NTNU

Norwegian University of
Science and Technology

NPT Online Broadband Test Tool

Lars-Petter Gunhildsberg Hansen
Ivar Conradi Østhus

Master of Science in Communication Technology
Submission date: June 2009
Supervisor: Poul Einar Heegaard, ITEM
Co-supervisor: Johan Foldøy, Post- og teletilsynet

Norwegian University of Science and Technology
Department of Telematics

Problem Description

Norwegian Post and Telecommunications Authority (NPT) will develop and release a tool for online bandwidth measurements during first half of 2009. This tool shall make broadband users able to measure the capacity of their broadband connections, and potentially; by use of the systems infrastructure (web server, test servers, database) let NPT evaluate whether Norwegian providers comply with their Principles of Network Neutrality.

NPT is interested in an evaluation of test tool's capability to reveal traffic discrimination in the customer-provider interface as well as interfaces between providers. In addition, we would like to see a suggestion for a test-setup able to point out any possible breaches on the aforementioned principles. The assignment will review existing tools, measurement techniques, study presentations of results, and discuss potential extension in the test tool, e.g. mobile terminal access capacity evaluation.

The assignment consists of the following tasks
1. Study active and passive measurements techniques
2. Review existing broadband test tool applications
3. Specify measurement scenarios for broadband testing
4. Testing of the measurement scenarios
5. Discuss aggregation and presentation of measurement statistics
6. Elaborate future broadband test tool applications

Assignment given: 15. January 2009
Supervisor: Poul Einar Heegaard, ITEM

Preface

This master's thesis is the final product of our research project carried out during the spring semester 2009. The thesis completes our 5 year study for the master's degree in Communication Technology with specialization in Networked Services and Multimedia Systems. The work has been accomplished at the Department of Telematics at Norwegian University of Science and Technology (NTNU).

The background for this thesis was the Norwegian Post and Telecommunications Authority's work with establishing a new on-line broadband test tool. We would like to thank Johan Foldøy, our supervisor at the Norwegian Post and Telecommunications Authority, for providing the background information that formed the foundation of our thesis.

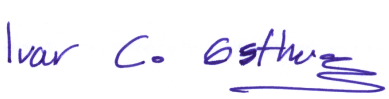
Poul Heegaard has been our supervising professor at NTNU and we would like to express our gratitude for all the feedback and guidance he has provided during the entire semester. Our weekly meetings have been of great value to us and all our discussions have helped us in maintaining a good perspective all along. In addition we appreciate the helpfulness with getting hold of the right people.

We would like to thank Jon Kåre Hellan from UNINETT for being helpful with access to a server in UNINETT's backbone network. This server became an essential part of our test setup.

Anders Solhaug from NextGenTel deserves thanks for the informative meeting we had in the early phase of our work. This meeting gave us insight in the challenges and opportunities that Internet service providers are faced with today.

Finally we would like to give thanks to Pål Sturla Sæther and Asbjørn Karstensen at the Department of Telematics for providing us with the hardware and software needed.

Trondheim, June 2009



Ivar Conradi Østhus

Lars-Petter Gunhildsberg Hansen

Summary

Many broadband subscribers suspect that they do not receive the data rate they are paying for. In order to verify that the broadband connection is compliant with the product purchased, subscribers can go on-line and choose between a myriad of available on-line broadband test tools with variable degree of precision. Today there exist no standardized methods to perform broadband evaluation for private subscribers.

We review and benchmark a selection of the available broadband test tools to reveal their strengths and weaknesses. Different tools have different approaches in their evaluation of network performance. Our studies show that most of the tools achieve acceptable accuracy for common Internet access data rates in Norway today. But when the data rate is increasing, the results from the different tools start to deviate. This is apparent for the upload rates in particular. The test methodology and the implementation technology are crucial for high bandwidth measurements.

The Norwegian Post and Telecommunications Authority will develop and release an on-line tool for evaluation of the end-users' Internet connections. We present the planned service and elaborate its possibilities and limitations.

Network neutrality is a concept that is quite ambiguous, and there exist many different interpretations. Based on the principles of network neutrality, developed by the Norwegian Post and Telecommunications Authority, we evaluate if the planned service is able to reveal breaches of network neutrality. We conclude that this is **not** possible with the planned service, mainly because of the limitations in the planned architecture combined with the complexity of network neutrality.

A broadband test tool should evaluate the quality of a broadband connection in context of its usage. We suggest a user profile scheme based on relevant services for different groups of users. Different services have requirements to different network characteristics, and this consequently determine what characteristics should be evaluated for each profile.

Lastly, we make use of our gained knowledge and recommend possible extensions and future applications for broadband evaluation.

Acronyms

ADSL	Asymmetric Digital Subscriber Line
AS	Autonomous System
ATM	Asynchronous Transfer Mode
BTC	Bulk Transfer Capacity
DNS	Domain Name System
DoS	Denial of Service
DPI	Deep Packet Inspection
DSL	Digital Subscriber Line
FPS	First Person Shooter
FTTH	Fiber To The Home
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IPTV	Internet Protocol Television
IRC	Internet Relay Chat
ISP	Internet Service Provider
IX	Internet Exchange Point
MSS	Maximum Segment Size
MTU	Maximum Transport Unit
NBTT	NPT Broadband Test Tool
NIX	Norwegian Internet eXchange
NPT	Norwegian Post and Telecommunications Authority
NRK	Norwegian Broadcasting Corporation
NTNU	Norwegian University of Science and Technology

PTS	Swedish Post and Telecom Agency
P2P	Peer-to-Peer
PSTN	The Public Switched Telephone Network
RED	Random Early Detection
RPC	Remote Procedure Call
RTCP	Real-time Transport Control Protocol
RTP	Real-time Transport Protocol
RTT	Round Trip Time
SLoPS	Self-Loading Periodic Streams
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
tc	Traffic Control
TBF	Token Bucket Filter
TCP	Transmission Control Protocol
ToPP	Trains of Packet Pairs
TTL	Time To Live
UDP	User Datagram Protocol
UiO	University of Oslo
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VoIP	Voice over IP
VPS	Variable Packet Size
WWW	World Wide Web

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The market for broadband Internet connections has exploded during the last few years and the majority of Norwegian households have access to a broadband Internet connection today [66]. The Internet service providers are continuously increasing the data rates of their broadband products in the fierce competition of the customers. Connection "speed" and price have become the most important selling points, and the operators are constantly trying to surpass each other. They all claim to deliver connections with a certain capacity, but the specifications are not uniform and often quite vague. "Up to" is a term that is commonly used by the Internet service providers when the capacity of broadband products are specified.

The usage of the private Internet connection has undergone a change as the typical contents communicated have developed from simple web pages to larger quantities of data and contents with real-time requirements. This development increases the users' demands related to the quality of the Internet connection. Customers are interested in getting the connection quality they pay for, and also be able to check whether this is the case or not. This raises a need for reliable broadband test tools.

1.1 Background and Motivation

The Norwegian Post and Telecommunications Authority (NPT) will develop and release an on-line tool for evaluation of the end-users' Internet connections. This work is inspired by the establishment of such a tool in Sweden where the Swedish Post and Telecom Agency (PTS) was participating.

There exists numerous on-line tools for broadband evaluation today, and this is a popular service provided by many different web pages. But NPT have expressed some concerns regarding the existing tools:

- *Most of these tools are financed by advertisement.* Commercial interests may affect how the test results are presented to the end-users. An example could be an Internet Service Provider (ISP) sponsoring the test tool, and therefore gain advantages.
- *The test servers are usually located in non-neutral locations.* The test server could be located in the network of one certain ISP. This may result in better test results for the end-users located in the same network as the test server, while users in other networks might experience relatively bad results.
- *The implementation quality of the tools varies.* Choices made during implementation of the tools may affect the test results significantly. This may again lead to wrong conclusions and mislead the end-users.

To overcome these concerns, NPT wish to establish a non-commercial tool with measurement points at neutral locations in the Internet. The new tool shall be released in 2009, and this work forms the basis for our thesis. We wish to study the possibilities and limitations of such a tool.

The numerous tools that already exist for broadband evaluation are based on quite a few different measurement techniques. They also do different assumptions and focus on different aspects of the broadband connection. We want to investigate the technological background for these tools, and also how these tools differ both in theory and in actual performance.

The most common tools focuses on how much data it is possible to push through the end-users' connections. We think there are other important aspects associated with a broadband connection in addition to the achievable throughput. Different users use different applications and thus have different needs. We want to study further possibilities for the broadband tools beyond what is provided today.

Network neutrality is a concept that has grown popular the last years. This is a concept that states how the networks constituting the Internet shall handle the traffic they carry. Network neutrality is a quite ambiguous concept, and there exist many different interpretations. NPT has developed guidelines describing the principles of network neutrality that many ISPs have endorsed. NPT as the telecommunication authority in Norway is interested in the possibilities of measuring how well the ISPs actions comply with the principles of network neutrality. Therefore we will evaluate a broadband test tool's capability of detecting breaches of network neutrality.

1.2 The Problem

As the problem description states, our assignment consists of the following tasks:

1. Study active and passive measurements techniques.
2. Review existing broadband test tool applications.
3. Specify measurement scenarios for broadband testing.
4. Testing of the measurement scenarios.
5. Discuss aggregation and presentation of measurement statistics.
6. Elaborate future broadband test tool applications

In addition to the aforementioned tasks we shall also investigate the concept of network neutrality, and the possibilities for an on-line broadband test tool to determine the network neutrality of an Internet connection.

1.3 Scope

We focus our work on private broadband subscribers. This is especially important to bear in mind when reading chapter 8 where we present user profiles. All the profiles, including the business profiles, are meant as user profiles for private customers' usage of their Internet connections. The target group of the broadband test tool NPT shall release is private broadband customers, thus we also focus our work on this group of customers. The business segment of the access market is much more complicated than the private market. The requirements to the specification and compliance of the service level agreement are usually strict and this raises demands for monitoring tools a bit more sophisticated than the ones we consider in this thesis. Therefore we consider the business market to fall out of scope for this thesis.

The target group of customers for the new NPT test tool give rise to another scope limitation. Throughout the thesis we focus on the Norwegian Internet scenario and the Internet infrastructure that exist in Norway. Typical broadband products and capacity specifications that are mentioned are based on the Norwegian market and the services offered there. We are aware of the fact that the Internet infrastructure in other countries may deviate significantly from the Norwegian scenario but we consider this as out of scope.

A last important scope limitation is related to the different bandwidth measurement techniques. The planned NPT on-line broadband test tool will be based on active measurement techniques, similar to much of the already existing on-line broadband test tools. Therefore we have chosen to focus on active measurement techniques in this thesis. We will not omit passive techniques from our presentation, but they will not be treated as thorough as the active techniques.

1.4 Approach to the Problem

The topic for our thesis is quite comprehensive and to get a good overview we will do a literature study and a presentation of the most important background theory. The study will consist of technological background for the Internet, measurement techniques for broadband evaluation and network neutrality. This work will prepare us for the following tasks and result in a written product that serves as a good theoretical background for the reader.

We will locate existing test tools to do a study of them and present the findings. A selection of on-line tools will undergo comprehensive testing. Each tool and its characteristics will be presented and we shall also give a discussion on the existing tools.

For the testing of the tools we have selected, we will make a test plan and execute the tests according to this. The results will be evaluated and the most important findings shall be presented in the report.

Since we are two persons working on this thesis, and many of the topics addressed are quite comprehensive, discussions are an important part of the work. We will continuously add discussions to our presentation in order to give a more balanced view of the current topic.

When presenting statements we will try as far as possible to support the claims with self-performed experiments and our own experienced data. When this is not possible, references will be used to support our statements.

A part of this thesis will contain our suggestions for further development and improvement of broadband evaluation tools. This part will be based on the knowledge we have acquired through the work with the thesis. Without getting into the formalities of epistemology, we can say that our approach is based on experimental learning.

1.5 Related Work

1.5.1 Measurement Lab

Measurement Lab is a partnership of the Open Technology Institute, the PlanetLab Consortium, Google Inc., and academic researchers [36]. They support and host tools that allow users to test their broadband connections. The tools shall both evaluate the performance as well as the transparency of the broadband connections. Measurement Lab is only at the beginning of its development. Measurement Lab continuously adds new tools and some of them were released early in spring 2009. Other tools will be released later.

When we started our thesis, only two tools were available:

- Network Diagnostic Tool
- Glasnost

We included both tools in our evaluation of existing tools.

1.5.2 Measurement Techniques

Extensive research has been carried out in the area of measurement techniques. We review and discuss the common active and passive techniques in chapter 5.

1.5.3 Existing Tools

There has been done extensive work on developing broadband evaluation tools. Most of the tools available today focus on data rates. We review a selection of these tools in chapter 6.

1.5.4 Our Contributions

Our work differs from the previous work because we analyze and benchmark the existing tools. We categories the tools and review which measurement techniques they use. By benchmarking the tools we get a good overview of what techniques that performs best in practice.

Throughout the thesis we will have a continuously discussion of NPT's planned service with regard to different important aspects of broadband evaluation, including network neutrality.

In this thesis we also suggest a user profile scheme based on relevant services for different groups of users. We believe this is a new approach to broadband evaluation not yet described in the literature.

1.6 Readers' Guide

We start our thesis by introducing NPT and the broadband tool they shall release in chapter 2. The establishment of this tool is the background for our thesis.

In chapter 3 the required technical background needed through the thesis is presented. This chapter constitutes a good reference for some of the technical challenges regarding broadband evaluation over the Internet. Readers with good knowledge of the protocols and functionality in the Internet probably do not need to read this chapter throughout, but it may provide a good perspective relevant for this thesis.

Chapter 4 introduce the concept of network neutrality. Because this is a relatively new and ambiguous concept we present our understanding of network neutrality from NPT's point of view.

We investigate what general measurements techniques that exists today in chapter 5. This is important to get a good overview of what techniques that exists,

how they perform the measurement, how the technique disturbs the network and what network characteristics they actually measure.

The presentation of measurement techniques from chapter 5 are followed up in chapter 6 where we present different existing tools that utilize some of the general techniques presented in the previous chapter. We have categorized the tools presented in two categories based on their implementation; on-line tools and stand-alone tools.

We have carried out a thorough evaluation of a selection of existing on-line test tools. Both the test setup and some of the results are presented in chapter 7.

In chapter 8 we have presented different groups of Internet users, and what requirements they have to their Internet connection. The tests performed and presented in chapter 9 shows some of the limitations in today's tools ability to serve different user groups with different requirements.

Measurement statistics is an important part of broadband tools. In chapter 10 we are evaluating how existing tools collect and present measurement statistics.

In chapter 11 we look into the future and present our ideas for further development of NPT's test tool and possible future application areas of such tools. We also present some possibilities for usage of statistics in the new NPT broadband tool.

Chapter 12 concludes the thesis.

Chapter 2

NPT and the Planned Broadband Test Tool

In this chapter we will shortly present the Norwegian Post and Telecommunications Authority and their responsibilities. Further we will introduce the planned online bandwidth measurement tool, which shall be released in 2009. This chapter will form a reference for the planned service which will be discussed throughout this thesis. We will only explain the planned functionality for the first release in this chapter. In chapter 11 we will discuss future possibilities for the planned service.

2.1 Norwegian Post and Telecommunications Authority (NPT)

NPT is an autonomous administrative agency under the Norwegian Ministry of Transport and Communications, with monitoring and regulatory responsibilities for the postal and telecommunications markets in Norway [48].

NPT possess many different responsibilities. In [49] NPT lists their current responsibilities:

- Monitoring of compliance with the legislation, regulations and licence requirements
- Supervising of telecommunications and postal services providers
- Supervising certification-service-providers issuing qualified electronic certificates
- Supervising registries assigning domain names under Norwegian country code toplevel domain
- Maintaining a register of telecommunications and postal services providers
- Preparing regulations
- Granting authorisations
- Product testing and approval
- Control of telecommunications terminals on the market
- Standardisation of telecommunications services
- Radio frequency management
- Number management
- Contingency planning and security
- International collaboration
- Advising the Ministry of Transport and Communications

2.2 NPT Broadband Test Tool (NBTT)

NPT will develop and release a tool for online bandwidth measurements during first half of 2009. Throughout this thesis, we will use the name NPT Broadband Test Tool (NBTT) for this tool.

In this section we will present the planned NBTT based on information provided by NPT.

2.2.1 Background and Motivation

Many broadband subscribers complain that they do not get the capacity they are paying for. Today there exists no standardized way to re-examine these complaints. Different providers might even state the provided capacity at different granularity which will make it almost impossible to compare different products.

The aim of establishing NBTT is to get a neutral, non-discriminating, non-commercialized broadband evaluation tool. The tool shall evaluate the different Internet subscriptions in the same way. From this the tool can gather statistical data about the actual measured data rates. This will allow end-users to compare different broadband products based on what is actually delivered. Hopefully this can force the ISPs to state their promised data rate at the same granularity.

Most ISPs today dimension their networks based on the assumptions that only a few customers use their full bandwidth at the same time. They also usually claim to deliver high bandwidth at low cost. A tool like NBTT can help both NPT and the ISPs to discover certain problem areas where the end-users do not get the promised bandwidth they are paying for. NBTT can also be used to reveal providers who consistently dimension their networks poorly.

2.2.2 Functionality

NBTT will be realized as a tool which runs within the end-user's web browser. The tool will establish a connection to a test server. The first version of NBTT will provide the following functionality:

1. Measure the download rate.
2. Measure the upload rate.
3. Measure the average delay.

A typical scenario where an end-user wants to evaluate his broadband connection is illustrated in figure 2.1. The user first request the page. The page, with the test embedded, is sent to the end-users web browser. Then the user can select "start test" which will start the test. The test will then be performed by exchanging data with a specialized web server. The final result is presented back to the user.

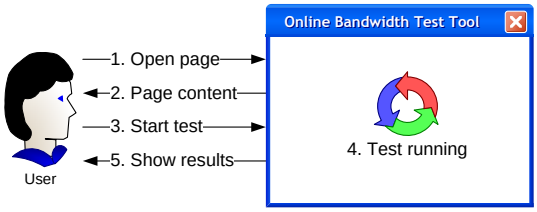


Figure 2.1: The figure shows a typical scenario where a user uses NBTT.

After the test is completed, the user will select what connection (technology and data rate) he uses and NBTT will evaluate whether the measured values is within

the expected range for this connection or not.

2.2.3 Technology

The measurement engine is developed by Ookla [55]. This engine is also used in Bredbandskollen, a similar service in Sweeden (see section 6.1.2), and Speedtest, a world wide bandwidth measurement service (see section 6.1.3).

The client side of the tool will be a Flash application which runs within the users' web browser. The server-side of the tool will be a specialized web server exchanging data with the client. The server will upload and download data from the client.

A database will contain both the raw result of each test as well as aggregated statistical information of interests.

2.2.4 Hosting

The NBTT tool will only be available at the Norwegian Internet eXchange (NIX) located in Oslo in the first release of the tool. NPT plans in a later release to host the service at various regional exchange points. We discuss and evaluate this possibility in section 11.1.1.

2.2.5 Statistics

The NBTT tool plans to collect much of the same statistics already done by Bredbandskollen. For each test performed there will be stored a database record consisting of:

- Date and time.
- The IP-address of the terminal.
- Measurement results (up- and down speeds, delay).
- The user's ISP.
- The geographical location of this IP-address.
- Browser and operating system used.
- The user's stated broadband capacity.
- An identification token connecting the measurement results to a browser cookie.
- An ID of the measurement server used.

We review the statistics provided by Bredbandskollen in section 10.1.3. In section 10.2 we discuss statistics for the planned NBTT service.

Chapter 3

Technical Background

This chapter introduces the technical background needed in the further reading this thesis. We will explain how the Internet is built up from a technical point of view, and how users connect to it. Important Internet protocols will also be introduced.

3.1 Internet

As the Internet have experienced a large growth the last years, the term "Internet" have become a part of everyday language. Our everyday life has become more and more dependent on this network called the Internet. As a reader of this report, you probably have a good technological understanding of the workings of the Internet. Much of the information in this section will therefore be well-known. But some concepts presented here might be useful in the further reading of the report. To be able to design a good broadband test it is important to know how the Internet works and how users connect to it.

3.1.1 Internet Terminology

An internetwork is defined as a collection of interconnected networks [68]. "The Internet" is a global internetwork that utilizes the TCP/IP protocol suite in the transmission and exchange of data. "The Internet" is an internetwork, but the opposite is obviously not necessarily true.

The terms "the Internet" and "World Wide Web" are often mistakenly mixed up, but they refer to two different aspects of the global data network. The Internet is the hardware and software infrastructure that provides connectivity between computers distributed all over the globe, while the World Wide Web (WWW) on the other hand is one of the services that are communicated over the Internet.

In the realization of the Internet, there are several different actors that serve different functions.

- **End-user** - The user that subscribe to an Internet connection from an ISP.
- **Internet access operator** - The company that owns and manage the access network used to deliver the Internet connection to the end-user.
- **Internet service provider** - The company that offers Internet subscriptions to their customers. The ISP may or may not own the access network that the connection is delivered on. If not, the ISP must lease capacity in the access network from an access operator.
- **Content provider** - The company that provides content over the Internet. Examples of content are newspapers, video services, community web pages etc. The number of content providers is endless.

3.1.2 Interconnecting Autonomous Systems

The Internet consists of a large amount of connected networks. The different networks are owned and controlled by the ISPs. The network of an ISP is often referred to as an Autonomous System (AS) [74]. To achieve connectivity the ISPs must connect their users to their network through an access network. The different ISPs must also connect their networks to all the other ISPs' networks to make an all-to-all communication possible, as shown in figure 3.1.

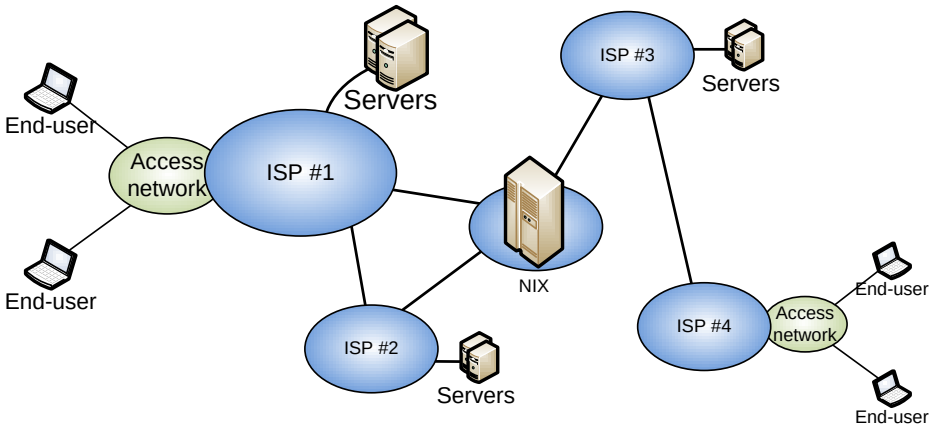


Figure 3.1: Interconnection of ISPs by peering and transit [46].

Some ISPs own their own access network. In Norway, most of the copper local loop is owned by Telenor of historical reasons. But because of the Local Loop Unbundling regulations, Telenor must oblige any reasonable request from ISPs that wish to lease access over the local loop. Therefore an ISP may provide Internet connection to the end-user without owning the access network used. Many of the xDSL-providers in Norway operate in this fashion, e.g. NextGenTel, Tele2, Ventelo etc. We will have a brief look at different types of access networks in section 3.1.3.

The Internet is based on the principle of global connectivity which means that any Internet user can reach any other Internet user as though they were on the same network. Hence in addition to connect the users to their networks, the ISPs must also interconnect their networks. Since the number of ISPs on the Internet are overwhelming, an all-to-all connection scheme would be extremely expensive, complicated and certainly impossible to implement. To solve this, the ISPs use a combination of peering and transit based on direct lines or Internet exchange points.

Peering refers to the case when two ISPs agree to mutually exchange Internet traffic. In this way it becomes possible for subscribers at the two different ISPs to communicate with each other. Peering does not include the obligation to carry traffic to third parties [41]. Peering requires a physical link between the two ASs and exchange of routing information. The physical link between two ISPs may be either a direct link, or a connection through an Internet Exchange Point (IX) [46]. Peering over a direct link are often called "private peering", while peering over an IX are referred to as "public peering" [79]. In figure 3.1 ISP #1 and ISP #2 are doing private peering, while ISP #1 and ISP #3 are doing public peering through the NIX. A peering agreement usually does not involve any kind of cash flow between the two ISPs. Both ISPs gain from the availability to each other's subscribers and the two ISPs consider each other as peers as long as the traffic pattern is fairly symmetrical. But if the traffic pattern becomes very asymmetrical, the peering might not be as beneficial for the ISP that receives the

most traffic. Therefore a peering agreement usually includes some description of the traffic symmetry. If the real traffic does not comply with this symmetry over a period of time, the peering agreement will probably be renegotiated in a way that is more beneficial for both parts.

Transit, a kind of billed peering [46], is an alternative to peering in the cases when two ISPs do not consider themselves as peers. An ISP can pay another ISP to provide connection towards the rest of the Internet. In this case, the one ISP buys transit from the other ISP. In figure 3.1 ISP #4 buys transit service from ISP #3. In this case ISP #3 transports the traffic originating or targeted at the rest of the Internet on behalf of ISP #4's customers. In most cases, the transit ISP carries traffic to and from its other customer ISPs, and to and from every destination on the Internet, as part of the transit arrangement [41]. This is an important difference from regular peering that usually does not include carrying traffic to a third party.

Internet exchange points are vital in the interconnection of the Internet. In Norway the Internet exchange points are called NIX and consist of 6 separate points of intercommunication; Trondheim, Tromsø, Bergen, Stavanger and two points in Oslo [51]. NIX usually refer to the points in Oslo (NIX1 and NIX2), but it is also used as a collective term for the six points of intercommunication distributed in the largest cities in Norway. NIX1 located at the University of Oslo is the point with by far the most customers connected, and most of the traffic handled by the NIX points are routed through this point [23].

The IX is a neutral point of intercommunication where different ISPs can connect their networks, and acquire connections to other ISPs networks. This simplifies the structure of the interconnection graph, since the IX will serve as a kind of a hub in the internetwork. Each ISP that wishes to connect itself to the IX must pay a yearly fee [46]. Then it gains access to a neutral medium that can be used to share routing information with all the other ISPs that it wants to peer with. A peering agreement is still needed, but with this arrangement the ISP does not need a direct link to each of its peers.

The different ISPs only manage their own network. One ISP has no control of the management of the other ISPs' networks. Therefore an ISP can only guarantee the quality of service within its own network. As we can see from figure 3.1, there are shown end-users both at ISP #1 and ISP #4. If the end-users at ISP #1 have paid for an Internet connection with a certain bandwidth, they have no guarantee that they may utilize this bandwidth when they are communicating with the end-users at ISP #4. There might be bottlenecks at the path between these end-users, or other problems that degrade the performance of the connection. If these problems exist within the network of ISP #4, then ISP #1 cannot do much to remedy. This is an issue that is crucial when it comes to bandwidth measurements. If a bandwidth measurement tool uses a test server placed in the network of an ISP with a troublesome network, then customers of another ISP may get false indications that their Internet connection does not comply with the specification. For this reason, a bandwidth test server should preferably be placed on a neutral location in the network. This is some of the motivation for the NBTT which NPT wish to place at the points of interconnection in Norway.

3.1.3 Connecting End-Users to the Internet

When a customer decides to buy an Internet connection from an ISP, the ISP need to make sure the customers equipment is physically connected to the ISP's network. This is the task of the access network. The access network constitutes the ISPs metro network in combination with different last mile technologies. Most ISPs provides services delivered over different last mile technologies. These technologies have quite different characteristics when it comes to data rates, and we will have a brief look at the different technologies in this section.

In table 3.1 we have collected some typical data rates for the most common last mile technologies. The numbers are based on the services delivered by the largest ISPs in Norway today. These numbers[1] are presented only to illustrate the incredible large span in data rates for different technologies. This is also interesting when it comes to designing a good bandwidth test tool. From the table, we can see that a bandwidth measurement tool needs to support Internet connections with rates in the interval 50 - 100 000 kbit/s to cover the most common access technologies.

Access Technology	Transmission medium	Typical bitrate, kbit/s download	Typical bitrate, kbit/s upload
Dial-up	Twisted pair	< 56	< 56
ISDN		< 64	< 64
ADSL		500 - 20 000	250 - 1000
VDSL2		30 000 - 40 000	5 000 - 20 000
Cable TV	Coaxial cable	1 000 - 30 000	750 - 2000
FTTH	Fiber cable	10 000 - 100 000	10 000 - 100 000
GPRS/EDGE	Wireless	100 - 200	50 - 75
UMTS		< 7 200	< 2 000
WiMAX		600 - 5 000	600 - 5 000
Wi-Fi*		< 54 000	< 54 000

* The bitrate of the Wi-Fi based Internet connection is dependent on the rate of the feeder network. 54 Mbit/s is just the theoretical maximum for the 802.11g standard.

Table 3.1: A comparison of typical data rates for different access technologies.

[1]The numbers in table 3.1 do not present the theoretically achievable data rates for the respective technologies, but rates commonly provided by Norwegian ISPs.

Dial-Up/ISDN These two access technologies uses the telephone network to provide Internet access. The user must dial-up the servers of the ISP, and are charged for the duration of the connection. The user-end of this connection usually consists of a modem, either external or built into the user's computer. These technologies are not so much used by private Internet end-users in Norway today. Their usage was more common during the 1990s and early 2000s. The massive roll out of Asymmetric Digital Subscriber Line (ADSL) in Norway during the early 2000 made most dial-up and ISDN connections obsolete. But despite the migration to broadband connections, there still exist a considerable number of Internet end-users with this type of connection [66].

Digital Subscriber Line (DSL) This is the most common access technology for broadband subscribers in Norway today [66]. Like Dial-Up/ISDN, the DSL technology also utilizes the telephone network to provide Internet access. DSL can provide much larger data rates than dial-up and ISDN because of more efficient utilization of the frequency spectrum. The most common type of DSL service in Norway is the ADSL service. This type of DSL divides the frequency spectrum asymmetrically, and uses the larger part for downstream traffic. Therefore this kind of services usually has a download data rate that is several times the upload rate. There are many different versions of the DSL technology, e.g. SDSL, ADSL, ADSL2, ADSL2+, VDSL and VDSL2. The main difference between these technologies is the allocation of the frequency spectrum, and hence they need different hardware at the endpoints of the connection. From table 3.1 we can see that VSDL2 is the DSL technology that provides the highest data rates today. One problem with the DSL technology is that as the distance from the call office increases the theoretical maximum data rate decreases. At the end-user side of a DSL connection, there must be a DSL modem, and these are usually external devices that are owned and managed by the ISP.

Cable This type of Internet access does the same thing with the cable-TV network as DSL does with the telephone network; it utilizes the unused frequency spectrum of an existing network to provide a data transfer service able to transport IP packets. A customer may buy an Internet access of this type from the cable-TV provider that operates in the area of interest. In Norway this service is provided by some of the cable-TV operators, e.g. Canal Digital and Get. The customer will have a cable modem installed on the user side of the connection, and as with DSL, this modem are usually owned and managed by the provider. The main difference between DSL and cable Internet is the underlying physical medium. Cable-TV uses coaxial cable, while DSL uses twisted pair. The coaxial cable can in general carry a larger bandwidth than the twisted pair, and it has better shielding which makes it more immune to signal noise [68].

Fiber To The Home (FTTH) Networks based on optical transmission are quite common in the backbone network, but still quite rare in the access part. This access technology cannot benefit from an already existing network like the DSL and Cable Internet can. Therefore it requires establishment of a new physical network based on optical fibers. This development is still in the initial phase

in Norway, and only end-users located in certain areas can choose this access technology today. The FTTH technology are still under development [25], and it is likely that this type of Internet access will be common in the future. The main advantage with this access technology is that it can support data rates many times the theoretical maximum of both DSL and cable Internet. At the user side of this type of connection there must be hardware capable of translating between the electrical and optical domain.

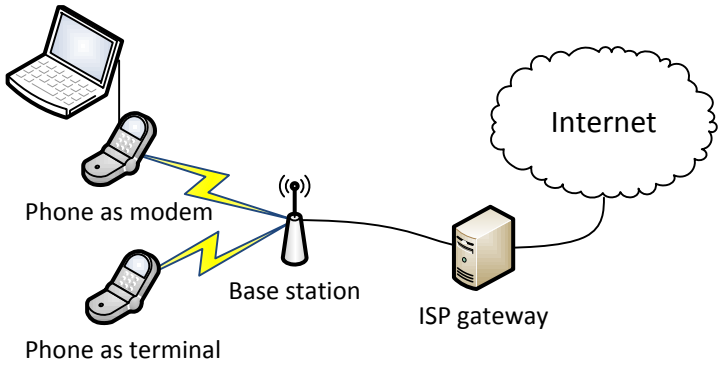


Figure 3.2: Simplified scenario with GPRS/EDGE/UMTS based Internet access.

GPRS/EDGE/UMTS These are mobile access technologies, and most mobile phones in use today support one or more of them. GPRS and EDGE are supported in the GSM network, while the UMTS network has its own standard for packet transmission. These technologies enable the end-users to be connected to the Internet while being mobile. Compared to the other access technologies presented here, the bandwidth of these mobile technologies is rather modest. But the increased mobility still makes these technologies attractive for end-users who need to be online everywhere. Figure 3.2 shows two typical user scenarios. The mobile phone can be used as a terminal alone, and the user accesses the Internet through the user interface of the phone. Another possibility is to use the telephone as a modem to connect a computer to the Internet. Many of the mobile operators also provide mobile Internet solutions with a dedicated modem that are able to connect to the mobile telephone network.

WiMAX This is an access technology that is used to provide Internet access in rural areas where other cable based alternatives are not available [43]. It can also be used to provide mobile broadband, as an alternative to the mobile technology presented in the previous paragraph. These two scenarios are shown in figure 3.3. Another possible usage for this technology is as a feeder network for a phone central which again provides broadband access over ADSL [43]. For fixed WiMAX the reach is typically around 20km, while for mobile WiMAX the reach is reduced to about 2-3 km. The end-users on WiMAX based broadband connections have a bandwidth that is larger than a typical UMTS user, and closer to a low-end DSL connection. Some of the Norwegian ISPs deliver WiMAX based broadband today in areas where DSL cannot be delivered.

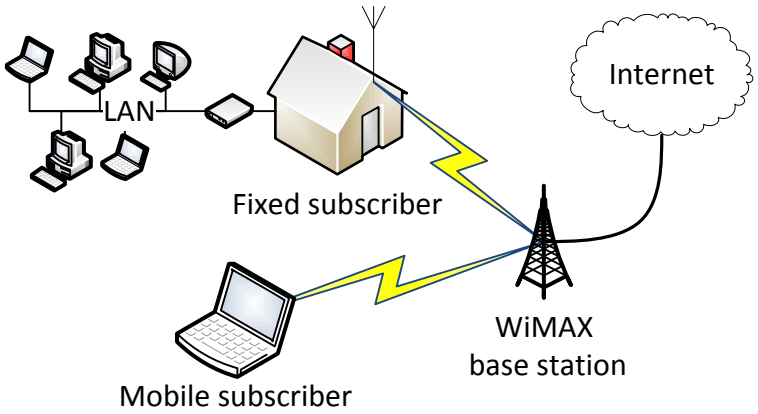


Figure 3.3: Simplified scenario with fixed and mobile WiMAX broadband subscribers.

Wi-Fi This is a technology that is much used in wireless local area networks, and this technology is supported in every laptop computers sold today. It has also become more common that mobile phones and other "gadgets" also support this technology. Although one would normally classify this as a local area technology, it can also be considered as an access technology under certain circumstances. If the end-user connects to an Internet hot spot, then the owner of the hot spot suddenly becomes the ISP that provides Internet connectivity to the end-user. And then, the wireless network may be viewed as a sort of access network. In table 3.1 the bandwidth of a typical Wi-Fi end-user is showed as maximum 54 Mbit/s. This is the theoretical maximum transfer rate for the 802.11g standard. But for a real Wi-Fi end-user, the bandwidth would be totally dependent on the bandwidth of the feeder network and its Internet connection. If the Internet hot spot has a 100 Mbit/s feeder network connected to a 100 Mbit/s Internet connection, and there are little cross traffic in the network, then the end-user might get at bandwidth that is closing up on the theoretical maximum of the wireless network. But the bandwidth acquired in such networks is usually significant lower than this theoretical maximum.

