SHEFFIELD HALLAM UNIVERSITY
SCHOOL OF COMPUTING AND MANAGEMENT SCIENCES

Project Title:	Investigation into the deployment of QoS over the JANET network and the costs and benefits of such an implementation.
Course:	BSc (Hons) Computing and Management Sciences (Final Year)
Name:	Benedict Horner
Supervisor:	Dr. Chuck Elliot

Acknowledgements

I would like to thank the following people who assisted in various ways throughout this project.

Dr. Chuck Elliot	For his support and advice throughout the project
Pat Myers and the rest of the Network Systems team at Manchester Computing: Darren Hughes, Malcolm Pitcher, Mike Robson and Anthony Ryan	For their support in making the implementation of this report possible by supplying the equipment needed and help where required
Jim Strom and Paul White at the e-learning centre (Manchester)	For the loan of the space and PC's for the implementation
Tom Kendrick	For his help on resources for IPv6.
Judith and David Horner	For their proof reading and support throughout the creation of this report

Abstract

This project's objective was to research both the JANET network and Quality of Service (QoS) mechanisms. These two pieces of research would then be brought together to find, through a practical implementation, how QoS might improve the JANET network's performance. Results from the implementation showed how a network on a small scale can have jitter on a UDP stream improved to less than 1% of its original level or worsened by 445% by QoS mechanisms. However it is generally the case that QoS mechanisms do create a more stable level of jitter across a network. This project's conclusions showed how QoS will be an important factor when improving the network of the future, but that a QoS rollout needs to be properly managed and tested, and all the costs and benefits fully understood before carrying out any QoS implementation.

Key Words:

JANET, QoS, Cisco, Networking, IP, Performance

Table of Contents

1. Introduction

1.0 Project Purpose

This project will analyse the feasibility of installing QoS (Quality of Service) on the JANET network and its subsidiary networks. It will describe the general topology of JANET and the details of the University of Manchester's network topology.

Once the topology analysis has been completed, there will follow a critical discussion of the varying QoS techniques available, thus to give an overview of how the technology works and how it would benefit a network such as JANET.

The report will then cover an implementation that will attempt to create a small network using Cisco QoS enabled hardware. A variety of different QoS strategies will then be implemented and tested on the network to identify and measure any improvements QoS may bring.

Through this topology and technology analysis, a time-line will be created in order to establish how QoS would be tested and finally implemented and if QoS would be cost effective.

After the implementation phases a proposal will be made in order to aid in the decision-making process of implementing QoS at Manchester and across JANET.

The deliverable for this project is an educative document for the JANET network managers advising them on the viability of the implementation of QoS, and a strategy outlining the potential benefits and costs involved in a network wide QoS policy.

1.2 JANET

During the late 1970s Universities and Research Councils within the United Kingdom operated their own computer networks. In 1984 these networks were consolidated into a single structure known as the Joint Academic Network or 'JANET'. The term 'superJANET' has now been introduced as a reference to the newer broadband components of the JANET network.

SuperJANET enables the academic community across the UK to do things that were once not possible with the early versions of the network. Broadband networking now allows users to transmit large amounts of audio and visual information. More importantly, this data can be transmitted across the network in real time enabling users and researchers to use sophisticated multimedia and real-time tools, for example videoconferencing, to develop applications such as group collaboration and distance learning.

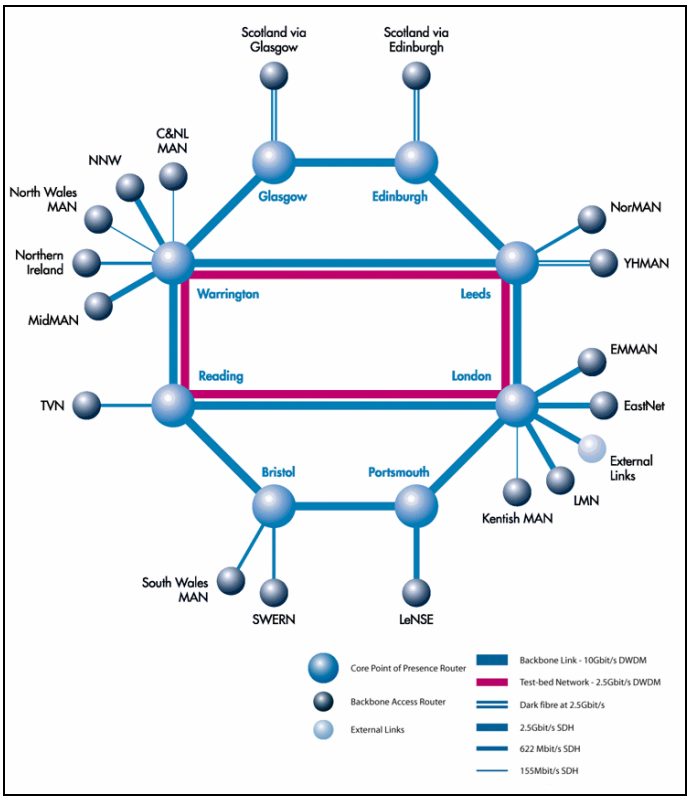


Figure 1.1 – Current superJANET underlying topology (UKERNA, 2003)

1. 3 QoS

QoS is a way of allocating resources in switches and routers so that data can get to its destination quickly, consistently and reliably. As applications are increasingly demanding high bandwidth and low delay, QoS is becoming a top criterion when network managers are looking at ways of improving their networks users' experience.

2. JANET

2.1 Architecture Overview

As previously mentioned, JANET is the wide-area network that was originally created in 1984 to serve the communication needs of the higher education and research sectors in the United Kingdom. In 2000, following the successful establishment of 'FEnet' in Wales (1996), JANET was extended to include the whole of the further education sector. As JANET grew, the JNT Association (known as UKERNA) was formed by the Higher Education Funding Councils for England, Scotland and Wales and the Office of Science and Technology in 1993 to run the JANET service. UKERNA is now responsible for overall management, operation and development of JANET and the provision of networking services to its customers. Since its conception JANET has grown from an X.25-based network in 1984 connecting only 50 sites, to an IP-based network currently supporting more than 700 direct connections. SuperJANET was the name given to the broadband capability of JANET, now the backbone component. The current SuperJANET4, came into operation at an initial 2.5Gbps in April 2001, superseding its 155Mbps predecessor, SuperJANET III.

The delivery of JANET services to customer sites (such as universities) is provided largely through a number of regional networks that have been developed since 1997. SuperJANET enables national and international connectivity through its high-speed links to the USA and mainland Europe, together with connections to commercial networks through the 'London InterNet eXchange' (LINX). The main operational centre for JANET is the Network Operations and Support Centre (NOSC), which operates under contract to UKERNA, and is situated at the University of London Computer Centre (ULCC). Detailed topology diagrams and explanations can be found in section 2.4 of this report.

2.2 JANET Uses and Technologies

Along with the usual network uses that can be found on most networks today such as web, FTP and e-mail, JANET is also used for more advanced purposes, for example videoconferencing and video-streaming; and is fundamental to e-science initiatives such as the UK GRID (http://www.escience-grid.org.uk). Many of the more advanced uses of JANET bring with them an increased demand for bandwidth; GRID computing, mentioned above, is used for bulk data transfer, dataset access, video and audio exchanges, medical imaging and interactive visualisation. All of these uses are very bandwidth intensive although some are more time critical (medical imaging), than others (bulk data transfer).

The amount of data expected to be produced and transferred through the GRID is extremely high. The particle physics section alone is expected to produce 1-7 Petabytes of data per year (equivalent to a continuous rate of 300Mbps – 2.1Gbps). Of course, along with particle physics, there are many other GRID uses and so it is easy to see how the levels of traffic could easily increase and overflow the links on superJANET.

Below is a table showing the varying uses of the trans-Atlantic traffic to and from JANET in the first quarter of 2000. It is obvious from the numbers that web traffic is the major contributor to use of the available bandwidth. As will be discussed later in section 3.2, web traffic is not a data flow that requires real-time service, and a delay of a few seconds would not be a great cause of concern for a user. With the current JANET network offering best-effort networking, real-time traffic such as videoconferencing is competing for bandwidth with web traffic.

	GigBytes	%misc TCP	%misc UDP	%FTP	%Mail	%DNS	%Web
From US	118,225.47	25.56	3.14	7.04	1.84	0.62	58.84
To US	47,577.01	44.70	2.24	17.79	2.38	0.92	29.06

Table 2.1 – Transatlantic JANET traffic

Another problem that faces the mangers of the JANET network is the network use habits of the students using the internet, via the academic institutions

connected to the JANET backbone. With the introduction of new uses of the internet such as peer-to-peer 'file sharing' that are known to be bandwidth 'hogs', network managers will need somehow to cap their use. Below is an example of a network (Arizona Western College) before and after peer-to-peer traffic was capped. It is obvious on the graph that the capping was started at 9am:

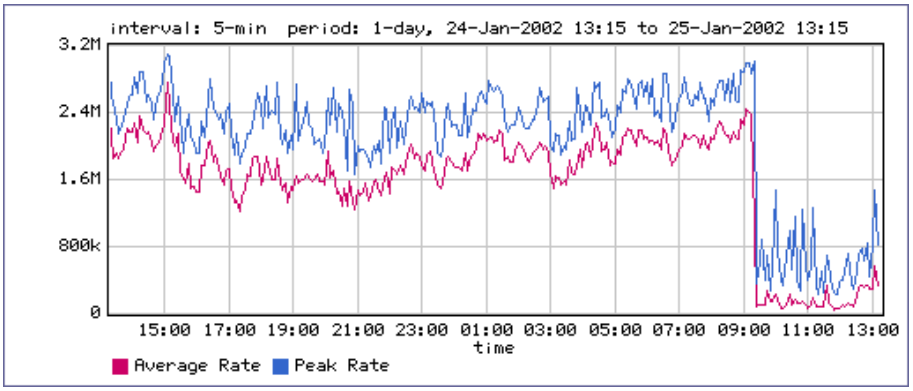


Figure 2.1 – Peer to Peer Traffic example (Arizona College, 2001)

With the huge variety of traffic that is sent across JANET everyday, it is clear that network managers will have to be very careful when deciding what traffic merits a better quality of service than another. These same managers will also have to constantly monitor the network, as new uses are appearing all the time.

2.3 In depth architecture

The implementation section within this report was carried out at the University of Manchester as facilities there were available for the testing of an implementation of QoS. The University was also able to give detailed topology diagrams of the University and its connections to the JANET backbone. For these reasons, this report will concentrate on the implementation of QoS on JANET and the MAN (Metropolitan Area Network) of the University of Manchester.

2.3.1 Manchester University Architecture

Below is a very simplified top-level topology diagram of the network used in Manchester. The colour of the different segments indicates which parts are managed by Manchester Computing (MC). Manchester Computing provides the computing facilities to the University of Manchester and to members of other UK academic institutions.

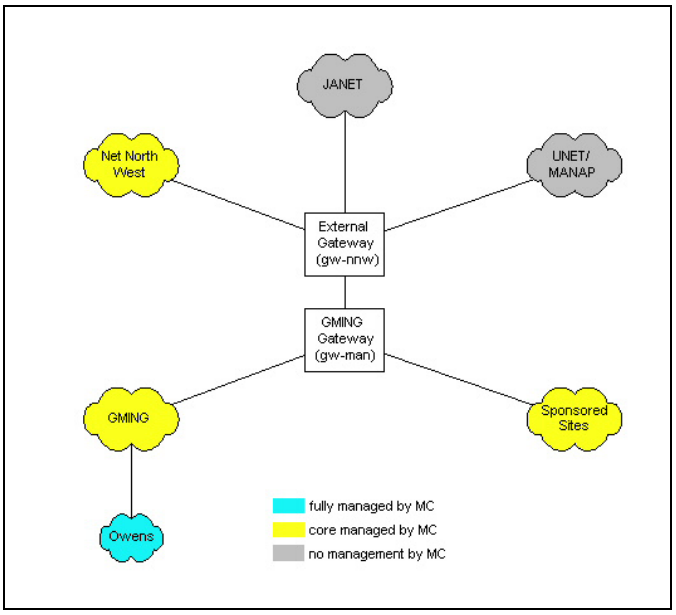


Figure 2.2 – Top Level Topology at Manchester University (Manchester Computing, 2003)

The next diagram focuses on the network that is fully managed by Manchester Computing at Owens Park. It shows the core routers that handle all the Owens Park traffic, the many networks that are supported by Owens Park and the types of links between them. By looking at this topology it becomes clear how complex the JANET network is. With 1000s of similar networks being connected to JANET, all with their own managers and network uses, the approach to implementing QoS across the whole of JANET will not be a 'quick-fix' that will suit all of JANET's users perfectly.

