RESOURCE MANAGEMENT IN THE FUTURE INTERNET

Mbalizethu Zandile Ntombifuthi Mtshali

(001114)

A dissertation submitted in fulfilment of the requirements for the degree of

Master of Science (Computer Science)

Department of Computer Science, Faculty of Science and Agriculture, University of Zululand

2005

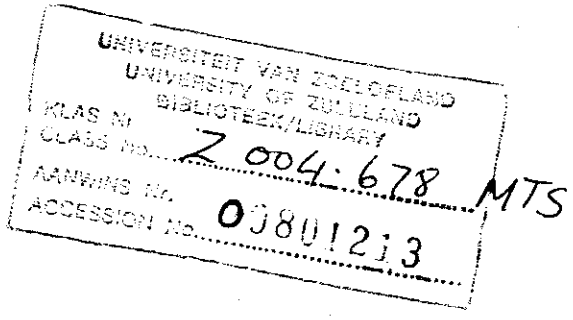
DECLARATION

I, Mbalizethu Zandile Ntombifuthi Mtshali, declare that this dissertation represents the authors' work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from published or unpublished work of others has been acknowledged in the text and a list of references is given.

_____

Signature of student

I dedicate this dissertation to my loving family. None of this would be possible without your love, support and encouragement. I am overwhelmed by your confidence in me. I love you very much.



ACKNOWLEDGMENTS

The author wishes to thank the following people for their support during the past two years of her research work:

"First and foremost, to my creator, God Almighty for giving me the strength, wisdom and perseverance to start and successfully finish my research work. He truly is my redeemer.

To one of my supervisors, Prof. Mathew Adigun, I am totally grateful to you for grooming me from the first year of my undergraduate degree up to my master's degree. You made computer science so challenging yet fascinating. Your advice, guidance and fatherly talks were always appreciated.

To Mr. George Ojong, who willingly assisted and guided me throughout my research work, you are the best mentor. Thank you for believing in me and giving me the courage to face up to my uncertainties. I am very appreciative to have had you in my life.

My colleagues and friends, I thank you all for your support and fruitful conversations we shared. I will always remember the endless hours spent in the Lab. I wish all of you the best for the future as we part ways to face the outside world.

On a final note, I would like to thank my sponsors, Telkom for providing me with the opportunity of joining one of their Centres of Excellence and funding me throughout my research project."

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The fast Internet growth has led to high demands for QoS provisioning mechanism to be implemented in the future Internet. This dissertation presents the development and implementation of a resource reservation scheme to allocate and manage resources in the future internet. Resource reservation as one of QoS techniques used for QoS provisioning in the network is crucial for many newly developed user applications to flourish and prosper.

The Scalable Efficient Reservation Protocol (SERP) is proposed to efficiently allocate and manage resources in the future Internet. The proposed scheme, Scalable Efficient Reservation Protocol (SERP), takes into consideration the fact that the future Internet will accommodate both real-time and non-real-time applications. Relevant design goals and principles were identified and these led to a formulation of a corresponding model.

A simulation of the model was conducted to evaluate the performance of the developed scheme. The experimental results do show that SERP reduces the router load with less states information to be maintained. Use of aggregate-based reservation further reduces states information. Consequently SERP improves scalability. The results further demonstrate that SERP has high request acceptance probability and more connections are established for end-to-end reservation.

CHAPTER ONE

1.INTRODUCTION

1.1. Overview

Since its inception, the Internet has become a powerful communication tool that has been adopted remarkably quickly throughout the whole world. The Internet carries data to provide access to information ubiquitously. The future Internet is still anticipated to carry far more data (for both real-time and non-real-time applications) than it does today and would do so more reliably. This growth, however, has placed high demands on the current Internet. The demand for Quality of Service (QoS) is amongst those. Provisioning of QoS is one fundamental demand that the Internet has to take care of.

The current Internet which was originally designed based on the Best Effort (BE) service model offers no QoS guarantees in transmission of real time applications. In the BE service model, packets may be dropped or discarded in case of congestion in the network.

The failure of the BE service model to provide some guarantees led to the development, by the Internet Engineering Task Force (IETF), of the Integrated Services (IntServ) and the Differentiated Services (DiffServ) models. IntServ [1] is a flow-based service model that provides hard end-to-end QoS guarantees over the Internet but is not scalable. DiffServ [2] is class-based and scalable but offers no hard QoS guarantee. Realising that these two proposed Internet architectures are not efficient in end-to-end QoS provisioning, researchers have

continued to propose other QoS models as prospective solutions. One proposal was to integrate IntServ and DiffServ ([3] and [4]). The main aim of this approach was to incorporate the best desirable features of these two architectures. Combing both approaches enables a network administrator to fine-tune the relationship between scalability and QoS to suit user demands and network capabilities [38].

In an Internet architecture, however, there must be techniques used to improve and provide QoS. Some of the techniques that can be used include admission control, traffic shaping, scheduling, and resource reservation. For any node to provide QoS, it may have to incorporate all of these techniques. This research study, however, focuses on a scheme to address reservation of network resources within the Internet.

Fundamentally, many problems we see in the Internet all come down to the issue of lack of explicit resource reservation. Packets get dropped or delayed because resources in the network are insufficient for all the traffic demands. For this reason, some method to manage the available network resources is essential. A resource management system controls access to scarce resources needed for processing especially with respect to real-time applications. It checks whether the additional service requests can be satisfied, if yes, the required resources are reserved for that application, otherwise the request is rejected [15].

A standardised resource reservation protocol proposed by the IETF is the Resource Reservation Protocol (RSVP) [9], [10], and [11]. RSVP is based on receiver orientation and offers QoS guarantees to applications. Soft state maintenance is applied in RSVP to provide robustness

and adaptability to route changes. However, RSVP lacks scalability due to its per-flow based reservations and soft state maintenance. Several RSVP extensions have been proposed to solve its drawbacks, and focus is being given to improving scalability of the reservation process at the same time reducing complexity.

Several other QoS resource reservation schemes to implement scalable solutions, have also been proposed. The design principle of these schemes is different from that of RSVP. In contrast to RSVP's receiver-oriented approach, many of these schemes implement sender-oriented reservation. Some proposed schemes use hard state maintenance to reduce the number of refresh control messages sent within the network. Other schemes proposed using data packets inline with control messages (in-band signaling) to minimise the number of control messages.

Another interesting idea is the aggregation of flows. Schemes based on this approach reduce the number of messages transmitted over the core network. In that way, overhead for processing several reservation processes is greatly reduced. Additionally, scalability within the core network is greatly improved.

Research effort is now being directed at schemes that will guarantee end-to-end QoS provisioning so as to accommodate both real time and non-real time applications. This dissertation proposes a resource reservation scheme, the Scalable and Efficient Reservation Protocol (SERP). Development of SERP emanated from the incorporation of the different design approaches proposed by several reservation schemes. However, we address the

challenges encountered in the Internet to offer scalability at the same time guaranteeing end-to-end QoS, especially for real-time applications.

1.2. Statement of the Problem

The current Internet does not guarantee QoS within the network. The Internet is required to support real-time services, which are envisaged to be the core of future network services. These types of services require that the delivery of information to an end host must satisfy certain quality requirements. However, transmission of real-time applications is still not receiving the desired QoS in terms of service delivery, delay, transmission rates and loss rate. Thus, a QoS provisioning mechanism that accommodates real-time application needs to be developed.

Resource reservation is an important mechanism for providing guaranteed services to applications. Though many resource reservation schemes have been proposed, they are faced with numerous problems such as; scalability, complexity, high processing overhead and failure to provide end to end guaranteed service. Consequently, there is still the need to propose a scalable, efficient and less complex solution to the QoS problem.

1.3. Research Motivation

Several efforts have been made to provision QoS on the Internet. QoS architectures to offer guaranteed services have been proposed. Within any QoS architecture, a resource reservation

process must be applied to reserve resources for applications prior to transmission, for QoS provisioning.

Resource reservation has received considerable attention of the research community resulting in numerous resource reservation protocols being developed. However, with the high demands anticipated for the future Internet, those schemes do not entirely meet the requirements. For instance, real-time applications require guaranteed service. However, end-to-end guaranteed service is still not provided.

Thus, this research is highly motivated by the need to provision QoS over the Internet by developing a new resource reservation protocol to offer guaranteed end-to-end QoS to current and future Internet applications.

As part of the departmental research on QoS provisioning, a novel QoS architecture is to be developed. A resource reservation protocol to be implemented within this architecture needs to be developed. Also, due to personal interest on resource reservation and in QoS, this research area was undertaken.

1.4. Research Goal and Objectives

1.4.1. Research Goal

The goal of this research is to propose an efficient resource reservation scheme, whose performance is simulated and evaluated.

1.4.2. Research Objectives

The research goal is achieved in terms of the following specific objectives:

a) To identify relevant design goals and principles for the proposed resource reservation scheme;

b) To formulate a corresponding model with control messages and protocol functionality;

c) To conduct a number of performance evaluation experiments with interpretation of the results.

1.5. Organisation of the Dissertation

The rest of the dissertation is organised as follows:

In chapter two the background concepts which form the basis of this research work is discussed. The chapter is an exposé on the current and the future Internet. Furthermore it describes the Internet Quality of Service as well as different QoS service models developed for QoS provisioning over the Internet. Special attention is given to resource reservation as one of QoS techniques being the focus of this research work. Finally, the chapter concludes with how a generic resource reservation protocol operates.

The focus of Chapter three is on resource reservation and existing related works. The chapter begins with a brief introduction on resource reservation. It is a descriptive overview of the

existing resource reservation schemes in the literature and led to presenting the design challenges associated with resource reservation. The chapter concludes with an overview of our own proposed scheme.

Chapter four describes in detail the development and simulation of the model for our proposed resource reservation scheme (SERP). The chapter begins by discussing SERP design goals and principles. This is followed by the operational functionality of SERP and concludes with a description of SERP processing in nodes.

The simulation, results, and analysis of the results are discussed in chapter five. This chapter is consequently is subdivided into two phases. The first phase, discusses the implementation. In this phase the simulation model, the simulation environment and simulation parameters are discussed. In the second phase, we present a number of performed experiments that have been used to study the performance of our proposed reservation scheme.

The conclusion of the dissertation is presented in chapter six. The chapter summarises the work conducted and described in this dissertation. Ideas for future work are also presented in this chapter.

CHAPTER TWO

2. BACKGROUND: Conceptual Definition and Terms

2.1. Introduction

This chapter gives a background of research work of the dissertation. Background concepts are explained and defined for a better insightful of the research background. Presented are concepts such as the Internet and Quality of Service (QoS), and how QoS could be provisioned and the mechanisms which could be used to improve its provisioning over the Internet. The main focus of the discussion is the resource reservation for QoS provisioning. The chapter concludes by discussing the resource reservation concepts and how a generic resource reservation protocol operates within the network.

2.2. The Internet

The Internet has become the most powerful communication architecture globally. Traditionally, the Internet was mainly used for information sharing, emails, web surfing and file transfer. However, the demands have currently increased enormously due to the increase in the number of Internet users and newly introduced applications.

The introduction of real-time applications, such as video conferencing and Internet telephony, brought to light the limitations of the current Internet being based on the Best-Effort (BE)

service model. This is due to the fact that in the BE model packets are treated the same and in case of network congestion packets may even be dropped. This Internet model occasionally loses, and re-orders packets. This creates delay and delay jitter, resulting in poor performance especially for the transmission of real-time applications, which have strict service requirements including limits on delay and packet loss rates.

Possible solutions to overcome this unsatisfactory situation include: over-provisioning the network, separate networks and Quality of Service (QoS) [6].

1) Over-provisioning of network entails increasing the network resources beyond the network capacity. This mechanism clearly leads to enormous wastage of valuable resources, consequently over-provisioning may not be the most desirable solution.

2) In separate network, each application type (one for real time and one for non-real time) is set up in separate networks so that there is no resource conflict. This mechanism, however, only solves the problem of non-real time applications interfering with real time applications. In addition, this also results in wastage of resources. Hence this mechanism is also not a desirable solution.

3) QoS provides consistent predictable data delivery service and satisfy user application requirements. Hence, QoS is the best way to handle conflict for network resources when the network is intended to service different types of traffic.

The future Internet is expected to carry a lot more data than it currently does. It is also expected to provide its users with QoS. Furthermore, the Internet will also become a seamless and ubiquitous global communication infrastructure.

2.3. Internet QoS

Quality of Service (QoS) is on the fore-front of networking technology [7]. QoS can be defined in a different number of ways. It can be defined as the set of services used to ensure that the source application and the network infrastructure have the capabilities to request, setup, and enforce the delivery of data [5]. QoS can also be defined as the capability of a network to provide better service to selected network traffic [6]. But generally, there is no common or formal definition of QoS.

Management of QoS can be viewed in a wider perspective. It can be viewed from two very different levels namely: Host QoS and Network QoS [8]. In this dissertation, the area of interest is on network level QoS.

Network QoS can be specified using the following characteristics of data transmissions: bandwidth, loss rate, delay and jitter [13]. Bandwidth represents the total amount of data sent on the network per time unit. Loss rate is the maximum number of packets that can be lost per time unit. Delay signifies the maximum amount of time that a packet will take to travel from the sending node to the receiving node. And jitter is the maximum variance in delay between successive packets.

Techniques that can be used to improve QoS as discussed in [13] include scheduling, traffic shaping, resource reservation, and admission control as illustrated in figure 2.1. Scheduling is a mechanism used to decide which packet to send first from multiple queues and to treat those packets in a fair and appropriate manner. Traffic shaping is used to control the traffic rate sent to the network. Resource reservation is used to reserve and allocate network resources before the actual transmission of data messages. And admission control is used by routers to accept or reject flows based on predefined parameters, that is, the flow's specification.

However, the focus of this research is on resource reservation. The research aim is to come up with a resource reservation mechanism that reserves resources for both real-time and non real-time applications being transmitted over the wired Internet.

