A study of the effect of MPLS on Quality of Service in Wireless LANs

Jacobus Schutte

Presented in partial fulfilment
of the requirements for the degree

MAGISTER IN COMPUTER AND ELECTRONIC ENGINEERING

Faculty of Engineering
School of Electrical, Electronic and Computer Engineering
North-West University
Potchefstroom Campus

Supervisor: Professor A.S.J. Helberg

31 October 2006

Abstract

The increasing availability of wireless technology, both at home and in the workplace, offers many advantages in terms of mobility and freedom of connectivity. However, switching from a wired to a wireless transmission medium also presents unique research challenges, especially in the field of Quality of Service (QoS). There are many difficulties associated with meeting user expectations in a wireless environment, due to the unreliable nature of wireless communications.

This study focuses on QoS provision in an infrastructure-mode 802.11 wireless LAN (WLAN), simulated in the OPNET Modeler network simulation package. The topology consists of 802.11b access networks that send data to each other via a wired core network. The core is designed to include some of the unreliable characteristics of a wireless network. Multi-Protocol Label Switching (MPLS) is deployed in the network backbone as a means to manage the allocation of available resources. We then study the effect MPLS has on the QoS of voice, video, and File Transfer Protocol (FTP) application traffic, as well as on the performance of the wireless access networks. Two experiments are performed, the first with limited core bandwidth, and the second to investigate the effect of a link failure on the level of QoS. For each experiment, the MPLS scenario is compared to two identical networks, one without any QoS present, and the other with Differentiated Services (DiffServ) scheduling.

The results from Experiment 1 show that MPLS traffic engineering is able to effectively manage available resources to provide for all application types. The baseline scenario is unable to guarantee acceptable QoS, while DiffServ favours real-time applications at the cost of FTP traffic. In Experiment 2, the use of backup MPLS Label Switched Paths (LSPs) ensures that application QoS remains relatively unaffected despite the link failure, while notable QoS degradation occurs in the other scenarios. In addition, the use of MPLS in the network core achieves the highest WLAN throughput in both experiments.

Our approach offers potential benefits for office or campus networks, both for ensuring adequate QoS for application traffic, and to increase the reliability of the network backbone. Our research on MPLS traffic engineering should also be applicable in a wireless-only environment.

Opsomming

Die toenemende beskikbaarheid van radiokommunikasie tegnologie bied die gebruiker baie voordele t.o.v. beweeglikheid en kommunikasiegerief. Die oorgang van bedrade netwerke na meer onbetroubare radio-skakels hou egter ook groot navorsingsuitdagings in, veral in die gebied van dienskwaliteit (Quality of Service).

Hierdie studie is gemoeid met die dienskwaliteit in 'n infrastruktuur 802.11b radio-netwerk, wat gesimuleer word in die OPNET Modeler simulasiepakket. Die topologie bestaan uit 802.11b toegangsnetwerke wat kommunikeer via 'n bedrade sentrale netwerk. Hierdie sentrale netwerk is ontwerp om verskeie van die onbetroubaarhede van 'n radionetwerk in te sluit. Die MPLS (Multi-Protocol Label Switching) protokol word in die sentrale netwerk gebruik om beskikbare hulpbronne soos bandwydte te bestuur. Ons bestudeer die effek van MPLS op die kwaliteit van videokonferensie, Internet telefonie, en dataoordrag (File Transfer Protocol) verkeer, asook die effek wat MPLS op die radio-toegangsnetwerk het. MPLS word vergelyk met 'n basiese netwerk sonder enige dienskwaliteit, asook 'n netwerk wat van gedifferensieerde dienste (Differentiated Services, of 'DiffServ') gebruik maak. Twee eksperimente word uitgevoer, waarvan die eerste die bandwydte in die sentrale netwerk beperk, en die tweede die gevolge van die faling van een van die kommunikasie-skakels ondersoek.

Eksperiment 1 se resultate toon dat MPLS in staat is om die beskikbare hulpbronne effektief aan die verskillende tipes verkeer toe te ken. Die basiese netwerk is nie in staat om aanvaarbare dienskwaliteit te lewer nie, terwyl DiffServ die multimedia verkeer bevoordeel ten koste van die dataverkeer. In Eksperiment 2 sien ons dat MPLS alternatiewe roetes kan gebruik om die dienskwaliteit te beskerm teen die gevolge van 'n skakel wat faal, terwyl die diens in die ander netwerke merkbaar verswak. Die MPLS netwerk toon ook die hoogste deursettempo van die 802.11b toegangsnetwerk in beide eksperimente.

Ons benadering hou potensiële voordele in vir gebruik in kantoor- of kampusnetwerke, om beide die dienskwaliteit en betroubaarheid van die netwerk te verbeter. Hierdie navorsing kan ook van toepassing wees in 'n omgewing wat slegs radiokommunikasie gebruik, met geen bedrade netwerke nie.

Acknowledgements

I dedicate this work to all those who helped make it possible.

To my God, Saviour, and Counsellor, Who gave me the strength, wisdom and insight to successfully complete this huge undertaking.

To Mari Louise, my lovely wife-to-be, for all her love, encouragement and support.

To my friends and family, who stood by me and supported me despite the amount of time I poured into my work.

To Professor Albert Helberg, for all his guidance and great ideas that helped shape this research into its present form.

To the Telkom Centre of Excellence program, for granting me the opportunity to pursue this research work.

To the people at OPNET technical support, for their help in solving a few knotty problems with the simulation.

To all the above, and anyone else I may have left out by mistake, I offer my deepest gratitude.

Humbly and sincerely yours,

Jaco Schutte

21 November 2006

Table of Contents

Table of Figures

Table of Tables

Important acronyms

AF – Assured Forwarding

AP – Access Point

BA – Behaviour Aggregate

BE – Best Effort

BSS – Basic Service Set

CBR – Constraint-Based Routing

CFP – Contention-Free Period

CLS – Controlled-Load Service

CoS – Class of Service

CP – Contention Period

CSMA/CA – Carrier Sense Multiple Access with Collision Avoidance

CSMA/CD – Carrier Sense Multiple Access with Collision Detection

CTS – Clear To Send

CW – Contention Window

DES – Discrete Event Simulator

DiffServ – Differentiated Services

DIFS – Distributed Interframe Space

DSCP – DiffServ Code Point

DSSS – Direct Sequence Spread Spectrum

EF – Expedited Forwarding

E-LSP – EXP-inferred Label Switched Path

ESS – Extended Service Set

EXP – Experimental bits in MPLS header

FEC – Forwarding Equivalence Class

FHSS – Frequency Hopping Spread Spectrum

FIFO – First In, First Out

FTP – File Transfer Protocol

GS – Guaranteed Service

IEEE – Institute of Electrical and Electronic Engineers

IETF – Internet Engineering Task Force

IntServ – Integrated Services

IP- Internet Protocol

IR – Infrared

ITU – International Telecommunications Union

LDP – Label Distribution Protocol

LER – Label Edge Router

L-LSP – Label-inferred Label Switched Path

LSP – Label Switched Path

LSR – Label Switching Router

MAC – Medium Access Control

MAD – Medium Access Delay

MPLS – Multi-Protocol Label Switching

NAV – Network Allocation Vector

NGN – Next-Generation Network

OFDM – Orthogonal Frequency Division Multiplexing

OSI – Open System Interconnect

OSPF – Open Shortest Path First

PC – Point Coordinator

PCF – Point Coordination Function

PHB – Per-Hop Behaviour

PHY – Physical Layer

PIFS – PCF Interframe Space

PQ – Priority Queuing

QoS – Quality of Service

RSVP – Resource Reservation Protocol

RTS – Request to Send

SIFS – Short Interframe Space

SLA – Service Level Agreement

STA – Station (wireless)

TCP – Transmission Control Protocol

TE – Traffic Engineering

ToS – Type of Service

UDP – User Datagram Protocol

VoIP – Voice over Internet Protocol

WLAN – Wireless Local Area Network

1. Introduction

1.1 Background

Few would dispute that communication is one of the most basic of human activities. It enables us to interact with others, to share thoughts, feelings and ideas, to have relationships with the people around us. Without the ability to communicate, life as we know it today would be impossible.

From a scientific point of view, communication technology advanced very rapidly during the past two centuries. The 100 years between 1840 and 1940 saw the invention of the telephone, radio, television, and also the digital electronics that form the basis of today's telecommunications networks [72]. These breakthroughs changed the way we do business, and also added a new dimension to entertainment, as radios and television sets became commonplace.

A crucial step towards truly global communication came with the creation of the Internet in the 1970's and 80's. This allowed the widespread use of applications such as electronic mail (e-mail), which was invented back in 1965 but only made commercially available to computer users in 1979, as the network deployment increased [73]. It now became possible to instantly deliver a message to someone on the other side of the world, at the touch of a button.

Today of course, long-distance communication is almost taken for granted. Communication networks providing fax, telephony, file transfer, and e-mail services are an integral part of the modern office, school, university, factory, transport hub and the like. The Internet is still growing exponentially, as can be seen in Figure 1, and is now accessible to an estimated 15.7% of the world's population [72]. Communication technology also continues to develop at a rapid rate, with the widespread laying of fibre-optic cables, and the shift from analogue to digital broadcasts.

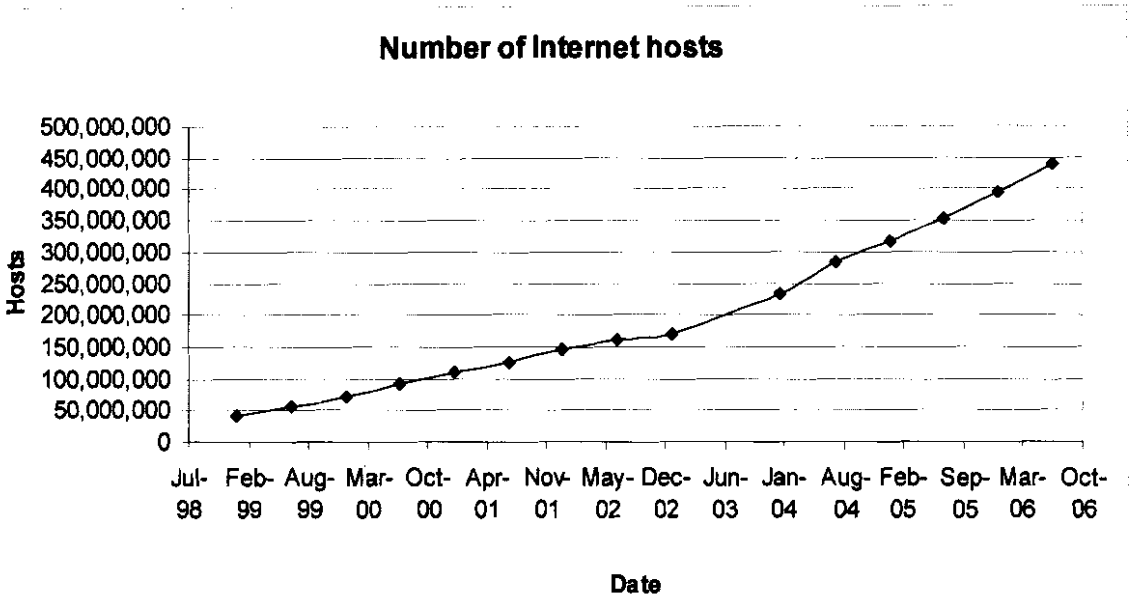


Figure 1 – Recent growth in the number of internet hosts [73]

Despite the unprecedented level of connectivity offered by the new and emerging communication techniques, one drawback remained. Until fairly recently, all the technologies mentioned above required the use of cables in one form or another. Telephones and traditional fixed computer networks use wires to send voice and data transmissions. Of course all electrical and electronic consumer devices require power, so even radios and televisions remain tethered to a cable, even though they are designed to receive wireless transmissions. This limits the mobility of the user by forcing him to remain in one place and in close proximity to the device.

The result of these limitations was an increasing interest in pure wireless communication. Some of the first major advances were made in the field of telephony. A radio telephone was tested by the Swedish police in 1946, and their use further increased during the 1950's, especially in Europe and the United States [74]. These devices enabled their users to roam about freely without being limited by a cable.

The mobile phone market really began to flourish in the 1980's with the arrival of the cellular phone. This growth has continued to such an extent that mobile phones are now more common than their fixed-line counterparts in many areas. In 2005 alone, 816.6 million mobile phones were sold across the globe [72].

Before long, wireless technology began to emerge in other areas of telecommunication. In the early 1990's, the wireless local area network (WLAN) made its appearance. The nodes in a WLAN communicate by means of radio links, thus eliminating the costly and time-consuming task of laying cables. Although this development had great potential, the first WLANs had limited speed, and lacked hardware compatibility. These early proprietary solutions were also expensive, and most companies preferred to retain their fixed networks [28].

To aid in the development of the WLAN, the Institute of Electrical and Electronic Engineers (IEEE) began work on a formal WLAN specification, resulting in the publication of the IEEE 802.11 standard in 1997 [21]. This offered data rates of 1 or 2 megabits per second (Mbps), transmitted either via infrared or in the 2.4 GHz radio band [56].

Despite these fairly low speeds, 802.11 had several advantages over proprietary WLANs. It allowed interoperability between products from different manufactures, offered better performance, greater range, and less susceptibility to interference from other devices [28]. The acceptance of 802.11 by the industry was swift, and today it is the leading WLAN technology available [1], [40], [59]. WLANs have been deployed in such diverse areas as hospitals, hotels, airports and academic institutions. Forrester Research in Cambridge, as quoted in [37], estimates that by 2003, 15% of industrial companies had already made use of wireless networks in their plants. It has been predicted that the total volume of mobile and wireless traffic will soon equal or even exceed that of fixed-line networks [2].

One of the main drivers behind the growing popularity of wireless networks is the demand for multimedia applications. Voice traffic was the primary reason for the astonishing growth of cellular networks. In a similar fashion, WLANs are increasingly used to carry multimedia traffic, such as voice over IP (VoIP), video conferencing, video telephony, and online games [2], [62]. However, providing these services in a wireless environment presents a considerable challenge. Real-time applications are much more resource-hungry than file transfers, e-mails or web traffic, and place very tight constraints on the network in terms of bandwidth, delay, and reliable packet delivery. These are commonly referred to as Quality of Service (QoS) parameters, and are used to measure the level of service the user can expect to receive from the network [5], [63].

It soon became apparent that legacy 802.11 WLAN was unable to provide the required QoS assurance for the growing volume of multimedia traffic. There are several reasons for this, many of them centred on the transmission medium itself. A wireless link is inherently less reliable than a wired one, due to noise, interference and the amount of signal loss that occurs during transmission [29]. Thus packets are frequently lost, resulting in an error rate more than three orders of magnitude greater than a wired LAN. Furthermore, the shared transmission medium causes packet collisions, forcing unwanted retransmissions which further increase delays [1], [5].

QoS provision has thus become a major research topic, not just in wireless but in fixed networks as well. Most of today's Internet Protocol (IP)-based networks provide only a best-effort (BE) service, which means that the user has no guarantees as to the level of service they will receive. The Internet Engineering Task Force (IETF) has developed several approaches in an attempt to add QoS to IP networks such as the Internet. Integrated Services (IntServ) appeared in 1994, followed by Differentiated Services (DiffServ) in 1998, and more recently, Multi-Protocol Label Switching (MPLS) [6], [30].

Unfortunately, the unique nature of a wireless network means that the techniques used in a wired network are not always directly applicable in a wireless one. The mobility of a wireless node causes changes in the network topology, and thus the link characteristics are variable and unpredictable, with link failures occurring more often than in fixed networks. As mentioned, the resources in a WLAN are also severely limited [1], [5], [24].

802.11 WLAN will very likely have an important role to play in the future of communication networks [1]. In recent years, there has been a shift towards network convergence, with the goal being the merging of separate network infrastructures into a common IP-based core network [2], [6]. This is known as the Next Generation Network (NGN) architecture [21], [30], [38]. As Figure 2 shows, the NGN will consist of an IP core network, accessed by a variety of edge networks. Wireless networks such as 802.11 WLAN will be expected to provide "anytime, anywhere" access to the users of the NGN [21]. Thus, the task of ensuring adequate QoS in wireless networks will continue to be of great importance in developing and enhancing the next generation of communication networks. This brings us to the subject of this study.

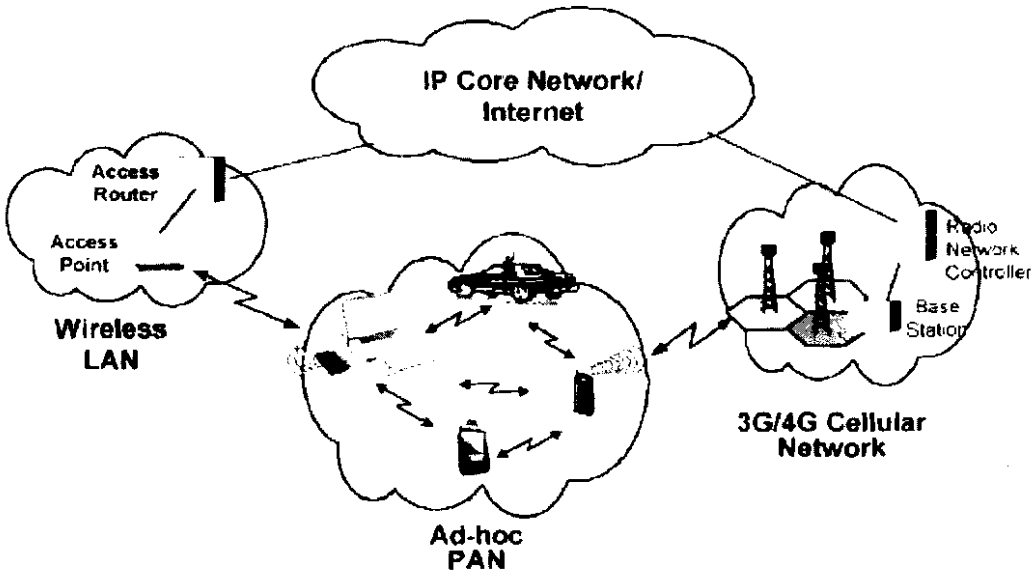


Figure 2 – Mobile communication in the NGN network [21]

1.2 Problem statement

A great deal of work has been done on the subject of QoS in wireless networks in general, and 802.11 in particular, some of which is reviewed in Chapter 3. This is an ongoing and very important effort, especially in view of the growing number of wireless network deployments.

In this study, we adopt a somewhat different approach to those used in the majority of the literature. As mentioned in the previous section, the current IP QoS frameworks (IntServ and DiffServ) were not designed with wireless networks in mind. MPLS, however, has certain attributes that make it more attractive for use in a network with limited resources. These include the low overhead it imposes on the routers, and support for traffic engineering (TE) [2], [55].

We propose to deploy MPLS in a network backbone connecting 802.11 WLAN access networks, with the aim of providing QoS for different types of application traffic. The network backbone will incorporate some of the characteristics commonly found in wireless networks, such as limited bandwidth. We will then use simulations to determine the effect MPLS has on the QoS of each application.

Based on the quantitative data gathered from the simulation we will make recommendations on the deployment of MPLS as a QoS technique in networks similar to ours in layout and topology. Areas of interest include office or campus networks that make use of WLANs to access a larger network infrastructure. The research data may assist administrators of such networks in deciding whether the deployment of MPLS will be of benefit to them. This research can also serve as a platform for future work involving the use MPLS in a completely wireless environment.

1.3 Scope of the research

Based on the literature survey as given in Chapter 2, the proposed research covers certain specific sub-problems.

MPLS was originally intended for use in large IP core networks, while our proposed area of research leans toward the office or campus environment. A suitable network configuration must be found that will support the deployment of MPLS.

The simulation was implemented in the OPNET Modeler simulation package, using the models described in [69]-[71]. A best-effort scenario was created in order to baseline the performance of the basic 802.11 standard. Having obtained this data, a quantitative comparison was drawn between the baseline network, a network using DiffServ, and the proposed MPLS solution, by investigating their performance under various network conditions.

Inevitably, there are limitations to our work which must be left for future study. The research data was limited to that obtained from the simulations, due to the current unavailability of an actual network that could serve as a test bed for our approach.

1.4 Document overview

The rest of this dissertation is organized as follows: Chapter 2 contains a broad literature study of the relevant subject matter. We discuss the concept of QoS, as well as taking an in-depth look at various aspects of 802.11 WLAN. Chapter 2 also reviews the current IP QoS techniques, and how they relate to this study.

Chapter 3 explains the methodology used in conducting the research. We firstly look at previous work done in the field of wireless QoS, with particular emphasis on 802.11 WLAN, and deployment of IP QoS technologies in a wireless environment. Based on this foundation, we next discuss the details and limitations of our proposed approach. Finally, we will explain the simulation setup, regarding the software used, the different scenarios, and details such as assumptions made and measured variables.

Chapter 4 contains the results obtained from each scenario. We use the quantitative data obtained from the simulation to compare the respective performances of a baseline, DiffServ, and MPLS network. These results are also discussed in this chapter.

In Chapter 5 we will draw conclusions from our findings and discuss the possible impact and benefits of this research, as well as making recommendations for future work related to this topic.

The references used as part of the study are given separately after Chapter 5, followed by the various appendices relevant to the study, but not included in the main body of the text.

2. Literature study

In this chapter we provide the theoretical background needed to place the rest of the research in context. The first step is to define the concept of QoS as used in this study, and how to measure it. Next, we look at the IP QoS technologies used in the study, namely DiffServ and MPLS. IntServ is also briefly discussed mainly for background purposes, but will not be used in the simulations. Lastly we discuss the IEEE 802.11 WLAN standard in some detail, including aspects such as operation, deployment, advantages and disadvantages.

2.1 Quality of Service overview

Data communications can be defined as the task of moving information from one place to another reliably, while also conforming to user requirements [51]. A great deal of information is transported by the various types of telecommunication networks, as we discussed in Chapter 1. As is the case with any person or object performing a service, the user has certain expectations regarding the level of service provided. A wealth of information such as we have available today is useless if the user is unable to access it when and where it is needed. The challenge of meeting this need has led to the concept of Quality of Service (QoS).

2.1.1 Defining QoS

Even though Quality of Service is a well-known term in the field of telecommunications, it has no single accepted definition. The literature provides quite a few versions, which we will briefly examine in order to gain a better understanding of what is meant by QoS.

In [1], QoS is defined as the ability of a network element such as an application, host or router, to provide some level of assurance for consistent data delivery. In [8], QoS is called the ability of a network to differentiate between traffic types and prioritize accordingly. In [17], QoS is described as the performance level of a service offered by the network to a user. The definition in [29] is very similar, saying that QoS describes a network behaviour that end users experience. A more complete definition found in [30] defines QoS as a set of specific requirements for a particular service provided by a network to the subscribed users.

Finally, Jerry Ash as quoted in [54], calls QoS "a set of service requirements to be met by the network while transporting a connection or flow; the collective effect of service performance which determines the degree of satisfaction of a user of the service."
Looking at these definitions, we can identify a few common factors:

i. QoS is linked to network behaviour.
ii. QoS places certain requirements on service provision.
iii. The goal of QoS is to achieve user satisfaction.

From an engineering perspective, the term "user satisfaction" creates a problem, namely that of qualitative versus quantitative QoS specification [29], [30]. *Qualitative* QoS involves the use of very informal terms to describe what the user requires of the network, such as "good service", "low delay", or "acceptable picture quality". An example of this type of QoS measurement is the Mean Opinion Score (MOS) test described by Recommendation P.800 of the International Telecommunications Union (ITU) [77], [78]. A group of individuals are asked to evaluate the quality of selected voice samples, and based on their perception the sample receives a score ranging from 1 (worst) to 5 (best). However, this evaluation is entirely subjective. Conversely, *Quantitative* QoS involves describing measurable network parameters through numerical values, such as "10 Mbps throughput" or "less than 1% packet loss".

How then does one translate the subjective experience of a user into concrete parameters that can be specified in the network design? The answer lies in determining what aspects of a particular service or application have a direct influence on the level of quality perceived by the user. In the case of a telephone call, for example, the quality of the service will be degraded if the mouth-to-ear delay between the speaker and the listener is long enough to hamper the conversation. Thus, transmission delay represents a quantitative, measurable factor that affects the QoS of the call. By identifying these quantitative parameters, it is possible to draw up precise requirements to describe the desired QoS of a voice call or other type of application. Our design goal is then to enable the network to meet or exceed these requirements.

2.1.2 QoS parameters

We will now look at the relevant network parameters used to measure the level of QoS. There are four that are generally accepted as being the most significant where application traffic is concerned, namely bandwidth, delay, jitter, and packet loss (See [7], [8], [17], [22], [51], [54], and [57]). A network generally carries many different types of traffic, and not all traffic is affected in the same way by these parameters. We pay particular attention to file transfer and real-time applications, as they are used further in the study.

Bandwidth and throughput – In the context of data transmission, bandwidth refers to the capacity of a transmission link, which is the amount of data it is capable of transferring per unit of time. It is usually measured in bits per second (bits/s). Bandwidth is, in effect, only the theoretical maximum capacity of the link. Of greater practical concern is the throughput, which is the amount of data per time unit that is actually transmitted by the link from the source to the destination. Throughput is almost always less than the bandwidth available, due to non-ideal channel conditions (errors, packet loss, etc). An 802.11b WLAN link, for example, has a stated capacity of 11 Mbps, but the actual achievable data rate has been measured at between 5.9 and 7.1 Mbps, depending on the protocol used to transmit the data [56].

Bandwidth and throughput are important from the user's point of view because they represent the 'speed' of the network. The higher the throughput, the more data can be sent at any given time. This enables the use of resource-intensive applications such as video conferencing. Inadequate throughput will result in the network receiving more data than it can reliably transfer. The result is a surfeit of packets in certain areas of the network, a condition known as congestion [8], [36]. The performance of a congested network rapidly deteriorates, sometimes to the point where no data is delivered at all (see Figure 3). Congestion also leads to increased delay, jitter and packet loss (see following sections).

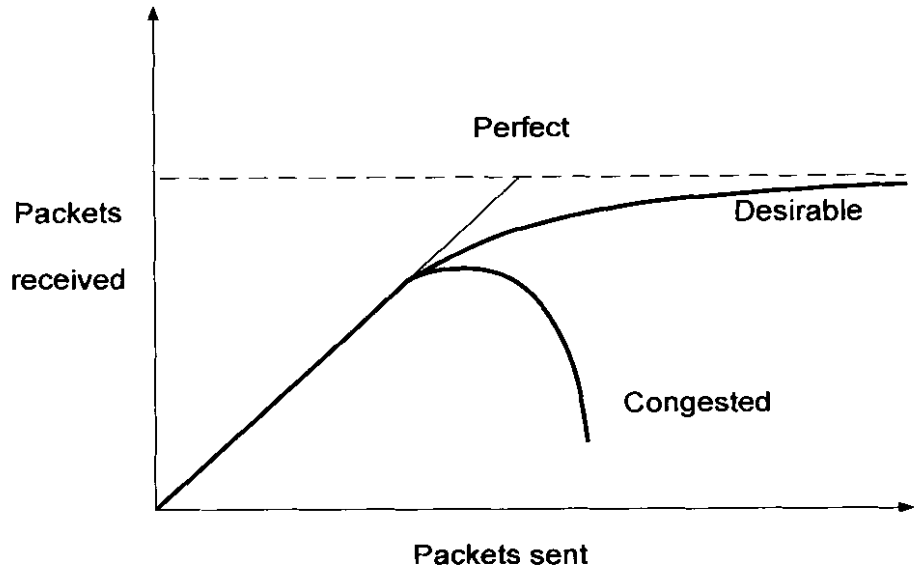


Figure 3 – The effect of congestion on packet delivery [51]

Delay – In any form of communication, the information does not just instantaneously appear at the destination. Likewise, data sent by a communication network is subject to a wait time or delay. There are several distinct types of delay that take place during the transmission of a packet [7]. Propagation delay is caused by the limited speed of a signal travelling across a link, and is governed by the physical properties of the transmission medium, whether optical fibre, radio link or copper wire. Switching delay is the time it takes for a network node such as a router to receive a packet and place it in the outbound queue. Scheduling or queuing delay is the time the packet spends in a queue or buffer before it is transmitted to the next node along the path.

Adding up all the different delays along the path gives the total delay, also referred to as *one-way delay*, which is defined as the time elapsed between sending a packet and its reception at the destination [51]. To a user, this represents the 'wait time' before the requested data is received. Depending on the type or current state of the network, the delay can range from a few milliseconds to several seconds.

File transfer, e-mail, and other non-real-time applications can tolerate fairly large delay times. However, delay is a crucial metric if the application being used is interactive. Returning to the example of the telephone call, a conversation becomes impractical if the round-trip delay is above 500 ms, which translates to a one-way delay of 250 ms [77].

ITU standard G.114 recommends a delay of no more than 150 ms from the speaker to the receiver for high-quality VoIP [7], [77], [78]. Delays between 150 – 400ms are acceptable but will degrade the perceived voice quality, while anything above 400ms represents unacceptable QoS. This principle also applies to video traffic since it similarly involves an interactive conversation.

As mentioned before, congestion in the network can severely impact the delay time. A packet may have to spend several seconds waiting in a queue before being transmitted, or could even be dropped if the buffers are filled to capacity. Lost packets then have to be retransmitted, further increasing the delay and adding to the frustration of the user. Ensuring low delay is thus an important step towards providing good QoS.

Jitter – Because of the various factors that impact the time it takes to deliver a packet, delay is difficult to predict, and the delay time between the reception of packets tends to vary [57]. This phenomenon is known as jitter. It is defined as the variation in delay between consecutive packets, and is usually measured in milliseconds [7], [22]. If the jitter value is positive, it means that delay times are increasing, whereas negative jitter indicates decreasing delays. Large jitter values, either positive or negative, correspond to sudden variations in packet delay.

Jitter is of little concern when transferring data such as e-mail, but has a significant effect on real-time applications. Ideally, the jitter value should be constant, meaning that each packet takes approximately the same time to traverse the network. But large delay variations will cause distortion in the playback of audio and video. When playing MPEG video over the Internet, for example, jitter of up to 2 seconds may be experienced [22]. To ensure good quality video or VoIP traffic at the receiving end, the jitter should be kept below 5 – 10 ms, with 30ms seen as the limit of what can be accepted [7], [82].

Packet loss – If a packet is transmitted but not received by the destination, it is considered lost. Packet loss is usually expressed as a percentage, indicating the ratio of packets lost out of those that were transmitted [82]. Loss can happen for several reasons. If an error occurs during transmission, the received packet may be corrupted, in which case it is usually discarded and re-sent. A poor quality link or a connection being lost altogether can prevent packets from being delivered.

However, the chief cause of packet loss is congestion in the network. Each node along the transmission path has limited queue or buffer space available. When all the queues are filled by an excess of traffic, the routers will start to drop any new packets they receive [36], [57]. If the network attempts to retransmit the discarded packets, the amount of traffic will increase, making the situation even worse (refer again to Figure 3).

Voice and video are more tolerant to packet loss than to delay and jitter, but data applications require a high degree of reliability [5], [34]. Lost packets represent lost data, which has to be recovered, or the user may receive a corrupted message or file. For high-quality data, the acceptable loss rate is less that 1% [7], [82].

2.1.3 Different approaches to QoS

IP networks were initially deployed without including a means to provide QoS for demanding multimedia applications such as voice and video. All traffic types received the same treatment, regardless of bandwidth or delay requirements. As we have seen, this is not acceptable for most applications [5], [54].

Service providers attempted to solve the resource problem by simply adding more bandwidth to the network. This solution has proved inadequate, since bandwidth is not the only important factor in ensuring good QoS. Also, continually upgrading the capacity of the network is a very expensive undertaking, forcing services providers to instead get the maximum performance from the resources at their immediate disposal [54].

Current practice is for the service provider and the user to make an agreement concerning the level of service the network must be able to provide. This is known as a Service Level Agreement (SLA), and is mainly specified in terms of the network parameters discussed in the previous section, namely bandwidth, delay, jitter, and packet loss. Broadly speaking, available QoS solutions fall into one of three categories:

Deterministic QoS – Also known as hard QoS, this approach guarantees a certain level of QoS for a given application or traffic flow [17], [51]. Very strict bounds are set in terms of delay, bandwidth, etc, and the network is expected to meet this requirement for the entire duration of a transmission. Hard QoS is usually achieved by some form of resource reservation mechanism, which ensures that the required resources will be available when needed.

Safety-critical applications such as remote surgery will typically have need of hard QoS guarantees [57].

Predictive QoS – Predictive or soft QoS cannot offer the same level of service guarantee as the hard QoS approach. Although soft QoS also sets bounds for the QoS parameters, they will only be met with a certain statistical probability at certain times [17], [51]. A soft QoS SLA may specify, for example, that the delay must be below 100 ms for 90% of the transmission duration.

Best-effort – If there is no QoS mechanism present in the network, all traffic types are handled as best-effort, and are thus equally susceptible to long delays or packet loss [5]. The routers forward all traffic on a first-in, first-out (FIFO) basis, and in the event of congestion occurring, the excess packets will be discarded regardless of traffic type. In a network which does have QoS control available, traffic with higher QoS requirements will be served first, and the best-effort traffic will be transmitted using whatever resources remain, without any service guarantees [51].

Regardless of the approach used, there are two necessary conditions for ensuring that QoS requirements are met, especially for applications that require a lot of resources. [54] lists them as follows:

- The application must have *guaranteed* bandwidth under all network conditions, including congestion and failures.
- The application must be handled according to *class*, meaning that certain traffic types will receive priority treatment from the network in terms of how to queue and forward packets and when to discard them.

Satisfying both these conditions simultaneously is a difficult task, and as a result not all networks are able to offer guaranteed levels of service.

In the following sections, we will discuss some of the frameworks designed to provide QoS in IP networks.

2.2 Integrated services

2.2.1 Background and working of IntServ

Integrated Services (IntServ) was the first of the architectural approaches to QoS developed by the IETF, and was introduced in 1994. The main goals were to improve the handling of real-time applications in IP networks, and to share the available bandwidth fairly among the different traffic classes. To achieve this, IntServ added two new services besides best-effort [15], [30], [54]:

Guaranteed Service (GS) is a hard-QoS service aimed at voice, video, and other time-bounded traffic types, and as a result includes assured bandwidth, no packet loss, and very strict limits on delay.
Controlled Load Service (CLS) uses the soft-QoS approach, and offers more predictable service levels than best-effort, though not guaranteed. It is intended for less demanding applications that can tolerate at least some delay and packet loss.

IntServ provides QoS on a per-flow basis [15], with a flow being defined as a stream of packets or datagrams between hosts that are created by the same application and therefore require the same level of QoS [8]. Depending on this level, a certain amount of resources is assigned to each flow by using the Resource Reservation Protocol (RSVP) [8], [51], [60]. RSVP is a receiver-oriented signalling protocol, meaning that the request for QoS comes from the receiving application. (See Figure 4 for details)

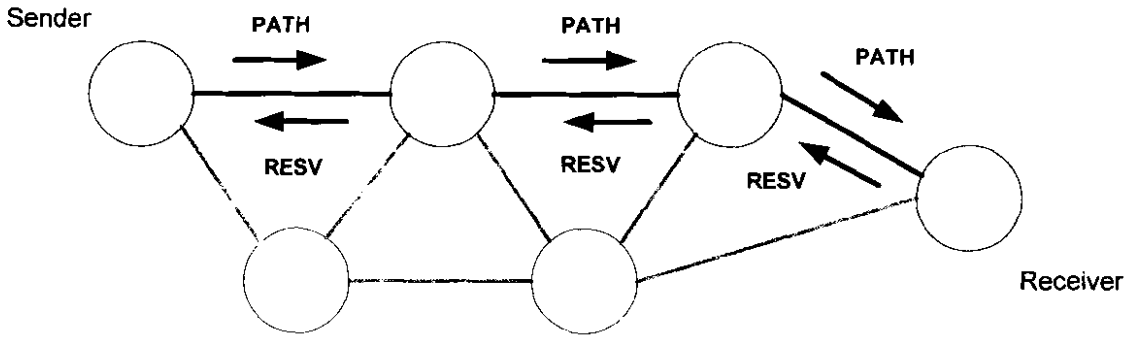


Figure 4 – IntServ / RSVP operation [51]

A path is created between the sender and receiver by sending a PATH message to all the intended recipients of the flow, which also includes a specification of the traffic offered by the sender. The receiver(s) then reply by sending reservation (RESV) messages back along the path, stating the QoS required by the receiving application, and requesting that the needed resources be assigned to that flow. At each hop along the return path, the node (such as a router) decides whether there are enough resources available to satisfy the request. If this is the case, the needed resources are reserved and the RESV message is sent to the next node, continuing until it reaches the sender. If at any stage a node has insufficient resources to comply with the request, an error message is sent to the receiver, and all the requested resources for that flow are released. Should the request be successful, the flow is then transmitted along the path reserved for it. Each node along the path uses a *packet classifier* to ensure that an incoming packet receives the specified level of QoS, and a *packet scheduler* to determine how each packet is forwarded to the next node along the path.

2.2.2 Limitations of IntServ

IntServ includes both guaranteed resources and class-based treatment, and therefore meets both the conditions for QoS. Unfortunately, this comes at a price. To guarantee that a flow will receive the requested level of QoS, each node along the intended path must have enough resources available. If even one node comes up short, guaranteed QoS cannot be provided and the request is denied [30]. But the greatest disadvantage of IntServ is the complexity involved in keeping track of all the traffic flows and their corresponding level of QoS [54]. Since resources are reserved on a node-by-node basis, each router has to maintain a current database of the state of the network and the flows it contains in order to classify an incoming packet correctly. This adds a significant amount of overhead to the network, a situation that grows worse if the network becomes large. Due to this lack of scalability, the deployment of IntServ has been severely limited [1], [54].

2.3 Differentiated services

2.3.1 DiffServ approach

The shortcomings of the IntServ approach led to the development of a somewhat different IP QoS architecture, namely Differentiated Services (DiffServ) [16], [30], [51], [54]. DiffServ does not offer the strict per-flow QoS of IntServ, but instead classifies the flows into one of a limited number of DiffServ classes [6]. A number of aggregated flows form a stream, which then receives a particular level of service from the network. This service level is based on an SLA between the source of the flow and the service provider. Having agreed on the QoS of the flow, it will be assigned to a stream that meets the requirements of the SLA. Combining individual flows in this manner reduces the load on the routers, and thus DiffServ is a simpler and more scalable solution than IntServ.

2.3.2 The working of DiffServ

The three elements used by DiffServ to provide QoS are *packet classification*, *traffic conditioning*, and *packet scheduling* [51]. Each of these takes place in a specific area of the DiffServ-enabled network, known as a DiffServ domain (see Figure 5).

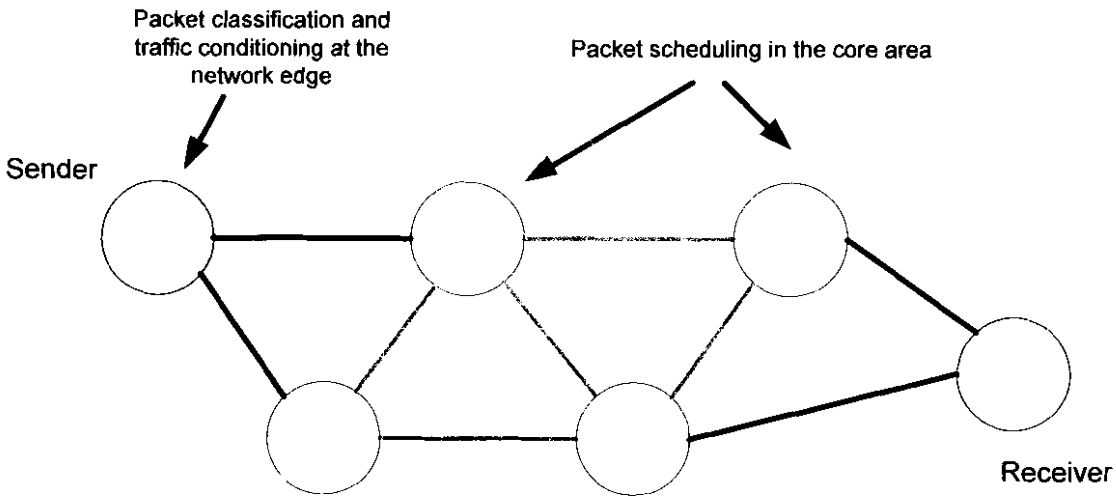


Figure 5 – The DiffServ domain [51]

Packet classification is done at the routers along the boundary of the DiffServ domain, also called edge routers. Each incoming packet is assigned a value according to the level of QoS agreed upon in the SLA. This value, known as the DiffServ Code Point (DSCP), consists of six bits in the packets' IP header, and is usually placed in the IP Type of Service (ToS) field (see Figure 6).

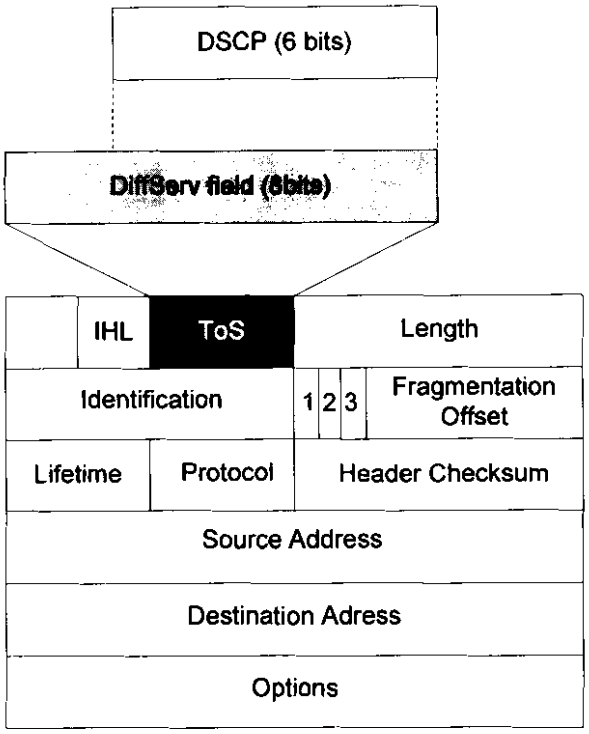


Figure 6 – The DSCP field in the IP header [51]

Traffic conditioning consists of inspecting the incoming traffic flow to ensure that it falls within the profile specified in the SLA. Out-of profile traffic will either be shaped to conform to the specification, or will be assigned to another service class.

Packet scheduling takes place in the core of the DiffServ domain. The DSCP value specifies what type of treatment the packet should receive from the network, with the six bits translating into 64 distinct behaviour types. All the packets with the same DSCP, and thus requiring the same level of service, are known as a Behaviour Aggregate (BA) [16], [54].

The forwarding treatment indicated by the DSCP value is called a Per-Hop Behaviour (PHB). A PHB specifies the forwarding priority that should be assigned to the flow, as well as bounds on delay, jitter, and packet loss. As the flow traverses the DiffServ domain, each core router

need only examine the DSCP of each packet to determine which PHB it requires. Currently, the DiffServ architecture includes three types of PHB [6], [7]:

- Expedited Forwarding (EF) class – EF is the premium DiffServ class, and is used to guarantee low loss, delay and jitter. Real-time traffic is usually marked as EF, because of the tighter bounds on these parameters.

