffServ Per Hop Behaviour (PHB) for different QoS classes. The *Multicast Forwarding* and *RPF Checking* blocks are similar to their conventional counterparts in the current Internet, except that the forwarding behaviour should conform to relevant permission from the sender access control block.

At the group member side, *Multicast Service Subscription* is the process through which interested receivers purchase multicast content from the content provider, who will later negotiate a mSLS with the INP for multicast content delivery toward these end users. This process can be achieved through out-of-band mechanisms and it is outside the scope of this thesis. *Group Join/leaving* is the action of these receivers to start/terminate receiving multicast content during the activating period of the corresponding mSLS.

### 1.3 Novelty

Since our concern is how to provision DiffServ aware multicast services from the standpoint of the INP, the major contribution of this thesis is the novel design and implementation of individual building blocks contained in the INP part. Nevertheless, we also address the remainder blocks when their relevant blocks in the INP part are specified in order not to lose generality. On the other hand, it should be noted that not all the blocks in the INP part result from our work (e.g., RPF checking), but we still include them so as to present a picture of the integrated architecture without loss of generality. Basically, the major contribution from this thesis includes the design and implementation of:
Chapter 1. Introduction

(1) Offline multicast traffic engineering with QoS constraints in the management plane;
(2) Dynamic DiffServ-aware routing (both overlay and non-overlay approaches) with QoS extension of group management in the control plane;
(3) Scalable inter/intra-domain multicast sender access control solutions in the control plane.

Compared to most of the existing QoS-aware multicast schemes, our proposed architecture provides a systematic solution to the incremental deployment of multicast applications with differentiated services. First, we push the complexity of QoS-aware routing optimisation to the management plane so that the network layer is kept as simple as possible. Moreover, we adopt underlying routing protocols such as PIM-SM for dynamic multicast tree construction, thus avoiding introducing a new QoS-aware protocol that cannot be supported by current IP routers [18, 20, 80]. In effect, although this proposed framework can be implemented through various approaches, we do not propose solutions based on explicit routing techniques such as Multi-protocol Label Switching (MPLS) [63] (see chapter 3). The advantage of this strategy is obvious: (1) Label Switching Path (LSP) scalability will not become an issue in multicast routing optimisation; (2) Efficient traffic engineering can be achieved using legacy IP routing and forwarding elements in the Internet. As far as we know, our work represents the first attempt to achieve QoS-aware multicast traffic engineering based on pure IP routing protocols. In addition to the basic hop-by-hop based PIM-SM routing, we also design and implement a seamless overlay scheme named Differentiated QoS Multicast (DQM) for supporting DiffServ aware multicast routing, specifically targeting bandwidth conservation as well as alleviating QoS states at core routers.

Another novelty of our work is the proposed multicast sender access control solution. We argue that effective prevention of DoS attacks is indispensable for the protection of precious network resources for authorised customers with QoS demands. From this point of view, the MSAC block is in effect a compensational component of the QoS-aware multicast architecture based on the open ASM service model. In this thesis we introduce both intra- and inter-domain multicast sender access control mechanisms, so that valid multicast data flows with QoS demands can be protected even if they are travelling across multiple domains.

1.4 Thesis Structure

This thesis is organised as follows. Chapter 1 presents the background and motivation for our work and a basic description of our contributions. Chapter 2 includes a comprehensive literature review on relevant research works. Chapter 3 is dedicated to the proposed offline multicast traffic engineering scheme assuming only hop-by-hop IP routing. In Chapters 4 and 5 we present
dynamic DiffServ-aware PIM-SM routing and group management, as well as an overlay-based approach named Differentiated QoS Multicast (DQM). In Chapter 6 we present our proposed intra- and inter-domain sender access control mechanism. We finally conclude the thesis and point to potential future research work in Chapter 7.
2 Literature Review

2.1 Multicast Service Models

2.1.1 IP Multicast

In the IP multicast model [26], sources send data packets to a logical IP address (known as class D address) ranging from 224.0.0.0 to 239.255.255.255. If any end host wants to receive the multicast traffic, it should obtain this specific class D address and become a group member. As we have previously mentioned, IP multicast is an open group model and its group management has the following characteristics: (1) A group may have multiple sources that only need to know the address of the group but have no idea of individual group members, i.e., receivers are anonymous to the information source. (2) A sender does not necessarily need to become a member of the group in order to send traffic to other group members. (3) Sources do not interact with each other, e.g., a source cannot prevent another one from sending traffic to the group and there are no priorities among sources in their sending behaviour.

Multicast routing protocols typically build multicast trees for delivering data from one or more sources to all the group members. Multicast trees can be classified into two categories: source specific trees and shared trees. The major different between the two types of trees is the following: each source specific tree contains only a single source that is normally the root of the tree while in shared trees multiple external sources can use the single tree to send data to all the group members.

2.1.2 Source Specific Multicast (SSM)

Due to the complex architecture of the traditional IP multicast service model and also the fact that many multicast applications are based on one-to-many communication, Source Specific Multicast (SSM [9]) has been proposed as a much simpler and manageable paradigm that can be deployed successfully on the Internet in the near future. In this service model, each group has a unique and well-known information source, while group members (called subscribers) can receive the information from the sender by subscribing to the associated channel. This type of service model
has addressed the nature of many applications that people are currently most interested in, e.g., Internet TV/radio, file distribution, etc.

In SSM, the traditional multicast group is substituted with a multicast channel identified by a tuple \((S, G)\), where \(S\) is the IP address of the unique information source and \(G\) is the class-D channel destination address within the range of 232/8. A single multicast distribution tree is constructed rooted at the source whose address has already been obtained by all the potential subscribers. In order to subscribe to the SSM service, end users should directly send explicit join requests along the reverse path back to the source \(S\), even if it is located in remote domains. In this scenario, the problem of inter-domain source discovery in the ASM model is successfully avoided even without the aid of MSDP. Another advantage of SSM is that since the group address is assembled with both source address and channel destination address, the whole range of the assigned address space can be used for each source, resulting in up to \(2^{24}\) available channels per sender. Since class D addresses are locally administered at each particular source, collisions will not take place even if two or more independent senders use exactly the same class D address. This is because each multicast session is not only identified by the channel destination address but also by the source address.

### 2.2 Multicast Group Management

At the receiver side, end hosts communicate with the Designated Router (DR) through the Internet Group Management Protocol (IGMP) in order to join/leave the group. In IGMPv2 [33], DRs attached to a LAN keep listening to any group membership report from end hosts on the same sub-network. Upon hearing an IGMP report with a new group address, the DR will trigger the underlying multicast routing protocol (e.g., PIM-SM) for grafting itself onto the associated multicast tree, so that the new group member is able to receive the group data packets via the DR. It should be noted that IGMPv2 allows a host to specify only the group address it is interested in receiving, which means that the traffic from all external sources to this group is received regardless whether the group member is interested in the data from all the specific sources.

IGMPv3 [17] has been developed as an SSM-aware protocol, but it is still backwards compatible with the conventional IP multicast. The distinct new feature of IGMPv3 is source filtering functionality by including two new modes: include() and exclude(). In the SSM scenario, the group membership report should include specific source addresses, and the group address should be within 232.0.0.0/8, so that the source-specific group join is triggered.
2.3 Multicast Routing

2.3.1 PIM-SM Routing

(1) Conventional PIM-SM

Dense mode routing protocols such as Distance Vector Multicast Routing Protocol (DVMRP) [75] and PIM Dense Mode (PIM-DM) [1], are based on flooding of multicast packets throughout the network, which suffers from scalability problems, especially when inter-domain multicast routing is considered. In contrast, PIM-SM is a receiver-oriented protocol, where each group member explicitly joins the existing delivery tree via a specific path. To achieve this, every multicast group is associated with a Rendezvous Point (RP), which can be regarded as a merging point of senders and receivers. When a receiver wants to join a multicast group G, it issues an IGMP membership report to its directly attached Designated Router (DR), and the DR will send a (*, G) PIM-SM join request hop-by-hop back towards the RP of the group. At the source side, all senders for this group simply encapsulate their multicast data and unicast the packets towards the RP (after performing the registration), from where the multicast packets are decapsulated and disseminated to all the group members. Once the DR has received multicast packets from the RP, it may choose to switch from the current shared RP tree to source specific trees by re-directing its join requests away from the RP towards individual sources. This type of source-specific join contains the address tuple (S, G) instead of (*, G), where S is the address of the individual source.

In PIM-SM, multicast routers utilise the underlying unicast routing table to perform Reverse Path Forwarding (RPF), which checks whether or not an interface is closest to the root (source or RP) of the tree. If a multicast packet is not received from the interface on the shortest path with which unicast traffic is delivered back to the source, it is then discarded for avoiding traffic loops. On the other hand, PIM-SM does not rely on any specific unicast routing protocol for RPF checking. This means that any underlying unicast routing table can be directly used as a reference to decide the shortest path for PIM-SM routing.

(2) PIM-SM adaptations

When PIM-SM is used in Source Specific Multicast (SSM) services, any (*, G) join packet must be suppressed, and individual (S, G) join requests should always be delivered to the source S explicitly. Moreover, the group address range of the group address G should be within 232.0.0.0/8, otherwise an error message will be triggered.

Bi-directional PIM (Bidir-PIM) is another mode of PIM-SM, specifically for group communications with multiple sources. In this adapted protocol, data packets are delivered along a bi-directional shared tree to the RP of the group. Bidir-PIM does not keep (S, G) group state,
and this aspect reduces the overall overhead in state maintenance within the domain, as only (*, G) group states are recorded at intermediate routers.

2.3.2 Inter-domain Multicast Routing

In the current IP multicast service model, PIM-SM itself does not have the functionality for enabling multicast packets from a source in one domain to reach a receiver in another. The PIM-SM DR for the source is only registered to the local RP, while the PIM-SM DR for the receivers send join requests towards the RP in their own domains. In this case, a mechanism is required for one RP to know about the existence of the sources if they are not located in the same PIM-SM domain.

(1) Multicast Source Discovery Protocol (MSDP)

To enable inter-domain multicast routing, Multicast Source Discovery Protocol (MSDP) [31] has been proposed and implemented over the Internet. As the name indicates, MSDP provides a mechanism to connect multiple PIM-SM domains so that RPs can exchange information on active sources of which they are aware. The basic operation of inter-domain multicast routing with the aid of MSDP is as follows. An MSDP RP constructs a Source Active (SA) message each time it receives a PIM-SM registration from a new local source, and it then sends the SA message to notify the RPs in its neighbour domains. When an RP receives from its external peer an SA message for a group for which interested receivers exist, it delivers the data down the shared tree to all the group members in its local domain. Once the receiver’s DR knows about the address of the remote source from the data packets on the RP tree, it performs an explicit join towards the specific remote source. In the Source Specific Multicast (SSM) service model, MSDP is not used, as designated routers already know about the existence of individual group sources even if they are located in foreign domains.

It should be noted that MSDP is NOT a multicast routing protocol, as it is not responsible for constructing inter-domain multicast trees. In effect, the routing task is still fulfilled by PIM-SM, which utilises the underlying Interior Gateway Protocol (e.g., OSPF/IS-IS) and inter-domain routing protocol (e.g., Multi-Protocol BGP [8]) for building the delivery tree across multiple domains.

(2) Multi-Protocol BGP (MBGP)

Currently Multi-Protocol BGP can be regarded as the de facto protocol for inter-domain multicast routing. The original task of MBGP is to advertise domain level Network Layer Reachability Information (NLRI) for non-IPv4 protocols and the address formats other than those of IPv4 unicast addresses. Compared to the conventional BGP, MBGP introduces two new attributes: MP_REACH_NLRI and MP_UNREACH_NLRI in UPDATE messages to advertise reachability
for non-IPv4 unicast traffic. To enable dedicated inter-domain multicast routing, MBGP is able to carry incongruent routes for unicast and multicast route by using different Subsequent Address Family Identifiers (SAFIs) in the attributes of MP_(UN)REACH_NLRI. According to [8], SAFI = 2 is dedicated to the advertisement of source addresses in IPv4 multicast routing. It should be noted that MBGP is not designed for constructing inter-domain multicast trees (still PIM-SM does this), but it provides domain level topology information for inter-domain PIM-SM group joins towards sources in other ASs.

(3) Border Gateway Multicast Protocol (BGMP) and Multicast Address Set Claim (MASC)

Border Gateway Multicast Protocol (BGMP) [73] has also been proposed as a long-term solution to inter-domain multicast routing. BGMP requires that each multicast group is associated with a single root domain. This is achieved through Multicast Address Set Claim Protocol (MASC), which is responsible for allocating multicast group addresses to a specific domain or Autonomous System (AS) over the Internet. From this point of view, BGMP and MASC work together to provide a mechanism for inter-domain multicast services. The group address distribution information is stored in the G-RIB table of each domain. By using the information in the populated G-RIB in individual domains, BGMP builds inter-domain bi-directional shared trees for active groups and then enables each domain to build source specific branches once a more optimal route has been explored.

2.3.3 QoS-aware Multicast Routing

Shortest path routing, though simple in implementation, does not always produce optimal performance in construction of multicast trees. In the literature, optimisation of multicast routing is generally formulated as the Steiner tree problem that has been proved to be NP-complete [36]. Unconstrained Steiner tree heuristics can be used to solve the problem of minimising overall tree cost [46, 72]. However, these heuristic algorithms do not attempt to cope with tree optimisation with end-to-end constraints, and hence they may not be well suited for multicast routing with such requirements. Nevertheless, there exist numerous research works that deal with a QoS-constrained Steiner tree problem, using QoS metrics such as delay, delay variation, bandwidth etc. Table 2-1 presents a brief summary on some classical heuristic algorithms for both constrained and unconstrained Steiner tree problems.
Chapter 2. Literature Review

<table>
<thead>
<tr>
<th>Name</th>
<th>Constraint(s)</th>
<th>Time Complexity</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>KMB [46]</td>
<td>None</td>
<td>$O(mn^2)$</td>
<td>L. Kou et al</td>
</tr>
<tr>
<td>TM [72]</td>
<td>None</td>
<td>$O(mn^2)$</td>
<td>A. Takahashi et al</td>
</tr>
<tr>
<td>KPP [43]</td>
<td>Delay</td>
<td>$O(mn^2)$</td>
<td>V. P. Kompella et al</td>
</tr>
<tr>
<td>BSMA [83]</td>
<td>Delay</td>
<td>$O(kn^2 \log(n))$</td>
<td>Q. Zhu et al</td>
</tr>
<tr>
<td>DVMA [64]</td>
<td>Delay &amp; Delay Variation</td>
<td>$O(klmn^4)$</td>
<td>G. Rouskas et al</td>
</tr>
<tr>
<td>GTM [49]</td>
<td>Bandwidth</td>
<td>$O(m^2n^3)$</td>
<td>C. P. Low et al</td>
</tr>
<tr>
<td>DCGMRP [50]</td>
<td>Delay &amp; Bandwidth</td>
<td>$O(TBm^2n)$</td>
<td>C. P. Low et al</td>
</tr>
</tbody>
</table>

Table 2-1 Summary of constrained and unconstrained Steiner tree solutions

In addition to the aforementioned centralised heuristic algorithms, there exists another category of distributed QoS-aware multicast routing protocols in the literature. In Yet Another Multicast (YAM) [18], when a new router intends to join a multicast tree with QoS requirements, it first performs a local search through flooding bid-order packets with controlled Time-to-live (TTL). Each on-tree router receiving the bid-order packet becomes a candidate node and returns a bidding message as an acknowledgement. On receiving the bidding messages from all the candidates, the new router will select one on-tree router with the best-offered QoS for group join. Banerjea et al extended YAM and proposed the QoS MIC protocol [80] in which both local search with bidding and multicast tree search are used to locate feasible join routes. Specifically, while performing local search, the joining router also sends a multicast join message to a tree manager that has full knowledge of the network and group membership. Finally, with possible aid from the tree manager, the new joining router may be able to find feasible path with desired end-to-end QoS demands. B. Yang et al proposed Multicast QoS (MQ) [82], as an integrated framework considering QoS routing, resource reservation and user heterogeneity. This genuine receiver-initiated approach inherits some basic characteristics of RSVP [15], such as quantitative QoS guarantees and recourse reservation merging from heterogeneous end users. Detailed description of MQ will be presented in Chapter 5, for comparison with our proposed overlay Differentiated QoS Multicast (DQM) scheme. QoS-aware multicast routing protocols also include QoS Multicast Routing Protocol (QMRP) [20], which is not considered in this thesis.
2.4 Differentiated Services

2.4.1 DiffServ Architecture

The Integrated Services (IntServ) model [14] was the first step towards supporting end-to-end QoS in the Internet. However, since it requires that each router keeps per-flow state, the Integrated Services model suffers from scalability problems in terms of both memory overhead and queue maintenance.

In the late 90's, IETF proposed the Differentiated Service (DiffServ) model [11] as an alternative solution to provide Internet QoS. The DiffServ architecture is currently seen as a promising solution for service differentiation in a large scale due to the fact that the core network is kept relatively simple, with most complexity confined at the network edge and the management plane. Admission control and traffic conditioning are performed at edge routers, while core routers simply treating traffic aggregates on a Per Hop Behaviour (PHB) basis according to the Differentiated Services Code Point (DSCP) in the packet header. Normally there exists a central agent known as Bandwidth Broker (BB) within each DiffServ domain, whose functionality is to intelligently manage bandwidth resources for individual transit traffic aggregates. When a customer wishes to receive Differentiated Services, s/he should first negotiate a Service Level Agreement (SLA) with the ISP, typically through the Bandwidth Broker, for specifying the packet treatment within the network. A detailed technical description of a SLA is known as Service Level Specification (SLS).

![Figure 2-1 Overview of DiffServ network](image-url)
As we have mentioned, the functionality of DiffServ edge routers is much more complicated than that of core routers, whose responsibility is to forward traffic aggregates according to the DSCP value being carried in the packet header. For DiffServ edge routers, traffic conditioning and admission control are the essential tasks. First, an edge router uses a classifier to identify the service class that should be given to the incoming traffic. Once classified, the traffic is submitted to the Meter, which measures the traffic to verify conformance to the agreed traffic profile. Thereafter, a Marker may perform marking on unmarked packets (DSCP=0000000) or even remarking on marked ones according to the result of the measurement. Finally, the Shaper/Dropper treats the marked packets according to their DSCP value so that the traffic is compliant with the traffic profile. The functionality of DiffServ edge routers is illustrated in Figure 2-2.

![Figure 2-2 DiffServ edge router functionality](image)

### 2.4.2 Multicast in DiffServ Networks

(1) The NRS Problem

The issue of integrating multicast services in DiffServ networks was first addressed in [12] (evolved from an early internet draft), where the Neglected Reservation Sub-tree problem (NRS) was specifically proposed and discussed. The authors found that in a DiffServ-aware multicast environment, it is possible that the actual resources consumed by the multicast traffic may exceed the pre-negotiated SLA. Since the multicast tree could branch at any core router with more traffic being generated, the amount of outgoing traffic from a domain may exceed the incoming traffic and thus consume additional resources. One typical scenario is illustrated in Figure 2-3, where egress router E2 joins the group (rooted at router H) without traffic conditioning at the edge of the DiffServ domain. As the new multicast tree branch is created from the core router to E2, the QoS treatment of existing traffic flowing along the path will be affected. According to [12], the NRS
problem can be solved by assigning a Lower than Best Effort (LBE) PHB to the newly branched traffic. In order to obtain higher grade of service, the joining node has to explicitly negotiate with the Bandwidth Broker (BB) for resource reservation. In case that the BB can allocate available bandwidth to the new branch, the new group member will receive the traffic based on its originally desired QoS class. Otherwise this branch has to remain in the LBE service class.

It is worth mentioning that the NRS problem can often occur if no proper resource provisioning is performed for multicast traffic, and this is exactly what is missing in the current DiffServ-aware multicast paradigms. We argue that NRS can be avoided successfully as long as network resources are optimally dimensioned in the management plane (e.g., offline multicast traffic engineering), and this is one of the major objectives of our work.

![Figure 2-3 The NRS Problem](image)

(2) DSMCast

DSMCast [70, 71] is a scalable framework that aims at completely stateless multicast in DiffServ networks. The main idea of DSMCast is that both the destination address of individual receivers and their QoS requests are embedded in the header of group data packets, other than being maintained within the DiffServ domain. Packets are replicated where necessary at core routers and delivered to individual receivers based on their unicast destination address contained in the packet header. In this sense, DSMCast does not make use of class D address as in the traditional IP multicast service model. During the replication procedure, DSCP values are also remarked according to the QoS requirements of individual downstream group members. In this scenario, core routers need maintain neither QoS states nor multicast group states, and this characteristic guarantees high scalability. On the other hand, DSMCast aborts the traditional IP multicast service model that has already been very popular throughout the Internet. Moreover, in case of large number of egress routers or receivers, DSMCast data transmission becomes inefficient due
to the relatively longer packet header that contains individual receiver's unicast address and their desired QoS classes.

