UNIVERSITY OF OSLO
Department of Informatics

in collaboration with

OSLO UNIVERSITY COLLEGE
Department of Computer Science

MASTER THESIS

ADMINISTRATION OF REMOTE COMPUTER NETWORKS

Stig Jarle Fjeldbo

May 23, 2005

Abstract

Today's computer networks have gone from typically being a small local area network, to wide area networks, where users and servers are interconnected with each other from all over the world. This development has gradually expanded as bandwidth has become higher and cheaper.

But when dealing with the network traffic, bandwidth is only one of the important properties. Delay, jitter and reliability are also important properties for the quality of network connection. This is because different applications has different needs, and therefore require different properties from the network.

System administrators are in an increasing degree involved with the troubleshooting of solving network problems concerning the quality of service for the different applications.

This thesis analyzed techniques for measuring, analyzing, presenting and interpreting the different properties for the administration of remote computer network. In this way system administrators can benefit from this thesis when administrating their remote computer networks.

Acknowledgements

First I would like to give my appreciation to University of Oslo and Oslo University College, for giving me this opportunity to write the master thesis.

I would especially give my appreciation to Mark Burgess and the other teachers who have worked hard for making this master program a reality.

I would also like to thank my thesis advisor, Tore M. Jonassen, and the other teachers at Oslo University Collage who has contributed to my master thesis.

My fellow students deserve my gratitude, for they have made these last two years a good experience. They have also contributed with ideas and support during the creation of this master thesis.

And last, my thanks to my family and friends who has provided support and patience, during this master thesis. I could never have done it without you...

Table of Contents

Chapter 1

Introduction

The most used network architecture is the client-server architecture. In a client-server architecture the server passively waits for a request, until the client actively sends a request to the server. The server then executes the request and sends the reply back to the client.

One of the first computer networks were isolated local area networks (LANs), with a client-server architecture. The clients were cheap terminals, attached to a screen and a keyboard. At the time, the clients required low network bandwidth. The only data transmitted was the keyboard activity sent to the server, and the screen updates sent back to the client.

The terminals used in these networks are classified as thin clients. This is because most of the processing is done at the server, while the client typically process keyboard input and screen output.

Some advantages with the thin client approach are:

- A lower hardware costs, as there is usually no need for disk, a lot of memory, or a powerful processor. This also creates a longer turnover time, because it takes a longer period of time before the equipment becomes obsolete.
- A lower administration cost, as the clients are almost completely managed from the server. All installations and upgrades are done on the servers, and not on each client.
- A higher client reliability, as the client hardware has less points of failure.
- Increased security, as no sensitive data ever resides on the client. The local environment is usually highly restricted, and the protection against malware is centralized on the servers.

The need to connection to other networks or clients from the existing network, created the next step for computer networks. The connection between the networks was typically created by leased lines or by dial-up connections. The new networks were called metropolitan area networks (MAN) or wide area networks (WAN) depending on the range of the networks. With the creation of these new networks, terminals could now connect to other servers in other networks, and process data in other computer environments.

The personal computer (PC) was intended to conquer the private marked, but the corporate marked also showed great interest. And as time went by, the pc replaced the terminal as the preferred client.

The pc can be a thick client, because it has a disk, memory and a powerful processor that allows the client to run its own operating system and programs. But even though the pc has the properties of a thick client, it can behave like a thin client. This all depends on the software that the pc is running. Applications like telnet and ssh emulates a thin client environment, because the applications connects to a remote server, and utilizes the resources which is provided by that server. Keyboard actions are sent to that server, and the server only replies with screen changes, just like in a thin client environment.

Traditionally the server had processed both the client environment and the production environment. But with the arrival of the pc, user environment processing could be removed from the servers, and done on the clients own processor. This meant a more efficient usage of the processing servers. In situations where the data could be stored on the pc itself, the processing of the production data could be executed on the local processor. But this moved the bottleneck away from the production server processors, and to the network bandwidth.

Some advantages with the thick client approach are:

- Lower server requirements, as a thick client does most of the application processing itself.
- Lower user environment network bandwidth usage, because there is no keyboard or screen data that has to be sent to and from the server.
- Higher system reliability, as the thick clients can operate even when the processing servers are unavailable.
- Better multimedia processing, because multimedia processing require high bandwidth and high performance processors.

The internet started off as a few computer networks interconnected with each other. The connection speed, at that time, was only about 64 kilobits per second (kb/s), and the connection between the networks was within the United States of America. Since then, hundreds of millions of people, all around the world, has connected to the internet.

The bandwidths available today typically range from 64 kilobits per second on dial-up connections, to gigabits per second on high performance broadband connections. The high bandwidth available on the internet today, enables new possibilities for network applications.

But bandwidth is not the only property for a good internet connection. Properties like delay, jitter, and reliability have become the main focus area in the resent years. Together these four properties make up the basis for quality of service (QoS).

As the internet service providers improve their quality of service, this enables organizations and businesses to structure their computer networks in new ways. There is no longer the need for one location where both the user environment and the production environment are located. Examples of these new possibilities are:

- Employees may connect to the production environment from their home.
- The same production environment can be used for several remote user environments.
- Multiple remote production environments can be interconnected.

This allows the business to easier create new locations in other countries. But for the system administrators who are used to operate in a local area network environment, this creates new problem areas. This is because most programs are intended to run in a local area network with low delay, low jitter, high bandwidth, and high reliability. The new tasks that the system administrator has to adapt to are how to locate and remove bottlenecks in remote computer networks. To do this, the understanding of what these quality of service properties do, and how to overcome them.

This thesis will investigate in the properties of quality of service, and use basic measurement tools for aiding the system administrators to measure and analyze their internet connection. This helps the system administrators, so that they can adapt their applications to their internet connections, or their internet connection to their applications.

Chapter 2

Background

2.1 Computer Networks

2.1.1 Network Classification

Computer networks are classified by the range of the network. Networks that range a few meters are classified as personal area networks (PAN). Networks that range a few hundred meters are classified as local area networks (LAN). When grouping several local area networks together, within a range of some kilometers, the network is classified as a metropolitan area network (MAN). And any networks ranging more then some kilometers are classified as Wide Area Networks (WAN).

Personal Area Network (PAN)

Personal devices interconnected within a few meters are considered to be a personal area network (PAN). The use of a personal area network may be communication between the personal devices and connection to higher level networks, like the Internet[1].

Personal area networks may be wired with computer buses such as USB and Firewire. Wireless personal area networks (Wireless PANs) are available through technologies such as IrDA and Bluetooth.

Local Area Network (LAN)

Devices interconnected within an area of 1000m2, which is the generally accepted maximum size for a LAN, are considered to be in a local area network (LAN)[2]. Low latency and high bandwidth are typically properties which describe a local area network[3].

Local area network technologies are Token Ring, Ethernet, Fast Ethernet, CDDI, FDDI and the newly emerging Gigabit Ethernet[2]. These technologies are typically design to run on either twisted-pair cables or optical fibre cables.

Wireless local area network (WLAN) technologies are 802.11a, 802.11b and 802.11g, although new technologies are soon to come. These new technologies will provide a wider range, higher bandwidth, increased security and quality of service[4].

Metropolitan Area Network (MAN)

Metropolitan area networks or MANs are large computer networks usually spanning a campus or a city. They typically use optical fibre connections to link their sites, running technologies like ATM, FDDI, SMDS or Gigabit Ethernet. The ATM, FDDI and SMDB technologies are beginning to be displaced by Gigabit Ethernet-based MANs[1][3].

Wide Area Network (WAN)

A wide area network or WAN is a computer network covering a wide geographical area, and are used to connect local area networks together. Wide area networks can be built to connect several private local area networks in a organization, or built by Internet service providers (ISPs) to provide an organization access to the Internet[1][3].

Wide area networks are typically built of leased lines, where a router connects the local area network to the private wide area network[5].

An alternative is to use the Internet, which provides a shared infrastructure and a high speed wide area network. Virtual private networks (VPNs) can use encryption and other techniques to make the connection secure and private[5].

2.1.2 Network Topologies

There are several possibilities when connecting several nodes together in a computer network. These possibilities are called network topologies.

The computer networks are designed by purpose and importance. While some networks require high bandwidth and high reliability, other networks require high bandwidth and low cost. The different topologies will suite different networks, depending on their needs.

Line

The nodes in a line topology have maximum two neighboring nodes. Data transmitted from one end of the network to the other end, will have to travel through all the other nodes in that network[6][7]. The line topology network is illustrated in figure 2.1.

This network topology is easy to create, and can span large distances because the nodes will act as repeaters. But because it lacks redundancy, it is highly dependent on the other nodes in the network. If one node fails, this will split the network because there are no alternative routes through the network[6][7].

Figure 2.1: The figure shows a Line Topology

Bus

The nodes in a bus topology are connected to a shared medium. All nodes in the network will receive the data transmitted through that shared medium, but only the node that the data are meant for will accept the data. All other nodes, will in most cases discard the data[6][7]. Figure 2.2 shows the network topology.

If a node fails, it will not affect the network, but if a link fails it can split the network, because there is no alternative route through the network[6][7].

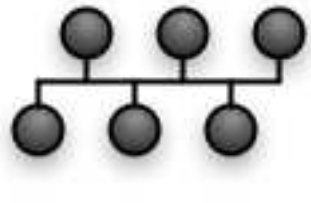

Figure 2.2: The figure shows a Bus topology

Ring

In a ring topology, all nodes are connected to two other nodes, this means that the network will create a logical ring, see figure 2.3. The ring topology is often the most expensive, and it tends to be inefficient because it have to travel through more nodes, then other topologies[6][7].

If a node fails it may impact other nodes, because in some implementations data are only transmitted one way through the network. It can then be considered as a line topology, with all its weaknesses. A way to solve this problem is to use a dual ring topology, where each node has four branches connected to it. This makes the topology more resistant to failures, but it also increases the cost[6][7].

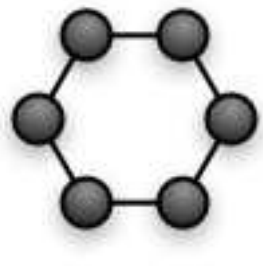

Figure 2.3: The figure shows a Ring topology.

Tree

In a tree network topology the nodes are arranged as a tree, see figure 2.4. The nodes connected as leaves act exactly as they would have been connected to a ring or bus topology. The nodes connected as non-leaf also act as they would have been connected

to a ring or bus topology, but they have several network cards, and will connect other leaves. It's important to note that no routing are done at the non-leaf nodes, they only relay data from their input to their output, like any other node[6][7].

If a link to a leaf or the node itself fails, it will only isolate the leaf node. The rest of the network will be unharmed. But if a non-leaf node fails, an entire section of the network will become isolated[6][7].

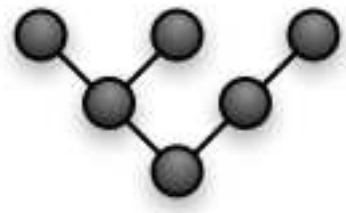

Figure 2.4: The figure shows a Tree topology.

Star

The star topology is a network topology where each node is connected to a central node. This central node retransmits all data received to all the other nodes connected to the central node[6][7]. See figure 2.5 for a graphical representation of the star network topology.

If a connection between a node and the central node is broken, this will only lead to the isolation of that node from the rest of the network. But if the central node is broken, the entire network will fail, and leave all the nodes isolated[6][7].

In local area network, this is the most common network topology, since it requires the least amount of transmission medium, and allows the network to be very adaptive. Today the central node has a very high reliability, and network redundancy is possible with the help of spanning tree algorithms[6].

Figure 2.5: The figure shows a Star topology.

Mesh

A mesh network topology has at least two nodes, with one or more paths to the other nodes. Figure 2.6 shows a mesh network topology. The mesh topology is the topology that most wide area networks use, such as the Internet[6].

The mesh topology is a compromise between a fully connected network topology and the star network topology. With a high reliability and a low link connection cost, the mesh topology is the economical preferred choice for wide area network topologies[6].

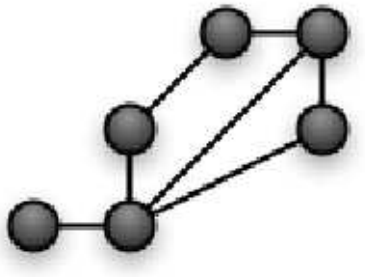

Figure 2.6: The figure shows a Mesh topology.

Fully Connected

A fully connected topology has direct links between all the nodes in the network, see figure 2.7. This is the most redundant and therefore the most reliable network, but it's also the most expensive because the direct links cost a lot of money to create and maintain[6].

A fully connected topology is also called a complete topology.

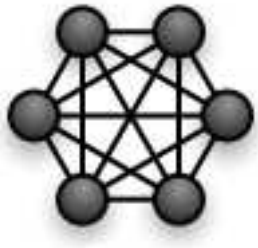

Figure 2.7: The figure shows a fully connected topology.

2.1.3 Network Transmission Media

The wired cables are the infrastructure of the network is the means for transmitting the data. Although they have the same purpose, they have different qualities and properties.

Twisted Pair Cable

The Twisted Pair cable is the most commonly used electrical cable. It is constructed by twisting two cables around each other, this will reduce the crosstalk. The more the cable is twisted, the more the crosstalk is reduced[1][3].

The unshielded twisted pair cable (UTP), is used in most telephone and computer networks. The UTP cables are standardized into several categories, which indicate signal integrity attributes. Category 5 cable is used in most Ethernet networks, but category 6 or 7 are probably the ones used in the 10-gigabit Ethernet standard[1][3].

The shielded twisted pair cable (STP) has a shield around the core which helps to protect against outside interference. Shielded Twisted Pair cable is used in token ring networks[1][3].

Coaxial Cable

The Coaxial Cable, commonly known as coax, is an electrical cable consisting of a round, insulated conducting wire, surrounded by an insulating spacer, surrounded by a cylindrical conducting sheath, usually surrounded by a final insulating layer[1][3].

Several versions of the coaxial cable are available. In networking the thick (0.5 inch diameter) and thin (0.25 inch diameter) coaxial cable are the most commonly used[1][3].

Optical Fibre Cable

Optical Fibre is a transparent thin fibre for transmitting light. It is made of glass or plastic, and because it is not affected by electromagnetic interference, it can operate on data rates well in excess of those possible with twisted-pair or coaxial cable[1][3].

2.1.4 Network Medium Access Control

The network infrastructure is created by a transmission media and a topology. But to utilize the infrastructure, protocols with standards and algorithms are necessary. Some of the most important medium access control standards are described below.

Ethernet

Ethernet was originally a frame-based computer networking technology for local area networks (LANs). The standard defines wiring and signaling for the physical layer, and frame formats and protocols for the media access control (MAC)/data link layer of the OSI model. The Ethernet is mostly standardized as IEEE's 802.3, and it carries traffic at the nominal rate of 10 Megabit per second (Mbit/s)[1][2].

Fast Ethernet which carry traffic at the nominal rate of 100 Mbit/s, and the Gigabit Ethernet which carry traffic at the nominal rate of 1 Gigabit per second (Gbit/s), are term describing various technologies for implementing Ethernet networking at 100 Mbit/s and 1 Gbit/s [1][2].

The Ethernet and Fast Ethernet standard supports only twisted pair cable, while Gigabit Ethernet supports both optical fibre and twisted pair cable[1][2].

The new 10-gigabit Ethernet standard encompasses seven different media types for LAN, MAN and WAN. It is currently specified by a supplementary standard, IEEE 802.3ae, and will be incorporated into a future revision of the IEEE 802.3 standard[1][2].

Unlike earlier Ethernet systems, 10-gigabit Ethernet is so far based entirely on the use of optical fibre connections. However a standard for 10-gigabit Ethernet over

twisted pairs, using Cat-6 or Cat-7 cable and planned for approval in 2006. Additionally, this developing standard is moving away from local area network design, with broadcasting to all nodes, towards a system which includes some elements of wide area routing. It is claimed that this system has high compatibility with earlier Ethernet and IEEE 802 networks[2].

The Ethernet, Fast Ethernet, Gigabit Ethernet and the upcoming 10-gigabit Ethernet has largely replaced all other LAN standards such as token ring, FDDI, and ARCNET[1][2].

Fibre-distributed data interface (FDDI)

Fibre-distributed data interface (FDDI) is a standard for data transmission in a local area network that can extend in range up to 200 km using a topology that is a dual-attached, counter-rotating token ring. In addition to being large geographically, an FDDI local area network can support thousands of users. The underlying medium is optical fibre, though it can be copper cable, in which case it may be called CDDI[6].

Point-to-Point Protocol (PPP)

The Point-to-Point Protocol is commonly used to establish a direct connection between two nodes. Its primary use has been to connect computers using a phone line, though it is also occasionally used over broadband connections. Many ISPs use PPP when providing customers with dial-up access[6].

Asynchronous Transfer Mode (ATM)

Asynchronous Transfer Mode is a cell relay network protocol which encodes data traffic into small fixed sized (53 byte) cells instead of variable sized packets as in packet-switched networks (such as the Internet Protocol or Ethernet)[1][2][6].

ATM provides a highly complex technology, with features intended for applications ranging from global telecommunication networks to private local area computer networks. ATM has been a partial success as a technology, with widespread deployment, but generally only used as a transport for IP traffic[2][6].

It's goal of providing a single integrated technology for LANs, public networks, and user services has largely failed.

2.1.5 Network Protocols

In a typical network, there are several protocols in use. The most known protocol is the Internet protocol suite, which are the building blocks of the Internet. But there are several important protocols both over and under these protocols. Examples are HTTP, FTP, SSH, MAC, ARP, FDDI, MPLS, etc.

Internet protocol suite

The Internet protocol suite is the set of communications protocols that implement the protocol stack on which the Internet runs. It is sometimes called the TCP/IP protocol

suite, after the two most important protocols in it: the Transmission Control Protocol (TCP) and the Internet Protocol (IP), which were also the first two defined[5][6][8][9].

Internet Protocol (IP)

The Internet Protocol (IP) is a data-oriented protocol used by source and destination hosts for communicating data across a packet-switched internetwork[5][9].

Data in an IP internetwork are sent in blocks referred to as packets or datagram's (the terms are basically synonymous in IP). In particular, in IP no setup is needed before a host tries to send packets to a host it has previously not communicated with[5][9].

The Internet Protocol provides an unreliable datagram service (also called best effort); i.e. it makes almost no guarantees about the packet. The packet may arrive damaged, it may be out of order (compared to other packets sent between the same hosts), it may be duplicated, or it may be dropped entirely. If an application needs reliability, it is provided by other means[5][6][8][9].

Transmission Control Protocol (TCP)

Transmission Control Protocol (TCP) is a connection-oriented, reliable delivery byte-stream transport layer communication protocol. It does the task of the transport layer in the simplified OSI model of computer networks[5][6][9].

In the Internet protocol suite, TCP is the intermediate layer between the Internet Protocol below it, and an application above it. Applications most often need reliable pipe-like connections to each other, whereas the Internet Protocol does not provide such streams, but rather only unreliable packets[5][6][9].

TCP connections contain three phases: connection establishment, data transfer and connection termination. A 3-way handshake is used to establish a connection. A four-way handshake is used to tear-down a connection. During connection establishment, parameters such as sequence numbers are initialized to help ensure ordered delivery and robustness[5][6][9].

User Datagram Protocol (UDP)

The User Datagram Protocol (UDP) is a minimal message-oriented transport layer protocol. In the TCP/IP model, UDP provides a very simple interface between a network layer below and an application layer above. UDP provides no guarantees for message delivery and a UDP sender retains no state on UDP messages once sent onto the network. UDP adds only application multiplexing and data checksumming on top of an IP datagram[5][6][9].

Internet Control Message Protocol (ICMP)

The Internet Control Message Protocol (ICMP) is a set of protocols used by networked nodes to send control data to the network[5][6][9].