- Assured Forwarding (AF) class – This class usually represents the majority of the traffic in a DiffServ domain. AF does not offer the same level of QoS as EF, and is used for less demanding traffic that requires better-than-best-effort service. AF traffic receives forwarding assurance from the routers, and thus is less susceptible to packet loss than best-effort traffic, but there are no guarantees as to the delay and jitter. There are twelve defined AF PHB's, which cater for traffic with varying levels of service requirements [54], [82]. These are listed in Table 1.

- Default PHB – all traffic not marked as AF or EF receives the default PHB. Traffic marked as Default is treated as best-effort traffic, with no service guarantees from the network, and will have the lowest priority at the routers.

Table 1 – DiffServ Assured Forwarding classes

Traffic priority level	DiffServ Code Point
Low	AF 11, AF 12, AF 13
Normal	AF 21, AF 22, AF 23
Medium	AF 31, AF 32, AF 33
High	AF 41, AF 42, AF 43

2.3.3 DiffServ disadvantages

DiffServ rectifies many of the shortcomings of IntServ, by being simpler, less resource-intensive, and thus easier to implement in large networks. Even so, DiffServ does not satisfy both conditions for QoS as stated in Section 2.1.3 [54]. It provides class-based treatment by means of the different PHB types, and is thus able to give certain traffic priority treatment at the routers. However, DiffServ does not consider the route the packets take through the DiffServ domain. This task is left to whatever routing protocol is being used.

Protocols such as Open Shortest Path First (OSPF) [51] tend to select the same path for all traffic with the same destination, regardless of class. This usually concentrates the traffic in a few links and neglects others, with the result that packets can easily be forwarded to a congested part of the network [11]. Consequently, DiffServ is not able to meet the first condition for QoS, which is guaranteed bandwidth.

2.4 Multi-protocol label switching

2.4.1 Introduction to MPLS

In the late 1990's, a type of technology known as tag switching emerged, with companies such as Ipsilon and Cisco being the main contributors [58]. Tag switching could perform packet switching over several different link technologies by using a label swapping procedure. This was done by adding a "tag" or label to the IP header of each packet, containing information similar to the postal code on a letter. The packet could then be forwarded based on this information instead of the IP destination address, resulting in a simple, high-speed process [60].

Multi-Protocol Label Switching or MPLS was developed and standardized by the IETF as a continuation of the work done on tag switching [52]. Like tag switching, MPLS is a packet-forwarding technique that makes use of label swapping to make routing decisions [2]. To achieve this, a label value called a 'shim' header is assigned to each packet as it enters the MPLS domain. The label value is fixed at 32 bits (see Figure 7), and is inserted between the packet's layer 2 and layer 3 headers [55]. As the packet travels through the MPLS domain, it is handled solely based on the label value, and the information in its original header is ignored. Forwarding packets by means of this technique is faster and less complex than using conventional IP routing tables [14].

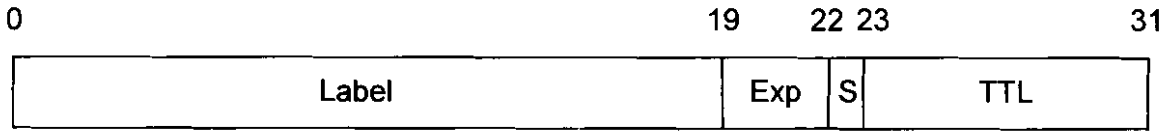


Figure 7 – The structure of the MPLS header [55]

The fields shown above have the following significance:

- Label: MPLS label value, 20 bits.
- Exp: For experimental use, 3 bits; currently used as a Class of Service (CoS) field.
- S: Bottom of stack, 1 bit. This bit is set to 1 if the current label is the last label in the label stack.
- TTL: Time to live, 8 bits. A decrementing hop counter that prevents packets from circulating indefinitely though the network. When the counter reaches zero, the packet is discarded.

The term "multi-protocol" is applied since MPLS can be used with any Layer 3 technology, not just IP, although IP traffic is the main area of interest [58]. MPLS was originally intended as a means to improve routing performance in IP core networks. More recently however, certain features of MPLS have caused it to be used increasingly as a way to provide QoS. MPLS enables the use of traffic engineering (TE) to improve the usage of available bandwidth, and can be combined with other technologies to differentiate between traffic types [9], [54], as we will explain in a subsequent section.

2.4.2 MPLS architecture and operation

MPLS is sometimes referred to as a "Layer 2.5" technology [79]. This is because it functions at, or more accurately, between, the link layer and network layer of the Open System Interconnect (OSI) protocol stack (See Figure 8). This is done by combining Layer 2 switching with Layer 3 routing functionality [8]. To this end, the MPLS architecture has two distinct planes, namely a control plane and a forwarding plane [10] (see Figure 9). This separation enables the two planes to operate independently of each other.

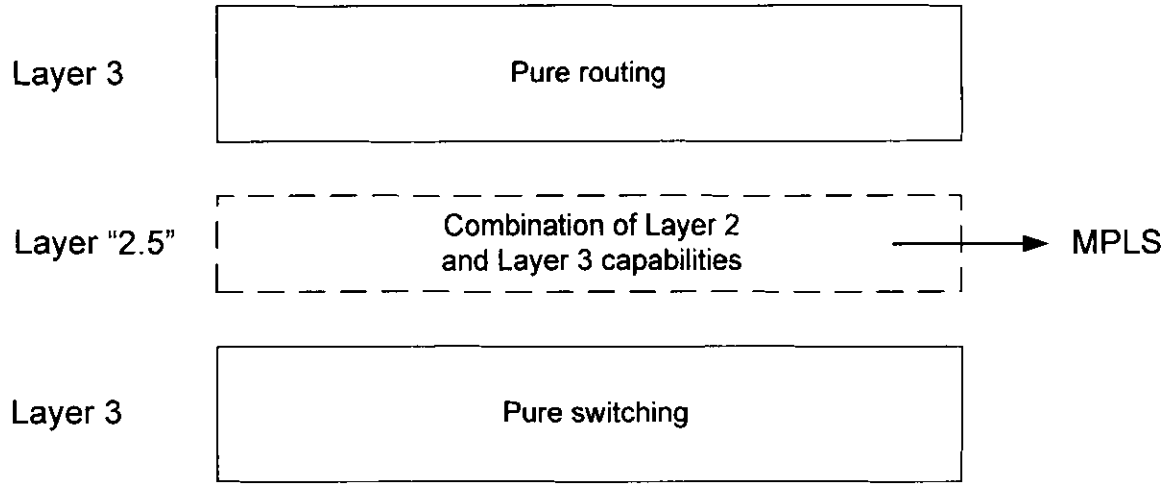


Figure 8 – Position of MPLS in the OSI stack [79]

The MPLS control plane handles the Layer 3 functionality of the network. It is responsible for maintaining the forwarding tables necessary to make a routing decision, and also for the distribution of control information concerning the network topology and the availability of resources. This is done by using existing IP routing and signalling protocols such as OSPF. Label distribution is done by means of a signalling protocol or label distribution protocol (LDP) [10], [46].

The forwarding plane performs the actual task of label swapping, as we will shortly explain. It is this plane that is used by the traffic when passing through an MPLS-enabled network element [10].

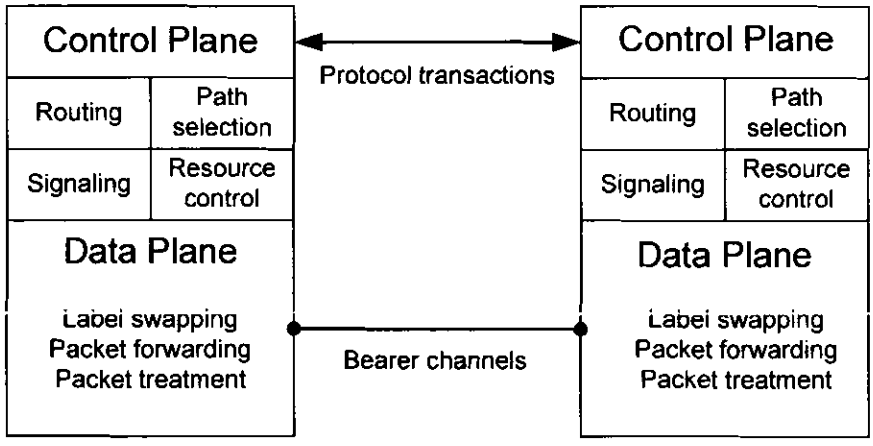


Figure 9 – The MPLS control and forwarding planes [10]

Figure 10 depicts a typical MPLS network topology. Routers at the edge of the MPLS domain are referred to as Label Edge Routers (LER's), while the core routers are Label Switching Routers (LSR's).

In a similar fashion to DiffServ, packet marking is performed at the edge of the network, by the LER's. Each incoming packet is classified and receives an initial label. The label assigns the packet to a Forwarding Equivalence Class (FEC). This assignment can be based on a variety of factors, such as the destination of the packet, its point of entry in the network, or the level of QoS it requires.

Generally speaking, packets belonging to the same FEC will receive the same label, and use the same path to traverse the MPLS network [2], [10]. This path is known as a Label Switched Path, or LSP. LSPs are set up by the MPLS control plane using the available routing information, but can also be created in response to a request from a flow requiring transport. An LSP is unidirectional, and thus at least two must be set up in order to carry duplex traffic [14].

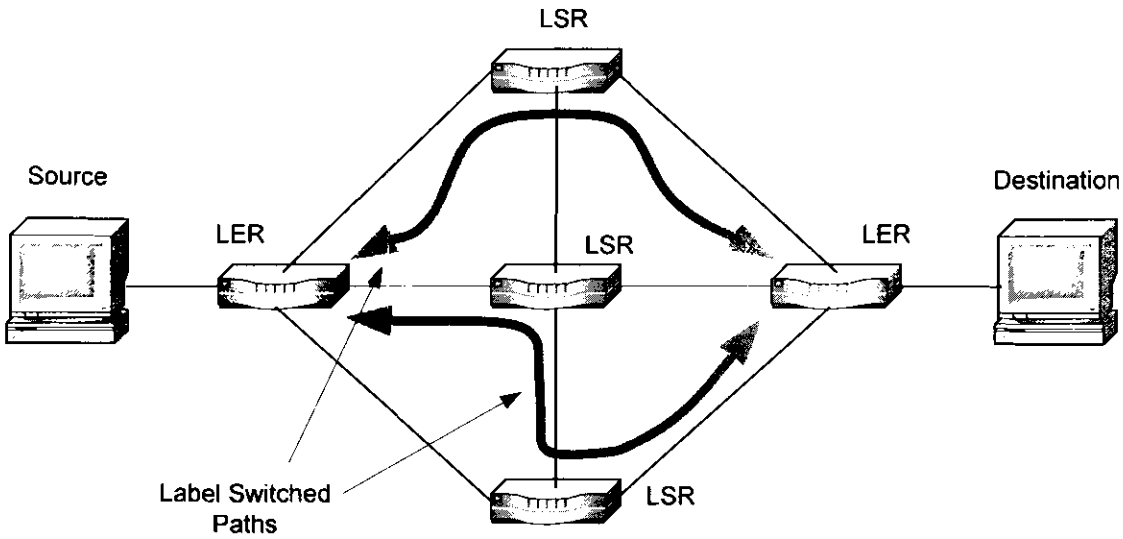


Figure 10 MPLS network topology

Figure 11 illustrates how the label swapping process takes place. The LDP is used to distribute the label values throughout the MPLS domain, and specifies the meaning of a label in terms of packet handling [14]. At each hop along the LSP the LSR checks the value of the incoming packet's label, and by simply performing a table lookup is able to determine the

destination of the next hop. Before forwarding the packet, the router performs label swapping, substituting the incoming label for a new one that will determine the FEC and path of the packet further downstream. When the packet exits the MPLS network, the label is removed and further forwarding takes place using conventional IP techniques.

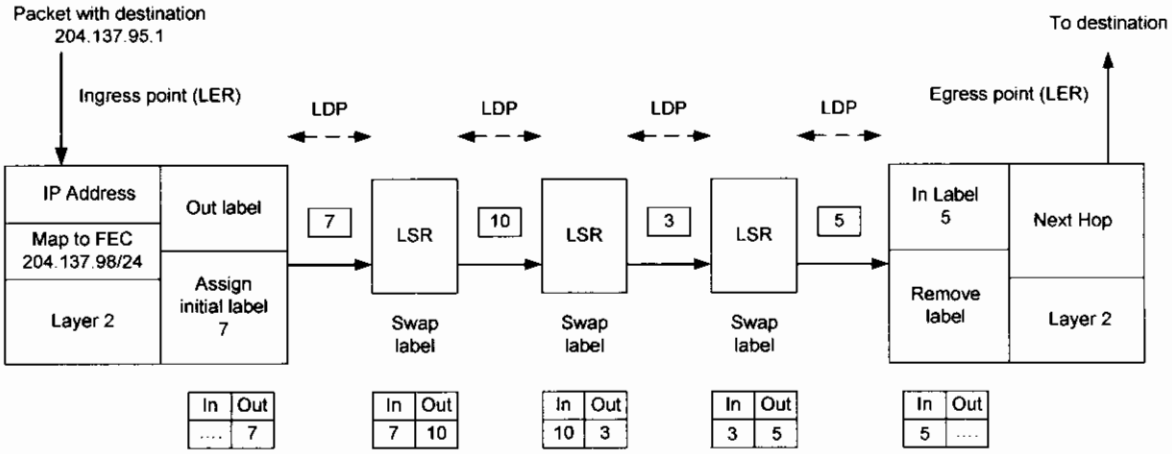


Figure 11 – Packet traversing an MPLS network [14]

It is possible to create two types of LSP. A *dynamic* LSP has a fixed entry point and destination, but the actual route taken by the packets can be dependent on several factors, such as the current state of the network. All this information is determined on a hop-by-hop basis by inspecting the label value. However, it is also possible to create a static or *explicit* LSP, with a path that is fixed at the LER where it originated. Traffic routed onto an explicit LSP will always follow the same path, making it equivalent to a virtual circuit. This is a very useful feature of MPLS and has many applications for QoS and traffic engineering, as we will discuss in the following sections [14].

A further important MPLS concept is that of traffic trunks. Traffic flows that receive the same MPLS label and thus belong to a single FEC can be aggregated and placed in the same trunk. A trunk is specified in terms of the characteristics of the traffic it carries, such as average throughput, expected maximum throughput, etc. All the flows in the trunk will receive similar treatment from the network. Traffic trunks are routable, and can be placed inside any LSP. Several trunks can be placed in the same LSP, or if preferred, a separate LSP can be created for each trunk (see Figure 12) [47].

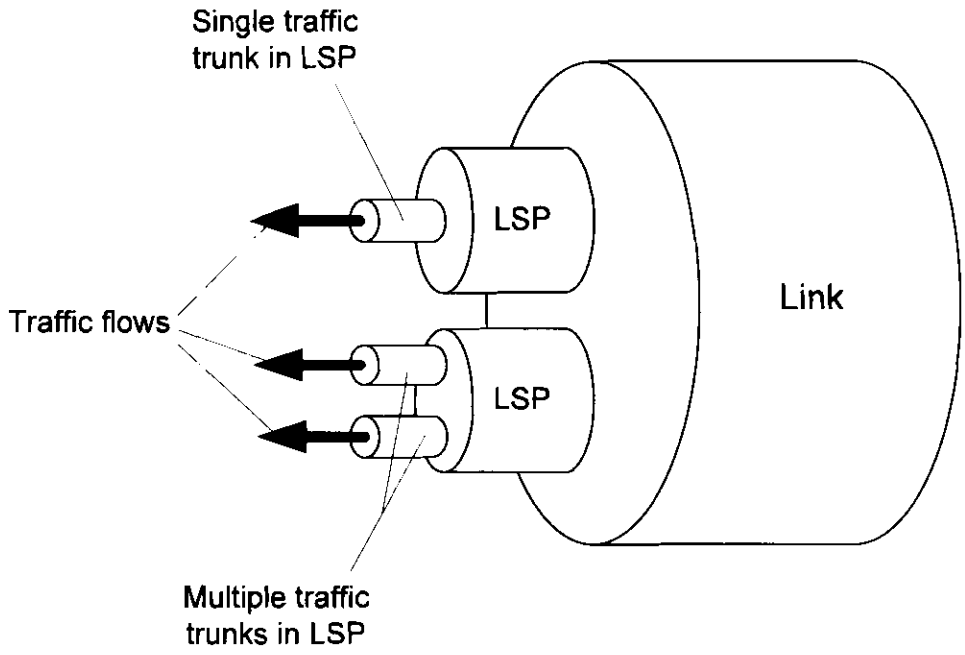


Figure 12 – MPLS traffic trunks inside an LSP [47]

2.4.3 Suitability of MPLS for traffic engineering

Traffic engineering can be defined as the process of controlling the flow of traffic through a network, with the goal to optimize its performance [10], [14]. One of the primary aims of traffic engineering is to avoid or relieve network congestion, which we briefly discussed in Section 2.1.2. IP networks are prone to congestion, since conventional IP routing protocols use algorithms like Dijkstra to compute the shortest path to a given destination, based on fairly simple metric such as distance or the number of links [14], [51]. This approach neglects to take resource availability or the current traffic characteristics into account, leading to some links being overlooked because they seem unsuitable to the protocol, while other links are over-used and thus become congested [10]. By employing traffic engineering, it is possible to place the traffic in the areas of the network that have the available resources to cope with it. This also simplifies the task of QoS provision.

IntServ, DiffServ and MPLS all have the capability to perform traffic engineering in some form. However, there are certain features of MPLS that make it particularly suitable for this task:

a) The separation between the control plane and forwarding plane inherent to the MPLS architecture is a great advantage for traffic engineering. The control plane has access to

information such as the available bandwidth of a link, and can base routing decisions on this information rather than the shortest path method of the IP routing protocols. The LDP then specifies the label values accordingly, causing the forwarding plane to forward the packets in the appropriate manner. This allows the best possible use of the available network resources [10].

b) MPLS is able to perform Constraint-Based Routing (CBR). This is done by adding extensions to the current IP routing protocols, enabling them to carry additional information regarding the available bandwidth of a link. When an application requests a path with a guaranteed amount of bandwidth, MPLS can then determine the best route to meet the request [10].

c) The ability of MPLS to specify explicit routes for traffic trunks and LSPs enables the creation of *tunnels*. A tunnel refers to both the traffic trunk and the LSP that carries it. Packets entering the tunnel will always follow the same route through the MPLS network [10], [14]. This can be useful for simply routing traffic away from congested areas. However, it is also possible to create a tunnel specifically for the use of a high-priority traffic flow like video. In this way, the more resource-intensive traffic can be explicitly routed to areas of the network with sufficient bandwidth to accommodate them, while the other flows can be assigned to links with less capacity.

2.4.4 Combining MPLS and DiffServ for QoS

In spite of the advantages offered by MPLS for traffic engineering and efficient resource management, it does have a drawback. Referring back to the necessary conditions for QoS stated in Section 2.1.3, we can see that MPLS satisfies the first condition and is able to guarantee bandwidth for traffic flows, by using CBR. However, MPLS is not designed to differentiate between different traffic types, and as such MPLS routers will not give preferential treatment to certain traffic regarding queuing or scheduling [10], [54].

One solution to the task of providing guaranteed QoS would be to combine DiffServ and MPLS, and in this way satisfy both conditions. This has led to the concept of DiffServ-aware MPLS traffic engineering [10], [54].

There are currently two methods for achieving cooperation between DiffServ and MPLS (see Figure 13). The first approach is to use the existing MPLS label as a reference to the FEC and its destination, while the 3 bits in the experimental (EXP) field of the MPLS shim header are used to specify a DiffServ behaviour aggregate and Per-Hop Behaviour. In this way, a single LSP can carry several different BA's, since the router can determine the required treatment of each packet by looking at the EXP bits. LSPs of this type are called EXP-inferred LSPs or E-LSPs.

The second option is to use the MPLS label value as a reference to the BA and PHB, while the EXP field only indicates the drop priority of the packet. An LSP will then only be used to carry one BA at a time, and is called a label-inferred LSP or L-LSP.

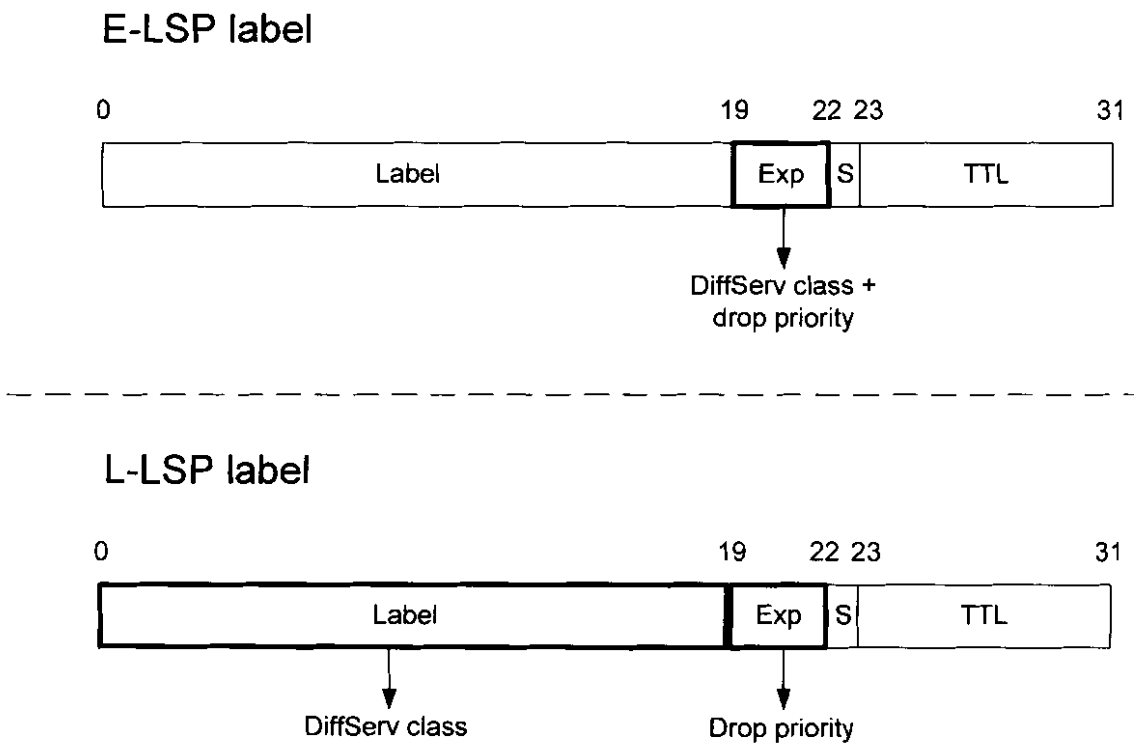


Figure 13 – The structure of the E-LSP and L-LSP headers [11]

Thus, by using the traffic engineering features of MPLS in combination with the class-based treatment of DiffServ, it is possible to guarantee both bandwidth and queuing priority for demanding traffic [54].

2.5 IEEE 802.11 wireless LAN

IEEE 802.11 WLAN was briefly introduced in Section 1.1. In this next section, we will discuss the standard in more detail, and further clarify the motivation behind the study.

2.5.1 The 802.11 WLAN standard

The original or legacy 802.11 standard was released in 1997 as a method to provide Ethernet networking over a radio link [59]. The specification covers the physical layer (PHY) and the Medium Access Control (MAC) sub-layer of the OSI protocol stack. The higher layers were left unchanged from the other 802.x LAN standards (see Figure 14). This facilitates easy interconnection between wired and wireless LAN's, since the higher layers cannot tell the difference between a mobile or stationary node. It also allows existing protocols such as TCP/IP to run in a WLAN environment [1].

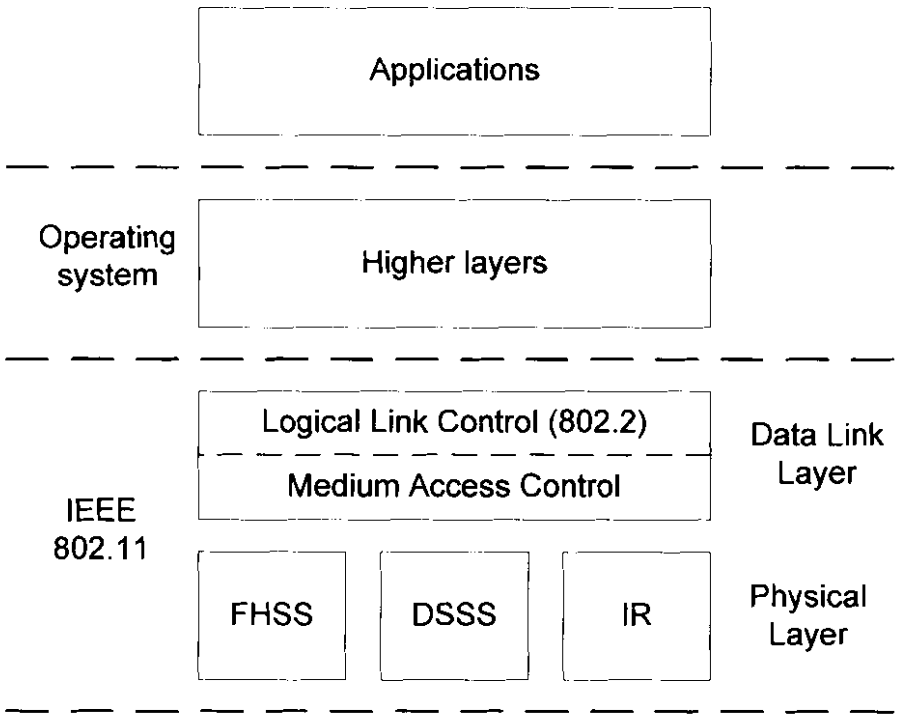


Figure 14 – The IEEE 802.11 protocol stack [21]

The 2.4 GHz band was chosen to deploy 802.11 WLAN. Three physical layer options were included in the original specification, these being Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DHSS), and infra-red (IR) [21], [56], [59].

FHSS uses numerous channels (up to 79) each with low bandwidth, and then changes frequencies every 0.4 seconds according to a pre-defined sequence. Although simple and effective, FHSS is limited to a maximum throughput of 2 Mbps.

DSSS divides the band into 14 channels, with the centre frequencies spaced 5 MHz apart. Many countries including the United States only allow the use of the first 11 channels. Since the channels are placed so close together, there is a fair amount of overlap between adjacent channels. Channels 1, 6 and 11 are the best choices to minimize interference from other transmissions (see Figure 15).

Although being specified in the original standard, the infra-red technique was never implemented. All three original PHY options were capable of data rates ranging from 1-2 Mbps.

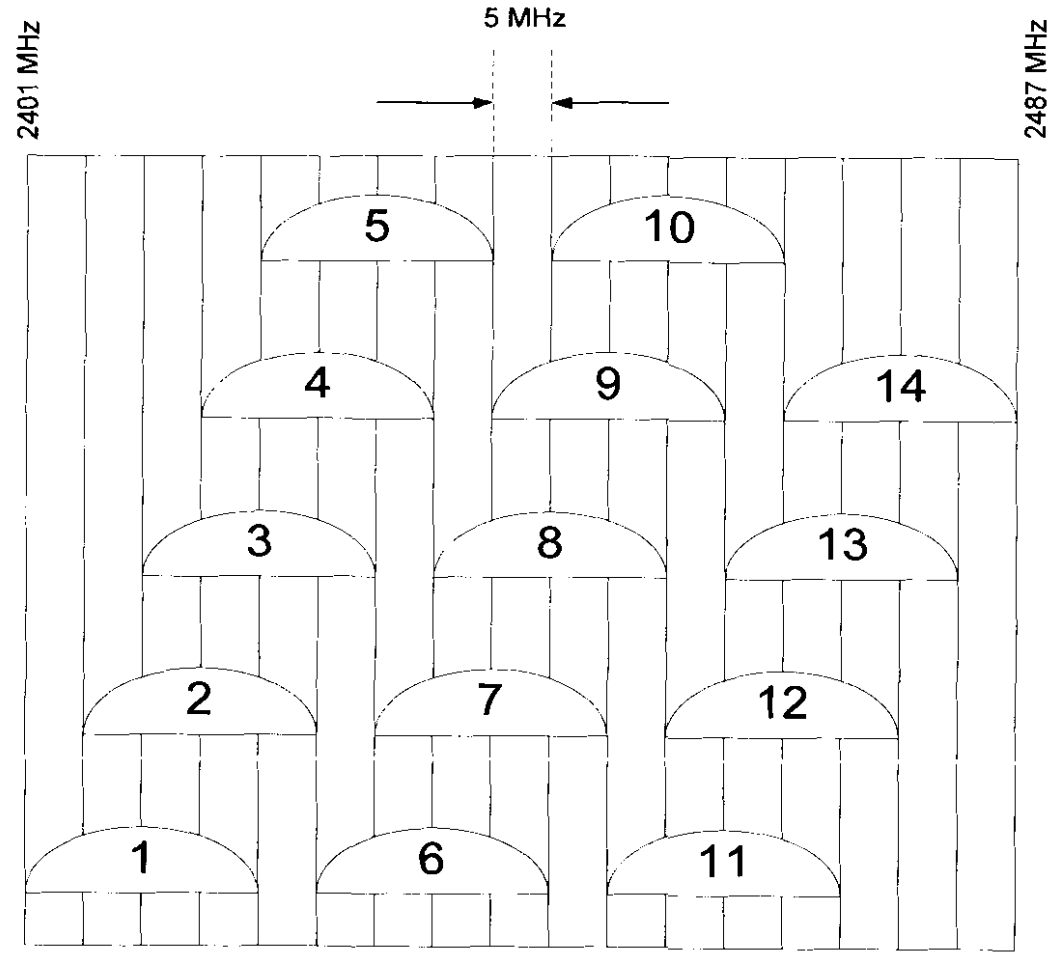


Figure 15 – Allocation of 802.11b channels [53]

The MAC protocol specified in the original standard is based on Carrier Sense Multiple Access (CSMA) used in 802.3 Ethernet. CSMA is designed to control access to a single shared transmission medium between multiple users, as is the case when using a radio link [21].

As mentioned earlier, the acceptance of the legacy WLAN standard was limited by its slow speed. To rectify this, the IEEE released the 802.11b specification in 1999. The MAC protocol remained relatively unchanged, but the PHY layer was enhanced to support data rates of up to 11 Mbps, using DSSS technology. Transmission range depends on the environment, but in typical deployments maximum throughput can be achieved over distances of 30 to 50 metres [59]. This increased throughput, combined with lower prices and the fact that 802.11b is backwards compatible with existing 802.11 DSSS systems, has made 802.11b very popular. It is the most widely deployed WLAN standard currently in use [56].

Additional extensions to the standard were added with the release of 802.11a. This version differs from the earlier versions in that it uses the 5 GHz frequency band. It also includes Orthogonal Frequency Division Multiplexing (OFDM) as a PHY layer modulation technology, and offers speeds of up to 54 Mbps. Nonetheless, 802.11a is much less common than the slower 802.11b, mainly because of the rapid adoption of the latter. A further disadvantage is that 802.11a is not compatible with 802.11b because of the significant physical layer differences between them. Also, transmitting at 5GHz limits the effective range of 802.11a to about one-seventh of that of 802.11b [59]

802.11g became available in 2003 and improves on many of the problematic areas associated with 802.11a. It also uses OFDM modulation and has a 54 Mbps data rate similar to 802.11a, but operates in the same 2.4 GHz band as 802.11b. Thus, b and g are compatible with each other, even though 802.11g has to lower its throughput when communicating with 802.11b terminals.

Numerous other proposed standards and task groups exist in the IEEE 802.11 working group, but these will not be discussed here. For a more complete list, see [56].

2.5.2 Basic operation of 802.11 Medium Access Control

Looking at topology, 802.11 WLAN is essentially a bus-based network. Each node connects to the radio medium, which in effect forms the network (see Figure 16). Since simultaneous transmissions taking place in the same frequency range tend to interfere with each other, only one node in the network is allowed to transmit at any given moment.

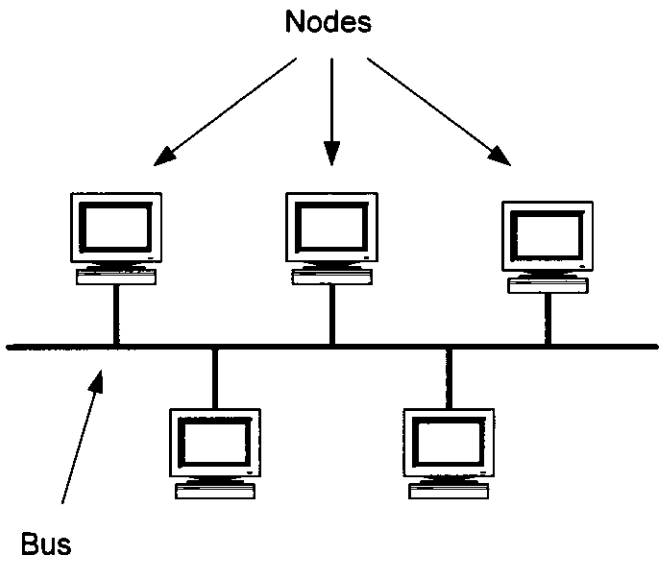


Figure 16 – A network using a bus topology

In order to control how the radio medium is accessed, 802.11 WLAN uses an access protocol known as Carrier Sense Multiple Access (CSMA), which is also employed in wired Ethernet networks. CSMA is a "listen before talk"-scheme that operates on the following principles, similar to a telephone conversation: [51]

- Listen before you start talking
- Don't talk at the same time as someone else
- Don't talk for long periods without giving the other person a chance
- If you start speaking at the same time as the other person, back off and wait for them to finish.

In practice, this means that the nodes in a network with CSMA will listen if the medium is available before attempting to transmit. If the link is being used by another node, the other nodes will wait until it finishes transmitting. This is commonly referred to as the Media Access Delay (MAD) [43].

In the event that two or more nodes begin to transmit simultaneously, the transmissions will interfere with each other, causing a collision. The data involved is lost and must then be re-sent. In wired Ethernet networks, the node continues to listen to the channel while transmitting. If it detects that the data present on the channel is different from what was originally sent, the node assumes that a collision has occurred and will then attempt to retransmit the lost data. This approach is called CSMA with Collision Detection, or CSMA/CD.

However, it is not possible to use CSMA/CD in a wireless environment. Because of the path loss associated with a radio transmission (see Ref [53]), the power level at the receiving end will be much lower than the transmitted power level. Thus a wireless node cannot transmit and listen to the channel at the same time. Consequently, 802.11 WLAN uses an adaptation of basic CSMA called CSMA with Collision Avoidance, or CSMA/CA. With this approach, the receiving node is required to send an acknowledgement (ACK) packet back to the sender, confirming that the data was successfully delivered. If the sender receives no ACK within a specified wait time, a collision is assumed and the data must be re-sent [3].

The legacy 802.11 standard defines two coordination functions used to provide access to the medium, namely the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF) [1]. We will now discuss both these in more detail.

a) Distributed Coordination Function (DCF)

DCF is the default access method used by 802.11 WLANs, and as such is mandatory in all networks using 802.11. DCF is distributed on every node and is designed for contention-based access, meaning that the nodes compete equally for a transmission opportunity. To this end, DCF uses CSMA/CA as described above.

A typical DCF transmission proceeds as follows: (see Figure 17)

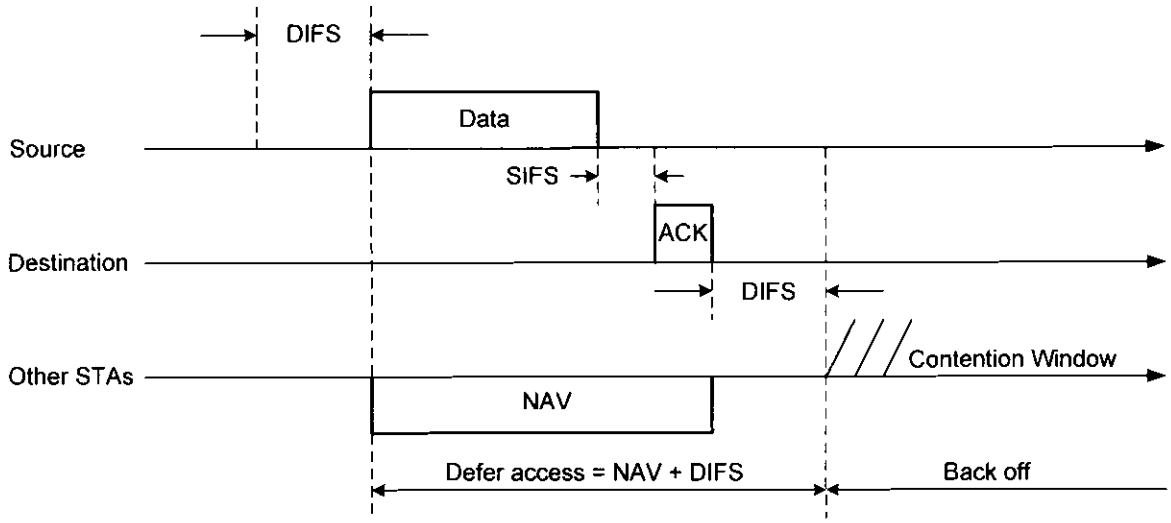


Figure 17 – DCF transmission [1]

A wireless node (henceforth called a station or STA) that has data to transmit must first sense whether the medium is available. This can be done on the PHY layer by measuring the signal strength and activity of other STAs, or on the MAC layer through virtual carrier sensing [1], [3], as we will explain shortly.

The medium must be free for at least a time period called a Distributed Interframe Space (DIFS), before the STA is allowed to transmit. As soon as this transmission begins, every other STA within range must remain silent until it completes. When the STA has finished transmitting its data, a further wait time occurs, known as a Short Interframe Space (SIFS), after which the receiving STA transmits an ACK packet to acknowledge reception of the data.

Thus each station reserves the channel for a total time of DATA + SIFS + ACK, as shown in Figure 17. This is called the Network Allocation Vector (NAV) and is updated at the beginning of every data transmission. This technique is called *virtual carrier sensing*, as it takes place at the MAC layer and is used to inform competing stations how long they will have to wait before trying to access the medium again.

At the end of the NAV, each STA wishing to transmit must then wait an additional DIFS, the purpose being to allow sufficient time for the ACK packet to be received by the STA that has just finished transmitting [3], [4]. This is important, since a retransmission will be triggered up to a preset retry limit if the ACK packet fails to arrive. In turn, other STAs will then again be prevented from transmitting their data.

The end of the DIFS wait time signals the start of the *contention window* (CW). Since there will probably be several STAs waiting to access the channel, they cannot be allowed to start transmitting simultaneously, because a collision would be inevitable. Thus, the contention window is divided into transmission slots. Each STA with data to transmit uniformly selects a random *back-off time*, according to the equation:

$$Back-off\_time = rand(0, CW) \times Slot\_time \qquad (1)$$

CW is the current value of the contention window, with a maximum value of 255 [4]. If a collision should occur, the current value of CW is doubled (up to CW_{max}) and a new back-off timer value is computed, to reduce the likelihood of any further collisions. After a successful transmission, CW is reset to a preset minimum value, CW_{min}. The slot time is dependent on the PHY layer properties of the channel, and the rand function selects a value in the interval (0, 1) according to a uniform distribution.

The back-off timer decreases for as long as the channel remains available. As soon as the timer runs out, the STA transmits its data. The other STAs will then freeze their timers, and continue the countdown once the channel becomes free for DIFS duration again. This ensures that every STA will at some point be allowed to access the medium.

An optional four-way handshake mechanism has been developed to improve the performance of basic DCF [1]. Instead of immediately transmitting its data upon gaining channel access, an STA first sends a request-to-send (RTS) packet to the intended destination. The receiving STA will then respond with a clear-to-send (CTS) packet if the surrounding area is clear of any other transmissions (see Figure 18). The RTS/CTS packets also specify how long the transmission will last based on the size of the data [18]. This allows the competing STAs to set their NAV values accordingly.

Since the RTS/CTS packets are small in size, a collision involving them will have a less significant impact than the collision of data packets, which can be a hundred times larger [1]. This mechanism is used to counteract the hidden terminal problem, which we will discuss in a later section.

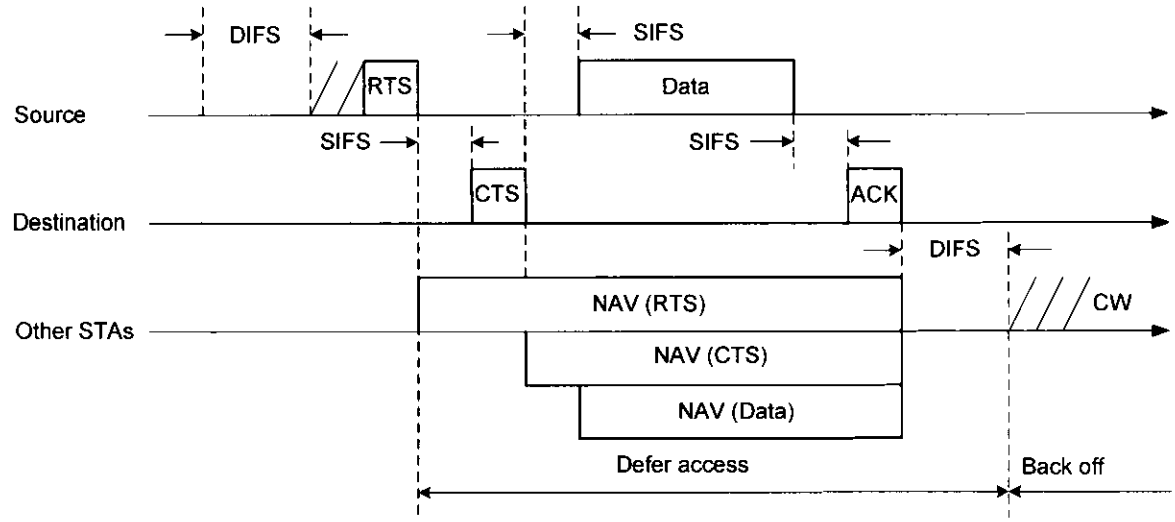


Figure 18 – DCF transmission with RTS / CTS mechanism [1]

b) Point Coordination Function (PCF)

Whereas the use of DCF is mandatory for al 802.11STAs, PCF [1] is an optional access method. Unlike the distributed DCF, PCF is a centralized polling scheme and requires the use of a central controller on one of the STAs. This controller, the Point Coordinator (PC) provides contention-free access to the wireless medium as opposed to the contention-based approach used in DCF. PCF was added in an attempt to support multimedia traffic in 802.11 WLAN [39]. By controlling the manner in which STAs are polled, it is possible to give a STA with real-time data priority access to the wireless channel.