3.2 TCP/IP – Protocols and Layers

TCP/IP is the Internet protocol suite and is composed of a set of communication protocols used in IP-based data networks. Transmission Control Protocol (TCP) and Internet Protocol (IP) gave name to this protocol suite since they are the two most important protocols, and also the two protocols that were defined first in this standard.

The Internet Protocol Suite may be viewed as a set of layers as shown in figure 3.4. Each layer provides different sets of functionality needed in the action of transmitting data. Each layer also provides a well-defined service to the upper layer protocols based on the services from the layer beneath.

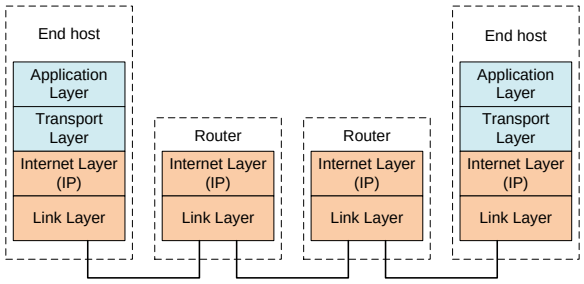


Figure 3.4: Internet protocol layers.

- **Application layer** – This layer contains all protocols that are concerned with process-to-process communication. Examples of application layer protocols are Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), BitTorrent, Simple Mail Transfer Protocol (SMTP) and Secure Shell (SSH). The protocols at the application layer depend on the underlying transport layer to establish host-to-host connections. The interface between the application and transport layer is defined by port numbers and sockets.

- **Transport layer** – This layer provides end-to-end data transfer. The transport layer is designed to allow peer entities on the source and destination host to carry on a conversation. There are two important transport protocols defined in TCP/IP, namely TCP and User Datagram Protocol (UDP). TCP is a reliable connection-oriented protocol that allows a byte stream originating at one host to be delivered without error on any other host on the Internet. UDP on the other hand is an unreliable, connectionless protocol that provides a "best-effort" service. We will examine TCP and UDP in greater detail in section 3.4 and 3.5.

- **Internet layer** – This layer provides a virtual network image of the Internet, and hides the physical architecture beneath from the transport layer above. IP is the most important protocol for data transmission in this layer. IP is a connectionless protocol that does not require any level of reliability from the lower layer protocols whose services it uses. The service provided by the IP protocol is classified as "best-effort". This means that it does not provide any reliability, flow control or error correction. This functionality must be provided by the transport layer if required. Routing of packets through the network to the correct receiver is an important functionality provided by the IP protocol. We will take a closer look at the IP protocol in section 3.3.

- **Link layer** – This layer is the interface to the actual network hardware. TCP/IP does not specify any protocols here, but can use almost any network interface available. This illustrates the flexibility of the TCP/IP protocol suite. Examples of link layer protocols are Ethernet and Asynchronous Transfer Mode (ATM).

3.3 Internet Protocol

IP is the main protocol in the Internet layer of the TCP/IP protocol suite. This protocol serves two main purposes in the Internet. The first is to deliver packets from a source host to a destination host based on the receiver address. The second is to provide fragmentation and reassembly of packets to support data links with different Maximum Transport Unit (MTU) sizes [57].

Data from an upper layer protocol are encapsulated in packets, and sent as self-contained data units. The encapsulation of packets makes it possible to use IP over a heterogeneous network consisting of a lot of different link layer technologies [78].

The operation of IP is connectionless, and each router forwards the IP-packets based solely on the destination address. The service delivered by IP is said to be "best-effort". This means that the delivery of each packet is not guaranteed, and IP does not implement any functionality to increase the reliability. Because of this, there are several faults that may occur during transmission of packets:

- Packet loss.
- Packet corruption.
- Packets delivered out of order.
- Packet duplication.

If the application that transmits the data cannot tolerate these faults, it should use a transport protocol that masks the errors. TCP is the most common choice.

Routing and addressing are two important parts of the IP-protocol. Routing is the process of building and maintaining the routing tables in each node. There are own protocols defined to handle this task. Addressing refers to how nodes and networks are assigned addresses. Both these subjects are quite comprehensive, and they will not be further presented since we consider them to fall outside the scope of this thesis.

3.4 Transport Control Protocol

In this section we will look at the most important transport protocol used in Internet today, namely TCP. TCP is widely used by many of the most popular applications and the application protocols these are based upon. Some examples are WWW, e-mail, FTP and SSH. TCP dominates the Internet traffic, and it has some special features that we need to be aware of if this protocol shall be used for bandwidth measurements. In this section we will briefly explain TCP and its workings before looking into the features of congestion control. For a more detailed explanation the reader is referred to [59] and [39].

3.4.1 TCP – Protocol Overview

TCP provides a connection oriented service on top of the connectionless service provided by the IP layer. The TCP service takes care of partitioning the data into chunks that fit the underlying packet format, makes sure the receiving application gets the data in the same order they were sent, retransmits data that are lost by the "best-effort" IP service and adjust the sending rate to a level acceptable both by the receiver and the intermediate network. A protocol with all this responsibility is bound to be rather complex and this is also the case with TCP.

TCP focuses on accurate delivery rather than timely delivery. Retransmission of lost or corrupt segments might introduce some delay, since all segments must be received in order. TCP is used in every application that needs the transferred data to be absolutely correct, for instance file transfer and e-mail. Real time applications that can tolerate some loss, but are depending on timely delivery would typically rather use a more light weight transport protocol like UDP.

The sender needs to adjust the sending rate to fit the minimum of what the receiver and the network can manage. The receiver advertises a receiver window to inform the sender about its buffer size. This will always be the maximum amount of data that can be sent before the sender must stop and wait for an acknowledgment from the receiver. The network itself and the traffic it carries is another factor that must be accounted for. The sender maintains a congestion window that is used to adjust the sending rate to the available bandwidth in the network. We will have a look at the algorithms used to maintain a congestion window. These algorithms serve two purposes:

- Prevent the sender from overwhelming the network and receiver with too much data.
- Enable the sender to efficiently utilize available bandwidth and react to the dynamic changes in available bandwidth.

3.4.2 Slow Start and Congestion Avoidance

When a TCP connection is established, the sender sets the congestion window to be the maximum size of one segment. The threshold value is initially set to a value, e.g. 64KB. This threshold is increased as the congestion window increases. When a timeout occurs, the threshold value is set to the half of the current congestion window. The sender starts sending the amount of data indicated by the congestion window before it waits for an acknowledgment from the receiver. If an acknowledgment is received before the timeout, the receiver doubles the congestion window, and sends this new amount of data before it again waits for an acknowledgment. This process continues and for each acknowledgment the sender gets in return, the congestion window is doubled. This algorithm is called slow start, even though it actually is an exponential growth in the congestion window [68]. This effectively increases the sending rate, and the congestion window keeps growing until one of three possible events occur:

1. The receiving window is reached. Then the slow start algorithm stops and the congestion window remains constant as long as no timeout occurs.

2. The threshold value is reached. This causes the slow start algorithm to stop, and an algorithm called congestion avoidance takes over. This algorithm is not that aggressive as slow start and it increases the congestion window with one segment size for each acknowledged burst. This is shown in Figure 3.5.

3. A timeout occurs. Then the current threshold value is set to half the current congestion window. What happens next depends on the TCP version. Either a new slow start is started with the congestion window reset, as shown in Figure 3.5 or the congestion window is halved and congestion avoidance with linear growing congestion window is restarted.

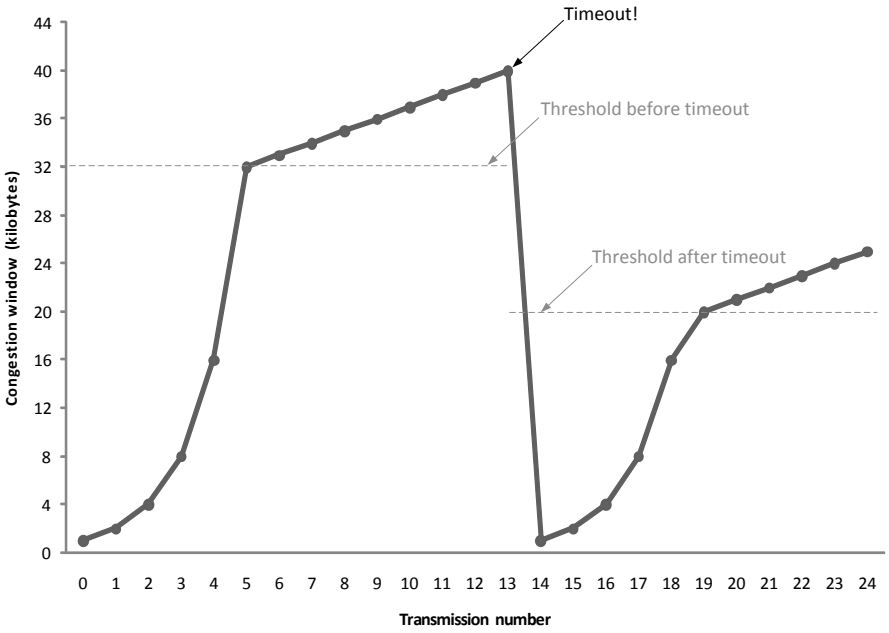


Figure 3.5: Example of congestion window during slow start and congestion avoidance [68].

As we can see from Figure 3.5, a TCP connection need some time to adapt the sending rate to the available bandwidth. This is very important to be aware of if a TCP connection is to be used for the purpose of bandwidth measurement. If we include the transient slow start period in the calculation of the bandwidth we will get lower results than when excluding this period from the calculations. If we wish to measure available bandwidth with one or more TCP connections it is important to give the connection(s) time to acquire the available bandwidth before the measurement starts. The experiment which is presented in section 6.1.7 and illustrated in figure 6.12 emphasizes this point.

3.5 User Datagram Protocol

UDP is one of the core transport protocols used in the Internet. This protocol enables a datagram mode of packet-switched communication. Applications can use UDP to send messages, also known as *datagrams*, to other applications with a minimum of protocol mechanism. The UDP is assumed used over the IP. Because UDP does not implement any bandwidth adaption or congestion control, the applications using UDP needs to take care of these operations themselves. They need to do this to prevent overloading of the network. This is why it is important for applications to be able to estimate the available bandwidth when using UDP as a transport protocol. The interested reader is referred to [56] where the protocol is formally defined.

3.5.1 UDP – Protocol Overview

UDP is a very simple protocol which does not guarantee reliability, ordering of packets or data integrity. The service provided by UDP is classified as a connectionless unreliable service where packets may arrive out of order, disappear or arrive duplicated. Applications that want to send a datagram to another application can do so without prior communications to set up special transmission channels. Error checking is usually not done at the network level, and only done by the application if needed.

The UDP does not implement any congestion control itself, like TCP. This basically means that a application utilizing UDP might consume all available bandwidth on the link in the case of congestion (and TCP will gradually back off). This effect is illustrated in figure 5.5.

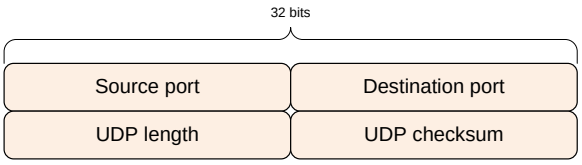


Figure 3.6: The UDP packet header.

UDP is basically IP with a short 8 bytes header added. The header is shown in figure 3.6. As seen in the figure, the header consists of two ports. The destination port represents where to deliver the datagram at the receiver. If the sender needs to receive an answer, the receiver can reply to the source port. The UDP length field defines the length of the payload. The UDP checksum field is optional and must be handled by the application.

3.5.2 UDP – Protocol Application

UDP is often used by time-sensitive applications such as Voice over IP (VoIP) and Internet Protocol Television (IPTV) where delayed packets are useless and dropping is preferred over retransmission. UDP is also commonly used by network

applications such as Domain Name System (DNS) and Remote Procedure Call (RPC).

3.6 Real-Time Transport Protocol

As mentioned in section 3.5, UDP is commonly used for time-sensitive applications such as audio-on-demand, video-on-demand, videoconferencing and other multimedia applications. As these applications emerged, people discovered that each application was reinventing more or less the same real-time transport protocol [68]. This resulted in the a more generic real-time protocol, Real-time Transport Protocol (RTP), first described in RFC 1889 published in 1996 [26] and later superseded by RFC 3550 in 2003 [27].

3.6.1 RTP – Protocol Overview

RTP uses primarily UDP to transport the information (but other transport layer protocols may be used). The packet nesting is shown in figure 3.7. As seen in the figure, RTP adds some extra information to the packets, and sends them over the network as regular UDP packets. The protocol does not provide any quality-of-service guaranties or ensures timely delivery. RTP includes sequence numbering which enables the receiver to reconstruct the sender's packet sequence.

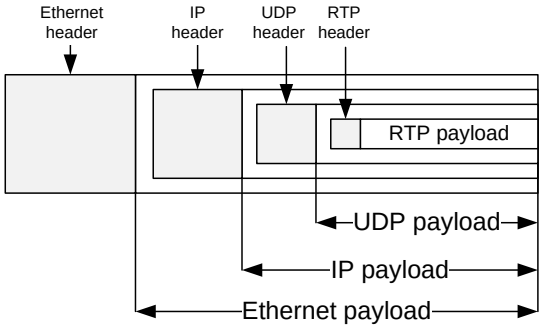


Figure 3.7: RTP packet nesting [68].

3.6.2 Congestion Control

All transport protocols needs to address congestion control in order to not let one stream consume all available bandwidth and RTP is no exception. The data transported over RTP is often inelastic (generated at a fixed or controlled rate) [27]. This reduces the risk of one RTP to consume all available bandwidth, as a TCP stream can. Because of the inelasticity the stream cannot reduce its load on the network arbitrary and the congestion control mechanism must be adapted to the specific application of RTP.

3.6.3 Real-Time Transport Control Protocol

Real-time Transport Control Protocol (RTCP) is usually used to monitor the quality of service and to convey information about the participants in an on-going session [27], while RTP transport the actual media streams (e.g. audio and video).

3.7 Hypertext Transfer Protocol

The Hypertext Transfer Protocol (HTTP) is one of the most important application level protocols in the Internet. We will present this protocol because of the extensive use of this protocol in existing on-line bandwidth measurement tools.

HTTP is the protocol used in the information exchange taking place when end-users requests content from web servers. HTTP takes care of the communication between the web browser and the web server, and all types of data are delivered through this protocol. The HTTP protocol is a general protocol that in addition to transporting hypertext also have other areas of applications [21], e.g. as communications protocol for SOAP messages when accessing *Web Services*. In this section we will limit the presentation to the browser-to-server communication scenario.

3.7.1 HTTP – Protocol Overview

HTTP is a client-server communication standard that follows the request/response message exchange pattern [21]. The protocol operates on *resources*, which is data objects or services that can be identified by a Uniform Resource Identifier (URI). HTTP requires a reliable transport protocol and are therefore implemented over TCP. But any protocol providing reliable transport could in theory be utilized [77].

The client which initiates a HTTP request are called *user agent*. On the other side of the communication channel is the *server* which is an application program that accepts connections in order to service requests by sending back responses [21]. A typical user agent is the web browser software running on the end-users computer. When the user wishes to fetch a resource from a web server, he instructs the browser to issue a HTTP request either by entering a Uniform Resource Locator (URL) in the address field, or by clicking on a hyperlink which contain an URL.

The HTTP protocol provides a set of methods that can be applied to a resource. Some of the most important methods are presented in table 3.2.

3.7.2 HTTP Request

The HTTP request have the following format:

Method	Description
GET	Request to load a Web page
POST	Append to a named resource (e.g. a Web Page)
HEAD	Request to load a Web page's header
TRACE	Echo the incoming request
OPTIONS	Query certain options

Table 3.2: Some of the most important HTTP request methods [68].

```
method request-URI HTTP-version
[headers]

[message-body]
```

The method field defines what operation that shall be done on the resource identified by the request-URI. The HTTP-version is also included. The request-header fields allow the client to pass additional information about the request and the client itself to the server [21]. One example is the User-Agent header, which allows the client to inform the server about its browser, operating system and other properties [68]. There are a lot of headers defined, but we will not present them here. For a complete description of all the headers, the reader is referred to [21]. The message body contains the payload of the HTTP request. The message-body can contain the data posted to a web page when the POST method is used.

3.7.3 HTTP Response

The HTTP response have the following format:

```
HTTP-version status-code reason-phrase
[headers]

[message-body]
```

The first line of a response message consist of the HTTP-version, a code reflecting the status of the response combined with a textual description of this status. The code is meant for interpretation by software, while the textual reason phrase are meant as a feedback to the human user [21]. This status shows whether the request was successful or not. It also identifies the problem if the request is not successful.

As with the request-headers, there are also numerous different response-headers. One of the headers we consider as important is the Cache-Control header. This header can be used to determine whether the response can be cached or not. If the time used to fetch a resource through HTTP is used to estimate the bandwidth, then it is crucial that the resource is fetched from the web server each time, and not from an intervening cache. We will not look further into the response headers here, so the reader is referred to [21] for a complete description of these.

3.7.4 HTTP in Action

Figure 3.8 shows a simple HTTP interaction. In this example, we have used telnet to issue the HTTP GET request and get the response written out on the screen.

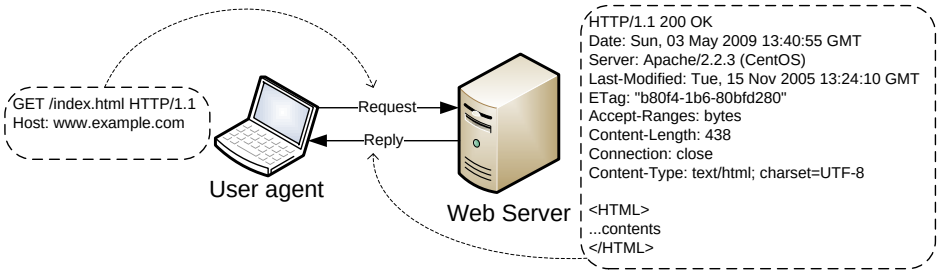


Figure 3.8: Example of a GET request and the consecutive response from the web server.

Chapter 4

Network Neutrality

The concept of a neutral Internet has been widely debated during the last few years. Some actions by the network operators have fueled this discussion. Network operators who have tried to discriminate traffic based on application or throttle traffic from one content provider in favor of another have made more people aware of the idea of network neutrality. Network neutrality is an ambiguously specified concept. Many different definitions exist, and much has been written on this subject. In this chapter we will present network neutrality and discuss some important aspects associated with this concept.

The inventor of the World Wide Web, Tim Berners Lee, expressed the following definition of network neutrality [8]:

> *"If I pay to connect to the Net with a certain quality of service, and you pay to connect with that or greater quality of service, then we can communicate at that level."*

In addition to say what net neutrality is, he also mentions what it is not. Network neutrality has nothing to do with free Internet access. Neither does it disallow customers to pay for different levels of quality of service. We agree with this interpretation.

4.1 Inherited Neutrality

Internet is based on the idea of a "stupid" core and "intelligence" placed at the edge nodes, i.e. in the end users computer [31]. The task of the Internet is to transport the bits and bytes from one end to the other end based on a receiver address. This is done without any regard to the semantics of the actual data being transported. A network that treats all traffic equally independent on contents or higher level protocols can be said to be neutral. This traditional mode of operation causes the network to be neutral with regard to traffic discrimination. In reality, the networks that constitute the Internet operates in a much more complex manner, and this inherited neutrality can no longer be taken for granted.

4.2 Principles of Network Neutrality

Discrimination is a keyword in the concept of network neutrality, but this concept has also been extended to cover other aspects of network provisioning as well. In Norway, NPT have composed a set of principles for network neutrality that they want the providers to agree on [47]:

1. Internet users are entitled to an Internet connection with a predefined capacity and quality.
2. Internet users are entitled to an Internet connection that enables them to:
   - Send and receive content of their choice.
   - Use services and run applications of their choice.
   - Connect hardware and use software of their choice that do not harm the network.
3. Internet users are entitled to an Internet connection that is free of discrimination with regard to type of application, service or content or based on sender or receiver address.

The first principle states that the properties of the Internet connection shall be agreed upon. The customer shall receive clear information on the capacity and quality of the Internet connection. In the case of Internet connection delivered on the same physical link as other services, the customer is entitled to information about how the resources are split between the services and how this affects the Internet connection.

The second principle states that the Internet connection must be possible to use in the way the end user wishes, as long as the activity is legal. One of the properties with the Internet technology is that any type of communication that is based on the Internet standards is supported. This property will be compromised as soon as the network operators start deciding what the customer can and cannot do with their Internet connection. Blocking of illegal or malicious traffic is an exception that is **not** considered to be in conflict with the principles of network neutrality.

The third principle states that the traffic carried over the Internet connection shall be transferred in a non-discriminating way. This means that every stream of data shall be treated equally, and not be discriminated based on application type, service type, content type and sender or receiver.

In case of congestion, it can be necessary to drop packets. In a neutral network, this dropping need to happen in a "fair" way without discrimination of customers or individual data streams. But since different customers often have paid for different capacity, there will probably also be different opinions on what is "fair" when it comes to packet dropping. A customer with a large capacity Internet connection will probably have much more packets in transmission than another customer with a small capacity Internet connection. If the network in a given time interval randomly drops 5 % of all the packets, the customer with the high capacity will lose more data than the customer with low capacity. In this case no discrimination has happened, but the high capacity customer will probably think that this scheme of degradation is unfair. Lacking a clear definition of "fairness", this third principle expresses that no unreasonable manipulation or degradation of individual data streams shall be done.

4.3 Breaches of Network Neutrality

Breaches of network neutrality happens as soon as the ISP begins to deliberately discriminate lawful traffic by degrading some content to prioritize another. This can be done in many ways, dependent on what the ISP wishes to accomplish. Limiting traffic from specific IP-addresses can be effective to degrade or even block content from specific content providers. An ISP may block all traffic going to/coming from a specific IP-address. This is done today to prevent customers from accessing illegal Internet sites, and this is not considered a breach of network neutrality. But if these means are used to block lawful hosts, then it becomes more problematic. Throttling traffic[1] based on TCP/UDP-port is one way of limiting the bandwidth consumption by specific applications. This approach requires an infrastructure that is able to read and act upon the transport protocol header in the packets, and this operation is expensive for large traffic volumes. It is also possible to use even more expensive solutions to filter out unwanted traffic based on the actual content in the packets and not only the headers. This is called Deep Packet Inspection (DPI). All these means will result in a breach of network neutrality when applied to lawful Internet traffic. This has already happened several times, and we will look closer at two concrete examples below.

4.3.1 NextGenTel vs. NRK

One of the most discussed examples of breach of network neutrality in Norway is the "NextGenTel vs. NRK" case from 2006. This specific case shows some of the problems arising when an Internet provider begins limiting the activities of the user.

[1] Throttling of traffic denotes the act of deliberately limit the bandwidth consumed.

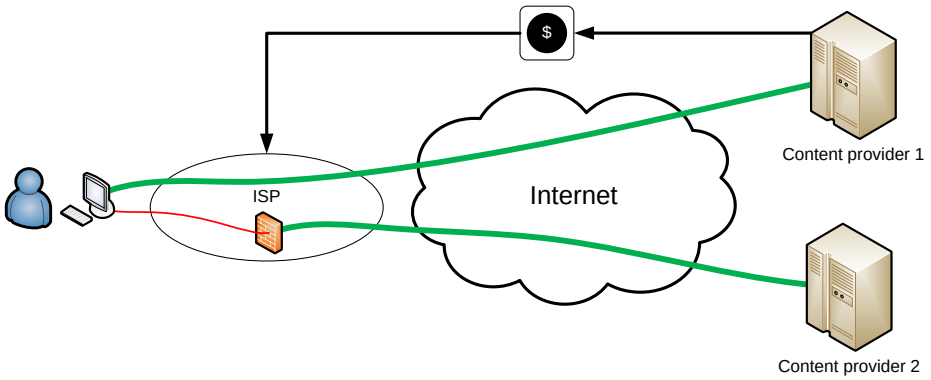


Figure 4.1: Example of a non-neutral network: The ISP is paid by content provider 1 to distribute their content. Content delivered from content provider 2 which does not pay the ISP is degraded by the ISP.

The two parts of this conflict are NextGenTel and Norwegian Broadcasting Corporation (NRK). NextGenTel is one of the largest providers of broadband connections to private customers in Norway. NRK is the Norwegian government-owned radio and television public broadcasting company, and one of the leading media companies in Norway. One of the services NRK delivers is web-TV where the user can choose to watch programs that have been aired earlier free of charge.

In June 2006 NextGenTel decided to deliberately limit the bandwidth from NRK's free web-TV service. NextGenTel claimed that they had limited the bandwidth on this free service, because they were not interested in making new investments to increase the bandwidth to cover a service that they do not receive any revenue from. Streaming of football matches from the Norwegian top division on the other hand, was a payment service where NextGenTel is paid to distribute the content, and bandwidth from this service was not limited in the same way as NRK's free web-TV service. For the customers of NextGenTel this meant that they would receive lower quality of service on content received from NRK compared to content received from providers that pay NextGenTel for distribution.

This is the start of a development that can change the whole nature of the Internet as we know it, and transform it from a neutral network to a network where the different ISPs decide what content and services you can and cannot consume, and at which quality you should receive different services. An illustration of this scenario is shown in Figure 4.1. The extreme version of this would be a scenario where every ISP blocks traffic from every content provider that does not pay the ISP for distribution. NPT wishes to prevent this development and preserve the neutral nature of the Internet by controlling the ISPs' actions.

4.3.2 Deutsche Telekom Blocking VoIP in Germany

Another good example of questionable behavior by an operator is the blocking of VoIP traffic on mobile terminals in the mobile network of T-Mobile. This mobile operator is owned by Deutsche Telekom, which has exclusive rights to the iPhone

in Germany. They have said they will cancel the contracts of subscribers who attempt to install workarounds to use Skype on their iPhone. So the prohibition is implemented both physical and contractual. T-Mobile also admits that they have been blocking usage of VoIP for two years [2].

This example shows how a provider may block traffic based on the application it belongs to. We regard this blocking to be in conflict with the principles of network neutrality.

4.4 What About the Content Providers?

The network neutrality principles provided by NPT clearly states that the Internet providers should not discriminate different services, application types or contents. In this section we want to view the network neutrality concept from the content providers' point of view.

4.4.1 TV2 Media Proxy

The current version of the principles does not limit the content providers to be neutral on their choice of what ISP that could access their content. A concrete example is TV2 who have an exclusive agreement with a selection of the Internet providers in Norway [70]. This agreement allows TV2 to have a local media proxy within the access network of the selected ISPs which allows the customers to retrieve the media content from TV2 in high quality. Customers of an ISP which does not have an agreement with TV2 can only retrieve the content in limited quality, even though they have paid for a high enough bandwidth to retrieve the high quality stream. This scenario is illustrated in figure 4.2 where TV2 has a local media proxy in the access network of ISP #2.

4.4.2 Possible Implications

The mentioned scenario is not in conflict with NPT's principles of network neutrality. Because content providers are not constrained by network neutrality it may lead to unwanted situations. We will describe some of the possible situations below. The reader should notice that it is hard to predict the future and these are just hypothetical predictions by the authors.

Multiple Internet Subscriptions to Access all Services

This situation is already a fact in the satellite television world, where you need to have multiple subscriptions to access all interesting channels. If the content providers are allowed to choose which ISPs who can access their content, we might get a situation in the future where the end-user need multiple Internet subscriptions to access different services.

More Expensive Internet Access

Today the end-user usually pays for an Internet access. Pay services and content are usually paid for separately. In the future one might get a situation where a lot of services are included in the basic Internet subscription, whether one uses these services or not. This might lead to increased price for a regular Internet access. In the television world, the analogy would be cable-TV subscriptions including a lot of channels many of the customers have no interest in.

Harder for new Competing Services

Today the Internet is relatively open and it has been quite easy for new services, such as Google, Twitter, Facebook and others, to capture the world and become worldwide services. This has been possible because there has been enough capacity in the backbone gluing the Internet together. One might see a future where the public traffic increase faster than the infrastructure, resulting in an overloaded public Internet. In this situation it would be quite hard for new competing services to emerge because they would need exclusive agreements with different ISPs. The large existing services might not allow these newcomers and demand the ISPs to block them out.

Even though we have purposed some hypothetical situations in this section, one should remember that most content providers are interested in reaching as much people as possible.

4.5 NBTT and Network Neutrality

NPT plans to develop and release NBTT in 2009. We presented the planned tool in section 2.2. The first version of the tool will have a server located at the NIX and data will be exchange between the end-user and the NBTT server. Is this test set-up able to reveal possible traffic discrimination?

To answer this question we have to start with looking at the planned system set-up, illustrated in figure 4.2. The figure shows end-user #1 performing a bandwidth test to the planned NBTT server (red line).

The planned system architecture for NBTT will be able to measure:

- **Bandwidth** – Both download speed and upload speed from/to the client can be measured with the planned test-setup by exchanging data of certain sizes.
- **Variation in Bandwidth** – By sampling the bandwidth it is possible to capture the experienced variations in bandwidth.
- **Latency** – Latency (delay) can be found by measuring the time a small amount of data needs to travel from the client to NBTT and back to the client again.

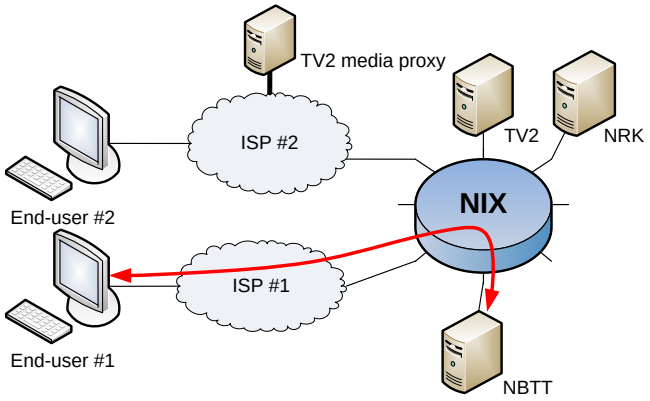


Figure 4.2: The figure shows a typical communication path from NBTT to the end-user performing a bandwidth test.

- **Variation in delay** – Variations in delay (jitter) can be estimated according to appendix A.8 in RFC1889 [26].

The first principle for network neutrality states that the Internet users should have a connection with a predefined capacity and quality. The capacity can be determined by the planned NBTT by sending data between the client and NBTT server. To verify the quality is a bit harder, because it depends on how we define quality and what quality parameters we have agreed upon. If we for example agree upon an up time of 99 % for the access connection it would be impossible to verify this with the planned tool. This is because we need to have connectivity to run the test. On the other hand, if we define the quality to be a specified maximum delay, variation in delay or variation in bandwidth it would be possible to verify this with the planned architecture.

The second principle states that the user should be allowed to use applications and send/receive content of their choice. This is possible to verify with the planned system architecture by emulating applications and their corresponding contents. By exchanging emulated content with the NBTT server the service can verify whether the content type is blocked. On the other hand it would be a rather time consuming task to simulate all types of applications and corresponding content. The architecture itself does not limit us from verifying this principle, but the overwhelming number of applications and services make it unpractical to verify them all.

The first part of the third principle states that there should not be any discrimination with regard to type of application, service type or content type. The planned architecture would be able to simulate different types of applications and compare the measured bandwidth between the different applications. If the deviations between the two measures are big, it is possible to assume that there exist discrimination of some sort. But one should be aware of those other factors such as competing traffic, sudden capacity changes in the network (link down) or alternative routes through the network that might affect the measurements. We must remember that the Internet is live and two successive bandwidth

measurements can get totally different results. If those situations are not taken into consideration, we could end up with a invalid conclusion.

The third principle also states that there should not be any discrimination with regard to the sender and receiver address. This is what NextGenTel did when rate limiting NRK from their network which we described in section 4.3.1. Because the planned NBTT only have one server location, this would be impossible to verify. This is impossible because there is no way to simulate different locations or sender/receiver addresses. We suggest extension to the current system set-up in section 11.1.2 where NBTT uses local test-processes in the content providers network.

The first version of NBTT will only exchange data between the client and the measurement server and will thus **not** be able to evaluate any of the network neutrality principles. We have in this section argued that the planned architecture will be able to emulate applications, services and content types. This can be used to verify that users may send/receive content of their choice. It can also be used to check if there is a discriminating trend between certain content types. But revealing discrimination between applications, services and content types are complicated tasks. There are many factors of uncertainty and successive bandwidth measurements can get different results for many reasons. Figure 4.2 also shows that certain content providers might use content proxies located within the ISPs' networks which complicate network neutrality evaluation even more.

Based on the discussion performed in this chapter, network neutrality will not be the main focus in the rest of this thesis. We will focus on the complicated task of broadband evaluation. In chapter 11 we will make use of our gained knowledge and discuss future possibilities for the planned NBTT. This chapter will also discuss possible extensions which enables NBTT to reveal trends which might imply breaches to the network neutrality principles.

Chapter 5

Measurement Techniques

In this chapter we will look into different broadband measurement techniques. We consider it very important to have a good knowledge on techniques existing today. This will enable us to gain deep knowledge from all the work that has been done in this area, which will be valuable in our later evaluation of existing tools.

Generally it is common to make a distinction between passive and active measurement techniques [9]. Passive techniques only observe the real-traffic on the link, while active techniques actively insert artificial traffic to measure the link capacity. Both techniques have their strengths and weaknesses. We remind the reader of the scope presented in section 1.3, and our main focus on active measurement techniques.

5.1 Measurement Parameters

Before we dive into the different passive and active measurement techniques we need to specify some performance parameters. It is important to know what we want to evaluate, before we can say something about how we should perform the evaluation. In the following we briefly introduce some of the important network performance metrics including: connectivity, delay, loss, throughput, utilization, available bandwidth and capacity bandwidth. Some of these metrics are defined differently in the literature, and we will present how we understand each metric. We will later in this section use these metrics in our discussions of the different measurement techniques.

- **Connectivity** – Connectivity refers to the ability to connect. This can be interpreted differently according to the specific context. For a UDP transfer, connectivity refers to the ability to transfer one packet from the sender to the receiver. For a TCP session, connectivity refers to the ability to establish and maintain a connection with packets flowing in both directions.
- **Delay** – Delay is usually defined to be the time required to move a packet from the source to the destination. Total network delay is usually composed of four components; processing delay, transmission delay, propagation delay and queuing delay.
  - **Processing delay** – Total time used to process the packet headers at the routers along the path.
  - **Transmission delay** – Time used to push the bits of a data unit onto the link. This is dependent on data unit size and bandwidth of the physical link.
  - **Propagation delay** – Total time it takes for the signal to propagate through the transmission medium. This is dependent on both distance and characteristics of the medium.
  - **Queuing delay** – Total time the data unit has spent waiting in queue at the intervening network nodes because of congestion. While the other three components contribute to a almost constant delay, the queuing delay can be very variable. This delay component depends on the cross traffic in the network.
- **Jitter** – The variation in delay (of received packets). Some real-time applications can be jitter sensitive.
- **Loss** – In the case of congestion or queuing in the network, the routers sometimes need to drop some packets resulting in loss. In some protocols, e.g. TCP, loss may be camouflaged. This results in a retransmission of lost data, which in turn lead to increased delay.
- **Throughput** – The achieved data rate including protocol overhead, e.g. packet header, retransmission, duplicates, etc.
- **Goodput** – The achieved data rate excluding protocol overhead, e.g. packet header, retransmission, duplicates, etc.

- **Utilization** – Used bandwidth divided by capacity bandwidth.
- **Bandwidth** – The data transmission rate, measured in bit/s. Bandwidth and throughput are considered synonyms in this thesis.
  - **Bottleneck link bandwidth** – The maximum transmission rate that can be achieved between two hosts at the endpoints of a given path in the absence of any competing traffic [28].
  - **Available bandwidth** – The amount of bandwidth on a link/path that is at one's disposal (maximum unused bandwidth) without affecting the existing flows in the network. Applications can usually not utilize all the available bandwidth due to small receive socket buffer and packet reordering [44].
  - **Capacity bandwidth** – The maximum total bandwidth a link/path can deliver [61].
  - **Achievable throughput** – The maximum total throughput one can achieve on a link/path. Tools using this technique typically use many parallel data streams to acquire a higher share of the total capacity bandwidth.

As we see there exists a lot of different performance metrics which characterize different aspects of a broadband connection. Many of the metrics are dependent on each other. A high loss rate can lead to increased delay for packets when utilizing a transport protocol supporting retransmission, e.g. TCP. The loss rate may also cause a reduced goodput. As we will see in the different techniques described later in this chapter, the performance metrics can be used to characterize a link and a path. As illustrated in figure 5.1 we see that a path consists of all the individual links traversed from the ingress to the egress node.