(3) QUASIMODO

QUASIMODO [10] is a control plane DiffServ multicast framework based on the IP multicast service model. The objective is to (A) provide flexible QoS support with respect to heterogeneous multicast groups, and (B) maintain compatibility with currently deployed multicast protocols.

In QUASIMODO, PIM-SM is selected as the reference multicast routing protocol. In order to accommodate QoS heterogeneity, DiffServ extensions have been made to PIM-SM join requests and the multicast forwarding table inside core routers. First, if a potential member decides to join the group with a certain QoS level, it should send out an adapted IGMP report (*, G, q) where q indicates the DiffServ service class this receiver desires to receive. Once the Designated Router receives the report, it will issue a (*, G, q) join request towards the RP, and this join request will explore a new tree branch that satisfies the requested QoS class.

On the other hand, in order to handle join requests with heterogeneous QoS demand, the multicast forwarding table inside core routers also needs to be extended accordingly. Specifically, the outgoing interface (oif) field of each group is appended with an additional DSCP entry, which is used to mark replicated packets that are forwarded on this particular outgoing interface. Figure 2-4 presents a typical structure of a DiffServ-aware multicast forwarding table in the QUASIMODO approach.

<table>
<thead>
<tr>
<th>Group address</th>
<th>iif</th>
<th>oif</th>
<th>DSCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>226.187.38.5</td>
<td>A</td>
<td>B</td>
<td>AF11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C</td>
<td>AF21</td>
</tr>
<tr>
<td>237.22.98.160</td>
<td>D</td>
<td>A</td>
<td>FF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C</td>
<td>BE</td>
</tr>
</tbody>
</table>

Figure 2-4 QUASIMODO multicast forwarding table

The routing dynamics of a particular group in QUASIMODO is basically how to update the oif list as well as its associated DSCP filed in the multicast forwarding table according to the received join requests with various QoS requirements. There are basically three scenarios when a core router receives a group G join request:

- The interface is not in the oif list of G. In this case the router will include this interface into the oif list, and record the desired QoS class carried in the join request to the DSCP field of
the forwarding entry. If this core router is not included in the distribution tree, it will forward the join request towards the RP or sources.

- The interface is in the oif list of group G and has equal or higher QoS class state than that indicated in the join request. In this case, the core router need not take any action.

- The interface is in the oif list of G but has lower QoS class state than the one indicated in the join request. In this case the core router will upgrade the DSCP value associated with this oif with the one that is carried in the newly arrived join request. Meanwhile a new join request with higher QoS class will be sent towards the RP or source, so that the QoS requirement of the new downstream member can be satisfied.

When group data is received on the incoming interface (iif), the core router replicates the packet and forwards its copies on all its interfaces in the oif list. The forwarding behaviour on each outgoing interface is uniquely based on the corresponding DSCP field in the group forwarding entry.

### 2.5 Traffic Engineering (TE)

Traffic Engineering (TE) is deemed an effective approach for supporting end-to-end QoS requirements, thanks to its power in improving the service capability of operational IP networks. In [4], TE is defined as large-scale network engineering for dealing with IP network performance evaluation and optimisation. The key task of TE is to enhance network performance while at the same time optimising resource utilisation. Based on the implementation method, traffic engineering approaches can be classified into Multi-Protocol Label Switching (MPLS)-based and pure IP-based.

#### 2.5.1 MPLS Based TE

Multi-Protocol Label Switching (MPLS) is an Internet Engineering Task Force (IETF) specified forwarding scheme. In MPLS, traffic delivery occurs on Label Switched Path (LSPs). An LSP is the path between ingress label switching routers (LSRs) and egress LSRs which a labelled packet traverses. At the boundary of an MPLS domain, LSRs classify IP packets into Forwarding Equivalence Classes (FECs) and append different labels for packet forwarding within the MPLS domain. The Label Distribution Protocol (LDP) is used to distribute label bindings during the setting-up of an LSP.

MPLS is a powerful technology for Internet traffic engineering, as it allows traffic to be directed through an arbitrary explicit route, which may not necessarily follow the shortest path, as it is the case in the current IP routing semantics. With MPLS-based TE, packets are encapsulated with
labels at ingress points, which are then used to forward these packets along a TE-aware explicit LSP. Typically, individual flows are aggregated into traffic trunks identified by FECs, which are then carried with LSPs between ingress and egress routers. In this case, the conventional shortest path based routing infrastructure (e.g., OSPF) is overridden with a tunneled MPLS explicit routing overlay. In order to support traffic-engineered explicit routing of these flow aggregates, two types of signalling protocols can be used for setting up and tearing down LSPs, namely RSVP-TE [5] and CR-LDP [41]. RSVP-TE is a soft-state signalling protocol that uses the RESV and PATH messages in Resource reservation Protocol (RSVP) [15] for a two-stage process in setting up LSPs. CR-LDP is a hard-state signalling protocol that runs over TCP and uses the Label distribution Protocol (LDP) [2] REQUEST and RESPONSE messages for setting up traffic-engineered paths. In order to disseminate TE information (e.g., reservable bandwidth) so that all nodes in the network have a consistent view of the associated traffic-engineering parameters, TE-extensions to OSPF (OSPF-TE) [42] and ISIS (ISIS-TE) [68] have been proposed to carry TE-aware link state advertisement for helping setting up traffic engineered LSPs.

In the literature, there have been numerous research works focusing on MPLS traffic engineering for unicast traffic. In [44], an online TE scheme was proposed for dynamic routing of individual traffic trunks without having a priori knowledge of future traffic demands. The major contribution of this piece of work is an efficient algorithm of routing traffic aggregates with minimum interference at some critical links. In the EU IST TEQUILA project [74], a sophisticated framework was proposed covering both management plane and control plane for MPLS based TE in IP Differentiated Services networks. For bandwidth conservation purposes, the TEQUILA approach adopts the Hose model for setting up point-to-multipoint LSPs, so that the optimisation task is formulated into a Steiner tree problem with QoS constraints such as end-to-end delay and packet loss.

Despite the progress for unicast services, traffic engineering for multicast flows remains largely a dark area till now. In the past few years, MPLS-based multicast TE has become a subject of interest, with a number of relevant research works becoming available. In [81], Steiner tree based heuristics are applied for performing multicast path selection only at the edge of MPLS domains, so that multicast TE within the network can be reduced to a unicast problem. In [45], the authors extended their TE scheme for unicast traffic [44], and proposed an online multicast TE scheme using Steiner tree heuristics, which also addresses the issue of minimising multicast flow interferences.
2.5.2 IP Based TE

Recently, the advent of pure IP-based TE solutions challenges MPLS-based approaches in that Internet traffic can also be effectively tuned through native hop-by-hop routing, without the associated complexity and cost of MPLS. In [77], the authors proved that any arbitrary set of loop-free routes can be represented into shortest paths with respect to a set of positive link weights, and [62] presented further analysis on the relevant issues in shortest path representability. The contributions from these works are of great significance since they indicate the feasibility of reducing general routing optimisation into shortest path based paradigms that can be directly handled by the underlying IP routers. As a typical application, B. Fortz and M. Thorup claimed that by optimising OSPF/IS-IS link weights for the purpose of load balancing, link congestion can be effectively avoided for unicast services [34]. The key idea of the proposed algorithm is to intelligently adjust the weight of a certain number of links that depart from one particular node, so that new paths with equal cost are created from this node towards the destination. As a result, the traffic originally travelling through one single path can be evenly split into other paths with equal OSPF/IS-IS weights. Figure 2-5 provides a simple illustration on the basic mechanism of the algorithm. Consider a destination node t and assume part of traffic demand going to t travels through an intermediate node x. The Fortz and Thorup strategy is to split the flow to t going through x evenly along all the links \((x, x_i)\) from x, if these links \((x, x_i)\) belong to the shortest path from x to t. This type of "local adjustment" needs special attention, since shifting traffic might incur additional congestion to other links. In order to avoid this oscillation phenomenon, the authors apply sophisticated Tabu search for obtaining best performance in load balancing.

In [69], the authors proposed a new scheme based on the manipulation of a set of next hops for routing prefixes, which is capable of achieving near optimal traffic distribution without any change of existing routing protocols and forwarding mechanisms. Three different heuristic algorithms were studied for optimally configuring the next hop of unicast destination prefixes.
Chapter 2. Literature Review

This approach is a typical strategy of making graceful trade-off between the performance and the overhead associated with the additional configuration needed.
Chapter 3

3 Offline Multicast Traffic Engineering (OMTE)

3.1 Introduction

As we have mentioned previously, the task of Offline Multicast Traffic Engineering (OMTE) is to map optimally the demanded multicast flows onto the physical network resources, and configure these resources in order to maximise the network service capability for external multicast sessions with QoS requirements. In the literature, it has been widely reckoned that end-to-end QoS should not be achieved at the expense of introducing significant overhead to the routing/signalling infrastructure, especially in the core network. Towards this end, our strategy is to push the network optimisation complexity to the management plane, which can be regarded as the complementary part to the conventional DiffServ and IP multicast control and data planes. From an implementation point of view, MPLS has become an attractive paradigm for the enforcement of TE-aware explicit routing, but INPs are always reluctant to deploy it at large scale for scalability and cost reasons. On the other hand, plain IP based TE solutions have also been available as it has been realised that Internet traffic can be effectively tuned through OSPF/ISIS link weight optimisation. The most notable approach is that of [34], which claims 50-110% efficiency improvement in comparison to simply setting link weights inversely proportionally to link capacity, as is the common practice today. In effect, through this approach not only unicast but also multicast traffic can be adjusted through modification of the underlying IGP link weights (a simple illustration will be presented in section 3.3). Motivated from this, we design and implement an efficient scheme for multicast traffic engineering in a plain IP environment without MPLS overlays. Of course, this does not mean that MPLS-based algorithms cannot be applied to OMTE; relevant approaches will be addressed in our future research work.

The rest of this chapter is organised as follows. Section 3.2 provides a brief illustration of the multicast Service Level Specification, from which multicast traffic matrix can be derived as an input to OMTE resource provisioning. In sections 3.3 and 3.4 we introduce the basic operation of the OMTE functional block, and we describe how it interacts with the control plane blocks such as Dynamic Multicast Routing (DMR). In sections 3.5 and 3.6 we present the problem
formulation of the traffic engineering objectives and also present a Genetic Algorithm (GA) based optimisation scheme as a solution. Section 3.7 is dedicated to the analysis of the GA approach, including the optimisation process and the associated computing overhead. Finally in section 3.8 we conduct simulations for the evaluation of the proposed algorithm in terms of efficiency, and we compare it to both conventional non-TE aware paradigms and MPLS associated solutions using explicit routing techniques.

3.2 Multicast Service Level (mSLS) Specification

In this thesis we assume the following business scenario: First, interested end users need to subscribe to the multicast service offered by the content provider, and if necessary, they may also specify their end-to-end QoS requirements. In order for the content provider to deploy offered multicast services, he should sign a multicast Service Level Specification (mSLS) with the INP so as to have the multicast data delivered to all the receivers with proper QoS treatment within the network. This type of mSLS should include: (1) source/group address, (2) ingress router from where the multicast traffic is injected; (3) egress routers (i.e., DRs) where subscribers are attached; and (4) QoS demand. Formally, a basic mSLS entry list for a particular INP can be expressed as follows:

\[
\text{mSLS}_i = \{\text{Src addr, Grp addr, ingress, [egress 1, ..., egress n], QoS demand}\}
\]

Figure 3-1 describes the business interactions between each of the three entities. From a business point of view, multicast subscribers first make all-in-one payment to the content provider, including the cost of both multicast content and the associated traffic delivery in the INP’s network. During the phase of mSLS negotiation, the INP charges the content provider for the resource consumption of multicast data transmission. Therefore the content provider will allocate part of the income from its customers for this payment to the INP. This type of sender-oriented pricing and charging follows exactly in the same style as for current unicast services on the Internet. It is noted that the work to be presented in this thesis will be based on this mSLS relationship assumption.

Once mSLS negotiations are finished, multicast traffic demands are summarised from individual mSLSs, and a multicast traffic matrix is derived and fed into the OMTE process for resource optimisation. It is also worth mentioning that, apart from mSLSs, the multicast traffic matrix can be alternatively (or better, in addition) obtained from traffic measurement at edge routers of the
network. The above two mechanisms allows the INP to perform multicast traffic engineering based on the forecasted traffic matrix. After the network resources are optimally provisioned through multicast TE, multicast subscribers are notified of QoS availability, and allowed to perform group joins and leavings at the edge of the network, which identifies the invocation of the signed mSLSs. From this description we can see that an mSLS is in effect initiated by the content provider but is effectively activated at the receiver side.

![Multicast business relationship diagram](image)

**Figure 3-1 Multicast business relationship**

### 3.3 OMTE Overview

As has been indicated in [4], the key task of traffic engineering is to enhance network performance while at the same time optimising resource utilisation. As for multicast services, the major concern of OMTE is to conserve bandwidth resources and balance the traffic load for maximising the service capability of the network without causing network congestion.

In contrast to the progress for unicast services, traffic engineering for multicast flows remains largely a dark area. In the past few years, MPLS-based multicast TE has become a subject of interest, with a number of relevant research works becoming available. Despite their efficiency and flexibility in path selections, these MPLS based schemes also suffer from potential scalability problems in terms of LSP maintenance. Compared to the unicast scenario, *point-to-multipoint* LSPs are compulsory for multicast traffic delivery, and hence the scalability issue becomes more pronounced and should not be overlooked. Another special consideration for multicast MPLS is that, engineered LSPs might work well if receivers are statically bound to the group address, but unfortunately this is not the case given the high dynamics of group membership over the Internet.

---

1 In this thesis we use the term *subscribers* to identify end users who have subscribed to the multicast service offered by the content provider, and when subscribers join the associated groups, they become *group members*. We also use the term *receiver* in general.
On the other hand, plain IP-based multicast traffic engineering without MPLS overlay is definitely a great idea, but unfortunately this research area has not yet been explored. The reason for this can be summarised as follows. First, PIM-SM uses the underlying IP unicast routing table for the construction of multicast trees, and hence it is difficult to decouple multicast traffic engineering from its unicast counterpart. Bandwidth optimisation for multicast traffic can be formulated as the directed Steiner tree problem, which is NP-complete. The enforcement of Steiner trees can be achieved through packet encapsulation and explicit routing mechanisms such as MPLS tunnelling. However, this approach lacks support from hop-by-hop protocols, due to Reverse Path Forwarding (RPF) in the IP multicast routing protocol family. In PIM-SM, if multicast packets are not received on the shortest path through which unicast traffic is delivered back to the source, they are discarded in order to avoid traffic loops.

In this chapter we investigate the feasibility of engineering multicast traffic based on plain IP routing protocols. In a similar fashion to the existing MPLS-based paradigms, our objective is to optimise the overall network resource consumption with QoS constraints (e.g., bandwidth guarantees), in order to increase the service capability for external multicast groups without causing network congestion. The enforcement of engineered PIM-SM path selections is via setting optimised link weights for the underlying link state routing protocols. In our proposed approach, PIM-SM follows the shortest path according to the pre-set link weights, whereas the resulting multicast tree is in effect a hop-count Steiner tree with minimum number of links involved, which implies that minimum bandwidth resources are consumed. We demonstrate this with the simple example of Figure 3-2. We assume that node A is the root of group X that contains member nodes E, F and G. If PIM-SM performs hop-count based shortest path (SP) routing, the total bandwidth consumed is 6 units (1 unit for each on-tree link), as shown in Figure 3-2(a). In effect, by applying Steiner tree heuristics to this simple example, it is easy to obtain the optimised multicast tree with 4 units of bandwidth consumption, as shown in Figure 3-2(b). This hop-count Steiner tree can be supported using explicit routing approaches such as MPLS tunnels. For example, in order to deliver multicast packets from node A to E via the engineered path, an LSP tunnel has to be set up along the non-shortest path $A \rightarrow C \rightarrow F \rightarrow E$. On the other hand, we notice that, by intelligently assigning link weights for the underlying link-state IGP protocol, we can still achieve the same effect in terms of bandwidth conservation, as PIM-SM join requests follow the shortest path in terms of this set of link weights (Figure 3-2(c)). From this example we can see that hop-count Steiner tree based multicast traffic engineering can be reduced to plain shortest path routing by introducing a set of optimised link weights. The advantage is that, through link weight setting as calculated by off-line network provisioning, IP routers are able to construct optimised multicast trees by simply using Dijkstra's shortest path algorithm. On the other hand,
other TE metrics than hop-count can also be considered in this optimisation problem, but in this chapter we only use the hop-count metric for illustrating the bandwidth conservation aspect.

Currently, one difficulty in implementation of this scheme is that plain unicast routing protocols such as OSPF and IS-IS do not provide independent set of link weights for different types of flows. Hence it is undesirable to set link weights exclusively for multicast without considering unicast traffic in the network. In order to decouple multicast from unicast path selection, our approach is based on the multiple-topology-enabled IGP (MT-IGP), e.g., Multi-topology extension of the IS-IS protocol (M-ISIS) [61] and OSPF protocol (MT-OSPF) [60], which is able to populate dedicated Multicast Routing Information Bases (M-RIBs, i.e. RPF tables) for PIM-SM routing. This multi-topology routing feature provides a mechanism to separate TE for multicast and unicast flows. For the rest of the thesis we will use M-ISIS as a typical example for illustration. We should also mention that in reality both multicast and unicast flows use the same set of physical links within the network. In this thesis, we assume the following decoupled TE scenario: First, unicast traffic engineering should be performed based on its own TE objective (e.g., load balancing using the scheme proposed in [34]). After that, the bandwidth resources allocated for the unicast traffic should be deduced from the physical link capacity, and our proposed multicast traffic engineering solution is then based on the residual bandwidth.

The optimisation of link weights through shortest path routing for indirectly obtaining one single Steiner tree in terms of hop-counts is \textit{NP-complete}, since this is an adapted version of the classical Steiner tree problem. In effect, a more practical problem concerning an INP for multicast traffic engineering is how to assign a set of unified link weights, so that all the multicast trees within the network consume minimum bandwidth resources. At the same time, we also consider an additional constraint that the total bandwidth allocated on each link for the overlapping multicast trees should not exceed its capacity. In this thesis we adopt a Genetic Algorithm (GA) [23]
Chapter 3. Offline Multicast Traffic Engineering

approach as off-line multicast traffic engineering for optimising overall bandwidth consumption for multiple multicast flows. More specifically, the M-ISIS link weights are adjusted in each GA generation so that the overall fitness is geared towards optimised network resource consumption with the constraint of bandwidth capacity. The key novelty of this work is that, in a similar fashion to the work in [34] for unicast traffic, multicast flows can also be optimised through hop-by-hop routing without relying on explicit MPLS tunnelling. As far as we know, our proposed methodology represents the first attempt to explore effective solutions to multicast traffic engineering based on the hop-by-hop routing semantics.

Finally, it is worth mentioning that, although we only consider the bandwidth requirement in our problem formulation, other QoS metrics such as delay, delay variation and packet loss can also be easily introduced as additional constraints. In effect, bandwidth itself plays a key role in QoS provisioning: once the bandwidth resources are optimally dimensioned, large-scale queuing delays and packet loss can be minimised by eliminating overloaded links. Towards this end, the major task is to guarantee that external traffic demand is efficiently mapped onto the network resources, so that ideally the accumulated bandwidth consumption on each link does not exceed its capacity. On the other hand, over-provisioning is not a desired solution since the INP must ensure efficient utilisation of its network resources to achieve maximum revenue. One common experience in coping with this dilemma is through delicate “under-provisioning” with over-admission of traffic demands from external multicast sources. In this scenario, despite the fact that some network links are estimated to be overloaded according to the forecasted traffic demand, the real-time service availability can still be maintained at a very high level when external mSLsSs are dynamically invoked (see section 4.6).

3.4 M-ISIS/PIM-SM based OMTE Framework

The traditional OSPF and IS-IS protocols only have a uni-dimensional viewpoint on the weight of each link in the network, and this influences path selections for both unicast and multicast traffic. In contrast, M-ISIS provides the original IS-IS protocol with the additional ability of viewing the weight of each link for different logical IP topologies independently. For multicast traffic, the Multi Topology identifier (MT-ID) field with value 3 in M-ISIS is dedicated to the multicast reverse path forwarding topology, i.e., the RPF table for PIM-SM can be populated using a set of independent link weights with MT-ID equal to 3. With this multi-topology capability, it becomes possible that PIM-SM based multicast routing is completely decoupled from the underlying routing table for unicast traffic.

