ADDIS ABABA UNIVERSITY
COLLEGE OF NATURAL SCIENCES
School OF INFORMATION SCIENCE

Developing WAN Optimization Model to improve the Performance of Business Critical Applications: The Case of UNECA

Performance benchmarking of critical applications is done to determine the impact level of Various WAN factors (bottlenecks), which is used for developing WAN Optimization Model to alleviate the impact of these factors accordingly.

A THESIS

SUBMITTED TO SCHOOL OF GRADUATE STUDIES OF ADDIS ABABA UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN INFORMATION SCIENCE.

By
Yacob Gobena

Advisor
Workshet Lamenew

JUNE 2013

ADDIS ABABA UNIVERSITY
COLLEGE OF NATURAL SCIENCES
School OF INFORMATION SCIENCE

Developing WAN Optimization Model to improve the Performance of Business Critical Applications: The Case of UNECA

By

Yacob Gobena

Approved by the Examining Board

_____________________________ __________
Chairperson, Examining Committee Signature

_____________________________ __________
Advisor Signature

_____________________________ __________
Examiner Signature

DECLARATION

The substance of this thesis is an original work of the author and due reverence and acknowledgement has been made, where necessary, to the work of others, no part of the thesis has already been accepted for any degree, and it is not being currently submitted in the candidature of any degree.

Yacob Gobena

ACKNOWLEGNEMNT

First and foremost, I praise God, the almighty, for giving me the necessary strength to complete this work.

My sincerest gratitude then goes to my advisor Workshet Lamenew for his constructive comments, overall guidance and commitment through the learning process of this master thesis. I attribute the level of my master's degree to his encouragement and effort; without him, this thesis would not have been completed or written. One simply could not wish for a better or friendlier advisor.

Many individuals have contributed support, idea expertise, insight and time to make this study possible. With this regard, I would like to thank my family, colleagues of my current job and friends for their support. Thank you all for your support. You will never know the depth of your influence on me and I will never forget your sacrifice to make this educational experience possible.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS and ACRONYMS

ACK	Acknowledgement
APP	Application
ATM	Asynchronous Transfer Mode
CIFS	Common Internet File System
CoS	Class of Service
CWND	Congestion Window
DifServ	Differentiated Services
DNS	Domain Name System
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HQ	Headquarters
IntServ	Integrated Services
IP	Internet Protocol
ITU	International Telecommunication Union
ISO	International Standard Organization
LAN	Local Area Network
LFN	Long Fat Network
MAC	Media Access Control
MAPI	Messaging Application Programming Interface
MPLS	Multi-Protocol Label Switching
MRTG	Multi Router Traffic Grapher
NFS	Network File System
OSI	Open System Interconnection
POP3	Post Office Protocol version 3
QoS	Quality of Service
RFC	Request for Comments
RWND	Receive Window
RPC	Remote Procedure Call
RTT	Round Trip Time
SDH	Synchronous Digital Hierarchy
SLB	Server Load Balancer

SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SONET	Synchronous Optical Network
SRO	Sub Regional Office
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VoIP	Voice over IP
VPN	Virtual Private Network
WAAS	Wide Area Application Services
WAFS	Wide Area File Services
WAN	Wide Area Network
WCCP	Web Cache Control Protocol
WLAN	Wireless Local Area Network

LIST OF APPENDICES

Key Terms used interchangeably

Network Issues, WAN factors, WAN conditions, and characteristics of WAN are terms used interchangeably in this report to describe the same networking concept.

ABSTRACT

Wide Area Networks (WANs) have natural characteristics such as high latency, congestion, low bandwidth and high packet loss rate. Due to these factors, the performance of enterprise applications can quickly become degraded when they are being accessed and used over the WAN link. In addition, most of the existing protocols are designed for Local Area Network (LAN) environments; therefore, several application protocols do not perform well over the WAN.

The purpose of this study is to investigate the impact of various WAN factors, with a view to develop a WAN optimization model that can improve the performance of business-critical applications over the WAN. Previous studies reported that an application optimization solution would be economically feasible and effective, if it is designed from the targeted organization WAN environment perspective. Hence, this study attempts to develop a WAN optimization solution particularly targeted for enterprise WAN environments, which have limited network resources.

To realize the goal of the study, various experiments have been undertaken in cases that are taken from the UNECA WAN environment. The experimentation is conducted through four major phases: in the first phase, application's traffic flow data is collected using Net-Flow analyzer tool from the UNECA WAN environment. Next, the collected traffic flow data is simulated and experimental environment is prepared using the OPNET Modeler tool. The third phase deals with analysis of the simulated traffic data for investigating critical application performance challenges, followed by experimenting different optimization methods, so that the appropriate technique is selected for each performance bottlenecks. Finally, based on the analysis result a WAN optimization Model is developed.

The developed model is designed to identify the real-time bottleneck status of each WAN link and apply optimization techniques accordingly, which are appropriate for improving the performance of each specific application traffic flow. This study suggests further research works for enhancing the efficiency and effectiveness of the Model.

CHAPTER ONE
INTRODUCTION

1.1 Background

During recent years, organizations tend to deploy their IT infrastructure over a WAN to increase productivity, support global collaboration, enhance geographical flexibility (with branch offices) and minimize costs, thus constituting to today's WAN centered environments. The public as well as the private companies are establishing readily deployable branch offices for providing 24/7 customer services to the clients. The corporate head offices as well as the branch offices have to share voice, data and video information over a geographically dispersed locations making use of the WAN technologies (Zhang, Ansari, & Wu, 2011).

Kansanen (2009) described that from the network perspective organizations have at least three different kinds of offices inside their network: branch offices, regional offices and data centers. Many companies prefer to have the employees, as close to the customer as possible and therefore the number of small branch offices within one corporation can be high.

Largest part of the network cost in such organizations comes from the servers, deployed infrastructure and the management of the servers in these branch offices. Only from the performance perspective, it is ideal for the branch office workforce to have a local server infrastructure, for enterprise applications. To reduce the costs and ease the management many companies have moved these local services into their data centers. Due to this consolidation of services, the branch office users have to access resources from bigger regional office or data center through WAN links. The large amount of small offices and at the same time the centralization of servers into data centers have made the WAN network a critical point for many organizations (Grevers & Christner, 2007).

Cisco (2008) states that when enterprise applications deployed in WAN environments, due to some network factors that are not common in LAN environments, their performance can quickly become degraded. Kansanen (2009) attempted in his research to describe the effects of these factors: as compared to LANs the available bandwidth in WANs is rather limited, which directly affects the application throughput over a WAN. Other obvious barriers in WANs are the high latency introduced by long transmission distance, protocol translation, and network congestions. The high latency in a WAN is a major factor causing the long application response time. Congestion causes packet loss and retransmissions, and leads to erratic behavior of the transport layer protocol, such as Transmission Control Protocol (TCP).

Moreover, most of the existing protocols are designed for LAN environments; therefore, applications that use these protocols do not perform well in WAN environments. The common practice of application developers is designing applications in corporate headquarters (HQ), where gigabit (1000 MB) LANs are the norm, while the typical branch office is often connected with T1 speed (1.5 MB) or lower. This difference creates a massive performance discrepancy in user experience; particularly branch/remote office users suffer the most from poor performance experience (IXIA, 2013).

A research survey (as cited in Chou, 2009) indicates that the mobile workforce, those who do not have an allocated corporate office space at all, and branch office workforce will make up more than 30 percent of the total global workforce by 2014. Briefly, the survey result shows that to enhance the productivity of these two workforce categories, there is a need to mitigate the application performance challenges in organizations' WAN environment.

With the proliferation of enterprise applications and increased complexity of systems and networks, it can be difficult to know how an application will perform on any WAN environment, and to identify the actual cause of performance issues when they arise. Since enterprise applications are rarely tested under WAN conditions (like bandwidth limitation, latency, throughput, congestion and packet loss), expectations of performance are often at variance to actual performance (Kerravala, 2009).

Cole & Ramaswamy (2000) discusses in their book how studying application performance over the WAN is so important for organizations. Businesses/Organizations needed to overcome the performance problems associated with the regionalization and globalization of their workforce and customer base. Users everywhere needed data and application access, but the performance of their existing applications over the WAN, is far from being LAN-like performance of applications around the world. This is particularly because, not enough attention has been paid for analyzing how applications actually perform over a WAN and on setting user expectation. Therefore, it is important to study WAN traffic characteristics of the client / server environment to cope with enterprise applications' performance issues.

This thesis attempts to propose an optimization model for improving the performance of enterprise applications, when used over the WAN by branch office workforces. The optimization model is proposed as a solution particularly for those enterprise WANs, which are having various network issues due to their limited network resource.

Christner, Seils, and Jin (2010) asserts that before examining how to improve application performance challenges caused by WAN factors, it is important to deal with vital issues like identifying factors, which are the major source of performance problems for that specific WAN environment, and determining the performance reaction of each enterprise application for these factors. Thus, this study undertakes experiments to evaluate the impact of network issues including bandwidth limitation, latency, throughput, congestion and packet loss on the performance of various business critical applications. These evaluation results are then used to develop the proposed optimization solution so that it will be effective to mitigate the impact of the analyzed network issues on the performance of WAN enterprise applications.

1.1.1 Motivation for the Research

Riverbed (2013) reported that the performance problem of applications would be even worse, when the enterprise WANs are not well equipped with the required network resources to run the applications. When network resources (like bandwidth, powerful WAN devices) are limited, the impact of WAN network factors including, latency, throughput, congestion and packet loss will be more significant. Hence, the study recommends that solving application performance issues should be given much attention particularly for WAN environments, which lacks the required network resources.

Clearly many organizations in developing countries including Ethiopia do not have the required financial power to supplement their WAN environment with resources (like bandwidth and powerful WAN devices). Based on the Riverbed (2013) report, it is logical to assume that enterprise applications' performance problem is more of our organizations' issues than developed countries. Thus, considering the limitation of the required WAN resources for organizations in our country, "investigating and improving the performance problem of enterprise applications when used through the WAN" is worth studying. Although, the scope of this thesis is limited to cases that are taken from UNECA WAN environment, the researcher is hopeful that this paper will serve as a valuable vehicle for further researches in other organizations.

Li et al. (2010) reported that identifying the source of performance problems before deploying any optimization solution is an effective approach to solve application performance problems. Even for a known common source of problems, their level of impact should be studied from the specific organization's perspective to deliver a better and appropriate solution. If we take, for instance, one network issue "congestion" its impact is quite different for a high bandwidth and low bandwidth WAN environment and similarly the technique applied to control its impact should be different. For these reasons, it is advisable to have an optimization solution, which is designed based on the experimental result that evaluates the impacts of various WAN conditions on the low resourced network environment.

Moreover, the researcher believes that an application-based solution, which applies an optimization technique based on the application type (applications reaction to real-time network situations), is an effective solution for organizations' WAN. This is because applying specific optimization technique that is designed for each application behavior provides better performance result. Such solution will also be suitable for new coming applications, as it can easily integrate a solution module that is designed for the new application behavior.

Applications today are becoming increasingly robust and complex compared to applications of ten years ago, making them more sensitive to network conditions, and it is certain that this trend will continue (Christner, Seils, & Jin, 2010). This issue obviously requires a corresponding performance improvement research attempts as the applications are becoming more complex and sensitive to network conditions. Chrisner et al. (2010) reported that the research domain that studies the application performance improvement over the WAN or over the Internet is very open and requires consistent research attempts as long as new applications are coming for use or existing applications are getting robust or complex.

The researcher of this study is inspired to work in this domain area, not only because of the recommendations and reports provided by various researches that are discussed above, but also because of the practical WAN situations observed in UNECA WAN environment as explained below.

1.1.2 UNECA WAN Scenario

United Nations Economic Commission for Africa (UNECA) is established by the Economic and Social Council (ECOSOC) of the United Nations (UN) in 1958 as one of the UN's five regional commissions. UNECA serves 54 Member States, which are organized in five sub regions: North Africa, West Africa, Central Africa, East Africa and Southern Africa. To serve its African Member States in a better way, the Economic Commission for Africa has opened, in 1963, Sub Regional offices (SROs), a representation in each one of the five sub regions.

ECA SROs develop and maintain sub-regional repositories of statistical information that feed into the common data bank and support analytical and research needs of the Commission.

The problem domain that is the performance degradation of critical applications over the WAN will be examined further by practical situations on UNECA WAN. The provided incident facts are a result of the researcher preliminary survey using Net-Flow analyzer monitoring tool and end users experience report on the presence and status of the problem (the preliminary result can be found in chapter 3 and Appendices).

Applications Performance Problem Overview

As stated earlier, ECA SROs produce repositories of statistical information and feed this information into a common data bank. In addition, to carry out the ECA’s mandate, the SRO employees use electronic resources located in HQs data center as well as collaborate with colleagues from other SROs using various media. To accomplish these tasks, WAN is deployed which connects the five SROs to UNECA HQ in Addis Ababa.

Different services and applications are consolidated in the HQ data center. And hence, the WAN is being used intensively by different services and applications: like Lotus Notes for email communications, Citrix for Finance and HR application, Video conferencing, Voice data for telephone, Active Directory replication, FTP for file transfer, Remote Desktop Protocol for monitoring servers, HTTP for Intranet, electronic applications (e-leave, e-billing and e-clearance) and many other services.

End users in SROs face intermittent application/service performance problem, which affects the operations of the organization. The problems include email delivery delay, finance/HR application is slow or unresponsive, telephone voice is not clear, during Video conferencing picture or voice is not clear, unable to transfer files using FTP and many other application performance problems. Furthermore, there is a near-future plan to consolidate data backup of all remote offices to the HQ that adds a substantial amount of traffic through the WAN and an ongoing project of implementing Active Directory, which is another source of WAN replication traffic. The addition of these new planned applications will make things worse in terms of performance, since it adds more traffics over each WAN link.

UNECA tried to add bandwidth to address the performance challenges and no noticeable performance change has been observed. In addition, the problem is intermittent and has no clear pattern. To mention a few incidents (supported with traffic data in section 3.2.1 and Appendices), one application performs well in SRO-CA and another application not in that same SRO: and the application that performs poorly in SRO-CA gives best response time for SRO-EA but the other application which was good in SRO-CA will be poor in SRO-EA. In general, applications perform differently in different SROs, even for SROs with similar Bandwidth.

The UNECA's WAN scenario is one of the main motivation factors for conducting this research: to find an answer for issues like what WAN conditions each critical application demands and how is it possible to design a solution that can fulfil each application's demand.

Proposed Optimization Solution

Various researches have been conducted to solve the application performance problems for those applications that are being used over the Internet and few other enterprise WANs. However as per Li et al. (2010) recommendation, the researcher of this study attempts to develop a WAN optimization solution that is entirely based on the experimentally identified source of performance problems that exist in the target organization WAN. The optimization solution is designed particularly for WANs with low infrastructure resources (in terms of Bandwidth, Topology, WAN devices and others). UNECA WAN environment, where cases are taken for this study's experiment, is not at its best in terms of network infrastructure resource. Because, the five SROs of UNECA are located in Africa and the resource limitation on these poor African countries' telecommunication infrastructure is directly reflected on UNECA WAN environment. The WAN optimization Model is also envisioned to apply a performance improvement technique based on the applications real time behavior revealed on the enterprise WAN environments.

To develop the WAN optimization model, the researcher of this study applies the basic concept "you cannot improve what you cannot measure and understand". Thus, first the target organization WAN (UNECA WAN environment) is simulated using the OPNET modeler tool. Various WAN traffic scenarios are also simulated using the tool so that experimentation is conducted to evaluate the impact level of WAN conditions on applications' performance and to analyze how applications react to these conditions. Optimization techniques are also being experimented in each scenario with OPNET Modeler tool and the result of this experiment is used to design and develop the optimization model that is capable of improving critical application's performance.

1.2 Statement of the problem

Applications, deployed over a WAN suffer performance degradation owing to natural characteristics of WANs such as high latency, congestion and high packet loss rate. Most of the existing protocols are not designed for WAN environments; therefore, several applications do not perform well under the WAN condition (Sevcik & Wetzel, 2008).

Aberdeen Group (2009) reported that with more branch office, remote and mobile users dependent on the WAN and with more new technologies and applications rollouts, managing the performance of enterprise applications become very difficult. New technology rollouts, convergence of voice and data networks, and multiple bandwidth intensive applications are driving organizations to add more bandwidth to support the role of their network in achieving strategic organizational goals. However, Riverbed (2013) argued that adding more bandwidth is a Shortsighted Tactic for addressing poor application performance problem. Because throwing more bandwidth may mask the worst symptoms of a poorly performing network for a while, but it does nothing to treat the underlying problem itself.

WAN optimization solution providers have options to improve the performance of enterprise applications. These solutions include various techniques for accelerating WAN traffics such as byte caching, file caching, application-specific compression tools, TCP acceleration tools (Xu, Hu, & Bhuyan, 2004). However Li et al. (2010) argued that it is economically infeasible and ineffective to apply any of the above optimization techniques without identifying the root cause of the problem. Moreover, Zhang et al. (2011) findings suggest that since many new challenges to content delivery over WANs are emerging and the scale of information data are growing rapidly, there is a need for a continuous study to improve applications performance over the WAN. Due to globalization, organizations located in Ethiopia also suffer with these new challenges thus, researches aimed at finding solutions for these performance challenges are essential.

Considering the Bandwidth limitation and cost, as well as the low resourced network infrastructure in our country, "investigating and improving the performance problem of critical applications when used through the WAN" is worth studying.

Thus, this study will answer the following Research questions:

- How does the enterprise WAN network performance bottleneck manifest itself?
- What is the impact level of these WAN network performance bottlenecks on end user response time and
- How is it possible to benchmark critical application performance over the WAN, based on the impact of these WAN network performance bottlenecks?
- How is it possible to develop a WAN optimization model that can identify and improve the performance problem of WAN critical applications in real-time?

1.3 Objectives

1.3.1 General Objective

The objective of this research is to investigate the performance impact of various network factors, with a view to develop a WAN Optimization Model that can improve the performance of business-critical applications, which are deployed over a WAN.

1.3.2 Specific Objectives

The research is organized through different phases, each with clearly stated objective. To achieve the goal of this thesis study, the specific objectives, which are mentioned below, should be realized.

- To identify the applications and services used through the WAN with their respective protocol information and their criticality for the organization business.
- To determine the major performance bottlenecks that affects critical application performance.
- To investigate the impact level of these bottlenecks on each critical application performance.
- To benchmark the performance of business-critical applications in terms of major performance bottlenecks,
- Based on the benchmarking result, to develop an algorithm, which is capable of identifying the application performance bottleneck in real time and accordingly optimizes the application traffic over the WAN.

1.4 Methodology

To realize the specific objectives and achieve the goal of the study the researcher employed a methodology as described below and which is represented by the flow chart in Figure 1.1.

1.4.1 Review of Related Literature

Relevant literatures (books, journals, articles, conference papers, research reports and web documents) pertaining to the research under consideration have been reviewed. The researcher conducted a literature review to assess the major issues and concepts in the field of application performance problems, application's protocol behavior, WAN bottlenecks, WAN optimization, application acceleration, and measurement of application performance. In addition, documentations of network analysis and network modeling tools have been reviewed.

Researches done so far in the area of performance benchmarking and optimization for application deployed over enterprise WAN's is reviewed in particular. In general, other related researches done on applications deployed over other private or public networks, including the Internet and wireless network have been reviewed.

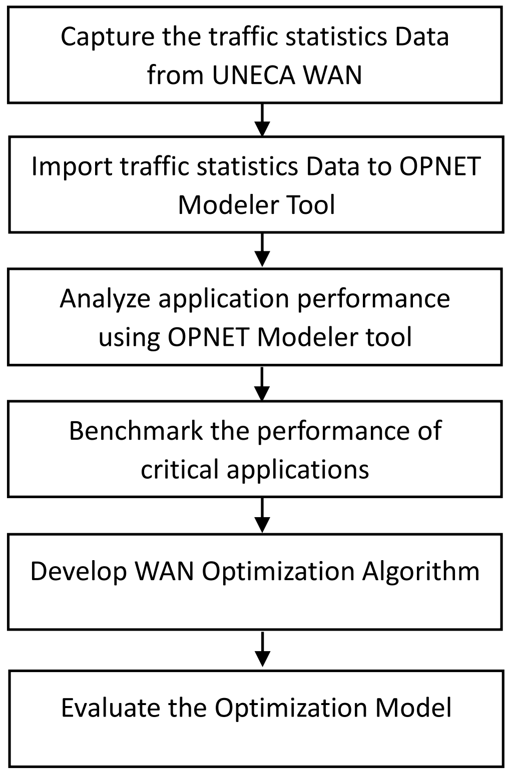


Figure 1.1 Workflow: for developing WAN Optimization Model (OPNET, 2008)

1.4.2 Data Collection and understanding

The empirical measurement of the target organization (UNECA) WAN is undertaken using Net-Flow analyzer to collect traffic statistics data. Data is collected about various features of the WAN, including WAN topology configuration, traffic flow statistics, various applications WAN usage statistics, end users experience report for all SRO WAN links. The traffic flow and application flow statistics data is collected for three months of period (January, February and March). These three months of period is selected because it can represent the 12-month traffic usage of UNECA WAN which are categorized into three levels of WAN usage. January represents the less number of traffic use because of the holiday season, February average number of traffic use and March high traffic usage since it is a beginning of the budget year.

Preliminary data analysis is also done to understand the collected data. Understanding helps to determine the major performance bottlenecks that influence application performances and to identify critical applications of the organization under the study.

Since the study focused on critical applications, selecting the list of critical applications to be experimented in this study is also prepared during this phase. The selection criterions are adopted from similar research attempts (Shunra, 2009) and it includes, how significant is the application for the business? How sensitive is the application protocol for WAN conditions? and how commonly the application is deployed over the WAN by organizations. High Priority is given to: 1) applications that support revenue-critical business processes, such as ERP, CRM systems and transaction processing 2) applications that are extensively used across the organizations' WAN, such as email, intranet portals and VoIP and 3) applications such as databases and "fat client" applications. In addition, real-time services such as VoIP and streaming video are given high priority.

1.4.3 Experimental Environment Setup (Modeling WAN)

The collected traffic data is used to prepare an experimental test bed by modeling the existing UNECA WAN environment using the OPNET Modeler tool. Modeling is done by importing actual WAN Network configuration and application traffic statistics data, which are collected in the previous phase. The modeled environment is the representation of UNECA WAN, since its topology, traffic flow pattern and application traffic usage is properly simulated and modeled by OPNET.

Traffic flow scenarios are also simulated and modeled from the actual traffic flow depending on the collected traffic statistic distribution. These models are used for further analysis to identify the behavior of critical applications and their source of performance problems. The scenarios are designed by changing, the link traffic load, type of application traffic using the link, link bandwidth, link delay value and by making one critical application traffic active at a time.

To check the model validity, average Response time of applications, throughput of WAN links and packet end-to-end delay measure is taken and evaluated.

1.4.4 Data Analysis

Experiments are conducted in the modeled environment and on the configured WAN scenarios so that the traffic flow is analyzed in terms of different performance metrics. The analysis is done using OPNET tool by examining the traffic flows in each scenario, which helps to determine the critical application's behavior in different scenarios and the performance impact of various WAN conditions on each critical applications. Traffic flow analysis helps to find a clear pattern in the performance behaviors of data, video and voice applications for different scenarios. Consequently, the pattern will be used for application performance benchmarking.

Typically, much of the analysis result data collected by OPNET Modeler for each scenario is placed in its output vector files. OPNET Modeler also provides a graphing and numerical processing environment in its Result Browser.

1.4.5 Application performance Benchmarking

Application performance benchmarking is a task of finding what WAN conditions each application demands to perform optimally over the WAN. Benchmarking requires the experimental result from data analysis phase, regarding each application. Based on the result, the performance of each critical application is benchmarked in terms of different WAN conditions.

1.4.6 Developing WAN Optimization Algorithm

This study is designed to adopt an approach, which uses the applications benchmarking task result in developing the optimization algorithms. Therefore, the critical application performance benchmarking result is used for developing the optimization algorithms. The develop algorithm is evaluated by its effectiveness on improving the performance of critical applications deployed over UNECA WAN environment.

1.4.7 Tools and Performance Metrics used

Net-Flow analyzer v9 is that is developed by Cisco is used in the data collection phase, to capture traffic statistics' data of the UNECA WAN. The traffic data provides information about the network utilization, the type of application using the network, application flows that exist between the network nodes, the top talkers (systems using the network) or top applications on the network and others.

OPNET Modeler is the essential tool used in almost all phases of this study. The OPNET Modeler environment incorporates tools, which are suitable for all phases of a study, including model design, simulation, data collection, and data analysis. For this study, it is used mainly for WAN model design, simulation and data analysis. ONET Modeler is used to create experimental test bed by simulating the traffic flow, application infrastructure and network topology of the enterprise WAN environment. Studying applications behavior and analyzing their performance in different factors/scenarios is also done using this tool.

Performance Metrics

To evaluate the effect of different factors on application response for WAN end users, different performance metrics are used by related research attempts. (Bai, Oladosu, & Williamson, 2007) Uses five metrics: loss rate, response time, retransmission rate, end-to-end delay and effective throughput for detailed performance measurement. Metrics like reply rate, throughput, response time, and error rate statistics are also used for a high-level over- view of the performance results.

OPNET modeler tool uses response time and packet end-to-end delay for analyzing applications performance. Therefore, this study uses response time as a primary performance metric for data applications (Email, Web and database applications) and packet end-to-end delay for multimedia applications (video conferencing applications and voice applications/telephone systems). Whenever necessary, other performance metrics are also used carefully without affecting the integrity of the evaluation result

1.5 Scope and Limitation of the Research

The WAN factors selected for experimentations are limited to the condition, which is mostly shown in under resourced corporate networks. These factors are also selected only from some layers of Open System Interconnect (OSI) Model.