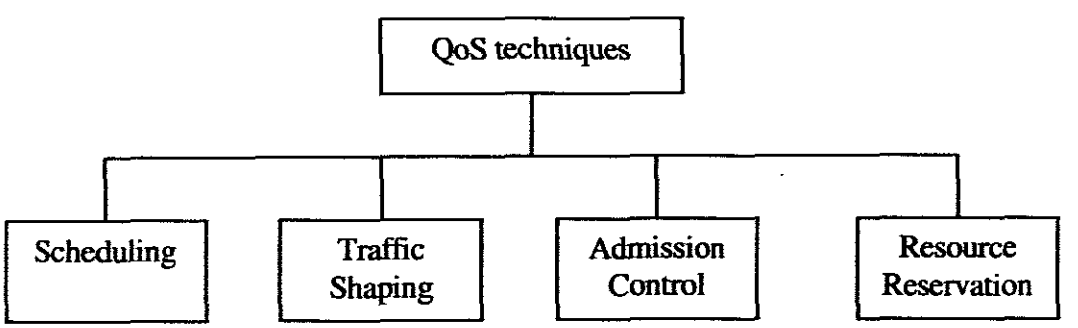


Figure 2.1 QoS provisioning techniques

2.3.1. Internet QoS Architectures

QoS architectures are designed to make network components QoS-aware. There has been several works done on designing QoS architectures. However, most current and emerging

standards are still in their early stages, but soon the rising need for QoS will propel them to the forefront.

Some of the Internet QoS service models proposed by the IETF and derivatives of them are: 1) Integrated Service (IntServ), 2) Differentiated Service (DiffServ) 3) IntServ over DiffServ and 4) Rainbow Services. IntServ and DiffServ were proposed by the IETF as QoS service models. However, these models had limitations. Realizing that these two proposed Internet architectures are not efficient in end-to-end QoS provisioning, researchers have proposed other QoS models. One of them is to integrate IntServ and DiffServ in order to incorporate the best desirable features of the two architectures. The following subsections overviews four Internet QoS service models and their relationships.

2.3.1.1. Integrated Services

Integrated Services (IntServ) [1] was proposed as an extension to the BE model to provide end-to-end QoS guarantees over the Internet. As a flow-based model, a flow is created from a source to a destination and informs interior nodes (routers) of resource requirements. Flow is defined as stream of packets with the same source address, destination address and port numbers. State information for each flow is maintained in the interior nodes. Control messages are periodically sent to refresh and update those reservation states. In addition to the BE service class, IntServ has two other service classes; Guaranteed Service (GS) and Controlled-Load (CL) service classes. GS [32] offers throughput and delay guarantees whilst CL [33] provides a

service equivalent to a best effort service in a lightly loaded network. Thus CL offers low loss, low delay and no absolute guarantees.

Even though IntServ does provide hard end-to-end QoS guarantees, it raises scalability concerns in routers since routers must maintain state information. Additionally, there are too many control messages sent into the network due to its per-flow based reservation.

2.3.1.2. Differentiated Services (DiffServ)

IETF proposed another Internet model, (DiffServ) [2], to solve the problems associated with IntServ. DiffServ maps multiple flows into few service classes to address IntServ's scalability challenge. DiffServ is a class-based model in contrast to IntServ's flow-based model. In addition to the Best Effort service class, two other service classes are defined: Assured forwarding (AF) and Expedited Forwarding (EF). AF [34] delivers packets with high assurance as long as the class traffic does not exceed traffic profile of the node. EF [35] assures low loss, low latency and ensured bandwidth. Even though DiffServ seems to be effective and scalable, its main limitation is that it does not offer hard end-to-end QoS guarantees.

2.3.1.3. IntServ over DiffServ

In [3], the authors propose a framework of IntServ over DiffServ in which end-to-end quantitative QoS is provided by applying IntServ end-to-end across a network with one or more DiffServ regions. DiffServ might not participate in end-to-end signalling. Within DiffServ regions, routers implement specific aggregate traffic control and the total amount of

traffic admitted in this region receives a certain Per-Hop Behaviour (PHB) limited by policing at the edge. The framework enables seamless inter-operation.

The main disadvantage in this framework is that edge nodes in the access network will not be aware of the traffic load in the nodes located within the DiffServ domain. Thus congestion within DiffServ region cannot be detected by these edge routers.

2.3.1.4. Rainbow Services

Rainbow Services [4] is one of the novel architectures which also adopted the approach of integration of IntServ and DiffServ. Unlike the previous architectures which adopted horizontal integration of IntServ and DiffServ, Rainbow Services is the formation of the vertical integration of IntServ and DiffServ. Proposed to provide a solid foundation for creating a robust, flexible, and scalable architecture, Rainbow Services serves a variety of real-time and non-real-time applications, both mobile and stationary.

Apart from the BE service class, four other service classes are incorporated: Guaranteed Constant Rate (GCR), Guaranteed Minimum Rate (GMR), Assured Delivery Service (ADS) and Assured Low-loss Service (ALS). In the GCR and GMR service classes, flows perform end-to-end reservations and are provided with hard guarantee while in the ADS and ALS service classes; flows perform per-hop reservations and are provided with soft or statistical guarantee [27].

The Rainbow Services has adopted the universal architecture for end-to-end QoS with a core network and access networks. The access network comprises of stationary hosts (stationery host Access Network (SH-AN)) or mobile hosts (Mobile Host Access Network (MH-AN)). The access networks are connected to core networks via edge routers [4]. Although Rainbow Services promises to improve QoS provisioning, there are still lots of issues to be dealt with. The QoS mechanisms in this architecture still need to be developed.

This dissertation documents the development of a resource reservation protocol that could be implemented in the Rainbow Services to offer end-to-end QoS.

2.4. Resource Reservation

Resource reservation enables Internet applications to obtain differing QoS for their data flows by allowing senders to reserve resources all the way towards the receivers. Resource reservation allocates resources for applications prior to data transmission and can be divided into: 1) immediate reservation and 2) advance reservation. This division is with respect to the time when the actual resource allocation occurs. In immediate reservation, users reserve and request to use resources at the same instance; while in advance reservation, resources are reserved in advance so that the desired QoS requirements could be guaranteed in advance.

A network, in its simplest form, consists of shared resources such as bandwidth and buffer, serving traffic from competing users. Lots of applications, apart from real-time applications have strict QoS requirements. Considering the variations in demand over time, unless resources

can be reserved in advance, there is no guarantee that the required resources will be available at the right time. It must be noted that QoS does not create resources but rather manages them so they can be used more efficiently to meet the wide range of application requirements. This is why advance resource reservation seems advantageous and more rational as the most constructive QoS provisioning mechanism. Additionally, using resource reservation can bring many benefits to the network. It can provide traffic protection and service guarantees to user flows inside the network so that the remotely located user populations can have acceptable and consistent communication over the Internet [30].

Furthermore, the network can never be free or unused. Network resources are scarce and if they were plentiful, then reservation of resources would not be necessary. Due to scarcity of resources, they must be reserved prior to transmission of applications. For real-time applications, advance resource reservation is very essential since these applications are sensitive to delay, jitter and bandwidth.

2.5. Generic Resource Reservation Protocol

Resource reservation can be implemented in various ways. A simplified overview of how resource reservation is carried out in the network can be described as follows:

Before a sender can start sending data packets in the network, a communication path has to be established first and to do this, resource request messages are created and sent by some sender

to a receiver via intermediate routers. At each router resources are reserved. The receiver will confirm reservation by sending a confirmation or feedback message to the sender.

Upon successful reservation indicated by a receipt of a confirmation message from the receiver, the sending host can then start data transmission immediately.

Resource reservation schemes can be classified in various categories according to the way their resource reservation is carried out. The categories include reservation orientation, reservation granularity, state maintenance, and signalling band. These categories can be described as following:

1) Reservation orientation defines the reservation initiator, their role in the traffic flow and the direction at which the reservation process is done. Thus reservation can either be sender-oriented or receiver-oriented.

2) Reservation granularity points to the format with which signaling messages are transmitted over the network. Signaling messages can be transmitted as individual flows or aggregated (multiple) flows. Thus there can either be per-flow based reservations or aggregate based reservations.

3) State maintenance indicates the approach in which reservation states are created and maintained within the routers. Reservation protocols can be divided into two categories: soft state protocols and hard-state protocols.

4) Signaling band represents the way signaling messages are isolated from the actual data messages while being transmitted over the network. Signaling messages can either be separated or piggybacked on normal data messages. Two forms of signaling exist, in-

band signaling and out-band signaling. Out-band signaling separates these messages and transmits them independently while in-band signaling sends these messages together.

CHAPTER THREE

3.LITERATURE REVIEW

3.1. Introduction

Resource reservation is a major QoS provisioning mechanism applied in QoS architectures to reserve resources for applications prior to transmission. Resource reservation has generated active research interests in the research community as a result of which numerous resource reservation protocols have been developed to offer QoS.

This chapter reviews mainly some of the existing schemes related to this work. The chapter concludes by briefly overviewing our own proposed scheme.

3.2. Existing Resource Reservation Schemes

The research study reported in this dissertation deals with resource reservation within the network and focuses on the wired network. Therefore, there is need to give an overview of the existing resource reservation schemes as classified according to different categories presented in the previous section.

3.2.1. Receiver-based schemes

In receiver-oriented reservation schemes, the receiver initiates the reservation process by making a reservation upon receiving some traffic description message from the sender

(source). An advantage of using a receiver-oriented scheme is that the receiver is provided with the capability to make reservations specifically tailored to its own QoS level requirements. Nevertheless, receiver-based reservation schemes add more complexity to the network and involves more signaling.

One popular receiver-oriented scheme is the Resource reSerVation Protocol (RSVP). RSVP [9], [10], and [11] is the standardised reservation scheme adopted by the IETF to reserve resources in the IntServ architecture [1]. RSVP is geared towards supporting real time applications by creating a flow and consequently making a resource reservation. Thus its main goal is to reserve resources so that applications can get the amount of bandwidth they need.

When reservation needs to be made with the RSVP protocol, a sender initially sends a PATH message consisting of traffic specification (TSpec) information to the destination address. Each RSVP-enabled router along the downstream route establishes a path-state that includes the previous source address of the PATH message. Upon receiving the PATH message, to make a resource reservation, the receiver sends a RESV (reservation request) message upstream towards the sender. In addition to the TSpec, the RESV message includes a request specification (RSpec) that indicates the type of services required. When each RSVP router along the upstream path receives the RESV message, it invokes an admission control process to authenticate the request and allocates the necessary resources. If the request cannot be satisfied (due to lack of resources or authentication failure), the router returns an error back to the receiver. If accepted, the router forwards the RESV message upstream to the next router. When the last router receives the RESV message and accepts the request, it sends a

confirmation message back to the receiver otherwise an error message is sent back to the receiver. The sender then starts to send data packets towards the receiver in the direction of the created communication path [10].

Although RSVP achieved the goals that it was initially intended for, the high demands caused by the Internet's growth, together with the emergence of real time multimedia applications have deteriorated RSVP's efficiency. Firstly, RSVP uses too many control messages. Scalability became a problematic issue due to its soft state maintenance of flows. Furthermore, since path messages are sent before the actual reservation process begins, the available resources may already be consumed by there reservation requests which came after the path message. Consequently RSVP is not scalable enough; it is complicated and also expensive to implement.

Another similar scheme that has been proposed is the Flow Initiation and Reservation Tree Protocol (FIRST) [18]. Just like in RSVP, reservation is receiver-oriented in order to handle heterogeneous receivers. FIRST uses a similar message to RSVP's PATH message to establish the states. A flow initiation message (similar to RSVP's RESV message) is then used to reserve resources in the routers. FIRST is also a stateful protocol but instead uses hard state to maintain the reservation information in the routers. An advantage of FIRST is that, as the flow initiation message goes towards the sender up the multicast tree, it searches every router checking if the service flow is already available. If the service flow is available at that node, the reservation stops there, otherwise it keeps going upstream until it reaches the sender. At the same time it reserves resources for the requested flow.

3.2.2. Sender-based scheme

A sender-based reservation scheme is an alternative approach to a receiver-based reservation scheme. In sender-oriented reservation schemes the sender initiates the reservation process by making a reservation request for resources in all nodes along the direction towards the receiver. Sender-oriented reservation schemes limit receivers from choosing the QoS level and also from determining the setup and teardown of reservations. Sender-orientation is carried out based on resource requirements of the application. Several sender-based reservation schemes have been proposed to overcome the limitations of the receiver-based reservation schemes. Another advantage of a sender-based reservation scheme is that the processing overhead is reduced. However sender-based approach limits receivers to have control in the reservation process.

In the work discussed in [16], the authors proposed the 'Yet another Sender Session Initiated Reservation' (YESSIR). Developed to simplify RSVP's complexity yet keeping some of RSVP's features, YESSIR was motivated by the fact that continuous multimedia applications require guaranteed QoS. YESSIR uses the Real Time Protocol (RTP) to deliver data and as a result YESSIR is built on top of Real Time Control Protocol (RTCP), a control protocol for RTP sessions. The reservation process starts when the sending host periodically multicasts RTCP sender reports (SR) to all members of the multicast group. Nodes receiving the RTCP-SR message attempt to make resource reservation according to the specification in the message. If the reservation is unsuccessful, the reason for failure is optionally attached to the SR. When the receiving host receives the SR, it attaches any failure messages to the receiver report (RR) and forwards to the sender. Upon receiving the RR with failure message, the

sender can choose to stop the session, continue the session if using partial reservation, or transmit and request fewer resources. In partial reservation, nodes which cannot make reservation will classify the flow as best effort, and forward the request without any error. Partial reservation is made with the hope that more nodes will be added, and a full reservation will eventually be the end result. As in RSVP, soft state is applied. To delete a resource reservation properly, an RTCP BYE message is used. YESSIR has its own limitations, however. The main disadvantage is that YESSIR can only be used with RTP sessions. Scalability is also an issue. There is also lack of error mechanisms. This may slow down any recoveries due to route change or reservation failures.

Another sender-oriented reservation scheme presented in [23] is the Dynamic Reservation Protocol (DRP). DRP was proposed to incorporate the principles of ATM [37] and RSVP but has its own unique feature as a dynamic sender-initiated reservation protocol. DRP's reservation process starts when the sender creates and forwards reservation messages, called RES packets, in-line with data packets. Included in the RES message is the sender's ceiling reservation type (CRT) which specifies the greatest QoS level it is willing to transmit. Upon receiving the RES packets, routers check for availability of resources. In case of reservation failure, partial reservation is allowed and the specific QoS violation bits are then set in the RES header. When the receiver receives the RES packets, it then forwards Return (RTN) packets which are reverse routed towards the sender. RTN messages also include receiver ceiling reservation type (CTRr) which is similar to CRT but specifies the greatest QoS level the receiver is willing to take. DRP seems advantageous because no prior reservation setup is

necessary, applications gain instant QoS, with the use of inline data packets. Allowing receiver heterogeneity makes DRP desirable for receivers in the same multicast group. Nevertheless, DRP has its problems such as overall scalability concerns with the use of data packets for reservation setup.