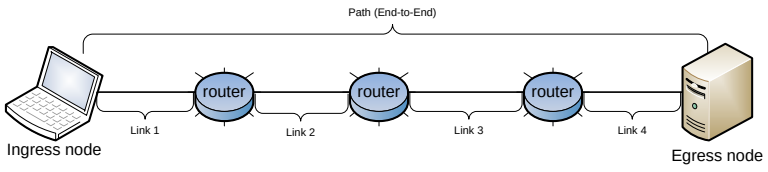


Figure 5.1: The figure shows the difference between a link and a path. A path consists of all the individual links from ingress node to the egress node.

5.2 Passive Measurement

Passive measurements involve observing the carried traffic in the network and do not require any coordination between the end hosts. Because passive measurement does not insert any traffic it cannot be used to measure connectivity. We must recognize that zero traffic on a link, or between two hosts, does not imply lack of connectivity between the two hosts.

Ideally passive measurements should be done without disturbing the measured network. Unfortunately this is not always possible to achieve. Passive

measurements can collect huge amount of measurement data, and this process might affect the performance of the network. If the collected data must be sent to a centralized database, this will cause a lot of network traffic which can affect the measured network negatively.

5.2.1 Traffic Monitoring

Network operators use passive measurement techniques to monitor the performance of the network. This information provides the network operators with a detailed view over their networks. In particular, congestions can be detected through periodic summaries of traffic load and packet loss on individual links; parts of the network exhibiting high delay or loss, as well as routing anomalies such as forwarding loops, can be identified by means of probes between pairs of nodes in the network [3].

It is possible for the operators to measure the available bandwidth on the different links. This is possible because network operators usually know the capacity of their own link and combined with the current traffic load, which can be obtained from interface counters, they can easily derive the available bandwidth.

5.2.2 Traffic Monitoring at End-Hosts

Passive bandwidth monitoring is usually implemented in applications sending or receiving large amount of data (e.g FTP clients, download process in a web browser, etc.). There also exist many applications that monitor and record the traffic coming in and going out of the computer interface. A good example is Wireshark described in appendix E.1. This tool captures all the packets entering and leaving the network interface of a computer. It is easy to see that this will both require a lot of storage and is very resource expensive. Usually we are not interested in *everything* flowing over the network and filtered, sampled and aggregated statistical information could be sufficient.

5.3 Active Measurement

Active measurement techniques insert traffic into the network in order to measure the network performance. The idea is to emulate *real* traffic, in order to say something about the network performance.

Example: A host record the time used to download data with a certain size to determine the bandwidth.

When utilizing active techniques it is important to remember that the inserted traffic generally influence the measurement, and these techniques will give a snapshot of the network performance at a certain time. Active measurement techniques have been criticized because they generate a lot of traffic that can disturb the network and thereby get an incorrect view of the network performance.

Example: Is it possible to send a packet into the network to determine if there is congestion? Your packet can be the one that causes the congestion!

The results derived from an active measurement are not always representative for the current network situation. This is because real-traffic flowing in the network is complex and affected by many different factors such as cross traffic, link failures, traffic types, service class and much more.

Intrusiveness is another major factor in designing a measurement technique. Generally a good active measurement technique should be as little intrusive as possible, which means that it should not significantly increase the network utilization, delays or losses. In [61] they say that:

> *"An active measurement tool is intrusive when its average probing traffic rate during the measurement process is significant to the available bandwidth in the path."*

These challenges require a measurement technique that is well designed. This is to ensure that we actually test the interesting metric without letting the current network situation affect our results, as well as our test should not affect the current network situation.

5.3.1 Capacity of Each Link in the Path

Variable Packet Size
Variable Packet Size (VPS) is a technique to measure the capacity on each link in the network path. The technique was first purposed and explored by Bellovin [7] and Jacobson [32]. The key element of this technique is to insert packets of varying lengths and measure the Round Trip Time (RTT) from source to each hop of the path as a function of the probing packet [61]. Just like in *traceroute* (explained in section 6.2.2) the VPS technique increment the Time To Live (TTL) field in order to measure the delay each packet experience at each hop. Packets of different size is needed to be able to solve the individual RTT to each hop. The interested reader is referred to [18], which explains *pathchar* in great detail. Pathchar is a simple application utilizing the VPS technique.

Unfortunately the VPS technique has many drawbacks. It is a slow technique which generates a lot of traffic. It is not considered an intrusive technique because it only sends one probing packet and waits for the reply before sending the next packet. Another challenge is that the routers are not built to send TTL timeout messages in a timely manner, and some routers does not generate these messages at all.

5.3.2 Available Bandwidth of Each Link in the Path

Today there exists no active technique that estimates the available bandwidth at each individual links in the network [61][1]. For an ISP, available bandwidth at each link can be easily be determined by passive measurements. This is possible

[1] We could not find any techniques in our studies either.

because the ISP knows the links' capacity and interface counters can be used to determine the current bandwidth usage. Thus will the available bandwidth be the difference between capacity and used bandwidth as shown in figure 5.2.

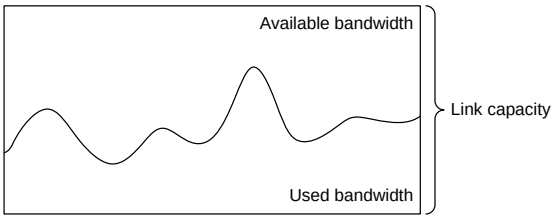


Figure 5.2: Available bandwith on a link with given capicity

5.3.3 End-to-End Connectivity

A connectivity test simply verifies that there is some connection between the ingress and egress node. This performance metric is usually verified by sending a simple request message and wait for a reply message (the most common way is to perform a ping, which is explained in section 6.2.1).

Generally connectivity can be divided in to two categories:

- **One-way connectivity** – There is connectivity in only one of the directions (from A to B) but not in the reverse path.
- **Two-way connectivity** – There is connectivity in both directions (from A to B and from B to A).

Some applications might only require connectivity in one of the directions to function. It is more common though to require two-way connectivity, e.g. TCP requires two-way connectivity to set-up a path from A to B.

5.3.4 End-to-End Delay

Delay is generally divided into two classes:

- **Bidirectional delay** – A common technique to measure bidirectional delay is to send a probe packet with an assigned timestamp and when this packet is received the RTT can be calculated.
- **Unidirectional delay** – Is more difficult to measure. This is because we require two cooperating hosts, and they must be strictly synchronized. The sender inserts a probe packet with an assigned timestamp and a sequence number and addresses the packet to the receiver. The receiver can then calculate the unidirectional delay from the sending timestamp and receiving timestamp.

5.3.5 End-to-End Loss

Loss is usually measured by sending a lot of probe packets into the network, and count how many of the packets that are not received in the other end. This can be done by asserting sequence number to each packet or require all packets to be acknowledged.

5.3.6 End-to-End Capacity

Packet Pair
Packet pair (also known as packet dispersion) is a well known technique to measure the end-to-end capacity of a path [61] [9]. This will usually be equal to the capacity of the smallest link in the path, also known as bottleneck capacity. The packet pair is illustrated in figure 5.3. As seen in the figure the two packets will be dispersed as the packets traverses a link of a certain capacity. The receiver will see the maximum dispersion experienced at the bottleneck link as the distance between the two packets. End-to-end capacity can be calculated from this.

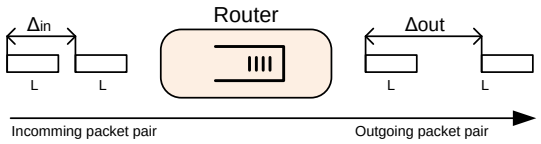


Figure 5.3: Packet dispersion before and after router queue [61].

The packet pair technique assumes that there is no cross traffic and that the packets will be queued next to each other in each link. These two assumptions do not hold in practice and can lead to capacity underestimation or even worse, capacity overestimation. Capacity overestimations occur if cross traffic delays the first packet more than the second packet, decreasing the dispersion.

Other techniques has been purposed to reduce the effect of cross traffic [17]. The idea is to send many packet pairs, often referred to as packet pair trains, and use statistical methods to filter out erroneous measurements. These techniques get slightly better results compared to plain packet pair probing. ToPP is a technique from this category that also can be used to measure end-to-end available bandwidth, and therefore we present this technique the next section.

5.3.7 End-to-End Available Bandwidth

Self-Loading Periodic Streams (SLoPS)
SLoPS tries to estimate the available end-to-end bandwidth [61]. The technique is based on a source sending equal sized packets at a certain rate, R, to the receiver. SLoPS tries to find the paths available bandwidth by slightly increasing the rate, and monitor the inter-arrival jitter experienced by the packets. Inter-arrival jitter can be estimated according to appendix A.8 in RFC1889 [26]. When R becomes greater than the paths available bandwidth, A, the stream will experience short

term overload and queuing in the bottleneck link. This will lead to increased one-way delay experienced by the probing packets (which results in a higher jitter). If the rate is lower than the available bandwidth, they will not cause queuing in the bottleneck link, and thereby their one way delay will not increase. The two scenarios described is shown in figure 5.4. From the left figure we can see the situation where the rate, R, is lower than the paths available bandwidth, A. In the right figure we see the opposite situation where the rate is higher than the available bandwidth. From the figure we can see that the one way delay increase significantly.

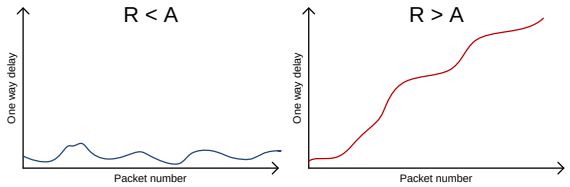


Figure 5.4: One way delay when the rate is lower (left) and higher (right) than the available bandwidth [61].

The SLoPS technique also keeps a "silent" period between successive streams in order to not stress the network to much with probing traffic. When the streams are sent at rate lower than the available bandwidth the intrusiveness of this technique is minimal. On the other hand when the rate is higher than the available bandwidth the intrusiveness is rather high. The silent period keeps the intrusiveness measured over time at a moderate level. The goal is to limit SLoPS to only use up to 10 percent of the available bandwidth.

Trains of Packet Pairs
Trains of Packet Pairs (ToPP) is another technique to measure the end-to-end available bandwidth [61]. In this technique the sender sends successive packet pairs to the receiver, at a gradually increasing rate. The basic idea behind this technique is analogues to SLoPS. If the rate is too high, the second packet will be queued, and the available bandwidth in the path can be calculated from the experienced delay. The major difference is that the ToPP technique also is able to calculate the bandwidth of the bottleneck link.

ToPP relies on sending many successive tests in order to get satisfying result. This is because the available bandwidth varies over time, and successive measurement can give a reasonable good statistical foundation for making a better estimate. This can give a good overview of the current situation in the network. Unfortunately this will lead to a very intrusive approach that might affect the measured network. A tradeoff could be to spread the measurements over time, leading to a rather slow algorithm for estimating the available bandwidth.

5.3.8 End-to-End Bulk Transfer Capacity

The end-to-end Bulk Transfer Capacity (BTC) is recommended by IETF as a metric for measuring a paths ability to transfer large files using TCP [40]. In [33]

they have the following definition of the BTC of a path:

> *"The BTC of a path in a certain time period is the throughput of a persistent (or 'bulk') TCP transfer through that path, when the transfer is only limited by the network resources and not by buffer, or other, limitations at the end-systems."*

It is important to recognize the difference between end-to-end available bandwidth and the paths BTC. The former gives a measure on the amount spare capacity of a path, independent of the transport protocol utilizing it. BTC on the other hand gives a measure on the experienced throughput of a single, persistent, TCP stream, that depends on TCP's congestion control and is only limited by network resources. Many parallel TCP connections might get a higher aggregated throughput than the BTC.

BTC is usually measured by using two cooperating hosts, where one acts as a sender and the other as a receiver. The idea is to send a specified amount of data from the sender to the receiver. The bulk of data is sent as a regular or emulated TCP stream, which implement regular congestion control.

BTC is considered intrusive because it sends large amounts of data which may affect the existing flows in the network.

5.3.9 Achievable Throughput

This technique uses (or misuses) TCP or UDP in order to transfer large amount of data and measure the achieved throughput. Tools using this technique usually lets the user "tune" TCP or UDP parameters and optionally use many parallel streams to get a higher share of the bandwidth.

This technique is considered extremely intrusive because it sends large amount of data which potentially can affect the existing flows in the network. The idea is to see how much bandwidth we can achieve from one host to another host under different conditions. One example is to test how high throughput a non-TCP friendly stream, such as UDP, can achieve in a network consisting of manly TCP flows. As the UDP stream will continue at a persistent rate the other flows following regular TCP congestion control will gradually back off to adapt to the current network situation.

The authors ran a test at the student campus showing that just one UDP was able "steal" bandwidth from an ongoing TCP session. The network had about 90 Mbit/s available bandwidth (TCP goodput), and the TCP session is started first. After ten seconds we started a UDP transfer at 80 Mbit/s. The result is shown in figure 5.5. We can see that the UDP session gets all the wanted bandwidth and the TCP session backs off immediately, left with the remaining 10 Mbit/s.

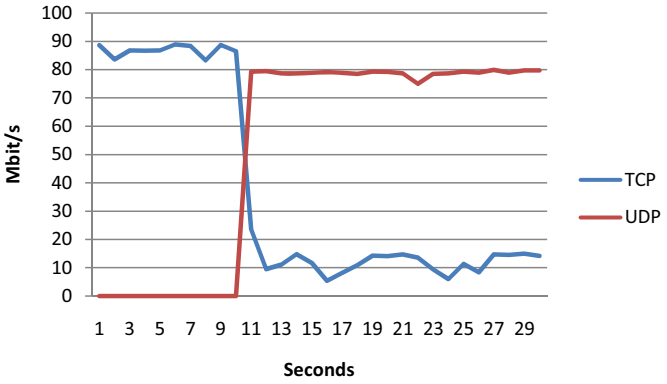


Figure 5.5: UDP "steals" bandwidth from a TCP session.

5.4 Discussion of the Techniques

Passive techniques can give information about the network performance experienced in the past. We can use the obtained information from these techniques to say something about what *has* happened and what happens right *now*. For instance we can say something about the highest bandwidth achieved this week or the current traffic situation. Passive measurements can be used to determine the available bandwidth of a path, or individual links of a path, but this requires knowledge of the link capacity and the current network situation. Current network information could be collected directly from the network by reading interface counters (for each hop in the network), but this would require access to retrieve this information. We doubt that the ISPs will grant access to this information in the near future.

Different active techniques have been introduced in this section. Some of the techniques have a link-to-link focus while others focus on the overall performance of the whole path, from end-to-end. From an ISP's point of view we can argue that the characteristics of each individual link are of great importance. The ISP can, if necessary, locate and upgrade/fix links with poor quality. For a subscriber it is more interesting to evaluate the whole path from one end to another. This is because the regular[2] subscriber does not know anything about the intermediate networks, nodes and/or network components. And if bottleneck links are found, the subscriber cannot do anything to correct them or choose a different path through the network. The Internet is "best-effort", paths are chosen arbitrary and thus the user has to use the path provided. End-to-end characteristics are thus more interesting for subscribers.

Intrusiveness is an important measure on how much a technique disturbs the rest of the traffic in a network. Generally it is usually beneficial for the applied technique to be as little intrusive as possible. The VPS algorithm for estimating link capacity is not considered intrusive because it sends only one probe packet; wait for a reply, before sending the next probe packet, which give a packet rate on

[2]Be aware of the difference between a regular and an experienced user.

one packet per RTT. From a subscribers point of view the end-to-end capacity will be determined by the slowest link. Usually this will be the last mile, connecting the user to the ISP's network. We also know that the typical access bandwidth for a subscriber is a small fraction of the overall capacity for an ISP. This limits the possible intrusiveness generated by one subscriber. If the active measure also is performed for a limited time, the overall intrusiveness is even smaller. The intrusiveness is thus not considered important in the context of broadband evaluation tools.

Available bandwidth can be measured with different techniques and we introduced ToPP and SLoPS. Both techniques are based on the idea of filling up the queues of each router along the path, and measure the delay between the received packets or trains of packets. If the delay increases, the threshold for available bandwidth has been surpassed and we can thus determine the available bandwidth. These techniques only give a rough estimate of the available bandwidth. For a subscriber the available bandwidth can never be larger than the capacity of the last mile, which usually is the bottleneck link. Available bandwidth techniques could be a good aid for selecting what server to download a big file from. Today this selection is usually performed on the basis of geographical locality and the closest server is chosen. A better approach could be to roughly estimate the available bandwidth and select the server that can provide the path with the highest spare capacity. This is discussed further in section 11.2.2.

We introduced BTC and explained how this differs from achievable throughput. Achievable throughput tries to push as much data as possible into the network with the result that other flows back off. BTC on the other hand uses one TCP connection following regular congestion control. This is less intrusive to other flows in the network. For a subscriber with a last mile bottleneck, BTC would be beneficial for competing traffic coming from the same home network. On the other hand a subscriber actively running a bandwidth test is probably not interested in the how much traffic is available if the link is shared evenly. Most likely a subscriber is interested in how much data she gets through her access. *Do I really get the bandwidth that I pay for?* In this case the bandwidth test technique should occupy as much of the last mile link capacity as possible during the test to ensure that other traffic[3] is not disturbing the results too much. To do this, she must use an achievable throughput technique.

We have seen that different techniques look at different network characteristics. It is important that that we choose the technique that evaluate the network in the correct context. The context will determine what network characteristics we should evaluate. A gamer would be more interested in keeping the delay as low as possible while a downloader would like to get the highest possible goodput, where delay only have minor importance. An application downloading an update could benefit from knowing the available bandwidth it can use without disturbing the rest of the traffic on the network. As we see there are different techniques fitting different needs. The technique we select must thus depend on what we want to evaluate.

[3]E.g. Traffic from other computers or applications looking for an update.

Chapter 6

Existing Test Tools

In this chapter we will look into some of the existing broadband test tools based on active measurement techniques. We have categorized the existing tests in two different categories, *on-line* and *stand-alone* tools. On-line tools are tools with a predefined server, and a small client application that runs within a browser. The service is often easy to use, and available through some sort of web page. Stand-alone tools on the other hand are separate tools that must be started manually and usually requires the user to set up some specialized server process.

6.1 On-line Test Tools

These tools are often implemented as small client applications which run inside the user's web browser. The implementations of these test tools are usually based on Java scripts, Java applets or Flash applications. The clients run tests towards a web server to download or upload content.

In our research we have found a lot of different on-line tools that help the end user in estimating the delivered bandwidth. They are all based on roughly the same idea of measuring the time spent on transferring data of a given size end-to-end, followed by some calculation of bandwidth based on these measurements. Some tools are quite sophisticated in their calculations, while others are as simple as can be. Since many of these tests are very similar, one can get a good understanding of their workings just by studying a small selection of these tests. In this section we will look closer at some of the on-line tools that we have studied during our research[1].

In Norway, a popular on-line test tool is the Speedometeret provided by ITavisen. This service have been promoted a lot, and they claim to have in average 60 000 measurements each week[2]. This service is also made available through other web sites, but then with an alternative name. Since this test is so much used, it seems natural to include it in our studies. Some of the background for this thesis is the NPT's wish for establishing a Norwegian version of the Swedish Bredbandskollen TPTEST that PTS have introduced in Sweden. Therefore it is obvious that Bredbandskollen TPTEST must be one of the tools we study. We have also included some more tools; MySpeed because of the more comprehensive feedback report it gives, and Speedtest because it have a lot of test servers distributed around the entire globe. Network Diagnosis Tool is included because it in addition to bandwidth measurements can identify whether network problems exist. It also has a more academic approach than the other commercial tools. Glasnost is at tool that focuses more on discrimination of BitTorrent traffic, and is therefore relevant in the discussion of network neutrality.

[1] We have tested a lot of tools in this category, but because of the similarities in these tools, we will only present the details for the ones we consider most relevant.

[2] Information provided by ITavisen.

6.1.1 Speedometeret

Speedometeret is a very simple bandwidth test with no manual configurability. A screen shot of the user interface is shown in figure 6.1. The test calculates your bandwidth for download and upload.

Name:	**Speedometeret**
Provider:	**ITavisen.no**
URL:	**http://itavisen.no/speedometeret**
Server Location:	**Oslo, Norway**
Technology:	**Java Applet**

Spedometeret uses a Java Applet, and requires that Java is enabled in the browser. The download rate is estimated by downloading a file with a predefined size and measuring the time used. The download rate is then calculated according to the following simple formula:

$$bandwidth = \frac{size\ of\ downloaded\ file}{time\ used\ to\ download\ file} \tag{6.1}$$

Speedometeret only utilize one TCP connection during the test. To increase the accuracy, the test will switch to another file with a larger size if the time used is lower than a predefined threshold. This is done to make the download time long enough so that the transient slow start period of TCP can be neglected (see section 3.4). The upload rate is estimated in the same way, but with a smaller amount of data. This is because the test is designed for the highly asymmetric capacity of an ADSL subscriber.

Figure 6.1: Bandwidth test: Speedometeret (Screen shot)

Speedometeret is an instance of a service provided by Aller Internett. Dinside.no and digi.no got their "Surfometer" which is the same service as Speedometeret with an alternative user interface. They also use the same test server.

JavaScript version
Speedometeret also has a simpler version that does not require Java. This is actually the previous version of this service. It is implemented as a simple JavaScript that most browser support out-of-the-box. This version is only capable of measuring download speed.

The measurement is done by downloading an image and measure the time used on this operation. This is in fact the simplest type of bandwidth measurement available. It is based on the assumption that the bandwidth utilization is 100 % during the entire download, and that HTTP- and image rendering overheads are neglectable. As we will see in section 6.1.7, this assumption is not necessarily correct. It also a known issue that timing in JavaScript can be significantly inaccurate [62]. With a small file size and large bandwidth, this inaccuracy can cause significant error in the final result.

Many of the bandwidth test tools available on the Internet today are very similar to this test, and does exactly the same by downloading an image while using JavaScript for timing. Since we consider this approach rather inaccurate we will not present other implementations of this type of tool.

6.1.2 Bredbandskollen TPTEST

Bredbandskollen TPTEST (from now on called Bredbandskollen) is a service provided by .SE[3] in corporation with the Swedish Post and Telecom Agency. The aim is to provide a non-commercial broadband evaluation tool for Swedish subscribers. A screen shot of the test is shown in figure 6.3. Bredbandskollen relies on a test engine developed by Ookla.

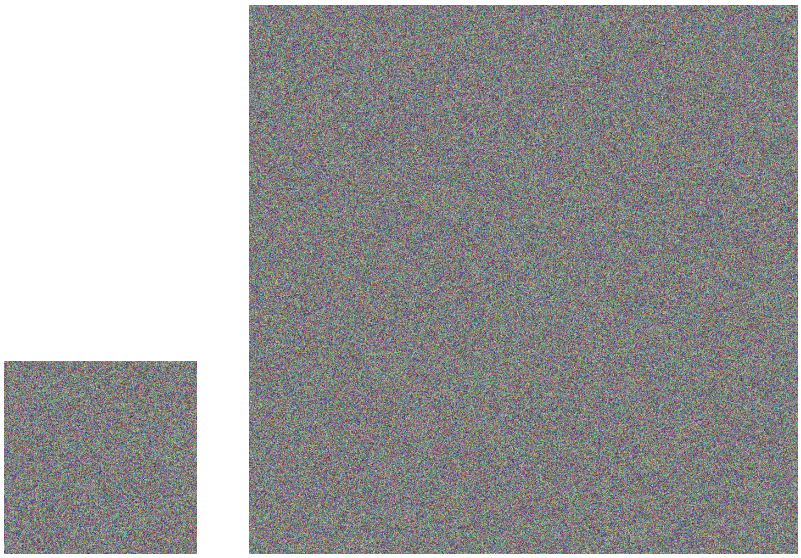
Name:	**Bredbandskollen TPTEST**
Provider:	**.SE**
URL:	**http://www.bredbandskollen.se**
Server Location:	**Stockholm, Sundsvall, Malmö and Göteborg in Sweden.**
Technology:	**Flash**

Bredbandskollen performs the following steps [64]:

1. A test-server is selected based on the location of the user. The user can manually override this operation by selecting another test-server.
2. 10 HTTP requests and replies are sent in order to measure the average RTT.
3. A pre-measurement is done by downloading a small image. This is done to roughly estimate the user's capacity and determines how much data to download and upload during the test. The image used is shown in figure 6.2a.
4. Two images are retrieved in parallel sessions. The size is determined in the pre-measurement. Figure 6.2b shows an image with a size of 2 MB. The size of the images must be large enough so that the transient slow-start period of TCP can be neglected (see section 3.4). The throughput is measured up to 30 times per second. The final download rate is determined by looking at ordered samples through a sliding window to eliminate anomalies.
5. Random data is generated and uploaded with HTTP POST messages to the test-server through two parallel sessions. The pre-measurement is used to determine how much data that should be uploaded. It is important that the data sent is large enough so the transient slow-start period of TCP can be neglected (see section 3.4). The time used to upload the data is used to calculate the upload rate.

The images used in the download test is shown in figure 6.2. From the pictures we can see that every pixel in the images is randomly generated. This is done to achieve an image without correlation between adjacent pixels, and thus prevent compression of the images.

[3]SE, Stiftelsen för Internetinfrastruktur.



(a) 350x350.jpg (b) 1000x1000.jpg

Figure 6.2: Images used for download-rate-test in Bredbandskollen and Speedtest.

Figure 6.3: Bandwidth test: Bredbandskollen (Screen shot)

6.1.3 Speedtest

Speedtest is an interesting tool that got a lot of servers located all over the world. The tests are performed within the web browser with HTTP GET and POST messages. The tool is a commercial service provided by Ookla and is free to use. The test methodology used by Speedtest is the same as for Bredbandskollen, but with a different interface. Also the statistics gathered and the presentation of the statistics is quite different. We review the statistics in chapter 10.

Name:	**Speedtest**
Provider:	**Ookla**
URL:	**http://www.speedtest.net/**
Server Location:	**Multiple, all over the World.**
Technology:	**Flash**

Figure 6.4 shows a screen shot of the service, and we can easily see the numerous number of possible server locations. Multiple locations allow the user not only to check the access capacity but also the possible bandwidth to specific locations all over the world.

Figure 6.4: Bandwidth test: Speedtest (Screen shot)

The operation of Speedtest is very similar to Bredbandskollen and follows these steps [54]:

1. A preferred test-server is selected based on the location of the user. The user can manually select another test-server.

2. 10 HTTP requests and replies is sent in order to measure the average RTT.

3. A pre-measurement is done by downloading a small image. This is done to roughly estimate the user's capacity and determines how much data to download and upload during the test. The image used is shown in figure 6.2a.

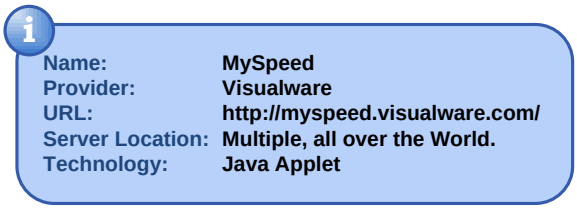
4. Two (or one) images are retrieved in parallel sessions. The size is determined in the pre-measurement. Figure 6.2b shows an image with a size of 2 MB. The size of the images must be large enough so that the transient slowstart period of TCP can be neglected (see section 3.4). The throughput is measured up to 30 times per second. The final download rate is determined by looking at ordered samples through a sliding window to eliminate anomalies.

5. Random data is generated and uploaded with HTTP POST messages to the test-server through one or two parallel sessions. The pre-measurement is used to determine how much data that should be uploaded. It is important that the data sent is large enough so the transient slowstart period of TCP can be neglected (see section 3.4). The time used to upload the data is used to calculate the upload rate.

One noticeable difference between Bredbandskollen and Speedtest is that Bredbandskollen always uses two TCP connections to measure the achievable throughput in both directions, while Speedtest varies the amount of connections in use, depending on the server configuration. The Speedtest servers located in Norway uses up to 6 simultaneous connections.

6.1.4 MySpeed

MySpeed is a more advanced broadband evaluation tool provided by Visualware. This on-line tool operates at the transport layer and not at the application layer like the previous described tools. MySpeed is implemented as a Java Applet and the data transfer is realized through TCP sockets.



MySpeed measures many different aspects of the connection. We will in the following explain each performance parameter from the test, illustrated in figure 6.6.

Download speed
This is the average download rate achieved during the test period measured in bit/s. This is calculated by transferring as much data as possible over a period of 8 seconds.

Upload speed
This is the average upload rate achieved during the test period measured in bit/s. This is calculated by transferring as much data as possible over a period of 8 seconds.

Quality of service
The Quality of Service metric is defined by the authors of MySpeed [72] to be:

$$Quality\ of\ Service = \frac{Minimum\ speed}{Maximum\ speed} \tag{6.2}$$

This parameter is used to verify that the connection delivers a continuous capacity. This is important for real-time applications, like VoIP or streaming of video, which requires a stable predictable bandwidth. The metric defined is quite simple, and does not by any means capture all aspects of quality of service. But as a metric for the degree of continuous delivery of capacity it has some expressivity.

Download pause
TCP pause or TCP delay, is the maximum time the receiver has to be idle waiting for the packets in a packet stream to arrive. TCP pause can be caused by congestion in the network, long RTT or because the sender has many active TCP connections. In figure 6.5 we have illustrated the effect a large RTT may cause to a TCP connection. The transmission time to transmit the receiver window is

smaller than the RTT. Due to the operation of TCP, this will force the connection to idle while the sender waits for acknowledge.

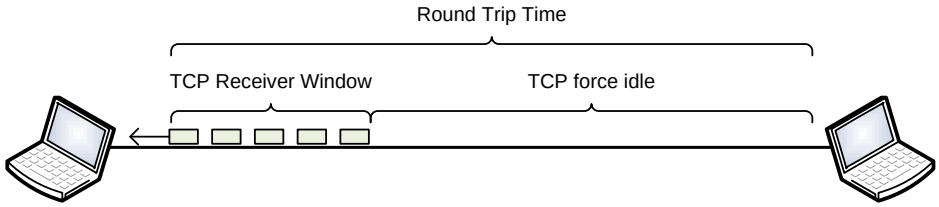


Figure 6.5: Effect of TCP pause

MySpeed reports both max pause and the average experienced pause. A high max pause is usually a sign of a problematic connection [73]. The delay illustrated in figure 6.7 is an effect of TCP pause. If the pause graph is very symmetrical it is most likely caused by traffic shaping (e.g. by the ISP) of the data flow and a fluctuating pause graph would indicate congestion or other network problems.

Round trip time to server
MySpeed also measures the RTT to the server. The test reports both the maximum RTT and average measured RTT.

A screen shot of the MySpeed tool is shown in figure 6.6. On the top we see the running test, which also indicate what type of connection your bandwidth represents. The bottom image is an overview over the detailed statistics gathered during the test, and is explained above. The test samples the amount of received data each millisecond and calculates the transfer speed and delay and plots this information in a graph, illustrated in figure 6.7. The figure shows the transfer and delay graph for both download (top) and upload (bottom).

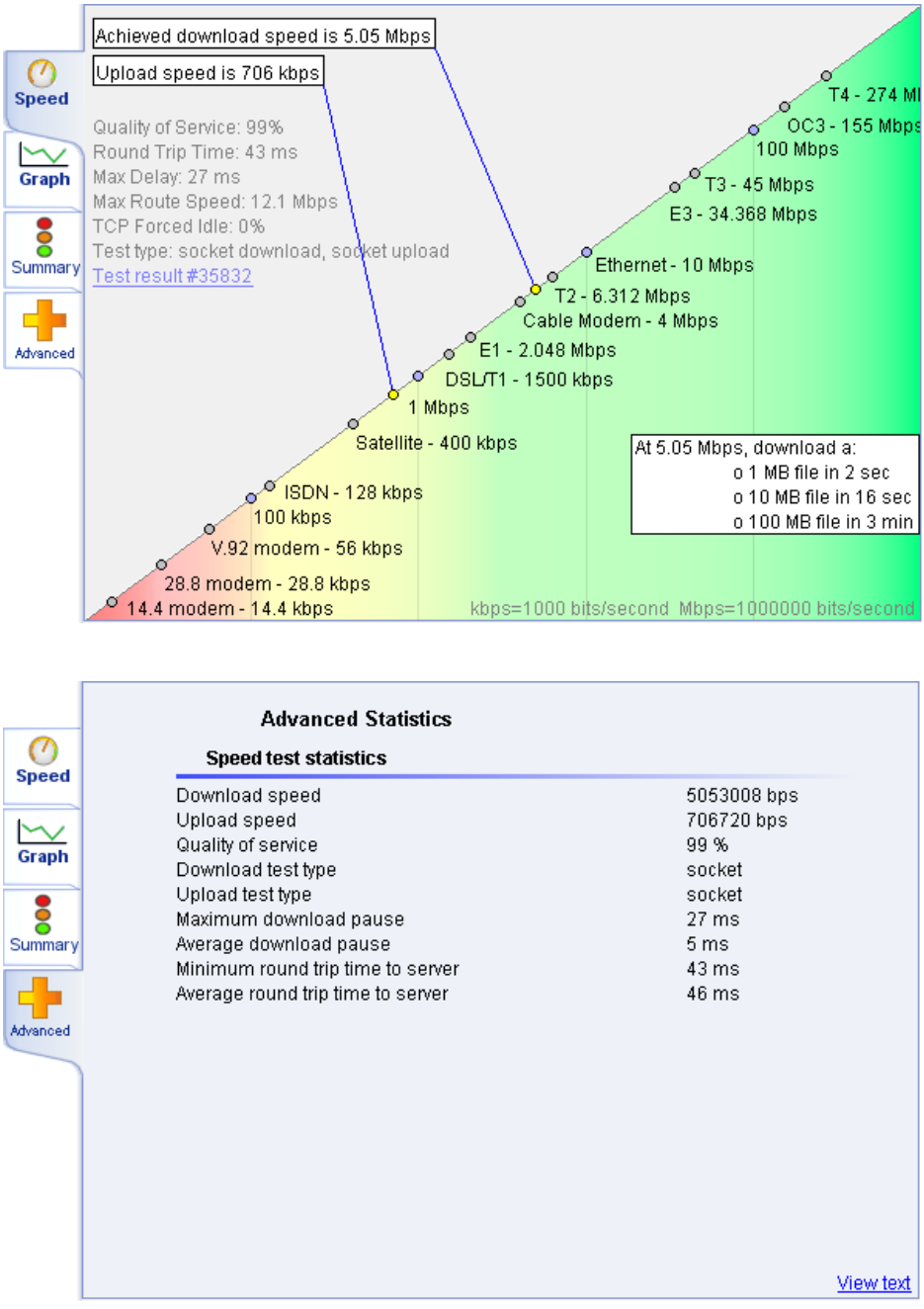


Figure 6.6: Top: Screen shot during test. Bottom: Advanced summary after test.