Figure 3-3 illustrates the basic framework of IP multicast traffic engineering through optimised M-ISIS link weight setting. First, the network topology (e.g., link capacity, edge router
Chapter 3. Offline Multicast Traffic Engineering

information) and the multicast “traffic matrix” are obtained as the input parameters for calculating the optimised link weights over an existing physical network infrastructure. The multicast traffic matrix can be derived by obtaining the following information from each group session: (1) traffic demand, and (2) root node (i.e., ingress router) and a set of egress routers with subscribers. As we have specified in section 3.2, an INP can estimate the multicast traffic matrix through traffic measurement and from the mSLSes with content providers. Based on the multicast traffic matrix, the optimised link weights are computed through off-line algorithms and configured in the routers that run the M-ISIS routing protocol with MT-ID equal to 3, which is dedicated to the multicast RPF table construction. On receiving Link State Advertisements (LSAs), each M-ISIS aware router computes shortest path trees according to this set of link weights and decides the NEXT_HOP router for a specific IP address/prefix. In Figure 3-2(c), the NEXT_HOP router computed by $E$ towards $A$ points to $F$ instead of $B$, since the path $E \rightarrow F \rightarrow C \rightarrow A$ has the lowest cost of 3 according to the assigned M-ISIS link weights. When a PIM-SM join request is received, the router simply looks up the RPF table and finds the proper NEXT_HOP for forwarding the packet. In this scenario, the delivery of PIM-SM group join requests follows an engineered path, thus the resulting multicast distribution tree from the root to individual members conforms to the TE requirement. In addition, the multicast forwarding information base (M-FIB) is dynamically updated for the incoming interface (iif) and outgoing interface (oif) list of each group. We can see from Figure 3-3 that, apart from the offline calculation and setting of link weights, there is no need for any other configuration or extensions to the current M-ISIS and PIM-SM protocols for multicast traffic engineering purposes.

It is also worth mentioning that, since this infrastructure is based on off-line multicast traffic engineering, large scale M-ISIS link weight setting takes place only at a relatively long-term resource provisioning cycle (RPC, e.g., weekly or monthly). On the other hand, online link weight adjustment according to multicast traffic dynamics can be addressed in future research work.
3.5 Problem Formulation

The following is the integer-programming formulation of the problem of computing bandwidth-constrained Steiner trees in terms hop counts with the objective of minimising overall bandwidth consumption. By setting the group-specific binary variables $x_{i,j}^g$ and $y_{i,j}^g$ for each uni-directional link $(i, j)$, a set of explicit multicast trees with minimum number of links is obtained, which implies that minimum bandwidth consumption is achieved. We first present some definitions below:

- $G$ — Total number of subscribed multicast groups;
- $r_g$ — Root node of group $g$ ($g = 1..G$);
- $V_g$ — Multicast subscriber set for group $g$ (for simplicity, we assume one subscriber per DR for each group);
- $T_g$ — Multicast delivery tree with active members for group $g$;
- $D_g$ — Bandwidth demand for group $g$ traffic on each link;
- $C_{i,j}$ — Bandwidth capacity of link $(i, j)$;
- $y_{i,j}^g$ — Equal to 1 if uni-directional link $(i, j)$ is included in the multicast tree for group $g$;
$x_{ij}^{g,k}$ — Equal to 1 if uni-directional link $(i, j)$ is on the multicast tree branch from the root node $r_g$ of group $g$ to the receiver node $k$ in the multicast tree;

The integer-programming problem of computing a set of bandwidth constrained Steiner trees with minimum overall bandwidth consumption is formulated as:

Minimise

$$\sum_{g=1}^{G} \sum_{(i,j)\in E} D_g \times y_{ij}^g$$

Subject to

$$\sum_{i\in V} x_{ih}^{g,k} - \sum_{j\in V} x_{ji}^{g,k} = \begin{cases} 1 & i = r_g \\ -1 & i = k, \ k \in V_g \\ 0 & i \neq r_g, i \notin V_g \end{cases} \quad (3.1)$$

$$x_{ij}^{g,k} \leq y_{ij}^g \quad (i, j) \in E, k \in V_g \quad (3.2)$$

$$x_{ij}^{g,k} = 0, 1 \quad (i, j) \in E, k \in V_g \quad (3.3)$$

$$y_{ij}^g = 0, 1 \quad (i, j) \in E \quad (3.4)$$

$$\sum_{g=1}^{G} y_{ij}^g \times D_g \leq C_{ij} \quad (i, j) \in E \quad (3.5)$$

The variables to be determined are $x_{ij}^{g,k}$ and $y_{ij}^g$ for every link $(i,j) \in E$. Constraint (3.1) ensures one unit of multicast flow from $r_g$ to every receiver $k \in V_g$. Constraint (3.2) guarantees that the amount of flows along link $(i,j)$ must be zero if this link is not included in the multicast tree for group $g$. $x_{ij}^{g,k}$ and $y_{ij}^g$ are confined to zero-one variables in constraints (3.3) and (3.4) for non-splitting of multicast flows. Finally it is required in constraint (3.5) that the total bandwidth consumption on each link should not exceed its capacity. Apart from this bandwidth requirement, other QoS constraints can be introduced in the above problem formulation, such as end-to-end delay, packet loss, etc. In this thesis we are only concerned with the bandwidth-constrained OMTE problem, but the proposed solution can still be adapted to include other QoS metrics.

As we have mentioned before, the enforcement of the above set of bandwidth-constrained hop-count Steiner trees can be achieved through an explicit routing overlay, e.g. through MPLS, on a
per group basis. However, the paths in the Steiner tree from $r$ to individual receiver $k \in V_g$ might not completely overlap with the shortest paths between them. This means that, in case of hop-by-hop routing, multicast traffic flowing on the Steiner tree will be discarded due to the network layer RPF check failure, if the packets are not received from the correct interface on the shortest path back to the source. In order to apply the above programming model to IP layer solutions, we introduce a unified M-ISIS link weight $w_i$ for each link $(i, j)$, and by properly setting those link weights it is guaranteed that the tree branch from $r$ to each subscriber $k \in V_g$ is the shortest path according to this set of weights. Put in other words, our strategy is to represent this set of explicit hop-count Steiner trees with shortest path trees through intelligent configuration of a unified set of link weights. Formally, the problem is to calculate a vector of link weights $W = \{w_i\} : w_i > 0$, so that for each optimised multicast tree $T_g$ ($g = 1..G$) with bandwidth constraints the following inequality holds:

For any on-tree path $P_{r_g \rightarrow k} \subseteq T_g$ (i.e., for each link $(i, j) \in P_{r_g \rightarrow k}$, $x_{ij} = 1$), $\forall P'_{r_g \rightarrow k} \not\subseteq T_g$

$$W(P_{r_g \rightarrow k}) \leq W(P'_{r_g \rightarrow k})$$

where $k \in V_g$.

According to [62], it is NP-hard to decide one set of unified link weights for converting an arbitrary group of explicit routes into shortest paths. This conclusion gives one of the reasons why MPLS explicit routing is able to outperform IP link-weight-setting based approaches, which lack flexibility in path selection for an arbitrary set of flows. In this chapter we address this issue within the scope of multicast traffic engineering where one set of M-ISIS link weights is optimised for controlling multiple multicast flows. Although it is not always possible to apply this type of shortest-path representability to any arbitrary set of explicit trees with one set of link weights (e.g., violence of loop-freedom), we notice that there always exists an approximation that can be geared towards the TE requirement. In this sense, our OMTE problem is formulated as follows: To assign a unified M-ISIS weight for each link, so that the shortest path trees according to this set of link weights consume minimum network resources. At the same time, it is also guaranteed that the bandwidth allocation on each link should not violate the capacity of individual links.

3.6 A Genetic Algorithm Based Solution

In comparison to MPLS traffic engineering, the task of link weight optimisation is more complicated since the search space for optimal path selection is much larger due to the wide range
of possible weights for individual links. In effect, a set of optimal multicast trees can be enforced potentially through multiple sets of link weights. To deal with this high complexity, metaheuristics are often adopted, e.g., Tabu search for unicast traffic engineering in [34]. In this paper we use another popular meta-heuristic - Genetic Algorithms, which is considered as a promising approach for global searching, for optimising resource provisioning. The basic mechanism of a Genetic Algorithm can be described as follows. First, a series of random solutions are obtained as the initial generation of chromosomes in the population. Thereafter, improved offsprings evolve iteratively from the parents by calculating their fitness. Chromosomes with higher fitness have higher probabilities of being inherited by the next generation. In each iteration, a new generation of approximations is created through the process of parent selection and reproduction. This is specifically achieved through genetic operators such as crossover and mutation. Finally, after a predefined number of generations, or the performance of fitness has reached its convergence, the resulting chromosome with the best fitness is selected as the final solution.

### 3.6.1 Encoding and Initial Population

In our GA approach each chromosome is represented by a link weight vector \( W = \langle w_1, w_2, \ldots, w_{|E|} \rangle \) where \(|E|\) is the total number of links in the network. The value of each weight is within the range from 1 to MAX_WEIGHT. In our experiments we define the value of MAX_WEIGHT to be 64 for reducing the search space. On the other hand, the population size is set to 100, with the initial values inside each chromosome randomly varying from 1 to MAX_WEIGHT. In addition to these randomly generated chromosomes, we add the solution of using hop-count as the link weight into the initial population. This is to guarantee that every link can potentially obtain the lowest link weight such that it has the chance to be included into the resulting trees.

### 3.6.2 Fitness Evaluation

Chromosomes are selected according to their fitness. In our approach, the bandwidth constraint is embedded into the fitness function as a penalty factor, so that the search space is explored with the potential feasible solutions. The fitness of each chromosome can be defined to be a two-dimensional function of the overall network load \((l_1)\) and excessive bandwidth allocated to overloaded links \((l_2)\), i.e.,

\[
\text{fitness} = f(l_1, l_2) = \frac{\mu}{\alpha \times l_1 + \beta \times l_2}
\]

where \(\alpha, \beta, \mu\) are manually configured coefficients.

In equation (3.6) \(l_1\) and \(l_2\) are expressed as follows:
Chapter 3. Offline Multicast Traffic Engineering

\[ l_1 = \sum_{g=1}^{G} \sum_{(i,j) \in E} D_g \times y_{ij}^g \]  
\[ l_2 = \sum_{(i,j) \in E} \omega_{ij} \times \left( \sum_{g=1}^{G} D_g \times y_{ij}^g - C_{ij} \right) \]  

\[ \omega_{ij} = \begin{cases} 0 & \text{if } \sum_{g=1}^{G} D_g \times y_{ij}^g \leq C_{ij} \\ 1 & \text{otherwise} \end{cases} \]  

We note from fitness function (3.6) that the objective is two-fold: first, chromosomes of the new generations should converge towards a set of Steiner trees in terms of hop counts with the lowest bandwidth consumption, and second, solutions obtained from the offspring should be feasible in that the total bandwidth allocated to the multicast flows travelling through each link should not exceed its capacity. The tuning of \( \alpha \) and \( \beta \) can be regarded as a tradeoff between overall bandwidth conservation and load balancing. For example, if we let \( \beta = 0 \) then the objective is to conserve bandwidth only, while setting \( \alpha = 0 \) infers to minimise link overloading within the network.

**Procedure Computing_Fitness(Chromosome i)**

**Begin**

Set the weight of each link in the network according to the gene values in chromosome \( i \);

For each multicast Group \( g \)

Compute the shortest path tree \( T_g \) rooted at \( r_g \), and spanning to all the receivers in \( V_g \);

For each link \( (i,j) \) in \( T_g \)

Update link load \( L_{ij} \) according to the bandwidth demand \( D_g \) of group \( g \);

End For

\( Load1 = 0; Load2 = 0; \)

For each link \( (i,j) \) in the network

\( Load1 = Load1 + L_{ij}; \)

If \( L_{ij} \geq C_{ij} \)

\( Load2 = Load2 + (L_{ij} - C_{ij}); \)

End For
Return fitness = \( f(L_{\text{load}1}, L_{\text{load}2}) \);

End;

Figure 3-4 Fitness calculation

### 3.6.3 Crossover and Mutation

According to the basic principle of Genetic Algorithms, chromosomes with better fitness value have higher probability of being inherited into the next generation. To achieve this, we first rank all the chromosomes in descending order according to their fitness, i.e., the chromosomes with high fitness (lower overall load) are placed on the top of the ranking list. Thereafter, we partition this ordered list into two disjoined sets, with the top 50 chromosomes belonging to the upper class \((UC)\) and the bottom 50 chromosomes to the lower class \((LC)\). During the crossover procedure, we select one parent chromosome \( C^i_U \) from \( UC \) and the other parent \( C^i_L \) from \( LC \) in generation \( i \) for creating the child \( C^{i+1} \) in generation \( i+1 \). Specifically, we use a crossover probability threshold \( K_c \in [0,0.5) \) to decide the genes of which parent to be inherited into the child chromosome in the next generation. We also introduce a mutation probability threshold \( K_m \) to randomly replace some old genes with new ones. In addition to this type of conventional mutation, we also find the congested link with the highest load in the chromosome of the new generation, and we randomly raise its link weight in an ad hoc manner so as to avoid hot spots. In non-congested conditions, this type of mutation the highest loaded link is suppressed.

**Procedure** Crossover\((C^i_U, C^i_L)\)

**Begin**

For all genes \( j = 1, \ldots, |E| \)

Generate \( r = \text{random}[0,1] \);

if \( r > K_c \)

\[ C^{i+1}(j) = C^i_U(j) \]

else if \( r > K_m \)

\[ C^{i+1}(j) = C^i_L(j) \]

else

\[ C^{i+1}(j) = \text{random}[1, \text{MAX}_\text{WEIGHT}] \]

End For

Find gene (link) \( t \) with the highest load in \( C^{i+1} \);

If \( L_t (\text{link load}) > \text{Cap}_t (\text{link capacity}) \)
3.7 GA Processing Analysis

3.7.1 Optimisation Scenarios

In the following examples, we assume that the scaled bandwidth capacity of each link is $10^5$. To illustrate how GA optimisation improves the performance step by step in each generation, we study the following two scenarios. From both of them, we can clearly observe the tradeoff between our main objectives of conserving network resources and guaranteeing feasible solutions.

In the first scenario, we set the maximum group traffic demand $D_g$ to be 4000 (i.e., $D_g$ for each group $g$ is uniformly distributed between 1 and 4000), so that none of the initial solutions in the first generation can satisfy the constraint of bandwidth capacity. From Figure 3-6(a) we can see that the maximum link load computed by the best chromosome in the initial generation is $1.16 \times 10^2$, which means that at least one link is overloaded by 16%. Starting from this set of infeasible solutions, the GA first manages to eliminate the overloaded links by decreasing the load of these congested links. In the figure, we observe that the maximum link load decreases drastically within the first 50 generations, and feasible individuals emerge from then on. Thereafter, the overall bandwidth consumption starts to drop significantly (shown in Figure 3-6(b)) with the maximum link load varying just below the bandwidth capacity for most of the period. Finally in the 500th generation the overall bandwidth consumption converges to the lowest value ($7.08 \times 10^6$), while the link with the highest load becomes nearly saturated ($9.7 \times 10^4$) but not overloaded.
In the second scenario, we set $D_s = 3000$, so that feasible solutions already exist in the initial population. In Figure 3-7(a) we can see that the load of the highest link is about 80% of the capacity in the first generation. When the GA optimisation starts, the overall bandwidth consumption decreases significantly as shown in Figure 3-7(b). During this period, the traffic distribution becomes less balanced, as the highest link load increases sharply within the first 10 generations. Although this value exceeds the bandwidth capacity occasionally, for most of the period the highest utilisation varies between 90% and 100%, thus feasible solutions are guaranteed in each generation.
3.7.2 Time Complexity Analysis

From the description in section 3.6.2, we can find that the computing of fitness takes up most processing time within each generation, compared to crossover and mutation. From Figure 3-4 it is easy to figure out that the time complexity of fitness calculation is $O(G | N|^3 + |E|)$, where $G$ is the total number of active groups, while $N$ and $E$ are number of nodes and links in the network. Given the fact that $|N|^3 >> E$, the overall time complexity of the GA algorithm is $O(MPG | N|^3)$, where $M$ is the predefined maximum generation and $P$ is population size. We ran the algorithm on a PC with Pentium IV 1.4G processor, and it took about 8 minutes to compute the optimal values for a network with 100 nodes and 100 groups, and $M$ and $P$ set to 500 and 100 respectively.
3.8 Simulation Results

3.8.1 Simulation Configuration

In this section, we evaluate the proposed approach through simulation. We adopt the Waxman's model in GT-ITM topology generator [52], an ideal model for creating *intra-domain* topologies, for constructing our network models. This approach distributes the nodes randomly on the rectangular grid and nodes are connected with the probability function:

\[ P(u,v) = \lambda \exp\left(-\frac{d(u,v)}{\rho L}\right) \]  

(3.10)

where \( d(u,v) \) is the distance between node \( u \) and \( v \) and \( L \) is the maximum possible distance between any pair of nodes in the network. The parameters \( \lambda \) and \( \rho \) (ranging \((0, 1)\)) can be modified to create the desired network model. A larger value of \( \lambda \) gives a node with a high average degree, and a small value of \( \rho \) increases the density of shorter links in comparison to longer ones. In our simulation we set the values of \( \lambda \) and \( \rho \) to be 0.2 respectively, and generate a random network of 100 nodes, out of which 50 are configured as Designated Routers (DRs) with attached group sources or receivers. The total number of groups is set to 100. It is worth mentioning that, given the fact that IP multicast has not been globally available over the Internet, it is difficult to estimate the actual scale of the relevant deployment within a typical ISP's network. However, we believe that the proposed simulation scale is valid for a medium to large-sized ISP.

The simulation parameters of the proposed Genetic Algorithm are illustrated in Table 3-1. Effectively, we also tried to tune the value of GA control parameters within the reasonable range (typically \( K_C \) and \( K_M \)), and we found the final results are not significantly sensitive to these changes. Throughout this thesis, we run 10 independent experiments for one data point for guaranteeing high confidence level. Apart from the GA approach, we also implemented two non-TE based hop-by-hop routing approaches and one explicit routing approach: (1) shortest path routing with random link weight setting (Random), (2) shortest path routing in terms of hop-counts (SPH), and (3) Steiner tree approach using the TM heuristic [72]. For this TM Steiner tree algorithm, we use hop count as the link weight, and the resulting trees are group specific, i.e., one Steiner tree is specifically constructed for each multicast group. The pseudo-code of using the TM heuristic for constructing multicast trees is presented in Appendix A.1. In the next section we will show that the TM heuristic has the best performance among the four algorithms in terms of

---

\(^2\) We also tune the value of \( \lambda \) and \( \rho \) within the reasonable range, and we found that this does not impact the simulation result significantly in all chapters.
bandwidth conservation. Nevertheless, it should be emphasised that this solution requires the setting up of MPLS tunnels for explicit routing on a per-group basis, and this cannot be achieved in a pure IP environment. Hence, the inclusion of the TM algorithm is only to use its performance as a lower bound reference for comparison with the other three hop-by-hop oriented approaches without MPLS tunnelling. It should also be noted that, as the TM algorithm is solely designed for reducing the total tree cost (bandwidth consumption in our case), it does not provide any other optimisation functionality such as reduction of link congestion when multiple multicast trees are constructed. From our simulation results we can also find that its performance in other evaluated metrics is not good even compared to shortest path routing with hop counts (SPH).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size ( (P) )</td>
<td>100</td>
<td>( \mu )</td>
<td>10^7</td>
</tr>
<tr>
<td>Maximum generation ( (M) )</td>
<td>500</td>
<td>( \alpha )</td>
<td>1.0</td>
</tr>
<tr>
<td>Maximum link weight ( (\text{MAX} _\text{WEIGHT}) )</td>
<td>64</td>
<td>( \beta )</td>
<td>10</td>
</tr>
<tr>
<td>Crossover probability threshold ( (K_C) )</td>
<td>0.30</td>
<td>Mutation threshold ( (K_M) )</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Table 3-1 GA parameter configuration

3.8.2 Performance Evaluation

We found from our simulation that shortest path routing with hop-counts (SPH) has higher capability in finding feasible solutions than random link weight setting approaches (shown later). Hence, we will start from the comparison between GA and SPH in the capability of exploring feasible solutions. Figure 3-8 presents the ratio of successful instances obtained by GA but not achieved in SPH. We define the Maximum Link Overload Rate (MLOR) as follows:

\[
MLOR = \max_{(i,j) \in E} \left( \frac{\sum_{g} D_g \times y_{ij}^g - C_{ij}}{C_{ij}} \right)
\]  

(3.11)

From this definition we can see that MLOR reflects the overloading scale of the most congested link (if any, i.e., MLOR>0). In the figure, when the value of MLOR computed by SPH is below 5%, GA can obtain feasible solutions (i.e. \( MLOR_{GA} \leq 0 \)) for 65% of these instances. We can also see that, with the increase of external bandwidth demands, the capability of GA in finding feasible solutions decreases. When the MLOR value of SPH grows up to 25% due to the higher external traffic demand, the success rate of GA drops to 5%. From this figure, it can be inferred that when the external group traffic demand is at the brink of causing network congestion, GA has higher capability of avoiding link overloading compared to other approaches. Obviously, it may
be the case that no feasible solution exists at all, if the external traffic demand exceeds a certain threshold.