ICMP differs in purpose from TCP and UDP in that it is usually not used directly by user network applications. One exception is the ping tool, which sends ICMP Echo Request messages (and receives Echo Response messages) to determine whether a host is reachable and how long packets take to get to and from that host[5][6][9].

2.2 Computer Security

To create a secure computer system, three properties are necessary[10]:

- Confidentiality
- Integrity
- Availability

2.2.1 Confidentiality

Confidentiality is about keeping data unavailable for non-authenticated users. This can be achieved by access control and encryption[1][10].

Access Control

Access control is about controlling who has access to specific resources. In a operating system functions are available to provide access control, but if these functions are bypassed or fail, the data are fully compromised[1][10].

Cryptography

Cryptography or encryption is a method to descramble the data so that it is only readable for authenticated users. Cryptographic methods can be used on data located in a operating system, or data transmitted through a insecure network[1][10].

2.2.2 Integrity

Integrity is about knowing that the data has not been changed by non-authenticated users[1][10].

Data Integrity

Data integrity is methods for controlling that the content of the data has not been changed. This can be achieved by using cryptographic message digest algorithms such as SHA-1 or MD5[10].

Origin Integrity

Origin integrity or authentication are methods for controlling the identity of the source (entity) of the data[10].

Information that can confirm the identity of the entity are[10]:

1. What the entity knows (such as passwords or secret information)
2. What the entity has (such as a badge or card)
3. What the entity is (such as fingerprints or retinal characteristics)
4. Where the entity is (such as in front of a particular terminal)

One or several of these information sources may be used to confirm the identity of the entity. Usually the security level defines how many of the information sources that are used[10].

2.2.3 Availability

Availability is about keeping the data accessable for authenticated users. Denial of service attacks are one of the methods for probitting availability[3][10].

Methods for ensuring high availability are by using cluster technologies like fail over clusters, that automatically changes to a secondary server if a error is detected on the primary server, or performance servers that share the load, so that there are enough resources to handle the load on the servers[3][10].

2.3 Measurements

Measurements are conducted in four stages[8]:

1. Data collection
2. Analysis
3. Presentation
4. Interpretation

2.3.1 Data Collection

The first stage collects the raw data from the network or computer. This can be done by active measurement, which are tool that generate traffic on the network to cunduct the measurements. Another name for active measurements are benchmarking. Another approach is passive measurements, which are contorary to active measurements in that they only monitor the network[8].

2.3.2 Analysis

In stage two, the raw data are processed in different ways to gather usefull information about the measurements[8]. Interesting data can be: minimun value, maximum value, mean value, median value, etc.

The Maximum

The maximum sample is the sample with the highest value[11].

The Minimum

The minimum sample is the sample with the lowest value[11].

The Median

The median of a set of samples is the sample for which there are an equal number of samples with a lesser value and an equal number with a greater value[11].

The Mean

The mean of a set of samples is the same as the average value, which can be found by the following formula[12]:

$$\langle v \rangle = \frac{v_1 + v_2 ... v_N}{N} = \frac{1}{N} \sum_{i=1}^{N} v_i \tag{2.1}$$

where v_{1-N} is the observation values, and N is the number of observations.

The Standard Deviation

The standard deviation can be found by the following formula[12]:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=0}^{N} \Delta g_i^2} \tag{2.2}$$

where g_{1-N} are guessed error values, and N is the number of observations.

The guessed error values are deviations of the measured data, and are found by this formula[12]:

$$\begin{aligned} \Delta g_1 &= \langle v \rangle - v_1 \\ \Delta g_2 &= \langle v \rangle - v_2 \\ &... \\ \Delta g_N &= \langle v \rangle - v_N \end{aligned}$$

"The standard deviation show the scatter in the data due to random influences. σ is the root mean square (RMS) of the assumed errors"[12].

A typical use of the standard deviation is in error bars on figures. This helps interpreting measurements as they give a more true picture, showing that the measuremetns are affected by random interference[12].

2.3.3 Presentation

In stage three, the raw and the processed data are visualized by creating graphs or charts. The visual aid, can help clearifying thrends in the data[8][13].

Timeseries

A time series diagram shows the x-axis in time, and the y-axis as the measured values. Time series diagrams are usefull for describing the measured data, and spotting trends. An example of a time series diagram can be found in figure 2.8.

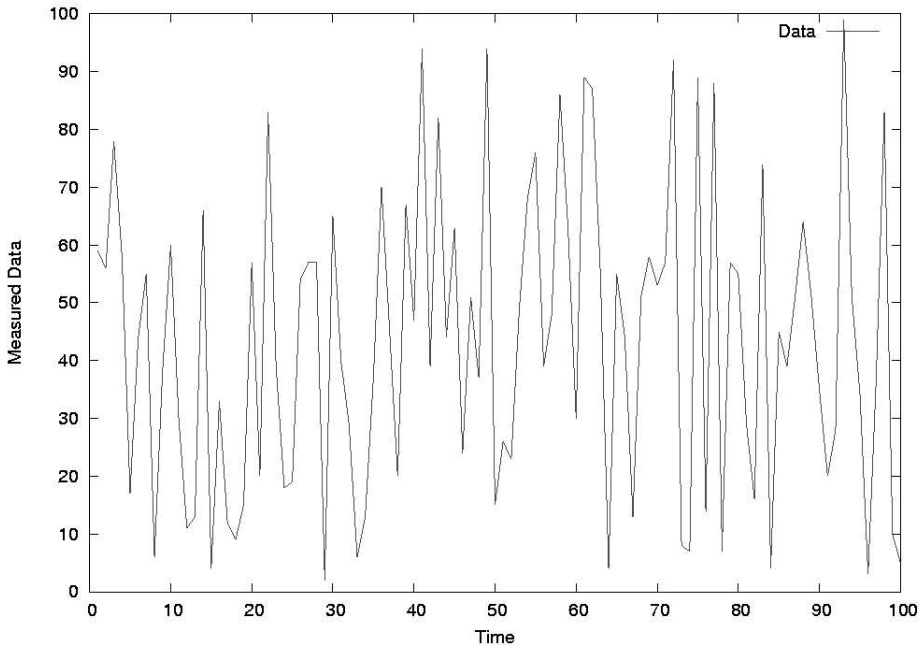

Figure 2.8: The figure shows a Timeseries diagram.

Histogram

A histogram diagram shows the x-axis as ranges of the measured data, and the y-axis as the frequency of the measured values within these ranges. Histograms are usefull for describing the distribution of the measured data. An example of a histogram diagram can be found in figure 2.9.

Phaseplot

A phaseplot diagram shows one of the axes as the measured data (i), and the other axes as the next value of the measured data (i+1). This is an efficient way of seeing if there are corrolations between the following value, or if it is completly random. An example of a phaseplot diagram can be found in figure 2.10.

2.3.4 Interpretation

The last stage makes use of the three other stages, and interpretates the data measured. This is usually the goal with the measurements, and typically require imagination and the necessarry skills within the field[8].

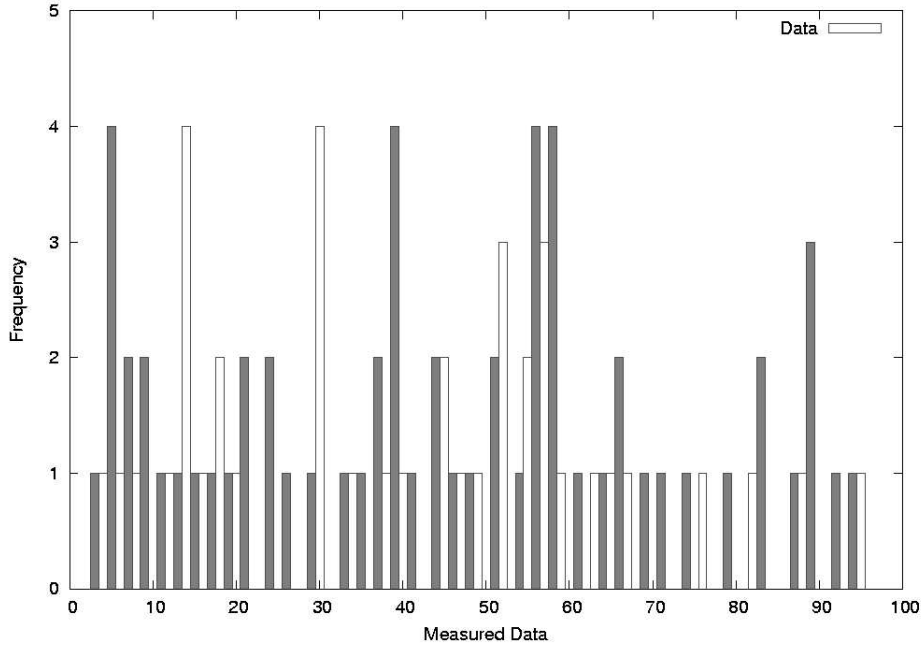

Figure 2.9: The figure shows a Histogram diagram.

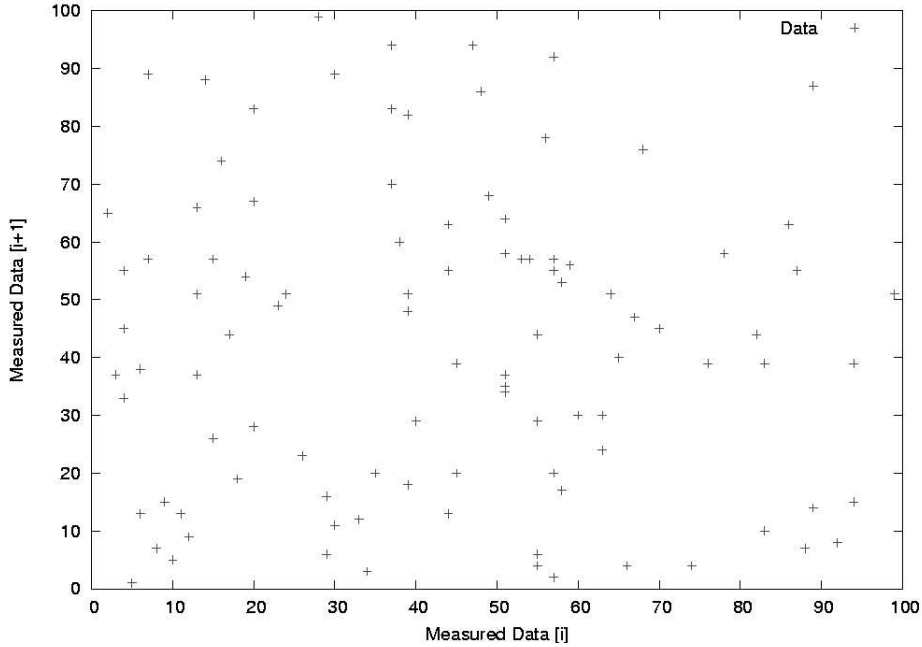

Figure 2.10: The figure shows a Phaseplot diagram.

Chapter 3

Literature Survey

3.1 Quality of Service

The stream of packet between two nodes in a network is called a flow. This flow will in a connection-oriented network follow the same route, but in a connectionless network, the packet may take different routes[1][14][15].

The problem with a connectionless network is that the routes may have different properties. The four main properties for a network connections are[1][15]:

1. Bandwidth
2. Delay
3. Jitter
4. Reliability

These four properties define the quality of service (QoS), that the flow requires. The QoS for the routes may not matter for some applications, but it may be crucial for others. Table 3.1 shows the stringency of several common applications[1].

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: How stringent the quality-of-service requirements are.

From the table, the following interpretation can be made[1]:

- The four first applications require a high reliability. This means that all bits have to be transmitted correctly. This is usually achieved by checksumming each packet and verifying that the checksum matches at both ends. If the packet is damaged, it will be retransmitted.
- The four last applications can tolerate errors, and will not require checksumming or verification.
- Applications like telephony and videoconferencing require a low delay, and are therefore highly dependable on the delay. These are typical real-time applications, and real-time has strict delay requirements.
- Applications like e-mail and file transfer, are more delay tolerant, as these are typical "store and use" applications.
- Web access and remote login applications are interactive programs that require a relative low delay.
- For real-time applications like the last four applications, a low delay between the packages are important. A burst of packets may become very uncomfortable for these real-time applications.
- The other four applications are more immune for jitter, as buffers can be used to smooth the connection.
- Applications like e-mail, remote login or telephony requires a low bandwidth.
- Applications with graphical data and/or sound data requires a higher bandwidth.

By creating a service level agreement (SLA) with the internet service provider (ISP) about the property of the internet connection, the ISP can guarantee that the quality of service is good, as long as the customer obeys the agreement[1].

3.1.1 Providing Quality of Service

Overprovisioning

One of the easiest methods for providing quality of service is to overprovision all the network equipment. By providing much router capacity, buffer space and bandwidth, there will be no congestion, and therefore the quality of service will be provided. The downside with this approach is that it is a very expensive method to solve the quality of service problem[1].

Buffering

By buffering the data on the receiving side, before the data is delivered, the delay will increase, but the jitter will smooth out. As jitter is the main problem for audio and video on demand applications, this technique may help[1].

For audio- and videoconferencing applications, and other real time applications, a high delay is not acceptable. A high delay feels uncomfortable for the user, and hence another solution has to be used for these types of applications.

Traffic Shaping

Traffic shaping is a technique used where the server and the client agrees on the shape of the traffic. So instead of sending bursts of traffic, which would create a unpredictable jitter and delay, the sending host knows how much data the client can accept, and transmits the data in a more uniform way[1].

Resource Reservation

By creating the path for the flow of the packets, during setup of a connection, and thus creating a virtual circuit. It is possible to reserve resources at all the routers on that path. This will ensure that the packages will arrive in order, with about the same delay, and so on[1].

The resources available for reservation at the routers are[1]:

1. Bandwidth
2. Buffer space
3. CPU cycles

When reserving bandwidth, a portion of the available bandwidth, is set aside for the flow. The reservation of buffer space is another resource of short supply. But reserving the buffer space will prevent dropping of packages. Packages are dropped if there are no available buffer space, and so they have to be retransmitted. This creates an even higher delay. The last resource available for reservation is CPU cycles. The router needs the CPU for processing which packet goes where. So when reserving CPU cycles for one flow, this will ensure that the flow will have a lower processing delay at each router[1].

Proportional Routing

The traditional approach for routing is to find the optimal route to reach the destination, and then send the packages to the next router on the route. This approach has its downside, in that it may congest the node[1].

An alternative approach would be to spread the load over multiple paths to the destination. To utilize the approach, local information at each router has to be available, as the routers generally has no overview over the network-wide traffic[1].

Packet Scheduling

As routers usually handle multiple flows of data, there is a need for scheduling the packets passing through the router. There are multiple scheduling algorithms available to solve the scheduling problem, but these usually provide methods for dividing the traffic in a fair way. A method for achieving quality of service is to schedule flows depending of the quality of service tag or a service level agreement[1].

3.1.2 Quality of Service Standards

IETF put a lot of effort into creating a architecture for streaming multimedia. They ended up with two different approaches.

Integrated Services

Integrated services, or flow-based quality of service, create the path through the network for the flow. This requires a setup for the connection, when the connection between two nodes is established[1][15][14].

The downside of this approach is that is does not scale well. Routers with thousands or millions of flows may crash or get congested because of the increased load of handling the different flows[1][15][14].

Differentiated Services

Differentiated services, or class-based quality of service, differentiate the quality of service dependable on the type of service field in the network protocol. The routers read the type of service field, a treat the packet according to the policy defined in the internet service providerŠs network[1][15][14].

The problem with this approach is that there is no common policy for the type of service field, and so when packages pass through different networks, the packets may be handled different then intended by the original sender[1][15][14].

3.2 Bandwidth

3.2.1 Theory

Bandwidth is a term used to describe the capacity of a link. It is the transmission rate for the link. A link able to transmit at 100 Mbps, has a bandwidth of 100 Mbps[16]. Table 3.2 lists some of the typical bandwidths provided by the most common medium access control technologies. Table 3.3 lists categories of wide area network connections provided by internet service providers[1].

Even though bandwidth is what is provided by the internet service provider, it is the throughput of the connection that is of interest for the customer. "Throughput is a measure of the amount of data that can be sent over a link in a given amount of time[16]".

The throughput is determined by the formula:

$$Throughput = \frac{Data\ Transferred}{Time} \tag{3.1}$$

The throughput is expressed in bits per second or packets per second. But when expressed in bits per second, the more the typical expression is kilobits (10^3 bits), megabits (10^6 bits) or gigabits (10^9 bits) per second, depending on the connection throughput.

The difference between throughput and bandwidth is that throughput measurements may be affected by considerable overhead that is not included in bandwidth

measurements. And therefore throughput is a more realistic estimator of the actual performance for the connection[16].

Description	Bits	Bytes
Ethernet (10base-X)	10 Mb/s	1,25 MB/s
Fast Ethernet (100base-X)	100 Mb/s	12,5 MB/s
FDDI	100 Mb/s	12,5 MB/s
Gigabit Ethernet (1000base-X)	1.000 Mb/s	125 MB/s

Table 3.2: Bandwidths provided in Local Area Networks

SONET		SDH	Data Rate (Mbps)		
Electrical	Optical	Optical	Gross	SPE	User
STS-1	OC-1		51,84	50,112	49,536
STS-3	OC-3	STM-1	155,52	150,336	148,608
STS-9	OC-9	STM-3	466,56	451,008	445,824
STS-12	OC-12	STM-4	622,08	601,344	594,432
STS-18	OC-18	STM-6	933,12	902,016	891,648
STS-24	OC-24	STM-8	1244,16	1202,688	1188,864
STS-36	OC-36	STM-12	1866,24	1804,032	1783,296
STS-48	OC-48	STM-16	2488,32	2405,376	2377,728
STS-192	OC-192	STM-64	9953,28	9621,504	9510,912

Table 3.3: Bandwidth technologies in Wide Area Networks

3.2.2 Data Collection

The two factors affecting the throughput are, the amount of data transferred, and the time it took to transfer that data.

Determining the throughput can be done in two ways:

1. Measuring the time it takes to transfer a predetermined amount of data.

2. Measuring the amount of data transferred in a predetermined amount of time.

Active Measurements

A simple method for actively measuring the throughput is to upload or download a file through ftp. This gives information about the file size and the time it took to transfer the file. The problem with this simple measurement approach is that the disk access needed to store or read the file, may interfere with the measurement[16].

Programs like `netperf`[17], `iperf`[18] and `ttcp`[19] use methods so that no disk access is necessary. This is done by reading and writing the transmitted data into the RAM. All of these programs use the same functionality for measuring the throughput, but they differ in functionality[16].

Of the three example programs here, `iperf` has the most functions. And can not only measure TCP throughput, but also UDP throughput, jitter and packet loss.

Pasive Measurements

Passive measurements do not add extra data to the network, but rather measures the current throughput, through a node. To capture the data flow, through the node, tools like `tcpdump`[20] must be used. These tools capture the data on a kernel level, and thus provide the raw data needed to perform analysis on the data. Other tools must then be used to analyze the data, and thus provide the throughput.

Tools like `tcpstat`[21] can be used to analyze the data captured by `tcpdump`, but `tcpstat` can also capture the data itself. The advantage with `tcpstat` monitoring the bandwidth itself is that there is no need to store the captured data, which is the case with `tcpdump`[16].

Both `tcpdump` and `tcpstat` can be used with filters, which filter away unwanted data. This can be useful for monitoring only upload traffic, download traffic, http traffic, ftp traffic, etc.

3.2.3 Analysis

The throughput measurements provide useful statistical information about the throughput of the node. If the node has used active measurements, the measurements show the throughput for the connection, while passive measurement tools show the utilization of the bandwidth.