It is also worth mentioning the work in [24] and [25] where the authors proposed the Sender Oriented Signaling (SOS) protocol which is a sender-initiated, simplex, lightweight protocol for a simplified guaranteed service. A timer is used when the reservation starts and if the timer times out, a new reservation is sent. Reservation request messages are sent continuously until the sender receives an acknowledgement from the receiver. In case of reservation failure, requests are dropped. Upon the receipt of acknowledgement, a confirmation message containing the same requests rate is sent back towards the receiver. Once the confirmation message is sent, data can be sent at the newly reserved rate. As routers encounter the confirmation message, the rate is then added to the confirmed rate field. Finally, a second acknowledgement is sent to acknowledge the confirmation message. Periodic refreshes are required to be sent once per-cycle during the session. The network treats them as per-class refreshes not per-flow refreshes and the refresh rate field is updated appropriately as refreshes are passed through the nodes. To end the reservation, a teardown message is sent. While the additional resource and complexity needed by routers to maintain per-flow state have been removed, there has been the need to introduce other complex features. Two separate garbage collection schemes are needed, as well as the requirement for a router to re-mark packets in case of route changes. Also, true guaranteed service is not achievable.

3.2.3. Per-flow based scheme

Flows are the smallest meaningful unit of traffic in an IP network with guaranteed QoS [12]. Thus in resource reservation, flows (as streams of packets) are used as control or signaling messages. Per-flow reservation minimizes the processing required within the network. Although per-flow reservation makes sense for long-lasting sessions, such as video conferencing, it is not well suited for Web traffic (e.g. file transfer, emails etc.). The overheads for setting up a reservation for each session are simply too high. To support per-flow reservation, each node in a network has to implement per-flow classification and scheduling and these mechanisms may not be able to cope with a very large number of flows at high speeds. RSVP and YESSIR, discussed in the previous section, are some of the schemes which adopted the per-flow reservation approach.

3.2.4. Aggregate based scheme

In these types of schemes multiple flows are aggregated (grouped together). Aggregation of flows reduces overhead for the reservation setup process. The main goal of this approach is to improve scalability within the network. Aggregate based reservations are also advantageous because to handle many reservations (such as in a flow based reservation), a router may have to suspend routing computation and packet forwarding due to hardware and CPU constraints. And since many routers do not have enough power to sustain hundreds of thousands of reservations, this makes aggregate reservation very desirable with respect to scalability concerns. While flow aggregation allows for scalability in the routers, end-to-end QoS is not totally guaranteed.

Several schemes which have adopted this approach are the work in [18], [20], and [22]. Aggregate-Resource reservation protocol (A-RSVP) [20] which is an RSVP extension, functions very similar to RSVP but it introduces some changes to reduce RSVP's communication overhead and also to reduce scalability problems encountered by RSVP. The first change was the use of aggregate resource reservation. Calendar-array time-slots are used to record aggregated resource and A-RSVP aggregates resources of the sessions whose PHOP are the same. PHOP depicts the previous hop address of the node which delivered the message ([9]). Another change was the giving up of periodic refresh mechanism. Thus rather than periodically sending refresh messages to update the states, the messages carry "Lifetime" parameters to inform the network about the period the resource is reserved.

Additionally, A-RSVP introduced two extra control messages, PATHCHANGED (notification message of route changes) and RESVM (message to modify resource reservations). RESVM is triggered by reservation increasing, reservation decreasing actively and passively. Active reservation decreasing occurs when reservation is cancelled because of PATHCHANGED message. Passive reservation decreasing occurs when reservation is cancelled due to expiration or PATH overwritten. Though A-RSVP seems to have solved RSVP's problem it has its shortcomings. Firstly, with the introduction of extra messages it raises scalability concerns. Also, end-to-end QoS is not totally achievable.

In [22], another aggregate-based reservation scheme is proposed. The scalable resource reservation protocol (SRP) is proposed to offer scalability by aggregating flows on links in a network. Aggregation is made possible by entrusting traffic control decisions to the end

systems. Per-flow state information is only required in the network edge and not in the routers. Protocol processing is simplified by attaching control information directly to data packets. Data packets are classified by the sender, by setting the packet's flag as request, reserved, or best-effort. The reservation process is then initiated by the sender. Data packets are sent with a request flag set. Upon receiving these packets, the router performs admission control. If admission control is successful, packets are forwarded with no changes otherwise the rejected packets' flag is changed to be best effort. Routers which have accepted request packets will then reserve appropriate resources, simply through the additive properties of the aggregate reservation. The receiver estimates the rate of the reservations which had been accepted and return this value as feedback. Based on that feedback, the sender can then transmit packets at the reserved rate marked with a reserved flag. This scheme achieves significant performance improvements especially with respect to scalability concerns from the lack of per-flow state. In addition, the protocol overhead processing required is simplified by attaching necessary control information into data packets. However, a major drawback is that there are complex mechanisms put in place at the network edges since routers must trust end host with traffic control decisions. Partial reservations might lead to resource starvations when multiple sources attempt to make reservations simultaneously.

The FIRST [18] protocol (discussed in section 2.4.3.1) similar to SRP, also reserves resources based on aggregates and uses estimators.

3.2.5. Soft-state reservation

In soft state maintenance, resources are reserved for a small amount of time. Hence a flow state has a short life cycle and unless refreshed or updated, it is deleted after some period of time (the default RSVP's refresh period ≈ 30 sec). Thus, end-systems periodically transmit messages into the network to renew the reservations otherwise resources are released after timeout period. Soft state deals easily with the following conditions: changes in routes, loss of control messages, node failures, reclamation of obsolete resources, and dynamic membership of multicast groups [19]. And for this reason, many of resource reservation protocols adopted this scheme [9] and [16]. However, although soft state does improve QoS, it causes scalability problems within the network due to periodic refreshes required to maintain states.

3.2.6. Hard-state reservation

Hard state maintenance is a converse approach to soft state maintenance [29]. States remain installed in the nodes unless explicitly removed by the receipt of a teardown message from the end system. Since management of hard state session is completely deterministic, the hard state setup approach must be reliable, with acknowledgements and retransmissions [17]. The hard state maintenance minimizes processing overhead at the routers. However, in a case where a path is broken, nodes may fail to release resources if nodes on path are unreachable. Schemes which adopted this approach include [9] and [18].

3.2.7. Out-band signaling

Out-band signaling separates control messages from data messages. Control information is sent separately from data packets. This approach alleviates the burden on routers. Control packets will request resources prior to transmission of data. In this way, there would not be much processing done in the router except to check if data packets correspond to the reservation states assigned to them.

However, this approach introduces scalability concerns since the number of signaling messages sent into the network increases. Schemes that implemented this approach include [9], [16], [18], and [20].

3.2.8. In-band signaling

Signaling messages are piggybacked in normal data packets, meaning that control information is sent with data packets. This minimizes the number of control or signaling messages sent within the network. In-band signaling improves scalability since there is minimal number of messages. However, sending data packets with control messages can create unnecessary burden on the routers if there are not sufficient resources to accommodate these packets. Additionally, processing of unreserved data packets can increase protocol processing time and this may constitute unnecessary delay. [16], [22], and [23] implemented this approach.

3.3. Resource Reservation Design Challenges

Reservation of resources is carried out according to each reservation protocol specification and design principles. The resource reservation schemes reviewed in section 3.2, in general experience the following design challenges: 1) scalability, 2) complexity, 3) high processing overhead, and 4) failure to provide 'reasonable' end-to-end guaranteed service. Each of these design challenges is discussed below:

3.3.1.1. Scalability

Reservation signaling has posed scalability concerns in the network, especially in the core network, which is frequently congested with thousands of received flows. The core network receives lots of signaling messages to process. Processing of these messages is not only time consuming but also creates a huge burden on the core network routers. This may result in the low utilisation of the network resources unless the network scales up.

3.3.1.2. Complexity

A lot of resource reservation protocols are complex. Complexity in reservation schemes is associated with the number of signaling messages a router has to process in order to complete one reservation session. The number of tasks required to be scheduled during the processing also contributes to complexity issues. The challenge is to reduce both the task burden as well as volume of message which a router must handle.

3.3.1.3. High protocol overhead

As flows are received by nodes, the resource reservation protocol implemented in a node processes each flow. Protocol processing overhead of a reservation protocol can be determined by factors such as: number of control messages sent, the sizes of these control messages and the refresh frequencies of control messages [9]. Designing a low protocol overhead reservation scheme becomes a challenge.

3.3.1.4. Failure to provide end-to-end guaranteed service

For real-time applications to be transmitted over the network, end-to-end QoS guarantee is required. Although many of the previously proposed resource reservation schemes proposed to offer end-to-end QoS, real end-to-end QoS has still not been achieved. End-to-end QoS is not achieved since end-to-end reservation is not easily realised in resource reservation and this is due to reservation failures and reservation timeouts. Thus designing a scheme to offer real end-to-end QoS guarantee is still a challenge.

3.4. Proposed Scheme

The previous section presented a number of resource reservation schemes that have been previously proposed to provide QoS guarantees in the Internet. These proposed schemes, however, experience some challenges one way or the other. In general, some of these schemes are not scalable, and some are complex. Some offer guaranteed QoS to user applications at an unreasonable rate.

We propose a novel protocol known as Scalable Efficient Reservation Protocol (SERP). The proposed scheme, SERP, aims at offering end-to-end QoS guarantees to user applications, both real-time and non real-time. SERP offers to address resource reservation challenges experienced in the Internet during the resource reservation process. Scalability as one of the challenges can be addressed by eliminating refreshing messages for state maintenance. Hard state maintenance is adopted instead. In this approach the number of control/signalling messages within the network are reduced. To further improve scalability, aggregate-based reservation is adopted in the core network to reduce the router load due to reservation states. Aggregate-based reservation is applied to flows heading towards the same destination network.

To efficiently allocate resources without wasting time and resources, the reservation is initiated by the sender. Sender-oriented reservations perform optimally since a resource request message from a sender is most likely to be successful in terms of consuming free resources compared to the one from the receiver. A receiver-based reservation can be too late to consume the might have been previously available resources by the time the receiver sends a reservation request message.

This research study assumes that true end-to-end QoS guarantee is still not offered in the Internet. SERP has been proposed to render end-to-end QoS guarantee by combing the following three special characteristics:

a) End-to-end resource reservation starting from the sending host to the receiving host;

b) Ability to satisfy the QoS level requirements of heterogeneous receivers;

c) Resource reservation is both sender and receiver oriented.

It should be noted that in this research study, we only considered the wired network whose nodes are stationary and interconnected by wired links.

CHAPTER FOUR

4. MODEL DEVELOPMENT

4.1. Introduction

In the previous chapter we presented a number of existing resource reservation schemes that were previously proposed to provide QoS guarantees in the Internet. These proposed schemes, however, have their limitations. In general, these schemes are not scalable, and some are complex. They offer guaranteed QoS to user applications at an unreasonable rate.

To address these limitations we propose Scalable Efficient Reservation Protocol (SERP). The proposed scheme, SERP, aims at offering end-to-end QoS guarantees to user applications, both real-time and non real-time applications. SERP offers scalability, complexity is however not addressed.

This chapter presents the model development activities that resulted in SERP. Section 4.2 covers the design goal and principles; SERP component and building blocks are described in section 4.3. The chapter concludes with a discussion of SERP processing modules in section 4.4.

4.2. SERP Design Goals and Principles

RSVP as a standard resource reservation protocol offers hard end-to-end QoS guarantees. However, there has been different reservation schemes developed to improve on RSVP. Several RSVP extensions, [19], [20], and [21] have been introduced to improve on RSVP's limitations. Although some of RSVP's limitations were solved, the proposed extensions added their own complex design issues. Our own design uses the features of the generic resource reservation protocol. We then incorporated new dynamic features that make our proposed reservation protocol scalable and efficient.

The design goals are to: 1) provide end-to-end reservation 2) accommodate receiver heterogeneity 3) provide scalability and minimal complexity 4) separate control packets from data packets and 5) provide low protocol overhead. A brief description for each of the just mentioned design goals is given below.

4.2.1. End-to-end reservation

A resource reservation protocol should support end-to-end QoS guarantees for a range of QoS sensitive applications, such as real-time multimedia applications. These applications are very sensitive to delay but can be tolerant to loss. Therefore, a resource reservation protocol should reserve resources from one end (the sending host) to the other end (the receiving host).

To achieve this, the sending host initiates the reservation process by sending a resource request message to the receiving host via routers. At the routers, admission control is performed and if

successful resources can then be reserved. This process proceeds until the request message reaches the receiver. Upon receiving the request message, the receiver would send a reply message to the sender as feedback or confirmation of reservation. In this way an end-to-end reservation is established and senders can start sending data packets via the reserved nodes, towards the receiver.

4.2.2. Sender-Receiver-oriented reservation

Existing schemes have adopted either sender-based or receiver-based reservation. RSVP-Lite [16], however, proposed no restriction on sender- or receiver- based reservation for its reservation process. This was proposed due to the fact that RSVP-Lite only accommodates unicast reservations and no merging is necessary from the receiver to the sender as in RSVP. It is up to upper level signaling protocols to decide which way to initiate.

However, in SERP we adopt a combined sender-receiver-based reservation. This allows senders to initiate the reservation process by instantly sending a resource request message downstream towards the receiver. Sender-based reservation allows a resource request to immediately consume available resources in the nodes they are making the request. While in receiver-based reservation, the process is delayed since nodes do not reserve resources until they receive a request message from the receiver. This may take considerable time and the resources that were previously available might be consumed by other resource requests by the time a request comes from the receiver.

SERP also allows receiver-based reservation in order to handle heterogeneous receivers; allowing receivers to modify the request message by specifying their own QoS level requirements. The modified request message from the receiver is sent upstream to the point where the receiver's QoS level is satisfied. Multicast reservations are accommodated.

4.2.3. Out-band signaling

Control messages are essential in setting up the reservation process. The use of data packets for reservation setup leads to complexity of the reservation process, and also amounts to high processing burden at routers. Additionally, there must be strict reliable process delivery mechanism to prevent loss of data in case of packet losses. If control messages are sent as separate packets, processing of flows is simpler and less complex than when the control messages are incorporated in data packets. It is for these reasons that we adopt out-band signaling whereby control messages are separated from data messages. SERP control messages include Request, Reply, Error and Tear messages. A description of each of these messages is given in section 4.3.1.

4.2.4. Hard state maintenance

As reservation messages travel downstream from the sender to the receiver, they leave behind reservation states. In soft state maintenance, these states are refreshed periodically otherwise they would be deleted after they timed-out. Soft state maintenance provides robustness but creates scalability problems in the core network due to frequent processing of refresh signaling messages. Often timers are used to keep track of the refresh period's expirations for the

reservation session. Maintenance of such timers and actions due to expirations of such timers also contribute to the router complexity and inefficiency.

