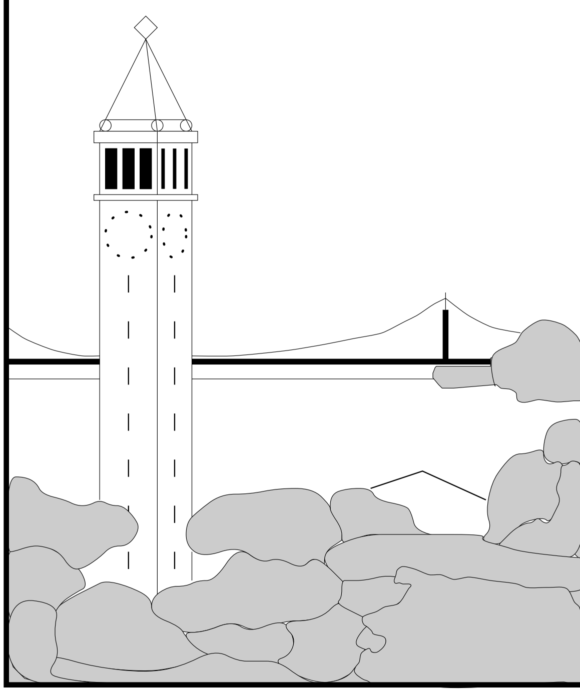
A Scalable Framework for IP-Network Resource Provisioning Through Aggregation and Hierarchical Control

Chen-Nee Chuah



Report No. UCB//CSD-1-1158

September 2001

Computer Science Division (EECS)
University of California
Berkeley, California 94720

Project Supported by Grant
No. 43517-23797-44-NDRXK

A Scalable Framework for IP-Network Resource Provisioning Through Aggregation and Hierarchical Control

by

Chen-Nee Chuah

B.S. (Rutgers, The State University of New Jersey) 1995
M.S. (University of California at Berkeley) 1997

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Engineering – Electrical Engineering and Computer Sciences

in the

GRADUATE DIVISION
of the
UNIVERSITY of CALIFORNIA at BERKELEY

Committee in charge:

Professor Randy H. Katz, Chair
Professor Jean Walrand
Professor J. George Shanthikumar

Fall 2001

The dissertation of Chen-Nee Chuah is approved:

Chair Date

Date

Date

University of California at Berkeley

Fall 2001

A Scalable Framework for IP-Network Resource Provisioning Through Aggregation and Hierarchical Control

Copyright Fall 2001
by
Chen-Nee Chuah

Abstract

A Scalable Framework for IP-Network Resource Provisioning Through Aggregation and Hierarchical Control

by

Chen-Nee Chuah
Doctor of Philosophy in Engineering – Electrical Engineering and Computer Sciences

University of California at Berkeley

Professor Randy H. Katz, Chair

There has been an increasing need to make the Internet architecture capable of meeting the diverse service requirements of newly emerging applications such as multimedia conferencing and e-Commerce. This thesis addresses the following question: *Is it possible to deliver latency-sensitive applications (LSAs), e.g., streaming audio and video, with satisfactory quality of service (QoS) in large-scale network without compromising scalability and bandwidth efficiency?*

Two previously proposed solutions are Integrated Services (Int-Serv) and Differentiated Service (Diff-Serv), but both have their own limitations that hinder their widespread deployments. Int-Serv requires per-flow signaling and state maintenance at every router (including edge and core routers) to provide per-flow performance guarantees. The main weakness of Int-Serv is its complexity that grows with the number of users. Diff-Serv, on the other hand, relies on packet marking at the edge router and class-based queuing at core routers to provide service differentiation. Although the Diff-Serv packet-level mechanisms are well-studied, many flow-level control problems and inter-domain resource allocation issues remain unresolved. In addition, existing work have mostly focused on network-level QoS (e.g., bandwidth guarantees, packet losses and delays) without considering how that relates to the application-level QoS, which matters the most to end users.

We have proposed an innovative distributed control architecture, the *Clearing House (CH)*, and a set of adaptive mechanisms to address these issues. Our design rationale is influenced by discussions with two major U. S. Internet service providers and driven by a realistic model of application-level performance requirements. Towards this end, we focus on Voice-over-IP (VoIP) as an example workload and perform subjective experiments to quantify the impact of packet losses and delays on perceived voice quality. Our results show that packet loss rate should be below 1% and per-hop delay should at most be 5 ms to guarantee high-quality VoIP delivery.

Two key ideas that contribute to the scalability of our CH architecture are: *aggregation* and *hierarchical control*. Our approach exploits the inherent hierarchy of the Internet structure and peering relationships between ISPs. In our model, various basic routing domains are aggregated to form logical domains (LDs), which can then be aggregated to form larger LDs and so forth. This introduces a hierarchical tree of the LDs, and a CH-node is associated with each LD. This hierarchical tree of CH-nodes forms a "virtual

overlay network" on top of existing wide-area network topology. The processing load and state maintenance required to manage an entire ISP domain are now distributed to various CH-nodes at different levels of granularity.

The CH-nodes establish bandwidth reservations on intra- and inter-domain links for aggregate traffic (trunk), rather than individual flows, so that no per-flow management is required at any routers. We approximate the arrival process of trunks as Gaussian, and measure their corresponding mean, μ, and variance, σ^2, during a chosen measurement window. Aggregate reservations are set up based on the measured μ, σ, and the QoS performance goal (e.g., tolerable loss rate). Using VoIP as an example workload, our simulations show that this technique can achieve a loss rate of 0.12% with only 8% over-provisioning.

In addition to resource reservations, two other essential resource control tasks within CH are admission control and traffic policing. Admission control is necessary for limiting the usage of resources by competing flows, while policing is useful for detecting and penalizing malicious flows (i.e., flows that violate their allocated share of bandwidth). For scalability, per-flow admission control is only performed at ingress points of an ISP domain, but it should consider the network-wide congestion level in estimating the impact of admitting new flows. Our scheme, Traffic-Matrix based Admission Control (TMAC), addresses this problem by leveraging the knowledge of the traffic distributions within an ISP and the link capacity constraints to compute the admission thresholds at ingress routers. Our simulation results show that TMAC can achieve 97% utilization level with less than 1% packet loss rate, which is sufficient for most voice applications.

We have also designed a scalable mechanism, called MDAP (Malicious Flow Detection via Aggregate Policing,) for detecting and policing malicious flows without keeping per-flow state at any edge routers. MDAP aggregates admitted flows for group policing without compromising the ability to identify individual malicious flows when necessary. The key insight behind MDAP is a coordinated way of assigning a unique flow-identifier Fid to every flow based on its ingress and egress point. As a result, the amount of state maintained by edge routers can be reduced from $O(n)$ to $O(\sqrt{n})$, where n is the number of admitted flows. We study the performance and robustness of MDAP through ns simulations. Our results show that we can successfully detect a majority (64-83%) of the malicious flows with almost zero false alarms. Packet losses suffered by legitimate flows due to undetected malicious activity are insignificant (0.02-0.9%). The average detection time for correctly identified malicious flows is less than 1/10 of the average flow lifetime.

From the above discussions, we concluded that the CH architecture is capable of satisfying the QoS objectives of LSAs (e.g., $<$ 1% loss rate and 150 ms delay) by exploiting statistical techniques and real-time traffic measurements. We evaluate the practicality and deployment issues of our approach through a lab prototype using VoIP as an example workload. Our implementation experience indicates that CH mechanisms introduce very minimal (at most 5%) processing overhead to an edge router.

Professor Randy H. Katz
Dissertation Committee Chair

To my parents,
my sisters, *Lan, Sim, Teen & Choo*,
my brother, *Teik*,
and *Mark*.

Contents

List of Figures

List of Tables

Acknowledgements

I am truly grateful to everyone who has directly or indirectly supported and helped me complete this Ph.D. dissertation. First and foremost, I would like to thank my advisor, Professor Randy Katz, for giving me a chance to work with him in the ICEBERG project about three years ago. Despite his heavy schedule, Randy takes the time to meet with his students every week, sometimes even on weekends. He reviewed my drafts of papers and dissertation with amazing turn-around times and offered many insightful feedback on my research. This thesis would not have been possible without his support and guidance. I am very fortunate to have been able to tap his technical and professional advice.

I thank my dissertation committee members, Professors Jean Walrand and George J. Shanthikumar, for their comments and suggestions during the course of developing my research agenda and completing this dissertation. I also thank Dr. Steve McCanne for chairing my qualifying exam committee and sharing his expertise in networking research.

In addition to my committee members, I benefited greatly from working with Professor Anthony Joseph and the ICEBERG group members. The concept of a clearing house for resource trading over the Internet was first suggested by Anthony, and set the stage for the first part of my dissertation. The design of the Clearing House (CH) architecture has since been greatly improved and refined through discussions with members of ICEBERG. I owe special thanks to my collaborator, Lakshminarayanan Subramanian, for constantly challenging my ideas and helping me understand the system aspects of a networking architecture. I sincerely thank Brian Shiratsuki and Keith Sklower for maintaining our systems, and for being very responsive whenever I ran into problems and called for help.

I have also learnt a lot from interacting with the faculty and student members of the Smart Networks project. I truly appreciate the technical feedback given to me by Professors Jean Walrand, Pravin Varaiya and Venkat Anantharam during my various presentations at the Smart Networks group meetings.

During the course of my graduate education at Berkeley, I was most fortunate to work with Professors Joseph Kahn and David Tse for my M. S. thesis. They both taught me how to be bold in formulating research problems, be thorough in analysis, and be concise in writing. Their guidance helped me survive my first year of graduate school and build a sound foundation for my research career. I also thank them for being extremely patient and supportive while I was searching for a new Ph.D. topic.

I am grateful to Drs. Supratik Bhattacharyya, Lee Breslau, Jennifer Rexford, Scott Shenker, Nina Taft, John Wroclawsky, and Dina Papaqianaki for their technical and career advice. I benefited greatly from discussions with them in person as well as through electronic mails, and I look forward to interacting with them in the future.

Like many other Berkeley system research groups, ICEBERG holds semi-annual retreats where students get the chance to present their work to industry sponsors and solicit early feedback. I have often received useful suggestions and criticism on my work during these retreats. In particular, I would like to thank Drs. Sally Floyd, Bryan Lyles, Reiner Ludwig, and Kevin Mills for their generosity in time and advice.

I spent two summers of graduate school in industry internships, during which I had a chance to interact with experts from the industry. During Summer 1997, I was an intern with Dr. Jerry Foschini at Lucent Technologies, NJ. Since then, Jerry has been a constant source of career guidance to me. I would like to thank him for his encouragement and support. I also thank Drs. Dimitry Chizhik, Mike Gans, Reinaldo Valenzuela, and

Jonathan Ling for a most stimulating research and learning atmosphere for exploring the various aspects of the BLAST (Bell-Labs Layered Space-Time) architecture.

In the summer of 1998, I interned again at Lucent Technologies, this time in UK. I thank my supervisor, Dr. Ioannis Kriaras, for making this internship possible. I was fortunate to witness and participate in the debate on what the Quality-of-Service (QoS) architecture and reservation protocol should be for UMTS (Universal Mobile Telecommunications Systems). Drs. Xiao Bao Cheng, Jin Yang, and Luca Salgarelli were extremely helpful to me as I learned my way around Cisco routers, internal traffic generators, and the RSVP protocol stack. Their suggestions and ideas helped shape my research agenda when I returned to Berkeley.

I would not have come to graduate school if not for the positive research experience as an undergraduate at Rutgers University. It all began when Professor David Goodman offered me a part-time position at the Wireless Information Network Laboratory (WINLAB) at the end of my sophomore year. I am grateful to Professors David Goodman, Roy Yates, and Christopher Rose for their patience in mentoring and guiding me throughout my very first research project. I thank the following WINLAB alumni: Drs. Sudheer Grandhi, Ching-Yao Huang, Sanjiv Nanda, Ashwin Sampath and Mohammad Saquib for their generosity in time and advice. I do truly appreciate the WINLAB staff members for their help and support.

My wonderful memories of graduate school were most enriched by my day-to-day interactions with fellow graduate students. I thank the members of Soda 473, the "happy family" of Soda 330, and other EECS comrades – Sharad Agarwal, Andy Begel, Yan Chen, Adam Costello, Wei-Dong Cui, Tom Henderson, Barbara Hohlt, Almudena Konrad, Morley Mao, Giao Nguyen, David Oppenheimer, Bhaskar Raman, Drew Roselli, Angela Schuett, Jimmy Shih, Amoolya Singh, Daniel Tan, Helen Wang, Tina Wong, and Tao Ye. They have helped maintain my sense of humor and make my Berkeley years more fun and colorful.

Gene Cheung, Fang Pei Chen, Tz Yin Lin, Jocelyn Nee, Ching Shang, Da-Shan Shiu, and Ma Yi have been with me from the very start when we first stepped into graduate school. They provided me with constant support and encouragement, especially when things got too difficult, and when I needed some boost of confidence. I also owe special thanks to my friends from my home town – Theen Theen Tan, Kean Hock Yeap, and Niny Khor, for always being there through thick and thin, and for ensuring that I had a balanced and fun-filled life outside of Berkeley.

I must thank Mary Byrnes, Ruth Gjerde, Peggy Lau, and Sheila Humphreys for helping me survive the bureaucracy at Berkeley. Their friendly attitude and strong dedication have made life so much more pleasant for graduate students in the EECS department. I truly appreciate their efforts and more importantly, their warm friendship. I also thank Bob Miller, Nathan Berneman, and Damon Hinson who managed our research group matters. They were wonderful to work with and always responded promptly to my requests and questions regarding administrative matters.

I cannot express enough of my gratitude to my parents, my sisters – Lan, Sim, Teen, and Choo, and my brother, Teik. Their constant love and support have always been the source of my strength and the reason I have come this far. They taught me the importance of hard work, discipline, and believing in oneself. Last but not least, I thank Mark Spiller for his love, encouragement, and faith in me. Mark's idealism has always been an endless source of inspiration for me. I also thank the Spiller family for their support and generosity.

Chapter 1

Introduction

1.1 Motivation

The startling growth of Internet and the flexibility of IP-based packet switched networks have expedited the convergence of data communications (packet switched), and voice- and video-based communications (traditionally circuit switched) into a single "IP-based" core architecture. We envision that the future infrastructure should be capable of interconnecting heterogeneous networks (both fixed and mobile, high and low bandwidth) and supporting a wide range of applications. Besides data transactions such as web surfing and e-commerce, the Internet today has been experiencing growth in audio and video streaming with the emergence of multimedia tools like Netmeeting and CuSeeMe, and Internet-telephony companies such as Dialpad.com. However, the existing Internet only offers a single best-effort class of service without any guarantees on delay variation, packet losses or available bandwidth. This can adversely impact the end-to-end performance of real-time applications. Clearly, the time has come to reexamine the design of Internet architecture.

Internet Scaling Challenges

More intelligence is needed in the Internet infrastructure to make it capable of providing different levels of performance assurance, e.g., guaranteed bandwidth and packet delivery within a delay bound. Nevertheless, any such research or implementation efforts are faced with the following challenges:

- **Scale in Number**
  The tremendous Internet growth over the past few years makes its scalability a crucial problem for network designers. The number of hosts advertised in the Domain Name Service (DNS) has more than doubled in two years, from 53 million in January 1999 to 110 million in January 2001 [1]. In the same time span, the number of users online worldwide increases from 150 million to 407.1 million [2]. Therefore, the feasibility of a QoS mechanism depends on whether it can scale well under unpredictable growth.

- **Growth in Heterogeneity**
  The scalability concerns are multi-dimensional, not just an issue of "number". The Internet carries traffic generated from a variety of applications with different traffic char-

acteristics and performance requirements, running on alternative hardware/software platforms. Users are connected via heterogeneous access networks, using devices that differ in capabilities. For example, Table 1.1 compares the traffic behavior and QoS requirements between the traditional data applications and emerging multimedia applications.

Applications	Traffic Behavior	QoS Requirements
Electronic mail (SMTP) file transfer (FTP) remote terminal (Telnet)	Small, batch file tranfers.	Very tolerant of delay. Bandwidth requirement: Low. Best effort.
HTML web browsing	A series of small, bursty file transfer	Tolerant of moderate delay. Bandwidth requirement: Varies. Best effort.
Client-server E-Commerce	Many small two-way transactions	Sensitive to loss & delay. Bandwidth requirement: Low to moderate. Must be reliable.
IP-based voice (VoIP) Real Audio	Constant or variable bit rate	Very sensitive to delay & jitter. Bandwidth requirement: Low Requires predictable delay & loss.
Streaming Video	Variable bit rate	Very sensitive to delay & jitter. Bandwidth requirement: High, variable. Requires predictable delay & loss.

Table 1.1: Heterogeneous traffic behavior and QoS requirements of Internet applications.

- **Distributed Internet Administration**
  The decentralized control of the Internet poses another technical challenge in providing end-to-end QoS. Privatization of the Internet, which was originally funded by DARPA/government, introduces separate and independent administrative or operational domains. These private organizations that operate the different networks and provide end users with access to the Internet are referred to as Internet Service Providers (ISPs). Due to time and market pressures, ISPs and equipment vendors seldom invest in developing infrastructures for the long term. Since the Internet technology was not originally designed to operate over separate domains, many interoperability issues remain unresolved. End-to-end QoS can truly be achieved only if there is a reliable and efficient way to manage inter-domain resource allocation and SLA negotiations.

- **Performance vs. Efficiency Trade-offs**
  Many QoS solutions involve many trade-offs, which often reflect the conflicting interests among the different participating organizations and entities (e.g., ISPs, transit providers) that constitute the networks. For example, network operators would like to admit as many users as possible to increase the network utilization level and profit, but by doing so, they might not be able to provide per-session performance guarantees

required by end-users.

1.2 Problem Definition

This dissertation attempts to answer the following question: *Is it possible to provide end-to-end Quality of Service (QoS) without compromising scalability, flexibility and bandwidth efficiency of the current Internet infrastructure?*

Statistical Quality of Service (QoS)

The term "Quality of Service" can mean many different things, e.g., low latency for Internet telephony; guaranteed bandwidth for streaming audio/video; predictable delivery for collaboration services through interactive Web applications; protection against competing users (e.g., network control for different grades of service); and higher expectations (e.g., minimum guarantees on service quality).

In short, QoS can be defined either at the application level or the network level:

- Application-level QoS characterizes how well user expectations are satisfied, and are usually subjective, e.g., clear voice, jitter-free video, etc.
- Network-level QoS refer to tangible measurements such as controlled latency, available bandwidth, packet loss rate, etc.

The relationship between the application and network-level QoS is not always straightforward, and is often neglected in the previous work. In this dissertation, we attempt to understand how the application-level QoS can be translated to network-centric parameters by analyzing VoIP as an example workload. Our findings are summarized in Section 1.4.2. Let us emphasize that our goal is to achieve statistical QoS, e.g., packet loss rate at 95 percentile or at most 150 ms end-to-end delay, but not hard guarantees, e.g., dedicated bandwidth per-session.

QoS Control Mechanisms

There are currently two approaches to enhance QoS over the Internet. The first relies on application-level QoS mechanisms to improve QoS without making changes to the network infrastructure. This includes modifying the application implementations to make them more adaptive to variations in packet delays and losses, e.g., various reconstruction methods at the Internet audio receiver [3], forward error correction (FEC) [4] and new transport protocols such as RTP/RTCP [5]. This class of mechanisms is out of scope of this dissertation.

The second approach tries to modify the network implementation to provide variable grades of service with some performance guarantees to a heterogeneous mix of Internet flows. There are five crucial components in an architecture that supports QoS:

1. **QoS Specification** The various network components must agree upon a universal way to (a) define the various service classes or performance guarantees that the network can provide, (b) specify QoS requirements and/or traffic characteristics of individual flows, and (c) map the individual QoS requirements in (b) to the service levels defined in (a).

2. **Resource Management and Admission Control** Certain control mechanisms are needed to allocate adequate resources to different service classes and limit the volume within each class, e.g., reservation protocols, capacity planning techniques, etc. The network must determine which resource reservation requests to grant or deny based on the current status of the network. If the admission control policy is conservative, better QoS guarantees are given to network clients, but fewer of them can be supported at the same time, leading to inefficient utilization of resources.

3. **Service Verification and Traffic Policing** There must be some means to verify that the admitted flows truly receive the performance guarantees promised by the network. At the same time, admitted flows must comply to their allocated bandwidth share and should be penalized otherwise.

4. **Packet Forwarding Mechanisms** Packets need to be sorted to their corresponding classes/queues and treated differently based on their QoS requirements. There has been an enormous research effort on this topic, including packet filters, traffic shapers, schedulers and buffer managers.

5. **QoS Routing** The basic function of QoS routing is to find a network path which satisfies the given QoS constraints, e.g., with sufficient bandwidth or minimal delay. QoS routing has been extensively studied, originally in the context of ATM and later in IP networks.

The QoS specification and packet forwarding mechanisms (No. 1 & 4) have been widely studied, and are moving quickly through the Internet standardization process [6]. Recently there is renewed interest in assessing the impact of deploying QoS routing protocols in IP networks [7, 8], which has attracted much debate. However, we will not further discuss these three topics in this dissertation. The second and third components (Resource Management and Traffic Policing) involved session-level control and have remained as open research topics; this dissertation focuses on these open issues.

In a nutshell, this thesis addresses how to provision IP-network resources properly so that latency-sensitive applications can deliver streams with predictable QoS. The proposed architecture and mechanisms must scale well with respect to the rate of Internet growth, and modifications to the network infrastructure should incur minimal overhead. Our contributions and results are summarized in Section 1.4.2.

1.3 State of the Art

Today, the most common approach deployed by Internet Service Providers (ISPs) to provide quality of service (QoS) is to over-provision and over-engineer their operational networks. This is an expensive and inefficient solution since it can take up to six months from planning the upgrade, to adding a new fiber and/or equipment and having the operation of the entire network fully tested. In addition, some of the links are at most 10% or 20% utilized due to this conservation approach. On the other hand, the end-to-end support is achieved by concatenation of bilateral peering agreements with neighboring ISPs. Such agreements describe the volume of traffic exchanged between the two networks, but the offered performance assurance is very vague and usually not verified.

Some ISPs offer Virtual Private Networks (VPNs) [9], as an alternative to private leased lines, to connect private corporate networks over the shared public Internet. Corporations have turned to VPN solutions to build a secured wide-area network (WAN) that can deliver performance and manageability to their various sites scattered around the country. VPN customers require, at a minimum, a predictable performance over VPN's secure tunnels, which is usually specified in the Service Level Agreements (SLAs). An SLA [10, 11] is a contract between the service provider and the customer that specifies the maximum packet transmission rate promised by the customer, and the desired service level in terms of delay, throughput, and loss characteristics. SLAs have traditionally focused on backbone performance, such as backbone/network availability, maximum or average delay, and mean time to notify customers of an network outage. Unfortunately, such coarse-grained guarantees do not reflect end-to-end performance of individual applications. In addition, VPNs can only provide guarantees for traffic traversing within the same ISP, and further development is needed to extend VPNs across multiple ISP domains.

1.3.1 Re-engineering the Internet

In recent years, there has been considerable research focused on extending the Internet architecture to provide better quality of service (QoS). Two major classes of approach that have been proposed to the IETF are: Integrated Services (Int-Serv) [12, 13] and Differentiated Services (Diff-Serv) [14, 15].

Integrated Services (Int-Serv) with RSVP signaling [16] introduces end-to-end per-flow reservations, such that each flow is guaranteed a certain amount of bandwidth at each router along its path from the source to the destination. However, this approach requires maintenance of individual flow states in the routers, and its signaling complexity grows with the number of users. As a result, such architecture may not scale well.

Differentiated Services (Diff-Serv), on the other hand, relies on packet marking and policing at the access or edge routers and different per-hop behaviors (PHB) [17, 18] at core routers to provide service differentiation to aggregate traffic. Edge routers (ERs) are boundary points at which a flow enters or leaves a Diff-Serv domain, while core routers (CRs) are internal routers within the domain. ERs may need to modify individual packets to ensure backward compatibility with external networks that do not support Diff-Serv. There have been comprehensive studies on the Diff-Serv packet forwarding mechanisms, such as scheduling, shaping, queue management, etc., but the understanding of the control framework at the session level is still relatively limited.

Recent proposals use agents known as bandwidth brokers (BB) [19, 20] to allocate preferred service to users as requested, and for configuring the network routers with the correct forwarding behavior for the defined service. BBs act as resource managers that provision resources at domain boundaries and negotiate SLA parameters with neighboring domains. An initial evaluation of bandwidth broker signaling can be found in [21]. However, it remains unclear how a BB computes the amount of resources needed for a service type, and how traffic fluctuation is reflected in the end-to-end resource allocation over multiple domains.

1.4 My Thesis: Scalable, End-to-end Resource Provisioning

In this dissertation, we propose a new control architecture to regulate end-to-end resource provisioning over IP-networks in a scalable manner. Such a control architecture is an essential component for delivering QoS and coordinating network management policies in the Internet [22], but it has not been well-studied.

We adopt the Two-Tier resource management model presented in [20], which breaks the problem down into two sub-problems:

1. **Inter-domain:** Design an architecture to provision resources at an aggregate scale across multiple domains.

2. **Intra-domain:** Design and develop mechanisms to manage intra-domain resources.

The QoS of delivered streams is usually limited by the bottleneck bandwidth along the path in the edge domains. Besides the inter- and intra-domain resource allocation in the backbone, we need to address the following issues at the edge domain:

- Admission control, which is necessary for limiting the usage of resources by competing flows.

- Traffic policing, which is useful to ensure that each admitted flow only uses its allocated share of bandwidth.

In proposal [20], a BB is associated with each domain to manage the internal resources and allocate inter-domain resource agreements. We expand on this model [20] and introduce a hierarchical control architecture, called the *Clearing House*, to distribute the various states and resource management tasks to different nodes.

1.4.1 Basic Assumptions

We make the following assumptions:

- We consider only two QoS classes: Best-effort and High priority. Examples of high priority traffic are latency sensitive applications (LSAs) such as streaming video and audio. Adequate resources must be allocated to this class of traffic to meet the delay and loss requirements. Traditional data transactions, e.g., FTP, e-mail, are carried as Best-effort traffic.

- Since the two QoS classes must co-exist, we need to strictly limit bandwidth allocated to the high-priority traffic so that it does not "starve" the best-effort class. Only high-priority traffic is admission-controlled.

- The networks are capable of providing at least two differential service levels and Diff-Serv primitives such as packet marking, classifier, leaky bucket policer, and rate-controlled priority scheduling [23].

- Every routing domain has the capability to monitor and collect statistics of incoming and outgoing traffic.

- Control paths (e.g., reservation requests) and data paths are separated. Control messages are sent as UDP message in the high priority class.

1.4.2 Contributions

This dissertation's proposed framework attempts to provide better end-to-end QoS, as offered by stateful networks (e.g., Int-Serv), while maintaining scalability and robustness found in stateless network architecture (e.g., Diff-Serv). Our specific contributions are:

- **Workload Modeling**
  First, we have developed a realistic model of the performance requirements of LSAs to drive the evaluation of our proposed architecture and resource control mechanisms. Using Voice-over-IP (VoIP) as an example workload, we performed subjective experiments to quantify the impact of packet losses and delays on user perceived voice quality. We found that loss rates within 1-2.5% are tolerable, but received voice streams become incomprehensible when more than 4% of the packets are lost. To be conservative, we assume the maximum packet loss rate should be less than 1% to deliver high-quality VoIP. To represent the diversity of LSA workloads, we collected 70 packet audio traces from a wide range of multimedia applications, including audio/video conferencing, two-way conversations, and distant learning. We used these traces as traffic workloads to drive a subset of our simulations.

- **Distributed Clearing House Architecture**
  We have designed a Clearing House (CH) architecture that facilitates intra- and inter-domain resource reservations based on statistical estimates of aggregate traffic over a particular link. Two key ideas, *aggregation* and *hierarchical control*, are employed in this design that make the CH scalable to a large user base. These are explained later in Section 1.4.3. We have extensively evaluated the efficiency and performance of the CH through trace-based simulation study.

- **Predictive Reservations**
  We have developed and evaluated a predictive reservation scheme that allocates resources in advance based on aggregate statistics of high-priority traffic. Reservations are set up for aggregate traffic, instead of individual flows, so that no individual state maintenance is required. A Gaussian predictor is used to estimate the required bandwidth based on the aggregate mean and variance of the high priority traffic over a particular measurement window. We quantify the performance characteristics of this approach by applying the predictor to a collection of actual Internet audio traces. Our trace-based simulations show that predictive reservation technique can achieve loss rate of 0.12% with only 8% over-provisioning.

- **New Service Model: IE-Pipes**
  We define a new service model that treats edge routers (ERs), but not hosts, as endpoints. ERs are ingress/egress points where traffic enters/exits the domain. All high-priority traffic that enters an ISP at ingress ER-s, and exits at egress ER-d belongs to the same aggregate denoted as IE-Pipe(s, d).

- **Traffic-matrix based admission control (TMAC)**
  In our model, per-flow admission control is only performed at the ingress router where the traffic enters the domain. We propose a new admission control scheme, TMAC,

that leverages the knowledge of network-wide *traffic matrix* within a domain to determine the admission threshold. A traffic matrix captures the distribution of aggregate traffic demand between every pair of ERs in a network domain, e.g., all IE-Pipes. Such information allows TMAC to take into account the dynamic fluctuations of traffic demands and congestion levels across domains, leading to more efficient resource allocation. Our simulation results show that TMAC can achieve 95% utilization level with less than 1% loss when a VoIP type workload is considered. TMAC does not require per-flow signaling or probing into the core network.

- **Malicious Flow Detection via Aggregate Policing (MDAP)**
  We present a scalable mechanism, MDAP, that allows us to aggregate flows for group policing without compromising the ability to uniquely identify a malicious flow when necessary. Traffic policing in the Diff-Serv literature usually refers to parameter-based packet filter mechanisms, which are useful in tracking and shaping per-flow usage. Here, policing refers to monitoring an aggregate group of admitted flows and identifying malicious flows within this aggregate. The words “malicious” and “misbehaving” are used interchangeably to describe admitted flows that violate their allocated share of bandwidth. The key result is that we can detect majority of the malicious flows without having to keep per-flow state information at the edge routers. Through distributed edge coordination, the amount of states maintained by any edge router is reduced from $O(n)$ to $O(\sqrt{n})$, where n is the number of admitted flows, while core routers are stateless.

  The choice of MDAP parameters determine the trade-offs among different performance indexes, e.g., non-detection rate and frequency of false alarms. We assume that preserving the QoS performance of legitimate flows is more important than punishing the malicious flows. Our simulation results show that MDAP can successfully detect a majority (64-83%) of the malicious flows with almost zero false-alarms. Packet losses suffered by innocent flows due to undetected malicious activity are insignificant (0.02-0.9%). The average detection time for correctly identified malicious flow is 26.9 seconds, which is less than 1/10 of the average flow lifetime.

- **Insights from Implementation Experience**
  We also seek insights on the issues involved in deploying our proposed QoS mechanisms over the current Internet. Towards this end, we have implemented the additional router functionalities required by the TMAC and MDAP schemes (e.g., traffic monitoring and policing, processing and forwarding control messages to LCH-nodes) on top of the Click [24] modular router. Our experiments show that the addition of these modules incur minimal processing overhead to an edge router. The maximum throughput degradation is only 5%. In all our experiments, we observe a minimum response time of 2.5 ms and a maximum of 7.2 ms at a load of 500 simultaneously active flows.

1.4.3 Key Design Principles: Aggregation and Hierarchical Control

In anticipation of the growth in the Internet traffic volume and the resource limitations in the routers and access links, it is crucial that our overlay control architecture is scalable and incrementally deployable. The key design principles behind our framework are:

- **Hierarchical control** We assume the network is composed of various basic routing domains, which can be aggregated to form *logical domains*. These logical domains (LDs) are then aggregated to form larger logical domains and so forth. This introduces a hierarchical tree of the LDs and a CH-node is associated with each LD. Individual CH-nodes are agents that manage aggregate reservations for all the links within the same domain at a particular hierarchical level. The reservations between neighboring domains are monitored by the parent CH-node. This hierarchical tree of CH-nodes form a "virtual overlay network" on top of existing wide-area network topology.

  The hierarchical structure allows the CH architecture to efficiently coordinate distributed decisions to adapt the inter-domain trunk reservations based on predictions of end-to-end traffic demand distributions. This leads to manageability in allocating resources across multiple domains and provides more predictable end-to-end QoS.

- **Aggregation** In our approach, reservations are set up only for aggregate traffic and not for individual flows. Therefore, no per-flow state maintenance is needed. We also propose to aggregate incoming flows for group policing at the ingress ER, and present a methodology to assign a unique flow-identifier intelligently such that we can trace a particular flow when necessary.

- **Leveraging traffic demand predictions** Internet data traffic exhibits burstiness at multiple time-scales. Although it is difficult to predict the bandwidth usage of a single flow, the behavior of the aggregate traffic is more predictable. The aggregate traffic statistics are useful in estimating future demand for the traffic aggregate as a whole. As the number of flows in the aggregate grows, the estimation gets closer to the actual traffic demand.

1.5 Dissertation Overview

The remainder of this dissertation is organized as follows. Chapter 2 reviews related work to this dissertation and addresses some of the differences between previous work and our approach. We discuss the latest development of the Diff-Serv bandwidth broker architecture, recent research in resource management for virtual private networks, measurement-based admission control and traffic policing.

Chapter 3 presents our research methodology and explains how we model the various workloads that drive the evaluation of our proposed architecture and mechanisms. In particular, we consider Voice-over-IP (VoIP) as an instant of latency-sensitive applications and have conducted a subjective experiment to map the packet loss rate to human perceived quality. This chapter also describes the property of 70 voice traces that were used in our simulation study. These traces were collected from various Internet multimedia applications.

Chapter 4 provides a high level architectural view of the distributed Clearing House, and discusses how our design choices are influenced by some observed properties and economic models of the current Internet infrastructure. We provide an illustration that explains how the CH-architecture can be deployed within an ISP domain and in multiple-ISP Scenario.

Chapter 5 describes how we leverage the predictability of aggregate traffic demand to set up and adapt intra- and inter-domain trunk reservations. We present a thorough evaluation of our Gaussian bandwidth predictor using both simulated traffic and real voice traces. We also explore the robustness of the Gaussian predictor with respect to traffic variability and measurement window. Our results show that we can achieve less than 1% loss rate with only 8% over-provisioning with a 1-minute window measurement. In this chapter, we propose a new scheme, the Traffic-Matrix based Admission Control (TMAC), that considers network-wide traffic demand distributions within an ISP (between each ingress and egress pair) in making admission control decisions.

Chapter 6 addresses the need for scalable, efficient traffic policing and malicious flow detection scheme at the edge domains to provide better end-to-end QoS. We extend the Clearing House architecture by adding traffic monitors at the edge routers and control mechanisms, called Furies, near the domain boundaries. To preserve the ability to uniquely identify a flow, in case it is malicious, Furies explicitly assigns a flow identifier, Fid to every admitted flow. Each Fid has two subfields: $FidIn$ and $FidEg$. $FidIn$ is assigned based on the ingress ER of the flow, and similarly $FidEg$ is assigned based on its egress ER. At the ingress(egress) ER, admitted flows are aggregated based on their $FidIn(FidEg)$ for group policing. The fact that each ER maintains only the aggregate state for each *group*, identified by Fid subfields, is crucial for the reduction of state information from $O(n)$ to $O(\sqrt(n))$, where n is the number of admitted flows. The details of the malicious flow detection scheme are described in this chapter.

Chapter 7 concludes the dissertation and discusses directions for future work.

Chapter 2

Related Work

There has been a wide spectrum of work during the last decade on Quality of Service (QoS) control and management in packet switched networks. This chapter only presents a survey of related work that serves as background to our research. In particular, we will survey recent development in: QoS control and router mechanisms in Section 2.1, resource provisioning techniques and bandwidth brokering architecture in Section 2.2, admission control based on passive measurements and active probings in Section 2.3, and traffic policing in Section 2.4. We will also discuss how our proposed architecture and mechanisms are distinct from or influenced by these prior efforts.

2.1 QoS Control in Packet Networks: An Overview

The emergence of IP telephony, video conferencing and other applications with very different throughput, loss and delay requirements are calling for substantial changes in the Internet infrastructure that was originally designed to offer a single, best-effort level of service. Providing different levels of service in the network requires new QoS control and management capabilities, which can be classified along two major axes: *data path* and *control path*. Data path mechanisms are responsible for classifying and mapping user packets to their intended service class and enforcing the treatment (e.g., amount of resources consumed or delay experienced) received by each service class. Control path mechanisms allow the users and the network to agree on service definitions. They are also needed to determine which users to grant service to, and appropriately allocate resources to each service class. The QoS mechanisms discussed in this section have largely been proposed for the IP layer (Layer 3) and developed to be application independent.

2.1.1 Schedulers and Buffer Management

Data path mechanisms are basic building blocks in a QoS-aware infrastructure. They control how packets access network resources, such as buffers and bandwidth, to provide service differentiation. The two corresponding mechanisms that have been long-standing research topics are (a) scheduling algorithms and (b) buffer management schemes, respectively. Scheduling mechanisms control which packets are selected for transmission on the link, while buffer management schemes decide which packets can be stored or dropped as they wait for transmission.

G. Apostolopoulos et al. reviews the different scheduling and buffer management schemes in [8] and discusses their associated trade-offs in terms of fairness, isolation, efficiency, performance and complexity. For example, Weighted Fair Queuing (WFQ) [26] and its many variants provide rate and delay guarantees to individual flows, while class based scheduling mechanisms, e.g., CBQ [27] provide aggregate service guarantees to the set of flows mapped into the same class. The finer granularity of per-flow information required for the former case comes at the cost of greater complexity. Examples of buffer management schemes include: First Come First Serve (FCFS), Early Packet Discard (EPD) [28] and Random Early Drop (RED) [29].

2.1.2 QoS Control Architecture

There are two major approaches currently under development in the Internet Engineering Task Force (IETF)[1]: Integrated-Services (Int-Serv) [12, 13] and Differentiated-Services (Diff-Serv) [14, 15].

Integrated Service

The philosophy behind Int-Serv is that routers must be able to reserve resources for individual flows to provide QoS guarantees to end users. Int-Serv QoS control framework supports two additional classes of service besides "best effort: (a) Guaranteed service [30] and (b) Controlled-load service [31]. Guaranteed service provides quantitative and hard (deterministic) guarantees, e.g., lossless transmission and upper-bound on end-to-end delay. This is useful for hard real-time applications that are intolerant of any datagram arriving after their play-back time [32]. Controlled-load service is intended to support a broad class of applications that are highly sensitive to overloaded conditions. It promises performance as good as in an "unloaded" datagram network, and provides no quantitative assurance. Both services must ensure that adequate bandwidth and packet processing resources are available to satisfy the level of service requested. This must be accomplished through active admission control. The following are the various Int-Serv components that are needed to provide end-to-end QoS, and many research contributions have been made to define their functionality and study their implementation issues:

- A signaling protocol to set up and tear down reservations, e.g., Resource ReSerVation Protocol (RSVP) [16].
- An application-level interface (API) for applications to communicate their QoS needs, e.g., Unix RSVP API (RAPI). [2]
- Per-flow scheduling in the network (e.g., WFQ [26]).

Unfortunately, Int-Serv faces other challenges that make immediate deployment infeasible. The increase in per flow state maintenance at routers is proportionally to the number of flows. This incurs huge storage and processing overhead at routers, and therefore does not scale well in the Internet core backbone. In addition, RSVP/Int-Serv Model needs

[1] `http://www.ietf.org/`

[2] The technical standard of Resource ReSerVation Protocol API (RAPI), developed by the Open Group, is available at `http://www.opengroup.org/onlinepubs/9619099/toc.htm`.

to work over different data-links such as Ethernet, and ATM. Therefore, mechanisms to map integrated services onto specific shared media are needed.

Differentiated Service

Diff-Serv, on the other hand, aggregate multiple flows with similar traffic characteristics and performance requirements into a few classes. This approach requires either end-user applications, first hop routers or *Ingress* routers (interface where packets enter an administrative domain) to mark the individual packets to indicate different service class, e.g., low delay, high throughput, etc. Currently this QoS information is carried in band within the packet in the Type of Service (TOS) field in IPv4 header or Differentiated Service (DS) field in IPv6 [33]. The backbone routers provide per-hop differential treatments to different service classes as defined by the Per Hop Behaviors (PHBs) [34]. Two service models have been proposed: assured service [17] and premium service [18]. Assured service is intended for customers that need reliable services from service providers. The customers themselves are responsible for deciding how their applications share the amount of bandwidth allocated. Premium Service provides low-delay and low-jitter service, and is suitable for Internet telephony, video-conferencing and for creating virtual lease lines for Virtual Private Networks (VPNs).

Diff-Serv approach has several advantages over Int-Serv:

- Diff-Serv is simpler than Int-Serv and does not require end-to-end signaling.
- Diff-Serv is efficient for core routers since classification and PHBs are based on a few bits rather than per-flow information. Since there are only a limited number of service classes indicated by the TOS field, the amount of state information is proportional to the number of classes rather than number of flows, and therefore Diff-Serv approach is more scalable than Int-Serv.
- Diff-Serv requires minimum change to the current network infrastructure. End hosts, routers or firewall can mark packets while intermediate routers/switches can employ active queue management to provide service differentiation based on bits in the packet headers.

Although flow aggregation improves scalability in Diff-Serv, it becomes unclear what level of statistical guarantees Diff-Serv can provide to individual flows, and if there exist such "guarantees" at all. In [35], Bolot analyzes the performance of two Diff-Serv service models: assured service and premium service. Several studies [36, 37] examine the loss and/or delay behaviors of Diff-Serv architecture using a variety of traffic models.