Trends are identified by presenting the measured data in a time series diagram, where the time is for a long duration of time. The longer the duration, the easier the trend may be to recognize.

The throughput distribution can be viewed in a histogram diagram.

3.3 Delay

3.3.1 Theory

The delay is the time it takes to send a packet or frame from a source node to a destination node. The delay is the product of thee delays, these are called[2][16][22]:

- Transmission delay
- Propagation delay
- Queuing delay

Transmission delay is the amount of time it takes to put the signal onto the cable. This depends on the transmission rate (or interface speed) and the size of the frame[16].

Propagation delay is the amount of time it takes for the signal to travel across the cable. This depends on the type of media used and the distance involved[16].

Queuing delay is the time it takes to process the packet or frame at intermediate devices such as routers and switches. It is called the queuing delay because most of time is spent in queues within the device[16].

The transmission delay and the propagation delay are quite predictable and stable delays, but queuing delays can introduce considerable variability[2][16].

The delay can be calculated by the following method:

$$\begin{aligned} d_1 &= t_1 + p_1 + q_1 \\ d_2 &= t_2 + p_2 + q_2 \\ &... \\ d_N &= t_N + p_N + q_N \end{aligned}$$

where d_{1-N} is the delay for each node on the path, t_{1-N}, is the transmission delay between each of the nodes on the path, p_{1-N} is the propagation delay between each of the nodes on the path, and q_{1-N} is the queuing delay for each node on the path.

The total delay is then:

$$Delay = \sum_{i=1}^{N} d_i \tag{3.2}$$

where d_{1-N} is the delay for each node on the path, and N is the number of nodes on the path to the destination.

The delay is expressed in time, and since the delay usually is quite small, it is expressed in ms.

3.3.2 Data Collection

To measure the delay, it is necessary to send a packet to a destination node, and somehow measure the delay between the nodes. The ICMP are designed to handle these kinds of operations, but this may create an unreliable result as the ICMP packets are usually prioritized compared to the IP packets.

There are two methods for measuring the delay between two nodes:

- One way delay
- Round trip delay

The one way delay is the time it takes the packet or frame to travel from the source to the destination node. The problem with measuring this value is the time synchronization needed to get a reliable result. If the clock is not perfectly synchronized for both hosts, the timestamps provide a false result. Time synchronization is provided through the network time protocol (NTP) and GPS. Tools like `sting`[23] provides one way delay measurements.

The round trip delay is the interval between the time a program sends a packet to a destination node, and the time for when an acknowledgment packet was received from the destination node. Tools like `ping`[24], `RTTometer`[25], `pinger`[26], and `smokeping`[27] all provide the possibility of measuring the round trip time.

Of the two measurement methods, round trip delay are the most useful for most applications, because there is an interaction between the two communicating hosts.

3.3.3 Analysis

The delay measurements can provide information as statistics of the connection that has been monitored. This can provide useful, when troubleshooting applications. By presenting the measurements in diagrams, the following information can be identified:

- Trends for the round trip time, for the measured connection. The trends can be viewed in a time series diagram.
- The distribution of the round trip time. This can be viewed in a histogram diagram.
- The congestion of the connection. Congestion is determined by a phase plot diagram.

In a phase-plot of a given measurement period there are three congestion regions as shown in figure 3.1[22]:

- Region I contains probe pairs that see empty queues and experience minimum RTT plus minor random overheads.
- Region III contains probe pairs that always see a queue. This is the region of persistent congestion.
- Region II contains probe pairs where one of the probes experiences queuing delay but the other does not, i.e., there is a transition in congestion state between adjacent probes. This is the region of transient congestion.

3.4 Jitter

3.4.1 Theory

Jitter is the variation in arrival times of successive packet from a source to a destination. And is determined by the difference experienced by subsequent packets, RTT_I and RTT_{I+1}[2][16][22]. The mathematical formula is:

$$Jitter_I = \sqrt{\frac{1}{2}(RTT_I - RTT_{I+1})} \tag{3.3}$$

3.4.2 Data Collection

The jitter can be measured by monitoring the round trip time for packets between two nodes. This can be done by passive measurement tools like `tcpdump`, that taps into the network and stores the relevant data. Other tools can then extract information from `tcpdump` which can then be analyzed.

Active measurement tools include all of those used to measure the round trip time.

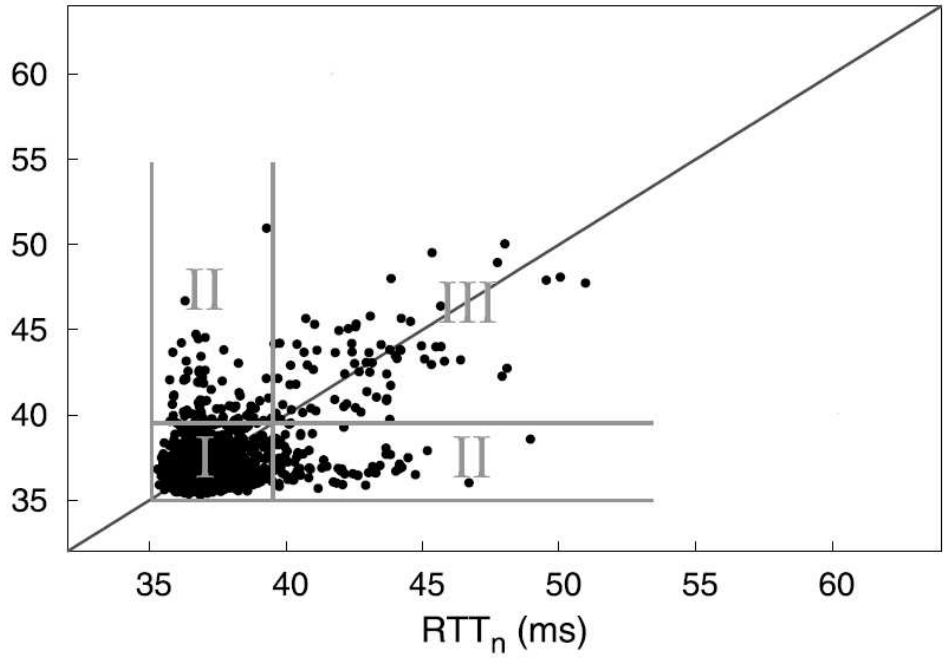

Figure 3.1: RTT in a phaseplot diagram.

3.4.3 Analysis

Showing the jitter in a time series diagram, shows the jitter during that time period, but reveals little information about the jitter itself.

More interesting information about the jitter comes from the distribution of the jitter. This can be viewed in a histogram chart. When the distribution of the jitter is mainly within a few seconds, the jitter can be qualified as low, but this depends on the requirement of the application. The desired distribution would be an exponential distribution. If the distribution is not within a few seconds, but rather spread across several second, the connection is unpredictable and has a high jitter value.

3.5 Reliability

3.5.1 Theory

Reliability is defined as "An attribute of any system that consistently produces the same results, preferably meeting or exceeding its specifications"[28].

A method to describe reliability is to use the failure rate, which describes how frequently something fails. A failure in network is when the packet does not reach its destination, before the time expire[29][30].

The failure rate (λ) has been defined as "The total number of failures within an item population, divided by the total time expended by that population, during a particular measurement interval under stated conditions. (MacDiarmid, et al.)". It has also been defined mathematically as the probability that a failure per unit time occurs

in a specified interval, which is often written in terms of the reliability function, $R(t)$, as,

$$\lambda = \frac{R(t_1) - R(t_2)}{(t_2 - t_1)R(t_1)} \tag{3.4}$$

where, t_1 and t_2 are the beginning and ending of a specified interval of time, and $R(t)$ is the reliability function, i.e. probability of no failure before time t.

The failure rate data can be obtained in several ways. The most common methods are[30]:

- Historical data about the device or system under consideration.
- Government and commercial failure rate data.
- Testing.

Historical data can be provided by the companies that produce the device or system. This can be used to produce the failure rates. Another approach is to use failure rate data provided by government or commercial companies. The last approach is to monitor and test the devices or system to generate failure data[30].

When monitoring a network connection or a node, packet loss is a measurement for measuring the fraction of packets sent from a measurement node to a destination node for which the measurement node does not receive an acknowledgment from the destination node[30].

3.5.2 Data Collection

The active measurement tools used for measuring the delay and jitter, are also suitable for measuring the failure rate.

3.5.3 Analysis

The interesting information gathered from the raw data conserving the reliability, is how many error there where during the measurement period. This is known as the error rate. It shows the number of errors divided on the time interval.

To retrieve reliable failure rate data, the testing should be performed over a relative large period of time. This removes uncertainty in the result.

Chapter 4

Methods

The state of a network link can be determined by measuring the throughput, the delay, the jitter and the packet loss. These four properties represent the quality of the link. In the following three case studies, methods for measuring these, and more properties will be shown.

4.1 Case One: Network Traffic

4.1.1 Motivation

By monitoring the network traffic for a node, information about the state of that node can be determined. This may provide the network administrator, with enough information to optimize the system performance, by removing bottlenecks. The system can represent the node, a subnet, or the whole network.

There are especially two locations that are of interest, when performing passive network measurements:

- The state of a service host, that provides a network service. Examples of network services are the DNS, DHCP, HTTP, and FTP services.

- The state of a network node, performing a routing or forwarding functions. Examples of such nodes are firewalls, virtual private networks (VPN), and routers.

4.1.2 Objective

By using a passive network measurement tool, two nodes are to be monitored for one day. The data gathered from these nodes are to be analyzed, and the state of the nodes is of interest.

4.1.3 Resources

In this experiment, the resources located in table 4.1 has been utilized.

Description	Node One	Node Two
Processor Model	Intel Pentium III	AMD Athlon(tm) XP
Processor Mhz	549,947Mhz	1852,314Mhz
Memory	640 MB	1024 MB
Network MAC	Fast Ethernet FD	Ethernet FD
Network Link	100 Mb/s	10 Mb/s
Internet Service Provider	UNINETT	Bredbandsbolaget
IP Address	128.39.73.19	83.227.111.133

Table 4.1: The resources utilized in Case One.

Description of Node One

Node One is a host located in a test lab at Oslo University College, in Oslo Norway. The operating system is GNU Linux Debian, where the version of Debian is "Sarge". The host is also running the following services: PostgreSQL, SMTP, HTTP, HTTPS, FTP, and SSH.

The host is connected to a local area network, which shares a 155Mb/s internet connection with the rest of the school. But as the local area network, and the host network interface card only supports Fast Ethernet, the maximum network speed is about 100Mb/s.

Description of Node Two

Node Two is a host serving as a firewall for a local area network, located in Oslo Norway. The operating system is GNU Linux Debian, where the version of Debian is "Woody". The host serves the firewall, which protects a local area network with services like HTTP, FTP, and SSH.

The host is connected to an Ethernet connection, providing 10 Mb/s internet connection. The ISP has a direct connection from Norway to Bredbandsbolaget's backbone network, which is located in Sweden.

4.1.4 Tools

To perform the passive measurements, a program has to run on the node that is monitored. The SNMP service is an alternative method for collecting the measurements. But it may also generate traffic on the network, if the collecting node is located on the monitored network. Another approach would have been to log all network traffic, with the help of tcpdump or an equitant program, and then later process the saved data. But this generates a lot of data, requires a lot of disk space, and it lacks the function to measure the state of the node.

A more suited program for the measurements conducted in this experiment, where `tcpstat`. `tcpstat` is a highly configurable program that measures some data, and may generate some statistics if wanted. Examples of the data that can be gathered are: bits per second, bytes since last measurement, ARP packets since last measurement, TCP packets since last measurement, ICMP packets since last measurement, etc.

The following command was executed on both nodes, and ran on the nodes for one day.

```
tcpstat -i eth0 \
         -o "%S %A %C %V %I %T %U %a %d %b %p %n %N %l \n"
```

The explanation of logged data can be found in the result chapter.

4.1.5 Predictions

The predictions for the result are:

- Node One, will never fully utilize the available bandwidth, but will probably utilize 100% of the processing power.
- Node Two, will fully utilize the bandwidth, but will probably not utilize the processing power.
- For both nodes, IP will dominate the network layer protocols, and TCP will dominate the transport layer protocols.

4.2 Case Two: Throughput

4.2.1 Motivation

By performing active measurements from one node to another node with equal link speed, the state of the connection can be determined. If the link speed is not as expected, countermeasures can be taken to locate and remove the bottleneck.

4.2.2 Objective

By using a active measurement tool, the connection between two nodes are to be benchmarked and analyzed. The test should provide enough information to see trends in the network, and determine if the node manage to utilize the available bandwidth. To remove uncertainties in the results, benchmarking should be executed from one node, to two other nodes.

4.2.3 Resources

In this experiment, the resources located in table 4.2 has been utilized.

Description of Node One

Node One is a host located in a test lab at Oslo University College, in Oslo Norway. The operating system is GNU Linux Debian, where the version of Debian is "Sarge". The host is also running the following services: PostgreSQL, SMTP, HTTP, HTTPS, FTP, and SSH.

The host is connected to a local area network, which shares a 155Mb/s internet connection with the rest of the school. But as the local area network, and the host network interface card only supports Fast Ethernet, the maximum network speed is about 100Mb/s.

Description	Node One	Node Two	Node Three
Processor Model	Intel P III	Intel P MMX	Intel P III
Processor Mhz	549,947Mhz	167,047Mhz	447,699Mhz
Memory	640 MB	96 MB	923 MB
Network MAC	Fast Ethernet FD	Fast Ethernet FD	Fast Ethernet FD
Network Link	100Mb/s	100Mb/s	100Mb/s
Internet Service Provider	UNINETT	UNINETT	UNINETT
IP Address	128.39.73.19	158.38.88.147	128.39.74.16

Table 4.2: The resources utilized in Case Two.

Description of Node Two

Node Two is a host located in the student housings at Molde University College, in Molde Norway. The operating system is GNU Linux Red Hat, where the version of Red Hat is "9.0".

The host is connected to a local area network, which shares a 155Mb/s internet connection with the rest of the school. But as the local area network, and the host network interface card only supports Fast Ethernet, the maximum network speed is about 100Mb/s.

The path from Node One to Node Two is shown in table 4.3.

Path ID	IP	Location
01	128.39.73.1	Oslo, Norway
02	158.36.84.21	Oslo, Norway
03	128.39.0.73	Oslo, Norway
04	128.39.46.249	Oslo, Norway
05	128.39.46.2	Trondheim, Norway
06	128.39.46.102	Trondheim, Norway
07	128.39.47.102	Ålesund, Norway
08	128.39.47.130	Molde, Norway
09	158.38.0.66	Molde, Norway
10	158.38.88.147	Molde, Norway

Table 4.3: Path from Node One to Node Two.

Description of Node Three

Node Three is a host located in the student network at Oslo University College, in Oslo Norway. The operating system is GNU Linux Debian, where the version of Debian is "Woody". The host is also running the following services: MYSQL, NTP, SMTP, HTTP, FTP, and SSH.

The host is connected to a local area network, which shares a 155Mb/s internet connection with the rest of the school. But as the local area network, and the host network interface card only supports Fast Ethernet, the maximum network speed is about 100Mb/s.

The path from Node One to node thee is shown in table 4.4.

Path ID	IP	Location
01	128.39.73.1	Oslo, Norway
02	128.39.74.16	Oslo, Norway

Table 4.4: Path from Node One to Node Three.

4.2.4 Tools

There are multiple tools that perform about the same function when performing active measurements. Known network throughput benchmarking tools are: `netperf`, `iperf`, `ttcp`, and ftp.

The tool chosen to test the throughput is `netperf`. To execute the experiment, a server node and a client node has to be installed on each of the nodes.

For the experiment, the server program was installed on Node Two and three, and the client software was installed on Node One. This setup was chosen, so that the process of benchmarking could be controlled from Node One. This minimizes the probability for interference from each measurement.

4.2.5 Predictions

The predictions for the result are as follows:

- As all nodes are attached to a overdimentioned network, the connection itself should not be a problem. And it should be possible to achieve full link utilization.
- Node Two could have a problem to achieve 100 Mb/s as the processing power is a bit low.
- All the nodes are connected to a school network. This will probably mean that the link has the highest load during the day. This is why there is a higher chance to achieve full link utilization during the night or weekends.

4.3 Case Three: Delay, Jitter and Packet Loss

4.3.1 Motivation

By using a active measurement tool that measures the delay between two nodes, the jitter and packet loss can be determined by using mathematical methods.

The delay can be measured as the time it takes for one packet to be sent from a host, until it is received at the destination. But as this requires that the clocks are perfectly synchronized, an alternative method is mostly used. This is to measure the delay in form of the round trip time.

The round trip time is measured as the time it takes for one packet to be sent from a node to a destination node, until another packet is received from the destination node.

4.3.2 Objective

The previous cases showed methods for measuring the throughput for the link. In this last case study, the delay of a network link is to be measured, and based on those measurements, the jitter and packet loss is to be determined.

The round trip time, from one node to three other nodes are to be measured for one week. This should provide enough information to make reasonable decisions about the link state.

4.3.3 Resources

In this experiment, the resources located in table 4.5 has been utilized.

Description	**Node One**
Processor Model	Intel Pentium III
Processor Mhz	549,947Mhz
Memory	640 MB
Network MAC	Fast Ethernet FD
Network Link	100 Mb/s
Internet Service Provider	UNINETT
IP Address	128.39.73.19

Table 4.5: The resources utilized in Case Three.

In addition, the link and processing power of three remote nodes has been utilized. As the active measurements does not require any installation or configuration of the destination nodes, the hardware configuration of Node Three and four are not know. The hardware configuration of Node Two can be found in table 4.2 (Node Two).

Description of Node One

Node One is a host located in a testlab at Oslo University College, in Oslo Norway. The operating system is GNU Linux Debian, where the version of Debian is "Sarge". The host is also running the following services: PostgreSQL, SMTP, HTTP, HTTPS, FTP, and SSH.

The host is connected to a local area network, which shares a 155Mb/s internet connection with the rest of the school. But as the local area network, and the host network interface card only supports fast ethernet, the maximum network speed is about 100Mb/s.

Description of Node Two

Node Two is a host located in the student housings at Molde University College, in Molde Norway. The operating system is GNU Linux Red Hat, where the version of Red Hat is "9.0".

The host is connected to a local area network, which shares a 155Mb/s internet connection with the rest of the school. But as the local area network, and the host

network interface card only supports Fast Ethernet, the maximum network speed is about 100Mb/s.

The path from Node One to Node Two is shown in table 4.3.

Description of Node Three

Node Three is a node that is a part of a cluster that serves the "www.vg.no" domain. This domain belongs to a Norwegian news paper called "Verdens Gang".

The path from Node One to Node Three can be view in table 4.6.

Path ID	IP	Location
01	128.39.73.1	Oslo, Norway
02	158.36.84.21	Oslo, Norway
03	128.39.0.73	Oslo, Norway
04	193.156.120.3	Oslo, Norway
05	193.75.3.6	Oslo, Norway
06	193.75.3.1	Oslo, Norway
07	193.69.165.11	Oslo, Norway
08	193.69.165.11	Oslo, Norway

Table 4.6: Path from Node One to Node Three.

Description of Node Four

Node Four is a node that is a part of a cluster that serves the "www.kernel.org" domain. This domain belongs to the official GNU Linux kernel.

The path from Node One to node four can be view in table 4.7.

Path ID	IP	Location
01	128.39.73.1	Oslo, Norway
02	158.36.84.21	Oslo, Norway
03	128.39.0.73	Oslo, Norway
04	128.39.46.249	Oslo, Norway
05	193.10.68.101	Oslo, Norway
06	193.10.68.29	Stockholm, Sweeden
07	213.242.69.21	Stockholm, Sweeden
08	213.242.68.201	Stockholm, Sweeden
09	212.187.128.25	London, England
10	4.68.128.106	New York, USA
11	64.159.1.130	San Jose, USA
12	4.68.114.158	San Jose, USA
13	209.245.146.251	San Jose, USA
14	192.5.4.233	San Jose, USA
15	204.152.191.5	San Jose, USA

Table 4.7: Path from Node One to Node Four.