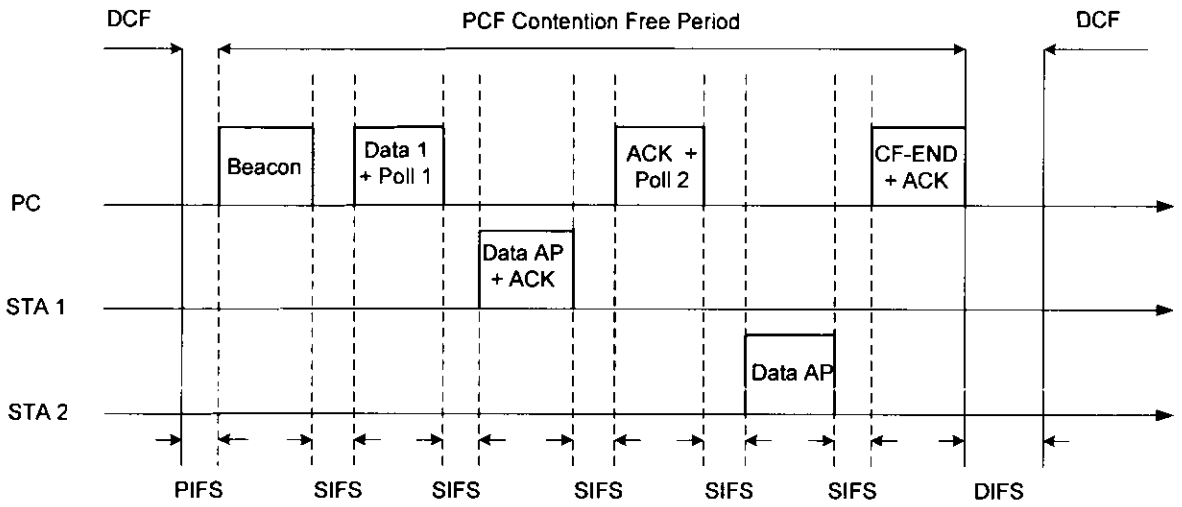


Figure 19 – PCF transmission [4]

Figure 19 shows the PCF access cycle, which consists of a contention-free period (CFP) and a contention period (CP). The PC waits for the medium to be idle for a time known as a PCF Interframe Space (PIFS). This is shorter than a DIFS and thus prevents any STA using DCF from gaining access before the PC. The PC first transmits a beacon frame, which informs all the STAs on its polling list of the start of the CFP period, and its expected maximum duration. This allows the STAs to set their NAV for the end of the CFP. The PC then polls the first STA on its list, asking it for data. Only the polled STA has permission to transmit, and all others must remain silent during this time. The polled STA responds after waiting for a SIFS with a packet that contains its data plus an ACK. After another SIFS wait time, the PC transmits a packet that polls the next STA and also sends an ACK to the first STA, acknowledging the reception of the data.

After all the STAs have been polled, the PC transmits a CF-END packet, signalling the end of the CFP. Each of the STAs will then reset the NAV and attempt to gain access using the default contention-based DCF. This contention period lasts until the PC transmits another beacon frame and the next CFP begins.

The contention-free access provided by PCF does offer certain advantages over DCF. A comparative study performed by Durbha and Sherman [64] confirms that PCF performs better than DCF in terms of throughput and delay for various traffic loads.

2.5.3 802.11 deployment

The IEEE 802.11 specification includes two different WLAN configurations: *Ad hoc* mode and *infrastructure* mode.

a) *Ad hoc mode* – An ad hoc 802.11 network is created whenever two or more WLAN STAs come within transmission range of each other [28]. The STAs communicate on a peer-to-peer basis, with no centralized control and no connection to any larger network backbone (see Figure 20). Ad hoc networks lack any type of infrastructure, since nodes are mobile and can join or leave the network at any time. Thus, an ad hoc network is very flexible and easy to deploy, since it doesn't require the use of wires and is self-configuring. This is ideal for quickly establishing communications in a situation where it wouldn't be feasible to set up a permanent network.

Current and possible future applications for ad hoc networks include disaster relief networks, construction site communications, inter-vehicle networks, wireless home and office networks, military communications, etc [23]

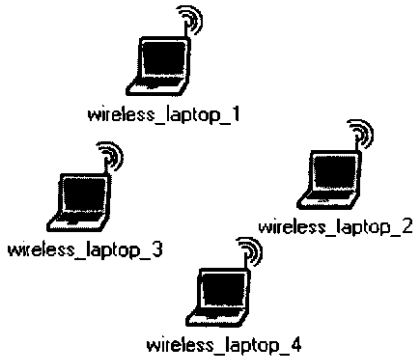


Figure 20 – A typical ad hoc network

b) *Infrastructure mode* – In infrastructure mode, one of the nodes in the network takes on the role of an Access Point (AP). This can be an STA, or a dedicated AP such as a wireless router. The AP is similar to the central node in a star topology network, since it connects to all the other STAs in the network. A group of STAs and their associated AP are referred to as a Basic Service Set (BSS) [64] (see Figure 21).

Instead of transmitting directly to each other as in ad hoc mode, all the communications in an infrastructure WLAN must go via the AP [1]. In the case of a network with PCF deployed, the point coordinator would be located on the AP. Thus in order to form part of a BSS, an STA must be within range of the AP.

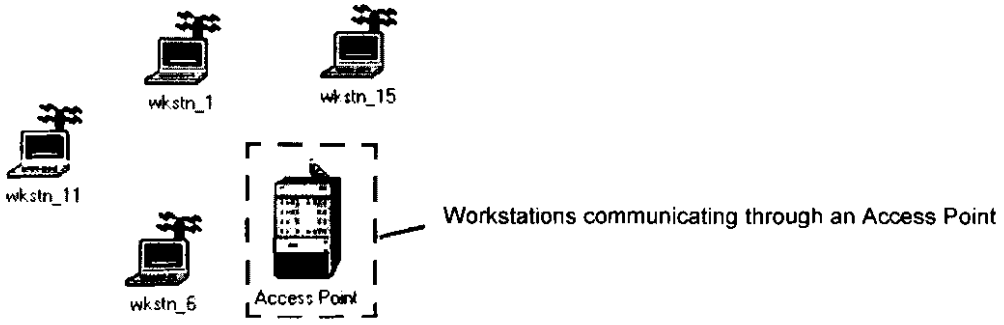


Figure 21 – A wireless LAN Basic Service Set (BSS) [71]

It is possible to set a WLAN up in infrastructure mode purely for communications within the BSS. More often however, the AP is used to connect the WLAN BSS to a larger network infrastructure (see Figure 22). A good example of this would be wireless "hot spots" that are commonly found in hotels and airports, where the AP is used to provide Internet access to anyone with a compatible wireless-capable computer.

The 802.11 standard also defines an Extended Service Set (ESS) for intra-BSS communication. In this case, the AP of each BSS connects to a common distribution network or backbone, such as a wired Ethernet network [64]. In this way, STAs from different BSSs can communicate even though they are out of wireless transmission range.

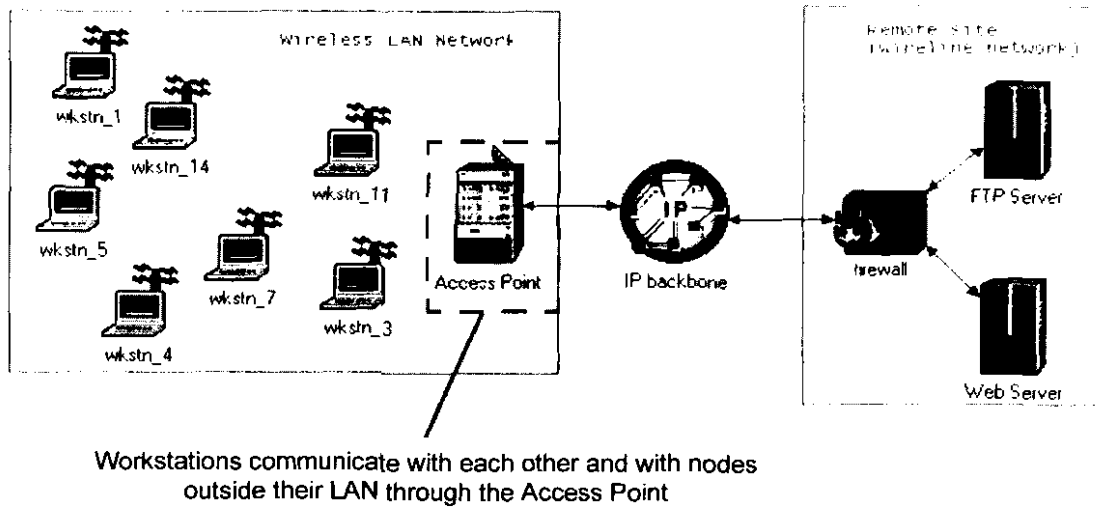


Figure 22 – An infrastructure WLAN connected to an IP backbone [71]

2.5.4 QoS and 802.11

As mentioned previously, IEEE 802.11 continues to be the most widely-accepted WLAN standard, with a growing market and a variety of uses. 802.11 offers many advantages: it is robust against failures, flexible, easy to deploy, and has become a cost-effective wireless solution [1]. Real-time applications have proliferated in the wireless environment, to the extent that 802.11 is expected to be the leading VoIP market by 2006 [37].

However, the major flaw of 802.11 WLAN still remains, namely a lack of QoS support. To a large extent, this can be blamed on the fact that the MAC scheme of the 802.11 standard was derived from wired Ethernet as used in IP networks. IP was never intended to provide QoS, even though in theory the Type of Service (ToS) field in the IP header can add service differentiation. In practice, most routers just ignore the ToS field and treat all packets with a single service level [51], unless an IP QoS technique such as DiffServ is deployed. At any rate, the bandwidth of a wired link is such that QoS poses less of a problem [1].

This is not the case in a wireless network. A wireless channel has very different characteristics from a wired one, and as such behaves in a dissimilar fashion. We highlighted the QoS requirements of real-time traffic in Section 2.1.2, and unfortunately 802.11 WLAN cannot currently support those requirements [5]. We will now describe the QoS shortcomings of both the PHY layer and the MAC sub-layer.

a) QoS difficulties of wireless communication

Ironically, the greatest benefit of a wireless link also brings about several disadvantages. Using the air as a transmission medium greatly increases the mobility of the wireless device, but the wireless medium is much less reliable than its wired counterpart. The following are the main problems inherent to a wireless transmission.

i. *Path loss* – The limited range of a wireless broadcast is due to path loss [53]. As the radio wave travels further from the source, it spreads out and becomes weaker. Eventually, the wave decays to a point where the information it carries can no longer be identified. The higher the frequency, the more severe the path loss becomes. Since 802.11 uses frequencies in the GHz range, it suffers a high degree of path loss.

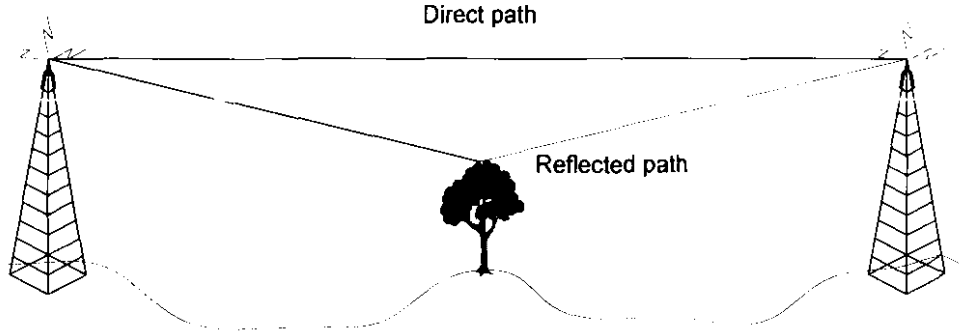


Figure 23 – Radio waves reflected by obstacles [53]

ii. *Multi-path interference* – Like other electromagnetic waves, radio waves are reflected by objects in their path (see Figure 23). This reflection will also travel towards the receiver, but it will arrive with a displacement in time and phase, due to having travelled a longer path. This will cause the received signal to become distorted or at worst, the waves will completely cancel each other if they are a half-wavelength out of phase [53] (see Figure 24).

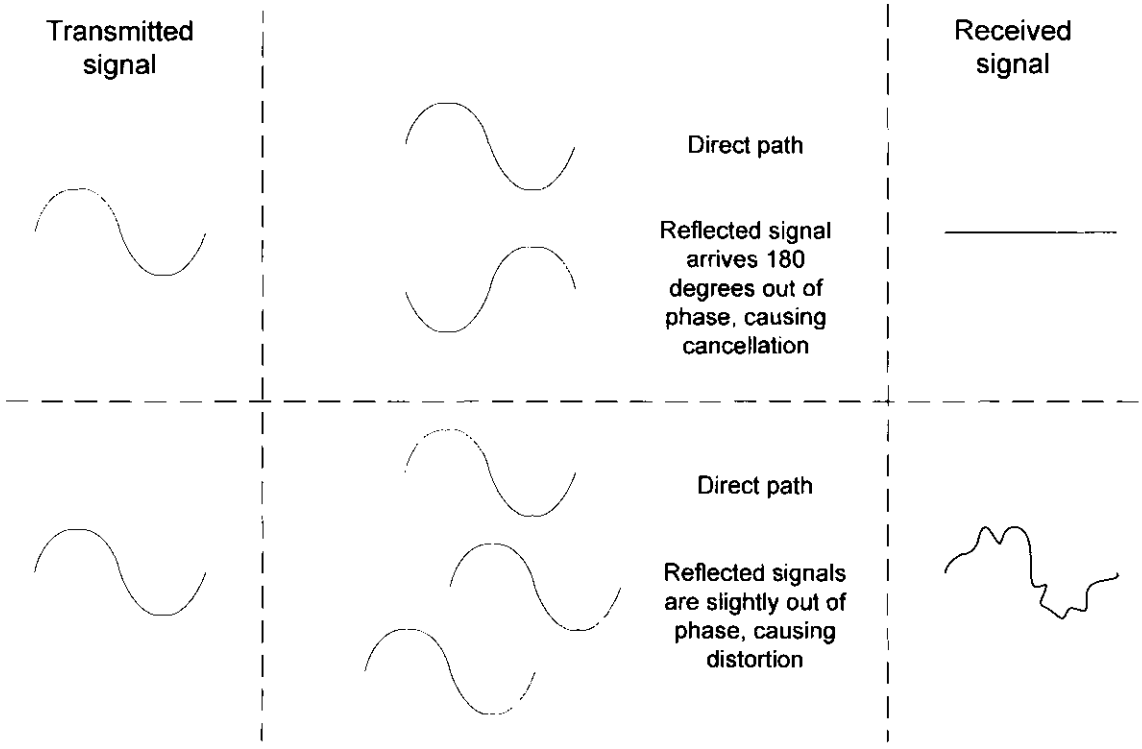


Figure 24 – Signal distortion caused by multi-path interference [53]

iii. *Jamming and signal interference* – As mentioned previously, the radio channel is a shared medium, and as such two transmissions of the same frequency will interfere with each other. Since the majority of 802.11 WLAN devices use the 2.4 GHz band, they incur interference from all other devices using the same frequency range. These include microwave ovens, cellular phones, and Bluetooth devices [56].

iv. *Limited resources* – The radio spectrum, which stretches from 9 kHz to about 300 GHz, unfortunately does not have limitless capacity. There are strict limitations on the amount of bandwidth available for a particular technology. For example, 802.11b can legally transmit between 2.4 and 2.487 GHz [53]. This limited bandwidth translates to lower throughputs compared to a wired link.

v. *Varying channel conditions* – The unrestricted mobility offered by a wireless device has a negative side-effect. As the user moves around, the length and conditions of the transmission path also change. This in turn affects parameters such as delay and jitter, and can cause unexpected signal loss, making it very difficult to route packets based on the state of the link [17].

All the factors mentioned above contribute to the unreliability of a wireless link. This increases the probability that a packet will be lost or corrupted as it travels to the destination. The error rate of a wireless channel has been measured at a thousand times greater than that of a wired link [1]. Since the 802.11 MAC retransmits any lost or corrupted packets, this high number of errors causes long delays, which real-time traffic cannot afford.

b) Limitations of the 802.11 MAC functions

i. *DCF* – DCF was designed to provide fair access to the transmission medium using CSMA. Unfortunately, this means that DCF is only able to support a best-effort service. An STA with real-time data to transmit competes for access on equal footing with every other STA, and must wait until it gains access to the medium. If there are several STAs present, the delay could conceivably be very long; rendering the data useless by the time it is delivered. Also, after the STA has managed to transmit, there is no way to predict when it will be able to gain access again, since the back-off time is chosen randomly. This of course causes variable delay times, resulting in jitter. Thus, DCF cannot provide QoS since it cannot distinguish between different traffic flows. This is clearly illustrated by simulations described in [1], which shows that the throughput and delay of all traffic flows are equally affected by increasing traffic volume.

A further problem associated with the CSMA protocol concerns the collision avoidance scheme. DCF is supposed to minimize collisions since an STA is only allowed to transmit if it senses that the channel is free, but consider the scenario in Figure 25:

Suppose both STA A and STA C want to transmit to STA B. Both can detect B, but due to the limited range of the wireless medium they cannot detect each other. Thus, when sensing the channel, both A and C will assume the medium is free and begin to transmit, causing a collision at B. This is called the *hidden terminal problem* [39].

This problem can be partially solved by using the RTS-CTS technique described in the previous section, but even then collisions are still possible given the right circumstances [39].

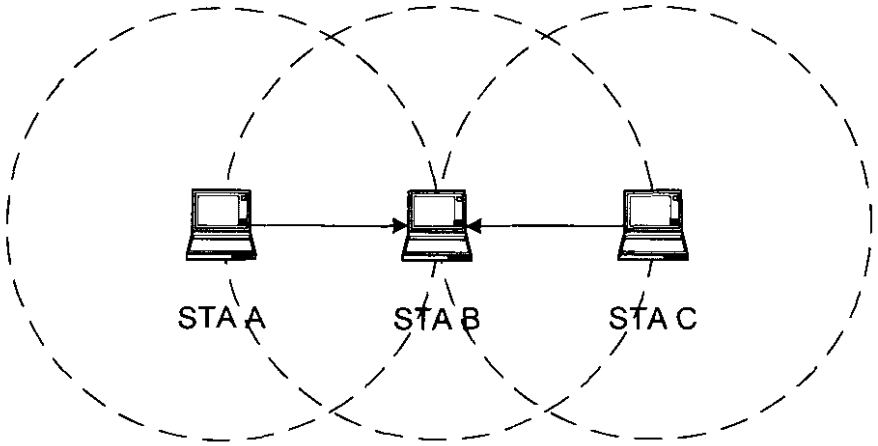


Figure 25 – The hidden terminal problem [39]

ii. *PCF* – Despite using a polling scheme in addition to the usual contention-based access, PCF also has limitations in providing QoS [1]. There are several reasons for this. Firstly, the central polling method is inefficient and wastes bandwidth, since all transmissions between STAs must go through the access point which contains the point coordinator, causing a bottleneck. When the network load becomes high, the AP may have trouble coping with all the incoming traffic, leading to long delays or packet loss. Also, if the AP should fail, the network will fail along with it.

A second difficulty associated with PCF is the use of the beacon frame that denotes the start of the contention-free period. In theory, a polled STA should be able to access the channel during the CFP without having to contend. However, since part of the PCF access cycle also includes a DCF period, the channel may be busy when the point coordinator attempts to transmit the beacon. This delays the start of the CFP, and hence also prevents the PC from granting access to STAs with high-priority traffic [1].

Lastly, there is no way to guarantee consistent delay times, even when polling all the STAs in a round-robin fashion. This is due to the variation in the length of the data a polled STA is allowed to transmit, which may be anything from 0 to 2346 bytes [1].

The variable channel conditions further compound this problem, making it impossible for the PC to predict how long a given STA will take to complete its transmission. Hence the PC is unable to ensure that a high-priority traffic flow is delivered with regular delays between packets.

2.6 Summary

In this chapter, we took a more in-depth look at the main topics pertaining to this study. We formulated a definition for QoS and discussed the major QoS parameters that will be used when conducting the simulations, namely bandwidth / throughput, delay, jitter, and packet loss.

We also examined several architectures designed to provide QoS in IP networks. Of these, Integrated Services was found to be unsuitable due to its complexity and resource requirements. Differentiated Services offers a simpler solution but with less stringent QoS guarantees. Multi-Protocol Label Switching is fast, has little overhead, and facilitates traffic engineering. MPLS and DiffServ can also be combined to satisfy both the necessary conditions for QoS, by employing DiffServ-aware traffic engineering.

Lastly, we described the various aspects of the IEEE 802.11 wireless LAN standard. Despite being simple and easy to deploy, 802.11 still suffers from an inability to provide QoS for multimedia traffic, for example video and VoIP. This is mainly due to the unpredictable wireless transmission medium, and the limitations of the DCF and PCF methods used to provide access to the wireless medium.

In the next chapter, we will use the insight gained from the literature to formulate and explain our approach to the stated research topic.

3. Method

It is important to place our work in a proper perspective regarding previous efforts of a similar nature. To this end, we review some of the research already done in the field of IP QoS as well as QoS in 802.11 and other wireless networks. Following that, we will describe our proposed solution and research methodology in detail. This chapter also includes an overview of the OPNET Modeler simulation package, and a complete account of each simulation scenario. We give particular attention to the chosen network topology, the assumptions and limitations of the simulation setup, and the reasoning behind our choice of simulation parameters.

3.1 Previous work

3.1.1 802.11 QoS research

A glance at the available literature will reveal that adding QoS to 802.11 WLAN is a very broad area of research. An exhaustive list of the current research activities falls outside the bounds of this study, and thus we briefly discuss some of the most recent developments in the field. The interested reader may refer to three very comprehensive QoS surveys included in the list of references. We list them here since they are useful for classifying 802.11 QoS approaches.

Ni, Romdhani and Turletti [1] focus on schemes to add service differentiation to 802.11 WLAN. These are aimed at the MAC sub-layer, can be applied either per-STA or per-flow, and are based on the existing DCF and PCF access methods. Each scheme proposes a method of enabling the 802.11 MAC to differentiate between traffic flows of different priorities, and thus ensure QoS for more demanding traffic. Also included are simulations that compare the performance of legacy 802.11 to the upcoming 802.11e standard, designed to add service differentiation and QoS to 802.11 WLAN.

TB Reddy et al [17] survey QoS problems and solutions in ad hoc wireless networks. They group wireless QoS solutions into two categories.

The first category is based on the approach used for QoS provision, while the second is based on the OSI protocol layer where the solution is deployed.

A further survey, also concerning ad hoc networks, is that done by Kumar et al [39]. They provide a discussion of the various MAC schemes used in ad hoc wireless networks, both contention-free and contention-based. This includes the legacy 802.11 MAC functions, as well as several proposals for MAC functions that are QoS-aware.

The shortcomings of the 802.11 MAC functions for QoS provision have been well documented, and consequently a fair amount of work has been done towards improving their performance. The main focus of the MAC layer research is to enable service differentiation and thus provide QoS for real-time traffic.

Aad and Castelluccia [3] propose four different ways to modify the DCF access method, and test them with Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic flows. Firstly, the contention window (CW) value is scaled according to priority, which works well for UDP traffic but not for TCP. Secondly, the CW_{min} value is modified, which improves the TCP performance but is less effective in prioritizing the flows. A third scheme is to assign different DIFS values to each STA according to priority. This works for both types of flow and ensures strict priorities. The last technique is to limit the maximum frame size of each STA depending on priority. This method is less complex and works well for both TCP and UDP traffic, but is less effective in a noisy channel than the DIFS scheme. The authors recommend the DIFS-based differentiation as the most effective of the four mechanisms.

Grilo et al [4] evaluate DCF and PCF performance in 802.11b and state that DCF is unable to guarantee QoS even with enhancements. They propose a scheduling mechanism called SETT (Scheduling based on Estimated Transmission Times) which accounts for the maximum delay of a transmission in order to compute a polling interval for each STA, and also distributes bandwidth fairly among STAs. SETT is compared to Weighted Deficit Round Robin (WDRR) for bursty traffic, VoIP and video conferencing. SETT performs better in most scenarios since WDRR does not guarantee delay times or prioritize traffic flows.

Al-Karaki and Chang [13] developed an improved version of PCF, called EPCF (Enhanced PCF) which they compare to the Hybrid Coordination Function (HCF) used in the 802.11e standard. The two are shown to have similar performances, but according to the authors their proposed technique is simpler and easier to implement in 802.11, since it makes use of the existing PCF.

Sundaresan et al [20] evaluate the performance of DCF in wireless ad hoc networks. They conclude that 802.11 MAC is inefficient in an ad hoc environment, and propose a new medium access scheme called Flow Based Medium Access (FBMA). FBMA operates on a per-flow basis, and uses different CW sizes as well as prioritized queuing to ensure fair access for different flows. Simulations show that this approach improves on the performance of CSMA/CA.

Yin et al [25] developed an analytical model to predict the performance of DCF based on such factors as number of STAs, traffic load, CW size, channel conditions, error rate, and packet size. They use the model to determine the optimum packet size and CW value to ensure the minimum number of errors. Smaller packets are less error-prone but waste channel resources, while large packets are more likely to encounter bit errors. Consequently, the choice of packet size depends on the channel conditions.

Aad et al [27] propose a change in the computation of CW values. Instead of resetting CW to its minimum value after a transmission, CW is slowly decreased to avoid the occurrence of future collisions. This reduces the amount of retransmissions and thus increases throughput while reducing delay and jitter. Simulations show comparable performance to legacy 802.11 under normal conditions, and improved performance in congested environments.

Yin and Leung [35] attempt to address the problem of reduced performance in an infrastructure WLAN due to all transmissions going through the AP. They propose AHADC, or ad hoc awareness direct connection. AHADC deliver intra-BSS packets directly as in an ad hoc network, thus reducing the traffic load on the AP. Simulations show that if a large portion of traffic is intra-BSS, this scheme reduces packet delay and increases the system throughput.

An analytical model to determine delay and jitter in an ad hoc network using DCF is presented by Carvalho and Garcia-Luna-Aceves in [44]. They use this to analyse the impact on delay of the initial CW value as well as the size of the network, with the RTS/CTS scheme enabled. Results show that a higher initial CW reduces both delay and jitter in a large network. In the case of packet size, smaller packets are found to correspond to lower delay and jitter.

Xiao [45] studies priority schemes for legacy 802.11 and 802.11e that make use of the back-off feature. The relevant parameters are initial CW size, the retry limit, and the window-increase factor. The author concludes that a small retry-limit is more suited to real-time traffic, since retransmitted packets will likely be too late to be useful. For non-real-time data, the retry limit needs to be larger to encourage reliable packet delivery. By differentiating the CW size and the window increase factor, it is possible both to reduce collisions and give different priorities to traffic types.

3.1.2 Research in IP QoS

This section takes a look at some of the existing research in IP QoS, with particular emphasis on DiffServ and MPLS.

Gyires and Wen [8] present an MPLS extension for IP traffic engineering in networks with long-range dependent traffic. It is designed to avoid the shortest path calculated by the IP routing protocols and instead route the traffic to less congested parts of the network. LSPs are automatically set up to improve the utilization of each link and to minimize delay, buffer usage, and packet loss. According to simulation results, the extension is effective in achieving this.

Kimura and Kamei [11] evaluate DiffServ-aware constrained-based routing (CBR) schemes for MPLS networks in terms of their ability to provide QoS. Two DiffServ classes, namely EF and best-effort, are used in the study. The algorithms for CBR computation are called *widest-shortest* (W-S), which chooses the path with the maximum available bandwidth, and *shortest-widest* (S-W), which chooses the shortest available path. The following four combinations are evaluated: W-S with EXP-inferred LSP (E-LSP), W-S with Label-inferred LSP (L-LSP), S-W with E-LSP, and S-W with L-LSP.

They also propose a fifth scheme, which uses S-W to compute paths for the best-effort traffic and W-S to compute paths for the EF-class traffic. Results show that there is no difference in the performance of E-LSPs or L-LSPs if the same CBR algorithm is used. In terms of delay and bandwidth, the proposed scheme performs at least as well as the first four.

Trimintzios et al [19] investigate the use of DiffServ and MPLS as QoS technologies in IP-based production networks, for example the communications between a main film studio and the mobile broadcasting studios. The authors develop a management and control system designed to meet the required SLA's and optimize the use of network resources. This system is based on the concept of a provisioning cycle, which is divided into several forecasting periods. Each forecasting period corresponds to a different network configuration, based on traffic forecasts and previously measured traffic data. DiffServ-aware MPLS traffic engineering is then employed to avoid network congestion, based on the expected traffic profile. This solution is intended to be cheaper and more flexible than the dedicated high-capacity networks currently in use for production traffic.

Shi and Mohan [33] investigate the use of flow distribution and flow splitting in MPLS networks. Flow distribution means that one LSP is used to carry an entire aggregated traffic flow, whereas flow splitting divides the same flow among several LSPs. The authors propose an algorithm that combines the two techniques to perform traffic engineering. Simulations show that this approach is effective in avoiding bottlenecks and improving the utilization of the network resources.

Fowler and Zeadally [43] propose a technique called Adaptive Queue Management (AQM) for mobile MPLS-based networks. Multimedia flows are isolated from the other traffic, and a threshold is set for the maximum allowable amount of multimedia traffic. This threshold is automatically adjusted based on the current state of the network. Factors such as the queue length at the routers and the media access delay (MAD) are taken into account. The simulations indicate that this approach decreases the amount of dropped packets since queue lengths are shortened, and also makes maximum use of the network resources.

Sun et al [47] analyse the impact of using MPLS traffic trunks in networks with both TCP and UDP traffic over links with limited bandwidth. Without MPLS, UDP traffic tends to dominate, since TCP reduces its transmission rate if it detects congestion in the network.

Mixing TCP and UDP on a single traffic truck in the same LSP has the same effect. The optimal solution is shown to be a separate trunk for each protocol, placed in different LSPs, which prevents the UDP traffic from interfering with TCP.

3.1.3 Summary

For the purpose of choosing a suitable approach to this study, we had to consider the current state of affairs concerning wireless networks and IP QoS, as well as likely future developments in this area. The work summarized in the previous sections provides only a glimpse of the current research efforts in these fields. Even so, it is clear that QoS provision in 802.11 WLANs remains a topic of great interest, with a wide variety of different approaches and proposed solutions. Likewise, a considerable amount of research is still being done on the use of DiffServ and MPLS, particularly concerning ways to avoid congestion, improve resource utilization, and provide real-time application QoS.

3.2 Proposed approach

3.2.1 Outline and motivation

The vast majority of the reviewed literature on 802.11 WLAN is concerned with adding some form of service differentiation to the 802.11 MAC layer, as a means of QoS provision. Consequently, we will not attempt to add to the amount of work already done in this particular direction. In addition, most of the existing research involves ad-hoc or intra-BSS communication between wireless STAs. Infrastructure-based WLANs that connect to a larger infrastructure are a less documented case. However, according to [35], the majority of WLANs are deployed in this manner.

Palmeri [2] specifically highlights this increasing convergence between fixed and wired networks, and also mentions that the continuing growth in mobile and wireless services may have an impact on the current fixed network infrastructure. Consider as an example the Next-Generation Network (NGN) architecture, which was briefly mentioned in Section 1.1. The NGN will be comprised of an IP-based core network, and several edge or access networks that provide coverage to the users.

It is expected that IEEE 802.11 WLAN will be a significant access network in the NGN, since WLANs are increasingly being used to supplement and even replace existing wired LAN deployments [35].

For this study, we propose to use a hybrid topology that includes both wireless access networks and a fixed backbone network connecting the access networks. This allows us to investigate certain aspects of combining wired and wireless communication in a single network infrastructure. Of particular interest is how the current lack of QoS in many IP networks will influence the level of QoS experienced by the user located in the wireless access network. The IP QoS techniques described in Chapter 2 will then be deployed in the core network to determine how they affect the access network in terms of QoS provision.

The level of QoS will be determined by sending different types of application traffic between the wireless access networks via the core network, and then measuring QoS metrics such as delay and jitter. In addition, other statistics such as link utilization, IP packet loss, or WLAN throughput can be gathered at strategic points in the network. These measurements will provide quantitative data that will enable us to determine the effectiveness of each QoS technique, as well as information regarding the interaction between the wired and wireless sections of the network.

3.2.2 Choice of QoS techniques

To select a suitable QoS technique for the study, we refer back to the IP QoS approaches we discussed in Chapter 2. For use in conjunction with a wireless access network, the requirements for IP QoS provision are [16]:

- Differentiating among applications and providing each with the needed QoS levels. Multimedia traffic in particular requires predictable network behaviour.
- Controlling delay, jitter and packet loss at the network layer to maintain performance.

IntServ is able to provide QoS with both class-based treatment and resource reservation. However, it is complex and adds significant overhead due to the amount of management traffic needed to keep track of all the flows. This drawback has limited the real-world deployment of IntServ, and also renders it inappropriate for use in a low-bandwidth setting like the one used in our study.

DiffServ is simpler and less demanding on the network than IntServ, and thus better suited to our approach. By aggregating traffic flows into DiffServ classes, each flow can receive forwarding treatment according to its requirements. But as we mentioned previously, DiffServ only determines the scheduling of the traffic at each node, and doesn't concern itself with *where* the traffic is sent. Consequently, all traffic flows may be forced to share the same link, which in a network with low capacity will cause congestion, packet loss, and long delays.

Of the available QoS approaches, MPLS seems the most appropriate choice for our purposes. The literature contains several proposals for wireless MPLS deployment, most notably [2], [16], [41], and [43]. These references mention several key advantages offered by MPLS that make it ideal for use in a wireless access network:

Guaranteed bandwidth provision – MPLS Label Switched Paths (LSPs) can be used to create an end-to-end tunnel with a guaranteed amount of resources for demanding flows such as video.
Service differentiation – Although this is not by default a feature of MPLS, different traffic types can be marked with different labels according to the QoS treatment they require.
Traffic engineering – The ability to specify explicit routes means that MPLS can make efficient use of the available network resources, by routing traffic onto less congested links to avoid delays.
Improved mobility and reliability – Dynamic LSPs can be used to rapidly re-route a connection in the case of a node or link failure.

The above factors are well suited to our stated approach, since MPLS is able to use the available resources to maximum effect. This can best be illustrated by making a quantitative performance comparison between several techniques. Thus we will use both DiffServ and MPLS in our simulation.

3.2.3 Architecture of our approach

Figure 26 shows the generalized layout of the network to be used in evaluating the QoS techniques. It consists of several edge or access networks that communicate with each other via the backbone or distribution system (DS). The edge networks are 802.11 WLANs that access the backbone via an AP.

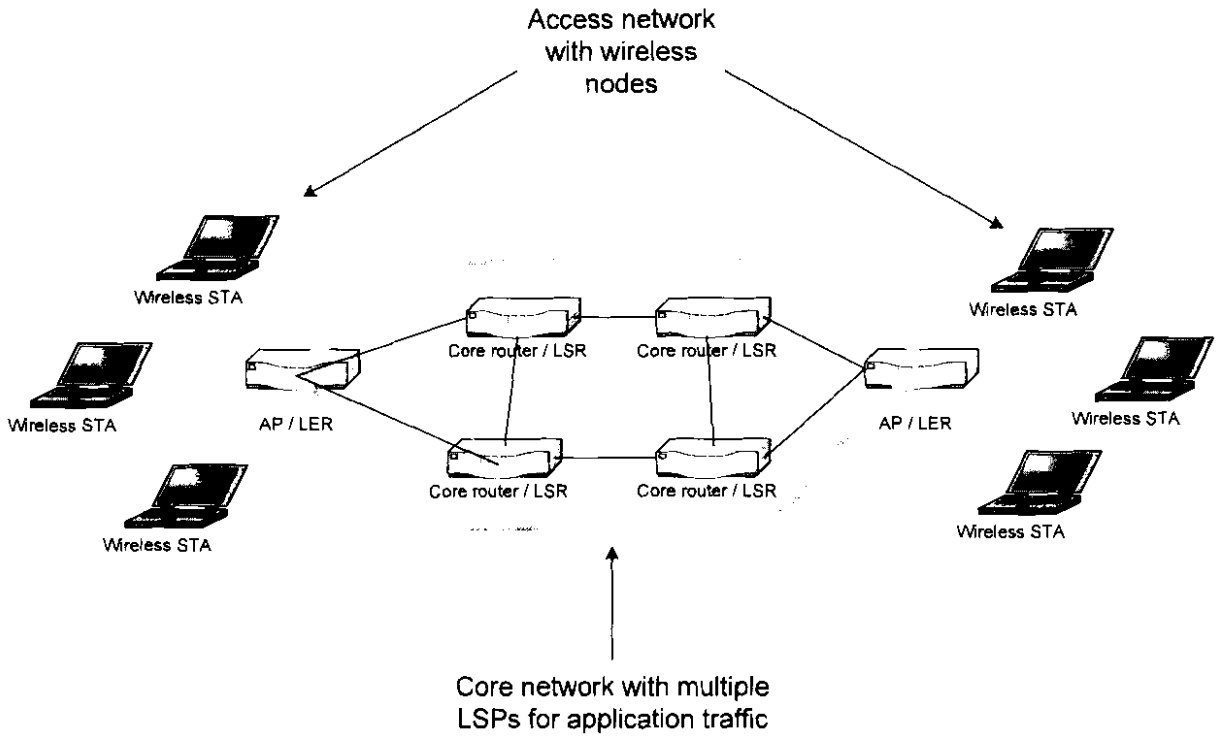


Figure 26 – General network layout of our approach

Architectures for deploying MPLS in networks similar to this one are given in [2] and [41]. The STAs transmit their data to the AP, which acts as the MPLS Label Edge Router (LER). It is here that the traffic flows are mapped into forwarding classes and assigned to an LSP. MPLS is also deployed on the intermediate core routers, which perform the function of Label Switching Routers (LSR), and perform packet forwarding and label swapping. Figure 27 illustrates the function of each node type in terms of the protocol stack, based on the mobile MPLS architecture shown in [41].

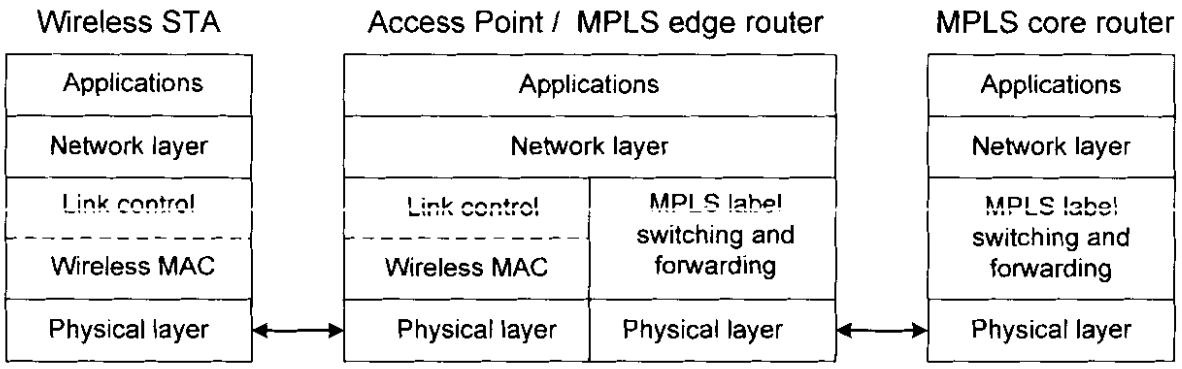


Figure 27 – Architecture of WLAN edge / MPLS core network [41]

Concerning MPLS Label Switched Path setup, the approach is to create several distinct paths through the core network, as proposed in [2]. The reason for this is two-fold. Firstly, we differentiate between traffic types by assigning each to its own LSP. Secondly, static or dynamic LSPs can be set up to ensure that the available bandwidth is fully utilized, while avoiding congested areas of the network. Static LSPs are manually created to specify an explicit path, while dynamic LSPs will attempt to find the best route based on the needs of the traffic flow. In this way we attempt to satisfy both the requirements for QoS, by providing class-based treatment and using traffic engineering to guarantee bandwidth.

3.3 Experimental setup

3.3.1 Reasons for the simulation approach

The most accurate way to obtain meaningful data for our study would undoubtedly be to build a real-world network and measure its performance. Unfortunately, the needed equipment isn't always readily available, and the expenditure associated with acquiring it creates the need for a more viable solution.

Gyires and Wen [8] discuss the topic of simulation techniques in some detail. While it is possible to describe the behaviour of certain network elements using mathematical equations, modelling an entire network is another matter. Networks are dynamic entities with behaviour that changes over time, and this cannot be represented using static equations.

The fastest and most cost-effective way of achieving our objective is to predict network performance via simulation. This approach enables us to construct the proposed network topology, evaluate it, and make changes where needed to achieve the desired result.

3.3.2 The OPNET Modeler network simulator

In recent years, network simulation packages have become quite readily available. The one chosen for this study is OPNET Modeler v 11.5, by OPNET Technologies Inc [80]. Modeler is a very comprehensive package that supports the modelling and simulation of a variety of different network technologies.

Several studies from the literature similar to this one have made use of the same software [8], [62], [64]. OPNET Modeler has several key features that make it a suitable choice for this study [75], [80]:

- OPNET is object-oriented, thus network elements are represented as objects with attributes that can be configured as needed.
- OPNET caters specifically for the modelling of communication networks.
- An OPNET model can be constructed using a graphical editor and interface, making it easier to represent a real-world system than when using a text-based interface.
- Statistics available for collection include overall network performance as well as more application-specific information.
- Simulation results can be graphically represented and analysed in various ways.
- The OPNET model library includes all the relevant protocols and LAN components, including MPLS and 802.11 WLAN.
- Radio links are modelled by the OPNET Wireless module.

OPNET Modeler has two distinct ways of modelling network traffic, namely explicit traffic and background traffic [65]. With explicit traffic, packets are modelled individually, and each packet transfer is represented as a discrete event. The relevant protocol effects such as segmentation, timeouts and retransmissions, media access, and QoS are also taken into account. Consequently, explicit traffic gives the most accurate results. Background traffic on the other hand, is modelled analytically, either as traffic flows or static resource loads. This approach does not model all the detail of the particular traffic flow, but represents it in an end-to-end fashion on the network layer. Lower layer functionality is not included.

Concerning the simulation environment, there are two available technologies (A third, called the Application Characterization Environment, is not included in our version of the OPNET package). Flow Analysis models network entities and protocols analytically using mathematical descriptions. This is a static approach, and has the advantage of fast computational times. Unfortunately, this comes at the expense of some accuracy. Flow Analysis is, among others, unable to model dynamic protocol effects such as windowing and flow control, and cannot calculate the delay variation of multimedia traffic.

The Discrete Event Simulator (DES) takes a rather different approach. Each network element is seen as an individual entity, and DES dynamically models the interactions between entities. Every change that occurs in the system causes an event, which in turn changes the state of the system; for example, a packet being placed on a communications link changes the state of the link. The results obtained from DES are superior to analytical methods, since it more closely mirrors the real-world behaviour of the network. The disadvantages to this approach are that a discrete simulation is computationally expensive and takes a fair amount of time to complete. DES also has high requirements in terms of memory usage.

Even though measurements taken from an actual network would be more accurate than a simulation, we have reason to believe that the results generated by OPNET Modeler are sufficient for our purposes. A study performed by Lucio et al [81] investigates the accuracy of two well-known network simulators when conducting packet-level network analysis. The measurements from an actual network test bed running Constant Bit Rate (CBR) data traffic and FTP traffic are compared to the results obtained from OPNET Modeler 9.0 and Network Simulator 2 (ns-2), a popular open-source simulator that is widely used in simulations found in the literature (See for example [1], [3], [25], and [44]).

The authors conclude that with proper fine-tuning, both simulators are capable of generating very accurate results compared to a real-world network. Ns-2 has the advantage of being freeware, and is thus very readily available, but Modeler is more user-friendly due to the graphical interface, and contains more features than Ns-2.