![Figure 3-8 GA Success rate vs. MLOR SPH](image)

**Figure 3-8 GA Success rate vs. MLOR SPH**

Figure 3-9 illustrates the feature of overall bandwidth conservation capability of individual schemes with the variation of maximum group traffic demand $D_g$. As it is expected, explicit routing with the TM heuristic achieves the lowest overall network loading while random link weight assignment results in the poorest performance. We can also see in the figure that the GA approach exhibits the best capability in conserving bandwidth among all the hop-by-hop routing schemes. Typically, when the network is under-utilised, our proposed GA approach exhibits significantly higher performance than the conventional IP based solutions without explicit routing. For example, when $D_g = 3000$ the overall bandwidth consumption of the Random and SPH solutions are higher than that of GA by 19.3% and 14.9% respectively. Compared with the TM heuristic that needs support from MPLS overlaying, the gap from GA is below 8%. However, when the external traffic demand grows, the performance of GA converges to that of the SPH approach. On the other hand, although the TM algorithm exhibits significant higher capability in bandwidth conservation when the external traffic demand grows ($D_g > 4000$), this does not mean what have been obtained are feasible solutions without introducing overloaded links.
The rest of the simulation evaluates the capability of alleviating network congestion in our proposed solution. From Figure 3-10 and Figure 3-11 we can see that in time of overwhelming multicast traffic demand, network congestion will be inevitable. In this scenario, a limited number of LSPs may be established only for rerouting the traffic flows that contribute to the link congestions to other under-utilised paths (if any) [13]. Since the scope and scale of congestion can be significantly reduced through our GA based approach, it is possible for an INP to identify individual flows that incur link overloading in a more tractable way. By reducing the scope and scale of overloaded links through our approach, the complexity of setting extra LSPs or excising admission control on excessive receivers can be significantly decreased.

Figure 3-10 shows the relationship between the proportion of overloaded links and the maximum group traffic demand $D_g$ in time of congestion. From the figure we can see that there exist more overloaded links within the network as $D_g$ increases. The most interesting result is that, through our GA optimisation, the number of overloaded links is significantly lower than all the other routing schemes. In the most congested situation ($D_g=6000$), the average rate of overloaded links computed by GA is only 1.4%, in contrast to 12.6% by random link weight setting, 8.6% by the TM heuristic, and 4.4% by SPH respectively. On the other hand, the amount of overloaded bandwidth occurred on the most congested links is another important parameter an INP could be interested in. An INP should avoid configuring the network resulting in hot spots with high MLOR. Through our simulations, we also find that the proposed GA approach achieves the lowest MLOR performance. In Figure 3-11, the overloading scale is 45% of the bandwidth.
capacity on the most congested link in the GA approach with $D_g$ equal to 6000, while this value reaches 110% and 59% in random link weight setting and SPH respectively. Even by using the explicit routing TM heuristic, the bandwidth is 78% of the original link capacity.

3.9 Summary
In this chapter we proposed an efficient scheme for OMTE with bandwidth constraint using Genetic Algorithms. By means of off-line optimising and pre-configuring M-ISIS link weights, traditional Steiner tree based multicast traffic engineering can be reduced to plain PIM-SM
shortest path routing that is widely supported in the current IP routers. Moreover, the GA-based approach also exhibits higher capability in finding feasible solutions and reducing network congestion. As far as we know, our proposed approach represents the first attempt to explore effective solutions to multicast traffic engineering based on the hop-by-hop routing semantics. This is in contrast to most of the current multicast traffic engineering schemes that require MPLS support. In the next chapter we will study the real-time performance of PIM-SM routing behaviour with this optimised M-ISIS link weight configuration in both best-effort and multi-service environments, such as DiffServ networks.
Chapter 4

4 DiffServ-aware Multicast Using QSSM

4.1 Introduction

In this chapter, we focus on control plane multicast semantics (including routing and group management etc.) in an IP differentiated services environment. The associated design and implementation is in effect the realisation of the Dynamic Multicast Routing (DMR) and Dynamic Group Management (DGM) functional blocks in the control plane of the proposed framework. In addition, we also investigate the issues of applying the OMTE algorithm in the management plane to the enforcement of the DMR behaviour in multi-service environments.

As we have specified in Chapter 2, the DiffServ architecture is seen as a promising technology for service differentiation in a large scale due to the fact that the core network is kept relatively simple, with most complexity confined at the network edge and the management plane. Admission control and traffic conditioning are performed at edge routers, while core routers simply treat traffic aggregates on a Per Hop Behaviour (PHB) basis according to the Differentiated Services Code Point (DSCP) in each packet. On the other hand, the fundamental principle of traditional IP multicast is to maintain group states where necessary within the network in order to route data to active receivers. We notice from the existing solutions to DiffServ aware multicasting that, in order to support heterogeneous QoS classes (QCs) requested from end users, DiffServ/multicast core routers need to maintain QC information for downstream group members in addition to the conventional group states \[10\]. Moreover, multicast protocols such as PIM-SM and IGMP are required to be extended for carrying QoS class information from heterogeneous receivers. These aspects impose not only additional memory overhead from a scalability point of view, but also backwards compatibility problems with existing multicast protocols. The basic reason for this undesired situation is that, the DiffServ framework caters mostly for sender-based unicast communication in which Service Level Specifications (SLSs) with a provider specify traffic entering the network at a particular ingress router. In the inherently receiver-initiated multicast paradigm, it is individual group members that demand various classes of service. Thus traffic from the multicast sender may need to reach group members with different QoS requirements on packet treatment.
In this chapter we propose a seamless integration of the emerging Source Specific Multicast (SSM [9]) service model and DiffServ infrastructure. By using the dedicated SSM group address to express and convey QoS requirement during group subscription from receivers, the fundamental conflict between sender-based DiffServ and receiver-oriented multicast can be gracefully handled. Since the proposed solution requires no extensions to existing router architecture and to underlying multicast protocols such as IGMP and PIM-SM, we believe that this approach can be directly deployed in a large scale on the Internet.

The organisation of the rest of this chapter is as follows. We first analyse router extension requirements with existing DiffServ-multicast approaches in section 4.2. Following that we present a detailed description of the proposed QSSM model in section 4.3, including QoS mapping, routing and group management issues. Finally in section 4.6 we present a performance evaluation of the control plane behaviour of QSSM, considering in particular the interaction with the management plane OmNE optimisations.

### 4.2 Router Extension Requirements with Conventional Approaches

Apart from DSMCast [70, 71], which is based on overlay approaches (explicit multicast), related schemes on DiffServ-aware IP multicasting include [10, 40, 81]. We can classify these solutions into two distinct categories regarding the strategy of maintaining heterogeneous QoS classes within multicast distribution trees. In [40], the authors propose that one specific distribution tree should be constructed for each QoS Class (QC) within one group, and different QC trees for the same group are completely independent of each other. In this scenario, one multicast session with \( n \) QoS classes needs \( n \) independent multicast trees. We call this approach QoS Specific Multicast (QSM) and it is illustrated in Figure 4-1(a). All the other approaches adopt one single tree that encapsulates multiple QoS classes, with individual tree branches reflecting heterogeneous QoS requirements. The key idea of this type of tree is that branches with lower classes can be directly grafted from those with higher classes for the same group. We call this strategy Hybrid QoS Multicast (HQM) and it is depicted in Figure 4-1(b). In this section we investigate core router extension requirements for implementing DiffServ-aware multicast by these approaches (assuming SSM trees).

Let's define first the following notations:

\[ |s| \] — Length of source address, which is 32 bits in IPv4.

\[ |G| \] — Length of SSM group address which can be distinguished by 24 bits in IPv4 (232.*.*.*).

\( X \) — Total number of interfaces in a router.
$Y$ — Total number of QoS classes an INP provides.

According to [39], the size of a typical forwarding entry (number of bits) for each $(S, G)$ group in a conventional SSM-aware router can be expressed as:

$$E_{SSM} = |S| + |G| + \log_2 X + X$$  \hspace{1cm} (4.1)

Assuming that the maximum number of interfaces per router is 16, a forwarding entry is $32 + 24 + 4 + 16 = 76$ bits long.

Now we consider the forwarding entry structure of QSM and HQM. In both QSM and HQM, join requests from receivers also carry the corresponding desired QC; when core routers receive these requests, the embedded QoS states are recorded at individual oifs (Figure 4-1). In QSM, since each oif should be associated with multiple classes of service for a particular $(S, G)$ group, the most efficient solution is to append a binary DSCP vector to each forwarding entry with each bit in the vector denoting a particular QC. The structure of this type of entry is shown in the lower part of Figure 4-1(a), and its size is:

$$E_{QSM} = |S| + |G| + 2 \times \log_2 X + Y$$  \hspace{1cm} (4.2)

If the maximum number of QoS classes provided by an INP is 64 (as proposed in [54]), then the length of one forwarding entry is 128 bits.

In HQM approaches, since a single tree is used to handle all QoS classes for a particular group, each oif needs to be associated with one unique QC for a specific group and thus the most straightforward solution is to append an encoded DSCP value to each forwarding entry (shown in lower part of Figure 4-1(b)). The size of such type of entry is:

$$E_{HQM} = |S| + |G| + 2 \times \log_2 X + \log_2 Y$$  \hspace{1cm} (4.3)

Again, given a maximum of 64 classes of service, the size of a forwarding entry is 70 bits. Note that this value is smaller than that of plain SSM, but it should be noted that both HQM and QSM forwarding entries are outgoing interface instead of group specific, i.e. a router has to maintain $k$ HQM/QSM forwarding entries for a given group where $k$ is the total number of oifs associated with it. In contrast, in conventional SSM each forwarding entry expressed in equation 4.3 is associated with the whole group rather than a specific outgoing interface.

From the above analysis, it is obvious that existing implementations of DiffServ-aware multicast requires extensions to the underlying forwarding table infrastructure for the inclusion of dedicated DSCP information on different outgoing interfaces. This type of extension violates the basic QoS stateless requirement by the traditional DiffServ model. Moreover, in order to convey
heterogeneous QoS requirements from end users, both IGMP and PIM-SM need to be extended to be DSCP-aware.

![Tree structure](image)

**Figure 4-1 QSM vs. HQM**

### 4.3 QSSM Semantics

#### 4.3.1 QoS Mapping

The proposed QoS-Source Specific Multicast (QSSM) scheme can be regarded as an integration of the Source Specific Multicast and Differentiated Services models, which both address scalability issues in multicast and service differentiation respectively. In SSM each group is identified by an address tuple (S, G) where S is the unique IP address of the information source and G is the destination channel address (in the 232.0.0.0/8 address range). Since channels exist on a per-source basis, issues such as class D address allocation and inter-domain source discovery, which are problematic in the IP multicast service model, are successfully eliminated. From a routing point of view, join requests from individual subscribers create a unique multicast delivery tree rooted at the well-known information source.
In the proposed QSSM architecture, the Internet Network Provider (INP) provides external content providers/receivers with finite classes of unified qualitative services, each of which is uniquely encoded into one (or a set of) class D address in the SSM address range 232.0.0.0/8. In such a situation, the interpretation of the SSM address tuple \((S, G)\) becomes straightforward: \(S\) identifies the address of the information source and \(G\) identifies the QoS service level (we name it QoS channel) available from \(S\). In effect, it is not a new idea to encode QoS classes with dedicated multicast group addresses: In Designation Set Grouping (DSG) [21] and Receiver-driven Layered Multicast (RLM) [76], sources send video streams with different QoS levels into different group addresses, and receivers are allowed to subscribe to the corresponding groups based on their individual capacities. However, it should be noted that there is a basic difference between QSSM and the above approaches: it is the INP that provides a set of unified QoS channels in QSSM, and group data from external sources is treated in these finite number of differentiated channels, hence it is possible for traffic aggregation from individual multicast sources in DiffServ networks. The distinct advantage of this scheme is that since the QoS class is embedded into the group address, no QoS-related states need to be maintained inside DiffServ core routers, and hence the forwarding table extension described in section 4.2 is not necessary. In order to support compatibility with the conventional DSCP-based forwarding in DiffServ environment, a logical mapping table is constructed with the responsibility of translating group address into a DSCP value that is associated with a specific QC (Figure 4-2). In section 4.3.2 we indicate that this type of mapping only needs to be maintained at edge routers. Through this mapping table, we are able to link receiver-initiated QoS requirements (encoded into the group address carried by join requests) with the actual data treatment (decided by the DSCP value). The overview of the QSSM architecture is shown in Figure 4-3, and detailed description on the relevant QoS mapping and data treatment is presented in section 4.3.2.

<table>
<thead>
<tr>
<th>SSM group address</th>
<th>DSCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G_1)</td>
<td>(EF)</td>
</tr>
<tr>
<td>(G_2)</td>
<td>(AF11)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>(G_n)</td>
<td>(BE)</td>
</tr>
</tbody>
</table>

Figure 4-2 QSSM mapping table

By effectively encoding QoS states into multicast addresses and maintaining these states within the network, no additional states need to be added to the existing multicast forwarding entries. In this sense, Source Specific Multicast can be integrated with DiffServ without QoS extensions of current multicast router architecture. On the other hand, the maximum number of QoS classes in
Chapter 4. DiffServ-aware Multicast Using QSSM

DiffServ is restricted by 6 bits of the DSCP field, and the allocation of 64 dedicated class D addresses will not cause any problem in the use of the SSM address range that contains $2^{24}$ addresses.

However, there is one restriction regarding the implementation of this approach. Since the QoS channel is source specific, it is impossible for a single source with a unique IP address S to send multiple data streams with different content. In the classic SSM model, an information source can be simultaneously in multiple groups because (S, G1) and (S, G2) are completely independent. To solve this problem, the INP can map multiple class D addresses for a single QoS channel. In this scenario, if a source wants to send multiple multicast flows for different sessions in a single QoS class, it can choose different class D addresses that are mapped onto the same QoS channel for individual groups. Although the join requests of the two sessions carry different group addresses, when the corresponding multicast flows are injected by the ingress router the DSCP will be marked to the same value for that specific QoS channel (see section 4.3.2).

![Figure 4-3 QSSM framework overview](image)

### 4.3.2 QSSM Tree Management

Recent research works have shown that construction of a unique hybrid delivery tree for all classes of service (i.e., HQM) might result in fairness problems for receivers with different service levels [71]. For example, the “Good Neighbour Effect” takes place when a group member subscribing to lower class is physically located near another receiver with a higher QoS class. Relevant simulation studies indicate that the two subscribers might receive group data with almost the same QoS performance, although they have subscribed/charged at different service/price levels. We should mention though that whether this QoS fairness issue is problematic or not from INP's viewpoint is still under debate.
In our proposed architecture, we build source specific trees on a per QoS class basis, i.e., different QoS channels for a specific source \( S \) are independently maintained even if some of them might have overlapping tree links within the DiffServ domain. The basic characteristic of QSSM is that one source specific tree only serves a particular QoS class and data packets delivered on this tree exhibit the same class of service. Although this is similar to the QSM approach, the basic difference lies in the fact that QSSM needs no forwarding table extension to introduce QoS states in DiffServ domains, since the relevant state is embedded into the SSM group address. Moreover, the QSSM multicast session should be source specific, which satisfies the fundamental requirement of the conventional SSM service model.

The construction of QSSM trees is illustrated in Figure 4-4. Once an end host \( R \) decides to join the QSSM tree rooted at source \( S \) with a desired QoS class, it first sends an IGMPv3 \((S, G)\) group membership request to its Designated Router (DR) at the edge of the DiffServ domain, where \( G \) is the associated QSSM group address mapped to the negotiated QoS channel. If there is not sufficient bandwidth for admitting the traffic invoked by the join request, this join packet will be dropped by the DR, and the user may alternatively opt to select a lower \((S, G)\) QoS channel. To achieve this, the DR (egress router) may apply the probing mechanism proposed in [10] for detecting potential congestion within each QoS class. A detailed specification on the DiffServ-aware group management between end hosts and DRs will be presented in section 4.5. On receiving the group membership request from \( R \), the DR will send a plain \((S, G)\) join request towards \( S \), and this join request packet will either reach the source \( S \), or it will be intercepted by an on-tree router with the same \((S, G)\) state. It should be noted that when core routers receive the QSSM join request, they only create plain \((S, G)\) state and they do not maintain any QoS-related information for the group, as it is required by the approaches presented in section 4.2. In effect core routers need not know about the mapping between QoS classes and group addresses.

Once the source \( S \) receives multiple \((S, G)\) join requests with different group address \( G \), it will map each of them onto independent QoS channels respectively. When the \((S, G)\) group traffic flows back into the DiffServ domain along the reverse path created by the join request, the ingress router (IR) will mark the data packets with the matching DSCP value according to the address \( G \) being carried. This type of marking is done by looking up the locally maintained mapping table between group address and DSCP value at the ingress router. Thereafter, traffic from \( S \) will flow along the \((S, G)\) tree back to the subscriber with the desired DSCP, based on which core routers will forward the packet to the proper DiffServ queue. If one \((S, G)\) join request is intercepted at a core router already having this state, a new branch is naturally grafted from the current \((S, G)\) tree, in a similar fashion to the conventional SSM join procedure. Moreover, replicated packets in the new branch still contain the original DSCP value since core routers never remark them at the branching point. In this scenario, it is guaranteed that the resulting source specific tree is QoS
Specific as well. From the core routers' point of view, multicast packets carrying common DSCP values can still be treated in an aggregate fashion, and furthermore, treatment of group data is exclusively based on the DSCP value in the packet header, instead of QoS states maintained at core routers. It should be noted that, if the INP maps multiple class D addresses to each QoS channel for allowing sources to send traffic to different group sessions, it is required that the mapping table at the IR should also record this multi-mapping information. As a result, when the ingress router injects flows with different class D address mapping to the same QoS channel, a common DSCP value will be marked so that traffic aggregation is still possible within each class.

Maintaining QSSM trees has the following advantages. First, inter-class fairness problems are avoided thanks to the QoS specific tree approach, and this has been proved in [71] by simulation. Second, there is no need to perform traffic reconditioning at core routers, because this is done at the edge of the DiffServ domain. Finally, since the QSSM group address is used as the carrier of QoS requirements from individual group members during the join procedure, neither the PIM-SM and IGMPv3 protocols nor core routers need to be extended with additional QoS information. On the other hand, QSSM results in wasting bandwidth in common links since replicated data with the same content are transmitted in multiple QoS channels simultaneously.

Figure 4-4 QSSM group join procedure

4.4 Inter-domain QSSM Operations

In this section we explain how QSSM trees are constructed and maintained across multiple Autonomous Systems (ASs). One of the challenges in handling inter-domain QoS delivery lies in the fact that INPs have heterogeneous DiffServ configuration policies. For example, each DiffServ domain might provide a different number of QoS classes, and for the purpose of flexibility, the DSCP identification for each class need not be necessarily consistent in all
domains. Since there is no yet mature solution for inter-domain DiffServ operation, we only discuss some preliminary directions towards inter-domain QSSM deployment.

In a similar fashion to DSCP usage, INPs should be allowed to map arbitrarily QSSM based group addresses to any class of service they provide within their own ASs. In this case, when two adjacent INPs set up a peering multicast Service Level Specification including compatible QoS class mapping and binding (i.e., multicast aggregates belonging to class i in domain A should be mapped to class j in domain B and vice versa) for their QoS capability extensions, the QSSM group address might not be identical for class i and j in the two DiffServ domains. Considering this difference in QoS class identification between different domains, we propose a mechanism for QSSM group address mapping at the edge of DiffServ domains. One basic assumption is that inter-domain QoS class mapping is part of the peering mSLS, but exactly how this is achieved is outside the scope of this work; the mechanism we describe is only concerned with extending this type of peering mSLS for QSSM deployment. Figure 4-5 illustrates a basic scenario on inter-domain QSSM management between two adjacent INPs. It should be obvious from the figure that the peering mSLS only involves QSSM group address conversion, with DSCP / PHB mapping hidden from external peers.