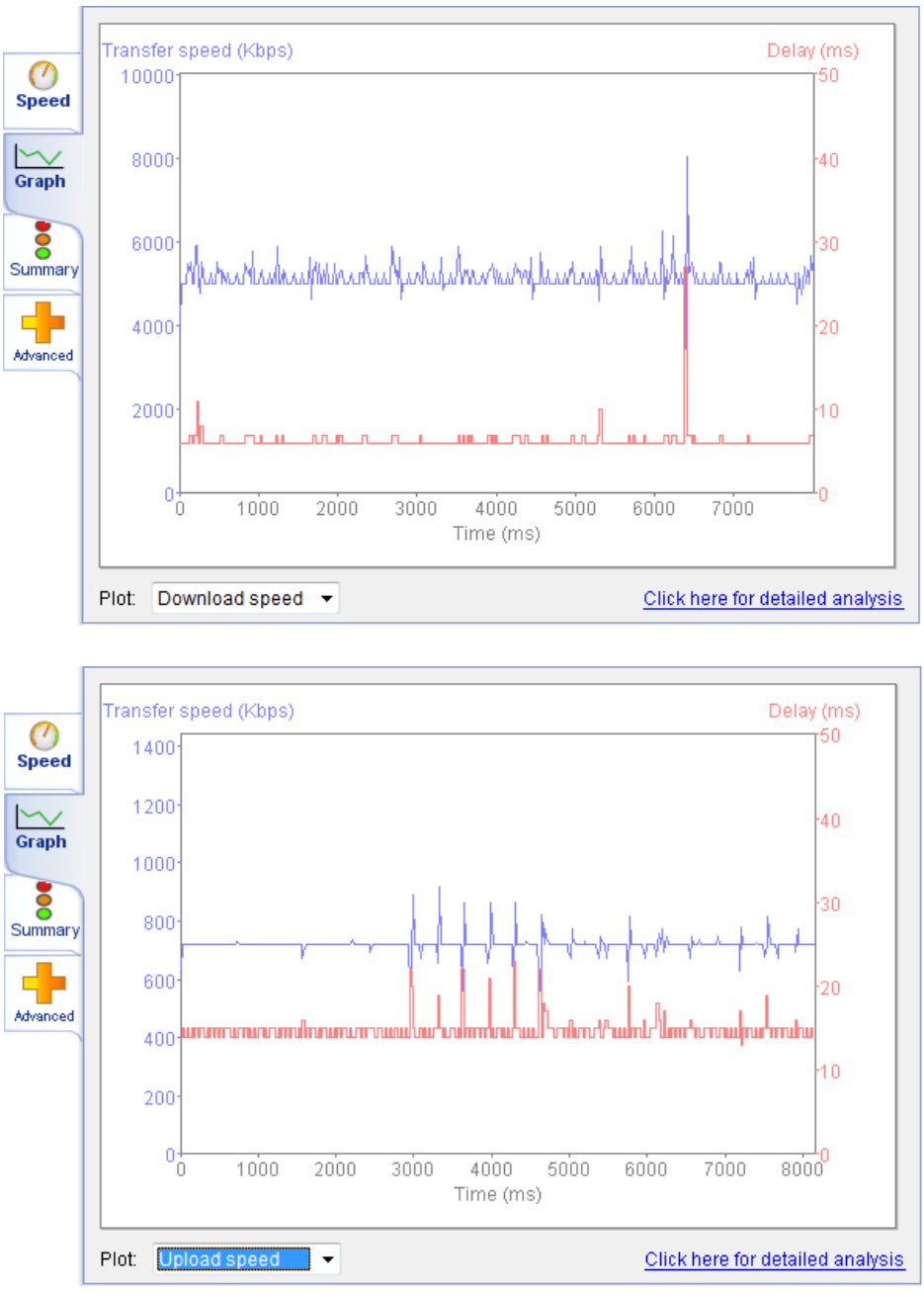


Figure 6.7: Graphs showing upload and download speed together with delay during test.

6.1.5 Glasnost

Glasnost is a tool that attempts to determine if the user's ISP is throttling or blocking BitTorrent traffic. BitTorrent is one of the most common used peer-to-peer file sharing protocol. The protocol is used to download files, where the user gets small fragments of the file from many different clients. The user also contributes by uploading completed fragments to other users. For a more detailed information about BitTorrent the reader is referred to [75].

Name:	**Glasnost**
Provider:	**Max Planck Institute for Software Systems**
URL:	**http://broadband.mpi-sws.org/transparency/bttest-mlab.php**
Server Location:	**Germany**
Technology:	**Java Applet**

There are many reasons for why the ISPs would like to throttle BitTorrent traffic:

- BitTorrent is a protocol that uses TCP aggressively. The client sets up a large number of TCP sessions in order to download one file, and thereby get a larger share of the available bandwidth.
- BitTorrent is known for its most common use, namely in distribution of pirate copied material. But this protocol is also used for legal purposes as well.
- Heavy BitTorrent users are known for using a lot of their available bandwidth, requiring the ISPs to dimension with a higher bandwidth per user.

In order to evaluate if the operator is throttling BitTorrent traffic the Glasnost tool performs the following steps:

1. Emulate BitTorrent traffic at well known BitTorrent ports. Download and upload test is performed. Compare with regular TCP traffic on the same port.
2. Emulate BitTorrent traffic at non-BitTorrent ports. Download and upload test is performed. Compare with regular TCP traffic on the same port.
3. Download and upload TCP traffic (non-BitTorrent traffic) on a well known BitTorrent port. Compare with regular TCP traffic on a non-BitTorrent port.

The results from the above mentioned tests are compared. If no major deviation is found, the test concludes that the ISP does not perform any BitTorrent throttling. Figure 6.8 shows a screenshot of Glasnost during the test, while figure 6.9 shows the result page shown after the test has completed. In this test the tool concludes that no BitTorrent throttling is detected.

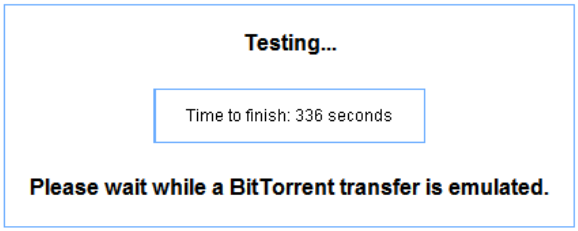
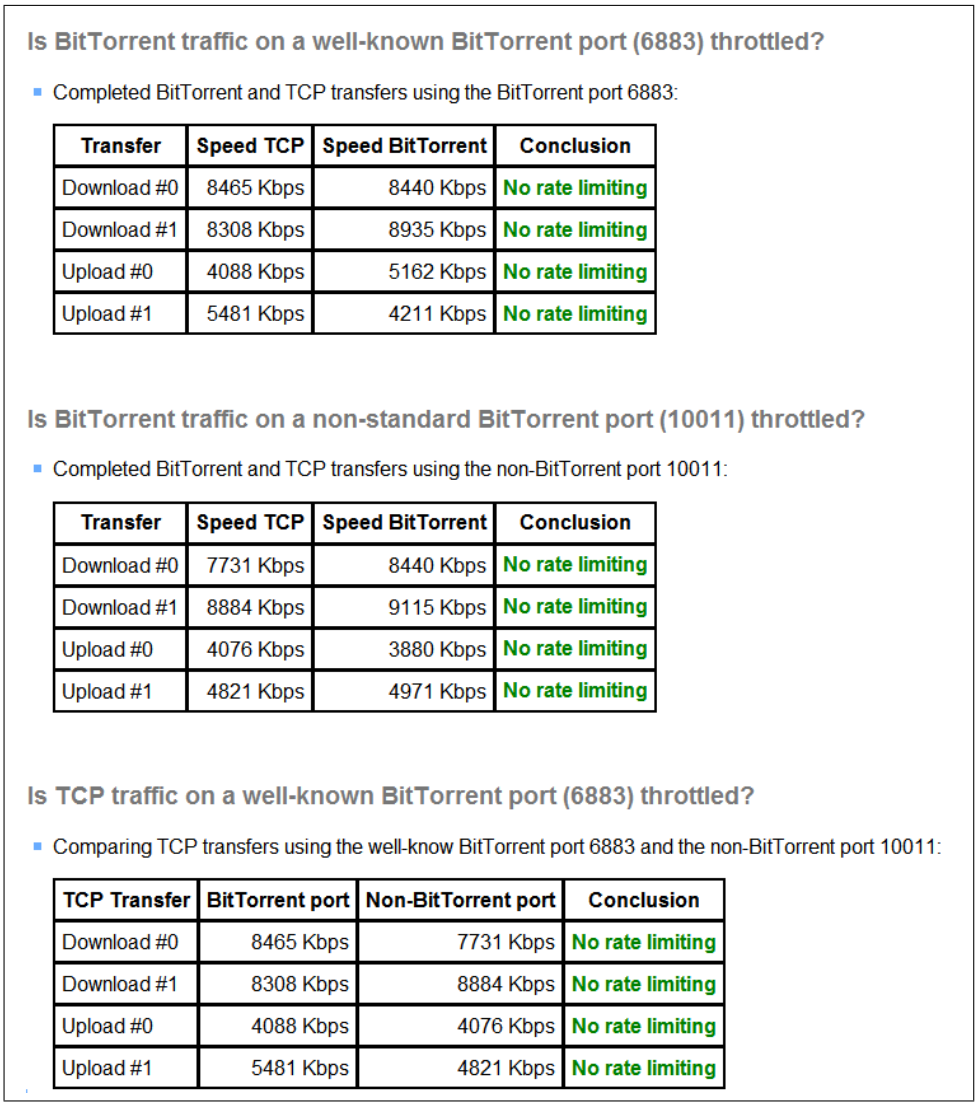


Figure 6.8: Bandwidth test: Glasnost (Screen shot)



Is BitTorrent traffic on a well-known BitTorrent port (6883) throttled?

- Completed BitTorrent and TCP transfers using the BitTorrent port 6883:

Transfer	Speed TCP	Speed BitTorrent	Conclusion
Download #0	8465 Kbps	8440 Kbps	No rate limiting
Download #1	8308 Kbps	8935 Kbps	No rate limiting
Upload #0	4088 Kbps	5162 Kbps	No rate limiting
Upload #1	5481 Kbps	4211 Kbps	No rate limiting

Is BitTorrent traffic on a non-standard BitTorrent port (10011) throttled?

- Completed BitTorrent and TCP transfers using the non-BitTorrent port 10011:

Transfer	Speed TCP	Speed BitTorrent	Conclusion
Download #0	7731 Kbps	8440 Kbps	No rate limiting
Download #1	8884 Kbps	9115 Kbps	No rate limiting
Upload #0	4076 Kbps	3880 Kbps	No rate limiting
Upload #1	4821 Kbps	4971 Kbps	No rate limiting

Is TCP traffic on a well-known BitTorrent port (6883) throttled?

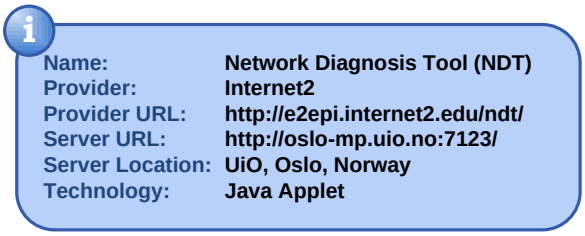
- Comparing TCP transfers using the well-know BitTorrent port 6883 and the non-BitTorrent port 10011:

TCP Transfer	BitTorrent port	Non-BitTorrent port	Conclusion
Download #0	8465 Kbps	7731 Kbps	No rate limiting
Download #1	8308 Kbps	8884 Kbps	No rate limiting
Upload #0	4088 Kbps	4076 Kbps	No rate limiting
Upload #1	5481 Kbps	4821 Kbps	No rate limiting

Figure 6.9: Glasnost result page, showing no BitTorrent throttling detected.

6.1.6 Network Diagnosis Tool (NDT)

The Network Diagnosis Tool (NDT) is in fact a tool that belongs in between our two categories. It can be characterized as an on-line tool, because when the server is set up, the user can do all the testing from within the web browser. But the tool requires that either the user or an administrator sets up a server, which is a non-trivial task. Therefore it also exhibits some characteristics that fit our category of stand-alone tools. But when the server is set up, the user does not need to know anything about the server other than the URL, therefore we have chosen to describe this tool in this section. UNINETT have instances of the NDT server installed on their servers in many locations. These servers are called "Målepåle" in Norwegian, which directly translated means "measurement pole". These NDT instances are accessible from the public Internet, and the address for one of them is shown in the box below.



The system is composed of a Java applet client program[4] and a pair of server programs that comprise a web server and an analysis engine [12]. The server programs must be installed on a Linux server with a Web100 enhanced kernel to capture TCP kernel variables during the test. These variables are used in the detection and calculation process. The requirement of a modified Linux kernel complicates the task of setting up the server.

The NDT test tries to determine three characteristics of the link between the server and the client. The first is to identify the speed of the slowest link on the end-to-end path. The second is whether the Ethernet duplex setting is full or half. The third is whether congestion is limiting the end-to-end throughput or not. The test can also identify two network error conditions, duplex mismatch and faulty cables respectively.

When the user starts the test, the server spawns a child process to handle the communications with the client. Then the client streams data over TCP for 10 seconds to the server to measure upload bandwidth. When the upload test is completed, the client starts to download data over TCP for 10 seconds. During the transfer the server captures the TCP kernel variables, and transmits these back to the client. The client then runs some detection algorithms to decide the characteristics of the end-to-end path. These algorithms are described in detail in [11]. A screen shot from the test window is shown in figure 6.10.

The results are presented to the user in multiple layer of detail. The most general results are shown in the main window. These are upload- and download speed.

[4]The NDT also provides a command line based client that actually is a modified version of the Iperf tool. We will not look further into that here.

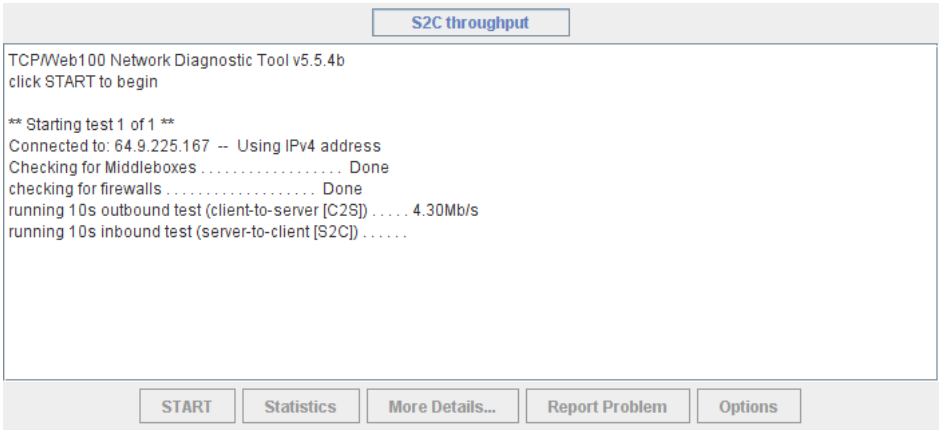


Figure 6.10: The main window of the NDT test during test procedure.

The link type of the slowest link on the end-to-end path is also identified. For more detailed information, the user may click the button "Statistics". This window show more information that can be interesting for the advanced user. The button "More Details..." opens a window with information that may be useful for an expert network administrator. This window shows the value of all the TCP kernel variables that has been captured.

The measured results are combined with calculated values derived from the TCP kernel variables at the server to determine if the throughput bottleneck is in the NDT server, the client computer or the network infrastructure. NDT can identify whether the network infrastructure is working properly, but it cannot identify where in the path a faulty or slow link is located [11].

The NDT tool is probably a little too complicated for the average broadband subscriber. But for broadband customers with an above average interest and understanding of computer networks it can provide useful information about the Internet connection, as long as they can get access to a server already running this tool. To setup the tool, and utilize all its possibilities you would probably need to be characterized as an expert user.

6.1.7 Discussion of On-line Test Tools

There exist a lot of on-line broadband test tools, and a selection of them has been presented in the previous sections. We have tried to select tools that differ as much as possible, since tools of this type are quite similar. In this section we will discuss the characteristics of the tools presented earlier. A comparison of some of the key attributes of the test are shown in table 6.1.

As we can see from table 6.1, all the tests measures download- and upload bandwidth. All tests except Speedometeret and Glasnost measure RTT. Glasnost, which actually is a different kind of tool than the others, is the only tool able to detect discrimination of the BitTorrent application protocol.

We started with Speedometeret from ITavisen. This is a very simple bandwidth test which measures the download and upload rate at the application layer. The test only measures the goodput seen from the HTTP point of view, not taking HTTP-overhead into account. This will result in an underestimation of the actual throughput, since the calculations only uses the size of the transmitted file, and not the total amount of transmitted data including HTTP-header data. This service is also available from other Norwegian sites hosted by Aller Internet. But when accessed from these other sites, the name of the service is "Surfometeret". We think this name is more suitable, since web surfing speed is what this test actually measures. Time used to transfer a file over HTTP cannot be used to determine how the connection behaves when used with other types of applications like streaming, VoIP or other. Another factor that limits the usefulness of Speedometeret is the lack of RTT/latency measurements. Response time is an important parameter in web surfing performance evaluation, and this parameter has been omitted in this tool. The time needed to transfer data of a given size is determined by the bandwidth. The time it takes before the transfer starts is given by the delay. A subscriber would probably not be satisfied with a connection with a large bandwidth if the average delay is several seconds. This will result in poor responsiveness and give an impression of a bad Internet connection, although the Spedometeret claims that the connection is as "fast" as promised by the ISP.

Two other tools that also use the HTTP-layer to handle the transfer of data are Bredbandskollen and Speedtest. We have used Wireshark to analyze the packets that are transferred during these tests, and found many similarities. The names and the structure of the files used in the download tests are exactly the same, and the pages used for HTTP POST in the upload tests are also identical in both tools. Therefore we suspect that these two tools are based on the same core, and just presented to the user with two different user interfaces[5]. The upload page used in both Bredbandskollen and Speedtest includes the size of the total HTTP POST message. We believe that this is used in the calculation of the upload speed, and this will increase the accuracy compared to Spedometeret that does not take the HTTP-overhead into account. Both these tests also only measures how well the connection works for web-surfing, and omits other applications.

MySpeed, NDT and Glasnost all use TCP directly for the data transmission.

[5]Later we have received internal NPT documents on the planned NBTT that also support our early suspicion. They plan to use the same measurement engine as Bredbandskollen developed by Ookla.

This means that the goodput transmitted on the TCP layer is what is measured, and not the slightly smaller amount of data transmitted in the HTTP layer as with Speedometeret. This will provide a slightly better level of accuracy. This type of measurement is a bit more application independent, but it still limits its predictability to applications that use TCP. So this kind of test cannot be used to test how well an application based on UDP will perform.

As far as we can see from our analysis, all the tests use some kind of transferred data/time to calculate the bandwidth of the connection. This method assumes that the transfer is able to instantly after startup consume 100 % of the bandwidth. Figure 6.11 is a sketch that illustrates how a TCP connection acquires bandwidth over time as a result of slow-start. It is apparent that the bandwidth

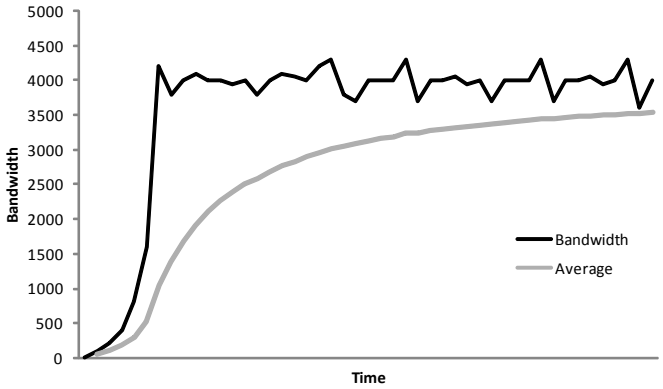


Figure 6.11: Example of TCP adaption to available bandwidth

utilization is not 100 % during the entire transfer. There is a transient slow-start period to begin with, and this can result in underestimation of the bandwidth if not special care is taken. We see from the average curve that a simple average is useless if the download period is too short. The duration of the slow-start phase is dependent on many parameters where RTT is the most important one. A TCP connection will always have a slow-start phase in the beginning even though it can be very short.

In bandwidth measurements the trick is to make sure the total time spent on transmitting the data is much greater than the duration of the slow-start phase. In this way the transient slow-start phase can be neglected since it only marginally affects the result. The tests we have evaluated solve this in different ways. Speedometeret, Bredbandskollen and Speedtest does a pre-calculation, and based on this value chooses a file that is large enough. In addition Bredbandskollen and Speedtest use a *sliding average* that efficiently eliminates the slow-start error by only including the last part of the download in the calculations. MySpeed and NDT that operates on the TCP layer have defined the transfer of data to last for 8 and 10 seconds respectively. Both these solutions prevent the slow-start in messing up the result. As mentioned in section 6.1.1 the JavaScript version of Speedometeret only downloads a predefined picture file. The problem is that one file size is only suitable for connections within a limited bandwidth interval. A

large bandwidth connection will result in completion of the download before the slow-start period is finished. A small bandwidth connection will result in a test that need a lot of time to finish. Neither is good for the customer, so we think adaption of file size is totally necessary in such tests.

We did an experiment with the Iperf[6] tool to illustrate the effects the file size may have on the reported bandwidth. In this experiment we used a computer connected to the network at NTNU as an Iperf server. This server should in theory have a 100 Mbit/s Internet connection. We also used a server placed in Oslo in the network of UNINETT as an Iperf client. To measure the bandwidth between the server and client we first ran an Iperf test for 20 seconds. This reported a bandwidth of 92029 kbit/s. Then we ran Iperf tests with defined transfer sizes from 128 KB up to 15 MB. The results of these tests are plotted in figure 6.12. The bandwidth reported in the 20 seconds test, are shown with the dashed line.

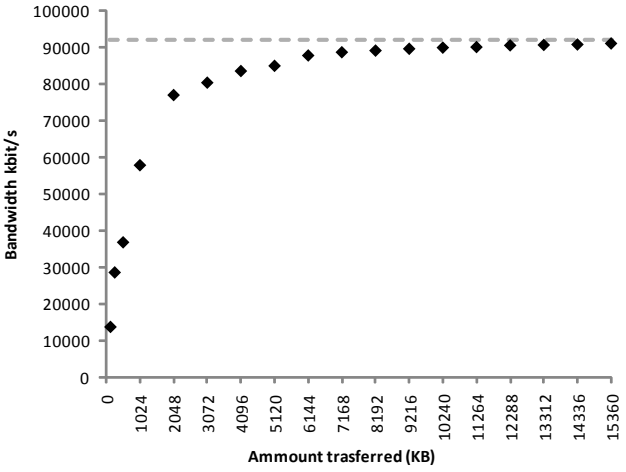


Figure 6.12: Bandwidth reported by Iperf when transferring different amounts of data per measurement.

As we can see from figure 6.12, the reported bandwidth is dependent on the amount of data transferred per test. With an Internet connection of almost 100 Mbit/s we see that a transfer of at least 10 MB are needed to get a result that are close to the "real" bandwidth. But even 15 MB is not enough to get an accurate result. It is also worth noting that the transfer of 15 MB took about 1,4 seconds to complete. So one could easily increase the download size further without increasing the test time beyond what is tolerable. When bandwidth tests are designed, they must be usable for Internet connections with quite different bandwidth characteristics. Our experiment shows that adjustment of download size in accordance with the bandwidth of the connection is essential.

One difference between the different tests is the number of connections used during the test. Speedometeret, MySpeed and NDT only use one TCP connection for

[6]The Iperf tool is presented in section 6.2.4.

download and upload, and thus measure the BTC. When testing a connection with a low packet loss, this will probably give an acceptable result. But if the loss probability becomes significant, the throughput of a single TCP connection might be considerably degraded. This is because dropping of an arbitrary packet will cause the single TCP connection to slow down as a result of the congestion control mechanism shown in figure 3.5. If multiple connections are used, dropping of one random packet will only cause one of the connections to slow down while the others continue at full speed. A test that uses multiple connections can therefore be considered more robust to packet loss as long as the loss rate is relatively small.

From table 6.1 we see that some of the tests allow the user to select which server the test shall be run against. Bredbandskollen has a few different test servers distributed in Sweden. Since this test is targeted at Swedish broadband customers, they only have servers in Sweden. Both Speedtest and MySpeed have many test servers distributed all around the globe. They both also have servers in Norway. The possibility to select server location can be useful for end-users that communicate a lot with hosts at specific geographic locations. One example is subscribers that use their broadband connection for on-line gaming. They will often need to connect to foreign servers, and a test to reveal the connection speed and latency against the country of interest may be of great value to these end-users. Other subscribers may have a lot of VoIP correspondence with persons located in different countries. A test server near the person they wish to communicate with can be useful if the subscriber wishes to test the quality the Internet connection can provide to the location of interest.

	Speedometeret	Speedometeret JS	Bredbandskollan TPTEST	Speedtest	MySpeed	Network Diagnosis Tool (NDT)	Glasnost
Measures/detects							
Download rate	X	X	X	X	X	X	X
Upload rate	X		X	X	X	X	X
Round trip time			X	X	X	X	
BitTorrent discrimination							X
Measurement parameter							
Bulk transfer capacity	X	X			X	X	
Acheivable throughput			X	X			
Implementation							
HTTP-based transfer	X	X	X	X			
TCP-based transfer					X	X	X
# connections download	1	1	2	2+	1	1	Many[3]
# connections upload	1	N/A	2	2+	1	1	Many[3]
Server select by user	No	No	Yes[1]	Yes	Yes	Yes[2]	No
Java applet	X				X	X	X
Flash			X	X			
JavaScript		X					

1. A few servers in Sweden can be selected

2. You can setup your own server, or use a server someone else has setup. This requires that you know the URL for the specific deployment.

3. This test checks for discrimination between transfers on different ports, so many connections needs to be used.

Table 6.1: Comparison of on-line test tools.

6.2 Stand-alone Tools

In this section we will look into broadband evaluation tools which are more advanced and usually require the tester to do something more than just to push a button. We will start with the basic tools, ping and traceroute, which are quite simple, but might still be some of the most important network evaluation tools ever invented. We will further dive into more advanced tools like Iperf and Netperf. These tools offers us a powerful way to measure different network performance parameters using either TCP or UDP. We will finally introduce two tools used to estimate the available bandwidth, pathload and Abget. Abget is interesting because this is the first tool to estimate the available end-to-end bandwidth which does not depend on a special server process. A simple web server holding a file with a suitable size is all Abget needs at the other end.

6.2.1 Ping

Ping is a tool for checking whether a host is reachable, measuring the RTT to a specified host and ping will also record any packet loss. The tool is included in almost any operating systems, including Windows and Linux.

Ping works by sending an Internet Control Message Protocol (ICMP) Echo Request/Reply messages, defined in RFC792 [58]. As seen in figure 6.13 the source sends an ICMP Echo Request message which includes a timestamp. This information is included in the ICMP Echo Replay message. From this the RTT can be calculated by the sender. Ping is usually performed a few successive times against the destination. From this statistics like maximum, minimum and average RTT are recorded as well as the loss rate.

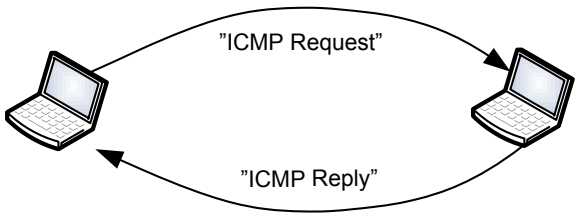


Figure 6.13: Functional overview of Ping. The source sends an ICMP Request message, and the receiver responds with an ICMP Reply message.

Ping can be performed without cooperation of the destination. The only requirement is that the ICMP Echo Request message is not filtered out on the path or at the destination. Unfortunately it is becoming more common to filter out this type of messages to avoid certain types of Denial of Service (DoS) attacks.

6.2.2 Traceroute

Traceroute is a network tool which can be used to determine the route taken by IP-packets across an IP network. In order to perform a traceroute the source host utilize the TTL-field of the IP-header and attempts to get an ICMP TIME_EXCEEDED response from each router along the path. Traceroute starts with sending a probe packet to the first hop, with a TTL equal to one. The first hop will then reply with TIME_EXCEEDED response message. This happens because each hop along the path should decrement the TTL field of an IP packed, and report back to the originating host when the TTL reaches zero. RTT can be calculated because the source knows when the request message was sent and when the TIME_EXCEEDED response is received. New probe packets are sent to succeeding hops with an increasing TTL value and the listen for the TIME_EXCEEDED response message sent from each hop along the path.

In figure 6.14 we have shown a traceroute example from UNINETTs server `ytelse2.uninett.no` to `www.vg.no`.

```
larsivar@ytelse2:~$ traceroute www.vg.no
traceroute to www.vg.no (193.69.165.21), 30 hops max, 40 byte packets
 1  oslo-gw4 (128.39.3.201)  0.423 ms  0.306 ms  0.355 ms
 2  oslo-gw3 (128.39.65.81)  0.288 ms  0.312 ms  0.248 ms
 3  stolav-gw1 (128.39.46.253)  0.297 ms  0.295 ms  0.255 ms
 4  xe-4-2-0.br1.osls.no.catchbone.net (193.156.120.3)  0.532 ms  0.549 ms  0.480 ms
 5  v4092.rs2.m323.no.catchbone.net (193.75.1.142)  0.654 ms  0.657 ms  0.549 ms
 6  193.69.165.11 (193.69.165.11)  0.669 ms  0.635 ms  0.672 ms
 7  193.69.165.11 (193.69.165.11)  0.672 ms  0.651 ms  0.679 ms
larsivar@ytelse2:~$
```

Figure 6.14: Example of a traceroute, we see all hops in the path, and the individual delays.

6.2.3 Hpcbench

Hpcbench is a Linux based network performance evaluation tool written in C. It can be used to measure the delay (RTT) and achievable throughput between two host [29].

UDP Communication
The UDP communication part of Hpcbench allows the user to perform latency tests (aka UDP ping) as well as UDP throughput tests.

TCP Communication
The TCP communication part allows the user to perform latency tests (aka TCP ping) as well as TCP throughput tests.

6.2.4 Iperf

Iperf is a quite sophisticated network performance testing tool. It is an open source project written in C++ and is able to create TCP and UDP data streams. This can be used to measure the throughput of a network. Iperf also allows the tester to tune certain parameters of the respective protocols enabling testing of a network, or alternatively for optimizing and/or tuning the network. Figure 6.16 shows JPerf which is a graphical front end for Iperf, and is written in Java. In addition, JPerf also provides graphs over the measured bandwidth.

Name:	**Iperf**
Provider:	**The Iperf Team**
URL:	**http://sourceforge.net/projects/iperf**
OS:	**Linux, Free BSD, MAC OSX, Windows**
Technology:	**C++**

Iperf can measure the following network parameters:

- **Bandwidth (uni- or bidirectional)** – measured with TCP-tests.
- **Maximum Segment Size (MSS)** – optionally print TCP maximum segment size (MTU - TCP/IP header). This feature is only supported on UNIX based systems.
- **Jitter** – measured with UDP-tests.
- **Loss** – measured through UDP-tests.

A client host and a server host needs to be set up in order to use Iperf. The client process must know the IP-address and port of the server process. The Iperf set up is illustrated in figure 6.15. Iperf can measure the throughput between the two ends, both unidirectional and bidirectional. The client can also specify if bidirectional bandwidth should be measured simultaneously or sequentially. Test duration can be specified in seconds or amount of data to transfer. The client may also specify the data to be transferred during the test.

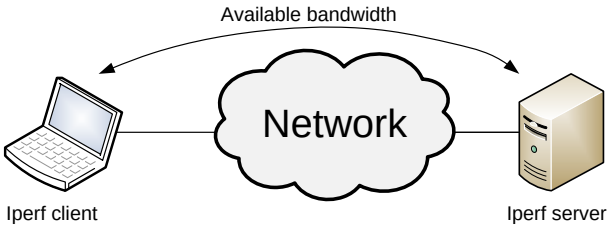


Figure 6.15: Iperf system setup.

TCP options
When running a TCP test, the tester may tune the following parameters:

- **Buffer length** – Specifies the length of the send/receive buffer.
- **TCP Window size** – The amount data that can be buffered during a connection, without an acknowledgment from the receiver.
- **MSS** – The maximum amount of data that can be transmitted at one time. Usually equal to MTU-TCP/IP-header[7].
- **TCP no delay** – disable Nagle's Algorithm. Nagl's algorithm collects a lot of small outgoing packets and sends them all at once. The reason for doing this is more formally described in RFC896 [42].

Default operating system values will be chosen if the tester does not specify the above mentioned parameters. By allowing the tester to manually specify these values, it is possible to tune the host-network configuration, e.g. , to achieve high rates over paths with large bandwidth-delay product). Iperf can optionally report back intermediate results in specified intervals (in seconds). When the test completes the program will report a final report containing the amount of data transmitted, duration and the measured bandwidth.

UDP options

When running UDP test the tester may tune the following parameters:

- **UDP Bandwidth** – The rate to send/receive at.
- **UDP Buffer size** – Specifies the length of the send/receive buffer.
- **UDP Packet size** – Specifies the size of each packet.

At the end of a UDP test the client sends a FIN message, signaling that the client has finished and the server responds with a packet containing measured statistics. The statics includes amount of data transferred, average bandwidth, number of lost packets and measured jitter. If the server does not respond to the FIN message, the client will resend it 10 times, before timing out.

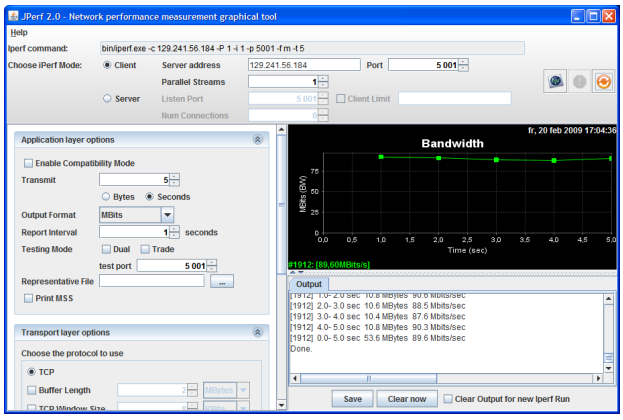
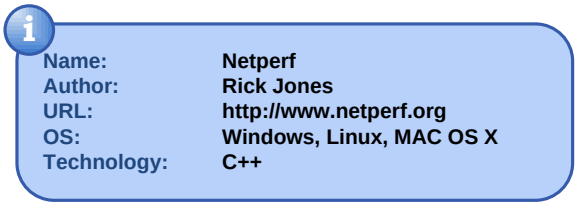


Figure 6.16: Bandwidth test: JPerf (Screen shot).

[7]On Ethernet/LAN: 1500bytes (MTU) - 40bytes(header) = 1460 bytes.

6.2.5 Netperf

Netperf is another network TCP/UDP performance benchmark tool. Its primary focus is on bulk data transfer and request/response performance using either TCP or UDP over standard sockets [35]. Netperf is a freely available open source tool written in C++.



Netperf is designed around the same client/server model as described for Iperf in section 6.2.4. One host is running a server process that the client processes on other hosts can connect to. In Netperf the client process will first establish a control connection over TCP regardless of the test being run. This connection is used to pass test configuration parameters from the client process to the server process and deliver results back to the client when the test finishes. There will be no traffic on the control channel during the test. Netperf can be used to benchmark different aspects of the network, and we will in the following go though the most common of them.

TCP Stream Performance

This is a common TCP network performance metric and is also referred to as BTC. The test measures the unidirectional (from client to server) TCP bulk transfer throughput under different settings. It is possible to change the following settings in order to "tune" TCP:

- **Socket buffer size** – Specifies the sender and/or the receiver buffer size. The TCP window size will usually be the same.
- **Message Size** – Is used to set send and/or receive message size. Using equal sized messages is a common way to distinguish messages sent over TCP.
- **TCP no delay** – Disable Nagle's algorithm, described in section 6.2.4.
- **Length** – Is used to specify the duration of the test. Can be specified in terms of seconds or in bytes to be sent.

If none of the above mentioned parameters are changed, the test will run a TCP stream test for 10 seconds with default system values for the TCP options. When the test completes it will report the measured throughput in bit/s.

UDP Stream Performance

The UDP stream performance test is very similar to the TCP stream test, with the difference that the messages are sent over UDP, an unreliable transport protocol. The available settings, used to "tune" UDP, are the same as described for TCP

(with the exception of "TCP no delay" parameter). It is important to notice that the socket buffer size must be larger than or equal to the message size, when running this performance test.

When the UDP stream test completes, the sending rate, the receiving rate, the number of sent messages, the number of received messages and the number of messages containing errors are reported back. The number of lost messages will then be the difference between the number of sent and received messages.

TCP Request/Response Performance
Request/Response is another performance metric which can be evaluated with Netperf. In Netperf this is measured as transactions per second for a given request and response size. It is still possible to specify the socket buffer size and the test duration, as for the stream test. The message size from the stream test is replaced with an option for specifying the request and/or the response size. When the test completes it gives the measured number of transactions per second, the TCP throughput (in/out) as well as the average RTT for the transactions.

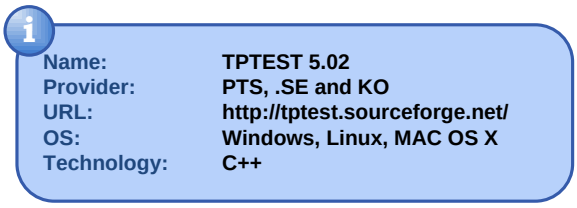
TCP Connect/Request/Response Performance
This test is similar to the previously described Request/Response test, with the exception that the test establishes a new connection for each request/response pair. The idea is to mimic the behavior of the HTTP 1.0 protocol[8].

UDP Request/Response Performance
This test works just like the TCP Request/Reply performance test, with the exception that the messages are transferred using UDP.

[8]HTTP 1.1 in use today have persistent connections and therefore does not establish a new TCP connection for each request.