2.1.3 The Big Picture

Figure 2.1 shows how the Internet infrastructure is evolving. While smaller-scaled local networks might be able to support RSVP signaling and per-flow queuing as in Int-Serv, we envision that the backbone will be more likely to provide coarse-grained service differentiation as in Diff-Serv approach. In this model, the heterogeneous devices and access networks are connected to the backbone through boundary routers, or edge routers (ER), which will mark the TOS or DS field value of the packets accordingly.

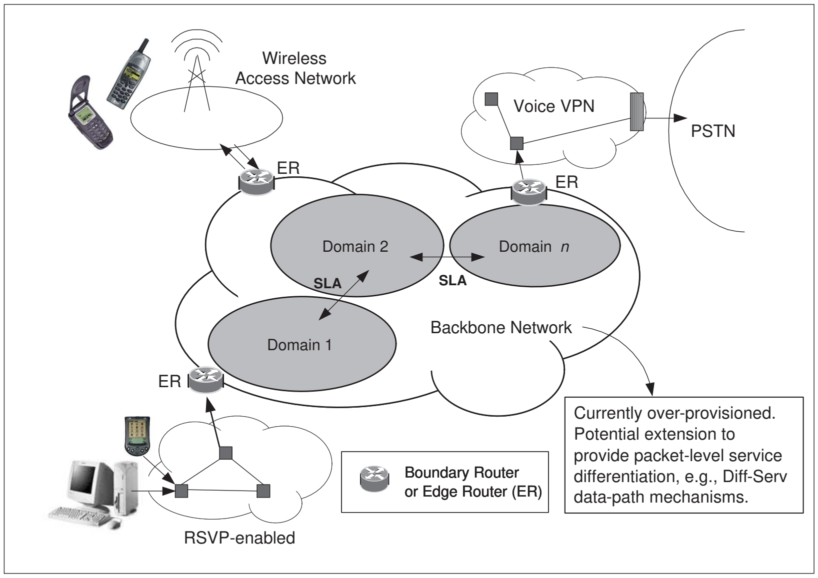


Figure 2.1: Distributed administration of Internet infrastructure with multiple ISP (Internet Service Provider) domains and heterogeneous access networks.

We adopted the Diff-Serv approach that separates data path mechanisms from admission control and resource allocation mechanisms that belong to the management plane. The network is viewed as a collection of various domains based on administrative boundaries, e.g., an organization's Intra-net or an ISP makes a domain. At domain boundaries, *service-level agreements* (SLAs) are made regarding to the amount of resources allocated to traffic that cross domains. A Service Level Agreement (SLA) also refers to the contract between a service provider and a customer. In this case, the SLA specifies a fixed peak bit rate, which the customer is responsible for not exceeding. All excess traffic is dropped. For better link utilization, dynamic SLAs should be supported so customers can request bandwidth on demand. However, the proper configuration of Diff-Serv mechanisms and traffic handling within each domain cannot work in isolation to achieve predictable end-to-end service model. They must be coordinated in a scalable manner across many devices in multiple domains to provide useful end-to-end services.

Figure 2.2 summarizes the various QoS control components of IP-networks in both data and control planes. As mentioned earlier, packet forwarding mechanisms (e.g., schedulers and buffer management) have been the subject of various studies while it is within the last five years that control architectures such as Diff-Serv and Int-Serve are being developed. We have described some of these earlier works in Section 2.1.1 and 2.1.2. The rest of this chapter will survey the related work in the areas of resource provisioning, admission control, and traffic policing (the shaded areas in Figure 2.2), which are the focus of our dissertation.

2.2 Network Resource Provisioning

As we described in the previous section, the management plane of the Diff-Serv

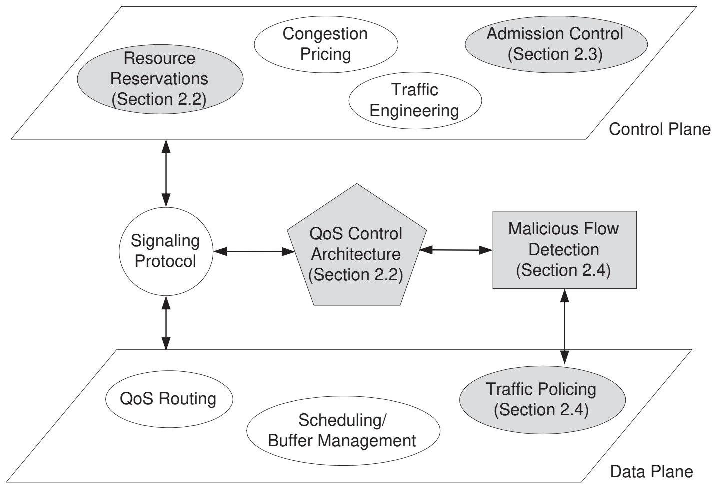


Figure 2.2: QoS control mechanisms in both control and data planes.

architecture is a design space that remains to be explored. In late 1997, the concept of "Bandwidth Brokers" was introduced by K. Nichols et al. in [38] as an entity in charge of resource management in an administrative domain. The Internet2 QoS working group [39] has then made an attempt to harmonize the different ideas and proposals to define a model of Bandwidth Broker (BB) to be deployed in an inter-domain Diff-Serv test-bed called Qbone [40].

The issues of resource allocation and management are not unique to Diff-Serv architecture. In fact, they are long-standing research topics on their own. Besides discussing the prior work on Diff-Serv bandwidth broker, this section also reviews other resource management techniques and architecture relevant to this dissertation, including: dynamic allocation for virtual private networks, capacity planning in telephone networks, advance reservation techniques, and pricing-based approach.

2.2.1 Diff-Serv Bandwidth Broker Architecture

Several bandwidth broker (BB) implementations have been proposed and analyzed in [19, 20, 21] as a scalable QoS provisioning mechanism over the Diff-Serv architecture. A BB is an agent that performs a subset of policy management functionality, including:

- admission control to limit the number of connection requests based on available resources in the network,
- intra-domain resource allocation to support the QoS services offered to users, and configuring routers with correct forwarding behavior, and
- automate inter-domain SLA negotiations.

In [21], the authors presented the broker signaling trade-offs in the context of the Swiss National Science Foundation project CATI [41], but they do not optimize end-to-end path selections. The Internet2 QoS working group have been investigating the inter-broker

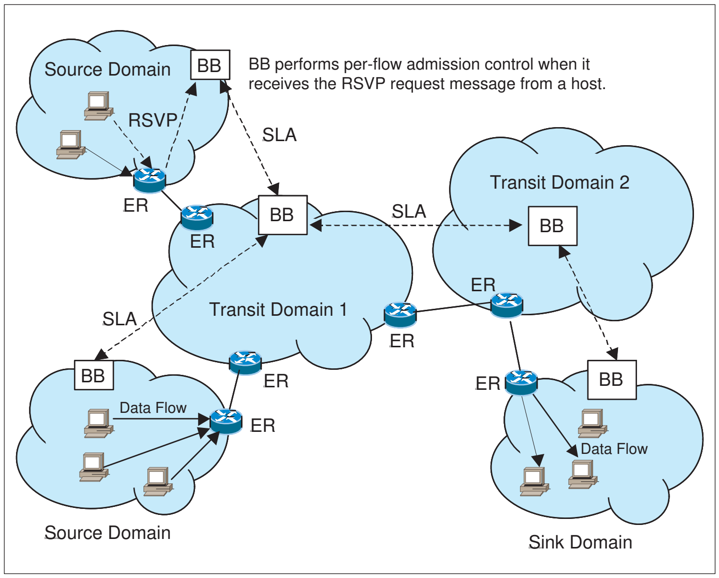


Figure 2.3: Two-tier model for Diff-Serv Bandwidth Broker architecture.

signaling to automate the adaptive reservation within and across domains. However, the BBs are currently configured manually, and many design decisions remain open.

Terzis et al. proposed a *Two-Tier* model [20] where resource allocation control is separated into two level hierarchy: inter-domain allocation and intra-domain allocation (Figure 2.3). In this case, the Bandwidth Broker has a dual role:

- manages internal resources within each administrative domain, which can be fined grained (per flow), and
- maintains bilateral SLA agreements with its neighboring BBs to allocate resources to aggregate traffic crossing domain borders.

This design reflects today's Internet two-level routing architecture that allows each Autonomous Systems to freely choose its own routing protocol internally (OSPF [42], RIP [43], or IS-IS[44]). To achieve cross-domain connectivity, neighboring domains use the Inter-domain routing protocol BGP [45] to exchange network reachability information.

A simplified RSVP is used in [20] as an intra-domain resource allocation protocol for traffic between edge routers, and admission control is performed on a hop-by-hop basis. The inter-domain allocation is adjusted dynamically based on a *additive increase multiplicative decrease* approach. The reservations are performed locally between two neighboring domains without reflecting the traffic and network variation in other domains that lie in the end-to-end path between source and destination networks. End-to-end QoS support is achieved through the concatenation of bilateral SLAs and adequate intra-domain resource allocations.

Discussion

Although we adopt the fundamental principle of the *two-tier* model for intra- and inter-domain resource allocation, our mechanisms and emphasis are significantly different from [20] or other BB proposals [40, 41].

1. We introduce a hierarchical structure, called the Clearing House (CH), within each administrative domain to manage resource allocation instead of employing a single Bandwidth Broker. The intra- and inter-domain resource control tasks are partitioned and distributed to various CH-nodes, so that the amount of state maintenance and processing per node is reduced.

2. Per-flow admission control is only performed by a local resource manager at the edge (leaf nodes) without propagating the reservation request hop-by-hop across the entire administrative domain. Our approach considers network-wide traffic distributions in choosing the admission threshold, rather than measuring the impact of admitting a new flow on a single node (ingress point) as in many previous solutions.

3. Inter-domain resource reservations are established in advance based on real-time measurements of aggregate traffic statistics, rather than the congestion control approach detailed in [20].

4. We propose a scalable approach to perform traffic policing and detect misbehaving flows within each domain, which are not addressed in the previous BB literature.

2.2.2 Virtual Private Networks

The paradigm of allocating resources for traffic aggregates has also been applied to managing Virtual Private Networks (VPNs). Reference [46] proposed a service model, called *hose*, which specifies the capacity required for aggregate traffic from one endpoint to the set of other endpoints in the VPN customer sites. Each *hose* is associated with a performance guarantee. The share of resources allocated to a *hose* is resized based on a set of traffic predictors, and the performance of this adaptive scheme is compared to static provisioning through trace-driven simulations. The authors consider traces of telephone calls over the AT&T national long distance network as well as data traffic on a large corporate private network. Results show that dynamic resizing method achieves a factor of 2 to 3 in capacity savings on access links over statically provisioned customer-pipes.

However, this work only considers a single ISP scenario. It is important to study how this technique can be extended to address inter-domain resource allocation in the case of multiple domains (ISPs). The ultimate challenge is how to provide end-to-end performance assurance through intelligent coordination of intra- and inter-domain resource control mechanisms without requiring a complex signaling protocol. Another desirable property of the solution is scalability with respect to both the number of users and the number of domains traversed.

2.2.3 Capacity Planning in Telecommunication Networks

The concept of hierarchical databases has long been used in telephone network switching, and for user mobility management in the PCS network. In both cases, the sessions are circuit switched or connection oriented, and each session generates a constant bit rate (CBR) traffic. The hierarchy of increasingly aggregated flows is common in the telephone network, but it is based on a fixed bit-interleaved digital multiplexing, as defined in the PDH standard [47], e.g., 24 telephone channels are carried at the T1 level (1.544 Mb/s). Each session is assigned a fixed time-slice of the resources.

The hierarchical structure of our proposed Clearing House architecture that allocate resources at different levels of aggregation is very similar to the telephone network. However, this dissertation explores a different problem space where all the sessions are connection-less, and individual flows can generate variable bit-rate traffic (due to compression), which allows statistical multiplexing at the packet level. The CH-architecture aggregates call requests and perform admission control decision in real-time based on the available bandwidth and network performance. As a result, we see constantly varying statistical multiplexing gains at different links, as opposed to a fixed multiplexing ratio achieved in the telephone trunks. In addition, routing (setting up a dedicated circuit) and resource allocation (allocating time-slice) are tied together in the telephone network when a connection is established. In our case, the two components are separate and we do not provide establish per-flow end-to-end reservations.

2.2.4 Advance Reservations

The need for advance resource reservation (ARR) has been recognized for applications such as video conferencing where network resources are required at a specific start time and for a given duration. Many studies [48]-[51] on this subject focus on performance modeling of ARR on a single link. The co-existence of immediate and advance reservations is addressed in [52, 53] where the authors show that network resources can be shared between the two without being pre-partitioned. Immediate and advance admission control are performed by agents [52] so that reservations can be provided without requiring any state maintenance at the routers. An important parameter is the *lookahead time*, the point at which the agents start making resources available for approaching advance reservations by rejecting immediate requests. It is assumed that individual users specify the bandwidth requirement at the time of requests, and for advance reservations, the duration is also specified.

We, on the other hand, consider advance reservation for the intra- and inter-domain traffic aggregate, instead of for individual sessions. The advance reservations are established based on aggregate traffic measurements without relying on how well individual flows keep to their bandwidth specifications.

2.2.5 Dynamic Packet State

I. Stoica et al., have proposed a new architecture called Scalable Core (SCORE) [54, 55] in which only edge routers perform per flow management, while core routers do not. The authors have shown that a SCORE network can achieve fair bandwidth allocation in [54], and provide end-to-end per flow delay and bandwidth guarantees (like Int-Serv) in [55]. The key technique behind SCORE is the Dynamic Packet State (DPS) approach that carries additional state information in each packet header. The packet header state is initialized by the ingress routers. The core routers process each packet based on the state carried in its header, update the state in the packet's header and forward it to the next hop. With DPS, the actions of edge and core routers along the path of a flow can be coordinated to implement distributed algorithms that deliver QoS assurance without maintaining per flow state at the core routers.

2.2.6 Pricing-based Approach

The Internet access has been predominantly sold based on flat monthly rates depending only on the size of access links, not on usage. However, there is a strong motivation to adopt "usage-sensitive pricing", as advocated by Professor Pravin Varaiya in the INFOCOM'99 keynote lecture:

> Flat-rate pricing encourages waste and requires 20 percent of users who account for 80 percent of the traffic to be subsidized by other users and other forms of revenue. Furthermore, flat-rate pricing is incompatible with quality-differentiated services.
>
> *Professor Pravin Varaiya*, University of California, Berkeley

In fact, the role of prices as essential resource allocation control signals has long been established. J. Sairamesh et al. proposed a new QoS provisioning methodology based on mathematical economic models in [56]. They compute the equilibrium prices based on the user demands, and from this determine the optimal allocation of buffer and link resources to each of the traffic classes. Results in [56] are based on a single-node model that has multiple output links with an output buffer. In another independent work, N. Semret et al. [57] introduce the Progressive Second Price (PSP) auction as a bandwidth pricing mechanism, and show that it achieves economic objectives (efficiency and incentive compatibility), while requiring small signaling and computational load. Further studies are needed to investigate the applicability of these results [56, 57] to large networks, and develop market based mechanisms to admit and route sessions over multiple domains. References [58] and [59] examine some of these issues. The former [58] considers a game theoretic model of capacity provisioning in a Diff-Serv Internet to maintain stable and consistent SLAs across multiple networks. The latter [59] introduced a hierarchical economy consisting of two types of markets (retail and wholesale) and three types of entities (service provider, domain broker, and users). The authors in [59] use retail marker estimation to determine the optical buying/selling strategies that maximizes profile while maintaining low blocking probability.

There is also a huge literature on Internet charging and billing mechanisms, mostly in the context of how users value services and react to price changes. Recently, a large-scale experiment called INDEX [60] was deployed to test users' willingness to pay for various Internet access option. The INDEX investigators conclude from the experimental data that differentiated services and usage-sensitive pricing would be better for both ISPs and users. However, the data also shows that metered billing has dramatically decreased usage. A. Odlyzko [61] attributed this decrease to very strong consumer preferences for simplicity, especially flat-rate pricing. The author's argument is based on an extensive and detailed analysis of the history of communication technologies reported in [61], including ordinary mail, telegraph, wired voice phone, cell phone, residential Internet access and private lines. For example, when AOL switched from usage-based pricing to flat-rate pricing in October 1996, the usage per person tripled in a year because the users found flat-rate pricing easier to understand. As a result of the increased usage, the total profit was greater when AOL used flat-rate pricing as opposed to usage-based pricing. This result indicates that the main weakness of usage-based pricing is its relative complexity compared to flat-rate pricing, which makes it less attractive to end users.

A detailed analysis of the effect of the various Internet pricing on user behavior and network efficiency is out of scope of this dissertation. We briefly mention recent work on pricing here because it provides an orthogonal degree of freedom to achieve Internet service differentiation. This thesis mainly focuses on network resource management and addresses the tradeoffs between efficiency and end-to-end performance.

2.3 Admission Control

Admission control is an essential component of any control architectures providing service differentiation. It determines whether flows requesting services are accepted (or rejected) depending on the available network resources to ensure that acceptable QoS levels are delivered the the admitted traffic. There are typically two classes of approach: *parameter-based* or *measurement-based* admission control. Parameter-based admission control algorithms are based on worst case bounds derived from the parameters describing the flow, and are typically more appropriate for providing hard-real time services. Their effectiveness depends on the ability to predict the traffic behavior based on client-specified parameters, and hinges on the ability of the flows to provide the best guesses of what these parameters are, and their lack of incentives to lie. These algorithms may result in low network utilization if the traffic is bursty. On the other hand, measurement-based admission control (MBACs) algorithms base their decisions on measurements of existing traffic rather than on worst-case bounds. Therefore, MBACs are best suited for providing soft real-time service, i.e. an enhanced QoS without hard guarantees.

In our architecture, we chose measurement-based over parameter-based admission control for two reasons. First, MBACs yield higher network utilization, and secondly it is difficult to describe Internet traffic with such diversity and unpredictability with a reasonably small set of parameters.

2.3.1 Measurement Based Admission Control

Many algorithms and principles outlined in the MBAC literature apply in our work, and we definitely benefit from results in [62, 64, 65] to name a few. L. Breslau et al. evaluated six different MBACs in [62] and results showed that all these algorithms achieved nearly identical performance in terms of their ability of balance the tradeoff of losses (QoS seen by individual users) and load (network utilization). Their study also revealed the following insights:

- measurement estimation and admission decision processes can be decoupled for many algorithms
- Differences in performance caused by flow heterogeneity should be addressed by policy, and rather than by algorithmic differences.
- MBACs appeared to cope well with long range dependence. In some of the simulation scenarios, they perform better than parameter-based algorithms.
- None of the MBACs evaluated are able to provide reliable performance tuning knobs that allow network operators to set a target performance level and actually match it.

In [65], the authors implemented and evaluated a new MBAC algorithm that exploits measured peak rate envelopes of the aggregate traffic. The “maximal rate envelope” is a function of a chosen interval length and captures the temporal autocorrelation structure of the aggregate flow. The MBAC uses this envelope to bound future packet arrivals, and ensures that the admission of new flow will not cause any buffer overflow. Packet losses and delays may occur due to the uncertainty of the prediction. The authors presented new theory to quantify the confidence level of a schedule-ability condition and predict loss probability when the condition is violated.

2.3.2 End-point Admission Control

Admission control in the traditional Int-Serv approach requires a signaling mechanism such as RSVP [16] to carry per-flow request to all the routers along the path. The routers must perform local admission control and keep per-flow state to ensure delivery of desired QoS. The significant burden placed on the routers limit the scalability of this approach. An alternative solution is *end-point admission control* where end-hosts probe the network to check for resource availability before establishing any connections. This is combined with the course-grained Diff-Serv router mechanisms and proper provisioning in the network to achieve QoS.

Recent proposals on end-point admission control [66]-[72] share similar architectures but differ significantly in the control algorithms. Prior to call establishment, the end host send probe packets at the data rate it would like to reserve. In [66] and [67], all data and probe packets are indistinguishable, and there is no differentiation of best-effort vs. real-time traffic. The packets are marked upon congestion (ECN congestion marks) [68], and flows must pay for the marked packets. In this case, admission control is an implicit service provided through price discrimination. The schemes described in [69] and [70] use packet drops instead of congestion marks to indicate congestion, and probe packets are sent in a separate (lower) priority class than data. The *endpoint* in [71, 72] refers to the edge router and not the host. In this setting, edge routers passively monitor paths to derive better estimates of the current network load. L. Breslau et al. provided a careful study of the architectural and performance issues inherent in endpoint admission control in [73].

Our proposed framework shares similar features as end-point admission control, that is the per-flow admission control is only performed at the edge. However, the details of our scheme differ significantly. We do not rely on per-hop signaling protocol or end-host probing to determine whether sufficient resources are available. Instead we leverage the knowledge of aggregate traffic distribution in the ISP domain between different ingress and egress routers to make admission control decisions. The details are discussed later in Chapter 5.

2.4 Traffic Policing

Traffic policing in the Diff-Serv literature usually refers to parameter-based packet filter mechanisms, which are useful in tracking and shaping per-flow usage. In this dissertation, policing refers to monitoring admitted traffic and identifying malicious flows. We use the words “malicious” and “misbehaving” interchangeably to describe admitted flows that violate their allocated share of bandwidth.

2.4.1 Stochastic Fair Blue

The most relevant work with respect to our traffic policing mechanism is Stochastic Fair Blue (SFB) proposed by W. Feng, et al. in [74]. SFB provides a scalable way to identify and rate-limit non-responsive flows using two independent algorithms BLUE [75] and a Bloom filter. BLUE is an active queue management algorithm that uses packet loss and link utilization history to manage congestion. It marks packets in the queue based on a probability that is incremented when a buffer overflow occurs. The rate at which it sends back congestion notification also increases with the marking probability. On the contrary, if the queue becomes empty or the link is idle, BLUE decreases this marking probability. Bloom filters are designed to uniquely classify objects through the use of multiple, independent hash functions. They are commonly used in word processing software applications as an efficient means to do spell checking or web caches to efficiently determine the existence of an object. Using bloom filters, SFB is able to classify flows with an extremely small amount of state and a small amount of buffer space.

The goal of SFB is to manage congestion and enforce fairness among a large number of flows. The basic algorithm is as follow:

- SFB maintains $N \times L$ accounting bins. The bins are organized in L levels with N bins in each level. There are L independent hash functions, and each is associated with one level of the accounting bins.
- Each hash function maps a flow into one of the N bins in that level. When a packet arrives at the queue, it is hashed into one of the N bins in each of the L levels.
- The accounting bins keep track of a marking/dropping probability, p_m as in BLUE, which is incremented when the bin goes above a threshold.
- The decision to mark a packet is based on $p_{\min}$, the minimum of p_m of all bins to which the flow is mapped into. If $p_{\min}$ is 1, the packet is identified as belonging to a non-responsive flow, and is rate-limited.

In short, SFB can effectively identify a single non-responsive flow in n^L flow aggregate using $O(L \cdot n)$ amount of state.

The idea of classifying good versus bad (non-responsive in SFB or misbehaving in our case) flows is similar, but the associated algorithm is different. Instead of Bloom filters, we classify packets into different groups for policing based on the Flow-Identifiers (Fids) carried in the packet header. We employ a set of token bucket filters (TBF) to police the traffic, and packets are dropped when the TBFs overflow. Active queue management such as BLUE is not considered in our scheme. The details of the detection scheme are outlined in Chapter 6.

2.5 Summary

Our survey indicates that QoS control mechanisms in the data path have been well studied. Some of the solutions such as Weighted Fair Queuing (WFQ) and Random Early Drop (RED) have been implemented in existing routers while other proposals are under development in the IETF (Section 2.1.1). On the other hand, we have relatively limited understanding of the control plane and many open issues remain to be resolved. One of the challenges is how to coordinate resource allocation within and across multiple domains in a scalable manner to provide end-to-end performance guarantees. Towards this end, Int-Serv and Diff-Serv have been developed as QoS-aware control architectures but each of these two solutions has its own limitations that hinder its wide-spread deployment (Section 2.1.2). Int-Serv, which requires per-flow signaling and state maintenance, does not scale well as the user population grows. Although Diff-Serv approach is scalable, it only manages to provide coarse-grained performance assurance. In either case, end-to-end QoS is impossible without inter-domain resource control mechanisms.

The earliest work to address inter-domain resource provisioning issues and attempt to bridge the gap between Int-Serv and Diff-Serv is the Bandwidth Broker (BB) Architecture (Section 2.2.1). However, the reservation and admission control mechanisms within the BB proposal only consider local measurements at a single node (ingress router) or between a single pair of neighboring domains (for inter-domain reservations) and fail to reflect the traffic fluctuations and congestion levels in other parts of the network.

In this dissertation, we propose a new architecture called Clearing House (CH) to provision the intra- and inter-domain link capacity to provide statistical QoS such as maximum packet loss rate and latency. The two key design principles that make CH scalable are: *hierarchical approach* and *aggregation*, which we will explain in detail in the next chapter. In short, an ISP can be partitioned to several smaller domains, each associated with a CH-node. The resource control tasks are then distributed to the various CH-nodes that form a hierarchical tree. The CH exploits the predictability of aggregate traffic to establish and adapt intra- and inter-domain reservations while requiring only aggregate state maintenance. In our model, per-flow admission control is only performed at the ingress routers, but our algorithm considers network-wide traffic distribution in making admission control decisions. We also provide a mechanism to police admitted flows and detect malicious flows to ensure that the end-to-end performance of legitimate flows is protected.

Table 2.1 and 2.2 compare how our approach is different from the previous work. The details of the CH architecture and its various resource control mechanisms are presented in Chapter 4, 5, and 6.

Table 2.1: Comparisons between the Clearing House approach and previously proposed architectures.

Proposed Architectures	Properties	Scalability	QoS Guarantees
Int-Serv	Flat structure. Uses RSVP protocol & soft state approach.	Limited. Per-flow signaling & state maintenance at all routers.	Strong per-flow, end-to-end QoS.
Diff-Serv	Flat structure. Uses DHCP in IP- headers to indicate traffic requirements.	Scales well. Only edge routers keep per-flow states.	Coarse-grained, per-hop performance assurance for traffic aggregates.
Bandwidth Broker	Two-tier model. One BB per domain to manage resources.	Scales well, except for large domains. BB & edge routers keep per-flow states.	Coarse-grained end-to-end performance via concatenating pair-wise SLAs.
Clearing House	Hierarhical structure.	Scales well. Edge routers keep aggregate states.	Statistical end-to-end QoS, e.g., maximum loss rate and delay. Btw Int-Serv and Diff-Serv.

Table 2.2: Comparisons between our resource control schemes and related work.

Proposed Solutions	Reservations (resv)	Admission Control (adc)	Traffic Policing (tp)
Int-Serv	Per-flow, end-to-end resv via RSVP based on user-specified parameters.	Per-flow and per-hop on end-to-end path using worst-case bounds.	Per-flow policing at all routers.
Diff-Serv	Per-traffic class, via configuring schedulers like WFQ.	Per-flow, only at ingress routers using single-node measurements.	Per-flow policing only at ingress routers.
Bandwidth Broker	Per-traffic class like Diff-Serv, and through pair-wise SLAs.	Per-flow, only at ingress routers using single-node measurements.	Not addressed.
Clearing House	Aggregate reservations for intra- & inter-domain traffic aggregates based on real-time traffic measurements.	Per-flow, only at ingress routers using estimated network-wide traffic distributions.	Aggregate policing with ability to detect individual malicious flows.

Chapter 3

Methodology

As described in Chapter 1, the main contribution of this thesis work is a comprehensive study of a lightweight, scalable control architecture built on top of Diff-Serv packet level mechanisms to provide better end-to-end QoS support for latency-sensitive applications. This chapter outlines a set of research methodologies that we follow to carry out our analysis, specifically, how we formulate the problems, examine existing systems and evaluate our proposed solutions. Section 3.1 presents the general framework that guides our study. In Section 3.2, we discuss how we model the various workloads and their performance requirements that later drive the performance analysis of our architecture. In Section 3.3, we describe the evaluation methodology: we conducted simulation experiments using both real traces and generated traffic, with a range of parameters driven by different performance goals.

3.1 General Framework

Our research methodology is best summarized by Figure 3.1. We follow an iterative process that comprises the following three phases:

- **Analysis**
  Before we know what constitutes a better QoS control architecture, we first need to model the characteristics of typical workloads and define their performance requirements. In particular, we focus on the latency-sensitive applications (LSAs) such as packet audio and video because this type of traffic require resource guarantees that are not supported by the current Internet. We will discuss mathematical models that capture the essence of real Internet workload. We also carry out a survey of prior work, as reported in Chapter 2, and determine the pros and cons of each approach.

- **Design**
  In our attempt to strike a balance between Diff-Serv, which provides differential treatment to traffic aggregates, and Int-Serv, which offers per-flow guarantees, we explore a new architecture called the Clearing House that combines features of the two previous approaches. In particular, we implement lightweight session-level control mechanisms such as resource reservations, admission control and traffic policing on top of a stateless Diff-Serv architecture. The initial design has been continuously refined based on

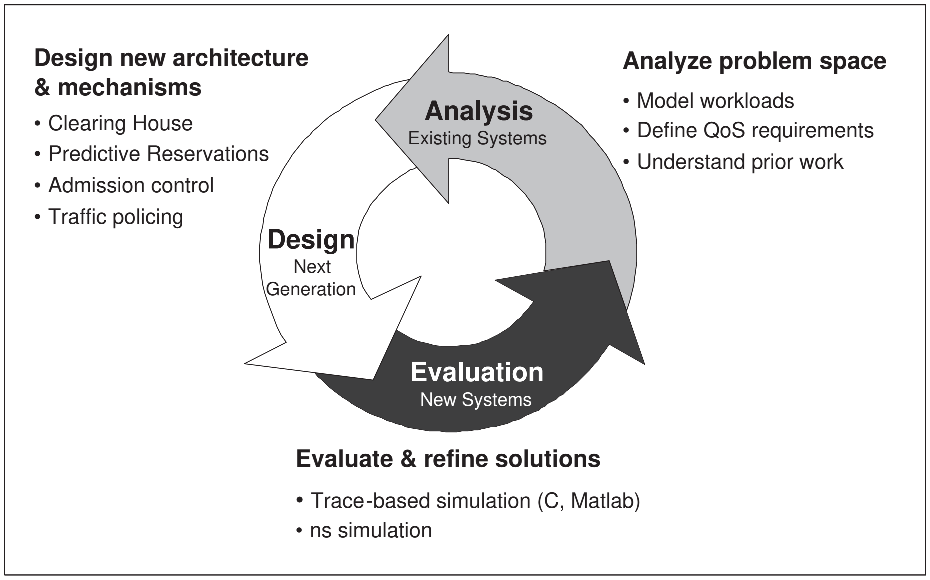


Figure 3.1: Iterative "Analysis, Design & Evaluation" phases.

discussions with researchers from two major Internet Service Providers that operate nationwide backbone networks in the United States.

- **Evaluation**
  The performance evaluation of our proposed architecture and algorithms are based on a combination of trace-based analysis, simulation experiments, and lab prototyping. We used simulations to examine scalability issues, determine the effect of various design parameters on system performance, and study the trade-offs involved at various operating points. The same set of experiments are often repeated for different "scenarios", where we vary the network topology, aggregate workload pattern, and individual source characteristics.

The next section provides detailed discussions on how we model packet audio applications that represent a typical LSA workload. Section 3.3 describes the general simulation settings but the details of each experiment will be presented in the subsequent three chapters where the corresponding algorithms and performance results are discussed.

3.2 Workload Modeling

In this dissertation, we consider two basic types of workload: data applications that can be sent as Best-effort traffic, and latency sensitive applications that require resource reservations and are sent as High-priority traffic. We have chosen Voice over IP (VoIP) as a representative workload of the latter, because interactive two-way conversations places a much more stringent delay requirements than other LSAs such as playback video. Section 3.2.1 presents the mathematical model that describes VoIP traffic. Section 3.2.2 documents the subjective testing and network measurements we carry out to quantify the performance requirement of VoIP in terms of network centric parameters such as delay and packet losses.

Besides VoIP, there are a wide variety of Internet audio applications that are also latency sensitive, including multimedia conferencing, distance learning, etc. To include these other applications in our analysis, we collected 70 packet audio traces from technical meetings, broadcasted lectures, and multimedia conferencing sessions. Section 3.2.3 documents the collection and analysis of these traces.

3.2.1 VoIP Traffic Model

VoIP refers to real-time delivery of packet voice across networks using the Internet protocols. The rapid growth of IP-based packet switched networks and the overall bandwidth efficiency of an integrated IP network make it an attractive candidate to transport voice connections. In fact, multiplexing data and voice results in a better bandwidth utilization than the traditional circuit-switched voice-or-nothing backbone in the PSTN (Public Switched Telephone Networks), which consists of over-engineered voice trunks. This justifies looking at VoIP as a workload for future Internet packet networks.

With silence suppression, each VoIP source can be modeled as an on-off Markov process. The alternating periods of activity and silence are exponentially distributed with average durations of $1/\beta$ and $1/\alpha$, respectively. An exponential variable X has the following density function:

$$f_X(x) = \begin{cases} ae^{-ax} & x > 0;\ a > 0 \\ 0, & \text{otherwise} \end{cases}$$

where $E[X] = 1/a$ and $var[X] = 1/a^2$.

The fraction of time that the voice source is "on" is $\frac{\alpha}{\alpha+\beta}$. We consider an average talk spurt of 30.83% and average silence period of 61.47% as recommended by the ITU-T specification [76] for conversational speech. In all our experiments, we set $1/\beta$ and $1/\alpha$ to be 1.004 s and 1.587 s, respectively. When the source is in the "on" state, fixed-size packets are generated at a constant interval. No packets are transmitted when the source is "off". The size of the packet and the rate at which the packets are sent depends on the corresponding voice codecs and compression schemes.

Let $X_i(t)$ be the instantaneous rate of voice connection i:

$$X_i(t) = \begin{cases} R & \text{when the source is active} \\ 0 & \text{when the source is silent} \end{cases} \tag{3.1}$$

where R is the voice bit rate (i.e., packet size/packet interval). The rate of transition from the state of transmitting "0 Kbps" to the state of "R Kbps" is λ while the reverse transition happens at the rate of μ.

Traditionally voice is Pulse Code Modulated (PCM) [77, 78] at 64 Kbps in the PSTN. PCM provides high quality reproduction of speech and comparable quality can be maintained with ADPCM [79]. Recent advances in compression technology have allowed highly compressed speech (16 Kbps and lower) that offer excellent voice quality in the absence of packet losses. In our experiments, we assume that the voice source generates constant bit rate (CBR) traffic of 80 Kbps when it is "active". [1] We use this on-off Markov process to generate VoIP traffic in our simulations (EXP1 model in Section 3.3).

[1] Assume 8 KHz, 8 bits/sample PCM codec was used with 20 ms frame per packet. With 12 byte RTP header, 8 byte UDP header and 20 byte IP header, the size of each voice packet = 20 (header) + 160 (data) = 200 bytes. The bandwidth required will be (200 x 8) bits/20 ms = 80 Kbps.

3.2.2 VoIP Performance Requirements

High quality interactive voice imposes many performance requirements on the underlying transport network. For example, one way end-to-end delay should be less than 150 ms to preserve the quality of interactive communication. In a circuit switched network, propagation delay is the only significant component in the one way end-to-end delay. In addition, this delay is constant component during the entire call duration, and therefore can be easily controlled. VoIP architecture, on the other hand, introduces new delay components such as: coding/decoding delay, packetization delay, queuing delays at intermediate routers/switches, and jitter compensation delay introduced by playout buffers. The multiplexing of VoIP and data traffic on shared links also introduces packet losses caused by buffer overflow at congested nodes. Latency and packet losses have adverse impact on the perceived voice quality, and therefore need to be bounded.

Our goal is to show how high quality voice can be supported with maximum utilization of resources if the network resource is provisioned properly and distributed admission control is implemented. To achieve this, we need to quantify the performance requirements of VoIP, by mapping the human perceived voice quality to the more tangible network centric parameters: packet loss and packet delay. Proper resource provisioning techniques can then be applied to provide statistical guarantees such as upper-bound for delay or loss profile.

Delay

In this dissertation, we ignore the delay introduced by the playout buffer. We also assume that:

- the end-to-end propagation delay is relatively constant and can be easily estimated,
- the sender uses the same codec throughout the call duration, and
- the sampling rate and packet size is fixed at the beginning of each call.

Since we are interested in investigating the effect of bandwidth allocation on voice quality, we try to segregate the effects of application-level QoS mechanisms. We assume that no application-level congestion control or rate adaptation are deployed at the voice sources. The only highly variable delay component in our model is queuing delay that occurs due to the multiplexing of voice packets, as well as the integration of voice and data over a shared link.

In our model, the end-to-end delay for VoIP are broken down to three components:

$$\text{packetization/transcoding} + \text{propagation} + \text{queuing delay},$$

as shown in Figure 3.2.

ITU-T Recommendation G. 114 [80] specifies that one-way transmission time for connections with adequately controlled echo should be in the 0-150 ms range to be acceptable for most user applications. We assume PCM transcoding introduces almost negligible delay if implemented in hardware (0.75 ms). The propagation delay is relatively constant and can be easily estimated. From [80], Public Land Mobile Systems contribute around 80 - 110 ms to one-way propagation time. Satellite systems introduce 12 ms at 1400 km altitude, and 110 ms at 14,000 km altitude. Optical fiber cable system contributes around 50-60 ms from

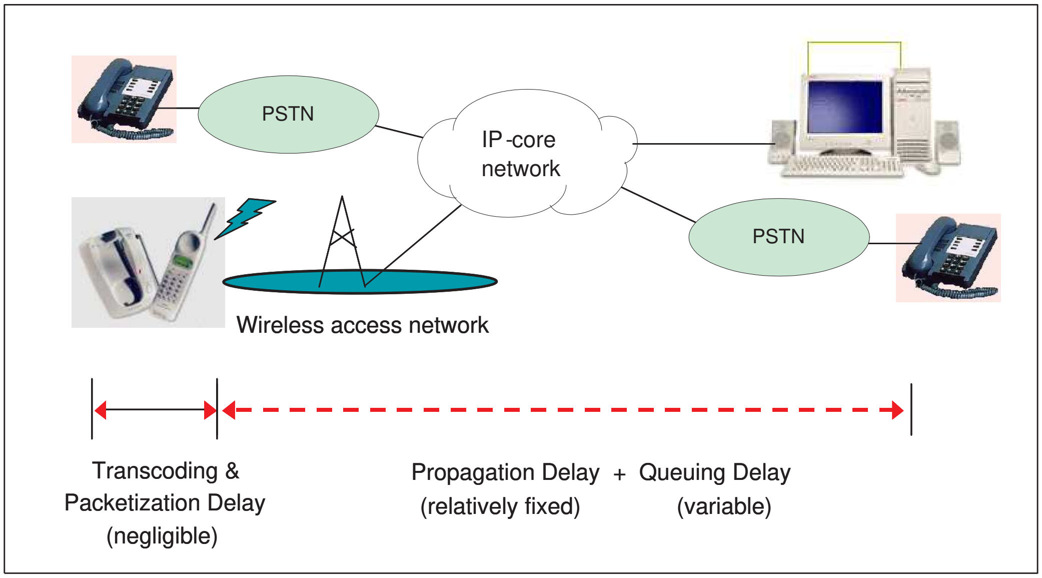


Figure 3.2: End-to-end delay components.

coast to coast in the United States. Assuming it takes 100 ms propagation delay for voice packets to be transported across the United States, the total queuing delay should be kept within 50 ms (150ms - propagation delay). Since queuing delay is the only variable part in our model, we need to budget the per hop queuing delay. From about 50 traceroute[2] [81] experiments, we found out that there were typically around 8-12 hops between a machines on the west coast and the east coast. Assuming that queuing delay is almost the same for each hop, we require the per hop queuing delay to be *at most 5 ms* and use this upper-bound to choose appropriate buffer size.

Packet Loss

Packet losses can cause further distortion beyond the unavoidable loss of information introduced by speech encoding/decoding and therefore should be minimized. We consider packet losses that are caused by buffer overflows in routers as well as discarding of delayed packets in the receiver playout buffer (i.e., if packets arrive at the receiver after too long a delay and miss the playout time, these packets are discarded and therefore considered lost). The impact of packet loss on voice quality is dependent on the voice codec used.

In the Fall of 1998, we used Visual Audio Tool (vat) [82] to run a simple subjective test to *map the packet loss rate to perceived voice quality*. Vat is a multi-party audio conferencing tool enabled by IP-Multicast [83]. We consider the following case: PCM codec with silence suppression, 8 kHz sampling rate, 8 bits per sample (contributing to 64 Kbps when the source is active), and 20 ms of voice samples per packet.

Figure 3.3 shows the experimental setup for the subjective test. The sound files of three sentences (about 6 seconds each) from the movie, “A Few Good Men” were down-

[2]Traceroute can be used to display the path taken by packets across network from one host to another host. This tool works by sending a series of UDP packets with different port numbers and TTL (Time To Live). A list of public servers that offer traceroute query service can be found at `http://www.traceroute.org/`.