4.3.4 Tools

There are multiple tools for measuring the round trip time, or one-way delays. The tools vary in methods for collecting and measuring the data. The biggest difference is what sort of packet which is used. The available packet formats can be ICMP, TCP or UDP.

The tool used to measure the data, is a modified Perl script that utilized the `ping` command which is available for most operating systems. The Perl script is a part of the `pinger` measurement package, which is used to measure round trip times from links all around the world.

The modified Perl script together width other scripts are available in the appendix.

4.3.5 Predictions

The predictions for the result are as follows:

- Node Three has the least amount of hops, and is located in Oslo, Norway, this node will probably have the lowest delay, the lowest jitter, and the lowest packet loss.
- Node Two has the second least amount of hops, and it is located in Norway, so this node will probably have a low delay, a predictable jitter, and a low packet loss.
- Node four is located in the United States of America. This will probably result in a high delay, with at times unpredictable jitter. The packet loss should however be relative low, with today's network link properties.

Chapter 5

Results

5.1 Case One: Network Traffic

The data collected from the two nodes by `tcpstat` program, consists of 17.280 lines, where each line represents one measurement. A sample from the measured data is located beneath.

Output 1 - *10 sample measurements from the tcpstat-node2.log file:*

```
...
1115157729 0 0 0 117 116 1 687.73 546.39 128742.40 23.40 117 80464 0.01
1115157734 0 0 0 138 136 2 668.99 556.25 147712.00 27.60 138 92320 0.01
1115157739 0 0 0 130 122 8 674.28 540.56 140249.60 26.00 130 87656 0.01
1115157744 0 0 0 118 117 1 708.32 556.59 133731.20 23.60 118 83582 0.01
1115157749 0 0 0 126 124 2 674.03 543.12 135884.80 25.20 126 84928 0.01
1115157754 0 0 0 135 133 2 640.15 552.20 138272.00 27.00 135 86420 0.01
1115157759 0 0 0 125 121 4 701.07 545.80 140214.40 25.00 125 87634 0.01
1115157764 0 0 0 125 123 2 675.81 557.62 135161.60 25.00 125 84476 0.01
1115157769 2 0 0 120 118 2 650.70 541.93 127017.60 24.40 122 79386 0.01
1115157774 1 0 0 123 121 2 723.53 572.89 143548.80 24.80 124 89718 0.01
...
```

A description of the columns can be found in table 5.1.

5.1.1 Analysis and Presentation

The `tcpstat` program extracts some data from the host node, but it also provides some data that are processed from the measured data. These values are based on the data that has been measured since last measurement. The following values are processed by `tcpstat` itself, and can be viewed in the raw data log:

- The average/mean packet size.
- The standard deviation of the size of each packet.
- The number of bits per second.
- The number of packets per second.

Column	Description
Column 01	Timestamp in UNIX time.
Column 02	The number of ARP packets.
Column 03	The number of ICMP and ICMPv6 packets.
Column 04	The number of IPv6 packets.
Column 05	The number of IPv4 packets.
Column 06	The number of TCP packets.
Column 07	The number of UDP packets.
Column 08	The average packet size.
Column 09	The standard deviation of the size of each packet.
Column 10	The number of bits per second.
Column 11	The number of packets per second.
Column 12	The number of packets.
Column 13	The number of bytes.
Column 14	The network "load" over the last minute, like in uptime.

Table 5.1: Description of the raw data.

Node One

The `tcpstat` program gathers information about the throughput, shown in packages and in bits per second. It shows how the packets are distributed between the different network protocols. And it shows the load of the CPU usage, on the node.

General statistical data from the measurements of Node One, can be viewed in table 5.2.

Description	Bits per second	Packets per second
Minimum value	147,20	0,40
Maximum value	58.299.084,80	28.406,80
Mean value	1.266.042,45	474,57
Median value	7.160,00	7,40
Standard deviation value	6.114.076,61	2.555,20

Table 5.2: Statistics for Node One.

The `tcpstat` program measures how many network layer packets that has passed through the system since the last measurement. Statistical data from these measurements, can be viewed in table 5.3.

The `tcpstat` program also measures how many transport layer packets that has passed through the system since the last measurement. Statistical data from these measurements, can be viewed in table 5.4.

The distribution of the transport layer protocols can be viewed in the pie chart in figure 5.1. The transport layer protocols are TCP and UDP, these protocols are encapsulated within the IP protocol.

The two first figures show the throughput, in megabits per second, for Node One. Figure 5.2 shows the measurement in a time series diagram, and figure 5.3 shows the

Description	ARP	ICMP	IPv6	IPv4
Minimum value	0	0	0	0
Maximum value	769	52	24	142.025
Mean value	3,08	5,94	0,06	2367,12
Median value	0,00	0,00	0,00	33,00
Standard deviation value	19,15	13,28	0,33	12.775,29
Sum packages	53.287	102.632	979	40.903.783
Distribution of packages	0,13%	0,25%	0,00%	99,62%

Table 5.3: Statistics of Node One's Network Layer Protocols.

Description	TCP	UDP
Minimum value	0	0
Maximum value	142.022	75.583
Mean value	1.507,92	850,85
Median value	9,00	14,00
Standard deviation value	10.417,22	7.052,29
Sum packages	26.056.907	14.702.654
Distribution of packages	63,93%	36,07%

Table 5.4: Statistics of Node One's Transport Layer Protocols.

distribution of the measurements, in a histogram diagram.

The next two figures show the throughput, in packets per second, for Node One. Figure 5.4 shows the measurements in a time series diagram, while figure 5.5 show the distribution of the measurements in a histogram diagram. Only the distribution from 0-50 packets per second is shown, as this represents 95,76% of the total measurements.

Figure 5.6 shows the CPU load of Node One at the time the measurement was conducted.

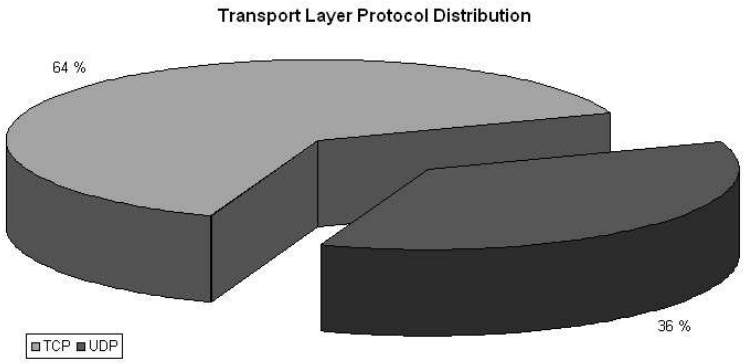

Figure 5.1: Distribution of the Transport Layer Protocols for Node One.

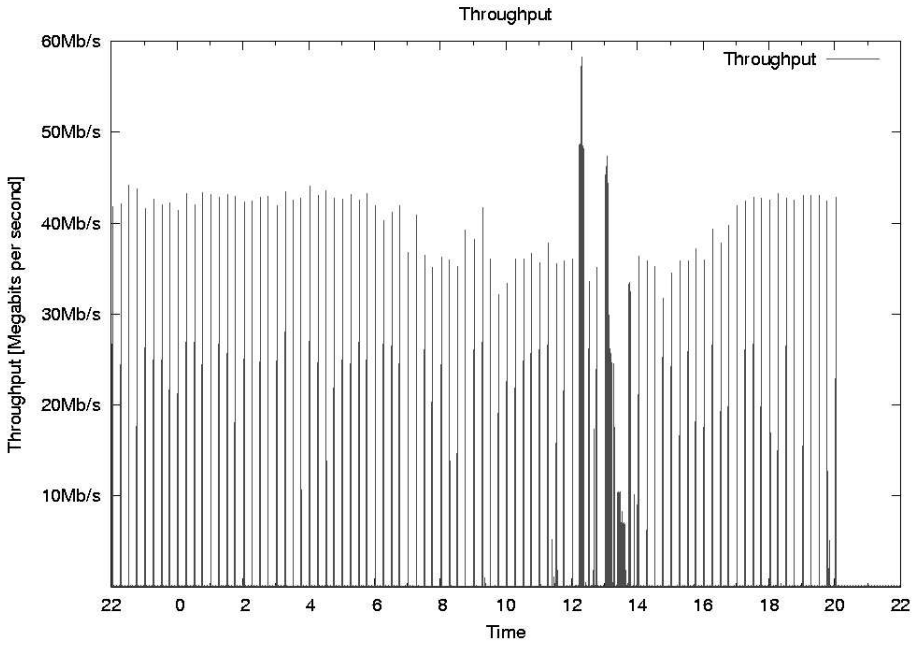

Figure 5.2: Throughput in bits per second for Node One.

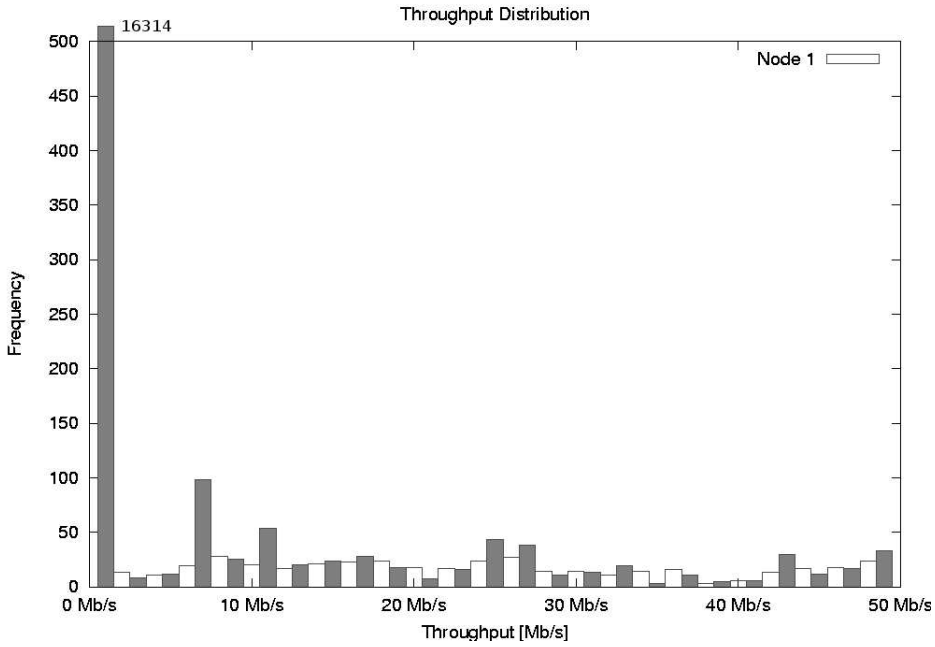

Figure 5.3: Distribution of the throughput (bps) for Node One.

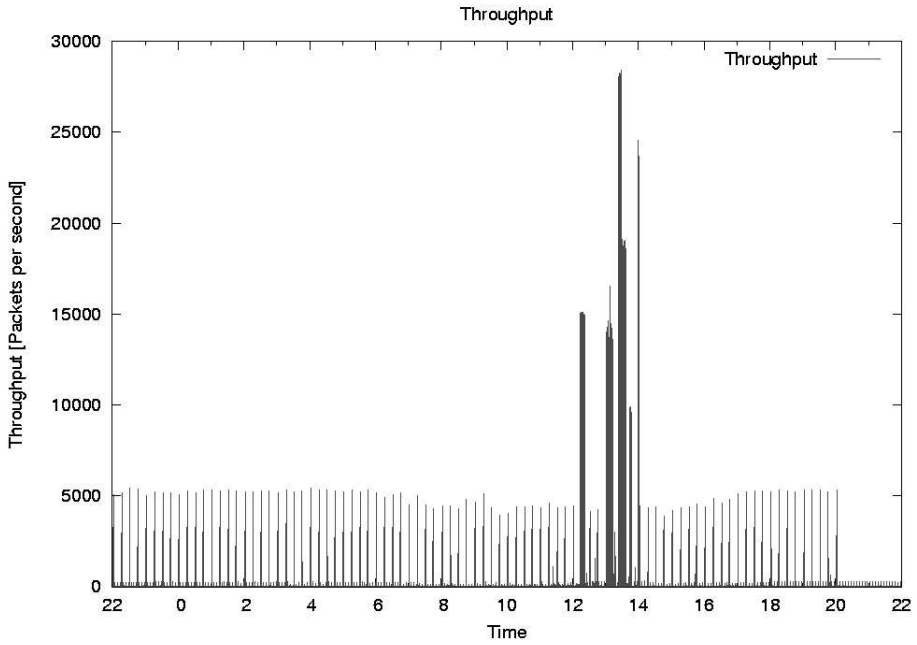

Figure 5.4: Throughput in packets per second for Node One.

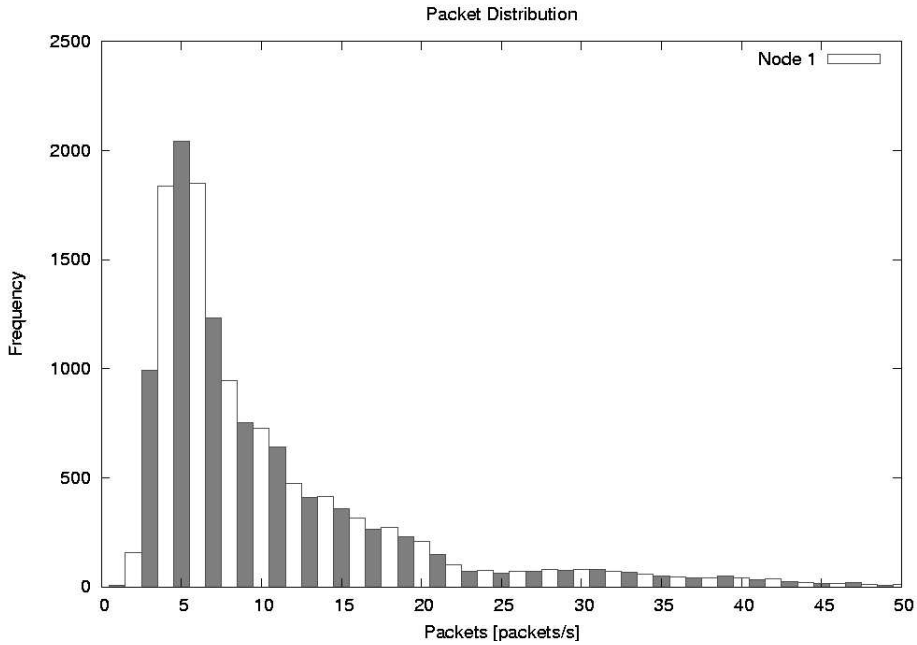

Figure 5.5: Distribution of the throughput (pps) for Node One.

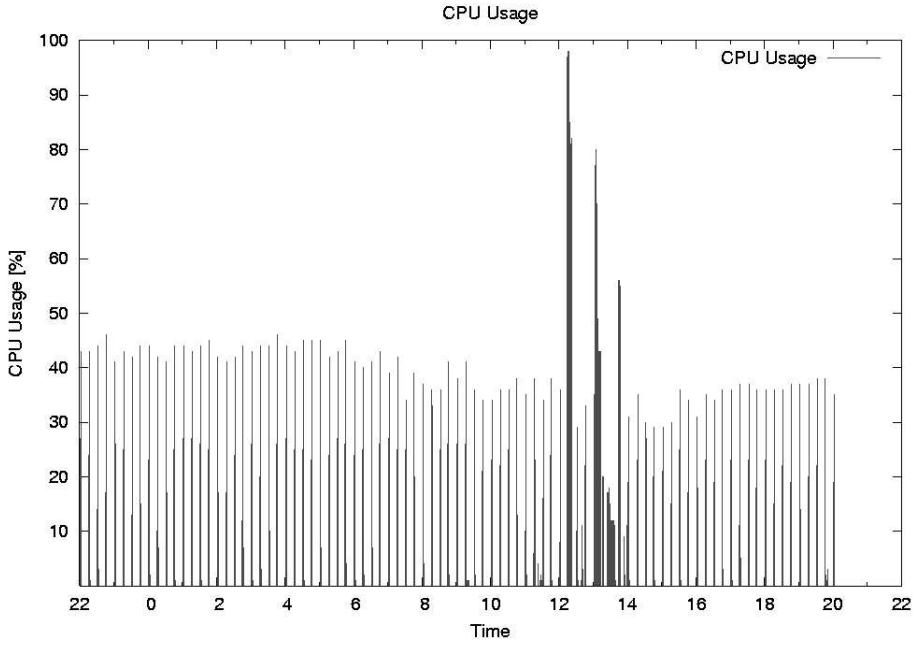

Figure 5.6: CPU usage for Node One.

Node Two

As with Node One, the same analysis and presentations has been done for Node Two.

The general statistical data from the measurements of Node Two, can be viewed in table 5.5.

Description	Bits per second	Packets per second
Minimum value	76,80	0,20
Maximum value	14.849.148,80	2.002,40
Mean value	564.568,72	86,81
Median value	5.704,00	3,20
Standard deviation value	1.832.270,03	256,26

Table 5.5: Statistics for Node Two.

The statistical data from the network layer packets, can be viewed in table 5.6.

Description	ARP	ICMP	IPv6	IPv4
Minimum value	0	0	0	1
Maximum value	16	14	0	10.012
Mean value	0,53	0,70	0,00	433,51
Median value	0,00	0,00	0,00	15,00
Standard deviation value	0,94	2,58	0,00	1.281,29
Sum packages	9.227	12.035,00	0	7.490.984
Distribution of packages	0,12%	0,16%	0,00%	99,72%

Table 5.6: Statistics of Node Two's Network Layer Protocols.

The statistical data from the transport layer packets, can be viewed in table 5.7.

Description	TCP	UDP
Minimum value	0	1
Maximum value	10.011	2.777
Mean value	429,33	3,46
Median value	7,00	2,00
Standard deviation value	1.279,95	28,26
Sum packages	7.418.787	59.865,00
Distribution of packages	99,20%	0,80%

Table 5.7: Statistics of Node Two's Transport Layer Protocols.

The distribution of the transport layer protocols can be viewed in the pie chart in figure 5.7.

Figure (figure 5.8) shows the CPU load of Node Two at the time the measurement was conducted.

The two next figures show the throughput, in megabits per second, for Node Two. Figure 5.9 shows the measurement in a time series diagram, and figure 5.10 shows the

distribution of the measurements, in a histogram diagram.

The two last figures show the throughput, in packets per second, for Node Two. Figure 5.11 shows the measurements in a time series diagram, while figure 5.12 show the distribution of the measurements in a histogram diagram.

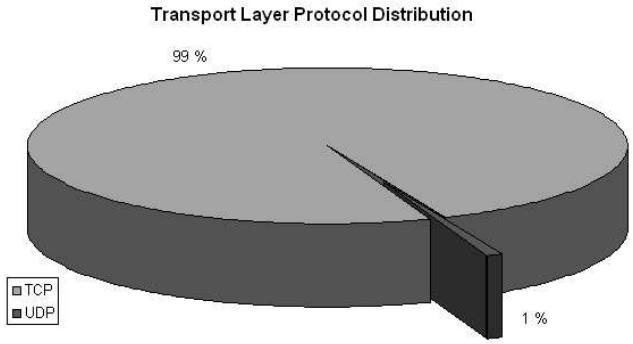

Figure 5.7: Distribution of the Transport Layer Protocols for Node Two.