If an end user wants to subscribe to a QSSM group whose source is located in a foreign domain, an inter-domain join request is issued, as in the conventional SSM group join. It should be noted that this user should choose one QoS channel available from its local domain. Suppose that the user selects QoS class i in its own domain A, then a \((S, G^i_A)\) join request will be sent towards the remote source S. Once this join request is admitted into the adjacent domain, say domain B, the
QSSM group address will be converted into $G_{b_j}$ based on the peering mSLS between domain A and B at the border node of domain B. Finally, what the source S or any grafting router already on the existing QSSM tree receives is the join request with a recognised group address in its own domain. When group traffic is transmitted back towards the new subscriber, the QSSM group address is also converted at the ingress router of the transit domains. When the data packet arrives at each DiffServ domain, the ingress router first changes its group address based on the peering mSLS, and then by looking up the local mapping table, it remarks the DSCP value according to the new QSSM address. In such a scenario, all the following core routers will use the proper queue for scheduling by checking the local DSCP value contained in the group data packets.

Another important consideration is the negotiation of the availability of the selected QoS class with the Bandwidth Broker (BB) of each domain for end-to-end QoS guarantees. This can be typically done with the local domain for receiving multicast traffic and the BB of that domain may negotiate it with its counterparts of other domains in a cascaded or star fashion. This points to models for inter-domain QoS management that are outside the scope of this work.

---

Figure 4-6 presents a simple example on how receiver R in domain A subscribes to the QSSM tree whose source S is in remote domain X. First R selects a local QoS channel and sends $(S, G_{ai})$ join request towards S. This request will follow the proper AS path by means of domain level RPF checking using the underlying Multi-protocol extended BGP (MBGP [8]). When this join request enters the transit domain B, the QSSM group address is changed into $G_{bj}$ at edge router B1 according to the mSLS between A3 and B1. Finally, what S receives is a $(S, G_{bj})$ join request that is locally recognised in domain X. When the $(S, G_{bj})$ group data flowing back towards R arrives at B3, this ingress router first changes the IP destination address to $G_{bj}$ and...
then remarks the DSCP value by looking up its mapping table. Since the DSCP value is exclusively remarked at ingress routers, it will not be changed from B3 to B1. In this case, when internal routers (e.g., B2) receive the packet, they can directly schedule this packet to the DiffServ queue associated with \( DSCP_{B1} \). Similarly, when the group data flows into domain A, the destination address is changed into \( G_{A1} \), and data packets are scheduled according to the new \( DSCP_{A1} \) remarked at A3, hence the new subscriber S receives data from the QoS channel \((S,G_{A1})\).

### 4.5 Dynamic Group Management in DiffServ Networks

Currently, IGMP is not capable of handling QoS heterogeneity. As far as DiffServ aware multicast services are concerned, IGMP needs QC extension for the management of heterogeneous QoS classes between egress routers (i.e. DRs) and end hosts. In both QSM and HQM schemes, the DR on each subnet not only takes care of \((S,G)^3\) group membership reports, but also needs to check the attached QC identifier (e.g., DSCP value), which identifies the requested QoS class from end hosts. In QSM, when the DR receives multiple \((S,G)^3\) group membership reports with different QC identifiers, it will treat them as different groups and always trigger one dedicated PIM-SM join request for each of them. In other words, IGMP operations are \((S,G,QC)\) specific instead of \((S,G)\) specific in its conventional version. Once the DR is connected to the individual QC trees for the \((S,G)\) group, it should prevent group members with a lower QC requirement from receiving the same \((S,G)\) group traffic in the higher class. In case of HQM, although IGMP group membership reports are also in \((S,G,QC)\) style, the DR only triggers one unique PIM-SM join request for the \((S,G,QC)\) report with the highest QC requirement, with all the other group membership reports being suppressed. In this case, all the other end hosts subscribing to lower QCs will automatically enjoy the multicast traffic with the “upgraded” QoS treatment. At this moment, whether this scenario is problematic from a fairness point of view is still an open issue.

In our proposed QSSM scheme, IGMP packets still retain the original \((S,G)\) format, as QC requirements from end hosts have already been identified with the group address G. When multiple receivers want to subscribe to the group provided by S with heterogeneous QCs, they only need to send different \((S,G_i)\) group membership reports to the subnet, where \(G_i\) is the SSM group address identifying the requested QoS class. On receiving different \((S,G_i)\) IGMP reports, the DR will send the corresponding \((S,G_i)\) PIM-SM join requests for each of them. Existing

---

3 In this thesis we assume source specific group joins in IGMP and PIM-SM in our illustration by default.
access control mechanisms for new group joins (e.g., GRIP in QUASIMODO) can also be adapted to QSSM, where the \((S, G)\) report will be suppressed once congestion at any intermediate router along the path to join is detected in the QoS channel identified by \(G\). Figure 4-7 presents a basic description of the difference in IGMP operation between QSM and QSSM, both of which are QC approaches.

![IGMP operation in QSM and QSSM](image)

**Figure 4-7 IGMP operation in QSM and QSSM**

### 4.6 Simulation Results

#### 4.6.1 Simulation Scenario

In this section we study the real-time performance of DMR in the control plane with dynamic group membership updates. Similar to the simulation scenario in section 3.8, we also evaluate the following four paradigms: (1) Random link weight setting (Random), (2) Shortest path routing with hop counts (SPH), (3) Steiner tree based explicit routing with TM algorithm (TM) and (4) OMTE optimised link weight with GA (GA). In section 4.6.2 we compare the performance of the above four approaches in terms of block rate and average network load. In section 4.6.3 we apply
the OMTE optimised link weight setting to the DMR block, and evaluate how it performs in DiffServ networks with multiple classes of service.

We emulate a sequence of events for group membership updates based on the static scenario by using the probability function proposed in [78], and we evaluate the real-time traffic condition with the group dynamics derived from the original static multicast traffic matrix. For each event, we first randomly select one group \( g \in G \), and then use the following probability function to decide whether this event is a group join or leave:

\[
P_s = \frac{\omega(|V_g| - m_g)}{\omega(|V_g| - m_g) + (1 - \omega)m_g}
\] (4.4)

In the function, \( m_g \) indicates the instant number of active members while \(|V_g|\) identifies the maximum size of group \( g \) (i.e. total number of subscribers). \( \omega \) ranging \([0, 1]\) is known as the invocation ratio that controls the density of each group. For example, \( \omega = 0 \) means that no group joins are invoked, while \( \omega = 1 \) indicates full group membership invocation. In our simulation we use this function for creating a series of events of group join/leave based on the static multicast traffic matrix. When a join request is issued for group \( g \) \((P_g > \text{a randomly created float number ranging from 0.0 to 1.0})\), a node \( v \in V_g \) but not yet on the multicast tree \( T_g \) is selected to join the group. Likewise, in case of a leave request for group \( g \), an on-tree node is randomly selected for pruning from \( T_g \).

### 4.6.2 Performance Comparison within One Class

In the following simulations, we assume that new group join requests will be blocked once network congestion (i.e., an overloaded link) has been detected. Figure 4-8 and Figure 4-9 show respectively one typical instance of the real-time performance (5000 events in group dynamics) in terms of overall network load\(^4\) and maximum link utilisation respectively, with \( D_g \) equal to 3000 and \( \omega \) equal to 1.0. In this condition the network is lightly loaded with no link congestions. From Figure 4-8 we can see that when the group dynamics converge to a steady state, the network load resulting from random link weight setting is the highest, while using the TM algorithm for MPLS explicit routing achieves the lowest resource consumption. We also find that the proposed link weight optimisation using the GA approach results in very low network load compared to other IP based approaches, and its performance is even very close to the TM explicit routing scheme. This

\(^4\) The network load is defined as the mean ratio of consumed bandwidth over the link capacity, so that it can directly reflect the performance of overall bandwidth conservation.
result is consistent with the static simulation scenario shown in Figure 3-9. As shown in Figure 4-9, the GA optimisation approach results in very high utilisation of the most heavily loaded link, which is only next to the Random link weight solution⁵. On the other hand, both the SPH and TM algorithms exhibit good performance in load balancing. Nevertheless, it should be noted that although the performance in maximum link utilisation by the GA approach is not as good as these two schemes, there is still no network congestion as all the links are under-utilised and the overall bandwidth resources are significantly conserved.

⁵ Before the group dynamics come to a steady phase, the maximum link utilisation of the GA approach is even higher than the random weight approach.
From Figure 4-10 and Figure 4-11 (typical instances for under-provisioning scenarios) we can see that the performance of the four approaches changes significantly in time of overwhelming traffic demand when Max $D_g$ is increased to 6000. First, both the GA and SPH approaches converge to the highest overall network load. On the other hand, explicit routing with the TM algorithm still achieves the lowest resource consumption, which remains the same with the scenario in Figure 4-8. From Figure 4-11 we see that all four schemes result in 100% utilisation in the highest loaded link due to the overwhelming traffic demand, and thus some new group joins are blocked due to the overloaded links. We can also see from this figure that the random approach first converges to the congested state while our proposed GA optimisation is the last to reach this phase. This implies that more group joins are likely to be rejected in the former while the least join requests will be blocked in the latter. In effect, group join blocks prevent the underlying multicast trees from consuming more network resources, and this explicitly explains why the overall network load of SPH and GA is higher than the random link weight approach in Figure 4-10, where a large number of group joins have failed due to overloaded links. Our subsequent simulation study will continue to focus on the statistics of group join blocks for the four approaches in different scenarios (e.g. with variations of $\omega$).
Figure 4-11 Real-time performance in maximum link utilisation (Max $D_s=6000$, $\omega=1$)

Figure 4-12 illustrates the overall block rate with the variation of the invocation ratio $\omega$ with respect to the 5000 group updates, while maximum $D_s$ is set to 6000. From the figure we can see that more group joins are rejected as the invocation ratio grows. The reason for this is that, bandwidth consumption increases when there are more active members in each group. Once the consumed bandwidth on any link reaches its capacity, new group joins are blocked due to the detected congestion. On the other hand, we notice that through sophisticated network dimensioning using the proposed MT-IGP link weight optimisation, group join blocks are significantly lower than in the other approaches. When $\omega$ increases from 0.5 to 1.0, the total number of blocks grows very slowly with our proposed GA solution, which is in contrast to all the other conventional methods. One interesting thing is that, compared to Figure 3-11 in the static scenario, although the provisioning performance of the GA approach results in 45% MLOR, the actual number of blocked join requests is quite low (2.1%) even in case of full group invocation. When $\omega < 0.7$, there are no blocked group join requests at all. The reason for this is that while there are overwhelming group joins, group leaves also take place at the same time, with used bandwidth resources returned to the network. Finally, it is also worth mentioning that the MPLS based Steiner tree approach does not exhibit strong capability in reducing the blocking rate, as the TM algorithm is solely greedy in bandwidth conservation and not in eliminating congested links.
Figure 4-12 Join block rate vs. invocation ratio \( \omega \)

Figure 4-13 shows the overall network load versus invocation ratio \( \omega \) with respect to the 5000 group updates. From the figure we can see that higher invocation ratio results in higher network load. On the other hand, the TM heuristic using MPLS explicit routing always achieves the lowest network load, which is in line with Figure 4-10. Moreover, we also notice that the network load of the GA optimisation is very close to that of the TM approach when \( \omega \) is relatively small, and this again indicates that the proposed solution exhibits strong capability in bandwidth conservation in time of light traffic loading. However, with the growth of \( \omega \), the network load by the GA approach increases more sharply than all the other approaches, and this is because more group joins are able to be accommodated successfully, while in the other approaches, especially the random link weight one, a large number of join requests are blocked due to network congestion so that the total bandwidth consumption is relatively lower. In effect, by studying the real-time performance in Figure 4-8 and Figure 4-10, the same conclusion can be drawn on the performance of network loading.
Chapter 4. DiffServ-aware Multicast Using QSSM

4.6.3 OMTE-Driven Performance in DiffServ Networks

The simulation study in this section addresses the issue of applying OMTE to DiffServ aware networks for service differentiation in multiple QoS classes. In this scenario, resource provisioning through MT-IGP link weight optimisation should be exercised within individual QCs independently. We demonstrate a simple case study of how to dimension successfully the network for multicast traffic and how to properly negotiate mSLSs (e.g., total number of mSLSs, maximum bandwidth demand for each QoS class) for effective service differentiation. In this multi-service environment, we target at providing different service availability in terms of block ratio of group joins as well as bandwidth requirements. In our simulation we assume three differentiated service classes: Gold service, Silver service and Bronze service. We allocate the bandwidth capacity of each link evenly to the three QCs (10000 units each), while imposing a different limit on the total number of established mSLSs and maximum bandwidth demand within each class. Fundamentally, we adopt the strategy of under-provisioning through "multiplexing" traffic within the network instead of over-provisioning bandwidth resources. In our simulation we dimension the network in such a fashion that customers subscribing to higher QCs are able to enjoy better service (high bandwidth demand and low ratio of join blocks in time of congestion). One possible network configuration is as follows: Only 50 mSLSs are established for the Gold service, with maximum bandwidth demand up to 1000 units. For Silver and Bronze services, the number of signed mSLS is 100 and 200, with maximum bandwidth demand fixed at 700 and 400 units respectively. During our simulation, we create 10000 membership updates that cover all the group dynamics within each QoS class. From the following performance study we can see that the

![Figure 4-13 Network load vs. invocation ratio \( \omega \)](image-url)
The above configuration of network resources and mSLSs is an example scenario for successful service differentiation.

Figure 4-14 and Figure 4-15 capture respectively the real-time performance of network load and maximum link utilisation during the invocation period of the first 5000 events, as at this stage convergence has been achieved. From Figure 4-14 we can see that the network load of each QC is clearly differentiated, with the Gold class having the lowest traffic loading and Bronze the highest. With this result of service dimensioning, link congestions are less likely to occur in higher quality classes, which means that the blocking rate of new group joins is lower. In addition, the performance of maximum link utilisation in Figure 4-15 also shows that link congestions first take place in the Bronze class while the Gold class virtually does not suffer from link overloading at all.

Table 4-1 illustrates the performance of block ratio with the variance of mSLS invocation ratio $\omega$ (from 0.5 to 1.0) in the three QoS classes. From the table we can see that in time of network congestion due to a high ratio of mSLS invocations, the performance of block rate in join requests is significantly differentiated. On the other hand, when the average invocation ratio is relatively low, virtually no group joins are blocked, as the network is under-utilised in all QoS classes. This phenomenon is also in line with the principle that service differentiation normally takes effect when bandwidth resources are scarce within the network.
Chapter 4. DiffServ-aware Multicast Using QSSM

Figure 4-15 Real-time performance in max. link utilisation. (ω=1)

<table>
<thead>
<tr>
<th></th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gold</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.18%</td>
</tr>
<tr>
<td>Silver</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.44%</td>
<td>1.05%</td>
<td>4.0%</td>
</tr>
<tr>
<td>Bronze</td>
<td>0.10%</td>
<td>0.36%</td>
<td>0.71%</td>
<td>1.28%</td>
<td>3.87%</td>
<td>9.17%</td>
</tr>
</tbody>
</table>

Table 4-1 Multi-QC block ratio performance comparison

4.7 Summary

In this chapter we introduced the QoS aware Source Specific Multicast (QSSM) scheme, which aims at scalable implementation of DiffServ based multicast services in the control plane. By encoding QoS classes with dedicated SSM group addresses, QoS extensions to the legacy multicast routers as well as protocols such as PIM-SM and IGMP in the conventional solutions can be successfully avoided. We also studied the interaction between OMTE in the management plane and QSSM, representing the DMR functional block in the control plane. Our simulation results show that, by optimised link weight configuration computed by OMTE, the real-time performance of DMR in both single class and multi-class environment can be significantly enhanced in terms of resource utilisation and service availability.
Chapter 5

5 DQM: An Alternative QoS Approach

5.1 Introduction

We saw in Chapter 4 that the QSSM scheme constructs multiple trees for every group session, with each dedicated to one QoS class. Building specific multicast trees for each QoS class exhibits less complexity in QoS provisioning, as service dimensioning for different QCs is independent, without potential inter-class interactions. However, extra bandwidth resources are consumed as replicated group packets are delivered to multiple QoS classes in parallel. On the other hand, the performance of OMTE driven QSSM introduced in the previous chapters depends heavily on the accuracy of the multicast traffic matrix. Unfortunately, in some cases multicast group dynamics might not behave in accordance with the predicted traffic demand. This means that service provisioning through offline traffic engineering might not be always efficient due to the inaccuracy of traffic forecast and estimation. Another inefficiency in offline based TE is that it lacks resilience in terms of fast rerouting functionality when the network topology changes due to link failures.

To overcome these problems, we propose an alternative control plane approach named Differentiated QoS Multicast (DQM). In this scheme, we inherit from QSSM the feature of using multicast group address for encoding QoS classes, but what differentiates DQM from QSSM is that only one single multicast tree is constructed, spanning multiple QCs. From this point of view, the DQM approach belongs to the Hybrid QoS Multicast (HQM) family introduced in the previous chapter. The distinct advantage of this strategy is that, not only bandwidth resources but also multicast group states within the network can be conserved through QC tree merging. In addition, we adopt online routing schemes for dynamic construction of multicast delivery trees without interactions from management plane components. This feature exhibits significant advantages in cases when an accurate traffic matrix cannot be obtained a priori, or end users demand fast backup services in time of link failure.

The rest of the chapter is organised as follows: Section 5.2 introduces Multicast QoS (MQ), from which we inherit the strategy of building hybrid trees for provisioning heterogeneous QoS demanded by end users. Section 5.3 is dedicated to the overview of the DQM scheme, including both routing and forwarding mechanisms. In section 5.4 we conduct simulation analysis and
evaluate the performance of DQM and related approaches such as MQ and QSSM. Finally in section 5.5 we summarise this chapter.

5.2 Review of the MQ Approach

The basic strategy of the MQ approach is to construct a single multicast tree for end users with arbitrarily heterogeneous QoS requirements. Being an integrated solution, MQ sets up a multicast distribution tree with quantitative QoS requirements, and it makes explicit bandwidth reservation for each group member during the tree construction phase. In a similar fashion to the ReSource Reservation Protocol (RSVP) [15], when there exist heterogeneous receivers resources are reserved up to the point where the paths to different receivers diverge. When a join request propagates upstream towards the source, it terminates at the point where there is already an existing QoS reservation equal to or greater than the one being requested. Figure 5-1 illustrates how different resource reservations are merged along the multicast join procedure. Suppose the requests from receivers A, B and C demand 10Mbps, 512kbps and 56kbps bandwidth respectively, their reservations are merged to the highest request at each hop as shown in the figure. MQ can also adapt to resource consumption with dynamic group membership. For example, if an on-tree router detects that the departing receiver originally requested the highest QoS, it will automatically shrink its reservation or even reshape the distribution tree to exactly satisfy the remaining participants. In Figure 5-1(b), we can see that when receiver A with the bandwidth requirement of 10Mbps wants to leave the multicast session, the remaining receiver B with 512kbps requirement will switch from the original “shared” path (S R1 R2 R4) with the capacity of 10Mbps to a shorter one (S R3 R4) which still satisfies its QoS demand for bandwidth optimisation purposes. From the description above, we can also see that MQ adopts online routing for QoS adaptations.

On the other hand, the mechanism for network resource allocation works in an accumulative fashion, i.e., bandwidth is reserved in sequence for various incoming QoS requests until the link becomes saturated. This approach is straightforward and simple, but might not be efficient in bandwidth allocation, especially in the case of highly dynamic group membership. From a deployment point of view, each on-tree router needs to maintain not only group states but also quantitative QoS demands for its downstream receivers, and this imposes heavy overhead, in a similar fashion to RSVP.
5.3 The proposed DQM solution

5.3.1 Overview

In comparison to MQ, DQM is a DiffServ based paradigm that deals with finite classes of service (known as QoS channels) other than arbitrary QoS demands. Figure 5-2 presents the basic structure of a DQM tree with three QoS channels. In this tree, upstream links reflect the highest QoS channel requirements, while tree branches with lower QoS channels can be grafted from those with higher channels. This feature is similar to other schemes that belong to the HQM family (e.g., MQ and QUASIMODO). However, since QoS information has been embedded into the multicast class D address, as in QSSM, the maintenance of a DQM tree is achieved exclusively by using group states, which also conforms to the conventional SSM model. Take Figure 5-2 as an example: we can see that individual QoS channels are encoded with SSM group addresses respectively, e.g., G3 identifies Gold service, G2 for Silver service, etc. Tree branches with (S, G1) state can be grafted from those with either (S, G2) or (S, G3) states, which implies that Bronze tree branches are allowed to be extracted from Gold and Silver ones, while Silver branches can only be extracted from Gold ones. In comparison to QSSM, another significant difference is that we apply online bandwidth constrained routing schemes, other than conventional shortest path routing with optimised IGP link weights, for exploring feasible join paths of new group members. It should be noted that, although bandwidth constrained routing provides higher flexibility in exploring paths, it also needs support from dedicated signalling and explicit routing mechanisms. From this point of view, we can regard DQM routing as an overlay approach for QoS aware multicast service provisioning. For simplicity, we only focus on the working algorithm of the proposed DQM routing, while other auxiliary mechanisms such as QoS advertisement are not addressed in this thesis.
The advantages of the proposed DQM scheme are summarised as follows. First, like QSSM, it solves the fundamental conflict between the stateless DiffServ service model and the state-based IP multicast mechanism. Second, compared to the QSSM approach, network bandwidth as well as multicast group states can be conserved, as heterogeneous QoS trees are merged into one single delivery tree for each group. Moreover, the DSCP value for indicating QoS channels is not necessary as packet treatment can be directly enforced according to the group address being carried.