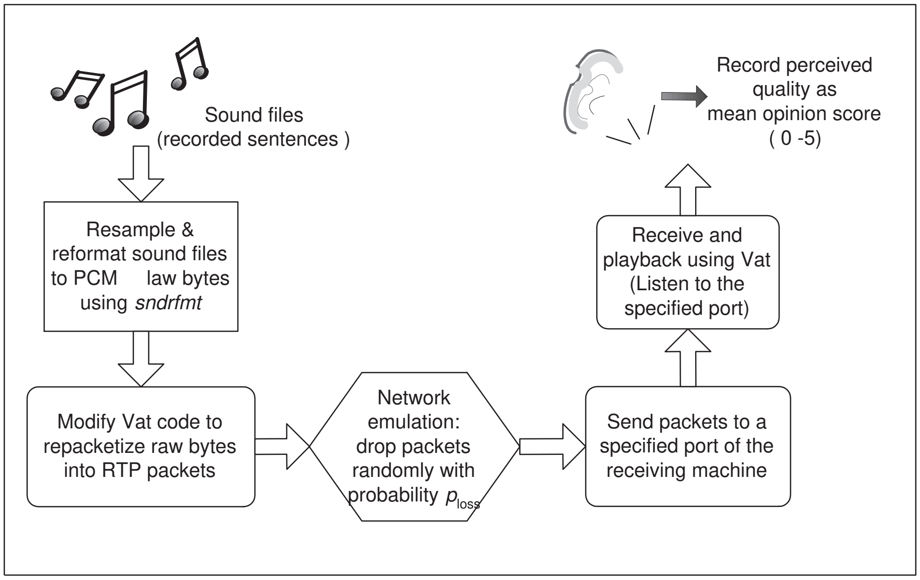


Figure 3.3: Experiment setup to carry out the subjective test that maps human perceived voice quality to different packet loss rates.

loaded and converted to PCM format with 8 kHz sampling rate. Since these sound files are in WAV format, we used *sndrfmt* program to resample the voice at 8KHz and convert the format to PCM and saved as μ-law bytes. *sndrfmt*[3] is a sound utility program that uses a library of audio hardware and sound-file access functions developed by Dan Ellis at the International Computer Science Institute (ICSI), Berkeley, CA. The voice samples were then packetized into RTP [5] packets with 12-byte RTP Header and sent through a simple network emulation that introduced random packet losses at different loss rates, p_{loss}. The packets are sent at every 20 ms interval. We ran vat at the receiving machine to listen to a specific port and playback the data. The perceived voice quality was scored on a numeric 0 to 5 scales with the following definitions: 5 = crystal clear, 4 = comprehensible but less clear; 3 = choppy speech; 2 = harder to comprehend sentences due to noise; 1 = can comprehend less than 50% of the sentence; 0 = gibberish noise. The same experiment was repeated for different p_{loss}, which was varied between 0 and 10%.

The result is plotted in Figure 3.4. Results show that the tolerable loss rates are within 1-2.5% and the speech becomes incomprehensible when more than 4% of the voice packets are lost. Note that packet voice using Forward Error Correction (FEC) [4] is more resilient to losses and therefore we would expect the curve to shift to the right in this case. On the other hand, the quality of voice connection using compressed speech is more sensitive to lost voice samples, and we expect the curve to shift to the left. The impact of packet loss on voice quality depends on the codec used, burstiness of losses, and frame sizes per packet, but this is out of scope of this project. For the rest of our analysis, we set the upper-bound packet loss rate to 1%, i.e., the QoS requirement is to send high-priority traffic from end-to-end with at most 1% loss rate.

[3]The complete "dpwelib" package that includes *sndrfmt* and many other utility programs can be downloaded from `http://www.icsi.berkeley.edu/~dpwe/dpwelib.html`.

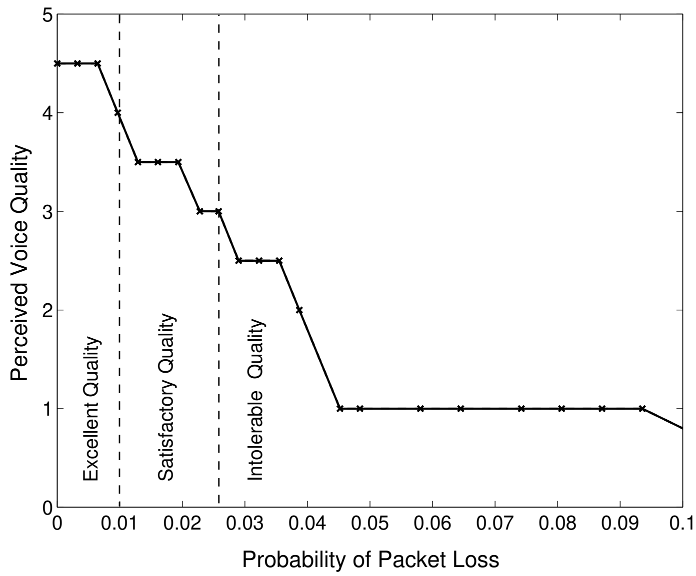


Figure 3.4: Subjective test results: how packet loss rate affect perceived voice quality.

3.2.3 Packet Audio Traces

To extend our analysis beyond VoIP, we collected 70 packet audio traces from a wide range of multimedia applications, including technical conference meetings, weekly lectures, technical demonstrations and social conversations. Based on these traces, we generate Internet workloads that have diverse characteristics to drive a subset of our simulation experiments.

These traces are classified into the following five categories according to their audio content:

- Traces collected from speakers who were giving a lecture or leading a discussion are classified as *lecture in classroom type*, where the audience (other participants) may interrupt the speakers with questions, leading to occasionally long pauses in the speakers' voice stream.

- Traces that represent participants who remain silent most of the time except for occasional questions or technical discussions are classified as *audience type.*

- In a situation where the speakers were participating in a group discussion or multimedia conference calls, the traces may have longer silence periods such as time spent listening to other participants or looking at a shared media board. These traces are classified as *conference call type.*

- The traces from pre-recorded technical demonstrations and lecture are classified as *pre-recorded speech type.*

- Voice traces when two speakers were engaged in social conversations or technical discussions are classified as *conversation type.*

Table 3.1: Summary of traffic traces.

Type	Number of Traces	Duration (minutes)	Voice Data (packets, MBytes)
Audience	32	(min) 1.26	616 pkt, 0.21 MB
		(max) 123.6	3747 pkt, 1.27 MB
Classroom lecture	11	(min) 4.4	6488 pkt, 2.21 MB
		(max) 71.8	100237 pkt, 34.1 MB
Conference call	26	(min) 0.5	528 pkt, 0.18 MB
		(max) 108.2	26819 pkt, 9.12 MB
Conversation	24	(min) 1.2	1553 pkt, 0.11 MB
		(max) 20.8	4287 pkt, 0.31 MB
Pre-recorded speech	1	6.6	9781 pkt, 3.33 MB

The traffic traces were generated by the following four sources and the breakdown of the traffic is tabulated in Table 3.1.

- **CSCW Electronic Classroom**
  58 traces were collected from a weekly Computer Science graduate-level class, Computer-Supported Cooperative Work (CSCW)[84] over 14 weeks in the Fall 1997. CSCW experimented with the idea of "Electronic Classroom" that was well-equipped with collaborative technology such as computers, video cameras, monitors, and a Xerox Live-Board. The class was held in a small, conference-style room. Some students would attend the course from their own office using remote collaboration tools (e.g., MASH tools like vic, vat and mb). 11 traces are classified as *classroom lecture*, 32 as *audience*, and 15 as *conference call*.

- **Research Groups' Multimedia Conferencing**
  11 traces were recorded from conference calls between professors, staff members, students and industrial sponsors of two research groups during January-September, 1998 and April-December 1999. All the traces are classified as *conference call*.

- **Pre-recorded Technical Demonstrations**
  We include in our analysis voice stream from pre-recorded technical demonstrations by graduate students, which we classified as *pre-recorded speech*.

- **CTS Test-bed with H.323 Gateway**
  24 traces were recorded from actual telephone conversations between students using the Computer Telephony Service (CTS) test-bed [85] from January-April 2000, where calls were made either from computer to computer, computer to normal PSTN phone or vice-versa via a H.323 Gateway.

All the participants in the CSCW class and multimedia conferencing communicate through three primary kinds of media: video, audio and shared white-board, using MASH [86] tools: vic, vat and "MediaBoard" (mb), respectively. These applications are launched on either Window-NT machines or Unix machines running Free-BSD. We are only interested in the voice packets recorded in these sessions/lecture.

Trace Processing

The voice traces were recorded according the MASH archive file formats [87, 88]. All data packets of one media type from a single source were stored in one file. Information such as the media type, the source identity, starting and ending time stamp were contained in the file header. The sender time stamp, receiver time stamp and sequence number of each packet were recorded. Voice packets were sent using RTP transmission format and 8 KHz 8 bits/sample PCM codec was used with 40 ms frame per packet. During the "talk" state, 340 bytes packets were generated every 40 ms (with 12 byte RTP, 8 byte UDP header and 320 bytes voice data).

We determined the talkspurt and silence periods by examining the interval between sender time stamps and locating gaps that were greater than 100 ms. Since the smallest meaningful element of speech, the phoneme, has an average size of 80-100 ms, we interpreted a pause smaller than 100 ms as a stop consonant or a minor break within the same talkspurt. We only ran statistical analysis on specific segments of the voice traces where actual conversations or lecture were in progress, and the rest of the traces were truncated. For example, a speaker sometimes had to restart his/her session because one of the tools (e.g., vic or "MediaBoard") failed to function. Although the voice packets were still recorded from the vat session, we truncated the packets recorded during the disruptions.

Section 3.3 will discuss how these traces are used in our simulation study.

3.3 Performance Evaluation

We have designed a new control architecture and resource provisioning mechanisms to deliver better QoS support to VoIP type workload outlined in previous section. The following three chapters (Chapter 4, 5, and 6) present the details of our proposed solutions and the design rationales behind them. To evaluate how well these mechanisms achieve our goals, we rely on a combination of simulation study and lab prototyping using both real-world and simulated topology. Besides network efficiency and end-to-end performance, we explore the architectural, scalability and practicality issues. It is also important to identify the degrees of freedom we have, e.g., the parameters that tune the several algorithms, and how they affect the trade-offs among contradicting performance goals, e.g., individual flow performance vs. overall network utilization.

3.3.1 Simulation Framework

Since it is infeasible to run large-scale experiments over actual wide-area networks, we resort to the following simulation experiments that capture the critical aspects of real-life Internet workloads and router technology:

Trace-based Simulation in C & Matlab

We developed a discrete-time event-driven C-simulator that implements the control logic of the Clearing House architecture and mechanisms. Two important inputs to the simulator are workload models and network topology. The arrival rate of the high-priority traffic is modeled as an independent Poisson process of intensity λ calls per second, and randomly pick from the pool of 70 traces (Section 3.2.3) to generate individual packet audio

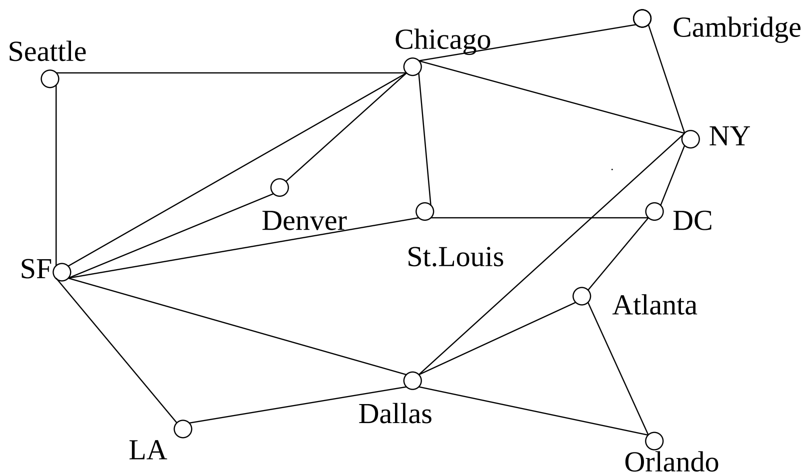


Figure 3.5: An example topology of a first-tier IP backbone network in the United States.

streams. We use the topology shown in Figure 3.5, which is an approximation of the AT&T WorldNet IP backbone as reported in [46].

With this simulator, we explored the efficiency and robustness of the CH-architecture in terms of resource utilization, call rejections and reservation setup time. The details of the experimental settings and simulation results are presented in Chapter 5. We also used Matlab to analyze the characteristics our proposed reservation scheme based on Gaussian traffic predictors.

ns Simulation

To study the packet-level dynamics and further evaluate our system in extreme cases with diverse types of workload, we ran more experiments using the ns simulator [89]. We constructed an overlay network on top of ns-objects such as nodes and links (implemented in C++), and added session-level control at the tcl level. We added modules to generate and process control messages transmitted using the ns UDP/IP protocol stack.

To evaluate the robustness of our proposed mechanisms against the diversity of Internet workloads, we consider four kinds of traffic source models in our simulations: EXP1, EXP2, CBR and PARETO. Each of these models has its own distinct statistical properties and can be used to represent a variety of latency sensitive applications, as discussed in the following.

1. **EXP1** has exponential on and off times as described in Section 3.2.1 with an average of 1.004 s and 1.587 s, respectively. The peak transmission rate is 80 Kbps, and the average is approximately 31 Kbps. EXP1 can be used to model voice applications, e.g., VoIP and audio conferencing, which use silence suppression.

2. **EXP2** also has exponential on and off times, but with an average of 100 ms and 900 ms, respectively. The peak rate is increased to 310 Kbps while keeping the average rate the same as EXP1, leading to a burstier source. EXP2 generates the most bursty

traffic among the four models that we consider and can be used to describe other workloads, such as video streams or multimedia conferencing applications, that have higher statistical variability than VoIP.

3. **CBR** is a constant bit rate source of 80 Kbps. Without silence suppression, packet voice streams can be represented using CBR model.

4. **PARETO** source has Pareto on and off times and has the same peak transmission rate (80 Kbps) as EXP1. A general Pareto density function is characterized by a shape parameter a and a scale parameter b:

$$f(x) = \frac{ab^a}{x^{a+1}} \text{ for } x \geq b.$$

   We set $a = 1.5$, and b is chosen such that the on and off times have the same average as EXP1 (1.004 s and 1.587 s, respectively). The aggregation of Pareto sources is known to exhibit long range dependencies [90, 91]. Hence, we use this model to describe interactive web applications that possess similar properties.

EXP1, EXP2 and CBR have exponential lifetimes with an average of 300s. The flow lifetimes of PARETO sources follow a log-normal distribution with average of 300 s.

We used ns to simulate different scenarios with these workloads to evaluate the admission control and malicious flow detection schemes. Results are documented in Chapter 6.

3.3.2 Lab Prototyping

We built a lab prototype of the Clearing House to evaluate certain performance metrics that could not be accurately quantified through simulations. One such metric is the overhead of implementing the various monitoring and policing mechanisms an edge router.

We extended the Click router [24] to support all the traffic policing and admission control functionalities of our architecture. Using this implementation, we measured the performance overhead incurred at an edge router, e.g., the degradation of system throughput. The current implementation works on Linux 2.2.16 and 2.2.17 kernels. Our architecture has been Operationally verified in our laboratory's test-bed. We will provide an overview of the implementation and performance measurements in Chapter 6.

3.4 Summary

This chapter gives an overview of our research methodology, including how we model the workloads of interest and how we evaluate the performance of our architecture through simulations and lab prototyping. The next three chapter document our technical contributions, design rationale and lessons learnt.

Chapter 4

The Clearing House: A Distributed QoS Control Architecture

The lack of a well-studied policy architecture to regulate network resource provisioning in a scalable manner has motivated our design of a Clearing House (CH) as an alternative solution. Examples of *network resources* include link capacity, buffer space, processing cycles at intermediate routers and storage space. In this dissertation, the word *resource* refers specifically to link capacity (bandwidth). The Clearing House is a distributed architecture that coordinates resource reservations within and across multiple domains based on statistical estimates of aggregate traffic demand. This chapter focuses on the CH architectural design, including its logical structure and the functionalities of its different components. In Section 4.1, we describe the design goals of the CH, and the assumptions we make about the network. In Section 4.2, we introduce background knowledge about the Internet network topology and traffic characteristics, and discuss how this knowledge affects our key design decisions. Section 4.3 and 4.4 provides an overview of the hierarchical CH-tree formation and the various resource control mechanisms. Section 4.5 summarizes the key features of the CH architecture, and discusses an example application where the CH is deployed to manage virtual private networks (VPNs).

4.1 Introduction

The concept of a clearing house has long been existent in the banking industry as an establishment where financial institutions adjust claims for checks and bills, and settle mutual accounts with each other. Even in the context of the Internet, the concept of the Clearing House is not entirely new. In 1995, a consortium of leading California Internet Service Providers formed the Packet Clearing House (PCH) [92] to coordinate the efficient exchange of data traffic from one network to another. The PCH member agreement includes cost of membership, peering connections and routing policy. For example, PCH members may exchange traffic between networks without any settlement fees. However, the PCH agreement does not provide any performance assurance or reflect any monetary compensations based on relative amount of traffic exchanged between members. Many architectural design issues involved in such an Internet Clearing House remain unexplored. On the other hand, increasing number of Internet companies are now offering on-line network resource brokerage by gathering guaranteed demand from the prospective customers and matching it with the sellers' capabilities. Examples include RateXChange's Real-Time Bandwidth Exchange (RTBX) [93], Arbinet Global Clearing Network's trading floor for minutes [94] and Priceline.com's future plan to offer time-block brokerage for domestic and international long-distance calls [95]. Such business models involve Clearing House mechanisms, which have not been studied carefully for the Internet scenario where bandwidth efficiency and QoS assurances are important.

4.1.1 Design Goals and Assumptions

Even today, most ISP backbone networks are managed manually. For example, ISPs rely on human operators to monitor their operational networks and reconfigure the routers when necessary, e.g., changing the preferred route between two endpoints to an alternate shortest path if congestions or link failures occur on the original path. In addition, the link weights for intra-domain routing protocols (OSPF [42] or IS-IS [44]) are chosen manually to perform load balancing. Several measurement studies [96, 97, 98] reveal that the distribution of Internet traffic over an ISP network can be highly unbalanced, e.g., link utilization on some critical links can be as high as 65-90% while other links are only 10% utilized. Obviously, there is a strong need for a more efficient way of allocating intra-domain resources (link capacity) and automate the required control process, especially for large ISPs that span over 500 nodes. Similar techniques are essential for managing resource allocation across multiple ISP domains to improve end-to-end network performance. Inter-domain resource control encounters an additional set of challenges that are not existent in the intra-domain case, including trust issues between different competing ISPs. For example, an ISP may not be willing to share information such as internal topology or backbone measurements with its neighboring peers. The lack of such knowledge can impact the effectiveness of inter-domain traffic engineering and resource management schemes.

In this chapter, we address the above issues by proposing a distributed control architecture, which we call Clearing House (CH), to coordinate intra- and inter-domain bandwidth allocation. An ISP can deploy a local CH to manage its backbone network, while relying on a third party (global) CH-node to coordinate inter-domain SLA negotiation and resource management. One of the basic design requirements of the CH architecture is to extend rather than replace the existing network devices, protocols, and implementations of inter-domain policies to minimize the development cost. The CH enhances the services and performance of the network by adding some functionality to the network access routers (or edge routers) and leveraging information from traffic monitoring devices.

The main goals that drive our design of the CH architecture are:

- **QoS Provisioning:** The CH attempts to provide an end-to-end coarse-grained QoS assurance by performing aggregate resource reservation along the path from source to destination host networks. This approach tradeoffs the fine-grained QoS assurance in order to preserve scalability and reduce signaling complexity. Per-flow admission control is performed only at the edge routers, but it should take into consideration the reservation status and traffic fluctuations within the domain.

- **Scalability:** The CH has a hierarchical tree structure that can incrementally scale to support a large user base (i.e., large geographic regions and large volume of simultaneous calls). We strive to minimize the number of states maintained in each node of the CH and the backbone routers.

- **Efficient Network Utilization:** The CH attempts to optimize the overall throughput while preserving the QoS of admitted calls by performing admission control based on information of the entire network stored in the CH database, e.g., reservation status and available bandwidth of inter-domain links. The accuracy of this information depends on the time granularity by which the database is updated.

- **Robustness Against Malicious Flows:** To preserve the overall QoS assurance and minimize the impact of malicious activity on the performance of innocent flows, the CH should provide mechanisms to police admitted flows, detect and isolate the malicious flows as fast as possible.

- **Secure Real-time Billing:** The CH is a distributed database that can store the billing prices, quality and latency provided by various ISPs. It can inform ISPs and customers about the available bandwidth, bandwidth demand, and reservation costs. This aspect of CH has been explored in [99] and is not a focus of this dissertation.

- **Support for Multicast Operations and Mobility:** The CH infrastructure can be easily extended to support multicast operations by coordinating resource reservations and cost-sharing between the group members at different level of the multicast tree. The CH can also keep track of the dynamic path changes and modify resource reservations accordingly to support mobility. This is part of future work, and is out of the scope of this dissertation.

We address mainly the first four design goals in our thesis work. Specifically, we design control mechanisms within CH to establish and negotiate aggregate resource reservations both within an ISP and between neighboring domains in a hierarchical manner. For example, the CH ensures that there is sufficient link capacity on intra- and inter-domain links to carry the VoIP traffic so that the maximum loss rate and delay are below the acceptable thresholds for perceived voice quality. We will not discuss how the reservation requests are translated to a specific traffic control agreement (TCA) that can be understood by the edge devices, or how these TCAs are delivered to the edge routers.

In designing the CH architecture, we make the following assumptions:

- The networks are capable of providing different service levels through a combination of packet marking, scheduling and queue management mechanisms. We assume network edge routers can verify whether the QoS assurance agreement is met by measuring the packet loss, average queuing delay, delay variance, etc.

- Every routing domain has the capability to monitor and collect statistics of the incoming and outgoing traffic. We assume this information is trustable, and will be used by CH to negotiate resource reservations with neighboring domains.

- Control paths (e.g., reservation requests) and data paths are separated. We decouple call setup and resource reservation procedures to reduce the overall response time and increase the system throughput.

- Only latency sensitive applications (LSA) such as voice and video need resource reservations and are classified as "high-priority". For the rest of our discussions, *flow* or *traffic* refer to high-priority packet streams that are affected by our session-level control mechanisms.

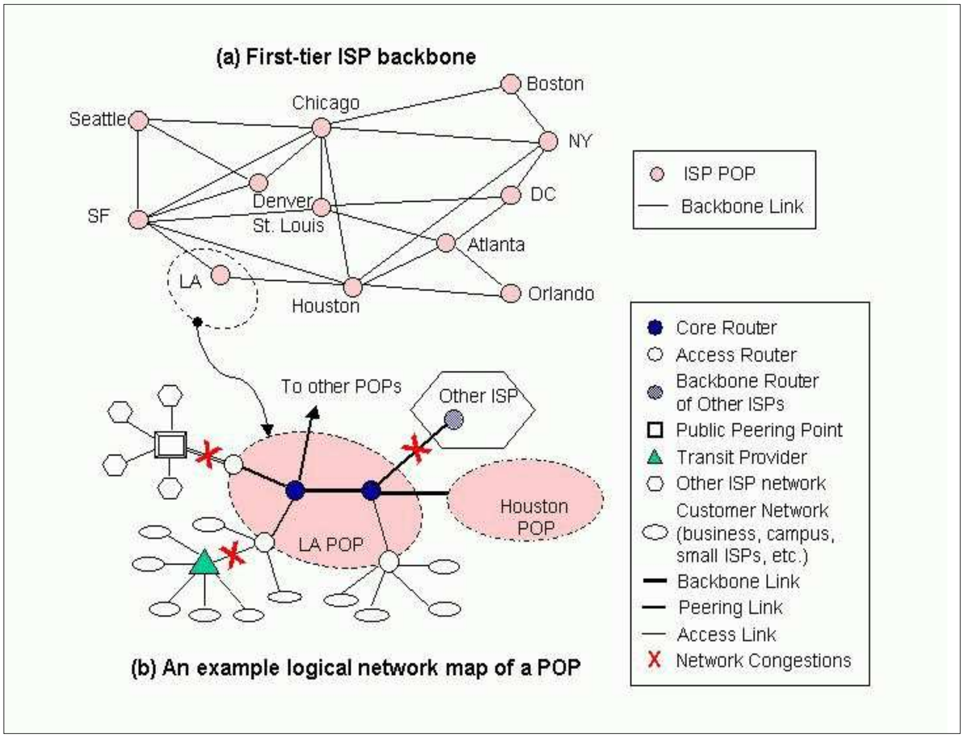


Figure 4.1: An example first-tier ISP backbone and its logical network map.

4.2 Design Rationale

In this section, we describe some basic properties of current Internet network topology, traffic characteristics and economic models based on our discussion with two major ISPs, and discuss how these considerations affect our design rationale.

4.2.1 Background: The Internet Structure

Internet Service Providers (ISPs) are segmented into tiers depending on the size of their network and number of subscribers. According to the xSP Forums[1], there are three ISP Tier-levels:

Tier-1: Backbones These are big ISPs either have their own nationwide backbone or over 1 million subscribers. There are about ten Tier-1 ISPs in the United States.

Tier-2: Regional Tier-2 ISPs usually own local regional backbone and/or support over 50,000 subscribers. These ISPs can also offer state or nationwide access services by peering or subscribing to Tier-1 ISPs.

Tier-3: Little, Local & Lots Local service providers that make up the majority of the ISPs belong to this category. Tier-3 providers typically support less than 50,000 users and offer local services only.

[1] `http://www.xspsite.net/isp/tiers.htm`

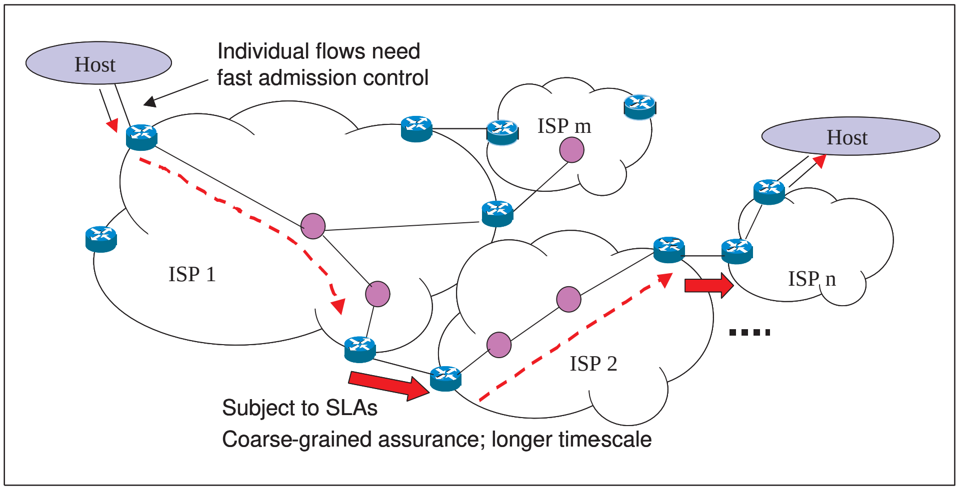


Figure 4.2: Multiple-ISP scenario.

A typical first-tier ISP backbone[2] in the United States generally has 15-25 Points-of-Presence (POPs) located in major cities throughout the country, as shown in Figure 4.1. The fan-out structure of POPs varies from city to city. For example, the number of edge routers (ERs) connected to the core routers (CRs) inside a POP, usually in the range of 10-20 [101], depends on the number of customers in that region of the country. A POP generally consists of high-speed backbone links (0.6-10 Gbps) connected to the core backbone and low-bandwidth edge links (45-155 Mbps) connected to Local Access Providers (LAP) or customer networks through ERs. 30-50 of such edge links can be terminated at the same ER. Customers that have direct connections to a POP include corporate networks, university campuses, web-hosting co-location sites, and modem pools. Individual users gain Internet access through LAPs. As mentioned in [102], the ISP backbone may also connect to neighboring private peers, public exchange points or transit providers via separate peering links. Past studies have indicated that it is at these interconnection points, where large amounts of traffic converge and the backbone pipes meet the narrow access links, that congestion occurs[3]. This often results in packet loss and unreliable QoS. The CH architecture is designed to improve end-to-end performance by rationing the number of high priority flows admitted by the edge router and managing network resources on congested links based on continuous network monitoring.

The task of resource provisioning is not confined to be just within an ISP domain, since end-to-end connections often span multiple domains, as shown in Figure 4.2. We treat the traffic demand coming from a private peer or transit provider differently from the high priority flows generated by end-hosts, because the resource allocation for both cases happen in vastly different time-scales and granularity. In the former, the traffic is usually subjected to peering agreements or Service-Level-Agreements (SLAs) [10] that reflect aggregate traffic performance (e.g., maximum round-trip delay). SLA renegotiation and the corresponding resource allocation decisions take place over longer time-scale, e.g., weeks or months. In

[2]For proprietary reasons, we do not have access to the exact backbone topology. Example ISP network maps are available from `http://www.cybergeography.org/atlas`.

[3]`http://www.internap.com`

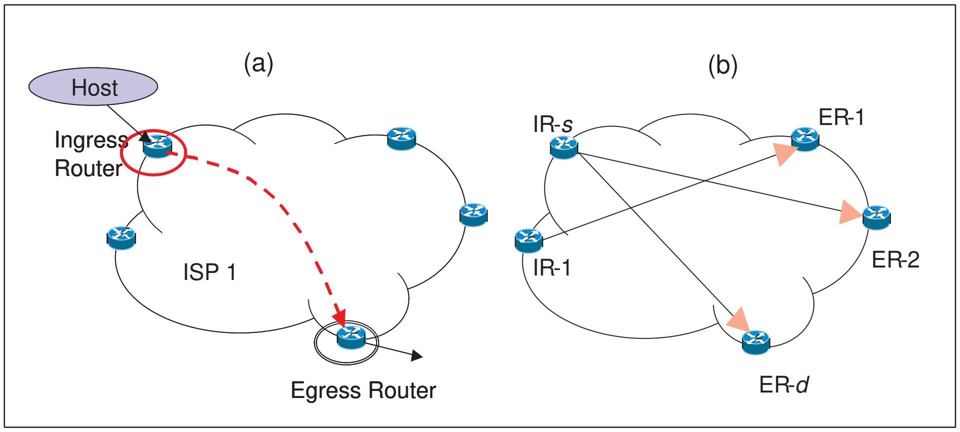


Figure 4.3: New service model: IE-Pipes(s, d) between specific ingress router IR-s and egress router ER-d.

the latter, the reservation requests from individual flows usually need fast admission control decisions, e.g., within ms, and the aggregate traffic demand fluctuate in a smaller time-scale, e.g., hours.

4.2.2 Key Design Decisions

The above observations about the Internet have led to three major design decisions that contribute to the scalability and robustness of our architecture.

Decision 1. New Service Model: IE-Pipes

Our first design decision is to treat ingress and egress routers within an ISP domain as endpoints instead of individual hosts. An ingress router (IR) is the point at which a flow enters into a domain, while an egress router (ER) is the exit point from the domain (Figure 4.3a). We define a new service model called IE-Pipe(s, d) that provides performance assurance for aggregate traffic between a specific IR-s, and a specific ER-d (Figure 4.3b).

Decision 2. Hierarchical Logical Structure

The Clearing House has a hybrid of flat and hierarchical logical structures. We introduce a local hierarchy within large ISP domains to distribute network management tasks to various CH-nodes. This helps reduce the amount of state information maintained at each CH-node, and there is no single point of failure. In addition, the distributed and hierarchical nature of the CH allows us to build in redundancy and system fault tolerance.

In our model, various basic domains (based on administrative or geographic boundaries) are aggregated to form *logical domains (LD)*, as shown in Fig. 4.4. These logical domains are then aggregated to form larger logical domains and so forth. This introduces a hierarchical tree of the LDs and a distributed CH architecture is associated with each LD. Individual CH-nodes can be thought of as agents that maintain aggregate reservations for all the links within the same domain at a particular hierarchical level. The reservations be-

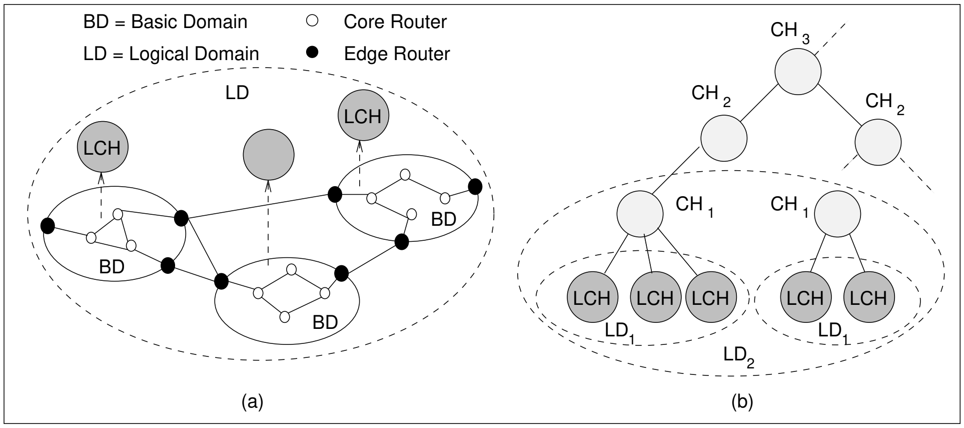


Figure 4.4: (a) Local Clearing House (LCHs) associated with their basic domains that lie within a single logical domain. (b) An hierarchical CH-tree with multiple levels of logical domains.

tween neighboring domains are monitored by the parent CH-node. This hierarchical tree of CH-nodes form a "virtual overlay network" on top of existing wide-area network topology.

In the multiple-ISP scenario where the ISPs do not share mutual trust, the top-level CH-node for each ISP maintain peer-to-peer relationships with one another to regulate resource allocation for aggregate traffic exchanged between two domains. In Section 4.3.1, we illustrate how the logical CH-tree is mapped to the physical network, and how CH can be deployed by various independent ISPs.

Decision 3. Decoupling Reservation and Admission Control

For scalability reasons, the CH does not maintain per-flow reservation at any routers. Instead, aggregate reservation is set up for a group of flows that share the same link and request for the same class of service (high priority). We assume that the path for a specific pair of ingress and egress routers (or IE-Pipes) can be determined from the underlying routing protocol, and it remains stable relative to the individual flow lifetime. The core routers only need a simple two-level priority scheduler to provide Expedited Forwarding (EF) service to high-priority packets. As a result, the edge routers only maintain aggregate state information, while the core routers remain stateless.

The actual resource allocation is decoupled from the admission control process, which is necessary to ensure that there is sufficient network resources to deliver the end-to-end QoS assurance. In our architecture, admission control for individual flows are performed only at the edge routers but it leverages the knowledge of the global *traffic matrix* and topology within an ISP domain. A traffic matrix captures the distribution of aggregate traffic demand between different pairs of ingress and egress routers. The CH infers this traffic matrix through passive monitoring of the traffic arrivals.

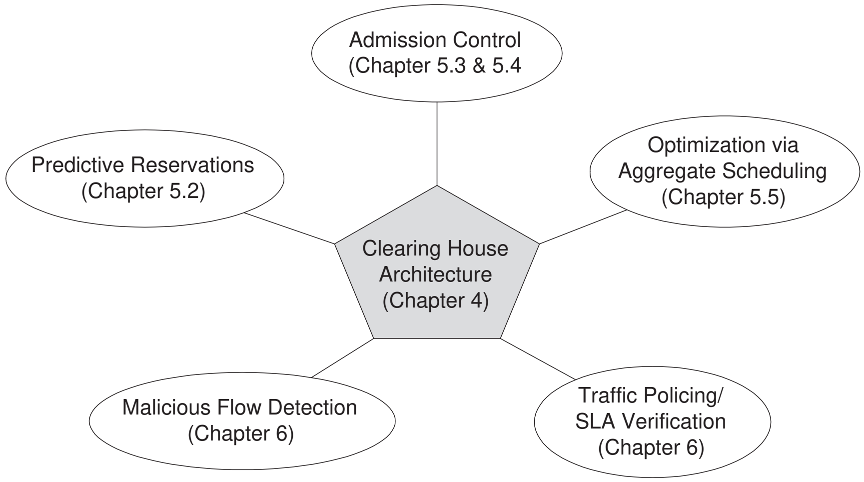


Figure 4.5: Thesis roadmap: The CH architecture and its various resource control mechanisms.

4.3 Clearing House Architecture

The CH is a distributed control architecture that performs four major resource management tasks:

- **Monitoring and Measurements**
  The CH collects aggregate traffic statistics through passive monitoring and measures the network performance such as packet loss rate and delay. It also estimates traffic matrix within each ISP domain by inferring the traffic demand distributions between every pair of ingress and egress routers.

- **Intra- and Inter- Domain Aggregate Reservations**
  Since LSA traffic has more stringent delay bounds, best-effort traffic must not preempt LSA traffic, and hence the latter should be served in a higher priority class. The CH architecture provides aggregate reservations for high-priority traffic within and across domains.

- **Admission Control**
  Since the LSA traffic must co-exist with current best-effort traffic, we need to limit bandwidth allocated to the LSA traffic to prevent starvation of best-effort class. In addition, the admission of high-priority flows that compete for network resources should be controlled to ensure satisfactory end-to-end performance. The CH leverages the knowledge of traffic matrix in performing admission control at the edge.

- **Traffic Policing for Malicious Flow Detection**
  Traffic policing is useful for monitoring admitted flows and detecting malicious behavior. The words *malicious* and *misbehaving* are used interchangeably to describe flows that violate their allocated share of bandwidth.

Figure 4.5 shows our thesis roadmap and the various CH mechanisms that we will address in Chapter 5 and 6. In the rest of this section, we illustrate how the hierarchical CHtrees are formed and discuss how the various CH control blocks interact with one another (the shaded area in Figure 4.5).

4.3.1 Hierarchical CH-Tree

First, we define several terms that we use in our discussions:

- A *basic domain* refers to a basic routing domain in the network. For example, a basic domain can be a small subset of backbone networks owned by a specific Internet Service Provider (ISP) which serves multiple host networks. We assume that the Internet can be divided into non-intersecting basic domains.
- A *logical domain (LD)* is a collection of adjacent basic domains that are clustered to form a larger domain, which may refer to geographic boundaries (e.g., states, or small countries) or for administrative reasons (e.g., campus, company etc). On the other hand, a big ISP backbone network can span across multiple domains.

The various logical domains can be clustered to form a larger logical domain. We can repeat the same process until we are left with one logical domain that represent the whole network. Together, these domains form a hierarchical tree, which we call a *CH-tree*. A distributed CH architecture is associated with every LD represented by a node in this tree. A CH-node at a particular level of the CH-tree maintains the reservation states of the LD, which is the union of all the sub-LDs whose states are maintained by its children CH-nodes. The actual number of CH-nodes in the distributed architecture will vary as a function of the size of the LD, and the level of the LD in the hierarchy. Mirror sites can be added to every CH-node to support fault tolerance and higher availability.

A CH in the hierarchy aggregates all inter-LD call requests to a particular domain and sends this aggregated request to the parent CH. In other words, all call requests between two LDs would be aggregated as a single request at a parent CH. Therefore, a CH of a LD that is a collection of K sub-LDs would contain $O(K^2)$ aggregate reservation requests. Only the CH at the local operators (at the leaf nodes of the CH-tree) maintain per-flow state information.

Although it is easy to extend the depth of the CH-tree to represent the whole network, our preliminary analysis considers the case of a two-level tree with one parent CH-node associated with an ISP domain and multiple children nodes (associated with basic domains). We quantify the performance of Clearing House and reservation strategies in this simple case and the simulation results are presented in Chapter 5.

4.3.2 An Illustration

A Single ISP Case

First, we consider a single ISP scenario. Since all the routers and CH-nodes within an ISP belong to the same administration (with the same AS number), we assume they share the same trust entity and therefore can exchange sensitive information such as traffic statistics and preferred routes among themselves. For ease of discussion, we denote the leaf nodes of the CH-tree as *Local Clearing House (LCH)* and the CH-node on top of the hierarchy for an ISP is called a *Parent Clearing House (PCH)*. Within an ISP, a basic domain (BD) typically represents a local Point of Presence (POP) that is connected to

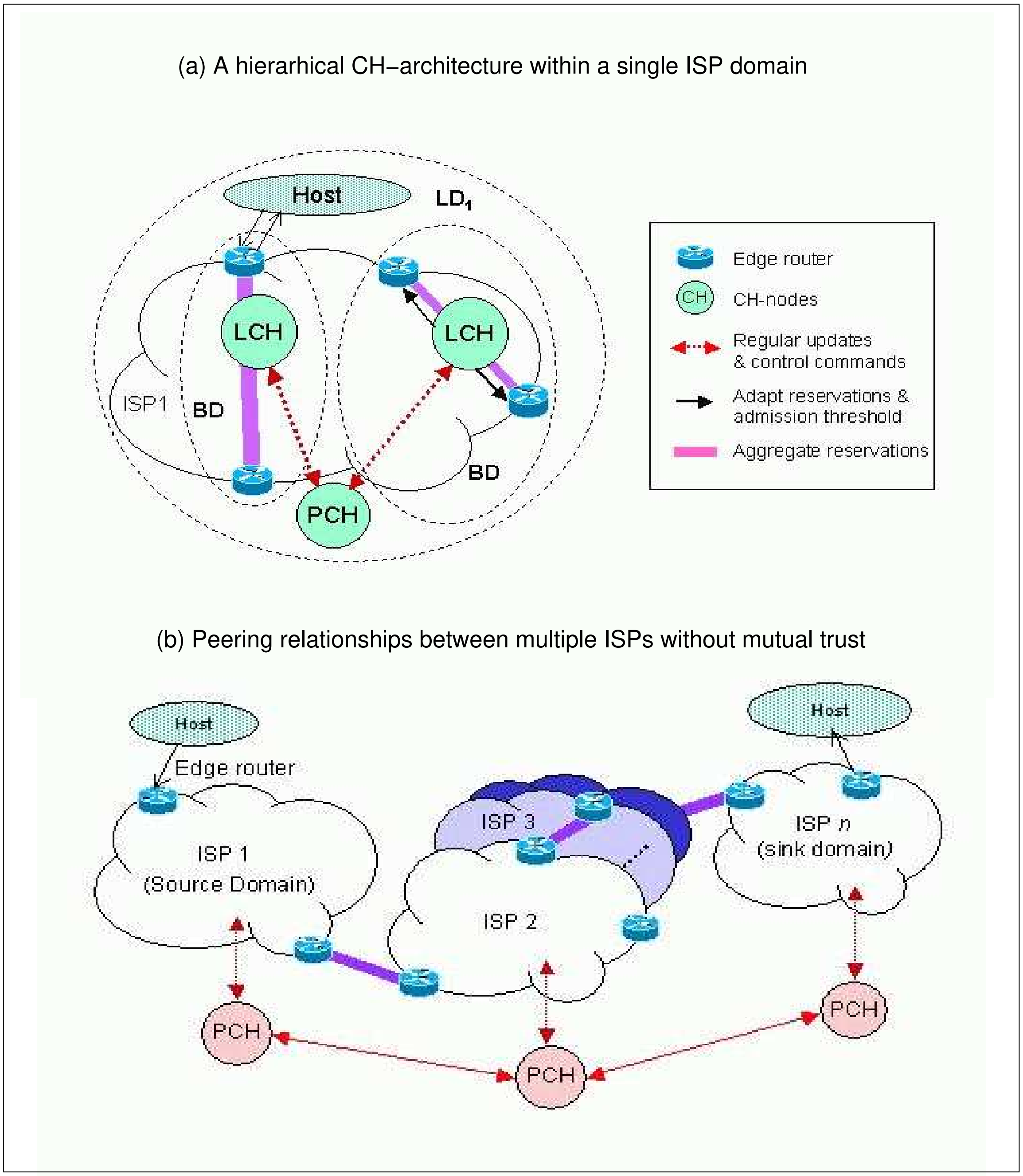


Figure 4.6: An illustration: A hybrid of flat and hierarchical CH-architecture within and across ISP domains.

downstream customer (host) networks. In general, a BD can be a subtree of a tier-1 or tier-2 ISP, or even the entire domain for a tier-3 ISP (as discussed in Section 4.2.1) that forms a regional network (e.g., a city or a county). There can be 10-25 nodes within a BD, depending on its geographical scope (city vs. county) and user population.