3.3.3 Design and layout of our simulation

The primary goal of our simulation is to provide accurate, useful data that will enable us to achieve the research objectives as stated in Chapter 1. This implies a careful choice of network topology, network technologies, traffic types, and simulation techniques to ensure that the resulting model provides the necessary answers. We now examine these design choices in more detail. This section deals with the default simulation setup; the different experimental scenarios as well as any alterations made to the basic layout are described in the following chapter.

i. *Network topology and OPNET library models* – In Section 3.2 we stated that our approach would focus specifically on infrastructure-based wireless LANs. Our basic network model appears in Figure 28. The layout is similar to that of a university campus network or an office network. Comparable topologies have been used in QoS and network performance studies, in particular [33], [47], and [54], the main difference being that our configuration uses wireless access networks.

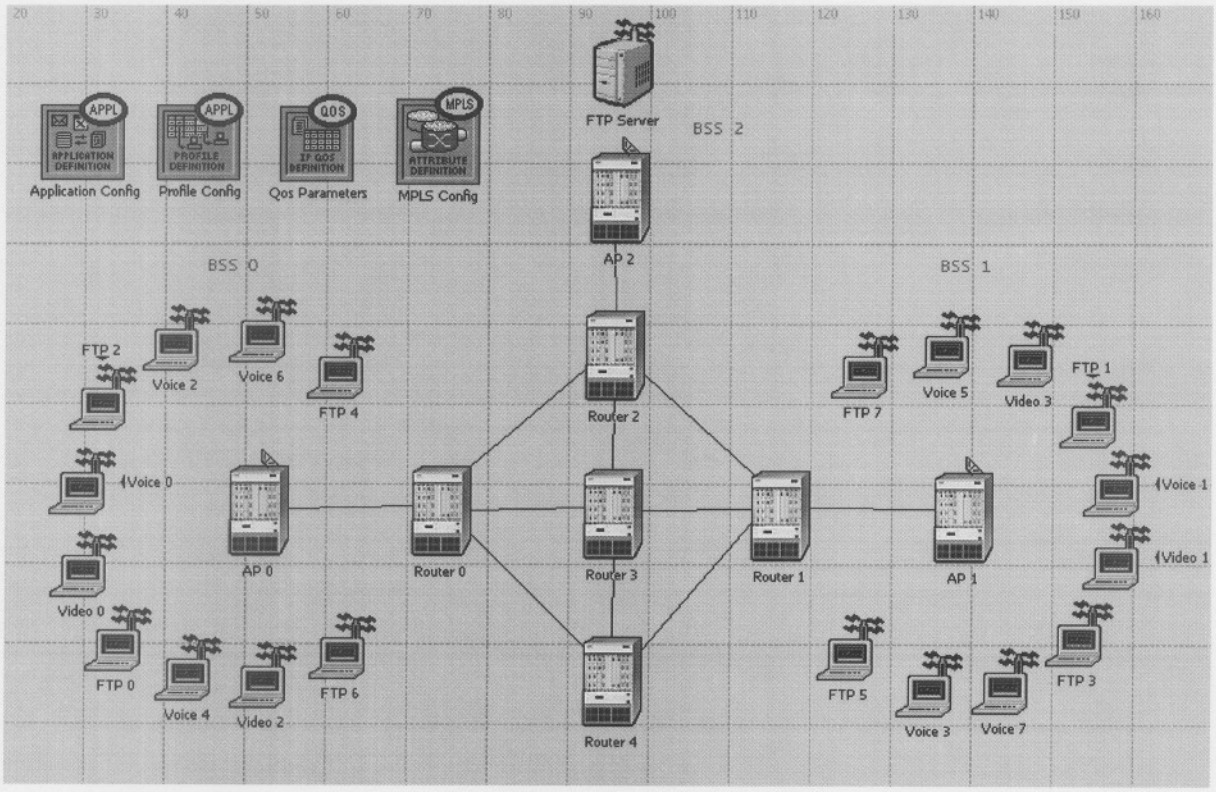


Figure 28 – The network simulated in OPNET Modeler

The network model comprises three 802.11 Basic Services Sets (BSS), labelled BSS 0, 1 and 2. BSS 0 and 1 each contain ten wireless LAN workstations, while BSS 2 comprises a single wireless FTP server. Each BSS is connected to the core network via an access point, AP 0, 1, and 2 respectively. The core network consists of five Ethernet routers, connected in a partial mesh via wired links. The links used between each AP and the core routers have higher bandwidth than the core links (10 Mbps vs. 1 Mbps). This link is shared by all traffic types, and thus we want to prevent it from becoming the potential bottleneck instead of the core network links. Table 2 describes the models selected from the Modeler library to construct the network.

Table 2 – Description of models used in the simulation

Model name	Role	No.	Description
Wlan_wkstn_adv	WLAN Station	20	A wireless node capable of running client/server applications over IP. It has a single WLAN connection
Wlan_ethernet_router_adv	WLAN Access Point	3	A WLAN router that can be used as an access point and also has a single Ethernet interface
Ethernet4_slip8_gtwy_adv	Core routers	5	An IP-base router with four Ethernet interfaces, that supports RIP and OSPF as routing protocols
Wlan_server_adv	FTP Server	1	A wireless server node that runs server applications over IP.
ppp_adv	Core links	8	A duplex link that connects two IP nodes with selectable data rates
10BaseT	AP-to-core links	3	A duplex10 Mbps Ethernet connection
Application Config	Application setup	1	Specifies applications that run in the network, using the available application types
Profile Config	Create traffic profiles	1	Used to create profiles that generate application-layer traffic on the network nodes
QoS Attribute Config	QoS settings	1	Defines QoS configuration for IP protocols
Mpls_config_object	MPLS settings	1	Used to create MPLS traffic trunks and FEC bindings

ii. *WLAN technologies and configuration* – Due to its widespread deployment in actual WLAN networks, IEEE 802.11b was selected as the WLAN technology for the access network. Each WLAN node is configured to operate at the maximum available data rate of 11 Mbps, using DSSS. The STAs use DCF for channel access.

The close proximity of the Service Sets to each other could lead to cross-channel interference. To prevent this, we use the optimal channel selection as discussed in Section 2.5. The channel assignments are Channel 1 for BSS 0, Channel 6 for BSS 1 and Channel 11 for BSS 2. We do not consider the effect of detail WLAN settings such as the contention window size or the retry limit, and where applicable they are left at the default value.

iii. *Traffic models and parameters* – Since our study concerns QoS, particularly of real-time multimedia traffic, accuracy is of paramount importance. When measuring metrics such as delay and jitter, a variation of a few milliseconds could mean the difference between acceptable or poor quality audio and video. For this reason, all the traffic used in our simulation will be of the explicit type.

Analytical background traffic as modelled in OPNET does not include the first two layers of the protocol stack, and as such is not suitable for this type of study.

For the choice of traffic types, we must include both multimedia and non-real-time traffic, as both types will be present in an actual network. This also enables us to contrast the impact that each of the QoS parameters has on different traffic flows. Previous studies on 802.11 QoS frequently make use of Voice over IP and video conferencing as real-time traffic, and data transfer such as FTP to represent non-real-time traffic [1], [4], [13], [62]. We will adhere to this practice, which is also advantageous from the modelling perspective. The OPNET model library contains built-in versions of application traffic, including VoIP, video conferencing, and FTP, each with several customizable attributes.

The flow of real-time traffic is configured as peer-to-peer connections between STAs, with one STA in each BSS. For example, STA Voice 0 in BSS 0 connects to Voice 1 in BSS 1, Voice 2 to Voice 3, etc. The same applies to video traffic, with Video 0 connecting to Video 1, etc. The FTP traffic flows between the FTP STAs and the FTP server in BSS 2. Half of the FTP traffic consists of uploads to the server, with the other half being downloads from the server.

It is interesting to note that FTP uses TCP as its transport protocol, while the real-time traffic uses UDP. As we mentioned in Section 3.1, UDP traffic tends to negatively affect the throughput of TCP traffic in a congested network, since TCP reduces its transmission rate in response to packet loss, while UDP does not [47]. Using both protocols in the simulation will also allow us to study the effect of this phenomenon on the QoS of the different applications.

Table 3 – Traffic used in the simulation

Application	STAs	Protocol	Parameters	Total load	Duration
File transfer (Heavy)	8	TCP/IP	Inter-request time: 20 s File size: 250000 bytes	100000 bytes/sec	100 sec. - End of sim.
Voice over IP call (GSM quality)	8	UDP/IP	Codec: GSM 32 Kbps Voice frames per packet: 3	64 000 bytes/sec	100 sec. - End of sim.
Video conferencing (Light)	4	UDP/IP	Packet size: 1500 bytes Frame rate: 20 fps	240 000 bytes/sec	100 sec. - End of sim.

Table 3 shows some of the significant traffic settings. The Application Configuration and Profile Configuration objects included in Modeler's object database are used to specify which traffic profiles are active, as well as their detailed parameters. Each traffic profile is activated at 100 seconds after the simulation starts (simulation time, not real time) and continues until the end of the simulation. This is a default setting, and is intended to allow the routing tables sufficient time to converge.

An important observation made by Zheng et al in [62] is that in the case of multiple traffic streams being handled by a single STA, traffic that uses small, frequent packets tends to fare better than traffic with large, infrequent frames. This applies to our simulation, since all traffic from a BSS flows via the AP. The default setting in Modeler for GSM voice traffic is one voice frame per packet. However, initial simulation results show that the other traffic types then suffer increased packet loss, due to the larger number of small voice packets flooding the buffer. So to reduce the number of voice packets and increase their size, we specify a value of three frames per packet.

iv. *Deployment of QoS technologies* – Application and node-specific QoS parameters can be set by editing the Application Configuration and QoS Attribute objects. The following simulation settings were used in the respective scenarios:

Baseline – The baseline scenario has no QoS mechanisms enabled. All applications are marked as Best Effort and the routers use the default FIFO scheme when forwarding traffic.

DiffServ – In the Application Configuration object, each different application is assigned a unique DiffServ Code Point according to its QoS requirements (see Table 4). FTP is marked as background traffic, but with greater emphasis on throughput than simple best-effort (code point AF11). Voice traffic receives code point AF43, which identifies it as multimedia traffic with high throughput and delay requirements. Video traffic is marked as EF, since it has very strict delay and throughput requirements.

Table 4 – DiffServ simulation parameters

Application	Per-Hop Behaviour	DiffServ code point	Scheduler
FTP	AF	AF11	Priority Queuing
VoIP	AF	AF43	Priority Queuing
Video	EF	EF	Priority Queuing

To handle the packet scheduling at each node, Priority Queuing (PQ) is enabled on the routers, by specifying it in the QoS configuration object. This decision is based on a study done by J. J. Smit [82] to evaluate the performance of various DiffServ schedulers. Of these, PQ and Weighted Fair Queuing (WFQ) are available in OPNET, but the use of WFQ is discouraged in [82]. PQ is a very simple scheme that serves the traffic with the highest priority first.

MPLS – Table 5 summarizes the MPLS simulation parameters. These are defined by means of the MPLS Configuration object. The first task is to assign each application to the proper Forwarding Equivalence Class (FEC), also known as FEC binding. This can be done based on parameters such as protocol, source / destination address, or Type of Service (ToS). The straightforward choice would be to use the ToS bits to distinguish different traffic types. Thus, in the Application Configuration object, we mark FTP as Standard, voice as Streaming Multimedia, and video as Reserved. Based on this distinction, three FEC's are created, called FTP_Class, Voice_Class and Video_Class. As each packet enters the MPLS domain, its ToS value will identify it as either voice, video or FTP, and thus ensure that it is mapped to the correct FEC. FEC binding takes place at the edge routers, namely Routers 0, 1, and 2.

Table 5 – MPLS simulation parameters

Application	ToS value	FEC	Traffic trunk	LSP assignment
FTP	Standard (Class 2)	FTP_Class	-	-
VoIP	Streaming Multimedia (Class 4)	Voice_Class	Voice_trunk	Voice LSP 0 - 3
Video	Reserved (Class7)	Video_Class	Video_trunk	Video LSP 0 - 3

The deployment of Label Switched Paths is done in similar fashion to the approaches used in [33], [43] and [47]. Firstly, we use flow splitting to utilize bandwidth more effectively.

Real-time flows coming from each BSS are mapped onto one of two available LSPs. This gives a total of four LSPs for voice traffic and four LSPs for video traffic, two in each direction (see Figure 29). The LSPs are static and each route is specified explicitly.

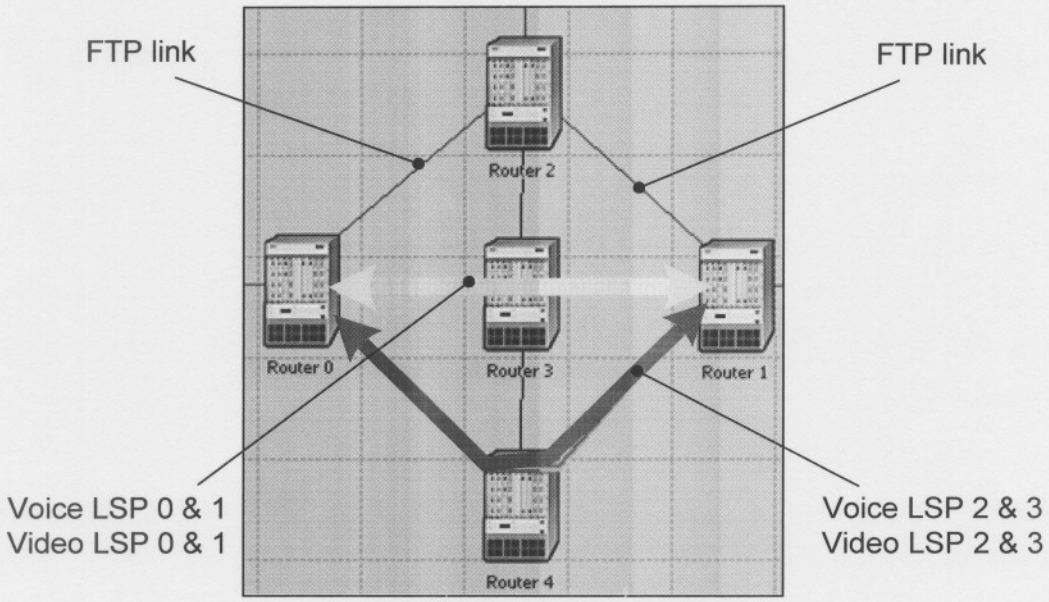


Figure 29 – LSP deployment in the core network

Next, we use the LSPs to direct the real-time flows away from the topmost two links connected to Router 2. The FTP traffic flowing to and from BSS 2 almost without exception uses these links, as they provide the most direct path. For this reason we re-route the real-time traffic to prevent them from mixing with the FTP flows.

Finally, each application must be placed in an appropriate traffic trunk. The trunk size is specified in terms of the characteristics of the traffic it carries, such as maximum and average bit rate. In this case the size of the trunk is obtained by dividing the traffic load of each application as shown in Table 2 by four, which gives 480 000 bits/s for video and 125 000 bits/s for voice traffic. To ensure sufficient capacity the trunks are sized at 500 000 and 150 000 bits/s respectively. FTP is not placed in a trunk since the FTP traffic profile varies considerably during the course of the simulation. A single traffic trunk is placed in each LSP.

v. *Simulation technique and parameters* – Our choice of explicit traffic eliminates Flow Analysis as a simulation technique, since only Discrete Event Simulation supports the use of explicit traffic. Also, while Flow analysis provides quicker results, to correctly analyse the QoS of different traffic types in our simulation requires the greater accuracy of DES.

Before executing the simulation, DES provides several configurable settings, such as time duration and simulation seed value. We simulate 600 seconds of network activity, which includes the first 100 seconds during which no traffic flows. The remaining 500 seconds are sufficient for the network to stabilize so that the results will not be affected by transitional effects [48], while also ensuring that the time it takes the simulation to complete remains within reasonable bounds. As a rule of thumb for our simulation, it takes approximately a minute real-time to simulate a minute's worth of network traffic.

The seed value is a used to set the initial state of OPNET's random number generator. This value affects any process in the network that has random elements built in. To ensure that the seed value does not cause inconsistencies in the results of the various simulation scenarios, we keep it constant for all simulation runs.

vi. *Measured parameters* – OPNET records two types of output statistics, namely object and global. Object statistics record an attribute of a specific node or link in the network. Global statistics measure the overall performance of a protocol or application, by averaging the results from the individual nodes. We use both types in the simulation to provide a complete picture of the network's performance. The statistic probes chosen for each object or application are listed in the next chapter together with their respective results.

3.3.4 Limitations and assumptions

The following refers to parameters that were either disregarded, or specifically altered for the purpose of the study.

i. *Mobility and network infrastructure* – We do not consider the effects of mobility in the simulation, and therefore all wireless nodes are stationary. The reason for this is that we are more concerned with the performance of the QoS technologies deployed in the network backbone. We also assume that the network infrastructure remains stationary,

with no nodes joining or leaving the network. This is justified since infrastructure networks are deployed on a more permanent basis than ad hoc networks. Also, the time span of the network traffic we simulate is short enough to assume that no arrivals or departures will take place.

ii. *Wireless effects* – In Section 2.5.4 we discussed the impact on QoS of wireless effects such as fading, path loss, interference, and the hidden terminal problem. As far as possible, we have attempted to eliminate these effects from the simulation. This is mainly because there is no intra-BSS communication between wireless nodes, except for the transmissions between the AP and each wireless STA. The STAs are placed close together to ensure that all are able to detect each other, thus reducing collisions and eliminating the hidden terminal problem. They are also in close proximity to the AP, which ensures that the effect of fading and path loss should be minimal. The disadvantages associated with medium access, however, cannot be eliminated. All STAs still have to compete for access using the CSMA/CA protocol.

iii. *Core bandwidth and traffic saturation* – We mentioned previously that one of our aims is to determine the ability of the applied QoS techniques to allocate network resources fairly. Two steps are taken to achieve this. Firstly, we use links with limited bandwidth in the core network. Secondly, the traffic volume is high enough to ensure that the network becomes congested. This is important, since it is difficult to measure QoS performance in an environment where sufficient resources are available to service all traffic, regardless of QoS deployment. Similar methods are used in [1] and [64].

3.3.5 Choice of miscellaneous variables and parameters

When constructing network models for simulations, it is important to focus on the factors and parameters that will make a significant contribution towards achieving the objective of the study [8]. There are also parameters that will undoubtedly affect the results, but are outside the scope of what we are trying to achieve with the simulation. These include the choice of routing protocols, packet size, buffer size, etc. Their values will typically remain constant throughout each simulation scenario. The following section documents our design choices regarding these variables, as motivated by the literature.

i. *Packet size* – The literature indicates that the size of the data packets can have a notable influence on 802.11 performance. In [48] Pellctta and Velayos state that the maximum packet size when using an AP with an Ethernet backbone is 1500 bytes. A study performed in [34] shows that the throughput of 802.11b increases as packets become larger, up to a maximum of 1472 bytes. Beyond this size, packets are no longer sent as single units but are fragmented into several smaller packets. Each fragment then receives its own IP header, which increases the overhead and as a result also decreases the effective throughput.

   The fragmentation threshold of 802.11 packets in OPNET is set at 2304 bytes. Based on this and the above information, we set the packet size of video conferencing traffic at 1500 bytes. For voice and FTP traffic, the size of individual packets is not a configurable attribute.

ii. *Buffer size* – The buffer refers to the amount of memory available on each STA or router for storing packets before they are forwarded to the next node. The choice of buffer size is important, since it directly impacts two of the QoS parameters, namely delay and packet loss [35], [63]. If the buffer is very large, queued packets have to wait a long time before being transmitted, which increases end-to-end delay. Conversely, making the buffer small will shorten the delay, but if the buffer is filled to capacity, any newly arriving packets will be dropped until some storage space becomes available.

   Zheng et al in [62] include various traffic types in their simulation and assume a finite buffer, with packets being dropped as soon as the buffer is full. Thanthry et al [63] investigate the effect that changing the buffer size has on voice traffic. They find that the end-to-end delay increases as the buffer size increases, but that less voice data is dropped. This only holds for a congested network, since the performance of the network is unaffected by the buffer size under low traffic load conditions.

   The wireless router model has built-in buffer sizes of up to 1024000 bits, although larger sizes can be manually specified. To prevent needlessly long delay times, we selected a smaller size of 256000 bits for the access points. The core routers have 16 MB of memory available by default.

iii. *Routing protocols* – The choice of routing protocol is fairly straightforward. Modeler enables Routing Information Protocol (RIP) as the default option. But [69] states that OSPF must be enabled in order to create MPLS LSPs. Therefore, we use OSPF on every router in the network, and also keep RIP as a backup option in the event that OSPF is unable to find a route.

3.4 Summary

In this chapter we explored some of the previous work done in the field of QoS research, in both wireless LANs and wired IP networks. In addition, we motivated our approach from the literature and described the design and layout of our simulated network. We now move to the task of gathering experimental data from the network via simulation.

4. Results and discussion

4.1 Introduction

In this chapter, we briefly discuss the types of experimental data gathered for our study, and then take an in-depth look at the simulation data itself. We perform two distinct experiments in order to gather results. The first is a direct comparison between the QoS techniques in a core network with limited bandwidth. The second experiment simulates a link failure between two of the core routers. It is also important to note that each experiment is comprised of three scenarios: The baseline network with no QoS mechanisms, a network with DiffServ enabled, and lastly a network with MPLS deployed (see Figure 30).

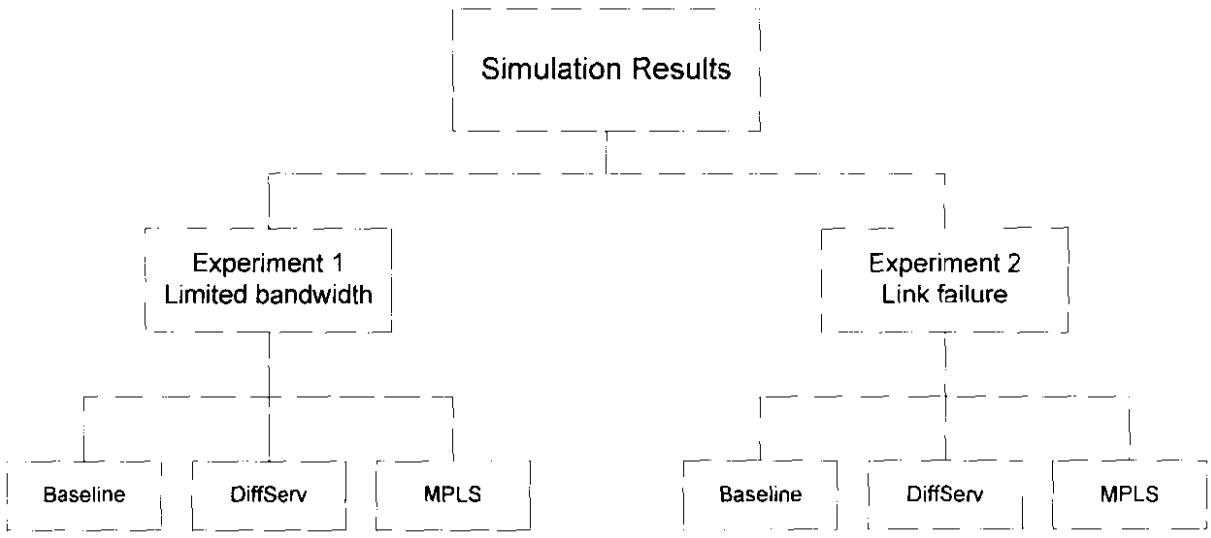


Figure 30 – Simulation hierarchy

In each experiment, we will use the measurements taken by the statistic probes listed in Table 3 to draw a comparison between the QoS performance of the different scenarios. This can be done by:

a) Directly comparing the raw numerical data obtained from each scenario.
b) Using OPNET's built-in statistical functions to process the data and present it in a different format.

Each statistic probe, whether measuring global or object statistics, gathers its data in vector format. The vector data can be plotted as shown in Figure 31. This figure is typical of the ones we will use throughout the chapter to illustrate the simulation results. However, we exported the figure data to a spreadsheet in order to improve the quality and presentation of the figures.

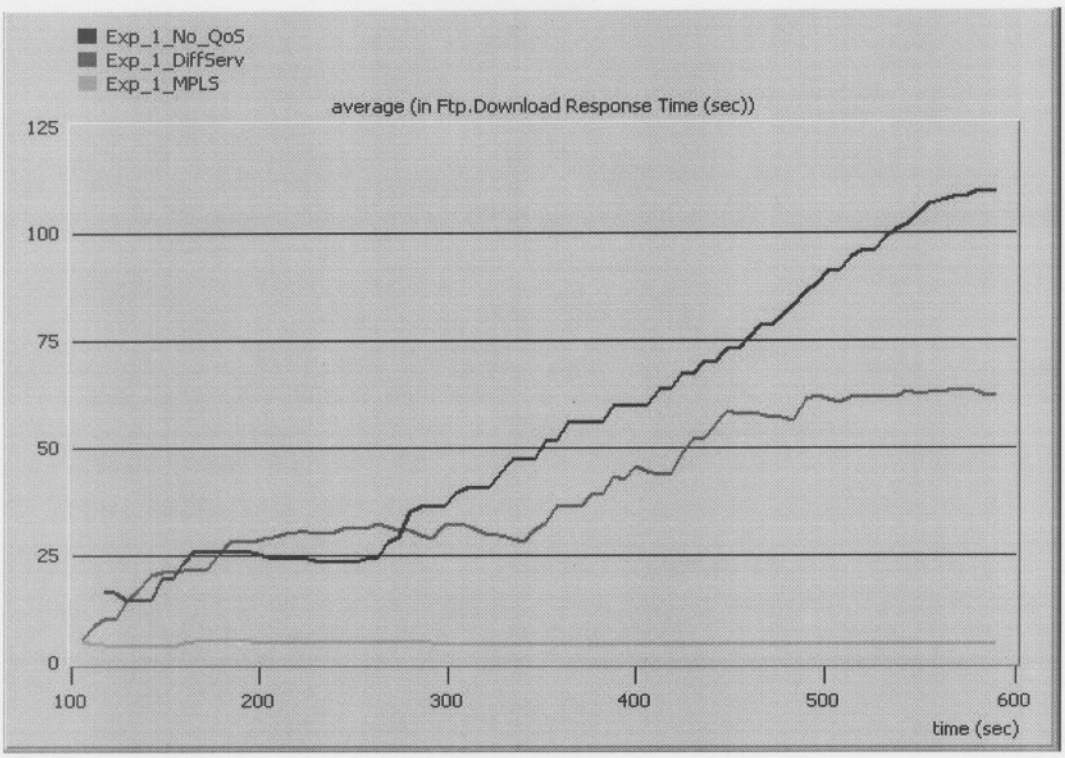


Figure 31 – Figure plotted by Modeler from data vector

Using the data vector, Modeler is also able to calculate statistics such as minimum and maximum values, sample mean, variance and standard deviation. This data can then be exported in text format as shown in Table 6.

These statistics can form the basis for a more compact representation of the results, for example only showing the sample mean of each scenario as opposed to plotting every data point in each vector. The set of raw data gathered from the simulation probes is too great to present in its entirety, therefore we use the processed data as often as possible, only reverting to the raw data where it is useful to illustrate a point or clarify a discussion. A more complete set of raw and statistical data is included on the supplied compact disc (see Appendices).

Table 6 – Sample of statistical data generated by Modeler

```
zone:              0
statistic:          'Exp_1_No_QoS: Ftp.Download Response Time
(sec).none'
length:            84
number of values:    84
horizontal, min:     100
        max:       598
vertical, min:       10.1133876533
        max:       205.170789604
initial value:       undefined
final value:         undefined
expected value:      109.495108827
sample mean:         109.495108827
variance:          3,783.74527383
standard deviation:  61.5121554965
** conf. intrvls valid if entries are independent samples
80% conf interval:  [     97.3307618561, 121.659455798     ]
90% conf interval:  [     93.8815392521, 125.108678402     ]
95% conf interval:  [     90.8917067803, 128.098510874     ]
98% conf interval:  [     87.4111621219, 131.579055532     ]
99% conf interval:  [     85.0544148116, 133.935802843     ]
** Operations List **
Original vector: Exp_1_No_QoS: Ftp.Download Response Time (sec).none
---------
------------------------------------------
```

Throughout both experiments, we will evaluate the performance of the real-time applications according to the bounds given in Table 7. These are based on established ITU or equivalent bounds used in previous studies of a similar nature [7], [77], [78], [82]. In the absence of such standards, as for example in the case of FTP traffic, the baseline scenario serves as the basis for comparison.

Table 7 – ITU bounds for real-time applications

Parameter	High quality	Acceptable quality	Unacceptable quality
Delay	< 150 ms	150 – 400 ms	> 400 ms
Jitter	< 10 ms	10 – 30 ms	> 30 ms

4.2 Experiment 1 – QoS provision with limited core bandwidth

4.2.1 Description

This experiment is a straightforward comparison between the different QoS techniques. The simulation is set up exactly as described in the previous chapter. The goal of this experiment is to evaluate how each QoS technique performs in a network with limited resources. Firstly, we look at link utilization, IP packet loss, and other factors to determine the behaviour of the core network. We will also pay specific attention to the performance of each application as described by the main QoS parameters. Finally, we examine the performance of the wireless access network to determine if and how it is affected by the QoS technologies used in the core network.

4.2.2 Link utilization and core network conditions

To a great extent, the conditions in the core directly influence the behaviour of the entire network. For this reason, we begin the examination of the simulation results by looking at the probe data gathered from the core links. In addition, we inspect the DES log messages generated by each scenario for warnings concerning IP traffic dropped, possible congested conditions, etc. Table 8 lists the statistic probes used to obtain this data.

Table 8 – Statistics recorded in the core network

Statistic name	Unit	Type	Description
Link utilization	Percentage	Object	Percentage consumption of available channel bandwidth, where 100.0 indicates full usage. For a duplex link, utilization is recorded in both directions.
IP traffic dropped	Packets/second	Global	The total number of IP datagrams dropped by all nodes in the network.

Measuring the link utilization enables us to form a picture of the volume of traffic flowing through the core, and where it is most highly concentrated. For this particular statistic, we only view the results from the links that carry traffic to and from BSS 0, as shown in Figure 32. Since the traffic deployment in BSS 1 is identical, the links connected to it should exhibit similar behaviour. Note that all links are duplex, and that each figure only represents the utilization in one of the two directions.

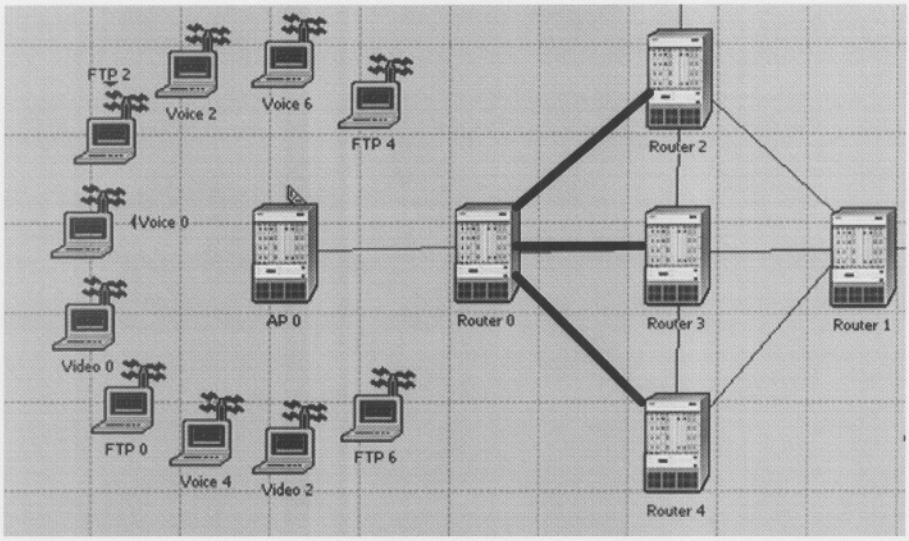


Figure 32 – Experiment 1: Links selected for measuring utilization

Figure 33 shows the average utilization of the link connecting Router 0 and Router 2, with the traffic flowing towards the latter. Since Router 2 is the entry point for all FTP traffic into the core network, this link should carry most, if not all, of the FTP traffic to and from BSS 0. The baseline and DiffServ scenarios both have an average utilization of over 70%. By default, Modeler limits maximum link utilization to 75%, meaning that in both cases this link is highly congested. In the MPLS scenario, all voice and video traffic flows were routed way from this link, and as a result it only reaches 20-30% of capacity. In the other two scenarios it carries a mixture of real-time and FTP traffic, which clearly exceeds the capacity of the link.

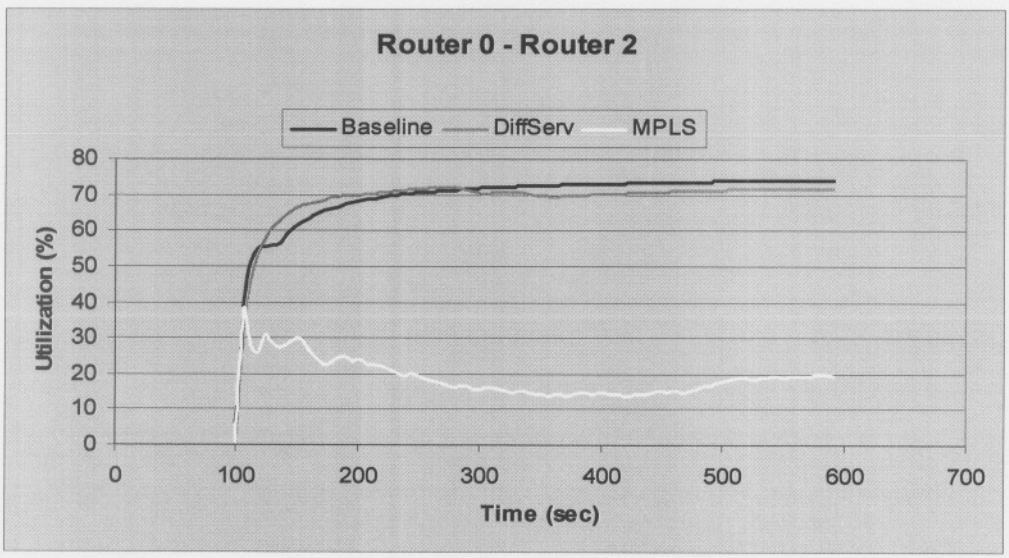


Figure 33 – Experiment 1: Average link utilization from Router 0 to Router 2

Next, we study the results from the link between Router 0 and Router 3, as displayed in Figure 34. In contrast to the previous case, this link is very much under-used by the baseline and DiffServ scenarios, only reaching 15% of capacity in both cases. In the MPLS scenario, the link carries one voice trunk and one video trunk, resulting in a 65% utilization figure. Though high, this is still below the maximum amount available.

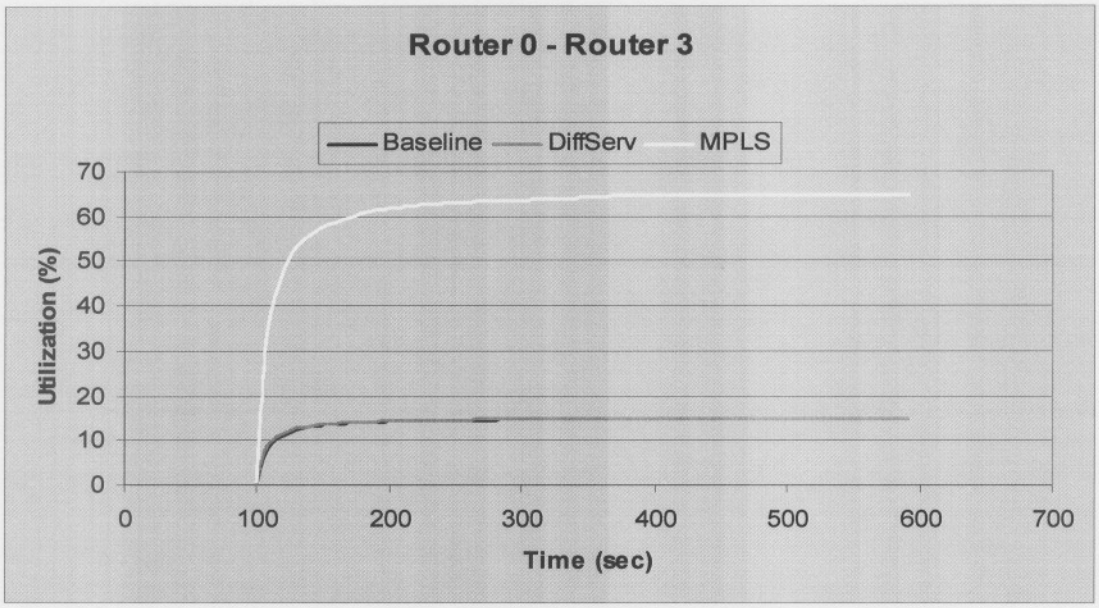
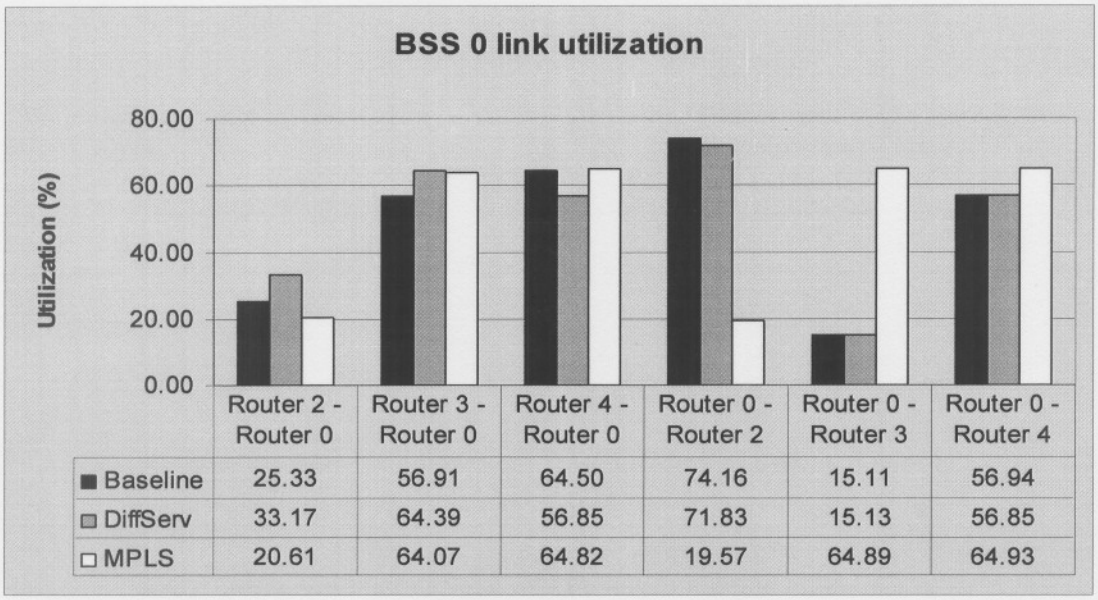


Figure 34 – Experiment 1: Average link utilization from Router 0 to Router 3

For a more complete overview, we plot the sample means of all link utilization to and from BSS 0 in Figure 35. The baseline and DiffServ scenarios display great variations in both outbound and inbound traffic, with figures ranging from 15% to 75%. These two scenarios both use OSPF for packet routing, which seems to flood some links to capacity while hardly using others. This supports the statement made in Chapter 2 that a DiffServ-enabled network is ignorant of available resources when routing traffic. Contrary to this, MPLS traffic engineering allows a more even spread of traffic in both directions, while still separating the real-time and multimedia flows. Average link utilization in this scenario never rises above 65% in either direction.



	Router 2 - Router 0	Router 3 - Router 0	Router 4 - Router 0	Router 0 - Router 2	Router 0 - Router 3	Router 0 - Router 4
■ Baseline	25.33	56.91	64.50	74.16	15.11	56.94
▨ DiffServ	33.17	64.39	56.85	71.83	15.13	56.85
□ MPLS	20.61	64.07	64.82	19.57	64.89	64.93

Figure 35 – Experiment 1: Mean link utilization to and from BSS 0

During each simulation run, Modeler generates DES log messages. These take the form of general information, notices, or warnings intended to inform the user of possible errors. Thus, the log messages can be used to help determine the severity of congestion and packet loss in each scenario. The log results are summarized in Table 9.

Table 9 – Experiment 1: Warning messages in DES log

Scenario	Protocol	Warning message	Nodes / routers affected
No QoS	IP	Packets have been dropped in the router for congestion reasons.	Router 0, Router 2
DiffServ	IP	Packets have been dropped in the router for congestion reasons.	Routers 0, 1 and 2
No QoS	TCP	TCP is retransmitting data segments which will cause additional overhead on the lower layers and links	FTP 2, FTP 5
DiffServ	TCP	TCP is retransmitting data segments which will cause additional overhead on the lower layers and links	FTP Server
MPLS	-	No warnings	-

Warnings are issued in both the baseline and DiffServ scenarios, indicating the presence of packet loss in the core routers due to congested conditions. The fact that more routers are affected in the DiffServ scenario seems to point towards more severe congestion than in the baseline network. Both these scenarios also warn of TCP retransmissions, meaning that one or more TCP sessions have timed out. However, no warning messages are present in the MPLS scenario log, which means that no packets have been dropped due to congestion.

To quantify this packet loss, we study the global IP traffic dropped statistic. Figure 36 shows that MPLS has a stable average loss rate of 2 packets per second. This can be attributed to transmission errors, since we know from the log that no packets were dropped due to congestion. The baseline scenario loss rate initially matches that of MPLS, but begins to increase steadily at about 300 seconds as congestion in the core sets in. The DiffServ scenario displays somewhat different behaviour, with packet loss beginning almost immediately, and remaining at 7-8 packets per second throughout the course of the simulation.

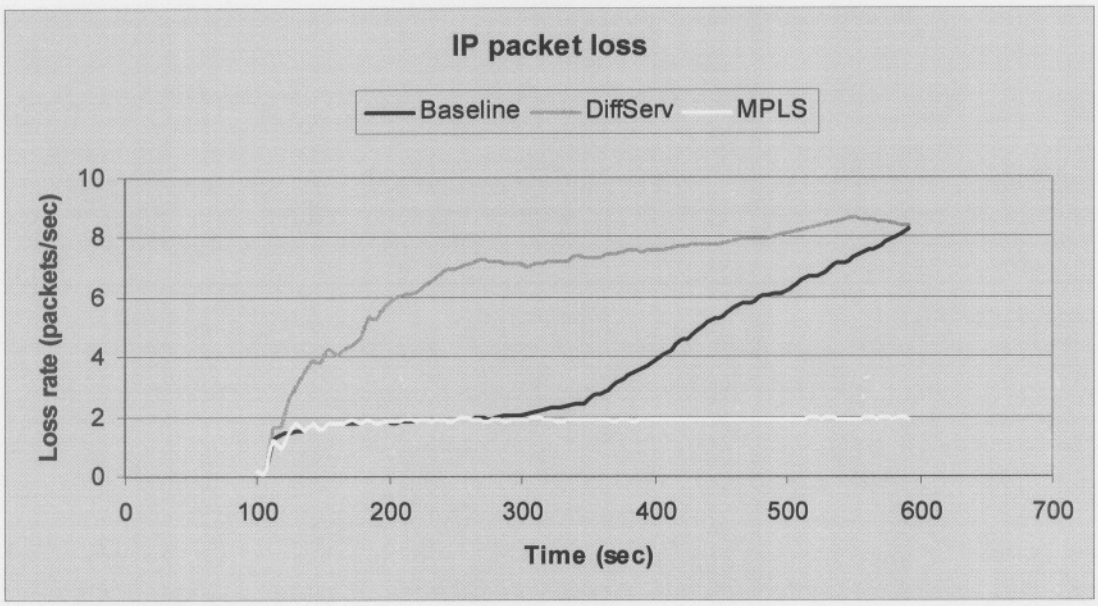


Figure 36 – Experiment 1: Average IP packet loss rate

We can summarize the above simulation results as follows: The baseline network treats all traffic equally, with no preferential treatment in terms of scheduling or routing. Traffic is forwarded via the shortest path, while other links remain underused. Over-utilized links cause congestion in the core, leading to an ever-increasing packet loss rate. It is reasonable to assume that this will be reflected in the application performance statistics.