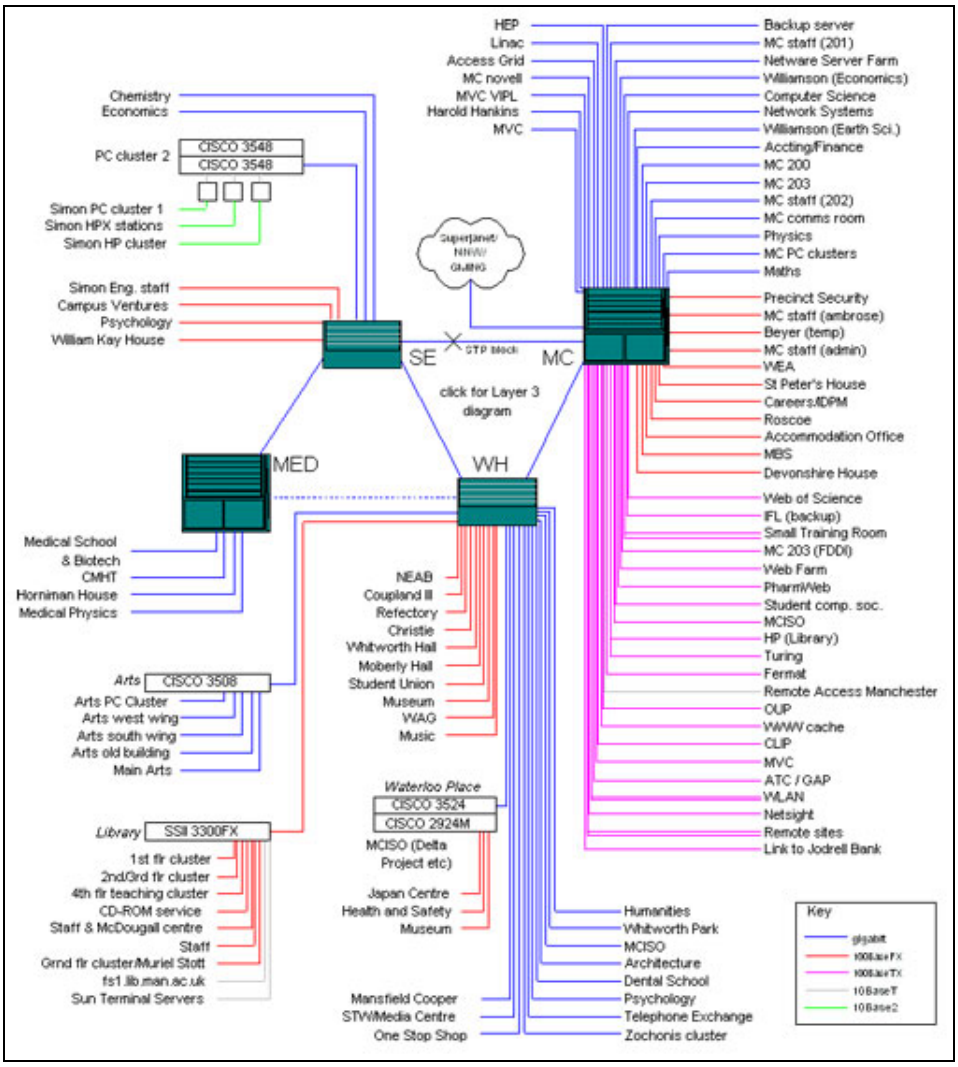


Figure 2.3 – Owens Park Topology (Manchester Computing, 2003)

2.3.2 Manchester University Network Uses

Manchester University is a very good example of a subsidiary network of JANET. The University supports the majority of JANET's uses such as videoconferencing, DNS, video-streaming, GRID computing and also a large amount of non-academic uses. Because of its wide variety of network uses, Manchester would be ideal to use as a test-bed network for the initial examination of the improvements that QoS might bring.

3. QoS

3.1 What is QoS?

Currently there are many definitions of what QoS is, partly due to both the words 'quality' and 'service' being ambiguous terms. Also QoS is a nascent technology with many important elements yet to be given final technical specifications from the Internet Engineering Task Force (IETF). Velte and Velte (2001) define QoS as, "a collection of run-time processes that actively manage bandwidth to provide committed levels of network service to applications and/or users. QoS implements a framework for service policy and action that extends end-to-end for serviced connections, even across autonomous systems." Put simply, QoS is a variety of mechanisms designed to favour some types of traffic over others.

3.2 Why QoS?

Until recently, increasing the bandwidth of a network was enough to solve most problems causing user dissatisfaction. When the network performance dropped below an acceptable level, the network manager would simply add another link to speed things up again.

The problem now for network managers is that the raw throughput of data is not the only requirement of today's users. Now the timing and availability of data is just as important to users as throughput. Many new networked applications such as video-streaming and 'Voice over IP' (VoIP) do not only use large amounts of bandwidth, but also add requirements such as timely content delivery. To be run successfully, VoIP needs a network with low 'jitter'. Jitter is caused when packets are delayed causing a signal to lose its timing references. Jitter can be seen by users as a conversation taking place out of sequence. It is caused by packets carrying the data across the network being delayed.

Below is a table showing the various QoS requirements for a number of network uses:

Application	Reliability	Delay	Jitter	Bandwidth
E-mail	High	Low	Low	Low
File Transfer	High	Low	Low	Medium
Web Access	High	Medium	Low	Medium
Remote Login	High	Medium	Medium	Low
Audio on Demand	Low	Low	High	Medium
Video on Demand	Low	Low	High	High
Telephony	Low	High	High	Low
Videoconferencing	Low	High	High	High

Table 3.1 – Stringency of various QoS requirements (Tanenbaum, 2003)

As shown in table 3.1 above, a TCP transfer such as an email can be run with a degree of jitter and delay on the network and not cause too much inconvenience to a user, but packet loss could make the email unreadable. On the other hand, a UDP transfer such as a VoIP call cannot handle much delay and jitter as the conversations fluidity would be lost, yet a small level of packet loss would not affect the call to any great degree.

3.3 Key Concepts

A degree of QoS has been used across the WAN links of JANET for many years now. JANET's WAN links are the biggest bottlenecks of the network. In an attempt to overcome this, WAN transport technologies such as frame-relay and ATM are used, which inherently offer a form of QoS.

Frame-relay uses the High-Level-Data-Link Control (HDLC) transport protocol. By using HDLC encapsulation, Frame Relay is able to create virtual circuits within a network, thus creating reliable connections that can be shared by many different applications. Frame Relay was designed to incorporate Explicit Congestion Notification (ECN). ECN is s flow control that consists of two bits that are within each frame header. These bits are used in the event of network congestion, by the network detecting it and then sending forward and backward ECNs (FECN and BECN) to the source and destination of the data. This aims to make the source and destination reduce their network utilisation and so reduce congestion.

ATM's use of QoS is more varied than that of Frame Relay. ATM automatically places data sent across the network into packets called cells. These cells are of a fixed length of 53 Bytes as shown below.

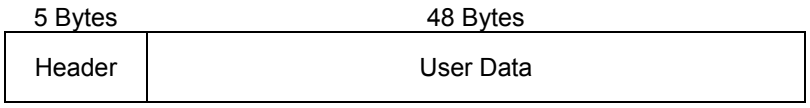


Figure 3.1 - An ATM Cell

Part of the header is a connection identifier, so that the sending and receiving hosts and all the intermediate routers can identify which cells belong to which connections. Due to the fixed size of the cells, routing can be carried out at a hardware level at great speed, thus making it much faster than IP which uses variable length packets that have to be routed by software. A result of ATM's fast routing method is that network congestion is automatically reduced along with reducing overheads on the routers' resources.

ATM also has a number of built-in QoS features which are, "negotiated prior to transmission, including the desired carrying capacity (in Mbps or cells/sec), the type of data in the payload (CBR or VBR) and the priority of the data (high or low)." (Corporate Information Systems, 1998). These QoS settings can be implemented on an ATM core to complement a surrounding IP based periphery using QoS.

However IP is the transport protocol used at the local levels of the connected LANs and MANs within JANET. All of the subsidiary networks of JANET at the academic facilities and beyond, utilise IP based architectures which use a wide variety of protocols and standards. This adds a problem with regard to QoS because IP is a 'best-effort' protocol, meaning that it is connectionless and once a packet is sent, it is left up to IP to decide what route that packet will take.

In addition to IP's best effort architecture, IP QoS is further complicated by:

- bandwidth being a limited resource, with every bps devoted to one application taking a bps from another
- a lack of set infrastructure supporting IP QoS. Unlike ATM and Frame Relay, IP has to handle multiple protocols that inevitably lead to an increase in complexity.

To deploy QoS across an autonomous network such as JANET requires a policy framework to sit above the IP networks. This will be in the form of an SLA that must be enforced throughout JANET to maintain a uniform policy across the topology. Although the subsidiary networks connected to JANET will probably have to design their own QoS policies, they must also be informed that those policies should be of a nature to complement the policy installed across the ATM core. This will enable a harmonious policy across the whole of JANET to the benefit of the users internal to the subsidiary networks and the managers of the superJANET core links.

3.4 QoS Types

There are four main areas:

- bandwidth provisioning
- bandwidth prioritisation
- congestion avoidance
- packet shaping

3.4.1 Bandwidth Provisioning

Provisioning of bandwidth is the clearing of a path through a network between nodes. Provisioning requires all nodes along a network flow's path to agree on a QoS procedure before any data transmission is made. A reservation may also need to make real-time adjustments to accommodate change to the flow's operating conditions, thus adding an extra level of complexity.

3.4.1.1 RSVP

Resource ReSerVation Protocol (RSVP) temporarily reserves an amount of bandwidth along a path to enable a minimum level of service for that flow. "RSVP does not transport application data but is rather an Internet control protocol" (RFC 2205, 1997). IP based applications can use RSVP to 'reserve' the amount of bandwidth they need in order to support the required level of QoS for their data stream. This is done through parameters such as minimum bandwidth, maximum delay jitter and maximum burst. RSVP is a complicated protocol. It works by defining a sender and a receiver host for each data flow. The sender node sends a 'PATH' message downstream collecting the device IDs that it passes through along the way to the receiver node. When the receiver gets the PATH message it sends back a 'RESV' message back along the same path to the sender node. The RESV message specifies all the QoS characteristics it requires and identifies all the nodes that it passes through 'sign-on' to the requested QoS. Once all the devices are signed-on, the data transfer may begin. Once the connection is finished, an explicit tear-down mechanism is used to ensure the reserved

resources are freed for the next user. If a reservation cannot be made, due to a possible lack of available bandwidth, the sending program has to decide whether to send the data using best effort, or to wait until a reservation can be made. During an RSVP enabled stream, refresh messages are sent through the route to ensure the resources are kept available. If refresh messages are not received, reservations will time-out and be dropped to release the bandwidth.

Unfortunately RSVP is not the best QoS mechanism for most networks. Among its shortcomings are:

- if just one router on the path is not RSVP-compatible, RSVP-based QoS will not work correctly
- because RSVP must configure each router along its path, there is a large overhead put onto the network to adopt the RSVP characteristics for each data stream
- if the reservation space is not fully used then this bandwidth is wasted, the un-used RSVP bandwidth is not re-distributed until the data stream has finished.

3.4.2 Bandwidth Prioritisation

A more reliable method of ensuring QoS is by using bandwidth prioritisation. Rather than RSVP's method of creating a hole through the network, bandwidth prioritisation takes the packets going through a network and sends on the most important ones first.

Simple prioritisation QoS is packet based. This means that the treatment a packet receives when being sent is determined by data within the packet itself. As shown in figure 3.2, in the header of an IP packet is an 8 bit ToS field. The IP ToS field contains 3 precedence bits that determine what level of priority the packet will receive when being sent on, so that more important packets can be placed in front of less important ones.

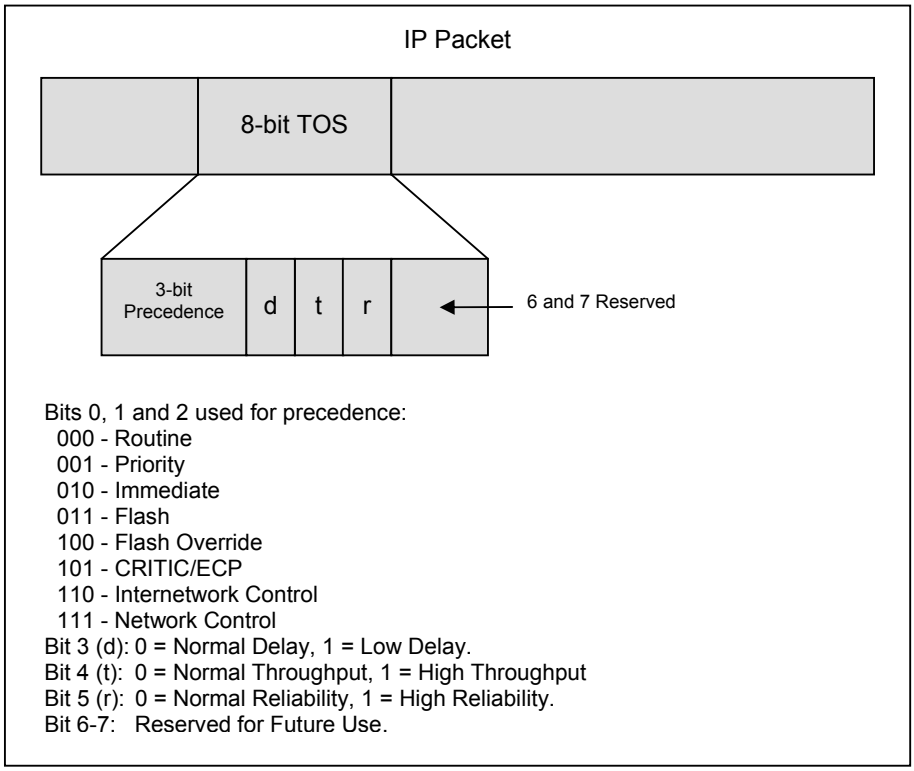


Figure 3.2 – IP Packet with ToS header field

This method of prioritisation means that each router a packet passes through will run its prioritisation independently from other routers on the network. This leads to a much lower level of overhead on the network in comparison to RSVP.

3.4.2.1 DiffServ

"Differentiated Services (DiffServ) is the most advanced method for managing traffic in terms of what is called Class of Service" (whatis.com, 2001). Class of Service (CoS) is slightly different to QoS in using 'best-effort' networking rather than defining levels of bandwidth and delivery times. CoS uses a lower level of granularity to QoS when controlling data flows by classifying different types of traffic into groups by use of the ToS bits in packet headers, so that they can be treated with their own levels of service priority.

DiffServ offers a different approach to managing packets than simple priority labelling. It uses an indication of how a given packet is to be forwarded and is known as the Per Hop Behaviour (PHB). The PHB

describes particular service levels in terms of bandwidth, queuing theory, and dropping (discarding the packet) decisions. These service levels are determined through complex policy or rule statements set on routers. A short policy example can be found in *Appendix A*. DiffServ redefines the IP ToS field bits to fit its own scheme by using 2 of the 8 bits for congestion notification and the remaining 6 bits for packet markings.

A router that has DiffServ enabled uses the ToS header field to determine how the implemented policy should treat the packet. An administrator can set up many different class definitions (much like first class and coach class on trains) that define how different types of traffic should be treated. An example would be VoIP traffic having a 'gold' class that does not drop packets and can use up to 80% of the available bandwidth, while the 'bronze' class for email traffic has a low packet drop threshold and is only allowed to use up to 15% of the available bandwidth. This means administrators can create very personalised policies which suit their network needs and can be adapted on a 'per-hop' basis.

3.4.3 Congestion Avoidance

There are two main mechanisms available to network managers when thinking about congestion avoidance. These are Random Early Detection (RED) and Weighted Random Early Detection (WRED). These two techniques are usually implemented on a network where there is no current congestion problem, but it is known that one may occur in the future. RED and WRED use queuing algorithms to sort traffic on a network and then decide which packets should be dropped to allow space for others.

3.4.3.1 Random Early Detection (RED)

RED is a proactive measure that can be implemented to relieve congestion in TCP flows. Instead of waiting until a congestion problem occurs, a router using RED drops packets as the congestion begins to emerge. As the name implies, RED will drop packets regardless of their type, class, or priority. As the congestion increases, so does the level of

packet dropping. RED's advantage over a policy such as DiffServ is that RED does not require any real overhead when deciding which packets to drop, as no queuing or packet re-ordering is needed. Of course, the QoS that RED produces is one very different to policies that do make use of queuing and packet shaping.