Figure 4.6a shows how a simple two-level CH-tree can be constructed within an ISP. This specific example shows two BDs (or POPs), each associated with a LCH, that are clustered to form a LD that represents the entire ISP. The LCHs send regular updates such as estimated traffic demand distributions and reservation status to upper-level CH-nodes (in this case, the PCH). The PCH collects traffic reports from all LCHs and construct the traffic matrix for the entire ISP.

Multiple-ISP Scenario

The local CH-hierarchy within ISP domains are hidden in Figure 4.6b. The independent service providers may not trust each other to share information about internal routing topology for traffic distributions. Therefore, we assume a flat structure at the top level where the PCH associated with each ISP form a peer-to-peer network to coordinate inter-ISP resource allocations. It is sufficient to reveal only aggregate traffic statistics, e.g., total bandwidth requirement to exchange high-priority traffic between any given pair of ISPs.

4.3.3 Local, Intermediate, and Parent Clearing House

We assume that the basic domains are non-overlapping to ensure that a user at a particular location has a unique LCH to contact for resource reservation or billing purposes.

The LCH is responsible for the following set of operations:

- An LCH keeps track of the amount of existing reservations and the available bandwidth on all the links between edge routers within its own BD. Based on the statistics of the intra-domain traffic, an LCH performs advance resource reservations on the intra-domain links. It also makes local admission control decisions when a new intra-domain reservation request arrives.

- An LCH also monitors the aggregated incoming and outgoing traffic exchanged with other neighboring BDs and uses these statistics to estimate the future bandwidth usage. The predicted bandwidth requirement for high-priority traffic that traverse between its own BD to every other BD within the ISP is reported to the CH-node higher up in the hierarchy (PCH in this example). If there are K BDs within the ISP domain, the traffic report would be $K-1$ vector.

- Based on the available network resources on the end-to-end path, the PCH will adjust the inter-domain reservations accordingly and sends updates to the LCH. Upon receiving acknowledgments from the PCH, the LCH will adapt resource allocation on its edge routers as well as the admission threshold for different IE-Pipes.

- If the existing inter-domain load is less than the allocated bandwidth, new requests will be admitted. Otherwise, the LCH aggregates inter-domain reservation requests as a single request and forwards it to the PCH. If there are sufficient network resources on the end-to-end path, the LCH will receive acknowledgments from the PCH to enhance the aggregate reservations, and the new requests will be admitted. Otherwise, the requests will be rejected.

In general, an intermediate CH-node acts as the coordinator among the various BDs and handles resource allocation for all inter-domain calls:

- A CH-node keeps track of the links that run between children sub-domains and their corresponding reservation status and network performance such as latency, average queuing delay, and packet loss rate.
- Based on the traffic statistics collected from all the children-LCHs, a CH estimates bandwidth usage on a particular inter-domain link and performs advance reservation accordingly (see Chapter 5).
- A CH-node aggregates reservation requests received from its children LCHs, and performs advance reservations for the inter-domain links that lie within its LD. If the reservation request involves links that connect to neighboring LDs at the same level, the reservation request will be forwarded to the upper-level CH. A CH-node services reservation requests for aggregated traffic instead of individual calls.

The parent CH-node (PCH) sits on top of the hierarchy for a particular ISP and adapts trunk reservations between different domains. In addition to the general tasks for CH-nodes described above, the PCH can advertise the costs of reserving bandwidth on their internal links to other neighboring PCHs. For example, the service providers can offer various prices based on the domain of the final destination (e.g., call Canada 7/9 cents/min) and the traffic load [103]. The PCH can then choose the optimal route that satisfied the performance constraints while minimizing the total costs.

4.4 CH Control Flows

Our architecture requires explicit REQUEST and TEARDOWN messages for each high-priority flow for admission control and accounting purposes. These control messages are generated by either a customer router or a proxy and are sent as UDP packets at the same priority level (high) as the data packets.

Figure 4.7 shows the essential control blocks within the LCH and how it interacts with the edge routers. It also shows the REQUEST, ACCEPT/REJECT and CONFIGURE control messages between the LCH, the edge routers and the host proxy. When a new REQUEST message arrives, the LCH performs admission control based on its current knowledge of intra- and inter-domain reservations and existing load. It will then respond with an ACCEPT or REJECT message. If the flow is accepted, the LCH needs to update the traffic-policing unit in the associated edge routers through CONFIGURE message. If necessary, it uses the same message to adapt the resource allocation at the routers.

4.4.1 Resource Reservations

The CH architecture can support two types of reservations: advanced and immediate reservations. An advanced reservation (AR) is time-limited and resources are allocated

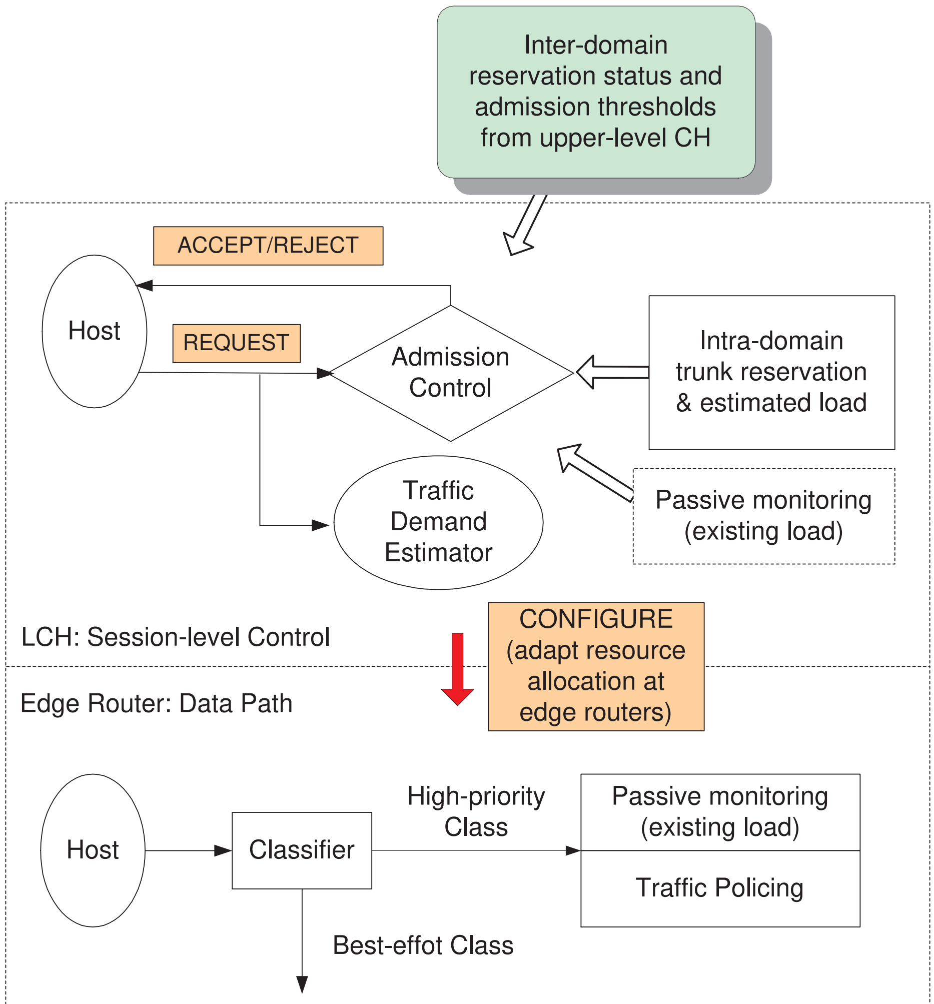


Figure 4.7: A logical view of the Local Clearing House (LCH) and its interaction with edge routers.

in advance based on statistical estimates of aggregate traffic over a particular link. We use advance reservations to reduce the call setup time, and the potential violation of QoS assurance if the traffic arrives before the resources are properly reserved. Such approach has been used for resource management in Virtual Private Networks (VPNs) as reported in [46]. Traffic statistics can be easily obtained by leveraging the existing traffic monitoring and measurement systems, through either third party organizations, e.g., MIDS Internet Weather Report (IWR) [104], Internet Traffic Report [105], or the ISPs themselves, e.g., Cable & Wireless USA [106] and AT&T IP Services [107]. We can also gather information from end nodes using software toolkit such as SPAND [108], which enables the networked applications to report the performance they perceive as they communicate with distant Internet hosts. Advance reservations only track the aggregate traffic pattern at a large time-scale (e.g., different hour of the day) and do not reflect the rapid fluctuations of local traffic volumes produced by end-users. Immediate reservations (IR), on the other hand, can be made on demand when the existing reservations become insufficient to accept the new admission requests. The local CH-nodes performs admission control to ensure that QoS assurance to the existing connections are not violated.

For our initial analysis, we consider the case with a two-level CH-tree within each ISP. We evaluate, with simulations, the performance, robustness and overheads of the CH architecture. The simulation framework and results are discussed in Chapter 5.

4.4.2 Caching and Aggregate Scheduling

We can employ two enhancements to improve the performance of the Clearing House, namely caching and RxW scheduling [109]. An LCH or GCH can cache intra-domain and inter-domain computed paths for previous reservation requests. This can reduce the service time of a reservation request at a CH. Since the number of logical domains maintained by a CH is small (10-50), a local cache can typically store all inter-domain paths. A local cache in a LCH can also store the price listings of various service providers to different destinations. RxW scheduling [109] is a very good algorithm for increasing the throughput of the CH. It schedules the aggregated call requests with the maximum value of $R \times W$, where R is the number of requests aggregated and W is the maximum waiting time of an aggregated request. This scheduling algorithm maximizes the throughput (number of call requests) serviced without unduly affecting the response time for call requests. Chapter 5 provides detailed discussions on how aggregate scheduling helps improve the CH efficiency and reduce the overall response time of reservation requests.

4.4.3 Admission Control

Whenever a sender wants to make a call to a receiver, there should be sufficient resources (e.g., link capacity and buffer space) along the particular path from the sender to the receiver to avoid packet losses and delays. Since on-line resource reservation is very costly, the goal of our design is to minimize the amount of per-link reservation that needs to be made for a particular call. Based on the reservation status within a domain, a particular path is chosen such that the number of new per-link resource reservations is minimized. If the LCH fails to locate any links with sufficient resources reserved to complete a chosen path, the ER will block the new call. The admission control decisions involve some trade-offs in the QoS assurance and the number of rejected calls.

So far, we have presented a general CH-architecture that can deploy both parameter-based and measurement-based admission control. We concentrate on the latter approach in this dissertation. The CH architecture allows an ISP to dynamically infer the traffic matrix within its own domain based on a collection of partial views from its sub-domains. With this knowledge, an ISP can make a better admission control decision for the incoming flows from its customer networks. A detailed description of the proposed scheme, Traffic-Matrix based Admission Control scheme (TMAC), can be found in Chapter 5.

4.4.4 Traffic Policing and Malicious Flow Detection

Another equally important task is to police the admitted flows to make sure that each flow uses its right share of allocated bandwidth and not more than that. To reduce the amount of state information maintained at the edge routers, we propose to aggregate flows for group policing. Chapter 6 presents the Malicious Flow Detection via Aggregate Policing (MDAP) scheme that is designed to uniquely identify malicious flows without keeping per-flow state at the edge router.

4.5 Summary and Discussions

We have designed a Clearing House architecture that coordinates intra- and inter-domain resource reservations for high-priority traffic. The scalability of the architecture is attributed to its hierarchical CH-tree structure and the aggregation of reservation requests at multiple levels of the logical tree. We use a Gaussian predictor to estimate bandwidth usage and set up reservations in advance to reduce the overall reservation setup time. The details of the predictive reservation scheme and evaluation of its effectiveness is presented in the next chapter (Chapter 5). Further analysis on how the throughput can be improved using aggregate scheduling algorithms is also reported.

In summary, the basic strengths of the Clearing House approach are:

- The state information that needs to be maintained by the entire ISP domain is shared between various CH-nodes in the local hierarchy. Since every CH-node maintains only the state for its own domain, this allow the entire CH-tree to scale better to a larger user base.

- The hierarchical model with aggregate reservations provides scalability of the architecture. Core routers do not need to maintain per-flow state information. The architecture supports easy insertion and deletion of the domains from the *CH-tree*. If a particular CH-node gets overloaded due to the growth of user-base, it is possible to split a logical domain (LD) into two or more sub-LDs, and create a new CH-node for the newly created LDs.

- The queue size of call requests in every CH is bounded due to aggregation of call requests at the children CH-nodes. This architecture is optimized for making decisions based on locality. End-to-end resource reservations can be set up quickly through the CH architecture, and therefore reducing the call setup time.

- The CH leverages knowledge of traffic matrix and routing topology within an ISP for admission control.

- Caching of inter-domain paths can enhance the performance of the system considerably.

4.5.1 VPN: An Example Application

Although we present the CH as a general architecture, one specific example where CH will be useful is for IT managers to manage a WAN (wide-area network) that interconnects corporate offices, remote and mobile employees. Corporations have turned to Internet VPNs to deliver performance, security and manageability to their various sites scattered across the country. However, existing SLAs[4] [10] between service providers (ISPs) and customers have focused on backbone performance guarantees, and do not reflect the end-to-end performance of individual applications. In addition, some fraction of the traffic may traverse multiple routing domains that belong to different ISPs. IT managers still face the challenge of provisioning the total capacity (VPN tunnels) efficiently among the various types of traffic to meet application requirements such as latency and reliability characteristics. A CH-architecture can be deployed in this case to handle intra- and inter- domain resource allocation. For example, IT managers can treat each corporate site as a basic domain, and introduce a CH-node at each site to monitor the traffic flow, adapt resource allocation, and re-negotiate SLAs with the corresponding ISPs when necessary. Various sites can be aggregated to form a larger LD, or several LDs, depending on the layout of the corporate network. The CH-nodes associated with these LDs can coordinate the aggregate resource allocation between domains that reflect on end-to-end performance requirements.

4.5.2 Other Resource Control Problems

In this thesis, we focus on using CH to allocate link capacity to provide statistical packet loss and delay guarantees to high-priority traffic. However, CH can be easily modified to address other resource allocation problems. The following are some examples:

Disk space allocation in storage area network: In today's high-technology economy, the storage needs for companies are growing exponentially due to the increasing dependence on "information". A storage area network (SAN)[5] is designed to meet this challenge by enabling any-to-any interconnection of servers and storage systems. It supports centralized management of both local and remote storage resources within an enterprise. The CH architecture can be used to perform load-balancing between different servers and control disk space allocations. For example, CH can exploit the data access patterns to determine how the storage system should be partitioned among a group of users. This can improve the overall disk utilization. CH can also be used to manage the sharing of link capacity between users in SAN to ensure rapid access to data and to avoid losses.

Distributed cache management for content distributions: Content distribution network (CDN) providers, such as Akamai[6], address the needs for e-commerce companies

[4] A service level agreement (SLA) is an explicit statement of the expectations and obligations that exist in a business relationship between two organizations: the service provider and the customer

[5] `http://www.storage.ibm.com/ibmsan/`

[6] `http://www.akamai.com`

to deliver streaming media, rich Web content and Internet applications to end customers with high performance and reliability. Akamai deploys thousands of servers (web caches) across hundreds of access networks worldwide. By placing the Web objects and applications at these caches that are close to the end users, Akamai eliminates the impact of congestion points in wide-area network on the performance of content delivery. The CH framework can be applied in this case to choose the optimal placement of these caches to achieve the required performance with the minimum number of caches. Using real-time measurements of cache performance, traffic load, and user access patterns, CH can perform load balancing to improve the overall efficiency of CDN networks.

Chapter 5

Intra- and Inter-Domain Resource Control

In Chapter 4, we presented the Clearing House (CH) as a resource provisioning architecture and highlighted some of the properties that contribute to its scalability. This chapter focuses on three specific control mechanisms within the CH architecture: resource reservations, admission control and aggregate scheduling of flow requests (the shaded areas in Figure 5.1).

For scalability, link capacity is reserved for aggregate traffic that share the same endpoints, which we call *trunks*, rather than individual flows. Assuming that the envelope of the aggregate traffic follows a Gaussian distribution, the intra- and inter-domain workload can then be characterized by its mean and variance. These statistics can be measured in real-time and used to dynamically adjust the trunk reservations on the associated links. The details of the predictive reservation strategies and the simulation study are described in Section 5.2.

In our architecture, admission control and resource reservations are decoupled (Section 4.2.2). While reservations are set up for aggregate traffic, per-flow admission control is performed only at the ingress point of an ISP domain to ensure that there are sufficient resources to satisfy the QoS requirement of the admitted traffic. In Section 5.3, we propose TMAC (Traffic-Matrix based Admission Control), an admission control policy that leverages the knowledge of traffic demand distributions between every pair of edge routers in a domain. The simulation study described in Section 5.4 is designed to explore the performance and robustness of TMAC.

In Section 5.5, we explore how the CH-node performs as the number of admission control requests grows large. We describe how TMAC can work across domains at different granularities using an example of a two-level CH architecture. This is a useful resource allocation technique for a first-tier ISP that spans multiple cities, each having its own local network called Point of Presence (POP). To improve the efficiency of the CH, individual requests for inter-domain flows are aggregated by a Local Clearing House (LCH) before they are forwarded to the parent CH-node for admission control. We explore an aggregate scheduling algorithm and evaluate, with simulations, its costs and benefits in terms of the reduction in setup time, call rejections and resource utilization by aggregating reservations.

Section 5.6 summarizes the key features of our proposed mechanisms and our simulation results. The results indicate that it is possible to provision the network for

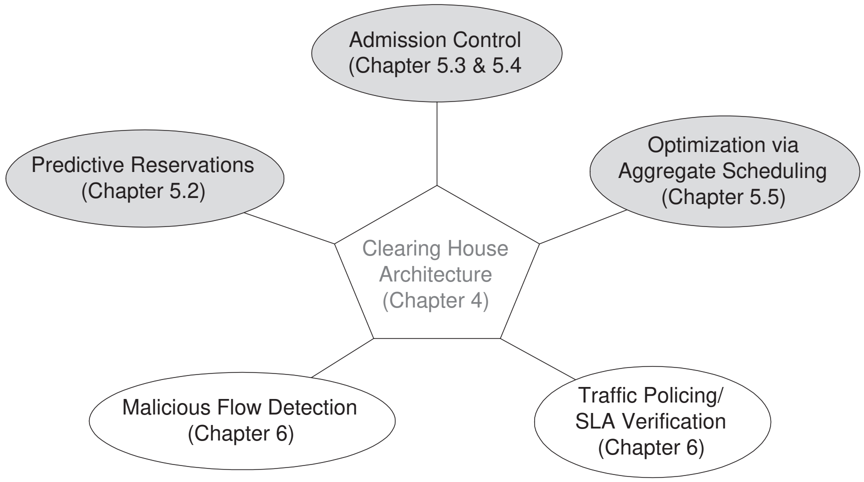


Figure 5.1: Thesis roadmap: The CH architecture and its various resource control mechanisms.

latency sensitive applications, such as VoIP, based on real-time measurements of aggregate traffic. For example, our proposed mechanisms manage to satisfy the QoS objective of VoIP (i.e., ¡ 1% loss rate) while maintaining 97% utilization level.

applications (i.e., less than 1% loss rate) with only 8% over-provisioning.

5.1 Introduction

5.1.1 Motivation

The unpredictable loss and delay in the conventional Internet can adversely impact the performance of real-time multimedia flows, such as audio and video conferencing. Toward this end, a number of architectures and algorithms have been devised to reserve network resources and ensure that the QoS objectives of such flows or class of traffic are satisfied. As described in Chapter 2, the two most well-studied approaches are Integrated Services (Int-Serv) and Differentiated Service (Diff-Serv). Int-Serv attempts to reserve network resources for each individual flow on all of its traversed routers. To achieve this, it requires per-flow management at each router (including edge and core routers) in both the control plane (e.g., signaling, state management, and admission control) and the data plane (e.g., per-flow scheduling and buffer management). Such approach faces fundamental scalability limitations in large-scale networks.

On the other hand, Diff-Serv pushes intelligence to the edge routers and only requires the core routers to provide differential treatment to traffic classes based on the TOS field in the packet headers [15]. Per-flow admission control is only performed at the edge routers. To avoid per-flow signaling to the core network, recent algorithms for *endpoint admission control*, e.g., [69]-[73] introduce the concept of active probing. The endpoints probe the network to detect the level of congestion (either through packet drops or ECN marks) and admit the flow when the measured level of congestion is below a certain threshold. The additional probing traffic is an overhead that increases with the frequency of probing, which affects the accuracy of measurements and network efficiency.

5.1.2 Our Approach

In an attempt to provide better QoS assurance as offered by stateful networks like Int-Serv, while maintaining the scalability of a stateless network architecture like Diff-Serv, the CH-architecture performs admission control only at the edge domain but maintains aggregate reservations for intra- and inter-domain high-priority traffic. Our proposed resource control mechanisms leverage a *network wide* understanding of the distribution of traffic load. As pointed out in [102], an accurate view of the traffic demands is crucial to guide the operation and management of an ISP network, e.g., effective traffic engineering, debugging performance problems, and planning the roll-out of new capacity. In Section 5.2 and 5.3, we explore how the knowledge of traffic demand distributions in an ISP network can be used for aggregate reservations and admission control.

Traffic Matrix Estimation

By defining a traffic demand as a volume of load originating from an ingress ER and destined to an egress ER, we can capture the *ISP-wide* distributions of traffic load as a *traffic matrix*, TM. If there are K edge routers (ERs) within the ISP, the traffic matrix will be $K \times K$, where each entry $K(ij)$ represents the total traffic load between a particular pair of ingress ER-i and egress ER-j.

As described in Chapter 4.2, a first-tier ISP in the United States typically has 15-25 POPs, and each POP has 10-20 ERs. It is possible to have as many as 25×20 ERs within the same ISP. In this case, K can be as large as 500. The ISP-wide traffic distributions can also be abstracted and represented at two levels of granularity:

- **POP-level Traffic Matrix,** M_{pop}: Each entry of $M_{\text{pop}}(i, j)$ represents the total arrivals (or envelope) of aggregate flows at POP-j that originate from POP-i.
- **Node-level Traffic Matrix,** M_{node}: Each entry of $M_{\text{node}}(s, d)$ represents the total arrivals (or envelope) of aggregate flows at egress ER-d that originate from ingress ER-s.

Figure 5.2 illustrates how a hierarchical CH estimates the traffic matrix for a large ISP that has multiple POPs. The ingress ERs perform passive monitoring of incoming traffic and estimate the distributions of the load to various egress ERs. Through regular updates, each ER-i within a local POP provides the LCH a partial view (a row) of the traffic matrix M_{node}, i.e., $M_{\text{node}}(i, j)$ for all egress ER-j within the same ISP. Based on these partial views, the LCH constructs the traffic matrix of M_{node} for admission control purposes. The LCH for a particular POP-a is also responsible for reporting the partial view of M_{pop} to the PCH. It estimates the traffic distribution from itself (POP-a) to other POPs within the same ISP by aggregating traffic measurements from ERs:

$$M_{\text{pop}}(a, d) = \sum_{i \in \mathcal{A}_a,\, j \in \mathcal{A}_d} M_{\text{node}}(i, j) \tag{5.1}$$

where $\mathcal{A}_k$ is the set of ERs within POP-k.

5.1.3 Main Contributions

There are three main contributions of this chapter:

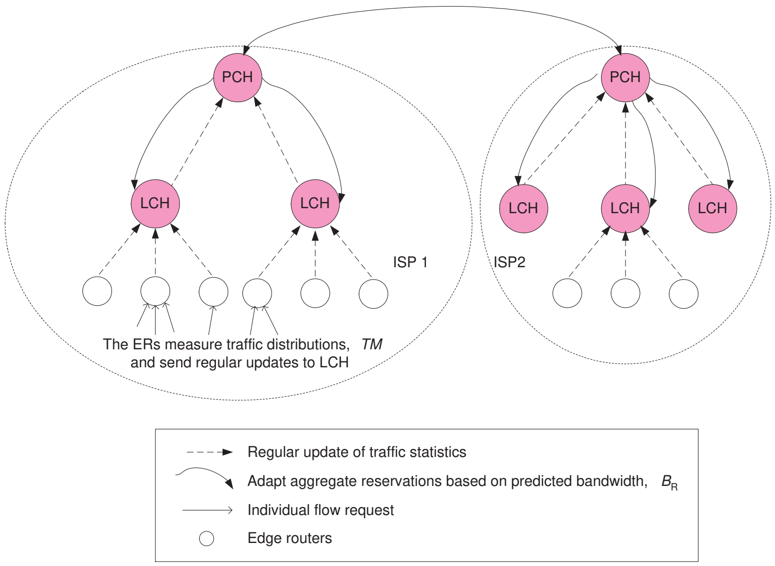


Figure 5.2: Hierarchical Clearing House architecture within large ISPs: the LCH performs resource management and admission control within a local POP, while the PCH maintains inter-POP and inter-ISP reservations.

- an aggregate reservation scheme that dynamically adapts bandwidth allocation to groups of flows based on real-time traffic measurements,
- an admission control algorithm (TMAC) that considers network-wide traffic distributions in estimating the impact of new flows, and
- an aggregate scheduling algorithm that enhances CH performance by classifying flow requests into different queues and processing them in batches.

We will now discuss the design rationale behind each of our solutions.

Aggregate Reservations

We use the online measurement of aggregate traffic statistics to *resize* the allocation of link capacity to high-priority traffic class. In Chapter 4, we defined a new service model, called IE-Pipe(s, d), that provides performance assurance for aggregate traffic (or trunks) between any specific pair of ingress ER-s and egress ER-d. The capacity requirement of an IE-Pipe is easier to characterize because the statistical variability in the individual flows is smoothed by aggregation into the same IE-Pipe. Section 5.2 describes how we use a traffic predictor to estimate the required capacity based on Gaussian approximation of the aggregate traffic arrival. Since only aggregate reservations are established for particular IE-Pipes and not for individual flows, our approach only requires maintenance of aggregate state information in all the routers.

The two performance indexes that we use to evaluate our aggregate reservation scheme are:

- Degradation of performance due to prediction error, e.g., packet loss rate, and

- Network inefficiency due to over-allocation of resources, e.g., percentage of idle bandwidth.

Note that an IE-Pipe is an example of intra-domain trunks. Similar technique can be applied to set up aggregate reservations for inter-domain trunks between two routers that belong to different domains. For the rest of our discussion, we use the more general term "trunk" to refer to the aggregate traffic between two specific endpoints that require resource reservation.

Traffic-Matrix Based Admission Control (TMAC)

Within the CH architecture, the Local Clearing Houses (LCHs) are responsible for enforcing admission control at all the ingress ERs in its domain. As discussed in Chapter 2.3, there are two classes of approach: parameter-based and measurement-based admission control. The former employs user-specified traffic parameters to estimate aggregate resource demands after accounting for statistical multiplexing gain. On the other hand, measurement-based admission control (MBAC) [62]-[67] uses online measurements of the aggregate flow instead of modeling individual flow characteristics. We choose MBAC over parameter-based approach for three reasons: (1) MBAC yields higher network utilization, (2) it is difficult to describe the Internet traffic with such diversity with a reasonably small set of parameters, and (3) by using aggregate traffic properties and distributed control, the MBAC approach has better scalability properties.

However, most existing MBAC algorithms have largely focused on provisioning resources at a single network node and do not take into account network-wide traffic load distributions and congestion levels. When the admission control decision is solely based on measurements of one single node, it may contribute to network congestion on other parts of the network, resulting in packet losses or long delays. For example, flows that are admitted at a single edge router not only affect the utilization level on the immediate outgoing link, but also on the subsequent links. Extending a single-node MBAC technique to multi-node environments would require coordination of state among nodes and involve tradeoffs between performance and signaling overhead. In Section 5.3, we propose a new admission control algorithm, TMAC, that takes these issues into account.

As mentioned earlier in Section 5.1.2, a network-wide traffic matrix can be estimated by combining on-line measurements from various ERs. TMAC leverages the knowledge of this traffic matrix to compute the allocation of link capacity to different IE-Pipes[1], denoted as the upper-bound matrix U. To achieve a "fair" sharing of resources, the available bandwidth on each link is split among the IE-Pipes in proportion to their estimated demands. The off-diagonal entries of the U matrix are used as admission thresholds at the ingress ERs. The U matrix is updated at regular intervals T_{u}, to reflect the dynamic fluctuations of traffic demands. In Section 5.4, we evaluate through simulations how the different values of T_{u} affect the performance of TMAC.

Our approach allows the admission control process to consider network-wide traffic distribution and link capacity constraints without sending per-flow signaling to all routers. TMAC also does not require active end-to-end probing of the network and therefore reduces

[1] As a reminder, an IE-Pipe can be viewed as a "virtual channel" between a specific pair of ingress and egress ERs.

bandwidth consumption by control traffic. The ultimate goal of an admission control algorithm is to admit as many flows as possible (for high network utilization) while satisfying the QoS requirements of the admitted flows. For evaluation purposes, we consider VoIP as a typical workload since the impact of packet loss rate on the perceived quality is well-understood (Chapter 3.2.2). However, our general methodology for aggregate reservation can also apply to other latency-sensitive workloads, e.g., video and interactive games. We measure the loss rate and network utilization achieved by TMAC in each of our simulations. Since TMAC lacks a priori knowledge of individual flow characteristics, we evaluate its robustness against workload diversity by considering a variety of source models. Results are presented in Section 5.4.

RxW Aggregate Scheduling

Processing and forwarding the reservation or admission control requests constitute part of the overhead in our CH architecture. In Section 5.5, we explore how an aggregate scheduling algorithm, called RxW, can be deployed to enhance CH performance by classifying individual requests into queues and scheduling them as an aggregate request. If we were to use a FIFO scheduler with a single queue, the queue length grows when the load is high. As a result, the average delay experienced by a particular request grows with the traffic load. On the other hand, RxW attempts to batch as many individual requests as possible and schedule them as a single request, while ensuring that the maximum delay experienced by an arbitrary request is bounded. To assess the effectiveness of this technique, we simulate a two-level CH architecture deployed within a typical ISP backbone topology. We study the trade-offs between improvement in system throughput and other performance indexes, such as the reduction in setup time and number of call rejections.

The following sections (Section 5.2-5.5) describe each of the above mechanisms in more details, and present the corresponding evaluation results.

5.2 Aggregate Reservations based on Gaussian Modeling

As mentioned earlier, reservations are set up for trunks (aggregate traffic) rather than individual flows in the CH architecture. To preserve the end-to-end QoS performance, the trunk reservations must reflect the fluctuations of traffic volumes produced by end-users. This section describes a predictive reservation technique whereby the intra- and inter-domain trunk reservations are adjusted dynamically at regular time intervals based on real-time measurements of traffic statistics.

5.2.1 Gaussian Traffic Predictor

Our approach is based on one fundamental concept: predictability of aggregate traffic. The capacity requirement of a trunk is easier to predict because the statistical variability in the individual flows are smoothed by the process of aggregation. We model the envelope of the arrival process of a traffic trunk as Gaussian distribution, which can be characterized by two simple parameters: its mean, μ, and variance, σ^2. The amount of reservations required to satisfy a particular QoS constraint, denoted as B_{R}, is then predicted as a function of μ and σ. We refer to this predictor as a Gaussian traffic predictor. The Gaussian modeling of the trunk behavior is justified as the following:

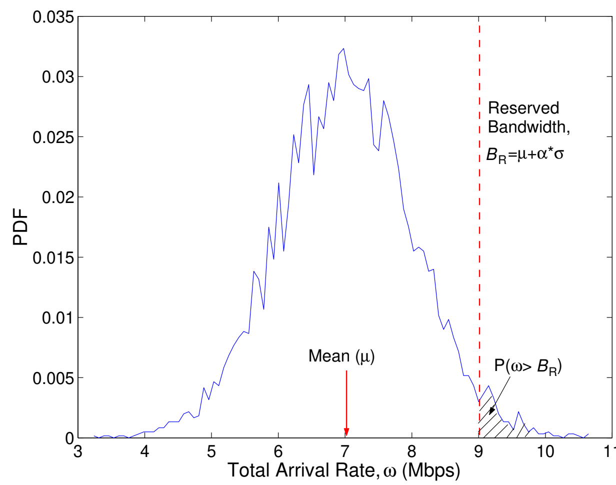


Figure 5.3: An example: Gaussian distribution and Q-function.

Gaussian Approximation When the number of individual flows gets large, the aggregate arrival rate ω tends to have a Gaussian distribution under the *Central Limit Theorem* [110]. In a bufferless fluid model, losses occur when the total arrival rate ω exceeds the reserved bandwidth B_{R}. In our analysis, we consider sufficient buffer space to hold the largest packet, and approximate the probability of packet loss as $p_{\mathrm{loss}} \approx P(\omega > B_{\mathrm{R}})$. Assume ω has a Gaussian distribution with a mean of μ and a standard deviation of σ, as shown in Figure 5.3. We estimate the required bandwidth as $B_{\mathrm{R}} = \mu + \alpha\sigma$, where α is a scalar factor that controls the extent to which the bandwidth predictor accommodates variability in the samples. If we reserve B_{R}, p_{loss} is approximately the integral of the shaded area in Figure 5.3:

$$p_{\mathrm{loss}} \approx P(\omega > \mu + \alpha\sigma) = Q(\alpha) \tag{5.2}$$

where $Q()$ is the complementary cumulative distribution (also known as tail distribution) of a standard Gaussian distribution. To ensure that QoS objectives of real-time applications such as VoIP are satisfied, α should be chosen carefully so that the loss rate is at most 1% (Chapter 3). If the Gaussian approximation holds, then $\alpha = Q^{-1}(p_{\mathrm{loss}})$.

Internet data traffic exhibits burstiness at multiple time-scales. Therefore, a predictor (B_{R}) based on a given sampling window can underestimate the bandwidth requirement that varies at shorter time-scales, resulting in possible violations of QoS guarantees. One option to cope with the changing per-flow requirements is to signal each change in flow activities at ER through the LCH to the core networks. However, this requires core networks to maintain per-flow state, leading to the same scaling problem faced by the Int-Serv architecture. In our design, the ERs measure mean, μ, and variance, σ^2, of the aggregate flows based on rates sampled during a specific measurement window, T_{mes}. At the end of each interval, the ERs send regular updates to the LCH, which uses these statistics to predict future bandwidth usage along a specific link.

If the predicted bandwidth, B_{R}, overestimates the actual bandwidth required, it will result in inefficient resource utilization. Over provisioning an inter-domain link for

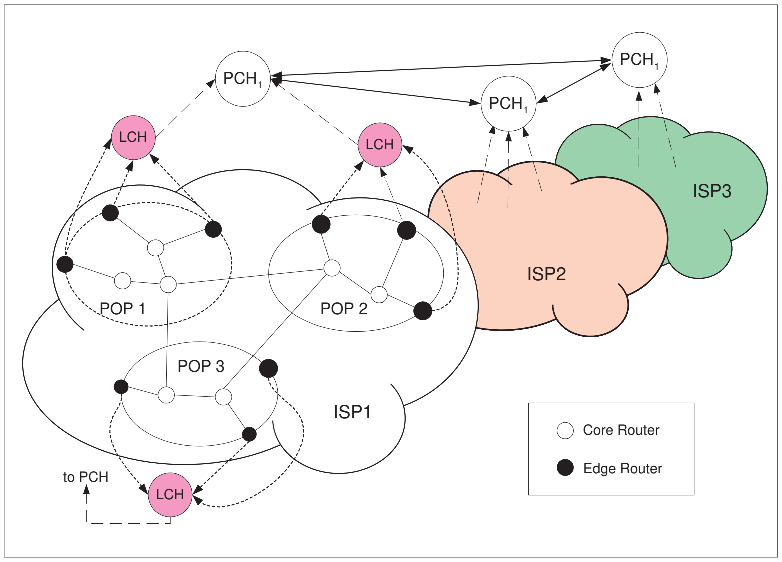


Figure 5.4: ISP1 is an example of large ISPs that has multiple POPs and an associated local CH architecture. The PCH-nodes of various ISPs maintain peering relationships to coordinate inter-domain resource allocations.

aggregate traffic originating from a particular source domain may result in unnecessary call rejections for traffic flows coming from other domains. The performance of of B_{R} heavily depends on the choice of T_{mea}, and the time-scale at which the traffic demand varies. We explore these tradeoffs in our simulation study (Section 5.2.3).

5.2.2 Deploying Gaussian Predictors

Now that we have described the basic principle behind Gaussian predictors, we will discuss how they can be deployed within an ISP for managing resource reservations.

Measuring Traffic Statistics

Figure 5.4 shows a typical scenario where a first-tier ISP network spans multiple basic domains or POPs. We assume each edge router (ER) or a third party prober can easily monitor the incoming and outgoing traffic on both the intra-domain links, and the links connecting to other neighboring domains. The Parent Clearing House (PCH) in each ISP can retrieve aggregate traffic statistics (e.g., mean and variance of existing load) and link status (e.g., amount of bandwidth reserved) by querying ERs or the LCH-nodes within its own network. This is not an unreasonable assumption: available are real-time reports on Internet traffic statistics and the current performance of major ISPs. For example, UUNET, Cable & Wireless, and AT&T advertise the backbone latency and loss measurements of their network on their company web sites. Network performance measurements can also be obtained from third-party monitoring architecture, such as KeyNote [111]. Keynote places probers in various strategic locations across the wide-area network to measure network performance, e.g., end-to-end delay and packet loss rate between two probers. A listing of other web sites that publish Internet traffic and demographic statistics can be found at `http://www.geog.ucl.ac.uk/casa/martin/statistics.html`.

All these observations imply that the ISPs are well-equipped with monitoring tools to measure the mean and variance of aggregate traffic, the essential input parameters required for deploying our aggregate reservation scheme.

Coordinated Aggregate Reservations

In the example scenario shown in Figure 5.4, the LCH within a local POP uses the Gaussian predictor to compute the required bandwidth B_{R} for aggregate traffic between every pair of ERs within its own basic domain. We assume that the LCH has complete knowledge of the network topology. The LCH can then computes the necessary resource allocation on all the links that form the path between two ERs and signals the required changes to the corresponding routers. Our approach requires two modifications at the ERs:

- A passive monitoring tool at the ERs to measure the aggregate traffic load that arrives and the distribution of the load to various ERs within the same domain. In other words, an ER i keeps track of a row of the $M_{\mathrm{node}}(ij)$, for all ER j within the same POP.
- The ERs should be able to interpret signaling from LCH and adjust resource allocation to the high-priority class accordingly. We assume ERs can support basic data-level differentiation through scheduling algorithms, e.g., rate-limited priority scheduler with two classes.

Similarly, the LCH keeps track of the mean and variance of aggregate traffic that flow into other POPs (a row of the M_{pop}) and forwards this information to the PCH. The PCH uses a Gaussian predictor to estimate bandwidth usage between different children sub-domains, and establishes trunk reservations between them. This process is repeated at different levels of the CH-tree, where predictive reservations are established on all the intra- and inter-domain links based on the predicted capacity requirements. At the top level, the PCH tracks the envelop of the aggregate traffic of the high-priority class exchanged between two ISPs. The PCHs can then coordinate among themselves to adapt the resource allocation on the inter-domain links and update the associated SLAs or peering agreements.

Note that the ERs monitor statistics of the aggregate flows rather than specific properties of the individual flows. The regular updates between different CH-nodes reflect aggregate traffic fluctuations and constitute part of the signaling overhead. As shown in Figure 5.2, the generation of these updates are decoupled from any (a) individual flow requests or (b) arrival of new updates from other ERs or CH-nodes. This ensures that individual flow arrivals and delayed copies of measurement reports do not trigger unnecessary signaling throughout the network. Our approach also does not require coordination of the reservation states among routers. The admission control algorithm for individual flow requests is discussed in Section 5.3.

5.2.3 Simulation Study

The first set of experiments explore the robustness of the Gaussian predictor with respect to traffic variability and measurement window, T_{mea}. We evaluate the bandwidth predictor using both simulated traffic and real voice traces.