Hence we adopt hard state maintenance. In hard state maintenance, states are maintained until they are explicitly removed. There is no need to periodically send refresh messages to maintain reservation states. This reduces the number of messages sent to the network. Although soft state maintenance deals very well with temporary node failures by recovering new route, in hard state maintenance, a "fast reroute mechanism", such as [32], can be applied to deal with such a scenario.

4.2.5. Aggregate-based reservations

Scalability is a major concern for QoS provisioning in the core of the Internet. The more scalable the network, the easier the network can be expanded. Therefore, we have decided to use the aggregation of reservations. Multiple aggregated flows are easier to maintain since aggregating reservations reduces the memory required to store the control state information. Furthermore, processing individual reservation requests on a per-flow basis may place a burden on routers, especially on core routers that receive thousands of such requests simultaneously.

4.3. SERP Components and Building Blocks

4.3.1. Control Messages

The basic protocol control (signalling) messages are REQUEST, REPLY, ERROR and TEAR as described in next subsections. The control messages implement the message format as discussed in section 4.3.1.5.

Figure 4.1 illustrates the direction of the control messages within the Internet with downstream (forward) direction being from sender to receiver whilst upstream (backward) direction is from the receiver to the sender. It is to be noted that the illustration is that of a sender-based reservation.

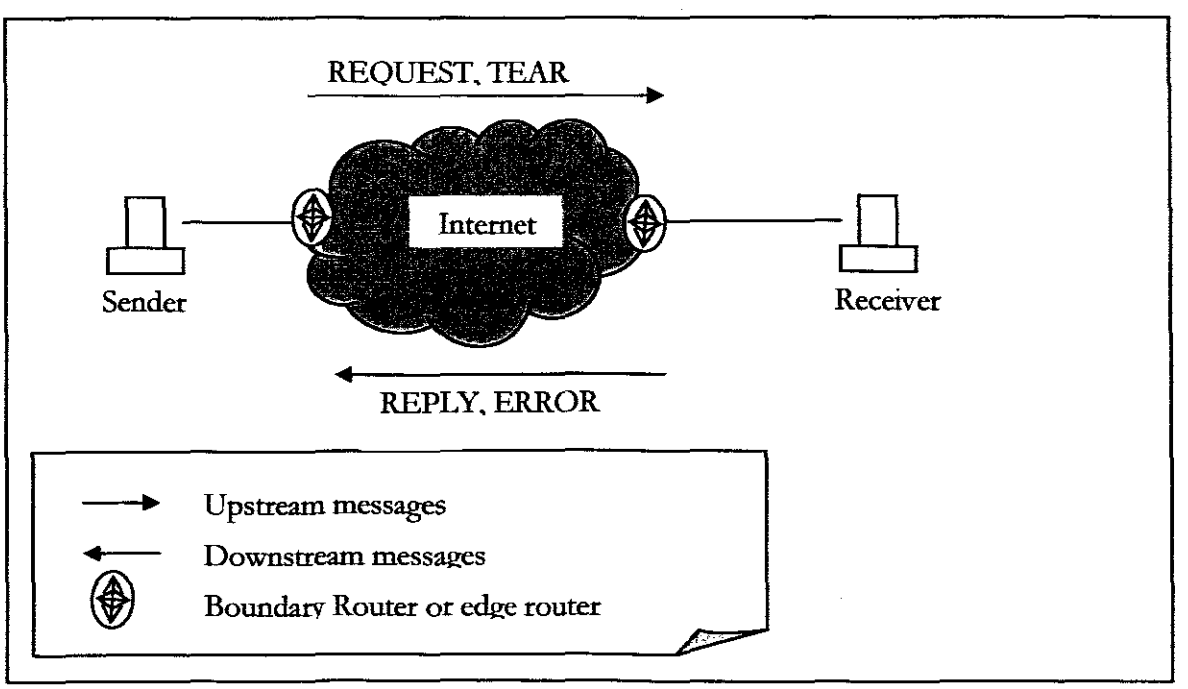


Figure 4.1 SERP control messages

4.3.1.1. REQUEST message

This is a reservation request message sent downstream by the sender (sender-based reservation) or upstream by the receiver (receiver-based reservation) to request reservation of resources. It includes QoS level requirement of a requesting node. This specifies the amount of resources a node would require to transmit its data.

Each Request message includes the amount of resources required to be reserved, source and destination address. The source address determines the point where the reservation starts and the destination address the point where it should end.

REQUEST message common part

- Session – contains the IP destination address and destination port to define a specific session of other flow that follow.
- Phop – carries the IP address of the last SERP node that sent this message. It is used to ensure correct reverse routing of upstream messages such as ERROR and REPLY messages.
- Sender Template – contains a sender IP address and optionally the sender port.
- RSpec – defines the amount of QoS resources desired, QoS level requirements specification.

- Service type – indicates a service type specified by the flow. If the specified value is "0000" service type is then BE, else if "0001" GCR is required, if "0010" service type is GMR.

4.3.1.2. REPLY message

This message is sent by nodes to deliver feedback information regarding the success or failure of the reservation request process. A reply is either a reject or an approval and it is based on the admission control decision.

The reply message would most often, in a sender-based reservation, be generated by the receiver and sent upstream to the sender. A router receiving a reply message need not process it but would transmit it upstream until the reply reaches the sender. The reply message may also originate from a router that acts as the temporary termination point for receiver reservation. In this case, the router sends the reply to the receiver.

REPLY message common part

- Session – as for REQUEST message
- Phop – as for REQUEST message.
- Reservation Confirmation – specifies the confirmation of the reservation request. The confirmation is either a reject or acceptance.

4.3.1.3. ERROR message

This message is sent by nodes to report error information about the failure of the reservation request process. A node sends an error message upstream to the previous node from which the request message was received. The error message ripples down until it reaches the sending host.

ERROR message common part

- Session – as for REQUEST message
- Phop – as for REQUEST message.
- ErrorSpec – specifies the error and includes IP address of the node that detected the error or where the error occurred.

4.3.1.4. TEAR message

The TEAR message is used to explicitly release resources at the reserved nodes without having to wait for the lifetime of the flow to expire. As nodes receive tear message, they check if a reservation state for the flow already exists, if it does then the reservation is deleted and resources are released for other flows to utilise.

TEAR message common part

- Session – as for REQUEST message

- Phop – as for REQUEST message.

4.3.1.5. SERP Message Format

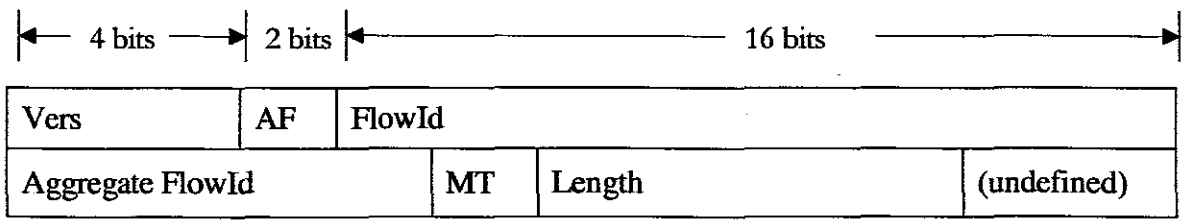


Figure 4.2 Common header

For experimental purposes, a SERP message consists of a common header followed by a common part consisting of objects. Some of the common header fields are as illustrated in figure 4.2. A brief description of each field in the common header follows:

Vers – a 4 bit field indicating the protocol version number

Aggregate Flag - a 2-bit field indicating whether the flow is part of an aggregated flow or not. If it is, the value in this field is set to 1 otherwise it is set to 0.

FlowId - a 16-bit field indicating the flow number. Each flow id is uniquely identified.

Aggregate FlowId (AF) - an 8-bit field indicating the unique flow number for aggregated flows. Each aggregate flow id is uniquely identified.

Message Type (MT) - a 2-bit field indicating the type of message. Field value "00" indicates a REQUEST, "01" the REPLY, "10" TEAR and "11" ERROR message.

Length – a 4-bit field indicating the total length in bytes of this SERP message, including the common header and the common part of objects that follow.

4.3.2. Aggregation

If end-to-end SERP reservation is carried out for each application's flows, routers in the core network may potentially need to process SERP messages for thousands of flows, and maintain state information for each of them. Thus, in the core network, aggregate-based reservation is carried out, whilst in the access network per-flow reservation takes place. Flow aggregation and de-aggregation is performed by edge routers which connect the access network and the core network.

When a node sends a request message, the destination address is used to determine the network to which the message is directed to. For uniform reservation, only flows which have the same destination network address and same service type are aggregated. The edge router does not consider where the request message originates from, as long as the destination network address matches the reservation. This allows request flows from different sources to be aggregated.

As illustrated in figure 4.3, when flows enter the core network at the edge router (ingress router), flows are aggregated by summing up the amount of resources requested by each flow. Within the core network these flows are sent as aggregate. And as flows leave the core network, the downstream edge router (egress router) de-aggregates flows back into their original individual flows.

In the request flow, an AggregateFlag is used to indicate whether the flow is part of an aggregated flow or not. If it is, the flag is set to '01' otherwise it is set to '00'. An AggregateFlowID is also defined to uniquely identify aggregate flows. This is used by the edge router at the destination network to de-aggregate flows into their original individual flows.

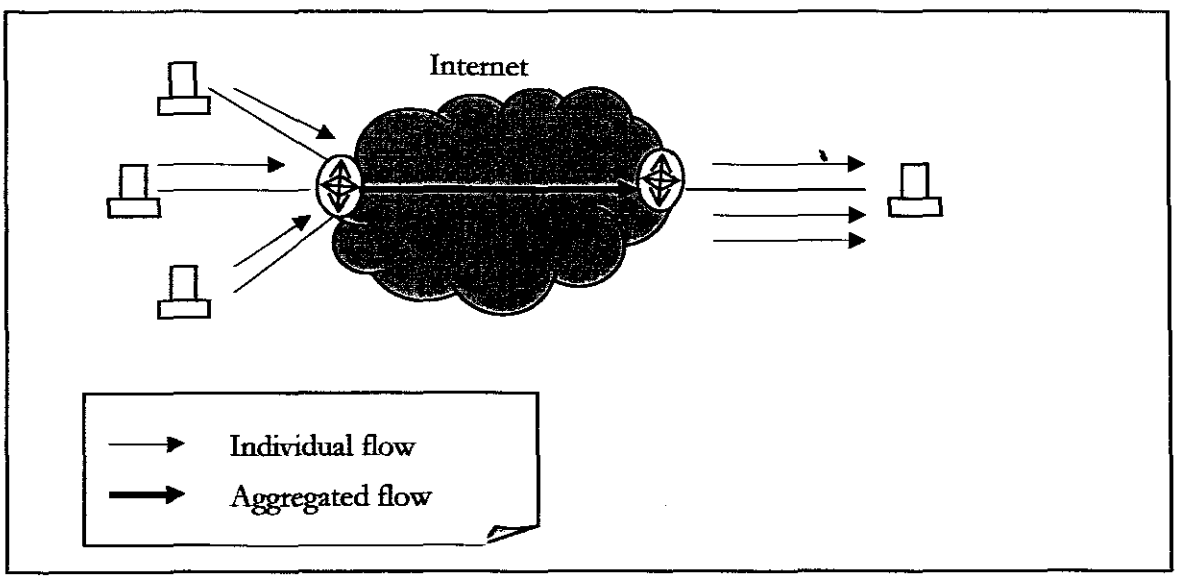


Figure 4.3 Per-flow and Aggregate Reservation

4.3.3. Protocol Functionality

To send a particular application, a host invokes a resource reservation process to request resources before the actual data transfer. The SERP reservation process communicates with two local decision modules, admission control and policy control (figure 4.4). Policy control determines whether a node has administrative permission to make the reservation. We assume that all nodes do, and thus policy control decision module is neglected. The admission control

checks for the availability of resources. If both processes succeed, the reservation module then communicates with packet classifier and packet scheduler to transmit the messages.

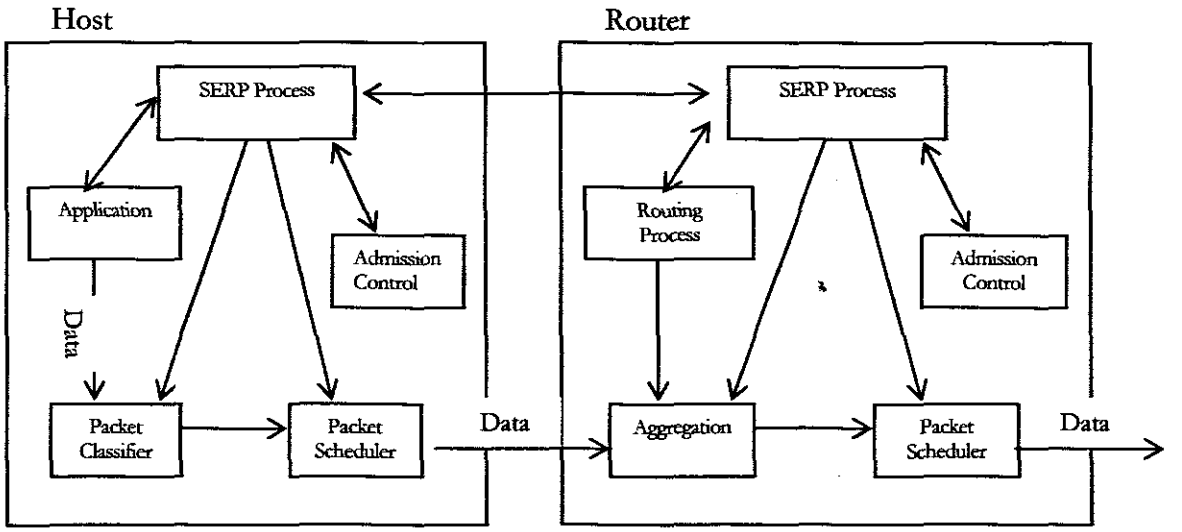


Figure 4.4 SERP Functional Diagram

The SERP reservation protocol module in the host then communicates with the one in the router. At the router, similar processes occur, except that the routing process is contacted for routing decisions. No specific routing protocol is designed for SERP; it makes use of existing ones provided by the IP network. If both processes fail, the request is rejected and an error message is then sent to the sender.

At the host node, packets are classified according to different service types and then submitted to the packet scheduler which queues the packets for transmission to the next node. At the router, the router just performs aggregation based on the type of service and their destination address. After aggregation, packets are then forwarded to the packet scheduler.