3.4.3.2 Weighted Random Early Detection (WRED)

As RED produces a QoS strategy that treats all packets equally, time-critical data flows have no priority over ones with no problem in being delayed. It could be said that RED is the fairest way to deliver QoS, but with JANET delivering time-critical data, a level of unfairness would probably be required. WRED offers an unfair method of dropping packets by assigning a weighting mechanism to data flows so that some data flows will have more packets dropped than others.

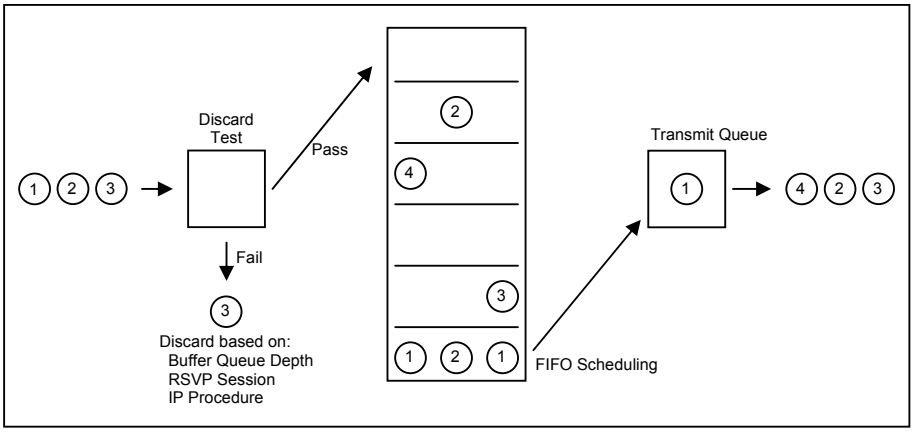


Figure 3.3 – Weighted Random Early Detection

WRED and RED differ from other congestion management techniques in the way that they work by anticipating congestion before it begins rather than controlling congestion once it has begun. As with RED, WRED tells the source of a data flow to reduce its transmission rate in order to try to reduce congestion.

WRED is primarily used in core routers on networks where network mangers expect bottlenecks to occur. Edge routers on the network will assign the IP precedence on each packet that enters the network and then

the core routers will use WRED to decide how each of the packets is treated.

3.4.4 Congestion Management

Congestion management is implemented by a network manager who has a bandwidth shortage. It offers managers a way of telling the network (via criteria set on the routers) which packets to send out first, which packets have to wait and which packets should be dropped. This is done through Weighted Fair Queuing and Class-Based Weighted Fair Queuing.

3.4.4.1 Weighted Fair Queuing (WFQ)

Weighted Fair Queuing (WFQ) is a strategy that allows throughput to a packet, based on the packet's precedence or weight. If the router detects congestion, the ToS bits are used to decide which packets are given priority and which are dropped. When the router becomes fully congested, and more packets are queuing up than can be physically handled, the router will use 'tail-drop'. WFQ ensures that more of the lower priority packets are dropped than the high priority ones.

WFQ uses an algorithm that gives each of the packet priorities a number from 0 to 7. This means, for example, that a packet with a priority of 4 is four times less likely to be dropped than a packet with a priority of 0.

Along with prioritising packets, WFQ also assigns a weight to each data flow. This determines the order in which the queued packets will be sent. WFQ is also RSVP aware, thus allowing it to allocate bandwidth for reserved flows.

WFQ ensures that flows do not run out of bandwidth to allow for predictable service. It does this by allowing low-volume traffic to receive preferential service so that it is not overwhelmed by a high volume transfer. High-volume traffic streams are therefore forced to share the remaining available bandwidth proportionally between themselves.

With WFQ being biased towards more 'traditional' data streams on networks, it may not be the most suitable on the JANET network, as many high-volume traffic streams on JANET can be time-critical. It may be that WFQ could be implemented alongside another QoS strategy that would better suit the needs of the JANET network.

3.4.4.2 Class-Based Weighted Fair Queuing (CBWFQ)

Class-Based Weighted Fair Queuing (CBWFQ) is a more flexible version of WFQ. Network managers are offered a finer level of granularity when controlling classes or priorities of data flows.

As an example CBWFQ would, instead of allowing only weights of 0 to 7, allow custom weightings such as: 1, 2, 4, 8, 16, 32, 64, 128. In this example the packets with priority 6 are now 32 times more likely to be dropped during a period of congestion.

Classes can be based on such criteria as protocols, access control lists and input interfaces. A queue is then created for each class and packets are routed to a queue according to their class. For each class the following criteria must be allocated:

- available bandwidth
- weight
- maximum queue limit before tail-drop

3.4.5 Packet Shaping

Data flows across networks are rarely sent in a uniform fashion. Packets can be sent in a 'bursty' manner making managing that traffic very difficult. To regulate the traffic into a more uniform flow, packet shaping can be used. This report will look at two methods of packet shaping: Leaky Bucket and Token Bucket. These methods are usually implemented on edge routers as a way of regulating the flows of data into a network.

3.4.5.1 Leaky Bucket

This method was first proposed by Turner (1986) and works very much as its name suggests. If water is poured into a leaky bucket, it begins to

fill but water drips out through the leak at a regular rate. If enough water goes into the bucket to fill it, water will also spill over the edge and be lost. Applying this to a network, the water is the data and when the water goes over the edge of the bucket, this is equivalent to packets being dropped when the queue is full. This is demonstrated in figure 3.4 below:

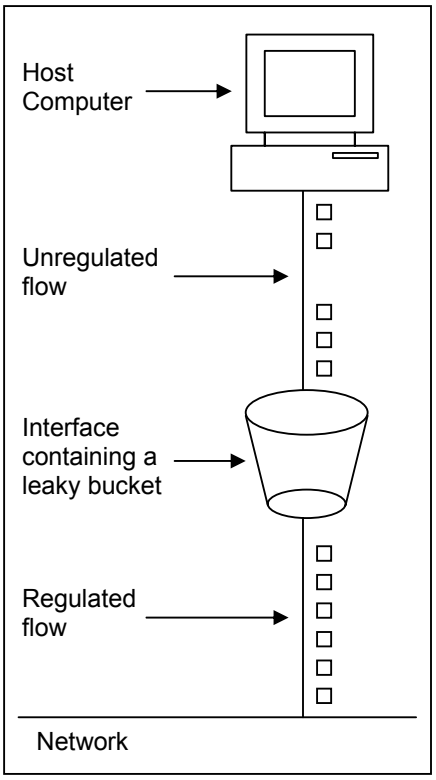


Figure 3.4 – Leaky Bucket Algorithm

Figure 3.4 shows how a computer sends irregular data flows to the network. However the router, using the leaky bucket algorithm then lets the data be forwarded per clock tick onto the network in a uniform rate, thus increasing the QoS of data onto the network by reducing the chance of congestion. When the packets being received are the same size (e.g. ATM) this algorithm can be used as described. If the packets being received are of variable sizes it is often advisable to allow a fixed number of bytes per clock tick, rather than just one packet per tick. If, for example, the tick limit is 1024 bytes per tick, a single 1024 byte will

be sent on its own, two 512 byte packets will be sent together, four 256 byte packets will be sent together and so on.

3.4.5.2 Token Bucket

The Token Bucket mechanism is quite different from the Leaky Bucket. A token bucket controls the transmission rate by the presence of 'tokens' in the bucket. A token is an abstract currency measured in bytes that must be available for the next FIFO packet to exit the network interface. There must be at least as many token bytes available as the number of bytes in the packet to be transmitted. The token bucket algorithm allows for some levels of 'burstiness' in the output stream, thus giving faster response to sudden bursts of input.

Packet shapers have limited use and application. They are normally implemented on networks that have a small number of WAN links and have the disadvantage of being inflexible once implemented, with changes to the policy being laborious task.

From all the QoS mechanisms mentioned in section 3.4 it was decided to concentrate on the ones below due to time/resource constraints:

- DiffServ
- Fair-Queuing
- Random Detection using a variety of policies
- A combination of DiffServ and RED
- Windows XP embedded QoS

These mechanisms are implemented in section 4.1 of this report.

3.5 Cisco QoS

The implementation section of this report (section 4) used a Cisco router. The majority of the networking hardware used at Manchester Computing is also Cisco based. Therefore this section will look at Cisco's approach to QoS and their suggestions when implementing a new QoS strategy.

Cisco have tried to make implementing QoS across an entire network as pain-free as possible. To this end they have created two pieces of software that can configure a variety of Cisco devices automatically through a user GUI rather than configuring each router and switch 'by hand' via the Cisco IOS (Internetwork Operating System). These programs are QoS Policy Manager (QPM) and QoS Device Manager (QDM) and they are discussed in the next section. The router used in section 4 of this report was programmed using the IOS interface as neither of the above pieces of software were available at the time of implementation.

3.5.1 QoS Policy Manager (QPM)

QPM offers a network manager an easy way to configure many Cisco routers and switches on a large network. It allows a manger to group routers together so that one policy can be deployed to many routers at the same time. For example, all edge routers might have the same CBWFQ policy and class definitions and at the same time all the core routers could have WRED, all with the same weighting policy.

A screen shot from QPM is shown below. It shows how a router can be assigned a QoS property such as CBWFQ through a simple drop-down menu rather than the user having to tackle setting up CBWFQ and its settings through the command line based Cisco IOS.

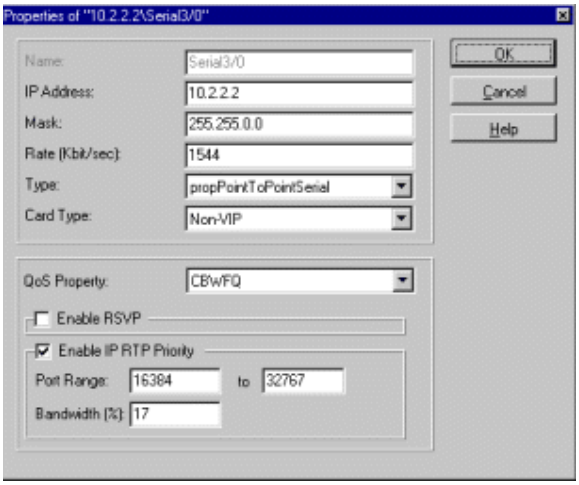


Figure 3.5 – Screen-shot of Cisco QPM

The main downfall of Cisco QPM is its price tag of £8,000; although this price would be easily justified by the number of man-hours it would cut from a manual QoS deployment.

3.5.2 QoS Device Manager (QDM)

Cisco QoS Device Manager is a small, free piece of software available from Cisco that helps network administrators configure and monitor QoS traffic classes and traffic policies on Cisco routers. QDM provides support for the following QoS mechanisms:

- minimum guaranteed bandwidth using Class-Based Weighted Fair Queuing (CBWFQ)
- low latency queuing
- rate limiting using traffic policing
- congestion control using traffic shaping
- congestion avoidance using Weighted Random Early Detection (WRED)
- packet marking/colouring
- fair bandwidth allocation using Flow-Based Weighted Fair Queuing (FBWFQ)

Cisco's website gives a list of the features and benefits of QDM, these are shown in table 3.2.

Feature	Comment/Description	Benefit
Wizards for quality of service (QoS) Configuration	Assists network administrators in the configuration of QoS functions on the device	• Makes it much easier to deploy QoS in the network • Helps teach users how to use and deploy QoS
Web and Java-Based Tool	QDM is stored on the router's Flash memory and downloaded to a Web browser	• Does not require a separate management station • Initial configuration of QDM is very easy
QoS Monitoring	Provides a number of graphs to display real-time QoS information	• Assists troubleshooting by allowing the network manager to easily see and visualize network status
QoS Baselining	Using NBAR, provides a view of protocols and applications currently in use on the network	• Baselining is required before deploying QoS features on the network; enabling one to easily baseline the network so that the proper QoS policy may be deployed

Table 3.2 – QDM Feature Overview
(Cisco, 2003)

As can be seen, this tool would be very useful to network managers wishing to undertake new QoS deployment, especially as the software provides a baselining tool that will allow for the analysis of any current QoS that is running on the network.

3.6 3rd Party QoS Delivery

Third party software vendors have now also realised the potential in QoS networking and so have developed software that is not platform specific, and in many cases includes additional functionality. One such piece of software is VizIq from Envoda (www.envoda.com). VizIq is a policy-based network management system that helps network managers automate device QoS configuration and prioritise applications. Traffic is managed by setting up classes that prioritise network traffic. These policies can then be automatically sent to any router or switch on a network, which gives a great deal of flexibility when deploying a new QoS strategy. VizIq also allows for time-based QoS, meaning automated changes in the network's QoS can be set to different times of the day or week. For example this would allow GRID

computing to have priority between 10pm and 7am, and then the network to resume normal operations from 7am to 10pm. VizIq can also work dynamically by checking network statistics and then change the QoS policy accordingly. This is demonstrated below:

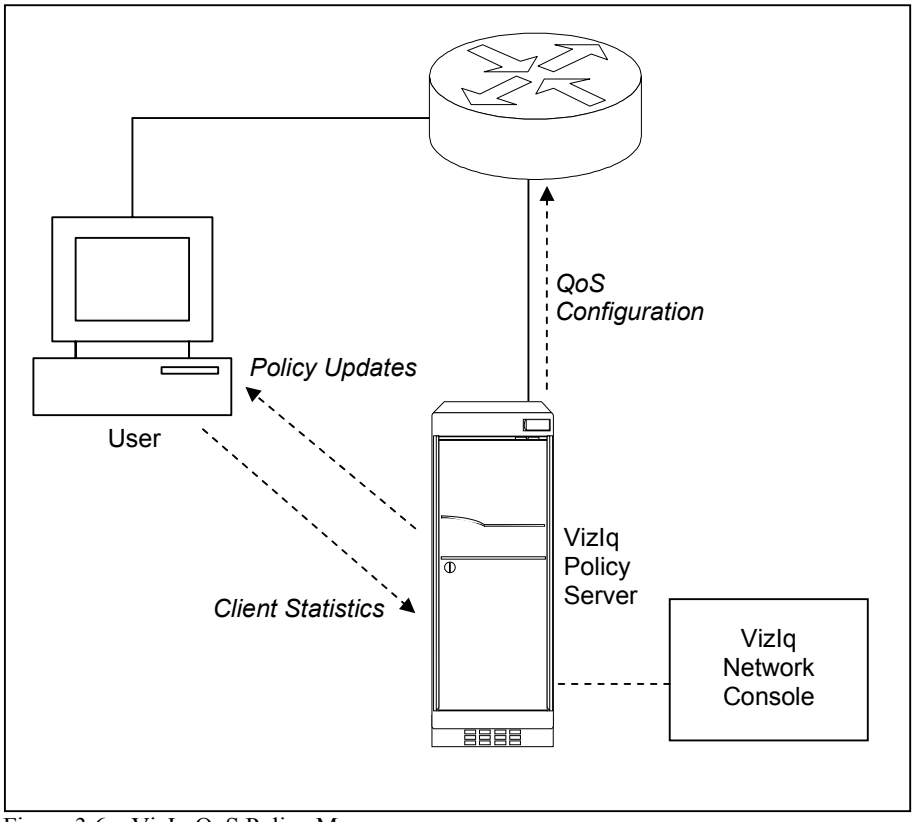


Figure 3.6 – VizIq QoS Policy Manager

With VizIq constantly monitoring the network for changes in bandwidth use it could become a valuable tool for network managers in assessing where the majority of the bandwidth is being used, and how best to manage availability.