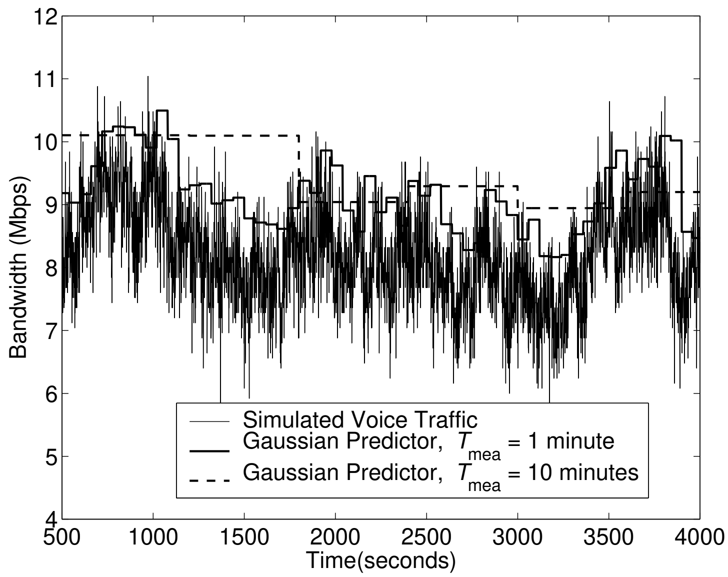


Figure 5.5: Gaussian predictors for simulated voice traffic with $T_{\text{mea}} = 1$ and 10 minutes. $\rho = 180$.

In the first case, we simulate individual voice sources based on the on-off Markov model (5.5.4) with a traffic load of $\rho = 180$ calls for a particular domain. We use a moving window of $\{1, 2, ..., 9, 10\}$ minutes for measurement and traffic predictions. Figure 5.5 shows a sample path of the aggregate traffic, along with the predicted bandwidth usage, B_{R} for $T_{\text{mea}} = 1$ and 10 minutes. Note that the predictor with $T_{\text{mea}} = 1$ minute tracks the actual capacity requirement better than the 10-minute predictor. If we allocate bandwidth based on the maximum rate (80 Kbps), we need a total bandwidth of $N \times 80$ Kbps, where N is the number of flows. We define multiplexing gain as: $(N \times 80)/B_{\text{R}}$. Advanced reservations based on 1-minute predictor achieves a multiplexing gain that ranges from 1.37 to 4.13 with mean of 1.62 and standard deviation of 0.318 when traffic load $\rho = 180$.

We repeat the experiments using packet voice traces collected from actual conversations, audio conferencing sessions, and electronic classroom applications (Chapter 3.2.3). The individual voice traces are aggregated according to a Poisson arrival model, and the sample path is plotted in Figure 5.6. We repeated the simulation for $T_{\text{mea}} = 1, 2, ..., 10$. For ease of following the graph, we only plotted the bandwidth allocation for two cases: $T_{\text{mea}} = 1$ and 10 minutes. The 1-minute predictor still tracks the actual bandwidth usage closely, but the 10-minute predictor fails to keep up with the smaller time-scale fluctuation. As a result, there are time intervals during which too much bandwidth is reserved, leading to inefficient resource utilization (e.g., between 500 and 1200 seconds in Figure 5.6). During some other intervals (e.g., between 1200 and 2400 seconds), the predictor underestimates the bandwidth requirement, leading to packet losses and failure to satisfy the QoS performance requirement.

In both cases, the probability of under-provisioning is much less than 1% for all ten values of T_{mea} taken into consideration. The observed p_{loss} is between 0.12-0.15% for the simulated traffic and is at most 0.082% when actual voice traces are used.

We measure the effectiveness of the Gaussian traffic predictor in terms of the under-

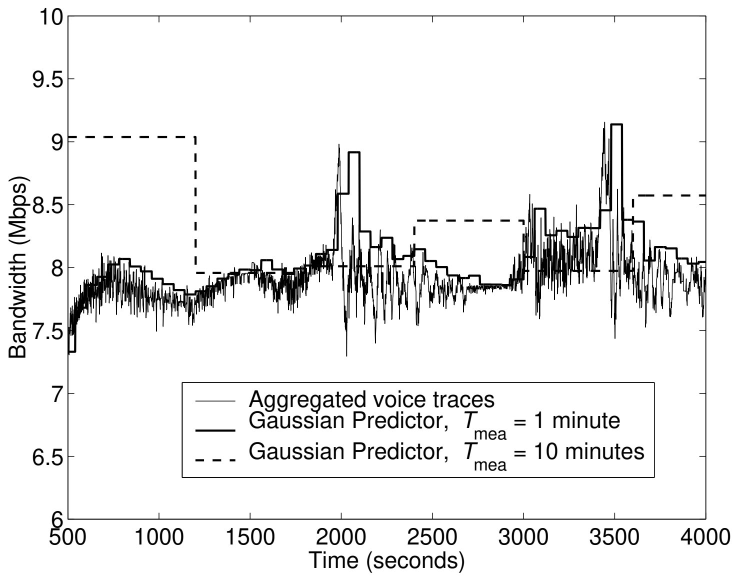


Figure 5.6: Gaussian predictors for actual voice traces with $T_{\text{mea}} = 1$ and 10 minutes.

utilized resources due to over-estimation of bandwidth requirement. The over-provisioning factor, f_{over}, is defined as:

$$f_{\text{over}} = \frac{B_{\text{R}}(l) - \sum_{(i,j)\in\mathcal{S}_l} \omega_{ij}}{\sum_{(i,j)\in\mathcal{S}_l} \omega_{ij}} \tag{5.3}$$

where $\mathcal{S}_l$ is the set of source-destination pairs that have traffic routed through link l. $B_{\text{R}}(l)$ is the amount of reserved bandwidth based on the Gaussian predictor; ω_{ij} is the actual arrival rate of traffic between source domain (or router) i and destination domain (or router) j.

We ran 100 simulations, each for 1 hour, using both simulated traffic and actual voice traces to evaluate the efficiency of the Gaussian predictor. The average f_{over} as a percentage (%) is plotted in Figure 5.7 for the two cases. Observe that the amount of over-allocation increases with T_{mea}, as the predictor becomes less responsive to the traffic fluctuations. For simulated traffic, the f_{over} increases from 12.3% to 15.5% when T_{mea} increases from 1 to 10 minutes. In the case of actual voice traces (dash line in Figure 5.7), f_{over} is more sensitive to the value of T_{mea}: f_{over} increases from 8% at $T_{\text{mea}} = 1$ minute to 32.5% at $T_{\text{mea}} = 10$ minutes. This implies that the actual voice traffic is more bursty than the traditional on-off Markov model for voice. We need smaller T_{mea}, i.e., 1 or 2 minutes, to track the aggregate traffic fluctuation.

In general, the required over-provisioning is higher for the case of actual voice traces than the simulated traffic because these traces were collected from a variety of multimedia applications (including multi-party audio conferencing and broadcasted lectures), where the individual source characteristics have higher statistical variability. This is not captured by the on-off Markov model that we used to simulate VoIP traffic for two-way conversations.

5.2.4 Discussions

Our simulation results show that it is possible to satisfy the QoS requirement of real-time voice traffic by ensuring that the packet loss rate due to insufficient resource

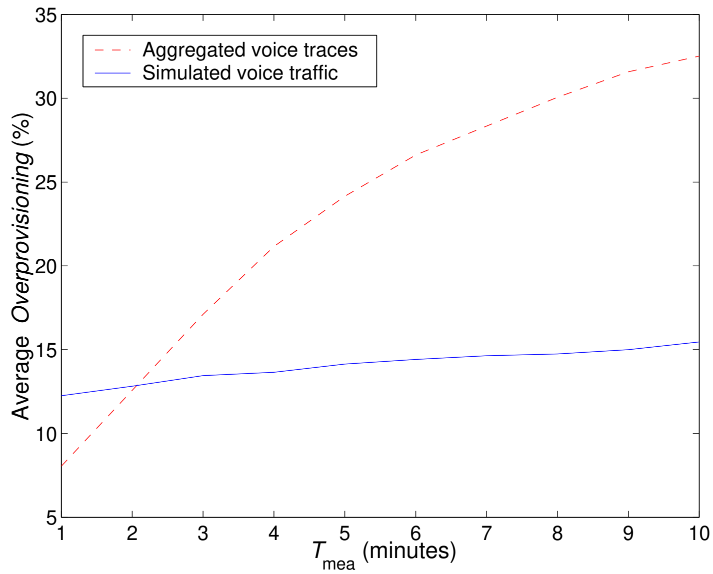


Figure 5.7: Average f_{over} (in %) when reservations are made based on Gaussian predictors for (a) aggregated voice traces, and (b) simulated voice traffic at ρ=180.

reservations is always below 1%. This can be achieved through a reasonable amount of over-provisioning to account for the prediction error. The prediction error is insignificant if the measurement window, T_{mea}, is small enough to track the statistical variability of the aggregate traffic loads. However, smaller T_{mea} means more control traffic is generated to send more frequent traffic measurement reports to the CH-nodes, leading to greater signaling overhead. On the other hand, the amount of over-provisioning required when larger T_{mea} values are used. Therefore T_{mea} controls a set of trade-offs among QoS performance, over-provisioning, and signaling overhead.

The magnitude of prediction error is dependent on the following two factors:

- **Level of aggregation:** Gaussian approximation holds when the number of flows aggregated, n, is large (> 50) [110]. Since the modeling accuracy increases with n, Gaussian traffic predictor should work well for describing the Internet workload in the backbone, since the value of n involved is large (over thousands of flows). However for low bandwidth links that observe less than 50 flows, our technique will not be as effective.

- **Source characteristics:** In general, the prediction error of our Gaussian approximation increases with the statistical variability of the individual sources within the traffic aggregate. As a result, one will need a larger amount of over-provisioning to achieve the same target loss rate. If the individual source has statistical properties that vary at multiple time-scales or is self-similar [90, 91], a more complicated model, e.g., fractal model [112] will be needed to describe the the aggregate traffic.

Although our evaluation is based on VoIP traffic, our general methodology for aggregate reservation can apply to other latency sensitive applications, such as real-time video. However, to improve the prediction accuracy, the knowledge of workload is important for choosing the best model to describe the arrival process of the aggregate traffic. For example, compressed video is known to generate traffic with variable characteristics at multiple time-scales. In [113], the authors characterize the video traffic at three explicit time scales: frame level, scene level and epoch level. We can replace the Gaussian predictor with this model to describe the aggregate behavior of video traffic more accurately, which will then reduce the amount of over-provisioning required.

Using VoIP as an example workload, we discuss how the per-flow admission control can be performed at the ingress points of a network domain in the next section.

5.3 Traffic Matrix Based Admission Control (TMAC)

Since the admission-controlled traffic must coexist with current best-effort traffic, we follow the same set of principles outlined in [73]. First, we need to strictly limit bandwidth allocated to the admission-controlled traffic so that it never borrows bandwidth from the best-effort class. Secondly, best-effort traffic must not pre-empt admission-controlled traffic, and hence the latter should be served in a higher priority class. In our model, the maximum capacity allocated to a high-priority traffic class on a particular link is bounded by a small fraction of its total capacity. This target fraction is denoted as β_{T}, which is typically 30%. This ensures that the edge-to-edge queuing delay within a domain is statistically bounded.

Our goal is to develop an admission control algorithm that can simultaneously achieve:

- a strong (multi-class) service model, e.g., packet loss for each IE-Pipe is bounded below 1%,
- high network utilization through flow and class level statistical sharing,
- scalability, e.g., no per-flow signaling or state management in the core routers, and
- fairness, e.g., distribute network-wide network resources in a fair manner among different IE-Pipes.

For ease of discussion, we define the following terms:

- **Demand Matrix**, D, captures the distribution of traffic demand within an ISP. Given an ISP network with K access routers, D is a $K \times K$ matrix, where each entry $D(s,d)$ represents the total bandwidth requested by high priority flows from ingress ER-s to egress ER-d. D is estimated based on the bandwidth requirement specified in each admission control request and accounts for both admitted and rejected flows.
- **Upper-bound Matrix**, U, is computed by splitting the bandwidth of the bottleneck links from ER-s to ER-d between all the competing IE-Pipes(s,d) in proportion to their estimated traffic demands, $D(s,d)$ (Section 5.3.1). $U(s,d)$ is used as a threshold for admission control.

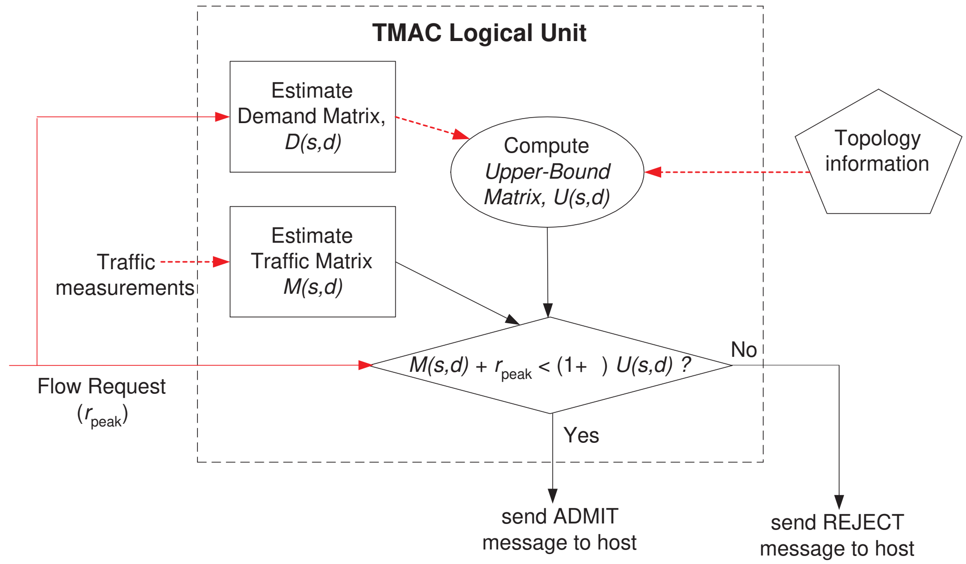


Figure 5.8: A logical view of the Traffic Matrix Based Admission Control (TMAC) unit.

5.3.1 Single Domain Case with One CH-node

First, we discuss how TMAC works in a single domain case where a single CH-node is responsible for resource management. For example, consider a small ISP as one logical domain without dividing it into smaller sub-domains or POPs. The CH-node serves as both the LCH and the PCH in this case.

Figure 5.8 shows the logical view of a TMAC unit. We assume that an admission controlled flow sends an explicit REQUEST message to the ingress ER in which it specifies the peak rate, r_{peak}, required to satisfy its QoS objective. When a new flow arrives at ER-s, the ER-s forwards the REQUEST message to the TMAC unit within the CH-node. The input link and the destination IP address of the REQUEST message are used to identify the end-points of demands, i.e., ingress ER-s and egress ER-d. This process requires information about destination prefixes associated with each egress link. We follow the same methodology outlined in [102], and keep a table of destination prefixes and the corresponding egress routers in the CH-node. Each `dest-prefix` consists of an aggregated network address advertised by the egress router and a mask length. The `dest-prefix` allows a set of destination IP addresses to be mapped to a specific egress router d. In practice, these `dest-prefix`'s can be determined from the forwarding tables of routers that terminate egress links. We assume the CH-node can obtain this information from the underlying routing protocol.

Upon receiving a new REQUEST message for a flow with r_{peak} between ingress ER-s and egress ER-d, the TMAC checks the following condition:

$$M_{\text{node}}(s,d) + r_{\text{peak}} < (1+\sigma)\cdot U(s,d) \tag{5.4}$$

where $M_{\text{node}}(s,d)$ is the estimated rate of admitted traffic between ER-s and ER-d, $U(s,d)$ is the admission threshold, and $0 \le \sigma \le 1$ is the hysteresis parameter. The new flow is admitted if the condition in (5.4) is not violated. Otherwise the flow is rejected. Setting

$\sigma > 0$ exploits statistical multiplexing of multiple flows and increases utilization level, but σ must be small enough to avoid oversubscription of resources and its corresponding adverse impact on the end-to-end performance. We investigate how the value of σ affect the TMAC performance in Section 5.4. We describe how D, M_{node} and U matrices are estimated in the next two subsections.

Estimating Demand Matrix and Usage Matrix

The traffic demand, $D(s,d)$ is estimated as simply the sum of the peak rate requested by individual flows (including both admitted and rejected requests):

$$D(s,d) = \sum_{\{f\}} r_{\text{peak}}(f) \tag{5.5}$$

where $\{f\}$ is the set of flows that enter the network at ingress ER-s and exits at egress ER-d.

We use a time-window estimator as described in [62] at each ingress ER-s to estimate the usage vector $M_{\text{node}}(s,:)$, where each entry $M_{\text{node}}(s,d)$ represents the rate of admitted traffic to each egress ER-d. The measurement window Δ_{T} is a multiple of Δ_{S}, and at the end of every Δ_{T}, $M_{\text{node}}(s,d)$ is set to the highest average load computed for any Δ_{S} in the previous window. All ingress ER-s send regular updates of their $M_{\text{node}}(s,:)$ vectors to the CH-node, which maintains the complete matrix M_{node}.

Computing Upper-bound Traffic Matrix

We assume that shortest paths for all ingress-egress pairs are known. Let $\mathcal{P}_{ij}$ be the set of links that form the shortest path from ingress router i to egress router j:

$$\mathcal{P}_{ij} = \{l_1, l_2, \ldots, l_h\}$$

where h is the number of hops from i to j.

For each link with capacity C_l, we limit the share of bandwidth allocated to high priority traffic as $\beta_{\text{T}} \cdot C_l$ where $0 < \beta_{\text{T}} < 1$. This available bandwidth should be split among different IE-Pipes that share the same link in proportion to the corresponding $D(s,d)$. For a particular IE-Pipe(s,d), the allocated bandwidth on link l is:

$$U_l(s,d) = \frac{D_l(s,d)}{\sum_{i,j \text{ s.t. } l \in \mathcal{P}_{ij},\, i \neq j} D_l(i,j)} \cdot \beta_{\text{T}} \cdot C_l \tag{5.6}$$

The admission threshold for IE-Pipe(s,d) can be determined as:

$$U(s,d) = \min_{l \in \mathcal{P}_{sd}} U_l(s,d), \quad \forall s,d, s \neq d. \tag{5.7}$$

We compute the entries, $U(s,d)$, from Eqs. (5.6) and (5.7) when $s \neq d$ and set $U(s,d) = 0$ when $s = d$. Since traffic demands for operational IP networks exhibit different time-of-day patterns (as reported in [102]), we need to update U often enough to reflect the dynamic fluctuations. The update interval is denoted as t_{u}, which is a parameter that we vary in our experiments.

Robustness to non-stationarity in the demand can be achieved by choosing the update interval, t_{u}, to be smaller than the time-scale at which $D(s,d)$ fluctuates. The sensitivity analysis of this parameter is discussed in Section 5.4.

5.3.2 Multiple Domain Case with Hierarchical TMAC

As the size of a domain grows, the number of admission control requests that need to be processed by the CH-nodes increases with the number of incoming flows. The order of the traffic matrix M_{node} also grows with the number of ERs. The direct application of TMAC to this scenario would eventually cause inefficiency and scalability problems. In Chapter 4, we solved the scalability problem of large-scale network by introducing a hierarchical CH architecture and distribute resource management talks to various level of CH-nodes. Similarly, we propose a hierarchical-TMAC (H-TMAC) that takes advantage of locality information and performs admission control in multiple *stages*.

A large ISP network can be divided into logical basic domains (BDs) with either geographic or administrative boundaries. For example, a basic domain can represent a local POP. Figure 5.2 shows an example of the two-level CH-architecture within two large ISP, one with two POPs, and the second one with three POPs. The LCH node within each POP maintain a sub-matrix of M_{node} that includes all ERs within its own domain. Therefore, the LCH can compute its own $U(s, d)$ for all its own ERs s, d. It is responsible for performing admission control for intra-domain flows, i.e., flows that arrive and exit from the same local domain. For inter-domain flows, the LCH will forward the requests to the PCH and the LCH of the destination BD for admission control.

To avoid having to forward every single flow request to the PCH, the LCH aggregates individual requests that share the same path, i.e., between the same pair of source and destination POPs, into an aggregated request for a higher bandwidth (sum of all r_{peak} requested) to the PCH. The aggregate request can be attached with a list of reservation identifiers for each individual sub-request, e.g., to differentiate bandwidth requirements from the source domain to each individual ER within the destination POP. The admission control is done in two stages:

1. The PCH forwards the aggregate request to the LCH in the destination POP which uses the attached list for finer-grained admission control via a local TMAC mechanism.

2. If the destination LCH accepts the aggregate request, the PCH checks if there is enough resources between inter-POP links to satisfy the request. Instead of Equation 5.4, it checks if
$$M_{\text{pop}}(a, b) + R(a, b) < \beta_{\text{T}} C(a, b) \tag{5.8}$$
where $M_{\text{pop}}(a, b)$ is the measured load from POP-a to b, $R(a, b)$ is the required bandwidth specified in the aggregate request, and $C(a, b)$ is the available bandwidth on the inter-domain link between POP-a and b.

The hierarchical CH architecture that deploys H-TMAC is an example of *collective tasking system*. In Section 5.5, we describe how aggregate scheduling mechanism is employed to enhance the throughput of CH-node and reduce the individual response time.

5.4 Performance Study: TMAC Characteristics

The aim of the simulation study is to evaluate the performance and robustness of TMAC. To evaluate the basic properties of TMAC, we consider a single domain case with one CH-node. We use the ns-Network Simulator [89] to implement the basic mechanisms

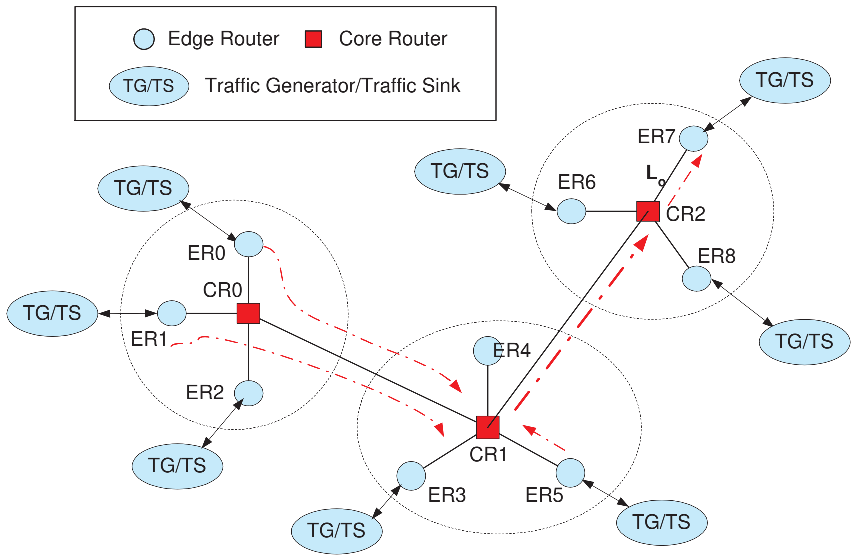


Figure 5.9: Simulation topology.

of a CH node. A time-window estimator is introduced at each input link to estimate the rate of existing flows. The admission control module is created as an NsObject and inserted before the ingress router. The admission control tasks (TMAC) and upper-bound matrix computation are implemented at the Tcl-level. Our CH-patch works for ns-v2.1b6.

5.4.1 Network Topology

Since it is infeasible to run large-scale Internet experiments over actual networks, we simulate a simple ISP topology shown in Figure 5.9 that consists of 9 edge routers (ERs) and 3 core routers (CRs) grouped into three local POPs. All the ERs can be both ingress and egress points for flows that enter and exit the ISP. The target utilization level β_{T} for high priority traffic is 0.3. Each router uses a simple priority scheduler, and the control messages are sent at the same priority as the data. In our simulations, the access links (from ERs to CRs) are 10 Mbps and the backbone links (between two CRs) are 100 Mbps each.

5.4.2 Traffic Generation

For proprietary reasons, we are unable to access the real traffic measurements from Internet backbone networks. We derive traffic models based on published results and discussions with researchers who have studied data sets collected by ISPs themselves.

In our simulations, we model the admission-controlled traffic as a Poisson arrival process. The arrival rate from a host network to an ingress router is denoted as $\lambda_i(t)$. We use the indices t to indicate the time-of-day dependence of the traffic demand as reported in [102] and [97], but do not attempt to differentiate the effect from different user groups (e.g., domestic consumer vs. domestic business, etc.). Data sets from the Sprint IP backbone monitoring project [97] indicate that the bandwidth consumption peaks between 10 a.m.

and 2 p.m. during the day and shows a dip from midnight to 3-4 a.m. These data sets also show that there is a smaller time-scale fluctuations throughout the day: about ± 10-15% changes at 30 minutes intervals.

To reflect the realistic traffic demand, we consider two scenarios:

1. Set $\lambda_i(t_o) = 1$ and inject ± 10-15% changes at T_ϕ=30 minutes interval for a two-hour simulation. This simulates the smaller time-scale behavior observed in [97].

2. Same as 1, but introduce a steep drop of 80% to $\lambda_i(t)$ after 30 minutes into the simulation. This simulates the abrupt change in bandwidth consumptions between the peak and off-peak hours.

The traffic distributions from an ingress ER-s to a set of egress ERs are based on a random probabilistic model.

We consider four different source models to generate individual flows: EXP1, EXP2, CBR and PARETO in our experiments. In Chapter 3, we discuss the rationale behind using these models to generate different workload for our experiments. Table 5.1 summarizes the characteristics of these four models and the type of applications that can be represented by each of them. EXP1, EXP2 and CBR have exponential lifetimes with an average of 300 seconds. The flow lifetimes of PARETO sources follow a log-normal distribution with average of 300 s. The aggregation of Pareto sources is known to exhibit long range dependencies [90, 91]. For all four cases, packets are 200 bytes in length. This correspond to sending 20 ms frame of digitized voice in a typical VoIP application that use PCM codec and 8KHz sampling rate.

Source Model	Traffic Characteristics	Example Applications
EXP1	Exponential on and off times with average of 1.004 and 1.587 seconds. Peak rate = 64 Kbps.	Voice-over-IP Internet audio
EXP2	Exponential on and off times with average of 0.1 and 0.9 seconds, respectively. Peak rate = 258 Kbps.	Bursty video clip Web transactions
CBR	Constant bit rate. Peak rate = 64 Kbps.	Voice-over-IP without silence suppression
PARETO	Pareto on and off times with average of 0.5 seconds each. Peak rate = 64 Kbps.	Heavy-tail web traffic

Table 5.1: Four traffic source models used to generate different workload for our ns-simulations.

5.4.3 Performance Evaluation

Using ns simulations, we evaluated how well the TMAC performs with respect to the goal of achieving high network utilization while maintaining low packet loss. All

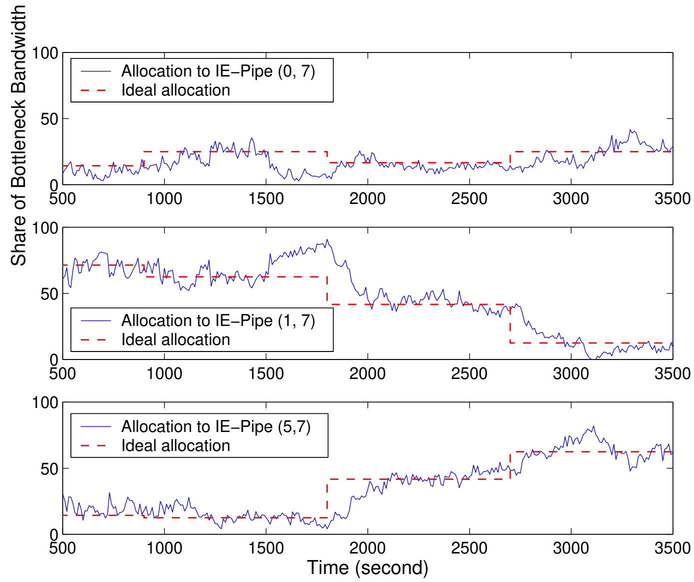


Figure 5.10: **Abrupt Traffic Fluctuations:** Share of bottleneck bandwidth allocated to each ingress-egress pair that share Link-3: $D(0,7)$, $D(1,7)$ and $D(5,7)$, $t_{\mathrm{u}} = 2$ minutes.

simulations were repeated using different seeds to the random number generator. The averages across all repetitions are reported in our results. In all our experiments, we use homogeneous flows unless specified otherwise. We assume that the peak rate specified by a flow is accurate, i.e., once the flow is admitted, its instantaneous transmission rate may fluctuate but it never exceeds the specified peak rate. The offered load is chosen such that blocking rates in these experiments are approximately 20%. The utilization level is defined as:

$$\frac{\text{Measured traffic load}}{0.3 \times \text{Link capacity}}.$$

Abrupt Traffic Fluctuations

In our first experiment, we examine how well TMAC reacts to dynamic fluctuations in traffic demand. We consider the case where all the flows that are destined for ER7 originate from either ER0, ER1 or ER5 (as marked by the dotted arrows in Figure 5.9). Figure 5.10 shows the bandwidth-sharing achieved on the bottleneck link L_{o} as a result of the TMAC policy when EXP1 source model is used. The x-axis shows simulation time and the y-axis shows the average bandwidth used by each of the three IE-Pipes: (0,7), (1,7) and (5,7) over ten-second intervals. All the EXP1 sources from ER0, ER1 and ER5 which are destined to ER7 compete for bandwidth on L_{o}. The upper-bound traffic matrix is updated at regular intervals of $t_{\mathrm{u}} = 2$ minutes in this case. We artificially introduce abrupt changes in traffic demand at time 900 s, 1800 s, and 2700 s. The dashed lines on Figure 5.10 show the ideal allocation of bandwidth among these three ingress-egress pairs if the traffic demand is known a priori. The solid lines show the actual link sharing achieved by TMAC. Results indicate that each ingress-egress pair gets roughly its ideal bandwidth allocation, with 8.1% - 19.9% deviation from the ideal values. On an average, it takes 2.7 minutes after each perturbation to converge to the ideal utilization level.

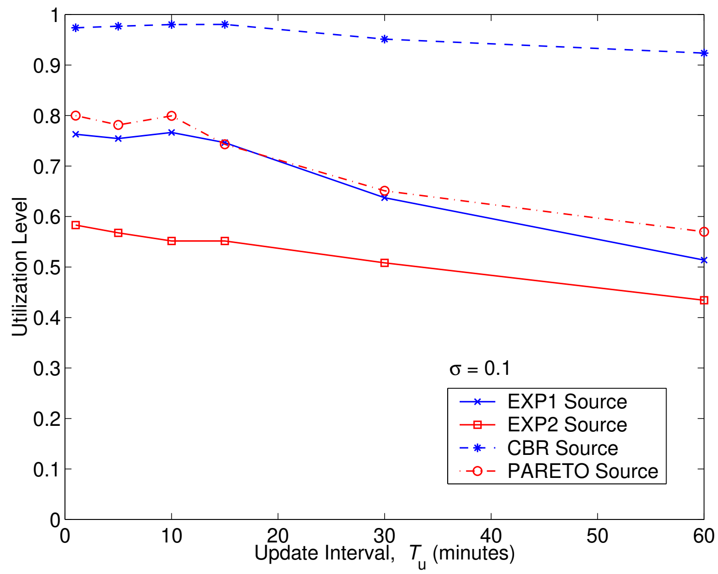


Figure 5.11: **Sensitivity analysis:** Link utilization vs. the update interval for upper-bound traffic matrix, t_u, for different source models.

Sensitivity Analysis

Since TMAC depends on the estimation of the upper-bound traffic matrix U, we need to understand how the utilization level is affected by the frequency at which U is updated. Figure 5.11 shows the network utilization level averaged over three bottleneck links as t_u is varied from 1 to 60 minutes for all four different source models (EXP1, EXP2, CBR and PARETO). We inject ± 10-15% random fluctuations in traffic demand as described in Section 5.4.2 at regular intervals of T_ϕ=30 minutes. For a given source model, each point on the curve shows the utilization level for a particular t_u averaged over 50 simulation runs with different random seeds. The reason for repeating the experiment 50 times is to get a large enough sample size to apply the central limit theorem and calculate the valid mean [114]. In all four cases, the average utilization level decreases as t_u is increased because the estimated U is less responsive to traffic fluctuations. However, the performance degradation is negligible for $1 \leq t_u \leq 10$ minutes. The utilization level starts to decrease as t_u is increased beyond 10 minutes. Since recomputation of U incurs processing overhead, it is desirable to choose a larger value of t_u. In this case, $t_u = 10$ minutes is the optimal choice since it is the largest possible t_u that does not degrade TMAC performance considerably.

The effect of t_u on TMAC performance varies depending on the statistical variability of the traffic source. With EXP1 sources, the utilization level decreases from 77% to 51%. Experimenting with CBR sources achieves the highest utilization level (92-98%) since the bandwidth usage of admitted flows is fairly deterministic. The presence of a burstier source (EXP2) reduces the utilization level (43-58%), while the long range dependent traffic (PARETO) achieves roughly the same utilization as (EXP1). In all cases except CBR source, packet loss is less than 0.3%. For CBR source, packet loss varies from 0.6% for t_u = 1 minute to 4.5% for t_u = 60 minutes.

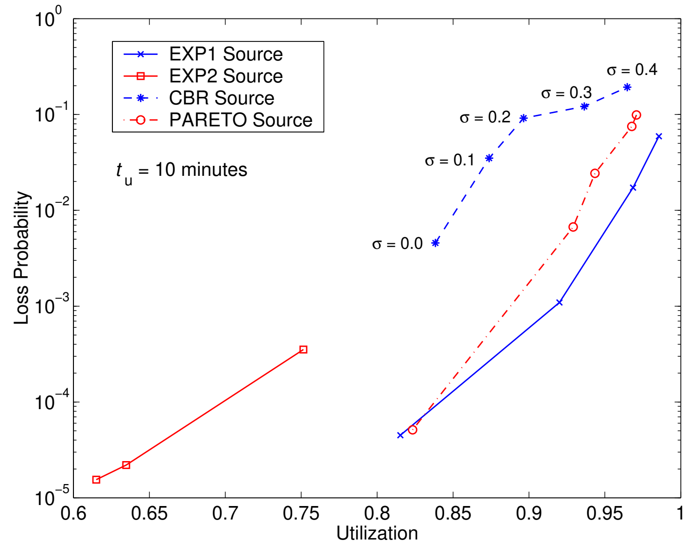


Figure 5.12: **Trade-offs:** Loss-load curves for four different source models as the control parameter σ is varied. t_u=10 minutes.

Trade-offs Between Loss and Utilization

There is a trade-off between packet loss and network utilization level in any admission control system. To increase the network utilization, one might admit more flows into the system. But this may lead to over-subscription of resources and a higher loss rate. Tuning the hysteresis parameter, σ, allows an ISP to operate at the desired utilization level. We are interested in the range of utilization and loss rates that can be achieved by varying σ for different source models. Results are plotted as a set of *loss-load* curves in Figure 5.12. Each point shows the average packet loss for a given utilization level as σ is varied from 0.0 to 0.5. The loss load curves have the same frontier as those of the measurement-based admission control algorithm (not shown here) reported in [62]. The burstier source (EXP2) shows a dramatically different range as compared to the three other traffic sources. Since the peak rate of EXP2 is 10 times larger than that of EXP1, and the admission control policy is based on peak rate, we tend to admit less number of EXP2 flows. However, since the activity cycle of EXP2 is very short (10%) and the used bandwidth is zero for a large amount of time, we observe this under-utilization of network resources.

5.4.4 Discussions

We have presented TMAC, a new admission control scheme that combines the Measured-Sum [63] algorithm (a measurement-based approach) and the traffic matrix knowledge in making admission control decisions (Section 5.3). It has been known that the MBAC approach can achieve higher network utilization than parameter-based algorithms, but can lead to occasional packet losses or delays [62, 63]. To ensure that TMAC can satisfy the application-level QoS requirement, e.g., 1% maximum loss rate for VoIP, we use the peak rate (rather than mean rate) of the new flow in the admission control process, as represented by Equation (5.4). We choose this conservative approach because TMAC does not have a priori knowledge of the workload models.

To evaluate the robustness of TMAC against the diversity of Internet workload, we use four different source models: EXP1, EXP2, CBR, and PARETO in our simulations.[2] The simulation results show that TMAC can achieve low loss rate but at the cost of network resources. For example, the number of flows is very small for EXP2 source model since TMAC uses peak rate for admission control, leading to extremely low network utilization. In this case, we want to increase the level of acceptable statistical multiplexing by tuning σ to allow admission of more flows and improve network efficiency. This implies that the knowledge of workload is useful to tune TMAC parameters for optimal performance. For example, with the EXP1 source (VoIP like traffic), TMAC can achieve 95% utilization level with less than 1% loss with $\sigma = 0.2$. In Section 5.4, we presented the tradeoff between link utilization and packet loss rate achieved by TMAC with different values of σ. This information allows ISPs to choose the appropriate operational point of their network based on their performance goals and customer flows' requirements.

There are two possible solutions to improve the performance of TMAC:

1. **Application Assisted TMAC:** In this approach, the admission controlled flows are required to specify its source characteristics in the REQUEST message. For example, it should provide average rate and burst size information besides to peak rate. The additional information can be used to choose the optimal value of σ.

2. **TMAC with Feedback Loop:** Instead of relying on the flows to provide accurate source parameters, this approach uses the measured loss rate and link utilization as feedback to fine-tune the TMAC parameter, σ. For example, σ should be decreased if loss rate is high, and increased if both link utilization and loss rate are low.

Another source of overhead for TMAC is the processing of individual request messages, especially in the Hierarchical-TMAC case when a flow crosses multiple domains. We address this issue in the next section.

5.5 Aggregate Scheduling in the CH Architecture

So far, we have presented the details of the aggregate reservation and admission control mechanisms within the Clearing House. Both mechanisms introduce the following overhead:

1. bandwidth consumed by control traffic, e.g., REQUEST message for admission control, traffic measurement reports, and signaling message to establish aggregate reservations at routers.

2. processing power of CH-nodes to service admission control (at flow level) and reservation (at trunk level) requests.

The processing of the resource control requests (item 2 mentioned above) is a potential bottleneck that might impact the performance of a CH-node. In this section, we explore how aggregate scheduling can enhance the throughput of the Clearing House in servicing the various resource control requests.

[2] EXP1 and CBR are used to model VoIP traffic with and without silent suppression, respectively. EXP2 sources have the highest statistical variability while PARETO sources are known to generate long-range dependency traffic [90, 91].

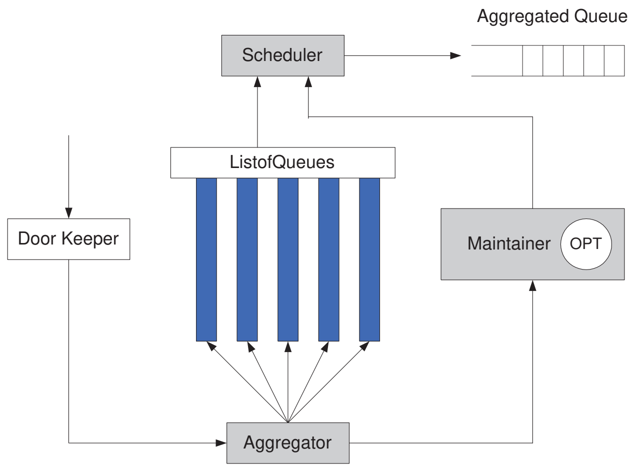


Figure 5.13: A general framework for aggregate scheduling.

5.5.1 Background: Aggregate Scheduling

Aggregate scheduling is a mechanism in which multiple requests of a particular type are aggregated into a single request for enhancing the performance of the system. The *level* of aggregation refers to the number of requests that have been aggregated. The cost of executing the aggregated request is much less than the cost incurred by executing each request separately.

Our implementation of a generic aggregate scheduling mechanism includes three distinct jobs: *aggregator*, *scheduler*, and *maintainer*. There are two data structures shared between these three jobs: ListofQueues and OPT data structures. Figure 5.13 shows a general aggregate scheduling framework.

Aggregator

The aggregator listens to incoming requests and places them into the request queue of a particular *class*. If the incoming request is of a new class, the aggregator creates a new queue for that class in the *List of Queues* data structure. The aggregator also keeps track of all the active classes and updates the status of the OPT data structure for every incoming request.

Scheduler

The scheduler schedules a *class* and not individual requests. The selection of a class is based on the optimization metric of a specific algorithm, e.g., the queue with the maximum number of requests or the longest waiting time. Once a class is scheduled, all requests of that class are aggregated into one request, and the aggregated request is processed.

Maintainer and OPT

The OPT data structure stores the status of the requests currently in the queues (for all the classes). The Maintainer is responsible for managing the OPT data structure. The OPT stores the following information for each particular class c

- $R(c)$, the total number of requests of class c.
- $W(c)$, the waiting time metric of the aggregated request, e.g., the total waiting time, the maximum or minimum waiting time, or the average waiting time.
- $A(c)$, an application specific metric for class c, e.g., the size of an object in broadcast scheduling.

Every aggregate scheduling algorithm is associated with an optimization metric, which is a function of $R(c)$, $W(c)$, and $A(c)$, and denoted as $f_c(R, W, A)$. The scheduler picks the class c with the maximum value of $f_c(R, W, A)$ to optimize its performance.

5.5.2 RxW Scheduling

In this section, we describe the RxW aggregate scheduling algorithm that we use for enhancing the CH performance.