6.2.6 TPTEST 5.02

TPTEST 5.02 (from now on called TPTEST) is the predecessor of Bredbandskollen and is a tool for evaluating a subscriber's broadband connection. This is a more complex tool that requires more knowledge from the tester. TPTEST is an open source tool written in C++ and can be compiled for almost any operating system including Windows and Linux.



The TPTEST tool depends on a TPTEST server in order to perform the bandwidth measurements. The tool automatically retrieves a list of available TPTEST servers[9]. All of the current servers are located in Sweden. The tool can be used to measure different aspects of the broadband connection and we will in the following present each one of them.

Availability

This test simply verifies that certain known web pages are available. The tester can also define own pages to test against.

Response time and Jitter

This test sends 20 probe packets of 100 bytes and measures the response time of each packet. The variation in the response times, also known as jitter, is presented in the result together with packet loss count and average, maximum and minimum response times.

Bandwidth

This test is designed to measure the upload and download rate. In order to measure the rate the tool starts with a small amount of data. If the test completes to fast, it will increase the amount data until the test last long enough to reach maximum capacity. The final result for upload and download rate is presented to the tester.

Self designed test

In addition to the predefined tests the tester might also define own test-cases[10]:

- **TCP test** – The tester can specify the TPTEST server to test against and the duration of the test in bytes or seconds.

[9] The updated list of servers is found on this location: `http://referens.sth.ip-performance.se/tptest3serverlist.txt`.

[10] We must ad that we had several issues with these tests, and the tool crashed a lot during the testing.

- **UDP test** – The tester can specify the TPTEST server to test against, number of packets per seconds, the packet size and the duration of the test in seconds.

TPTEST also records the results from each test above, and presents the user with comparison with previous result as well as minimum, maximum and average values for each test. In addition TPTEST includes a few common system tools like netstat, ping and traceroute in addition to the above mentioned tests. These tests are executed by calling the system tools, and TPTEST presents the results within its own graphical user interface.

6.2.7 Available End-to-End Bandwidth Tools

There has been made a lot of tools trying to estimate the available end-to-end bandwidth[11]. Because many of them do roughly the same with minor differences on implementation we will only introduce the most important ones. It is also a known fact that many of these tools are rather inaccurate and takes a long time to complete.

Pathload
Pathload is a tool used to estimate the available end-to-end bandwidth from one host (sender) to another host (receiver). This tool is based to the SLoPS technique described in section 5.3.7. As explained SLoPS tries to determine if the current rate is larger than the available bandwidth by looking at the variation in one-way delay. Pathload introduces the concept of a fleet of streams. Each fleet contains many streams, used for sampling.

In Pathload the sender transmits a fleet of N streams of UDP packets to the receiver at a certain rate. At the receiver side Pathload looks at the variation of one-way delay (jitter) of successive packets in each stream. Inter-arrival jitter can be estimated according to appendix A.8 in RFC1889 [26]. If a large portion of the streams in one fleet has an increasing trend the entire fleet is said to have an increasing trend and the next fleet will have a lower rate than the current fleet. Pathload terminates when [16]:

- The rate of two successive fleet is less than a user-specified resolution, or...
- ...the available bandwidth varies in a "grey area", which is larger than the user-specified resolution.

Pathload reports available bandwidth in terms of a range with a lower and upper limit. The center is the average available bandwidth measured during the test and the edges represents the variation of the available bandwidth.

The reader is referred to [16] for more detailed information about Pathload.

Abget
Abget is another tool for estimating the available end-to-end bandwidth from

[11] E.g. pathChirp, Spruce, IGI/PTR, cProbe and others.

one host to any other end host running a web server. The interesting part of this tool is that it is able to connect to any TCP based web server on the Internet in contrast to most other available bandwidth estimation tools which requires the tester to have access to both hosts.

Abget is based on an iterative algorithm that is similar to SLoPS, just like Pathload. Because there is no way to force TCP server to send packets at a certain rate the Abget had to implement a work around. The idea is to use a limited advertised window and "fake" ACKs. The ACKs are considered fake because they are sent in advance of the incoming data segment in order to control the sending rate from the server.

Abget depends on a large file at the web server used in order to get accurate estimates. The more information the user includes about the expected results, the faster Abget completes the testing process.

Abget is also able to measure the available upload rate, from the user to the server. This is done by sending HTTP requests in many overlapping segments. This step is repeated at different rates in order to estimate the available bandwidth in the upload direction.

The tool report back the measured available bandwidth within the requested resolution defined by the tester running Abget. The interested reader is referred to [15] to get more information of how Abget is implemented and how this tool works.

6.2.8 Discussion of Stand-alone Tools

In this section we have introduced a selection of important active stand-alone measurement tools. In our studies we found a lot of available tools and we have tried to introduce tools that evaluate different network characteristics. Where no documentation is found we have used Wireshark in order to find out how the tools works, what protocols they use and what messages that are exchanged. Even though some of the presented tools, like TPTEST, Iperf and Netperf, try to cover many of the different network parameters, none of them covers all aspects. The tools also try to evaluate many of the same aspects with different approaches. All the stand-alone tools evaluated, except Abget, are implemented to use regular transport protocols (TCP, UDP or ICMP) without any specific application protocol on top. This enables us to evaluate the network performance without the behavior of application protocols affecting the obtained results. Even though tools like Iperf and Netperf only depends on regular transport protocols we are still able to use these tools to mimic the behavior of application layer protocols, like HTTP. This is possible because they are quite configurable and it is easy to specify the content through well defined scripts.

Ping is an excellent tool for checking the bidirectional connectivity from one host to another. If we use ping to measure the RTT we need to be aware of that we measure the RTT at the ICMP level and that the end host is not set to respond to these messages in a timely manner (e.g. if the host is busy with something else) as mentioned in section 6.2.1. Usually ICMP processing is implemented at the network level of the operating system, which means that the host does not need to involve the application level in order to issue a ICMP-Response. This gives good response times even though the end-host will only respond "as fast as possible". Ping has been studied and found to be quite stable in a controlled environment [4]. They found the Linux version to be accurate to +-0.1 ms while the Windows XP ping reported RTTs between 0 and 1 ms smaller than the passively measured RTT.

It is important to be aware of that the RTT measured with ping is **not** an estimate on how fast we could expect response from a web server, which requires set-up of many TCP-connections before returning a response. Ping also suffers from the fact that many operators give a lower priority to ping messages or even worse, block them completely. Because of this there exists a high uncertainty of the measured RTT values from the ping application in an uncontrolled environment, such as the Internet.

Traceroute is a great tool for finding the path traversed by packets through the network. The tool itself relies on the same ICMP protocol, just like ping. Traceroute is used to measure the RTT to each hop along the path. From this we can find which links in our path contributing the most to the delay.

Loss can also be measured with the ping application. Ping records the number of packets sent, and number of received responses, where the difference is the experienced loss. Ping is excellent for measuring the loss rate on a poor access-connection, e.g. a bad wire or poor wireless connection. It is also important to recognize that small packets (less than MTU) gives lower loss rate than larger packet (larger than MTU). This is because larger packets are fragmented

into many smaller chunks when transmitted over the network. In order not to experience a loss it is required that all the independent chunks are transmitted correctly and that none of the chunks are lost. Using ping to measure the overall loss probability from one-host to another over the Internet is another story. The challenge is that a lot of ISPs does not prioritize ICMP packets, used by ping. This means that it is a higher probability that these messages are dropped in case of congestion. When most of the regular traffic is transported with UDP or TCP, it is not a good idea to measure the overall loss rate between hosts with ICMP packets.

Iperf and Netperf have another approach in order to measure the overall loss probability. These tools uses UDP to measure the overall loss rate. This is a better approach because UDP is often used to transmit time-sensitive information where loss will affect the obtained quality[12].

Measuring the overall loss rate with TCP is an impossible task. This is because the protocol in itself hides the experienced loss from the application utilizing TCP. But the loss rate will affect the TCP stream and force it to lower the transmission rate, as explained in section 3.4.

TPTEST uses ping to measure the RTT, jitter and loss rate and presents the results in its own graphical user interface. This means that TPTEST does not offer anything more than the regular ping application. Actually it offers less, because you are not able to specify any parameters, such as message size or the number of probe packets.

Iperf, Netperf and TPTEST are all tools which can be used to measure the BTC explained in section 5.3.8. While Iperf and Netperf are applications which allows (or requires) the user to set-up its own server, TPTEST depends on the pre-established TPTEST-servers all over Sweden. For advanced network debugging and performance tuning it is essential to be able to define both hosts in the end-to-end path being evaluated. For subscribers sitting at home only wanting to check the quality of their own access-connection it is essential that the other side is already established and waiting for connections. But the server location is still important and will affect the obtained results. A user located in Norway running a BTC test to a server located in Sweden will get a measure on the capacity for the whole path. We must recognize that characteristics like the available capacity between the two countries could affect the obtained results.

It is also possible to use Iperf and Netperf to measure the achievable bandwidth. Both tools offer two different approaches:

1. Use multiple TCP sessions at the same time. This will give the application a higher share of the path-capacity because of the TCP congestion algorithm described in 3.4.

2. Use UDP to transport the data. By using UDP at a higher rate than the available bandwidth, other protocols implementing congestion control will back off, while UDP will continue at the same rate, despite of packet losses.

[12] Again we should be aware of that some ISPs could be interested in down-prioritize non-TCP-friendly protocols such as UDP to increased the experienced quality of TCP based applications.

Both Iperf and Netperf allow the user to define the content of the messages, which makes it possible to use these tools to mimic application layer protocols. This is especially interesting in the case where we would like to evaluate how the network can handle different application scenarios.

Available bandwidth can be measured with tools like abget and pathload. The motivation for these tools is to estimate a paths' available bandwidth, and to be as little intrusive as possible. It is important to notice that available bandwidth differs from BTC as we have explained in section 5.3.8. This metric is beneficial for many reasons. Firstly applications utilizing UDP as transport protocol needs to know what rate they can send at, as UDP does not implement any flow-control. Secondly it could allow automatic update services to run in the background and only use the available bandwidth and thus not disturbing the rest of the network traffic.

Pathload is a fast and accurate available bandwidth estimation tool [33]. In Pathload the sender transmits periodic UDP to the receiver at a certain rate. Abget has another approach where the client is able to estimate the available bandwidth to "any" web server on the Internet holding a file with a suitable size. The idea is to make available bandwidth estimation as easy as ping and traceroute. Abget is able to do this because this tool modifies the TCP behavior and utilize *ACKs*. The downside of this is that this requires abget to run with root privileges and is dependent on libraries that only are available on UNIX based systems. Abget also requires the user to specify upper and lower limits of the expected results in order to get an accurate result. Both tools estimate the available bandwidth to be in a certain range, often referred as a "gray area", which indicates the uncertainty of the obtained results. In a network with live traffic it is almost impossible to get a correct estimate on the available bandwidth, as this will vary over time.

In table 6.2 we have tried to summarize the tools introduced in this section. From the table we can see what network performance metrics the different tools covers, and what protocols they use in the evaluation.

From this we can conclude that there is not any tool today that evaluates all aspects of a broadband connection. We have also seen that the different aspects can be evaluated with different approaches.

	Ping	Traceroute	Iperf	Netperf	TPTEST 5.02	Pathload	Abget
Measurement metric							
Round Trip Time	X	X		X[1]	X[2]		
Loss	X		X	X	X[2]		
Jitter	X		X		X[2]		
Bulk Transfer Capacity			X	X	X		
Achievable bandwidth			X[3]		X[4]		
Available bandwidth						X	X
Request/Response perf.				X			
Protocol							
TCP			X	X	X		X
UDP			X	X	X	X	
ICMP	X	X			X		

1. RTT measured for Request/Response
2. Measured with the *ping* tool
3. Measured with parallel TCP connections or a UDP connection
4. Measured with a UDP connection

Table 6.2: Comparison of stand-alone test tools.

Chapter 7

Evaluation of Existing Tools

We have in the previous chapter introduced a lot of different available tools which subscribers can use to evaluate their broadband connections. In this chapter we want to find out the accuracy of the different on-line broadband test tools. The evaluation has been done by running all the test tools multiple times at Internet connections with various data rates and compare the obtained results with optimal and calibrated Iperf-measurements performed against a server located near NIX. The server is located within UNINETT, one hop away from NIX. We assume that this single hop is neglectable, and consider the server to be located at NIX. This location is equivalent with the planned location for the first NBTT server.

7.1 Test Setup

We will in this section describe the test setup illustrated in figure 7.1. From the figure we can see that the Test Computer connects through the Bandwidth Limit Server, which operates as a network bridge that also control the traffic rate flowing through this bridge. If traffic is coming too fast, the server will start dropping packets, to ensure that the traffic rate does not exceed the specified limit in both directions. The Bandwidth Limit server is an attempt to simulate a regular ADSL broadband connection. This test setup allows us to use any type of computer hardware supporting regular networking as the Test Computer.

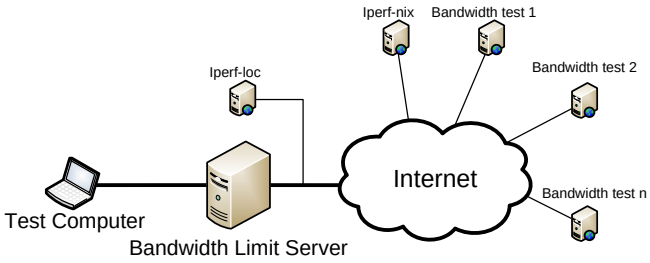


Figure 7.1: Functional overview of the Test Setup.

7.1.1 Bandwidth Limit Server

In order to evaluate the different tools at different data rates we need a way to limit the download and upload rate. This makes it possible to simulate different broadband access networks at different speeds. Today there exist a lot of different applications which can be used to limit, or shape, the traffic to a specified rate. We tested a lot of different shaping tools, including Traffic Shaper XP Client [14], NetLimiter [65], Trickle [19] and Wonder Shaper [30], but found that all of them are quite inaccurate, unstable and delivers unpredictable results. The problem with these applications is that they operate in user mode, which add extra variable delay and CPU scheduling problems. To account for this they usually let through more traffic if the traffic is bursty or if they use many TCP-connections like Speedtest do, see section 6.1.3. These limitations makes it impossible to compare the different bandwidth test tools, as some of them uses one connection, while others uses multiple. We need a rate limiting technique that allows us to limit the total traffic rate to a specified rate, whether it is one flow or multiple simultaneous flows.

Traffic Control

Our solution was to use the Linux based tool Traffic Control (tc) [10]. This is a tool used to configure the Traffic Control module in the Linux kernel. This tool can be configured to, among many other things, shape the traffic to a specified rate using a Token Bucket Filter (TBF) which utilize a Random

Early Detection (RED) queue discipline. The tc tool is a stable shaping tool which delivers accurate and predictable bandwidth rates. The Linux server is configured as a network bridge, and limits the traffic in both directions according to a specified rate. We have illustrated the bridge configuration of the server in figure 7.2. By adding a RED queue and a TBF with a defined output rate we can control the rate in and out of our bridge. This will basically emulate a typical slow ADSL link. The server also uses a queue in both directions[1], which will add some extra delay to the sent and received packets. This delay should account for some of the delay one should expect from a regular ADSL access connection.

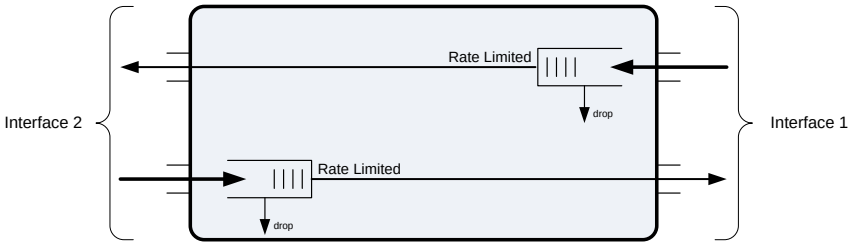


Figure 7.2: Network interfaces of the Bandwidth Limit Server.

The bridge set up script is included in appendix A.1. When this script is executed it will create a bridge between the two interfaces of the Linux Server. To enable bandwidth rate limiting we have written another script, included in A.2. This script takes the wanted bandwidth rates as parameters (in kbit/s) and makes it very easy to switch between different bandwidth levels.

RED Queue

As mentioned above we use a RED queue discipline [22]. RED is an active queue mechanism that tries to keep the average throughput high and the average queue size as low as possible. In regular tail drop queue disciplines the server would queue as many packets as possible and drop packets as soon as the queue is filled up. When a packet is dropped, this is a signal to the ongoing TCP connections to back off. If there are many simultaneous TCP flows, they will all be asked to back off at the same time. This will lead to a fluctuating traffic pattern, where the network are constantly congested or underutilized in turns [81].

RED notifies TCP that the queue is filling up by randomly dropping packets with a probability calculated from the average queue size. Figure 7.3 illustrate how a RED queue operates. When a packet arrives, the average queue length is calculated. If the average length is below the minimum threshold the packet is added to the queue. When the average size is between the minimum and maximum threshold the dropping probability is calculated. The dropping probability grows when the average queue length grows. If the average queue length is above the maximum defined threshold the packet is dropped.

[1] Also NextGenTel uses queues in both directions for their ADSL subscribers.

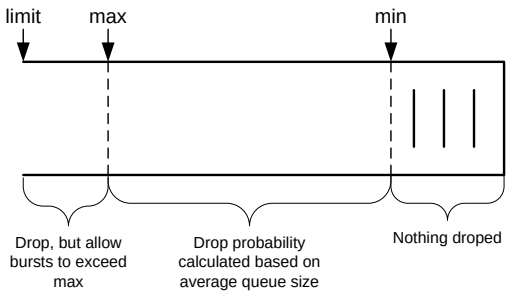


Figure 7.3: Random Early Detection Queue Discipline

7.1.2 Test Computer

The test computer can be any computer supporting regular IP networking. This is because we use a dedicated Linux Server as a network bridge implementing the rate limiting functionality in both directions. We have chosen to use a computer running Windows XP and the on-line test tools have been executed within the Internet Explorer browser (because of Flash upload rate issues introduced in section 7.3.2). We closed all programs/services we did not need to prevent other programs from affecting our results. We also used the passive bandwidth monitoring tools DU Meter [37] and NetLimiter [65] to verify that there is nothing else using our network connection during the testing.

7.2 Pitfalls with Test Setup

When performing a live traffic test over the Internet it is important to be aware of the many factors that might affect the obtained results. We will in this section list up these factors and explain why they can affect the result.

- **Cross traffic** – The Internet is a live unpredictable environment with many competing users. This might lead to sudden capacity competition at certain exchange points.
- **Inaccurate rate limiting** – Accurate rate limiting is very hard to achieve as explained in the previous section. If the applied rate limit technique is inaccurate this could affect the results and might lead to a wrong conclusion.
- **Queue discipline** – The applied queue discipline might affect the results. We have chosen to use RED which makes it easier for TCP flows to adapt to the available rate.
- **Layer used for testing** – Some of the tested tools perform the testing at the application layer (as HTTP) in the protocol stack (e.g. Speedtest), while others (e.g. Iperf and MySpeed) perform the testing with pure TCP sockets. If this is not accounted for it will be hard to compare the obtained results.

- **Test Computer** – If the test computer uses the network connection during the testing (e.g. Windows checks for updates) this would affect the measured bandwidth. We have used passive monitoring to ensure that this does not happen.
- **Broadband test server location** – If the server we measure our bandwidth against is located far away, or on a path with limited bandwidth, then it is hard to measure the bandwidth accurately. We experienced this problem with the MySpeed server located in Ålesund.
- **Poor cables** – Poor network cables with high loss probability will most likely affect the results.
- **Unreliable switches** – At our office we use a simple Unex switch to share our Internet connection. We found this switch a bit unreliable, providing a varying bandwidth. The solution was to not use this switch while testing.

7.3 Results

To evaluate the different on-line test tools we connected our Test Computer to the Internet and executed each broadband test-tools *ten* times at the different rates. By running each tool multiple times at each rate we ensure that transient capacity changes on the Internet should not influence the final result too much. We have also eliminated results that obviously deviate from the expected result.

The tested rates are 400 kbit/s, 800 kbit/s, 1600 kbit/s, 3200 kbit/s, 6400 kbit/s and 12800 kbit/s in both upload and download direction. We could not use higher upload rates because we discovered some Windows related upload issues with TCP on NTNU at rates higher than 13000 kbit/s. These issues are explained in more detail in appendix D. The implemented rate limitation is performed at the network level, while all tools try to measure the actual payload throughput (goodput) of TCP or HTTP. TCP communication includes a lot of overhead (TCP and lower layer headers). To find the actual TCP goodput we have established two Iperf-test servers. One server is located locally (Iperf-loc), on the same switch as the Bandwidth Limit Server, while the other, Iperf-nix, is located near NIX as illustrated in figure 7.1. The Iperf-nix server gives a natural location of a bandwidth test-tool, at/or nearby a neutral exchange point on the Internet. We use this server to calibrate our results and all other tools are compared with Iperf-nix. The Iperf-loc is used to verify the results reported by Iperf-nix and ensures that nothing else disturbs our results.

We will further in this section present some of our findings from our study of the bandwidth test tools. The complete test plan, with results and all relevant graphs, is included in the digital appendix B.1.

7.3.1 Download Results

Generally all tools (except Speedometeret JavaScript described below) performed well for all rates up to 6400 kbit/s. Figure 7.4 shows how much each tool

deviates on average from Iperf-nix in kbit/s. The figure illustrates that most tools underestimates the bandwidth, except Bredbandskollen and Speedtest, which seems to overestimate the download rate[2]. The figure also illustrates that Speedometeret generally measures almost the correct value with a little underestimation. We think this is because Speedometert uses HTTP to perform the bandwidth test without accounting for the overhead, and thus actually measure the HTTP payload throughput. MySpeed (in Ålesund) underestimate the highest rate. We suspect that this is a result of the server location which is far away from the Norwegian backbone, and we have to pass links of 2.5 Gbit/s to reach the MySpeed server in Ålesund.

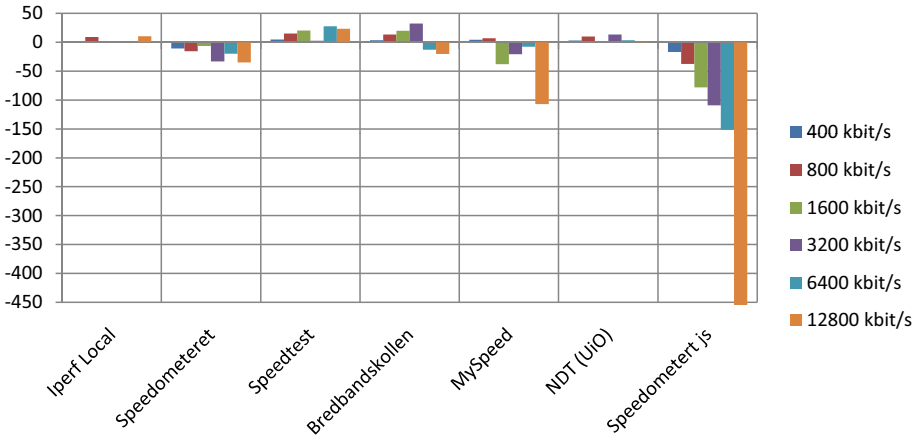


Figure 7.4: Download rate deviation from Iperf-nix in kbit/s.

In figure 7.5 we have presented the percentage overestimation performed by Speedtest and Bredbandskollen at the various download rates. Why these tools overestimate the bandwidth is hard to say when we don't have access to the source code. But we suspects that this is a result of the functionality of how these tools estimates the throughput, explained in section 6.1.2. The throughput is measured up to 30 times per second. The final download rate is determined by looking at ordered samples through a sliding window to eliminate anomalies. This will result in slightly higher results as it does not include the transient slow-start in the final result. Other anomalies, such as small bandwidth drops are also eliminated from the final results. The figure also shows that Bredbandskollen underestimates (instead of overestimating) the two highest rates. We think this is because Bredbandskollen is located in Sweeden, where the data has to travel longer and also has to compete with inter-country traffic.

Figure 7.6 shows the result for all tools with a network layer download rate limitation at 3200 kbit/s. From the figure we can see that Iperf-nix and Iperf-local has almost identical results, and the difference between the maximum and

[2]To verify that Speedtest and Bredbandskollen actually gets a higher bandwidth we performed bandwidth measurements against Iperf-nix, Speedtest and Bredbandskollen at our private ADSL-connection (NextGenTel). We found the same "overestimation" in these measurements. This verifies that our Bandwidth Limit Server works like it should.

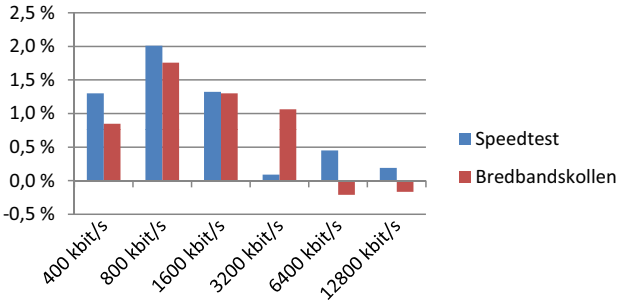


Figure 7.5: Speedtest and Bredbandskollen percentage deviation from Iperf-nix.

minimum measured value is low. We can also see how inaccurate Speedometeret JavaScript version is, with a huge difference in highest and lowest measured value, where the highest measured bandwidth is lower than the lowest measured Iperf-nix value. The graph also illustrates that the variation in the measured rate is larger for the tools located at a long distance from NIX.

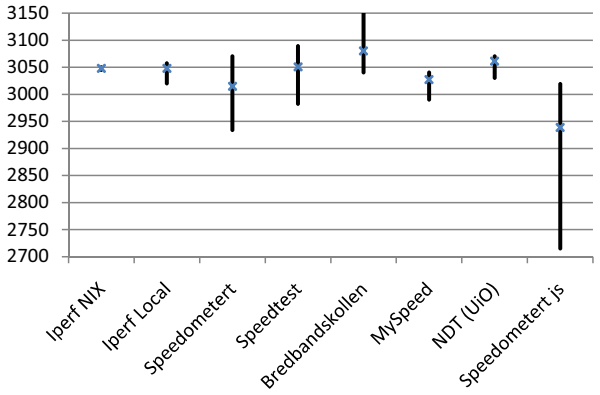


Figure 7.6: Minimum, maximum and average download rates at 3200 kbit/s.

Speedometeret JavaScript

In section 6.1.1 we introduced the JavaScript version of Speedometeret. We claimed that JavaScript is not a good idea for bandwidth estimation, mainly because of JavaScript timing issues. To verify our statement we included this test-tool in our evaluation, and the results is shown in figure 7.7. The chart shows that the deviation in kbit/s measured against Iperf-nix increases as the download rate increases. The chart speaks for itself and we can see that this test underestimates the bandwidth for all rates. For a 12,8 Mbit/s link the test underestimates the bandwidth with almost 1000 kbit/s.

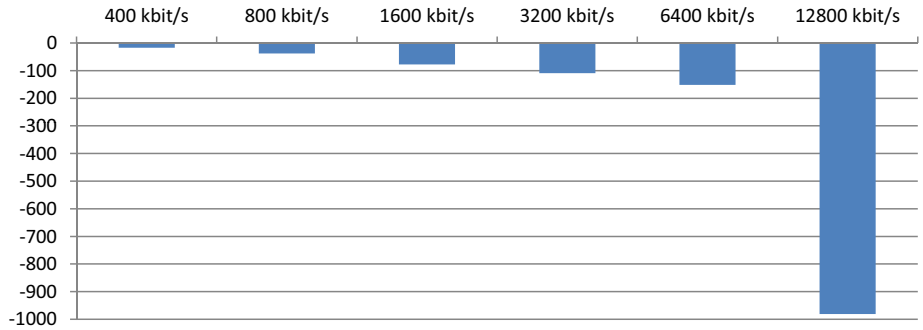


Figure 7.7: Speedometeret JavaScript download rate deviation from Iperf-nix in kbit/s.

High Speed Download Testing

As more and more subscribers get higher and higher download rates, it is becoming more important that the broadband evaluation tools can handle these increased rates. We are in this test interested in checking if the different tools are able to handle high download rates, and we tested them for 20 Mbit/s, 40 Mbit/s and 60 Mbit/s. We compared all results against the results acquired when we used Iperf-nix with multiple TCP connections. Iperf-nix and Iperf-loc has been set up to measure achievable throughput, explained in section 5.3.9. This ensures that we get a higher share of the bandwidth. We have also included Iperf-nix-BTC which uses one connection and thus measures the BTC as explained in section 5.3.8. MySpeed is excluded from this test as this tool cannot handle our high rates. A simple 100 Mbit/s test was performed and MySpeed reported 22 Mbit/s. We think this is a result of MySpeed's location as we explained earlier. Because of a lot of competing traffic at University of Oslo (UiO) during the test we had to switch to a NDT server located at NTNU. NDT thus measures the local BTC.

The average measured deviation from Iperf-nix is illustrated in figure 7.8. We can see that all tools can handle the high rates fairly well except Speedometeret which underestimates the 60 Mbit/s connection with 12 %. It must also be taken into consideration that Speedometeret only uses one connection and thus measure the BTC, explained in section 5.3.8.

Iperf-nix-BTC which only uses one TCP connection is not able to utilize the whole bandwidth. We can also see from the figure that the BTC transfer is more exposed to competing traffic at higher rates. Even though NDT also measures the BTC traffic, it only has to compete with traffic at NTNU. We suspect that higher rates are needed to get the same effect.

Bredbandskollen underestimates the bandwidth with 0,86 % at the highest rate. We think this is because this service is located in Sweeden and must compete with inter-country traffic, as we explained earlier.

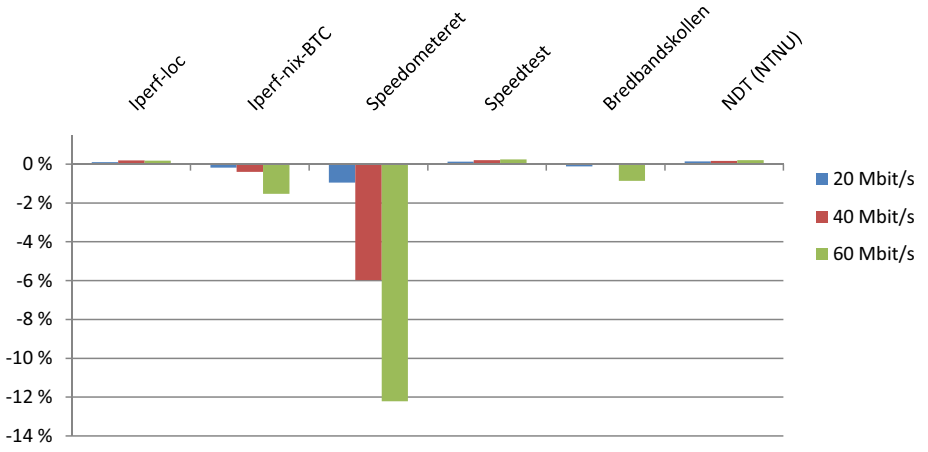


Figure 7.8: Percentage deviation from Iperf-nix download rate at 20 Mbit/s, 40 Mbit/s and 60 Mbit/s.

7.3.2 Upload results

The upload test has been calibrated by sending data from our Test Computer to Iperf-nix. We have chosen to measure the upload rate for Iperf-nix at the receiver side (receiver download rate), which implies a measuring interval lasting from the first bit of data is received until the last bit of data is received. The results are presented in figure 7.9 which shows the upload deviation from Iperf-nix for each tool. When we increase the rates to 6400 kbit/s and above, we see that Speedometeret, Speedtest and Bredbandskollen really start to underestimate the upload rate. We think this is because these tools are designed for asymmetrical bandwidths where they assume that the upload rate is much smaller than the download rate and too little data is sent. Speedometert completes its upload test at the highest rate in just 0,6 seconds which is an indication that a too small amount of data is used. Speedometeret, Bredbandskollen and Speedtest are the only tools tested that uses HTTP POST messages to upload data. In figure 7.10 we have presented the results of the upload test at 3200 kbit/s and 12800 kbit/s. This figure illustrates that the underestimation grows for higher upload rates. The figure also shows that the bandwidth varies more at higher rates, as it is more exposed to cross traffic.

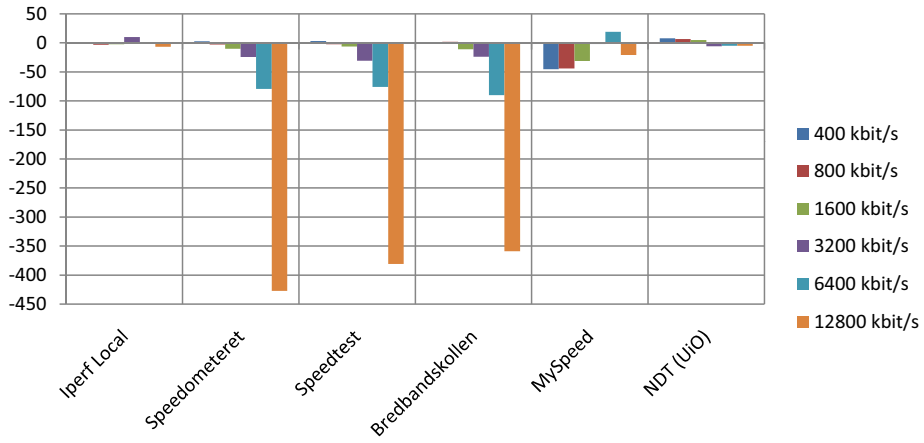


Figure 7.9: Upload rate deviation from Iperf-nix in kbit/s.

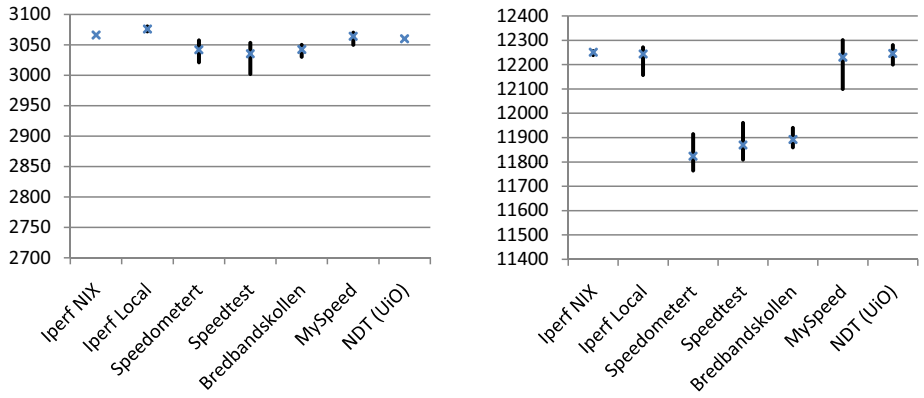


Figure 7.10: Minimum, maximum and average upload rates at 3200 kbit/s (left) and 12800 kbit/s (right).

Flash Upload Issues

During our testing we discovered that the Mozilla Flash plug-in has some issues with high speed upload rates. The problem seems to be related to the Windows based Flash plug-in[3] used by Opera, Firefox and Safari. Figure 7.11 shows the obtained upload rates with different browsers and operating systems when our Test Computer was connected directly to a 100 Mbit/s connection and executed the Bredbandskollen test tool. From the figure it is easy to see that the upload rates are significantly lower for Firefox and Opera on the Windows platform, than for Internet Explorer in Windows and Firefox in Linux. We performed multiple tests at different hours to ensure that the results are correct. Opera and Firefox use the same Flash plug-in in Windows, but Internet Explorer in Windows and Firefox in Linux use different plug-ins. From this we can conclude that there seems to be something wrong with the Flash plug-in and this error manifests itself when measuring high upload rates with HTTP POST messages[4]. We have not been able to find any published information regarding this issue. A parallel study submitted by Artur Janc to the Faculty of Worcester Polytechnic Institute in January 2009 looked in to different ways to perform evaluation network characteristics within the web browser. Janc reports the same upload issues with the Mozilla Flash plug-in in his master thesis [34]. This limitation with the Flash plug-in must be taken in to consideration when designing a broadband test tool that should support high speed clients.