The DiffServ network gives preferential treatment in terms of scheduling, but does not take the available bandwidth of the links into account when forwarding traffic. This, like in the baseline network, leads to congested conditions in the core links, which can be seen from the similarities in link utilization between the two scenarios. This also explains the early onset of packet loss in the routers, which immediately react to the congestion by dropping lower priority traffic.

The MPLS network separates FTP traffic from the multimedia flows, giving each a dedicated path. Traffic is spread more evenly across the links, preventing any one link from becoming congested. The effectiveness of this technique is demonstrated by the absence of packet loss (due to congestion) in the routers.

The following three sections focus on the performance of the application traffic.

4.2.3 Video conferencing

Of all the applications in the experiment, video conferencing is the most demanding in terms of total throughput. In addition, strict bounds must be placed on delay and jitter to guarantee adequate QoS to the user. To determine how video traffic fared in each scenario, we use the statistics described in Table 10. Note that Modeler does not record jitter for video traffic, but the variance in packet end-to-end delay times. Larger variance implies greater variation in delay, which would also correspond with high jitter values.

Table 10 – Video conferencing statistic probes

Statistic name	Unit	Type	Description
Video conferencing traffic sent	Bytes/sec	Global	Average amount of traffic submitted to the transport layers by all video conferencing applications in the network.
Video conferencing traffic received	Bytes/sec	Global	Average traffic forwarded to the voice applications in the network by the transport layers.
Video conferencing end-to-end delay	Seconds	Global	The time taken to send a video packet to the destination node application layer.
Video conferencing delay variation	-	Global	Variance among end-to-end delays for video packets

Looking first at throughput in terms of traffic received, we see from Figure 37 that the throughput for all scenarios is fairly similar. The flows are stable, except for a slight drop-off experienced by the baseline scenario from about 500 seconds onward. This is most likely due to the mounting congestion in the core network seen in the previous section, leading to increased video packet loss.

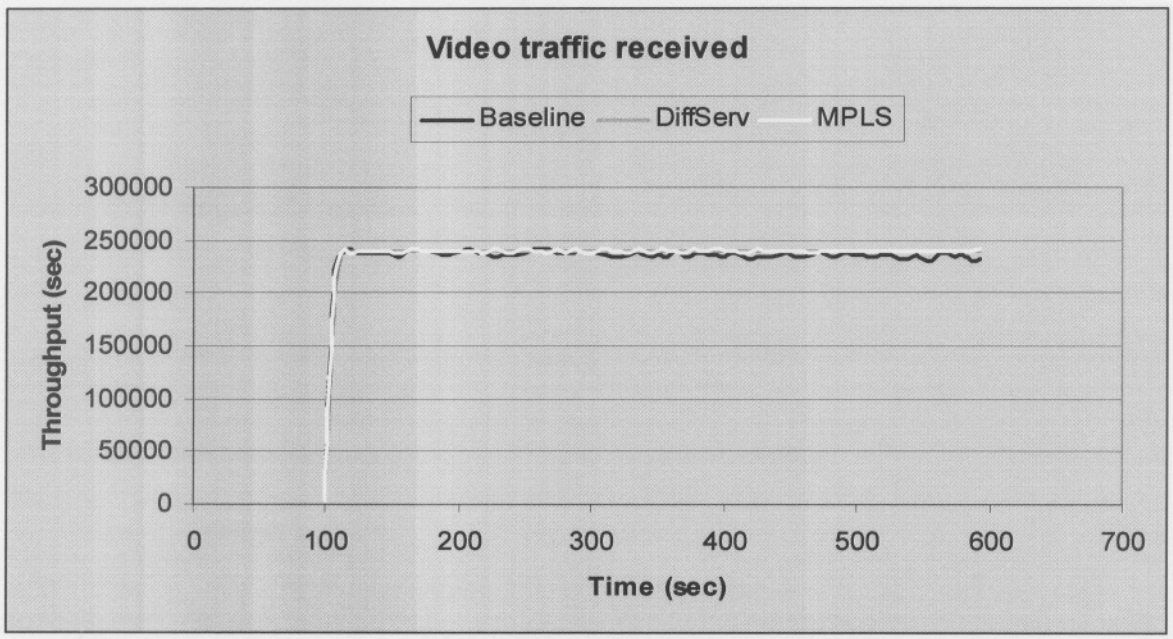


Figure 37 – Experiment 1: Video traffic received

The end-to-end delay of video traffic is shown in Figure 38. Here the lack of QoS in the baseline scenario is unmistakable. The delay increases steadily as the core links become more saturated with traffic, settling at a maximum of about 1.75 seconds between 300 and 400 seconds. This is high enough to seriously hamper the quality of any video conversation between the communicating parties. In contrast, the DiffServ and MPLS scenarios both have stable delays below 100ms, although the exact value cannot be seen from the figure.

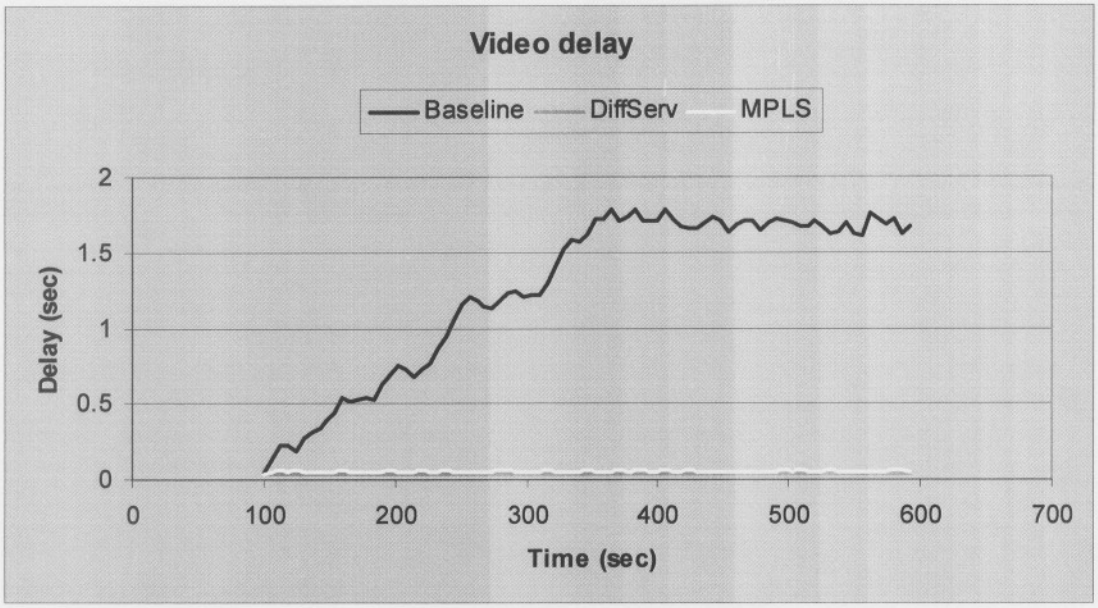
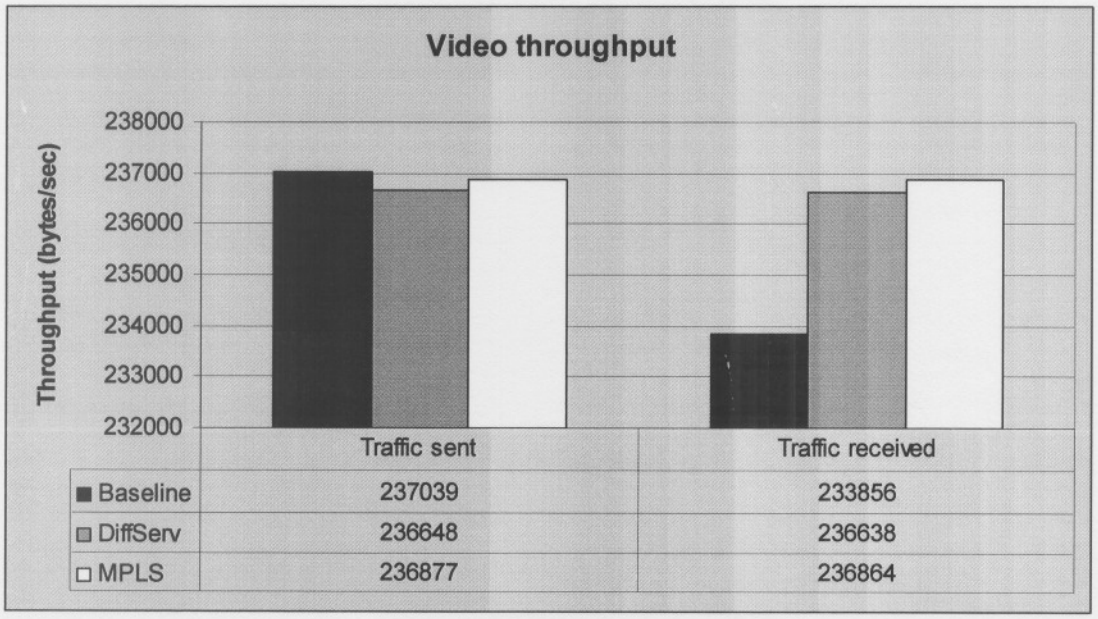


Figure 38 – Experiment 1: Video end-to-end delay

To further clarify the situation, we plot the sample mean from each scenario for the respective parameters. As can be seen from Figure 39, there is little to choose between the scenarios in terms of throughput. The baseline scenario sends the most traffic but receives the least, recording approximately a 1.3 % loss in throughput as calculated from the values shown. DiffServ and MPLS both have a 99.9% delivery rate in terms of the traffic sent.



	Traffic sent	Traffic received
Baseline	237039	233856
DiffServ	236648	236638
MPLS	236877	236864

Figure 39 – Experiment 1: Mean values of video throughput

A different picture emerges when examining the delay values shown in Figure 40. As seen previously, the baseline network has an extremely high average delay of 1.26 seconds. The constantly increasing delay is reflected in the high delay variation value of 3.04. The scenarios with QoS present perform significantly better. DiffServ has the lowest mean delay of 41 ms, compared to the 54 ms of the MPLS scenario. The delay variation of the DiffServ network is five orders of magnitude lower than the baseline at 0.0001, with a value of 0.0005 for MPLS. These extremely small mean variations confirm that both flows are stable, with no sudden drops in throughput which would also impact the delay time.

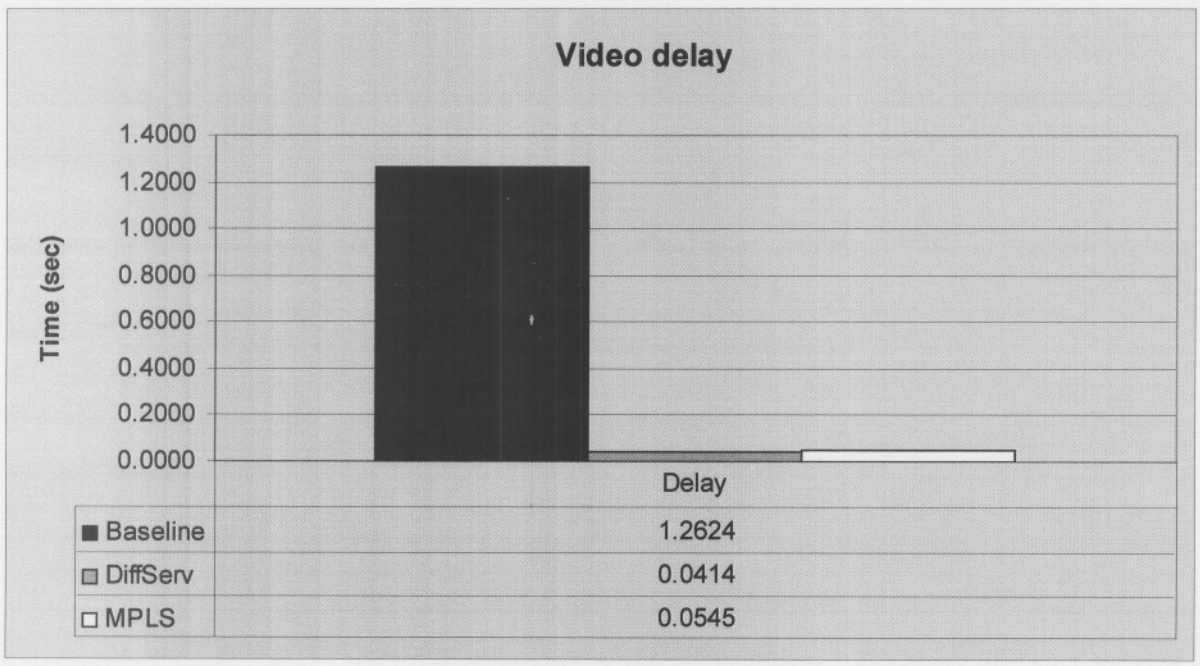


Figure 40 – Experiment 1: Mean values of video delay

We see from the results that a congested network with no QoS mechanism present has a disastrous effect on video traffic. Packet loss in this case is minimal, but the QoS of the application is severely degraded in terms of delay and packet delay variation. It would be all but impossible to hold a video conferencing session with delay times of over a second, and a large variation in frame inter-arrival time. Contrary to this, the results indicate that both QoS mechanisms are able to guarantee a better than acceptable level of video QoS. DiffServ appears to have slightly better overall performance than MPLS, but both are well within the 150 ms delay bound for high-quality video.

4.2.4 Voice over IP

As mentioned in the previous chapter, the voice load is much less than that of video or FTP traffic. However, since VoIP is also real-time, it has similar requirements than video in terms of packet loss, delay, and jitter. The measured parameters for voice traffic appear in Table 11.

Table 11 – Statistic probes measuring voice traffic

Statistic name	Unit	Type	Description
Voice traffic sent	Bytes/sec	Global	Average amount of traffic submitted to the transport layers by all voice applications in the network.
Voice traffic received	Bytes/sec	Global	Average traffic forwarded to the voice applications in the network by the transport layers.
Voice end-to-end delay	Seconds	Global	The time taken to send a voice packet to the destination node application layer.
Voice jitter	Seconds	Global	Difference in delay between consecutive voice packets

Figure 41 plots the rate at which voice traffic is received in each scenario. What is immediately evident is that voice shows strikingly similar performance to video traffic in terms of throughput. Once again, the baseline scenario appears to have a slight drop-off towards the ends of the simulation. This corresponds with the increasing packet loss observed in the baseline network from about 300 seconds onwards. The DiffServ and MPLS video flows both appear stable, with no visible drops or major fluctuations.

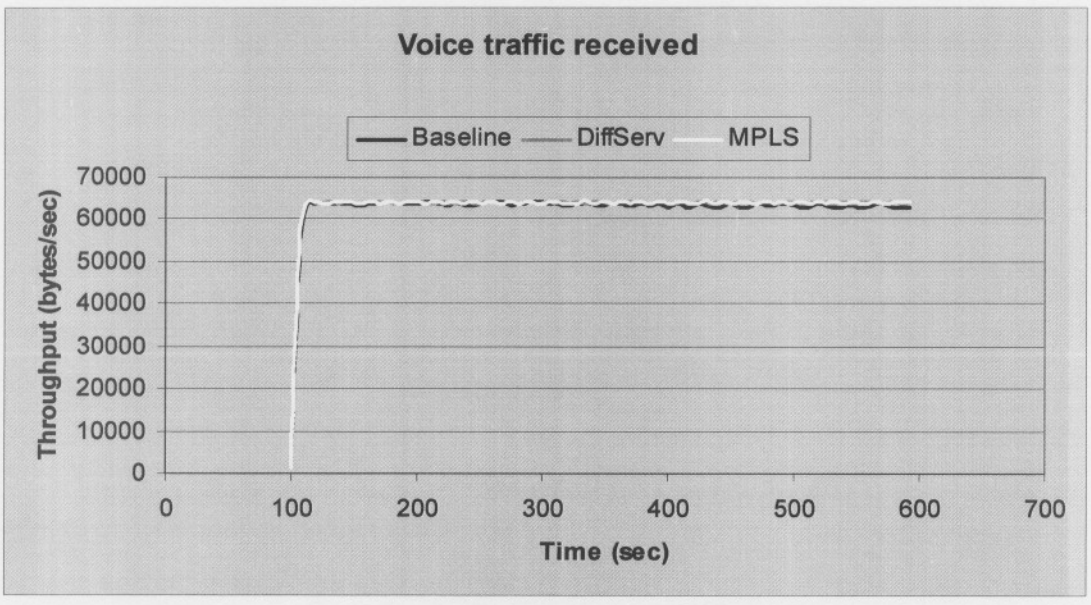


Figure 41 – Experiment 1: VoIP traffic received

As far as end-to-end delay is concerned, shown in Figure 42, we also witness results comparable to those of video traffic. DiffServ and MPLS are able to provide consistent delay times of 200 ms or less, while the lack of QoS in the baseline network causes delay values approaching 1 second when it stabilizes at around 350 seconds simulation time. This equates to a round-trip delay of 2 seconds, which is four times higher than the delay threshold for a practical voice conversation [77].

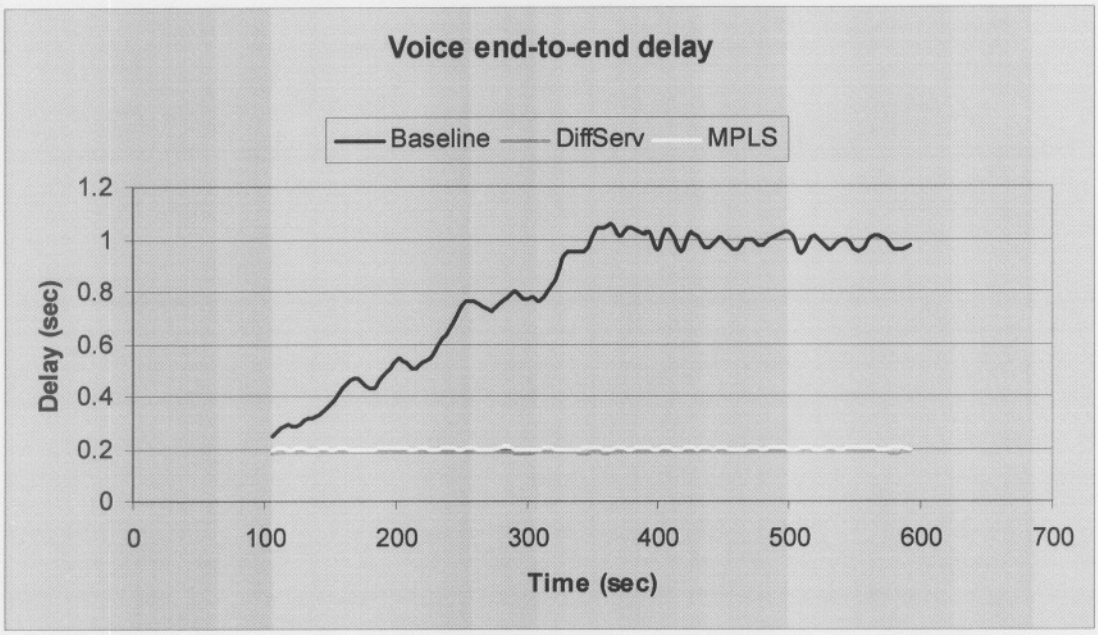


Figure 42 – Experiment 1: VoIP end-to-end delay

The mean throughput values obtained from each scenario as shown in Figure 43, are nearly identical, with only a 28 bytes/second difference from the lowest (no QoS) to the highest (MPLS). As with video traffic, the baseline receives noticeably less traffic in comparison with the other two. Even so the difference between the sending and receiving rate is less than 1%. The DiffServ and MPLS scenarios both record a throughput figure of 99.9%.

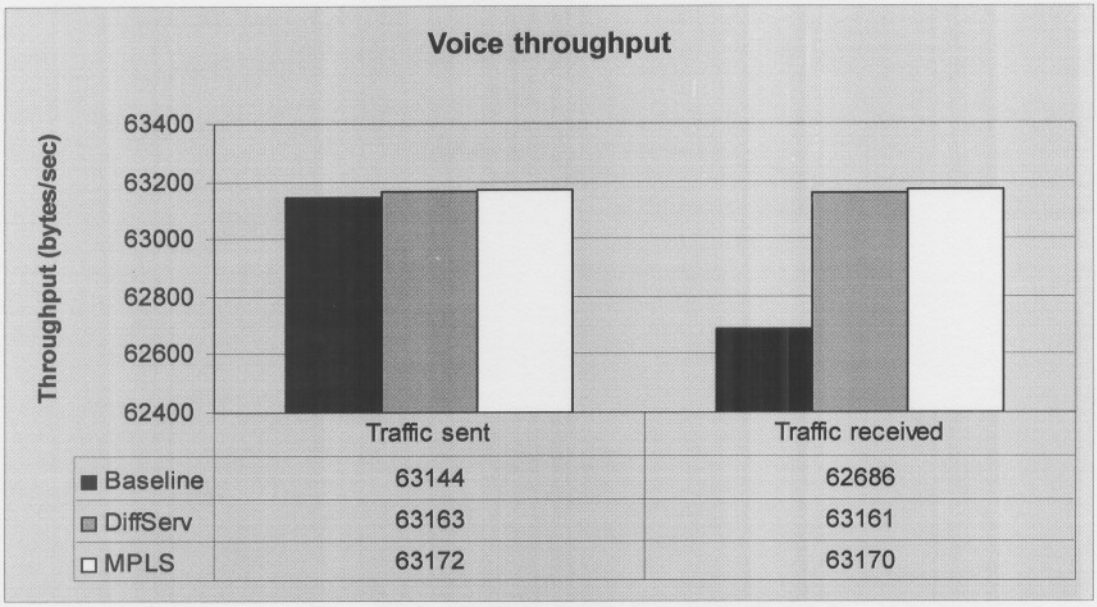


Figure 43 – Experiment 1: Mean VoIP throughput

As can be seen from Figure 44, the mean delay of voice traffic in the baseline scenario is four times higher than the DiffServ and MPLS networks. DiffServ has a slight edge on MPLS, of 14 ms on average. Both are higher than the recommended 150 ms for high-quality voice, but still within acceptable bounds for a voice conversation. What is interesting to note is that the respective mean delays of voice traffic in these two scenarios are much higher than the 41 ms and 54 ms measured for video traffic.

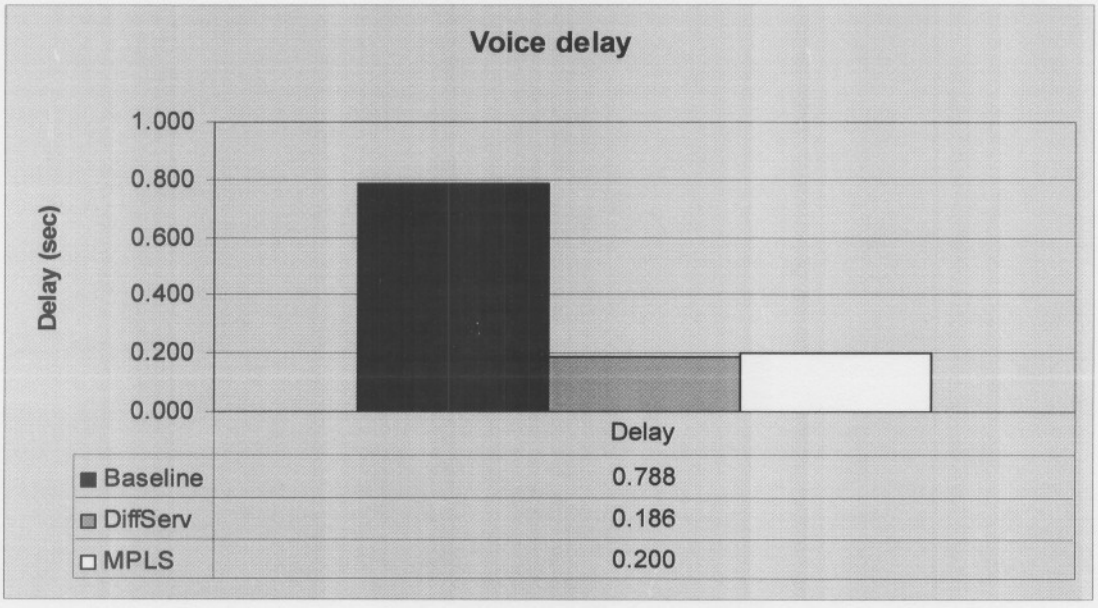


Figure 44 – Experiment 1: Mean VoIP delay

Extremely low voice jitter was present in each scenario, as can be seen in Figure 45. The no-QoS baseline network has a jitter value measure in tens of milliseconds. This is even more pronounced in the DiffServ and MPLS scenarios, with hardly any jitter being recorded. The near-absence of jitter points towards very consistent voice delivery, which is an important component of QoS provision.

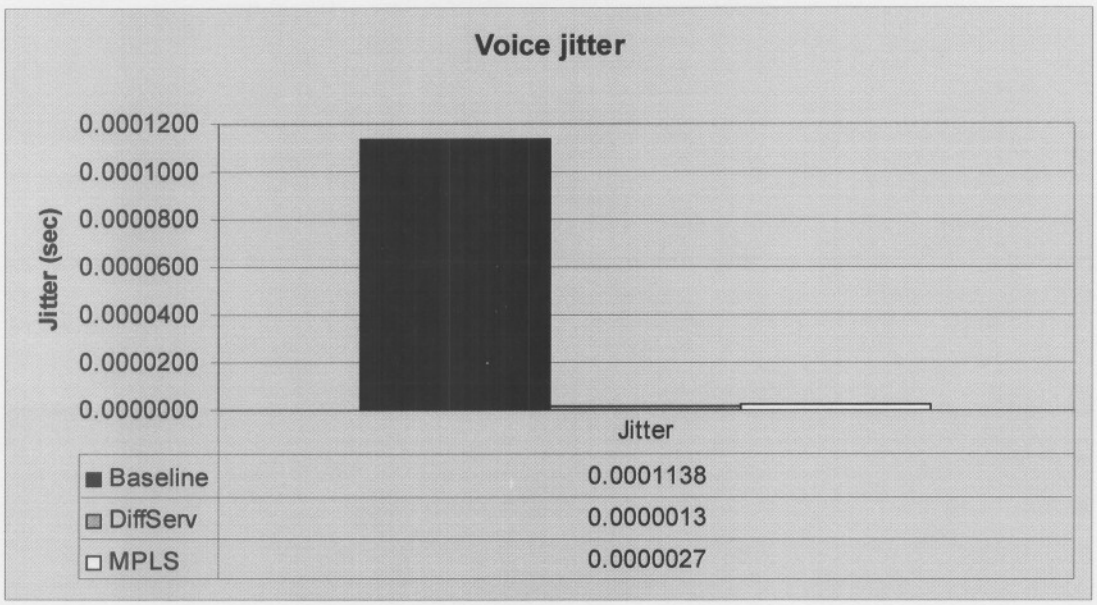


Figure 45 – Experiment 1: Mean VoIP jitter values

Summarizing the voice results, we see that voice traffic displays similar characteristics in terms of performance to video traffic. Voice packet loss is minimal, but the absence of service differentiation in the baseline network has a severe impact on voice end-to-end delay, as was the case with video traffic, even though the total voice throughput is much lower. This reduced throughput benefits the voice flows in terms of jitter, with all three scenarios being well below the recommended high-quality 10 ms bound.

A notable feature of the results is that voice records higher delay values than video traffic in the two QoS scenarios. In the case of DiffServ, this can be explained by the lower DSCP priority value assigned to voice traffic. Video thus gets priority at the routers, causing longer voice delays, especially when the link is busy. In the MPLS scenario, voice and video are mapped to different LSPs, yet share the same core links.

MPLS doesn't differentiate between traffic in terms of forwarding behaviour, thus the most logical explanation would be that the high volume of video traffic leads to the voice packets spending more wait time in the buffers. Unfortunately, we were not able to gather the needed data from the simulation to verify this theory.

The results show that despite their different approaches, both DiffServ and MPLS are able to provide sufficient QoS to voice traffic.

4.2.5 FTP traffic

Having compared the performance of the multimedia flows, we now turn to the results from the non-real-time traffic. We evaluate FTP performance in terms of the following parameters:

Table 12 – FTP statistics recorded

Statistic name	Unit	Type	Description
FTP download response time	Seconds	Global	The time elapsed between sending a request and receiving the response packet.
FTP upload response time	Seconds	Global	Time elapsed between sending a file and receiving a response.
FTP traffic sent	Bytes/sec	Global	Average amount of traffic submitted to the transport layers by all FTP applications in the network.
FTP traffic received	Bytes/sec	Global	Average traffic forwarded to the FTP applications in the network by the transport layers.

Figure 46 shows the average throughput of FTP traffic sent by each scenario in bytes/second. The reaction of the TCP protocol to congestion in the core network is clearly visible. In each case, the initial throughput is high, followed by a downward trend as TCP reduces the transmission rate. The throughput of both the DiffServ and MPLS scenarios initially falls below that of the baseline scenario. Notice however, that from 400 seconds onwards the MPLS scenario throughput begins to increase. This might indicate that TCP doesn't perceive the network as being congested and is reacting by raising its transmission rate. This view is supported by the results we observed concerning the state of the core network in the MPLS scenario, since it contains no warnings about TCP retransmissions.

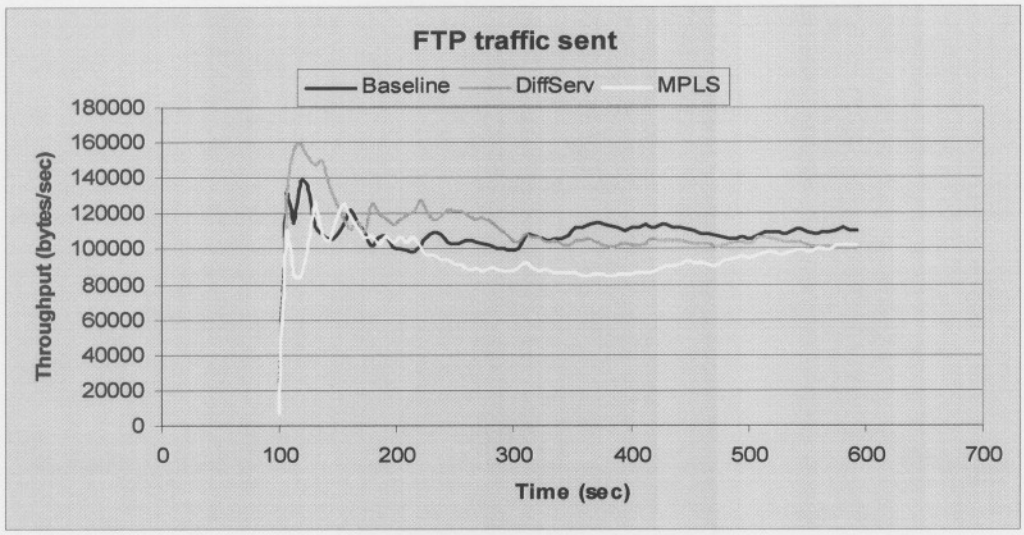


Figure 46 – Experiment 1: Average FTP throughput

From the perspective of an FTP user, download response time is a very important metric, since it represents the amount of wait time before a requested file is delivered. The average download response time for the three scenarios can be seen in Figure 47. The baseline network's response time continually increases as the core network becomes more saturated with traffic. From about 500 seconds onwards, receiving an FTP response takes between 150 and 200 seconds. This is to be expected, since there are no QoS mechanisms present. However, the DiffServ scenario shows a similar trend. There are great variations in the response time, with some values of over 200 seconds being recorded. In contrast, the MPLS network performs consistently, with stable delay times that never rise above 10 seconds.

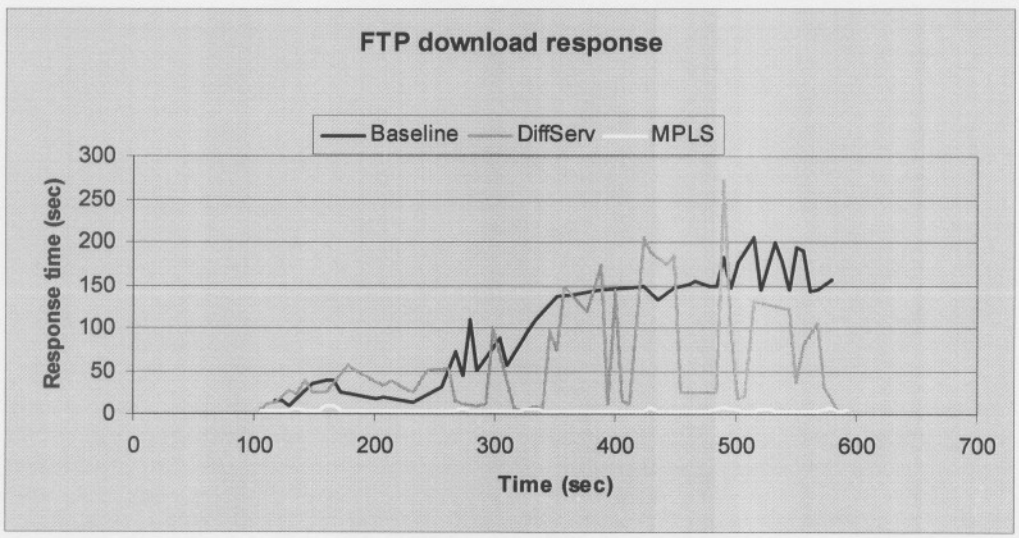
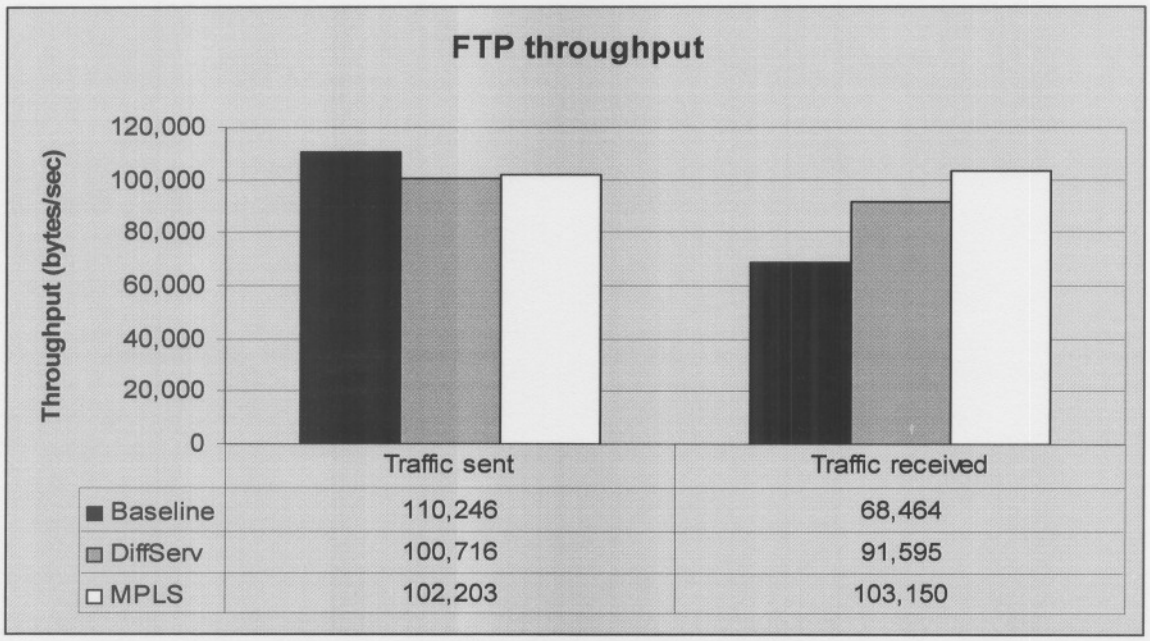


Figure 47 – Experiment 1: Average FTP download response time

Taking the sample mean from each scenario's vector data, we plotted the values of both sent and received traffic in Figure 48. This clearly indicates the comparative throughput performance of each scenario. The baseline network sends the most traffic, but the rate at which it is received is only 62% of the sending rate, as can be calculated from the data provided below the figure. DiffServ improves this to 91 %, and in the MPLS scenario requested FTP traffic is delivered at the same rate at which it is sent.



	Traffic sent	Traffic received
Baseline	110,246	68,464
DiffServ	100,716	91,595
MPLS	102,203	103,150

Figure 48 – Experiment 1: Sample means of FTP throughput

Looking at the mean response times shown in Figure 49, the MPLS scenario clearly outperforms the other two. The congestion in the baseline network causes delays of over 100 seconds before a response is received. The DiffServ network performs better than the baseline, but FTP traffic still endures long delays even though it receives Assured Forwarding treatment from the routers The is also some inconsistency between upload and download performance, since it takes 20 seconds less on average to upload a file to the server than to download from the server. This can most likely be attributed to the variation in link utilization to and from the FTP access network. For example, the link between Router 2 and Router 0 is 31% utilized in one direction and 72% utilized in the opposite direction. The response times in the MPLS scenario are consistently the best, averaging about 4 seconds for both uploads and downloads.

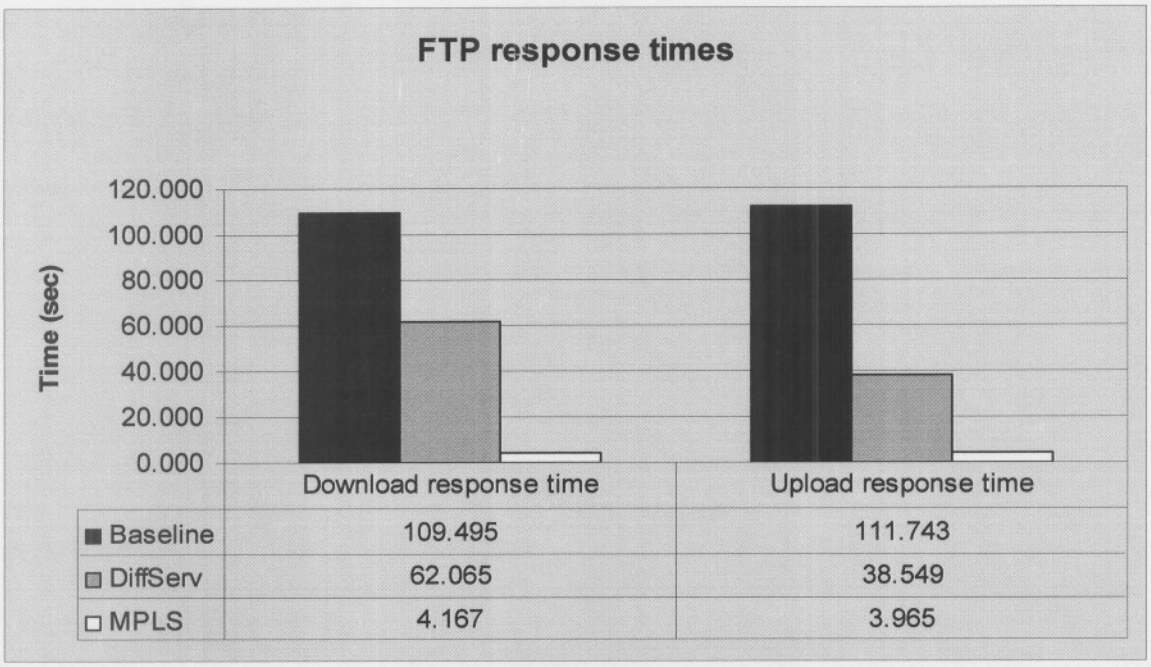


Figure 49 – Experiment 1: Sample means of FTP response time

To summarize, the MPLS scenario gives the best FTP performance by some margin, both in terms of throughput and response time. The primary reason for this can be traced back to the link utilization profile shown earlier in the chapter. Because the baseline and DiffServ scenarios do not take available bandwidth into account, the most often used links become congested by traffic. Mixing FTP and real-time traffic on the same links has a very detrimental effect on the former, since its lower DiffServ priority means that the FTP traffic will always be served last. This causes FTP sessions to time-out, leading to retransmissions which further increase delays, as indicated by the warning messages in the DES log. Also, the TCP protocol used by FTP detects the congested conditions and decreases its output accordingly.

In the MPLS scenario, no time-outs or retransmissions take place, and FTP delivery is both faster and more consistent than in the other scenarios. Clearly, the approach of routing real-time traffic away from links using FTP has proven to be more effective than assigning DiffServ priorities at the routers.

4.2.6 *Effect on the wireless access network*

The final section of this experiment focuses on the primary research question: Does adding QoS, and specifically MPLS, to the core network have an effect on the wireless access network? In an attempt to answer this question, we examine the following wireless LAN simulation parameters:

Table 13 – WLAN statistics collected

Statistic name	Unit	Type	Description
Wireless LAN data dropped (buffer)	Bits/sec	Global	Total size of higher layer packets dropped by WLAN MAC due to full buffers
Wireless LAN throughput	Bits/sec	Global	Total traffic forwarded by WLAN to the higher layers
Wireless LAN delay	Seconds	Global	End-to-end delay of all WLAN packets received by the WLAN MAC in the network and forwarded to higher layers.
Wireless LAN media access delay	Seconds	Global	Total queue and contention delays of WLAN packets forwarded to the MAC.

Figure 50 shows the total throughput of WLAN for each scenario. All three achieve similar figures, averaging around 7 Mbps. The MPLS scenario, however, shows greater variation in throughput than the baseline or DiffServ networks. This results from the intermittent bursts of the FTP traffic, which is less visible in the other scenarios due to their lower FTP throughput.

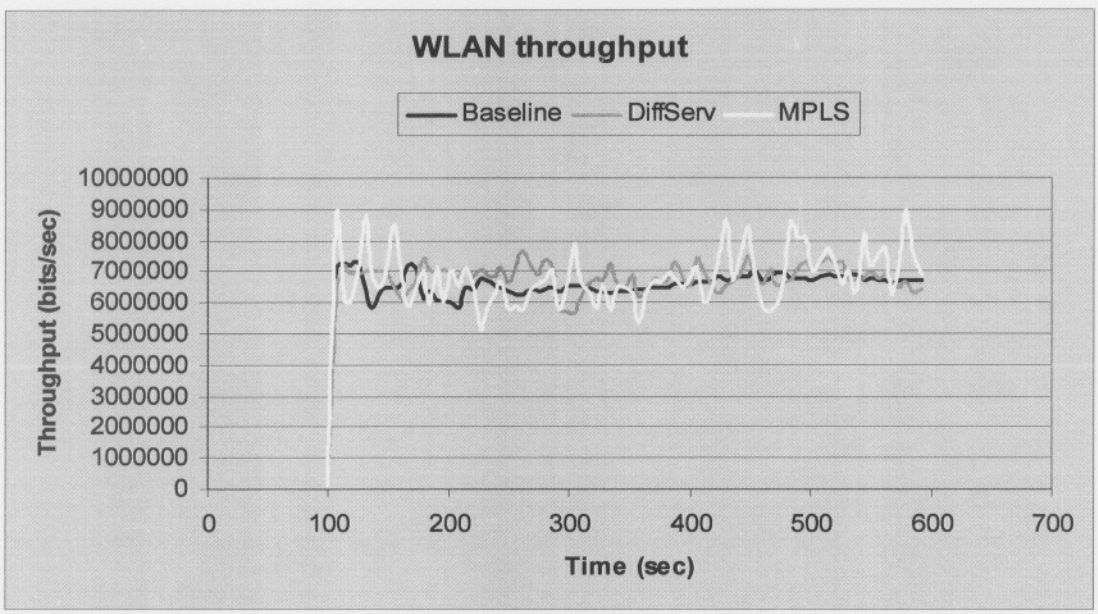


Figure 50 – Experiment 1: WLAN throughput

By plotting the mean value from each scenario (Figure 51), we can clearly see that both scenarios with QoS deployed have a higher WLAN throughput than the baseline scenario, achieving an approximate 3 % improvement. The increased throughput results from the reduction in congestion compared to the baseline network. The decrease in end-to-end delay and the reduction of TCP retransmissions, allow the access networks to send traffic at a higher rate.