4. Implementation

4.1 Manchester University

The purpose of this small scale implementation was to create a 'proof-of-concept' network that ran a number of basic QoS configurations. These configurations were tested against a baseline of the network running with no QoS strategy.

4.1.1 Testing Plan

The equipment used in the implementation was as follows:

- 2 Dell 700Mhz PC's (Windows XP)
- 1 Cisco 7200 Router
  - with 2 Gigabit Ethernet Interfaces (GigEth 0/1 and GigEth 0/2)
- 1 Toshiba Laptop (Router admin)
- 2 Crossed-over CAT5 cables
- 1 Serial cable

The test network was setup as show below:

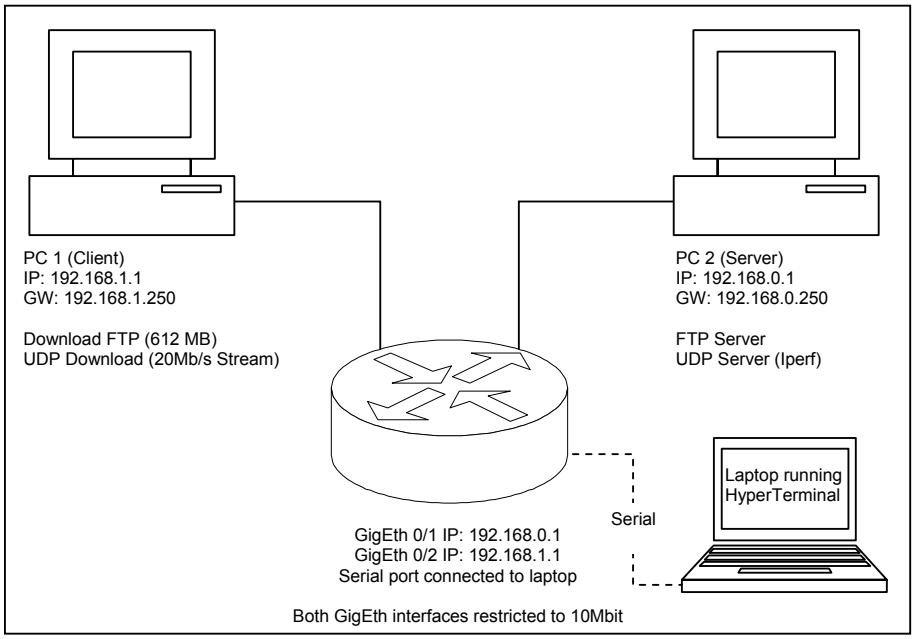


Figure 4.1 – Test Network

The Cisco 7200 Router comes with 2 interfaces that can handle 1000/100/10/Auto speeds. Due to the limited number of PCs used in the test, it was decided that all the interfaces would be limited to 10Mbit so that the interface could be easily saturated with data, thus forcing the router to use the QoS strategies implemented. More information regarding the Cisco 7200 Router used in the implementation can be found at: http://www.cisco.com/univercd/cc/td/doc/product/core/7200vx/72vxicg/over6vx.pdf

PC 1 was connected to the router via the GigEth 0/1 interface and placed on a class C network of 192.168.1.0. This PC would act as the client. PC 2 was connected to the router via the GigEth 0/2 interface and was placed on a class C network of 192.168.0.0. This PC would act as the server.

The QoS tests to be run were as follow:

QoS Strategy	Implemented on just Interface GigEth 0/1	Implemented on both interfaces	Test ID
No QoS (Baseline)	-	-	1
DiffServ	✓		2 – 1
DiffServ		✓	2 – 2
Fair-Queue	✓		3 – 1
Fair-Queue		✓	3 – 2
Random-detect (DSCP)	✓		4 – 1
Random-detect (DSCP)		✓	4 – 2
Random-detect (DSCP/Flow)	✓		5 – 1
Random-detect (DSCP/Flow)		✓	5 – 2
Random-detect (Prec)	✓		6 – 1
Random-detect (Prec)		✓	6 – 2
Random Detect (Prec + DiffServ)	✓		7 – 1
Random Detect (Prec + DiffServ)		✓	7 – 2
XP QoS enabled (No router QoS)	-	-	8

Table 4.1 – Test Plan

The QoS tests outlined in table 4.1 were chosen as these were the QoS mechanisms available on the Cisco 7200 Router that was being used. These mechanisms were covered earlier in section 3.4. For an example of the routers running-configuration see *Appendix B*.

For fairness during the tests, the two PCs were put on their most basic default settings. The inbuilt QoS of the Windows XP operating system was disabled on all the tests other than test 8. All virus scanning programs, firewalls and other non-essential applications and services were disabled for the duration of the tests.

To generate traffic across the network that could be tested for QoS, TCP traffic in the form of a large FTP upload was used together with a 1Mbit UDP stream (representing a video stream) created using Iperf. By using these two transfers it was possible to see how the UDP stream would be handled along-side a bandwidth-heavy TCP based transfer. During the UDP transfer, jitter levels were monitored and recorded for later analysis as jitter levels would be a good indicator of QoS improving the data transfer quality.

Below is a photograph of the test network put together at the University of Manchester:

Figure 4.2 – Final test network

4.1.1.1 Iperf

Iperf is a tool developed by 'The Distributed Applications Support Team' (DAST - http://dast.nlanr.net) at the National Laboratory for Applied Network Research (NLANR) to measure the maximum bandwidth of a network, allowing the tuning of various parameters and UDP characteristics. Iperf reports bandwidth, delay jitter and datagram loss. Iperf thus lends itself very well to QoS testing as it shows all the statistics that are needed to test for QoS improvements, especially with regard to jitter.

Iperf is used via a command line interface between 2 PCs. One PC acts as a server and one as a client. Below are the client and server configurations and explanations of the various switches used:

Client:

```
iperf -c 192.168.0.1 -u -t 300
```

`iperf`	- Calls Iperf executable
`-c 192.168.0.1`	- Act as a client and connect to a server at 192.168.0.1
`-u`	- UDP protocol to be used
`-t 300`	- Run test for 300 seconds

Server:

```
iperf -s -u -i 10
```

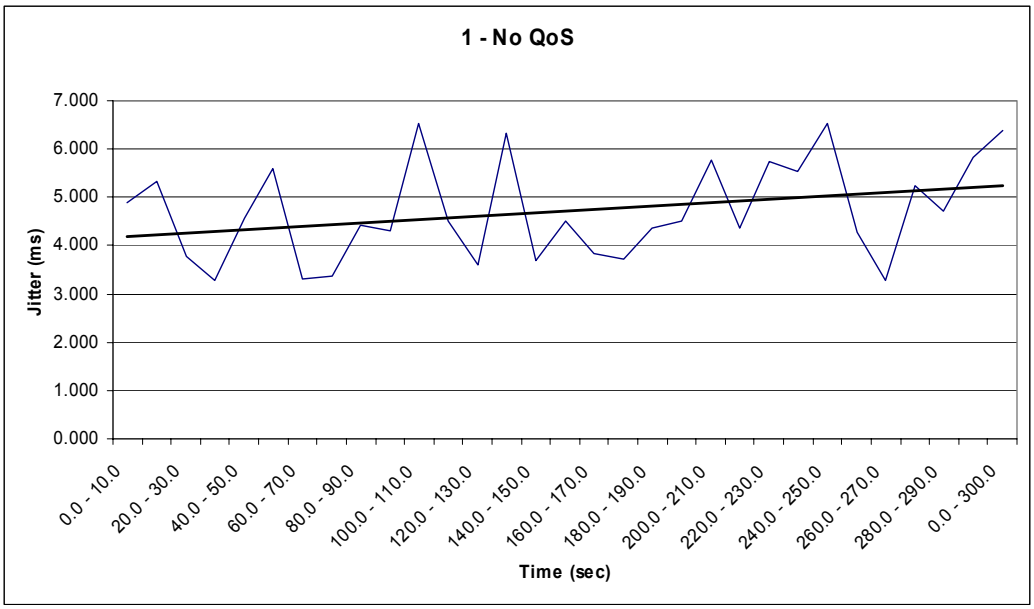
`iperf`	- Calls Iperf executable
`-s`	- Act as a server and listen on the default port (5001) for client requests
`-u`	- UDP protocol to be used
`-i 10`	- Make log entries every 10 seconds
`-o iperflog.txt`	- Output the log to a text file called 'iperflog.txt'

For each QoS test the FTP file transfer was started and run for the full duration of the test. Once the FTP transfer had been initiated, the Iperf test was run for 300 seconds. This imitated how, for example, a video stream would be handled when a bandwidth-heavy TCP-based transfer was already operating across a network. Once finished, the results were collected and stored, the router re-configured for the next QoS strategy and the test run again. An appropriate scale was used to graph the jitter levels for easy comparison between tests.

4.1.3 Results

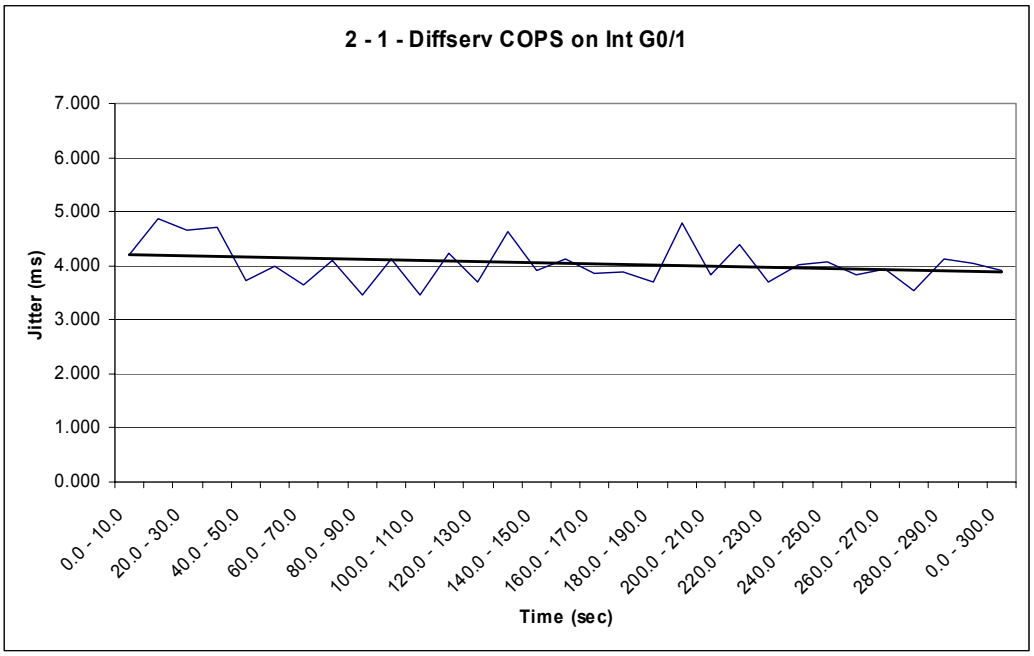
The results produced by Iperf were outputted to text files after each test. These results were then imported into Excel for analysis. An example of the raw Iperf output can be found in Appendix C.

4.1.3.1 Test 1 - No QoS (Baseline)



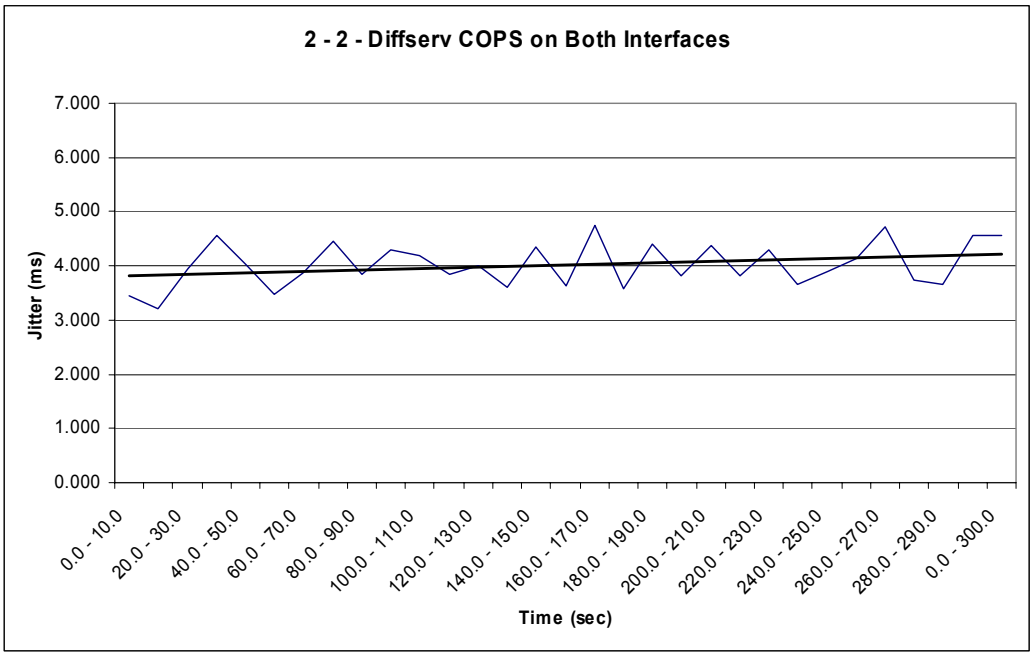
The baseline produced results against which all the other tests that use QoS were compared. The baseline shows an upward trend for jitter during the test with an uneven level of jitter. This clearly shows that with no QoS installed, a busy network running a variety of services such as TCP and UDP, will inevitably produce jitter and thus interfere with any real-time applications such as VoIP. It should also be noted that a 2% packet loss was recorded during the 270 – 280 second time frame. On all other tests carried out a 0% packet loss was recorded apart from Fair Queuing where a 65% packet loss was present.