On the other hand, as QoS provisioning for one group session involves multiple QoS channels, it is relatively difficult to provide a sophisticated management-plane dimensioning scheme for DQM, as we have applied OMTE to QSSM.

### 5.3.2 DQM Forwarding

Once an intermediate router receives \((S, G)\) join requests with different values of \(G\) associated with various QoS channels from subscribers, it will merge all of them and will only send a single \((S, G_m)\) join request towards \(S\), where \(G_m\) is the class D address associated with the highest QoS channel being requested. Using this approach, a single tree is constructed for all QoS channels of a group session. Detailed descriptions of DQM group join and leave procedures will be presented in section 5.3.3. In accordance with the conventional SSM terminology, we still define the interface from which a join request is received as the outgoing interface (oif) and the one used to deliver unicast data to the source as the incoming interface (iif). When the router receives group data from its iif, it will take the following steps to forward the packets (shown in Figure 5-3, assuming \(QoS(G) > QoS(G')\)).
Chapter 5. DQM: An Alternative QoS Approach

(1) Check the group state(s) associated with the source S on each outgoing interface and replicate the packet where necessary.

(2) Copy the value of G contained in the (S, G) state of each outgoing interface to the IP destination field in the replicated packet (if the two are not consistent).

(3) Assign the data packet to the priority queue associated with the relevant QoS channel at the outgoing interface based on the (S, G) state.

Step (2) is necessary because the value of G contained in the packet indicates how this packet will be treated in the next on-tree router towards end hosts with heterogeneous QoS channel subscriptions. Remember that the group states are created by (S, G) join requests for different QoS classes, and the way data packets are treated in each router is uniquely identified by the value of G contained in the (S, G) state; in this way, data packets can be forwarded according to the QoS requirements of individual receivers. On the other hand, packets from different sources but with the same class D address in their (S, G) address tuples are treated aggregately in the corresponding queues. To achieve this, intermediate routers should be configured so that each priority queue is associated with a group address at outgoing interfaces (see Figure 5-4). This figure also illustrates how data from different sources but with a common group address is treated aggregately in a specific queue of an intermediate router.

Figure 5-3 Packet replication in DQM
5.3.3 DQM Routing

In this section we discuss how a source-specific DQM tree is constructed through online routing algorithms, according to dynamic group memberships. Specifically, we will discuss the mechanism of QoS channel subscription and unsubscription in the following two sub-sections.

5.3.3.1 QoS Channel Subscription

The construction of DQM trees is source specific. Once an end host R decides to join the DQM tree rooted at source S with a desired QoS channel, it will send an IGMPv3 \((S, G)\) group membership request to its Designated Router (DR) at the edge of the DiffServ domain, where \(G\) is the associated DQM group address mapped to the subscribed QoS channel. On receiving the membership report, the DR will submit to the source a plain SSM based PIM-SM \((S, G_i)\) join request that does not contain any extra QoS class information from the new group member \(R^6\). This join request will follow a feasible path with sufficient available bandwidth for supporting channel \(G_i\) towards the source \(S\), and this is done through online constraint-based routing with respect to the available bandwidth in that channel. One routing algorithm is described as follows: First, all the links without sufficient available bandwidth in the \((S, G_i)\) channel are pruned from the network, and then the router will perform shortest path routing based on the “reduced”

---

6 The DR on the LAN only sends one join request for the highest QoS channel when multiple group membership reports for different QoS channels appear on the LAN. When the multicast traffic flows back to the DR, the DR is responsible for forwarding replicated packets onto the LAN with different \((S, G)\) addresses that are remarked for heterogeneous group members with different QoS channel subscriptions.
network topology towards the source S. When the (S, Gj) join request reaches a router that has already received traffic from the source S with the same or higher QoS channel, i.e., with group state (S, Gj) where G \leq G_j^7, then the join procedure terminates and this interface is added to the list of group (S, Gj). Thereafter, data packets from S are replicated and forwarded to this interface with the class D address of the new packets modified from G_j to G_i. This way, a new tree branch is grafted from the current QoS channel that has equal or higher service level.

If the (S, Gj) join request reaches a router with the highest available QoS channel (S, G_i) where G_i > G_j (i.e., a router with lower QoS channel for S), the join will continue to explore a feasible path that satisfies the new requirement of the (S, Gj) channel subscription. Once a path with desired QoS channel has been set up and this particular router has received traffic from the (S, Gj) channel, it will tear down the (S, Gj) channel on the original path with lower QoS level. It should also be noted that the procedure of tearing down the (S, Gj) channel might invoke another internal join request from an on-tree router, where (S, G_j) is the highest local channel it maintains and there exist other channels with lower QoS. The flowchart for group join is presented in Figure 5-5, and it is worth mentioning that this flowchart also includes the steps for handling internal group joins invoked by QoS channel unsubscriptions, which will be specified later.

7 We assume that higher class D address is associated with higher QoS channel, i.e., G_i > G_j
QoS(G_i) > QoS(G_j)
In Figure 5-6, we assume that initially there already exists a single QoS channel constructed by \((S, G_2)\) subscriptions from both receivers R1 and R2 (Figure 5-6(a)). After some time router D receives a \((S, G_1)\) subscription from R3 where \(G_1 < G_2\), i.e., a subscription with a lower QoS channel. In this case D will send a join request towards S and this request will terminate at router B that has already received group data from S for a higher QoS channel (shown in Figure 5-6(b)).
In Figure 5-6(c), we assume that router E receives a (S, G3) join request from R4 where G3 > G2. In this case a new feasible path that satisfies the QoS demand of (S, G3) channel is constructed, shown with the solid line in the figure. When router C receives data traffic from S in the (S, G3) channel, it will tear down the original (S, G2) channel back to S. When router B has detected the pruning, it finds that it has also maintained a lower QoS channel for R3, namely (S, G1). Therefore, it will first send a (S, G1) join request back to S. When detecting that group data from S comes in the new channel (S, G1), router B will tear down the original (S, G2) channel on link AB as shown in Figure 5-6(d).

![Figure 5-6 Dynamic QoS channel subscription](image-url)
5.3.3.2 QoS Channel Unsubscription

Suppose that a particular router is currently receiving multicast traffic from source S with QoS channel \((S, G_i)\). When it detects no \((S, G_i)\) subscribers attached and wants to leave the channel, it will stop sending \((S, G_i)\) join requests towards the source S. When the \((S, G_i)\) state times out at the off, the upstream router will check all its oifs with QoS channels associated with S. There exist three possible cases as follows (illustrated in Figure 5-7):

1. There exists at least one \((S, G_j)\) state where \(G_j \neq G_i\), or there are other oifs for \((S, G_i)\); then the router will simply stop forwarding traffic on the \((S, G_i)\) channel at this timed out oif, and it will not need to take any further actions;

2. There does not exist any \((S, G_j)\) state where \(G_j = G_i\), and this interface is the only oif for \((S, G_i)\); then the router will check the status of all the remaining QoS channels associated with S, it will select the class D address \(G_{m}\) associated to the highest QoS channel currently requested and it will send an internal \((S, G_{m})\) join request towards the source S. Once this router has received data traffic from the \((S, G_{m})\) channel, it will stop sending \((S, G_i)\) join requests on its incoming interface. Special considerations are required for internal join requests invoked by channel unsubscriptions, and we will discuss this issue in detail using an example.

If this is the last subscriber for S, the router will simply stop sending any \((S, G)\) join request towards the source and hence it will break from the tree.

The flowchart for QoS channel unsubscription is provided in Figure 5-8.

We still follow the example in Figure 5-6 to illustrate the QoS channel unsubscription procedure. Starting from Figure 5-6(d), we assume that receiver R4 unsubscribes from the \((S, G3)\) channel, and we will show how DQM efficiently adapts the tree for the remaining group members. When router E notices this unsubscription, it finds out that the highest remaining active channel is \((S,
Chapter 5. DQM: An Alternative QoS Approach

G2) for R1, and hence it first sends a (S, G2) join request towards S. When the upstream router C detects that there currently exists a higher QoS channel (S, G3) on the interface from which this lower (S, G2) join request was received, it assumes that this downstream router E is downgrading its QoS requirement due to a high QoS channel unsubscription it has noticed. Meanwhile router C finds out that, after E has downgraded its requirement to (S, G2), the remaining highest channel becomes exactly (S, G2), and hence it will send an internal (S, G2) request towards S. (This procedure is also described in Figure 5-5 for internal group joins invoked by the relevant QoS channel unsubscription, identified by the * branch). We assume that router B is the next hop, and that its highest QoS channel is (S, G1), which is lower that what has been requested from router C. In this case, router B first forwards the (S, G2) request to the upstream router A, and once (S, G2) traffic comes from A, it will stop sending (S, G1) join requests on the same path, so that the (S, G1) channel will be deleted on link AB after the channel state times out. Once the (S, G2) traffic reaches router C from B, router C will stop sending (S, G3) join request to router F, so that in a similar fashion is pruned from the (S, G3) channel. Similarly, router F will prune itself from the tree since it is not receiving any join request for the source S. As a result, the adapted DQM tree is restored to that of Figure 5-6(b) after receiver R4 unsubscribes from the (S, G3) QoS channel.

It should be noted that the basic mechanism in this routing with loop-freedom guarantees applies also to other HQM schemes that follow the approach of building hybrid QoS trees. However, none of these schemes have investigated a detailed online QoS routing scenario according to group dynamics as we consider here. Moreover, QoS routing in those schemes needs not only group states but also extra QoS information (i.e. DSCP). This requirement introduces additional overhead to DiffServ core routers, as we have also previously indicated. Finally, it is worth mentioning that in DQM, a boundary router issues join/leave requests only when the first receiver for a new (S, G) session joins or the last member leaves the group. This strategy of pushing group management to the edge of the network reduces significantly the frequency of reshaping delivery trees within a domain.
Chapter 5. DQM: An Alternative QoS Approach

5.4 Simulation Study

We adopt the simulation scenario illustrated in last chapter, including network topology and group membership dynamics generation model. In our simulation, the average number of group members varies from 10 to 40 in steps of 5. We assume that the INP provides three QoS channels, namely Gold, Silver and Bronze, and that the subscription bandwidth for these three channels is 8Mbps, 4Mbps and 2Mbps per receiver respectively. Within the network, the bandwidth capacity of each link varies from 10Mbps to 45Mbps in an even distribution. The bandwidth capacity of each link is partitioned in the following proportion: 50% for Gold, 30% for Silver and 20% for Bronze respectively. Among all the receivers, we assume that 20% of them subscribe to Gold,
30% to Silver and 50% to Bronze. First of all, we investigate bandwidth conservation performance, and comparisons are made between DQM and QSM (e.g., QSSM, QUASIMODO) that applies separate trees in different QoS classes. Following that we focus on the capability of traffic engineering in terms of network utilisation between the DQM and MQ approaches. Finally, we compare the scalability in terms of memory overhead for group state maintenance between DQM and QSSM, as both embed QoS class information into group addresses.

We define the bandwidth conservation overhead for a particular channel C (C could be Gold, Silver or Bronze) as follows:

\[ O_C = 1 - \frac{U_{DQM}^C}{U_{QSM}^C} \]  

where \( U_{DQM}^C \) is the bandwidth utilisation of channel C by DQM, and \( U_{QSM}^C \) is that by using QSM schemes with independent QoS tree maintenance. Similarly, we define the overhead for all channels as:

\[ O_T = 1 - \frac{U_{DQM}}{U_{QSM}} \]  

where \( U_{DQM} \) is the overall link utilisation by DQM and \( U_{QSM} \) is that by QSM.

Figure 5-9 illustrates the overhead for both individual QoS channels and overall bandwidth conservation. We observe that in DQM bandwidth for non-gold channels can always be conserved and the corresponding overhead varies from 0.33 to 0.46. Obviously, bandwidth for the Gold channel is not conserved at any time, as it cannot be merged into any other QoS channel. Regarding the overall bandwidth conservation, we find that the aggregated overhead varies from 0.19 to 0.23, i.e., by using QoS channel merging in DQM, the average bandwidth consumption is 81.3% to 84% that of QSM approaches.
Another interesting empirical study is the traffic engineering capability of DQM and MQ in terms of link utilisation, bandwidth consumption, etc. In DQM, network bandwidth is pre-allocated to specific traffic aggregates of individual QoS channels, and this is very similar to the general strategy of DiffServ. In contrast, MQ/RSVP allows the overall bandwidth to be accumulatively reserved by QoS demands until the link has become saturated. In the following simulation, we examine the performance of load balancing in DQM and MQ/RSVP. According to bandwidth utilisation, we classify network links into the following three categories: (1) High load link with overall utilisation above 50%; (2) Medium load link with overall utilisation between 20% and 50%; and (3) Low load link with overall utilisation below 20%. Table 5-1 presents the proportion of these three types of links in the network with the average number of subscribers varying from 10 to 50. From the table we can see that DQM performs better in terms of load balancing since traffic is more evenly distributed. For example, when the average number of subscribers is below 30, none of the network links become highly loaded in DQM. In contrast, MQ always results in hotspots with utilisation above 50% even when the average number of subscribers is 10. From the table we can also see that the proportion of low load link in DQM is consistently higher than that in MQ.
Table 5-1  Traffic distribution comparison with MQ

<table>
<thead>
<tr>
<th>Number of subscribers</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DQM</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High load link</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.07%</td>
<td>0.12%</td>
</tr>
<tr>
<td>Medium load link</td>
<td>1.2%</td>
<td>2.6%</td>
<td>4.1%</td>
<td>4.9%</td>
<td>5.4%</td>
</tr>
<tr>
<td>Low load link</td>
<td>98.8%</td>
<td>97.4%</td>
<td>95.9%</td>
<td>95.0%</td>
<td>94.5%</td>
</tr>
<tr>
<td><strong>MQ</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High load link</td>
<td>0.23%</td>
<td>0.41%</td>
<td>0.86%</td>
<td>1.33%</td>
<td>1.58%</td>
</tr>
<tr>
<td>Medium load link</td>
<td>1.7%</td>
<td>3.1%</td>
<td>4.1%</td>
<td>4.7%</td>
<td>6.7%</td>
</tr>
<tr>
<td>Low load link</td>
<td>98.1%</td>
<td>96.5%</td>
<td>95.0%</td>
<td>94.0%</td>
<td>91.7%</td>
</tr>
</tbody>
</table>

We also investigate the overall link utilisation of DQM and MQ, and the simulation results are presented in Figure 5-10. From the figure we can see that the average link utilisation of DQM is consistently higher than that of MQ by a small margin, e.g., when the average number of subscribers is fixed at 50, the link utilisation of DQM is 4.7% higher than that of MQ. From the empirical results in Table 5-1 and Figure 5-10, we can infer that the better performance of DQM's load balancing is in effect at the expense of higher bandwidth consumption, but the relevant cost is very small (i.e., up to maximum 5% higher than MQ).

![Figure 5-10 Overall link utilisation comparison with MQ](image-url)

In addition to the previous evaluation based on traffic characteristics, we also investigate the scalability aspect in terms of memory consumption for group state maintenance. We scope the comparison between approaches that use group states to identify differentiated QoS classes, i.e.,
DQM and QSSM. Figure 5-11 shows the average number of channel states that are maintained at each router, i.e., the total number of forwarding entries (expressed in equation 4.1) in the network divided by the number of routers. From the figure we can see that the number of channel states needed per router increases as the group size grows. On the other hand, by using QoS channel merging in DQM, the burden of maintaining group states can be alleviated significantly, e.g., when the number of receivers is fixed at 40, the router memory overhead of using DQM channel merging is 83.5% that of QSSM, which needs dedicated trees for each QoS class. In the extreme case, in DQM the ingress router for the source S only maintains one (S, G) state, where G corresponds to the highest QoS channel requested from all the downstream receivers. In contrast, QSSM requires that the first hop router of the source maintain as many group states as the total number of QoS classes being subscribed. Figure 5-12 depicts the memory overhead for maintaining individual QoS channel states in both schemes. We can see that the total number of states for the Gold service is exactly the same in QSSM and DQM. This is because the branches for the highest QoS channel cannot be grafted onto any other tree. Given the same group subscription scenario, QSSM and DQM always form an identical tree shape for the Gold channel. On the other hand, by comparing (a) and (b) in Figure 5-12, we also notice that DQM is able to conserve group states for lower class channels, namely the Silver and Bronze classes in our simulation. For example, when the average number of subscribers reaches 40, the number of group states for the Silver channel in DQM is 80% that in QSSM, and for the Bronze channel the corresponding value is 72.1%.
In this chapter we introduced the Differentiated QoS Multicast (DQM) scheme, aiming at providing an overlay control plane mechanism for multicast QoS heterogeneity. The key idea of DQM is to construct a hybrid delivery tree per group with heterogeneous QoS classes instead of maintaining multiple trees for individual QCs. With this strategy, benefits come not only from better performance in terms of bandwidth conservation but also higher scalability in terms of router memory requirement. In addition, online constraint-based routing provides DQM the
flexibility of exploring in real-time join paths with QoS requirements, and this is in contrast to the
OMTE-based approach that relies on accurate traffic forecast mechanisms a priori.

In our future work, we will also investigate how DQM trees can be constructed across multiple
domains for the provisioning of inter-domain QoS multicast services, as it has been done in
QSSM.
6 Multicast Sender Access Control

6.1 Introduction

In addition to the QoS provisioning schemes introduced in the previous chapters, protection of network resources from Denial of Service (DoS) attacks is also indispensable. As has recently been indicated in [3, 66], any new research on QoS mechanisms or architecture ought to specifically address potential security issues. Deploying existing QoS mechanisms (e.g., DiffServ) across an inter-domain boundary creates a significant and easily exploited DoS vulnerability for any network that provides inter-domain QoS. Having this in mind, in this chapter we propose fundamental anti-DoS mechanisms, at both intra- and inter-domain level. We in fact address the issue of multicast sender access control in the IP multicast service model. As we have already explained, IP multicast is also known as “Any Source Multicast (ASM)” since any information source, even outside a group, can send data to a multicast address without any control mechanism. This means that in the current service model group management is not stringent enough to control both senders and receivers. While ASM provides a highly flexible architecture for group communications, it is also vulnerable to malicious sources without a valid mSLS in terms of DoS attacks. The weakness in controlling the behaviour of sources has been improved in the Source Specific Multicast (SSM) to some extent. In SSM each group is identified by an address tuple (S, G) where S is the unique address of the information source and G is the destination channel address. A single multicast tree is built, rooted at the well-known source for delivering data to all subscribers. In this situation, centralised group authorisation and authentication can be achieved at the root of the single source through application level mechanisms.

On the other hand, there exist many other applications based on a many-to-many or even a peer-to-peer communication model, such as multi-party videoconferencing, Distributed Interactive Simulation (DIS), online Internet games, etc. For this type of applications, bi-directional shared trees, such as Core Based Tree (CBT) [7], Bi-directional PIM (Bidir-PIM) [37], and RAMA style Simple Multicast [58, 59], are efficient routing schemes for delivering data among peering hosts. Figure 6-1 illustrates the difference between uni- and bi-directional multicast routing: assuming that router C is the core/root of the multicast tree, in uni-directional routing each information source should unicast its data to C from where packets are forwarded to all receivers. In contrast,
bi-directional routing allows traffic from sources to be delivered directly to group members without necessarily having to pass through the core; this results not only in smaller end-to-end delay and lower bandwidth consumption, but also in lower memory overhead for maintaining (S, G) states for individual sources. However, since there is no single point for centralised group access control, sender authorisation and authentication become difficult challenges. For example, an invalid host without an established mSLS may attempt a DoS attack by flooding bogus data from any point of the bi-directional shared multicast tree. Sender access control for bi-directional trees in IP multicast is not catered for in the specification of any of the corresponding routing protocols. In addition, it is not known if Source Specific Multicast can be extended to bi-directional multicast routing, hence the source filtering function of IGMPv3 may not apply to the underlying protocols. A potential solution that has been proposed in the literature is to periodically "push" the entire sender access list down to all the on-tree routers, so that only data from authorised senders can be accepted and forwarded to the bi-directional tree (Full Policy Maintenance, FPM [16, 58]). This simple access control mechanism has also been adopted in RAMA-style Simple Multicast. However, this approach is not scalable, especially when many multicast sessions are active and/or large group sizes with many senders are involved.