The enterprise applications, which are experimented and analyzed in this study, are chosen based on their criticality level mainly considering the study target organization's business environment. The metrics used for evaluating application performance are restricted to response time and end to end delay measures due to time constraint.

The scope of this thesis limited to investigating the performance of only five branch offices of the UNECA WAN and the WAN nodes considered in the study is geographically limited to Africa.

1.6 Applications of Results and Beneficiaries

Organizations can use these findings to pinpoint potential areas of concern or WAN bottleneck and enhance their already deployed critical application performance. Moreover, when an organization plan to deploy new applications over the WAN, they have to know in advance at what WAN conditions this application performs best. One of the output of this research that determines the optimized conditions demanded by each application could be significant for the deployment.

Essentially, through further works, the optimization model framework developed in this study can be transformed into commercial application to be used by enterprise WANs. Mainly those enterprises' WAN with low network resources can be benefited from this optimization solution as it is designed to address the common performance issues of such WAN environments.

This research has a contribution not only on the optimization algorithm but also provides a methodological approach on how to study the performance problem of an enterprise WAN.

1.7 Thesis organization

The thesis report consists of six chapters. The first chapter deals with the general overview of the study including background, statement of the problem, objectives and methodology of the research.

In chapter 2, network performance is discussed from the WAN perspective concentrating on corporate networks. Factors affecting network performance are reviewed. The chapter also concentrates on the performance problems caused by TCP and application protocols used in corporate networks.

Chapter 3 is the empirical part of the work. It deals with the experimentations conducted on UNECA WAN to investigate the impact of different factors and evaluate various optimization techniques on critical application performance. First, the critical applications of the target network are defined. After that based on the outlined methodology the traffic measurement and analysis is undertaken.

Chapter 4 demonstrates the findings from the experimentation and discusses the result. The output of the experiment are also analyzed and interpreted in this chapter. Key issues, regarding the performance of critical applications are explained in terms various WAN factors as per the findings.

In chapter 5, benchmarking the performance of critical application in terms of WAN factors are explained, as the task is essential for developing the model. Then, discusses the WAN Optimization Model development process. The details of optimization model architectures, building blocks and algorithms are presented in this chapter.

Chapter 6 presents conclusions of the research and implications for future work in the area of WAN application performance improvement.

CHAPTER TWO
LITERATURE REVIEW

In this chapter, an attempt has been made to review literatures on the concepts of network architecture, different layer protocols and common WAN performance factors. The purpose of this chapter is to describe, networking technologies and several interacting protocols that are related to application performance issues and then give the readers theoretical backgrounds and concepts.

2.1 WAN Performance

Often the term performance in networks is used to describe the performance of applications and the impact on end users' experience (Kansanen, 2009). Likewise, in this thesis the network performance is studied from the application performance point of view. The focus is on the network performance in WAN, although many of the characteristics can be also found in LAN. In this section, first the reason why application performance over the WAN is an important issue for most organizations is discussed. Next different WAN Technologies explained since it gives conceptual background on the technologies involved when information transferred from applications server to client's computers. Then the different factors affecting network performance are presented.

2.1.1 Organization's WAN and Performance

Today's organizations rely heavily upon the concept of an Anywhere Enterprise® as defined by Yankee Group an Anywhere Enterprise enables corporate users to work anywhere using any device (Kerravala, 2009). The Yankee Group Anywhere Enterprise Lifestyle Survey (2008) found that only 40 percent of workers operate from corporate HQ, while 41 percent is based out of branch offices, 11 percent from home and the remaining 8 percent from the field/on the road.

The above survey could not directly apply to African countries however, in my opinion the IT development trend shows in a few years the concept of Anywhere Enterprise will be used widely in many organizations in Ethiopia or other developing countries. Currently in Ethiopia Business sectors like Banking, Insurance, Airlines, UN agencies and many Government organizations have branch and remote offices in different locations. This work style shift makes organizations to rely heavily upon their WAN network performance to drive day-to-day operations and support employees and customers.

However, with more branch office, remote and mobile users dependent on the WAN and more Web-based applications coming online every day, the amount of traffic crossing organizations' WAN networks is growing exponentially and creating a consistent user experience for remote offices workforces become very difficult. Due to this reason, WAN application performance problem is drawing attention of many researchers (Aberdeen Group, 2009).

Continuing the concept of Anywhere Enterprise, distributed organizations (organizations with branch office, remote and mobile workers) come in all sizes and shapes. Even small organizations may have multiple locations connected by data networks around the world. For large enterprises, even the simplest usable network is a complicated arrangement, while the most elaborate and sophisticated networks have unbelievable layers of replicated and interlocking functionality. Whether a distributed organization is small or large, whether its networks are simple or elaborate, each faces similar business challenges: making the distributed organization efficient and effective in its use of the Wide-Area Networks (WANs) that tie the organization together (Kansanen, 2009).

Often these business challenges show up as distinct IT and WAN issues at remote offices, satellite offices, or branch offices. Poor application performance and poor control of information at branches are both common examples of ineffective use of WANs, which leads to reduced productivity and dangerous exposure to liability. Excessive spending on network bandwidth and high administrative costs for branches are both common examples of inefficient use of WAN resources: throwing away money that could be put to better use (Forrester Consulting, 2010).

Cole and Ramaswamy (2000) discuss in their book how studying application performance over the WAN is so important for organizations. Businesses/Organizations needed to overcome the performance problems associated with the regionalization and globalization of their workforce and customer base. Users everywhere needed data and application access, but the performance of their existing applications over the WAN, is far from being LAN-like performance of applications around the world. This is particularly because, not enough attention has been paid to how applications actually perform over a WAN and on setting user expectation. Therefore, it is important to study WAN traffic characteristics of client/server applications to deal with Networked application performance issues.

Networked applications are pervasive in business, and dependence on them keeps increasing. An outage, or even poor performance, can cripple day-to-day business, affecting revenues directly through lost sales and indirectly through decreased productivity. With the proliferation of enterprise applications and increased complexity of systems and networks, it can be difficult to know how an application will perform on any given network, and to identify the actual cause of performance issues when they arise. Because enterprise applications are rarely tested under WAN conditions (a network with issues like bandwidth limitation, latency, throughput, congestion and packet loss) expectations of performance are often at variance to actual performance (IXIA, 2013). Most application developers create applications in corporate HQ where gigabit (1000 MB) LANs are the norm, while the typical branch office is often connected with T1 speed (1.5 MB) or lower. This means the branch worker is often connected to applications with speeds that are more than 600 times slower than the corporate developer or a worker located in the corporate HQ. This difference in performance creates a massive discrepancy in user experience (Kerravala, 2009).

These challenges drives researchers to study how to solve the effectiveness and efficiency problems at branches and remote locations by making the WAN perform more like a LAN. And these studies also recommend future attempts on how to: deliver dramatic performance improvements, especially for those applications or protocols that show the worst degradation when running over a WAN, and 2) contend with a wide spectrum of protocols that are crossing the networks of a distributed organization (Taneja Group, 2005).

2.1.2 Wide Area Networking

In this section, fundamental concepts about Internetworking and WAN Technologies that are available in the world market and the most used WAN technologies by organizations in Ethiopia including UNECA is discussed.

Open System Interconnect (OSI) Model

Similar to the protocol practices used for the Internet, the OSI model establishes the framework for WAN connectivity (Kurose & Ross, 2008). The model was developed by the International

Organization for Standardization (ISO) in 1984, and it is now considered the primary architectural model for inter-networking communications. This protocol stack helps to divide the tasks involved with moving information between networked computers into seven smaller, more manageable task groups. A task or group of tasks is then assigned to each of the seven OSI layers. These seven layers include the application layer, the presentation layer, the session layer, the transport layer, the network layer, the data link layer, and the physical layer. (Grevers & Christner, 2007) The upper layers of the OSI model deal with application issues and generally are implemented only in software. The highest layer, the application layer, is closest to the end user. Both users and application layer processes interact with software applications that contain a communications component. The lower layers of the OSI model handle data transport issues. The physical layer and the data link layer are implemented in hardware and software. The lowest layer, the physical layer, is closest to the physical network medium.

The OSI model provides a conceptual framework for communication between computers, but the model itself is not a method of communication. Actual communication is made possible by using communication protocols. A protocol implements the functions of one or more of the OSI layers. A wide variety of communication protocols exist. Some of these protocols include LAN protocols, WAN protocols, network protocols, and routing protocols. LAN protocols operate at the physical and data link layers of the OSI model and define communication over the various LAN media. WAN protocols operate at the lowest three layers of the OSI model and define communication over the various wide-area media (Cisco, 2008).

Information being transferred from a software application in one computer system to a software application in another must pass through the OSI layers. For example, if a software application in System A has information to transmit to a software application in System B, the application program in System A will pass its information to the application layer (Layer 7) of System A. The application layer then passes the information to the presentation layer (Layer 6), which relays the data to the session layer (Layer 5), and so on down to the physical layer (Layer 1). At the physical layer, the information is placed on the physical network medium and is sent across the medium to System B. The physical layer of System B removes the information from the physical medium, and then its physical layer passes the information up to the data link layer (Layer 2), which passes it to the network layer (Layer 3), and so on, until it reaches the application layer (Layer 7) of

System B. Finally, the application layer of System B passes the information to the recipient application program to complete the communication process (Jenson, 2009).

A given layer in the OSI model generally communicates with three other OSI layers: the layer directly above it, the layer directly below it, and its peer layer in other networked computer systems. The data link layer in System A, for example, communicates with the network layer of System A, the physical layer of System A, and the data link layer in System B. One OSI layer communicates with another layer to make use of the services provided by the second layer. The services provided by adjacent layers help a given OSI layer communicate with its peer layer in other computer systems. Three basic elements are involved in layer services: the service user, the service provider, and the service access point (SAP). In this context, the service user is the OSI layer that requests services from an adjacent OSI layer. The service provider is the OSI layer that provides services to service users. OSI layers can provide services to multiple service users. The SAP is a conceptual location at which one OSI layer can request the services of another OSI layer.

According to Jenson (2009), the seven OSI layers use various forms of control information to communicate with their peer layers in other computer systems. This control information consists of specific requests and instructions that are exchanged between peer OSI layers. Control information typically takes one of two forms: headers and trailers. Headers are prepended to data that has been passed down from upper layers. Trailers are appended to data that has been passed down from upper layers. An OSI layer is not required to attach a header or a trailer to data from upper layers. Headers, trailers, and data are relative concepts, depending on the layer that analyzes the information unit. At the network layer, for example, an information unit consists of a Layer 3 header and data. At the data link layer, however, all the information passed down by the network layer (the Layer 3 header and the data) is treated as data. In other words, the data portion of an information unit at a given OSI layer potentially can contain headers, trailers, and data from all the higher layers. This is known as encapsulation.

The information exchange process occurs between peer OSI layers. Each layer in the source system adds control information to data, and each layer in the destination system analyzes and removes the control information from that data. If System A has data from a software application to send to System B, the data is passed to the application layer. The application layer in System A then communicates any control information required by the application layer in System B by prepending a header to the data. The resulting information unit (a header and the data) is passed to the presentation layer, which prepends its own header containing control information intended for the presentation layer in System B. The information unit grows in size as each layer prepends its own header (and, in some cases, a trailer) that contains control information to be used by its peer layer in System B. At the physical layer, the entire information unit is placed onto the network medium. The physical layer in System B receives the information unit and passes it to the data link layer. The data link layer in System B then reads the control information contained in the header prepended by the data link layer in System A. The header is then removed, and the remainder of the information unit is passed to the network layer. Each layer performs the same actions: The layer reads the header from its peer layer, strips it off, and passes the remaining information unit to the next highest layer. After the application layer performs these actions, the data is passed to the recipient software application in System B, in exactly the form in which it was transmitted by the application in System A.

Wan Technology

A WAN is a data communications network that covers a relatively broad geographic area and that often uses transmission facilities provided by common carriers, such as telephone companies. WAN technologies generally function at the lower three layers of the OSI reference model: the physical layer, the data link layer, and the network layer as shown in Figure 2.1. WANs use numerous types of devices that are specific to WAN environments. WAN switches, access servers, modems, CSU/DSUs, and ISDN terminal adapters. Other devices found in WAN environments that are used in WAN implementations include routers, ATM switches, and multiplexers. Different WAN technologies are available and some of them are discussed in the following sections as reviewed from (Cisco, 2008).

Point-to-Point Links

A point-to-point link provides a single, pre-established WAN communications path from the customer premises through a carrier network, such as a telephone company, to a remote network. Point-to-point lines are usually leased from a carrier and thus are often called leased lines. For a point-to-point line, the carrier allocates pairs of wire and facility hardware to your line only. These circuits are generally priced based on bandwidth required and distance between the two connected points. Point-to-point links are generally more expensive than shared services such as Frame Relay.

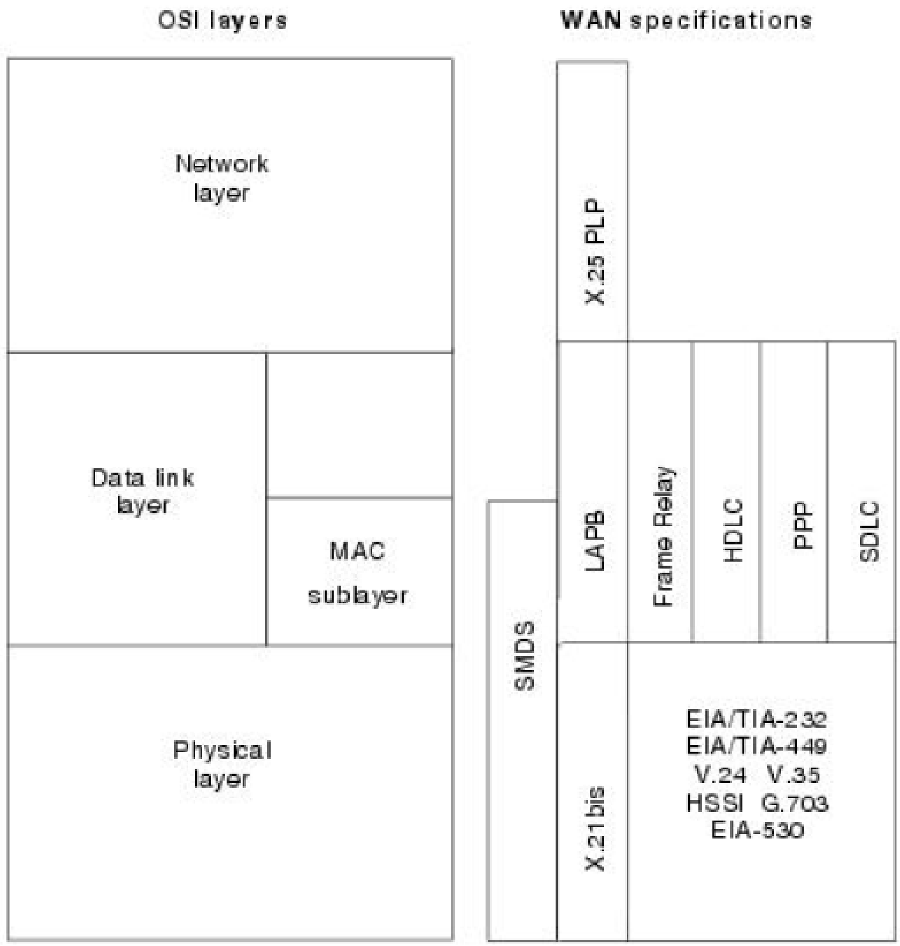


Figure 2.1 WAN Technologies Operate at the Lowest Levels of the OSI Model

Circuit Switching

Switched circuits allow data connections that can be initiated when needed and terminated when communication is complete. This works much like a normal telephone line works for voice communication. Integrated Services Digital Network (ISDN) is a good example of circuit switching. When a router has data for a remote site, the switched circuit is initiated with the circuit number of the remote network. In the case of ISDN circuits, the device actually places a call to the telephone number of the remote ISDN circuit. When the two networks are connected and authenticated, they can transfer data. When the data transmission is complete, the call can be terminated.

Packet Switching

Packet switching is a WAN technology in which users share common carrier resources. Because this allows the carrier to make more efficient use of its infrastructure, the cost to the customer is generally much better than with point-to-point lines. In a packet switching setup, networks have connections into the carrier's network, and many customers share the carrier's network. The carrier can then create virtual circuits between customers' sites by which packets of data are delivered from one to the other through the network. The section of the carrier's network that is shared is often referred to as a cloud. Some examples of packet-switching networks include Asynchronous Transfer Mode (ATM), Frame Relay, Switched Multimegabit Data Services (SMDS), and X.25. The virtual connections between customer sites are often referred to as a virtual circuit.

WAN Virtual Circuits

A virtual circuit is a logical circuit created within a shared network between two network devices. Two types of virtual circuits exist: switched virtual circuits (SVCs) and permanent virtual circuits (PVCs). SVCs are virtual circuits that are dynamically established on demand and terminated when transmission is complete. Communication over an SVC consists of three phases: circuit establishment, data transfer, and circuit termination. The establishment phase involves creating the virtual circuit between the source and destination devices. Data transfer involves transmitting data between the devices over the virtual circuit, and the circuit termination phase involves tearing down the virtual circuit between the source and destination devices. SVCs are used in situations in which data transmission between devices is sporadic, largely because SVCs increase bandwidth used due to the circuit establishment and termination phases, but they decrease the cost associated with constant virtual circuit availability. PVC is a permanently established virtual circuit that consists of one mode: data transfer. PVCs are used in situations in which data transfer between devices is constant. PVCs decrease the bandwidth use associated with the establishment and termination of virtual circuits, but they increase costs due to constant virtual circuit availability. PVCs are generally configured by the service provider when an order is placed for service.

WAN Dialup Services

Dialup services offer cost-effective methods for connectivity across WANs. Two popular dialup implementations are dial-on-demand routing (DDR) and dial backup. DDR is a technique whereby a router can dynamically initiate a call on a switched circuit when it needs to send data. In a DDR setup, the router is configured to initiate the call when certain criteria are met, such as a particular type of network traffic needing to be transmitted. When the connection is made, traffic passes over the line. The router configuration specifies an idle timer that tells the router to drop the connection when the circuit has remained idle for a certain period. Dial backup is another way of configuring DDR. However, in dial backup, the switched circuit is used to provide backup service for another type of circuit, such as point-to-point or packet switching. The router is configured so that when a failure is detected on the primary circuit, the dial backup line is initiated. The dial backup line then supports the WAN connection until the primary circuit is restored. When this occurs, the dial backup connection is terminated.

The remainder of section 2.1 describes factors affecting WAN performance, which includes bandwidth, latency, throughput, congestion and packet loss.

2.1.3 WAN Bandwidth

Bandwidth describes the capacity of the communication link and it is usually measured are bits per second (bps) (Kansanen, 2009). As an example, corporate LANs can have bandwidths from 100Mbp to 1Gbps, whereas WAN links can be 2-50 Mbps. The small branch offices can have WAN connections of 1Mbps or even less. Due to the slower transmission rate, bandwidth usually causes performance problems only in WANs. The difference in bandwidth creates a bottleneck between the LAN and the WAN. These problems, oversubscription and utilization, and are explained in the coming sections.

Oversubscription

The performance related problem caused by different bandwidths used inside network is called oversubscription. When data is coming from a higher rate network to a lower rate network the lower rate part might get oversubscribed. Oversubscription causes queues in the devices, usually routers, handling the change of bandwidth and thus slows down the transmission (Cole & Ramaswamy, 2000).

Figure 2.1 illustrates the possible points of oversubscription in a WAN. As an example let us assume that the LAN part of the connection is 100 Mbps Fast Ethernet and the WAN part is constructed with a point-to-point T1 (1,544 Mbps) links. These two parts are connected with routers to each other. The first point of oversubscription can happen already inside the LAN when the three client workstations (each having a100 Mbps link to the switch) are there joined to one circuit of only 100 Mbps. Connecting three lines into one in the LAN causes 3:1 oversubscription as can be seen from the figure. When moving from LAN to WAN the difference in the network speed means an oversubscription of 67:1 (100 Mbps / 1,544 Mbps). From these two the oversubscription presented in LAN is not causing as much performance problems as the oversubscription when entering the WAN. The LAN oversubscription might not even be present at all times since the three workstations might not be in use at the same time.

Utilization

Utilization is a way to measure the performance of the network. Utilization percentage describes how much of the total network bandwidth, network capacity, is in use at a certain point of time. The best way to detect over utilization in the network is to monitor it constantly. Measurements can be taken for instance from 8 working hours, the busiest hour of the day and from the busiest 15 minutes of the day. Charles (2000) gives an example for these reference values: for the 8-hour working day utilization should be 20% or less and for the busiest hour the average utilization should not exceed 30%. In addition, the 15 minutes average should be less than 50%. Bartlett, Sevcik and Moore (2008) state that overall utilization should be most of the time less than 35%. It is hard to define when the performance is good - these reference values might change based on the network and how it is used. One important factor in defining whether the network is over utilized is to get the opinions of end users. In general, the reference values are based on the examinations of the statistical properties of the traffic.

2.1.4 Latency and delay

Latency is used for describing the time that data travels in the network. It can be expressed as one-way latency or as roundtrip latency. One-way latency, also known as delay, simply means the time it takes for data to travel from the transmitting node to the receiving node. Roundtrip latency, also known as Round Trip Time (RTT), measures the time data travels from transmitting node to the receiver plus the time that it takes for the transmitting node to get a response (acknowledgement) from the receiving node. When studying application performance, the most commonly used form of latency is the RTT (Cole & Ramaswamy, 2000).

Latency can be divided into smaller delay components that together generate the overall network latency. These components are propagation delay, serialization delay, processing delay and forwarding delay. The following sections discuss the different components. It is important to make the distinction between the different types of delay, since some of them are fixed and some are at least partly controllable. Understanding what parts of the delay can be controlled becomes important when talking about improving the network performance. Figure 2.2 below has an example of a network showing how the different parts of the delay affect the overall network latency.

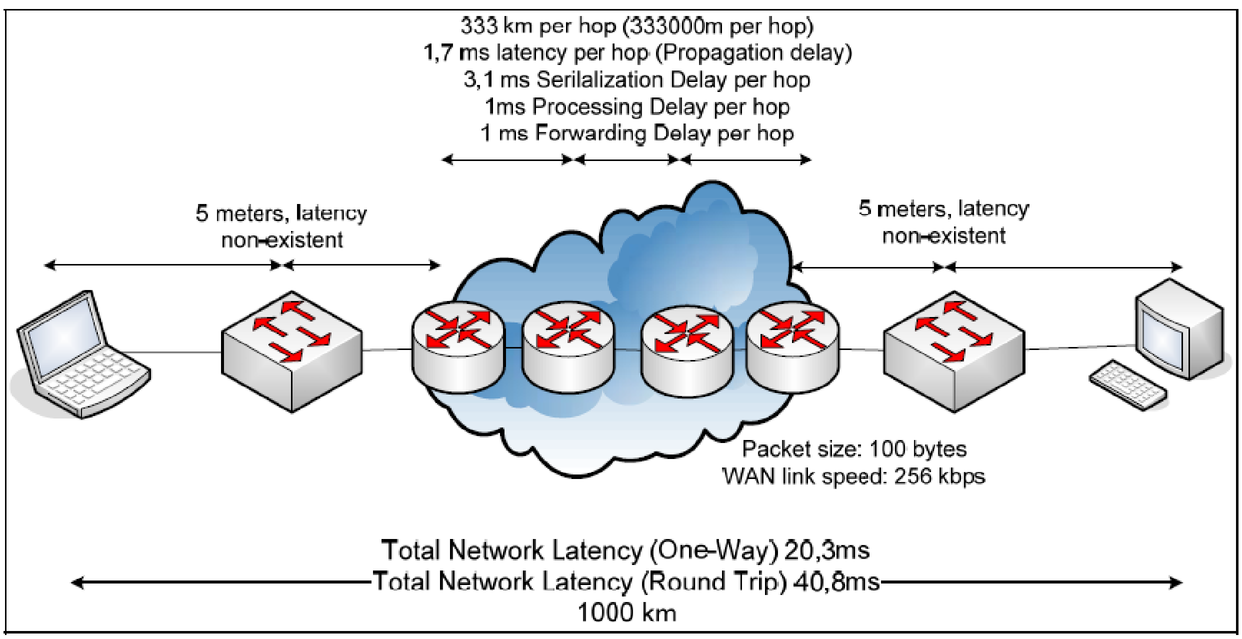


Figure 2.2 Example of how the overall network latency is constructed from different. The different types of delay are presented per hop and the network consists of 3 hops inside the WAN. The total latency per hop is approximately 6.7 ms which created a 20.3 ms one-way latency.

Propagation Delay

Propagation delay is a form of delay that is caused by the distance between nodes and physics in terms of how fast data can be transferred in the network. Propagation delay is one of the fixed factors affecting the overall network latency. Propagation delay is measured as the time the data packet spends to go through the network. The speed packets can be transferred is called propagation velocity and it is normally around 2 / 3 of the speed or light (3x108m/s). Propagation delay becomes significant in long distances, which is usually the case in WANs. In Figure 2.2 for example a connection of 999 km of distance between the sending and receiving end, including three 333 km hops, would have the propagation delay of approximately 5ms and RTT of 10ms. Propagation delay Tp is defined as

$$T_p = d / v,$$

Where d is physical distance (m) and v is propagation velocity (~ 2x108) (Grevers & Christner, 2007).

Serialization Delay

Serialization delay is measured as the time it takes to move bits of a packet into the line. It consists of the size of the packet, network medium and speed of the interface. Usually serialization delay is more significant in lower-speed networks. In Figure 2.2 above we assume the line speed to be 256kbps and the packet size 100 bytes. This creates a serialization delay of 3,1ms per hop. As an example of a high-speed network in a 1Gbps link the same delay would be 100 ns. Serialization delay T s is defined as

$$T_s = s/r,$$

Where s is size of packets (bits) and r is transmission rate (bps).