4.1.3.2 Test 2 - 1 - DiffServ on Int G0/1



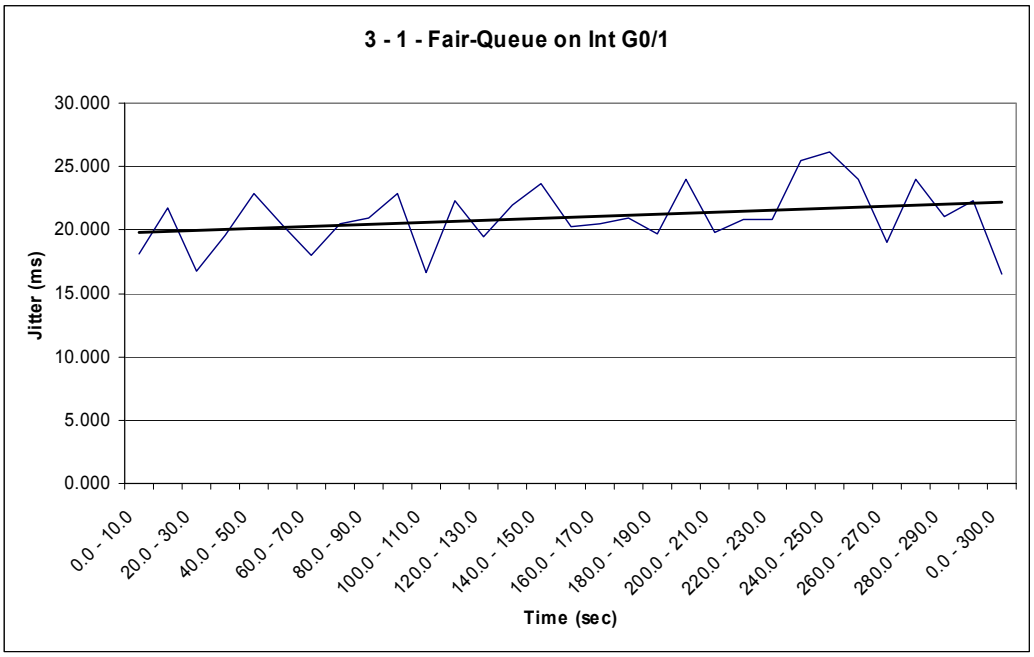
The second test was the implementation of DiffServ on the GigEth 0/1 interface. Here the level of jitter is slightly lower than that of the baseline test and it should also be noted that the levels of jitter remain very consistent across the entire test. In a larger network with higher levels of traffic, it would be as important to keep a level amount of jitter as it would be to keep a low level of jitter.

4.1.3.3 Test 2 - 2 - DiffServ on Both Interfaces



Implementing DiffServ on both interfaces made little difference to the results of the test that ran on only one of the interfaces. Again a steady level of jitter was present throughout.

4.1.3.4 Test 3 - 1 - Fair-Queue on Int G0/1



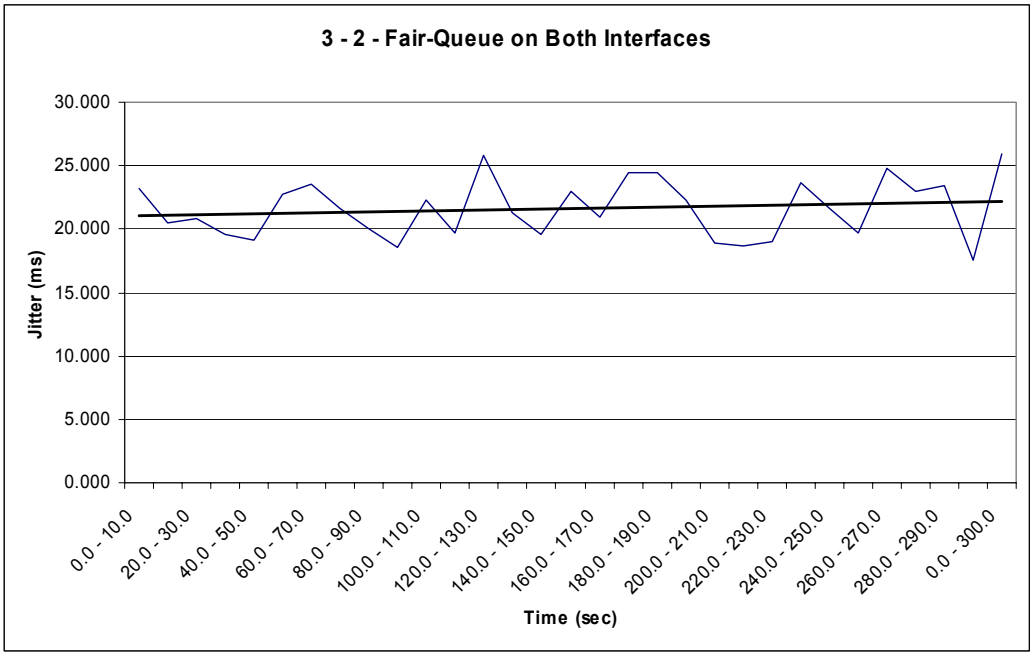
The graph above and the graph in section 4.1.3.5 have had their y axis adjusted to accommodate the higher levels of jitter recorded in the tests.

The fair-queuing tests produced very surprising results that required double checking. The levels of delay in both fair-queue tests were notably different to any of the other tests. The average jitter level for the first test was 20.996 ms as compared to the average jitter of the baseline at 4.715 ms.

After research into why the levels of jitter would be so high for fair-queuing it was found that fair-queuing works best alongside a network running RSVP. As the test network was not running RSVP it was obvious that the queuing mechanism had to work without it. Fair-queuing is also best used in an environment where the traffic going through the network is known, and so policies can be set up to handle the various types of traffic encountered. With the test network, the default fair-queue settings which were used obviously do not suit the network's uses. It would be beneficial in further tests to develop a full policy map that took into account all the traffic types and then gave them a more suitable class-based weighting so that network jitter could be lowered

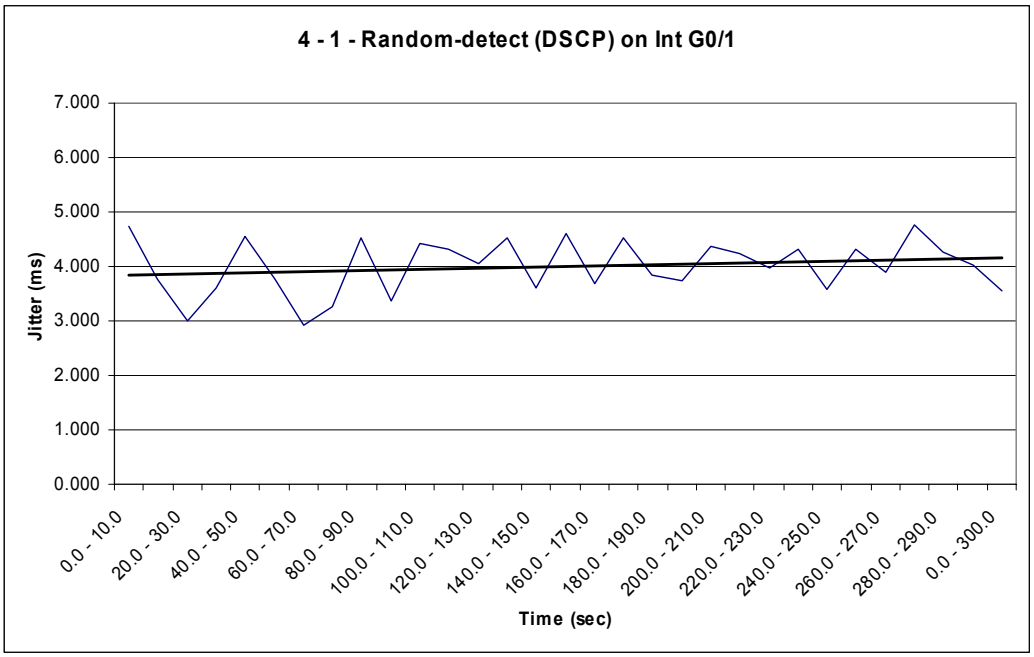
to a more acceptable level. Both fair-queue tests also produced an average throughput of 365Kbit/s instead of 1.45Mbit/s and they had an average UDP packet drop rate of 65%.

4.1.3.5 Test 3 - 2 - Fair-Queue on Both Interfaces



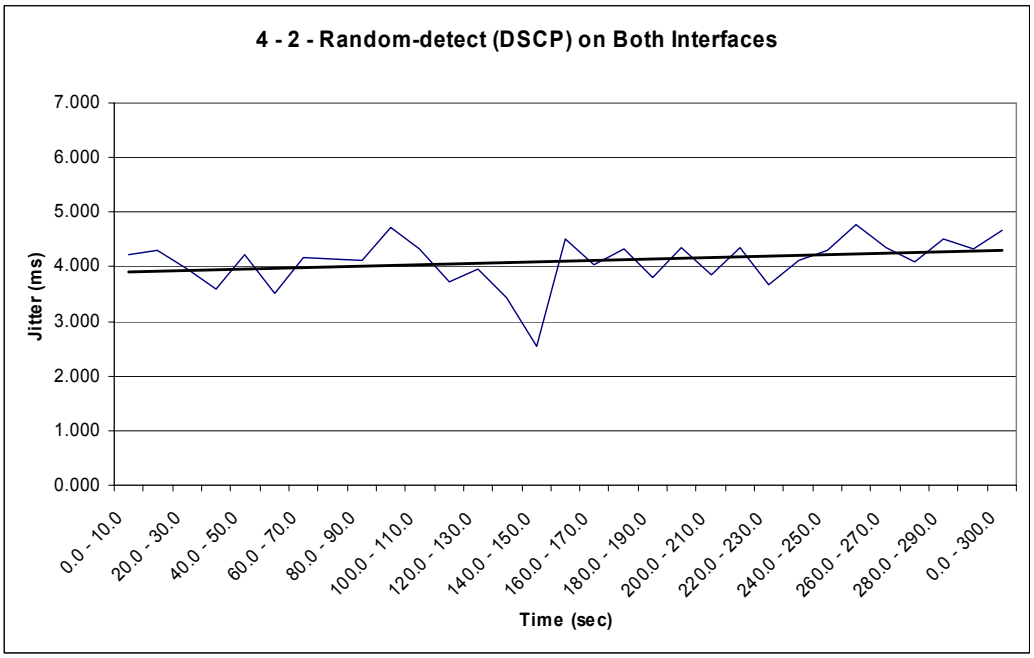
Very similar results were produced using fair-queuing on both interfaces. This is due to the same reasons outlined above.

4.1.3.6 Test 4 - 1 - Random-detect (DSCP) on Int G0/1



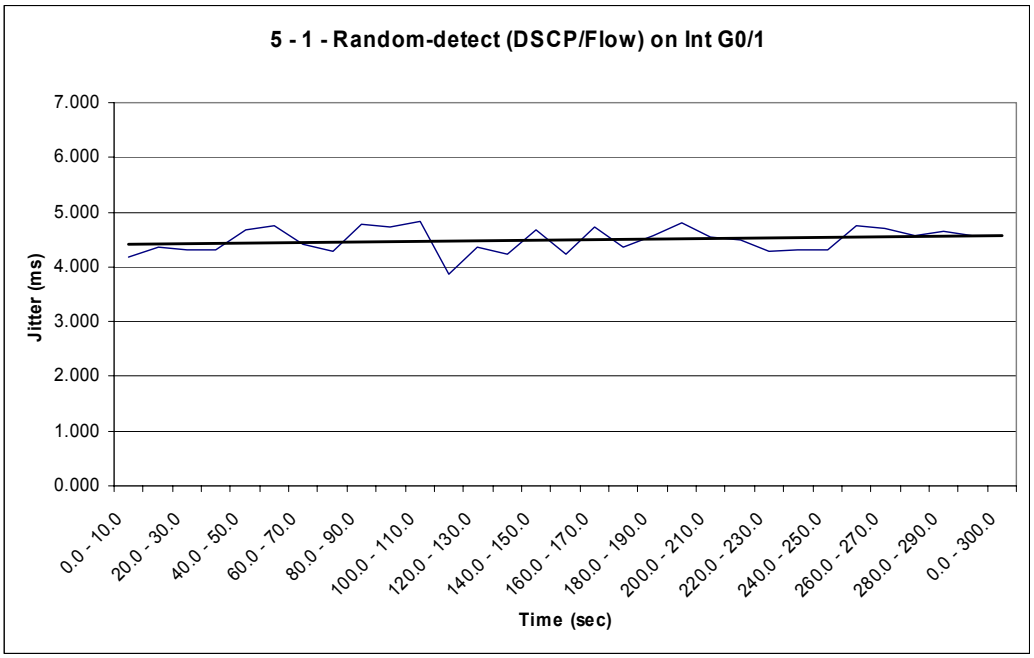
Random-detect with DSCP produced a slightly more stable UDP stream than the baseline. Random-detect should affect the more robust protocol of TCP which would explain why none of the UDP packets were dropped during the test.