In this chapter we investigate the scalability issue of the FPM scheme, and propose an efficient and scalable sender access control mechanism for bi-directional trees in IP multicast. It should be noted that the underlying assumption on trust model is exactly the same as that in [16] and [58]. The basic idea is to deploy access policy for authorised senders (typically with a valid mSLS) on the tree routers only where necessary, so that traffic from unauthorised senders is policed and discarded once it arrives on the bi-directional shared tree. Our scheme has little impact on the current bi-directional routing protocols and can be directly implemented without modifying the
basic function of the current routing protocols. Moreover, the memory overhead introduced is much smaller than that proposed in [16] and [58].

The rest of the chapter is organised as follows: Section 6.2 gives the overview of our dynamic maintenance of the access control policy. Sections 6.3 and 6.4 introduce sender authorisation and authentication in both intra- and inter-domain scenario. Considerations for multi-access networks are especially discussed in section 6.5. In section 6.6 we assess the scalability of our proposed scheme and we finally present a summary in section 6.7.

6.2 Sender Authorisation and Authentication Overview

6.2.1 Sender Access Control Deployment

Compared with source specific trees and even uni-directional shared trees such as conventional PIM-SM, in which external source filtering can be performed at the single source or Rendezvous Point (RP) where the registrations of all the senders are processed and authorised, in bi-directional trees this is much more difficult since data from any source is directly forwarded to the whole tree once it hits the first on-tree router. In effect, since there is no single point for centralised sender access control, source authorisation and authentication has to be deployed at the routing level. As we have already mentioned, the simplest solution for this is to periodically broadcast the entire access control list down to all the routers on the bi-directional tree for deciding whether or not to accept data. However, this method is only feasible when a few small-sized groups with limited number of senders are considered. For large scale multicast applications, if we do not send the whole policy to all the on-tree routers for scalability reasons, three questions need to be answered as stated in [16]: (1) How to efficiently distribute the list where necessary? (2) How to find edge routers that act as the trust boundary? (3) How to avoid constant lookups for new sources? In fact if we try to statically mount the access control policy to an existing bi-directional multicast tree, none of the above three questions can be easily answered.

It should be noted that most multicast applications are highly dynamic in nature, with frequent join/leaving of group members and even information senders. Hence the corresponding control policy should also be dynamically managed. Here we propose an efficient sender-initiated distribution mechanism of the access control list during the phase of multicast tree construction. The key idea is that each on-tree router only adds its downstream senders to the local Sender Access Control List (SACL) during their join procedure, and the senders in the access list are activated by a notification from the core. In fact, only the core has the right to decide whether or not to accept the sources and it also maintains the entire SACL for all the authorised senders. Packets from an unauthorised host (even if it is actually on the tree) will be discarded once they
reach any on-tree router. To achieve this, all senders must first register with the core before they can send data to the group. When a registration packet hits an on-tree router, the unicast address of the sender is added to the SACL of each router on the way. Under this scheme, the access policy for a particular sender is deployed on the branch from the first on-tree router where the registration is received along to the core router. We define the interface from which this registration packet is received as the *Downstream Interface (DI)* and the one used to deliver unicast data to the core as the *Upstream Interface (UI)*. The format of each SACL entry is \((G, S, I)\) where \(G\) indicates the group address, \(S\) identifies the sender and \(I\) is the downstream interface from which the corresponding registration packet was received. Once the registration reaches the core, the latter will contact a Source Authorisation Server (SAS) for deciding whether to accept the new sender. If the SAS has approved the join, the core will send an “activating packet” back to the source, and once each on-tree router receives this packet, it will activate the source in its SACL so that it is able to send data to the bi-directional tree from then on. In such a scenario, an activated source can only send group data to the tree via the path where its SACL entry has been recorded, i.e., even if a sender has been authorised, it cannot send data to the group from other branches. Source authentication entries kept in each SACL are maintained in soft state for flexibility, and this requires that information sources should periodically send “refresh” packets up to the core to keep their state alive in the upstream routers. This action is especially necessary when a source is temporarily not sending data for a period. When data packets are received from a particular sender, the on-tree router can assume that this source is still alive and will automatically refresh its state. If a particular link between the data source and the core fails, the corresponding state will time out and become obsolete. In this case, the host has to seek an alternative path to perform re-registration for continuing sending group data.

One common assumption is made for both the proposed scheme and the FPM solution: adjacent routers always trust each other within each domain. On the other hand, the difference between our dynamic access control scheme and the Full Policy Maintenance mechanism (FPM) is illustrated in Figure 6-2, parts (a) and (b) respectively. It should be noted that, only the on-tree routers having received the sending request from the new source \(h\) (in grey colour) need to maintain the policy for \(h\). This is more scalable compared with the approach in which all on-tree routers keep the entire sender list. However, this requires that the sender should send multicast data to the bi-directional tree *only* from the designated ingress router (router A in Figure 6-2).
6.2.2 Data Authentication and Forwarding

When a router receives a data packet from one of its downstream interfaces, it will first check if there exists such an entry for the data source in its local SACL. If the router cannot find a matching entry that contains the unicast address of the source, the data packet is discarded. Otherwise if the corresponding entry has been found, the router will verify if this packet comes from the same interface as the one recorded in the SACL entry. Only if the data packet has passed these two authentication mechanisms, it will be forwarded to the upstream interface and the other interfaces with the group state, i.e., interfaces where receivers are attached. On the other hand, when a data packet comes from the upstream interface, the router will always forward it to all the other interfaces with group state without performing any authentication. Although the router cannot judge if this packet is from a registered sender since it comes from the upstream router, there exist only two possibilities: either the upstream router has the SACL entry for the data source or it has received the packet from its own parent router in the tree. The extreme case is that none of the intermediate ancestral routers have such an entry and then we have to backtrack to the core. Since the core has recorded entries for all the registered senders and it never forwards any unauthenticated packet on its downstream interfaces, we can safely conclude that each on-tree router can trust its parent, hence packets received from the upstream interface are always from valid senders. By maintaining such a trust chain residing on the bi-directional routing tree, sender access control for information sources can be achieved in a scalable fashion. However, this does not include the case of routers attached on multi-access networks such as LANs, we will discuss the relevant considerations and required additional operations in section 6.5.
6.3 Intra-domain Access Control Policy

6.3.1 Sender Access Control List Construction and Activation

In the IP Multicast architecture, information sources and receivers (referred to as group members) are treated separately. However, in many interactive applications, a host may act in both roles simultaneously. In this section, we classify senders of a multicast session as follows: if a host wants to be both sender and receiver, it must join the multicast group and become a Send-Receive capable member (SR-member, SRM). Otherwise if the host only wants to send data to the group without receiving any, it may choose to act either as a Send-Only member (SO-member, SOM) or a Non-Member Sender (NMS). In the former case, the host must join the bi-directional tree in order to send data, and its Designated Router (DR) will forward the packets on the upstream interface as well as other interfaces with the group state (if any). In the IP multicast model, information sources are allowed to send data to the group without becoming members. Hence, if the host is not interested in the information from the group, it may also choose to act as a non-member sender. In this case, the host must unicast its data packets towards the core. Once the data packet hits the first on-tree router and passes the corresponding source authentication, it will be forwarded to all the other interfaces with group state. We discuss below the exact mechanism for SACL construction and activation; the description is based on the CBT routing protocol but it can also apply to other bi-directional routing schemes such as Bidir-PIM and Border Gateway Multicast Protocol (BGMP). A detailed flowchart for SACL construction and activation on each router is presented in Figure 6-3, while SACL-based data authentication and forwarding is shown in Figure 6-4.
In the flow branch of handling notification packets, we use A to identify the interface from which the notification packet is received whereas B to identify a particular downstream interface with pending downstream SR/SO members.

UI and DI stand for upstream/downstream interface respectively.

Figure 6-3 Flowchart for Router Actions in SACL Management
Chapter 6. Multicast Sender Access Control

Figure 6-4 Flowchart for Router Actions in SACL Management

(1) SR-member Join

When the Designated Router receives a group G membership report from a SR-member S on the LAN, it will send a join request towards the core. Here we note that the group membership report cannot be suppressed at the DR if it is submitted from a send-capable member. Once a router receives this join-request packet from one of its interfaces, say, A, then the (G, S, A) entry is added to its SACL. If the router is not been on the shared tree, a (*, G) state is created with the interface leading to the core as the upstream interface and A is set to the downstream interface. At the same time interface A is also added to the DI list (i.e., oif list of group G in the M-FIB) for group G, so that data from other sources can be forwarded to S via A. If the router already has (*, G) state, but A is not in the interface list with group state, then it is added to the DI list. Thereafter, the router just forwards the join-request to the core via its upstream interface based on the underlying unicast routing table. On the other hand, if the router receives any join packet from its upstream interface, the packet will be dropped.

(2) SO-member Join

For SRM and SOM joins, the DI list is interchangeable with oif list, and UI is in effect if.

105
In a similar fashion to SR-member joins, the DR of a SO-member also sends a join-request up to the core and when the router receives this request from its interface A, the (G, S, A) entry is added to the local SACL. If the router is not yet on the tree, (*, G) state will be generated but interface A is not added to the DI/oif list for group G. This guarantees that A will not forward group data to a send-only member S later on.

(3) Non-Member Sender (NMS) Registration

Here we use the terminology “registration” instead of “join request”, since this host is not a group member and does not need to be on the tree in order to send group data. The registration packet from the Non-Member Sender is unicast towards the core and when it hits the first router with (*, G) state, the (G, S, A) entry is created in the local SACL of all the on-tree routers on the way to the core. It should be noted that if a router is not on the tree, it does not maintain SACL for the group even if it has received the registration.

Finally, if a receive-only member (also known as the group member in the conventional IP multicast model) wants to join the group, the join request invokes a (*, G) state if the router is not on the tree, but no new SACL entries need to be created. Moreover, once the join request hits any on-tree router, a join-notification is immediately sent back (necessary for hard-state based routing protocols) without informing the core.

Once each on-tree router receives the activating notification from the core, the (G, S, A) entry is activated so that data from S can be forwarded on this router. Thereafter, the router will forward this notification packet exclusively on interface A (the interface from which the original join request/registration was received) for activating the corresponding SACL on its downstream routers. Route selection of notification packets is not shortest path routing, but is based on the interface where the pending join request/registration is attached. Hence interface A might not be on the shortest path back to the sender S if the network link metric is asymmetric. This type of forwarding guarantees that the notification packet will finally follow the reversed path back to reach the source’s designated router. It is also worth mentioning that in the following two cases the notification is not valid and should be discarded: (i) There is not a matching (G, S, A) entry and (ii) the notification packet does not come from the upstream interface. The second is necessary, because notification packets should exclusively originate from the core, and they must not appear on any interfaces except the upstream interface.

We have to mention that the proposed SACL mechanism is not able to prevent flooding attacks by spoofing. For example, if S forges and uses the IP address of a remote authorised source, on-tree routers with the associated SACL entry will not be able to identify this type of IP spoofing. It should be noted though that even the full policy maintenance mechanism is not able to solve this
problem either: relevant solutions should be implemented at lower levels and are outside the scope of this work.

### 6.3.2 An Example for Intra-domain Access Policy

A simple network model is shown in Figure 6-5(a). We assume that node A is the core router and all the Designated Routers of potential members of group G should send join request to this node. Hosts H1-H5 are attached to the individual routers as shown in the figure.

Initially we assume that H1 wants to join the group. Its DR (router B) will create (*, G) state and send the join request to the core A. Since H1 is an SR-member that can both send and receive data, each of the routers from which the join request has passed will add this sender to its local SAACL. Hence both routers B and A will have the SAACL entry (G, H1, 1), since they both receive the join request from interface 1. Host H2 wants to only send data to group G, so it may choose to join as a SO-member or just act as a NMS. In the first case, its DR (router C) will create (*, G) state indicating that this router is on the tree and then add H2 to its SAACL. Thereafter, router C will send a join request indicating H2 is a SO-member towards the core; when B receives this request, it will also add H2 to its local SAACL and then forward the join-request packet to A. Since H2 does not want to receive data from the group, the link BC becomes a send-only branch. To achieve this, router B will not add B3 to the interface list with group state. If H2 chooses to act as the Non-Member Sender, router C will not create (*, G) state or SAACL for the group but will send a registration packet towards A. When this packet hits an on-tree router, i.e. B in our example, H2 will be added to the local SAACL of all the routers on the way. When sending group messages, router C needs to unicast the data to the core by setting the corresponding IP destination address to A. When the data reaches B and passes the SAACL authentication, the IP destination address is changed to the group address originally contained in the option field of the data packet, and the message is forwarded to interfaces B1 and B2 to get to H1 and the core respectively. After H3 and H4 join the group, the resulting shared tree is shown in Figure 6-5(b) with the SAACLs of each on-tree router. It should be noted that H4 is a receive-only member, and hence routers B, F and A need not add it to their local SAACLs. Suppose router F has received group data from H3 on interface F3, it will check in its local SAACL if H3 is an authorised sender. When data passes the address and interface authentications, it is forwarded to both interfaces F1 and F2. When group data is received on the upstream interface F1, since its parent A is a trusted router (the data source should be either H1 or H2), it is forwarded to F2 and F3 immediately without authentication. However, if the non-registered host H5 wants to send data to the group, this will not be forwarded to the bi-directional tree due to the SAACL authentication failure at router F.
6.4 Inter-domain Access Control Policy

6.4.1 Basic Descriptions

As we have mentioned above, on-tree routers only maintain the access policy for the downstream senders. However, if large-scale groups with many senders or many concurrent sessions are considered, the size of the SACL in the routers near the core will become a heavy burden for those routers. In this section we discuss how this situation can be improved with the aid of inter-domain IP multicast routing semantics.
Chapter 6. Multicast Sender Access Control

Our idea is based on hierarchical access control policy to achieve scalability. Routers only maintain SACL for the downstream senders in the local domain and do not need to add sources from downstream domains to their local SACLs. In other words, all the senders for the group are only authenticated in the local domain. In the root domain, the core needs to keep entries only for its local senders; however in order to retain the function of authorising and activating information sources from remote domains, on receiving their registrations the core router needs to contact the source authorisation server residing in the local domain, which decides whether or not to accept the requests.

For each domain, a unique border router (BR) is elected as the “policy agent” and keeps the entire SACL for all the senders in the local domain, and we name this the Designated Border Router (DBR) for the domain. In fact the DBR can be regarded as core of the sub-tree in the local domain, and the common practice for this is to set the best exit BR towards the root domain as the DBR. In this sense, all the data from an upstream domain can only be injected to the local domain from the unique DBR and all the senders in this domain can only use this DBR to send data up towards the core. This mechanism abides to the “3rd party independence” policy in that data from any sender must be internally delivered to all the local receivers without flowing out of the domain. This requires that joins from different hosts (including both senders and receivers) merge at a common point inside the domain. In BGP-4, all the edge routers of a stub domain know for which unicast prefix(es) each of them is acting as the egress router, this satisfies the above requirement of “path convergence” of internal joins we just mentioned.

Since individual sender authentication is performed within each domain and invalid data never gets any chance to flow out of the local domain, the on-tree BR of the upstream domain will always trust its downstream DBR and will assume that all the data packets coming from it originate from authorised senders. Hence, when a packet leaves its local and enters remote domains, no further authentication is needed. This also avoids constant lookups when the authenticated data is travelling on the bi-directional tree.

6.4.2 Inter-domain SACL Construction and Activation

Since Border Gateway Multicast Routing (BGMP [73]) is considered as the long-term solution to inter-domain multicast routing, in this section we will take BGMP as an example to illustrate how sender access control policy can be deployed in inter-domain applications.

First we discuss how the Designated Router for a group member sender submits its join request and how it is added to the SACL and activated. This applies to both SR-members and SO-members, the only difference between the two being whether or not to add the interface from which the join-request was received to the ofl list of the group in the M-FIB. Only if an on-tree
router receives a join request from a sender in the local domain, it will add this sender to its SACL, otherwise the router will just forward the join request towards the core without updating its local SACL. In the transit domain, in order for the intermediate routers to know how to deliver correctly the activation notification on the reversed path back to remote senders, they need to create temporary SACL states for them. Once the activation packet has been received from the core and delivered back to the remote sources, this type of transit states are deleted.

In Figure 6-6, when host S wants to become a S0-member to send data, its DR (router A) sends a join request towards the DBR router B, which has the best exit to the root domain. All the internal routers receiving this request will add S to their local SACLs. Since B is the core of the sub-tree for the local domain, it also needs to create a SACL entry for host S once it receives the join request from its Multicast Interior Gateway Protocol (M-IGP, e.g., Bidir-PIM) component. Thereafter, B finds in its Group Routing Information Base (G-RIB) that the best route to the root domain is via its external peer C in the transit domain, so router B will send the BGMP join request towards C via its BGMP component. Once router C receives the join request, it creates (*, G) state (if it has not been on the tree), as well as a transit entry for S in its local SACL. When C finds out that the best exit toward the root domain is D, it just forwards the join request to this internal BGMP peer, and hence router D becomes the DBR of the transit domain for group G. Suppose Bidir-PIM is the M-IGP, the RP in this transit domain should be placed at D, and router C will use its M-IGP component to send the join request towards D. When this join request travels through the transit domain, all of the internal routers along the way create a transit SACL entry for S. After the join request reaches the root domain and the core router F authorises the new sender by contacting the access control server and sends back the activating-notification, all the on-tree routers (including internal on-tree routers and the DBR) in the transit domain just forward it back (based on the transit SACL entry for S) towards the local domain where the new sender S is located. After the notification packet has been forwarded on the interface leading back towards S, the transit state is deleted. When the packet enters the local domain, all the on-tree routers (i.e., B and A in Figure 6-6) will activate S in their SACLs.

![Figure 6-6 Inter-domain join and activation](image)

**Figure 6-6 Inter-domain join and activation**
As we have mentioned previously, a send-only host may also choose to act as a Non-Member Sender (NMS). However, there are some restrictions when inter-domain multicast routing is involved. If a send-only host is located in the domain where there are no receivers (we call this domain a send-only domain), then the host should join the bi-directional tree as a SO-member other than a Non-Member Sender (NMS). Otherwise, if the host acts as a NMS, its registration packet will not hit any on-tree router until it enters remote domains. This forces the on-tree router there to add a sender from another domain to its local SACL, which does not conform to the rule that on-tree routers only maintain access policy for senders in the local domain. On the other hand, if the host joins as a SO-member and since its DR will be on the tree, the authentication can be achieved by the on-tree routers in the local domain. It should be noted that for any on-tree routers in the send-only domain, the interface from which the join request for the SO-member is received is not added into the group's oif list (which is always empty in a send-only domain for the group), and hence group traffic will not flow into the local domain at any time.

### 6.4.3 An Example for Inter-domain Access Policy

An example for inter-domain sender access control is given in Figure 6-7. C is the core router and domains X, Y and Z are remote domains with respect to the core C. Hosts a, b, c and d are attached to the routers in different domains. Also suppose that host a only wants to receive data from the group, hosts b and c want to both send and receive, while host d only wants to send data to the group. In this case, X is a receive-only domain and Z is a send-only domain. X1, Y1 and Z1 are border routers that have been selected as the DBR for each domain. According to our inter-domain access control scheme, on-tree routers have the SACL entry for downstream senders in the local domain, and each DBR has the policy for all the senders in the local domain. Hence, Y1 has the entry for hosts b and c in its SACL while the SACL of X1 contains no entries at all. Although X is the parent domain of Y and Z which both contain active senders, all the on-tree routers in X do not need to add these remote senders to their SACLs. In fact, data from Y and Z has already been authenticated by their own DBRs (i.e. Y1 and Z1) before it flows out of the local domains. Since host d only wants to send data to the group and there are no other receivers in domain Z, host d should join as a send-only member. Otherwise, if d acts as a non-member sender and sends its registration packet towards the core, this makes the first on-tree router (X2) add d to its SACL, but this is not scalable because on-tree routers are forced to add senders from remote domains. On the other hand, if host d joins as a send-only member, the shared tree will span to its DR, i.e. Z2, and then the authentication can be performed at the routers in the local domain.
BGMP provides also the mechanism for building source-specific branches between border routers. Now we assume PIM-SM is used for constructing multicast delivery trees in domain Y shown in Figure 6-7. At a certain time the DR in domain Y, i.e. Y3 or Y4, may wish to receive data from host d in domain Z via the shortest path branch instead of the current shared tree. Hence (S, G) state is originated and passed to the border router Y5, which is the best exit to domain Z, but not the DBR of domain Y and also not on the shared tree. When Y5 receives the source specific join, it will create (S, G) state and then send the corresponding BGMP source specific join towards Z1. On the other hand, since Z1 is the DBR of domain Z, intra-domain sender authentication has been performed before the traffic is sent to Z1’s BGMP component for delivery to remote domains. In fact, Y5 will only receive and accept data originated from host d in domain Z due to its (S, G) state filtering. Once Y5 receives the data from host d, it can directly forward the packets to all the receivers in the local domain, as the RPF check can be passed. When the DR receives the data from d via the shortest path branch, it will send a source specific prune message up towards the root domain to avoid data duplication. From this example, we can also see that source specific tree can also interoperate with the proposed sender access control in the receiver’s domain (note that the M-IGP in domain Y is not a bi-directional).