For RxW scheduling, the optimization metric is based on $R(c)$ and $W(c)$, and are application independent. Here, $R(c)$ refers to the total number of requests, while $W(c)$ refers to the maximum waiting time of a request in the queue of class c. The scheduler schedules the class with the maximum value of $R(c) \times W(c)$. RxW scheduling is well studied in [115]. Several approximate variants of RxW scheduling are discussed in [115] but a detailed comparison of how these variants perform in our application is beyond our scope.

5.5.3 Aggregate Scheduling and the Clearing House

Every CH-architecture has control over a set of domains. We assume that all the flows between a specific pair of ERs within the same ISP domain share the same primary path. In this case, forwarding and processing each admission control call request from end-to-end is the bottleneck in servicing the call. We define a class c for every pair of ERs. All flow requests between the same pair of ERs can be aggregated into a single request with higher bandwidth requirement. The total bandwidth computation and discovering the path between a pair of ERs is done only once for the set of requests. We can also use a cache to store the different computed paths for future requests and update it when there are routing changes.

5.5.4 Simulation Framework

We have developed a simulator that simulates the actions of a two-level Clearing House architecture. The CH is treated as a database in which, the reservations along the various links in the topology are maintained. The CH-node has the following simple structure:

```
typedef struct {
  Database *database;
```

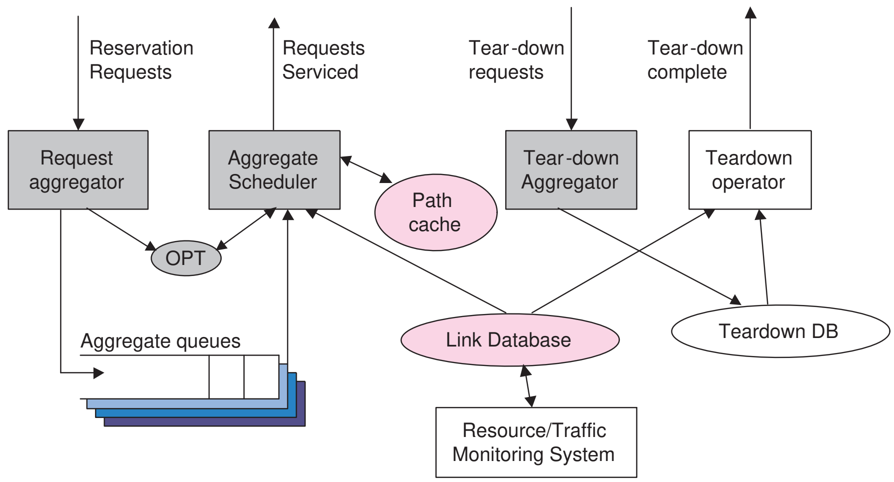


Figure 5.14: Simulation model.

```
  // Database of all links in topology
  IntexTable *itable;
  // Hash index for efficient database access
  PendingQueue *pq;
  // Pending queue of call requests
  TeardownQueue *tdq;
  // Pending queue of tear-down requests
  TeardownAggregator *tda;
  // Database of aggregated tear-downs
  AggregateRequests *arqueue;
  // Queue of aggregated requests
  Network_Stats *net;
  // Network Statistics
  Cache_Paths *cp;
  // Cache of Shortest Paths
}ClearingHouse;
```

There are four important components in our Clearing House simulator. As illustrated in Figure 5.14, they are the call setup aggregator, the call setup scheduler, the call tear-down aggregator and the call tear-down scheduler. These four processes are scheduled by the global scheduler in a weighted round-robin fashion. RxW scheduling is employed by the call setup scheduler and the cache is used for storing previously computed shortest paths between different pair of ERs. The link database stores the entire ISP network topology and propagation delay along the various links.

Network Topology

For our simulations, we use the topology shown in Figure 5.15, which is an approximation of the AT&T Worldnet IP backbone as reported in [46]. Assume an ISP network that has to interconnect different POPs that reside in 12 important cities in the USA. Each

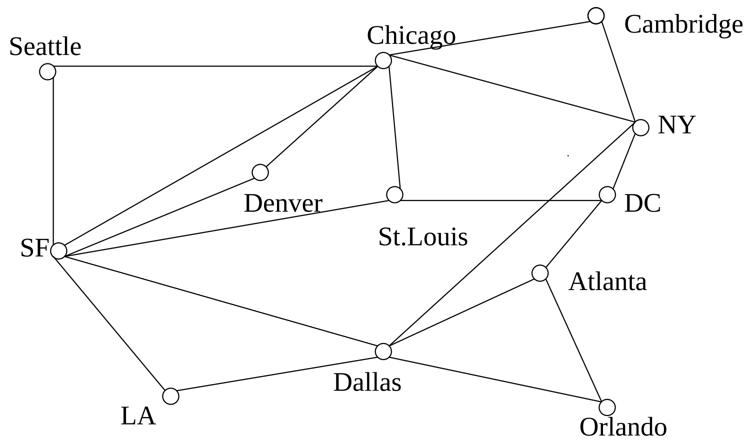


Figure 5.15: Topology of the IP backbone with 12 basic domains.

POP is represented by a basic domain (BD), and they are interconnected by links with limited capacity. A Local Clearing House (LCH) is associated with each BD. Inter-domain call requests generated based on a weighted distribution and sent to the LCHs. The 12 domains are grouped to form one larger LD and top CH-node (PCH) is introduced to service aggregate reservation requests between multiple domains.

Workload Models

We use voice traffic as a workload to drive the evaluation of the Clearing House. The call arrival rate in each domain i is modeled as an independent Poisson process of intensity λ_i calls per second, and the call duration is exponentially distributed with a mean of $1/\mu$=120 s. We define the traffic load arriving at each LCH i as $\rho_i = \frac{\lambda}{\mu}$, where $i = 1,\ 2,\ \ldots,\ 12$.

5.5.5 Performance Evaluation

In this section, we study the performance characteristics of a single PCH-node, and evaluate the scalability issues involved in deploying the CH architectures in large-scale networks.

In our simulations, a single PCH node keeps track of the reservations along the various links in the topology given in Figure 5.15. The 12 LCH-nodes perform local admission control while the PCH processes call requests between POPs and coordinate aggregate reservations on the various inter-domain links. The reservation status is maintained in the back-end database which is constantly updated.

Given this setting, we test the performance of this node under various loads. We define the load as the number of inter-domain reservation requests per second arriving at the PCH node. A weight proportional to the population of the city is associated with every node in the topology. The calling pattern is derived from a probabilistic model in which the probability associated with every node is proportional to the weight of the node.

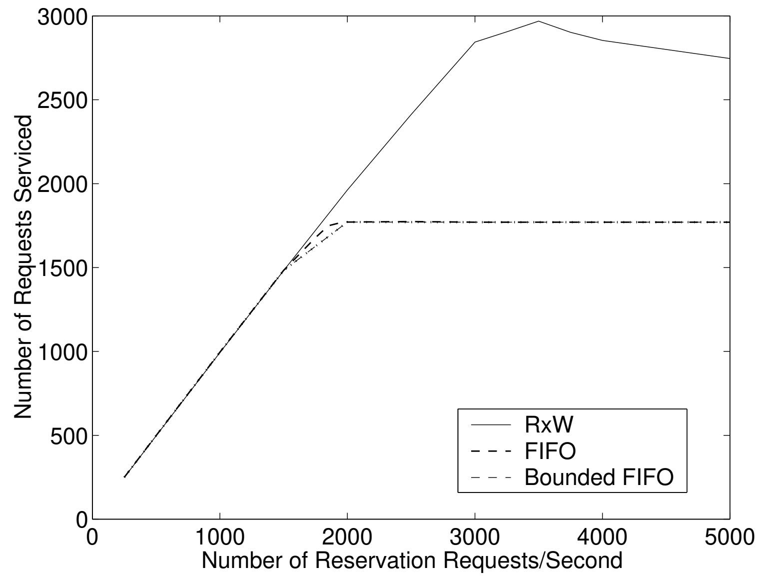


Figure 5.16: Throughput of a Clearing House node as the traffic load is varied.

We use three different scheduling policies to evaluate the CH-architecture. They are:

1. **RxW Scheduling:** as described in Section 5.5.2.
2. **FIFO Scheduling:** This is the normal scheduling policy based on the first come first serve principle.
3. **Bounded FIFO:** Bounded FIFO refers to FIFO scheduling with bounded response time, i.e., requests with high waiting times are directly dropped.

We measure the throughput, mean response time, mean tear-down time and call blocking rate for varying loads. We also measure the fairness of the different scheduling policies in terms of the variability of individual response time.

Throughput Characteristics

Throughput is measured as the number of calls serviced by the CH-node per second. From Figure 5.16, we observe that the peak throughput obtained using RxW is 71% more than the peak obtained using FIFO. By introducing a bounded response time policy in FIFO scheduling, the throughput is unaltered. Using RxW scheduling, the CH can successfully process 3500 calls/s while the CH can only take a load of 1850 calls/s using FIFO scheduling. The throughput of RxW drops once the load increases beyond 3750 calls/s.

Call-Blocking Characteristics

A call is "blocked" when the reservation request is dropped by the scheduler either due to insufficient resources or excessive load. The blocking rate obtained using RxW

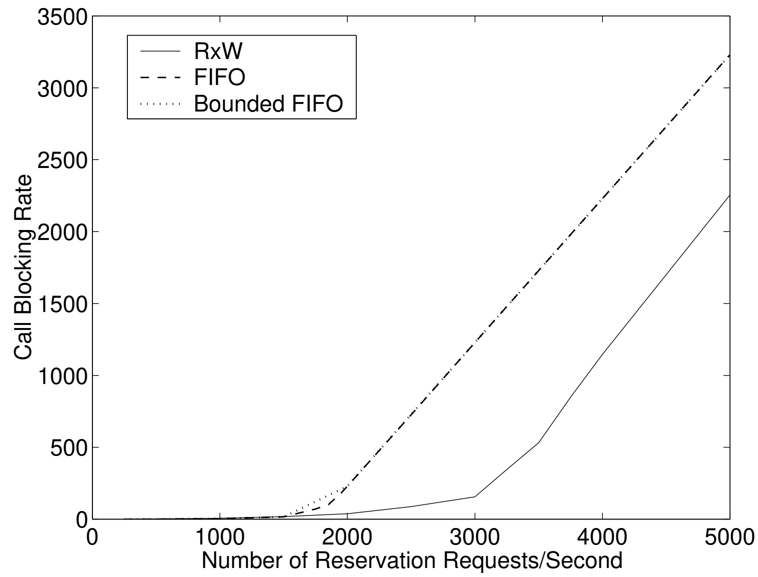


Figure 5.17: Call blocking rate as the traffic load is varied.

scheduling is much less than that of FIFO scheduling. The call blocking rate of FIFO scheduling is unaffected by the bounded response time constraint. The call blocking rate is negligible until a load of 1500 calls/s. For RxW scheduling, the call blocking rate is less than 10% until a load of 3200 calls/s. After a certain threshold, the blocking rate increases linearly with the load indicating a saturation point of throughput.

Response-time Characteristics

The response time is defined as the time taken to service a call request by the CH. In Figure 5.18, we plot the mean response time as a function of the load for the three scheduling policies. The response time of bounded FIFO is always lower than 0.5 s and is much lesser than that of normal FIFO after a load of 2000 calls/s. The mean response time of RxW increases linearly after 2500 calls/s. After a load of 1850 calls/s, the mean response time of FIFO scheduling shoots up rapidly and is an order of magnitude larger than bounded FIFO or RxW scheduling.

Tear-down Characteristics

It is important to measure the time taken to tear down the reservations along a particular path. Figure 5.19 shows some very interesting properties of the tear-down response time. When the throughput of the system decreases at 3700 calls/s load, the tear-down response time drops by 5 ms for RxW scheduling and stabilizes at 16 ms. The mean tear-down time for RxW scheduling shows a steep increase after a load of 2500 calls/s while the mean tear-down time for FIFO policies stabilizes at 6ms beyond a load of 1850 calls/s. The number of tear-down requests is proportional to the throughput of the system at a specified load. Hence, when the throughput of FIFO and RxW scheduling stabilizes at 2950 calls/s and 1770 calls/s at loads of 3500 calls/s and 1850 calls/s, the mean-tear down time stabilizes.

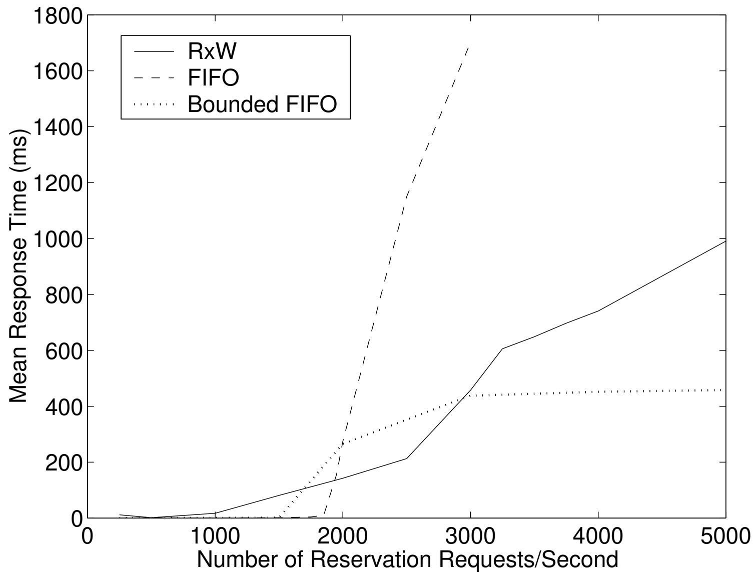


Figure 5.18: Mean response time as a function of traffic load.

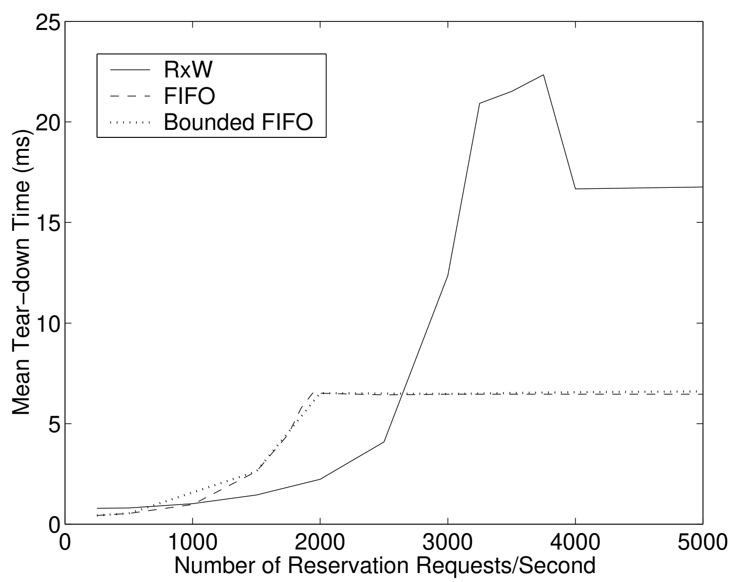


Figure 5.19: Tear-down response time as a function of traffic load.

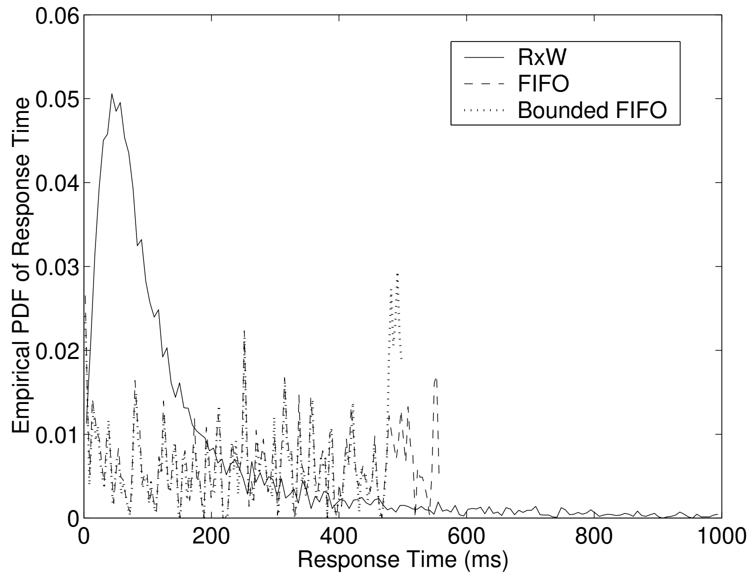


Figure 5.20: Distribution of response time at a load of 2000 requests per second.

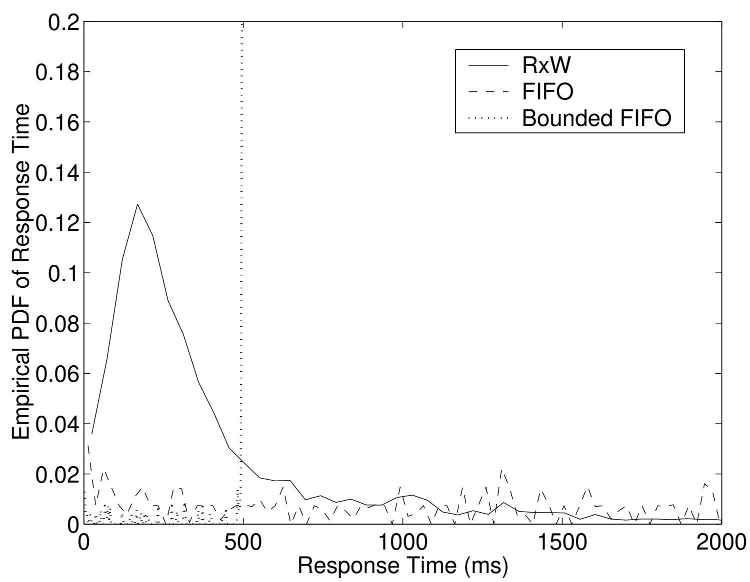


Figure 5.21: Distribution of response time at a load of 3000 requests per second.

Fairness of our Approach

We determine the fairness of our approach by studying the variations of response time. In a normal FIFO queue model, the response time of a call request would be proportional to the size of the queue. Since RxW scheduling tries to optimize the throughput of the CH, there might be a few requests which might have to wait for a long time before getting scheduled. Such requests will observe a high response time. In Figure 5.20 and Figure 5.21, we plot the variations of response time at two critical loads equal to 2000 calls/s and 3000 calls/s.

FIFO scheduling reaches its stable region at a load of 2000 calls/s, while the throughput of RxW scheduling is close to its maximum at 3000 calls/s. At a load of 2000 calls/s (Figure 5.20), a huge percentage (> 80%) of requests in RxW scheduling have a response time in the range of 50-220 ms while the response time of FIFO is distributed over the range of 30-600 ms. For the Bounded FIFO case, the majority (30%) of the requests experience response time of about 500 ms, and the response time of the remaining requests are distributed between 0-500 ms. At a load of 3000 calls/s (Figure 5.21), RxW scheduling processes most of the requests within 500 ms. A small percentage of the requests (<2%) in RxW scheduling suffer due to loss of aggregation and have a high response time (>1 second). The response time of FIFO is distributed equally between 30 ms -2 seconds. In other words, the response time of the normal FIFO are almost three times longer when the load increases from 2000 calls/s to 3000 calls/s. For Bounded FIFO, a huge percentage (>90%) of the requests have a response time between 450-500 ms.

5.6 Summary

To make the Clearing House scalable to a large user base, we introduced a hierarchical structure where resource control tasks are delegated to distributed CH-nodes in Chapter 4. In this chapter, we describe three of these resource control mechanisms: an aggregate reservation technique based on Gaussian predictors, a Traffic-Matrix Based Admission Control (TMAC) algorithm, and a RxW aggregate scheduling mechanism for servicing the flow requests. We have conducted thorough simulation experiments to study how each of these algorithm performs and the tradeoffs involved in tuning its parameters. The following are some of the observations:

Aggregate Reservations The key insight behind our approach is predictability of aggregate traffic (trunks). We approximate the arrival process of the intra- and inter-domain trunks as Gaussian, and measure their corresponding mean, μ, and variance, σ^2, during the measurement window of length T_{mea}. Aggregate reservations are set up based on the measured μ and σ. The value of T_{mea} should be small enough to track the traffic fluctuations, but not too small to incur unnecessary signaling overhead. We infer from our simulations that additional knowledge of workload is useful in choosing the optimum value of T_{mea}. Using VoIP as an example workload, our results show that this reservation technique can satisfy the QoS objective of voice applications (i.e., $< 1\%$ loss rate) with only 8% over-provisioning, when $T_{\mathrm{mea}} = 1$ minute.

This aggregate reservation scheme can be applied to other Internet workloads as long as the number of flows being aggregated is large enough for the Gaussian approximation to hold. For multiple time-scale traffic such as compressed video, the Gaussian predictor needs to be replaced by a more complicated model to capture the statistical variability at different time-scales.

TMAC Although admission control is only performed at ingress points of an ISP domain, it must to take into account the network-wide congestion level to avoid any potential impact on the performance of existing flows. TMAC addresses this by leveraging the knowledge of the traffic demand distributions within an ISP to compute the admission threshold, $U(s,d)$ for a particular flow that enters at ingress ER-s, and exits at egress ER-d. In our simulations, we explore the tradeoff between link utilization and packet loss rate as the value of σ, which controls the level of statistical multiplexing, is varied. In general, the network utilization levels increases with the value of σ, but it also becomes more likely to admit too many flows, resulting in degradation of QoS performance (higher packet losses). With VoIP traffic, TMAC can achieve 97% utilization level with less than 1% packet loss rate. To support the diverse Internet workload, TMAC needs to tune its control parameter dynamically based on either (a) the measured network performance through an on-line feedback loop, or (b) source parameters specified by end applications.

RxW Aggregate Scheduling We consider RxW scheduling for servicing reservation requests, and simulation results show that high throughput can be obtained while keeping reasonably low response time of an individual reservation request. The peak throughput obtained using RxW shows 71% improvement over the throughput obtained using FIFO. This implies that a CH-node can support greater number of users (better scaling property) using aggregate scheduling.

Chapter 6

Furies: Malicious Flow Detection via Aggregate Policing

As described in [22], a control architecture for provisioning network resources is essential for ensuring end-to-end Quality-of-Service (QoS) for IP-based latency sensitive applications. In Chapter 4, we proposed a Clearing House architecture to coordinate bandwidth allocation within and across domains to achieve statistical delay and packet loss bounds. Chapter 5 discussed how the CH-nodes adapts intra- and inter-domain aggregate reservations based on Gaussian predictors. In that chapter, we also presented the Traffic-Matrix based Admission Control (TMAC) scheme that takes into account network-wide traffic distributions within an ISP.

In addition to bandwidth reservations and admission control, an equally important resource control task is *traffic policing*, i.e., verifying that each admitted flow uses only its allocated share of network resources, and does not lie about its bandwidth requirement. This chapter focuses on malicious flow detection and traffic policing mechanisms, as shown in our thesis roadmap (Figure 6.1). We have designed a new control framework, called Furies[1], for detecting and policing malicious flows without having to keep per-flow state information at any edge routers. The words "malicious" and "misbehaving" are used interchangeably in our discussions to describe an admitted flow that violate its allocated share of bandwidth.

We show the scalability and the practicality of Furies through simulations, a prototype, and an implementation. Section 6.1 provides a brief overview of the design challenges, our contributions and evaluation methodology. Section 6.2 highlights the key features of Furies, and discusses how we arrive at several design choices based on our understanding of current Internet infrastructure. We also discuss how Furies can be extended to police inter-domain traffic subject to SLAs, but our main focus is on detecting individual flows.

Malicious flow detection (MFD) is an example of on-line change detection problems [116], in which one needs to detect the occurrence of abnormal traffic behavior as soon as possible, but with a low rate of false alarms. Section 6.3 describes the detection algorithm, called *M*alicious flow *D*etection via *A*ggregate *P*olicing (MDAP), that we have developed. The design of MDAP is driven by two goals: (a) to protect the well-behaved flows against resource depletion due to malicious activity, and (b) to identify and eventually penalize the malicious flows. Another desirable property is robustness with respect to noise

[1] Furies is the Roman name of the three Greek goddesses responsible for tormenting evildoers.

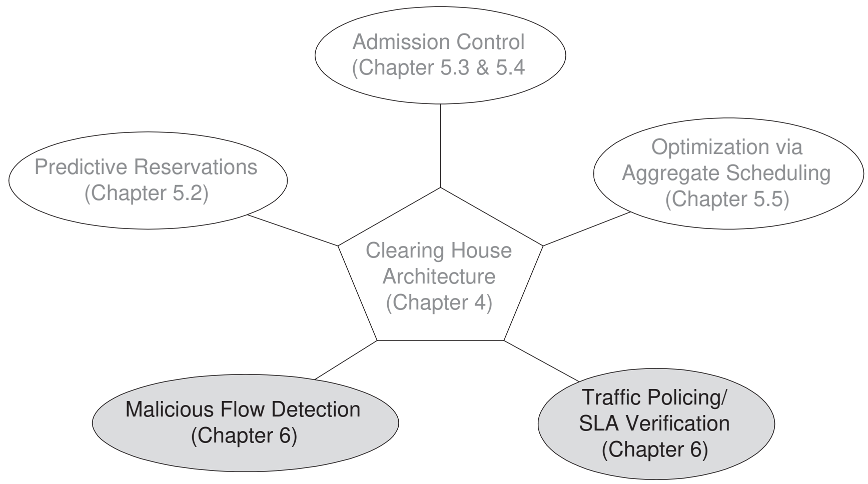


Figure 6.1: Thesis roadmap: The CH architecture and its various resource control mechanisms.

conditions and errors in workload modeling. The simulations discussed in Section 6.4 are designed to evaluate the effectiveness and robustness of MDAP. We also study the trade-offs between different performance criteria by tuning the parameters of MDAP.

Section 6.5 provides a brief description of our implementation of Furies based on Click Router [24], and demonstrates that the processing overhead that Furies introduced at an edge router is insignificant. Deployment issues are addressed in Section 6.6. Section 6.7 summarizes key results presented in this chapter.

6.1 Introduction

6.1.1 Motivation

Designing a control framework that performs scalable traffic policing in the edge networks faces numerous challenges. The most important is scalability. The overhead imposed by implementing the policies, such as amount of states maintained and processing time, should be bounded as the number of flows grows, i.e., should scale at most linearly with the number of flows. Another important requirement is compatibility with the existing Internet architecture. By compatibility, we mean that the proposed policies should be incrementally deployable and be able to reuse rather than replace the primitives supported by the existing network. The scalability, performance and deployment issues that might affect the design of a MFD scheme are not well understood. We attempt to bridge this gap in our work.

6.1.2 Traffic Policing and Malicious Flow Detection

After a high-priority flow is admitted, it is important to verify that this flow uses only its allocated share of network resources. A natural approach is to monitor every admitted flow at its ingress router, i.e., the entry point to a network domain. *Traffic policing* in the Diff-Serv literature usually refers to parameter-based packet filter mechanisms, which are useful in tracking and shaping per-flow usage. However, this requires every edge router to maintain per-flow information and incurs significant processing overhead, leading to poor scalability. In this chapter, *policing* refers to monitoring aggregate groups of admitted flows and detecting specific groups that violate their total allocated bandwidth. The ultimate goal is to uniquely identify the malicious flows within these detected groups. It is crucial to detect and penalize these malicious flows because they can potentially cause other flows that share the same link to suffer from congestion, resulting in delays and packet losses.

6.1.3 Performance Indexes

In general, the four intuitive performance indexes for designing and evaluating an on-line change detection system [116] are:

1. probability of *false alarms* (i.e., a flow is detected as malicious when it is not), P_{fa}
2. probability of *non-detection* (i.e., a malicious flow is not detected), P_{non},
3. delay for detection, t_{det}, and
4. magnitude of *detected change* (e.g., how much do malicious flows exceed their allocated bandwidth).

The choice of the parameters for a detection algorithm usually involve a set of trade-offs among these performance indexes. For example, if we choose a large value as the threshold for detection, the probability of false alarms decreases, but the probability of non-detection increases. Therefore, these indexes may be in different order of importance depending on the applications.

Since the main goal of this dissertation is to deliver end-to-end QoS assurance to latency sensitive applications, the first criteria is to protect the well-behaved flows that use legitimate amount of resources against malicious behavior. Identifying and penalizing the malicious flow itself is secondary, as long as the impact of the malicious activity of undetected flows on other well-behaved flows is negligible. This implies that malicious flows that cause the most damage should be detected as soon as possible and penalized, e.g., through packet drops, while smaller malicious flows are tolerable even if they are not identified. The exact magnitude of violation (detected change) is not important. To quantify how well the performance of well-behaved flows is preserved against malicious activity, we define P_{mis} as the probability of incorrectly dropped packets from the good flows. In summary, an ideal MFD algorithm in this case should achieve the following (in the order of importance):

- close to zero P_{fa},
- close to zero P_{mis},
- maximum possible successful detection probability, P_{fa}, (or 1 - P_{non}), and
- small t_{det},

6.1.4 Our Contributions

We propose a new control framework, Furies, that can be deployed by an Autonomous System (AS)[2] to detect malicious flows in an efficient and scalable manner. Our approach exploits the Internet's hierarchical structure and the economic relationships between Internet Service Providers (ISPs) to form special grouping of flows for aggregate policing. Within the CH-architecture, Furies resides in the core of a Local Clearing House (Section 4.4: Figure 4.7) as the main resource controller. An LCH can also host other logical entities that provide services such as security and billing, but a detailed analysis of those entities is out of the scope of this dissertation.

The main contribution of this chapter is designing, developing and evaluating MDAP (*M*alicious flow *D*etection via *A*ggregate *P*olicing), a mechanism within the Furies framework that detects malicious behavior through intelligent coordination of edge routers and aggregate policing. MDAP requires edge routers to maintain only a small amount of aggregate state information, and chooses its parameters based on the performance criteria listed at the end of Section 6.1.3 (as for an ideal MFD algorithm). Two other desirable properties that we consider are robustness and minimal processing overhead. Since MDAP does not require a priori knowledge of the individual flow characteristics, we evaluate its sensitivity to modeling errors in Section 6.4 by considering a variety of source models. We also simulate different extreme scenarios to investigate whether MDAP is robust with respect to noise conditions, e.g., when large and small flows are aggregated together for policing, or when the percentage of malicious flows are varied. For MDAP to scale well with a large user population, its processing overhead should be minimal. We study this issue through implementation in Section 6.5.

In our model, the source provider's network is responsible for monitoring the admitted flows and detecting malicious behavior. The subsequent ASs (ISPs or transit providers) only attempt to verify whether the inter-domain traffic from their upstream peers or customers adhere to their respective Service-Level Agreements (SLAs) [10]. An SLA specifies the expected level of service offered by a provider to the customer traffic it carries, e.g., maximum delay or packet loss guarantees, provided that the customer does not exceed the total subscribed bandwidth.

The key insight behind Furies is a coordinated way of assigning a flow-identifier, Fid, to every admitted *flow*, which allows aggregation of flows for traffic policing without compromising the ability to uniquely identify a flow if it is malicious. Each Fid has two subfields: $FidIn$ and $FidEg$. At ingress routers, admitted flows are aggregated based on their $FidIn$ for group policing. Similarly, egress routers police flows with the same $FidEg$ as an aggregate. The fact that each edge router maintains only the aggregate state for each *group*, identified by Fid subfields, is crucial for the reduction of state from $O(n)$, which would be required if each flow were policed individually, to $O(\sqrt{n})$, where n is the number of admitted flows.

Furies does not strive to provide hard end-to-end guarantees but rather statistical QoS, as provided by soft real-time services. We assume that only the high-priority traffic needs resource reservation and are admission controlled. Furies can co-exist with any Measurement-Based Admission Control (MBAC) [62] algorithms previously proposed. For

[2]An AS is a network domain administered by a single organization, e.g., an ISP, a transit provider, a campus or corporate network.

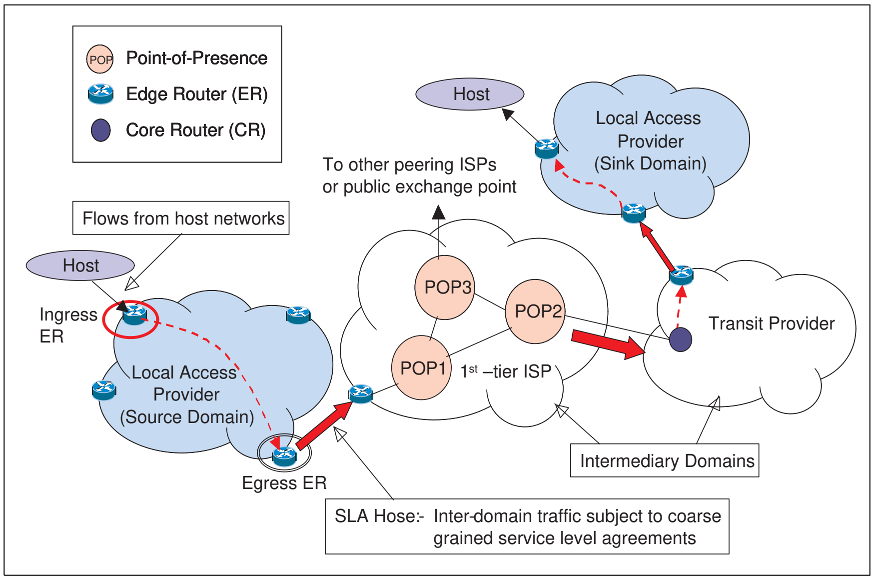


Figure 6.2: An example logical map of the Internet infrastructure.

evaluation purposes, we use the Measured Sum (MS) algorithm [63]. A detailed comparison of the different MBAC algorithms is out of the scope of this chapter.

6.2 Design Rationale

Our design decisions are influenced by the inherent Internet hierarchy as described in Chapter 4.2.1. Figure 6.2 illustrates the logical relationship between host networks, LAPs, transit providers and ISPs. For ease of discussion, we introduce the following definitions:

- A **flow** refers to a high-priority stream identified by its source and destination IP addresses.
- A **source domain** refers to the first AS (usually LAP or ISP) that a flow from a host network is routed to.
- A **hose** is a collection of flows that crosses from one domain to another, e.g. between two peering ISPs, from LAPs to transit providers or to first-tier ISPs.
- An **Ingress ER** is the edge router where a flow enters a domain and an **Egress ER** is where it exits.

Furies attempts to preserve end-to-end performance assurance by auditing the high-priority traffic that is admitted into a domain. The following characteristics are responsible for the scalability and robustness of our architecture.

6.2.1 Furies Service Model

We treat the traffic coming from another provider's domain differently from the individual flows from host networks, because resource allocation decisions for these cases happen in vastly different time-scales and granularity. In the former, the transit traffic (hose) is usually subjected to peering agreements or SLAs [10] that reflect aggregate traffic performance (e.g., average packet loss) and stay effective over long time-scale, e.g., weeks or months. In the latter, the reservation requests from individual flows usually need fast resource control decisions, e.g., within milliseconds, and finer-grained performance guarantees.

We assume that the source domain (e.g., the LAP shown in Figure 6.2) is responsible for policing individual flows from host networks and ensuring that the aggregate traffic (hoses) entering the subsequent ISP domains do not violate the associated SLAs. In other words, it is important for the source domain to identify and stop malicious flows before they leave its network. The subsequent ISPs will treat these flows as part of a "hose" coming from the source domain and police them as an aggregate.

Furies uses the "core-stateless" principles [54], in that our architecture differentiates between edge and core nodes. While edge nodes do perform per flow management, core nodes do not and therefore can be efficiently implemented at high speeds. This significantly reduces the implementation complexity since no state information is maintained in the core. Our architecture builds on many of the existing Diff-Serv primitives, including:

Packet marking and classification The Diff-Serv Code Point (DSCP) [15], which is the first six bits in the TOS byte in the IP-header, is marked to differentiate packets from different classes of applications, e.g., high-priority voice flows vs. best-effort data traffic. Bits in the TOS octet are set at the network edges or administrative boundaries. Core routers simply classify the packets into different queues based on their DSCP values.

Expedited forwarding PHB The granularity of service provisioning is a "class" in Diff-Serv, and multiple flows with the same DSCP value are mapped on to a single per-hop behavior (PHB) at a router. We consider expedited forwarding PHB [18] because it is appropriate for applications that require a hard guarantee on the delay and jitter, such as VoIP. It can be implemented using a priority scheduler that always schedule packets from high-priority queue first whenever it is not empty.

Traffic policer In Diff-Serv, the Traffic Conditioning Agreement (TCA) [15] provider and the customer may specify that packets submitted for a certain service level (as specified by the DSCP) and are deemed to be non-conforming may be re-marked to a lower service level. This remarking is performed by a traffic policer. The simplest form of policing is dropping the packets, which is sufficient most of the time.

6.2.2 Flow-Identifiers and Group Policing

Since a typical ER in a local POP can observe up to 5000 [97] simultaneous flows, per-flow policing is not ideal because it incurs a huge processing overhead. Instead, we propose to aggregate flows for group policing. To be able to uniquely identify a malicious flow, Furies assigns each admitted flow a unique 32-bit flow-identifier, Fid, which is inserted in the packet header. This Fid is explicitly assigned by Furies and is not assumed as a random number by the flow. Every Fid consists of two 16-bit sub-fields: $FidIn$ and $FidEg$. All flows that enter or exit at a particular ER are aggregated into different groups based on their $Fids$. Each of these groups is identified with a unique group-identifier. The

FidIn and *FidEg* subfields of a flow refer to the group identifiers used by its ingress and egress ER, respectively.

Furies aggregates all flows that share the same subfield *FidIn* (or *FidEg*) and polices them as a group at the associated ER. Every flow is policed at both its ingress and egress ER in two distinct groups, thereby increasing the chances of detecting malicious flows. Every ER maintains only the aggregate state for each group and hence does not store any per-flow state.

6.2.3 Assumptions

We make the following assumptions in our design:

- All routers can support two QoS classes: Best-effort and High-priority. From Chapter 3.2.2, we found that the perceived quality of VoIP application is satisfactory if the packet loss rate is less than 1% and per hop queuing delay is minimal (less than 5 ms). We use this as a guideline for the performance requirement of high-priority class.
- For any particular flow that enters a domain, we can infer the associated egress ER from the underlying routing protocol. We do not modify any routing decisions.
- Packets that violate traffic profiles can either be dropped or re-marked to lower priority levels at the traffic policers. Our architecture can support both cases equally well, but for simplicity, we choose packet dropping as the default.
- Furies requires explicit REQUEST and TEARDOWN messages for admitting and releasing every flow. The signaling messages are generated by either the customer router or a proxy and sent as UDP packets at the same level of priority (high).
- We assume the reservation agent or proxy that issues the REQUEST message (on behalf of the host) is also responsible for inserting the assigned 32-bit *Fid* in the packet header if the flow is admitted.

6.3 Furies Architecture and MDAP Mechanisms

In this section, we provide a detailed description of our algorithms for malicious flow detection.

6.3.1 Components of Furies

Furies adds two components to edge routers (ERs) to implement its policies: Traffic Monitors (TMs) and Traffic Policers (TPs). Each ISP domain has a Resource Manager (RM) that interacts with the ERs continuously to admit new flows and coordinate edge policing.

The RM is a logical unit that can be physically placed at the fault monitoring point or policy server in an ISP. The RM maintains the repository of assigned *Fids* and their allocated bandwidths in the Fid-Repository (FR). It also contains the admission control policy. Figure 6.3a illustrates how the control messages flow between the various Furies components. When a new REQUEST message arrives at ER-s, it is forwarded to the RM, which performs admission control and assigns an *Fid* if admitted. Upon successful admission, the RM then sends an ACCEPT message along with the *Fid* back to the host. It also

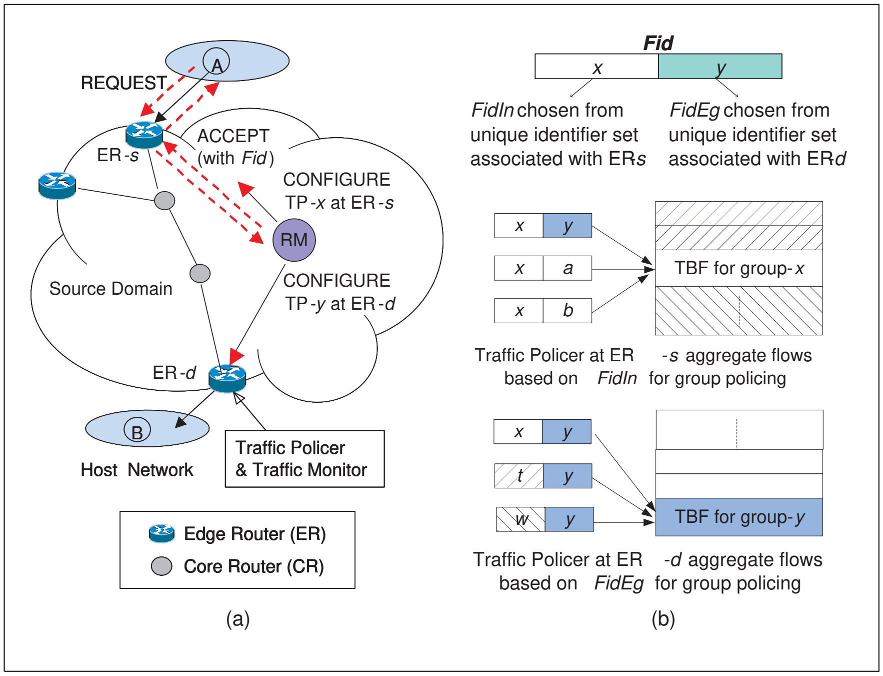


Figure 6.3: (a) Components of Furies. (b) MDAP mechanisms.

updates the traffic policers at the associated ingress and egress ER using the CONFIGURE message. Otherwise, the RM sends a REJECT message.