4.1.3.7 Test 4 - 2 - Random-detect (DSCP) on Both Interfaces



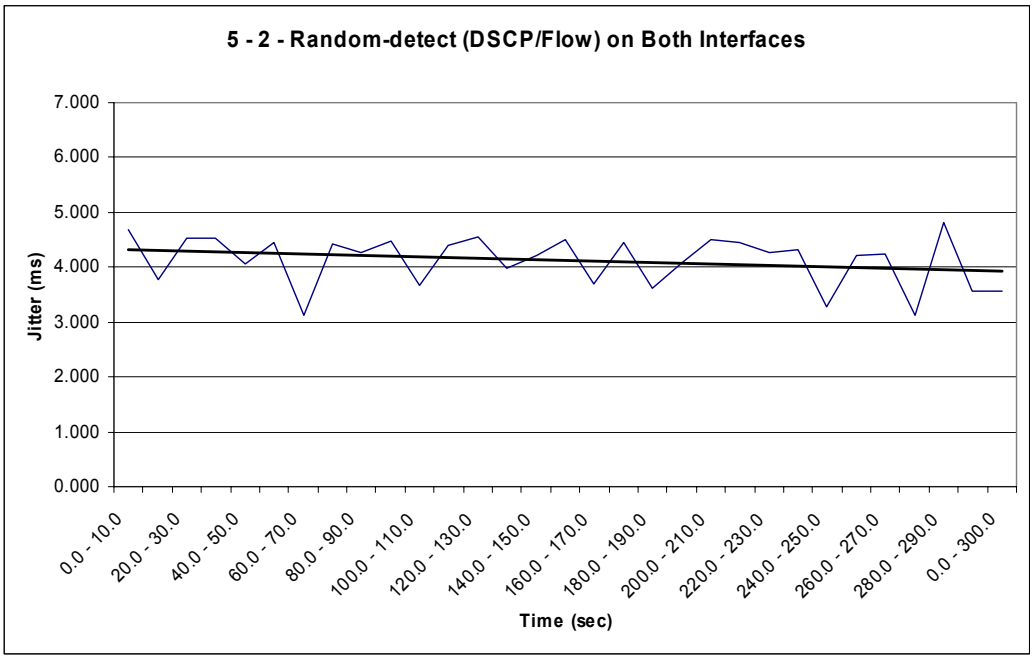
With both interfaces running Random-Detect with DSCP, the levels of jitter were stabilised further. It is clear from the results that at the 240 – 250 second point of the test, there was a large drop in jitter for 10 seconds. This level seems to be a rogue result, as in a re-run, a similar drop was not recorded.

4.1.3.8 Test 5 - 1 - Random-detect (DSCP/Flow) on Int G0/1



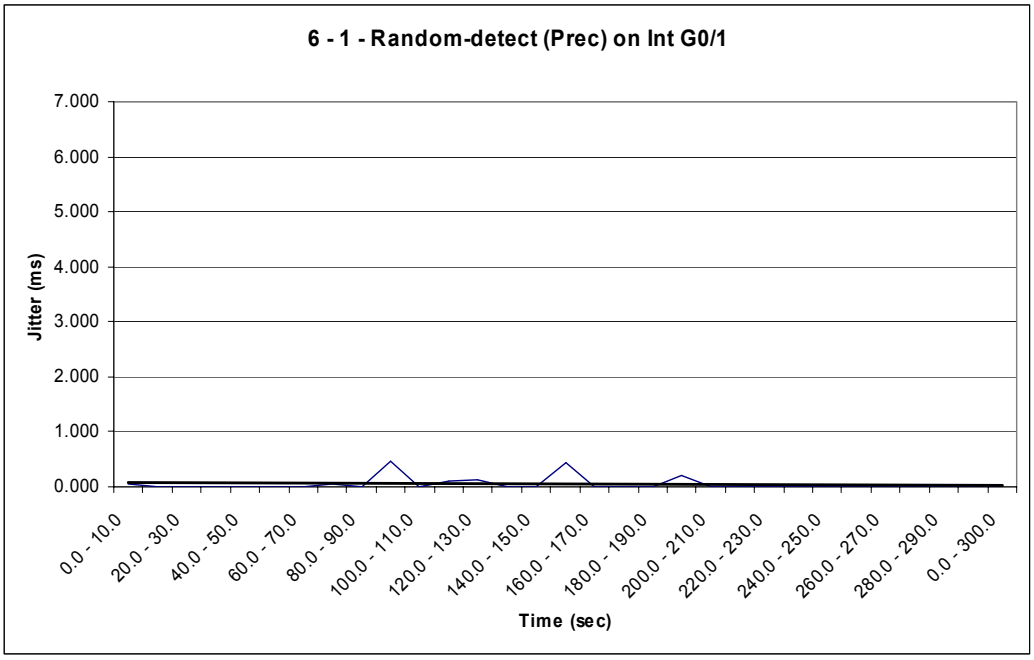
The flow-based Random-Detect results matched all expectations. As flow-based Random-Detect treats all data flows evenly, it would be expected that the average jitter would increase slightly compared to DSCP-based. Flow-based Random-Detect also prohibits one flow from monopolising the available bandwidth, thus explaining the steady level of jitter throughout.

4.1.3.9 Test 5 - 2 - Random-detect (DSCP/Flow) on Both Interfaces



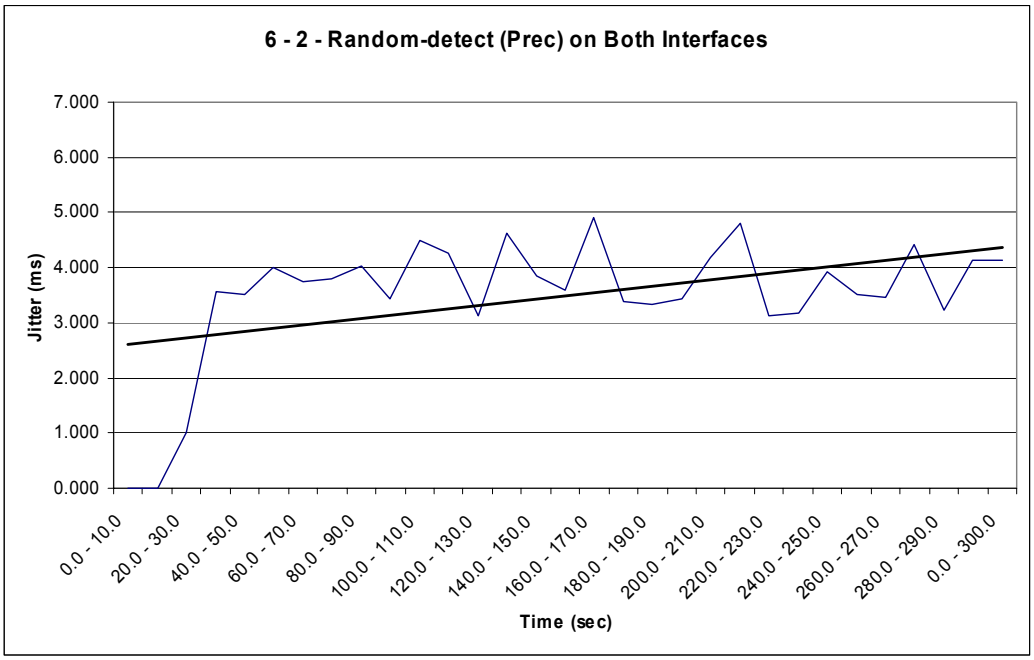
As can be seen, when flow-based Random-Detect is implemented on both interfaces, the jitter level is steadily decreased. Interestingly the stability of jitter here is less than that of the previous test. This would suggest that although having both interfaces using flow-based ransom-detect improves jitter levels, it does add slightly less stability. This will need to be looked at again in further tests.

4.1.3.10 Test 6 - 1 - Random-detect (Prec) on Int G0/1



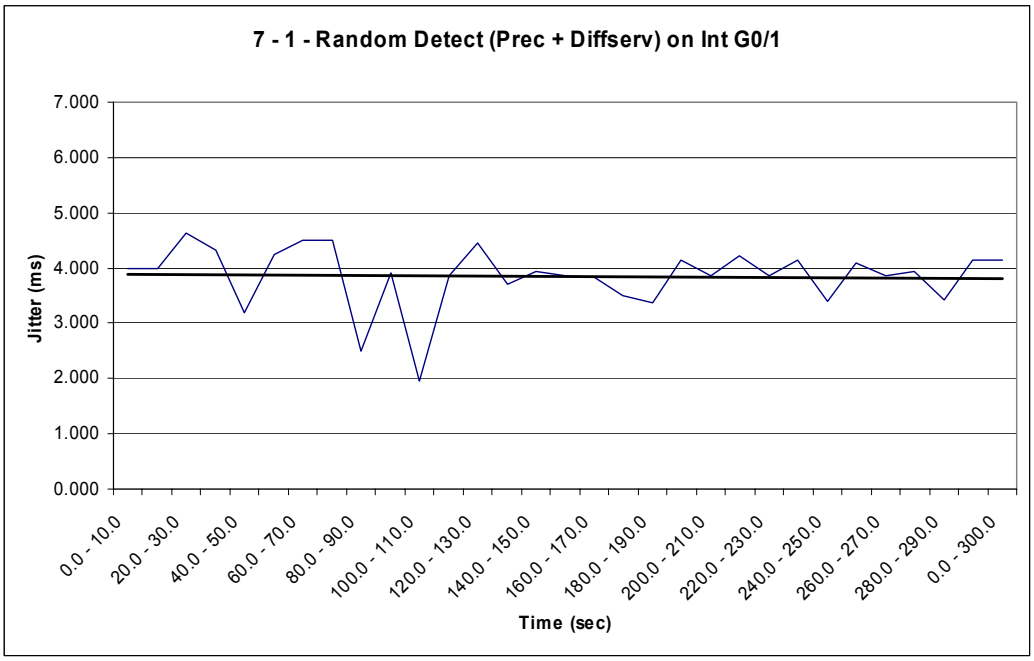
Results from the test were again as expected. The random-detect using IP precedence showed that the router's default settings allow a much greater precedence to UDP packets over TCP packets. This is shown through the average level of jitter being just 0.047ms, as compared to the baseline average of 4.715ms.

4.1.3.11 Test 6 - 2 - Random-detect (Prec) on Both Interfaces



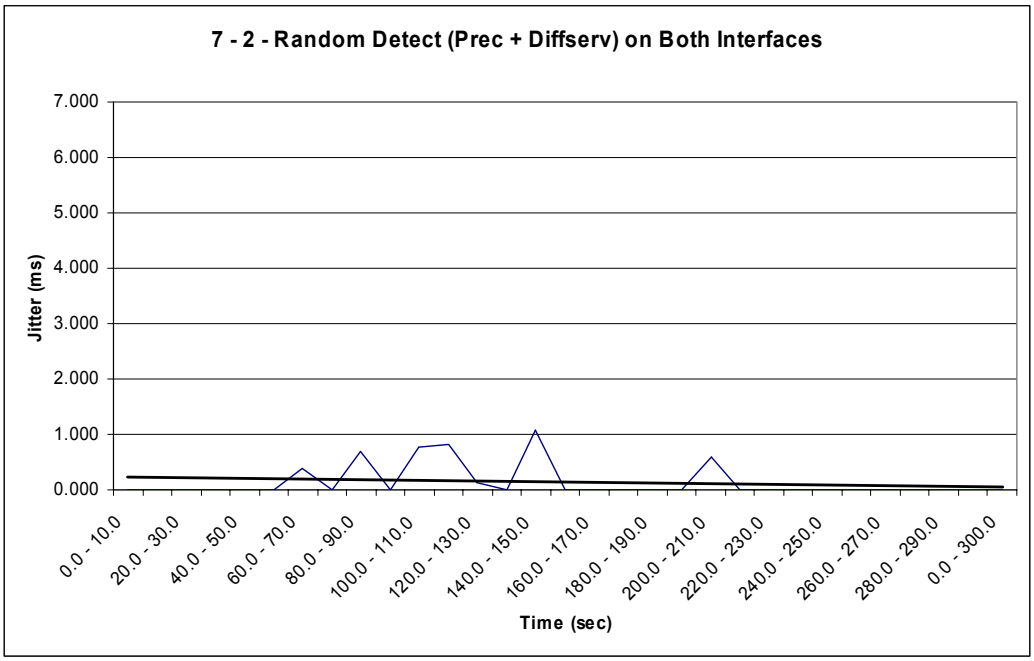
These results were very surprising. It seems that when Random-detect (Prec) was installed on both interfaces the jitter levels were very similar to the baseline. Apparently the two policies running side-by-side were conflicting and cancelling each other out. This may be an area for research in further studies.

4.1.3.12 Test 7 - 1 - Random Detect (Prec + DiffServ) on Int G0/1



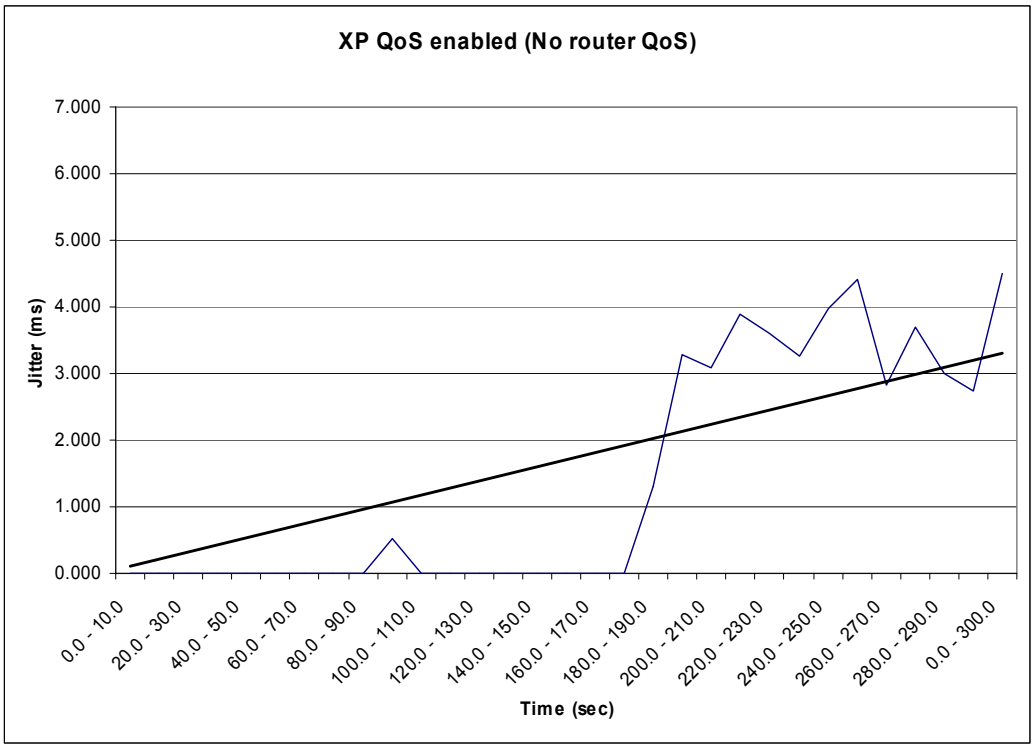
Combining Random Detect (Prec) and DiffServ provided slightly more stability to the jitter levels on the network than the baseline, but did not produce any results that would show a marked difference in performance.

4.1.3.13 Test 7 - 2 - Random Detect (Prec + DiffServ) on Both Interfaces



The anomaly noted in the test in section 4.1.3.11 was also apparent in this test. However this time when the policy was implemented on both interfaces, the jitter levels were significantly reduced - almost to 0. This seems to suggest that combining Random Detect and DiffServ on both interfaces provides a very favourable result for a UDP stream whilst running it on only one interface does not.