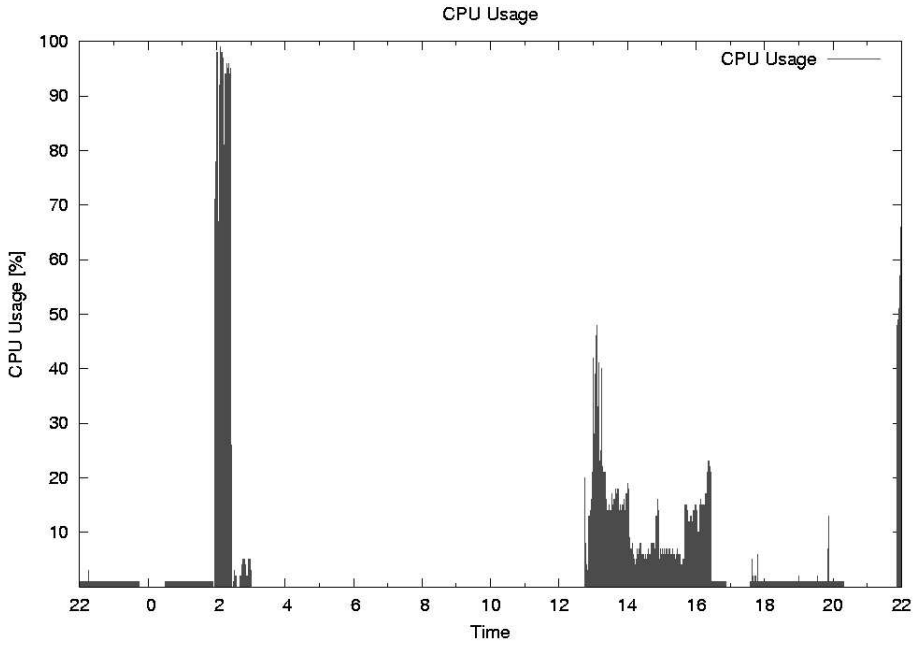

Figure 5.8: CPU usage for Node Two.

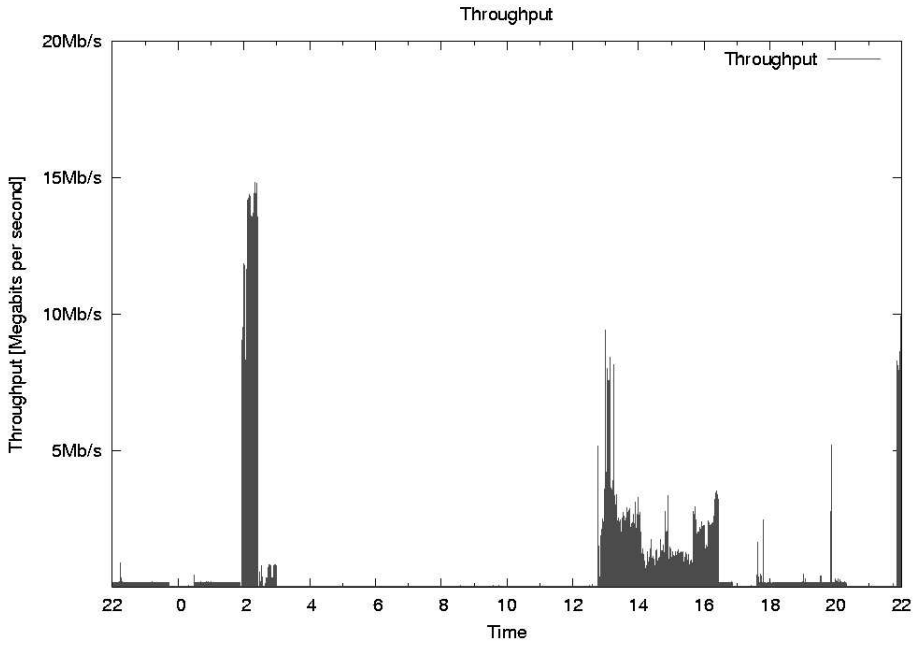

Figure 5.9: Throughput in bits per second for Node Two.

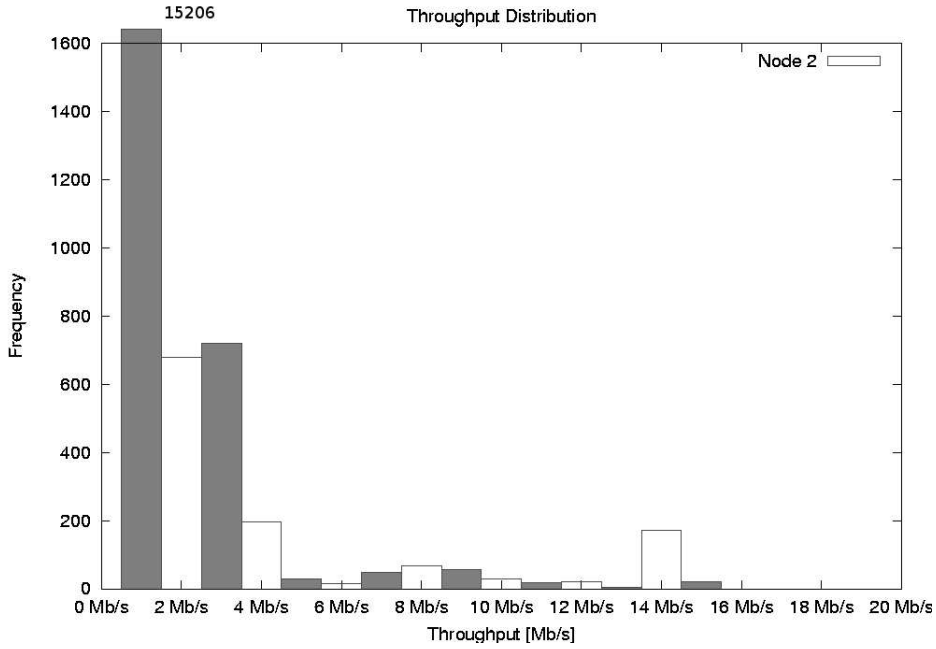

Figure 5.10: Distribution of the throughput (bps) for Node Two.

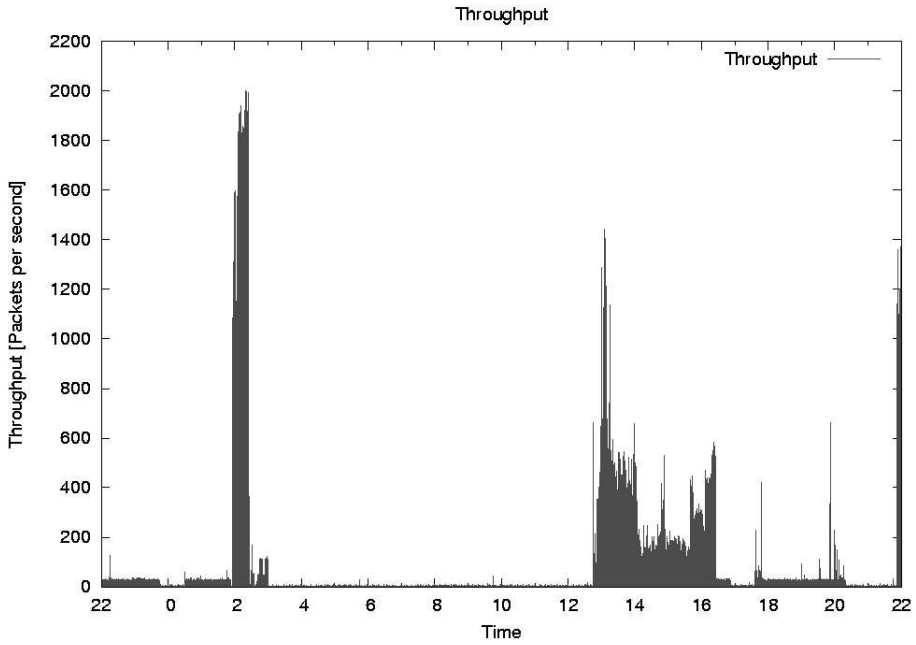

Figure 5.11: Throughput in packets per second for Node Two.

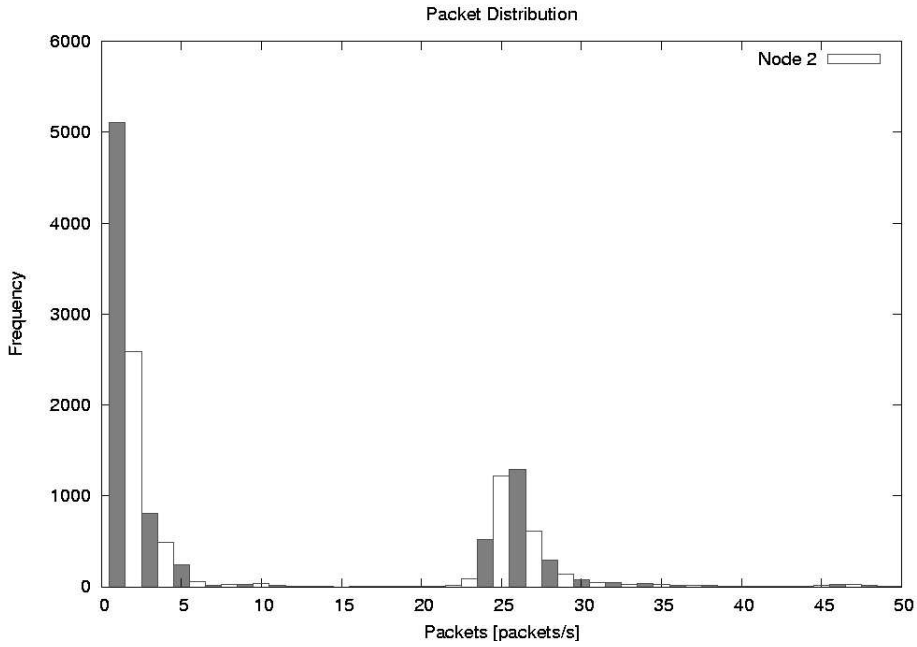

Figure 5.12: Distribution of the throughput (pps) for Node Two.

5.1.2 Interpretation

Node One

From the raw data, the analysis, and the presentations, the following information can be interpreted about the state of Node One's network traffic.

The general statistics shows that during the 24 hours that the node was monitored, it reached a max throughput of 58 Mb/s, which is a network utilization of 29%, as this is a 100 Mb/s full-duplex connection. The mean throughput was 1Mb/s, which gives a mean network utilization of 0,5%. The general statistics also show that the standard deviation is 6 Mb/s, which would indicate that the traffic was retrieved and/or sent in bursts.

In the statistics created by the data from the network layer, one can see that IPv4 dominates the network layer, with a 99,62% margin. But there were over 100.000 ICMP packets sent and/or received during the 24 hours measurements, which means that more then one ICMP packet was sent and/or received every second during that measurement period.

The transport layer statistics show that of the IP traffic, 64% is TCP traffic, and 36% is UDP traffic, this is illustrated in figure 5.1.

Interpretations of the other five figures are as follows:

Figure 5.2: Figure 5.2 presents the throughput measurements, in bits per second, for one day. The figure shows a relative idle connection, with multiple bursts about four times an hour until around 20:00. As the data show similarities with the data gathered in Case Two, this could lead to the assumption that the active measurements conducted in Case Two has been captured by the passive measurements conducted by `tcpstat` in this case.

The monitored traffic is quite periodic, except between 12:00 and 14:00 where the figure shows shorter bursts of downloads or uploads, with a peak at about 60Mb/s. This is in itself interesting, if there is a correlation between these measurements and the measurements done in Case Two, as that measurement show a top throughput at about 40-45Mb/s, while at least 60Mb/s is a possible throughput. This could indicate that it is the internet connection or the other node that was the bottleneck in that experiment.

Figure 5.3: Figure 5.3 presents the distribution of the measured throughput data, in bits per second. It verifies the observations made from figure 5.2 that the connection is mainly idle, with 16.314 measurements between 0 and 1 Mb/s. Figure 5.3 also shows that the remaining 4000 measurements are relatively evenly distributed between 1 Mb/s and 50Mb/s.

Figure 5.4: Figure 5.4 presents the throughput measurements, in packets per second, for one day. The figure shows that about 5000 packets are sent or received in the periodic bursts every 15 minutes. The periodic intervals, and the predictable packet amount, strengthen the hypothesis that this is an experiment, using an active measurement tools.

As pointed out in figure 5.2 there is a change in the throughput between 12:00 and 14:00, this also shows in figure 5.4 as an increased packet amount, with a peak at

about 30.000 packets per second. This is about six times higher, compared with the burst peaks at 5000 packets per second.

The high packet count between 12:00 and 14:00 could indicate some sort of denial of service attempt or more plausible, the usage of a bittorrent client. This is a plausible assumption, as bittorrent is used on this node to download GNU Linux distributions, which is distributed through the bittorrent network.

Figure 5.4 also verifies the observation from figure 5.2, about the halt of bursts at 20:00.

Figure 5.5: Figure 5.5 presents the distribution of the measured throughput data, in packets per second. The figure shows that most of the time, only 3-15 packets per second are passed through the node. This verifies the claim that the connection is mostly idle, as 3-15 packets per second in most situations would be considered as an idle connection, at least for a 100Mb/s full-duplex internet connection.

Figure 5.6: Figure 5.6 presents the CPU usage of the node, during the network traffic measurements. The figure shows a definite correlation between the throughput and the CPU usage. The figure also indicates that 60Mb/s could be the throughput limit, as the CPU reaches 100% utilization at that point. But higher throughput may be possible if the software creating the throughput is requiring much processing power. Lower CPU intensive software may get a higher throughput.

As with figure 5.4, figure 5.6 verifies the observation from figure 5.2 that the bursts stop at 20:00.

Node Two

From the raw data, the analysis, and the presentations, the following information can be interpreted about the state of Node Two's network traffic.

The general statistics shows that during the 24 hours that the node was monitored, it reached a max throughput of 15 Mb/s, which is a network utilization of 29%, as this is a 10 Mb/s full-duplex connection. The mean throughput was 500Kb/s, which gives a mean network utilization of 5%. The general statistics also show that the standard deviation is 1,8 Mb/s, which would indicate that the traffic was retrieved and/or sent in bursts.

In the statistics created by the data from the network layer, one can see that IPv4 dominates the network layer, with a 99,72% margin. The transport layer statistics show that of the IP traffic, 99% is TCP traffic, and 1% is UDP traffic, this is illustrated in figure 5.7.

Interpretations of the other five figures are as follows:

Figure 5.9: Figure 5.9 presents the throughput measurements, in bits per second, for one day. The figure shows a relative idle connection, with some exceptions when something has been downloaded or uploaded. Around 2:00 the graph passed 10 Mb/s which means that there has to be both upload and downloads, as it is a 10 Mb/s full-duplex internet connection.

From the shape of the graph, it looks like the connection is relative stable and providing predictable patterns.

Figure 5.10: Figure 5.10 presents the distribution of the measured throughput data, in bits per second. It shows that the connection is mostly idle, as the throughput between 0 and 1 Mb/s is dominating in frequency. The usual throughput is 2-3 Mb/s, but also 4 Mb/s and 14 Mb/s occurs quite often.

Figure 5.11: Figure 5.11 presents the throughput measurements, in packets per second, for one day. The graph corresponds with the graph from figure 5.9. This graph has a normal amount of packets per second, compared to the throughput.

Figure 5.12: Figure 5.12 presents the distribution of the measured throughput data, in packets per second. The figure shows that there are either 0-5 packets per second, or 24-28 packets per second.

Figure 5.8: Figure 5.8 presents the CPU usage of the node, during the network traffic measurements. The figure shows that on this node there is also a definite correlation between the throughput and the CPU usage, as with the previous node. The figure also indicates that 15Mb/s could be the throughput limit, as the CPU reaches 100% utilization at that point. But this is a bit strange since this has a considerable better performance then the previous node.

5.2 Case Two: Throughput

The data measured between the two nodes by the `netperf` program, consists of 650 lines, where each line represents one measurement. A sample from the measured data are located beneath.

Output 2 *- 15 sample measurements from the netperf-128.39.74.16.log file:*

```
...
1114651931 87380 16384 16384 10.01 41.11
1114652771 87380 16384 16384 10.00 40.63
1114653732 87380 16384 16384 10.00 41.26
1114654632 87380 16384 16384 10.01 40.93
1114655532 87380 16384 16384 10.01 40.97
1114656371 87380 16384 16384 10.01 40.95
1114657332 87380 16384 16384 10.01 40.55
1114658232 87380 16384 16384 10.01 40.69
1114659132 87380 16384 16384 10.00 41.46
1114659971 87380 16384 16384 10.01 40.83
1114660931 87380 16384 16384 10.00 41.24
1114661831 87380 16384 16384 10.01 41.21
1114662731 87380 16384 16384 10.02 28.60
1114663571 87380 16384 16384 10.01 31.03
1114664532 87380 16384 16384 10.01 40.89
...
```

A description of the columns can be found in table 5.8.

Column	Description
Column 01	Timestamp in UNIX time.
Column 02	The buffer socket size for the recieving host.
Column 03	The buffer socket size for the sending host.
Column 04	The send size for the message.
Column 05	Elapsed time, in seconds.
Column 06	The throughput expressed in 10^6 bits per second.

Table 5.8: Description of the raw data.

5.2.1 Analysis and Presentation

The throughput of the measurements is included in the data logs, and is shown in the last column as megabits per second.

Node One to Node Two

Statistical data from the measurements between Node One and Node Two, can be viewed in table 5.9.

A summery of the distribution of the throughput data, can be viewed in table 5.10.

A time series graph, and a histogram graph of the data are presented in figure 5.13, and in figure 5.14.

Description	Value
Minimum value	0,00 Mb/s
Maximum value	25,70 Mb/s
Mean value	20,20 Mb/s
Median value	22,10 Mb/s

Table 5.9: Statistical data between Node One and Node Two

Throughput	in Frequency	in Percentage
00 Mb/s - 05 Mb/s	22	3%
05 Mb/s - 10 Mb/s	19	3%
10 Mb/s - 15 Mb/s	34	5%
15 Mb/s - 20 Mb/s	140	21%
20 Mb/s - 25 Mb/s	353	53%
25 Mb/s - 30 Mb/s	104	15%

Table 5.10: Throughput Distribution between Node One and Node Two

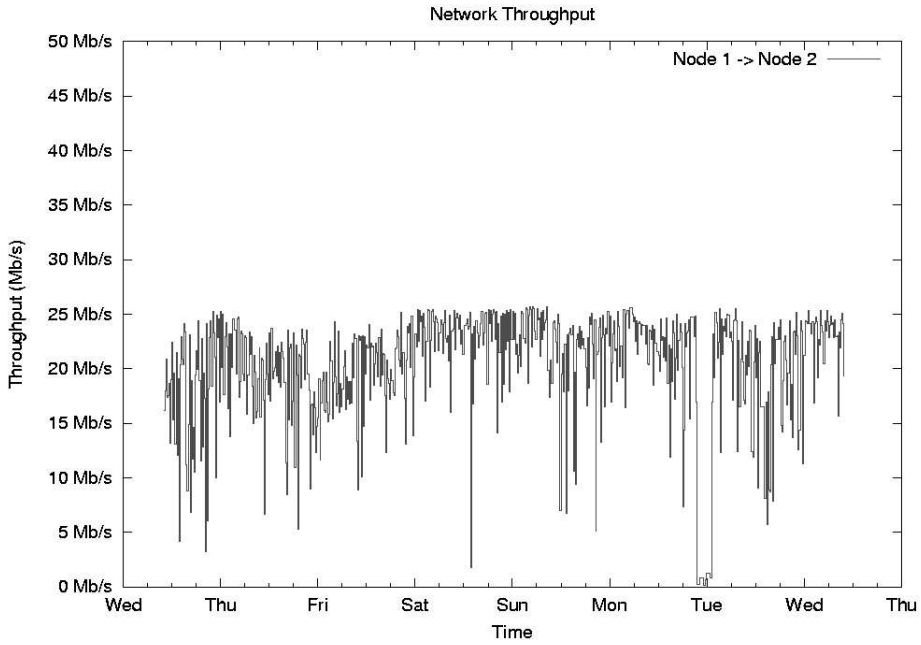

Figure 5.13: The figure shows a Throughput between Node One and Node Two.