Processing Delay

Time that it requires for a network node like a router, switch or for example accelerator to perform required actions on the packet is called processing delay. In a router processing delay means the comparison of a piece of data to the access list. The forwarding architecture has to also be counted in processing delay: the node can either wait until the entire packet is received before it makes any decisions what to do with it (store and forward) or the forwarding of the packet can start as soon as the header is received. For instance, in a router the processing delay can vary between less than 1ms to even 10ms when the router is congested. In the Figure 2.2 the processing delay is estimated to be 1ms for each hop in the network, making a total of 3ms end to end processing delay (Cole & Ramaswamy, 2000).

Forwarding Delay

In routers or switches the time spent on deciding where to forward the packet. For example in a router the packet could go through in 1 ms and under a load it could take 3 to 5 ms for the same job. In Figure 2.2 the forwarding delay is estimated to be 1ms per hop creating a total of 3ms forwarding delay for the end to end connection (Cole & Ramaswamy, 2000; Grevers & Christner, 2007).

2.1.5 Packet loss

Transmission systems and data networks are not a perfect system. As a result not all packet transmitted is received by the corresponding layer and this will lead to packet losses. In the event of a packet loss, the transmitter will not receive an acknowledgment for the lost packet, and it will have to retransmit the packet. Packet loss is not a scenario that can be proactively reported to a transmitter: that is, a router that drops a particular packet generally does not notify a transmitting node that a specific packet has been dropped due a congested queue. Packet loss is generally handled reactively by a transmitting node based on the acknowledgments that are received from the recipient or the lack of recipient of said acknowledgements (Kansanen, 2009). Several factors contribute to packet losses:

Transmission Errors

When a transmitter sends a bit of information down a transmission path, it is not always interpreted correctly by the receiving entity on the transmission path. For this reason, protocol suites include error detection or error correction capabilities. Predominantly, error detection methods are employed in most modern WAN networking protocols, due to the relatively low bit error probabilities on transmission systems. These protocols, when detecting an error in a packet, discard the entire data packet. This appears to the higher layer protocols as a packet loss.

Buffer overflows

Data communications equipment that employs statistical packet multiplexing maintains finite data buffers. Due to the statistical nature of the packet loads, there are occasions when the amount of data to be buffered exceeds the size of the data buffers and packet losses occur. To detect packet losses, protocol suites employ packet sequencing by numbering the packets in the order in which they are sent. A lost packet will result in the receiver seeing a packet number or several numbers skipped over.

Transmitter time-outs
When a transmitting host sends a packet, it sets a timer. When the transmitter receives the acknowledgment for the packet, it terminates the timer. If the value of the timer exceeds a threshold, that is, the transmitter time-out value, the transmitter assumes that the packet was lost in transmission, and the transmitter enters into an error recovery state. Time-outs may occur due to packet losses or due to excessively long queuing delays in transit.

Out-of-sequence receptions
Some network technologies ensure that the order of the packets sent to the network is the same as the order of the packets delivered by the network. Some networks do not maintain packet sequencing, most notably IP router networks. Some transport protocol implementations may discard out-of-sequence packets instead of storing them and reordering the incoming packet stream. In this case, out-of-sequence packets will affect the performance of throughput systems in a fashion similar to other packet loss mechanisms.

2.1.6 Throughput
Throughput is the rate of successful data transfer in the network. Grevers and Christner (2007) define throughput as a sum of 3 different parameters: network capacity, latency and packet loss. Capacity means the maximum amount of information that can be transferred between two network nodes. The throughput of a network is never more than the capacity of the slowest hop within that network. For example if the network connection inside a branch office is 1Gbps and it is connected through a 1,5Mbps WAN link to the data center router and again the data center devices are connected to each other with a 1Gbps connection, the throughput of the connection would never exceed the 1,5Mbps. For throughput latency means the time it takes to transfer data between two nodes and also the distance of the nodes. Packet loss adds the element of lost data to the sum, how many packets are dropped for example due to congestion. In addition to these 3 parameters, when talking about throughput for an application, the transport protocol itself can limit the throughput. In cases like this adding more capacity to the network would not necessarily improve the throughput at all.

2.2 End to End protocol

The necessary technological background regarding each layers in the TCP/IP protocol suite and how they behave when used through the WAN is well examined and presented in the coming parts. As it is mentioned in Chapter 1, to achieve the objective of this study it is required to measure and predict the impact of different layers of Internetworking protocols on the performance of critical applications used over the WAN. Hence, the experimental part of the study is based on the facts presented here, which are regarding each TCP/IP protocol suite layers behaviors and limitations over the WAN.

This section presents literature reviews regarding the more well-known protocols and functions that make up TCP/IP. What do these technologies and standards do? What layer of the TCP/IP reference model does each fall into and why? What is a transport protocol and how does it work to ensure data is moved between nodes? How does it create performance limitations? What are the problems caused by End to End protocol (TCP) .These are just a few questions that is discussed in the pages to come. The section concentrates on End-to-End protocol (TCP) and their impact on applications performance. IP layer is also briefed while the application protocols are discussed in the coming section 2.3.

2.2.1 TCP/IP Protocol Suite

TCP/IP is the name that refers to the group of protocols that it encompasses. This group of protocols is known as the TCP/IP protocol suite. It is called TCP/IP because of the two main protocols that are part of the group: TCP and IP. The TCP/IP protocol suite is also known as the Internet protocol suite, as TCP/IP is pretty much the backbone of the Internet (and the majority of all networks out there) (Edwards & Bramante, 2009).

The TCP/IP protocol suite allows data communication to take place. No matter what the node is, who it was made by, which operating system software is running, and where the node is located, TCP/IP makes it work. TCP/IP has kept up with the tremendous growth that the Internet (as well as networks in general) has experienced. The possibilities seem endless and may very well be.

Developers of networking protocols adhere to a layered approach. Each layer is responsible for a different portion of the data communication that is occurring at any time. There are many protocols that are part of the TCP/IP protocol suite. Each protocol functions within a layer of the TCP/IP model, depending on its function. Figure 2.3 shows an example of the TCP/IP model, how it corresponds to the OSI model, and some of the more well - known protocols that are served at each layer (Edwards & Bramante, 2009).

The layers in the TCP/IP reference model roughly correspond to one or more layers of the OSI reference model that are discussed in section 2.1.1. Protocols of the upper layers can focus on the layer they are a member of, without concerning themselves with the functions performed by the lower levels. This is huge during the development of the protocol, as it enables developers to focus on the development at each layer, rather than worrying about an all-encompassing standard (Noble, 2002).

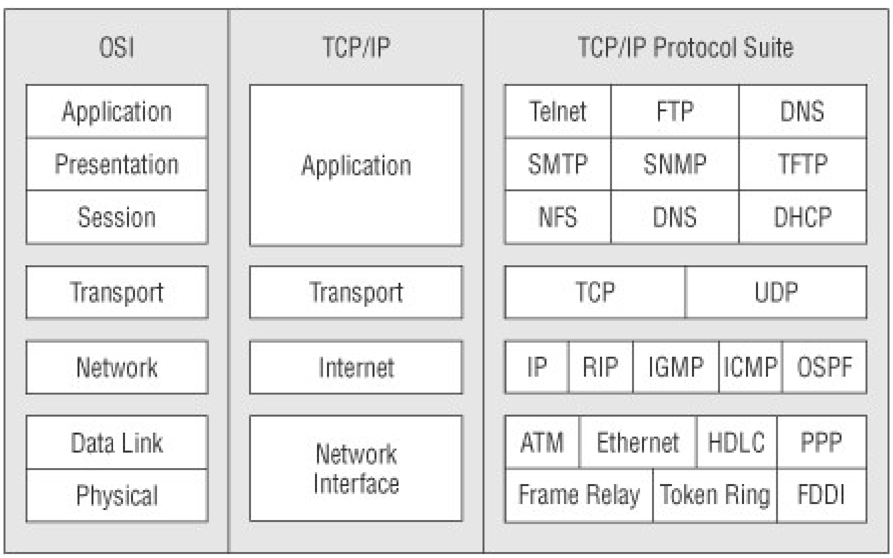


Figure 2.3 TCP/IP reference model layering

The Internet Protocol (IP) is the primary network-layer protocol in the TCP/IP protocol suite and is documented in RFC 791 (Postel, 1981). It is a network-layer (Layer 3) protocol that contains addressing information and some control information that enables packets to be routed. Along with the TCP, IP represents the heart of the Internet protocols. IP has two primary responsibilities: providing connectionless, best-effort delivery of datagrams through an internetwork; and providing fragmentation and reassembly of datagrams to support data links with different maximum-transmission unit (MTU) sizes (Cisco, 2008).

As per Jenson (2009), the IP simply specifies how packets must be formed. Its simplicity provides for the required flexibility that facilitate networking. The protocol is both stateless and connectionless. Stateless means packets can traverse the network and arrive in any order. Connectionless means that packet delivery is unreliable or that there is no acknowledgement or verification of delivery. The IP standard accommodates a variety of underlying network technologies. WANs and LANs can connect regardless of network speeds, connection-orientation, or physical medium (wired, wireless, radio, fiber optics, free space optics, etc.) as long as the IP is adhered to. This packet formation is understood by all components in the network, which allow the packets to be routed from its source to its destination. Additionally, since IP is a published standard specifying exactly how packets need to be formed, multiple vendors can design network components that are interoperable, making IP the protocol that stitches LANs and WANs together. The IP header or preamble to the packet (Figure 2.4) contains the layer 3 IP addresses that are read by layer 3 devices (routers). This is encapsulation at work and the underlying principle that makes internetworking flexible.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |     DSCP      |          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       IP Source Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    IP Destination Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.4 IP Header RFC 791 *(Postel, 1981)*

2.2.2 Transmission Control Protocol (TCP)

In an internetwork, a layer must exist between applications and the underlying network infrastructure. This layer, called the transport layer, not only helps to ensure that data is moved between nodes, but also helps nodes understand how the network is performing so that they can adapt accordingly (Grevers & Christner, 2007).

End-to-end protocols are categorized in the transport layer (layer 4) of the OSI model or layer 2 of the TCP/IP reference model and also named by Transport protocol. Moreover, they are the other critical protocol in the TCP/IP protocol suite and provide instructions on how data is transferred from one end user to another. Remember from the previous section that the IP (layer 3) provides a best-effort delivery infrastructure. It is layer 4's responsibility to execute that delivery. The transport layer adds a port number after the IP address to properly route the packets to the correct port on the end user's machine. The address scheme now looks something like this: XXX.XXX.XXX.XXX:21 (Jenson, 2009).

There are several transport layer (end-to-end) protocols but the two most frequently used are UDP and TCP. TCP provides guaranteed or reliable packet delivery at a bandwidth premium, while UDP provides faster service with no guarantee of delivery at minimal bandwidth cost. This study is about critical application performances and since this application uses TCP, the focus of this section is TCP end-to-end protocol.

TCP is currently the most widely used Internet transport protocol. In 2002, TCP traffic accounted for 95% of the Internet Protocol (IP) network traffic [27]. This was due to a variety of popular Internet applications and protocols. Web (HTTP), file transfer (FTP), and e-mail (SMTP) rely on TCP as the underlying transport protocol. Internet applications that rely on TCP today are likely to do so in the future (Zeng, 2006).

TCP provides reliable transmission of data in an IP environment. Among the services TCP provides are stream data transfer, reliability, efficient flow control, full-duplex operation, and multiplexing. With stream data transfer, TCP delivers an unstructured stream of bytes identified by sequence numbers. This service benefits applications because they do not have to chop data into blocks before handing it off to TCP. Instead, TCP groups bytes into segments and passes them to IP for delivery. TCP offers reliability by providing connection-oriented, end-to-end reliable packet delivery through an internetwork. (Jenson, 2009) It does this by sequencing bytes with a forwarding acknowledgment number that indicates to the destination the next byte the source expects to receive as shown in fig 2.5. Bytes not acknowledged within a specified period are retransmitted. The reliability mechanism of TCP allows devices to deal with lost, delayed, duplicate, or misread packets. A time-out mechanism allows devices to detect lost packets and request retransmission. TCP offers efficient flow control, which means that, when sending acknowledgments back to the source, the receiving TCP process indicates the highest sequence number it can receive without overflowing its internal buffers. Full-duplex operation means that TCP processes can both send and receive at the same time. Finally, TCP multiplexing means that numerous simultaneous upper-layer conversations can be multiplexed over a single connection (Cisco, 2008).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledgement Number                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Data |       |               |                               |
| Offset|Reserve|     Flags     |            Window             |
|       |       |               |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    TCP Options                |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.5. TCP Header RFC 793 *(Postel, 1981)*

Before data can be sent between two disparate application processes on two disparate nodes, a connection must first be established. Once the connection is established, TCP provides guaranteed reliable delivery of data between the two application processes. In the next part of this section, connection oriented service and guaranteed delivery mechanism of TCP is further conferred.

Connection-Oriented Service

The TCP connection is established through a three-way handshake agreement that occurs between two sockets on two nodes that wish to exchange data. A socket is defined as the network identifier of a node coupled with the port number that is associated with the application process that desires to communicate with a peer (Grevers & Christner, 2007).

During the establishment of the connection, the two nodes exchange information relevant to the parameters of the conversation. This information includes (Grevers & Christner, 2007)

- Source and destination TCP ports**:** The ports that are associated with application processes on each of the nodes that would like to exchange application data.

- Initial sequence number: Each device notifies the other what sequence number should be used for the beginning of the transmission.
- Window size: The advertised receive window size; that is, the amount of data that the advertising node can safely hold in its socket receive buffer.
- Options: Optional header fields commonly used for extensions to TCP behavior. For instance, this could include features such as window scaling and selective acknowledgment that were not included as part of the TCP RFC but can augment TCP behavior.

The TCP connection oriented service is elaborated with the following scenario, if an Internet user wants to use Internet Explorer to access http://www.uneca.org, the user's computer would first have to resolve the name www.uneca.org to an IP address, and then attempt to establish a TCP connection to the web server that is hosting www.uneca.org using the well-known port for HTTP (TCP port 80) unless a port number was specified. If the web server that is hosting www.uneca.org is accepting connections on TCP port 80, the connection would likely establish successfully. During the connection establishment, the server and client would tell one another how much data they can receive into their socket buffer (window size) and what initial sequence number to use for the initial transmission of data. As data is exchanged, this number would increment to allow the receiving node to know the appropriate ordering of data. During the life of the connection, TCP employs checksum functionality to provide a fairly accurate measure of the integrity of the data.

Once the connection is established between two nodes (IP addresses) and two application process identifiers (TCP ports), the application processes using those two ports on the two disparate nodes can begin to exchange application layer data. For instance, once a connection is established, a user can submit a GET request to the web server that it is connected to in order to begin downloading objects from a web page, or a user can begin to exchange control messages using SMTP or POP3 to transmit or receive an e-mail message. TCP connection establishment is shown in Figure 2.6.

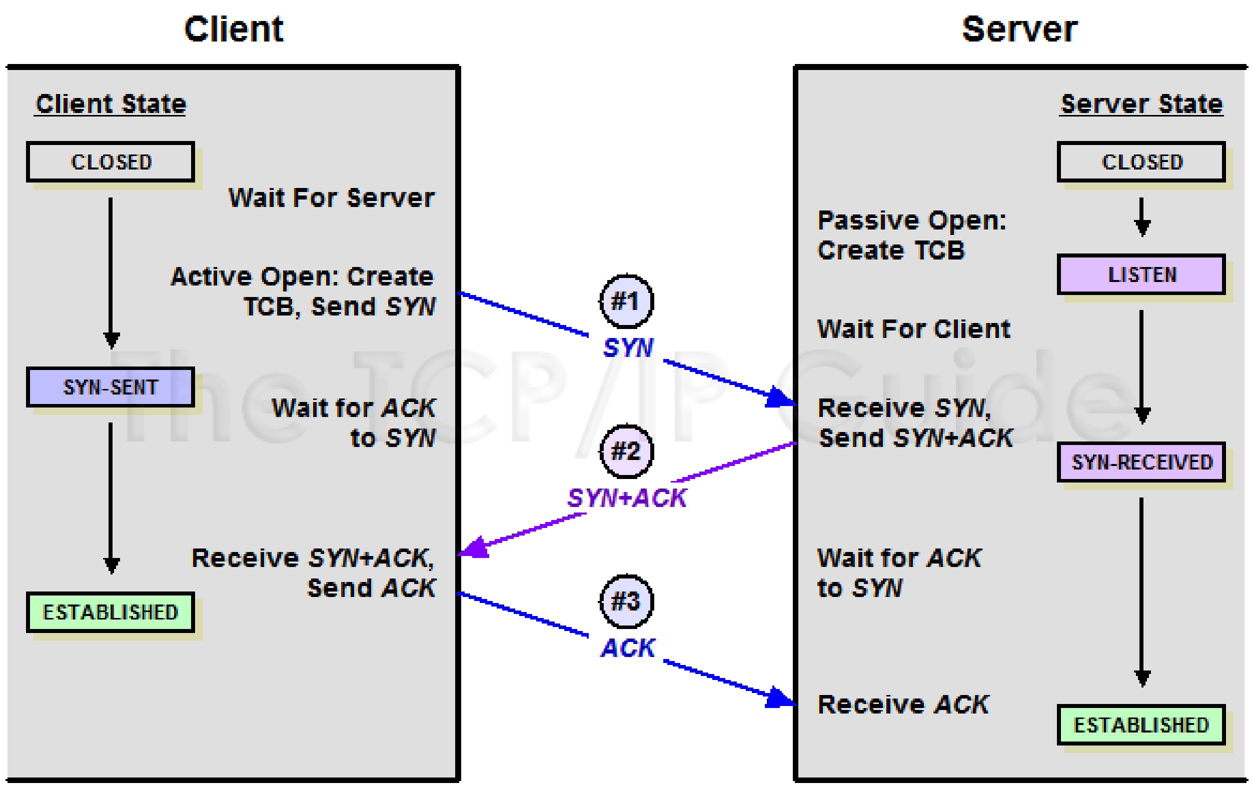


Figure 2.6 TCP Connection Establishment

Guaranteed Delivery

Once transmission commences, application data is drained from application buffers on the transmitting process into the node's socket buffer. TCP then negotiates the transmission of data from the socket transmission buffer to the recipient node (that is, the draining of the buffer) based on the availability of resources on the recipient node to receive the data as dictated by the initially advertised window size and the current window size. Given that application data blocks may be quite large, TCP performs the task of breaking the data into segments, each with a sequence number that identifies the relative ordering of the portions of data that have been transmitted. If the node receives the segments out of order, TCP can reorder them according to the sequence number. If TCP buffers become full for one of the following reasons, a blocking condition could occur (Grevers & Christner, 2007):

TCP transmit buffer becomes full: The transmit buffer on the transmitting node can become full if network conditions prevent delivery of data or if the recipient is overwhelmed and cannot receive additional data. While the receiving node is unable to receive more data, applications may be allowed to continue to add data to the transmit buffer to await service. With the blockade of data waiting in the transmit buffer, unable to be transmitted, applications on the transmitting node may become blocked (that is, momentary or prolonged pauses in transmission). In such a situation, new data cannot be written into the transmit buffer on the transmitting node unless space is available in that buffer, which generally cannot be freed until the recipient is able to receive more data or the network is able to deliver data again.

TCP receive buffer becomes full: Commonly caused by the receiving application not being able to extract data from the socket receive buffer quickly enough. For instance, an overloaded server, i.e. one that is receiving data at a rate greater than the rate at which it can process data, would exhibit this characteristic. As the receive buffer becomes full, new data cannot be accepted from the network for this socket and must be dropped, which indicates a congestion event to the transmitting node.

The above conditions shows, how TCP acts as an intermediary buffer between the network and applications within a node. Which could led to this dependent relations: applications performance is directly related to the Transport protocol operation performance.

In the beginning of this section, some critical points have been mentioned that needs to be addressed to discover necessary fact relations between TCP/IP protocol suite technologies and application performances. What do TCP/IP protocol suite technologies and standards do? What layer of the TCP/IP reference model does each fall into and why? What is a transport protocol and how does it work to ensure data is moved between nodes? Is discussed so far. The Next part of the section attempted to unveil the most critical points, how does TCP create performance limitations? What are the problems caused by End-to-End protocol (TCP) when used over the WAN?

2.2.3 Transport Protocol Limitations

While the transport layer is an unlikely candidate for application performance woes, it can become a problem, primarily because the transport protocols in broad use today were designed in 1981. Today's application demands and network topologies differ greatly from the networks of the early 1980s. For instance, 300 baud was considered blazing fast at the time that TCP was created. Applications in 1981 were commonly a text-oriented applications (and largely terminal oriented), whereas today even the most ill-equipped corporate user can easily move files that are tens upon hundreds of megabytes in size during a single transfer. Although the network has changed, TCP remains relevant in today's dynamic and ever-changing network environment. TCP has undergone only minor changes in the past 25 years, and those changes are in the form of extensions rather than wholesale rewrites of the protocol (Grevers & Christner, 2007).

Internet applications that rely on TCP today are likely to do so in the future. With a growing deployment of WAN networks, it is important to support these applications in both LAN and WAN environments. Hence, WAN networks will also require good TCP performance. WAN networks have different characteristics compared to LAN networks. TCP, which was carefully designed and tuned to perform well in LAN networks, suffers performance degradation when deployed in WAN networks (Zeng, 2006).

As discussed in the previous section, TCP is a connection-oriented protocol, and it has to establish a connection between the sending and the receiving node before any data can be sent. This takes 1.5 round trips (RTT). After the connection established a number of protocol and application messages are exchanged between the sender and receiver and all these messages add latency usually with one round trip. The amount of round trips that a protocol uses for the communication might have an effect on the overall network latency when distances are long. This is not the biggest factor in application performance, but it adds some overhead (Cisco, 2008).

Kelly (2003) states that TCP has 2 mechanisms that affect application performance slow start and congestion avoidance. TCP uses these two mechanisms to determine the capacity of the network and to adapt to possible changes in the network. Slow start is a mechanism that recognizes the network capacity. TCP enters the slow start phase when initializing the connection. In slow start phase, a variable called congestion window (CWND) defines how many packets can be sent through the connection on every RTT. The congestion window is maintained by the sending node. The data transmission starts with a CWND of one packet and it is increased with one packet for every received acknowledgement. This makes the CWND to grow exponentially per RTT:

$$CWND_{i+1} = 2 \times CWND_i \ .$$

The connection stays in the slow start phase until congestion (packet loss) occurs. When a packet loss has been encountered for the first time, it is assumed that the maximum capacity of the network has been reached and TCP enters the congestion avoidance phase. At the same time the CWND size is dropped to

$$CWND_{i+1} = CWND_i / 2$$

In the congestion avoidance phase, the CWND is increased by 1/CWND for each received acknowledgement. This means that the CWND is increased with one packet per RTT:

$$CWND_{i+1} = CWND_I + 1$$

Slow start and congestion avoidance are useful in many ways but can cause problems in networks with high bandwidth and high latency, where it can take a long time to get the window size to the maximum so that most of the available bandwidth could be used. These types of networks are commonly known as long fat networks, LFNs. The term high-speed WAN is also used to describe the networks that can suffer from the poor performance due to these TCP characteristics. Wide area high-speed networks are characterized with speed higher than 100Mbps and RTT over 50ms (Kelly, 2003).

To get an idea of how large influence the two TCP mechanisms have on the throughput of a connection we can calculate the average congestion window. Average congestion window (packets) is defined as:

$$W = BR / (8D)$$

Where B (bps) is throughput, R (s) is RTT and D (bytes) is packet size (Floyd, March 2004). As an example, let us assume that a connection has a RTT of 100ms and a packet size of 1500 bytes. For a 1Gbps bandwidth, the average congestion window would be 8333 packets. In case of congestion, the CWND drops to about 4170 packets, which corresponds a transmission rate of 500Mbps. In the congestion avoidance, it would require about 4170 round trip times to achieve again the transmission rate of 1Gbps, and it would take 7 minutes to do that.

The most common TCP implementation found today, TCP Reno, reduces the congestion window by 50 percent upon encountering packet loss. Although reducing the congestion window by 50 percent does not necessarily correlate to a 50 percent decrease in connection throughput, this reduction can certainly constrain a connection's ability to fully utilize the bandwidth available on the link (Christner, Seils, & Jin, 2010).

Besson (2000) proposed a fluid based model to describe the behavior of a TCP connection over a WAN, and showed that two successive bottlenecks seriously affect the performance of TCP, even during congestion avoidance. The author also derived analytical expressions for the maximum window size, throughput, and round-trip time of a TCP connection.

Some of the challenges associated with TCP behavior on application performance in WAN environments have been discussed; the next section introduces the challenges caused by the application itself when used over the WAN.

2.3 Application Performance

This section discusses the WAN performance problems studied in the previous part from the application perspective. First, what performance requirements the applications required is examined. In addition, some common protocols and applications used in the organizations networks and the performance problems they are facing in WANs are introduced.

Applications use their own protocols, like Hypertext Transfer Protocol (HTTP), on top of TCP. These application protocols usually add extra messages to the data transfer since they normally define how the data exchange is done. This means that there are multiple messages sent over the network before any actual data is sent. Grevers and Christner (2007) use term chatty to describe this kind of application protocols. In short distances this chattiness of an application does not cause problems due to small latencies, but as distances between the sending and receiving node grows also the latency increases and it takes more time for messages to go through the network. Most commonly used chatty application protocols are HTTP, Common Internet File System (CIFS), Network File System (NFS) and Remote Procedure call (RPC). These protocols are covered later in this section.

Since application uses different protocols on top of TCP and these application protocols react differently in different WAN conditions, it is quite necessary to determine the application's (and the end-to-end protocol's) level of sensitivity to these WAN conditions. As it is mentioned in chapter 1 the purpose of this study is to experimentally analyze and determine these protocols reaction on WAN links. This will help enterprises/Organizations select the appropriate methodology/ technology to reduce the apparent effect on applications performance. The next sections 2.31 and 2.3.2 discusses how varies WAN conditions affect applications protocol work and the different applications protocol reactions to these conditions respectively.