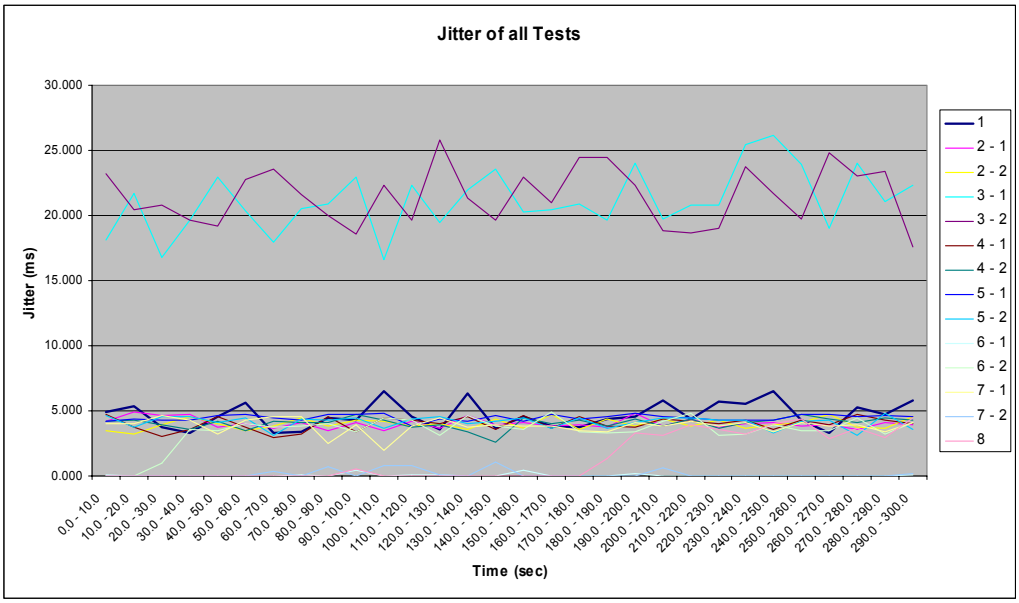
4.1.3.14 Test 8 - XP QoS enabled (No router QoS)



In the XP QoS test the router was set to have no QoS support whatsoever. Instead the QoS built into the Windows XP operating system on both PCs was enabled. XP uses QoS by giving a share of the total available bandwidth to all processes equally. XP allows a process explicitly to reserve 20% of the available link speed. If the process is not using the full 20% then XP allows other processes to make use of the available bandwidth. This would explain the jump at the 180 second mark during the test where the jitter rises to above 3ms. As the UDP stream was only using 1.05Mbit/s of the 10Mbit available (10%), it could be concluded that XP has given extra bandwidth and resources to the larger TCP stream at the 180 second mark, thus lowering resources on the router available to the UDP stream and increasing the UDP jitter. More information on how XP QoS operated can be found on the Microsoft website at: http://support.microsoft.com/?kbid=316666

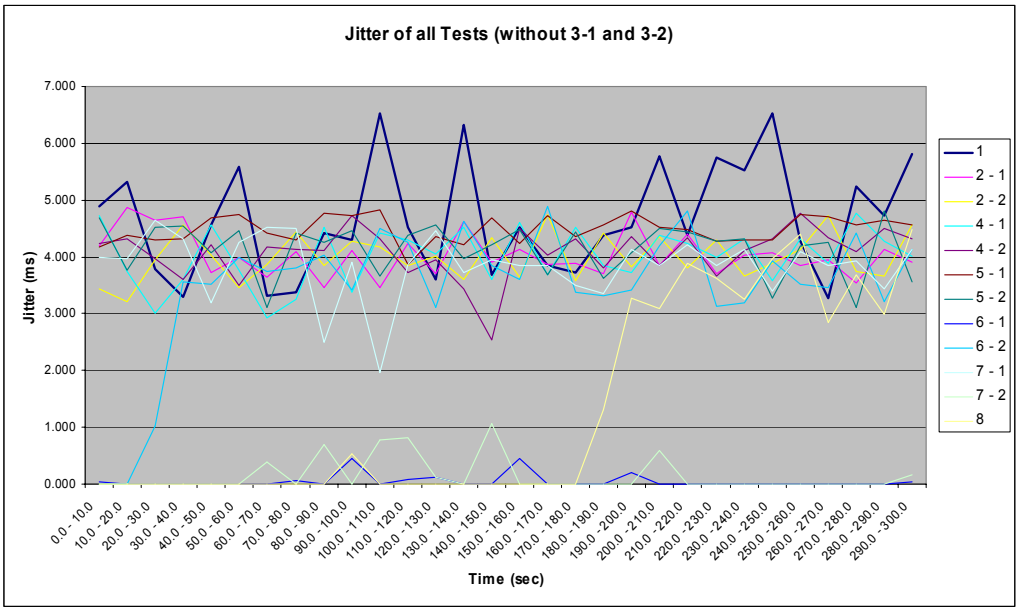
4.1.4 Analysis

Below are the jitter results from all the tests overlaid:



It can clearly be seen that tests 3-1 and 3-2 produced comparatively poor results. Tests 3-1 and 3-2 used the 'Fair-Queue' QoS strategy. These results can best be explained by the default settings on the Cisco 7200 router for fair-queuing not being matched very well for the purpose of giving a small UDP transfer priority over a large TCP transfer.

To enable better analysis of the results, test '3-1' and '3-2' have been taken out of the graph below, and the scale of the y axis re-configured accordingly:



In the graph above, the baseline has been highlighted to enable easy comparison with the other QoS tests. There is an obvious trend amongst many of the tests that sit in the 3 – 4ms range. These tests seem to suggest that the average level of jitter, although not decreased, is made more stable by these QoS strategies.

The tests that use random-detect with IP precedence are by far the most impressive, with average jitter levels at 0.047ms. This clearly shows that with precedence being set to favour UDP traffic, jitter levels can be virtually eliminated as a worry across a network running real-time applications.

5. Proposal to JANET Managers

The implementation of QoS across the entire JANET network is one of the largest nationwide IT projects that the UK will see this decade. With the JANET network supporting millions of separate computers, tens of thousands of routers and 100s of sub-networks; by definition this will be the biggest European QoS implementation. However, while there are overwhelming reasons for the implementation of QoS, as described in table 5.2 the risks involved are equally high.

Throughout the process of this evaluation and study, even though the scale of the network used was miniscule in comparison with the above, the one thing that has become most clear is the need to balance the improvement created by the QoS implementation with the potential dangers and costs of a poor implementation. Tests 3-1 and 3-2 showed how an ill-considered QoS strategy can throw up anomalies.

Although QoS has great potential to improve network performance it is not the only way. SuperJANET is constantly being upgraded and Manchester University are progressing to 10Gbps links very soon.

An overview of the options available to the JANET managers for network improvements and their relevant strengths and weakness is proposed below:

	Strength	**Weakness**	**Probable Outcome**	**Potential Threat**
Increase bandwidth	Easy	Expensive	Proven improved performance	Inflexible and limited
Congestion charge	Self regulating	Political and technical difficulties	Revenue generator	Possible revolt/abuse
QoS strategy	Flexible/ Rapid	Possible implementation and management problems	Improved performance	Reduced performance

Table 5.1 –Analysis of Network Improvements

Apart from QoS, the options outlined above have all been tried before in an effort to improve network quality. The other options have been shown to work up to a point, but now the JANET managers need to take a leap of faith and introduce QoS. Such a move will need to be carefully managed, but if undertaken correctly, QoS could provide a cost-effective alternative to other available options. Of course, a QoS implementation will bring with it a whole host of costs and benefits, an example of these is illustrated in the table below:

Benefits	Underlying Costs
Improved performance to users of time-sensitive applications such as VoIP	Slight lowering in performance to users of lower priority traffic such as e-mail
Improved control over the network for network managers	Initial 'teething' problems when first QoS implementation is rolled out, and possibility of reducing some performance levels
Possibility of charging for premium bandwidth access	Time and money spent on initial verification of QoS strategy and its implementation
JANET becoming more flexible to accepting new network uses	Extra administration required when new network use is introduced
Managers able to reduce non-priority bandwidth 'hogs', thus lowering network overheads	Each bps taken for one service is a bps taken from another
Make JANET network managers more aware of who is using their network and for what reasons	Possibility of abuse of QoS policies by users sending non-priority traffic with fake priority IP headers
Reduced expenditure on continually improving bandwidth due to improved use of existing links	Extra bandwidth will still need to be introduced at some point, but not as often as is currently done

Table 5.2 – Costs and Benefits of JANET QoS

A network on the scale of JANET demands a clear strategy for deploying any implementation. It is the view of this report that a timetable of deployment should be created that will act as a 'roadmap' for QoS implementation. A

provisional outline timescale could be as shown in table 5.3.

Date	Task
Month 1	Creation of JANET QoS team with representatives from: UKERNA, academic, industry and research.
Month 2	Request sent out to JANET member networks for QoS interest
Month 3	Networks selected as test base
Month 4	Sniffers deployed on test network for 3 months to gather information on network usages
Month 7	Data analysed by QoS team and beta QoS strategy created
Month 9	QoS strategy deployed across test networks
Month 10	Test networks are monitored for performance improvements
Month 10 onwards	Test networks report back to QoS team on performance
Month 12	If the implementation is successful the QoS policy will be expanded within the test networks towards the JANET core (superJANET)
Month 14	Final QoS policy decided and deployed across JANET on all edge networks
Month 15	QoS strategy implemented towards superJANET once proven successful
Month 18	JANET QoS implemented section-by-section with constant monitoring of performance

Table 5.3 – Timeline for QoS deployment across JANET

By following such a roadmap, JANET would hopefully have a well constructed QoS policy that would provide a sustainable improvement in network performance while also saving costs on bandwidth usage.

Of course rarely do projects go ahead as expected, so changes to the roadmap should be anticipated and dealt with throughout the project's life. Areas that may affect the progress of the QOS deployment include: internal politics within subsidiary JANET networks; budgetary constraints; staff training and availability and obtaining 'buy-in' from all network managers within JANET. Another area which must be monitored closely is the introduction of IPv6, as this new version of IP has a much more advanced and pre-planned approach to QoS and, if implemented, could affect how a QoS deployment would be carried out.

To enable the most efficient deployment of QoS across the networks connected to JANET it is clear that implementing QoS 'by hand' on routers will be very time consuming. Instead, the use of a third party QoS deployment tool such as VizIq should be promoted to network managers so that any changes to the JANET QoS policy in the future can be distributed immediately across the networks. This will also help network managers that will be running multiple vendor routers and switches and will result in less training required for networking staff.

Throughout the QoS deployment network managers must be reminded that the QoS implementation will be a 'trial and error' task and that the QoS policy is unlikely to work fully first time. It was clearly demonstrated in a small way within this project how a good QoS policy can help and a bad one can hinder. Even though the implementation was on an extremely small scale, QoS was shown to lower jitter to less than 1% of the jitter on a best-effort network or make jitter worse by a huge 445%.

A balance must be found that provides the best delivery of the JANET service while also meeting/exceeding JANET's user expectations.

The benefits of QoS can only be assessed when looking at all elements of the overall operation; as a QoS policy on one segment may not complement the policy on another segment as expected.

6. Project Evaluations

6.1 Critical Evaluation

This section will concentrate on how well this project was carried out

6.1.1 Project Selection

The project was undertaken for two main reasons:

- The issue of QoS is a new and exciting area within modern networking. It affords network managers the opportunity to offer a whole new level of service to their customers, one which can be tailor-made to individual network segments or users.
- After gaining experience in IP videoconferencing, the author saw how bandwidth is the single deciding factor in how well a videoconference will run. It was obvious that if QoS were running, the IP videoconferencing service would become much improved and would possibly become a lucrative area for generating revenue.

6.1.2 Methodology Selection

Unfortunately, with QoS being such a new area in networking, there are no tried and tested methodologies for implementation. Furthermore, as every QoS implementation is tailor-made for each network, there will never be a 'one size fits all' solution. The QoS mechanisms used and investigated within this report cover the wide variety of QoS mechanisms available and aim to show how tests can be performed to asses the affect that QoS can have on the complete network. A major factor in selecting which QoS mechanisms to use in the implementation was which QoS mechanisms were available on the Cisco 7200 router being used and how much time was available to configure each mechanism.

6.1.3 Investigation and Analysis

The improvements (and possible hindrance) that QoS can bring to a network were extremely well illustrated by the fact that a QoS strategy could either improve jitter by lowering it to 0.047ms or degrade QoS by worsening jitter to 20.996ms. This investigation, although on an extremely small scale, was able to show to a network manager how important it is not just to implement QoS

but also to make sure that it is implemented correctly for that network. Through the analysis of the data collected it was also possible to see that QoS can throw up anomalous results, such as in tests 4.1.3.10 and 4.1.3.11, where implementing the same QoS mechanism on both the input and output interface will have a very different effect to just installing QoS on one interface.

When looking at the graphs in section 4.1.3 it should always be remembered that although the levels of jitter for the UDP stream were lowered in some cases, a lowering in the jitter for the UDP stream would mean degradation in the service given to the TCP stream. In a future implementation, data should also be collected on how the QoS affects the TCP stream to ensure that it is kept within reasonable levels.

6.1.4 Design and Implementation

Although the resources available for the implementation within this report meant that the network used was very simplistic, it did nevertheless provide a meaningful indication of how QoS can affect network performance. The QoS mechanisms chosen for the implementation were selected for their wide variety of ways of handling QoS so that it could be shown how different types of QoS handle traffic across a network. There are many improvements that could have been made to the implementation given more time and resources (covered in section 7.2), but with those that were available, this project was successful in so far as it showed how a well implemented QoS mechanism can enhance a network and a badly chosen QoS mechanism can also severely damage performance.

When looking at the graphs in section 4.1.3 it should be noted that, although the levels of jitter for the UDP stream were lowered in some cases, a lowering in the jitter for the UDP stream would most probably mean degradation in the service given to the TCP stream.

6.1.5 Time Management

The time management throughout this project did not waver from the original plan. All the tests were carried out exactly as planned, due in part to the level

of preparation and co-operation shown by the Network Systems team at Manchester Computing prior to the implementation phase.

The timescale first set out in the project specification was followed and adhered to, which shows the initial time plan was well thought out and represented a realistic view as to how the project would proceed.