4.3.4. Operation in Nodes

4.3.4.1. REQUEST message processing

When a router receives a flow's request, it checks whether a reservation for that flow has not been previously established. For a flow whose reservation has been established, router neglects the request. Otherwise, for a request flow whose reservation has not been established, the router processes the request as illustrated in figure 4.5.

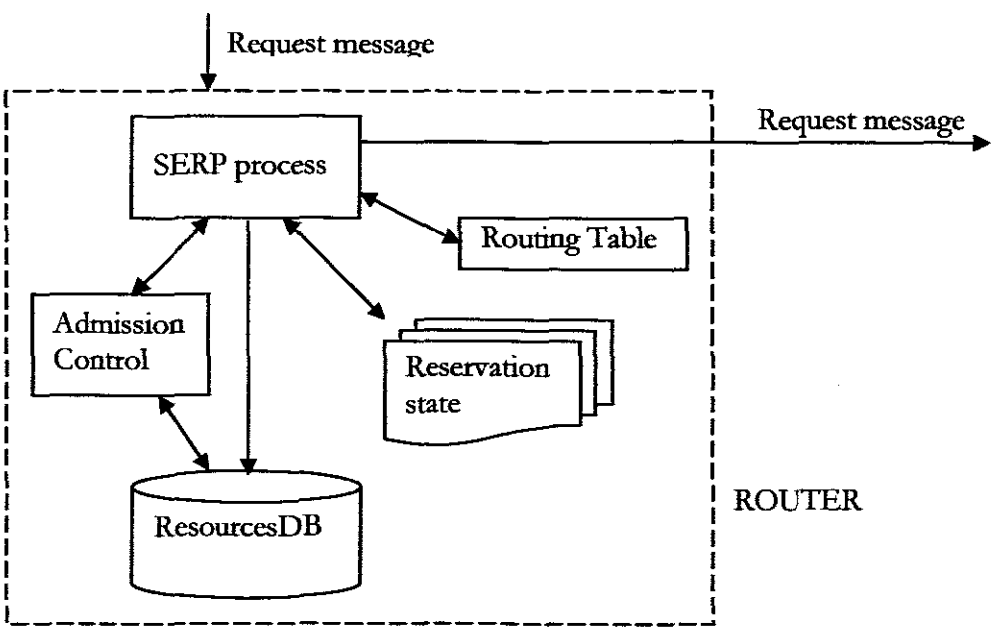


Figure 4.5 Router Processing of REQUEST message

Upon receipt of a new REQUEST flow, the SERP process invokes admission control which then determines the availability of resources from the resources database (ResourcesDB). In every node, the ResourcesDB stores the total bandwidth and the used bandwidth.

If sufficient resources are available in the node, resources are reserved, the ResourcesDB is updated accordingly and the reservation state is then recorded. The reservation state records the request message with it corresponding objects. The SERP process then checks the routing table for the next hop address of the next node to send to. The REQUEST message is then forwarded to the next node. Figure 4.6 illustrates the sequence of a REQUEST message as it is forwarded from one node to another.

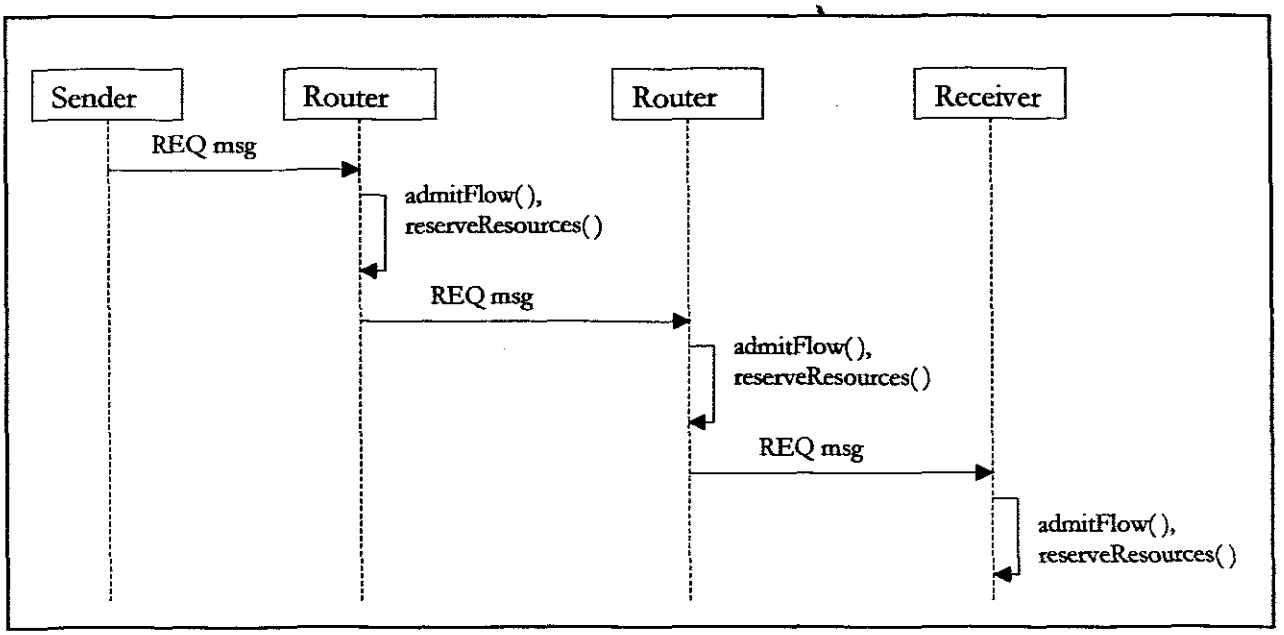


Figure 4.6 Sequence diagram for REQUEST message processing

4.3.4.2. TEAR message processing

When a router receives a TEAR message the SERP process checks if a reservation state for the specified flow is active. If found the reservation state is deactivated and the ResourcesDB is updated such that the resources that were reserved for that flow are freed

In a case where the specified flow is part of an aggregate flow, the reservation state is updated. Let the total amount of resources reserved for an aggregate flow be R. And let r_i be the flow rate requested by flow i. Then

$$R = \sum_{i=1}^{n} r_i \qquad (1)$$

Where n is the initial number of flows aggregated into a single flow and $i \in \{1, 2, 3, \ldots, n\}$.

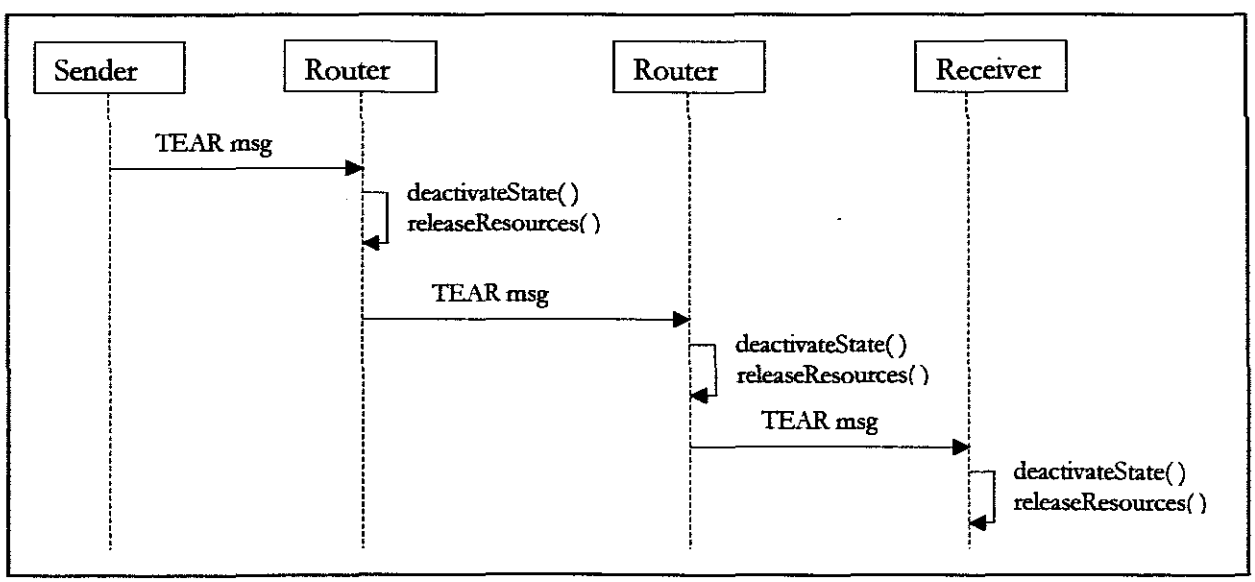


Figure 4.7 Sequence diagram for TEAR message processing

The minimum number of aggregate flow is 1. After resources are released for one flow, the new aggregated flow is reduced and it becomes:

$$R = \sum_{i=1}^{n-1} r_i \qquad (2)$$

The sequence of the TEAR message as it is forwarded from one node to another is illustrated in figure 4.7.

4.3.4.3. ERROR message processing

When an ERROR message arrives at the router, the router forwards the message upstream to an upward node. The ERROR ripples up until it reaches the sender.

ERROR messages do not modify any state maintained in nodes. They are only reported to sender application. Figure 4.8 illustrated the sequence of the ERROR message as it is forwarded from one node to another. In the figure, ERROR msg-1 represents an error message generated by the first router, ERROR msg-2 by the second router, and ERROR msg-3 by the receiver.

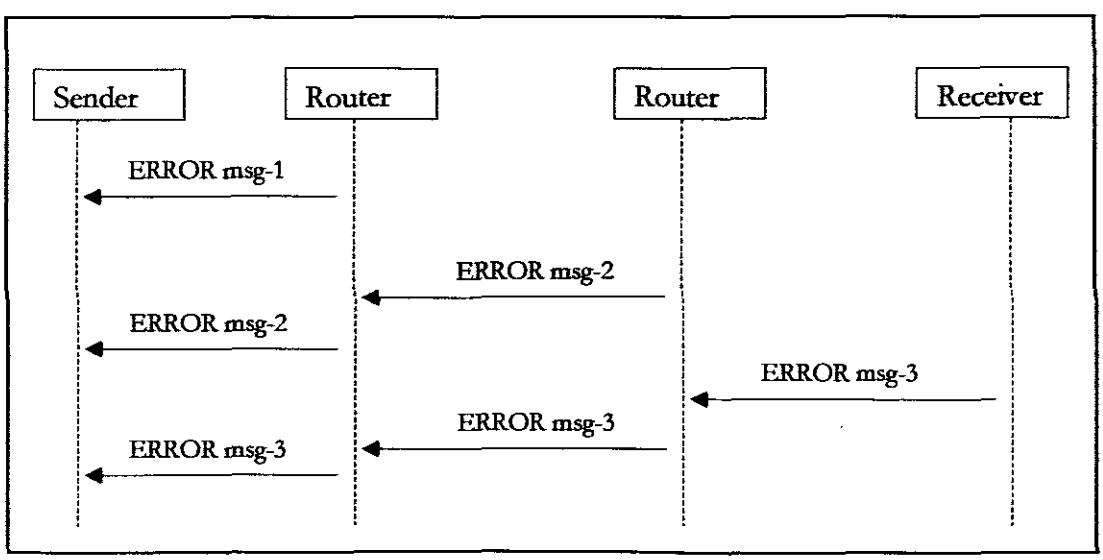


Figure 4.8 Sequence diagram for ERROR message processing

4.3.4.4. REPLY message processing

When a router receives a REPLY message, no processing is done except that the REPLY message is forwarded to the next upstream router until it reaches the sender of the request. The sequence of a REPLY message as it is forwarded from one node to another is illustrated in figure 4.9.

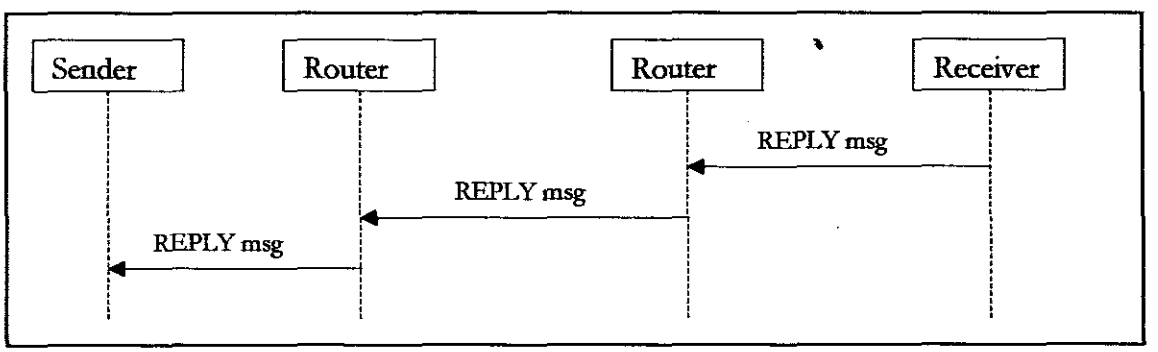


Figure 4.9 Sequence diagram for REPLY message processing

4.3.5. SERP Message Sequence Diagram

The interaction of SERP messages is represented in figure 4.10. The sequence diagram depicts the sequence of actions taken by the various network entities in forwarding SERP control messages and data packets.

The process is initiated by the sending host which creates and forwards the Request message (Req msg.) to the next intermediate router. Assuming that the next intermediate router is edge router, the edge router creates an aggregate message (Aggr-Req msg.) by setting the aggregation flag. The aggregate request message is forwarded downstream through core routers

until it reaches the other edge router of the receiver network. This edge router de-aggregates the aggregated request message back to the Request message by setting aggregation flag.

The admission control which is followed by resource reservation is performed in routers and receivers. Upon success of both processes, resources are reserved and the Request message is passed to the next node. However, at the receiver, a Reply message is created and sent to the egress router. This then sends an aggregated reply message to the ingress edge router which then sends a reply message to the sender. Sender then starts sending data packets along the reserved nodes. To end the communication, the sender creates and forwards a TEAR message downstream towards the receiver.