2.3.1 WAN and Application Performance

The performance of an application is constantly impacted by the barriers and limitations of the protocols that support the given application and the network that resides between the communicating nodes (Christner, Seils, & Jin, 2010). As discussed earlier in TCP protocols, applications protocols also have been designed for LAN environment and hence they react poorly when introduced to WAN conditions such as latency, bandwidth, congestion, and packet loss. Additional factors impact application performance, such as how the common protocols consume available network bandwidth and how traffic is transferred between devices. Furthermore, other application performance barriers can be present in the endpoints themselves, from something as basic as the network card to something as complex as the block size configured for a local file system.

Applications have different kind of requirements for the network. An application can for example demand low latency or then it can require a faultless transfer to work well in the network. It is important to find out what kind of bandwidth the application needs and what kind of latency and error rate are acceptable. This section discusses the performance requirements of applications.

The staggering drop in bandwidth as traffic moves from the LAN to the WAN is obvious and well understood. The effects of latency are sometimes less obvious, but latency often slows application performance even when ample bandwidth is available. According to Juniper Networks, Inc (2005) analysis, “Network managers who do not spend time addressing the latency issue will not meet the service levels that global applications and business processes demand.” Finally, application contention becomes far more prevalent on bandwidth-restricted WAN links, sometimes getting worse because of addressing the bandwidth limitation.

Bandwidth and Applications

All WANs, when compared to LANs, can be said to be limited in bandwidth. If there is not enough bandwidth for the application, the impacts on performance can be drastic especially for applications, which are Bandwidth intensive. For applications that send large amounts of data in the network the bandwidth easily becomes an issue. Siegel (2006) has collected a set of questions that can help in identifying these bandwidth related problems that applications have:

- Amount of data transferred in both directions (download / upload)?
- How much repetitive patterns data contains?
- How much of the data will be sent only once through the network?
- Is there a variation in the rate data is sent to the network?

In a LAN environment where nodes are connected to the same switch or in a campus network where nodes are connected within proximity to one another, available bandwidth is generally much higher than that required by the two communicating nodes. However, as nodes become spread across a larger, more complex internetwork, network oversubscription might be encountered, or points of aggregation within the network might be encountered. These two factors have a significant impact on application performance, because the point of oversubscription or point of aggregation must effectively throttle transmission from attached networks and negotiate access from the higher-capacity network to the lower-speed, oversubscribed network. In this way, not only does the network element need to manage access to the smaller WAN bandwidth link, but it must also handle the pacing of traffic onto the node through queuing (Grevers & Christner, 2007).

The performance challenges created by bandwidth disparity, oversubscription, and aggregation can quickly degrade application performance. Not only are each of the communicating nodes strangled because their flows are negotiated onto a lower-speed link, but the flows themselves remain strangled even when returning to a higher-capacity link. From the perspective of the server, each of the clients is only sending small amounts of data at a time (due to network oversubscription and bandwidth disparity). The overall application performance is throttled because the server can then only respond to the requests as they are being received.

Application Data

Studying what kind of data the application is sending is important when it comes to improving application performance. A large part of the data going through a WAN link is repetitive. Repetitive data is usual in the Internet, intranet and some client-server applications. In a Web page, for example, this would mean a photo or instructions on how the page is formatted. In a client-server application repetitive data can be for instance a particular file that is sent over the network many times. There can be a lot of repetitive data traffic in corporate networks. Multiple users can use same files during one business day or an email can be sent to whole department at the same time (Siegel, 2006).

Latency and Applications

The sum of all of the components (physics, transport protocols, application protocols, and congestion) adds up to the amount of perceived network latency in hierarchal WAN. As it has been discussed in sections 2.14 and 2.2.3 clearly, the largest impact from the perspective of latency comes from physics. Put simply, every network connection between two nodes spans some distance, and sending data across any distance takes some amount of time. Although physical layer is the major cause for latency, the transport layer can certainly add incremental amounts of latency due to connection management, retransmission, guaranteed delivery, and flow control, which is thoroughly examined in section 2.2. The third component that adds to the amount of perceived latency is application protocols according to (Christner, Seils, & Jin, 2010).

Latency affects some applications more than others do. Siegel (2006) mentions Voice over IP (VoIP) traffic as an example of an application that is affected easily by latency. With VoIP as users expect to hear the person at the other end of the line in real time. The International Telecommunication Union (ITU) has published a recommendation ITU-TG.114 [18] on one-way transmission times. For VoIP the G.144 sets recommendation 150ms or less as a good latency level and 300ms as an acceptable level for really long distance calls. Latency can be also a problem for protocols that use many round-trips for simple functions like initializing the connection. For example if an application needs 20 roundtrips to perform certain function with a one-way delay (latency) of 100ms, the total delay would be 4 seconds. As users are used to delays of only milliseconds they might think that the application has failed to perform the task. Due to poor protocol design, it is not unusual for an application to take hundreds of even thousands of round-trips for relatively simple tasks (Siegel, 2006). For a node-to-node connection over a complex, hierarchal network, the effect of latency on applications can become quite worse.

Error Rates for Applications

Many application protocols require low packet loss to function well. TCP has a way to recover from packet loss or incorrect packets but as discussed in the previous section 2.2 this slows down the network. VoIP and multimedia applications can handle a small amount of errors but a high error rate in a VoIP conversation can make the talking impossible to understand. Due to the real-time nature of VoIP, the resend function of TCP is not suitable for it. Error rate analysis can be done to estimate the error sensitivity of an application (Siegel, 2006).

For data applications, which usually require perfect data transmission, standard TCP prefers a data channel with infrequent, non-burst errors and without packet loss due to congestion. Packet loss should not occur during data bursts; the burst should be buffered without loss within the data channel. For data applications using other protocols to guarantee perfect delivery, there may be less sensitivity to individual packet loss but more sensitivity to end-to-end latency, which might cause session timeout and abandonment. For VoIP and multimedia, the sensitivity of the application to packet loss and to the pattern of packet loss should be investigated. Typically, uniform packet loss of 1% should not greatly damage VoIP and multimedia transmissions, but this depends on the particular application (Siegel, 2006).

Application Contention-Prioritizing applications

Once applications hit the restricted bandwidth available on the WAN, they must contend for access to that precious bandwidth. Increasing bandwidth through compression techniques reduces contention, but it is not possible for IT to fully eliminate the possibility of contention. IT needs a simple but effective approach to resolving application contention. For a better application performance, proper bandwidth allocation and determining which applications should get priority across the WAN is one factor. To apply the bandwidth allocation some parameters like application flow characteristics, is required. Moreover, using this application-based information and business policies, it is possible to set prioritization requirements.

2.3.2 Applications and Application Protocols

In this section, common application protocols and applications used in corporate networks are presented. As applications work in different manners, it is essential to know how the application protocols are constructed. TCP and its impacts to the performance of applications were discussed in section 2.2. This section concentrates on the protocols that work on top of TCP. Main characteristics of each protocol and their capability to work in WAN are discussed to get an understanding of the possible effects on the application performance.

Messaging Application Programming Interface - MAPI

Messaging Application Programming Interface (MAPI) is a protocol that is usually used together with some other protocol like Remote Procedure Call (RPC). MAPI is used for directing messages but RPC is actually the protocol used for the message transfer. Microsoft Exchange is an example of an application that functions in this way. MAPI breaks down the email messages into small data blocks for sending across the network. Each block has to be acknowledged before sending the next block. Therefore sending one email through the network requires thousands of RTTs. The way MAPI works is different within different versions of the Microsoft Exchange. The recent versions have improvements to reduce the performance effects of the protocol when used in WANs. Microsoft Exchange 2003 for example uses local cached copies of data in order to reduce unnecessary traffic over WANs. There are also many other applications that use MAPI in similar ways (Juniper Networks, Inc., 2005).

Common Internet File Service – CIFS

Common Internet File Service (CIFS) is a protocol used for instance for accessing, reading and writing remote files from a server. It is based on Server Message Block (SMB), a protocol developed by Microsoft. It also has a method for controlling file permissions. CIFS was designed for LANs for controlling local file shares that users can access. CIFS requires a lot of message exchange in order to ensure correct usage of the files. As an example Grevers and Christner (2007) state that, opening a 1.5 MB file requires over 1000 message exchanges. The amount of messages is related to the multiple tasks CIFS is handling. These are for example: user authentication, fining correct disk share, checking user permissions, asking which file to open and handling file locks. The file is also broken into multiple data blocks, which all need to be send one at the time through the network. Server only sends the next block after it has received the acknowledgement form the client. Due to this considerable amount of messages that are also sent one at the time, CIFS is said to be a chatty protocol. Therefore, the performance of CIFS suffers when used in WAN, and increased latency reduces performance even more. CIFS contains some optimization methods that reduce the amount of messages during the use of files. Figure 2.7 presents an example of the messages that can be exchanged between the client and the server while a user opens a file from the file server.

Hypertext Transfer Protocol – HTTP

Hypertext Transfer Protocol (HTTP) is the protocol used for Internet traffic. Web pages contain typically many small-embedded objects like pictures, JavaScript codes or Cascading Style Sheets (CSS). In order to transfer the objects the protocol might have to open additional TCP connections. Both the objects and the TCP connections needed for the transfer additional round trips for opening the web page. When a web page is opened for the first time, an HTML file is retrieved first. The HTML file contains links to all other objects needed for displaying the web page completely. Usually the objects are downloaded one at a time and the next object will be downloaded only after the current object has been received completely. Sending and receiving requests one at a time has an effect on the application performance in networks where latency is high. HTTP uses local caching and the second time the same HTML file is downloaded HTTP gets the objects from the local cache. This has some positive effect to the performance although HTTP still needs to check from the server that the objects in the cache have not been changed. This results in the same amount of RTTs as fetching the objects. HTTP has become a popular transport method within corporate applications. For example, applications that area actually client-server based can use web-browser and HTTP to deliver the application to remote workstations (Kansanen, 2009).

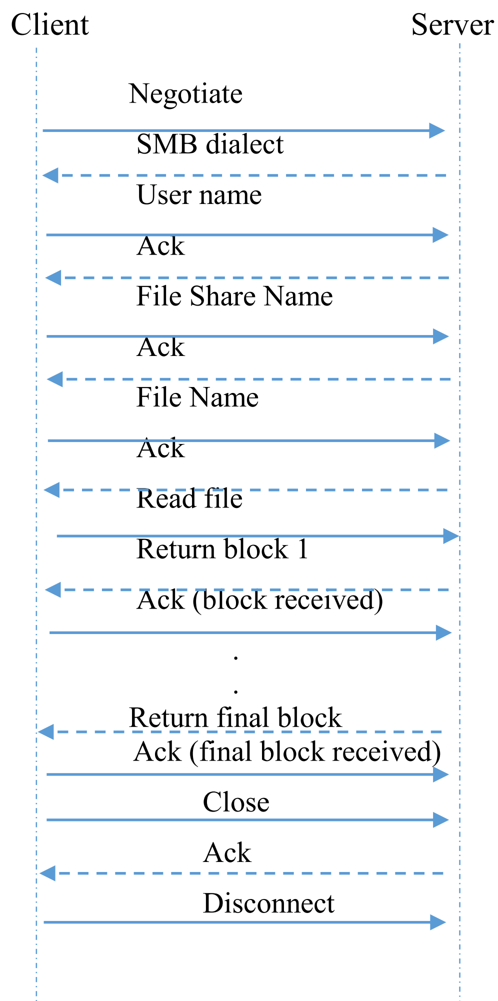


Figure 2.7 CIFS – a rough illustration of the messages that can be changed during opening a file from a server. First steps are: user authentication, finding correct disk share, checking user permissions, asking which file to open. After that the data can be exchanged, the number of messages related to the data exchanged is directly linked to the file size.

Secure Sockets Layer - SSL

Encrypting critical business processes has a significant role in corporate networking. More and more applications are used through the web and require encryption. There are many possibilities for improving the security of these applications. Secure Sockets Layer (SSL) has become the most used. SSL can be used with protocols like HTTP, File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP). SSL was developed first by Netscape. IETF has published Transport Layer Security (TLS) standard, RFC 2246 (Dierks & Allen, 1999) and RFC 5246 (Dierks & Rescorla, 2008) which is based on the latest version of SSL (3.0). Chou (2002) discusses the two main phases of SSL: handshake and data transmission. Handshake is done between client and server and the purpose of the handshake is to determine the secret-key parameters. These secret keys are then used during the data transfer to encrypt and decrypt the data sent over the network. The encrypted data cannot be seen by network devices like routers or accelerators. This makes analyzing or for example, prioritization of the SSL encrypted data difficult. SSL also demands a lot of processing power, especially during the handshake phase, which also affects the performance. BlueCoat (2007) mentions that one of the downsides of SSL is that also spyware and peer-to-peer applications, like instant messaging, exploit the secure SSL tunnel.

SAP

SAP is one of the most used business applications. There are two graphical user interfaces (GUI), SAP-GUI (also known as SAP R/3) and SAP-Web (also known as MySAP). Currently SAP-GUI, which is also the original GUI, is the more popular one. In general SAP is said to have a better performance over network than many other applications. Sevcik and Wetzel (2008) discuss SAP and the performance problems it is facing in WAN. SAP performance can become a problem due to congestion, latency or denial-of-service attacks. Latency, for example, can become a problem for SAP due to the fact that it normally is a centralized service, served from the data centers to the users.

FTP

Many network administrators consider the FTP a "necessary evil." Legacy applications and logging hosts commonly use FTP for simple, authenticated file transfers. FTP is viewed as a simple solution but with a potential for a major impact. Everything from the NIC of the server to the WAN link that the traffic will traverse is impacted by the manner in which FTP transfers content. FTP can disrupt the ability to pass other traffic at the same time an FTP transfer is taking place. By nature, FTP consumes as much bandwidth as possible during its transfer of content, based on what TCP is allowed to consume. FTP tends to use large data buffers, meaning it will try to leverage all of the buffer capacity that TCP can allocate to it (Cisco, 2008).

FTP is such a common protocol on the Internet and WAN networks. RFC 959 states "[t]he objectives of FTP are: 1) to promote sharing of computer programs, data and/or files, 2) to encourage indirect or implicit use of remote computers, 3) to shield a user from variation in file storage systems among hosts, and 4) to transfer data reliably and efficiently" (Jenson, 2009)

Precautions exist within many operating systems and third-party applications that allow the administrator to define an upper limit to any given FTP transfer, preventing congestion situations.

FTP is not fault tolerant and, by nature, is very unforgiving to disruptions. In many cases, a network disruption requires that the file be retransmitted from the beginning. Some application vendors have written client and server programs that leverage FTP as a control and data transfer protocol that allow for continuation of a failed transfer, but these mechanisms are not built into FTP as a protocol itself (Grevers & Christner, 2007).

Simple Mail Transfer Protocol (SMTP)

Networking was created to make communication and file sharing more efficient. The most widely used electronic communication application is e-mail. E-mail is made possible by the SMTP. RFC 821 states that SMTP was designed to "transfer mail reliably and efficiently". FTP and HTTP also facilitate reliable and efficient communication, but they require both the sender and receiver to be connected at the same time. To facilitate communication with a host that is not on the network, SMTP is used (Jenson, 2009).

E-mail traffic is user-oriented, meaning it is sent from one user to another user, not from one computer to another computer. This means that the standard IP addressing will not work to deliver an e-mail to a user, simply because the user is not always receiving their e-mail at the same computer. Instead, a user accesses their e-mail from a particular server that is always connected to the Internet. This server serves as a middle-man that receives the message when the user is not logged in. When the server identifies that the user is logged in, it will deliver the message to the user. Kozierok (2005) describes the process in three steps:

1. Transaction Initiation and Sender identification: The sender establishes a connection with the SMTP server, informing the server that it wants to send a message. This message includes the e-mail address of the sender.

2. Recipient Identification: The sender tells the SMTP server the email address of the recipient.

3. Mail Transfer: Sender transfers the e-mail message to the SMTP server ("The TCP/IP Guide").

The process does not end here. If the e-mail address is not a local SMTP server address, then the server has to look up the address and forward the message to the appropriate SMTP server. When the recipient logs onto their email server, they connect via a SMTP connection to retrieve the message.

SMTP comes in a variety of formats; however, for the purposes of this research it has two main sections: the message header, which contains important control and descriptive data; and the body/payload that carries the data. These connections normally occur on port 25.

The point of this description is to illustrate that SMTP is more complex than FTP or HTTP. It is a more 'chatty' protocol as it frequently sets up and tears down TCP connections for short-term use. Additionally, since SMTP is a connection-oriented and user-oriented protocol, it becomes a good surrogate to represent BFT, an application that is chatty by nature. Understanding the protocol is essential to identifying its behavior during application performance testing.

2.4 Measurements of Application Performance

This section provides the basic application performance measurement background necessary to analyze complex scenarios commonly encountered in today's WAN data network traffics. Different approaches are available to measure the application performance from the end users perspective. The metrics and approaches used by some studies and Network traffic tools are reviewed and presented in this section as follows.

2.4.1 Approaches used in Related Works

Nahum et al. (2001) uses three metrics for determining the WWW server performance: observed throughput, response time as seen by a client, and maximum throughput or capacity, when they study the effect of WAN characteristics on the server.

Bestavros and Kim (1997) evaluate the performance of different TCP types and versions using four metrics: loss rate, response time, retransmission rate, and effective throughput. They conduct experiment by increasing the packet size toward the marginal then the chance of a cell in a packet being dropped at a switch increases and for small buffer sizes, this phenomenon becomes more remarkable, resulting in near 100% packet loss rate and poor performance.

Bai et al. (2007) studies Performance benchmarking of wireless Web servers and extract the application-layer statistics on HTTP behaviors (e.g., reply rate, throughput, response time, and error rate) for a high-level overview of the performance results. Performance data in their experiments come primarily from httperf and the wireless network analyzer, though they collect some performance data, such as netstat information, on client and server machines as well. The httperf tool reports application-layer statistics on HTTP behaviors. The also attempt to get detailed performance data from the wireless network analyzer, enabling traffic analysis from the MAC layer to the HTTP layer.

According to Cole and Ramaswamy (2000), an approach to Computing Response Times Computing application-level response times is usually a complex task, especially if it involves background load. The task is especially complex in the case of two-tier client/server applications. However, it is useful and relatively easy to calculate baseline end user response times under the assumption of unloaded network resources. These baseline response times are useful in troubleshooting WAN performance problems for client/server applications. As per Cole & Ramaswamy, The parameters used for computing response times are total number of bytes transferred in any one direction, segment size , receiver advertised window size, optimal window size for the connection, preferred end user response time (in seconds), one-way propagation delay and protocol overhead factor((S + Overheadbytes) /S).

Sevcik and Wetzel (2008) conducts experimental measurements on the performance of different applications in WAN. Applications used for performance measurments are CIFS, MAPI, HTTP and SAP, which are known for thier chattiness nature. As basis for the evaluation, Sevcik and Wetzel have done measurements on actual data on typical user tasks performed with these applications. For example, for CIFS, the tests included opening of Microsoft PowerPoint files ranging in size from 10KB to 1MB. For MAPI, loading emails 10KB-1MB from Microsoft Exchange Server. From the X-axis "Payload per Task (bytes)" it is possible to see the amount of bytes required to transfer through the network per task per application. From the Y-axis "Turns per Task" it is possible to see the number of client-server interactions (one turn consists of 2 messages sent over the network) needed for performing the task e.g. opening of a file.. The measurements here show that for opening a 1MB file with CIFS it requires approximately 800 turns (1600 messages) over the network. Sevcik and Wetzel use this performance measurement approach for their study and they be able to determine about behavior of different applications. For instance according to their result, CIFS suffers most and SAP GUI least from the centralization of services into data centers and there seems to be also some difference in the way different protocols behave when the payload per task grows, MAPI seems to handle the growth in the task size better than CIFS or Web traffic.

Tsykin and Langshaw (1999) provide four broad techniques for application performance measurement:

- Application instrumentation: modifying the application at the source code level to collect performance data.
- Client instrumentation: inserting hooks into the client environment to collect data on activities such as operating system interrupts and/or messages (as with Microsoft Windows).
- Wire Sniffing: monitoring, decoding and analyzing either raw network (sniffer) traffic or server network packets (i.e., TCP/IP).
- Benchmarking: application scripts periodically executed and measured.

Mirza et al. (2010) proposed a machine learning approach to predict TCP throughput, which was implemented in a tool called PathPerf. The test results showed that PathPerf could predict TCP throughput accurately over diverse wide area paths.

2.4.2 Network Traffic Analysis Tools

Network is an environment with millions of variables and Traffic Analysis Tools are a way of getting an idea of how much and what kind of traffic goes through the network. Network Traffic Analysis is useful for detecting performance problems in the network. Network Traffic Analysis tools are set up to monitor routers or switches in the network. Network devices like routers and switches support the Simple Network Management Protocol (SNMP) that enables the monitoring. Net-Flow and Network Based Application Recognition (NBAR) are examples of mechanisms that are used in Network Traffic Analysis. Many WAN accelerators have also implemented network monitoring in them (Kansanen, 2009). In this thesis, a Net-Flow based monitoring tool is used for evaluating the traffic in the experimentation part of the study in chapter 3. Therefore, the following sections discuss Net-Flow based monitoring and analysis in more detail.

The term flow describes packets travelling in the network between two nodes. A flow has always a source and a destination, which are defined by OSI network layer IP addresses and OSI transport layer source and destination port numbers. Net-Flow is a service developed by Cisco, which allows access to view the flow information of the network. Latest version is called Net-Flow v9 and the IP Flow Information Export (IPFIX) standard (RFC3917) is based on it. With Net-Flow, it is possible to view the network utilization, get detailed information on the application using the network and to see how many flows exist between the network nodes. In corporate networks, Net-Flow helps to determine what kind of prioritization of traffic would be most beneficial.

Net-Flow enabled routers or switches have a cache where they store all IP flows that pass through the device. These IP flows contain attributes like source IP address, destination IP address, IP protocol number and type (e.g. TCP, UDP), source port, destination port, Type of Service (ToS) identifier. If packets contain same attributes these packets are grouped to show the overall (identical) traffic. On top of these attributes, information like timestamp, subnet mask or TCP flags can be collected.

The collected data can be viewed with different kind of tools either in real time or later from a server, also known as a collector, which stores the traffic history for a longer period. The IP flows are sent to the collector as UPD packets where one packet usually contains 30 to 50 flows. Collecting the traffic history is helpful for getting a long-term view of the operation and usage of the network. Normally the traffic history is used, for example, to create reports about the top talkers (systems using the network) or top applications in the network. The real time view of the traffic can on the other hand help in detection problems in the network, for example if something is overloading a link the cause could be detected by viewing the traffic.

Many applications exist that allow for in-depth and thorough analysis of Net-Flow data, including products from Cisco, CA, Hewlett-Packard, InfoVista, NetQoS, and many others. These applications are helpful in analyzing the data presented by Net-Flow and correlating the data into various reports, including these (Grevers & Christner, 2007):

- Top talkers: The nodes that consume the most bandwidth
- Top applications: The applications that consume the most bandwidth

Many of these applications also couple other mechanisms for analyzing performance metrics such as Simple Network Management Protocol (SNMP) polling, remote monitoring (RMON), and traffic analysis using port mirroring. For example, Figure 2.8 shows a report generated using NetQoS SuperAgent that provides insight into who the top talkers on a given network are.

Figure 2.8 shows another report generated by NetQoS SuperAgent that displays the top applications found on the network

2.5 Improving WAN performance

Section 2.1 discussed the general WAN performance problems, for instance problems caused by latency. Section 2.2 and 2.3 concentrated on the performance problems that transport protocols and different application protocols have when used in WANs. This section presents WAN acceleration techniques for improving WAN performance and review of various literature attempts.

In today's world there are various WAN optimization solution providers like Riverbed, Bluecoat, Juniper and Cisco, and each of them have their own features (Balakrishnan, 2011). These solution providers have many different options to improve the performance of application, including different techniques for accelerating WAN traffics such as byte caching, file caching, application-specific compression tools, TCP acceleration tools. However, is it a good solution to apply WAN optimizers to get all the feature? No, because first it is not feasible and second it is not economical. According to Li, et al. (2010) it is economically infeasible and ineffective to apply any of the optimization techniques without identifying biggest bang for the buck or the root cause of the problem. And as it is mentioned in Chapter 1, the goal of this research is to benchmark the performance of critical applications when used over the WAN and this will in turn narrow down the optimization strategies that are needed for the applications. So one of the significance of this research is to help organizations select the best performance improvement techniques for their

WAN environment, from the available WAN optimization techniques that are presented in the coming part of this section.

Improving WAN performance can be divided into two categories: Quality of Service (QoS) and WAN acceleration (Kansanen, 2009). The main focus in this thesis is on WAN acceleration. The term WAN accelerator is used to describe a device designed to accelerate the applications used in the network. Also terms WAN optimization (section 2.5.1) and application acceleration (section 2.5.2) are used to refer to the services provided by the WAN accelerators.

2.5.1 Wan Optimization

According to Zhang, et al. WAN acceleration technologies aim, to accelerate a broad range of applications and protocols, mitigate the impact of latency and loss, and minimize bandwidth consumption.

Fist type of WAN acceleration technique is WAN optimization, which offers transport protocol optimization and compression. The most common techniques employed by WAN optimization to maximize application performance across the WAN include TCP optimization, data de-duplication and compression. All of these try to solve the problems caused by transport protocols and network utilization. These techniques provide solutions for the performance problems described in the section 2.1 and 2.2. In addition, this section presents the following most commonly used WAN optimization techniques.