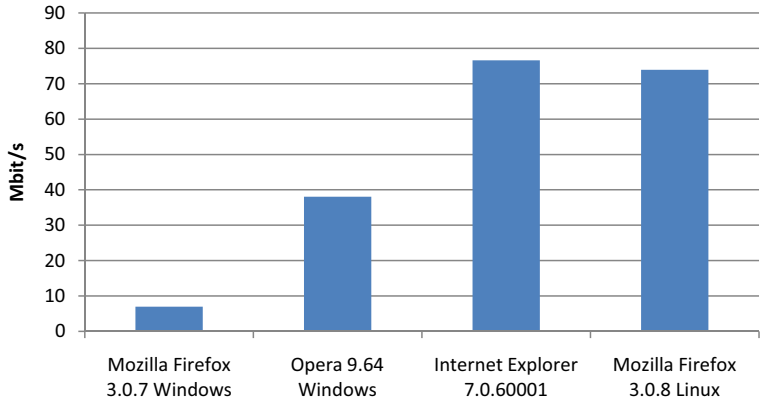


Figure 7.11: Obtained upload rate with Flash in various browsers.

7.3.3 Glasnost

In this test we wanted to check if Glasnost, introduced in section 6.1.5 detects rate limiting at well known BitTorrent ports (6881, 6882, 6883). We tested the following configurations:

[3]We used Shockwave Flash 10.0 r22 (Mozilla Version), but had the same problems under version 9.0.

[4]We have also performed testing with a high speed fiber connection from Lyse Tele showing the same difference in upload speed obtained by Firefox and Internet Explorer.

1. **65% Reduction** – Regular traffic limited at 2300 kbit/s. BitTorrent ports limited at 800 kbit/s.
2. **56% Reduction** – Regular traffic limited at 2300 kbit/s. BitTorrent ports limited at 1000 kbit/s.
3. **50% Reduction** – Regular traffic limited at 2000 kbit/s. BitTorrent ports limited at 1000 kbit/s.
4. **33% Reduction** – Regular traffic limited at 1500 kbit/s. BitTorrent ports limited at 1000 kbit/s.

The results is included in appendix C. What we found is that Glasnost report potential bandwidth discrimination for the first two cases where the BitTorrent port traffic is reduced with more than 50%. No discrimination is reported when the traffic is reduced with 50% or less.

7.3.4 Evaluation of NBTT

The planned NBTT will be based on the measurement engine developed by Ookla, also used by both Bredbandskollen and Speedtest. Because of this we can expect that NBTT should perform equivalent to these services. Because NBTT is planned to be located at optimal locations in the Norwegian Internet infrastructure we should also assume that the results will be slightly more accurate and the service will provide less variations in the obtained results.

Since NBTT will be developed in Flash, we should also be aware that NBTT will have the same high speed upload issues demonstrated for Bredbandskollen shown in figure 7.11.

7.3.5 Concluding Remarks

We have in this chapter tested different tools end-users can use to evaluate their broadband access. Generally we have seen that most of the tools are able to measure the download rate rather accurately up to 12,8 Mbit/s. When we tested the tools at higher download rates (20 Mbit/s, 40 Mbit/s and 60 Mbit/s) we found that Speedtest and Bredbandskollen handled these rates well. Speedometeret on the other hand failed to measure these rates accurately. We think this can be caused by a combination of the server location and the fact that this tool measures the BTC while Speedtest and Bredbandskollen measures the achievable throughput.

The evaluation also showed that server location is important. MySpeed server located in Ålesund failed to measure bandwidths above 20 Mbit/s.

The evaluation of Speedometeret JavaScript showed that JavaScript is not a good choice for measuring bandwidth. This is mainly because of timing issues in JavaScript explained in section 6.1.7.

All tools utilizing HTTP POST to measure the upload bandwidth failed to do this accurately, especially for rates above 3.2 Mbit/s. This can be a result of these

tools assuming asynchronous bandwidths, and thus uses too little data to measure the upload rate. Speedometeret completed the upload test at 12,8 Mbit/s in just 0,6 seconds.

We also discovered upload issues with the Windows version of the Flash plug-in for Mozilla for high-speed upload rates. As the source code for Flash is not publicly available we were not able to figure out what causes these problems, but we think it is an important issue to take in to consideration. Janc reports the same upload issues with the Mozilla Flash plug-in in his studies [34].

Glasnost is able to detect possible traffic throttling if BitTorrent traffic gets less than 50 % of the rate regular traffic gets. This means that an ISP can throttle BitTorrent traffic up 50 % without this tool reporting anything suspicious. Traffic throttling is hard to reveal because of the many possible reasons for why the bandwidth varies over time. Glasnost performs many measurements in sequence and different values do not necessarily mean that traffic throttling has occurred.

NBTT is planned to use the same test engine used by Bredbandskollen and Speedtest, and we can thus expect that the obtained results for this service will be close to the results obtained by these tools.

Overall we think most of the available tools perform well for common bandwidth rates today. For higher rates, especially upload rates, there is still some room for improvements.

Chapter 8

Measurement Scenarios for Broadband Testing

In chapter 5 we introduced different network performance quality parameters and we argued that different users have different needs. For some services high bandwidth is of great importance, while others are more sensitive about delay, jitter or loss rate. A broadband test tool should evaluate the quality of a broadband connection in context of its usage. In this section we will define different typical scenarios, or profiles, which try to cover different end-users' needs. An end-user can of course fulfill multiple profiles (a family sharing an Internet connection). By defining these profiles it will be possible to run specific tests to verify that a certain access connection fulfills the service needs of the selected end-user profile. We remind the reader of the scope of this thesis from section 1.3 and the focus on private broadband subscriptions. When we describe a business user we refer to the private broadband subscriber who uses his connection for business purposes.

8.1 End-user Profiles

Today there exist a lot of different types of end-users, ranging from technological pioneers to the slow adapting traditionalists [52]. If we also add the life-stage dimension ranging from youth to seniors it is easy to see that defining end-user roles can be a quite complex task [52]. Also within each life-stage there is a high risk of anomalies not fitting their role definition (e.g. grandmother playing games on-line). Because of all this complexity we think end-user role definition is a study on its own which must include a comprehensive user survey and is therefore out of scope for this thesis.

We will in this section define five high-level end-user profiles including: *private basic*, *private gaming*, *private multimedia*, *business basic* and *business multimedia*. These profiles are presented in figure 8.1. From the figure we can see the distinction between private and business profiles. We have made this distinction because business users typically make use of different services and therefore have different requirements to the access connection. We also see that there is a hierarchy where we expect that *private gaming* users also utilize the services of the *private basic* profile. The same applies for the multimedia profiles for both private and business.

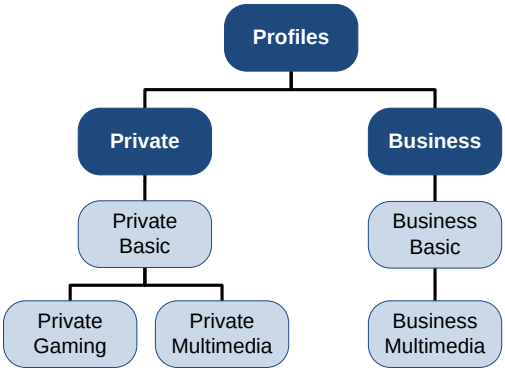


Figure 8.1: Classification of high-level end-user profiles.

By defining profiles it will be easier to verify whether an Internet subscription fulfills the correct needs. The idea is that an end-user wanting to evaluate the broadband connection can run a basic test or select an appropriate profile. The profile classifications should be as broad as possible to make it easy to select the correct profile and not confuse the end-user. Figure 8.2 shows a typical five-step scenario where the user accesses the on-line bandwidth test tool and selects what profile to use in the evaluation of the access connection.

The scenario presented in figure 8.2 follows these steps:

1. The user accesses the on-line bandwidth test tool.
2. The on-line test tool presents the available profiles to the user. A basic profile is selected by default.

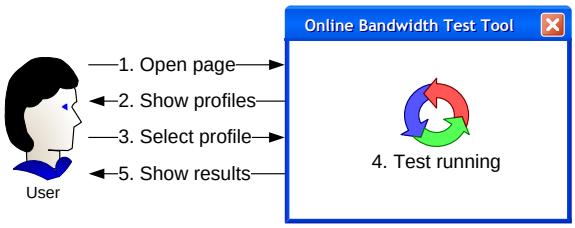


Figure 8.2: Five-step bandwidth test scenario where the user selects matching profile.

3. The user optionally selects a predefined matching profile, or define a custom profile. A default profile will be used if no profile is selected.

4. The on-line test tool start evaluating the end-user access-connection. The evaluation is performed on the basis of the specified profile.

5. When the on-line test tool completes it presents the final result to the end-user which includes a assessment of how the connection satisfy the selected profile's requirements.

We will in the following briefly introduce each profile and list typical services which fit within the respective profile. Some users have special needs, or overlap between multiple profiles. This make it hard to define general profiles covering all users. Because of this we have included a *custom* profile which allows the user to specify his own services. This makes it possible for advanced users to customize their own profiles, and evaluate whether their connection performs acceptable for those services. The services will be further discussed in section 8.2.

8.1.1 Profile: Private Basic

This profile is intended to fit most of the private end-users. As this is a general profile it should contain all services we could expect a *regular* private subscriber to be interested in. All the services covered by this profile are the main driver for why end-users want to access the Internet today. Who would be interested in connecting to the Internet if they are unable to read web pages or send emails? This profile should at least verify the performance of the following services:

- **Web Surfing**
- **E-mail**
- **Instant Messaging**
- **File Transfer**
- **Peer-to-Peer**

In the future, when new services are introduced and the majority of the end-users have adopted these, we must also add those services.

8.1.2 Profile: Private Gaming

This profile tries to capture the needs of an end-user interested in playing games on-line. We should expect that a user in this profile also is interested in all the basic services. A gamer will need to access these common services for the same reasons all private users need. Who would be interested in an Internet subscription only able to play games? Gamers has a special need to communicate with other people through various forums and instant messaging services, chat during the game play and download game updates. *Private gaming* is thus a sub-profile of *private basic*. It is also common to have voice conversations with other players while playing. The *private gaming* profile should verify the performance of the services in the *private basic* profile as well as:

- **Gaming**
- **VoIP**

8.1.3 Profile: Private Multimedia

Today a lot of multimedia services are provided over the Internet. Some of these services are new inventions like Youtube and Joost, while others are old multimedia services finding its way to the Internet, like Television over Internet (IPTV) or Telephony over Internet (VoIP). This profile should try to cover the commonly available multimedia services. *Private multimedia* is a sub-profile of *private basic*. This profile should thus verify the performance of the services found in the *private basic* profile as well as:

- **Streaming**
- **VoIP**
- **Video-over-IP**
- **IPTV**

8.1.4 Profile: Business Basic

A business user differs from a private user in that the connection is used in a business-context and the required services differ slightly. This profile should at least evaluate the following services:

- **Web Surfing**
- **E-mail**
- **File Transfer**
- **Remote Desktop**

8.1.5 Profile: Business Multimedia

The *Business multimedia* profile covers the business users with multimedia requirements. For business users this usually includes regular telephony over Internet (VoIP) and/or video-telephony over Internet as well as all the services in the *basic business* profile. For this profile it is thus necessary to evaluate the services from the basic profile as well as:

- **VoIP**
- **Video-over-IP**

8.1.6 Profile: Custom

For some end-users none of the specified profiles fulfills their needs entirely. They might have overlapping needs, or use special type of services, such as Secure Shell, which does not belong to any of the specified profiles. These users should have the possibility to define their own custom profile by selecting from all the services. Appropriate network performance evaluation should be based on the selected services.

8.2 Services

In this section we will describe services commonly used by end-user subscribers. This includes everything from regular services like web surfing and e-mail to business services like remote desktop and file transfer. Lastly, we will introduce game streaming, which is a new highly demanding concept. We will describe what each service is used for, including concrete examples, and discuss performance parameters which are important for this service. Some of the services in this section does not belong to any of the predefined profiles, but should be available as a selectable option in the *custom profile*.

8.2.1 Performance Parameters

For each service we will list what performance parameter the connection must satisfy to give an acceptable user experience. We have chosen to use the following performance quality parameters:

- **High Bandwidth** – The service has no upper or lower limit of the variation in bandwidth, as long as it on average is high.
- **Predictable Bandwidth** – The service will typically use a predefined data rate and be degraded if it is offered a lower data rate. The bandwidth variation should be low.
- **Low delay** – The delay should be low in order to not degrade the quality of the service.

- **Moderate delay** – The service can tolerate slightly higher delay (typical < 2 seconds) than those services requiring a low delay (typical < 0.2 seconds).
- **Low jitter** – The variation in delay should be low in order to not degrade the quality of the service.
- **Low loss** – The loss should be low in order to not degrade the quality of the service.

The requirement for each performance parameter will depend on specific application providing the various services.

8.2.2 Web Surfing

The WWW is a huge framework for accessing linked documents spread all over the Internet [68]. These documents may contain text, pictures, video and other multimedia content. Web surfing refers to the phenomena of going from one site to another, "surfing the web".

Today the web is extremely popular and many people need to access various web pages to perform everyday tasks, and is usually just as important in a business context as in a private context. Examples includes: newspapers, information searching (Google and Yahoo), social sites (e.g. Facebook and twitter) and Internet banking.

Performance Parameters
For Web surfing it is important to have a decent download bandwidth rate to get a good *surfing* experience. This is important because web-page content is downloaded to the end-users web browser, where it is read when the download is completed.

Example: We checked the size of Norwegian TV2 webpage (tv2.no) to illustrate why high bandwidth is needed for Web surfing. On April 22, 2009 their front page had a total content-size of 3 MB. Just to download all this content on a 2 Mbit/s connection we would require about 13 seconds.

When surfing the web the user navigate from one cite to another by following hyperlinks. To get a good surfing experience we thus require response on our actions within acceptable time periods. We would probably not be satisfied if we have to wait a few seconds before we get response. Seen from a technical point of view, web surfing requires a lot of signaling. Typically it requires connection set up (3 times RTT) and then it needs to download an index file which contains information on what other files it needs to download, which optionally also includes information on additional files. To get a feeling of a responsive web we thus require a *moderate delay*.

To sum up, Web surfing requires:

- **High Bandwidth (download)**
- **Moderate Delay**

8.2.3 E-mail

Today electronic mail, or e-mail, is a very common form of communication in both a private and a business context. In many situations e-mail has already replaced ordinary "snailmail". E-mails can contain attachments such as documents, presentations, images, audio and video.

Performance Parameters
E-mail is a store-and-forward type of communication. The user write an e-mail, transmits the e-mail to a server which is responsible for transmitting the message to the receiver. The most common way to write, send, receive and read e-mails today is through specialized e-mail applications. As these applications send and regularly checks for incoming messages in the background, it is only required to have a decent download and upload rate.

To sum up, e-mail requires:

- **High Bandwidth**

8.2.4 Instant Messaging

Keeping in touch with friends and colleagues is very important. Today more and more people are using instant messaging services for this purpose. Instant messaging is a system for sending short instant text messages between users. Examples includes Windows Live Messenger, Google Talk, and Internet Relay Chat (IRC).

Performance Parameters
It is quite common to have near-synchronous dialogues over an instant messaging system. For this reason the delay should be within acceptable levels to provide a decent service, we require a *moderate delay*. Because instant messaging only transmits short text messages there is no strict requirement to the bandwidth (but of course one must have some form of connectivity). Instant messaging clients typically provides the possibility for the user to transmit files between each other, but this should be regarded as a file transfer, see section 8.2.5.

To sum up, instant messaging requires:

- **Moderate Delay**

8.2.5 File Transfer

File transfer is a generic term for the act of transmitting files over a computer network or the Internet [76]. There exists many different ways and protocols to perform a file transfer. A file transfer can be from one user to another or from a server to a user. Most end-users needs file transfer because they want to download new programs, upgrades to existing programs, or to share photos (e.g. upload images to Flickr).

Business users might use network sharing techniques to access network resources, and we will in this thesis categorize this as a file transfer.

Performance Parameters
File transfer is usually used to move files of a certain size. Because the transfer itself last for a certain amount of time the signaling needed for set-up can be neglected. The major driver for a good file transfer experience is the bandwidth rate.

To sum up, file transfer requires:

- **High Bandwidth**

8.2.6 Peer-to-Peer

Peer-to-Peer (P2P) is a distributed overlay network where the end-hosts, "the peers", are connected to each other over the Internet. This enables file sharing without a centralized server. Each "peer" in a P2P becomes a client and a server [1].

Performance Parameters
P2P networks can in principles be used for all types of distributed systems such as file transfer, media streaming and Internet telephony, but is usually used to transfer files between end-hosts.

To sum up, P2P requires:

- **High Bandwidth**

8.2.7 Gaming

On-line gaming is a technology rather than a genre; a mechanism for connecting players together rather than a particular pattern of gameplay [63]. On-line gaming is to play games over the Internet. Examples include Age Of Conan, Quake 3, Counter Strike and World of Warcraft. There exists many types of gaming and we will primary consider real-time sensitive games.

Performance Parameters
On-line gaming typically tend to use small highly periodic UDP packets. They are highly periodic because of predictable state updates between clients and servers. On-line gaming has low bandwidth requirements, but they require the bandwidth to be larger than a certain threshold. Low latency and low jitter is required because of the real-time game-logic. Packet loss quickly degrades the user experience to an unpredictable level [20].

To sum up, on-line gaming requires:

- **Predictable Bandwidth**

- **Low Delay**
- **Low Jitter**
- **Low Loss**

8.2.8 Streaming

Multimedia streaming is a technique used for transferring information so that it can be processed as a steady and continuous stream [52]. This technique makes it possible for the receiver to start consuming the content before the entire file is received. Typical application of streaming includes:

- **Audio-on-Demand** – Streaming of music (e.g. Internet Radio).
- **Video-on-Demand** – Streaming of video (e.g. Youtube).

Performance Parameters
Usually media streaming delivers the content in a constant rate dependent on the streamed media. This requires the bandwidth to not be lower than the required rate. If the content is delivered using UDP, packet loss will degrade the user experience of the streamed media. Generally delay variation (jitter) should be low, but some jitter can be accounted for by using buffers at the receiver.

To sum up, streaming requires:

- **Predictable Bandwidth**
- **Low Jitter**
- **Low Loss**

8.2.9 VoIP

VoIP is a general term for transporting voice communication over the Internet and is often referred to as Internet telephony. Some of the VoIP systems interface the The Public Switched Telephone Network (PSTN), while others do not.

Performance Parameters
VoIP has small bandwidth requirements in terms of size but requires a predictable bandwidth able to handle the steady stream of packets. Delay and jitter is important network characteristics for VoIP and Video-over-IP because of the real-time characteristics [84]. Loss will degrade the user experience.

To sum up, VoIP requires:

- **Predictable Bandwidth**
- **Low Delay**
- **Low Jitter**
- **Low Loss**

8.2.10 Video-over-IP

Video-over-IP is a general term for video-telephony over the Internet. This enables the participants to communicate using live pictures and voice.

Performance Parameters
Video-over-IP will, as VoIP, produce a constant stream of data which requires a predictable bandwidth (this rate will of course be higher than for VoIP). Video-over-IP is basically the same service as VoIP, but with added video functionality. Therefore the important performance parameters are the same as for VoIP, see section 8.2.9.

8.2.11 IPTV

IPTV is a general term for Television broadcasted over Internet and other packet based networks. IPTV can be can be compared with streaming where the difference is that the content is live while streaming typically offers on-demand capabilities.

Performance Parameters
The important performance parameters is the same as for streaming see section 8.2.8.

8.2.12 Remote Desktop

Remote Desktop refers to a software or an OS feature allowing graphical applications to be run remotely on a server, while being displayed locally [80].

Performance Parameters
Remote desktop generates a constant stream of all the updates from the server to client. The user can usually define how large this stream can be (in Windows from 28.8 kbit/s to 10 Mbit/s). To make the remote desktop feel responsive it is important to have a low delay. This is important because when the user performs an action, this action has to travel to the server, the server performs the action, and the result is transported back to the client.

To sum up, remote desktop requires:

- **Predictable Bandwidth**
- **Low Delay**

8.2.13 Secure Shell / Telnet

Secure Shell is a network protocol that allows data to be exchanged using a secure channel between two networked devices [82]. It was designed to replace the unsafe

and insecure Telnet protocol. Secure Shell is usually used to access shell accounts on Linux and Unix based systems.

Performance Parameters
When using Secure Shell to access shell accounts the user operates in a console environment without graphical support. This basically sets low requirement to the bandwidth. A high delay on the other hand can degrade user experience. This is because when the user types a command on his keyboard, the command has to travel to the server, and back again, before it is presented on the screen.

Secure Shell can also be used to transfer files, but this should be regarded as a file transfer, see section 8.2.5.

To sum up, Secure Shell requires:

- **Low Delay**

8.2.14 Game Streaming

Game streaming is new gaming concept introduced at Game Developer's Conference in San Francisco this year (2009) by OnLive [53]. Figure 8.3 shows how the OnLive works. The idea is to allow clients to stream games from the OnLive servers. This will allow game playing with low-end user equipment at the client side and still provide high-quality images. The user does not need to upgrade hardware or software. Having the game discs are unnecessary as the game is streamed directly to the end-user.

Figure 8.3: How OnLive Works [53].

Performance Parameters
Game streaming can be looked at as a combination of streaming, remote desktop and gaming. OnLive claims they can provide HDTV quality at a 5 Mbit/s connection [53]. In addition to high bandwidth, it will require a predictable bandwidth, as the bandwidth for the service should not be lower than a specified threshold.

To provide a good user-experience, ordinary gaming requires ultra-low delay from the instant the user performs a controller action until the action is reflected on

the screen. In the OnLive case, the action has to be transported to the OnLive server, which calculates the effect of the action and renders the screen image, before it is transported back to the user. This puts extreme requirements to the two-way network delay.

A Service like OnLive should probably use UDP to transport the data downstream to the user because of the real-time characteristics. Loss is a possibility when using UDP, and this would degrade the user experience.

To sum up, game streaming requires:

- **High Bandwidth**
- **Predictable Bandwidth**
- **Low Delay**
- **Low Jitter**
- **Low Loss**

8.2.15 Summary Services

Figure 8.4 illustrates the previously introduced services. From the figure we can see which basic network performance parameters we need to evaluate for each service. The different services, or even the different applications providing a service, will have different requirements for the performance parameters. E.g. some games will have higher requirements to a low delay than others. We think it is also important to use the correct transport protocol when evaluating the quality for the different services.

8.3 Profiles and Performance Parameters

If we combine the information about which services a profile contain with the information about the important performance parameter for each service, we get information about what performance parameters we need to evaluate for each profile. This information is presented in table 8.1. From this table we can see which performance parameters we must evaluate for each profile and what transport protocols that should be used in the evaluation.

8.3.1 What Profiles Can NBTT Evaluate?

In section 2.2 we introduced the NBTT service. This service will use HTTP throughput to evaluate the subscribers download and upload rate as well as the RTT (delay). HTTP is an application protocol which uses TCP for transport. Table 8.1 shows which network parameters we must evaluate for each profile. The table also show which transport protocol we should use to evaluate the different performance parameters. Because NBTT will evaluate download and upload bandwidth, as well as RTT and use TCP as transport protocol in the evaluation,

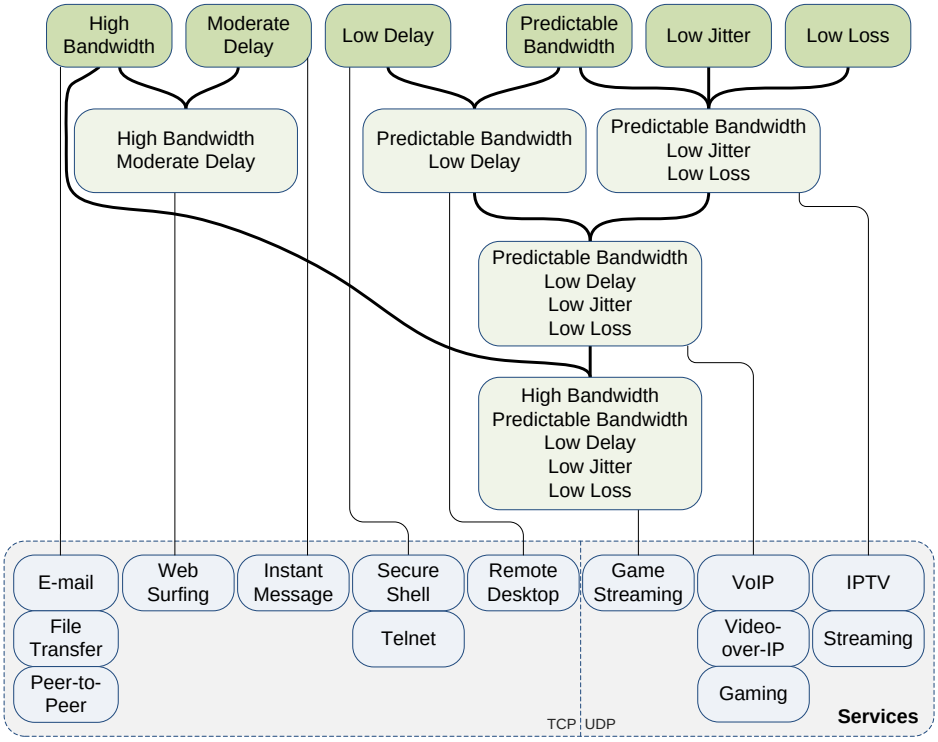


Figure 8.4: Performance parameters we need to evaluate for each service.

	Basic	Gaming	Multimedia	Basic	Multimedia
Bandwidth (down)	T	T	T	T	T
Bandwidth (up)	T	T	T	T	T
Delay	T	TU	TU	T	TU
Predictable Bandwidth		U	U	T	U
Loss		U	U		U
Jitter		U	U		U
	Private			**Business**	

T = Evaluate using TCP
U = Evaluate using UDP
TU = Evaluate using TCP and UDP

Table 8.1: Profiles and performance parameters.

the service will only be able to evaluate the performance for the *private basic* profile. We think this is a good starting point for the NBTT service since this profile is aimed to fit the majority of the end-users. In the future the tool can be expanded to cover all profiles. This is further discussed in section 11.1.4.

In section 11.1.4 we also suggest how profile measurements can be used to reveal breaches to NPT's network neutrality principles.

Chapter 9

Testing of Measurement Scenarios

In this chapter we will investigate three different measurement scenarios. We want to illustrate that the planned NBTT is not able to cover all scenarios in different environments. Concrete scenarios, with explicit network performance parameter requirements are used to show that the values measured by NBTT are not enough to predict the experienced quality for certain services. An access connection can be rated as good even if it only on average delivers the desired quality. In this chapter we show that average measured values cannot be used to verify that the quality is good enough to support all services.

The tested scenarios include:

- Gaming
- VoIP
- Predictable bandwidth

9.1 Test Set-up

Each scenario will be tested on three different access networks. We have chosen to use multiple access technologies to get a broader evaluation of the planned service. NBTT is planned to provide the same functionality as Bredbandskollen (section 6.1.2). We will thus compare our analysis against values measured with Bredbandskollen. Please notice that this is not going to be a performance evaluation of the different access networks, but rather an investigation of whether the planned functionality of NBTT is able to cover different important scenarios.

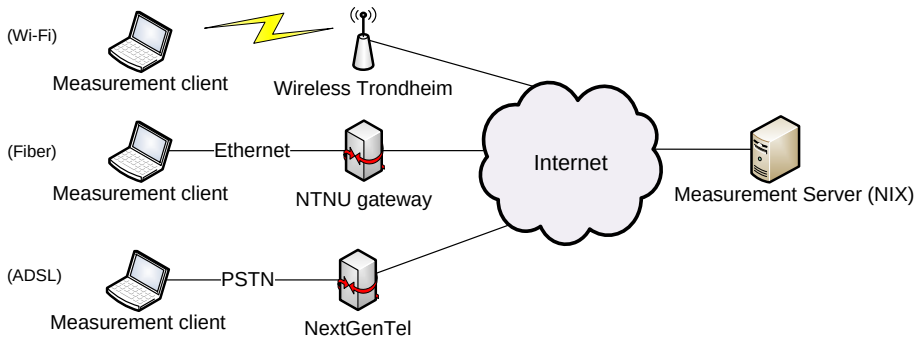


Figure 9.1: The access networks used for scenario testing.

The three access networks used in our testing are shown in figure 9.1 and includes the following technologies:

- **Fiber** – NTNU campus network with fiber all the way from NIX to NTNU campus. The last hundred meters are regular Ethernet.
- **Wi-Fi** – Wireless Trondheim [69][1]. We have on purpose chosen a location far away from the nearest base-station for this connection. This will result in a poorer and a more unpredictable connection, which is exactly our intention. We want to see how such an environment may affect the result.
- **ADSL** – Private home broadband connection delivered by NextGenTel.

We have executed the Bredbandskollen tool for each connection and the measured values are presented in table 9.1. These values will be used in our presentation of each of the scenarios. Keep in mind that there will be some minor differences in values measured by Bredbandskollen (in Stockholm, Sweden) and the values measured against NIX (in Oslo, Norway).

Paramteter	**Wi-Fi**	**Fiber**	**ADSL**
Download rate (Average)	0.50 Mbit/s	94.28 Mbit/s	5.38 Mbit/s
Upload rate (Average)	0.50 Mbit/s	19.97 Mbit/s	0.73 Mbit/s
Delay (measured with HTTP)	16 ms	15 ms	35 ms

Table 9.1: Measured values with Bredbandskollen (Stockholm, Sweden).

[1] Wireless Trondheim is a huge Wi-Fi network covering Trondheim center.

Control Environment
To check the actual values for all the important performance parameters we needed to set up a control environment. This includes sending traffic from the measurement server to the measurement client. Traffic is also sent in the opposite direction for both the gaming and the VoIP scenario. The data sent (packet length and rate) and the transport protocol used is dependent on the protocol used by the scenario we want to evaluate. Data is generated and sent with the Iperf application, introduced in section 6.2.4. We will also use Iperf to measure the jitter and loss rate. Hpcbench (section 6.2.3) is used to measure the UDP delay.

9.2 Scenario Testing

For each scenario we will use a specific application with explicit requirements which is typical for the profile. Of course, as mentioned in section 8.2.15, different applications providing a service will have different network performance requirements in order to provide a good experience for the end user.

9.2.1 Scenario: Gaming

In this scenario we would like to check whether the three different access connections, Wi-Fi, Fiber and ADSL, are able to play the First Person Shooter (FPS) game Quake 3 with desirable quality (8 players). Quake 3 is a very fast and responsive game and the ping time is crucial for who is winning or losing [83].

Network Requirements
The important network performance parameters for a gaming service (FPS) were introduced in section 8.2.7. The bandwidth must be predictable, meaning it should not go below a lower threshold value. FPS gaming also has certain delay requirements as well as the loss should not be excessive. Data is transported with UDP. Because Quake 3 has good loss recovery techniques it allows a loss rate up to 35 % before the user experience is noticeable decreased [83]. Other FPS games, such as Halo, stops working if the loss rate exceeds 4 % [83]. In table 9.2 we have listed the Quake 3 requirements needed to get a satisfying game experience. Even though we in this table requires the delay to be below 150 ms it should be noted that there is a huge difference between 50 ms and 150 ms delay [38].

Control Environment
To set-up our control environment we need to know the traffic characteristics of Quake 3. We will use a simplified model based on the analysis performed in [60], where we will only use the average packets lengths and the maximum bandwidth rates in both directions (client to/from server). The maximum rates occur when multiple actions emerge simultaneously. Based on [60] we have chosen to use an average packet length of 65 bytes and a rate of 50 kbit/s from the client to the server. From the server to the client we have chosen an average packet length of

100 bytes and a 100 kbit/s rate. The environment set-up is illustrated in figure 9.2.

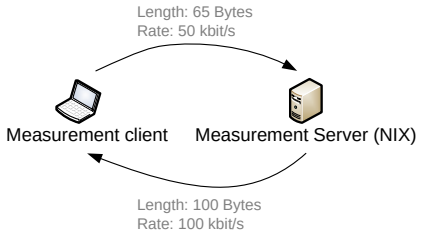


Figure 9.2: Environment for the Game Scenario.

Can Bredbandskollen predict the quality for gaming?
In table 9.1 we present the values measured with Bredbandskollen. From these values it is easy to mistakenly believe that all the access networks are more than good enough to satisfy the Quake 3 network performance requirements presented in table 9.2. This is not a valid conclusion because Bredbandskollen:

- ... measures the average download/upload rate. We have no guarantees that the bandwidth is not below certain thresholds in shorter time periods.
- ... measures the average response time for HTTP messages (uses multiple TCP packets). Quake 3 needs to know the delay for short UDP packets.
- ... does not measure the Jitter.
- ... does not measure the Loss rate.
- ... and the game server are not co-located.

Based on the control environment we have measured the actual values for the Quake 3 application. These values are presented in table 9.2. The measured delay for fiber and ADSL to the game server was smaller than the delay measured with Bredbandskollen. This is because the servers are not co-located. From the table we can see that the Wi-Fi access is not able to provide a satisfying game experience. This is because the bandwidth is too unpredictable combined with a too high delay variation compared to the requirement. We sent a constant stream of data (100 kbit/s) from the server to the client. We observe a huge bandwidth variation with a big difference of the lowest and highest sampled rate. The same effect was observed in the upload direction. Such high bandwidth variations lead to packets being queued in the network with the effect of higher delay. The reported delay for Wi-Fi with Bredbandskollen was 16 ms while we measured the delay to be 44 ms. We think this is caused by the high delay variation. Table 9.2 also shows that the delay is lowest for the Fiber connection, but only the ADSL provides no loss during our test.

Another interesting observation is that the Bredbandskollen measured the ADSL delay to be 35 ms while Quake 3 has an actual delay of 24.34 ms over ADSL. This is because Bredbandskollen measures the delay over HTTP, which uses much longer

Paramteter	Wi-Fi	Fiber	ADSL	Requirement
Down-rate				
min (kbit/s)	**40**	100	99	
avg (kbit/s)	99	100	100	10-90 kbit/s [60]
max (kbit/s)	**140**	100	101	
Up-rate				
min (kbit/s)	**6**	49	49	
avg (kbit/s)	49	50	50	14-50 kbit/s [60]
max (kbit/s)	**100**	50	50	
RTT (ms)	44	8.24	24.34	$<$ 150 ms [6]
Jitter				
server (ms)	**60**	0.12	0.31	$<$ 30 ms [5]
client (ms)	**118**	0.04	0.56	
Loss				
server (%)	0.80	0	0	$<$ 35 % [83]
client (%)	1.00	0.01	0	

Table 9.2: Quake 3 requirements is listed in the right column. The actual measured performance values with the gaming control environment are presented in the center column. Unacceptable values are emphasized in red.

TCP packets[2]. We used Wireshark and found that the delay-test performed by Bredbandskollen uses TCP packets with a size of about 400 bytes. A longer packet require a longer transfer time which can become a significant part of the delay, especially for ADSL links with a limited upload rate. To show the effect of packet sizes over ADSL links we have plotted the RTT for various TCP payload sizes for both fiber and ADSL in figure 9.3 (we excluded Wi-Fi because of the huge delay variation). The figure also illustrate that the added transfer delay is neglectable if we have sufficient transfer capacity, as we can see for the fiber measure.

9.2.2 Scenario: VoIP

The VoIP scenario is used to check if the three different access networks are able to provide acceptable network qualities required by a VoIP conversation. We will use the G.711, an ITU-T standard for audio compression and transport.

Network Requirements
The important network performance parameters was discussed in section 8.2.9. The bandwidth must be predictable because a VoIP session generates a constant and predictable stream of data. The rate for G.711 is 64 kbit/s [13]. The total one-way end-to-end delay (from mouth to ear) should not exceed 150 ms to provide

[2]The added distance to Stockholm will also add some extra delay.

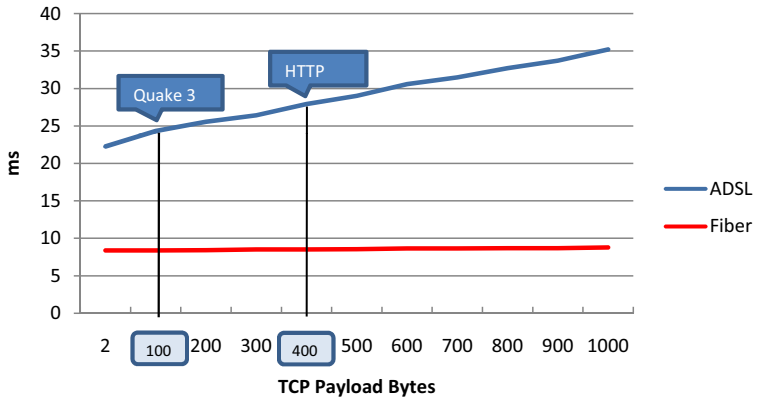


Figure 9.3: Measured RTT for various packet sizes for ADSL and fiber.

a good user experience [67]. The coding delay for G.711 is 40 ms [71] and the one-way network delay should thus not exceed 110 ms. From this we can roughly require that the RTT (two-way delay) should not exceed two times one-way delay, which is 220 ms[3]. Jitter buffers are used to transform asynchronous packet arrivals into a synchronous stream of packets. This is done by adding a variable delay at the receiving end. Cisco has performed extensive lab testing and found that voice quality degrades significantly when jitter consistently exceeds 30 ms [67]. Data is transported with UDP.