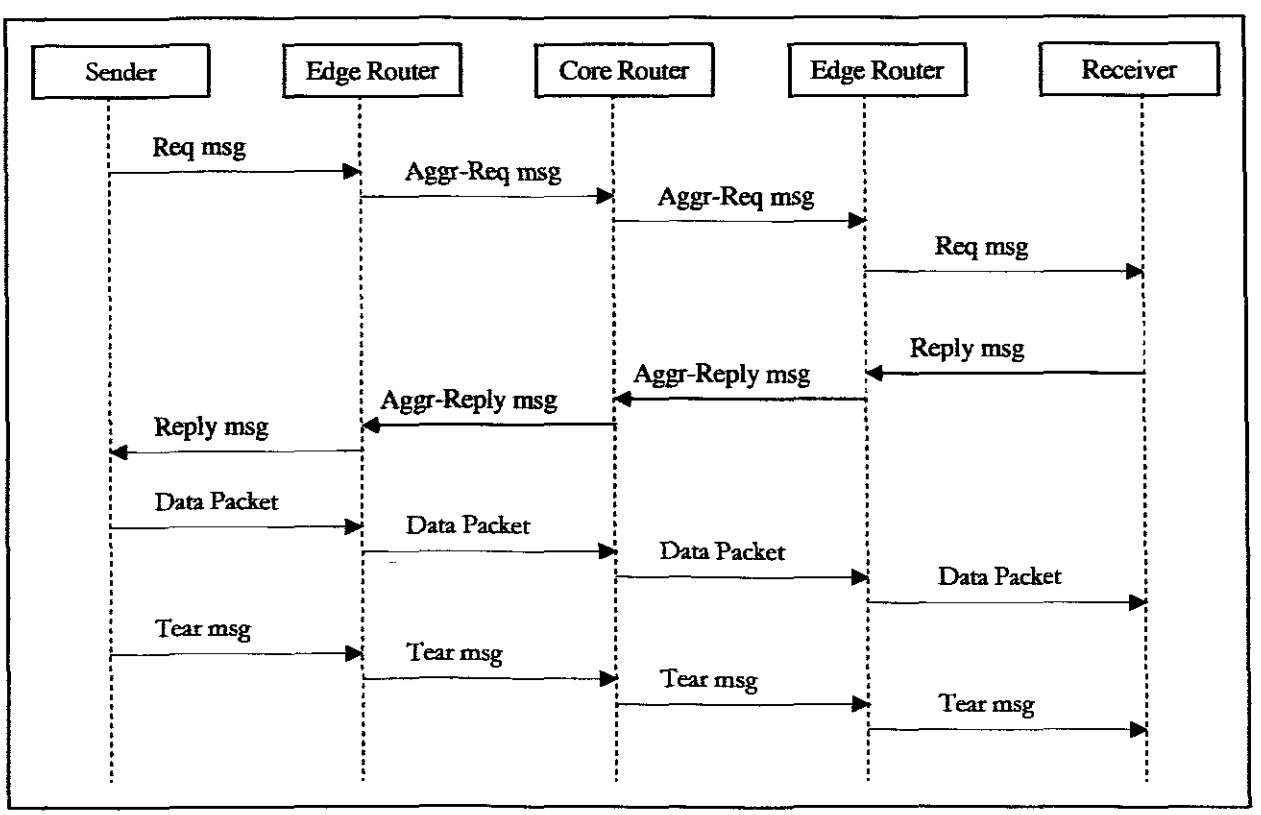


Figure 4.10 SERP Message Sequence diagram

4.3.6. Accommodating receiver heterogeneity

It is possible that a receiving host cannot accommodate the QoS level requirements specified in the request message. In this occurrence, as part of our design goal, the receiving host is allowed to modify the flow's requested QoS level and sends it own modified request upstream.

In a special unicast situation, when the reservation request reaches a network with low fidelity receiver and the resources required are too high. The low fidelity receiver can do backward reservation right up to the sending host.

It can also be applied in a multicast situation where the reservation goes to one receiver successfully but the other receiver cannot accommodate the reservation because of low fidelity. As illustrated in figure 4.11 if the reservation has reached a router in this network, the low fidelity receiver could make backward reservation.

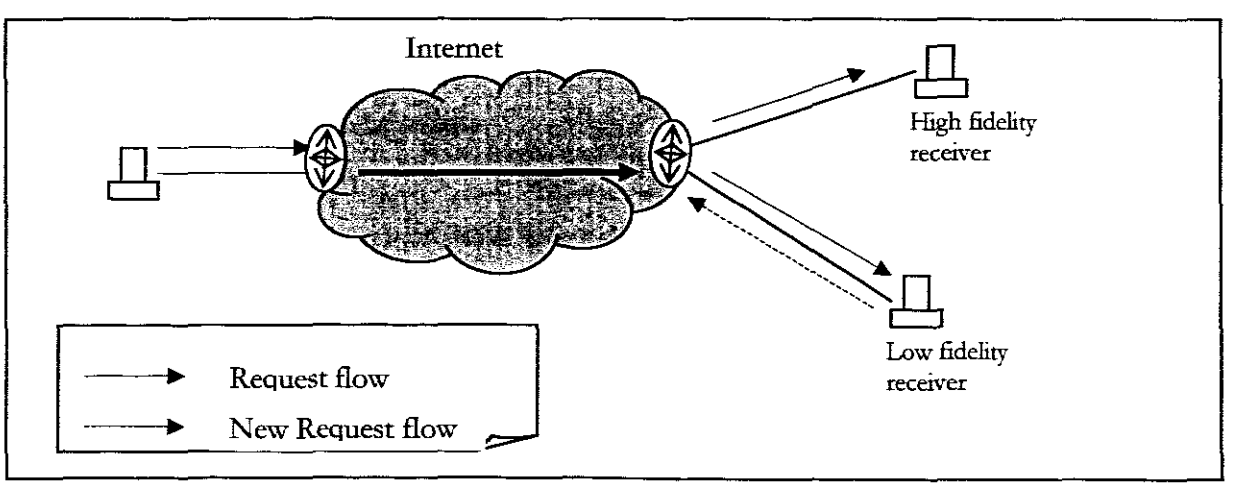


Figure 4.11 SERP receiver heterogeneity

4.4. SERP processing modules

This section presents the pseudo-code of the most important processing modules in SERP which are admission control and resource reservation. The notations used in these pseudo-codes are listed as follows:

a) **r** : resource requested

b) flow(**r**) : a flow requesting **r** resources

c) $\mathbf{R_{Total}}$: Total resources

d) $\mathbf{R_{Free}}$: Free resources

e) $\mathbf{R_{Used}}$: Used resources

In the simulation, the amount of requested resources (**r**) was considered a constant data rate for each flow so as to observe how the number of states varies with the number of flows without impacting on non-uniformly of the resources.

4.4.1. Admission control module

As a Request message traverses from one network node to the other to reach the receiver, different processes are invoked. Amongst other components, the admission control is one reservation setup process which needs to be fulfilled. Based on the amount of committed local resources provided by the reservation database table, the admission control determines whether the node has sufficient resources to satisfy the QoS requirement of the reservation session without having to violate QoS guarantees of any ongoing sessions. After receiving a request

message, SERP invokes admission control. The admission control process returns a Boolean value of '*accept*' when there are sufficient resources or a '*reject*' value in case of insufficient resources in the node.

The admission control algorithm used in SERP is illustrated in Figure 4.12. A basic admission control algorithm was used in the simulation to aid in the reservation process. We used the basic algorithm so as not to detract ourselves from the real task at hand which was to develop a resource reservation protocol.

```
Pre-condition: Identify traffic type, class of REQUEST flow
                    R_Total > 0
                    R_Free <= R_Total
                    R_Used >= 0

Function admissionControl(flow(r))
  R_Free = R_Total - R_Used
   If r <= R_Free then
      return accept
   else
      return reject
   end if
End Function

Post-condition: accept/reject flow
```

Figure 4.12 Admission control algorithm

4.4.2. SERP reservation module

SERP reservation module performs a very important role in resource reservation within nodes. As a key module of this simulation system, the SERP module plays the following multiple roles:

1) Accept incoming control messages: upon receiving any of SERP control messages, SERP reservation module accepts the messages and invokes the necessary processing module.

2) Setup and maintain reservation state: upon successful admission control, SERP reserves resources and then creates a reservation state for the requested flow. Reservation states are maintained in the nodes unless explicitly deactivated by the receipt of a TEAR message.

3) Update resources database: SERP is responsible for continuous update of the resource database. The algorithm for SERP reservation module is given in Figure 4.13.

Pre-condition: receive a request message

Function reserveResources(flow(**r**))

If admissionControl(flow(**r**)) is *accept* then

Serve flow

$R_{Free} = R_{Used} + \mathbf{r}$

$R_{Free} = R_{Total} - R_{Used}$

Create reservation state (**r**)

Update resourceDB( )

Else If admissionControl(flow(**r**)) is *reject* then

Drop flow

Send ERROR flow message

End if

End function

Post-condition: Serve/Drop flow

Figure 4.13 SERP reservation algorithm

CHAPTER FIVE

5. SIMULATION, RESULTS AND ANALYSIS

5.1. Introduction

Simulation was conducted to evaluate the performance of the proposed model. In this chapter we present the simulation, its results for tests conducted using the specified simulation parameters. Consequently this chapter is divided into 2 parts. The first part, in section 5.2, discusses the simulation of SERP. Section 5.3 covers the second part with details of experiments conducted to evaluate SERP's performance.

In the experimental design, two differently behaving reservation schemes were set up for simulation. One scheme emulated our own proposed reservation scheme, SERP and another one that emulated RSVP. SERP and RSVP were then compared using different scenarios. Each experiment is presented in section 5.2. In each experiment, the accumulated results are tabulated and illustrated in a graphical structure. The analysis of the results then follows at the end of the experiment.

5.2. SERP implementation

5.2.1. Simulation Model

To evaluate the model's performance, simulation was conducted. The simulation network is illustrated in figure 5.1. It consists of seven nodes made up of four end hosts and three routers. To keep it simple, two access networks are represented; Network A and Network B. In Network A the two end hosts are represented as A1 and A2 and in Network B as B1 and B2. Network A represents the sending network whilst Network B represents the destination network. Thus requests are directed downstream from the left to the right. Each sending host could initiate a resource reservation, and so sender-to-receiver communication can be based on a one-to-one, one-to- two, two-to-one or two-to-two relation.

Each node was assigned resources (bandwidth and buffer). The nodes are connected via links. A link transmits packets on a hop-by-hop basis between two nodes. The link's capacity was assumed to be high enough to transmit all flows.

In the access networks (Network A and Network B), per-flow reservation was carried out while in the core network, individual flow reservations were aggregated. The routers that connect the access network and the core network are referred to as the edge routers or boundary routers. In the simulation network topology, R1 and R3 act as edge routers such that R1 performs flow aggregation while R3 performs de-aggregation.

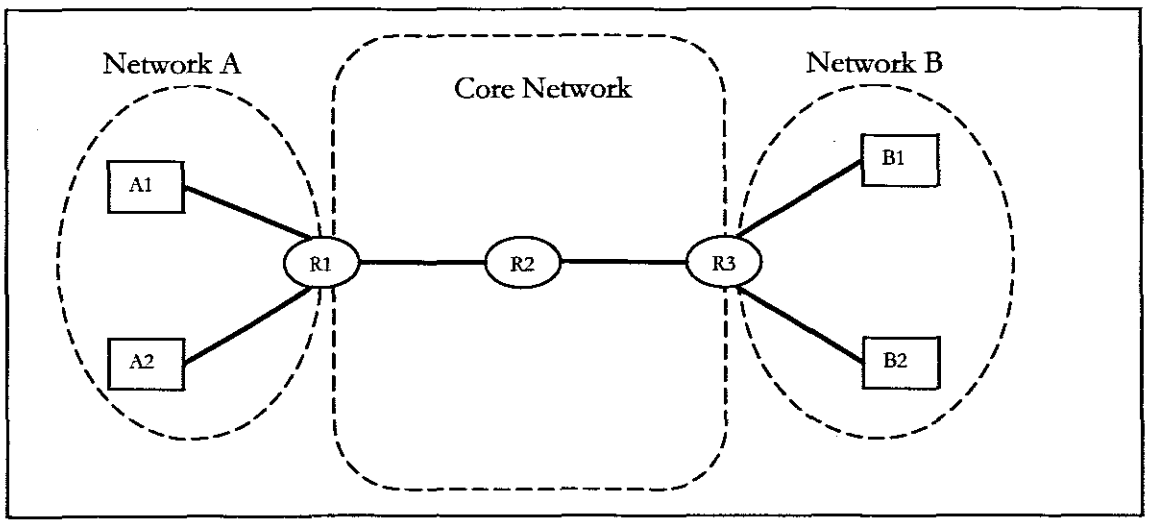


Figure 5.1 Simulation Network Topology

When the sender(s) and the receiver(s) have been specified together with the type of service requested, a request flow is generated to request resources. Data packets for a flow are also generated from a user's input message. These data packets are sent sequentially after successful reservation of resources. The number of desired resources is determined by the amount of resources required to transmit the data flows. Each node was assigned a total amount of resources (bandwidth).

It was assumed that there were not any link failures and that all links worked correctly throughout the simulation. Reservations were assumed to be of Guaranteed Service (either guaranteed constant rate (GCR) or Guaranteed Minimum Rate (GMR)) so as to perform end-to-end reservation.

5.2.2. Simulation Environment

Simulation was implemented using Microsoft Visual Basic.Net on the Microsoft Visual Studio.Net 2003 environment. The .Net package was selected because it is totally object oriented and has simplified development efforts. This makes VB.net advantageous and easily deployable. Microsoft Access 2003 was used for storage of the database and of the simulation results statistics. The simulation was run on a PC running Microsoft Windows XP operating system with an Intel Pentium 4 processor.

The objects identified that interact in the simulation include: network, router, packet, links, routing table, and nodes. A complete class diagram can be found in the Appendix.

5.2.3. Simulation Parameters

The simulation parameters used to evaluate our proposed model's performance were load and time. The load is the number of flow sessions (accepted and failed). Simulation time was also used as parameter to check how the SERP behaves as the simulation progressed. For experimental purposes, flow load was set between 0 and 30 and the simulation time was varied from 0 to 300 seconds. The flow's request messages were sent at constant rate but with different amount of required resources for each flow.

5.3. Experiments

The goal of the experiments was to observe the behaviour and the effect of using SERP in a network for resource reservation. The set of experiments conducted evaluated the efficiency and the scalability properties of SERP with varying load and time. We present the details of the experiments along with the results and their importance in achieving the research design objectives.

5.3.1. State maintenance

5.3.1.1. Test

According to design, nodes maintain state information upon successful reservation setup. The number of states maintained creates a load in the nodes and this can cause scalability problems. In this experiment, the simulation was configured to observe if in SERP, as the load is increased, how much states are maintained in the nodes. Two sets of experiments were conducted to demonstrate how the states maintained in routers perform with the increasing load.

In this first scenario, we looked at how reservation states are maintained in both schemes (SERP and RSVP). For both schemes, flow load was increased from 1 to 20 (number of flows) while keeping the amount of requested resource the same for all request flows.