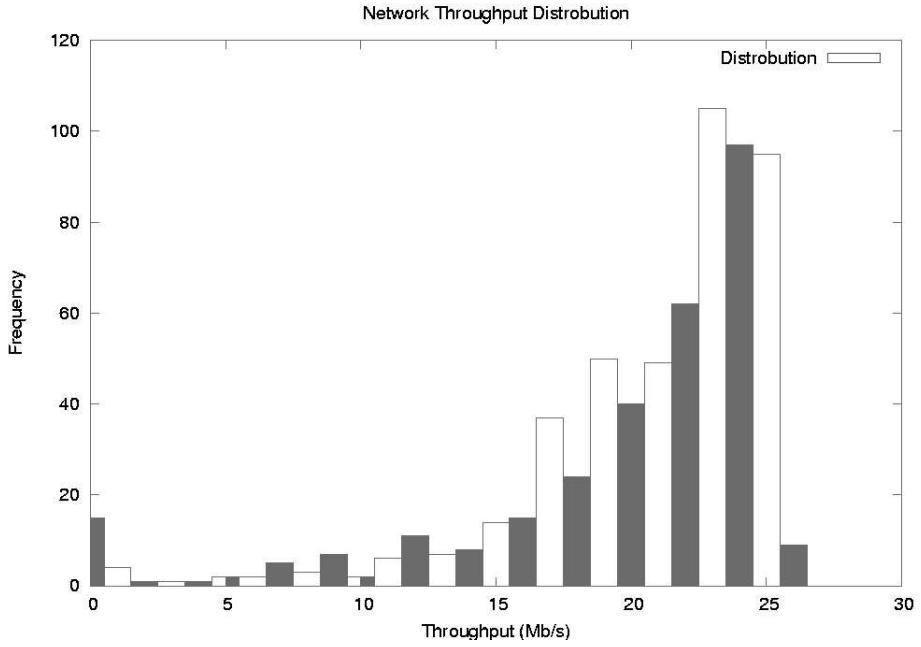

Figure 5.14: The figure shows the distribution of the throughput between Node One and Node Two.

Node One to Node Three

Statistical data from the measurements between Node One and Node Three, can be viewed in table 5.11.

Description	Value
Minimum value	0,00 Mb/s
Maximum value	42,50 Mb/s
Mean value	35,40 Mb/s
Median value	39,30 Mb/s

Table 5.11: Statistical data between Node One and Node Three

A summery of the distribution of the throughput data, can be viewed in table 5.12.

Throughput	in Frequency	in Percentage
00 Mb/s - 05 Mb/s	16	2%
05 Mb/s - 10 Mb/s	0	0%
10 Mb/s - 15 Mb/s	1	0%
15 Mb/s - 20 Mb/s	12	2%
20 Mb/s - 25 Mb/s	40	6%
25 Mb/s - 30 Mb/s	62	9%
30 Mb/s - 35 Mb/s	102	15%
35 Mb/s - 40 Mb/s	138	21%
40 Mb/s - 45 Mb/s	301	45%
45 Mb/s - 50 Mb/s	0	0%

Table 5.12: Throughput distribution between Node One and Node Three

A time series graph, and a histogram graph of the data are presented in figure 5.15, and in figure 5.16.

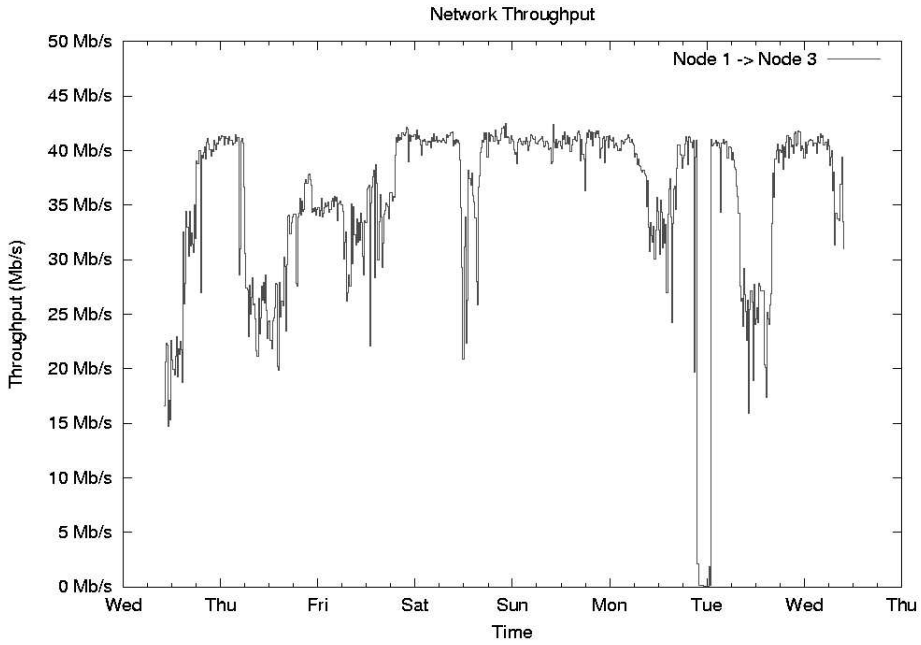

Figure 5.15: The figure shows a Throughput between Node One and Node Three.

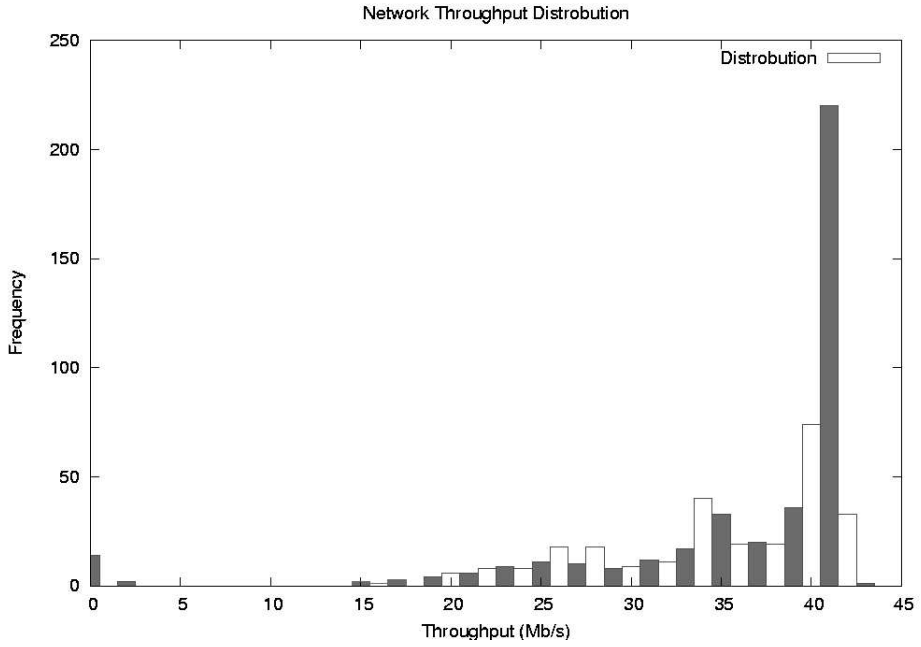

Figure 5.16: The figure shows the distribution of the throughput between Node One and Node Three.

5.2.2 Interpretation

Node One to Node Two

The statistics show a mean throughput value of 22,20 Mb/s, which is a 22,2% utilization of the network bandwidth. But it reached a maximum throughput of 25,70 Mb/s, which is a 25,7% utilization of the network bandwidth.

Figure 5.13 presents the throughput measurements in a time series diagram, which shows a relative smooth graph, with a few exceptions. One of the exceptions is at Tuesday when multiple samples registered low or no connection to the destination node.

Figure 5.14 presents the distribution of the measurements in a histogram diagram. It shows that most of the throughput measurements where between 15 Mb/s and 25Mb/s.

Node One to Node Three

The statistics show a mean throughput value of 35,40 Mb/s, which is a 35,4% utilization of the network bandwidth. But it reached a maximum throughput of 42,50 Mb/s, which is a 42,5% utilization of the network bandwidth.

Figure 5.15 presents the throughput measurements in a time series diagram, which shows an even smoother graph, also this with a few exceptions. One of the exceptions is also at Tuesday at the same time as with the measurements between Node One and Node Two. Even these measurements show low or no connection to the destination node.

There are also four other clear throughput performance drops during the measurement period, but these are created because of congestion, as the throughput only drops from about 40 Mb/s to about 25 Mb/s. These drops occur in working hours, and the congestion is probably due to a heavier load on Node Three and at the shared internet connection.

Figure 5.16 presents the distribution of the measurements in a histogram diagram. It clearly shows that 41 Mb/s is the most frequent throughput utilization, with 40 Mb/s as a distant second.

Comments

The measurements show that the connection between Node One and Node Two provides about 22 Mb/s, while the same measurements between Node One and Node Three provides about 35 Mb/s.

This may be explained by a probably a higher load on the connection between Oslo and Molde, then from one room to another room inside Oslo University College. But if it had only been the connection, there would be periods where the throughput had been considerable higher.

Another explanation could be that the nodes have to low hardware resources to utilize the bandwidth of 100 Mb/s, and that it is the hardware which is the bottleneck.

Even though this proved useful to illustrate the effectiveness of the tool, it could have been interesting to have had the hardware resources to achieve a throughput of 100Mb/s.

5.3 Case Three: Delay, Jitter and Packet Loss

The data collected from the three nodes by the `pinger` script, consists of 2.000 lines, where each line represents one measurement. A sample from the measured data is located beneath.

Output 3 *- 10 sample measurements from the icmp-node4-100b-timeseries.txt file:*

```
...
1114615202 100 10 10 185.8 187.7 190.7 - \
   186.1 188.1 187.9 186.6 187.9 190.7 185.8 187.4 186.7 190.7
1114615502 100 10 10 185.7 187.3 192.0 - \
   186.0 188.0 192.0 185.7 188.1 187.8 185.9 186.7 187.3 185.8
1114615802 100 10 10 185.7 189.2 203.0 - \
   191.0 185.8 186.6 185.7 187.8 185.8 189.1 189.0 203.0 188.3
1114616102 100 10 10 185.8 190.1 213.3 - \
   190.4 186.3 185.9 188.4 213.3 188.2 185.9 189.1 187.9 185.8
1114616402 100 10 10 185.6 188.1 192.1 - \
   185.8 190.1 189.2 190.8 185.7 185.6 185.8 190.6 192.1 186.0
1114616702 100 10 10 185.7 189.6 198.8 - \
   186.6 189.0 191.1 188.3 185.7 198.8 191.4 186.0 192.4 187.4
1114617002 100 10 10 185.5 189.5 199.7 - \
   186.8 193.0 187.9 185.6 185.5 199.7 185.8 187.9 194.9 188.6
1114617302 100 10 10 186.0 190.7 199.6 - \
   187.5 187.7 186.1 186.0 199.6 199.4 190.3 187.8 195.4 188.1
1114617602 100 10 10 185.6 189.3 200.4 - \
   187.7 186.2 187.9 187.3 187.8 200.4 187.7 188.2 194.7 185.6
1114617903 100 10 10 185.7 189.1 207.9 - \
   185.9 188.0 186.1 189.6 189.8 186.1 186.0 186.3 185.7 207.9
...
```

A description of the columns can be found in table 5.13.

Column	Description
Column 01	Timestamp in UNIX time.
Column 02	Package size, expressed in bytes.
Column 03	Number of ICMP packages sent.
Column 04	Number of ICMP packages received.
Column 05	The minimum RTT of the ten packages sent in that measurement.
Column 06	The mean RTT for the ten packages sent in that measurement.
Column 07	The maximum RTT of the ten packages sent in that measurement.
Column 08	Roundtrip time (RTT) measurement number 01.
Column 09	Roundtrip time (RTT) measurement number 02.
Column 10	Roundtrip time (RTT) measurement number 03.
Column 11	Roundtrip time (RTT) measurement number 04.
Column 12	Roundtrip time (RTT) measurement number 05.
Column 13	Roundtrip time (RTT) measurement number 06.
Column 14	Roundtrip time (RTT) measurement number 07.
Column 15	Roundtrip time (RTT) measurement number 08.
Column 16	Roundtrip time (RTT) measurement number 09.
Column 17	Roundtrip time (RTT) measurement number 10.

Table 5.13: Description of the raw data.

5.3.1 Analysis and Presentation

The `pinger` script extracts data gathered from the active measurements, it also provides some processed data based on the measured data. The following values are processed from the ten round trip time values measured for that measurement:

- The minimum RTT of the ten packages sent in that measurement
- The mean RTT for the ten packages sent in that measurement.
- The maximum RTT of the ten packages sent in that measurement.

Node One to Node Two

Statistical data from the measurements between Node One and Node Two, can be viewed in table 5.14.

Description	Value
Minimum value	16, 80 ms
Maximum value	199, 80 ms
Mean value	23, 10 ms
Median value	18, 30 ms
Standard deviation value	16, 60 ms

Table 5.14: Statistical data between Node One and Node Two.

A summery of the distribution of the round trip time data, can be viewed in table 5.15.

Round Trip Time	in Frequency	in Percentage
-> 17 ms	0	0,00%
18 ms	7122	34,80%
19 ms	6171	30,20%
20 ms	1740	8,50%
21 ms	934	4,50%
22 ms	448	2,20%
23 ms	306	1,50%
24 ms - 200 ms	3717	18,20%

Table 5.15: Round Trip Time (RTT) between Node One and Node Two.

A summery of the distribution of the jitter, can be viewed in table 5.16.

The packet loss rate between Node One and Node Two, was $\frac{602}{20160}$, as there where 602 error, and a total of 20160 packets.

To present the measured and analyzed data, the following four figures are used:

- Figure 5.17 shows how the delay varies during the week, through a time series diagram.

Jitter	in Frequency	in Percentage
0 ms	1158	5,70%
1 ms	10703	53,10%
2 ms	2052	10,20%
3 ms	887	4,40%
4 ms	536	2,70%
5 ms	481	2,40%
6 ms ->	4342	21,50%

Table 5.16: Distribution of jitter between Node One and Node Two.

- Figure 5.18 shows the distrobution of the delay, in a histogram diagram.
- Figure 5.19 shows the RTT measurements in a phase plot diagram.
- Figure 5.20 show the distribution of the jitter, in a histogram diagram.

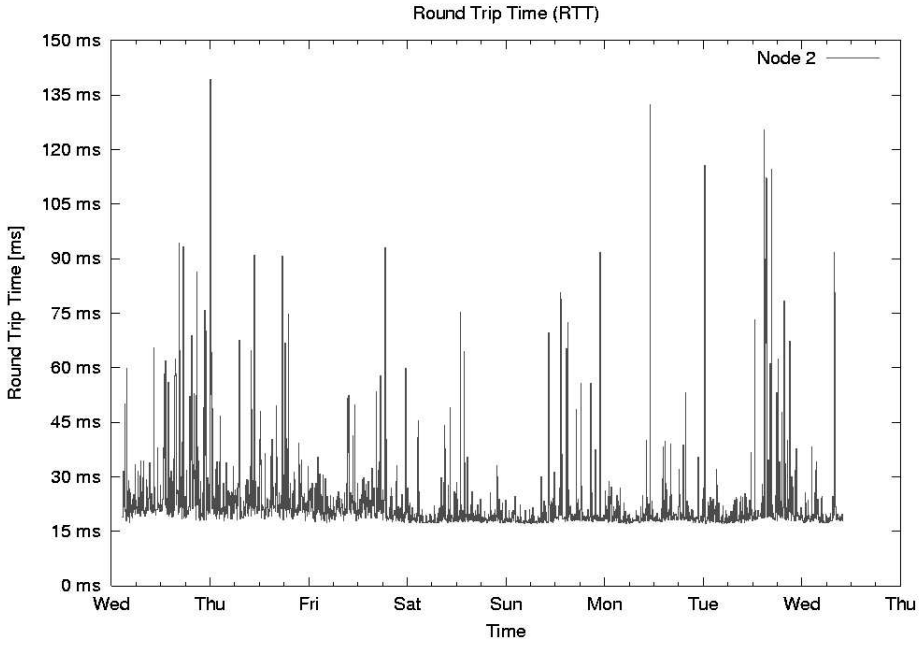

Figure 5.17: Delay between Node One & Node Two.

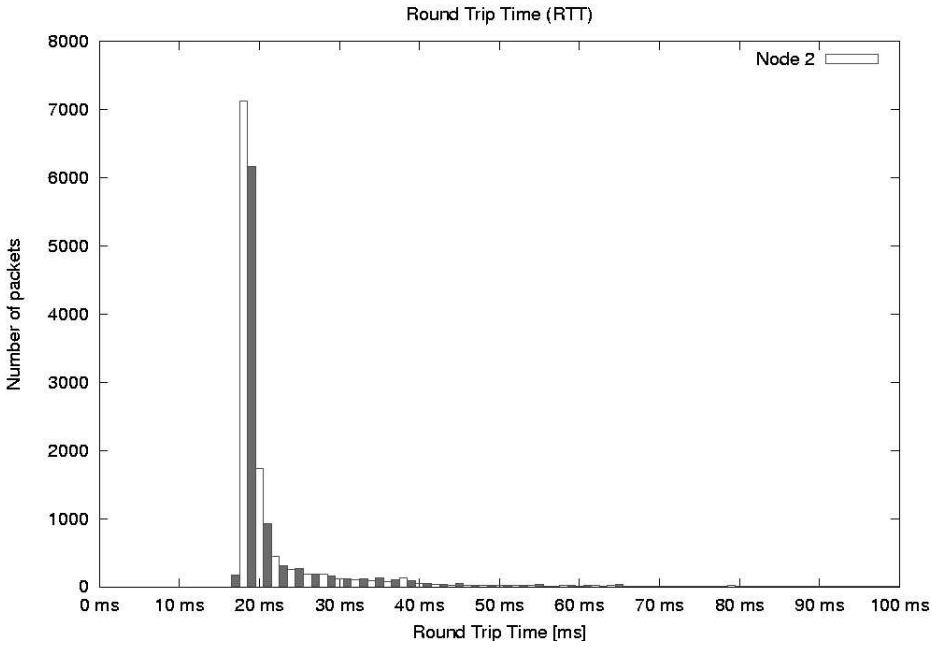

Figure 5.18: Histogram of Node One & Node Two.