Control Environment
To set-up our control environment we need to know the traffic characteristics of G.711. We will used the traffic characteristics described in [13]. Figure 9.4 shows our system set-up where the server is located at NIX. We have, according to [13], chosen to use packet lengths of 160 bytes and a payload rate of 64 kbit/s in both directions.

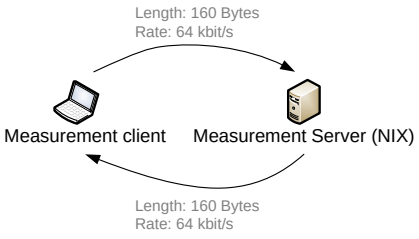


Figure 9.4: Environment for the VoIP (G.711) Scenario.

Can Bredbandskollen predict the quality for VoIP?
For the same reasons as discussed for the game scenario, it is not enough to

[3]This is a very rough simplification that works well for links with almost symmetrical one-way delay. A RTT of 220 ms will represent an absolute maximum, since higher values will imply one-way delay greater than 110 ms in at least one of the directions.

perform a network evaluation with Bredbandskollen in order to determine if an access network can provide a good VoIP quality. When comparing the requirements and measured values in table 9.3 we see that the delay variation for the Wi-Fi connection is too high to provide a VoIP conversation with acceptable quality. Also the bandwidth varies too much, resulting in added delay and potentially loss because of late arriving voice-packets. We also see that the delay is lowest for the Fiber connection, but only ADSL provides no loss during our test. Both the ADSL and Fiber connection provides a constant predictable stream of data at 64 kbit/s.

Paramteter	**Wi-Fi**	**Fiber**	**ADSL**	**Requirement**
Down-rate				
min (kbit/s)	**39**	64	64	
avg (kbit/s)	64	64	64	64 kbit/s [13]
max (kbit/s)	**104**	64	64	
Up-rate				
min (kbit/s)	**19**	63	63	
avg (kbit/s)	64	64	64	64 kbit/s [13]
max (kbit/s)	**125**	65	65	
RTT (ms)	44	8.24	24.34	$<$ 220 ms [67]
Jitter				
server (ms)	**45**	0.32	0.15	$<$ 30 ms [67]
client (ms)	**104**	0.04	0.41	
Loss				
server (%)	0.22	0	0	$<$ 1 % [67]
client (%)	0.12	0.002	0	

Table 9.3: VoIP (G.711) requirements is listed in the right column. The actual measured performance values with the VoIP control environment are presented in the center column. Unacceptable values are emphasized in red.

9.2.3 Scenario: Predictable Bandwidth

In this scenario we want to investigate how constant the bandwidth actually is. Bredbandskollen measured our bandwidth to a certain value for each of the different access networks, but this is only the average rate measured for a time period. How much does the bandwidth vary? This is important in order to know how much bandwidth we can expect to get and it represents the predictable bandwidth. This performance parameter is important for a service who wants to send a constant stream of data. We have in this scenario transported data with TCP from our measurement server to our measurement client.

Wireless Trondheim
We sampled the download rate for the Wi-Fi connection in 1 second intervals.

The result is presented in figure 9.5 where the average bandwidth is the blue line and equals the download rate measured by Bredbandskollen (to the right). As the graph illustrates, Bredbandskollen only tells the tester what the bandwidth is on average, but fails to say something about the variation. This is important information for applications who want to use a constant stream of data, such as video streaming, gaming and video-over-IP. We can see from the figure that the network over the measured 30 seconds is not able guarantee more than 180 kbit/s even though the average measured rate is 500 kbit/s.

Figure 9.5: Bandwidth sampling (1 second interval) of Wi-Fi (left) and reported values from Bredbandskollen (right).

NTNU Campus Network

We performed the same sampling for the fiber connection, presented in figure 9.6. In the figure we clearly see that the sampled download rate is roughly located around the average download rate. We can also notice a sudden drop around 14 seconds which probably was caused by competing traffic.

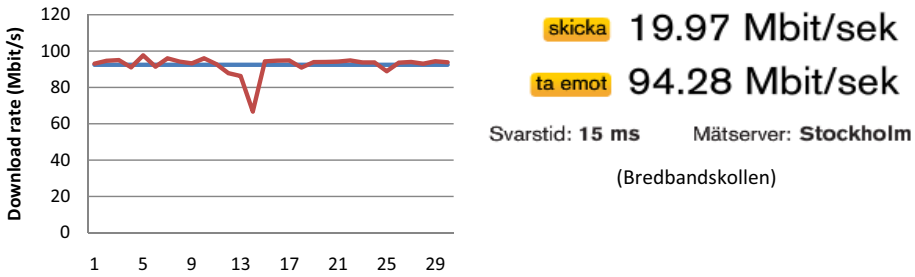


Figure 9.6: Bandwidth sampling (1 second interval) of Fiber (left) and reported values from Bredbandskollen (right).

ADSL Broadband Connection

At last we performed download rate sampling for our ADSL connection, presented in figure 9.7. The graph clearly illustrates the effect of bandwidth policing performed by the ISP. If the bandwidth is too high (above a certain threshold) the queue becomes full and packets are dropped. This results in an alternating high/low bandwidth where the average bandwidth is the blue line in the figure. If an application wants to send a constant stream of data it should probably not

exceed 4.5 Mbit/s to ensure that the subscribers download-queues are not filled up. Even though the bandwidth has this alternating behavior we should notice that most applications, such as streaming services and VoIP utilize buffers to smooth out these small bandwidth variations. The buffers will add some extra delay to the transferred data.

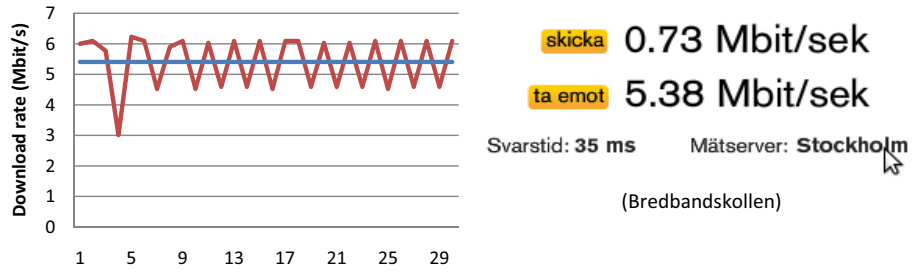


Figure 9.7: Bandwidth sampling (1 second interval) of ADSL (left) and reported values from Bredbandskollen (right).

9.2.4 Summary

In this section we have pointed out that an average measured performance parameters are not sufficient to reveal possible variations, which might be essential for some services. In section 11.1.3 we suggest some new ways of presenting the measured bandwidth, and how the measurements could be aggregated for statistical purposes.

Chapter 10

Measurement Statistics

In this chapter we will see how on-line test tools may collect and present statistics of the results acquired in each test. A tool with many users will provide a greater value beyond simple bandwidth measurement as soon as relevant statistics are presented. One single test result can give the end-user some quantification of the quality of her Internet connection at the moment. But when a large number of measurement results from a large number of users are combined, they may reveal trends that cannot be seen when only isolated measurements are evaluated. The statistics can be used by end-users who wish to compare different ISPs before buying or changing an Internet connection. The ISPs may use the statistics as an indicator of possible existing problems in their networks. The authorities, NPT in Norway, will also benefit from the statistics since systematic under-provisioning can be made visible and acted upon.

10.1 Statistics in existing tools

Some of the tools reviewed in chapter 6 records and presents measurement statistics. But as we shall see the presentation of the statistics and level of detail are varying much.

10.1.1 Speedometeret

Speedometeret collects statistics for each test performed, but do not provide any continuously updated statics to the users. Instead the results are published as a single article in the ITavisen newspaper each month. For a result to be a part of the statistics, the subscriber must have Internet connection from one of the ISPs who's IP-range is known to ITavisen. Then it is possible to associate the result with a certain subscription type from that ISP. An extract from the topmost part of the list of providers and subscriptions are shown in figure 10.1. As we see from this extract the results are averaged for each subscription type and they are ordered by kbit/s per NOK in descending order.

Resultatlisten for Oktober 2008

Operatør/abonnement	Snitt nedl.	Antall målt	kr/ mnd	kbps/kr nedl.	kbps/kr totalt*
BredbandsService Bredbånd Hjemme 10 Mbit/s (10240/10240)	8392	21	395	21.25	42.49
NTE Bredbånd AS Internett Familie (10000/10000)	9223	173	449	20.54	41.08
Lyse Super 50/50 Mbit/s (5000/50000)	4175	11	1450	2.88	31.68
Jæren Kabelnett 4/4 (4000/4000)	4093	33	275	14.88	29.76
NTE Bredbånd AS Internett Ekspress (25000/15000)	12854	22	699	18.39	29.42
Canal Digital Mega (20000/1800)	14486	79	547	26.48	28.87
Start.no Boost! (24992/1250)	8093	21	298	27.16	28.52
Canal Digital Mega (16000/1600)	13474	74	547	24.63	27.10
EB Altibox Internett Familie (10000/3000)	8777	118	449	19.55	25.41
Lyse Ekspress 25/15 Mbit/s (2500/15000)	2348	11	699	3.36	23.51

Figure 10.1: Extract from the statistics presented on ITavisen.

The fact that the measurements are directly connected to a specific subscription type from the user's ISP increases the usefulness of the statistics. But since the user is trusted with the task of selecting the right type of subscription from a list, the statistics soon become less trustworthy. We have also experienced that the list does not cover all the subscriptions one ISP delivers. The ISPs constantly upgrade their assortment of subscriptions, and connection speed for already existing subscriptions are also upgraded from time to time. This may lead to end-users who do not know exactly what subscription type and connection speed they pay for[1]. All this uncertainty of the user input, results in statistics of

[1]Neither of the authors of this thesis knew exact connection speed for their home Internet access at semester startup.

low value. ITavisen seem to have stopped publishing updated statistics, and the newest available statistics per April 2009 are the results from October 2008.

10.1.2 Speedtest

Speedtest provides some statistics with coarse granularity to the user. Average download and upload rate are given for continents, countries and regions. These numbers are an average of every tests performed in the area of interest and do not give any information about different ISPs or subscriptions. In addition a list of the top 10 ISPs in the selected country or region are provided, as shown in figure 10.2. These ISPs are rated according to their average download and upload rates. Since the ISPs usually provide numerous different subscriptions and connection speeds, this statistics doses not really give much valuable information to the user at all. An ISP that provides mobile solutions with a relatively low data rate compared to the high-end stationary solutions will not get a good rating in this statistics. The presentation of the results actually favors the ISPs that only provide a few subscriptions with high data-rate at the expense of the ISPs with a wide assortment of subscriptions.

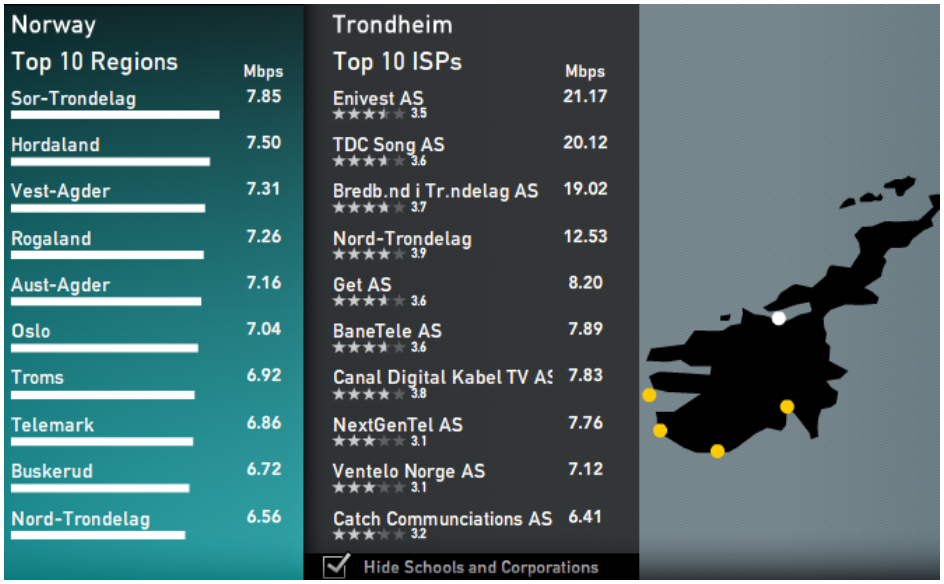


Figure 10.2: Statistics from Speedtest when Trondheim and download speed is selected.

When you run the Speedtest, a cookie with a unique identification is saved to your computer. This is done to provide a page with personal statistics. On this page you can see your previous results with download and upload speed together with latency. Each result is associated with a time stamp and an IP-address. If you have done measurements from different networks, you can select to view only the results from one specific IP-address. This statistics can be useful if you have experienced some problems with your Internet connection and wish to look for trends in your measurements over time.

10.1.3 Bredbandskollen

The statistics provided by Bredbandskollen are more detailed than the previous mentioned tools. The information stored for each test performed are [64]:

- Date and time of the test execution.
- Measurement results:
  - Upload speed
  - Download speed
  - Latency
- An ID of the measurement server used.
- The IP-address of the terminal executing the test.
- The users ISP, determined from the IP-address.
- The geographical location of the users ISP.
- Browser and operating system used for test execution.
- The user's stated peak capacity.
- An user ID which identify the cookie stored on the user's computer.

When a test is completed, the user is asked to select what type of connection he has. Based on predefined intervals for each type of connection, the measurement gets a rating that is good, acceptable or not acceptable [64]. This classification is only based on the download speed achieved, so it does not take the upload speed into account. The choices available for the user are not real subscriptions, but rather predefined connection speed intervals for download speed combined with different access technologies. This is necessary to keep the statistics general and not dependent on specific products from each ISP, as is the case with the ITavisen statistics. This also makes it easier for the user to choose the right alternative, but at the cost of less accurate statistics. One example is the interval 12-24 Mbit/s ADSL. There exist many different subscriptions that fit into this interval, and we think it is unfair to compare the results from one ISP with a 12 Mbit/s subscription with another ISP with a 20 Mbit/s subscription.

In figure 10.3 we have presented a screen shot of the statistics page provided. Here we can see different possible choices of display. The topmost tabs decide how the results shall be listed. The listing may be grouped by ISPs, connection speed or county. In figure 10.3 we have selected to view the results for each ISP. Then we can choose which connection speed the users have registered, and in which county the tests of interest were executed. In addition we may decide from which month we would like to view the results. The list shown after these parameters are selected may be sorted by different criteria. As default the operator with the most registered measurements are shown first. But the list may also be sorted by average download or upload speed, latency, percentage of tests within the acceptable, not acceptable or good rating. As far as we can see, there is no choice available to see the results in a larger, or smaller, time interval than one month.

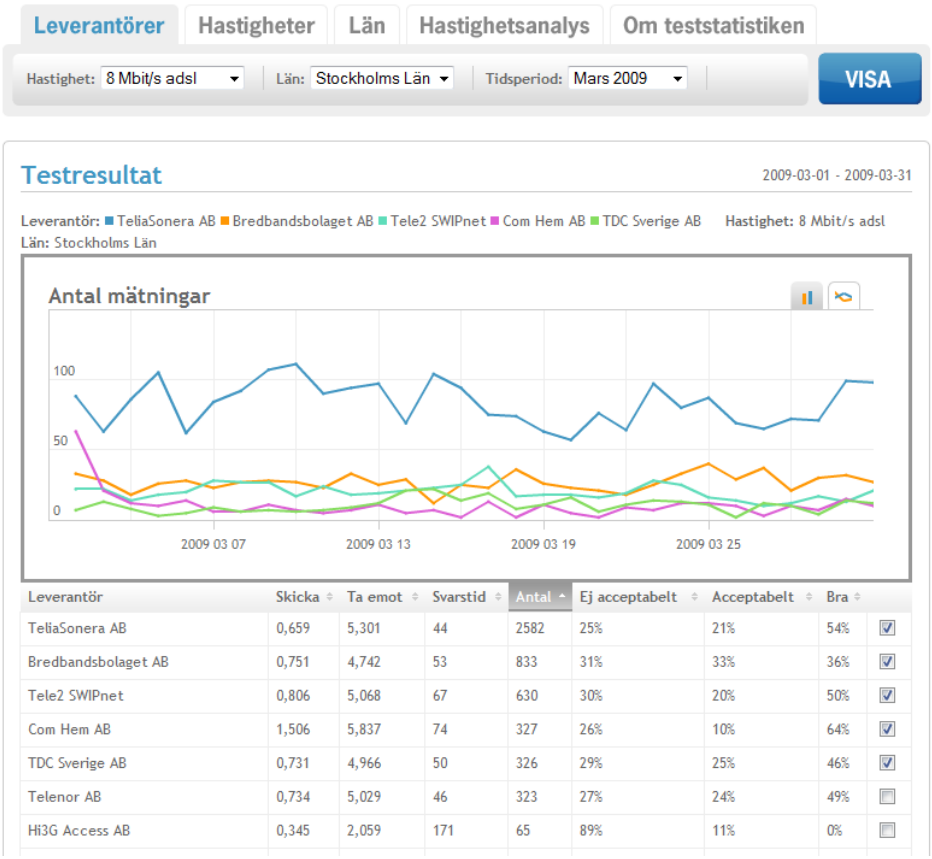


Figure 10.3: The statistics page of Bredbandskollen.

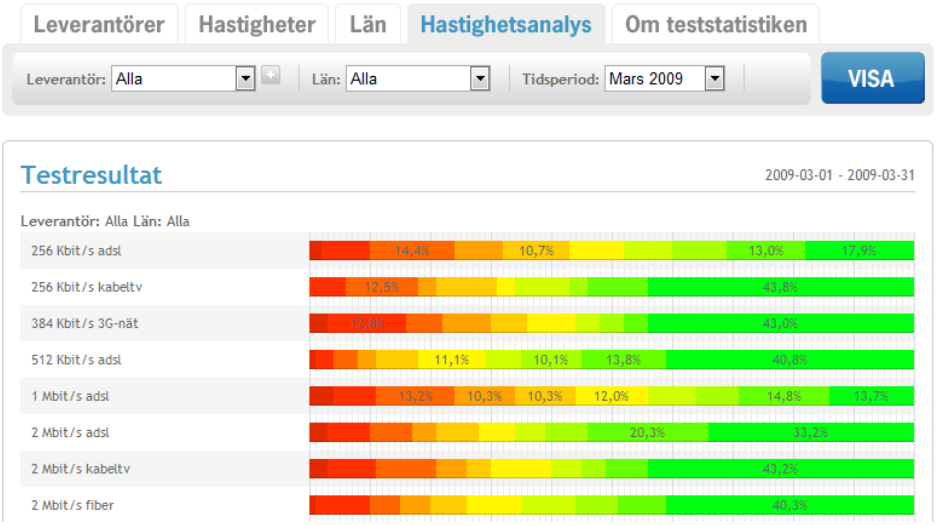


Figure 10.4: Speed analysis from Bredbandskollen.

The speed analysis tab are shown in figure 10.4. This function provides a tool to compare how the achieved download speeds are distributed. The different test results are shown with different color based on the deviation from the connection speed stated by the user. The color of each bar indicated the deviation from the stated connection speed. The width of each bar shows how many percent of the test that falls within each interval of deviation. The light green bar, for example, shows the tests that have a deviation less than 10 % of the stated connection speed. A good connection will then typically be characterized by a wide green area and a red area as thin as possible.

10.2 Statistics in NPT Broadband Test Tool

The new broadband test tool that NPT are going to release in 2009, NBTT, aims to be the new leading bandwidth test tool in Norway. NPT estimates that NBTT will have at least 200 000 tests performed per month[2]. With such large number of tests it is obvious that the results of these tests may form the basis for some statistics with great value for NPT, the ISPs and the end-users. The challenge is to present the statistics in a way that reveals its potential value.

As described in section 2.2, NBTT will be very similar to the Swedish Bredbandskollen and delivered by the same company that has delivered Bredbandskollen. The measurement will be done in the same way, and the database will include the same type of data. Therefore, this new tool will have the potential to provide statistics at least on the same level as Bredbandskollen, and hopefully even better.

In section 11.1.3 we will discuss proposals for further development of the statistics presented to the users of NBTT.

10.3 Measurement Selection Bias

A broadband test tool like NBTT is an active broadband test where the end-user is the one who decides that a measurement shall be done, and when it shall be done. It is important to be aware of this, since it may have some important effects on the resulting measurement statistics. Users that choose to do the measurement themselves cannot be said to represent a real random selection.

Many end-users will not even think of the possibility of testing their broadband connection. These are the users that do not have much insight in network technology, and are happy as long as they get access to the newspapers they usually read. This type of end-users may therefore be underrepresented in the statistics.

There are also other biases that can affect the statistical results of an broadband test tool. To see this, we must look at what kind of end-user who actually uses

[2]200 000 measurements are the "guesstimate" received from NPT based on the number of visitors on telepriser.no and the Swedish bredbandskollen.se.

such tools. We believe that most of the measurements performed are done by end-users who:

- Experience problems with their Internet connection.
- Consider changing ISP or subscription.
- Have just changed ISP or subscription, and would like to test the new connection's performance.

The first category of users consists of the ones who experience problems with their Internet connection. These problems may have many different causes, either in the ISP domain, or the home network domain. Nevertheless the measurements made by these users will have bad results compared to what type of connection the end-user is paying for.

The second category is the users who consider changing subscription. The main incentives an end-user have to do this is that the existing connection does not perform satisfactory or that another ISP can provide the same or a better service at less cost. Among this category of end-users there will therefore be a part that is not satisfied with their connection, and their results will affect the statistics.

The third category is users who do not necessarily have any problems with their Internet connection. They just want to check the characteristics of the service they receive. This category will therefore not have an overrepresentation of users with troublesome Internet access.

To sum up, this section suggests that the resulting selection of users can be biased. An overrepresentation of end-users that experience problems with their connection is a possible outcome. At the same time there might be groups of users that are underrepresented. This is important to have in mind when evaluating such statistics.

Chapter 11

Future Aspects

During our studies of the existing tools we gained much knowledge of what is already done in the area of Internet access evaluation. We have also explored areas not yet covered by these tools. In this chapter we want to make use of our gained knowledge and recommend possible extensions to the planned NBTT. These suggestions should hopefully contribute to a richer and more complete service, which can be used to evaluate more than just the momentary achievable throughput.

The last section of this chapter suggests various future applications of a test tool such as NBTT. A rich tool with many features can hopefully be used to more than just serving people who experience problems with their connection.

11.1 Future Possibilities for NBTT

The planned NBTT was introduced in section 2.2. We saw that NPT plans to use the measurements engine developed by Ookla [55]. The Ookla engine is a closed source. This will of course constrain the possible extensions of the applied measurement engine to be developed by Ookla. In this section we will not consider whether it is feasible or not to extend this engine, but rater suggest extensions freed from this constraint.

11.1.1 Multiple Locations

NPT plans in a later release of NBTT to host the service at various regional exchange points. In this section we will review the effects (positive and negative) of multiple hosting locations for NBTT.

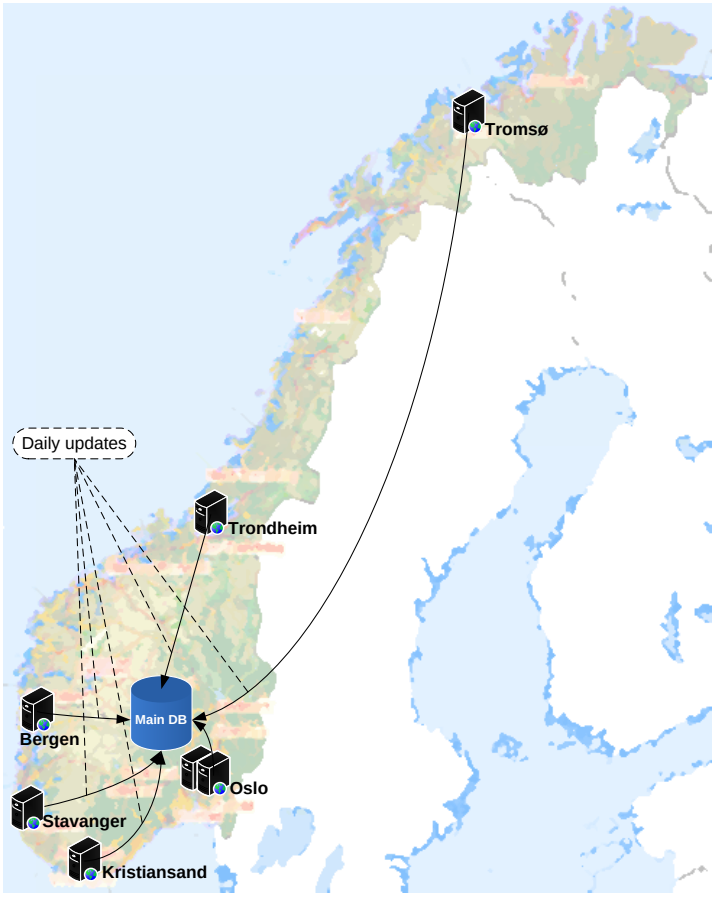


Figure 11.1: The figure shows possible locations for a broadband test in Norway.

Possible hosting locations in Norway is illustrated in figure 11.1. In the figure we have suggested some of the largest cities in Norway (regional exchange points)

as possible hosting locations. We also suggest that each local server collects daily statics and exchange updated statistics with the main database, preferably at night hours. Each local server should also contain the aggregated statistics available for the end-user. Basically this means that each local instance can work independently from the centralized server. This will heavily decrease the possible network load caused by NBTT. Multiple instances will also provide redundancy, which will increase the fault tolerance of the service.

By having local servers at regional exchange point we will decrease the overall load from the service, as well as we isolate what the service actually measure. If the NBTT server is located near the end-user (but of course outside the access network) we get a good measure on the possible access network bandwidth without other factors such as cross traffic, the physical distance and the Norwegian infrastructure to affect the obtained results. These factors might also be out of the ISPs' control. On the other hand we should take into consideration that most of the large Internet content providers in Norway are located in Oslo. In addition the international traffic to/from Norway is usually routed through Oslo. Most subscribers are not interested in their local access network capacity, but rather what capacity they can expect into the core network, close to the content of interest. In chapter 7 we saw that there usually is small deviation from running a local test in Trondheim (Iperf-loc) versus running a test at NIX (Iperf-nix). Some ISPs might also speculate in providing high access bandwidth with the cost of a higher probability of congestion at the interface out of their network, resulting in a lower actual bandwidth.

To sum up:

1. Local hosting is beneficial to isolate the access network in the measurement, but...
2. ...in the Norwegian Internet scenario it should be possible for the end-users to specify what server to test against. This will also allow the subscribers to perform debugging on their own, e.g. checking the bandwidth to another city of interest.

11.1.2 Network Neutrality and Local Test-Processes

In this section we suggest an extension to the current NBTT system set-up which might be used to discover discrimination with regard to content provider based on sender or receiver address (the last part of the third principle of network neutrality, see chapter 4).

The idea is to have a local test process running in the content providers' network. This process can be called by the end-user performing the test and data can be exchanged to measure the bandwidth (up and down). We have illustrated the possible test set-up in figure 11.2. Even though this figure shows that both content providers are reached though NIX we should notice that this is not always the case. Some content providers are even located within a ISPs network. Special peering agreements between ISPs or between a ISP and a content provider might affect the path the data travels. But because we suggest putting a NBTT process within the content providers' network it does not actually matters how the data

travels (it will of course be harder to compare the measured bandwidths). Each test process should be identical with regard to applied protocol and measurement technique. This allows comparison of individual measurements.

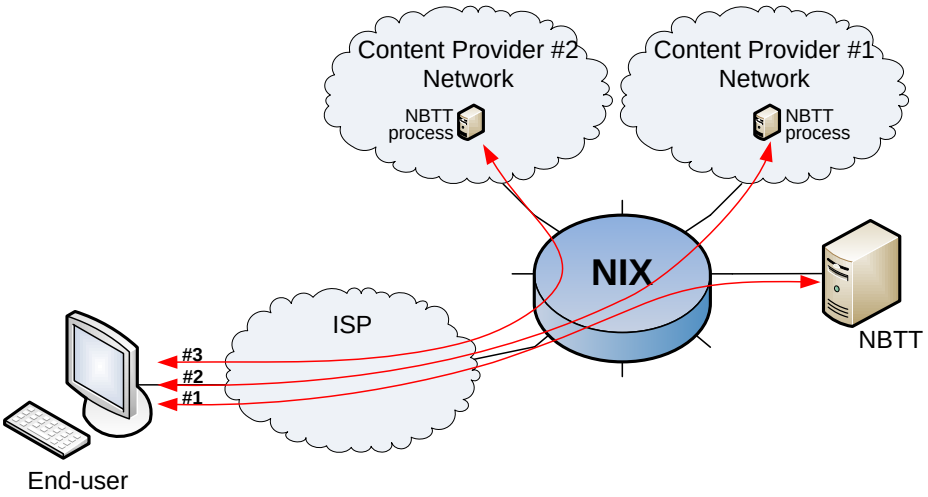


Figure 11.2: Possible extension of NBTT which might discover network neutrality breaches.

The figure illustrates the following measuring strategy:

- Test **#1** – Measure bandwidth against NBTT server. This value forms a reference for what we could expect an optimal value to be.
- Test **#2** – Measure bandwidth against NBTT test-process at content provider #1.
- Test **#3** – Measure bandwidth against NBTT test-process at content provider #2.

After completing the measures the obtained values can be compared against each other, and if they deviate one can assume possible discrimination between the content providers. One must notice that a single measure can not reveal traffic discrimination. This is because a difference in measured bandwidth can be a result of:

1. Sudden capacity variations in the Internet.
2. One of the content providers having trouble with the network (e.g. a lot of simultaneous users).
3. Different peering agreements among the ISPs, or possible agreements between a ISP and a specific content provider.

Because of all these possible explanations for why we measured a difference between the content providers we suggest to use multiple measures from the collected statistics when determining if there exists possible discrimination. Aggregated measures performed by multiple users from the same ISP can be used to check whether there is a discriminating trend. But we should notice

that a possible observed trend can be a result of either network problems at the content provider or peering agreements as mentioned above.

There exist enormous numbers of content providers. It would be unfeasible to run a test against all content providers. Thus we suggest that a network neutrality test should have two possible options:

1. Only a selection of important Norwegian content providers, where NPT suspects possible discrimination, are selected by default for the network neutrality test.
2. The end-user can select which content providers she wants to test the bandwidth against.

A single measure cannot verify whether there exists traffic discrimination or not, but it can inform the user on what speeds she gets to a variety of important content providers. If also statistics from these measures are gathered it can be important information for a user who wants to buy an Internet subscription. Aggregated statistics from these measures can also be used to see if there is a discrimination trend.

11.1.3 Statistics

In this section we will present some proposals for further development of the statistics presented to the users of NBTT. These are mostly ideas that we think could contribute in making the statistics more expressive.

Automated Subscription Determination

As we have seen in chapter 10, the already existing tools have some challenges in how the connection speed is determined for each test. Two problems we have identified are that the user is trusted with the task of selecting the correct speed, and that the selection is not always intuitive.

Both problems can be solved if the identification of subscription type could be automated. We will propose an idea for solution here. If every ISP were required to offer a service to NPT where subscription type could be identified for a certain IP-address, then the user selection as a source of error could be eliminated. This scenario is illustrated in figure 11.3.

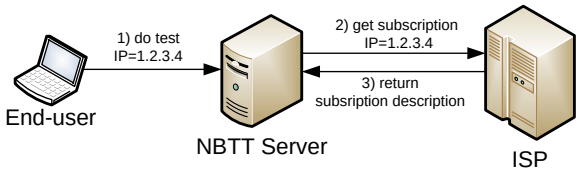


Figure 11.3: Possible scenario where ISPs provide subscription information

If this solution were implemented, then each measurement in the database could be connected to an existing broadband product. In addition each broadband product could be mapped to a defined general connection speed, like the intervals used in Bredbandskollen. This will result in a more accurate statistics, while still keeping it general enough to survive in a continuously changing broadband market.

Price Comparison

One of the goals of the NPT is to make sure that the consumers get access to communication services at reasonable prices by promoting effective competition in the telecommunications markets [48]. One of the ways NPT can contribute to maintain a healthy level of competition, is to make price comparison more available to the end-user. This is what they have done with the telepriser.no website for different telecommunications services [50]. On this site the end-user may compare different broadband subscriptions based on price, and promised data rate.

Since promised data rate, and experienced data rate are not always the same, it could be beneficial to also take measurement statistics into account before comparing different subscriptions. As a natural extension of the statistics collected in NBTT, it could be possible to register what the users pay for their Internet connection. This could be used to calculate some sort of NOK per kbit/s, based on the achieved test result. The users of NBTT could then be presented statistics of price per kbit/s from different ISPs delivering broadband connections in the area where the user lives. Then it will be easy to compare one ISP versus other ISPs based on how well they score on the NOK per kbit/s scale.

It is worth mentioning that the registration of what users pay for their Internet connection has the same weakness as registration of connection speed if the user is trusted with the task of entering the right value. We suggest that price could be collected directly from the ISPs in the same way as the subscription type shown in figure 11.3.

Measurements Distributed in Time

Figure 11.4 shows a screen shot of the statistics provided for all ports on NIX1. Most of the Internet traffic within Norway passes through this point, and the graph shows a significant variation in the amount of traffic at different times of the day. It is reasonable to believe that this traffic variations may affect the measurement results for a bandwidth measurement tool like NBTT. Therefore it could be interesting if the statistics could be aggregated in a way that makes it possible to view results from different time of the day, the week, the month and even the year.

If one ISP frequently delivers lower capacity to its users during the afternoon, this kind of statistics can be used to reveal this under-provisioning. Comparison can also be made, and a user that requires a good service “24/7” will then probably choose an ISP that have good results in the statistics at all times of the day.

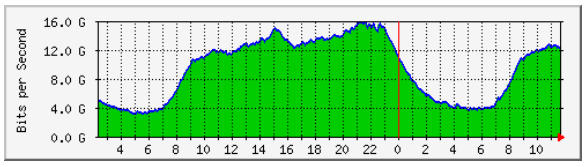


Figure 11.4: Traffic statistics for all ports on NIX1 28.04.2009 [45]

.

Measurement Presentation

Based on how MySpeed, introduced in section 6.1.4, presents the measured bandwidth and our experience in the testing of predictable bandwidth in section 9.2.3 we suggest a new way to present the measured bandwidth. We suggest that experienced users should have the possibility to view a graph over the sampled bandwidth during the testing, as we did in section 9.2.3. This way it will be possible for the end-user to see how much the bandwidth varies during the testing. This is important because a lot of applications, especially those utilizing UDP for transport (and thus do not implement standard congestion control algorithms), have certain requirements on how much the bandwidth can vary. Of course other important network performance parameters, such as the delay, jitter and loss could also be presented in a graph.

Because it is impractical to store a full set of samples from a bandwidth test we suggest using a histogram to store the values. The intervals, or the bins, of the histogram, should be selected according to typical applications and their requirements. This requires deep knowledge and studies of the available applications and Internet services used today and we will thus not suggest what these intervals should be. We have however suggested how such a histogram can be presented to the end-user in figure 11.5. The histogram is a hypothetical example of the sampled bandwidth for a user who have executed NBTT ten times on the same connection. It is of course possible for users to switch locations, and thus have different access capacities. We should not aggregate measured bandwidths from different access networks. This is to ensure that the measurements are comparable. When we have stored the previous sampled bandwidths for an end-user on one connection we can present the user with probabilities for a certain bandwidth. E.g from figure 11.5 we can assume that the user can expect to get a bandwidth equal to or greater than 4 Mbit/s with 96 % certainty.

The end-user is often interested in comparing his measurements with others having the same subscription and ISP. This will enable the user to evaluate whether the obtained results are representative for his subscription or not. We have illustrated a possible presentation in figure 11.6. In this histogram we have both presented the end-user result and the overall result from users having the same subscription and ISP. If the end-user acquires worse results than the overall average this can be an indication of last mile network issues. In figure 11.6 we see that the “My Results” only have 59 % of the measurements above 5 Mbit/s while “others with the same subscription” have 75 % of the measurements above 5 Mbit/s.