TCP optimization

Currently, TCP is the most popular transport layer protocol for connection-oriented reliable data transfer in the Internet. TCP is the de facto standard for Internet-based commercial communication networks. The growth trends of TCP connections in WANs were investigated in (Paxson, 1994). However, as discussed in section 2.2, TCP is well known to have poor performance under conditions of moderate to high packet loss and end-to-end latency. Many researches attempted TCP optimization in different approaches, which includes Slow Start Enhancement, Congestion Control Enhancement, TCP Transparent Proxy and TCP Offload Engine. The coming part of this section presents the review done on different literatures, which attempt TCP optimization using the above mentioned approaches (Zhang, Ansari, & Wu, 2011).

A systematic experimental study of IP transport technologies, including TCP and UDP, over 10 Gbps wide-area connections was performed in (Rao, et al., Oct. 2009). The experiments verified the low TCP throughput in the WAN due to high network latency, and the experimental results showed that the encryption devices have positive effects on TCP throughput due to their on-board buffers.

Many studies have observed that TCP performance suffers from the TCP slow start mechanism in high-speed long-delay networks. TCP slow start can be enhanced from two aspects by setting ssthresh intelligently and adjusting the congestion window innovatively. Hoe (1996) proposed to enhance TCP slow start performance by setting a better initial value of ssthresh (slow start threshold) to be the estimated value of bandwidth delay product, which is measured by the packet pair method. There are also some other works that focus on improving the estimate of ssthresh with a more accurate measurement of bandwidth delay product. Aron and Druschel (1998) proposed to use multiple packet pairs to iteratively improve the estimate of ssthresh. Paced Start (Hu & Steenkiste, Nov. 2003) uses packet trains to estimate the available bandwidth and ssthresh. These methods avoid TCP from prematurely switching to the congestion avoidance phase, but it may suffer temporary queue overflow and multiple packet losses when the bottleneck buffer is not big enough as compared to the bandwidth delay product. Adaptive Start (Wang, Pau, Yamada, Sanadidi, & Gerla, March 2004) was proposed to reset ssthresh repeatedly to a more appropriate value by using eligible rate estimation. Adaptive start increases the congestion window (cwnd) size efficiently without packet overflows.

Several TCP slow start enhancements focus on intelligent adjustment of the congestion window size. At the beginning of the transmission, the exponential increase of the congestion window size is necessary to increase the bandwidth utilization quickly. However, it is too aggressive as the connection nears its equilibrium, leading to multiple packet losses. Smooth Start (Wang & Williamson, July 1998) improves the TCP slow start performance as the congestion window size approaches the connection equilibrium by splitting the slow-start into two phases, filling phase and probing phase. In the filling phase, the congestion window size is adjusted the same manner as traditional slow start, while it is increased more slowly in the probing phase. How to distinguish these phases is not addressed in smooth start. An additional threshold max ssthresh is introduced in Limited Slow Start (Floyd, March 2004). The congestion window size doubles per RTT, the same as traditional slow start, if the congestion window size is smaller than max ssthresh; otherwise, the congestion window size is increased by a fixed amount of max ssthresh packets per RTT. Limited slow start reduces the number of drops in the TCP slow start phase, but max ssthresh is required to be set statistically prior to starting a TCP connection.

Lu et al. (2010) proposed a sender-side enhancement to slow start by introducing a two-phase approach, linear increase and adjustive increase, to probe bandwidth more efficiently with TCP Vegas congestion-detection scheme. A certain threshold of queue length is used to signaling queue build-up. In the linear increase phase, TCP is started in the same manner as traditional slow start until the queue length exceeds the threshold. The congestion window size increment slows down to one packet per RTT to drain the temporary queue to avoid buffer overflow and multiple packet losses. Upon sensing the queue below the threshold, the sender enters the adjustive increase phase to probe for the available bandwidth more intelligently.

TCP variants have been developed for wide-area transport to achieve Gbps throughput levels. TCP Vegas (Brakmo & Peterson, Oct. 1995) uses network buffer delay as an implicit congestion signal as opposed to drops. This approach may prove to be successful, but is challenging to implement. In wireless and wired-wireless hybrid networks, TCP-Jersey enhances the available bandwidth estimations to improve the TCP performance by distinguishing the wireless packet losses and the congestion packet losses. By examining several significant TCP performance issues, such as unfair bandwidth allocation and throughput degradation, in wireless environments with window control theory, Hiroki et al. (2010) found that the static additive-increase multiplicative decrease (AIMD) window control policy employed by TCP is the basic cause for its performance degradation. In order to overcome the performance degradation caused by the static AIMD congestion window control, explicitly synchronized TCP (ESTCP) (Nishiyama, Ansari, & Kato, April 2010) was proposed by deploying a dynamic AIMD window control mechanism, which integrates the feedback information from networks nodes. High-speed TCP modifies the congestion control mechanism with large congestion windows, and hence, High-speed TCP window will grow faster than standard TCP and recover from losses more quickly. This behavior allows High-speed to quickly utilize the available bandwidth in networks with large bandwidth delay products. A simple sender side alteration to the TCP congestion window update algorithm, Scalable TCP, was proposed to improve throughput in highs-speed WANs. Numerical evaluation of the congestion control method of Scalable TCP and its impacts on other existing TCP versions were reported in (Tekala & Szabo, 2005).

The long delay is one of the main root causes for the TCP performance degradation over WANs. TCP transparent proxy involves breaking of long end-to-end control loops to several smaller feedback control loops by intercepting and relaying TCP connections within the network. The decrease in feedback delay accelerates the reaction of TCP flows to packet loss more quickly; hence, the accelerated TCP flows can achieve higher throughput performance.

Ananth et al. (2003) introduced the idea of implementing a single logical end-to-end connection as a series of cascaded TCP connections. They analyzed the characteristics of the throughput of a split TCP connection analytically and proved the TCP throughput improvement by splitting the TCP connection. A flow aggregation based transparent TCP acceleration proxy was proposed and developed in (Chakravorty, Katti, J, & I, April 2003) for GPRS network. The proxy splits TCP connections into wired and wireless parts transparently, and aggregates the connections destined to the same mobile hosts due to their statistical dependence to maximize performance of the wireless link while inter-networking with unmodified TCP peers.

A Control for High-Throughput Adaptive Resilient Transport (CHART) system (Brassil, McGeer, Rajagopalan, Sharma, & Yalagandula, Jan. 2009) was designed and developed by HP and its partners to improve TCP/IP performance and service quality guarantees with a careful re-engineering Internet layer 3 and layer 4 protocols. The CHART system enhances TCP/IP performance through two principal architectural innovations to the Internet Layer 3 and Layer 4 protocols. One is the fine-grained signaling and sensing within the network infrastructure to detect link failures and route around them. The other one is the explicit agreement between end hosts and the routing infrastructure on transmission rate, which will permit the end hosts to transmit at the agreed rate independent of loss and delay. To achieve these two innovations, CHART couples a network-wide real-time network monitoring service, a new generation of network elements which monitor and control flows to explicitly assign available bandwidth to new and existing flows, and a new TCP driver, TCP-Trinity, which implements the explicit-rate protocol on the end hosts to accelerate data transmission by bypassing the slow-start and congestion avoidance phases of data transfer. Either manual modification on the end-hosts or route configuration changes are required by the original CHART TCP accelerators.

TCP Offload Engine (TOE) is a technology used in network interface cards (NIC) to offload TCP processing to the network controller. Moving complex TCP processing from end-systems into specified networking hardware reduces the system load and frees up resources. The experiment evaluation in (Feng, Balaji, Baron, Bhuyan, & Panda, Aug. 2005) provided some insights into the effectiveness of the TOE stack in scientific as well as commercial domains.

A transparent TCP acceleration technique was proposed in (Ladiwala, Ramaswamy, & Wolf, March 2009) by using network processors to increase TCP throughput inside the network without requiring any changes in end-system TCP implementations, and is thus undetectable to the end-system. The architecture of this TCP accelerator implemented by splitting a long end-to-end TCP connection into several small TCP connections. Two packet-forwarding paths are implemented in the acceleration nodes. Non-TCP packets or packets that cannot be accelerated due to resource constraint will be forwarded without any modification. Hence, packet classification is required to distribute packets to go through TCP acceleration or not. TCP accelerator involves IP input process, TCP acceleration, IP output processing, and a large TCP state memory storage. Network processors equipped with this transparent TCP accelerator can opportunistically act as TCP proxies by terminating TCP connections and opening a new connection towards the destination.

Compression

Compression is very important to minimize the amount of bandwidth consumed on a link during transfer across the WAN in which bandwidth is quite limited. It can improve bandwidth utilization efficiency, thereby reducing bandwidth congestion; it can also reduce the amount of transit time for given data to traverse the WAN by reducing the transmitted data. Therefore, compression substantially optimizes data transmission over the network. A comparative study between various text file compression techniques is reported in (Al-laham & Emary, April 2007). A survey on XML compression is presented in (Sakr, 2009). Another survey on lossless image compression methods is presented in (Shukla, Alwani, & Tiwari, April 2010). A survey on image and video compression is covered in (Clarke, 1999). HTTP is the most popular application-layer protocol in the Internet. HTTP compression is very important to enhance the performance of HTTP applications. HTTP compression techniques can be categorized into two schemes: HTTP Protocol

Aware Compression (HPAC) and HTTP Bi-Stream Compression (HBSC) schemes. By exploiting the characteristics of the HTTP protocol, HPAC jointly uses three different encoding schemes, namely, Stationary Binary Encoding (SBE), Dynamic Binary Encoding (DBE), and Header Delta Encoding (HDE), to perform compression. SBE can compress a significant amount of ASCII text present in the message, including all header segments except request-URI (Uniform Resource Identifier) and header field values, into a few bytes. The compressed information is static, and does not need to be exchanged between the compressor and de-compressor. All those segments of the HTTP header that cannot be compressed by SBE will be compressed by DBE. HDE is developed based on the observation that HTTP headers do not change much for an HTTP transaction and a response message does not change much from a server to a client. Hence, tremendous information can be compressed by sending only the changes of a new header from a reference.

Data De-duplication

Data de-duplication, also called redundancy elimination (Mandagere, Zhou, Smith, & Uttamchandani, 2008), is another data reduction technique and a derivative of data compression. Data compression reduces the file size by eliminating redundant data contained in a document, while data de-duplication identifies duplicate data elements, such as an entire file (Bolosky, Corbin, Goebel, & Douceur, Aug. 2000) and data block (Spring & Wetherall, 2000), and eliminates both intra-file and inter-file data redundancy, hence reducing the data to be transferred or stored. When multiple instances of the same data element are detected, only one single copy of the data element is transferred or stored. The redundant data element is replaced with a reference or pointer to the unique data copy. Based on the algorithm granularity, data de-duplication algorithms can be classified into three categories: whole file hashing (Bolosky, Corbin, Goebel, & Douceur, Aug. 2000), sub-file hashing (Spring & Wetherall, 2000), and delta encoding (Hunt, Vo, & Tichy, 1996). Traditional data de-duplication operates at the application layer, such as object caching, to eliminate redundant data transfers. With the rapid growth of network traffic in the Internet, data redundancy elimination techniques operating on individual packets have been deployed recently (Saha, Lukyanenko, & Yla-Jaaski, April 2011) based on different chunking and sampling methods.

The main idea of packet-level redundancy elimination is to identify and eliminate redundant chunks across packets. A large-scale trace-driven study on the efficiency of packet-level redundancy elimination has been reported in (Anand, Muthukrishnan, Akella, & Ramjee, 2009). This study showed that packet-level redundancy elimination techniques can obtain average bandwidth savings of 15-60% when deployed at access links of the service providers or between routers.

Compression and data de-duplication help to overcome the problems caused by oversubscription. Changing the amount of data sent over WAN minimizes the effects of bandwidth discrepancy between LAN and WAN (Kansanen, 2009).

2.5.2 Application Acceleration

Application acceleration tries to improve the application layer operation in WAN. Application acceleration can be further categorized into object caching, prefetching (read-ahead) and Quality of Service. These techniques are discussed in this section. In section 2.3, I discussed the performance problems that the common applications used in corporate networks have. This section presents solutions for improving the performance of these application protocols. Because different application protocols work in different ways, it is not necessarily possible to apply all these techniques to every protocol (Grevers & Christner, 2007).

Caching

Caching is considered to be an effective approach to reduce network traffic and application response time. Based on the location of caches, they can be deployed at the client side, proxy side, and server side. Owing to the limited capacity of a single cache, caches can also work cooperatively to serve a large number of clients. Cooperative caching can be set up hierarchically, distributively, or in a hybrid mode. From the type of the cached objects, caches can be classified as function caching and content caching. A hierarchical classification of caching solutions is shown in Figure 2.9 (Zhang, Ansari, & Wu, 2011).

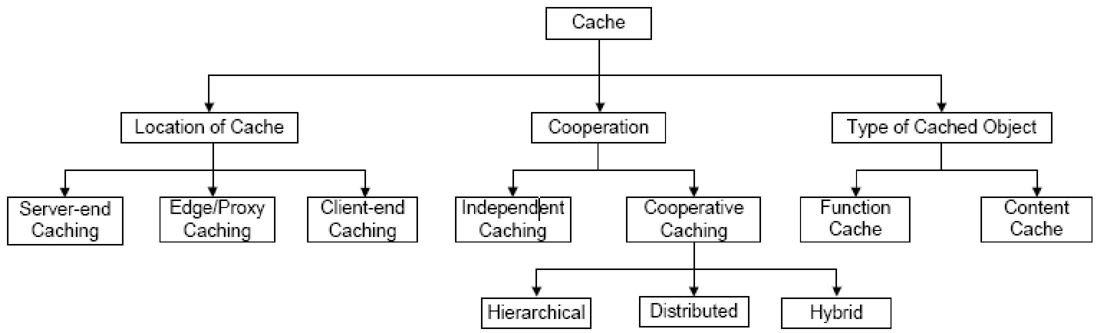


Figure. 2.9 caching classification hierarchy

Location of cache: Client side caches are placed very close to and even at the clients. All the popular web browsers, including Microsoft Internet Explorer and Mozilla Firefox, use part of the storage space on client computers to keep records of recently accessed web content for later reference to reduce the bandwidth used for web traffic and user perceived latency.

Several solutions have been proposed to employ client cache cooperation to improve client-side caching efficiency. Squirrel (Iyer, Rowstron, & Druschel, 2002), a decentralized, peer-to-peer web cache, was proposed to enable web browsers on client computers to share their local caches to form an efficient and scalable web cache. In Squirrel, each participating node runs an instance of Squirrel, and thus web browsers will issue their requests to the Squirrel proxy running on the same machine. If the requested object is un-cacheable, the request will be forwarded to the origin server directly. Otherwise, the Squirrel proxy will check the local cache. If the local cache does not have the requested object, Squirrel will forward the request to some other node in the network. Squirrel uses a self-organizing peer-to-peer routing algorithm, called Pastry, to map the requested object URL as a key to a node in the network to which the request will be forwarded. One drawback of this approach is that it neglects the diverse availabilities and capabilities among client machines. The whole system performance might be affected by some low capacity intermediate nodes since it takes several hops before an object request is served.

Contrary to the client-side caching, server-end caches are placed very close to the origin servers. Server-end caching can reduce server load and improve response time especially when the client stress is high. For edge/proxy side caching (Vakali, 2001), caches are placed between the client and the server. According to the results reported in (Kroeger, Long, & Mogul, Dec. 1997), local proxy caching could reduce user perceived latency by at best 26%.

In distributed caching systems, there are only institutional caches at the edge of the network. The distributed caching system requires some mechanisms to cooperate these institutional caches to serve each other's cache misses. Several mechanisms have been proposed so far, including the query-based approach, content list based approach, and hash function based approach. Rodriguez et al. (2001) proposed analytical models to study and compare the performance of both hierarchical and distributed caching. The derived models can be used to calculate the user perceived latency, the bandwidth utilization, the disk space requirements, and the load generated by each cache cooperating scheme. In order to maximize the advantages and minimize the weaknesses of both hierarchical and distributed caching architectures, hybrid caching architecture has been proposed (Rodriguez, Spanner, & Biersack, Aug. 2001). In a hybrid-caching scheme, the cooperation among caches may be limited to the same level or at a higher level caches only. Rabinovich et al. (1998) proposed a hybrid-caching scheme, in which the cooperation is limited between the neighboring caches to avoid obtaining documents from distant or slower caches. The performance of the hybrid scheme and the optimal number of caches that should cooperate at each caching level to minimize user retrieval latency has been investigated in (Rodriguez, Spanner, & Biersack, Aug. 2001).

Prefetching

Although caching offers benefits to improve application performance across the WAN, passive caching has a limitation on reducing application latency due to low hit rates (Kroeger, Long, & Mogul, Dec. 1997). Abrams et al. (1996) examined the hit rates for various workloads to investigate the removal policies for caching within the Web. Kroeger et al. (1997) confirmed a similar observation that local proxy caching could reduce latency by at best 26% under several scenarios. They also found that the benefit of caching is limited by the frequency update of objects in the web.

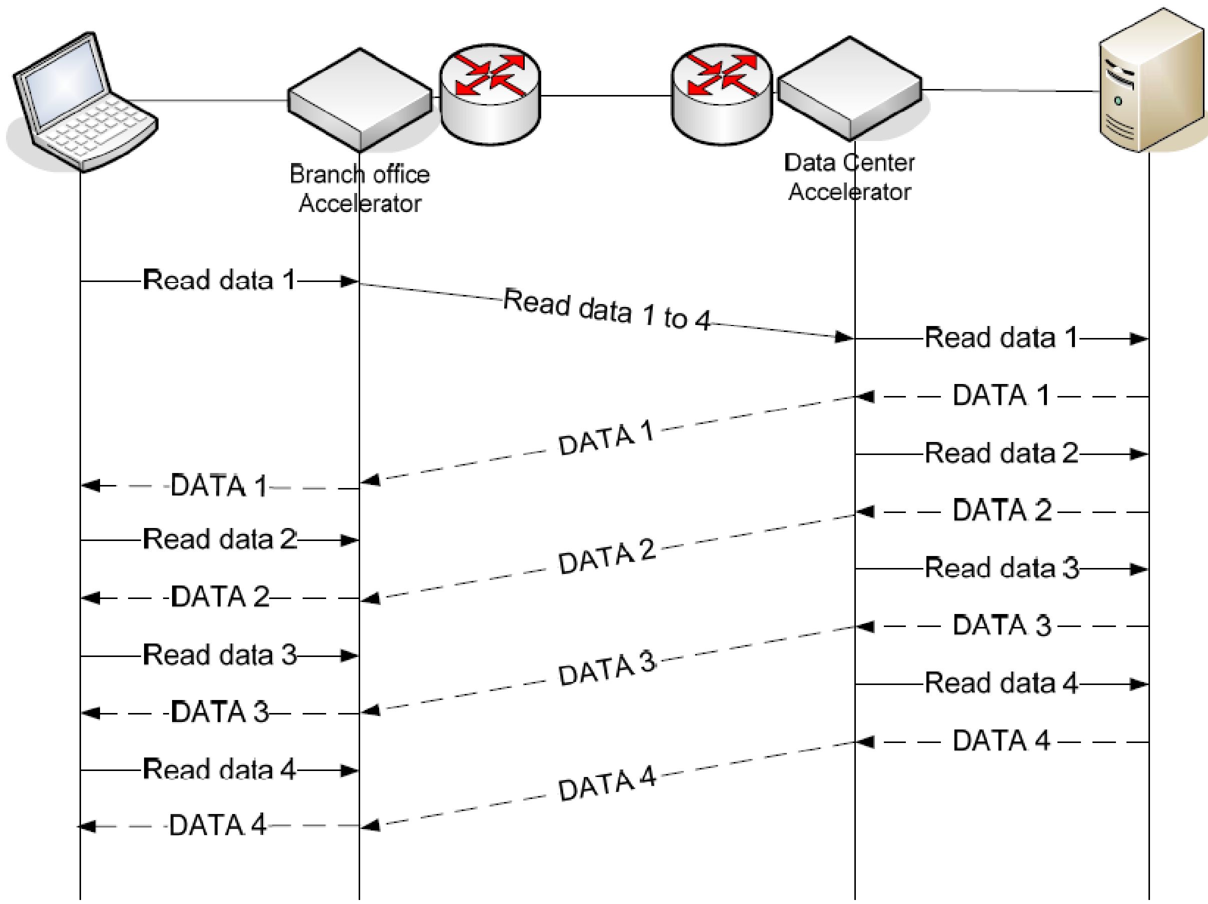


Figure 2.10: Read - Ahead for CIFS. When a client request comes to the branch office accelerator to read data 1, the accelerator will request also data 3 to 4 from the server. When client requests the remaining 3 data blocks, these are already at the branch office accelerator and can be served to the client quickly

Prefetching (or proactive caching) is aimed at overcoming the limitations of passive caching by proactively speculative retrieval of a resource into a cache in the anticipation of subsequent demand requests. Figure 2.10 above shows an example of how prefetching (read ahead) functions with CIFS. Client requests first sequence of data from the server. Instead of just requesting the first sequence of data the accelerator will request 3 more sequences (Grevers & Christner, 2007).

The experiments reported in (Kroeger, Long, & Mogul, Dec. 1997) showed that prefetching doubles the latency reduction achieved by caching, and a combination of caching and prefetching can provide a better solution to reduce latency than caching and prefetching alone. So far, several web-prefetching architectures have been proposed according to the locations of the prediction engine and prefetching engine, which are two main elements of web prefetching architecture. In addition, many prediction algorithms have been proposed, and how far in advance a prefetching algorithm is able to prefetch an object is a significant factor in its ability to reduce latency. The effectiveness of prefetching in addressing the limitations of passive caching has been demonstrated by several studies. Most research on prefetching focused on the prediction algorithm.

Quality of Service

Normally IP networks use best effort delivery for handling the IP flows. Best effort delivery means that all traffic is treated equal and the flows are handled in the order they enter the network device. This might cause problems from the corporate network perspective. Treating the business critical applications with the same priority as Internet traffic for example, could have a big effect on the performance of the business critical applications (Kansanen, 2009).

Integrated services (IntServ) and differentiated services (DiffServ) are considered as the 2 ways to implement quality of service in corporate networks. IntServ is older and it is more complex to implement than DiffServ. Thus, most of the corporate networks use DiffServ as the QoS method. Both use Classes of Service (CoS) to set different kind of priorities for the traffic. In DiffServ the QoS is done by assigning application traffic to different CoS queues in the network based on priority. For example a corporate network could have different CoS for voice traffic, business critical applications, email and Internet browsing. Even if Internet, set to the lowest CoS, would be used more than other applications, it would not consume all the bandwidth as some of the bandwidth is reserved for other CoS levels (Willis, 2005).

Quality of Service provides methods with which it is possible to define different service levels for different types of traffic. For corporate IP networks, the quality of service became possible after the invention of MPLS IP VPN networks. Before this ATM networks already had QoS in them. QoS is a part of the functions improving WAN performance and many WAN accelerator appliances offer it as one of their services. Although many corporations already have implemented the router based QoS for small companies the QoS offered by the accelerators might be a desirable feature (Newman, Aug 2007).

2.6 Application Performance Benchmarking

The point of implementing WAN optimization is not to get packets from one router to another faster. It is to improve response times for end-users as they perform real-world business tasks. Benchmarks should therefore be based on improvements experienced at the desktop by remote users everywhere across the organization –especially those experiencing the slowest performance with the most important business applications. Benchmarking under current production conditions alone is insufficient. Those conditions will certainly change as users, locations and applications are added. So WAN optimization solutions should be benchmarked under various types of projected utilization and routing conditions – including worst-case and/or disaster recovery scenarios (Shunra, 2009).

Benchmarks are designed to mimic a particular type of workload on a component or system. Synthetic benchmarks do this by specially created programs that impose the workload on the component. Application benchmarks run real-world programs on the system. While application benchmarks usually give a much better measure of real-world performance on a given system, synthetic benchmarks are useful for testing individual components, like a hard disk or networking device (Wikipedia, 2010).

For an application, it is no longer sufficient to test functionality alone. Testing an application to ensure that it satisfies the desired performance parameters is fast becoming the norm. Performance testing and benchmarking an application can be performed before deploying the product in the production environment or after the deployment. If the business is expected to grow, the organizations should ascertain the application ability to support the extended business volumes on existing infrastructure and the potential areas of concern or bottlenecks in terms of performance. It is essential to identify the areas where enhancements, tuning and upgrade are required. Performance testing and benchmarking provide proof of the performance of the product and set a baseline for further enhancement in the application (Puri, 2008).

Performance benchmarks are used to evaluate applications or server performances, and thus have a large impact on what performance optimizations are applied to application Infrastructure and servers. A key challenge is to precisely predict the impacts of WAN performance factors on the application response time (Nahum, Rosu, Seshan, & Almeida, 2001).

A realistic application benchmarking experiment should capture, in some reasonable way, the behavior of latest applications when used through different WAN conditions as well as the behavior of end-users interacting with these applications (Li, et al., 2010).

To fully exploit today's cloud platforms trends, WAN optimization solution providers and application developers need access to modern tools and benchmarks to design, experiment, and enhance suitable optimization solutions and applications (Cecchet, Udayabhanu, & Wood, 2011).

2.6.1 Related Research Works

Many are working to address the issues of poor network performance and the unpredictability of end to- end network bandwidth availability. To address unpredictability, the Network Weather Service project is working to predict the network bandwidth available between two sites on the Internet based on statistical forecasting. Efforts to address poor network performance include Diffserv, Quality of Service (QoS) Reservation, Bandwidth Brokering, and network and application tuning efforts (Noble, 2002).

File size, type and popularity. Authors of benchmarks such as SURGE and SPECWeb96 developed their file size and request distribution by analyzing the logs of many popular Web sites. As usage patterns have changed over time, some benchmarks such as SPECWeb99 have been updated with more recent request distributions and the addition of dynamic content. Many benchmarks view request-related information as the key to understanding Web server performance and reproduce little else. Request arrival and rate. Most benchmarks attempt to stress the server by requesting objects as quickly as possible. However, some benchmarks attempt to recreate more realistic arrival distributions. The SURGE benchmark uses the concept of "user equivalents" to recreate the active and idle periods of typical users. Another benchmark, Scalable Client (s-client), uses an "open-loop" architecture to produce a fixed request arrival rate regardless of server load. Both these approaches cause the server to deal with the bursty arrival of requests, which can significantly affect server performance (Nahum, Rosu, Seshan, & Almeida, 2001).