For experimental purposes, a single router was used to check the number of states maintained in it. The amount of resources in that router was increased such that all requests were successful.

5.3.1.2. Results

The results of the conducted experiment specified in scenario 1 are graphically illustrated in figure 5.2. Figure 5.2 shows how reservation state is maintained in SERP and RSVP schemes. The corresponding captured data is presented in Table 5.1.The flow load was increased from 1 to 20 flows while keeping the amount of requested resources constant for each flow.

Table 5.1 Reservation states for SERP and RSVP

Reservation states	Load (number of flows)																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
SERP	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
RSVP	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40

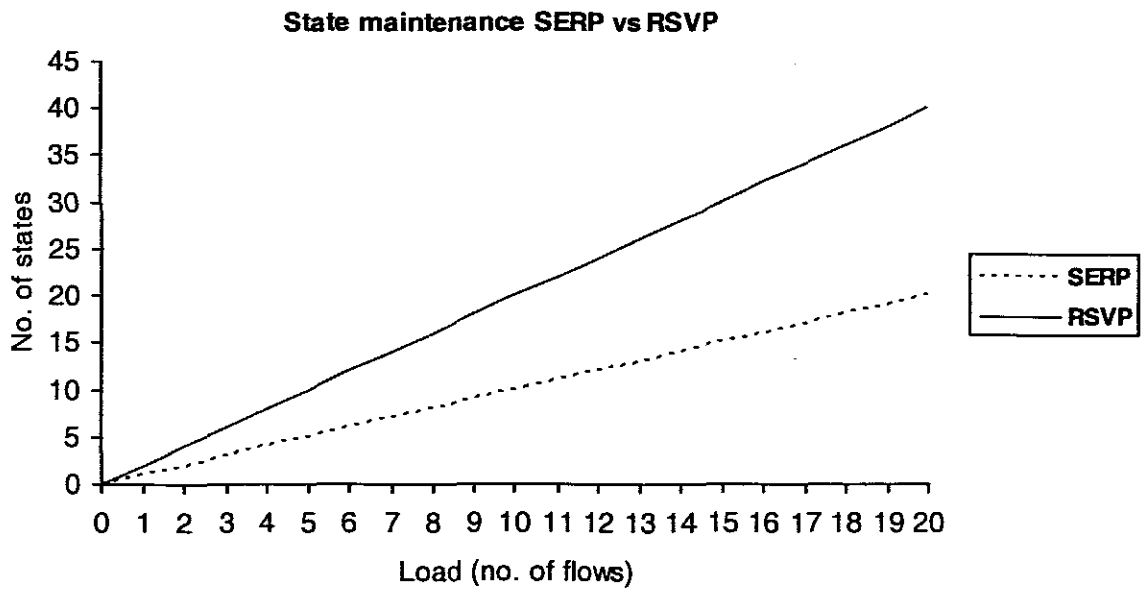


Figure 5.2 State maintenance evaluations, SERP and RSVP

5.3.1.3. Analysis

From the achieved results, it can be seen that in SERP the number of maintained reservation states increase linearly as the flow load increase but a lower rate compared to RSVP. This is because in SERP for one flow, only one reservation state is maintained. However, in RSVP, as the number of flow load increases, the number of reservation states maintained increases enormously. This enormous increase is due to both Path state and Resv state information that is maintained in the routers.

5.3.2. Aggregation on State Maintenance

5.3.2.1. Test

In this experimental test, a similar configuration as in scenario 1 was used. However, a core router was monitored to get the number of cumulated reservation states. A core router differs from an access router because it maintains reservation states for aggregate flows. In the first instance evaluation was done for the cumulated number of reservation states when aggregation was not carried out and again tested when aggregation was carried out.

For experimental purposes it was assumed that an aggregate can only accommodate five flows at a time. Naturally, real networks may use higher flows per aggregate. In SERP the number of aggregated flows depend on whether flows are destined for the same network and are of the same service class or not. The aggregated flows were thus assumed to accommodate these requirements.

5.3.2.2. Results

The results of the conducted experiment specified in second test are graphically illustrated in figure 5.3. Figure 5.3 shows the number of cumulated states for SERP with and without aggregation. The corresponding captured data is presented in Table 5.2. The flow load was increased from 1 to 20 flows while keeping the amount of requested resources constant for each flow.

Table 5.2 Reservation states for SERP with and without aggregation

SERP Reservation states	Load (number of flows)				
	0	5	10	15	20
Without aggregation	0	5	10	15	20
With aggregation	0	1	2	3	4

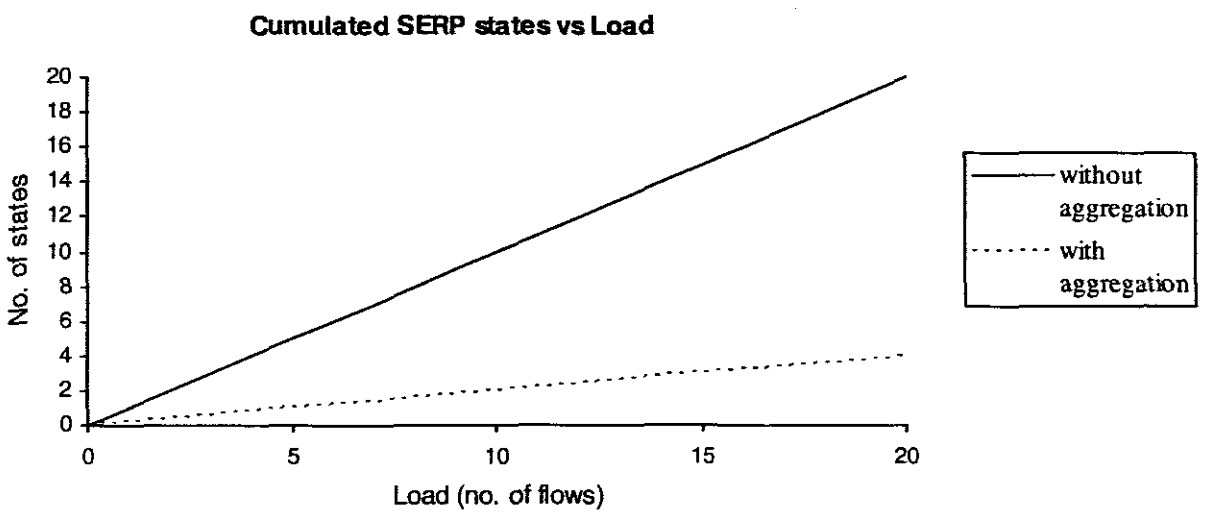


Figure 5.3 Cumulated States for SERP with and without aggregation

5.3.2.3. Analysis

The results show that the flow aggregation further reduces the number of reservation states maintained in the routers. In fact the number of maintained state is kept at minimum. As the load increases, the number of cumulated states also increases but with a lower rate. And if

routers could aggregate more than 5 flows, the rate for aggregate states could be lowered even further.

Based on the results of scenario 1 and scenario 2, it was observed that the number of reservation states maintained in both access and core routers are greatly reduced and as a result, scalability is improved. As a result, SERP-capable nodes improve scalability within the network.

5.3.3. Acceptance Probability

5.3.3.1. Test

In reality, the number of reservation requests received by nodes varies from every single node. In the second set of experiment, acceptance probability was evaluated. Acceptance probability is taken as the probability that reservation requests will be accepted. In this experiment, the percentage of accepted reservation requests was evaluated over the total transmitted load. The load was varied from 1 to 20 request flows. The aim was to check how nodes react in terms of request acceptance with increasing load.

Ideally, it is favourable that the number of accepted requested flows to be not affected by the load increase. Although a decrease in the number of accepted requests flows was anticipated, but a percentage above 50 would be acceptable.

5.3.3.2. Results

Figure 5.4 shows how variation in load affects the acceptance probability in both SERP and RSVP. Table 5.3 illustrates the corresponding captured data for figure 5.4. The flow load was increased from 1 to 20 flows while keeping the amount of requested resources constant for each flow.

Table 5.3 Percentage of accepted flows for SERP and RSVP as load varies

Reservation Requests	Load (number of flows)						
	1	5	10	15	20	25	30
Accepted SERP	100	90	60	66	70	64	60
Accepted RSVP	100	90	60	62	61	51	47

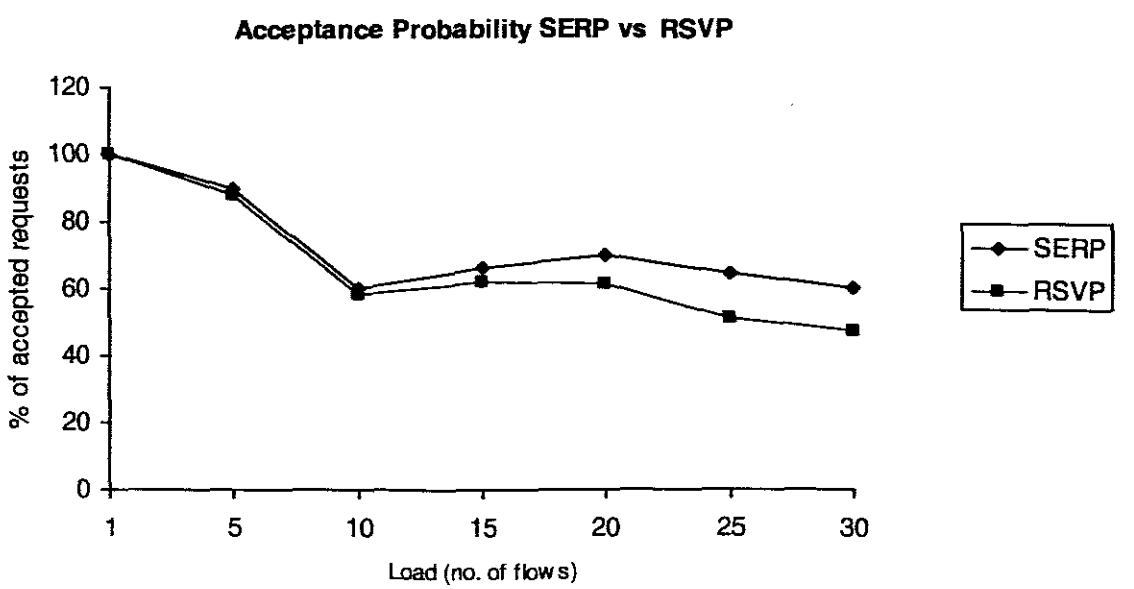


Figure 5.4 Acceptance Probabilities, SERP vs. RSVP

5.3.3.3. Analysis

From the plot it is observed that, in both SERP and RSVP, as anticipated, the number of accepted requests decreases as the flow load increases. However, the percentage of accepted request flows stays above 50% with SERP. RSVP's rate however slides below 50% and this means more RSVP request flows are rejected. Rejection in RSVP is even more serious since all the routers will have to delete the path state whereas in SERP, only those routers which are from sender downstream up to the point of rejection, that will perform the deletion of state information.

5.3.4. Connection establishment

5.3.4.1. Test

For end-to-end resource reservation to be realised, connection from sender to receiver needs to be established. The connection is established when a sender sends request messages to the receiver and the receiver will have to send a positive confirmation for connection to be established. The aim of this experiment was to check the number of connections that are established, in both SERP and RSVP, as the simulation time increases. In the experiment, the parameter time was varied from 0 to 600 seconds. A single request flow was transmitted per second with a constant amount of required resources.

In SERP, connection is established once the sender receives a REPLY message from the receiver whilst in RSVP the connection is established on the receipt of a RESV message.

5.3.4.2. Results

The results of the conducted connection establishment experiment are graphically illustrated in figure 5.5. The graph illustrates the number of connections established in both SERP and RSVP over varying time. The corresponding captured data is presented in Table 5.4.

Table 5.4 Connection establishment over time

Number of established connections	Time (seconds)												
	0	50	100	150	200	250	300	350	400	450	500	550	600
SERP	0	1	3	8	9	9	9	10	11	13	15	15	15
RSVP	0	1	1	3	4	5	5	5	7	9	11	12	12

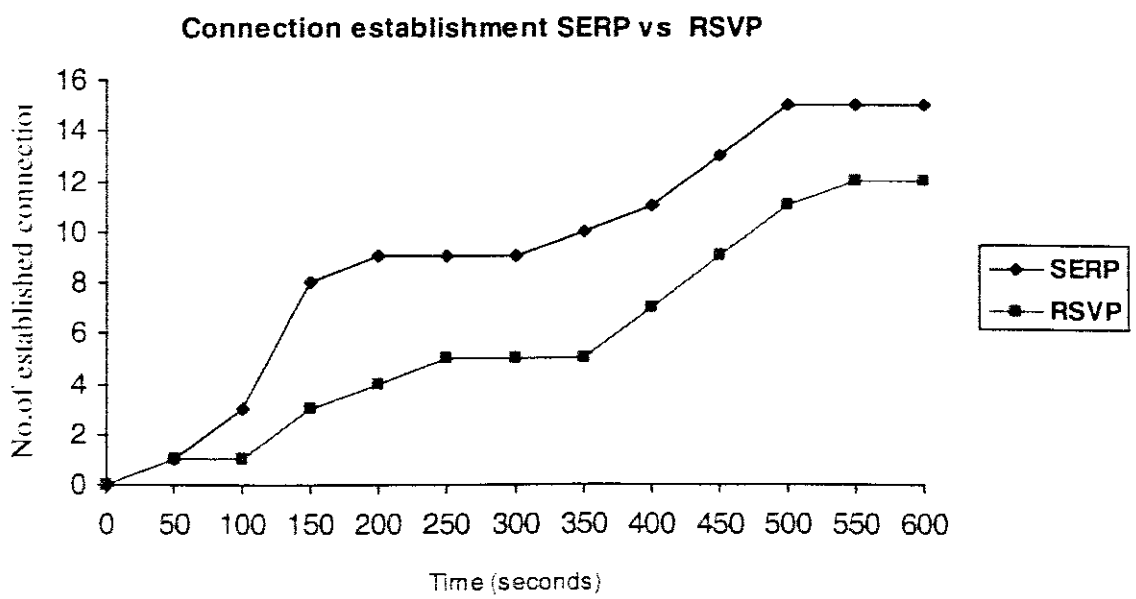


Figure 5.4 Connection establishments, SERP vs. RSVP

5.3.4.3. Analysis

The graphical results demonstrate that as time is increased, more connections are established in SERP compared to RSVP. The sender-initiated reservation requests clearly consume more resources compared to a receiver-based one. It is noticeable in both schemes that there are certain times where the number of connections remains almost constant. For example, in SERP, from 150 to almost 350 seconds, the number of established connection is constant. This behaviour is caused mainly by rejections to insufficient available resources. The graph increases immediately after some free resources are available for consumption.