A Traffic Monitor (TM) at each ingress ER passively measures the rate of admitted traffic and updates the RM periodically. These updates are used by the RM for measurement-based admission control (Chapter 5.3). A Traffic Policer (TP) is introduced at each ER to police groups of flows identified by subfields of their $Fids$. In the example shown in Figure 6.3b, the new flow is assigned an Fid with the first subfield, $FidIn$ equals to x, and $FidEg$ equals y. At ER-s, the new flow is aggregated with other flows with $FidIn$ $= x$ for group policing. At ER-d, this flow is grouped with other flows with $FidEg = y$ for policing. Figure 6.4 highlights the major control blocks within the RM, which will be discussed in detail in the following section.

6.3.2 Fid Assignment and Releasing

Furies assigns each edge router ER-i a set of M unique group identifiers, denoted as $\mathcal{A}_i = \{x_{i1}, x_{i2}, \ldots, x_{iM}\}$, where each member is a 16-bit binary number and is unique across the set $\mathcal{A}_i$. The sets $\mathcal{A}_i$ and $\mathcal{A}_j$ associated with any two ER-i and j are mutually disjoint.

Any Fid of an admitted flow should satisfy the following properties:

1. If the flow is routed from ER-s to ER-d, then $FidIn \in \mathcal{A}_s$, and $FidEg \in \mathcal{A}_d$.

2. No other flow should have the same Fid.

3. Flows with the same $FidIn$ (or $FidEg$) have similar bandwidth requirements.

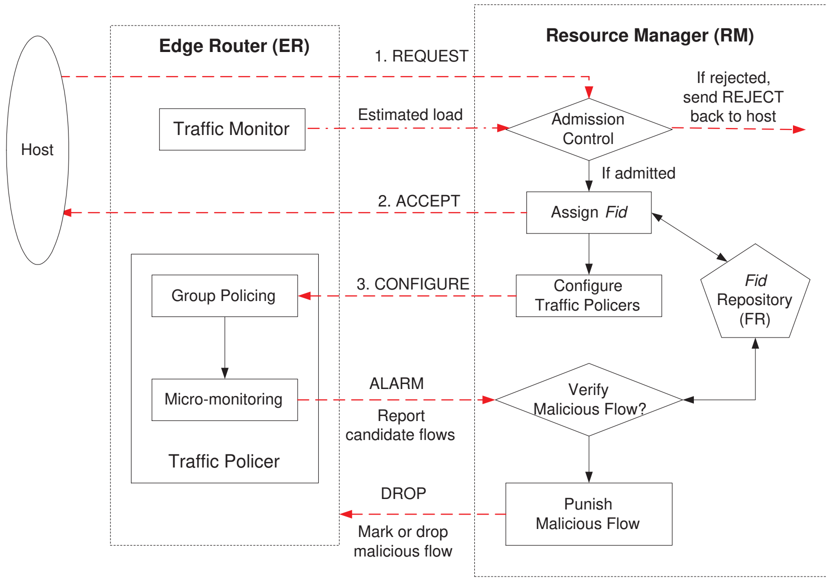


Figure 6.4: The logical flow of control messages between an edge router (ER) and a Resource Manager (RM).

When a new flow from ER-s to ER-d is admitted, the RM picks a random x_s from $\mathcal{A}_s$ and a random y_d from $\mathcal{A}_d$ such that the Fid with $FidIn = x_s$ and $FidEg = y_d$ satisfies the above properties. This Fid is assigned to the flow, and a new entry with this Fid and its allocated bandwidth is added to the Fid-Repository (FR). The total number of flows that can be uniquely identified in this scheme is $K \cdot M^2$ for a particular ingress ER, where K is the total number of potential egress ERs, each having its own unique set of identifiers. We assume the admitted flow will send a TEARDOWN message to the ingress ER when it terminates. The TEARDOWN message contains the Fid, and its allocated peak rate. Upon receiving the TEARDOWN message, the RM updates the FR accordingly and releases the Fid.

Demand for Fids

In [97], the authors built a passive monitoring infrastructure over their backbone to collect and analyze real Internet traffic. They observed that the typical number of simultaneously active flows on an ingress link (between an edge router and a core router) is between 300 and 5000. From the trends in application usage seen at the NASA Ames Internet Exchange [117], we estimate that about 10% of these will be latency sensitive applications such as streaming media and online gaming [3] Even if the demand for Fids increases 10 times in future, we need at most $M = \sqrt{5000}$, which is roughly 71 unique identifiers per $\mathcal{A}$ set. Based on the discussion in Section 6.2[4], the total number of ERs within an ISP, K, is in the order of 150-500. Therefore, the total number of unique identifiers

[3] Results in [117] are based on analysis of Internet traffic trace data collected at Ames Internet Exchange (AIX) over 10 months, from May 1999 through March 2000. On average, the fraction of Real Audio Traffic packets seen at AIX is between 0.5-6% of the total traffic, while the fraction of online gaming traffic is between 0.5-4%.

[4] K= number of POPs $\times$ number of ERs/pop.

required for the whole ISP ($M \times K$) is roughly 35,500 in this case. Allocating 16-bits ($2^{16} = 65536$) for each subfield should be more than sufficient for producing mutually disjoint $\mathcal{A}_i$ for all routers.

If the demand ever exceeds the available Fids, we can either (a) increase the size of Fids, or (b) allow a small probability of collisions by recycling used Fids. A comprehensive analysis of malicious flow detection with Fid collisions is part of our future work.

Explicitly Assigned vs User-Selected Fids

In our approach, we attempt to maximize the level of flow aggregation without compromising the uniqueness of $Fids$, thereby minimizing the number of groups to be policed at every ER. An explicit assignment of $Fids$ can achieve this since the RM can keep track of the availability of individual $Fids$ and allocate unused ones to new flow requests. For example, if there are n flows from an ingress ER to a specific egress ER, an explicit assignment can preserve uniqueness by maintaining only $\sqrt{n}$ groups at each of the two routers. It also allows the aggregation of flows with similar bandwidth requirements into a common group for policing to increase effectiveness of group policing.

On the other hand, if flows were allowed to assume their own $Fids$, then it would be necessary to maintain a membership function to map the random $Fids$ to a particular group. The cost of performing an online grouping of flows based on these functions would be very high because the $Fids$ are continuously changing. Techniques like Stochastic Fair Blue (SFB) [75] that use random hash functions cannot be applied because they do not provide an inverse mapping from group identifiers to actual $Fids$. Thus, using SFB alone does not provide a direct mechanism to verify whether a suspected flow is truly misbehaving.

6.3.3 Group Policing

Furies deploys a set of Token Bucket Filters (TBF) [118]) at both ingress and egress ERs for policing the traffic generated by admitted flows with the matching group identifiers ($FidIn$ or $FidEg$). Every group identifier, $x \in \mathcal{A}_i$, is associated with a TBF with two parameters, r_{tot} and b_{tot}, where r_{tot} is the total average rate of admitted flows and b_{tot} is the total burst size. When a new flow between ER-s and ER-d is admitted, the RM sends a CONFIGURE message that specifies the $FidIn$ and $FidEg$ of the admitted flow to the ingress ER-s and egress ER-d, respectively, along with its average rate r and burst-size b. TP_s and TP_d will then update the the TBF with the matching $FidIn$ and $FidEg$ accordingly. Packets that violate the associated traffic profile are discarded. Each TP keeps track of the dropped packets and reports the statistics to the RM.

Since the policing at the TP is performed on a group of flows sharing the same 16-bit subfield of Fid, the amount of state information maintained at the ingress ER is proportional to M, the number of unique identifiers in the set, $\mathcal{A}$. Consider an example ISP domain with K edge routers and M=100. Each ER maintains only 100 pieces of state information, but an arbitrary router can admit up to $K \times 10,000$ flows with unique Fids. A per-flow policing scheme would have require each ER to maintain all $K \times 10,000$ states.

6.3.4 MDAP Detection Process

There are two stages in the MDAP detection process:

- Guess candidate flows within the group that violate aggregate bandwidth allocation, and
- Verify which of these flows are truly malicious.

Providing Best Guesses

As an example, let a flow with $Fid = [f, g]$ be malicious. All flows with the same $FidIn = f$ will be policed as an aggregate in the same Token Bucket at the ingress ER, TP_s, regardless of what their $FidEg$ is. If the total allocated rate of $FidIn = f$ is violated, the affected TP reports f to the RM using an ALARM message (Figure 6.4). However, this information alone is insufficient for pinpointing the exact misbehaving flow, because there can be as many as $K \cdot M$ flows with the same $FidIn$ that could potentially be malicious. If the TP at an egress ER $FidEg = g$ also sends an ALARM message, the RM guesses that a flow with Fid that contains both f and g as its subfield is malicious, and submits this Fid for verification.

However, a malicious flow may not always be detected at both its ingress and egress ERs. To improve the effectiveness of MDAP, we introduce a "micro-policing" mode. Whenever a group TBF that violates the aggregate rate is identified, Furies requires the edge routers to randomly sample individual flows within this group for a duration of t_{mp}. At the end of this period, the associated ER identifies n_{mp} largest flows, and report their Fids, along with the sampled peak rate, to the RM. Normally the potentially malicious flows are the ones that transmit at a much higher rate relative to other members. The value of t_{mp} should be as small as possible to ensure short detection time (t_{det}), but it has to be large enough to observe malicious behavior, especially if the flow is bursty. From our trace analysis (Chapter 3.2.3, [119]), we observed that the audio trace with the minimum activity cycle has mean silence period of 4.9 seconds. Using this as a rough guideline, we choose $t_{\text{mp}} = 5$ seconds. On the other hand, the choice of n_{mp} depends on the relative number of malicious flows in the network. Large value of n_{mp} provides better chances of catching all the malicious flows (if they are many), but incurs higher processing overhead. We perform sensitivity analysis by varying the value of n_{mp} from 1 to 10 and found that the performance difference is negligible for $n_{\text{mp}} \geq 5$, given that 10% of the flows are malicious. We use n_{mp}=5 in all our experiments.

Verifying Malicious Behavior

For each reported flow with $Fid = b$, the RM compares the allocated rate, r_b with the measured peak rate m_b reported in the ALARM message:

$$m_b < (1 + \epsilon) \cdot r_b \tag{6.1}$$

where ϵ is a hysteresis parameter to absorb transient behavior of bursty traffic. If the condition in (6.1) is violated, the flow is considered misbehaving. ϵ is typically between 0 and 0.05. A counter associated with this flow is incremented for every such violation of condition (6.1). To reduce the probability of false alarm, we introduce a second hysteresis parameter, η, which determines the minimum number of violations before a flow is reported as misbehaving. A reasonable range for η is between one and five.

Several actions can be taken against a detected flow, e.g., dropping all the packets of this flow, demoting all its packets to best-effort, or charging more for the connection. Such a penalty would require keeping some state information at the edge router, but only for a very small subset of flows that misbehave. The choice of the penalty function is dependent on the business goals of the providers, which will not be addressed in this dissertation. Instead, we focus on providing insights into the technical design of MFD scheme itself.

6.3.5 Policing of SLA Traffic

Service Level Agreements (SLAs) are contracts between service providers and customers, in which the providers stipulate their commitment to deliver numerous service levels to their customers with an agreed-upon pricing scheme. In return, the customers are bound to only use the service they pay for and will be penalized otherwise. For example, consider a customer that signs up for fractional T-1[5] line, say for 25%, which is equivalent to 386 Kbps. The customer is required by the SLA to keep the peak link utilization below 386 Kbps. Delivery of excess traffic beyond that is not guaranteed by the service provider.

SLA Performance Guarantees

Previously, SLAs have primarily focus on backbone performance such as reliability and availability, e.g., a typical SLA requirement for T-1 and frame relay service should be 99.93% availability or 60 seconds of disrupted service in 24 hours. In recent years, ISP competition has fueled stronger SLAs that specify packet losses and delay guarantees. For instance, Cable & Wireless is offering its dedicated Internet access customers a protective guarantee that they will experience an average latency of no more than 70 ms per month across the ISP's Internet backbone. It has also added a packet-loss protection guarantee of no more than 1% over one month. However, this implies that the customer packets can experience 1 full second delay for one day, and 35 ms for all the other 29 days in a month, and still average less than 70 ms per month. It would be impossible for the customer to run any real-time audio or video applications for that particular day where the delay is greater than the acceptable range for satisfactory perceived quality (Chapter 3.2.2). With the increasing diversity of Internet applications, there is a strong need for the ISPs to provide finer-grained performance guarantees that refect application-level requirements.

SLA Traffic Policing

SLA validation tools are currently available (e.g., from VisualNetworks[6] and TeleChoice[7]) for business users to measure the latency, packet loss and network availability, and to validate whether the service providers meet the SLAs. One area where all ISPs are lacking is in real-time monitoring tools to verify whether the customers abide by the SLAs in terms of generating traffic. This is crucial, so that the ISPs can offer more specific SLAs with finer-grained performance assurance (as discussed earlier) through admission control, load balancing, and capacity planning.

[5] A T1 line can carry 24 digitized voice channels or it can carry data at a rate of 1.544 Mbps.
[6] `http://www.visualnetworks.com/`
[7] `http://www.telechoice.com/`

Furies does not address how the new SLAs are specified or negotiated. Instead, we propose a way to police the SLA traffic that enter a domain. As a basic rule, Furies favors individual flows (as discussed in previous section) over SLA-traffic, i.e., SLA traffic has lower priority than flows with assigned Fids, and SLA packets will be remarked or dropped first when congestion happens.

We assume a domain k will issue an SLA request to reserve the required bandwidth, R_{kq}, to carry a "hose" from its domain to the neighboring domain q. For a domain q, each admitted SLA-hose from its neighboring domain k is assigned an identifier S_k^q. A traffic profile consisting of two parameters: S_k^q and the allocated rate R_{kq} is maintained at the ingress ER-i where the SLA traffic enters. All the packets from SLA-flows with the matching S_k^q are policed as an aggregate. If the sampled aggregate rate exceeds the allocated rate R_{kq}, the packets are marked non-conformant and dropped. A control message that indicates the violation is sent to the originating domain p.

6.3.6 Other Issues

In this section, we discuss how MDAP can be modified to cope with diverse traffic loads, untrusted networks, and changing routing policies.

Hiding in the Aggregate

Although group policing allows the Furies architecture to scale, it limits the effectiveness of MDAP because a malicious flow can "hide" in the aggregate. This can happen when:

- the aggregate usage of the group is less than the total allocated rate because certain flows are under-utilizing their resources or the percentage of over-utilization is less than the threshold ϵ, and
- the malicious flow is relatively small compared to other larger, yet legitimate, flows in a misbehaving group.

To address this problem, Furies introduces redundancy by deploying TP at every egress point as well. By assigning a unique Fid to every flow, we ensure that no two flows are in the same group in both the associated ingress and egress ERs. Essentially every flow is policed in the aggregate at two distinct points to maximize the number of malicious flows that are detected. Secondly, we assign flows with similar bandwidth requirement into the same group (Property 3 of Fids in Section 6.3.2).

Bogus Identifiers

It is possible for an external malicious user/application to create its own Fid which is valid but has not been explicitly assigned by the RM. We refer to such identifiers as *bogus* Fids. This problem can be easily solved by using a secure hash function with a secret key within an ISP. Assume that the RM and the ERs share a secret key, K, for a hash function $h()$. Given an $Fid = [f, g]$, the secure-Fid allocated to the flow is set to $[h_K(f), h_K(g)]$, from which f, g cannot be inferred without knowing K. The associated ingress (egress) ER can authenticate the flow by comparing the $FidIn$ ($FidEg$) of the secure-Fid with the hashed values of valid group identifiers (e.g., f and g) of the ER.

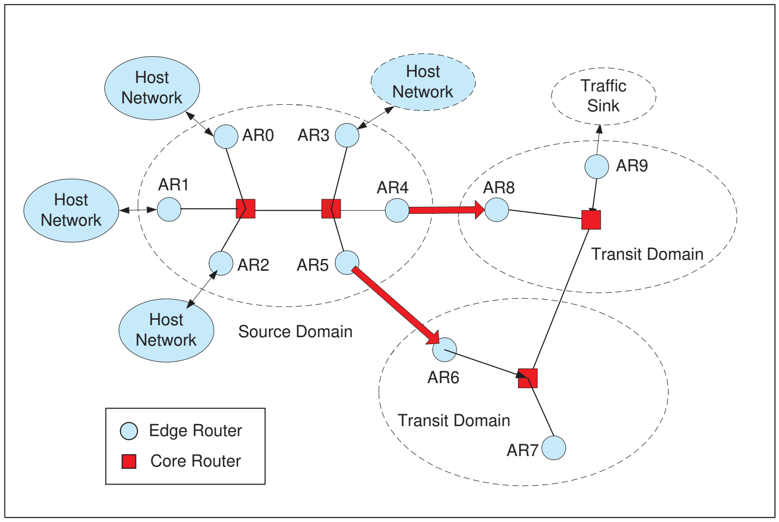


Figure 6.5: Simulation topology.

Routing Changes

When a routing change occurs and causes an active flow to change its ingress or egress points, the previously assigned Fid may not match the group identifiers in one of the new ERs or both. Whenever this happens, we can either (1) remark the packets of this flow as best-effort, or (2) contact the RM for re-admission of this flow with the new endpoints. Further investigation is needed to understand the the performance and security issues of both approaches. However, we believe such events are rare based on our discussions with two first-tier ISPs.

6.4 Simulation Study

The aim of the simulation study is to evaluate the performance and robustness of Furies. We use the ns-network simulator [89] to implement the basic mechanisms of Furies. The TP[8] is implemented as a connector in front of a node, and a time-window estimator is introduced at each input link to estimate the rate of existing flows. The admission control module is created as an NsObject and inserted before the ingress ERs. The various tasks of the RM in Furies are implemented at the Tcl-level. Our Furies-patch works for ns-2.1b6.

6.4.1 Network Topology

We use ns to simulate a simple subgraph of the Internet topology as shown in Figure 6.5. It consists of 4 ASs (a source domain, two transit domains and one traffic sink) and captures the general properties of the Internet, including the POP structure of large ISPs and peering relationship with transit domains. Flows from host networks enter and exit the source domain through edge routers, AR0-AR5, where they are aggregated for policing using the MDAP scheme. The transit traffic (or hoses) that traverse from the

[8]We modify the Diff-Serv module contributed by Sean Murphy, http://www.teltec.dcu.ie/ murphys/ns-work/diffserv/index.html.

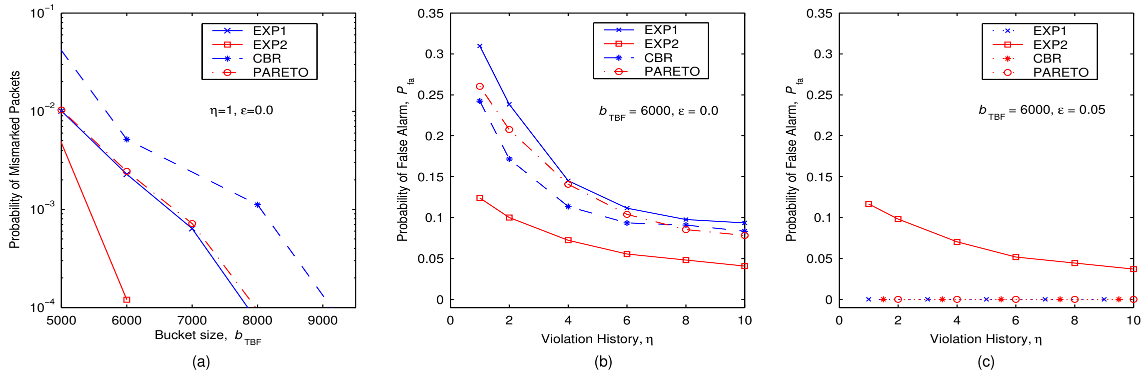


Figure 6.6: Case 1: Zero Malicious Flows.

source domain (at AR4 and AR5) to another two AS domains are subject to SLA policing at AR6 and AR8, respectively. All routers support priority scheduling and all control signaling between Furies-RM (not shown on the figure) and the ERs is carried in UDP messages.

6.4.2 Traffic Generation

Similar to Chapter 5.4, the arrival process of the admission-controlled traffic is modeled as Poisson with arrival denoted as $\lambda_i(t)$. We use the indices t to indicate the time-of-day dependence of the traffic demand as reported in [102] and [97]. To reflect the realistic traffic demand, we introduce $\pm$ 10-15% changes to $\lambda_i(t)$ at a regular interval of 30 minutes. The traffic distributions from an ingress ER to a set of egress ERs are based on a random probabilistic model. We also analyze two extreme cases where: (a) the traffic is equally distributed to all egress ERs and (b) some egress ERs attract heavy traffic while the other egress ERs are relatively idle. Neither cases affect the results. This is not surprising since the design of MDAP is independent of the traffic distribution model.

Since MDAP does not assume a priori knowledge of source traffic we need to evaluate its robustness with respect to variation in workload characteristics. We use four kinds of traffic source models in our experiments: EXP1, EXP2, CBR and PARETO. In Chapter 3 we discussed the characteristics of each model and how it can be used to describe certain types applications. This information is summarized in Table 5.1, which can be found in Chapter 5.4.2.

6.4.3 Performance Evaluation

To quantify the effectiveness of the malicious flow detection (MDAP) scheme in Furies, we are interested in the following two events: successful detection and false alarm (Section 6.1.3). The probability of successful detection P_{sd} is approximated as the fraction of malicious flows that are actually detected. Similarly, the probability of false alarm P_{fa} is the fraction of normal flows that are incorrectly reported as misbehaving. Since the flows are policed as an aggregate, malicious flows can cause packets from complying flows to be dropped. The probability of a packet being incorrectly dropped, denoted as P_{mis}, quantifies the impact of malicious flows on end-to-end performance seen by other member flows. The goal of MDAP is to maximize P_{sd}, while minimizing P_{fa} and P_{mis}.

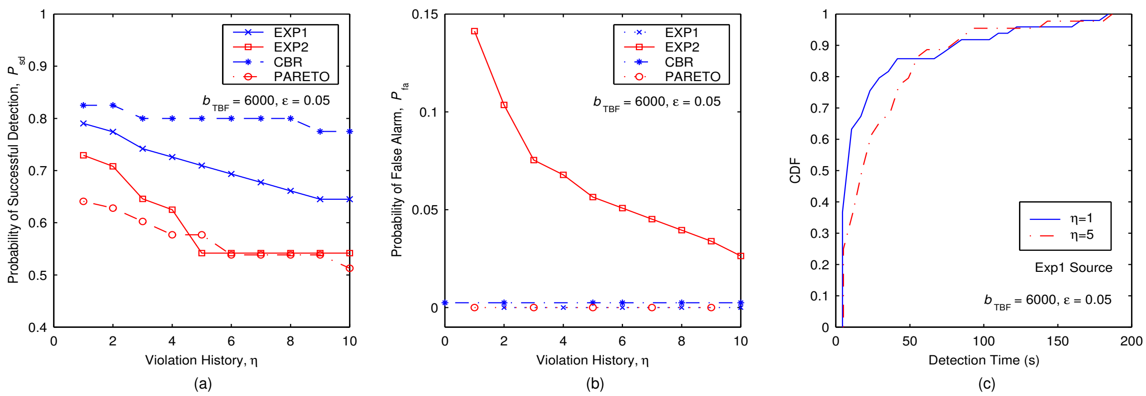


Figure 6.7: Case 2: Many small homogeneous flows; a small fraction, γ =0.1, misbehave.

Test Scenarios

To examine the robustness of MDAP, we simulate four extreme cases:

Case 1: This is the reference case with zero malicious flows.

Case 2: This arrangement is similar to Case 1, but now a small fraction, γ, of the flows are misbehaving.

Case 3: We assign Fids such that one large flow and many simultaneous small flows are grouped together for policing. The peak rate of the large flow is 10 times larger than the peak rate of a small flow. All of the small flows are compliant, and only the large flow misbehaves.

Case 4 Again, we consider a mixture of one large flow and many simultaneous flows like Case 3. However, the large flow is compliant this time, and a fraction γ of the small flows are malicious.

We repeat each experiment using four different source models: EXP1, EXP2, CBR and PARETO. For each scenario, the simulation was repeated 10 times with different random seeds, and the average P_{sd}, P_{fa}, and P_{mis} was computed. Each simulation ran for 1000s. All experiments were performed under high load with 20% blocking probability. A malicious source requests allocation for r Kbps but sends traffic at a higher rate, randomly chosen between $1.1 \cdot r$ and $1.2 \cdot r$ Kbps (10-20% violation). The average and peak rate for each source model is the same as described in Section 6.4.

Results and Discussions

Case 1: The experiments in Case 1 are intended for understanding the limitations of the MDAP and its performance sensitivity with respect to different choices of design parameters. Ideally, none of the flows should be reported as "misbehaving", but the transient behavior of bursty traffic could momentarily overflow the Token Bucket Filters (TBFs) and be interpreted as malicious, leading to a "false alarm". Figure 6.6a shows how the choice of bucket size, b_{TBF} at the Traffic Policers (TP) affects the

Table 6.1: Case 3: One large malicious flow and many small complying flows. η = 5, b_{TBF} = 6000, ϵ = 0.05.

Source	EXP1	EXP2	CBR	PARETO
P_{sd}	1.0	1.0	1.0	1.0
P_{fa}	0.0077	0.027	0.012	0.0013
P_{mis}	0.003	0.00021	0.0072	0.0037

probability of mis-marked packets, P_{mis}. A smaller value of b_{TBF} is more effective in detecting misbehaving flows, but there should be enough tokens to allow the legitimate packets to pass, and keep the P_{mis} low. Except for the CBR traffic, P_{mis} is below 0.01 for other source models. For the CBR source, increasing b_{TBF} to 6000 is sufficient to reduce P_{mis} to 0.005. The two hysteresis parameters η and ϵ determine under what conditions a flow is reported as “malicious” (Section 6.3.4), but have no effect on P_{mis}.

We can relax the condition for MDAP by increasing ϵ and η, and this helps to reduce the number of false alarms. Figure 6.6b and 6.6c study how P_{fa} varies as a function of η for $\epsilon = 0.0$ and 0.05. For a 0% tolerance level in MDAP (i.e., ϵ=0), P_{fa} decreases gradually as η is increased. However, we notice that P_{fa} drastically decreases for all the source models when the tolerance level is increased to 5%. This indicates that P_{fa} is more sensitive to ϵ than η. For the rest of the experiments, we choose ϵ=0.05 and b_{TBF} = 6000.

Case 2: Increasing η causes a delay in reporting misbehaving flows and may adversely impact the effectiveness of MDAP. We examine this issue in Case 2. We set the value of γ (fraction of misbehaving flows) to be 0.1. Figure 6.7a and 6.7b show the variation of P_{sd} and P_{fa} as η is increased from 1 to 10. The effect of η on P_{sd} for the CBR source is minimal. For the other source models, P_{sd} decreases sharply when η is increased and the rate of decrease varies across the source models. From Figure 6.7b, we can infer that only in the case of the EXP2 source model is P_{fa} sensitive to the value of η. With $\eta = 1$, we can detect most of the malicious flows with EXP1 (79%), CBR (83%) and PARETO (64%) sources with virtually zero P_{fa}. In the case of EXP2, there is a trade-off between maximizing P_{sd} and minimizing P_{fa} as we choose the value for η. This indicates that burstier sources are more difficult to detect.

We also measured the detection time for each correctly identified malicious flow and plotted the distributions in Figure 6.7c. With $\eta = 1$, the average detection time is 26.9 seconds, which is less than 1/10 of the average duration of a flow. 90% of the flows are detected within 78.9 seconds. When we increase η to 5, the average detection time increases to 33.8 seconds, which is still reasonably fast. The 90^{th}-percentile detection time is 80.4 seconds in this case.

The simulations in Case 2 show the basic results of MDAP hold across different source models. Although long range dependent traffic like PARETO is harder to detect, we can achieve a reasonable success rate (0.64) with zero false alarms. The presence of burstier sources, EXP2, pose challenges to the MDAP scheme, and we need to choose the value for η carefully to maximize P_{sd} while keeping P_{fa} reasonably small.

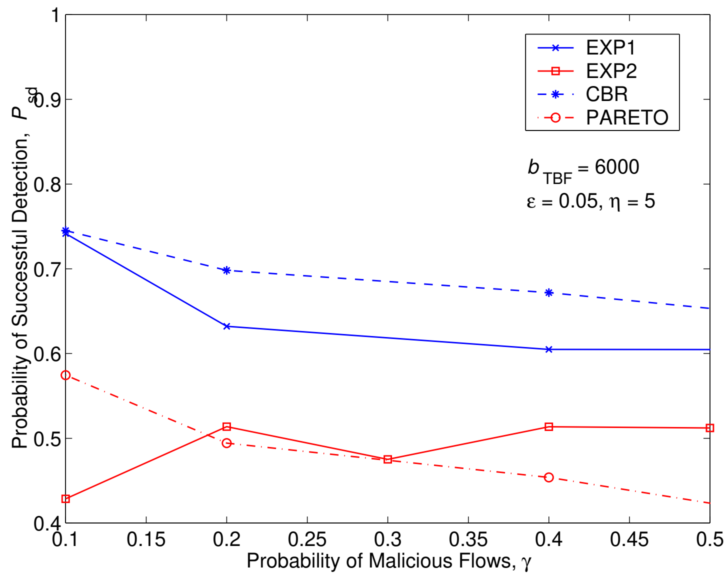


Figure 6.8: Case 4: One large flow (r_l) and many small flows (r_s); γ of small flows misbehave.

Case 3: We have so far considered only homogeneous flows with the same rates. In the next two cases, we consider an extreme case where there is one large flow with peak rate r_l and many small flows with peak rate r_s, where $r_l = 10 \times r_s$. We model the small flows the same way as in Case 1 and repeat our experiments for four different source models. In Case 3, only the large flow is misbehaving. The results are summarized in Table 6.1. For all source models, we always successfully detect the misbehaving large flow and P_{fa} is less than 0.03.

Case 4: Case 4 addresses the scenario where misbehaving small flows "hide" in the aggregate with another large flow. The probability of a small flow being malicious is γ. Intuitively, we suspect that detection is harder in this case, because the misbehaving flows can "steal" the idle bandwidth allocated to the large flow. Since the traffic policer can only enforce the total allocated rate, the malicious flows may not be detected. Figure 6.8 shows the P_{sd} achieved for different values of γ and Table 6.2 summarize P_{sd}, P_{fa} and P_{mis} for $\gamma = 0.1$ and 0.5. Surprisingly, we notice that the P_{sd} achieved with $\gamma = 0.1$ for EXP1, CBR, and PARETO sources are fairly close to the results in Case 2, where there is no large flow. But for EXP2, the success rate is significantly smaller (P_{sd}=0.43 in Case 4 as supposed to 0.54 in Case 1). When γ increases, the success rate P_{sd} decreases for EXP1, CBR and PARETO source models. With EXP2, P_{sd} fluctuates as we vary γ, and is actually higher at γ=0.5 than γ=0.1. This is because the active cycle of EXP2 is very short (10%), and can easily go undetected if it coincides with the idle period of the large flow. However, when the fraction of malicious flows increases, there is an increased likelihood that some of the malicious flows will synchronize or overlap in their active cycles, leading to overflow of the TBF at the traffic policer. When the aggregate rate is violated, all the flows sharing the same subfield ($FidIn$ or $FidEg$) will be monitored individually (micro-monitoring) and the malicious flow can be correctly identified. The probability of false alarms P_{fa}

Table 6.2: Case 4: One large flow and many small flows. γ of small flows misbehave. $\eta = 5$, $b_{\mathrm{TBF}} = 6000$, $\epsilon = 0.05$.

Source Model	EXP1	EXP2	CBR	PARETO
$\gamma = 0.1$				
P_{sd}	0.74	0.43	0.75	0.57
P_{fa}	0.00066	0.011	0.0	0.0
P_{mis}	0.0032	0.00016	0.0047	0.0028
$\gamma = 0.5$				
P_{sd}	0.61	0.51	0.67	0.39
P_{fa}	0.00067	0.025	0.0	0.0
P_{mis}	0.0030	0.00047	0.0088	0.0022

Table 6.3: Comparisons between heterogeneous and homogeneous source models: $\gamma = 0.1$, $b_{\mathrm{TBF}} = 6000, \epsilon = 0.05$.

Source Model	HET	EXP1	EXP2	CBR	PARETO
$\eta = 1$					
P_{sd}	0.55	0.79	0.73	0.83	0.64
P_{fa}	0.0	0.0	0.14	0.0025	0.0
P_{mis}	0.0030	0.0028	0.00016	0.0079	0.0031
$\eta = 5$					
P_{sd}	0.55	0.71	0.54	0.80	0.58
P_{fa}	0.0	0.0	0.060	0.0025	0/0
P_{mis}	0.0030	0.0028	0.00016	0.0079	0.0031

and mis-marked packets P_{mis} are negligible in this case across different values of γ and source models.

Further Sensitivity Analysis

So far, we have been considering flows with homogeneous source characteristics in our simulations. The next experiment uses a random mixture of the four different source models (EXP1, EXP2, CBR and PARETO), each with different peak rates, idle times and burst times. Each arriving flow chooses among these source models at random. We repeat the experiment in Case 2, with η=1 and 5 using heterogeneous flows (HET), and compare the results with Case 2 where homogeneous flows are used. Results are summarized in Table 6.3. With HET sources, the success rate P_{sd} is lower than all the other homogeneous source models, but the differences in P_{fa} and P_{mis} are negligible. It is difficult to tune the hysteresis or TBF parameters to optimize the overall performance since the source characteristics are not known a priori.

We also repeat the Case 2 experiment using the EXP1 source with the following modifications:

a. MDAP without micro-policing mode, and

b. MDAP deployed at ingress ERs only.

Results show that only 23% of malicious flows are detected in (a), and 53% in (b), which is significantly lower than 79%, when MDAP is deployed at both ingress and egress ERs (Figure 6.7).

6.5 Implementation and Prototyping

In this section, we provide a brief description of how we implement the Furies architecture on top of the Click modular router [24]. We use this implementation to evaluate certain performance metrics which could not be accurately quantified through simulations. One such important metric is the overhead incurred at an edge router by adding Furies control functionalities. The current implementation works on Linux 2.2.16 and 2.2.17 kernels.

We also built a proof-of-concept Furies prototype on Millennium[9], a powerful computational testbed in our laboratory. We verified all the features of Furies by setting up a virtual network of seven ingress-egress POPs over this testbed.

6.5.1 Overview of Implementation

Click is a Linux-based software architecture developed by Kohler, et al. at MIT [24] for building flexible and configurable routers. A Click router is assembled from packet processing modules called elements. Individual elements implement simple router functions like packet classification, queuing and scheduling. We extend the Click router to support two additional elements: the reservation agent (RA) and the monitoring agent (MA). The RA is responsible for directing flow requests to the Furies Resource Manager (RM) and forwarding responses from the RM to the client. The MA handles the traffic monitoring and policing of admitted flows, i.e., the functions of the TMs and TPs in the Furies architecture (Section 6.3.1).

The communication between the Click router and the Resource Manager is performed through SNMP. In order to enhance the throughput of the Click router, we reduce the number of context switches required for processing the control packets from the RM by batching the messages from the RM to the Click router.

6.5.2 Performance Evaluation

Using our implementation, we measure the performance overhead of adding the RA and MA in an edge router. To obtain a realistic picture of this overhead, we compare the maximum throughput obtained from our implementation to a basic implementation of Click which did not contain any of the monitoring tools (hereafter referred to as default Click). A quantification of this overhead is necessary to determine whether it is practical to deploy Furies.

[9]Millennium Project, http://www.millennium.berkeley.edu/.

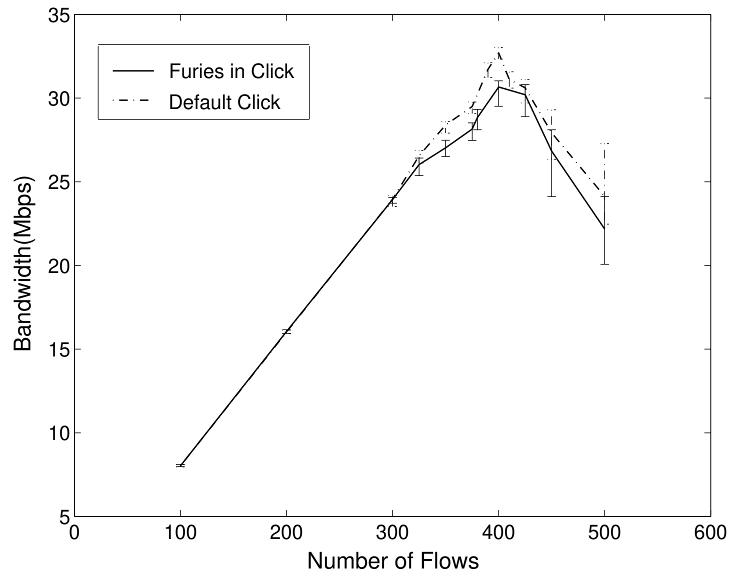


Figure 6.9: Throughput comparisons between Furies+Click and default Click.

Experimental Setup and Methodology

For evaluation purposes, we set up our own cluster of machines over a 10.2.2.0/24 network. The experimental setup consisted of a total of six Intel PCs running Linux. A Pentium-III 650Mhz machine was used as the Click router. This machine used a 3com 3c905 100 Mbps Ethernet controller and had 256 KB of L2 cache. Another machine, a Celeron-400 Mhz with a 128 KB of L2 cache was used as one of the traffic generators. The other traffic generators were Dell 6350, 4-processor 650 Mhz machines. We use two more machines as Sink and Resource Manager. All these machines are connected to a backbone router using 100 Mbps connections. The router is a Bay Networks Accelar-1100B router with the capacity to support 16 100 Mbps Ethernet ports.

To make our measurements more realistic, we used an IP routing table from a BBN planet edge router [120]. The routing table contained 43,872 entries. We disabled the IP-lookup cache, and compared the lookup time with the time taken for traffic monitoring. The traffic statistics was periodically sent to the RM every 100 ms. We modified Mgen [121], a publicly available traffic modeling software, to generate traffic for our experiments.

Experimental Results

In our first experiment, we measured the maximum throughput of our implementation at different loads and compared it to a default Click router. A basic flow in our setup has a bandwidth of 80 Kbps and a packet size of 1024 bytes. As the number of flows increases, the amount of policing and state needed at the edge router also increase.

Figure 6.9 compares the throughput of our Click implementation of Furies and the default Click. We vary the number of flows to generate varying loads. Since an ER can observe up to 500 latency-sensitive flows (Section 6.3.2), we consider a load of 100-500

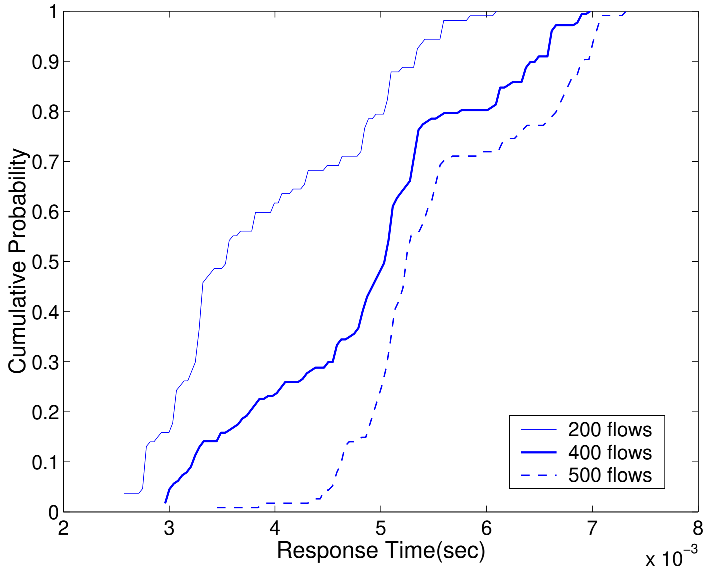


Figure 6.10: Response time for processing flow requests for varying loads.

flows[10]. From the figure, we can infer that the percentage overhead (percentage difference in throughput) is insignificant. The maximum percentage difference observed in our experiments is 5%. This occurs at a load of 400 flows. The default Click achieves a peak throughput of 32 Mbps while our implementation achieves 30.5 Mbps. At smaller loads, the overhead is negligible.

In the second experiment, we measured the response time for flow allocation at different loads. We achieved a particular load in the system by maintaining a constant number of active flows and sending dummy flow requests to the Click router from the Traffic generator. There are three stages in the process of obtaining a response for a flow request: the RA in Click forwards the request to the RM, the RM performs admission control on the flow, and the RM sends the response to the requesting entity through the RA.

In Figure 6.10, we plot the cumulative distribution of the response time at three different loads: 200 flows (small load), 400 flows (saturation point) and 500 flows (very high load). From the graph, we can observe that the mean response time increases as the load increases and the CDF shifts to the right. In all our experiments, we observe a minimum response time of 2.5 ms and a maximum of 7.2 ms. Our results indicate that the standard deviation of our response time is high. This can be attributed to the batching of responses at the RM and timer-based processing of flow requests at the Click router. However, this variance is tolerable since the mean response time is small.

6.5.3 Discussions

We found it easy to build extra router mechanisms on top of Click. Our entire implementation consists of 4463 lines of C++ code and the effort taken to integrate our code with Click was minimal. From our experiments, we observed a negligible performance

[10]Our setup did not allow us to generate beyond 500 flows.

difference between the interrupt-driven kernel mode and the interrupt-driven user mode of Click. In both cases, the processing overhead of deploying the Furies mechanisms on a Click router is minimal ($\leq 5\%$). Our implementation did not use the *Poll-device* element optimization of Click. We also used a much larger routing table than the one used in [24]. Studies in [122, 123] have shown that increasing the size of the routing table can adversely affect the throughput of the system. However, Click does not support many optimizations for fast table lookups. The performance of Click has also been optimized for DEC 21140 Tulip 64-bit PCI controllers and our setup used a 32-bit 3com Ethernet controller. Though these issues may affect the net throughput of our system, we believe that they will not change the percentage difference of the throughput.