Single Protocol Scripts in (Jenson, 2009): The Throughput script was used to baseline test the link for maximum throughput. This script sends the specified file size from one endpoint to the other and waits for an acknowledgement (Ixia, 2007, p. 8-83). File size was incremented from 100KB to 1 MB to 10 MB. This script, uncelebrated, provided the baseline to compare accelerated results.

The FTP get script was used to simulate an FTP get command. File size was incremented from 100 KB to 1 MB to 10 MB. When this file is run bi-directionally, it is equivalent to FTP get and put commands being run at the same time. The HTTP gif script was used to simulate the transfer of graphic files from an HTTP server. File size was incremented from 100 KB to 1 MB to 10 MB.

The SMTP script was used to simulate typical e-mail traffic. This script includes an additional 20-byte header along with the selected file size. Since e-mail traffic typically consists of smaller files these scripts were modified as such. File size was incremented from 1 KB to 100KB to 1 MB. The NetMtgv script was used to simulate streaming video, a UDP protocol, with factory set defaults. As illustrated in the research work, the TCP accelerator does not touch UDP traffic, referred to as pass through, and therefore limited testing was done with this script. A summary of the script configuration for the individual protocol tests is provided in Table.

The demand to improve WAN applications performance has existed for quite some time and, as a result, there have been many previous efforts to measure and characterize the performance of these applications. These efforts can be roughly divided into benchmarking studies and analyses of WAN network conditions.

Li, et al. (2010) studied how to accurately predict and benchmark the performance impact of different WAN conditions on the web service to narrow down the optimization strategies that could bring the most benefits. This study introduces WebProphet, a system that automates performance prediction for web services. WebProphet is a model that employs a novel technique based on timing perturbation to extract WAN applications dependencies, and then uses these dependencies to predict the application performance impact of changes to the handling of the objects. A key challenge is to precisely identify web object dependencies, as these are essential for predicting performance in an accurate and scalable manner. This experimental study decomposes the loading time of critical application into client delay, network delay, and server delay. The client delay is due to various client application activities such as page rendering and Java script execution. The network delay can be further decomposed into DNS lookup time, TCP three-way handshake time, and data transfer time. TCP handshake time and data transfer time are influenced by network path conditions such as RTT and packet loss. The server delay is produced by various server processing tasks such as retrieving static content or generating dynamic content.

The researchers tested WebProphet to the Search and Maps services of Google and Yahoo, find WebProphet predicts the median and 95th percentiles of the page load time distribution with an error rate smaller than 16% in most cases. Using Yahoo Maps as an example, they find that WebProphet reduces the problem of performance optimization to a small number of web objects whose optimization would reduce the page load time by nearly 40%.

Bai, Oladosu & Williamson (2007) conducts experimental Studies on performance benchmarking of Wireless Web Servers. The researchers attempted to benchmark the performance capabilities of wireless Web servers in short-lived ad hoc network environments. Network traffic measurements are conducted on an in-building IEEE 802.11b wireless ad hoc network, using a wireless-enabled Apache Web server, several wireless clients, and a wireless network traffic analyzer. Their experiments focuses on the HTTP transaction rate and end-to-end throughput achievable in such an ad hoc network environment, and the impacts of factors such as Web object size, number of clients, and persistent HTTP connections. The researcher's results show that the wireless network bottleneck manifests itself in several ways: inefficient HTTP performance, client-side packet losses, server-side packet losses, network thrashing, and unfairness among Web clients. Persistent HTTP connections offer up to 350% improvement in HTTP transaction rate and user-level throughput, while also improving fairness for mobile clients accessing content from a wireless Web server.

Nahum, Rosu, Seshan and Almedia (2001) attempted to show how WAN conditions can affect WWW server performance. The researchers examine these effects using an experimental test-bed, which emulates WAN characteristics in a live setting, by introducing factors such as delay and packet loss in a controlled and reproducible fashion. Their result demonstrate that when more realistic wide area conditions are introduced, servers exhibit very different performance properties and scaling behaviors, which are not exposed by existing benchmarks running on LANs. This paper also shows that observed throughputs that can give misleading information about server performance, and recommends maximum throughput, or capacity, as a more useful metric. They have also discovered that packet losses can reduce server capacity by as much as 50 percent and increase response time as seen by the client. The researchers show that using TCP SACK can reduce client response time, without reducing server capacity. Issues considered in the paper include: how does server performance scale with load? How do packet delay and loss affect observed throughput? What is the impact of loss and delay on response time as seen by the client? How do TCP variants such as SACK and New Reno influence performance? And how do delay and loss affect server capacity?

Cecchet, Udayabhanu and Wood (2011) examined realistic Benchmarking of Web Applications to improve the performance of today's web application that support rich client interactivity. This research focuses on HTTP application. They have tried to capture real latencies perceived by users geographically distributed on the Internet for benchmarking modern Web applications such as Wikibooks and demonstrate the effect of geographically distributed load on the performance. The effect of Internet round-trip times also captured and Web performance benchmarking "at scale" by leveraging modern public clouds---by using a number of cloud-based client instances, possibly in different geographic regions, is supported. Their research make contributions by providing empirical results on the need to capture the behavior of real Web browsers during Web load injection.

2.7 Tool for Applications and Network Modeling

OPNET Modeler

OPNET Modeler provides a comprehensive development environment supporting the modeling of communication networks and distributed systems. Both behavior and performance of modeled systems can be analyzed by performing discrete event simulations. The OPNET Modeler environment incorporates tools for all phases of a study, including model design, simulation, data collection, and data analysis.

OPNET Modeler is used to construct models for two different purposes: to study system behavior and performance and to deliver a modeling environment to "end users" with other OPNET analysis software products.

OPNET Modeler is a vast software package with an extensive set of features designed to support general network modeling and to provide specific support for particular types of network simulation projects. Some of the most important capabilities of OPNET Modeler are:

- Object orientation—Systems specified in OPNET Modeler consists of objects, each with configurable sets of attributes. Objects belong to classes, which provide them with their characteristics in terms of behavior and capability. Definitions of new classes are supported to address as wide a scope of systems as possible. Classes can also be derived from other classes, or "specialized" to provide more specific support for particular applications.

- Specialized in communication networks and information systems—OPNET Modeler provides many constructs relating to communications and information processing, providing high leverage for modeling of networks and distributed systems.
- Hierarchical models—OPNET models are hierarchical, naturally paralleling the structure of actual communication networks.
- Graphical specification—wherever possible, models are entered via graphical editors. These editors provide an intuitive mapping from the modeled system to the OPNET model specification.
- Flexibility to develop detailed custom models—OPNET Modeler provides a flexible, high-level programming language with extensive support for communications and distributed systems. This environment allows realistic modeling of all communications protocols, algorithms, and transmission technologies.

Typical Applications of OPNET Modeler

Because of the capabilities described above, OPNET Modeler can be used as a platform to develop models of a wide range of systems. Some examples of possible applications are listed below with specific mention of supporting features:

- Standards-based LAN and WAN performance modeling—detailed library models provide major local-area and wide-area network protocols. Configurable application models are also provided by the library, or new ones can be created.
- Internetwork planning—hierarchical topology definitions allow arbitrarily deep nesting of sub-networks and nodes and large networks are efficiently modeled; scalable, stochastic, and/or deterministic models can be used to generate network traffic.
- Research and development in communications architectures and protocols—OPNET Modeler allows specification of fully general logic and provides extensive support for communications-related applications. Finite state machines provide a natural representation for protocols.
- Distributed sensor and control networks, on-board systems—OPNET Modeler allows development of sophisticated, adaptive, application-level models, as well as underlying

communications protocols and links. Customized performance metrics can be computed and recorded, scripted and/or stochastic inputs can be used to drive the simulation model, and processes can dynamically monitor the state of objects in the system via formal interfaces provided by statistic wires.

- Resource sizing—accurate, detailed modeling of a resource's request-processing policies is required to provide precise estimates of its performance when subjected to peak demand (for example, a packet switch's processing delay can depend on the specific contents and type of each packet as well as its order of arrival). Queuing capabilities of Proto-C provide easy-to-use commands for modeling sophisticated queuing and service policies; library models are provided for many standard resource types.

OPNET Modeler Architecture

OPNET Modeler provides a comprehensive development environment for modeling and performance-evaluation of communication networks and distributed systems. The package consists of a number of tools, each one focusing on particular aspects of the modeling task. These tools fall into three major categories that correspond to the three phases of modeling and simulation projects:

1. Model Specification and Modeling Communications with Packets
2. Data Collection and Simulation
3. Analysis

These phases are necessarily performed in sequence. They generally form a cycle, with a return to Specification following Analysis. Specification is actually divided into two parts: initial specification and re-specification, with only the latter belonging to the cycle, as illustrated in the following Figure 2.11.

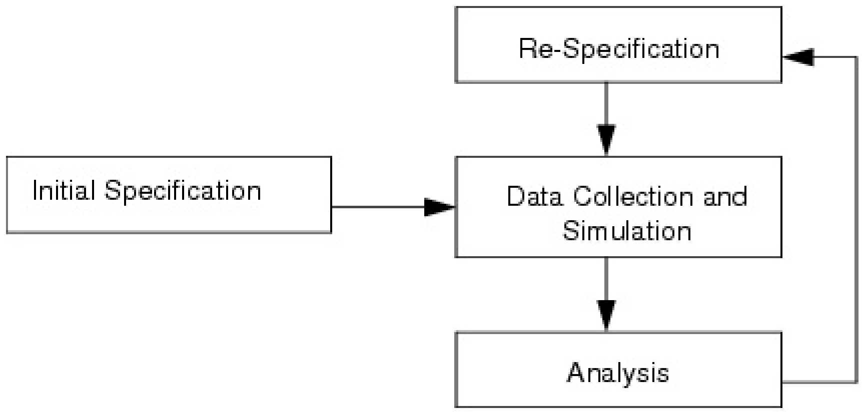


Figure 2.11 Simulation Project Cycle

1- Model Specification

Model specification is the task of developing a representation of the system that is to be studied. OPNET Modeler supports the concept of model reuse so that most models are based on lower level models developed beforehand and stored in model libraries. Ultimately, however all models are based on the basic concepts and primitive building blocks supplied by the OPNET Modeler environment.

OPNET Modeler supports model specification with a number of tools, called editors, which capture the characteristics of a modeled system's behavior. Because it is based on a suite of editors that address different aspects of a model, OPNET Modeler is able to offer specific capabilities to address the diverse issues encountered in networks and distributed systems. To present the model developer with an intuitive interface, these editors handle the required modeling information in a manner that is parallel to the structure of real network systems. Therefore, the model-specification editors are organized hierarchically. Models built in the Project Editor rely on elements specified in the Node Editor; in turn, when working in the Node Editor, you use models defined in the Process Editor and External System Editor. The remaining editors are used to define various data models, typically tables of values that are later referenced by process- or node-level models.

Modeling Domains

The Network, Node, Process, and External System modeling environments are sometimes referred to as the modeling domains of OPNET Modeler because they span all the hierarchical levels of a model. The remaining specification editors correspond to no particular modeling domain because they mainly support the three principal editors. As mentioned earlier, the capabilities offered by the modeling domains mirror the types of structures found in an actual network system; the issues addressed by each domain are summarized in the following table 2.1 and then briefly described in the remainder of this section.

A network model may contain any number of communicating entities called nodes. Nodes are instances of node models, developed using the Node Editor. Modelers can develop their own library of customized node models, implementing any functionality they require. Network models consist of nodes and links that can be deployed within a geographical context. OPNET Modeler comes with fixed nodes and point-to-point and bus links; the Wireless functionality adds mobile and satellite nodes and radio links. Fixed, mobile, and satellite sub-network objects provide hierarchy in the network model and are used to break down complexity into multiple levels. Subnets can contain various combinations of nodes, links, and other subnets, and can be nested to any depth.

Domain	Editor	Modeling Focus
Network Domain	Project	Network topology described in terms of subnetworks, nodes, links, and geographical context.
Node Domain	Node	Node internal architecture described in terms of functional elements and data flow between them.
Process Domain	Process	Behavior of processes (protocols, algorithms, applications), specified using finite state machines and extended high-level language.
External System Domain	External System	Interfaces to models provided by other simulators running concurrently with a discrete event simulation (a cosimulation).

Table 2.1 Modeling Domains

Node models consist of modules and connections. Modules are information sources, sinks, and processors. Some modules have pre-defined behavior; processor and queue modules are programmable via their process model. Connections (packet streams and statistic wires) allow information to flow between modules.

Process models define behavior for programmable modules. A process is an instance of a process model and operates within one module. Initially, each programmable module contains only one process; however, processes can create additional child processes dynamically. These can in turn create additional processes themselves. This paradigm is well suited to modeling systems with dynamically instantiated contexts, like certain protocols, or multi-tasking operating systems. Processes respond to interrupts, which indicate that events of interest have occurred such as the arrival of a message or the expiration of a timer. When a process is interrupted, it takes actions in response and then blocks, awaiting a new interrupt. It may also invoke another process; its execution is suspended until the invoked process blocks.

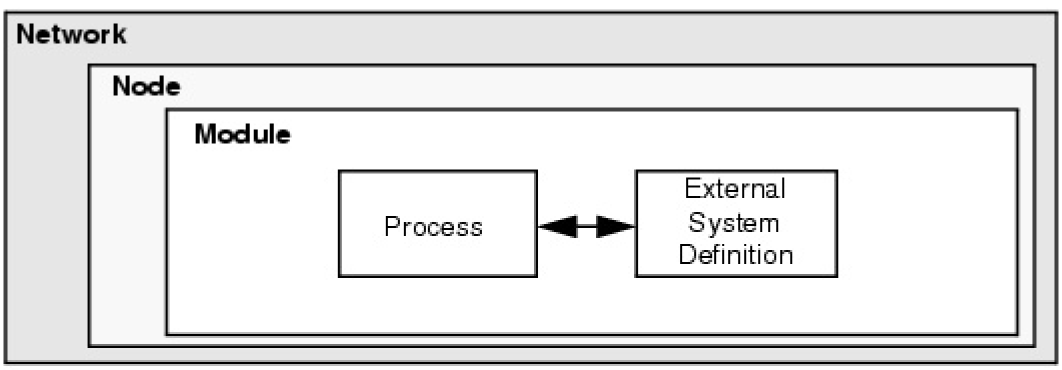


Figure 2.12 Relationship of hierarchical level in models

Objects are the building blocks of models and appear in each of the modeling domains. Some objects are created explicitly by the user; others are implicitly created by OPNET Modeler. During a simulation, certain types of objects can be created dynamically. Objects provide attributes as a means of controlling their behavior. The attributes constitute part of the object's interface. Each attribute has a name, a value, and properties. Properties specify the rules governing the attribute's use, including its data type, allowable values, suggested values, and documentation

Object	Modeling Domain	Model Type
Node (all types)	Network	Node
Link (all types)	Network	Link
Processor	Node	Process
Queue	Node	Process
External System	Node	Process, ESD

Table 2.2 Objects Supporting Model Assignment

2- Modeling Communications with Packets

The previous sections describe the support OPNET Modeler provides for representing the structure of a network, including the elements that appear at different hierarchical levels. Communication of data takes place at all levels of this hierarchy and most of the supported objects play some role in implementing this communication: processes can be considered to be both the sources and sinks of data and control information; the node domain is the context in which processes communicate information with each other and with physical layer transmitters and receivers; and the network domain allows nodes to exchange information with each other via various types of communication links.

There are many forms of communication that are supported by the OPNET Modeler modeling environment. However, there is one fundamental structure, called packet that provides the most commonly used mechanism for information exchange. Packets are objects that contain formatted information that can change dynamically. They can be stored by and transferred between objects in each of the modeling domains. In the Node Domain, packets typically travel over streams; in the Network Domain, they are typically transferred over links.

3- Data Collection and Simulation

The objective of most modeling efforts is to obtain measures of a system's performance or to make observations concerning a system's behavior. OPNET Modeler supports these activities by creating an executable model of the system. Provided that the model is sufficiently representative of the actual system, OPNET Modeler allows realistic estimates of performance and behavior to be obtained by executing simulations. Several mechanisms are provided to collect the desired data from one or more simulations of a system.

Discrete event simulations are capable of producing many types of output. Of course, because of the general programmability of process models and link models, developers can define their own types of output files, including text reports, proprietary binary files, etc.

4- Analysis

The third phase of the simulation project involves examining the results collected during simulation. Typically, much of the data collected by simulation runs is placed in output vector files. OPNET Modeler provides a graphing and numerical processing environment in the Results Browser of the Project Editor.

CHAPTER THREE
EXPERIMENTATION

In this chapter, various experiments undertaken on cases that are taken from UNECA WAN environment is presented. Section 3.1 provides an overview about the target organization and its WAN environment. In section 3.2, the experimentation tasks done using the methodology indicated in the introduction part is discussed. These experiments are conducted to identify the critical application performance problems, and to evaluate various performance optimization techniques.

3.1 Business Understanding

In this section, an overview about the study target organization, UNECA, is given which is taken from the official UNECA website (UNECA, 2013). Then the UNECA WAN environment is illustrated and finally critical applications that are selected for the experiment are presented.

3.1.1 UNECA

UNECA was established in 1958 with a mandate to promote the economic and social development of its member states. UNECA, with its HQ located in Addis Ababa, serves 54 Member States, which are organized in five sub regions: North Africa, West Africa, Central Africa, East Africa and Southern Africa.

In order to better serve its African Member States, the Economic Commission for Africa has opened, in 1963, a representation (office) in each one of the five sub regions. By virtue of being closer to member States, ECA sub-regional offices are better equipped to prepare country and sub-regional profiles that include risk analysis for the member States of the Economic Commission for Africa with a focus on strengthening the relationship with National Statistical Offices in the countries of the sub-region to produce and use quality statistical information.

ECA sub-regional offices develop and maintain sub-regional repositories of statistical information that feed into the common databank and support analytical and research needs of the Commission. The representative offices in five sub regions of Africa is described below

Sub-Regional Office for Central Africa (SRO-CA)

Located in Yaounde, Cameroon, the Sub-Regional Office for Central Africa of the Economic Commission for Africa (ECA/SRO-CA covers the following member States: Cameroon, Chad, Central African Republic, Congo, Equatorial Guinea and Sao Tome & Principe. Given its remit to support the Regional Economic Communities, namely CEMAC and ECCAS as well as their specialized institutions, ECA/SRO-CA also collaborates with Angola, Burundi and the Democratic Republic of Congo. The Office further works with non-governmental organizations, the civil society, universities, research institutions and the private sector.

Sub-Regional Office for Eastern Africa (SRO-EA)

SRO-EA is based in Kigali, Rwanda and covers Burundi, Comoros, D.R Congo, Djibouti, Ethiopia, Eritrea, Kenya, Madagascar, Rwanda, Seychelles, Somalia, South Sudan, Tanzania and Uganda. The SRO-EA serves two Regional Economic Communities (RECs): Intergovernmental Authority on Development (IGAD) and East African Community (EAC); and Three intergovernmental Organizations (IGOs): The Economic Community of the Great Lakes Countries (CEPGL), The International Conference on the Great Lakes Region (ICGLR) and Indian Ocean Commission (IOC).

Sub-Regional Office for North Africa (SRO-NA)

The ECA Office for North Africa is located in Rabat, Morocco, and serves seven member states: Algeria, Egypt, Libya, Mauritania, Morocco, Sudan, and Tunisia. The Office facilitates networking and information exchange among development actors in North Africa and provides a link between these actors and ECA, thereby helping disseminate ECA's policy recommendations and technical publications.

Sub-Regional Office for Southern Africa (SRO-SA)

ECA established Southern Africa Sub-regional presence in in Lusaka, Zambia in 1966. The office serves eleven countries: Angola, Botswana, Lesotho, Malawi, Mauritius, Mozambique, Namibia, South Africa, Swaziland, Zambia and Zimbabwe. SRO-SA seeks to strengthen the capacities of member States for regional integration along specific priorities of the Southern African sub-region, the NEPAD framework and Millennium Development Goals.

Sub-Regional Office for West Africa (SRO-WA)

The Sub-Regional Office for West Africa of the Economic Commission for Africa (ECA/SRO-WA) is one of the five Sub-Regional Offices in Africa. Located in Niamey, Niger it serves the 15 countries members of the Economic Community of Western African Countries (ECOWAS): Benin, Burkina Faso, Cape Verde, Côte d'Ivoire, the Gambia, Ghana, Guinea, Guinea-Bissau, Liberia, Mali, Niger, Nigeria, Senegal, Sierra Leone, and Togo.

3.1.2 UNECA WAN and Critical Applications

As stated in the previous section ECA sub-regional offices produce and use quality statistical information, maintain sub-regional repositories of statistical information and feed this information into common data bank located in one of the sub-regional offices. Moreover to carry out the ECA's mandate, the sub-regional office staffs uses electronic resources located in HQ-data center as well as collaborate with colleagues from other sub-regional offices using various media.

To help achieve the above mentioned tasks, WAN is deployed which connects the five SROs and UN HQ in New York to UNECA HQ in Addis Ababa. The UNECA WAN architecture is shown in Figure 3.1 and it doesn't include the UN HQ link since the scope of this thesis limited to studying the performance of the five sub-regional offices' WAN.

The WAN technology used is Satellite communication for some sub-regional offices and Virtual Private Network (VPN) for the others. Since the initial deployment, the WAN has evolved through some changes, including satellite bandwidth upgrade and VPN deployment to improve its performance. Yet, the attained WAN performance improvement did not enhance the WAN critical application performance to the desired level (section 3.2.1 and appendices, support these facts with data).

Different services and applications are consolidated in the HQ data center. Thus, the WAN is being used intensively by different services and applications like Lotus Notes for email communications, Citrix for Finance and HR application, Video conferencing, Voice data for telephone, Active Directory replication, FTP for file transfer, Remote Desktop Protocol for monitoring servers, HTTP for Intranet, electronic applications (e-leave, e-billing and e-clearance) and many other services.

Selection of critical Applications

The traffic flow of any WAN environment comprises of a variety of applications including business-critical, less critical and unknown applications (Shunra, 2009). One of the WAN optimization approaches used in this study is improving the performance of these business critical applications at the expense of less-critical and unknown applications.

For this study, critical applications of UNECA WAN are selected based on the methodology defined in chapter 1. Applications, which are business critical for UNECA, are given higher priority (researcher interviews management bodies to determine business critical applications). Applications, which are more sensitive to various WAN conditions, are given the next priority. Thus, Video conferencing system, Voice application, email application protocol, Http protocol (Web application), and multi-tier Database applications are given higher priority and selected as critical applications of the organization for this study.

3.2 WAN Traffic Analysis

As discussed in Chapter 2, there are different factures that cause poor application performance, at the same time, different solution to mitigate these problems. Li et al. (2010) argued that it is not economical to apply all solutions and not effective to pick any solution, thus the purpose of this experiment is, investigating the impact of each factor and evaluating various optimization techniques, on critical application performance. The experimentation phase of the study analyzes different factors that have been discussed in literature review part of this thesis from the target organization perspective. In the analysis part emphasize is given on examining the degree of impact by each factor on the selected critical application performance.

The experiment requires, manipulating different factors like disabling the specific application server, changing the protocols properties of the online application, which can affect the production environment. There are also fixed factors that cannot be changed in actual environment. Modeling the existing UNECA WAN environment makes it possible, what cannot be experimented easily in the actual environment.

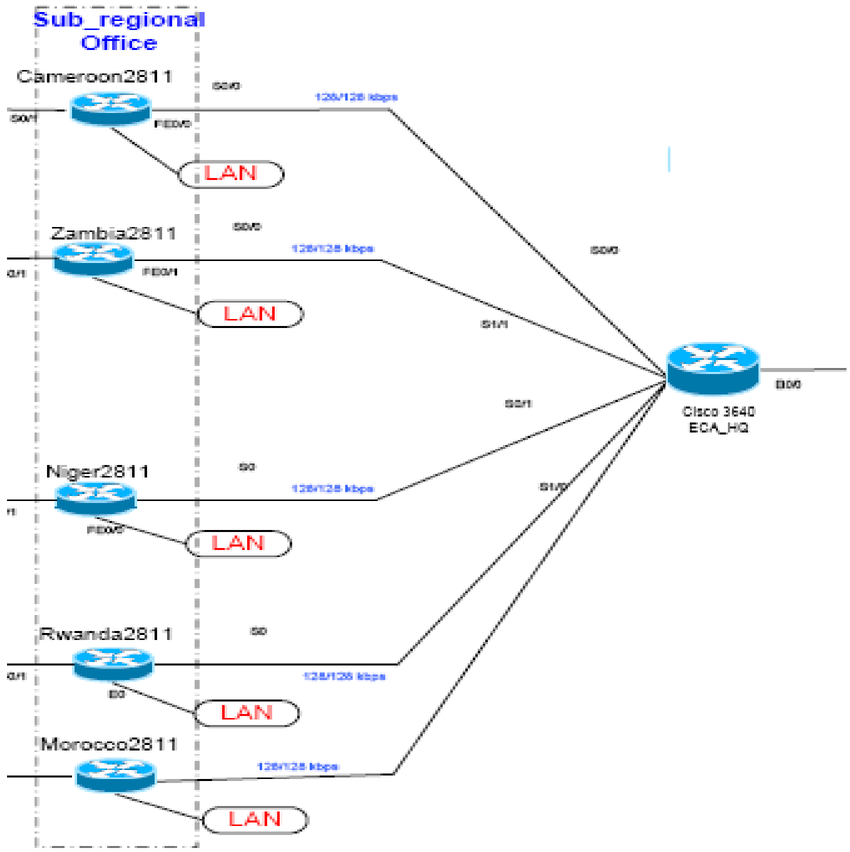


Figure 3.1 UNECA WAN Architecture

Thus, this experiment uses a modeling approach using the OPNET Modeler tool. Basically, the approach has three major phases, data collection, modeling the experimental-environment and data analysis. After collecting data from the Actual WAN environment, the data is used to realistically simulate the environment by OPNET Modeler tool and then various traffic flow scenarios are simulated, which represents the actual environment ‘s WAN conditions, finally the original and simulated traffic flows are analyzed using the OPNET tool. Various configuration and traffic data are collected to make the modeled WAN environment, a realistic representation of the actual UNECA WAN Environment.