6.2 Future Improvements

The implementation within this report was on an extremely small scale covering only the default settings for the QoS built into the Cisco 7200 router. In any future implementations it would be advisable to use a more 'realistic' network with multiple routers over a larger area. Also the traffic used on the network could be closer to what would be expected on the actual JANET network. This could easily be generated through the use of a program such as NetIQ's Chariot. Chariot is a bandwidth creation utility that offers a great level of granularity in both traffic selection and analysis. Packet sniffers across the network would provide a realistic representation of JANETs traffic that could be put into Chariot. Chariot would then reproduce this traffic and allow analysis of how QoS would improve network performance.

A further improvement to an implementation test would be to use a network simulation tool such as OPNET that supports QoS simulation. OPNET would allow the network managers to re-build a large network segment with very fine detail and run real network traffic within the simulation with QoS. This would mean changes could be made to the network without any risk of affecting the real-world network's performance.

As for the network use and topology research that was undertaken within this report, the scope could be widened in future to include a variety of subsidiary JANET networks, rather than just concentrating on Manchester University. This would give a clearer indication to the JANET managers of what the requirements of the subsidiary sites are and how QoS would best serve them.

6.3 Conclusion

The area of QoS is in development and still has a long way to go, especially with the introduction of new network technologies such as IPv6. With this in mind, the managers of JANET should look carefully at what they and their users expect from QoS and what can be realistically delivered. With software products that aid the deployment of QoS continually being created, it is obvious that QoS is a technology that is here to stay. The network managers of JANET and its subsidiary networks will need to embrace QoS in a positive and committed way, to ensure that they will continue to offer the best possible service to their users for many years to come.

References

Arizona College, 2001, *Peer to Peer Network Effects*, [online], last accessed 04th April 2003 at URL:
http://virgil.azwestern.edu/~seth/pics/bandwidth.gif

Cisco, 2002, *Cisco 7200 VXR Router Product Overview*, [online], last accessed 04th April 2003 at URL:
http://www.cisco.com/univercd/cc/td/doc/product/core/7200vx/72vxicg/over6vx.pdf

Cisco, 2003, *Cisco QoS Device Manager Overview*, [online], last accessed 04th April 2003 at URL:
http://www.cisco.com/warp/public/cc/pd/nemnsw/qodvmn/prodlit/qdm_ds.pdf

Corporate Information Systems, 1998, *Understanding ATM Networking*, Sheffield Hallam University

Microsft.com, 2001, *Windows XP QoS*, [online], last accessed 04th April 2003 at URL:
http://support.microsoft.com/?kbid=316666

RFC 2205, 1997, *Resource ReSerVation Protocol (RSVP)*, [online], last accessed 04th April 2003 at URL:
ftp://ftp.rfc-editor.org/in-notes/rfc2205.txt

Tanenbaum, 2003, *Computer Networks*, Fourth Edition, Pearson Education International

Velte and Velte, 2001, *Cisco: A Beginner's Guide*, Second Edition, Osbourne

WhatIs.com, 2001, *Differentiated Services Definition*, [online], last accessed 04th April 2003 at URL:
http://whatis.techtarget.com/definition/0,,sid9_gci213845,00.html

WhatIs.com, 2000, *RSVP Definition*, [online], last accessed 04th April 2003 at URL:
http://searchsystemsmanagement.techtarget.com/sDefinition/0,,sid20_gci214274,00.html

Bibliography

Aurrecoechea, Campbell and Hauw, 1999, *A Survey of QoS Architectures*, Columbia University

Constable and Price, March 2001, *The Need for Quality of Service Mechanisms*, UKERNA,

Cisco, 1999, *Internetworking Technology Handbook 1999*, Chapter 46 – QoS Networking, Cisco

Cisco, 2001, *Internetworking Technology Handbook 2001*, Chapter 49 – QoS Networking, Cisco

Cisco, 2002, Cisco *7200 VXR Router Installation and Configuration Guide*, [online], last accessed 04th April 2003 at URL:
http://www.cisco.com/application/pdf/en/us/guest/products/ps341/c1099/ccmigration_09186a00800eec5e.pdf

CramSession.com, 2002, *Cisco Internetwork Design 3.0 - Links and Resources*, [online], last accessed 04th April 2003 at URL:
http://studyguides.cramsession.com/cramsession/cisco/cid3/links.asp

Distributed Applications Support Team, 2003, *Iperf*, [online], last accessed 04th April 2003 at URL:
http://dast.nlanr.net/Projects/Iperf/

Fitzgerald and Dennis, 1999, *Business Data Communications and Networking*, Sixth Edition, John Wiley and Sons

Freesoft.org, 2003, *IP Packet Structure*, [online], last accessed 04th April 2003 at URL:
http://www.freesoft.org/CIE/Course/Section3/7.htm

gridpp.ac.uk, 2003, *The UK Grid for Particle Physics*, [online], last accessed 04th April 2003 at URL:
http://www.gridpp.ac.uk/

JA.net, 2003, *JANET QoS Development*, [online], last accessed 04th April 2003 at URL:
http://www.ja.net/development/qos/qos_dev.html

Leinwand, 2001, *Cisco Router Configuration*, Second Edition, Cisco Press

Lucent Technologies, 1999, *QoS: The IP Solution*, Lucent Technologies,

NetIQ, 2003, *NetIQ: Chariot*, [online], last accessed 04th April 2003 at URL:
http://www.netiq.com/products/chr/default.asp

Nortel, 1998, *IP QoS: A Bold New Network*, Nortel

nwfusion.com, 1998, *QoS in English*, [online], last accessed 04th April 2003 at URL:
http://www.nwfusion.com/netresources/0817qos.html

RFC 2212, 1997, *Specification of Guaranteed Quality of Service*, [online], last accessed 04th April 2003 at URL:
ftp://ftp.rfc-editor.org/in-notes/rfc2212.txt

RFC 2990, 2000, *Next Steps for the IP QoS Architecture*, [online], last accessed 04th April 2003 at URL:
ftp://ftp.rfc-editor.org/in-notes/rfc2990.txt

RFC 2998, 2000, *A Framework for Integrated Services Operation over DiffServ Networks*, [online], last accessed 04th April 2003 at URL:
ftp://ftp.rfc-editor.org/in-notes/rfc2998.txt

RFC 3387, 2002, *Considerations from the Service Management Research Group (SMRG) on QoS in the IP Network*, [online], last accessed 04th April 2003 at URL:
ftp://ftp.rfc-editor.org/in-notes/rfc3387.txt

UKERNA, July 2001, *Report of Quality of Service Think Tank*, UKERNA,
Hagen, 2002, *IPv6 Essentials*, Chapter 6, First Edition, O'Reilly

WhatIs.com, 2002, *CoS Definition*, [online], last accessed 04th April 2003 at URL:
http://whatis.techtarget.com/definition/0,,sid9_gci213827,00.html

Appendix A

Example policy map excerpt for a Cisco 7200 router

```
Policy Map VOIP
     Class platinum
       Weighted Fair Queuing
             Strict Priority
             Bandwidth 500 (kbps) Burst 12500 (Bytes)
     Class gold
       Weighted Fair Queuing
             Bandwidth 35 (%) Max Threshold 64 (packets)
     Class silver
       Weighted Fair Queuing
             Bandwidth 25 (%) Max Threshold 64 (packets)
     Class bronze
       Weighted Fair Queuing
             Bandwidth 15 (%) Max Threshold 64 (packets)
     Class best-effort
      police 56000 1750 1750 conform-action set-dscp-transmit 0
      exceed-action drop violate-action drop
```

Appendix B

Cisco 7200 Configuration

Default running config of Cisco 7200 router:

```
gw-QOS_test#show running-config
Building configuration...

Current configuration : 1106 bytes
!
version 12.2
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname gw-QOS_test
!
enable secret 5 $1$/Aag$wnuJOkua1FjziMAd46kbh.
enable password *******
!
ip subnet-zero
!
!
!
interface GigabitEthernet0/1
 ip address 192.168.0.250 255.255.255.0
 no ip mroute-cache
 duplex auto
 speed 10
 media-type rj45
 no negotiation auto
 no cdp enable
!
interface GigabitEthernet0/2
 ip address 192.168.1.250 255.255.255.0
 no ip mroute-cache
 duplex auto
 speed 10
 media-type rj45
 no negotiation auto
 no cdp enable
!
interface GigabitEthernet0/3
 no ip address
 no ip mroute-cache
 shutdown
 duplex auto
 speed auto
 media-type rj45
 no negotiation auto
 no cdp enable
!
ip classless
no ip http server
ip pim bidir-enable
```

```
!
dialer-list 1 protocol ip permit
dialer-list 1 protocol ipx permit
!
!
call rsvp-sync
!
!
mgcp profile default
!
dial-peer cor custom
!
!
!
gatekeeper
 shutdown
!
!
line con 0
 exec-timeout 0 0
 stopbits 1
line aux 0
 stopbits 1
line vty 0 4
 password *********
 login
!
!
end
```

Interfaces on Cisco 7200 router showing default configuration with cleared queues

---

```
gw-QOS_test#show int g0/1

GigabitEthernet0/1 is up, line protocol is up
  Hardware is BCM1250 Internal MAC, address is 000b.bf90.381b
  (bia 000b.bf90.381b)
  Internet address is 192.168.0.250/24
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 10Mb/s, media type is RJ45
  output flow-control is off, input flow-control is off
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:03:02, output 00:00:04, output hang never
  Last clearing of "show interface" counters 00:01:15
  Queueing strategy: fifo
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     0 input packets with dribble condition detected
     8 packets output, 480 bytes, 0 underruns
```

```
     0 output errors, 0 collisions, 0 interface resets
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
```

```
gw-QOS_test#show int g0/2

GigabitEthernet0/2 is up, line protocol is up
  Hardware is BCM1250 Internal MAC, address is 000b.bf90.381a
  (bia 000b.bf90.381a)
  Internet address is 192.168.1.250/24
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 10Mb/s, media type is RJ45
  output flow-control is off, input flow-control is off
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:20, output 00:00:05, output hang never
  Last clearing of "show interface" counters 00:00:05
  Queueing strategy: fifo
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 27000 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     0 input packets with dribble condition detected
     1 packets output, 60 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
```

Appendix C

Raw Iperf Output Example:

```
[1960] local 192.168.0.1 port 5001 connected with 192.168.1.1 port 1199
[ ID] Interval    Transfer   Bandwidth    Jitter  Lost/Total Datagrams
[1960] 0.0-10.0 sec 1.73 MBytes 1.45 Mbits/sec 1.829 ms  0/ 1232 (0%)
[1960] 10.0-20.0 sec 1.73 MBytes 1.45 Mbits/sec 3.556 ms  0/ 1232 (0%)
[1960] 20.0-30.0 sec 1.76 MBytes 1.48 Mbits/sec 2.129 ms  0/ 1259 (0%)
[1960] 30.0-40.0 sec 1.73 MBytes 1.45 Mbits/sec 2.756 ms  0/ 1232 (0%)
[1960] 40.0-50.0 sec 1.73 MBytes 1.45 Mbits/sec 3.311 ms  0/ 1233 (0%)
[1960] 50.0-60.0 sec 1.73 MBytes 1.45 Mbits/sec 2.189 ms  0/ 1231 (0%)
[1960] 60.0-70.0 sec 1.73 MBytes 1.45 Mbits/sec 3.313 ms  0/ 1231 (0%)
[1960] 70.0-80.0 sec 1.73 MBytes 1.45 Mbits/sec 1.845 ms  0/ 1234 (0%)
[1960] 80.0-90.0 sec 1.73 MBytes 1.45 Mbits/sec 1.841 ms  0/ 1233 (0%)
[1960] 90.0-100.0 sec 1.72 MBytes 1.45 Mbits/sec 2.625 ms  0/ 1230 (0%)
[1960] 100.0-110.0 sec 1.73 MBytes 1.45 Mbits/sec 1.783 ms  0/ 1237 (0%)
[1960] 110.0-120.0 sec 1.95 MBytes 1.64 Mbits/sec 2.717 ms  0/ 1393 (0%)
[1960] 120.0-130.0 sec 1.73 MBytes 1.45 Mbits/sec 1.981 ms  0/ 1233 (0%)
[1960] 130.0-140.0 sec 1.73 MBytes 1.45 Mbits/sec 2.804 ms  0/ 1231 (0%)
[1960] 140.0-150.0 sec 1.73 MBytes 1.45 Mbits/sec 1.271 ms  0/ 1232 (0%)
[1960] 150.0-160.0 sec 1.73 MBytes 1.45 Mbits/sec 1.878 ms  0/ 1231 (0%)
[1960] 160.0-170.0 sec 1.73 MBytes 1.45 Mbits/sec 1.412 ms  0/ 1232 (0%)
[1960] 170.0-180.0 sec 1.73 MBytes 1.45 Mbits/sec 2.050 ms  0/ 1236 (0%)
[1960] 180.0-190.0 sec 1.72 MBytes 1.45 Mbits/sec 0.908 ms  0/ 1230 (0%)
[1960] 190.0-200.0 sec 1.72 MBytes 1.45 Mbits/sec 1.443 ms  0/ 1230 (0%)
[1960] 200.0-210.0 sec 1.72 MBytes 1.45 Mbits/sec 1.568 ms  0/ 1230 (0%)
[1960] 210.0-220.0 sec 1.73 MBytes 1.45 Mbits/sec 1.338 ms  0/ 1231 (0%)
[1960] 220.0-230.0 sec 1.73 MBytes 1.45 Mbits/sec 1.305 ms  0/ 1232 (0%)
[1960] 230.0-240.0 sec 1.73 MBytes 1.45 Mbits/sec 1.022 ms  0/ 1231 (0%)
[1960] 240.0-250.0 sec 1.72 MBytes 1.44 Mbits/sec 0.887 ms  0/ 1229 (0%)
[1960] 250.0-260.0 sec 1.73 MBytes 1.45 Mbits/sec 2.236 ms  0/ 1232 (0%)
[1960] 260.0-270.0 sec 1.73 MBytes 1.45 Mbits/sec 1.244 ms  0/ 1234 (0%)
[1960] 270.0-280.0 sec 1.72 MBytes 1.44 Mbits/sec 0.633 ms  27/ 1251 (2.2%)
[1960] 280.0-290.0 sec 1.72 MBytes 1.45 Mbits/sec 1.108 ms  0/ 1230 (0%)
[1960] 290.0-300.0 sec 1.72 MBytes 1.44 Mbits/sec 0.890 ms  0/ 1228 (0%)
[1960] 0.0-300.0 sec 52.1 MBytes 1.46 Mbits/sec 0.890 ms  27/37160 (0.073%)
```