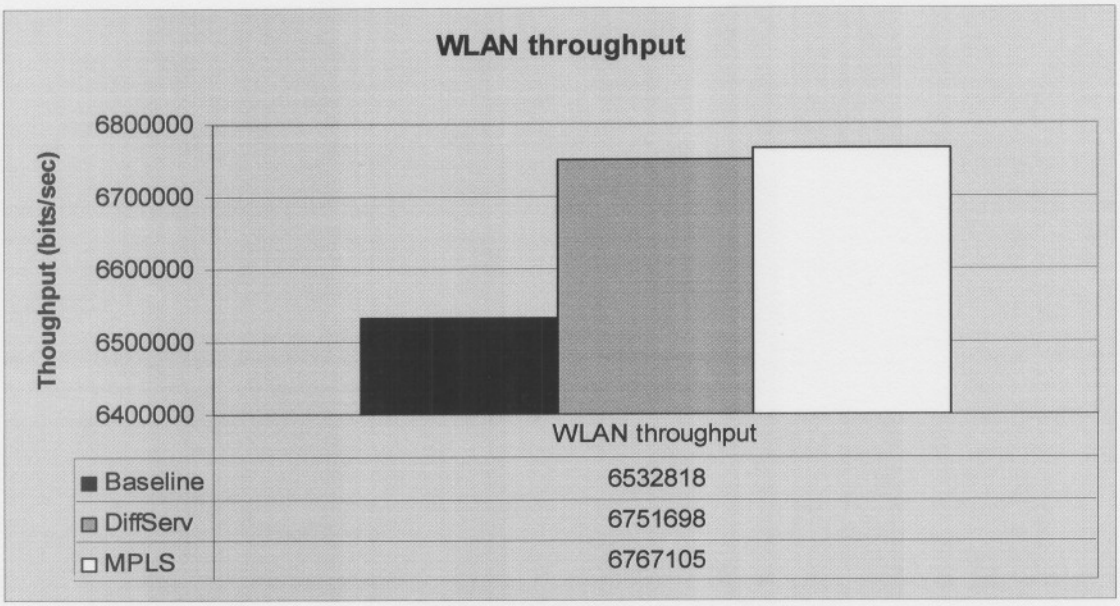


Figure 51 – Experiment 1: Mean WLAN throughput

The WLAN delay metric recorded by Modeler is a measure of the total time taken for a packet to be received by the MAC, processed and forwarded to the transport layer. Of these delays, the one that has the greatest impact on QoS is Media Access Delay (MAD), which represents the time an STA has to wait before being allowed to access the medium, as well as certain processing and queuing delays. The delay results are shown in Figure 52. All scenarios demonstrate very low access delays, mainly due to the close proximity of the STAs to the AP, which diminishes wireless effects such as the hidden terminal problem. It is worth noting, however, that the DiffServ network shows higher average delays than the baseline or MPLS networks. This is the effect of using Priority Queuing, which introduces extra delays by prioritizing the traffic in the buffer. The MPLS and baseline scenarios forward the traffic without differentiating between types at a per-node level and thus have very similar delay times.

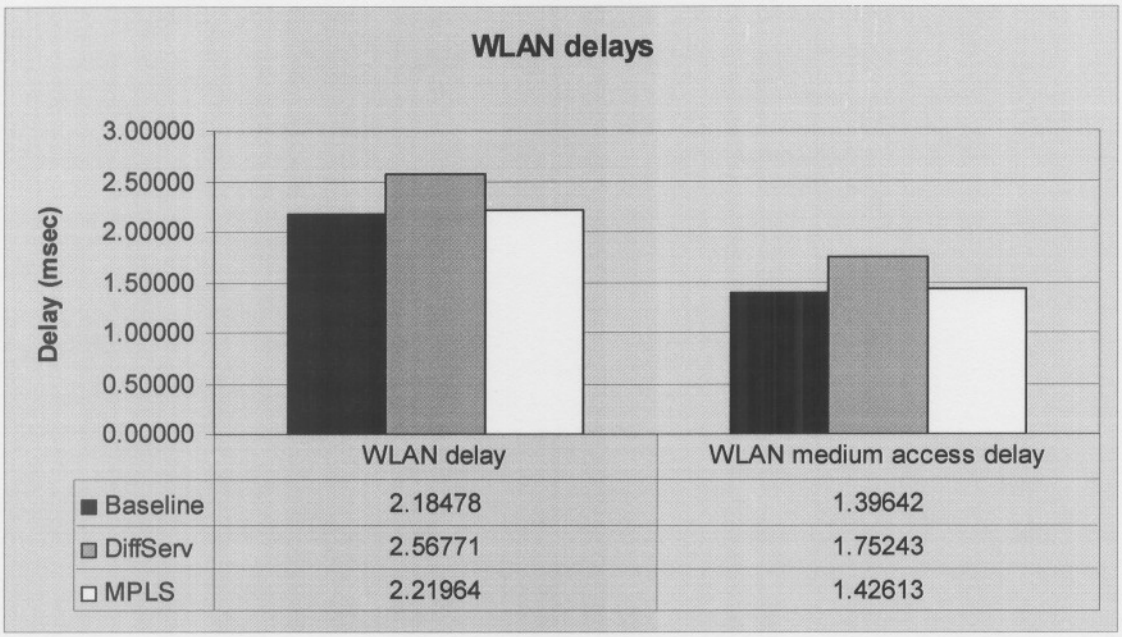


Figure 52 – Experiment 1: WLAN delay and media access delay

A further effect of DiffServ Priority Queuing is an increase in buffer utilization corresponding to the increased wait time in the buffer. As a result, some data is lost at approximately 550 seconds due to WLAN buffer overflow. This does not occur in the baseline or MPLS scenarios.

The WLAN results demonstrate that adding QoS to the core network does indeed influence the performance of the access network, albeit to a limited extent in this experiment. Both QoS-enabled scenarios achieved a higher average throughput than the baseline network. Access delay shows little, if any change, although DiffServ Priority Queuing does cause a slight increase in access delays, and also some data loss due to increased buffer usage.

4.2.7 Summary of Experiment 1

We summarize the results from this experiment by using the experimental data to quantify the effect of each QoS technique on the no-QoS baseline network.

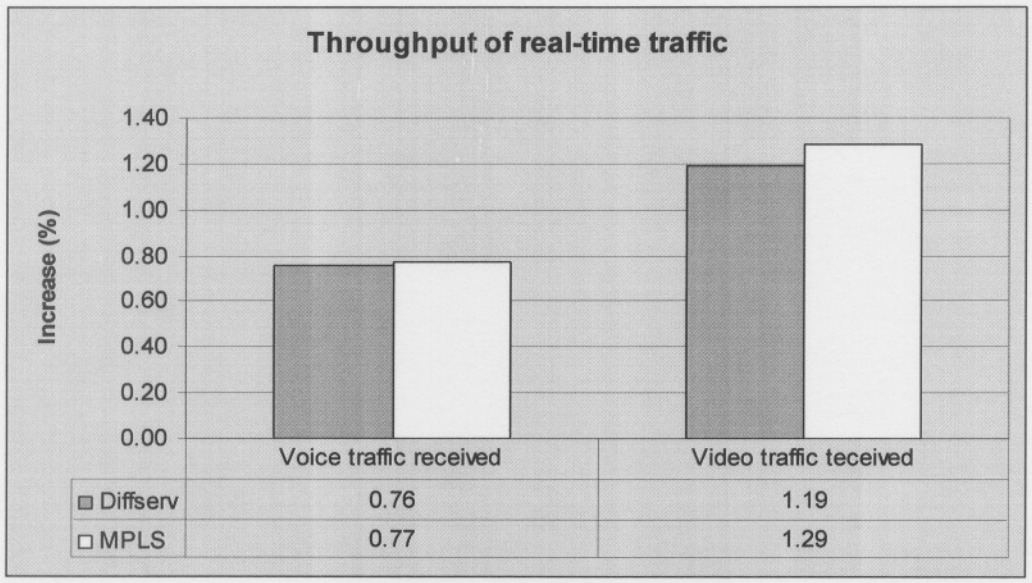


Figure 53 – Experiment 1: Effect of QoS on real-time throughput

Figures 53 and 54 are concerned with the real-time voice and video traffic. Minimal gains were made in terms of total throughput for both traffic types in either scenario. The major improvements were in those of delay and jitter, where both DiffServ and MPLS reduced the delay and jitter values recorded in the baseline network by 75% or more. DiffServ performs marginally better in this area, but never by more than 2% over MPLS. Thus, MPLS traffic engineering was able to provide multimedia QoS comparable to that of DiffServ, by efficiently managing the available bandwidth in the core network.

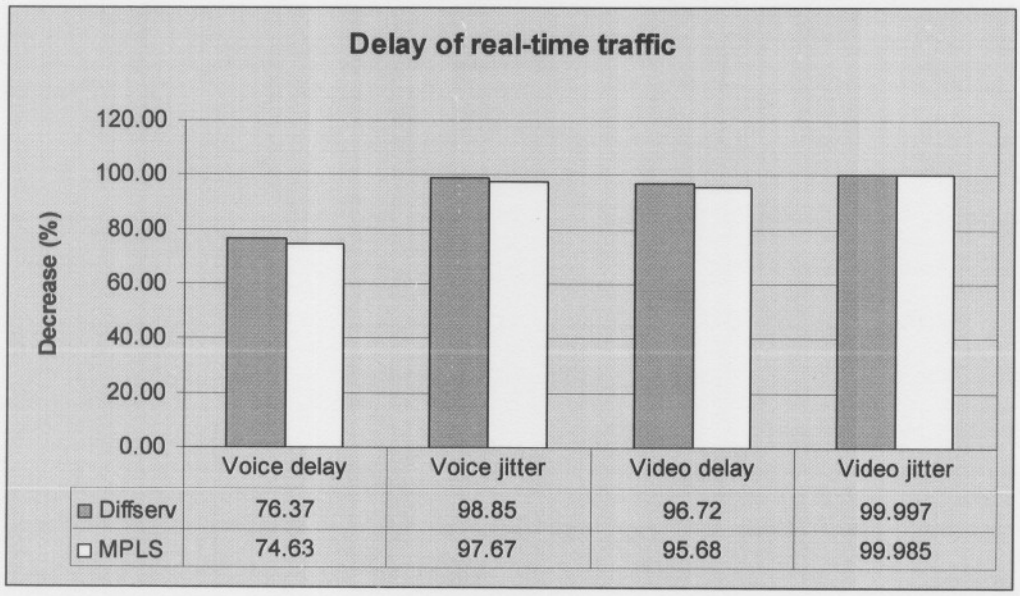


Figure 54 – Experiment 1: Effect of QoS on real-time delay and jitter

However, the DiffServ approach reveals its shortcomings where the non-real-time FTP traffic is concerned. DiffServ was unable to fairly distribute available resources on the over-utilized links. In order to ensure adequate real-time performance, the lower priority FTP traffic is delayed on the links shared by both types. As observed from the previously shown data, this results in unpredictable FTP performance, which worsens as the simulation progresses. The MPLS approach avoids this by separating FTP and multimedia flows, and as a result achieves a 50% increase in throughput and a 96% reduction in response times over the baseline network, as shown in Figure 55.

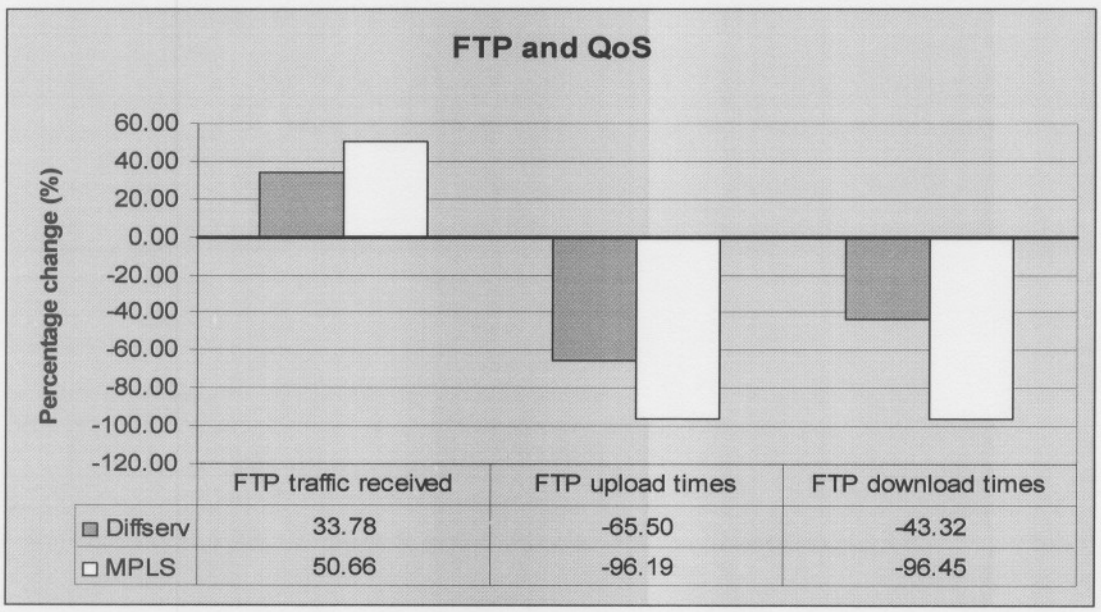


Figure 55 – Experiment 1: Effect of QoS on FTP traffic

Adding QoS to the core network was shown to have a small yet noticeable impact on the wireless access networks (see Figure 56). Both DiffServ and MPLS achieve a 3 % gain in terms of WLAN throughput. As far as delay and media access delay are concerned, we observe that DiffServ causes increased delay in both cases, due to the added packet prioritizing that takes place in the buffers. MPLS has a minimal effect on the WLAN delays, while still providing the same level of QoS to real-time flows as DiffServ, and significantly improving the performance of FTP traffic.

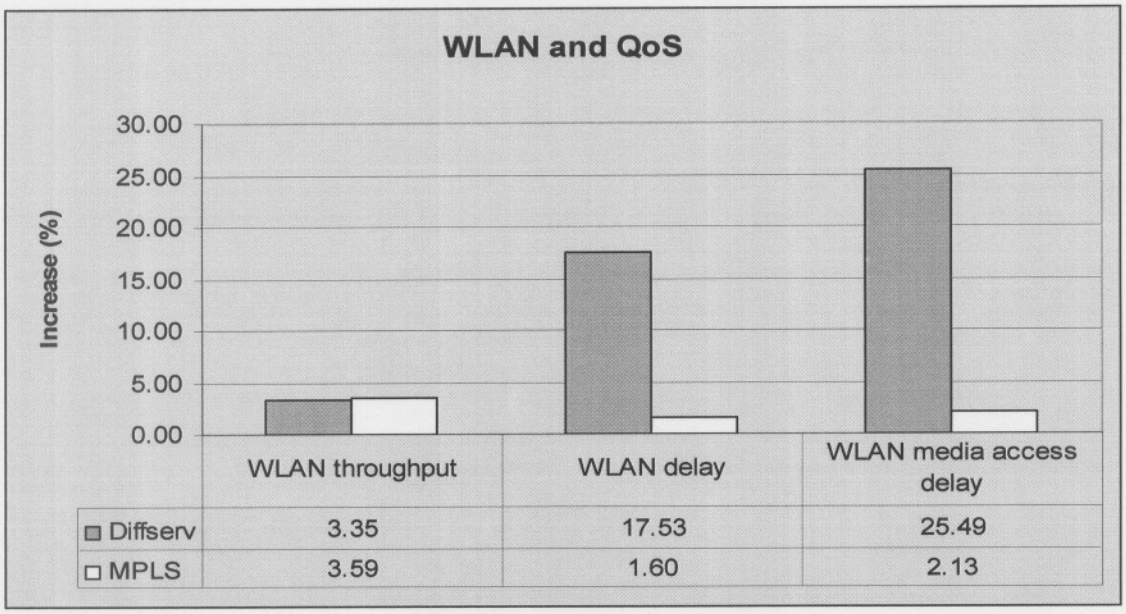


Figure 56 – Experiment 1: Effect of QoS on the WLAN access network

4.3 Experiment 2 – Reaction to link failure

4.3.1 Description

For Experiment 2, we simulate a link failure in the core of the network. In this way, we can study how DiffServ and MPLS react when having to find a new route to the destination, and how this impacts the network and application QoS. The following changes were made to the default simulation setup described in Chapter 3:

i. *Network topology and OPNET library models* – The link between Router 0 and Router 4, as seen in Figure 57, is set to fail after 350 out of the total 600 seconds by using the failure / recovery object supplied in the Modeler object database. Since the core network in Experiment 1 had little or no spare capacity, the default 75% link utilization limit was disabled. This will allow each scenario to fully use the available link bandwidth of 1 Mbps.

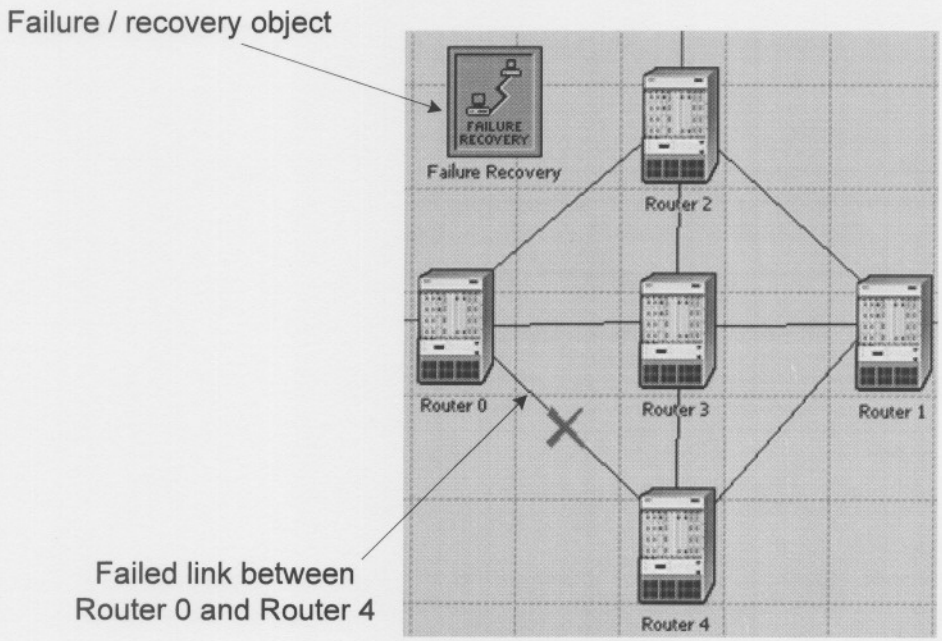


Figure 57 – Simulated link failure

ii. *Deployment of QoS technologies* – No changes were made to the QoS configuration of the baseline and DiffServ scenarios, since they employ per-node QoS by means of packet scheduling and not routing or forwarding. OSPF routing was used to find a new route in place of the failed link and update the routing tables accordingly.

In contrast to this approach, MPLS traffic engineering allows the creation of explicit backup routes in case the primary LSP should fail. Looking at the original MPLS configuration, there are four LSPs that will be affected by the link failure, namely Voice LSP 2 and 3, as well as Video LSP 2 and 3. For the video LSPs, two backup routes are configured, both using the same path through the core network but in opposite directions (see Figure 58). They are specifically routed to include the previously unused links between Routers 2, 3, and 4.

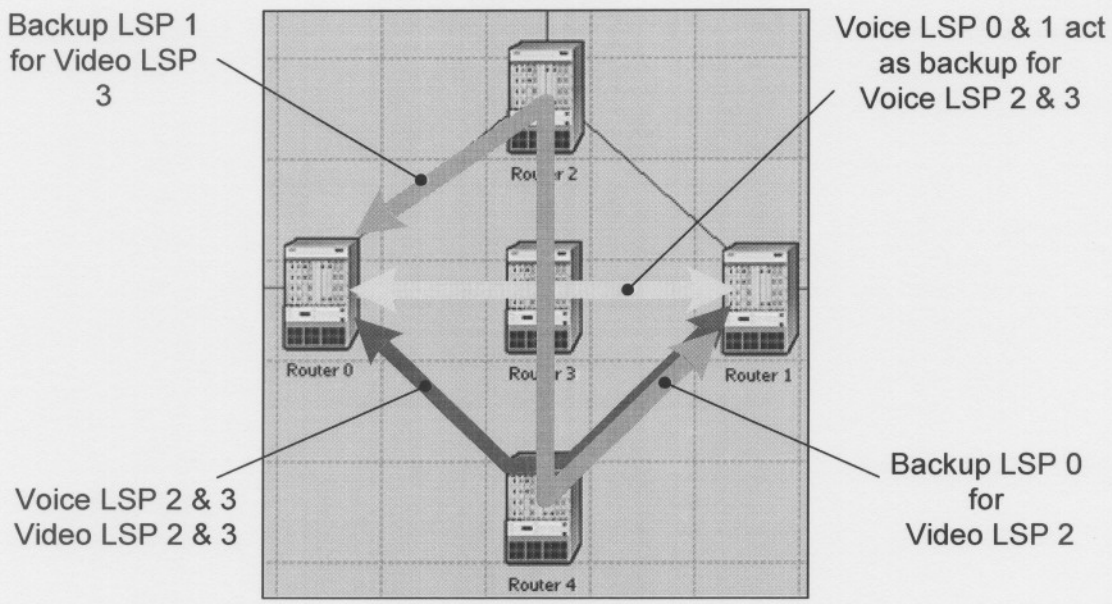


Figure 58 – Backup LSPs in the MPLS scenarios

These LSPs are dynamic as opposed to the static LSPs used in Experiment 1. Dynamic LSPs as simulated by Modeler have several additional attributes in comparison with the static model. They can be created or destroyed at specific times, and will automatically attempt to reroute themselves in case of link or node failures, based on user-specified parameters. These parameters are summarized in Table 14.

Table 14 – Detail parameters of backup LSPs

Name	Model type	Setup	End	Source	Destination
Backup LSP 0	MPLS_E_LSP_DYNAMIC	350 sec	End of sim	Router 0	Router 1
Backup LSP 1	MPLS_E_LSP_DYNAMIC	350 sec	End of sim	Router 1	Router 0

Upon discovering that the primary route has failed, any traffic flow originating at Router 0 or 1 will search for another viable route based on the available LSPs. To ensure that traffic switches to the backup LSPs when the link failure occurs, the LSP assignment configured at Routers 0 and 1 is changed to include these backup routes.

The voice application traffic does not reroute to the backup LSPs, but instead switches to the remaining voice LSPs 0 and 1, making use of the bandwidth still available on the links between Routers 0, 3, and 1.

iii. *Measured parameters* – The same statistics were recorded as in the previous experiment, with exceptions being noted in the text where applicable. The main difference between Experiments 1 and 2 lies in the way we interpreted the measured data. In the first experiment the focus was on overall QoS performance. In this experiment, we pay particular attention to the changes in network behaviour and QoS brought about by the link failure. For example, it would be more relevant to examine average video delay before and after the failure than to consider the overall average value as in Experiment 1.

4.3.2 Link utilization and core network conditions

Figure 59 shows the links measured in this experiment, using the same link utilization probes as before.

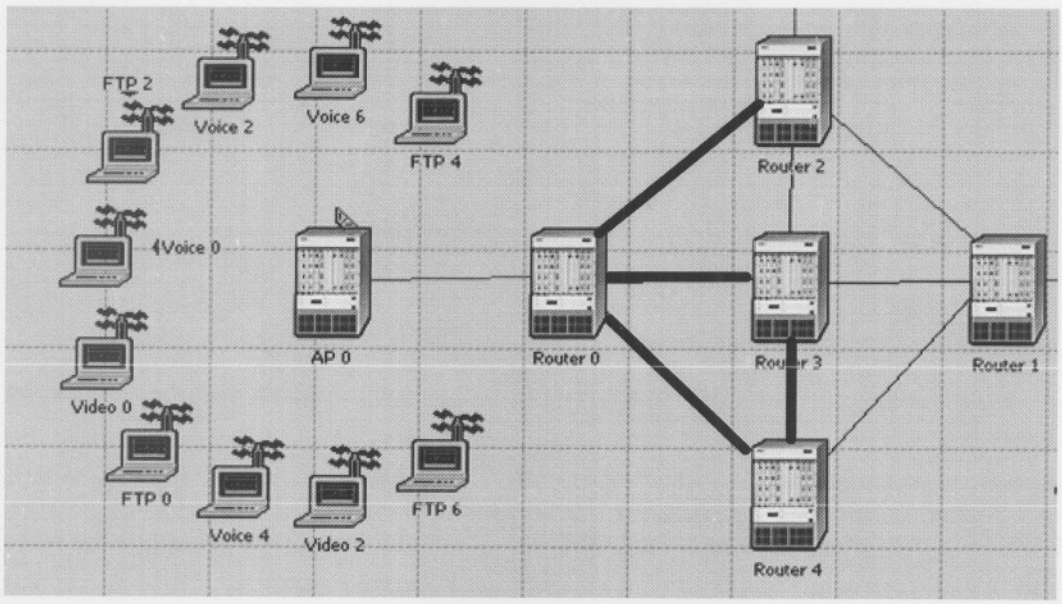


Figure 59 – Link statistics measured for Experiment 2

The data in Figure 60 confirms that the link between Router 0 and Router 4 did indeed fail after 350 seconds, since no traffic flows on the link from this time onwards. Notice that the baseline and DiffServ scenarios use the link to 100% capacity until it fails, while MPLS maintains the 65% figure recorded in Experiment 1.

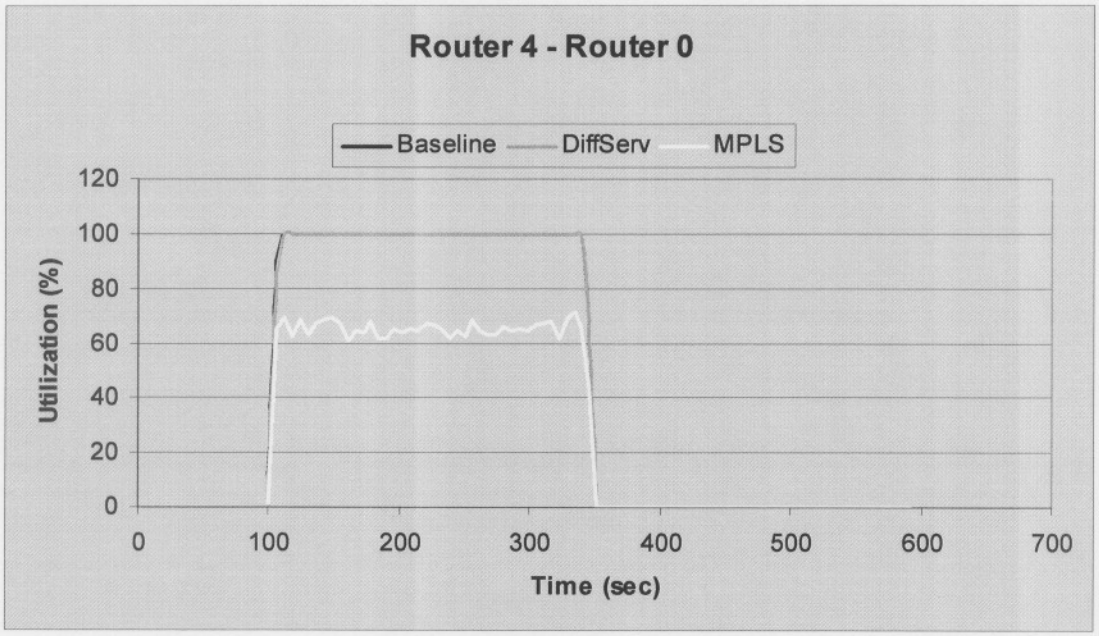


Figure 60 – Experiment 2: Link utilization from Router 4 to Router 0

For comparison, we also plot the results from Router 3 to Router 0. As in the first experiment, the baseline and DiffServ scenarios display great variation in link utilization. The link shown in Figure 61 is almost completely unused until the failure occurs, at which time the DiffServ network reroutes some traffic across it. The baseline network sends a brief burst of data, after which the link utilization returns to 0%. Even though there is more bandwidth available than in the case of Experiment 1, there is still a marked tendency by the networks using OSPF to overuse some links, and ignore others with plenty of capacity available.

The MPLS scenario maintains a steady 65% utilization up to 350 seconds, which then increases to 80% as it picks up some of the rerouted traffic. The following sections will clarify exactly how the traffic from the failed link was distributed.

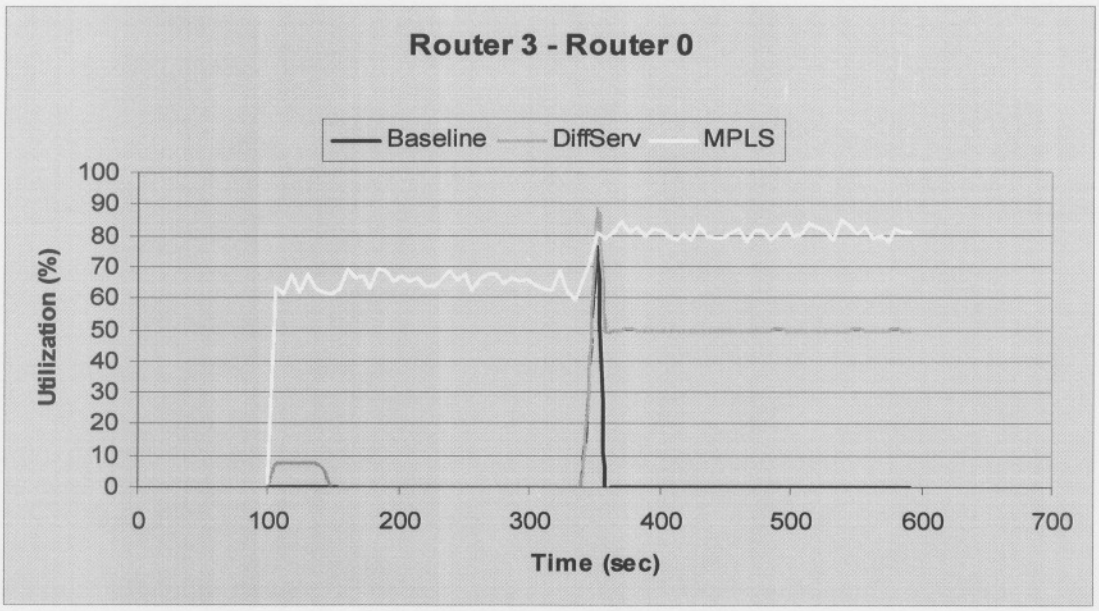


Figure 61 – Experiment 2: Link utilization from Router 3 to Router 0

The link connecting Router 3 and Router 4 was unused by any of the scenarios in Experiment 1. By examining Figure 62, it becomes clear that this is still the case with the baseline and DiffServ networks, even though it represents a possible alternative link to bypass the failed one between Router 0 and Router 4. MPLS traffic engineering is able to make use of this link in order to route the backup LSPs, resulting in the 50% utilization from the point of the failure onwards. We will expand on this observation in the next section.

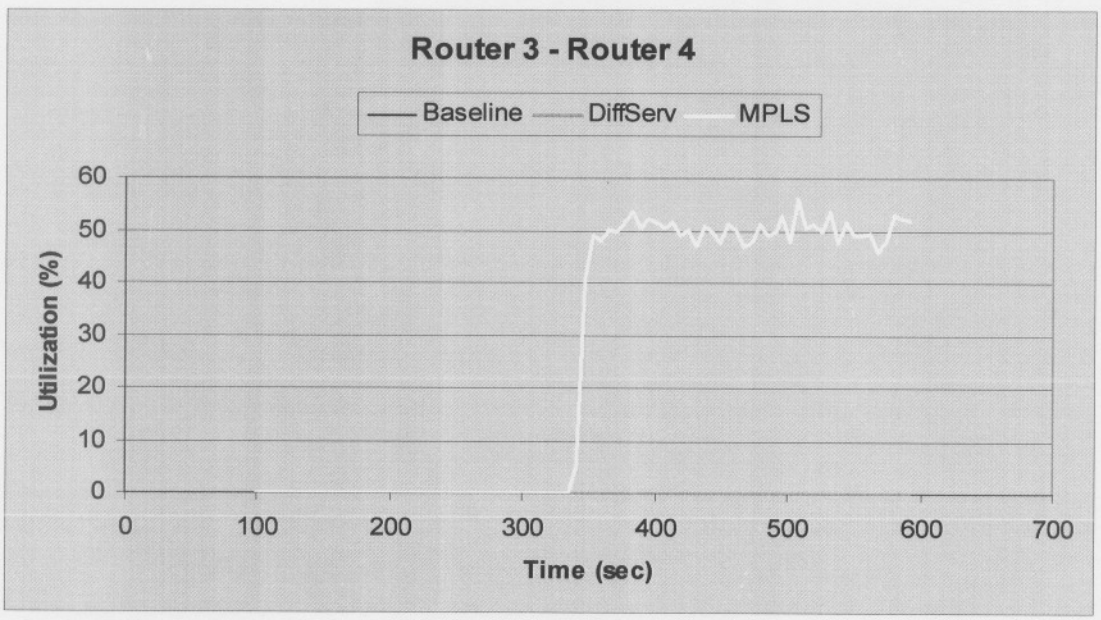
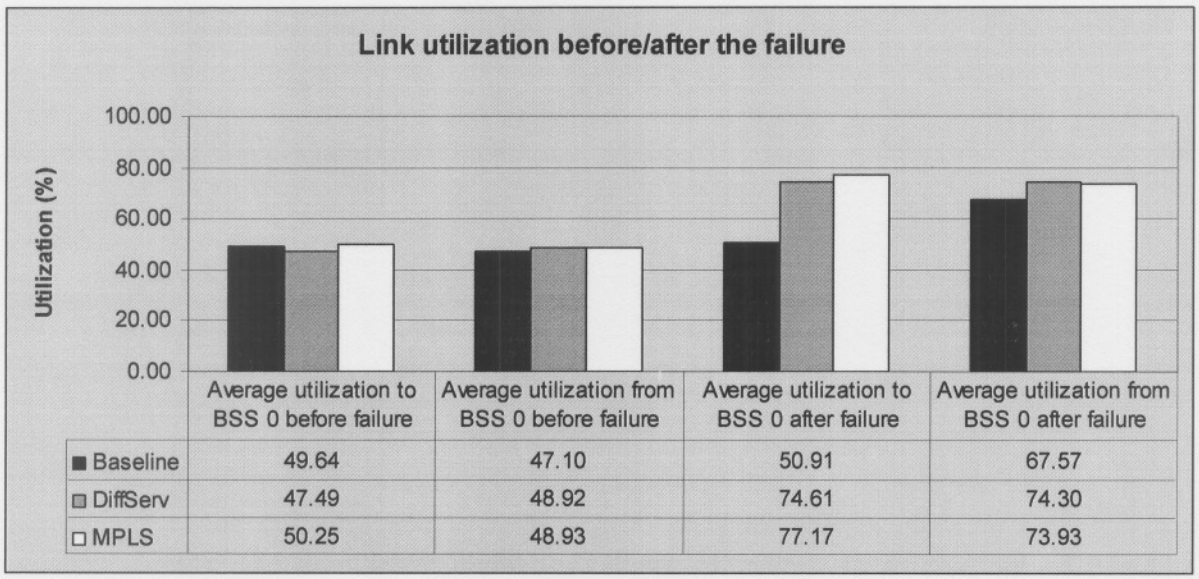


Figure 62 – Experiment 2: Link utilization from Router 3 to Router 4.

Using both the vector and statistical data obtained from the simulation, we plot the average utilization of the links connected to BSS 0, both before and after the failure (see Figure 63). Note that we discount the link between Router 0 and Router 4 in the second instance, since it is no longer a valid route. As can be expected, the average amount of traffic per link increases significantly after the link failure, although to a lesser extent in the baseline scenario. The reason for this, as will become clear presently, is the amount of traffic lost in the core network.



	Average utilization to BSS 0 before failure	Average utilization from BSS 0 before failure	Average utilization to BSS 0 after failure	Average utilization from BSS 0 after failure
Baseline	49.64	47.10	50.91	67.57
DiffServ	47.49	48.92	74.61	74.30
MPLS	50.25	48.93	77.17	73.93

Figure 63 – Experiment 2: Increase in average link utilization after the failure

As before, the DES log shown in Table 15 contains warning messages concerning packet loss and retransmissions. The baseline scenario issues three packet loss messages and one TCP retransmission message. All of these, with a single exception, occur after the link failure, indicating the increased congestion in the core resulting from the higher link utilization.

The log from the DiffServ scenario, however, contains five warning messages before the failure and a single one after it. The first warning is issued only 8 seconds after the start of the traffic profiles, pointing to a very early congestion of traffic in the network.

As in the previous experiment, the MPLS scenario log contains no indications of packet loss or retransmissions.

Table 15 – Experiment 2: DES log warning messages

Scenario	Protocol	Warning message	Time (sec)	Nodes / routers affected
No QoS	IP	Packets have been dropped in the router for congestion reasons.	149, 362, 385	Routers 0, 1 and 2
DiffServ	IP	Packets have been dropped in the router for congestion reasons.	108, 110, 184, 297, 390	Routers 0 and 2
No QoS	TCP	TCP is retransmitting data segments which will cause additional overhead on the lower layers and links	369	FTP 7
DiffServ	TCP	TCP is retransmitting data segments which will cause additional overhead on the lower layers and links	110	FTP 7
MPLS	-	No warnings	-	-

The baseline graph in Figure 64 clearly demonstrates the effect of the link failure on the baseline network, with the loss rate increasing from just over 20 packets/second to 140 packets/second by the end of the simulation. The early packet loss recorded by the DES log in the DiffServ scenario can also be seen. Apart from this, the DiffServ and MPLS scenarios remain largely unaffected by the link failure where packet loss is concerned.

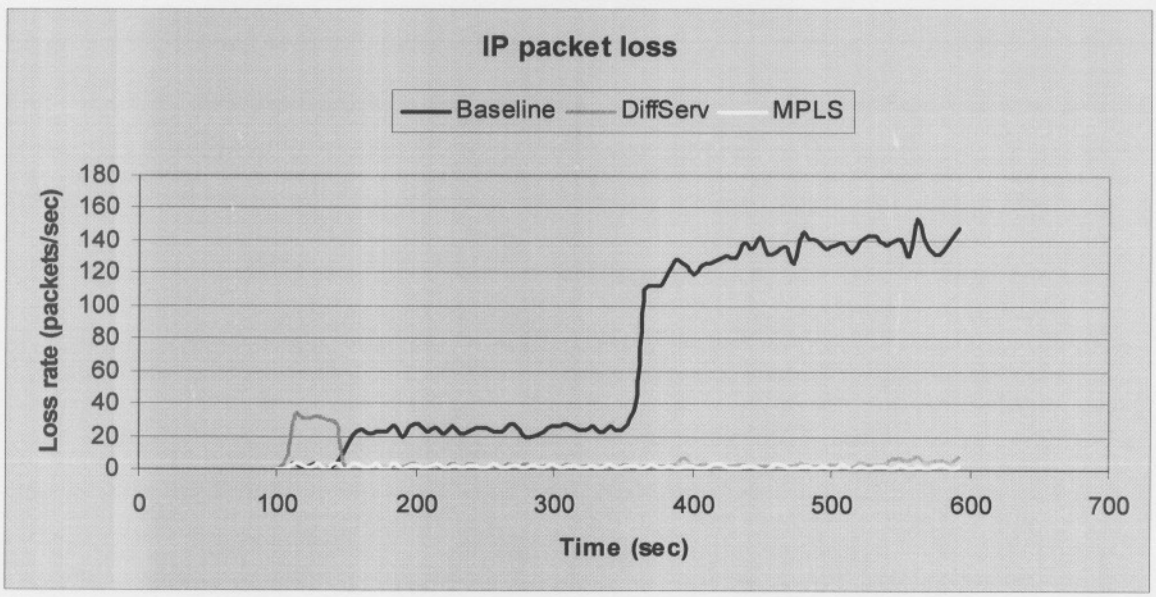
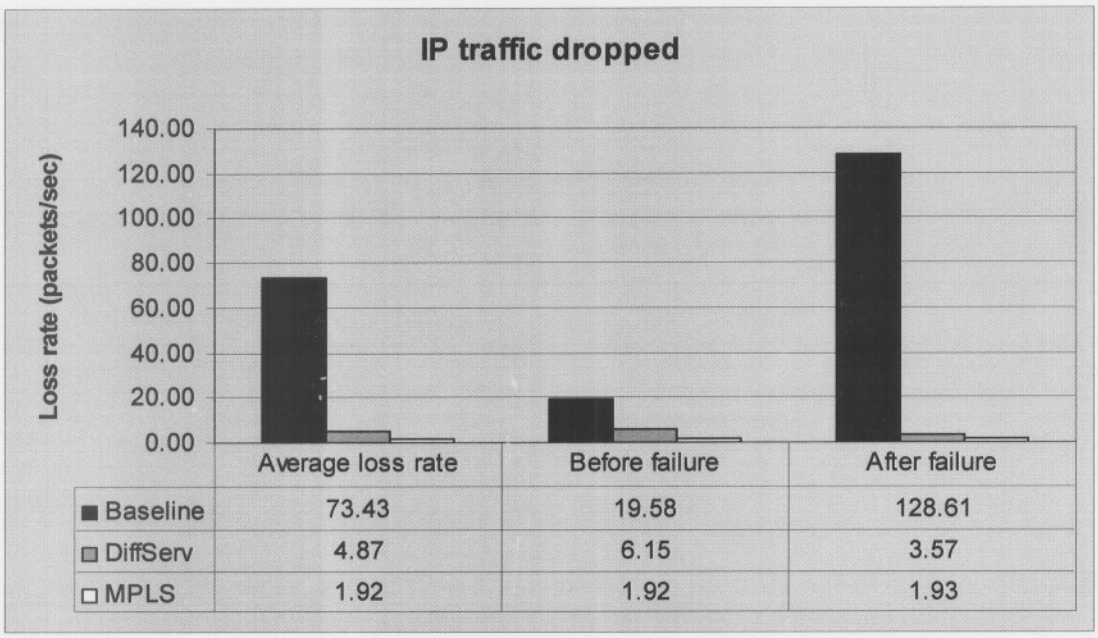


Figure 64 – Experiment 2: IP packet loss rate

We use the numerical simulation data to compare the sample means of packet loss rate before and after the link failure in Figure 65. The baseline scenario suffers the worst, losing six times more packets after the failure than before it. Since traffic is forwarded on a FIFO basis with no QoS mechanisms present, there are no guarantees in terms of loss prevention. This high amount of packet loss also explains the lower increase in link utilization relative to the QoS-enabled networks.