3.2.1 Tools and Performance Metrics used

Tools and Performance metrics that are described in the methodology section, chapter 1, is used for experimentation. Mainly, Net-Flow analyzer v9 is used in the data collection phase and

OPNET Modeler tool is used in almost all phases of this study, including model design, simulation, data collection, and data analysis. As clearly presented in chapter 1 (Methodology section), in this experiment, response time is used as a performance metric for data applications (Email, Web and database applications) and Packet end to end delay measured in seconds is a primary performance metric for multimedia applications (video conferencing applications and voice applications/telephone systems).

3.2.2 Data collection and understanding

This section discusses the data collection phase and the preliminary data analysis, for understanding the collected data. Understanding the traffic data helps to determine the major performance bottlenecks that affects application performances. However, the level of performance influence by these and other potential factors will be experimented in the analysis phase to achieve the objective of the study.

3.2.2.1 Data collection

Data is collected about various features of the UNECA WAN environment, which include existing WAN topology configuration, traffic flow statistics, applications' bandwidth usage statistics and end users experience report for all SROs WAN links.

Configuration information of WAN devices is generated and copied to a text file by running a series of commands, which are listed below, on each routers of UNECA WAN environment.

```
term leng 0
show running-config
show version,
show interfaces,
show cdp neighbors detail
show mpls traffic-eng tunnels
```

The output text file holds sufficient information to represent the UNECA WAN environment. The text file contains information regarding type of WAN devices, type of interfaces, each interface's configurations, connectivity to neighbors, tunnel type, WAN technology used, the bandwidth allocated, the routing protocol used and many other details that are captured using show commands. This output text file will be given to OPNET Modeler to model the WAN topology for experimental test bed setup. Part of configuration file is shown in Table 3.1 below.

Interface Type	Option	Value
VPN	Interface	Tunnel26/
Serial	Interface	Serial0/1/1.1 point-to-point
Serial	Description	Connection to SRO-1
Serial	IP address	IP address Subnet mask
Serial	IP address	IP address Subnet mask (secondary)
VPN	Tunnel source	HQ Router IP address
VPN	Tunnel destination	SRO Router IP address
VPN	Keepalive	10 3
VPN	IP	Virtual-reassembly
Serial	Encapsulation	Frame-relay
Serial	Frame-relay interface	dlci 100
Serial	Class	mlp576
Serial	Bandwidth	256 Kbs

Table 3.1 Part of configuration file for WAN topology

Traffic flow data is collected from HQ router, which is the HUB for all SRO WANS and both the incoming (WAN IN) and outgoing (WAN OUT) interfaces of the router are configured to send traffic to Net-Flow Analyzer. The tool is configured to collect the traffic flow and application flow statistics of UNECA WAN for three months of period and the statistics summary is presented in the next section.

3.2.2.2 Data Understanding

The collected data is used for data understanding and preliminary study to determine the major performance bottlenecks that affect the enterprise applications' performances. The collected statistics data summary is presented in two ways, which are WAN links overall traffic flow statistics and application distribution in the traffic flows.

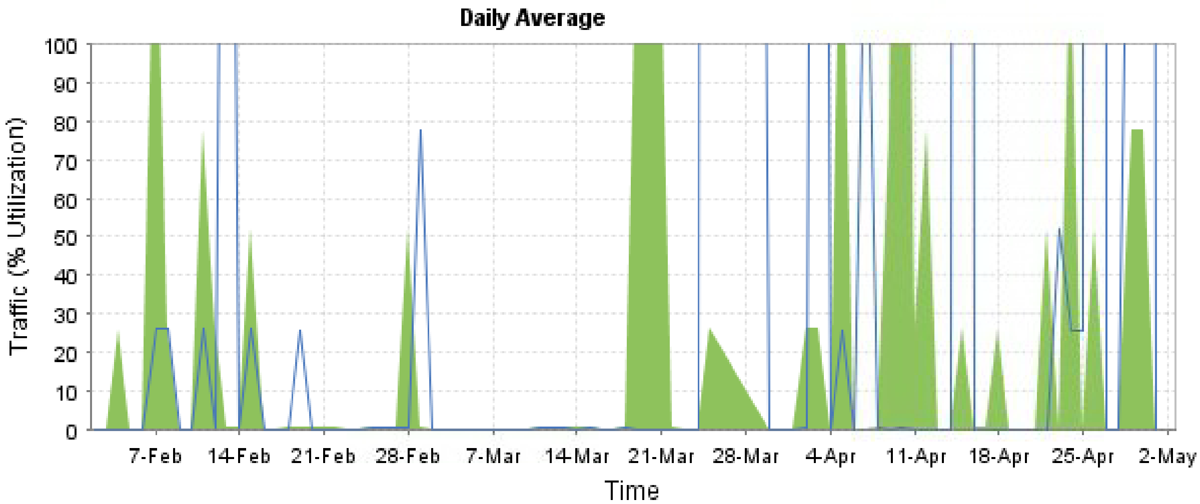


Figure 3.2 Daily average Traffic Utilization of SRO WAN Link

As shown above, Figures 3.2 and 3.3 present the overall traffic flow statistics, which are collected from SRO hub router, located in HQ data center. The graph in green indicates “Traffic IN”, that is a traffic coming to the interface and the blue color is for “Traffic OUT”, that is traffic going out of the interface. The traffic utilization percentage by SRO WAN link is shown by taking daily and weekly average of the three months period. The Traffic utilization reaches in some points (most of them are beginning of the Week) to 100% or more. These WAN links overutilization explains the observations that I have mentioned in section 3.1.2, poor end users experience when accessing applications from HQ.

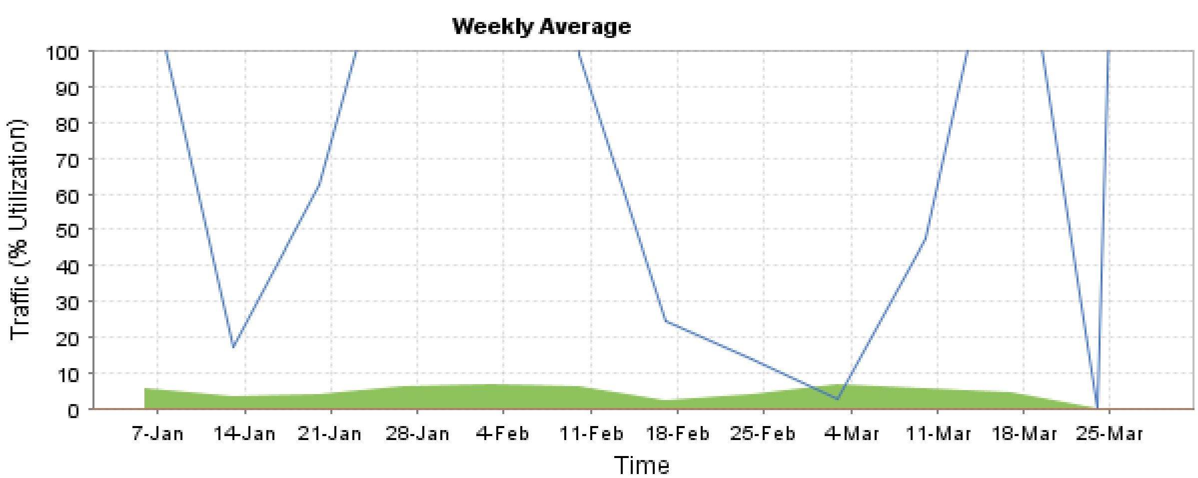


Figure 3.3 Weekly average Traffic Utilization of SRO WAN Link

* Utilization is calculated with Link Speed of 1.0 Mbps

Category	Total	Max		Min		Avg	Standard Deviation	95th Percentile
		1 min	Weekly	1 min	Weekly			
IN	40.81 GB	27.28%	6.41%	0.00%	0.08%	4.50%	2.22	-
OUT	1.55 TB	858993.87%	795.52%	0.00%	0.05%	171.49%	242.87	-

Table 3.2 SRO WAN Link Traffic IN and Traffic out statistics in terms of utilization

The traffic IN, which is a traffic flow in to the HQ data center from SRO users and Traffic Out, which is a traffic flow to the SRO users from HQ data center, is not proportional as shown in Table 3.2. The utilization is calculated using 1Mbps, which is the highest speed from the five SROs WAN link.

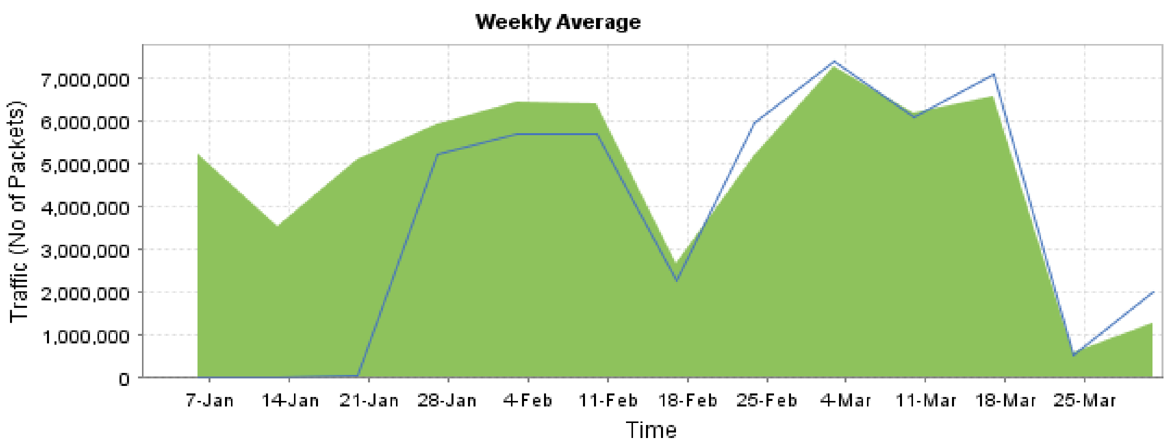
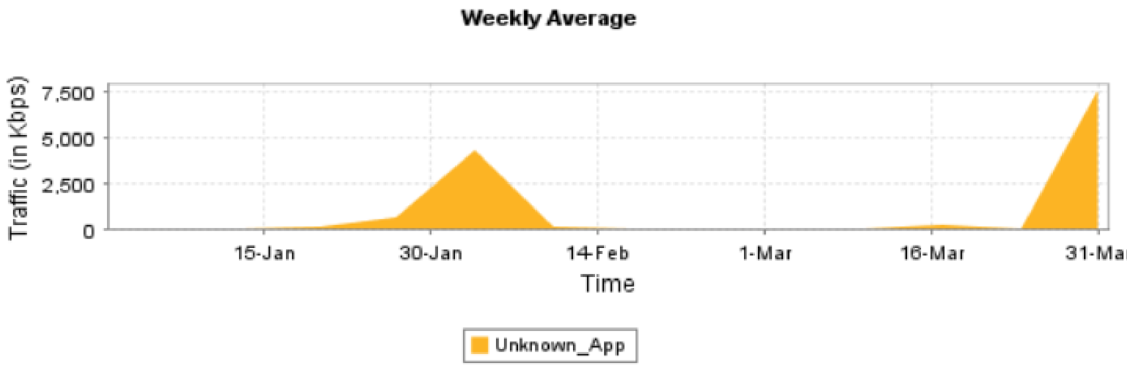


Figure 3.4 Weekly average of traffic flow in no of packets for an SRO WAN link

Category	Total	Max		Min		Avg	Standard Deviation	95th Percentile
		1 min	Weekly	1 min	Weekly			
IN	6.2312972E7	7258067.0	7258067.00	559950.0	559950.00	5192747.67	2173843	-
OUT	4.8026739E7	7413557.0	7413557.00	3997.0	3997.00	4002228.25	2934067	-

Table 3.3 SRO WAN Link Traffic IN and Traffic out statistics in terms of no of packets

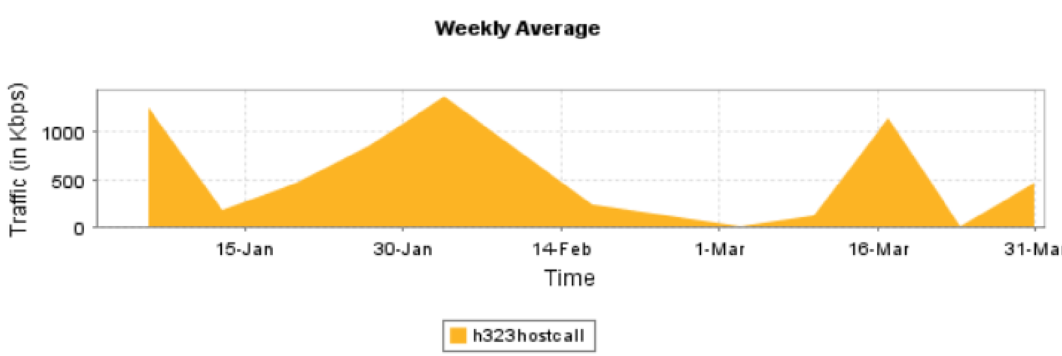
Fig 3.4 presents traffic flow statistics of the WAN links in terms of No of packets used, by taking a weekly average from the three months' traffic data. Table 3.3 shows No of packets used for both IN and OUT traffic flow. The fact derived from data in Table 3.2, which is the unbalanced Traffic IN and Traffic OUT distribution, is also applied here except that the average no of packets for Traffic IN flow, is higher.



ApplicationOUT	Traffic	Percentage of total traffic
Unknown_App	980.38 GB	63 %

Figure 3.5 Traffic utilization by Unknown applications

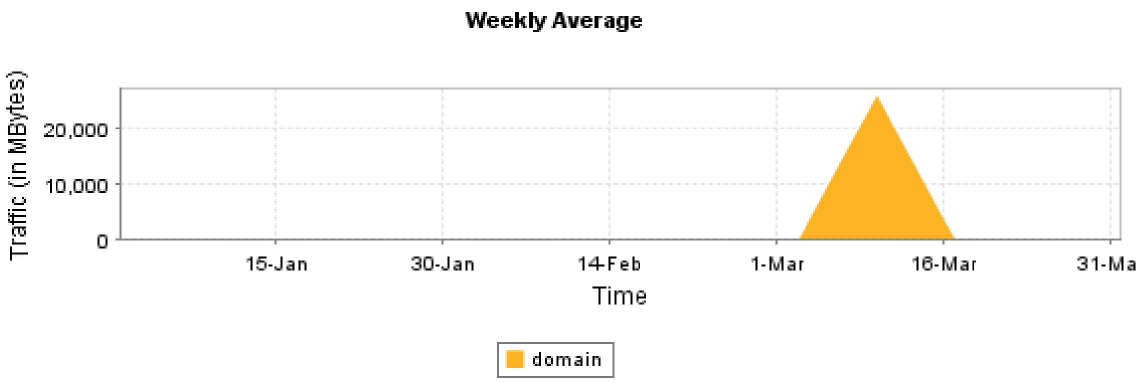
Net-Flow analyzer also takes a snapshot of all applications traffic distribution in every traffic flows through each WAN link. The top four applications by their volume contributions are selected and a statistics summary of their distribution in each SRO WAN link is presented. Figure 3.5 shows Unknown applications traffic volume in the traffic flow captured for a three-month period. The tool, categorized applications as unknown if the application does not use a well-known TCP or UDP protocol which is registered in Internet assigned numbers authority (IANA). The critical applications that we have selected for this experiment uses a registered port and are not part of this applications category. Critical applications usage will be described in the coming parts by comparing with unknown applications traffic utilization.



ApplicationOUT	Traffic	Percentage of total traffic
h323hostcall	523.99 GB	34 %

Figure 3.6 Traffic utilization by Video and voice applications

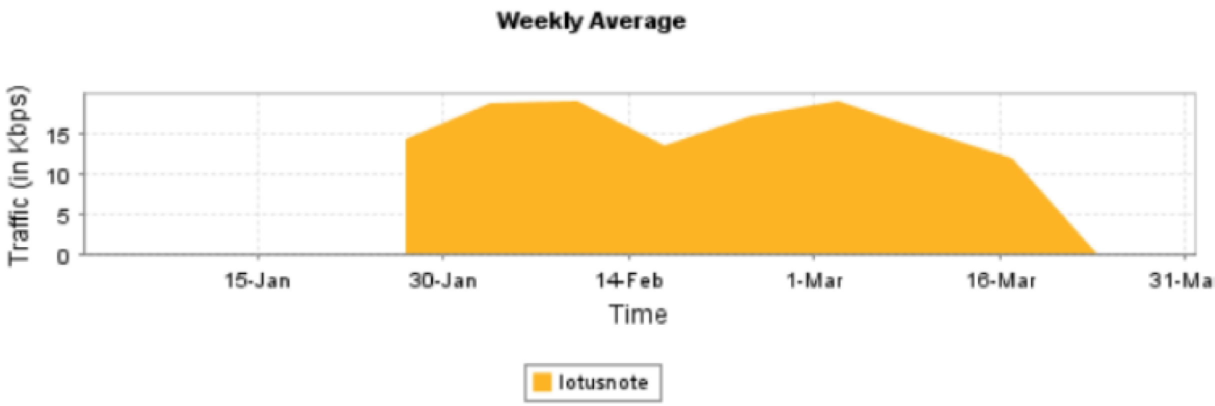
Unknown applications contribute 63% of the total traffic flow, which is 980.38GB from the total Traffic OUT volume, 1.55TB. However, the distribution is not uniform; it is very high during certain periods and very low in another time. These kind of issues are investigated in the analysis phase of the experiment, the scope of this phase is to understand the data from its distribution.



ApplicationOUT	Traffic	Percentage of total traffic
domain	26.22 GB	2 %

Figure 3.7 Traffic utilization by Domain applications

H323hostcall is an application protocol used for Video conferencing and voice applications. As show in Figure 3.6, multimedia applications contribute 34% of the traffic flow, which is nearly half of the unknown applications utilization. Figure 3.7 represent domain applications traffic volume. Domain applications are those applications that are responsible for domain name system (DNS) service. DNS servers are deployed in each SRO, to provide name resolution services: these servers are configured as a slave DNS and configured to pull data from master, which is in the HQ. The domain traffic, caused by data replication between the slave and master DNS servers, constitutes 2% of the traffic flow. This traffic is the third application category in terms of utilization and it is not member of the list of critical applications selected for the experiment. These kind of traffics are taken as a potential factor to be investigated further in the experimentation phase for a source of critical applications performance problem.

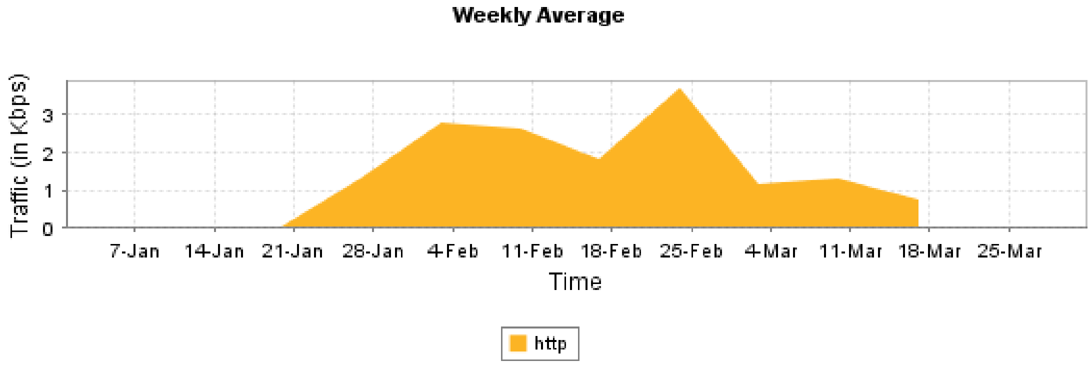


ApplicationOUT	Traffic	Percentage of total traffic
lotusnote	9.68 GB	1 %

Figure 3.8 Traffic utilization by email application

The utilization graph in Figure 3.7 indicates that the domain traffic is heavily used for short period as the graph indicates. Apparently this kind of application flow statistics is helpful for analyzing, the performance impacts of traffics having unique properties. Thus, the effect of this domain traffic is examined in the analysis phase.

Email application is the other selected critical application for performance analysis and benchmarking. The study target organization, deployed Lotus note for email application and all sub-regional and HQ clients rely on this email application for communication and collaboration. Figure 3.8 depicts that lotus notes applications traffic payload is only 1% of the total traffic flow.



ApplicationOUT	Traffic	Percentage of total traffic
http	1.15 GB	<1%

Figure 3.9 Traffic utilization by Web applications

Application	DSCP	Traffic
https	Default	245.68 MB

Figure 3.10 IMIS application traffic usage

There are some major web applications deployed in the data center and supposed to be accessed frequently by users from all SROs with HTTP protocol. The volume of this web application is less than 1% of the total traffic (Figure 3.9). IMIS, which is a multitier database application (UNECA's ERP application), uses https protocol. Unlike other critical applications, it is used by few end users from each SRO. However, the application is used intensively regardless of the number of users' accessing it. Figure 3.10 shows this application takes only 245MB from the total traffic volume and it is less than 0.02%.

The remaining portion of this chapter introduce the configuration of the experimental test bed using OPNET Modeler and the traffic flow analysis that is done to determine the performance impact of various WAN conditions on each critical applications.

3.2.3 Experimental Environment Setup (Modeling WAN Traffic)

In this phase, the experimental test bed is prepared by modeling the existing UNECA WAN environment using OPNET Modeler tool. Modeling is done by importing WAN Network configuration and applications' traffic flow, which are collected in the previous phase. Average Response time of applications, throughput of WAN links and packet end to end delay measure is taken from the modeled UNECA WAN to check the model validity. The modeled environment is the representation of UNECA WAN and it includes the following components.

WAN Topology Model: Device Configuration Imports (DCI) module of OPNET Modeler is used to generate this model. The topology model is created by importing the device configuration file as shown in Figure 3.11. The configuration text file, which is prepared in the data collection phase, is selected as source file for this procedure. The missed configuration information is indicated by the model assistant and is configured manually after the model is generated. The network model summary report is generated after the import process, which shows total files imported and devices created (please refer Appendix III for this report).

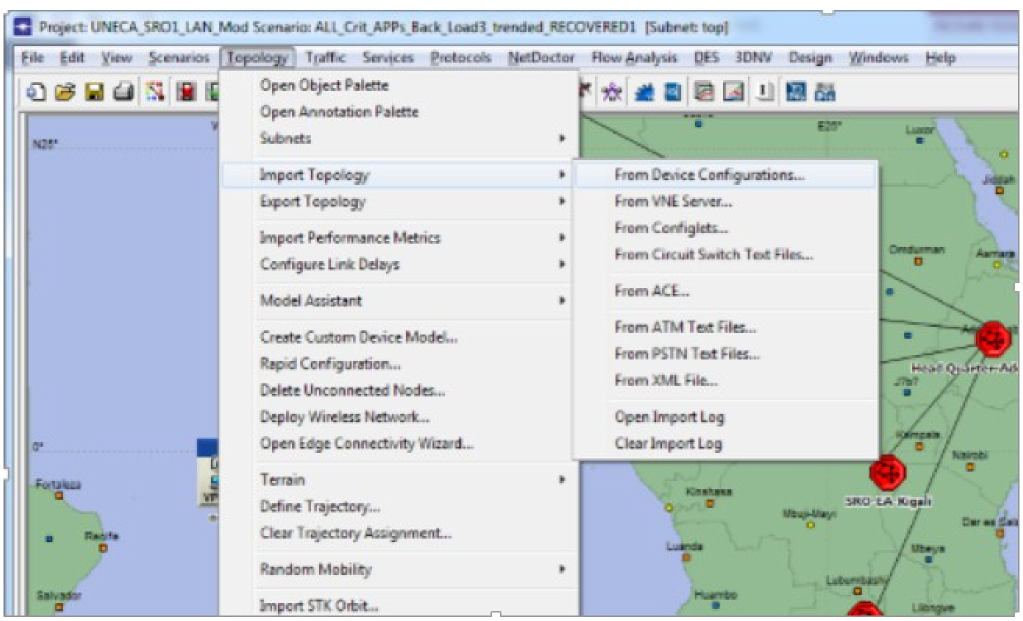


Figure 3.11 Importing Configuration data for Topology

Traffic flow Model: is the second model that completes the experimental environment setup. The model is created by importing the traffic flow that is collected from the target WAN environment.

Traffic center is used to import traffic flows collected by Net-Flow analyzer as shown in Figure 3.12.