Figure 6-7 Example for Inter-domain sender access control

6.5 Operations on Multi-access Networks

Special consideration is necessary for protecting group members from unauthorised sources attached to multi-access networks such as LANs. As we have mentioned, if an on-tree router receives data packets from its upstream interface, it will always forward them to all the other
interfaces with group state, assuming that these come from an authorised information source. However this may not be the case if the upstream interface of an on-tree router is attached to a broadcast network. When an unauthorised host wants to send data with group address to the multi-access LAN, a corresponding mechanism must be provided to prevent these packets from being delivered to all the downstream group members. To achieve this, once the Designated Router (DR) on the LAN receives such a packet from its downstream interface, if it cannot find a matching access entry for the data source in its SACL, it will discard the packet, and at the same time it will send a “forbidding” control packet containing the unicast address of the unauthorised host to the LAN from its downstream interface. Taking the CBT routing protocol as an example, the IP destination address of this forbidding packet should be “all-cbt-router address (224.0.0.15)” and the value of TTL is set to 1. Once the downstream router receives this packet on its upstream interface, it will stop forwarding the data with this unicast address that originates from an unregistered host attached to the LAN. Hence all the downstream session members will only receive little amount of unauthorised data for a short period of time. In terms of implementation, the downstream on-tree routers should maintain a “forbid list” of unauthorised hosts recorded. Since all the possible unauthorised hosts can only originate from the local LAN, this list should not introduce much overhead to the routers. In Figure 6-8, we assume that the unauthorised host S sends data to the group. When the DR (router A) cannot find the corresponding entry in its local SACL, it immediately discards the packet and sends a “forbidding” packet containing the address of S to the LAN. Once the downstream router B receives the forbidding packet, it will stop forwarding data coming from S. If S sends data by maliciously using a different network prefix, both routers A and B will notice that the source address is not contained in their SACL list, or the data does not come from the correct interface; as such, they will not forward relevant packets which means that this type of malicious flooding will be only restricted to the local network.

Figure 6-8 Access control operation on LANs
With respect to inter-domain routing, further consideration is necessary for data travelling towards the core. This is because routers in transit domains do not have entries for remote senders in their SACLs. Take Figure 6-8 as an example and suppose that the LAN is located in a transit domain where there are no local authorised senders, hence router A’s SACL is empty. If there is data appearing on the LAN destined to the group address, there are only two possibilities: (1) the data came from a downstream domain and was forwarded to the LAN by router B; (2) a local unregistered host attached to the LAN (e.g., host S) sent the data. It is obvious that in the former case router A should pick up the packet and forward it towards the core, and for the latter, it should just discard the packet and send the corresponding "forbidding" packet to the LAN. This requires that the router is able to distinguish between packets coming from remote domains and packets coming from hosts directly attached to the LAN, which can be easily done by checking the source address prefix.

### 6.6 SACL Scalability Analysis

#### 6.6.1 Simulation Scenario

In this section we discuss scalability issues regarding router memory consumption in both intra- and inter-domain scenario. It is obvious that the maximum memory space needed in maintaining a SACL is $O(ks)$ where $k$ is the number of multicast groups and $s$ is the number of senders in the group. In fact, this is exactly the size of SACL in the core router. However, since on-tree routers need not keep the access policy for all sources but only for downstream senders, the average size of SACL in each on-tree router is significantly smaller.

We can regard the bi-directional shared tree as a hierarchical structure with the core at the top level, i.e., level 0. Since each of the on-tree routers adds its downstream senders to its local SACL, then the SACL size $S_i$ of router $i$ in the shared tree $T$ can be expressed as follows:

\[
S_i^H = \sum_{(i,j) \in T} S_j^{H+1}
\]  

and the average SACL size per on-tree router is:

\[
\overline{S} = \frac{\sum_{i=0}^{H} \sum_{j=0}^{L_i} S_j^H}{\sum_{i=0}^{H} Y_i}
\]  

where $H$ is the number of hops from the core to the farthest on-tree router (or maximum level) and $L_i$ is the number of routers on level $i$, while
Chapter 6. Multicast Sender Access Control

\[ Y_i = \begin{cases} 
1 & \text{if router } i \text{ is included in the shared tree} \\
0 & \text{otherwise} 
\end{cases} \]  \hfill (6.3)

For the intra-domain network topology generation, we still apply the Waxman's model provided in GT-ITM, with receivers randomly distributed within the network. Nevertheless, we notice that the scalability of our proposed solution in the Internet largely depends on the shape of the distribution tree. Taking this fact into consideration, we will analyse in our future work the relevant performance of the proposed scheme based on existing research achievements in the multicast tree topology over the Internet [19].

In addition, in order to evaluate the performance of the proposed mechanism at a larger scale, we also conducted the simulation based on inter-domain routing scenarios. In our simulation model, 100 routers are contained in each of the 11 peering domains (shown in Figure 6-9), and hence altogether 1100 nodes are involved in the inter-domain simulation model. In our experiment we assumed that the core of the inter-domain bi-directional tree is located in Node 0.

![Figure 6-9 Inter-domain simulation model](image)

6.6.2 Intra-domain Scalability

First of all, we study the relationship between average SACL size per on-tree router and total number of senders. In the simulation we generate a random network with 100 routers with the core router also being randomly selected. The total number of senders varies from 10 to 50 in steps of 10, while the group size is fixed at 50. We study three typical situations regarding the sending host type:

(1) All senders are also receivers (AM);

(2) 50% senders are also receivers (HM);
(3) None of the senders are receivers (NM).

All send-only hosts choose to act as Non-Member Senders (NMS) without joining the bi-directional tree.

From Figure 6-10 we can see that the average SACL size grows as the number of senders increases. However, we observe that even when the number of senders reaches a size as large as 50, the average SACL size is still very small (less than 4 on average). This is in significant contrast with the strategy of “Full Policy Maintenance” (FPM) on each router. Further comparison between the two methods is presented in Table 6-1. From the figure we can also see that if all the senders are also receivers on the bi-directional tree (case AM), this results in a larger average SACL size. On the other hand, if none of the senders is a receiver (case NM), the corresponding SACL size is smaller. This phenomenon is expected because given the fixed number of receivers on the bi-directional tree as well as the sender group, the larger the proportion of senders coming from receiver set, the larger the resulting average SACL size. However this gap decreases with larger sender group size.
Next we study the effect on SA CL size resulting from the senders' choice of acting as a Send-Only Member (SOM) or a Non-Member Sender (NMS). As we have mentioned, a host only wishing to send data to the group can decide to act as a SOM or NMS. Figure 6-11 illustrates the relationship between the SA CL size and total number of senders. The group size is fixed at 50 and the number of senders varies from 5 to 40 in steps of 5. It should be noted that in this simulation all group members are receive-only hosts and do not send any data to the group. From the figure we can see that the SA CL size also grows with the increase of the number of external senders. Moreover, if all the hosts join the bi-directional tree and act as Send-Only Members (SOM), the average SA CL size is smaller. The reason for this is obvious: if the hosts choose to take the role of SOMs, this will make the bi-directional tree expand for including the DRs of these senders. Since the number of on-tree routers grows up while the total number of senders remains the same, the resulting average SA CL size becomes smaller. On the other hand, if all of the hosts act as Non-Member-Senders, the figure of the shared tree does not change and no more on-tree routers are involved, and hence the average size of SA CL is relatively larger compared to the SOM scenario.
Chapter 6. Multicast Sender Access Control

We continue to study the relationship between the average SACL size and the group size with the number of senders fixed at 20. We still let these senders choose to act as a SOM or NMS respectively. From Figure 6-12 we can see that the SACL size decreases with the growth of the group size in both cases. On the other hand, a SOM join results in smaller average SACL size compared with a NMS one. The gap is more significant with fewer receivers. This is because if senders choose to act as SOMs, they have to join the tree and generate many send-only branches, i.e., more routers are involved in the bi-directional tree. If the hosts just send data without becoming group members, the shared tree will not span to any of these senders, so the number of on-tree routers is independent of the number of senders. When the group size is small (e.g., 10 receivers), the size of the bi-directional tree will increase significantly to include all the senders if they join as SOMs, hence the gap is bigger for a small set of receivers.

![Figure 6-12 Average SACL Size vs. group size](image)

<table>
<thead>
<tr>
<th>S</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPM</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
</tr>
<tr>
<td>SOM</td>
<td>0.65</td>
<td>1.27</td>
<td>1.82</td>
<td>2.3</td>
</tr>
<tr>
<td>NMS</td>
<td>0.73</td>
<td>1.4</td>
<td>2.09</td>
<td>2.73</td>
</tr>
</tbody>
</table>

Table 6-1 Comparison with FPM (Average SACL size)
We also present a comparison between our method and the "Full Policy Maintenance" (FPM) strategy regarding a router's memory consumption. Table 6-1 gives the relationship of SACL size and total number of senders (S). From the table we can see that the length of the access list recorded in each on-tree router in the FPM mechanism is exactly the number of active senders. This imposes very big overhead on all routers compared with our proposed scheme. It is inferred that the maximum number of SACL size of our proposed scheme is also the number of active senders (e.g., the core itself). However the number of such heavily burdened routers is significantly smaller than in FPM. To verify this, we conducted the simulation focusing on the SACL entry distribution in the whole network. In Figure 6-13, the X-axis indicates the SACL size and the Y-axis shows the number of the routers with that number of SACL entries. We still take peer-to-peer applications as our example and the group size is fixed at 30. From the figure we can see that the proposed scheme imposes a very light memory burden on tree routers, e.g., on average 31.7 out of 100 routers have a SACL size equal to 1, and only one router (the core) has the SACL size equal to the group size of 30; this is consistent with the analysis presented above.

### 6.6.3 Inter-domain Scalability

In this section, we present our SACL size analysis based on the inter-domain topology shown in Figure 6-9. We compare the performance between the hierarchical approach we presented in section 6.4 with the non-hierarchical one that forces each on-tree router to maintain SACL entries for all downstream senders, including those in foreign domains. In this simulation we assume that all receivers are also sources, and we evaluate the SACL size performance with the variation of number of senders from 10 to 50 in each domain (altogether 110 to 550 senders). Figure 6-14(a)
illustrates the overall average SACL size for all the 11 domains. We notice that by using the hierarchical approach the average SACL size is reduced by 25.8%, since on-tree routers need only to maintain SACL entries for their local downstream sources. On the other hand, the difference between the two schemes can most significantly be reflected at the root domain, as it is shown in Figure 6-14(b). If the hierarchical solution is adopted, the SACL size performance in the root domain is very similar to that in any other domain (shown in Figure 6-14(a)). In contrast, we can observe that the SACL size in the non-hierarchical approach is much larger than that of the hierarchical one, e.g., when the number of senders in each domain is 50, the average size in the root domain is almost 3 times that of the hierarchical solution. This result is expected because all the inter-domain join requests will enter the root domain to reach the core, thus imposing a heavy burden to the routers near the core. Moreover, we can infer that the core router itself has to maintain 550 entries for all the sources in the non-hierarchical approach, while in the hierarchical one only 50 SACL entries are needed for the local sources.

(a) Average Performance
In section 6.4.2 we mentioned that if a sender comes from a domain without any group members, it should join the tree as a send-only member. In this case a router needs not maintain SACL entries for remote sources, and hence the average router overhead can also be reduced. We consider the following scenario: suppose domains 3, 6 and 9 are sender-only domains that contain no group members, and there are 50 receivers in each of the rest domains respectively. Now we let the number of senders that are not receivers in each domain vary from 5 to 40 in steps of 5 (there are no SR-members in any domain), and we evaluate the performance between the choice of acting as SOMs and NMSs for these sources. In a similar fashion to the previous experiment, we consider both the average performance of all domains and the typical performance of the root domain. From Figure 6-15(a) we can observe that the average performance of all the 11 domains is very similar to that of the intra-domain scenario. On the other hand, we also observe that the gap between SOM and NMS is much more obvious in the root domain as it is shown in Figure 6-15(b). When the total number of sending hosts reaches 40 per domain, the SACL size of SOM is only 47% that of NMS in domain 0. This result is also expected because the NMS schemes forces inter-domain SACL entry maintenance, and typically the on-tree routers need only to record SACL entries for all the hosts in remote send-only domains (domains 3, 6 and 9 in our simulation scenario).
Finally we evaluate the effect of the multicast group size on inter-domain SACL scalability, in a similar fashion to the intra-domain scenario. We fix the number of send-only hosts in each domain to be 20 and also assume that there are no SR-members, and the per-domain group size varies from 10 to 50. We still compare between the SOM and NMS cases. By comparing Figure 6-12 and Figure 6-16, we can observe that the performance of SOM and NMS in the inter-domain scenario is very similar to the intra-domain one: the gap between the two becomes less obvious as the group size grows in each domain. On the other hand, the inter-domain SACL size does not
even increase in comparison to the intra-domain case presented in Figure 6-12. Hence we can draw the conclusion that the proposed sender access control scheme scales well for inter-domain bi-directional trees due to the fact that sender access control is restricted within individual domains.

Figure 6-16 Average SACL Size vs. group size (Inter-domain)

6.7 Summary

Providing mechanisms for anti-DoS attacks is an important issue for an INP that provisions QoS for its customers. In this chapter we propose an efficient mechanism of sender access control for bi-directional multicast trees in the IP multicast service model. Each on-tree router maintains dynamically the access policy for its downstream senders. With this scheme, data packets from unauthorised hosts are discarded once they hit any on-tree router. As such, group members do not receive irrelevant data, and network service availability is guaranteed since the multicast tree is protected from denial-of-service attacks such as data flooding from unauthorised hosts without a valid mSLS. In order to achieve scalability for large-scale multicast applications with many information sources and in order to accommodate more concurrent multicast sessions, we also extend our control mechanism to inter-domain routing where a hierarchical access policy is maintained on the bi-directional tree. Simulation results show that the memory overhead of our scheme is quite lightweight, resulting in good scalability even for inter-domain bi-directional multicast routing schemes. The proposed sender access control solution can be regarded as an additional component to our QoS aware multicast service provisioning in that it provides a scalable but sophisticated scheme for resource protection.

Nevertheless, there still exists another type of source based DoS attacks: malicious sources can send a huge number of registrations/group-joins to cause the explosion of SACLs, and this will
result in the exhaustion of router memory. Our proposed scheme based on soft state is able to reduce the risk, as pending SACL states are timed out without receiving the activation notification from the core. In this case the memory resource consumed by useless SACL entries are dynamically returned to the system. In our future work we will investigate the trade-off between the mechanisms for detecting this two types of DoS attacks (traffic based and memory based).

In addition, the proposed algorithm cannot protect end users and dimensioned network resources from attacks by spoofing sources. Another serious concern is that, DoS can also come from malicious multicast receivers. It has been realised that not only multicast routing/forwarding table explosion may take place but also bandwidth resources may be exhausted if an attacking host sends out a huge number of multicast group join requests that cause disastrous multicast tree expansion. We will address all these DoS associated issues in our future research work.
Chapter 7

7 Conclusion

7.1 Summary

In this thesis we designed and implemented a scalable three-layer framework for provisioning QoS aware multicast services in DiffServ networks. Detailed contributions from this thesis are summarised as follows.

First of all, we proposed the Offline Multicast Traffic Engineering (OMTE) scheme in the management plane, aiming at systematic network provisioning for multicast services with bandwidth guarantees. Different from the exiting approaches that adopt the MPLS explicit routing technology, our proposed scheme shifts to plain PIM-SM routing with respect to optimised IGP link weights. The key idea is that, through deliberate configuration of M-ISIS link weights, hop count Steiner trees with bandwidth constraints are represented into shortest path trees with respect to this set of weights. As a result, network bandwidth resources are conserved and link congestions are eliminated/alleviated. As no MPLS tunnels are required in this solution, the associated scalability issues in terms of LSP state maintenance can be successfully eliminated. We applied Genetic Algorithms to solve the formulated optimisation problem, and the simulation results showed that, traffic sub-optimality and network congestion can be significantly improved compared to conventional non-TE-capable paradigms. Moreover, the performance of bandwidth conservation in our proposed scheme is even comparable to the existing MPLS based approaches using classic Steiner heuristics.

In the control plane, we proposed QoS aware Source Specific Multicast (QSSM) that can be regarded as an extension to the conventional SSM service model. In our solution, we used multicast group address for encoding QoS class information (e.g., DSCP value), so that there is no need for QoS extensions to the underlying multicast protocols as well as to IP routers in the network. On the other hand, we built dedicated multicast delivery trees in different QoS classes, and this strategy simplifies the task of offline network dimensioning as inter-class interactions in multicast routing is not necessary. We also evaluated the performance of QSSM with the guidance from optimised link weight configuration in the management plane. We found from our simulation experiments that the network service capability can be drastically enhanced through the cooperation between OMTE and QSSM in the proposed framework.
We also designed another overlay scheme known as Differentiated QoS Multicast (DQM) in the control plane. The significant difference from QSSM is that, one single hybrid multicast tree is constructed, which is able to span heterogeneous QoS channels. In this scenario, replicated group data need not be delivered in different QoS classes in parallel. As a result, both bandwidth consumption and overhead in multicast QoS state maintenance can be reduced compared to the strategy of constructing dedicated delivery trees for each QoS class. Furthermore, we applied an online routing scheme with bandwidth constraints for multicast tree construction and adaptation, which is more flexible compared to the offline based approaches, but this has not been studied extensively in the existing works on DiffServ-aware multicast routing semantics.

Finally, we envisage that the protection of provisioned network resources from Denial-of-Service (DoS) attacks is a critical issue that has not received much attention, particularly for multicast services. For a very long period, research efforts on network QoS and security have been orthogonal to each other, till recently when it has been argued that QoS mechanisms should specifically involve network security functionality in terms of facing DoS attacks [3]. In our work, we focused on IP multicast sender access control for bi-directional trees that is the most vulnerable to attacks from malicious hosts. In some preliminary solutions, all the on-tree routers should be aware of every authorised sending host, while in our proposed scheme, the same effect can be achieved by deploying the sender access control list (SACL) on downstream tree branches only. Our simulation results show that the memory overhead of the proposed access control scheme is lightweight, resulting in good scalability even for inter-domain bi-directional multicast routing schemes.

### 7.2 Future Work

First, we will continue to design and implement the building blocks that have not been specifically addressed in the proposed framework, i.e. QoS-aware multicast group management at egress routers. One of the significant challenges is that we still lack sophisticated admission control mechanisms for group members at egress routers, network congestion is liable to occur if there are no restrictions on overwhelming group joins.

In the management plane, our major efforts will shift to off-line multicast traffic engineering across multiple ASs so as to enable inter-domain multicast services with QoS requirements. In addition, we will investigate multicast TE with more QoS metrics such as end-to-end delay, packet loss, etc. Another important research dimension is to explore efficient online optimisation solutions in time of significant dynamics of traffic as well as network link failures. At the time of writing, preliminary solutions had been proposed in this direction for unicast traffic [35], and we
recon that it is equally important to come up with corresponding mechanisms for multicast services.

Finally, we will enhance the multicast sender access control scheme with respect to both intra- and inter-domain scope and we will particularly explore effective solutions to protect network resources and end users from DoS attacks by spoofing sources. In addition, we also intend to investigate how to prevent malicious receivers from DoS attacks through sending bogus group join requests.
References


[38] M. Handley, “Internet Denial of Services Considerations”, Internet Draft, draft-iat-dos-01.txt, May 2004, work in progress


[66] S. Shalnov et al, "Quality of Service and Denial of Service", Proc. ACM SIGCOMM 2003, the RIPQOS Workshop


Appendices

A-1. Pseudo Code of the TM Algorithm for general Steiner tree problem

Procedure $TM()$
Begin
For each multicast group $g$
Compute the shortest paths from all nodes to all members in $V_g \cup \{r_g\}$;
Set $T_g = \{r_g\}$ and $E_g = \Phi$;
$Counts = 1$;
While ($Counts < |V_g|$)
Find a path $P(v,u)$ where $v \in T_g$ and $u \in V_g - T_g$, such that
\[ C(v,u) = \min_{i \in T_g, j \in V_g - T_g} C(i,j) \]
$T_g = T_g \cup \{\text{all nodes in } P(v,u) \text{ except } v\}$;
$E_g = E_g \cup \{\text{all links in } P(v,u)\}$;
Decrease the available bandwidth of each link in $P(v,u)$ by $D_g$;
$Counts = Counts + 1$;
End While
End For
End