The deployment of QoS in the DiffServ and MPLS scenarios results in vastly reduced amounts of dropped traffic in comparison with the baseline network. DiffServ achieves this by the drop precedence indicated in the DSCP, while MPLS traffic engineering manages bandwidth to prevent link congestion. Of the two approaches, MPLS has the lowest overall packet loss, in addition to near-identical loss rates before and after the link failure. This demonstrates the ability of MPLS traffic engineering to prevent packet loss by successfully rerouting traffic to alternative links.



	Average loss rate	Before failure	After failure
Baseline	73.43	19.58	128.61
DiffServ	4.87	6.15	3.57
MPLS	1.92	1.92	1.93

Figure 65 – Experiment 2: Effect of the link failure on mean packet loss rate

The following sections will examine the results gathered from the application traffic.

4.3.3 Rerouting the application traffic with MPLS

Before looking at the applications themselves, we present some results recorded from the MPLS LPSs, which illustrate the operation of MPLS traffic engineering. In order to do so, the following statistics are recorded during the simulation (see Table 16):

Table 16 – Experiment 2: LSP statistics recorded

Statistic name	Unit	Type	Description
Traffic in	Bits/sec	Object	Total traffic sent into the LSP at the ingress end of the tunnel.
Traffic out	Bits/sec	Object	Total traffic received from the LSP at the egress end of the tunnel.
Delay	Seconds	Object	Delay experienced by packet in the LSP, i.e. time spent by the packet within the Label Switched Path.

Taking Voice and Video LSP 2 as examples, Figure 66 clearly shows the traffic flow being cut off at 350 seconds as a result of the link failure. At this point, the LSP is considered as having failed, and no more traffic will be routed onto it.

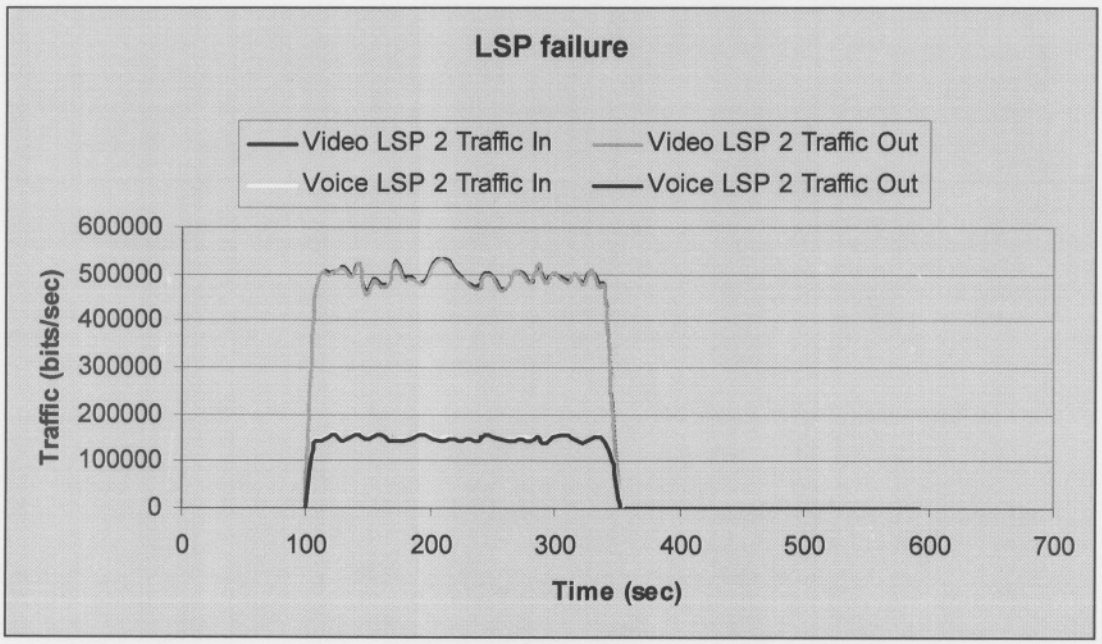


Figure 66 – Experiment 2: LSP reaction to link failure

Looking first at the voice traffic, Figure 67 shows how MPLS reacts to the failure. The 150000 bits/sec voice traffic from Voice LSP2 is immediately switched to Voice LSP 0, thus doubling its previous traffic load.

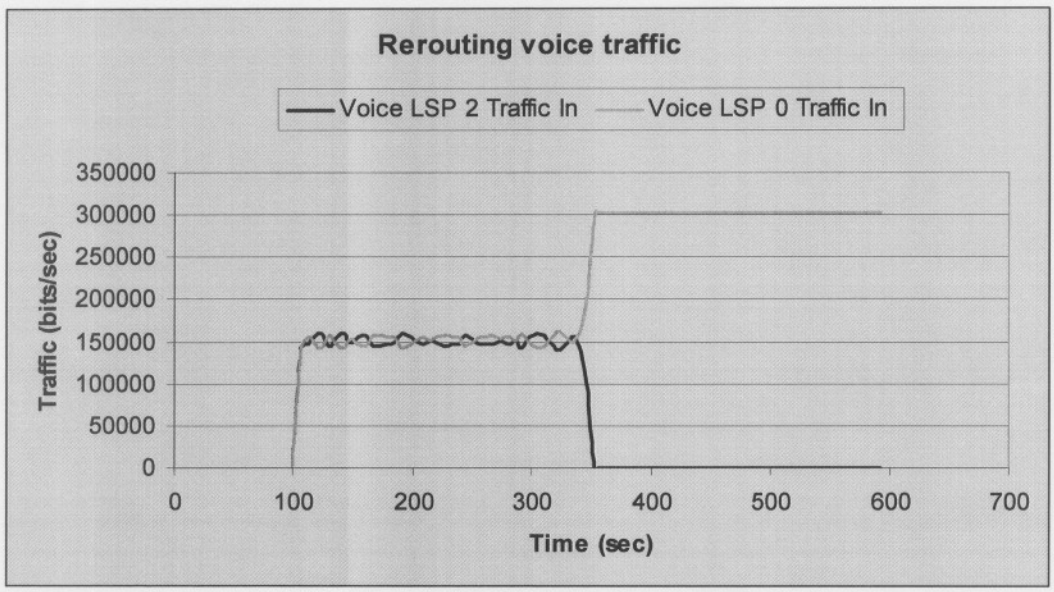


Figure 67 – Experiment 2: Rerouting voice application traffic to voice LSP 0

The increased traffic load can reasonably be expected to have an effect on the delay of the existing voice LSPs. This effect can be seen in Figure x. Although each voice LSP now carries twice as much traffic as before, the delay increases by only 3-4 ms after the failure.

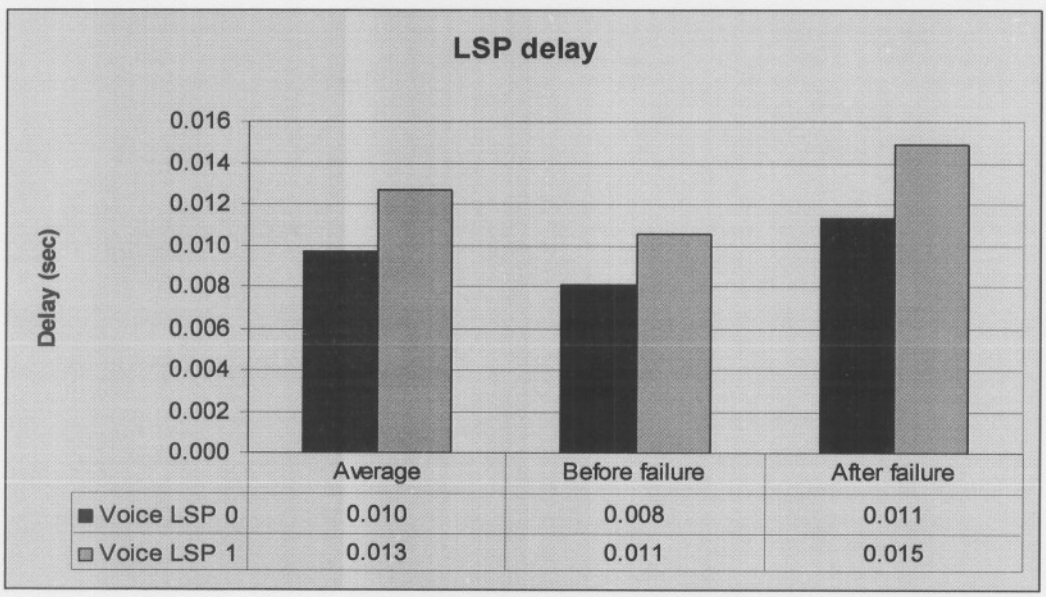


Figure 68 – Experiment 2: Increase in voice LSP delay

In a similar fashion, the video traffic from the failed video LSPs are rerouted to the backup LSPs when the link failure occurs, as shown in Figure 69. The 500000 bits/sec of traffic that previously used Video LSP 2, reappears on Backup LSP 0 at approximately 350 seconds simulated time.

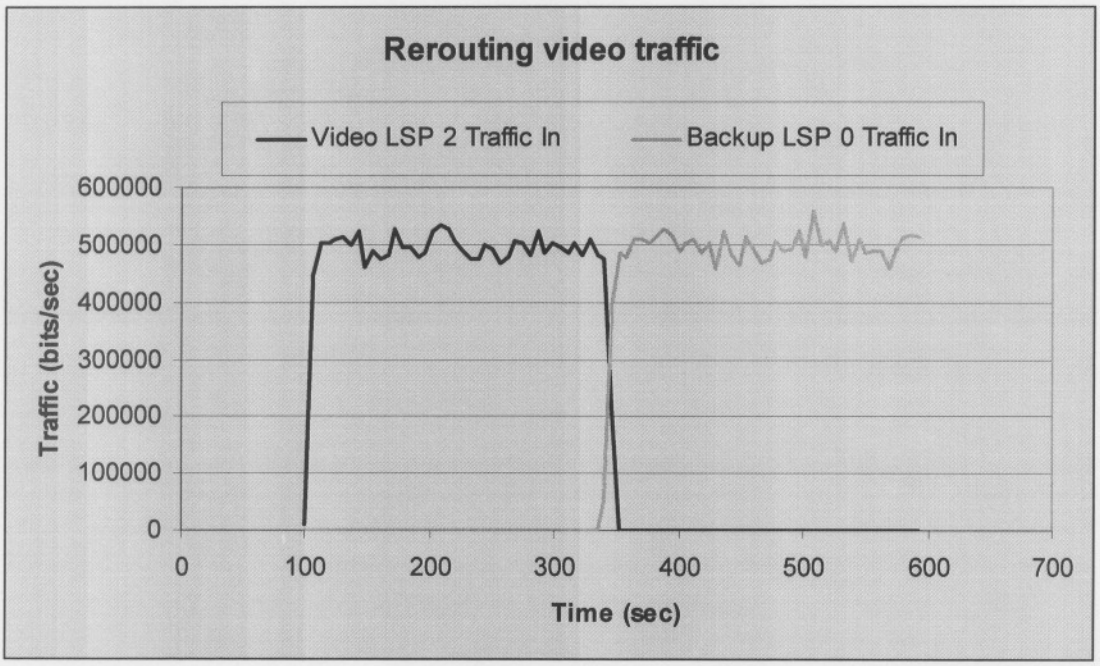


Figure 69 – Experiment 2: Rerouting video traffic from Video LSP 2

From the results shown above, we see that MPLS was able to reroute al the traffic from the failed link and LSPs to new routes through the core network. In the following sections we will investigate how this affected the application traffic.

4.3.4 Video conferencing

As can be seen from Figure 70, the rate of video traffic received in the baseline scenario decreases by approximately 50000 bytes/second after the link failure. This shows the extent to which the increased packet loss and congestion affects video traffic. The DiffServ and MPLS scenarios have no noticeable decrease in throughput, although the flows appear less stable from 350 seconds onwards.

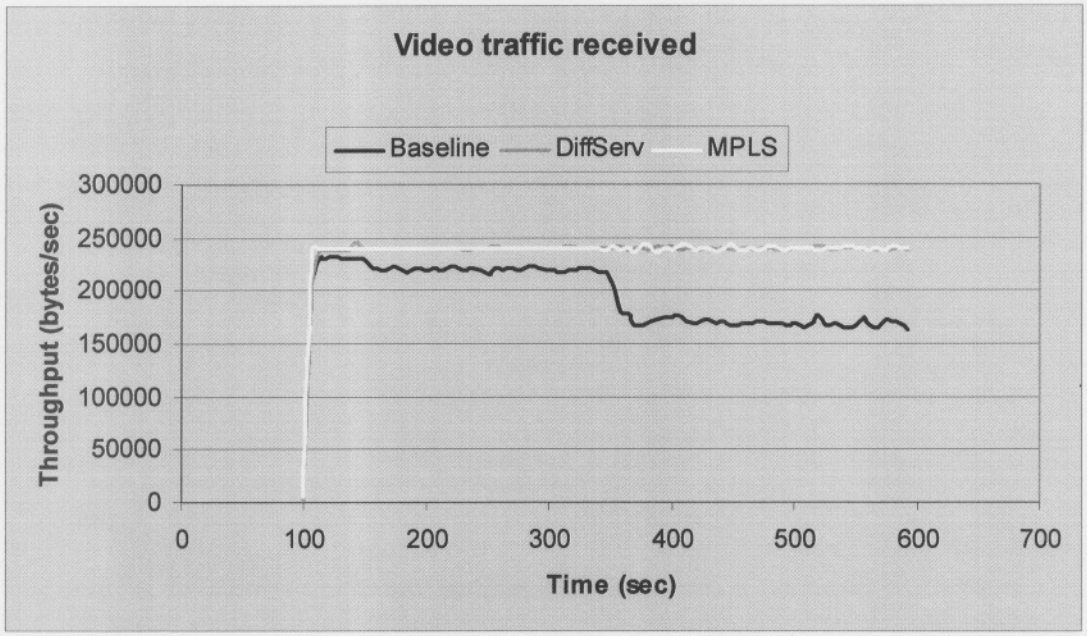


Figure 70 – Experiment 2: Video conferencing traffic received

Figure 71 shows that the already high video delay in the baseline scenario becomes even more pronounced after the link failure, almost immediately increasing from 2 seconds to 6 seconds, and then to over 8 seconds by the end of the simulation. There is also an observable difference in the delay before and after the failure for the QoS-enabled scenarios, with the DiffServ video delay visibly increasing from about 450 seconds onwards.

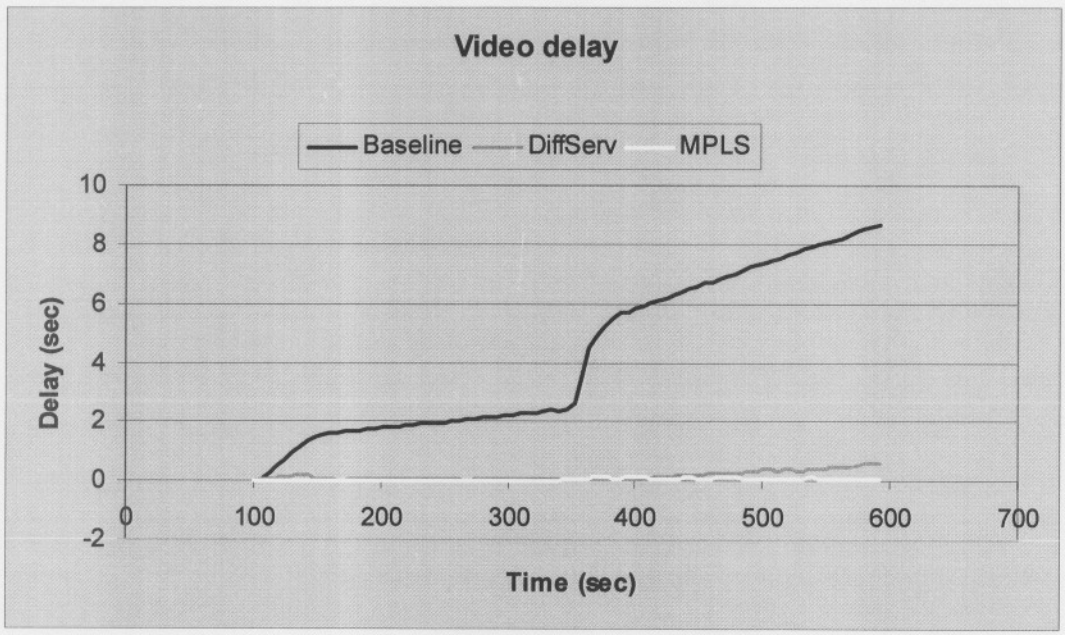
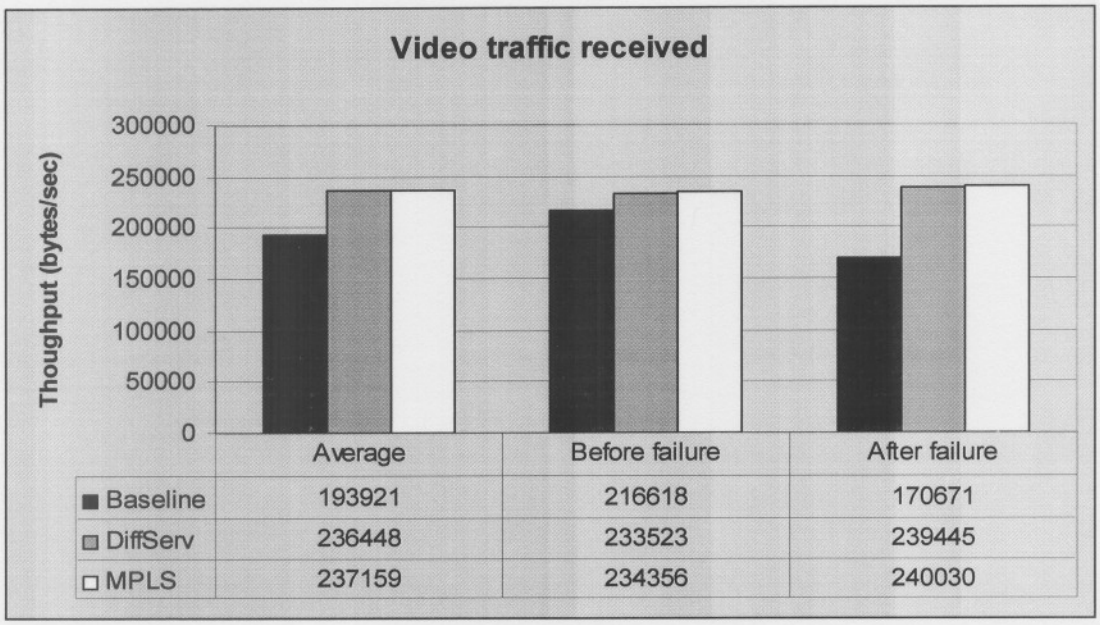


Figure 71 – Experiment 2: Video conferencing delay

Using the same data as shown above, we compare the average video throughput before and after 350 seconds with the sample mean obtained from the statistical data (see Figure 72). The baseline scenario has the lowest overall throughput and the most pronounced reaction to the link failure, with average throughput dropping by 46000 bytes/second. The video throughput of the DiffServ and MPLS scenarios remain stable, showing that both techniques successfully prevent video throughput from decreasing in the wake of the link failure.



	Average	Before failure	After failure
Baseline	193921	216618	170671
DiffServ	236448	233523	239445
MPLS	237159	234356	240030

Figure 72 – Experiment 2: Effect of the link failure on video throughput

The very high video delay values recorded in the baseline scenario means we would require an exponential scale for all scenarios to be visible. However, in an effort to be consistent with the other figures as regards presentation, we only compare the DiffServ and MPLS scenarios in Figure 73. For all three cases, the DiffServ network is outperformed by MPLS. Both are able to maintain the video delay well below the high-quality bound of 150 ms before the failure. However, the average delay of the DiffServ scenario jumps more than 200 ms after the link fails, increasing the overall average to over 150 ms. While still acceptable, this can no longer be classified as a high-quality video transmission. MPLS limits this delay increase to 50 ms on average, and the overall average of 63 ms is still more than 50% below the recommended ITU bound.

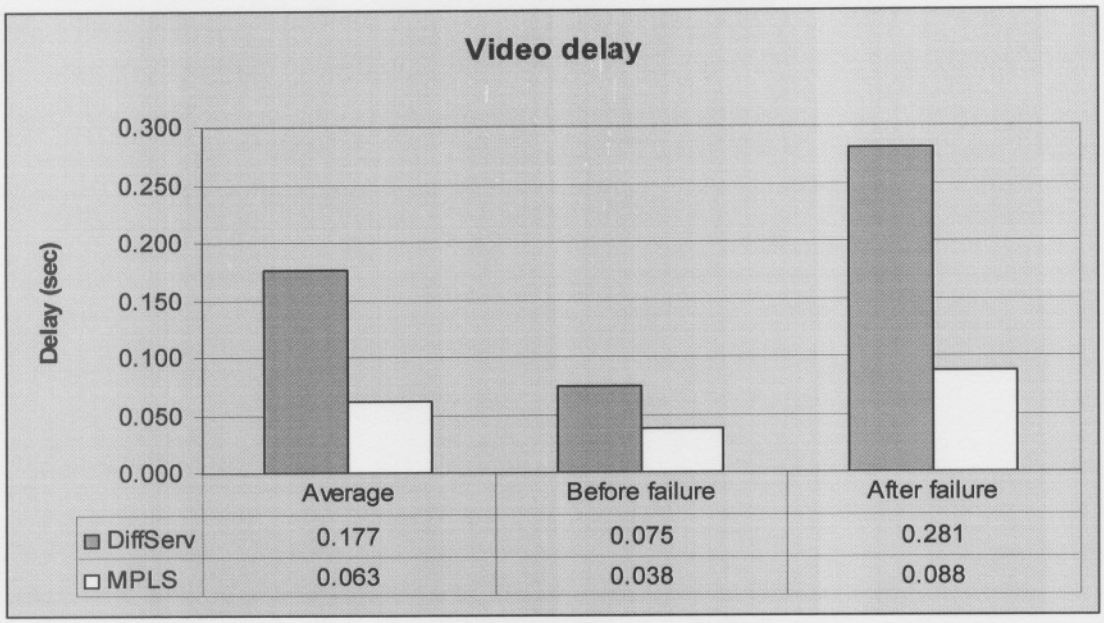


Figure 73 – Experiment 2: Effect of the link failure on video delay

The video delay variation depicted in Figure 74 shows the same behaviour as the delay. The increased variance after the link failure indicates greater variation in the video packet inter-arrival time. Once again MPLS records the lowest values, both before and after the failure occurs.

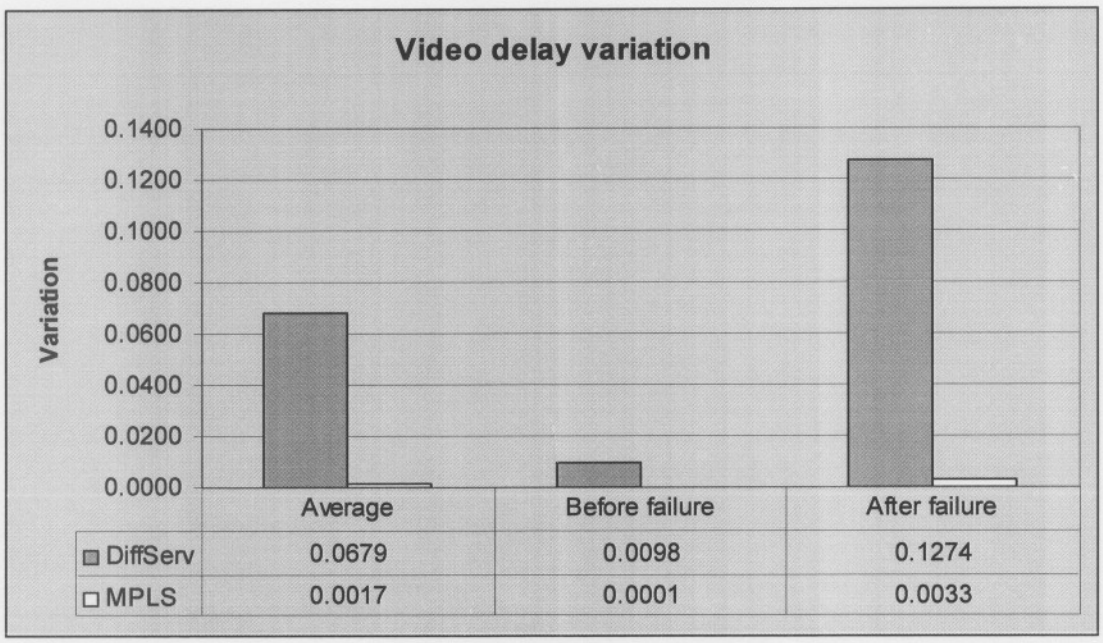


Figure 74 – Experiment 2: Effect of the link failure on video delay variation

In summary, MPLS traffic engineering was shown to be effective in preserving the QoS of the video application by rerouting the traffic from the failed link to the backup LSPs. By making use of the spare capacity available in the core, MPLS prevents any one link from being over-used, thus minimizing the increase in delay and delay variation. Since DiffServ only affects packet scheduling and not routing, the spare capacity remains unused. Thus the higher average link utilization caused by the link failure results in increased end-to-end delays and greater delay variation, to the extent where the recommended delay bound for high-quality video conferencing is no longer met. This occurs despite the fact that video traffic is marked for Expedited Forwarding, which is the highest available DiffServ QoS priority.

4.3.5 Voice over IP

As is the case with video traffic, we see from Figure 75 that voice throughput in the baseline scenario decreases significantly from 350 seconds onwards. The DiffServ scenario shows the effect of the early congestion recorded by the DES log, but maintains a stable throughput even after the link failure. The MPLS scenario displays no drop-off in terms of traffic received.

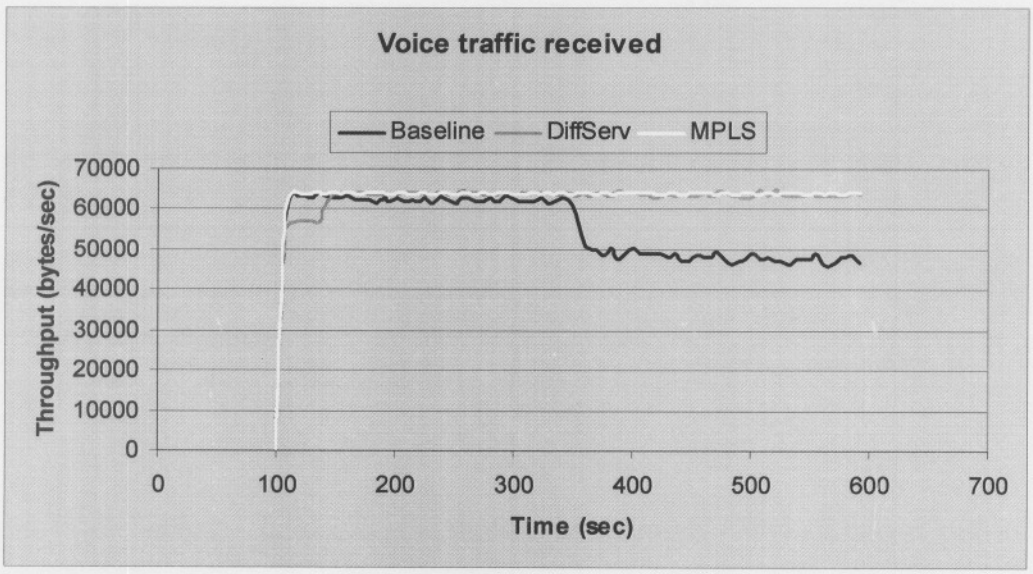


Figure 75 – Experiment 2: Voice traffic received

The baseline scenario voice delay increases by 2 seconds directly after the failure occurs, and continues to increase up to 5 seconds (see Figure 76). The increase in voice delay from about 400 seconds in the DiffServ scenario is also apparent, even more so than was the case with video. The MPLS scenario however, maintains a stable voice delay throughout the simulation.

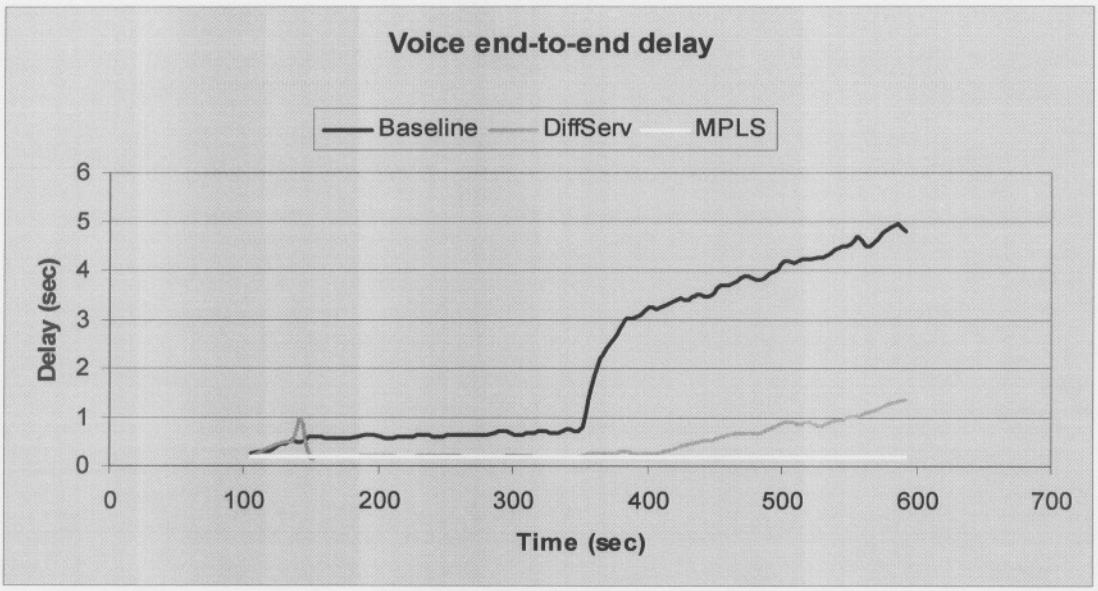
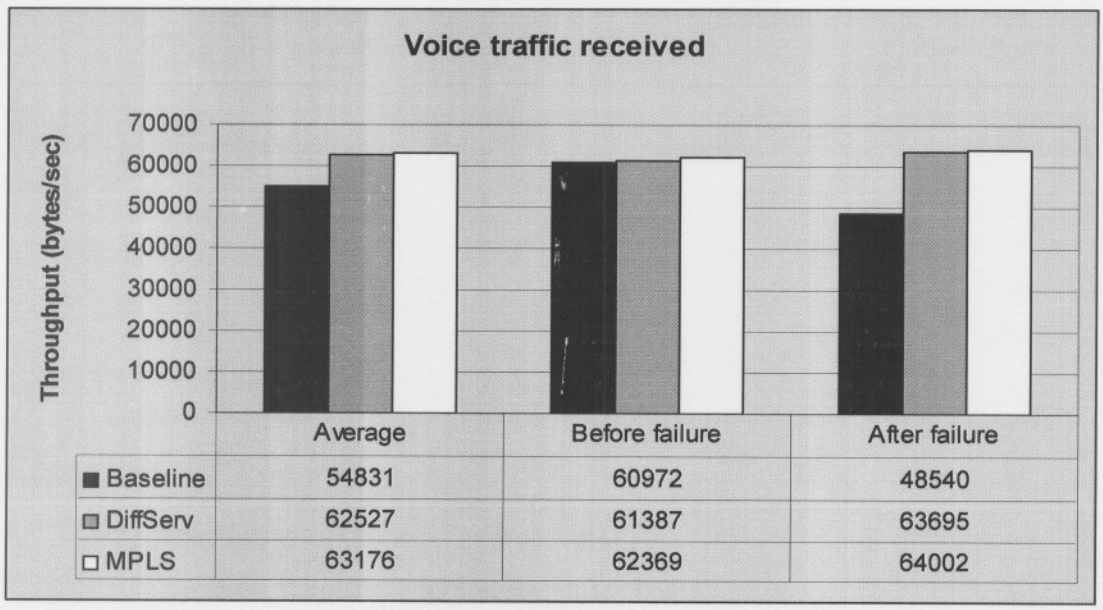


Figure 76 – Experiment 2: Voice end-to-end delay

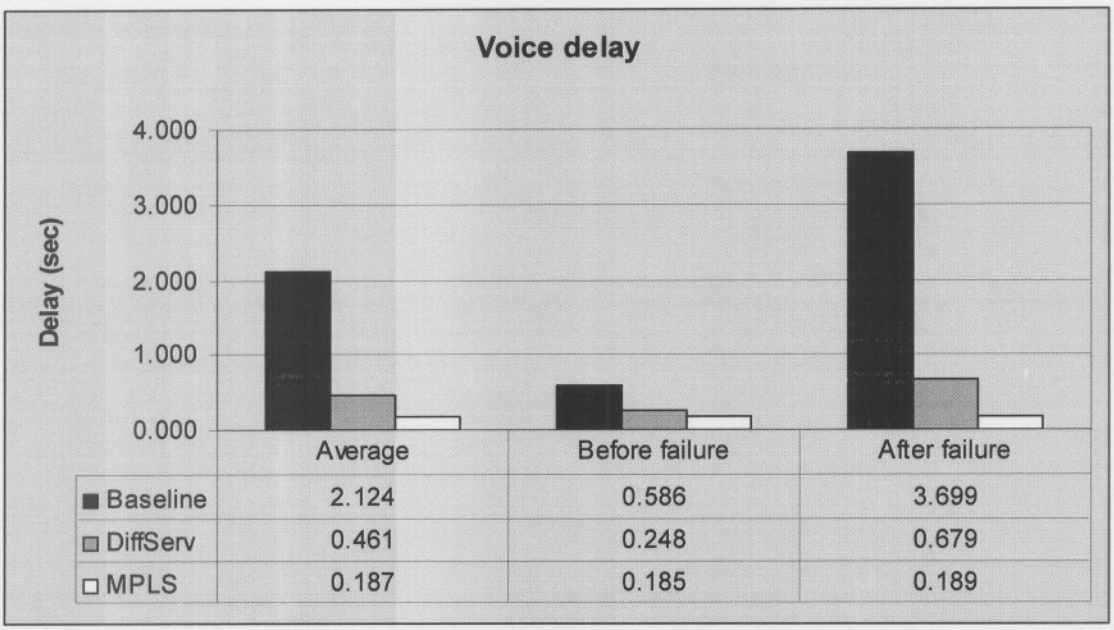
As before, we compare the statistical sample mean of throughput, delay, and jitter with the average values recorded before and after the link failure. Throughput, shown in Figure 77, drops noticeably in the baseline scenario but remains stable in the QoS-enable scenarios.



	Average	Before failure	After failure
Baseline	54831	60972	48540
DiffServ	62527	61387	63695
MPLS	63176	62369	64002

Figure 77 – Experiment 2: Effect of the link failure on voice throughput

Average voice delay in the baseline scenario is never below half a second, and increases to 3.7 seconds as a result of the link failure (see Figure 78). The DiffServ value of 248 ms recorded before the failure is still within acceptable limits, but then rises to 679 ms after the failure, or almost 500 ms average, well above the maximum acceptable delay bound of 400 ms. In contrast, voice delay in the MPLS scenario is nearly unaffected by the link failure, with an average difference of only 4 ms resulting from rerouting the voice traffic.



	Average	Before failure	After failure
Baseline	2.124	0.586	3.699
DiffServ	0.461	0.248	0.679
MPLS	0.187	0.185	0.189

Figure 78 – Experiment 2: Effect of link failure on voice delay

With such stable delay values, the voice jitter in the MPLS scenario averages only 2 ms (see Figure 79). In the case of DiffServ, the jitter is more than four times greater after the failure, resulting in an average value of 177 ms, which far exceeds the maximum 30 ms ITU bound. As was the case with video, we omit the results from the baseline scenario to better present the data from the other two scenarios.

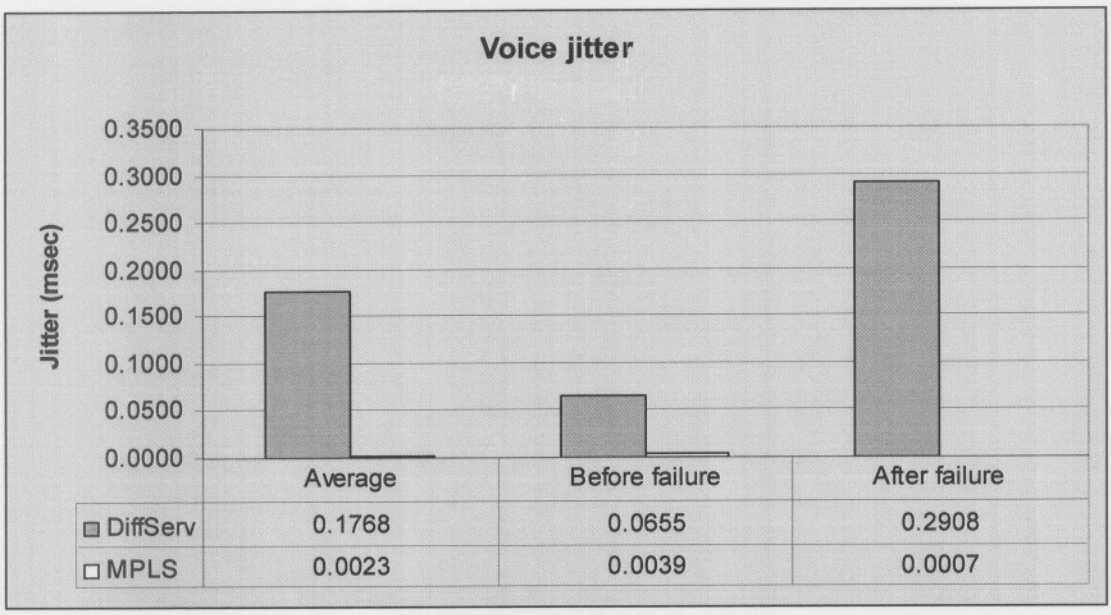


Figure 79 – Experiment 2: Effect of link failure on voice jitter

As is the case with video traffic, MPLS traffic engineering successfully maintains the QoS of the VoIP application by rerouting the traffic from the failed link to areas of the core with spare capacity available. In the DiffServ scenario, we see that voice traffic is subject to significant increases in delay and jitter after the link failure. Once again, this results from the overuse of certain links, which DiffServ is unable to rectify. Voice is affected even more than video traffic, since video has greater DiffServ priority in terms of scheduling. This forces voice packets to wait in the buffer while video packets are serviced, further increasing voice end-to-end delay.

4.3.6 FTP traffic

We now investigate the performance of the non-real-time FTP traffic and how it is influenced by the link failure. The average FTP traffic received as shown in Figure 80 again illustrates the working of the TCP protocol. All three scenarios have an initial throughput of over 140000 bytes/second, which rapidly decreases due to FTP congestion control. From the failure time of 350 seconds and onwards, the baseline scenario shows a clear decrease in throughput. A slightly less prominent drop-off occurs in the DiffServ scenario, while for MPLS the throughput apparently remains constant after the failure.

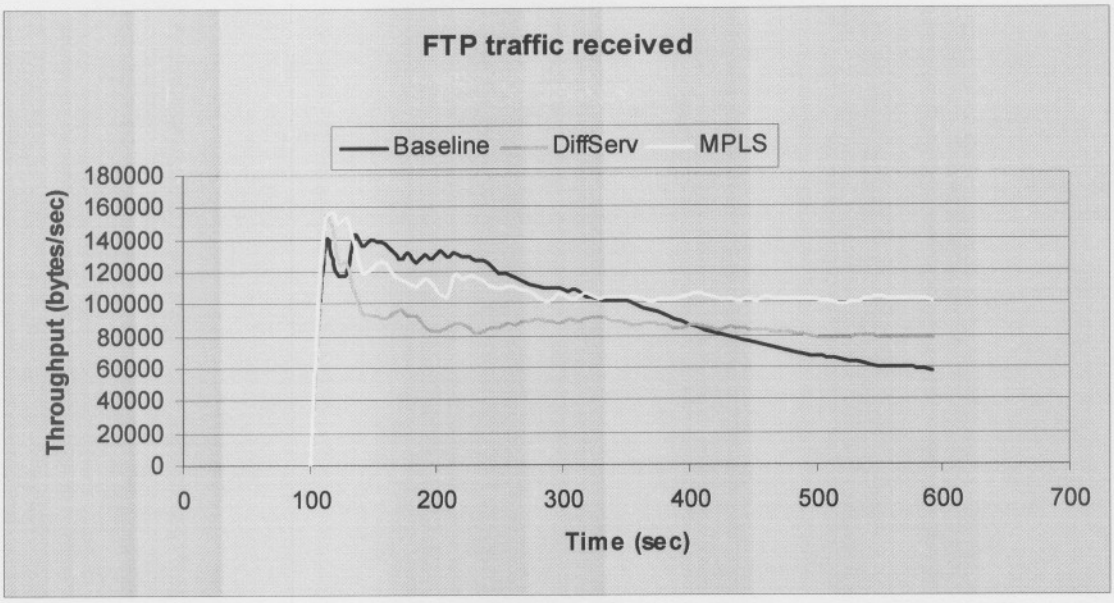


Figure 80 – Experiment 2: Average FTP traffic received

The download response seen in Figure 81 conveys further information. The baseline scenario records no response times between approximately 400 and 550 seconds, indicating the absence of successful downloads during that time. In total, the data vector contains only five downloads after the failure, with the last two recording response times of 200 seconds or more. This corresponds to the decrease in FTP traffic received in the previous figure.

The DiffServ scenario likewise shows downloads occurring less frequently, with increasing download delays. However, this does not appear to be the case with the MPLS scenario.