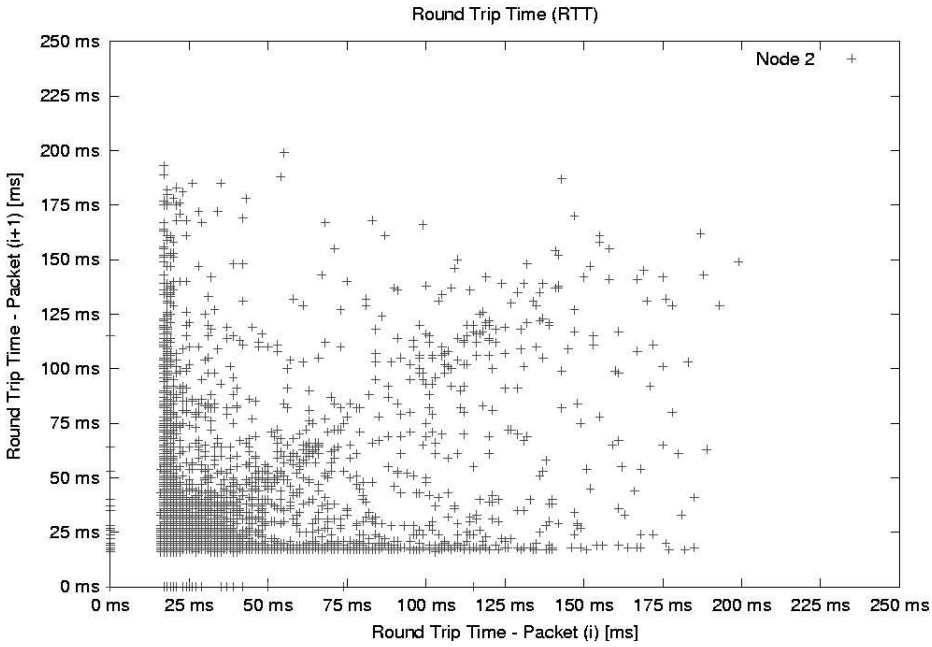

Figure 5.19: Phase plot of Node One & Node Two.

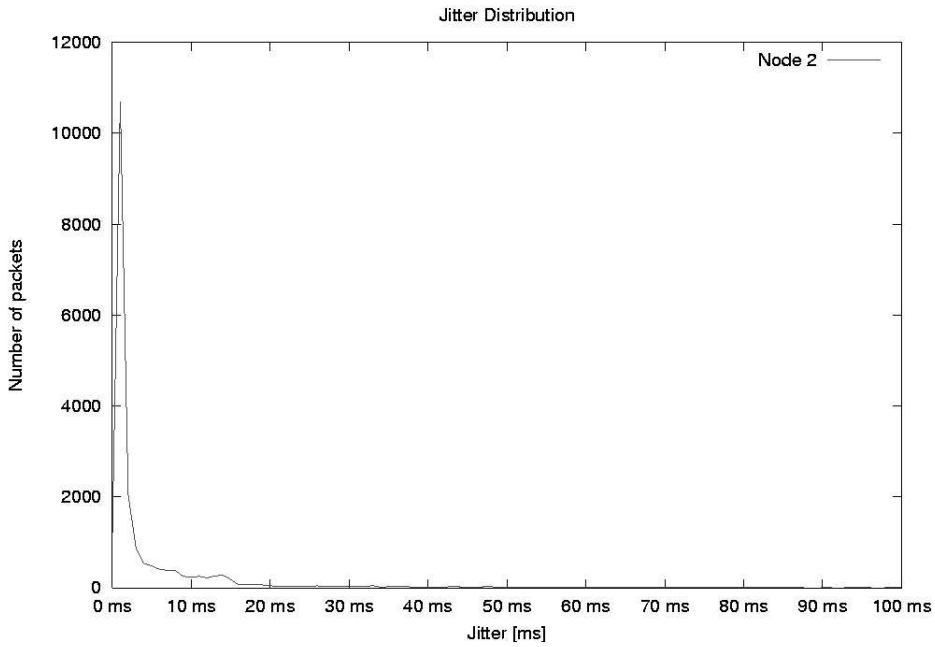

Figure 5.20: Distribution of jitter between Node One & Node Two.

Node One to Node Three

Statistical data from the measurements between Node One and Node Three, can be viewed in table 5.17.

Description	Value
Minimum value	1,00 ms
Maximum value	635,30 ms
Mean value	8,70 ms
Median value	1,60 ms
Standard deviation value	25,25 ms

Table 5.17: Statistical data between Node One and Node Three.

A summery of the distribution of the round trip time data, can be viewed in table 5.18.

Round Trip Time	in Frequency	in Percentage
2 ms	13580	69,40%
3 ms	1845	9,40%
4 ms	731	3,70%
5 ms	482	2,50%
6 ms	288	1,50%
7 ms ->	2640	13,50%

Table 5.18: Round Trip Time (RTT) between Node One and Node Three.

A summery of the distribution of the jitter, can be viewed in table 5.19.

Jitter	in Frequency	in Percentage
0 ms	2602	12,90%
1 ms	11331	56,20%
2 ms	1283	6,40%
3 ms	714	3,50%
4 ms	313	1,60%
5 ms ->	3916	19,40%

Table 5.19: Jitter between Node One and Node Three.

The packet loss rate between Node One and Node Three, was $\frac{594}{20160}$, as there where 594 error, and a total of 20160 packets.

To present the measured and analyzed data, the following four figures are used:

- Figure 5.21 shows how the delay varies during the week, through a time series diagram.
- Figure 5.22 shows the distribution of the measurements, in a histogram diagram.

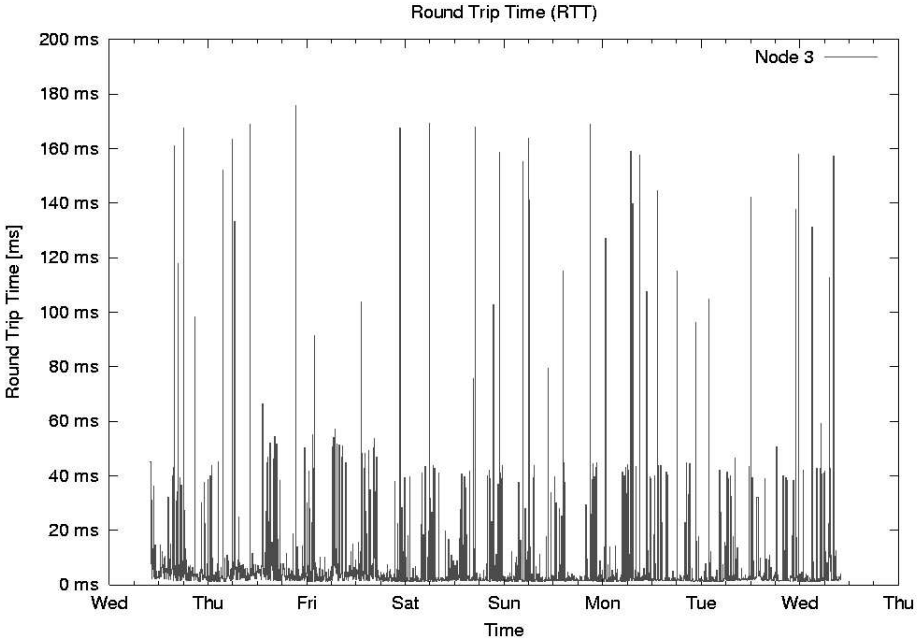

Figure 5.21: Delay between Node One & Node Three.

- Figure 5.23 shows the RTT measurements in a phase plot diagram.
- Figure 5.24 show the distribution of the jitter, in a histogram diagram.

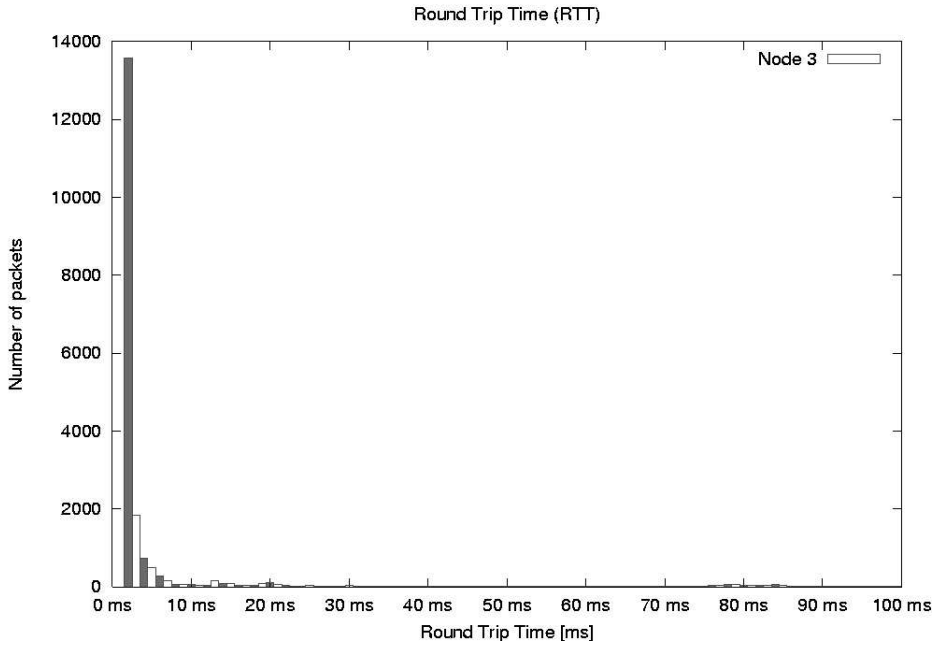

Figure 5.22: Histogram of Node One & Node Three.

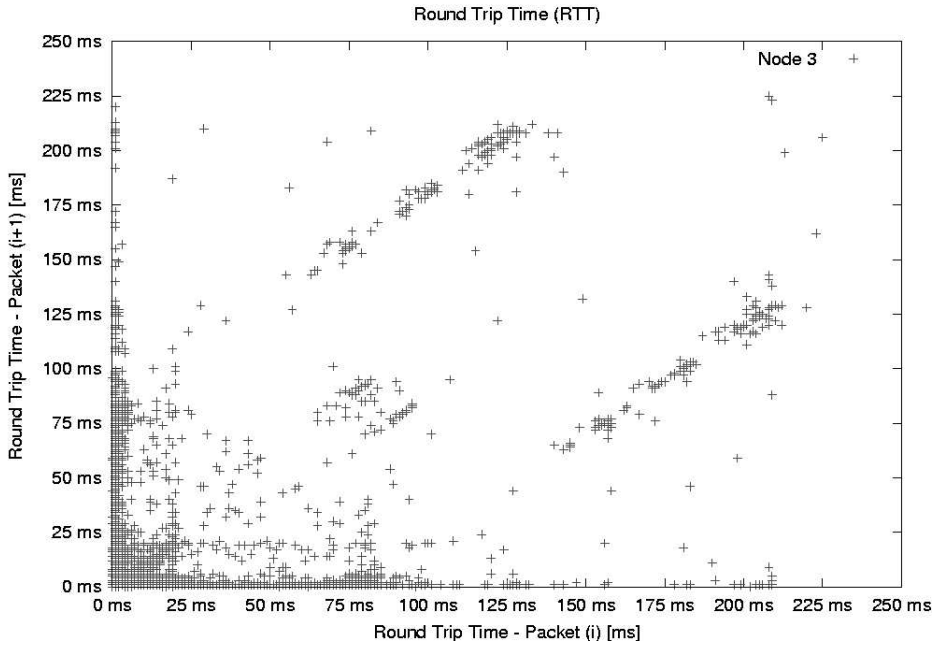

Figure 5.23: Phase plot of Node One & Node Three.

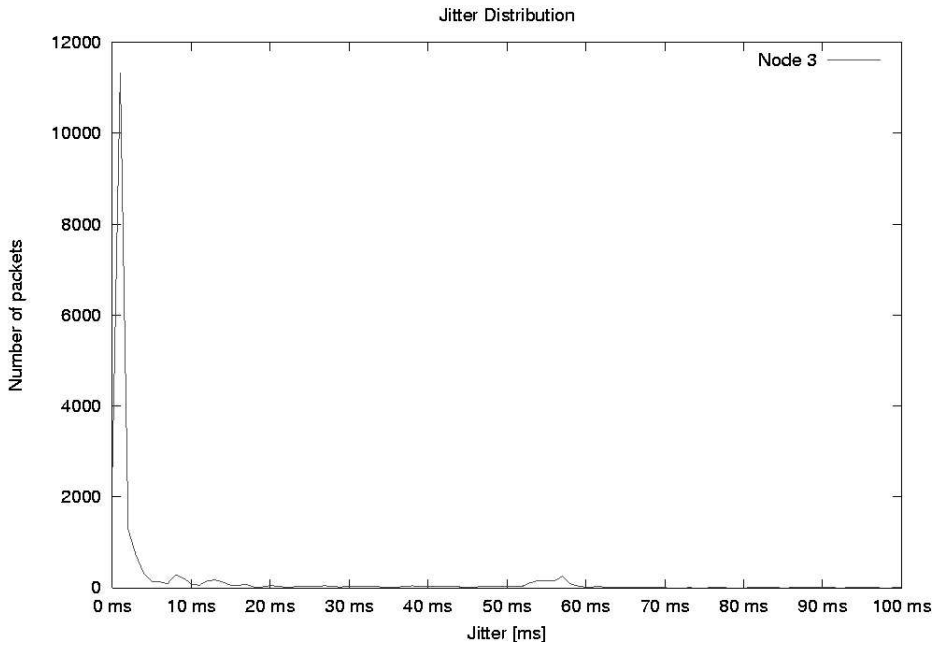

Figure 5.24: Distribution of jitter between Node One & Node Three.

Node One to Node Four

Statistical data from the measurements between Node One and Node Three, can be viewed in table 5.20.

Description	Value
Minimum value	184,70 ms
Maximum value	310,30 ms
Mean value	188,50 ms
Median value	187,60 ms
Standard deviation value	37,40 ms

Table 5.20: Statistical data between Node One and node four.

A summery of the distribution of the round trip time data, can be viewed in table 5.21.

Round Trip Time	in Frequency	in Percentage
-> 0ms	0	0%
185 ms	22	0,10%
186 ms	6041	31,20%
187 ms	2716	14,00%
188 ms	4015	20,80%
189 ms	2683	13,90%
190 ms	813	4,20%
191 ms	954	4,90%
192 ms ->	2107	10,90%

Table 5.21: Round Trip Time (RTT) between Node One and Node Four.

A summery of the distribution of the jitter, can be viewed in table 5.22.

Jitter	in Frequency	in Percentage
0 ms	1430	7,10%
1 ms	7295	36,20%
2 ms	6178	30,70%
3 ms	1548	7,70%
4 ms	970	4,80%
5 ms ->	2738	13,60%

Table 5.22: Jitter between Node One and Node Four.

The packet loss rate between Node One and node four, was $\frac{809}{20160}$, as there where 809 error, and a total of 20160 packets.

To present the measured and analyzed data, the following four figures are used:

- Figure 5.25 shows how the delay varies during the week, through a time series diagram.

- Figure 5.26 shows the distribution of the measurements, in a histogram diagram.
- Figure 5.27 shows the RTT measurements in a phase plot diagram.
- Figure 5.28 show the distribution of the jitter, in a histogram diagram.

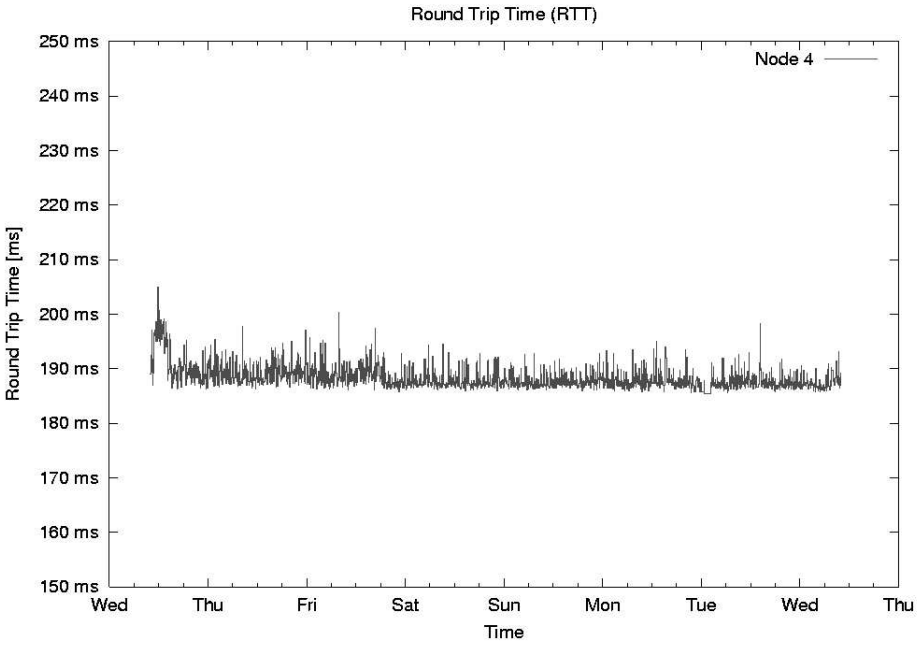

Figure 5.25: Delay between Node One & Node Four.

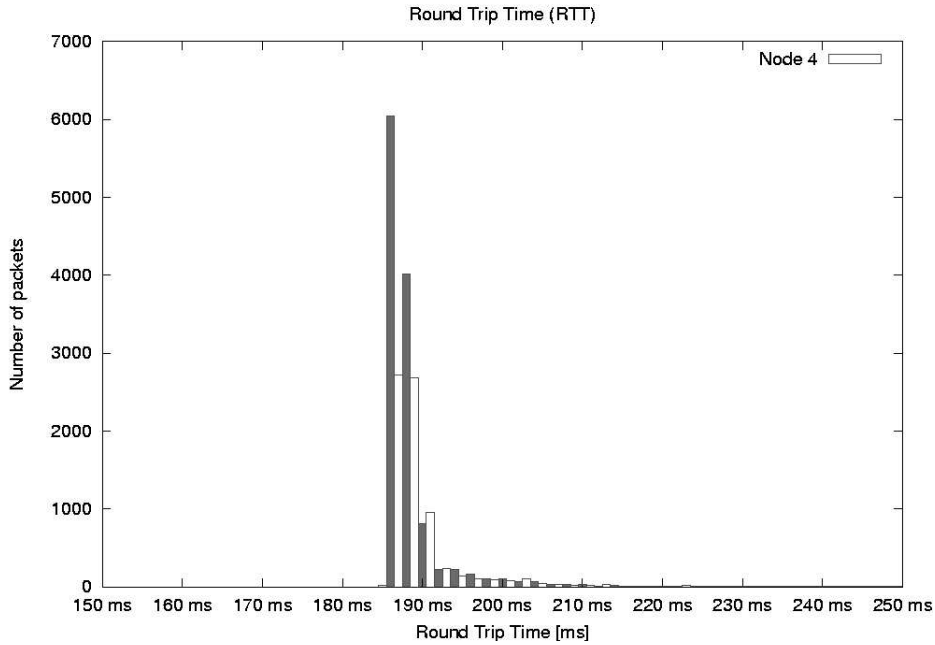

Figure 5.26: Histogram of Node One & Node Four.

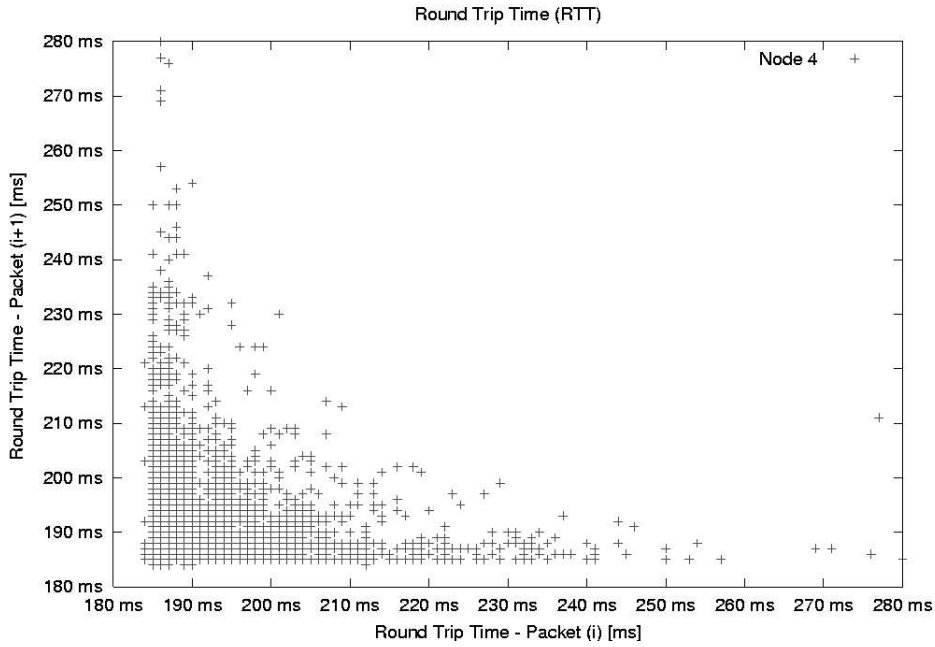

Figure 5.27: Phase plot of Node One & Node Four.