CHAPTER SIX

6. CONCLUSIONS

6.1. Introduction

This dissertation highlighted the challenges encountered in the Internet due to high Internet demands placed by newly developed user applications. These challenges brought up needs for QoS provisioning techniques to be revised and improved. Specifically, the investigation focused on resource reservation as a QoS provisioning technique. The overall goal is the design and analysis of a scalable and efficient reservation protocol to offer end-to-end QoS guarantees to user applications. We named the protocol, Scalable Efficient Reservation Protocol abbreviated as SERP.

This chapter reviews the developed resource reservation scheme. The achievements of the research work are reviewed. The chapter also contains a reasonable critique of the work with a view to suggesting how the work could be extended in the future. To conclude the chapter, we presented ideas for future work and the direction of the future efforts to increase the functionality of the developed scheme.

6.2. Summary

Due to high resource reservation challenges experienced in the Internet during the reservation process, Scalable Efficient Reservation Protocol (SERP) has been proposed. SERP will offer scalability and efficiency to the Internet by performing reservation of resources in the network.

The first objective was to identify relevant design goals and principles for the proposed resource reservation scheme. To achieve this objective SERP was established based on various resource reservation design principles from already existing schemes from which some dynamic principles were identified. Existing design principles were adopted and reused to solve some of the reservation design challenges which were encountered in the literature which are: scalability, complexity, failure to provide end-to-end QoS and high-protocol processing overhead.

The second objective that was set was to formulate a corresponding model with control messages and protocol functionality. This objective was realised by adopting the generic resource reservation protocol's operation. Thus SERP's four control messages (REQUEST, REPLY, ERROR, and TEAR) are used for reservation setup, termination and error reporting. Also, SERP operates based on the following features: SERP is both sender- and receiver-oriented. Hard state maintenance is applied. Aggregate-based reservations are used within the core network, and end-to-end reservation is guaranteed for applications.

Finally, the third objective was to conduct a number of performance evaluations with interpretation of the results. To achieve this objective, a simulation of the proposed model was conducted. The simulations conducted show that SERP performs better when compared with RSVP. Several experiments were conducted using the simulation parameters (load and simulation time) and various performance metrics such as cumulated reservation states, acceptance probability, and connection establishment.

SERP has its own limitations however. Firstly, there is no mechanism used to evaluate whether the processing overhead is high or low. Also, the use of a simple admission control limits the SERP process. Besides, only stationary nodes were taken into consideration. Despite its limitations, SERP's concepts could be the basis for the resource reservation protocol of the future Internet.

6.3. Future Work

The results obtained from the simulations showed suitability of the proposed scheme for real-time applications. However, the simulations are only an approximation of the reality; therefore another primary goal in future is to observe the behaviour of the SERP in real networks. A test-bed network could be constructed, with SERP as the resource reservation protocol where both real-time applications and non-real-time applications are used.

To evaluate SERP's processing overhead, some performance metrics such as round trip time could be used to evaluate the processing overhead. The performance parameters used in this work could also be used.

A class-based admission control algorithm (work-in progress) proposed for Rainbow Service could be used with SERP instead of the basic one used in this research work. SERP can be further extended to the mobile environments, where mobility parameters for the wireless environment could be used.

REFERENCES

[1] Braden R, Clark D, and Shenker S, "Integrated Services in the Internet Architecture: an Overview", IETF RFC 1633, July 1994.

[2] Blake S, Black D, Calson M, Davies E, Wang Z, and Weiss W, "An Architecture for Differentiated Services", IETF RFC 2475, December 1998.

[3] Bernet Y, Yavatakar R, Ford P, Bakes F, Zhang L, Speer M, Braden R, and Dacie B, "A framework for Integrated Services Operation Over DiffServ Network", RFC 2998, November 2000.

[4] Ojong G, and Takawira F, "Rainbow Services: A New Architecture for QoS Provisioning in the Future Internet", Proceedings of SATNAC 2003, September 2003.

[5] Xiao X, and Ni L, "Internet QoS: A Big Picture", IEEE Network Magazine, March/April 1999.

[6] Dhesikan S, "Quality of Service for IP Videoconferencing", Engineering White Paper, Cisco Systems, June 1st, 2001.

[7] Cisco IOS QoS, "Quality of Service", Internetworking Technologies Handbook, Chapter 49, Posted: February 2003, http://www.cisco.com/warp/public/732/Tech/quality.shtml, last accessed on 5 December 2005.

[8] Chakravorty R, Kar S, and Farjami P, "End-to-End Internet Quality of Service (QoS): An Overview of Issues, Architectures and Frameworks", Proceedings of ICIT 2000, December 2000.

[9] Zhang L, Deering S, Estrin D, Shenker S, and Zappala D, "RSVP: A New Resource Reservation Protocol", IEEE Network Magazine, Vol. 7, pp 8-18, September 1993.

[10] White P, "RSVP and Integrated Services in the Internet: A Tutorial", IEEE Communications Magazine, pp. 100-106, May 1997.

[11] Cisco systems, "Resource Reservation Protocol", Internetworking Technologies Handbook, Chapter 48, February 2002, http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/rsvp.htm, last accessed on 2 December 2005.

[12] Passmore D, "Going With The Flow", BUSINESS COMMUNICATION REVIEW, 1 June 2003.

[13] Forouzan B, "Data Communications and Networking", Third Edition, McGraw-Hill, International Edition 2003.

[14] Zhang G and Mouftah H, "End-to-End QoS Guarantee over Diffserv Networks", Proceedings of Sixth IEEE Symposium on Computer and Communication, IEEE 2001.

[15] Wolf L, Delgrossi L, Steinmetz R, Schaller S, and Witting H, "Issues of Reserving Resources in Advance", Proceedings of 5th NOSSDAV, April 1995.

[16] Pan P and Schulzrinne H, "YESSIR: A Simple Reservation Mechanism for the Internet", ACM SIGCOMM Computer Communication Review, Volume 29, Issue 2, April 1999.

[17] Reid D and Katchabaw M, "SCAR: A Stateless Approach to Achieving Scalable Quality of Service", Proceedings of the 2005 International Conference on Internet, June 2005.

[18] Chung T, Chang H, and Leung V, "Flow initiation and reservation tree (FIRST)", Proceedings of IEEE Conference on Communications, Computers and Signal Processing, September 1999.

[19] Mathy L, Hutchison D, Schmid S, and Simpson S, "REDO RSVP: Efficient Signalling for Multimedia in the Internet", Proceedings of IDMS'99, France, October 1999.

[20] Huizhou Z, Binbin C, and Wei Y, "Aggregate-Resource Reservation Protocol", http://www.apan.net/2003_busan/10.doc, last accessed on 10 November 2005.

[21] Fu X, and Kappler C, "Towards RSVP Lite: Light-weight RSVP for Generic Signaling", Proceedings of AINA 2003, Xian, China, March 2003.

[22] Almesberger W, Le Boudec J, and Ferrari T, "Scalable Resource Reservation for the Internet", IEEE Conference on Protocols for Multimedia Systems –Multimedia Networking, Nov 1997.

[23] White P and Crawford J, "A Dynamic Sender-Initiated Reservation Protocol for the Internet", 8th IETF Conference on High Performance Networking, Vienna, September 21-25, 1998.

[24] Ossipov E and Karlisson G, "SOS: Sender Oriented Signaling for a Simplified Guaranteed Service", Proceedings of QoFIS'02, 2002.

[25] Ossipov E, "The design and application of a simplified guaranteed service for the Internet", Licentiate thesis, KTH, Sweden, 2003.

[26] Fu X, "Development of QoS Signaling Protocols in the Internet", Proceedings of IEEE LCN Conference, Germany, October 2003.

[27] Ojong G, and Takawira F, "Bandwidth Allocation for Rainbow Services", Proceedings of AFRICON 2004, September 2004.

[28] Vali D, Paskalis S, Merakos L, and Kaloxylos A, "A Survey of Internet QoS Signaling", IEEE Communications Surveys, Fourth Quarter 2004, Volume 6, No. 4.

[29] Ji P, Ge Z, Kurose J, and Towsley D, "A Comparison of Hard-state and Soft-state Signaling Protocols", Proceedings of SIGCOMM'03, August 2003.

[30] Pan P, Hahne E, and Schulzrinee H, "Do We Need Resource Reservation?", April 2001.

[31] Fu H and Knightly E, "A Simple Model of Real-Time Flow Aggregation", IEEE/ACM Transactions on Networkings, Volume 11, No. 3, June 2003.

[32] Shenker S, Partridge C, and Guerin R, "Specification of Guaranteed Quality of Service", RFC 2212, September 1997.

[33] Wroclawski J, "Specification of the Controlled-Load Network Element Service", RFC 2211, September 1997.

[34] Heinanen J, Baker F, Weiss W, and Wroclawski J, "Assured Forwarding PHB Group", RFC 2598, June 1999.

[35] Jacobson V, Nichols K, and Poduri K, "An Expedited Forwarding PHB", RFC 2598, June 1999.

[36] Pointurier Y, "MPLS Unicast Fast Reroute", http://www.cs.virginia.edu/~mngroup/projects/mpls/, last accessed on 7 October 2005.

[37] McDysan E, and Darren S, "ATM Theory and Application", New York: McGraw-Hill, 1998, http://www.atmforum.com, last accessed on 7 October 2005.

[38] Welzl M and Mühlhäuser M, "Scalability and Quality of Service: A trade-Off?", IEEE Communications Magazine, June 2003.

APPENDIX

A. Class Diagram and Description

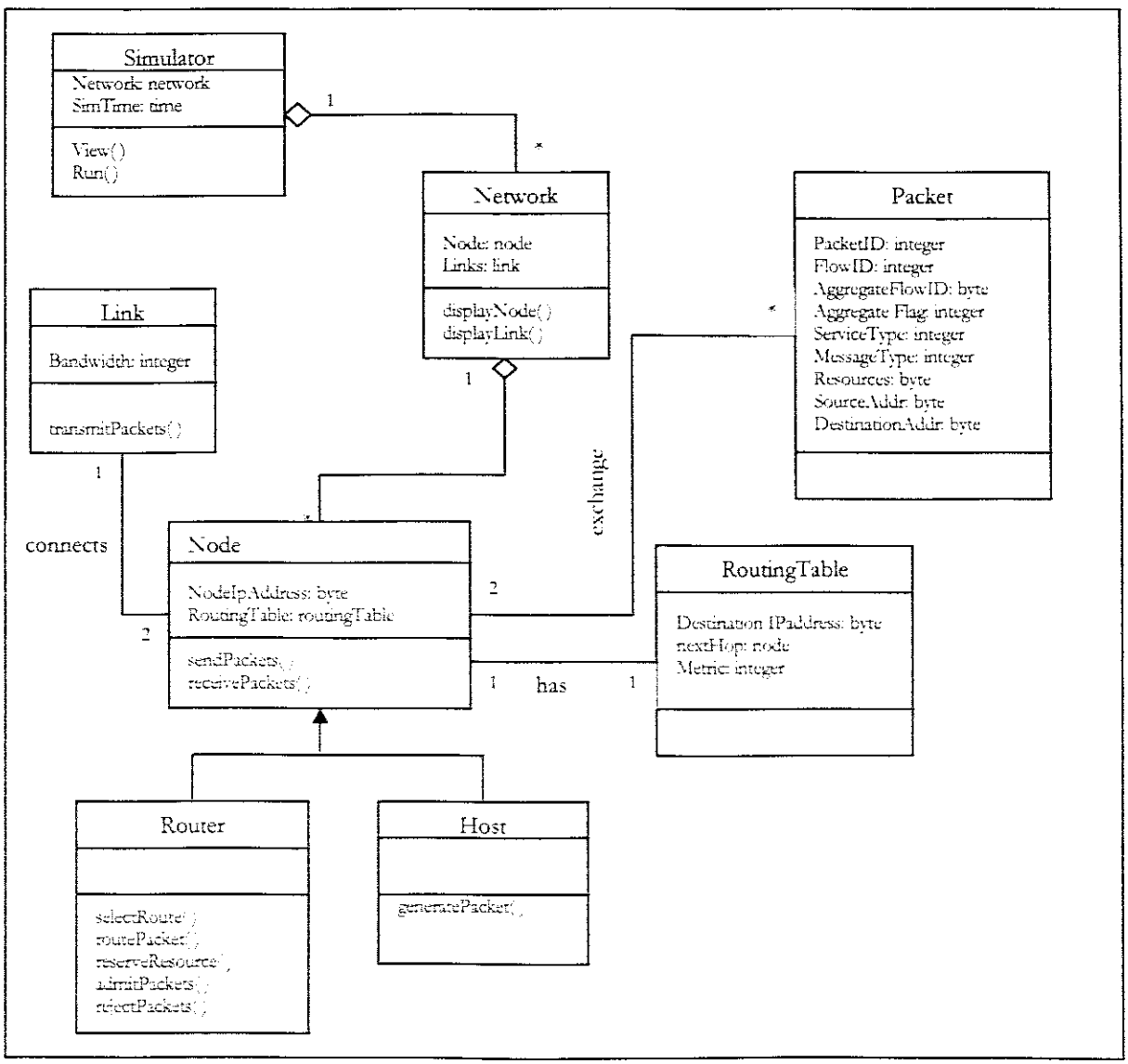


Figure A.1 Class Diagram

Classes Definition

Simulator: This is the container class for all classes. It consists of network with their nodes. Within the Simulator, users are able to enter and view simulation statistics.

Network: This class contains the entities of nodes connected by links. Within the network, nodes and links can be added and displayed.

Packet: This class represents the message that is transmitted over the network. It consists of the following properties, the source IP address, destination IP address, Message Type, and Service Type.

Node: This class is one of the fundamental building blocks of the network. There are two types of nodes associated with the network class used in the simulation, a router and a host. Each node is uniquely identified by an IP address and has a routing table which assists nodes in selecting the next best route for a flow to take. Nodes are able to send and receive packets.

Link: This class represents the connections between nodes in the network. A link transmits packets from one node to another. A link thus connects exactly two nodes.

Router: This class represents a node that acts as a gateway for packets from source to destination. A router utilizes a routing table to select the best route options for routing packets from one node to another. A router performs admission control and resource reservation for the packets (flows).

Host: This class represents a node in the access network. A host is able to send (when acting as a sender) and receive (when acting as a receiver) packets.

RoutingTable: This class represents a composite class of node class. A routing table class contains entries for route selection. The fields in the routing table include Destination IP address, the next Hop node IP address and Metric. The metric used was based on the number of hops the message will take to reach its destination.