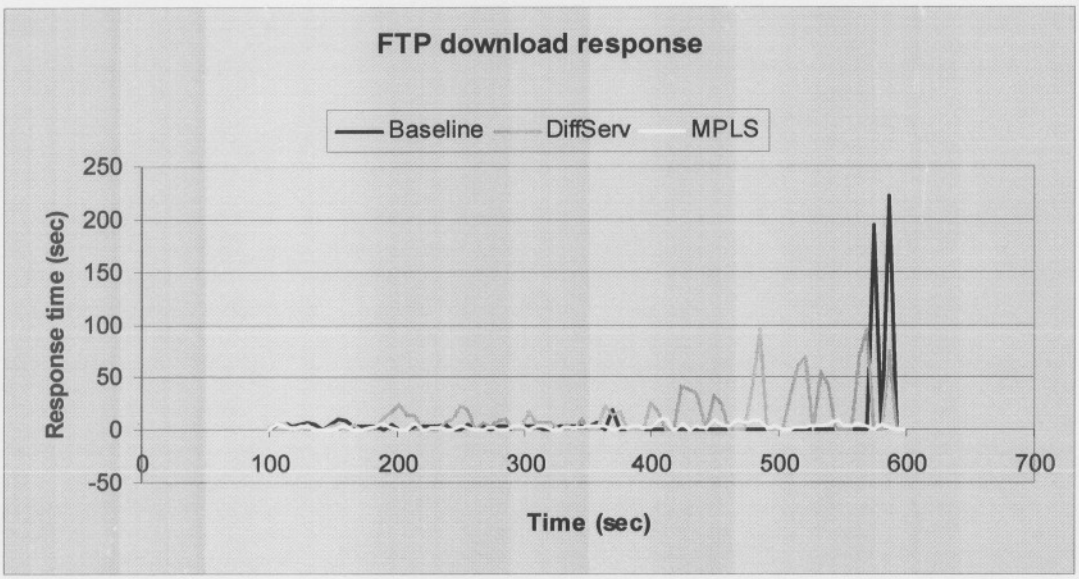
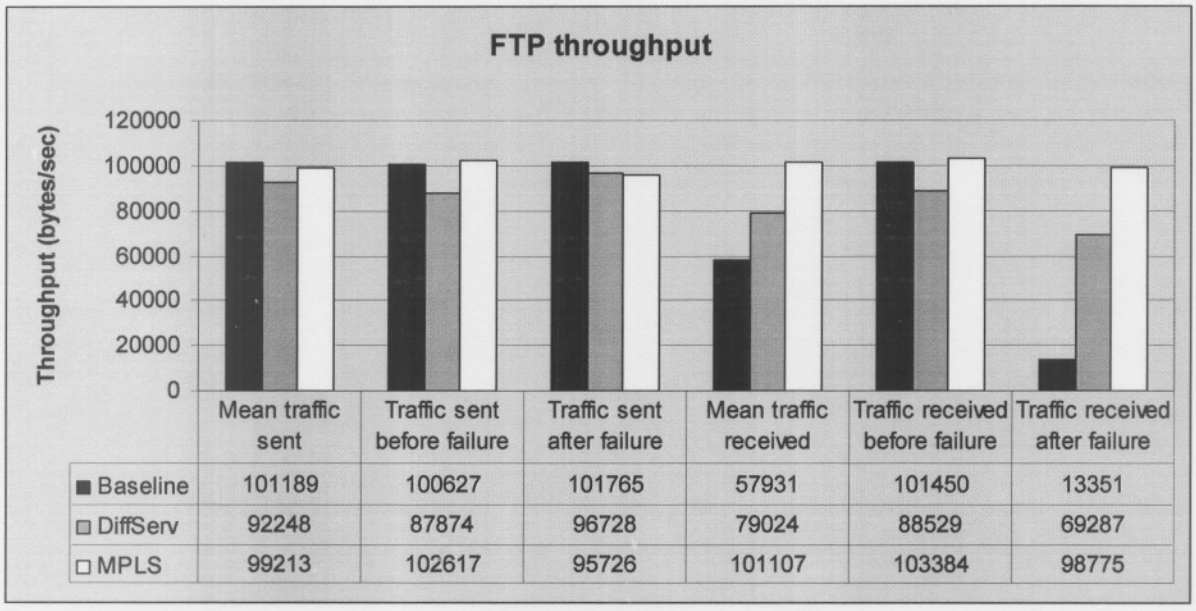


Figure 81 – Experiment 2: FTP download response time

To clarify the above results, we compare the sample means of all FTP statistics with the values recorded before and after the link failure.

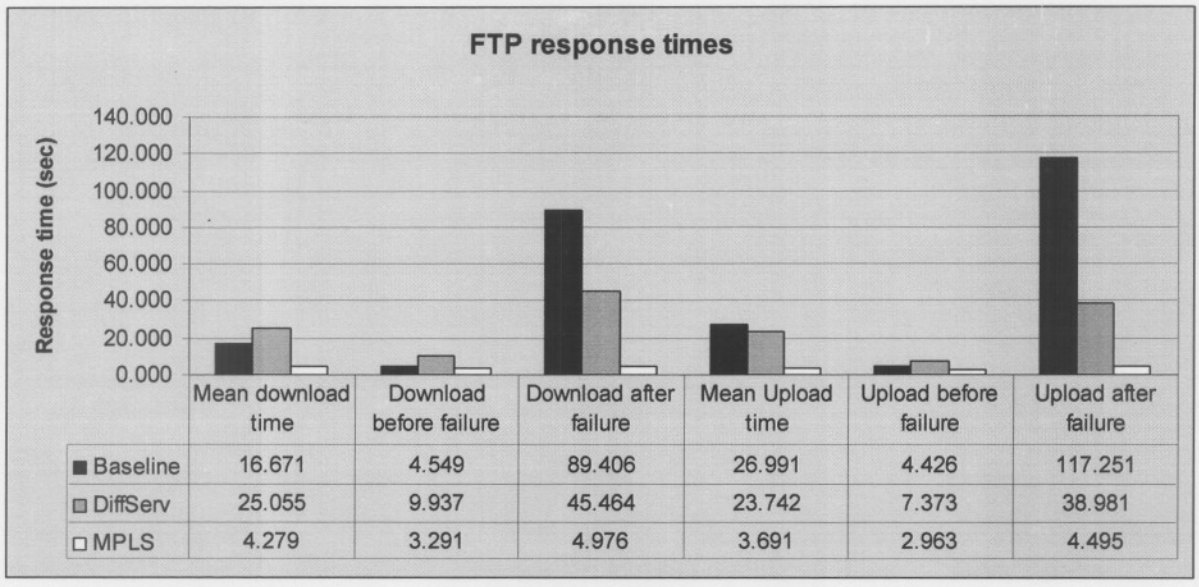
Referring to Figure 82, the baseline scenario sends the highest amount of traffic overall before the failure occurs, due to the fact that FTP receives the same QoS treatment as the real-time applications. After the failure however, the lack of QoS and the increase in congestion cause the average rate of FTP traffic that is actually received to fall by a factor of seven. DiffServ is able to cope better with the failure, but the throughput of the received FTP traffic nonetheless decreases by 22%. The MPLS scenario has the highest mean throughput after the failure, and maintains the rate of received FTP traffic at 95 % of its value prior to the failure.



	Mean traffic sent	Traffic sent before failure	Traffic sent after failure	Mean traffic received	Traffic received before failure	Traffic received after failure
■ Baseline	101189	100627	101765	57931	101450	13351
□ DiffServ	92248	87874	96728	79024	88529	69287
□ MPLS	99213	102617	95726	101107	103384	98775

Figure 82 – Experiment 2: Effect of link failure on FTP throughput

The FTP response times appear in Figure 83. Again the baseline scenario comes off worst of the three, with upload and download delays increasing by 80 seconds or more due to the link failure. In the case of DiffServ, FTP delays increase by about 30 seconds, while the MPLS scenario has a variation of less than two seconds brought on by the link failure.



	Mean download time	Download before failure	Download after failure	Mean Upload time	Upload before failure	Upload after failure
■ Baseline	16.671	4.549	89.406	26.991	4.426	117.251
▨ DiffServ	25.055	9.937	45.464	23.742	7.373	38.981
□ MPLS	4.279	3.291	4.976	3.691	2.963	4.495

Figure 83 – Experiment 2: Effect of link failure on FTP response times

As in the previous experiment, the benefit of using MPLS traffic engineering to separate the FTP and real-time flows is clearly visible. In the baseline scenario, higher link utilization and congestion causes TCP to reduce its transmission rate, which lowers the throughput of FTP traffic. Increased congestion also leads to retransmissions, as witnessed in the DES log, resulting in longer FTP response times. In the DiffServ-enabled network, FTP traffic is once again disadvantaged by a lower forwarding priority compared to the real-time applications. By employing traffic trunks and LSPs to route the real-time applications, MPLS is able to ensure that the available bandwidth is used fairly, which benefits FTP traffic both in terms of throughput and response delays.

4.3.7 Effect on the wireless access network

In this section we will investigate whether the occurrence of the link failure and the subsequent changes in the traffic flows had an effect on the performance of the wireless LAN access networks.

Figure 84 depicts the total throughput of all WLAN access networks for the three scenarios. As in the previous experiment, there is considerable variation present due to the constantly changing FTP traffic profile. Looking firstly at the baseline network, a clear decrease in throughput is visible from the 350 second mark onwards. In addition, the throughput becomes more stable.

Both changes are a result of the dramatic drop in the rate of FTP traffic observed in the previous section. A decrease in throughput variation is also seen in the DiffServ scenario, which likewise receives less FTP traffic than before. The sustained throughput of FTP traffic is evident in the data from the MPLS scenario.

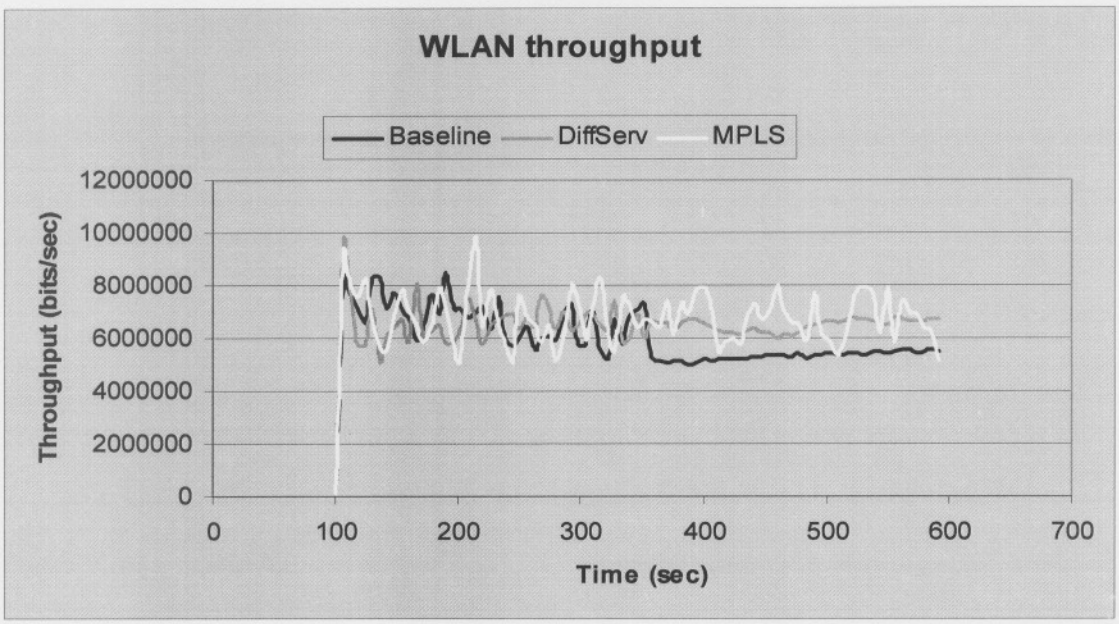


Figure 84 – Experiment 2: Total WLAN throughput

The above observations are confirmed by the sample means taken before and after 350 seconds (see Figure 85). Al three scenarios experience decreased throughput, but this is more pronounced in the baseline scenario, with a loss of approximately 20%. By adding QoS to the core network, this figure is reduced to around 2% for the other two scenarios. The MPLS scenario achieves the highest average throughput in all three instances shown in the figure.

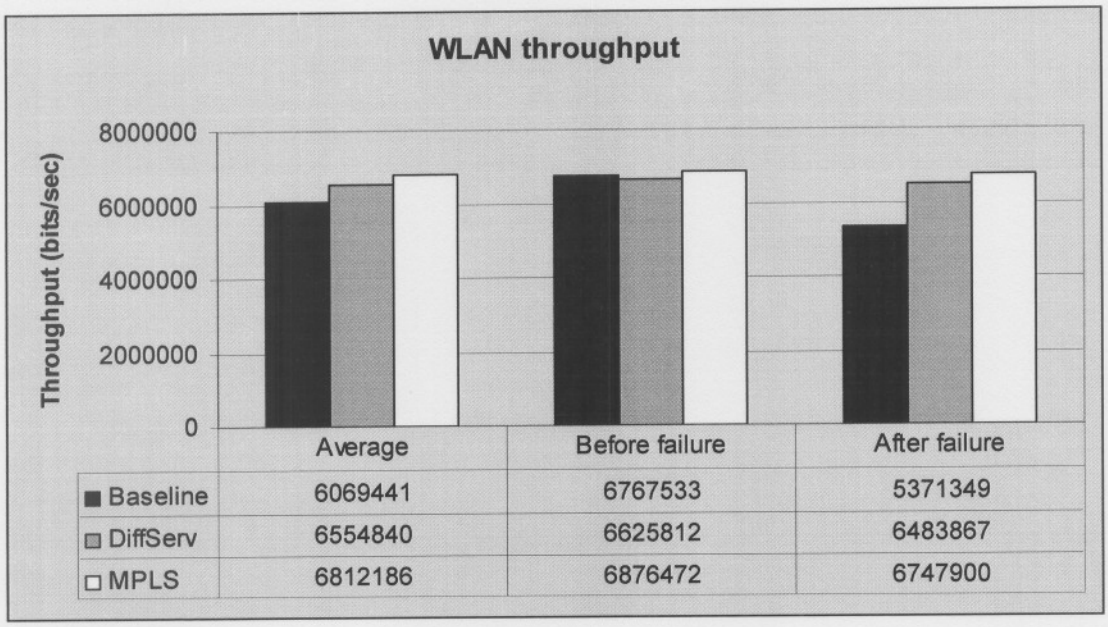


Figure 85 – Experiment 2: Effect of link failure on WLAN throughput

Looking at the media access delay in Figure 86, the MPLS scenario has higher average delay values than the other scenarios by some margin. The most likely cause is the higher WLAN throughput in comparison with the baseline and DiffServ networks, which increases the amount of packets in the buffer and thus the processing delays.

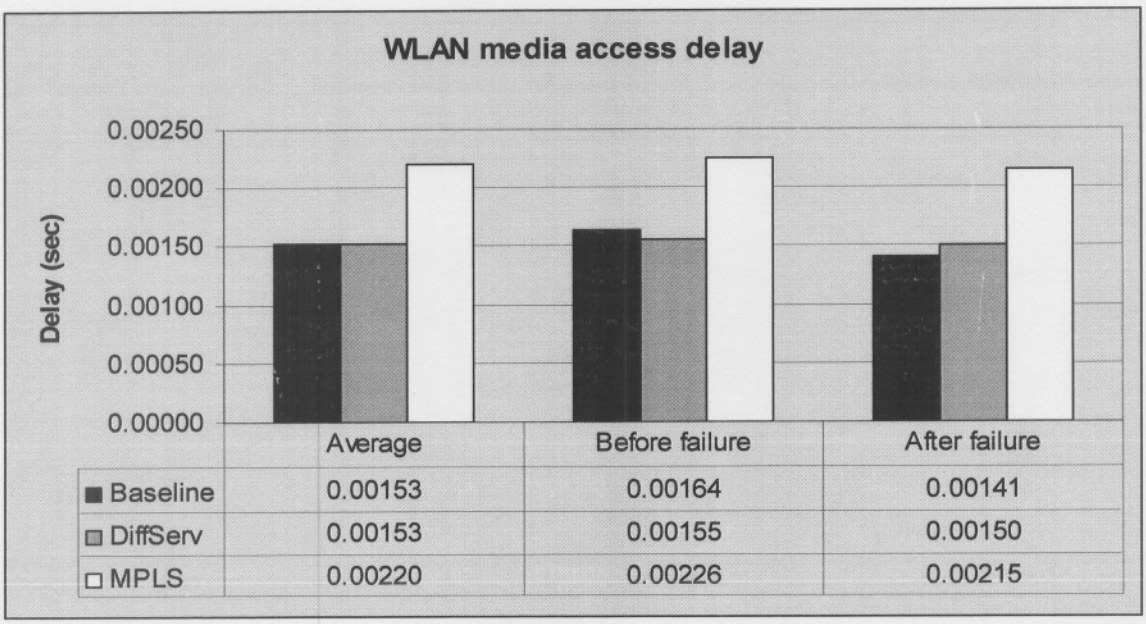


Figure 86 – Experiment 2: Effect of link failure on WLAN media access delay

We confirm the above assessment by comparing the average queue size of Access Point 0 for each scenario in Figure 87. The MPLS scenario consistently has the largest packet queue throughout the simulation. This is indicative of the greater volume of traffic processed by the AP in the MPLS scenario compared to the baseline and DiffServ scenarios.

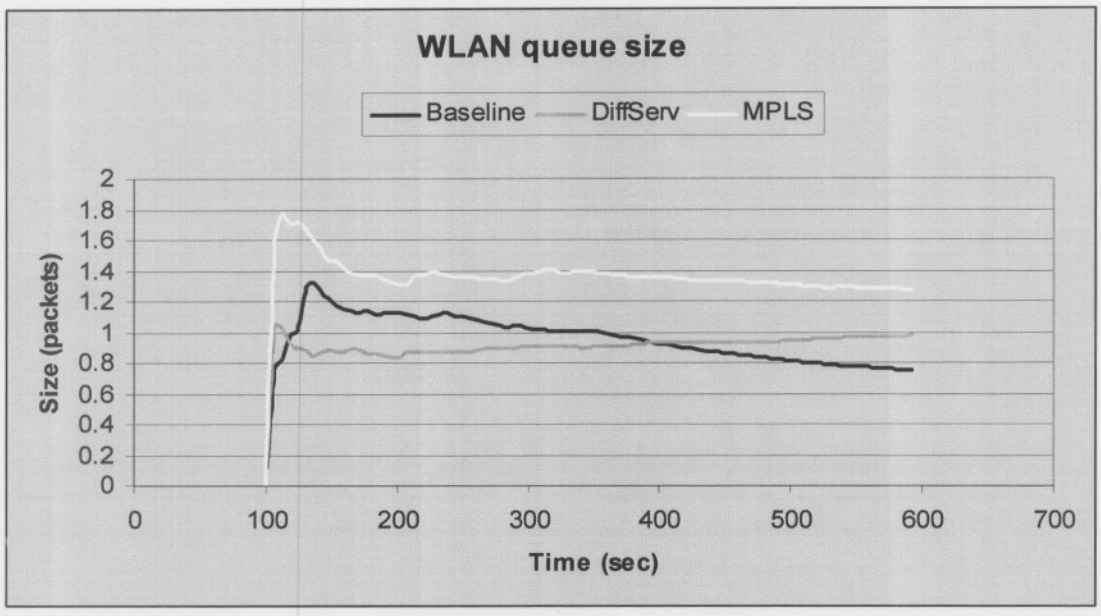


Figure 87 – Experiment 2: Average WLAN queue size of Access Point 0

4.3.8 Summary of Experiment 2

Using the ITU bounds as a guideline, we compare the performance of DiffServ and MPLS in terms of maintaining application QoS after the link failure (see Table 17).

Table 17 – Experiment 2: Changes in level of QoS due to link failure

Metric	DiffServ		MPLS		Degradation	
	Before	After	Before	After	DiffServ	MPLS
Voice delay (ms)	0.248 – Acceptable	0.679 - Unacceptable	0.185 – Acceptable	0.189 – Acceptable	Yes	No
Voice jitter (ms)	0.065 – Unacceptable	0.291 – Unacceptable	0.004 – High quality	0.001 – High quality	N/A	No
Video delay (ms)	0.075 – High quality	0.281 – Acceptable	0.038 – High quality	0.088 – High quality	Yes	No

For two out of three application metrics, the DiffServ scenario suffers degradation in QoS from a higher to a lower level. In the case of voice jitter, the QoS level was classified 'unacceptable' even before the link failure occurred. In the MPLS scenario the QoS level remains unaffected, even though the delay increases for both voice and video traffic.

Table 18 shows how the non-real-time FTP traffic was affected by the link failure in the DiffServ and MPLS scenarios. DiffServ records a 10% improvement in FTP traffic sent after the failure, but the rate of received traffic drops by more than 20%, while the response times show massive increases of more than 350%. MPLS is able to limit the degradation in FTP performance to no more than a 5-7% decrease in throughput and a 50% increase in response times. Although this 50% increase may appear large, the actual response times are still below 5 seconds on average.

Table 18 – Experiment 2: Effect of link failure on FTP performance

Metric	DiffServ		MPLS		Change (%)	
	Before	After	Before	After	DiffServ	MPLS
Traffic sent (bytes/s)	87874	96728	102617	95726	10.08	-6.71
Traffic received (bytes/s)	88529	69287	103384	98775	-21.74	-4.46
Upload response (sec)	7.373	38.981	2.963	4.495	428.69	51.67
Download response (sec)	9.937	45.464	3.291	4.976	357.52	51.18

The results from Experiment 2 demonstrate the effectiveness of MPLS traffic engineering in preventing loss of application QoS due to link failure in the core network. In addition, we see that the inability of DiffServ to consider available resources results in service degradation for both real-time and FTP applications.

4.4 Summary

From examining the results presented in this chapter, we may conclude that the experimental design as described in Chapter 3 was successful in providing the required data for our study. We now proceed to the final conclusions and recommendations that follow from the results.

5. Conclusions and recommendations

In this chapter, we conclude our research by measuring the experimental results against the stated outcomes and objectives of the study. We also make some recommendations concerning how real-world networks may benefit from this research. Finally, we highlight potential areas of future research that may improve or expand on this work.

5.1 Concluding remarks

The primary objectives of the study were to:

i. Deploy MPLS in the core of an infrastructure WLAN network in an attempt to provide QoS to different application types. This approach would emphasize effective management of available resources rather than the per-node scheduling method used by DiffServ.
ii. Determine the effect that deployment of MPLS in the backbone has on the performance of the wireless access networks.

5.1.1 Application QoS

Our design goals were to differentiate among applications with different QoS requirements, and then to control parameters such as delay, jitter and packet loss to ensure that these QoS requirements were met. We compared the MPLS results to those obtained from identical networks with a) no QoS deployed, and b) DiffServ queuing and scheduling in place. From an in-depth examination of the results, several conclusions may be drawn.

The first experiment investigated the ability of MPLS to add QoS to a network with limited core bandwidth. The baseline scenario is a clear example of a network that is unable to meet user requirements in terms of QoS. The standard IP routing used in this scenario fails to take available resources into account, and simply forwards traffic along the shortest path between source and destination. This results in some links being used very sparingly, while others constantly run at full capacity. These over-utilized links eventually cause congestion in the core routers, resulting in buffer overflows and packet loss.

Since each application receives the same treatment in terms of scheduling and forwarding, both multimedia and file transfer flows are equally affected by these conditions. The network is unable to meet the strict delay and jitter requirements of the voice and video applications, resulting in unacceptable levels of real-time performance as measured by the ITU standard. FTP traffic suffers due to TCP retransmissions caused by the traffic overload, which is reflected by the excessively long response times for file uploads and downloads.

Enabling DiffServ scheduling on the routers vastly improves the performance of real-time applications. The delay results for video conferencing conform to the ITU requirements for high-quality video transmission, while VoIP is of acceptable quality both in terms of delay and jitter. However, this approach has some distinct disadvantages. DiffServ is concerned with scheduling treatment and not with forwarding, and thus the DiffServ scenario displays similar behaviour to the baseline network in that some links are overused and other underused. Also, routing both real-time and FTP flows along the same path extracts a heavy toll on the file transfer traffic. In order to meet the delay and jitter requirements of voice and video, the lower priority FTP traffic is always serviced last at the routers. As in the baseline scenarios, this causes retransmissions, long response times, and reduced FTP throughput.

In terms of allocating resources fairly, MPLS performs the best of the tested scenarios. Routing traffic along explicit LSPs prevents any one link from being over-utilized, all but eliminating packet loss in the core network. Despite the absence of per-node QoS offered by DiffServ, the bandwidth allocation approach of MPLS is able to achieve similar real-time performance. Voice and video delay in the MPLS scenario are only fractionally higher than the respective DiffServ values, and are well within acceptable ITU bounds. The advantage of MPLS is that this can be achieved without starving the non-real-time traffic. Using dedicated paths for each application has the effect of separating non-real-time traffic from the multimedia application traffic. As a result, the MPLS scenario has by far the lowest and most consistent FTP response times, as well as the highest FTP throughput.

The goal of the second experiment was to investigate the impact of a link failure on the application QoS, using essentially the same simulation setup and QoS techniques as before. Once again, the lack of service differentiation or QoS guarantees in the baseline scenario result in unacceptable levels of delay and jitter for real-time traffic. After the link failure occurs, the situation becomes even worse as more traffic is forced onto the remaining links.

FTP throughput falls to a fraction of its former value, while packet loss in the core network greatly increases, along with the delay and jitter of the voice and video traffic.

The DiffServ scenario is able to provide acceptable levels of QoS to all applications before the link failure. However, as with the baseline scenario, DiffServ does not take available bandwidth into account when routing traffic. Consequently the link failure causes a notable loss in performance in the DiffServ scenario. Significant gains in the delay and jitter of the real-time applications mean that they no longer conform to the ITU bounds, while FTP also shows a decrease in throughput and longer response times than before the failure.

MPLS traffic engineering proves to be the best solution of the scenarios tested. By using backup LSPs to re-route traffic along previously unused links, MPLS is able to prevent any serious loss in the performance of the application traffic. The throughput of both FTP and real-time traffic remains consistent after the failure. Delay and jitter show marginal increases, but still remain within the stated QoS bounds. The MPLS scenario also shows no signs of increased packet loss after the link failure.

We can conclude from the experimental results that MPLS traffic engineering allows the effective management of the available network resources. Each application can be allocated sufficient resources to ensure acceptable QoS by ITU standards. MPLS is able to meet similar delay and jitter bounds than the high-priority DiffServ classes, while also providing much better throughput and response times for FTP traffic. By rerouting traffic from failed links, MPLS also makes the core network more robust against such failures.

It is worth noting that although we used a wired backbone in the experiment, the results will also be applicable to a wireless backbone. Since limited bandwidth and frequent link failures are typical of a wireless environment, adding these attributes to the core network also provides some insights as to how MPLS would perform in a wireless-only environment.

5.1.2 Concerning wireless access

Our second main objective was to investigate the interaction between the wireless access network and the wired backbone. In particular, we investigated the effect that adding QoS mechanisms to the core network would have on the access networks.

The results suggest that the throughput of the access network is the parameter most affected by the condition of the core network. In both experiments, the non-QoS baseline scenario had lower WLAN throughput than the DiffServ and MPLS scenarios. This is a reflection of the congestion and packet loss occurring in the core of the baseline network. Long delays and retransmission attempts limit the rate at which the STAs in the access networks are able to send traffic. This is most apparent in Experiment 2, where the baseline network shows a clear drop in WLAN throughput after the link failure. In both experiments, MPLS achieved the highest WLAN throughput, since the traffic engineering approach was able to keep delay and packet loss at a minimum.

Due to the topology of the wireless access networks, the WLAN delay values recorded by the simulation were negligible compared to the delays caused by the backbone component. The results show that the behaviour of the core network has little if any effect on the delay of the access network, except for the slight increase caused by DiffServ Priority Queuing.

However, it is worth noting at this point that the results would have been noticeably different for a less idealized wireless setup. Adding more wireless STAs to the access network would increase the per-STA media access delay. Making the STAs mobile instead of stationary would cause unpredictable changes in access delays, and could potentially introduce wireless effects such as the hidden terminal problem, path loss, or increased transmission errors.

All these factors would affect the QoS of the applications running on the wireless STAs. Since the legacy 802.11 protocol is still unable to offer service differentiation, adding QoS mechanisms to the core network alone would be less effective in a non-ideal access network. We can conclude from the results that in order to ensure acceptable QoS for all applications under a variety of conditions, there needs to be some form of QoS mechanism present in the access network as well.

5.2 Recommendations

5.2.1 Potential areas of application

The network topology we used in this study is similar in layout to existing ones used in campus or office networks. The main difference is that we have included wireless access networks, whereas currently the majority of workstations in the office or university are likely to use wires for connecting to the larger infrastructure. However, the results show that MPLS traffic engineering, though designed for large IP backbones, can be applied just as effectively to a relatively small wired backbone. MPLS offers many potential benefits including QoS for real-time applications, and increased robustness against node or link failures in the core network. This will aid in the successful deployment of applications such as video conferencing for meetings or distance learning, and the use of VoIP instead of current fixed-line telephone services.

5.3 Future work

5.3.1 Migrating to a wireless-only core network

A further extension to our current work would be to change from a wired to a wireless backbone by using wireless routers. MPLS has proven to be an effective QoS technique in a limited bandwidth environment, and also to guard against the consequences of a link failure. Limited bandwidth and unreliable links are both characteristics of wireless transmissions, and thus the use of MPLS in wireless backbones may prove advantageous for enabling QoS provision in this unpredictable environment.

5.3.2 DiffServ-aware MPLS traffic engineering

Referring back to Section 2.4, a possible improvement to our current approach would be to integrate the capabilities of DiffServ and MPLS in a combined QoS solution. This hybrid approach would employ DiffServ to provide queuing and scheduling priorities at every node, while MPLS would be used to ensure guaranteed bandwidth by routing traffic onto selected LSPs. This approach would satisfy both requirements of QoS provision mentioned in Chapter 2, namely guaranteed bandwidth and class-based treatment.

References

[1] Ni Q, et al, "A survey of QoS enhancements for IEEE 802.11 wireless LAN", *Wireless Communications and Mobile Computing*, vol. 4, no. 5, pp. 547–566, 2004

[2] Palmeri F, "An MPLS-based architecture for scalable QoS and traffic engineering in converged multiservice mobile IP networks", *Computer Networks*, vol. 47, no.2, pp. 257–269, 2005.

[3] Aad I, Castelluccia C, "Priorities in WLANs", *Computer Networks*, vol. 41, pp. 505–526, 2003.

[4] Grilo A, et al, "IP QoS support in IEEE 802.11b WLANs", *Computer Communications*, vol. 26, pp. 1918–1930, 2003.

[5] Baghaei N, Hunt R, "Review of quality of service performance in wireless LANs and 3G multimedia application services", *Computer Communications*, vol. 27, pp. 1684–1692, 2004.

[6] Vandermeulen F, et al, "A generic architecture for management and control of end-to-end quality of service over multiple domains", *Computer Communications*, vol. 25, pp. 149–168, 2002.

[7] Filsfils C, Evans J, "Engineering a multiservice IP backbone to support tight SLA's", *Computer Networks*, vol. 49, pp. 131–148, 2002.

[8] Gyires T, Wen H J, "Extension of Multiprotocol Label Switching for long-range dependent traffic: QoS routing and performance in IP networks", *Computer Standards & Interfaces*, vol. 27, pp. 117–132, 2005.

[9] Fineberg V, et al, "The MPLS UNI and end-to-end QoS", *Business Communications Review*, pp. 27- 32, Dec 2004.

[10] Awduche D O, Jabbari B, "Internet traffic engineering using multi-protocol label switching (MPLS)", *Computer Networks*, vol. 40, pp. 111–129, 2002.

[11] Kimura T, Kamei S, "QoS evaluation of DiffServ-aware constraint-based routing schemes for multi-protocol label switching networks", *Computer Communications*, vol. 27, pp. 147–152, 2004.

[12] Cui J, et al, "AQoSM: Scalable QoS multicast provisioning in Diff-Serv networks", *Computer Networks*, vol 50, pp. 80-105, 2006.

[13] Al-Karaki J N, Chang J M, "Quality of service support in IEEE 802.11 wireless ad-hoc networks", *Ad Hoc Networks*, vol. 2, pp. 265–281, 2004.

[14] Kimura T, Kamei S, "QoS evaluation of diffserv-aware constraint-based routing schemes for multi-protocol label switching networks", *Computer Communications*, vol. 27, pp. 147–152, 2004.

[15] Agarwal A, Wang K, "Supporting Quality of Service in IP multicast networks", *Computer Communications*, vol. 26, pp. 1533–1540, 2003.

[16] Chuah M, et al, "Quality of Service in Third-Generation IP-Based Radio Access Networks", *Bell Labs Technical Journal*, vol. 7, no. 2, pp. 67–89, 2002.

[17] Reddy T B, et al, "Quality of service provisioning in ad hoc wireless networks: A survey of issues and solutions", *Ad Hoc Networks*, vol. 4, pp. 83–124, 2006.

[18] Xu S, Saadawi T, "Revealing the problems with 802.11 medium access control protocol in multi-hop wireless ad hoc networks", *Computer Networks*, vol. 38, pp. 531–548, 2002.

[19] Trimintzios P, et al, "Quality of service provisioning through traffic engineering with applicability to IP-based production networks", *Computer Communications*, vol. 26, pp. 845–860, 2003.

[20] Sundaresan K, et al, "IEEE 802.11 over multi-hop wireless networks: Problems and new perspectives", *Ad Hoc Networks*, vol. 2, pp. 109–132, 2004.

[21] Zahariadis T, "Evolution of the Wireless PAN and LAN standards", *Computer Standards & Interfaces*, vol. 26, pp. 175–185, 2004.

[22] Zhang L, Zheng L, Ngee K S, "Effect of delay and delay jitter on voice/video over IP", *Computer Communications*, vol. 25, pp. 863–873, 2002.

[23] Remondo D, Niemegeers I G, "Ad hoc networking in future wireless communications", *Computer Communications*, vol. 26, pp. 36–40, 2003.

[24] Phanse K S, DaSilva L A, "Addressing the requirements of QoS management for wireless ad hoc networks", *Computer Communications,* vol. 26, pp. 1263–1273, 2003.

[25] Yin J, Wang X, Argawal D P, "Modeling and optimization of wireless local area network", *Computer Communications*, vol. 28, pp. 1204–1213, 2005.

[26] Ziouva E, Antonakopoulos T, "CSMA/CA performance under high traffic conditions: throughput and delay analysis", *Computer Communications*, vol. 25, pp. 313–321, 2002.

[27] Aad I, et al, "Enhancing IEEE 802.11 MAC in congested environments", *Computer Communications*, vol. 28, pp. 1605–1617, 2005.

[28] Kane J, Yen D C, "Breaking the barriers of connectivity: an analysis of the wireless LAN", *Computer Standards & Interfaces*, vol. 24, pp. 5–20, 2002.

[29] Farkas K, et al, "Real-time service provisioning for mobile and wireless networks", *Computer Communications*, vol. 29, pp. 540–550, 2006.

[30] Jamalipour A, Lorenz P, "End-to-end QoS support for IP and multimedia traffic in heterogeneous mobile networks", *Computer Communications*, vol.29, pp. 671–682, 2006.

[31] Akyildiz I, et al, "Wireless mesh networks: a survey", *Computer Networks*, vol. 47, pp. 445-487, 2005.

[32] Lorchat J, Noel T, "Overcoming the IEEE 802.11 paradox for realtime multimedia traffic", to appear in *Computer Communications*, 2006.

[33] Shi T J, Mohan G, "An efficient traffic engineering approach based on flow distribution and splitting in MPLS networks", *Computer Communications*, vol. 29, pp. 1284-1291, 2006.

[34] Kappes M, "An experimental performance analysis of MAC multicast in 802.11b networks for VoIP traffic", *Computer Communications*, vol. 29, pp. 938–948, 2006.

[35] Yin Z, Leung V C M, "Performance improvements of integrating ad hoc operations into infrastructure IEEE 802.11 wireless local area networks", *Computer Communications*, vol. 28, pp. 1123–1137, 2005.

[36] Kliazovich D, Granelli F, "Cross-layer congestion control in ad-hoc wireless networks", to appear in *Ad Hoc Networks*, 2005.

[37] Egea-Lopez E, et al, "Wireless communications deployment in industry", *Computers in industry*, vol. 56, pp. 29–53, 2005.

[38] Maniatis S, et al, "Dynamic resource management for QoS provisioning over next-generation IP-based wireless networks", *Computer Communications*, vol. 29, pp. 730–740, 2006.

[39] Kumar S, Raghavan V S, Deng J, "Medium Access Control Protocols for ad hoc wireless networks: A survey", Ad Hoc Networks, vol. 4, pp. 326–358, 2006.

[40] Mangold S, et al, "IEEE 802.11e Wireless LAN for Quality of Service", *Proceedings of European Wireless*, Florence, Italy, February 2002.

[41] Vijayaramgam S, Ganesan S, "QoS Implementation for MPLS based Wireless Networks", *Proceedings of the ASEE Conference*, Oakland University, Michigan, April 2002.

[42] Aad I, Casteluccia C, "Introducing service differentiation into IEEE 802.11", *Proceedings of the Fifth IEEE Symposium on Computers & Communications (ISCC '00)*, 2000.

[43] Fowler S, Zeadally S, "Adaptive Queue Management on Micro-MPLS-based Wireless Networks for Multimedia Traffic", *Proceedings of the 4th Annual Communications Networks and Services Research Conference (CNSR '06)*, 2006.

[44] Carvalho M M, Garcia-Luna-Aceves J J, "Delay Analysis of IEEE 802.11 in Single-Hop Networks", *Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP '03)*, 2003.

[45] Xiao Y, "An Analysis for Differentiated Services in IEEE 802.11 and IEEE 802.11e Wireless LANs", *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS '04)*, 2004.

[46] Hunt R, "IP Quality of Service Architectures", *Proceedings of the Ninth IEEE International Conference on Networks (ICON '01)*, 2001.

[47] Sun W, Bhaniramka P, Jain R, "Quality of Service using Traffic Engineering over MPLS: An Analysis", *Proceedings of the 25th Annual IEEE Conference on Local Computer Networks (LCN '00)*, 2000.

[48] Pelletta E, Velayos H, "Performance measurements of the saturation throughput in IEEE 802.11 access points", *Proceedings of the Third International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'05)*, 2005.

[49] Rouhana N, Horlait E, "Differentiated Services and Integrated Services use of MPLS", *Proceedings of the Fifth IEEE Symposium on Computers & Communications (ISCC '00)*, 2000.

[50] Girish M K, Zhou B, Hu J, "Formulation of the Traffic Engineering Problems in MPLS Based IP Networks", *Proceedings of the Fifth IEEE Symposium on Computers & Communications (ISCC '00), 2000.*

[51] Irvine J, Harle D, *Data Communications and Networks: An Engineering Approach*, John Wiley & Sons Ltd, West Sussex, England, 2002

[52] Young, P, *Electronic communication techniques*, 5th ed, Pearson Education Inc, Upper Saddle River, New Jersey, 2004

[53] Olexa R, *Implementing 802.11, 802.16, and 802.20 Wireless Networks: Planning, Troubleshooting and Operations*, Elsevier Inc, USA, 2005.

[54] Fineberg V, "QoS Support in MPLS Networks", MPLS/Frame Relay Alliance white paper, [http://www.mplsforum.org/tech/library.shtml#wp], May 2003, accessed January 2006.

[55] Cisco Systems, "MPLS FAQ for Beginners", [*http://www.cisco.com/warp/public/105/mpls_faq_4649.shtml#q16*], August 2005, accessed January 2006.

[56] Wikipedia online encyclopedia, "IEEE 802.11", [*http://en.wikipedia.org/wiki/802.11*], January 2006, accessed January 2006.

[57] Wikipedia online encyclopedia, "Quality of Service", [*http://en.wikipedia.org/wiki/Quality_of_service*], January 2006, accessed January 2006.

[58] Welcher P J, "Introduction to MPLS", [*http://www.netcraftsmen.net/welcher/papers/mplsintro.html*], August 2000, accessed January 2005.

[59] Kapp S, "802.11: Leaving the wire behind", IEEE Internet Computing Jan/Feb 2002 [*http://computer.org/internet*], accessed June 2006

[60] Hunt R, "Evolving technologies for new internet applications", IEEE Internet Computing Sep/Oct 1999 [*http://computer.org/internet*], accessed June 2006

[61] Intel Corporation, "Understanding Wi-Fi and WiMAX as Metro-Access Solutions", Wi-Fi and WiMAX Solutions White Paper, [*www.intel.com/netcomms/technologies/wimax/304471.pdf*], accessed June 2006.

[62] Zheng L, et al, "Fairness of IEEE 802.11 Distributed Coordination Function for Multimedia Applications", *OPNET contributed Papers library, [https://enterprise16.opnet.com/support/biblio.html],* accessed October 2006.

[63] Thanthry N, et al, "Issues in Deploying Wireless Networks", *OPNET contributed Papers library, [https://enterprise16.opnet.com/support/biblio.html],* accessed October 2006.

[64] Durbha P, Sherman M, "Quality of Service (QoS) in IEEE 802.11 Wireless Local Area Networks: Evaluation of Distributed Coordination Function (DCF) and Point Coordination Function (PCF)", *OPNET contributed Papers library, [https://enterprise16.opnet.com/support/biblio.html],* accessed October 2006.

[65] OPNET methodologies and case studies, "Representing Network Traffic", *[https://enterprise16.opnet.com/Temp_Priv/support/methodologies/115.html],* accessed October 2006.

[66] OPNET methodologies and case studies, "Adding QoS Services to an MPLS-Enabled Network", *[https://enterprise16.opnet.com/Temp_Priv/support/methodologies/115.html],* accessed October 2006.

[67] OPNET methodologies and case studies, "Strategies for Deploying MPLS", *[https://enterprise16.opnet.com/Temp_Priv/support/methodologies/115.html]*, accessed October 2006.

[68] OPNET methodologies and case studies, "Case Study: Simulation-Based Analysis of MPLS Traffic Engineering", *[https://enterprise16.opnet.com/Temp_Priv/support/methodologies/115.html]*, accessed October 2006.

[69] OPNET Modeler documentation, "MPLS Model User Guide", ver. 11.5.A, 30 August 2005.

[70] OPNET Modeler documentation, "IP QoS Model User Guide", ver. 11.5.A ,30 August 2005.

[71] OPNET Modeler documentation, "Wireless LAN Model User Guide", ver. 11.*5.A,* 30 August 2005.

[72] Wikipedia online encyclopedia, "Telecommunication", [*http://en.wikipedia.org/wiki/telecommunication*], accessed June 2006.

[73] Wikipedia online encyclopedia, "History of the Internet", [*http://en.wikipedia.org/wiki/History_of_the_Internet*], accessed June 2006.

[74] Wikipedia online encyclopedia, "History of mobile phones", [*http://en.wikipedia.org/wiki/History_of_mobile_phones*], accessed June 2006.

[75] OPNET Modeler documentation, "Modeling Overview", *ver. 11.5.A,* released 30 Aug 2005.

[76] OPNET Modeler documentation, "Modeling and Simulation Basics", *ver. 11.5.A,* released 30 Aug 2005.

[77] Boger Y, "Fine-tuning Voice over Packet services", VoIP white paper, [*http://www.protocols.com/papers/voip2.htm*], accessed November 2006.

[78] Global IP Sound product demonstration, "VoIP - Better than PSTN?", [*http://www.globalipsound.com/demo/tutorial.php*], accessed November 2006.

[79] Network tutorials session 1801, "Introduction to MPLS", *Proceedings of OPNETWORK 2003*, Washington DC, August 2003.

[80] OPNET Technologies Inc, "OPNET Modeler", [*http://www.opnet.com/products/modeler/home-2.html*], accessed November 2006.

[81] G. Flores-Lucio, et al, "OPNET-Modeler and NS-2: Comparing the Accuracy of Network Simulators for Packet-Level Analysis using a Network Testbed", *3rd WEAS International Conference on Simulation, Modelling and Optimization (ICOSMO 2003)*, vol. 2, pp. 700-707, Crete, 2003.

[82] Smit J J, "*A simulation study to evaluate the performance of schedulers in a Differentiated Services network*", Faculty of Engineering, University of Johannesburg, Johannesburg, South Africa, November 2004.

Appendices

A compact disc containing additional information is supplied with this report. The contents of the disc are as follows:

- An electronic copy of this document, labelled J_Schutte_M_Eng_Dissertation.doc.
- A folder, labelled OPNET Simulation, which contains a complete saved copy of the simulation as implemented in OPNET Modeler 11.5A. The simulations can be repeated in any working installation of OPNET Modeler version 10.5 and above.
- Two spreadsheet files, labelled Experiment_1_Results.xls and Experiment_2_ Results.xls. These documents contain the experimental data gathered from the OPNET simulation, as well as the figures and associated calculations used in the report.