6.6 Deployment Issues

This section describes some of the issues involved in deploying Furies in the existing Internet.

6.6.1 Distributed RM Implementation

Although the Resource Manager (RM) is described as a single logical entity within a domain (Section 6.3.1), it can be implemented as a distributed architecture across POPs. Every POP of an ISP usually has a fault monitoring facility to continuously manage the link and router status in its network. The additional functions of the RM can be implemented as additional pieces of software that reside in these monitoring facilities. For example, a *local-RM* of a POP can maintain part of the domain's database, FR (Section 6.3.1), by tracking $Fids$ where at least one of the $FidIn$ or $FidEg$ is a valid group identifier of an edge router within the same POP. For every flow that is admitted, a new entry with its Fid and allocated bandwidth is created in the partial FR databases at both its ingress and egress POPs. Similarly, when a flow stops, we remove the flow's entry from the local RMs in its ingress and egress POPs, respectively.

6.6.2 Changes to Routers

To deploy Furies, no changes are required in the core routers, while the policing and monitoring need to be added to the edge routers. From our Click implementation, we infer that the modifications needed to add the extra Furies mechanisms in an edge router is minimal.

6.6.3 Virtual Private Networks

Furies can be deployed within an ISP to support Virtual Private Network (VPN) customers. The mechanisms of Furies can be applied to provide an abstraction of a VPN overlay network by treating the VPN endpoints as ingress and egress points. We do not need to change any of the underlying admission control or traffic policing mechanisms.

6.6.4 Multiple ISPs

If indeed Furies is deployed, it will be done incrementally and not all ISPs will be Furies-enabled. A flow that passes through multiple ISPs may not receive the best performance guarantees if some intermediary ISPs do not support Furies. Furies can be extended to support a *Confederation* of ISPs, which is a group of ISPs that coordinate among each other to set up a larger virtual ISP through peering agreements. Every peering point is associated with a set of SLAs, and a flow that is admitted in one ISP can be guaranteed its level of service within the confederation. Furies also provides a way for sharing resources across ISPs and does not assume any trust relationships between ISPs.

6.7 Summary

Although detection of malicious flows has long been recognized as an important aspect of resource control, a practical and scalable way of implementing it has not been studied in great detail. This chapter proposes the Furies architecture for policing incoming flows and detecting malicious behavior without requiring per-flow state maintenance at any edge routers. By aggregating flows for group policing, Furies only requires $O(\sqrt{n})$ state maintenance at edge routers (where n is the number of flows), which is substantially better than previous approaches. Extensive simulations show that Furies-MDAP is effective and robust across a variety of source models and extreme cases. For VoIP type traffic (EXP1), MDAP can successfully detect 79% of malicious flows with zero false alarms and less than 0.1% incorrectly dropped packets. However, further study is needed to improve the detection of bursty malicious sources. Our approach has significant practical value since Furies incurs very minimal processing overhead at edge routers and can be incrementally deployed. 90% of the malicious flows are detected within 1/4 of their average life time, and the average detection time is 26.9 seconds or 1/10 of flow life time.

MDAP provides a way to select a small subset of the flows, which are most likely to be malicious, for individual sampling and further verification. Since flows are aggregated for policing in general, the processing overhead and state maintenance required are minimal. Therefore, the optimal operational range for MDAP is when only a small fraction of the total admitted flows is malicious. The advantage of MDAP over per-flow policing schemes is the most significant in this case. Otherwise, if the majority of the flows are malicious, a large subset of the flows would be subjected to individual sampling and the overhead of MDAP approaches that of per-flow policing schemes.

Tuning the parameters of MDAP involves tradeoffs among different performance indexes, e.g., frequency of false alarms vs non-detections. The optimal choice of parameters are often dependent on the operational goals and business models of the network providers.

Chapter 7

Conclusions and Future Work

This chapter concludes the dissertation by summarizing the major contributions of the thesis and suggesting some key directions for future work.

7.1 Thesis Summary

The focus of this thesis is to analyze how adaptive resource control mechanisms can be coordinated intelligently in large-scale networks to support the QoS objectives of latency sensitive applications (LSA) such as real-time voice and video streams. Our goal is to understand how to support the demanding QoS requirements as well as a stateful approach (e.g., Int-Serv), yet maintaining the scalability or efficiency found in stateless network architectures (e.g., Diff-Serv). The major contributions and results of this thesis have four core components:

- workload modeling,
- architecture,
- specific algorithms for predictive reservations, admission control and traffic policing, and
- insights gained from simulations and implementation experience,

each of which is discussed in the following sub-sections.

7.1.1 Workload Modeling

First, we need to model the characteristics and performance requirements of LSAs to drive the design and evaluation of the resource control schemes we proposed, including the predictive reservation technique, TMAC-*Traffic-Matrix based Admission Control* and MDAP-*Malicious Flow Detection via Aggregate Policing*. One common missing piece in the previous work is the measure of how well these network-level QoS control mechanisms satisfy application-level performance requirements, such as perceived audio/video quality or user experience. To bridge this gap, we focus on Voice-over-IP (VoIP) as an example workload and perform subjective experiments to quantify the impact of packet losses and delays on perceived voice quality.

From our subjective testing (Chapter 3.2.2), we found that loss rates within 1-2.5% are tolerable but received voice streams become incomprehensible when more than 4% of the packets are lost. To be conservative, we choose 1% as the maximum packet loss rate allowed in a QoS-aware architecture to deliver high-quality VoIP. A separate study based on traceroute experiments revealed that the propagation delay contributes to the largest part of the end-to-end delay, but the queuing delay is the most variable component. Therefore, we need to budget the per hop queuing delay so that the one-way transmission delay of a VoIP packet stays below 150 ms, the recommended value for an acceptable user experience [80]. Our design requires that per hop queuing delay be at most *5 ms*, and we use this upper-bound to choose an appropriate buffer size.

For performance evaluation purposes, we use VoIP as a typical LSA workload because the impact of network congestion on its perceived quality is well-understood. To capture the characteristics of the diverse LSA workloads (other than VoIP), we collected 70 packet audio traces from a wide range of multimedia applications, including audio/video conferencing and distant learning. We used these traces to generate traffic for our simulations by modeling the inter-arrival process as Poisson.

7.1.2 Clearing House Architecture

We have designed a Clearing House (CH) architecture that facilitates resource reservations across multiple routing domains. Two key ideas that contribute to the scalability of the CH to a large user base are: *hierarchy* and *aggregation*. We treat the wide-area network as a collection of smaller routing domains called a *basic domains (BD)*. For example, a basic domain (BD) can be a local POP network of an ISP, a Local Access Provider (LAP) or a campus network. Several BDs can be aggregated to form *logical domains (LDs)* based on geographical or administrative boundary. This introduces a hierarchical tree of logical domains, and a CH-node is associated with each logical domain to regulate the intra-domain aggregate reservations. As described in Chapter 4, the CH serves as a distributed resource controlling system in which the processing load and state maintenance are distributed to various nodes at different level of granularity. In our model, per-flow admission control is only performed at the ingress edge router (ER) using the admission threshold computed by the Local Clearing House (LCH). On the other hand, resource reservations are established for aggregate flows that share the same pair of ingress-egress points. These reservations are adapted dynamically based on predicted bandwidth requirement based on Gaussian approximation. The CH-nodes are also responsible for computing the traffic matrix at node or POP-level (Chapter 5.1.2), and propagating the information up the CH-tree.

7.1.3 Resource Control Mechanisms

IP network resources need to be provisioned properly to protect LSA flows, e.g., through admission control at the edge and aggregate reservations on intra- and inter-domain links. In Chapter 5 and 6, we analyzed the resource control problems posed by both scalability and performance challenges. We solved them using a combination of passive traffic monitoring, session-level control techniques and enhancements to edge router mechanisms. The following are some of our findings:

Aggregate Reservations using Gaussian Predictors In our CH approach, reservations are set up for aggregate traffic, rather than individual flows, so that no individual state maintenance is required at any routers. We exploit the observation that when many flows are aggregated, the total arrival rate can be approximated by a Gaussian distribution under the Central Limit Theorem [110]. We introduced a passive monitoring tool at edge routers to measure the aggregate mean and variance of the high-priority traffic over a measurement window, T_{mea}. A Gaussian predictor (Chapter 5.2) is then used to estimate the future bandwidth requirement based on the measured mean, variance and QoS performance goal. Our trace-based simulations show that predictive reservation technique can achieve loss rate of 0.12% with only 8% over-provisioning when $T_{\text{mea}} = 1$ minute. Gaussian predictor is robust if T_{mea} is smaller than the time-scale at which bandwidth demand varies, regardless of the number of flows being aggregated.

Traffic Matrix based Admission Control (TMAC) Our scheme, TMAC, leverages the knowledge of network-wide traffic demand distributions and topology to compute the admission threshold, $U(s,d)$, for IE-Pipes between every pair of ingress router s, and egress router d. $U(s,d)$ is computed by splitting the bandwidth on a link shared by several IE-Pipes in proportion to their aggregate traffic demand. Based on $U(s,d)$ and the measured existing load, TMAC determines if a flow should be admitted at an ingress router (Chapter 5.3). Our simulation results show that TMAC can achieve 95% utilization level with less than 1% loss and 20% statistical multiplexing when a VoIP type workload is considered.

RxW Aggregate Scheduling The bottleneck of a CH-architecture lies in processing and forwarding the inter-domain reservation requests for admission control at different levels of granularity. We use RxW aggregate scheduling (Chapter 5.5.2) to enhance the CH performance by classifying the requests that share the same path into the same queue and scheduling them as an aggregate request. RxW shows a 71% improvement over the throughput achieved using a standard First-In-First-Out (FIFO) queue. It also reduces the mean reservation set-up time by greater than 4 times at a load of 3000 requests/second. More than 80% of the individual set up time is less than 200 ms.

Malicious Flow Detection via Aggregate Policing (MDAP) We propose a scalable mechanism, MDAP, for policing and detecting malicious flows without having to keep per-flow state at any edge routers. Our approach exploits the Internet's hierarchical structure and the inter-ISP economical relationships to impose special rules to aggregate flows for policing. The first provider's network is responsible for monitoring individual flows and detecting malicious behavior. The subsequent ISPs only attempt to verify the adherence of cross-domains traffic to their respective Service-Level Agreements (SLAs). Through distributed edge coordination, the amount of states maintained by any edge router is reduced from $O(n)$ to $O(\sqrt{n})$, where n is the number of admitted flows, while core routers are stateless. The parameters of MDAP detectors control the trade-offs among different performance indexes, e.g., non-detection rate and frequency of false alarms. The optimal choice of parameters is often dependent on the operational goals of the network providers.

In this thesis, we consider preserving the QoS performance of legitimate flows to be more important than punishing the malicious flows. With this in mind, we choose the parameters to minimize the degradation of QoS performance and detect as many malicious flows as possible. Our results show that MDAP can successfully detect a majority (64-83%) of the malicious flows with almost zero false-alarms. Packet losses suffered by innocent flows due to undetected malicious activity are insignificant (0.02-0.9%). The average detection time for correctly identified malicious flow is 26.9 seconds, which is less than 1/10 of the average flow lifetime.

7.1.4 Key Design Principles and Lessons

In addition to solving specific resource allocation problems discussed above, we also gather a set of general principles and lessons through our design process, simulation and implementation experience.

Hierarchical Approach In our work, we develop the idea of *hierarchical organization with distributed control* as a design methodology by which network-wide information is abstracted and aggregated before it is exposed to different levels of control points (e.g., CH-nodes). Resource control decisions are made in a distributed manner with proper propagation of information and coordination among control points. For example, an LCH collects traffic statistics for every ingress-egress pair but only propagates pop-level traffic distribution to its parent CH-node (PCH). The LCH performs and maintains aggregate reservations within its own loca domain independently of other LCHs, while the PCH coordinates inter-domain reservations. This is a powerful technique that allows the CH architecture to scale well with respect to the growing user population and size of Internet.

Aggregation A second design principle that we use in designing many of our schemes is the notion of traffic aggregation. Although individual traffic fluctuation is hard to predict, the characteristics of aggregate traffic are reasonably well described with a small set parameters. For examples, when many flows are aggregated together, the total arrival rate approaches a Gaussian distribution that are characterized by its mean and variance. Chapter 5 discusses this principle in detail in the context of predictive reservation technique and traffic matrix estimation.

Lessons from Implementation Building a prototype of our system is useful in evaluating its practicality and deployment issues. Our TMAC and MDAP schemes require current routers to support additional functionalities, e.g., traffic monitoring and policing, processing and forwarding control messages to/from LCH-nodes. Our implementation using Click router [24] shows that the addition of these modules add minimal processing overhead to an edge router. The maximum throughput degradation is only 5%. We also found it easy to build extra router mechanisms on top of Click because of its modularity. The entire implementation consists of 4463 lines of C++ code.

From the above discussions, we concluded that the CH architecture is capable of delivering statistical end-to-end QoS assurance (e.g., $< 1\%$ loss rate and 150 ms delay) without requiring per-flow signaling or state maintenance. By leveraging the predictability of aggregate traffic, we can allocate resources more efficiently through our aggregate reservation technique. We show that TMCA can be responsive to network-wide traffic fluctuations by adjusting the admission threshold and enforcing per-flow admission control only at the ingress routers of the network. On the other hand, MDAP provides a way to select a small subset of the flows, which are most likely to be malicious, for individual sampling and further verification. Our approach has significant practical value since edge routers are required to maintain only aggregate state information and the processing overhead is minimal.

7.2 Future Directions

There are a few interesting future research directions that are either direct extensions of our work, or are motivated by the more general problem of designing next-generation Internet capable of supporting multiple QoS levels.

7.2.1 Signaling for Resource Control/Policy Distributions

In this dissertation, we focused on specific resource control mechanisms, and used the UDP protocol to exchange control messages between edge routers and CH-nodes. We have not addressed several signaling protocols that are designed specifically for carrying reservation requests or distributing policy, such as *R*esource Re*S*er*V*ation *P*rotocl (RSVP), *YE*t Another *S*ender *S*ession *I*nternet *R*eservation (YESSIR), *B*order *G*ateway *R*eservation *P*rotocol (BGRP), and *C*ommon *O*pen *P*olicy *S*ervice (COPS) protocol.

RSVP [16] is a receiver-initiated protocol designed to support end-to-end resource reservation. It uses a soft-state approach to maintain reservation status for each request on every router that lie in the path. However, it suffers from two major problems: complexity and scalability. YESSIR [124], on the other hand, attempts to reduce the processing overhead using a sender-initiated approach. It builds on top of RTCP [5], supports shared reservation and associated flow merging. Implementation results in [124] show that YESSIR reduces a session setup time by a ratio of 4.3:1. While RSVP and YESSIR support Int-Serv [13] service models that provide per-flow performance assurance, BGRP [125] relies on Diff-Serv [15] for data forwarding and hence deals with the granularity of packet classes (e.g., 32 AF/EF [17, 18] code points). BGRP builds a sink tree for each destination domain and aggregates bandwidth reservations from all data sources to that domain. Since BRGP only requires routers to maintain aggregate reservation status for each sink tree, the processing overhead scales linearly with the number of Internet domain, M. Even aggregated versions of RSVP have an overhead of $O(N^2)$, where N is the number of distinct source-destination pairs and generally much greater than M.

While we expect these techniques (RSVP, YESSIR, BGRP) to apply to the CH architecture, certain modifications are needed to accommodate the hierarchical organization of CH-nodes, e.g., the pre-segregated granularity of resource control at different CH-tree levels. Besides, it remains an open question whether the same protocol should be used to both (a) coordinate resource control decisions among CH-nodes, and (b) configure the routers and maintain reservation status. An alternative approach is to use a lightweight overlay protocol for (a) and a separate protocol such as COPS [126] to distribute resource control decisions to the affected routers. COPS is a simple query and response protocol that can be used to exchange policy information between a policy server known as Policy Decision Point (PDP) and its clients or Policy Enforcement Points (PEPs). In our model, the CH-nodes are PDPs while edge routers are PEPs.

7.2.2 Effect of Routing Instability

Throughout our work, we assume that we know the shortest paths between each pair of ingress-egress border routers, and the routes remain stable. For example, by explicitly knowing the path of different IE-Pipes, we can compute the ideal allocation of bandwidth to different ingress-egress pairs based on their estimated demand and use that as the admission threshold. This dissertation does not address the effect of dynamic routing changes on resource reservations and admission control decisions. There are two possible ways to resolve the route instability problem:

- Provide *buffer resources* by increasing the percentage of over-provisioning (for aggregate reservations) on each link, and reducing the level of statistical multiplexing (for TMAC) for each IE-Pipe.

- Design an explicit path discovery and monitoring mechanism for each IE-Pipe to detect route changes.

Each of these solutions introduces overhead in terms of idle resources or signaling overhead, and further study is needed to evaluate their performance and robustness.

7.2.3 Inter-Domain Traffic Engineering

Towards the goal of providing QoS assurance to specific traffic class (LSAs in our case), we only optimize resource allocation based on measured traffic distributions and a set of given routes. Our mechanisms do not influence the routing decision itself. To achieve a balanced link utilization level across a network, one might need to split the traffic between the same source-destination pairs among different routes. Such load-balancing operation is one aspect of traffic engineering [127], which entails design, provisioning and tuning of operational internet networks.

Until now, research in the areas of traffic engineering has primarily focused on measurement and control aspects of intra-domain routing and resource allocation. Techniques already in use or in advanced development include Asynchronous Transfer Mode (ATM) [128, 129], Frame Relay overlay models [130], Multiple Protocol Label Switching(MPLS) [131], and constraint-based routing (also known as QOS routing) [132]. Many open problems remain to be solved to extend these solutions across autonomous system boundaries, and this has been an active research area. For example, there is an on-going debate whether the current MPLS label assignment algorithm is sufficient for inter-domain traffic engineering.

Another unresolved problem is whether Border Gateway Protocol (BGP) [45] should be used for both inter-domain routing and traffic engineering. BGP was originally designed for connectivity. Although it allows an AS to enforce certain policies or to state preferred routes through the use of its different option fields, the actual operation is cumbersome and inefficient. A thorough performance study is needed to determine if extending BGP to support inter-domain traffic engineering is a scalable solution.

7.2.4 Security Issues

There are various security and privacy issues related to deploying the Clearing House architecture and mechanisms over the Internet, which are administered by multiple independent and (sometimes) competing parties. Some of these issues are explained below:

Trust Management We observe that the CH-nodes of a particular ISP may need to access monitoring points that lie in other ISPs' networks for making resource allocation decisions. This raises an important issue: how can one specify policies for controlled exposure of information, such as aggregate traffic statistics or backbone performance, without having to reveal internal network state information. For example, it might be mutually beneficial to different ISPs to route the inter-domain traffic along less congested paths, but this requires information about the neighboring domains beyond that available from end-to-end measurements.

Security Management In addition to trust, any intra- or inter-domain QoS signaling or routing should be protected against tempering, hijacking and denial-of-service attack. For example, an ISP needs a mechanism to verify the authencity and authorization of the routing or measurement information provided by other parties. Initial work in this area includes Secure Border Gateway Protocol (S-BGP) [133] and the ARQoS project [134].

Secure Billing As a resource controlling system, the CH should support billing mechanisms for settlements between peering ISPs, as well as between an ISP and its customers. It must also enable recording of billing information securely and verify the charging model, e.g., does the bill correspond to the legitimate charges invoked only by the participating ISPs? does a customer access a service from the local network points as they specify?

Privacy Management Ideally, the legitimate user should not need to reveal his/her identify to every provider in the call path simply for billing purpose: only he user's home provider would need to record the details of a call, such as the date and place of origin. Since the CH is the intermediary system in admission control, maintenance and perhaps, billing, it has a significant role to play in the overall privacy maintenance.

Secure Fid Assignments Since the MDAP mechanism (Chapter 6.3) requires assignments of Fids to admitted flows, it is possible for an external malicious user/application to create its own Fid which is valid but has not been explicitly assigned by the our system. As discussed in Chapter 6.3.6, one potential solution is to use a hash-function with shared secret key between the resource control points and every edge router. To adopt this approach, issues such as key distribution and maintenance need to be resolved.

7.3 Conclusions

From our design experience and the evaluation results discussed throughout this dissertation, we conclude that statistical techniques can be employed to provision the network resources efficiently, through reservation, admission control and traffic policing, to support latency sensitive applications without per-flow management at every router. We illustrate how these techniques work within a distributed control architecture called the Clearing House. The main strengths of our approach are scalability, support for distributed management, and robustness against traffic fluctuations. Although CH supports incremental deployment, the main operational challenge it faces is trust and policy management between different ISPs.

Our work establishes a foundation for further studies on inter-domain routing and traffic engineering issues. For example, some of our findings provide insights into making better routing decisions and negotiating more meaningful SLAs. There is also a set of interesting open security problems associated with deploying CH architecture and mechanisms.

Bibliography

[1] Network's Wizard website, `http://www.nw.com`.

[2] NUA Internet Survey website, `http://www.nua.ie/surveys/how_many_online/index.html`.

[3] V. Hardman, M. A. Sasse, M. J. Handley, and A. Watson, "Reliable Audio for Use over the Internet," *Proc. INET*, 1995.

[4] S. F. Bolot and D. Towsley, "Adaptive FEC-Based Error Control for Interactive Audio in the Internet," *Proc. IEEE Infocom*, March 1999.

[5] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *RFC 1889*, Internet Engineering Task Force, January 1996.

[6] IETF Working Groups in Transport Area, `http://www.ietf.org/html.charters/wg-dir.html`.

[7] Z. Wang and J. Crowcroft, "QoS Routing for Supporting Resource Reservation," *IEEE Journal on Selected Areas in Communications*, September 1996.

[8] G. Apostolopoulos, R. Guerin, S. Kamat, A. Orda and S. K. Tripathi, "Intra-Domain QoS Routing in IP Networks: A Feasibility and Cost/Benefit Analysis," *IEEE Special Issue on Integrated and Differentiated Services for the Internet*, September 1999.

[9] B. Gleeson, A. Lin, J. Heinanen, and G. Armitage, "A Framework for IP Based Virtual Private Networks," *Internet draft: draft-gleeson-vpn-framework-00.txt*, September 1998.

[10] D. Verma, *Supporting Service Level Agreements on IP Networks.* Macmillan Technical Publishing, 1999.

[11] E. Clark, "SLAs: In Search of the Performance Payoff," *Network Magazine*, May 1999.

[12] Integrated Services Working Group, `http://www.ietf.org/html.charters/intserv-charter.html`.

[13] S. Shenker and J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements," *RFC 2215*, Internet Engineering Task Force, September 1997.

[14] Differentiated Services Working Group,
`http://www.ietf.org/html.charters/diffserv-charter.html`.

[15] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," *RFC 2475*, Internet Engineering Task Force, December 1998.

[16] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin, "Resource ReSerVation Protocol (RSVP): Version 1 Functional Specification," *RFC 2205*, Internet Engineering Task Force, September 1997.

[17] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," *RFC 2597*, Internet Engineering Task Force, June 1999.

[18] V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB," *RFC 2598*, Internet Engineering Task Force, June 1999.

[19] Group For Advanced Information Technology (GAIT), "CA*net II Differentiated Services: Bandwidth Broker System Specification," *GAIT Internal Document*, British Columbia Institute of Technology, October 1998.
`http://www.gait.bcit.ca/Projects/bandwidth\_broker/index.html`.

[20] A. Terzis, L. Wang, J. Ogawa, and L. Zhang, "A Two-Tier Resource Management Model For The Internet," *Global Internet*, pp. 1808-17, December 1999.

[21] M. Günter and T. Braun, "Evaluation of Bandwidth Broker Signaling," *Proc. IEEE International Conference on Network Protocols*, pp. 145-52, October 1999.

[22] R. Rajan, D. Verma, S. Kamat, E. Felstaine, and S. Herzog, "A Policy Framework for Integrated and Differentiated Services in the Internet," *IEEE Network Magazine*, vol. 13, no. 5, pp. 36-41, September/October 1999.

[23] H. Zhang and D. Ferrari, "Rate-controlled service disciplines," *Journal of High Speed Networks*, vol. 3, no. 4, pp. 389-412, 1994.

[24] E. Kohler, R. Morris, B. Chen, J. Jannotti and M. F. Kaashoek, "The Click Modular Router," *ACM Transactions on Computer Systems*, vol. 18, no. 4, November 2000.

[25] R. Guerin and V. Peris, "Quality-of-Service in Packet Networks: Basic Mechanisms and Directions," *Computer Networks*, vol. 31, no. 3, pp. 169-179, February 1999.

[26] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queuing Algorithm," *Journal of Internetworking: Research and Experience*, vol. 1, pp. 3-26, January 1990.

[27] S. Floyd and V. Jacobson, "Link-Sharing and Resource Management Models for Packet Networks," *IEEE/ACM Trans. Networking*, vol. 3, no. 4, pp. 365-413, August 1993.

[28] A. Romanow and S. Floyd, "Dynamics of TCP Traffic over ATM Networks," *IEEE Journal on Selected Areas in Communications*, vol 13, no. 4, pp. 633-41, May 1995.

[29] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397-413, August 1993.

[30] S. Shenker, C. Partridge, and R. Guerin, "Specification of Guaranteed Quality of Service," *RFC 2212*, Internet Engineering Task Force, September 1997.

[31] J. Wroclawski, "Specification of the Controlled-Load Network Element Service," *RFC 2211*, Internet Engineering Task Force, September 1997.

[32] D. D. Clark, S. Shenker, and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Architecture and Mechanism," *Proc. ACM Sigcomm*, pp. 14-26, August 1992.

[33] K. Nichols, S. Blake, F. Baker and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," *RFC 2474*, Internet Engineering Task Force, December 1998.

[34] S. Brim, B. Carpenter, and F. Le Faucheur, "Per Hop Behavior Identification Codes," *RFC 2836*, Internet Engineering Task Force, May 2000.

[35] M. May, J. Bolot, A. Jean-Marie and C. Diot, "Simple Performance Models of Differentiated Services Schemes for the Internet," *Proc. IEEE Infocom*, vol.3, pp. 1385-94, March 99.

[36] L. Breslau and S. Shenker, "Is Service Priority Useful in Networks," *Proc. of ACM Sigmetrics*, pp. 66-77, June 1998.

[37] H. Naser, A. Leon-Garcia, and O. Aboul-Magd, "Voice over Differentiated Services," *Internet Draft: ¡draft-naser-voice-diffserv-eval-00.txt¿*, Internet Engineering Task Force, December 1998.

[38] K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet," *Internet Draft*, Internet Engineering Task Force, November 1997.

[39] Internet2 QoS Working Group, `http://www.internet2.edu/qos`.

[40] Ben Teitelbaum (editor), "QBone Architecture (v1.0)," *Internet2 QoS Working Group Draft*, August 1999.

[41] B. Stiller, T. Braun, M. Günter, B. Plattner, "The CATI Project: Charging and Accounting Technology for the Internet," *Multimedia Applications, Services and Techniques (ECMAST)*, vol. 1629, pp. 281-96, May 1999.

[42] Open Shortest Path First IGP (ospf), IETF Working Group, `http://www.ietf.org/html.charters/ospf-charter.html`.

[43] Routing Information Protocol (rip), IETF Working Group, `http://www.ietf.org/html.charters/rip-charter.html`.

[44] IS-IS for IP Internets (isis), IETF Working Group, `http://www.ietf.org/html.charters/isis-charter.html`.

[45] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)," *RFC 1771*, Internet Engineering Task Force, March 1995.

[46] N. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan and J. E. Van der Merwe, "A Flexible Model for Resource Management in Virtual Private Networks," *Proc. ACM Sigcomm*, pp. 95-108, September 1999.

[47] ITU-T Recommendation G.702, *Digital Hierarchy Bit Rates*, 1988.

[48] J. T. Virtamo, "A Model of Reservation Systems," *IEEE Trans. on Communications*, vol. 40, no. 1, pp. 109-18, January 1992.

[49] M. Degermark, T. Kohler, S. Pink, and O. Schelen, "Advance Reservation for Predicted Services," *Proc. 5th International Workshop NOSSDAV*, pp. 3-14, April 1995.

[50] D. Ferrari, A. Gupta, and G. Ventre, "Distributed Advance Reservation of Real-Time Connections," *Proc. 5th International Workshop NOSSDAV*, pp. 16-27, April 1995.

[51] L. C. Wolf, L. Delgrossi, R. Steinmetz, S. Schaller, and H. Wittig, "Issues of Reserving Resources in Advance," *Proc. 5th International Workshop NOSSDAV*, pp. 28-38, April 1995.

[52] O. Schelen and S. Pink, "Sharing Resources Through Advance Reservation Agents," *Proc. 5th International Workshop on Quality of Service (IWQoS)*, pp. 265-76, May 1997.

[53] O. Schelen and S. Pink, "Resource Sharing in Advance Reservation Agents," *Journal of High Speed Networks: Special Issue on Multimedia Networking*, vol. 13, no. 5, pp. 36-41, September/October 1999.

[54] I. Stoica, S. Shenker and H. Zhang, "Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks," *Proc. ACM Sigcomm*, pp. 118-130, August 1998.

[55] I. Stoica and H. Zhang, "Providing guaranteed services without per flow management," *Proc. ACM SICGOMM*, pp. 81-94, September 1999.

[56] J. Sairamesh, D. F. Ferguson and Y. Yemini, "An Approach to Pricing, Optimal Allocation and Quality of Service Provisioning in High-Speed Packet Networks," *Proc. IEEE Infocom*, vol. 3, pp. 1111-19, April 1995.

[57] N. Semret, R. R.-F. Liao, A. T. Campbell, and A. A. Lazar, "Market Pricing of Differentiated Internet Services," *Proc. IEEE/IFIP International Workshop on Quality of Service*, pp. 184-93, 1999.

[58] N. Semret, R. R.-F. Liao, A. T. Campbell, and A. A. Lazar, "Peering and Provisioning of Differentiated Internet Services," *Proc. IEEE Infocom*, vol.2, pp. 414-20, 2000.

[59] E. W. Fulp and D. S. Reeves, "Optimal Provisioning and Pricing of Internet Differentiated Services in Hierarchical Markets," submitted to International Conference on Networking, 2001.

[60] R. Edell and P. Varaiya, "Providing Internet Access: What we learn from INDEX," *IEEE Network*, vol. 13, no. 5, pp. 18-25, 1999.

[61] A. Odlyzko, "Internet Pricing and The History of Communications," to appear in *Computer Networks*.

[62] S. Jamin, P. Danzig, S. Shenker and L. Zhang, "A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks," *IEEE/ACM Trans. in Networking*, vol. 5, no.1, pp. 56-70, February 1997.

[63] S. Jamin, S. Shenker, and P. Danzig, "Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service," *Proc. IEEE Infocom*, April 1997.

[64] M. Grossglauser and D. Tse, "A Framework for Robust Measurement-Based Admission Control," *Proc. ACM Sigcomm*, vol. 27, no. 4, pp. 237-248, September 1997.

[65] E. W. Knightly and J. Qiu, "Measurement-based Admission Control with Aggregate Traffic Envelopes," *IEEE/ACM Trans. on Networking*, vol. 9, no. 2, pp. 199-210, April 2001.

[66] R. Gibbens and F. Kelly, "Distributed Connection Acceptance Control for a Connectionless network," *Proc. ITC*, vol. 2, pp. 941-52, June 1999.

[67] F. Kelly, P. Key, and S. Zachary, "Distributed Admission Control," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 2617-2628, 2000.

[68] K. Ramakrishnan and S. Floyd, "A Proposal to Add Explicit Congestion Notification(ECN) to IP," *RFC 2481*, Internet Engineering Task Force, 1999.

[69] G. Bianchi, A. Capone, and C. Petrioli, "Throughput Analysis of End-to-End Measurement-Based Admission Control in IP," *Proc. IEEE Infocom*, March 2000.

[70] V. Elek, G. Karlsson, and R. Ronngren, "Admission Control based on End-to-End Measurements," *Proc. IEEE Infocom*, March 2000.

[71] C. Centinkaya and E. Knightly, "Egress Admission Control," *Proc. IEEE Infocom*, March 2000.

[72] C. Centinkaya, Vi. Kanodia and E. W. Knightly, "Scalable Services via Egress Admission Control," *IEEE Trans. on Multimedia*, vol. 3, no. 1, pp. 69-81, March 2001.

[73] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang, "Endpoint Admission Control: Architectural Issues and Performance," *ACM Sigcomm*, pp. 57-69, October 2000.

[74] W. Feng, D. Kandlur, D. Saha, and K. Shin, "Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness," *Proc. IEEE Infocom*, April 2001.

[75] W. Feng, D. Kandlur, D. Saha and K. Shin, "Blue: An Alternative Approach To Active Queue Management," to appear in *Proc. NOSSDAV*, June 2001.

[76] ITU-T Recommendation P. 59, "Artificial Conversational Speech," 1993.

[77] A. H. Reeves, "Telecommunications of the Future with Pulse Code Modulation," *Canadian Electronics Engineering*, vol. 12, no. 12, pp. 54-6, December 1968.

[78] ITU-T Recommendation G.711, "Pulse Code Modulation (PCM) of Voice Frequencies," November 1988.

[79] ITU-T Recommendation G.721, "32 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)," November 1988.

[80] ITU-T Recommendation G. 114, "General Characteristics of International Telephone Connections nd International Telephone Circuits: One-Way Transmission Time," February 1996.

[81] G. Malkin, "Traceroute Using an IP Option," *RFC 1393*, Internet Engineering Task Force, January 1993.

[82] V. Jacobson and S. McCanne, Visual Audio Tool, Lawrence Berkeley National Laboratory - Network Research Group. Software available at `ftp://ftp.ee.lbl.gov/conferencing/vat`.

[83] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C-G. Liu, and L. Wei, "An Architecture for Wide-area Multicast Routing," *IEEE/ACM Trans. on Networking*, vol. 4, no. 2, April 1996.

[84] "CS294-7: CSCW using CSCW," U. C. Berkeley, `http://bmrc.berkeley.edu/courseware/cscw/fall97/`.

[85] ICEBERG Computer Telephony Service (CTS), `http://iceberg.cs.berkeley.edu/CTS/`.

[86] Mash System and Software, `http://www.openmash.org/users/tools/index.html`.

[87] Mash Archive System Documentation, `http://www.openmash.org/users/tools/usage/archive-usage.html`.

[88] A. Schuett, S. Raman, Y. Chawathe, S. McCanne, and R. Katz, "A Soft-state Protocol for Accessing Multimedia Archives," *Proc. 8th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, pp. 29-39, July 1998.

[89] The Network Simulator version 2 (ns-2), `http://www.isi.edu/nsnam/ns/`.

[90] W. Willinger, M. Taqqu, R. Sherman and D. Wilson, "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level," *ACM Proc. Sigcomm*, pp. 100-113, August 1995.

[91] M. Crovella and A. Bestavros, "Self-Similarity In World Wide Web Traffic: Evidence and Possible Causes," *IEEE/ACM Trans. on Networking*, vol.5, no.6, pp. 835-46, December 1997.

[92] California's Internet Interchange: Packet Clearing House, 1995. `http://www.pch.net/`.

[93] RateXChange, `http://www.ratexchange.com/`.

[94] Arbinet Global Clearing Network, `http://www.arbinet.com/`.

[95] Priceline.com, `http://travel.priceline.com/infoctr/comingsoon/welcome.asp`.

[96] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford, "NetScope: Traffic engineering for IP networks," *IEEE Network Magazine, special issue on Internet traffic engineering*, pp. 11-19, March/April 2000.

[97] C. Fraleigh, S. Moon, C. Diot, B. Lyles, and F. Tobagi, "Architecture of a Passive Monitoring System for Backbone IP Networks," Submitted for publication, October 2000.

[98] S. Bhattacharyya, C. Diot, J. Jetcheva, N. Taft, "Pop-level and access-link-level traffic dynamic in a tier-1 POP," *Proc. ACM SIGCOMM Internet Measurement Workshop (IMW)*, November 2001.

[99] R. Gummadi and R. Katz, "A lightweight secure hot billing scheme for mobile networks using a clearing house," unpublished.

[100] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz, "Characterizing the Internet hierarchy from multiple vantage points," *UC Berkeley Technical Report CSD-01-1151*, August 2001.

[101] A. Sridharan, S. Bhattacharyya, C. Diot, R. Guerin, J. Jetcheva and N. Taft, "On the Impact of Traffic Aggregation on Routing Performance," *Proc. 17th International Teletraffic Congress*, September 2001.

[102] A. Feldman, A. Greenberg, C. Lund, N. Reingold, J. Rexford and F. True, "Deriving Traffic Demands for Operational IP Networks: Methodology and Experience," *ACM Proc. Sigcomm*, pp. August 2000.

[103] Y. Paschalidis and J. N. Tsitsiklis, "Congestion-dependent pricing of network services," *Technical Report*, Systems Group, Department of Manufacturing Engineering, Boston University, 1998.

[104] Matrix Information and Directory Services Inc. (MIDS), `http://www.mids.org/weather/`.

[105] Internet Traffic Report, `http://www.internettrafficreport.com/`.

[106] Cable & Wireless USA Real Time Internet Traffic Statistics, `http://traffic.cwusa.com/`.

[107] AT&T IP Network Statistics, `http://ipnetwork.bgtmo.ip.att.net/`.

[108] S. Seshan, M. Stemm and R. H. Katz, "SPAND: shared passive network performance discovery," *Proc. of 1st Usenix Symposium on Internet Technologies and Systems*, pp. 135-46, December 1997, `http://www.cs.berkeley.edu/~stemm/spand/index.html`.

[109] D. Aksoy and M. Franklin, "Scheduling for large-scale on-demand data broadcasting," *Proc. of IEEE Infocom*, pp. 651-659, March 1998.

[110] H. Cramer, *Mathematical Methods of Statistics*, Princeton University Press, 1946

[111] Keynote, The Internet Performance Authority, `http://www.keynote.com/`.

[112] T. D. Dang, S. Molnar, and A. Vidacs, "Investigation of Fractal Properties in Data Traffic," *Journal on Communications*, vol. XLIX, pp. 12-18, November-December 1998.

[113] T. Borsos, L. Gyorfi, and A. Gyorgy, "On the Second Order Characterization in Traffic Modeling," *Proc. IFIP Working Conf. on Performance Modelling and Evaluation of ATM & IP Networks*, June 2001.

[114] P. T. Strait, *A First Course in Probability and Statistics with Applications*, Harcourt Brace Jovanovich, 1989.

[115] D. Aksoy and M. Franklin, Scheduling for Large-scale On-demand Data Broadcasting, *Proc. IEEE Infocom*, pp. 651-59, March 1998.

[116] M. Basseville and I. V. Nikiforov, *Detection of Abrupt Changes: Theory and Application*, Prentice-Hall, Inc., Englewood Cliffs, NJ, April 1993.

[117] S. McCreary and K. C. Claffy, "Trends in Wide Area IP Traffic Patterns: A View from Ames Internet Exchange," *CAIDA Technical Report*, May 2000. `http://www.caida.org/outreach/papers/AIX0005/`.

[118] S. Shenker, C. Partidge, and R. Guerin, "Specification of Guaranteed Quality of Service," *RFC 2212*, Internet Engineering Task Force, September 1997.

[119] C-N. Chuah and R. H. Katz, "Statistical Analysis of Packet Voice Traffic in Internet Multimedia Applications," unpublished, July 2000.

[120] IP Routing tables, `http://www.merit.edu/~ipma/tools/lookingglass.html`.

[121] The Naval Research Laboratory (NRL), Multi-Generator (MGEN) Toolset, `http://manimac.itd.nrl.navy.mil/MGEN/`.

[122] M. Degermark, A. Brodnix, S. Carlsson, and S. Pink, "Small forwarding tables for fast routing lookups," *Proc. ACM Sigcomm*, pp.3-14, October 1997.

[123] M. Waldvogel, G. Varghese, J. Turner, and B. Plattner, "Scalable high speed IP routing lookups," *Proc.ACM Sigcomm*, pp 25-38, October 1997.

[124] P. Pan and H. Schulzrinne, "YESSIR: A Simple Reservation Mechanism for the Internet," *Computer Communication Review*, vol. 29, no. 2, April 1999.

[125] P. Pan, E. Hahne, and H. Schulzrinne, "BGRP: A Tree-Based Aggregation Protocol for Inter-Domain Reservations," *Journal of Communications and Networks*, vol. 2, no. 2, pp. 157-67, June 2000.

[126] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry, "The COPS (Common Open Policy Service) Protocol," *RFC 2748*, Internet Engineering Task Force, January 2000.

[127] Internet Traffic Engineering Working Group, `http://www.ietf.org/html.charters/tewg-charter.html`.

[128] M. De Prycker, *Asynchronous Transfer Mode: Solution for Broadband ISDN*, Ellis Horwood, Chichester, England, 1991.

[129] B. S. Davie, "A Host-network Interface Architecture for ATM," *ACM Proc. Sigcomm*, pp. 307-16, September 1991.

[130] Frame Relay Service MIB (frnetmib) Working Group, `http://www.ietf.org/html.charters/frnetmib-charter.html`.

[131] Multiprotocol Label Switching (mpls) Working Group, `http://www.ietf.org/html.charters/mpls-charter.html`.

[132] E. Crawley, R. Nair, B. Rajagopalan, and H. sandick, "A Framework for QoS-Based Routing in the Internet." *RFC 2386*, Internet Engineering Task Force, August 1998.

[133] S. Kent, C. Lynn, and K. Seo, "Secure Border Gateway Protocol (S-BGP)," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 4, pp. 582-92, April 2000.

[134] The ARQoS Project-Protection of Network Quality of Service Against Denial of Service Attacks, `http://arqos.csc.ncsu.edu`.