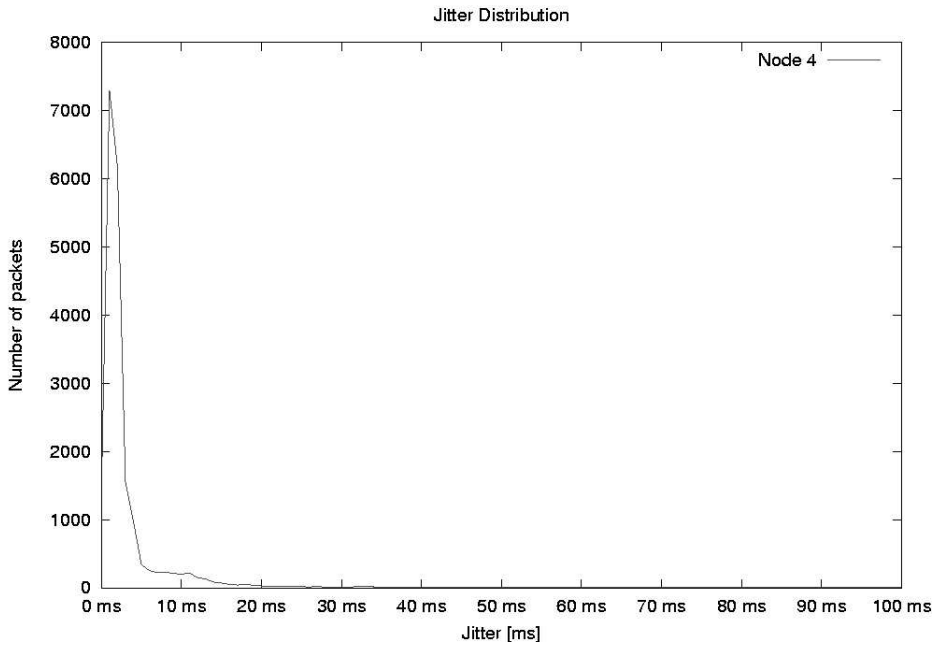

Figure 5.28: Distribution of jitter between Node One & node four.

5.3.2 Interpretation

Node One to Node Two

The measurements show that there where 602 errors out of a total of 20.160 packets. Of the packets that where received, the minimal round trip time was 16,80 ms, the maximum round trip time was 199,80 ms, and the mean round trip time was 23,10 ms.

Figure 5.17 presents the mean value, for the ten samples sent in each measurement, in a time series diagram. The figure shows a relative unpredictable pattern, where the transmission delay and the propagation delay probably mainly represents the predictable 18 ms minimal delay, and the queuing delay represents the random delay. The standard deviation of 16,60 ms seems to correspond with figure 5.17.

Figure 5.18 presents the distribution of the measured round trip times, in a histogram. The figure shows that most packets arrive within 18 ms to 20 ms, which is a low delay.

Figure 5.19 present the RTT_i and RTT_{i+1} packets in a phase plot diagram. This diagram is useful for recognizing congestion. The figure show that most samples are in region I (bottom left corner), which suggests a low congestion, where the packet pass easily through the routers on the path. But there are some congestion in the measurements, showing in region II and III.

Figure 5.20 presents a distribution of the jitter, in a histogram. From the figure it becomes clear that the majority of the packets arrive within 4 ms, the rest of the packets arrive within the next 15 ms, with few exceptions.

The figures show that the delays are low, the jitter is low, and the packet loss is also low. This shows that the internet connection between Node One and two are of excellent quality. But since there where some bursts of data, this could mean that the connection does not widely support quality of service, but rather achieves the result by the use of overprovisioning.

Node One to Node Three

The measurements show that there where 594 errors out of a total of 20160 packets. Of the packets that where received, the minimal round trip time was 1,00 ms, the maximum round trip time was 635,30 ms, and the mean round trip time was 8,70 ms.

Figure 5.21 presents the mean value, for the ten samples sent in each measurement, in a time series diagram. The figure has a relative predictable pattern, where the transmission delay and the propagation delay represents the predictable 1-2 ms minimum delay, and the queuing delay represents the random delay. The pattern shows bursts of data from about 5 ms, to about 40 ms, and occasionally bursts up to 160 ms, but the bursts occur at fixed periods of the day, indicating periods of high load. The standard deviation of 25,25 ms seems to correspond with figure 5.21.

Figure 5.22 presents the distribution of the measured round trip times, in a histogram. The figure shows an exponential graph, where most of the packets arrive within 1 ms, which is a very low delay.

Figure 5.23 present the RTT_i and RTT_{i+1} packets in a phase plot diagram. The figure show that most samples are in region I (bottom left corner), which suggests a low congestion. But there are some congestion in the measurements, showing in region II and III.

Figure 5.24 presents a distribution of the jitter, in a histogram. From the figure it becomes clear that the majority of the packets arrive within 4 ms, then some more packets arrive around 10 ms, then some more packets at around 15 ms, and finally the last packets arrive within 60 ms.

The figures show that the delays are low, the jitter is low to medium, and the packet loss is also low. This shows that the internet connection between Node One and three are of good quality, with a very low delay. But because of the bursts of data, the connection does probably not widely support quality of service, but rather achieves the result by the use of overprovisioning. This is at times successful, but during the peaks, may provide problems for the jitter sensitive applications.

Node One to Node Four

The measurements show that there where 809 errors out of a total of 20160 packets. Of the packets that where received, the minimal round trip time was 184,70 ms, the maximum round trip time was 310,30 ms, and the mean round trip time was 188,50 ms.

Figure 5.25 presents the mean value, for the ten samples sent in each measurement, in a time series diagram. The figure has a very predictable pattern, where the transmission delay and the propagation delay represents the predictable 185 ms minimum delay, and the queuing delay represents the random delay, but the delay is only about 10 ms. The standard deviation of 37,40 ms seems not to correspond with figure 5.25.

Figure 5.26 presents the distribution of the measured round trip times, in a histogram. The figure shows that most of the packets arrive within the range of 184 ms to 191 ms, which is a relative large delay.

Figure 5.27 present the RTT_i and RTT_{i+1} packets in a phase plot diagram. The figure show that most samples are in region I (bottom left corner), which suggests a low congestion. The figure also shows that there are some transition delays, where one of the packets are queued, while the other packets pass right through, but these transitions does not occur often.

Figure 5.28 presents a distribution of the jitter, in a histogram. From the figure it becomes clear that the majority of the packets arrive within 6 ms, and the rest of the packets arrive within 20 ms.

The figures show that the delays are medium, the jitter is low, and the packet loss is also low. This shows that the internet connection between Node One and four are of good quality, but with a medium delay. This delay is probably caused by the propagation delay, from the United States of Americas west coast to Europe, through the Atlantic Ocean.

But because of the lack of bursts, this connection probably supports quality of service, by the use of some sort of traffic shaping method.

Chapter 6

Conclusion

The objective with this master thesis was to assist network and system-administrators in administration of remote computer networks.

This was primary done by identifying the properties for securing the remote computer networks, and the properties that are important for the quality of the services. The properties provide quality of service for the connection between the remote computer networks.

Secondary, some simple methods for analyzing and presenting the measured data was identified. These methods simplify the interpretation part of the administration of remote computer networks.

The three case studies were created to demonstrate the functionality for some of the tools used to measure the four properties in quality of service.

In Case One, the objective was to make use of passive throughput measurement tools, to monitor the traffic on two different nodes for one day. The `tcpstat` tool successfully measured the data on both nodes, and provided enough information to create a good understanding of what had happened on the network for the last 24 hours. The only mistake in these two experiments was that the filter functionality in `tcpstat` should had been used to filter input and output traffic. But as the objective was demonstrate, the experiments can still be classified as successful.

In Case Two, the objective was to make use of active throughput measurement tools, to benchmark the network connection between two different nodes for one week. The `netperf` tool successfully measured the throughput from one node, to two other nodes situated at different computer networks. In this case, the results did not match the predictions, and this had probably something to do with limited hardware resources at the nodes. It would have been interesting to remove that bottleneck, and have really tested the network throughput between two high performance nodes, or at least include the CPU of the measurement nodes during the measurement. But again the objective was to demonstrate the active throughput tools, and since `netperf` performed as expected, the experiment can still be classified as successful.

In Case Three, the objective was to make use of active delay measurement tools, to benchmark the network connection between three different nodes for one week. The round trip time measurement that was done in the experiment can be used to find both the delay and the jitter of a connection. The packet loss data can be used to figure out the reliability of the connection. These three properties provide important information

about the quality of service for the connection. And with the available information relative important assumptions could be done to describe the quality of service for these network connections. This helps the system administrator in designing the network for services that require different quality of service properties. This experiment can definitely be classified as successful.

The three case studies together demonstrated all aspects of quality of service measurements, and together with the analysis and presentation methods described in this thesis, the network and system administrators should be able to administrate remote computer networks.

Bibliography

[1] Andrew S. Tanenbaum. *Computer Networks, Fourth Edition*. Prentice Hall, 2003.

[2] Kevin Hamilton Kennedy Clark. *Cisco LAN Switching (CCIE Professional Development)*. Cisco Press, 1999.

[3] Annabel Z. Dodd. *The Essential Guide to Telecommunications, Second Edition*. Prentice Hall PTR, 1999.

[4] Sergio Verdú. Wireless bandwidth in the making. *IEEE*, 2000.

[5] Gene Spafford Simson Garfinkel, Alan Schwartz. *Practical Unix & Internet Security, 3rd Edition*. O'Reilly, 2003.

[6] Fred Halsall. *Data Communications, Computer Networks and Open Systems, Fourth Edition*. Addison-Wesley, 1996.

[7] William Stallings. Local networks. *ACM*, 1984.

[8] Mahbub Hassan and Raj Jain. *High Performance TCP/IP Networking*. Pearson Prentice Hall, 2004.

[9] W. Richard Stevens. *The Protocols (TCP/IP Illustrated, Volume 1)*. Addison-Wesley, 1993.

[10] Matt Bishop. *Computer Security, Art and Science*. Addison-Wesley, 2002.

[11] Cross-Industry Working Team. Internet service performance: Data analysis and visualization. Technical report, The Cross-Industry Working Team (XIWT), 2000.

[12] Mark Burgess. *Analytical Network And System Administration*. Wiley, 2004.

[13] B. Kleiner P.A. Tukey J.M. Chambers, W.S. Cleveland. *Graphical Methods for Data Analysis*. Duxbury Press, 1983.

[14] Lionel M. Ni Xipeng Xiao. Internet qos - a big picture. *IEEE*, 1999.

[15] Bruce S. Davie Larry L. Peterson. *Computer Networks - A System Approach, Second Edition*. Morgan Kaufmann, 2000.

[16] Joseph D. Sloan. *Network Troubleshooting Tools*. O'Reilly, 2001.

[17] Rick Jones. The netperf website. *http://www.netperf.org/*, May 2005.

[18] The Board of Trustees of the University of Illinois. The iperf website. *http://dast.nlanr.net/Projects/Iperf/*, May 2005.

[19] The ttcp website. *http://www.pcausa.com/Utilities/pcattcp.htm*, May 2005.

[20] The tcpdump website. *http://www.tcpdump.org/*, May 2005.

[21] Paul Herman. The tcpstat website. *http://www.frenchfries.net/paul/tcpstat/*, May 2005.

[22] Sugih Jamin Amgad Zeitoun, Zhiheng Wang. Rttometer: Measuring path minimum rtt with confidence. Technical report, The University of Michigan, Ann Arbor, 2003.

[23] Stefan Savage. The sting website. *http://www.cs.washington.edu/homes/savage/sting/*, May 2005.

[24] PING 127.0.0.1 Computer Services. The ping website. *http://www.ping127001.com/pingpage.htm*, May 2005.

[25] Zhiheng Wang Amgad Zeitoun. The rttometer website. *http://idmaps.eecs.umich.edu/rttometer/*, May 2005.

[26] The pinger website. *http://www-iepm.slac.stanford.edu/pinger/tools/software.html*, May 2005.

[27] Tobias Oetiker. The smokeping website. *http://people.ee.ethz.ch/ oetiker/webtools/smokeping/*, May 2005.

[28] Dictionary.com. Dictionary.com's website. *http://dictionary.com/*, May 2005.

[29] Daniel P. Siewiorek Jim Gray. High-availability computer systems. *IEEE*, 1991.

[30] Miroslaw Malek Allen M. Johnson. Survey of software tools for evaluating reliability, availability, and serviceability. *ACM Computing Surveys*, 1988.

Appendix A

Online Resources

This appendix only includes some data used in this thesis. To retrieve the rest of the material, visit the master programs website at "http://nasa.iu.hio.no/" and follow the links, to retrieve the data.

Avaliable resources on the web are:

- Datalogs
- Measurement scripts
- GNUplot scripts

Appendix B

Experiment Setup

The following scripts can be used to recreate the setup of the experiments.

Case Two: Throughput

Script 1 - *startExperiment.sh:*

```
#!/bin/sh

sleep 1m
timestamp1=`date +%s`;
result1=`netperf -P0 -H 158.38.88.147`;
sleep 1m
timestamp2=`date +%s`;
result2=`netperf -P0 -H 128.39.74.16`;

echo $timestamp1 $result1 >> /usr/local/netperf/netperf-158.38.88.147.log
echo $timestamp2 $result2 >> /usr/local/netperf/netperf-128.39.74.16.log
```

Appendix C

Graph scripts

In the report, GNUplot has been used as the tool to present the data in the graphs and charts.

The following scripts show the precedure for how to create the graphs, but in some situations, the node number has to be exchanged to create the correct graph.

Case One: Network Traffic

GNUplot script 1 - *result-tcpstat-node1-bps-histogram.gpl:*

```
set terminal postscript color
set title "Throughput Distribution"
set xlabel "Throughput [Mb/s]"
set ylabel "Frequency"
set output "result-tcpstat-node1-bps-histogram.ps"

set format x "%g Mb/s"
set xrange [0:50]
set yrange [0:500]
set xtics 10
set ytics 50

plot "tcpstat-node1-bps-histogram.txt" using 1:2 title "Node One" with boxes
```

GNUplot script 2 - *result-tcpstat-node1-bps-timeseries.gpl:*

```
set terminal postscript color
set title "Throughput"

set output "result-tcpstat-node1-bps-timeseries.ps"

set ylabel "Throughput [Megabits per second]"
set yrange [0:60000000]
set ytics ("10Mb/s" 10000000, "20Mb/s" 20000000, "30Mb/s" 30000000, "40Mb/s" 400
00000, "50Mb/s" 50000000, "60Mb/s" 60000000)

set xlabel "Time"
set xdata time
set timefmt "%s"
set format x "%k"

plot "tcpstat-node1.log" using 1:10 title "Throughput" with impulses
```

GNUplot script 3 - *result-tcpstat-node1-cpu-timeseries.gpl:*

```
set terminal postscript color
set title "CPU Usage"

set output "result-tcpstat-node1-cpu-timeseries.ps"

set ylabel "CPU Usage [%]"
set yrange [0.0:1.0]
set ytics ("10" 0.1, "20" 0.2, "30" 0.3, "40" 0.4, "50" 0.5, "60" 0.6, "70" 0.7,
 "80" 0.8, "90" 0.9, "100" 1.0)

set xlabel "Time"
set xdata time
set timefmt "%s"
set format x "%k"

plot "tcpstat-node1.log" using 1:14 title "CPU Usage" with impulses
```

GNUplot script 4 - *result-tcpstat-node1-pps-histogram.gpl:*

```
set terminal postscript color
set title "Packet Distribution"
set xlabel "Packets [packets/s]"
set ylabel "Frequency"
set output "result-tcpstat-node1-pps-histogram.ps"

set xrange [0:50]

set xtics 5

plot "tcpstat-node1-pps-histogram.txt" using 1:2 title "Node One" with boxes
```

GNUplot script 5 - *result-tcpstat-node1-pps-timeseries.gpl:*

```
set terminal postscript color
set title "Throughput"

set output "result-tcpstat-node1-pps-timeseries.ps"

set ylabel "Throughput [Packets per second]"
set yrange [0:30000]
set ytics 5000

set xlabel "Time"
set xdata time
set timefmt "%s"
set format x "%k"

plot "tcpstat-node1.log" using 1:11 title "Throughput" with impulses
```

Case Two: Throughput

GNUplot script 6 - *result-throughput-image1a.gpl:*

```
set terminal postscript color
set title "Network Throughput"
set xlabel "Time"
set ylabel "Throughput (Mb/s)"
set output "result-throughput-image1a.ps"

set yrange [0:50]
set ytics 5
set format y "%g Mb/s"

set xdata time
set timefmt "%s"
set format x "%a"

plot "netperf-158.38.88.147.log" using 1:6 title "Node One -> Node Two" with steps
```

GNUplot script 7 - *result-throughput-image1b.gpl:*

```
set terminal postscript color
set title "Network Throughput Distribution"
set xlabel "Throughput (Mb/s)"
set ylabel "Frequency"
set output "result-throughput-image1b.ps"

plot "netperf-158.38.88.147-dist.txt" using 1:2 title "Distribution" with boxes
```

Case Three: Delay, Jitter and Packet Loss

GNUplot script 8 - *result-icmp-node4-timeseries.gpl:*

```
set terminal postscript color
set title "Round Trip Time (RTT)"
set xlabel "Time"
set ylabel "Round Trip Time [ms]"
set output "result-icmp-node4-100b-timeseries.ps"

set yrange [150:250]
set ytics 10
set format y "%g ms"

set xdata time
set timefmt "%s"
set format x "%a"

plot "icmp-node4-100b-timeseries.txt" using 1:6 title "Node 4" with steps
```

GNUplot script 9 - *result-icmp-node4-histogram.gpl:*

```
set terminal postscript color
set title "Round Trip Time (RTT)"
set xlabel "Round Trip Time [ms]"
set ylabel "Number of packets"
set output "result-icmp-node4-100b-histogram.ps"

set xrange [150:250]
set xtics 10
set format x "%g ms"

plot "icmp-node4-100b-histogram.txt" using 1:2 title "Node 4" with boxes
```

GNUplot script 10 - *result-icmp-node4-jitter.gpl:*

```
set terminal postscript color
set title "Round Trip Time (RTT)"
set ylabel "Round Trip Time - Packet (i+1) [ms] "
set xlabel "Round Trip Time - Packet (i) [ms]"
set output "result-icmp-node4-100b-phaseplot.ps"

set xrange [180:280]
set xtics 10
set format x "%g ms"

set yrange [180:280]
set ytics 10
set format y "%g ms"

plot "icmp-node4-100b-phaseplot.txt" using 1:2 title "Node 4" with points
```

GNUplot script 11 - *result-icmp-node4-phaseplot.gpl:*

```
set terminal postscript color
set title "Jitter"
set xlabel "Time"
set ylabel "Jitter [ms]"
set output "result-icmp-node4-100b-jitter.ps"

set yrange [0:50]
set ytics 5
set format y "%g ms"

set xdata time
set timefmt "%s"
set format x "%a"

plot "icmp-node4-100b-jitter.txt" using 1:3 title "Node 4" with steps
```