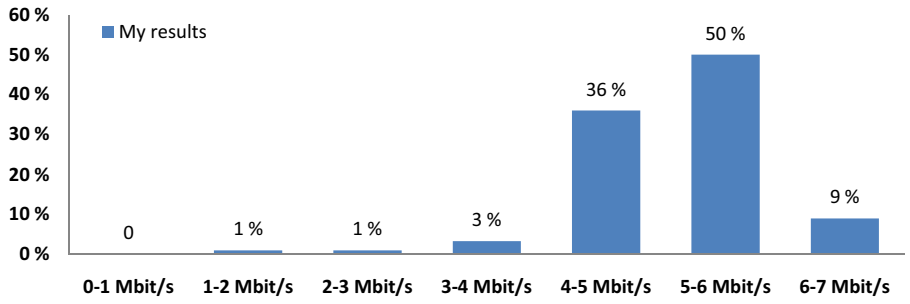


Figure 11.5: A possible histogram over the measured bandwidths for an end-user.

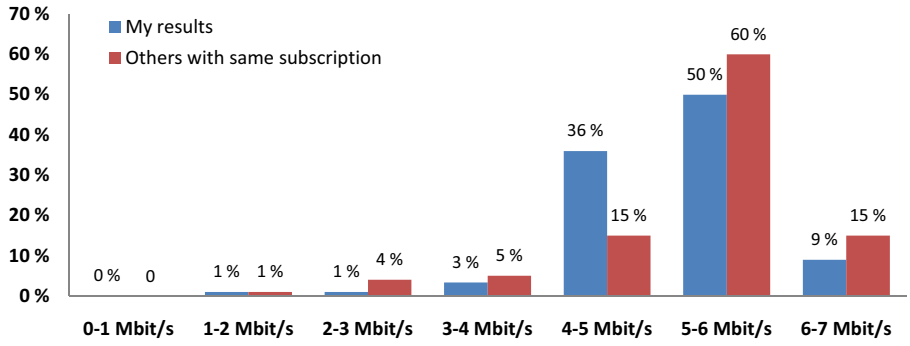


Figure 11.6: A possible histogram over the measured bandwidths for an end-user compared with other users with same subscription and ISP.

When presenting this type of statistics to the user it is important to take the uncertainty into consideration. A single measurement should not be presented to the end-user as a statistical representative selection. The user must be informed of the validity of the measured statistics. We suggest that this type of presentation should require the end-user to perform a given number of measurements before charts are presented.

11.1.4 Profile Extension

Evaluate all Profiles

In chapter 8 we suggested to classify the end-users into different user profiles. Even though we can expect that most of the end-users fit the basic profile, there still exist users with other needs. We suggests that the NBTT should be expanded to be capable of evaluating all the user profiles. If all the different profiles are covered by NBTT it will be easier for the end-user to evaluate whether an Internet subscription fits the actual end-user needs. The services should also gather statistics from individual measurements. This way NBTT can also guide end-users to buy the cheapest Internet subscription which fits the actual end-user needs.

To be able to evaluate all profiles, NBTT must be expanded to evaluate the performance of all basic network parameters (bandwidth, delay, jitter, loss and predictable bandwidth) presented in table 8.1. The service must also evaluate the different parameters with the corresponding transport protocols and use packets with the correct size (keep in mind the measured RTT for ADSL in figure 9.3). This ensures an accurate performance evaluation.

The challenge with this extension is emulating all the different possible services accurately and as simple as possible. This requires extensive research of the available services today. Another challenge is to integrate the extension with the planned measurement engine from Ookla, which is a closed source.

Profiles and Network Neutrality

By measuring the performance of profiles and their specific services we can actually use the obtained information to reveal breaches to NPT's second and third network neutrality principle presented in section 4.2.

By simulating different services (packet size, transport protocol, ports etc.) it is possible to collect measured data for various services and applications. This information can be compared and if they deviate much it is possible to assume possible traffic discrimination among the services. But as we discussed in section 11.1.2 we should be aware that a single measure is not enough to reveal traffic discrimination. But because each measure is done to the same server we can use statistics to find whether it seems to be a discriminating trend between different services for one ISP.

User Profile Statistics

The different user-profiles we have defined in chapter 8 can potentially add some value to the measurement statistics. If the measurements in NBTT were extended to include support for user profiles, then the statistics may also include information relevant to the different profiles.

An end-user that belongs to a certain user profile would probably be interested in which ISPs that provide the best service for her demands. This can be done by ranking the different ISPs or subscriptions based on the measured values that are important for the different profiles. For each user profile, the relevant parameters can be weighted so that the ones important for one profile become more significant in the comparison.

11.2 Future Applications

In this section we will look into possible future applications of a service such as NBTT. Some of these applications requires modification of the planned NBTT service.

11.2.1 Mobile Terminal Access Capacity Evaluation

Terminals of multiple access channels have to share the capacity of a certain multi user access point. This will cause the measured bandwidth for a terminal to vary according to how many concurrent terminals the terminal is competing with. We suggest that clients running NBTT from a hot-spot Wi-Fi connection or on a GSM/UMTS connection also provides the unique identification for the access point they currently use. For a Wi-Fi connection this would be the MAC address of the wireless gateway and for a GSM/UMTS connection this would be the base station identification. NBTT will thus be able to collect multiple measurements and can provide the mobile user with valuable information. This will for example allow NBTT to rate the measured performance according to previous measured values for this access point.

11.2.2 Available Bandwidth Test to Select Download Server

Today there exist different download services which provide files for download[1]. These services are popular because they got many servers located at multiple locations which allows the user to select a *close* server, which hopefully can provide a high bandwidth. The problem with these services is that the server is selected (sometimes also automatically) based on location and not the currently available bandwidth. We suggest that NBTT could provide a client able to measure the available bandwidth between the user and a selection of possible

[1] Such as `http://download.com`.

servers (maybe utilizing some of the same techniques used by Abget, introduced in section 6.2.7), from this a optimal download server can be selected.

11.2.3 Passive Measurement Combined with NBTT

The planned NBTT relies on active measurement techniques. Active techniques requires artificial data to be injected in to the network, and this have certain drawbacks discussed in section 5.3. Passive measurements, introduced in section 5.2, only observe the already existing traffic, and do not insert traffic. There are several possible approaches to enforce passive measurements in the planned NBTT. We suggest an extension to NBTT where the idea is to develop a client side application, which passively collect network performance statistics by sniffing and analyzing the packets at the end-user equipment. This can either be a lightweight application at the end-user machine, or integrated in the modem connecting the end-user to Internet. By passively collecting statistics we ensure that the data represents actual experienced performance. Passively recording all packets flowing across an interface would be quite resource consuming and it is thus important that the passively measured data are sampled and aggregated rationally. This allows the application to be lightweight which is essential for end-users to take the trouble of installing it. We also believe that it has to offer the end-user some value to motivate use of the application. These added values could be more precise evaluation of the broadband connection, better presentation of statistical data, real-time monitoring of link status and new features such as highest data rate last month, sent data last week, average bandwidth utilization etc.

This scheme has certain benefits, briefly mentioned below:

- **Implicit Profiles**
  If passive statistics is collected we can determine the end-user profile from the collected statistics. This makes the profile implicit, and the user does not need to select her profile.
- **Network Neutrality**
  Passive measured network performance will give valuable information in revealing traffic discrimination based on content type, application, service and receiver or sender address.
- **Discovering ISPs' Busy Hours**
  ISP busy hour happens when there are many simultaneous users and the aggregated traffic is closing up on the network capacity. By passively measuring the end-users' bandwidth we can gather statistics which can be used to reveal the ISPs' busy hour.

Chapter 12

Conclusion

When designing a broadband evaluation tool it is important to know what factors might affect the obtained results. One important aspect is the properties of the transport protocol used in the evaluation. In chapter 3 we reviewed the important properties of the common transport protocols used in the Internet today. We showed that TCP have certain properties such as slow-start and congestion avoidance which will affect the measured bandwidth, if not accounted for during the evaluation.

Norwegian Post and Telecommunications Authority (NPT) plans to develop a neutral on-line broadband evaluation service and release it in 2009. We have used the name NPT Broadband Test Tool (NBTT) throughout this thesis when referring to this service. NBTT will be developed by the same company that developed Bredbandskollen and is thus heavily inspired by this service. NPT plans to use the same measurement engine, the same technology and collect the same statistics.

NPT was interested in an evaluation of the planned NBTT's capability of revealing traffic discrimination and breaches to the principles of network neutrality. In chapter 4 we reviewed the network neutrality principles and found that the planned service is **not** able to evaluate any of the network neutrality principles. This is mainly because of the limitations in the planned architecture combined with the complexity of network neutrality.

In chapter 6 we reviewed existing broadband evaluation tools based on active measurement techniques. The existing tools were categorized into two groups, namely on-line and stand-alone. Some of the tools evaluated the same performance parameters with different approaches. None of the studied tools covers all network performance parameters introduced in section 5.1.

In chapter 7 we benchmarked some of the most common on-line tools. We established a measurement server at Norwegian Internet eXchange (NIX). A dedicated Linux bridge was set-up to enforce controlled bandwidth limitation. The motivation for setting up a Linux bridge is that none of the available bandwidth shaping tools running in the operating system's user mode were able to limit the bandwidth sufficiently accurate for our needs. By connecting this

bandwidth limiting bridge between a computer and the Internet access, we could efficiently simulate different access data rates.

Further in chapter 7 we found that most of the tools, including Bredbandskollen, performed well for common asymmetrical access-bandwidths used in Norway today. NBTT is planned to use the same test engine used by Bredbandskollen, thus we can expect that the obtained results for NBTT will be close to the results obtained with Bredbandskollen. For bandwidths above 20 Mbit/s we found a difference between tools measuring Bulk Transfer Capacity (BTC) and tools measuring the achievable throughput. One example is Speedometeret, which measures BTC, underestimating a 60 Mbit/s download rate with about 12 %. Tools measuring the achievable throughput, including Bredbandskollen, gave good results for rates up to 60 Mbit/s. All tools using HTTP POST, including Bredbandskollen, had problems with upload rates above 3.2 Mbit/s.

Chapter 7 also shows that implementation technology chosen will affect the obtained results. JavaScript was found to be inaccurate and underestimated a 12.8 Mbit/s download rate with almost 10 %. This is mainly caused by timing inaccuracy in the JavaScript implementation. We also found that the Flash plug-in for Mozilla in Windows have issues with high upload data rates. This is of great importance for the new NBTT which will be implemented in Flash.

A broadband test tool should evaluate the quality of a broadband connection in context of its usage. In chapter 8 we have categorized different profiles covering different types of users. The profiles we defined were: *private basic*, *private gaming*, *private multimedia*, *business basic* and *business multimedia*. We categorized the profiles by their service types. For each service we investigated the significant network performance parameters. From this we deduced what network performance parameters that must be evaluated for each profile. We found that the planned NBTT will only be able to evaluate the performance of the *private basic* profile. As a first release this is adequate but as a future-oriented service, NBTT should be expanded to include profile based evaluations. In chapter 9 we showed that average measured values can be insufficient when predicting the experienced quality for some services. We showed that Bredbandskollen concluded that our Wi-Fi connection quality was sufficient for gaming and VoIP, while our detailed analysis disproved this conclusion. Bredbandskollen's average values conceal the variations in both delay and bandwidth.

In chapter 10 we reviewed the statistics collected and presented by the existing tools. We found that the tools providing statistics differed in their presentation. Because of NBTT's similarities with Bredbandskollen, it will have the possibility to provide statistics at least on the same level as Bredbandskollen. During our studies we discovered new possibilities for the planned NBTT, which will make the statistics more expressive to the users. We suggested price comparison, automated subscription determination, measurements distributed in time and user profile statistics. These possibilities was introduced in section 11.1.3.

In chapter 11 we made use of our newly gained knowledge and recommended possible extensions to the planned NBTT. We reviewed the implications of using multiple server locations. Hosting the service at multiple locations is beneficial because the measurement server will then be located close to the end-

user, reducing the number of factors which may affect the obtained results. In addition, multiple servers allow load sharing and provide redundancy for fault tolerance. The drawback of hosting NBTT at multiple locations is that most of the Norwegian Internet traffic is routed through the NIX, located in Oslo. Therefore a good measurement result to a local measurement server might not reflect the actual Internet performance.

We suggested in section 11.1.2 to use local test-processes located within the content providers network. This could, in combination with statistics, be used to reveal breaches to NPT's network neutrality principle three. Because enormous numbers of content providers exist, it would be unfeasible to run a test against all of them. We suggested that a predefined selection of the most important content providers could be a step in the right direction.

In this thesis we have treated a huge and comprehensive subject, and we have only been able to look into some of the many important aspects. Broadband evaluation and network neutrality are subjects that will form the foundation for more research and academic work in the future. Hopefully NPT will take advantage of our findings and our recommendations for future development of NBTT.

References

[1] Karl Aberer and Manfred Hauswirth. An Overview on Peer-to-Peer Information Systems. In *Workshop on Distributed Data and Structures*. WDAS, 2002.

[2] Kristen Allen. T-Mobile blocks Skype for German iPhones. Internet, 2009. `http://www.thelocal.de/sci-tech/20090331-18359.html` Last visited: 30.04.2009.

[3] Leopoldo Angrisani, Salvatore D. Antonio, Marcello Esposito, and Michele Vadursi. Techniques for available bandwidth measurement in IP networks: A performance comparison. Passive and Active Network Measurement, 5th International Workshop, PAM 2004, Antibes Juan-les-Pins, France, April 19-20, 2004, 2004.

[4] Patrik Arlos and Markus Fiedler. Accuracy Evaluation of Ping and J-OWAMP. *Swedish National Computer Networking Workshop*, 2006.

[5] Grenville Armitage and Lawrence Stewart. Limitations of using Real-World, Public Servers to Estimate Jitter Tolerance Of First Person Shooter Games. In *ACE '04: Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 257–262, New York, NY, USA, 2004. ACM.

[6] Grenville J. Armitage. An Experimental Estimation of Latency Sensitivity In Multiplayer Quake 3. Internet, April 2003. `http://caia.swin.edu.au/reports/030405A/CAIA-TR-030405A.pdf` Last visited: 02.05.2009.

[7] S. Bellovin. A Best-Case Network Performance Model. ATT Research, February 1992.

[8] Tim Berners-Lee. Net Neutrality: This is serious. Internet, 2006. `http://dig.csail.mit.edu/breadcrumbs/node/144` Last visited: 22.05.2009.

[9] Tønnes Brekne, Marius Clemetsen, Poul Heegaard, Tone Ingvaldsen, and Brynjar Viken. State of the Art in Performance Monitoring and Measurements. Telenor, Scientific Report, 2002. `www.telenor.com/rd/pub/rep02/R_15.pdf`.

[10] Martin A. Brown. Traffic Control HOWTO, Version 1.0.2. Internet, Oct 2006.
http://tldp.org/HOWTO/html_single/Traffic-Control-HOWTO/
Last visited: 23.03.2009.

[11] Richard A Carlson. Developing the Web100 Based Network Diagnostic Tool (NDT). Internet, 2003.
http://www.nlanr.net/PAM2003/PAM2003papers/3703.pdf
Last visited: 22.05.2009.

[12] Richard A Carlson. Network Diagnostic Tool (NDT). Internet, 2004.
http://e2epi.internet2.edu/ndt/
Last visited: 22.05.2009.

[13] Cisco. Voice Over IP - Per Call Bandwidth Consumption. Internet, May 2005.
http://www.cisco.com/application/pdf/paws/7934/bwidth_consume.pdf
Last visited: 02.05.2009.

[14] Bandwidth Controller. Traffic Shaper XP. Internet, 2007.
http://bandwidthcontroller.com/
Last visited: 24.03.2009.

[15] D.Antoniades, M.Athanatos, A.Papadogiannakis, E.P.Markatos, and C. Dovrolis. Available bandwidth measurement as simple as running wget. *Passive and Active Measurements (PAM) conference*, 2006.

[16] Constantine Dovrolis. Pathload tutorial. Internet, April 2003.
http://www.cc.gatech.edu/fac/Constantinos.Dovrolis/bw-est/pathload_tutorial.html
Last visited: 22.05.2009.

[17] Constantinos Dovrolis, Parameswaran Ramanathan, and David Moore. What do packet dispersion techniques measure. In *In Proceedings of IEEE INFOCOM*, pages 905–914, 2001.

[18] Allen B. Downey. Using pathchar to estimate Internet link characteristics. In *SIGCOMM '99: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 241–250, New York, NY, USA, 1999. ACM.

[19] Marius Aamodt Eriksen. Trickle. Internet, 2007.
monkey.org/~marius/trickle/
Last visited: 24.03.2009.

[20] Wu-chang Feng, Francis Chang, Wu-chi Feng, and Jonathan Walpole. A Traffic Characterization of Popular On-line Games. *IEEE/ACM Trans. Netw.*, 13(3):488–500, 2005.

[21] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC2616 - Hypertext Transfer Protocol – HTTP/1.1. Internet, 1999.
http://www.ietf.org/rfc/rfc2616.txt.

[22] Sally Floyd and Van Jacobson. Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM TRANSACTIONS ON NETWORKING. VOL I . NO 1., Aug 1993.

[23] Universitetets Senter for Informasjonsteknologi (USIT). ISPs at NIX. Internet, 2009.
`http://www.uio.no/nix/nix-ops.html`
Last visited: 16.04.2009.

[24] Wireshark Foundation. Wireshark. Internet, 2009.
`http://www.wireshark.org/`
Last visited: 12.05.2009.

[25] Grobe, K. and Elbers, J.-P. PON in adolescence: from TDMA to WDM-PON. *Communications Magazine, IEEE*, 46(1):26–34, January 2008.

[26] R. Frederick H. Schulzrinne, S. Casner and V. Jacobson. RFC1889 - RTP: A Transport Protocol for Real-Time Applications. Internet, January 1996.
`http://www.ietf.org/rfc/rfc1889.txt`.

[27] R. Frederick H. Schulzrinne, S. Casner and V. Jacobson. RFC3550 - RTP: A Transport Protocol for Real-Time Applications. Internet, July 2003.
`http://www.ietf.org/rfc/rfc3550.txt`.

[28] K. Harfoush, A. Bestavros, and J. Byers. Measuring Bottleneck Bandwidth of Targeted Path Segments. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 3:2079–2089 vol.3, March-3 April 2003.

[29] Ben Huang. Hpcbench - High Performance Networks Benchmarking. Internet, 2009.
`http://hpcbench.sourceforge.net/index.html`
Last visited: 24.04.2009.

[30] Bert Hubert. The Wonder Shaper. Internet, 2002.
`lartc.org/wondershaper/`
Last visited: 24.03.2009.

[31] David S. Isenberg. The Dawn of the "Stupid Network". *netWorker*, 2(1):24–31, 1998.

[32] V. Jacobson. pathchar: A Tool to Infer Characteristics of Internet Paths. Internet, April 1997.
`ftp://ftp.ee.lbl.gov/pathchar/`.

[33] Manish Jain and Constantinos Dovrolis. End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput. *SIGCOMM Comput. Commun. Rev.*, 32(4):295–308, 2002.

[34] Artur Janc. Network Performance Evaluation within the Web Browser Sandbox. Master's thesis, WORCESTER POLYTECHNIC INSTITUTE, Worcester, MA, USA, January 2009.
`www.wpi.edu/Pubs/ETD/Available/etd-011909-150148/unrestricted/artur-janc-msc-thesis.pdf`.

[35] Rick Jones. Netperf Manual. Internet, 1995.
http://www.netperf.org/svn/netperf2/tags/netperf-2.4.3/doc/netperf.html
Last visited: 22.05.2009.

[36] Measurement Lab. Measurement lab. Internet, 2009.
http://www.measurementlab.net
Last visited: 22.05.2009.

[37] Hagel Technologies Ltd. DU Meter 4.01. Internet, 2009.
http://www.dumeter.com/
Last visited: 18.05.2009.

[38] Trier M. Highspeed-Internet. In *GameStar (Gaming Magazine)*, pages 164–165, March 2002.

[39] V. Paxson M. Allman and W. Stevens. RFC2581 - TCP Congestion Control. Internet, April 1999.
http://www.ietf.org/rfc/rfc2581.txt.

[40] M. Mathis and M. Allman. RFC3148 - A Framework for Defining Empirical Bulk Transfer Capacity Metrics. Internet, July 2001.
http://www.ietf.org/rfc/rfc3148.txt.

[41] Terrence P. McGarty. Peering, Transit, Interconnection: Internet Access in Central Europe. Internet, 2002.
http://hdl.handle.net/1721.1/1495
Last visited: 22.05.2009.

[42] John Nagle. RFC896 - Congestion Control in IP/TCP Internetworks. Internet, January 1984.
http://tools.ietf.org/html/rfc896.

[43] NextGenTel. Hva er WiMAX. Internet, 2008.
http://wimax.no/hvaerwimax/
Last visited: 30.03.2009.

[44] Peter Steenkiste Ningning Hu. Evaluation and Characterization of Available Bandwidth Probing Techniques. IEEE Journal On Selected Areas In Communications, VOL. 21, NO. 6, AUGUST 2003.

[45] NIX. Nix Statistikk. Internet, 2009.
http://mrtg.uio.no/mrtg/nix/
Last visited: 28.04.2009.

[46] NPT. Om nettnøytralitet - Prinsipper for nøytralitet på Internett Versjon 1.0. Internet, 2008.
http://www.npt.no/iKnowBase/Content/107040/Nettnoytralitet_prinsippnotat.pdf
Last visited: 11.03.2009.

[47] NPT. Nettnøytralitet - Retningslinjer for nøytralitet på Internett Version 1.0. Internet, 2009.
http://www.npt.no/iKnowBase/Content/109607/Retningslinjer_for_

nettnoytralitet.pdf
Last visited: 11.03.2009.

[48] NPT. Post- og teletilsynet. Internet, 2009.
http://www.npt.no/
Last visited: 11.03.2009.

[49] NPT. Post- og teletilsynet. Internet, 2009.
http://www.npt.no/portal/page/portal/PG_NPT_NO_EN/PAG_NPT_EN_HOME/PAG_MAIN_TEXT?p_d_i=-121&p_d_c=&p_d_v=47781
Last visited: 08.05.2009.

[50] Post og teletilsynet. Telepriser.no - Om telepriser. Internet, 2009.
http://www.telepriser.no
Last visited: 04.05.2009.

[51] Kjetil Olsen. NIX forside. Internet, 2007.
http://www.uio.no/nix/
Last visited: 16.04.2009.

[52] Anders Olsson. *Understanding Changing Telecommunications*. John Wiley and Sons, Ltd, 2003.

[53] Inc OnLive. OnLive. Internet, April 2009.
http://www.onlive.com/index.html
Last visited: 24.04.2009.

[54] Ookla. Speedtest. Internet, 2008.
http://speedtest.ookla.com/docs/usage.html
Last visited: 20.05.2009.

[55] Ookla. Ookla Inc. Internet, 2009.
http://www.ookla.com
Last visited: 24.04.2009.

[56] J. Postel. RFC768 - User Datagram Protocol. Internet, August 1980.
http://www.ietf.org/rfc/rfc0768.txt.

[57] J. Postel. RFC791 - Internet Protocol. Internet, September 1981.
http://www.ietf.org/rfc/rfc0791.txt.

[58] J. Postel. RFC792 - Internet Control Message Protocol. Internet, September 1981.
http://www.ietf.org/rfc/rfc0792.txt.

[59] J. Postel. RFC793 - Transmission Control Protocol. Internet, September 1981.
http://www.ietf.org/rfc/rfc0793.txt.

[60] Mark Pozzobon. Quake 3 Packet and Traffic Characteristics. Internet, December 2002.
http://caia.swin.edu.au/genius/021220A/
Last visited: 02.05.2009.

[61] R. S. Prasad, M. Murray, C. Dovrolis, and K. Claffy. Bandwidth estimation: metrics, measurement techniques, and tools. IEEE NETWORK, VOL 17; PART 6, pages 27-35, 2003.

[62] John Resig. Accuracy of JavaScript Time. Internet, 2008.
`http://ejohn.org/blog/accuracy-of-javascript-time/`
Last visited: 16.04.2009.

[63] Andrew Rolling. *Fundamentals of Game Design*. Prentice Hall, 2006.

[64] .SE. Bredbandskollen TPTEST. Internet, January, 2009.
`http://www.bredbandskollen.se/about.html`
Last visited: 22.05.2009.

[65] Locktime Software. NetLimiter - Ultimate Bandwidth Shaper. Internet, 2008.
`http://netlimiter.com/`
Last visited: 12.05.2009.

[66] Statistisk Sentralbyrå. Andel med ulike typer Internett-abonnement, etter husholdningstype, husholdningsinntekt, kjønn, alder, utdanning og arbeidssituasjon. 2. kvartal 2008. Prosent. Internet, 2009.
`http://www.ssb.no/emner/10/03/ikthus/tab-2008-09-18-02.html`
Last visited: 26.03.2009.

[67] Tim Szigeti and Christina Hattingh. Quality of Service Design Overview. Internet, December 2004.
`http://www.ciscopress.com/articles/article.asp?p=357102`
Last visited: 03.05.2009.

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

[69] Trådløse Trondheim. Trådløse Trondheim nettside. Internet, 2009.
`http://tradlosetrondheim.no`
Last visited: 22.05.2009.

[70] TV2. Hvorfor får jeg bare se video i 590 kbps? Internet, 2009.
`http://webtvsupport.tv2.no/index.php?_m=knowledgebase&_a=viewarticle&kbarticleid=36&nav=0`
Last visited: 09.05.2009.

[71] A. Tyagi, J.K. Muppala, and H. De Meer. VoIP support on Differentiated Services using Expedited Forwarding. *Performance, Computing, and Communications Conference, 2000. IPCCC '00. Conference Proceeding of the IEEE International*, pages 574–580, Feb 2000.

[72] Visualware. MySpeed Quality of Service. Internet, 2007.
`http://www.myspeed.com/whitepapers/qos.html`
Last visited: 22.05.2009.

[73] Visualware. Test My Connection Results. Internet, 2008.
`http://myspeed.visualware.com/resultsinfo.html`
Last visited: 22.05.2009.

[74] Wikipedia. Autonomous system (Internet). Internet, 2009.
http://en.wikipedia.org/w/index.php?title=Autonomous_system_(Internet)&oldid=280806795
Last visited: 31.03.2009.

[75] Wikipedia. BitTorrent (protocol). Internet, 2009.
http://en.wikipedia.org/wiki/BitTorrent_(protocol)
Last visited: 19.02.2009.

[76] Wikipedia. File transfer. Internet, 2009.
http://en.wikipedia.org/w/index.php?title=File_transfer&oldid=281715175
Last visited: 23.04.2009.

[77] Wikipedia. Hypertext Transfer Protocol. Internet, 2009.
http://en.wikipedia.org/w/index.php?title=Hypertext_Transfer_Protocol&oldid=287482253
Last visited: 03.05.2009.

[78] Wikipedia. Internet Protocol. Internet, 2009.
http://en.wikipedia.org/w/index.php?title=Internet_Protocol&oldid=276704784
Last visited: 16.03.2009.

[79] Wikipedia. Peering. Internet, 2009.
http://en.wikipedia.org/w/index.php?title=Peering&oldid=274727871
Last visited: 12.03.2009.

[80] Wikipedia. Remote desktop software. Internet, 2009.
http://en.wikipedia.org/w/index.php?title=Remote_desktop_software&oldid=284758227
Last visited: 24.04.2009.

[81] Wikipedia. Random Early Detection. Internet, March 2009.
http://en.wikipedia.org/w/index.php?title=Random_early_detection&oldid=275962194
Last visited: 24.03.2009.

[82] T. Ylonen and C. Lonvick. RFC4252 - The Secure Shell (SSH) Authentication Protocol. Internet, January 2006.
http://www.ietf.org/rfc/rfc4252.txt.

[83] Sebastian Zander and Grenville Armitage. Empirically Measuring the QoS Sensitivity of Interactive Online Game Players. Internet, 2004.
http://caia.swin.edu.au/pubs/ATNAC04/zander-armitage-ATNAC2004.pdf
Last visited: 02.05.2009.

[84] Liren Zhang, Li Zheng, and Koh Soo Ngee. Effect of delay and delay jitter on voice/video over IP. *Computer Communications*, 25(9):863 – 873, 2002.

Appendix A

Server Scripts

A.1 Bridge Script

The bash script we used to set up our Linux Server as a bridge is shown below. The script requires two interfaces (eth0 and eth1) and will enable a bridge over these two interfaces.

```
#!/bin/bash

# Enable both interfaces
ifconfig eth0 0.0.0.0 up
ifconfig eth1 0.0.0.0 up

# Add them to the bridge
brctl addbr br0
brctl addif br0 eth0
brctl addif br0 eth1

# Enable the bridge
ifconfig br0 up

# Get an ip-address for the bridge
dhclient3 br0
```

A.2 Rate Limit Script

The bash script we used to set up the rate limit for our Linux bridge is shown below. The scripts assumes that the bridge is set up and between interface *eth0* and *eth1*. The script takes to two input parameters, both should be integers and represents the limit rate in kbit/s.

```
#!/bin/bash
# Lars-Ivar's bandwidth limit script
#######################################

# Input params
download=$1"kbit"
upload=$2"kbit"
a=$1
ceil=$((a+0))"kbit"

# Calculate RED-values
bw=$1*1000
red_max=$((bw/8*1/4))
red_min=$((red_max/3))
red_limit=$((red_max*8))
red_avpkt=1000
red_burst=$(((2*red_min+red_max)/(3*red_avpkt)))
red_prb_down=0.01
red_prb_up=0.01
red_bw=$1

# REMOVE previous values
tc qdisc del dev eth0 root 2> /dev/null > /dev/null
tc qdisc del dev eth1 root 2> /dev/null > /dev/null

# ADD HTB shaping and queue disipline on download traffic
tc qdisc add dev eth1 root handle 1: htb default 20
tc class add dev eth1 parent 1:1 classid 1:20 htb rate
  $download ceil $ceil prio 0
tc qdisc add dev eth1 parent 1:20 handle 20: red limit
  $red_limit min $red_min max $red_max burst $red_burst
  avpkt $red_avpkt bandwidth $red_bw probability
  $red_prb_down

# ADD HTB shaping and queue disipline on upload traffic
tc qdisc add dev eth0 root handle 1: htb default 20
tc class add dev eth0 parent 1:1 classid 1:20 htb rate
  $upload ceil $ceil prio 0
tc qdisc add dev eth0 parent 1:20 handle 20: red limit
  $red_limit min $red_min max $red_max burst $red_burst
  avpkt $red_avpkt bandwidth $red_bw probability
  $red_prb_up
```

Appendix B

Electronic Attachment

B.1 Evaluation of Existing Tools - Test Results

Complete test plan, results and associated graphs are included in the Microsoft Office Excel 2007 spreadsheet file `complete_test_results.xlsx`.

Appendix C

Glasnost Test Results

65 % Reduction - Regular traffic 2300 kbit/s. BitTorrent ports limited at 800 kbit/s.

TCP Transfer	BitTorrent port	Non-BitTorrent port	Conclusion
Download #0	780 kbps	2014 kbps	Potential rate limiting
Download #1	780 kbps	1882 kbps	Potential rate limiting
Upload #0	2096 kbps	2258 kbps	No rate limiting
Upload #1	1828 kbps	2246 kbps	No rate limiting

56 % Reduction - Regular traffic 2300 kbit/s. BitTorrent ports limited at 1000 kbit/s.

TCP Transfer	BitTorrent port	Non-BitTorrent port	Conclusion
Download #0	909 kbps	2097 kbps	Potential rate limiting
Download #1	915 kbps	2162 kbps	Potential rate limiting
Upload #0	2191 kbps	2157 kbps	No rate limiting
Upload #1	2212 kbps	2166 kbps	No rate limiting

50 % Reduction - Regular traffic 2000 kbit/s. BitTorrent ports limited at 1000 kbit/s.

TCP Transfer	BitTorrent port	Non-BitTorrent port	Conclusion
Download #0	919 kbps	1658 kbps	No rate limiting
Download #1	921 kbps	1681 kbps	No rate limiting
Upload #0	1780 kbps	1871 kbps	No rate limiting
Upload #1	1864 kbps	1857 kbps	No rate limiting

33 % Reduction - Regular traffic 1500 kbit/s. BitTorrent ports limited at 1000 kbit/s.

TCP Transfer	BitTorrent port	Non-BitTorrent port	Conclusion
Download #0	919 kbps	1352 kbps	No rate limiting
Download #1	921 kbps	1348 kbps	No rate limiting
Upload #0	1352 kbps	1357 kbps	No rate limiting
Upload #1	1358 kbps	1346 kbps	No rate limiting

Figure C.1: Glasnost test results.

Appendix D

NTNU Upload Rate Issues

During our testing we discovered a strange phenomenon regarding the upload rates from NTNU. What we found is that a Windows user is only able to get an upload rate of about 14 Mbit/s per TCP connection from NTNU to Oslo. If the user establish two connections he would get about 28 Mbit/s etc. We tested multiple servers located at various places in Oslo, and multiple locations at NTNU (including Realfag, Stripa, Hovedbygget and EL-Bygget) all showing the same results.

Within the NTNU network all Windows clients was able to obtain full speed for a single TCP connection.

Linux on the other hand is able to get 100 Mbit/s on a single TCP connection from NTNU to the same servers in Oslo. But if we booted the same computer in Windows, with the same hardware specifications, the rate was again limited to 14 Mbit/s.

We reported our findings to Vidar Stokke at ITEA Systemdrift. He was not able to explain our findings, but promised to look into it. Hopefully have we discovered some kind of misconfiguration somewhere and this can be corrected, improving the NTNU network!

Appendix E

Tools Used in Analysis

E.1 Wireshark

Wireshark is the world's foremost network protocol analyzer, and is the de facto standard across many industries and educational institutions [24].

We have used Wireshark to analyze the traffic sent across the network interface when running the different bandwidth measurement tools. In Wireshark it is possible to analyze the traffic on different protocol levels and show the content of the packets being transferred. It also possible to filter out traffic on specific IP addresses or protocols. This is of great value when evaluating the functionality of the bandwidth tools. For more information on Wireshark, see [24].

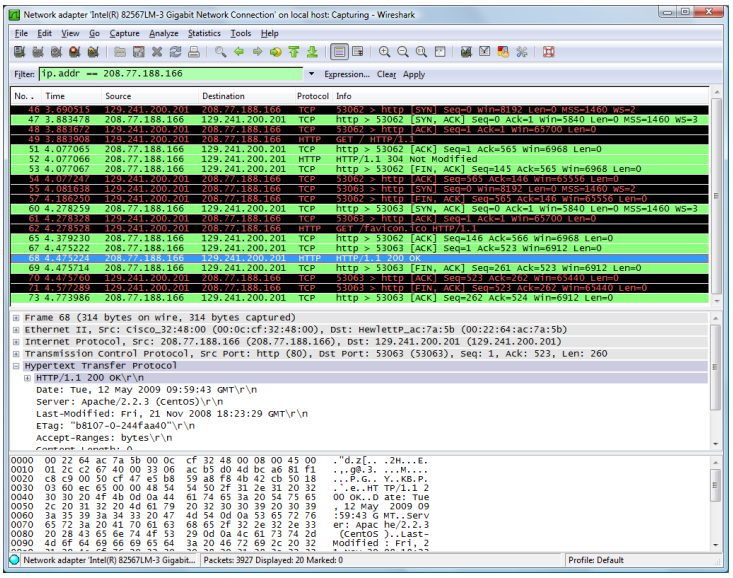


Figure E.1: Screen shot of Wireshark when monitoring a HTTP request performed against www.example.com

E.2 NetLimiter

NetLimiter is an Internet traffic control and monitoring tool. You can use NetLimiter to set download/upload transfer rate limits per applications or even per connection and monitor their Internet traffic [65].

NetLimiter is a traffic shaper that operates on the application level. As explained in section 7.1 this type of traffic shaping did not satisfy our needs in the testing of the bandwidth measurement tools. But the monitoring part of NetLimiter is useful to keep an eye on the Internet traffic of the network interface. This enables us to eliminate the tests where other processes use the Internet connection to transfer data, e.g. Windows searching for new updates.

E.3 DU Meter

Du Meter is a passive measurement tool which presents the data rate coming in and out of the network interface. The tool collect traffic information and the gathered information can be presented as various reports. More information about the tool is found in [37].