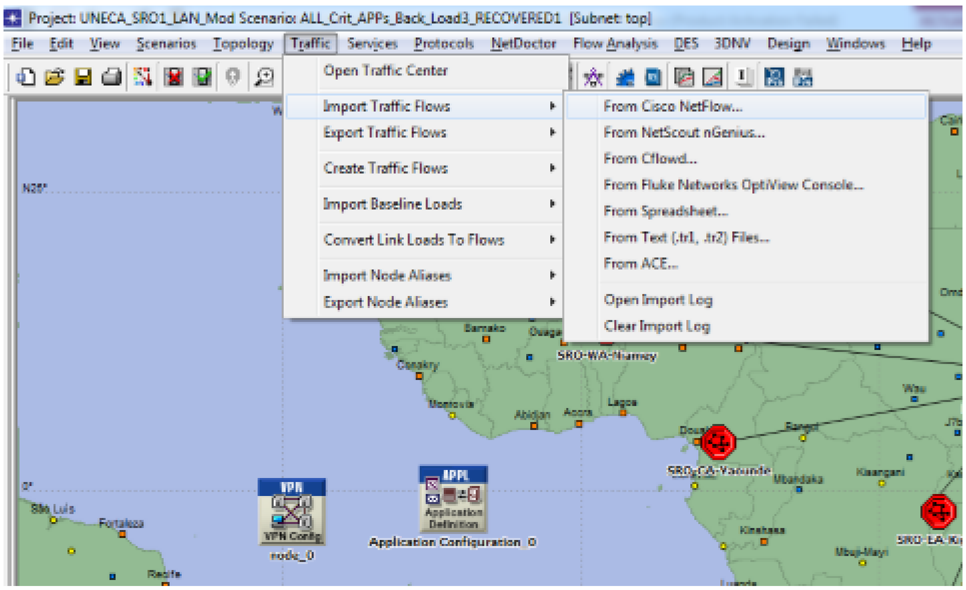


Figure 3.12 Traffic center: for importing traffic from Net-Flow

Traffic flow is a variable condition in the WAN comparing to topology, which do not change frequently unless there is a planned upgrade. Thus, various traffic flow scenario is also simulated and modeled from the actual traffic flow depending on the collected traffic statistics distribution. These simulated traffic flow models are used for further analysis to identify the behavior of critical applications and their source of performance problems.

3.2.3.1 WAN Simulation Topology

OPNET Modeler simulated the topology of UNECA WAN architecture as illustrated in Figure 3.13 from the imported router configuration data. Figure 3.14 shows part of the topology detail of the HQ data center. The topology represents a typical WAN Environment that connects SROs LANs to HQ through WAN. LANs, routers, switches, application nodes and links are depicted in this topology.

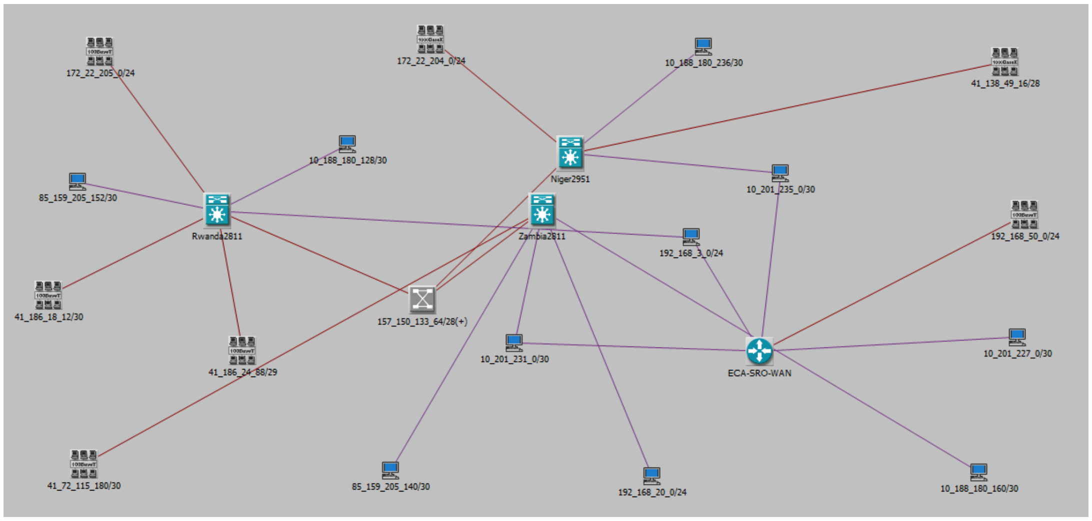


Figure 3.13 UNECA WAN simulated topology (OPNET Modeler)

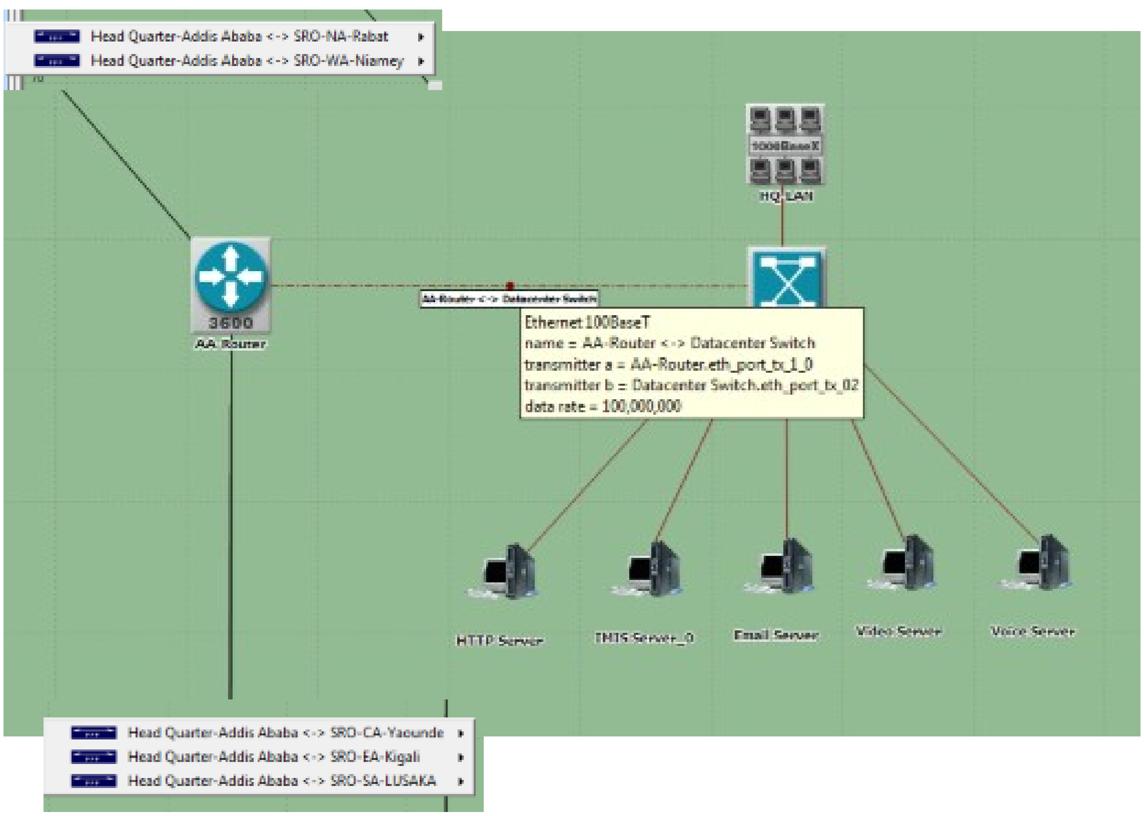


Figure 3.14 HQ link simulated topology (OPNET Modeler)

3.2.3.2 Modeling Actual Traffic Flows and Scenario Design

The traffic flow data prepared in the Data collection phase is used in this modeling phase in two ways 1) it will be modeled as it is, so that later the traffic will be analyzed 2) to create various traffic flows so that the analysis phase will get more analytical power for problem identification. The analysis phase uses these scenarios for narrowing down the causes of each critical applications' performance problem, depending on the collected traffic statistics distribution.

3.2.4 Data Analysis

Since the WAN topology and traffic flow is modeled, the environment is now ready for diagnosing applications performance problems and exploring various performance fixes. AppDoctor module of the OPNET tool is used in this phase to:

- Pinpoint network and application bottlenecks
- Diagnose application problems
- Explore proposed fixes to existing applications
- Investigate application performance under varying configurations and network conditions

AppDoctor automates the process of application troubleshooting, finds the major components of delay, and determines the root cause of application performance problems. Diagnosis tests the transactions in the traffic flow against issues that often cause performance problems in network-based applications, grouped by category. Values that cross a specified threshold are marked as bottlenecks or potential bottlenecks. Table 3.4 shows the major bottlenecks reported by the AppDoctor Diagnosis that causes poor application performance.

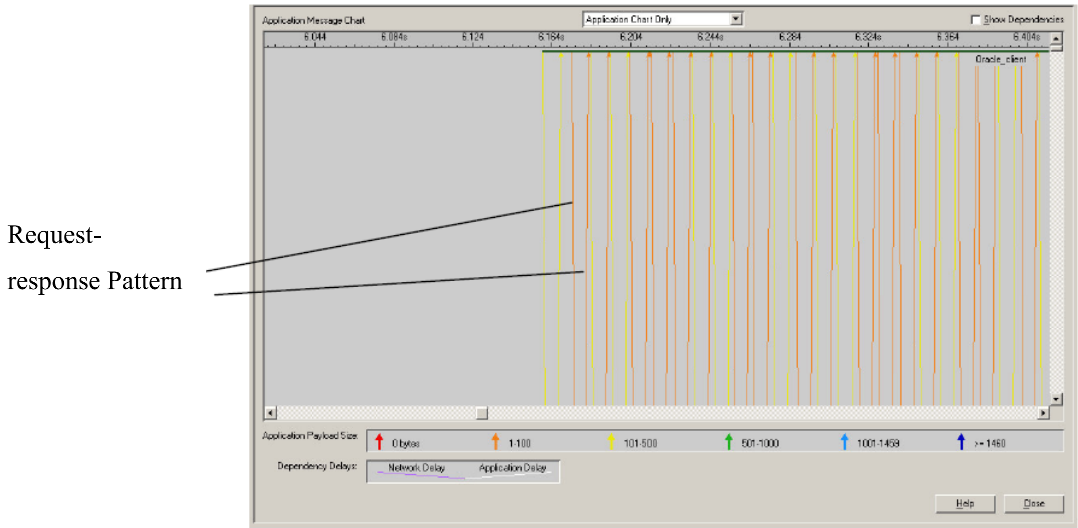
	Total	Client <-> Server
Protocol Overhead	Bottleneck	Bottleneck
Chattiness	No Bottleneck	No Bottleneck
Network Cost of Chattiness	No Bottleneck	No Bottleneck
Propagation Delay	Bottleneck	Bottleneck
Transmission Delay	Bottleneck	Bottleneck
Protocol/Congestion Delay	Bottleneck	Bottleneck
Connection Resets	No Bottleneck	No Bottleneck
Retransmissions	Bottleneck	Bottleneck
Out of Sequence Packets	No Bottleneck	No Bottleneck
TCP Windowing (A -> B)	Not Applicable	No Bottleneck

Table 3.4 Application bottlenecks reported by the AppDoctor Diagnosis.

The Diagnosis result with a detailed description of the reported five bottlenecks and suggestions for correcting each bottleneck is described below.

- **Protocol overhead**: Large amounts of protocol overhead are increasing the utilization of your network. Threshold: 30.0%, Value: 34.0% - a lower value is better. Make your application send fewer, larger application messages. This will utilize network resources more efficiently.
- **Propagation Delay**: is contributing significantly to the application response time. Threshold: 30.0%, Value: 31.4% - a lower value is better. Consider moving the affected tiers physically closer together and/or if the application has a "Chattiness" bottleneck, addressing the application's chattiness
- **Transmission delay**: is contributing significantly to the application response time. Threshold: 30.0%, Value: 32.5% - a lower value is better. Consider using a faster link or sending less data.
- **Protocol/congestion delay**: is contributing significantly to the application response time. Threshold: 30.0%, Value: 68.5% - a lower value is better. Consider changing the parameters of your transport protocol and/or upgrading the bandwidth of your network.
- **Retransmissions**: there are many packet retransmissions. The network may be heavily congested, or there may be an error-prone link. Threshold: 3.0%, Value: 4.1% - a lower value is better.

AppDoctor’s Summary of Delays has revealed issues with both the network and applications. The Diagnosis function of AppDoctor also provides a detailed analysis of the bottlenecks for further



insight into the cause of this delay.

Figure 3.15 Data Exchange Chart

The Data Exchange Chart shown in Figure 3.15 provides additional insight in to various causes. The chart shows part of individual application messages flow: the arrowhead indicates the direction of the flow and the colors represent applications packet size.

This application flow chart shows that some applications transaction uses many small size messages (as indicated by the orange and yellow colors) and it seems to repeat a simple request and response pattern with many application turns. Application turn is each change of direction by the application as the direction of data flow changes.

The AppDoctor Diagnosis is done on traffic flow that is captured from target organization WAN. Since imported traffic flow is not filtered, it contains traffics generated by all critical applications, (Video, Voice, Email, database, Http) and other non-critical applications. Hence, the result of diagnosis shown above represents the major causes of performance problem in the WAN environment as a whole. However, the impact of these bottlenecks on each application performance should be determined to achieve the objective of the study.

To investigate the impact of reported bottlenecks on each applications, new scenario (scenario1 shown below) is designed that contains only one critical application traffic at a time, which is extracted from original captured traffic flow. Packet filters are useful components of OPNET for filtering out traffic of interest from imported traffic data. Packet filtering is done based on host, ports or protocols. Since the host address, transport protocol and application port of each critical applications server is known, the traffic flow for each application is extracted by filtering packets using these parameters.

Scenario 1: All critical data applications are modeled alone so that performance diagnosis is executed on each applications traffic flow to test application's transactions against issues that often cause performance problems in network-based applications.

Additional traffic flow scenarios that are listed below are designed, to analyze further the performance of each application under reported bottlenecks. The scenarios are also used to evaluate the impact of solutions suggested by AppDoctor for correcting each bottleneck.

Scenario 2: This scenario is designed to examine the effect of congestion, which is one of the reported bottleneck, on each application performance.

Each application traffic flow with and without background traffic load is modeled. It is designed, by changing the traffic load parameter in traffic information menu from the link configuration window shown in Figure 3.16.

Scenario 3: This scenario is designed to examine the effect of propagation delay, which is one of the reported bottleneck, on each application performance. In addition, the solutions suggested by AppDoctor for propagation delay are evaluated in this scenario.

It is designed, by changing the delay parameter from the link configuration window shown in Figure 3.16.

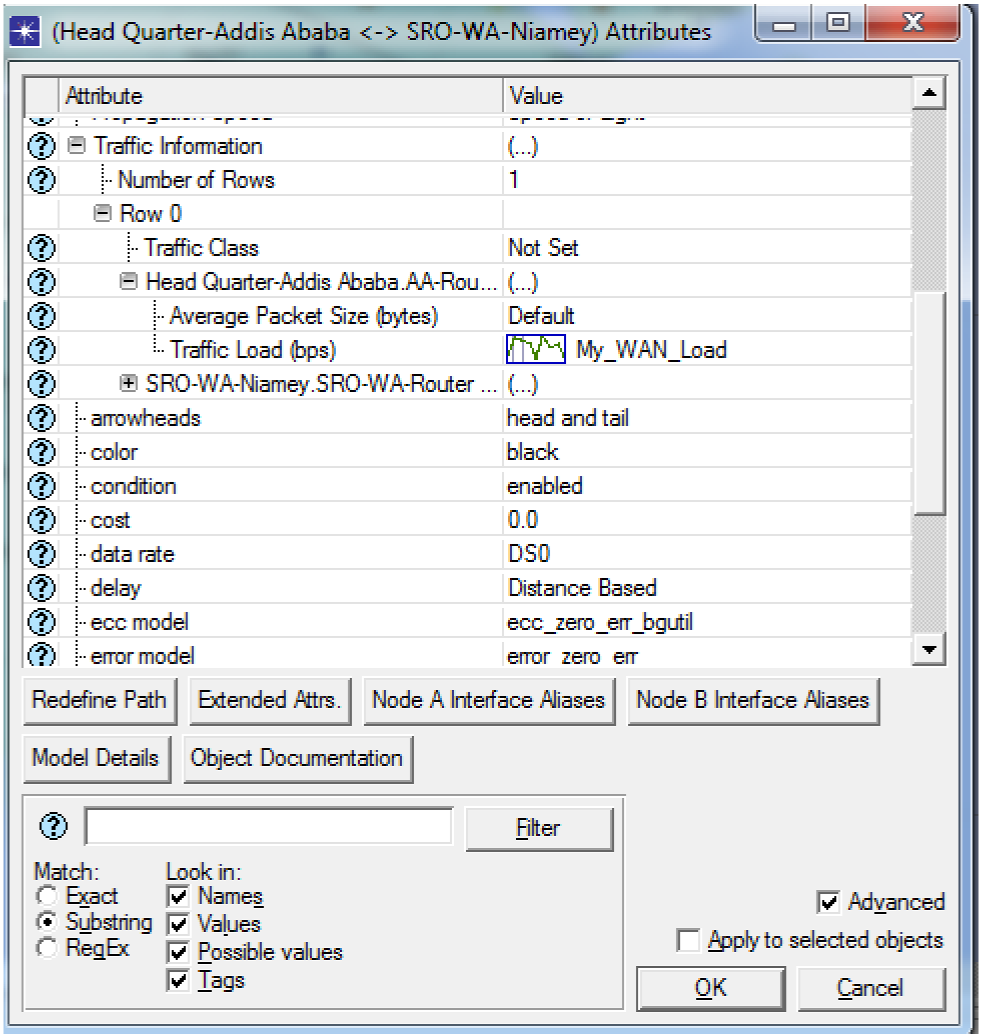


Figure 3.16 HQ to SRO-WA WAN link configuration

Scenario 3: This scenario is designed to examine the effect of transmission delay, which is one of the reported bottleneck, on each application performance. In addition, the solutions suggested by AppDoctor for transmission delay are evaluated in this scenario.

It is designed, by changing the data rate parameter from the link configuration window shown in Figure 3.16.

Scenario 4: This scenario is designed to examine the effect of transport protocol delay, which is one of the reported bottleneck, on each application performance. In addition, the solutions suggested by AppDoctor for protocol delay are evaluated in this scenario.

It is designed, by modeling only critical data applications without video and voice applications, (since only data applications use TCP)

Scenario 6: This scenario is designed to examine the effect of other reported delays, on each application performance.

It is designed, by modeling different combination of critical applications at one time, to determine the impact of one application on the other.

Using OPNET Modeler all simulations are executed one by one and then the traffic flow is analyzed. Critical applications performances are evaluated with each scenarios designed above. OPNET collects the analysis result data regarding impact in performance of each critical application. The analysis result collected is presented using various parameters, which are suitable for interpretations and determining the impact level of each WAN conditions on critical applications performance.

OPNET modeler tool uses response time and packet end-to-end delay for analyzing applications performance. Therefore, these performance metrics are primarily used in this research to evaluate the effect of bottlenecks and various optimization techniques on applications performance.

The major experimental analysis results after analyzing the traffics on all scenarios are summarized and discussed in the next chapter.

CHAPTER FOUR

RESULTS AND DISCUSSION

This chapter is the core part of the thesis, since it presents and discusses the major analysis results of the experimentation. Key issues, regarding the performance challenges of critical applications are explained in terms of various WAN factors.

According to the experimental analysis result, applications' performance requirements vary based on their type of protocol. SMTP a protocol for email behaves differently than HTTP a protocol for web in different WAN situations. Latency may affect one application significantly than others, or application that does not require low latency may require low link error rate. It is therefore important to know the impacts of WAN factors on application performance and on their supporting end-to-end protocols.

This chapter discusses the analysis result provided by OPNET after diagnosing all critical application performance using various scenarios designed in 3.2.4. The first section discusses the experimental results, which evaluate the impacts of TCP and layer seven protocol on application performance. The second section is about the impact of WAN links end-to-end delay (propagation delay), packet loss and bandwidth, then the behavior and requirement of each critical application when sharing WAN links is presented in the third section. Finally, the analysis result will be presented which examines the impact of non-critical applications (background traffic) bandwidth usage and protocol overhead on end users critical application response time.

4.1 Impacts of TCP

Tables below presents impacts of TCP on applications performance using TCP traffic analysis. Table 4.1, Table 4.2 and Table 4.3 shows TCP statistic's data for web, email and DB applications.

Statistic	Average	Maximum	Minimum
TCP Delay (sec)	0.8776	8.8240	0.0391
TCP Retransmission Count	117.67	272.00	79.00
TCP Segment Delay (sec)	0.6517	1.8444	0.0453

Table 4.1 Traffic flow with Http application only

Statistic	Average	Maximum	Minimum
TCP Delay (sec)	10.358	89.239	0.054
TCP Retransmission Count	125.15	354.00	86.00
TCP Segment Delay (sec)	1.1733	4.7161	0.0270

Table 4.2 Traffic flow with Email application only

Statistic	Average	Maximum	Minimum
TCP Delay (sec)	18.797	71.530	0.047
TCP Retransmission Count	160.05	386.00	92.00
TCP Segment Delay (sec)	0.41342	0.60196	0.07355

Table 4.3 Traffic flow with database application only

HTTP traffic has lower TCP connection delay and lower TCP retransmission count than the other two. Email traffic has lower TCP connection delay and lower TCP retransmission count than database. Database traffic is the highest in TCP delay and retransmission but the lowest in TCP segment delay.

Figures 4.1 (A) and 4.1 (B) show how the TCP connection delays of each application when sharing the same WAN link at a time. Figure 4.1(A) is the analysis result taken from the scenario, which is designed using with a traffic flow that contains all critical application.

TCP Delay (sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
IMIS Server_0	0.45456	0.45456
HTTP Server	0.07848	0.08803

A) When Traffic flow contains all applications traffic

TCP Delay (sec)

Statistic sampling period is 18 seconds.

Sort By Node	Sorted By Average	Sort By Peak
Email Server	10.683	129.86
IMIS Server_0	8.589	31.09
HTTP Server	1.724	20.52

B) When Traffic flow contains all applications traffic except video

Figure 4.1 TCP delay (sec)

As the result shows email application has reported no TCP delay since the application make no single TCP connection or no email traffic is received by the users in this scenario (Figure 4.2). Http also reported the lowest TCP delay. Figure 4.1(B) is the analysis taken from the scenario traffic flow without multimedia applications. Similarity in this case email application reported the highest TCP delay and IMIS the second highest. This result shows that TCP segment delay and TCP retransmission count has impact on applications TCP connection time.

Email application TCP segment latency (delay) is high as presented in Table 4.2 and this application reported no TCP connection or high latency TCP connection that results in no email traffic being received by any user (in a case scenario) as indicated in Figure 4.2. Thus, this high email segment delay is one of the factor for email applications performance problem. IMIS (database application) on the other hand has high TCP retransmission count which results in poor database application performance.

Client Email Traffic Received (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-CA-Yaounde.SRO-LAN	0	0
SRO-EA-Kigali.SRO-EA-LAN	0	0
SRO-NA-Rabat.LAN	0	0
SRO-SA-LUSAKA.SRO-SA-LAN	0	0
SRO-WA-Niamey.-LAN	0	0

Figure 4.2 Email applications traffic receiving rate in packets/sec (during link congestion)

TCP delay and Optimization Techniques

The above experimental results show that TCP segment delay and TCP retransmission count have an impact on applications performance. Email has high TCP segment delay and Database application on the other hand has high TCP retransmission count. Which explains both applications performance is affected by TCP delay.

TCP is reducing the rate at which the application can send data because of congestion control, flow control, send window sizes, and/or receive window sizes. When an application is sending bulk data over a high-bandwidth and high- latency network, TCP window sizes must be large enough to permit TCP to send many packets in a row without having to wait for TCP ACKs. TCP delay is caused by two factors 1) TCP segment delay: If TCP window size is small, when an application is sending bulk data over a high-bandwidth and high- latency network, 2) High TCP retransmission count: when there is congestion and TCP flow control mechanism is affecting communication (OPNET, 2008). Various solution suggested by OPNET AppDoctor is experimented and the techniques listed below provides better result.

	TCP delay	Email Download Response Time (sec)
Node	36.264	24.78
Email Server	16.594	22.64
	10.68	5.34

Table 4.4 Effect of TCP Delay on email response time

	TCP delay	Database entry Response Time (sec)
Node	36.326	33.18
DB Server	18.88	10.85
	8.58	1.84

Table 4.5 Effect of TCP Delay on Database response time

As shown in Table 4.4 and 4.5 by decreasing the TCP delay, it is possible to enhance the performance of email and database applications. For email application, TCP delay is controlled by increasing TCP window size of the application server whereas for database application by enabling TCP extensions such as "Fast Retransmit" and "Fast Recover" it is possible to limit the impact of TCP delay. These settings can be modified on the server nodes of the modeled environment by changing different application parameters as required.

SMTP delay and Optimization Techniques

The analysis result on impacts of TCP shows that when all applications are using the WAN and makes the link congested, email application cannot send and receive packets unlike other experimented critical applications, as shown in Figure 4.2. The Data Exchange Chart in section 3.2.4 notified that some applications are sending many small size application segments. The following discussion is based on these two facts.

According to Stevens (2001), TCP is a full-duplex protocol, yet SMTP (email application) uses TCP in a half-duplex fashion. The client sends a command then stops and waits for the reply. The client send multiple small size commands like HELO, MAIL, RCPT, DATA, and QUIT commands before sending the body data. Moreover, this half-duplex operation of SMTP fools the slow start mechanism when the network is running near capacity.

Some applications generate series of small segments, since each packet has a header, a 1-byte small segment application will become 41-byte packets: 20 bytes for the IP header, 20 bytes for the TCP header. These small packets (called tiny-grams) are normally not a problem in LANs, since most LANs are not congested, but these tiny-grams can add to congestion on WANs. A simple and elegant solution was proposed in RFC 896 (Nagle, 1984), called the Nagle algorithm. This algorithm says that no additional small segments can be sent until the acknowledgment is received. However, there are times when the Nagle algorithm needs to be turned off, like for applications that require small messages to be delivered without delay in order to work properly (Stevens, 2001).

The above discussion explains the question, why email application (using SMTP) receives no packet when link is congested, which is because of Nagle's algorithm (since SMTP has small segment commands) and SMTP's half-duplex operation over TCP. Hence, by disabling Nagle's algorithm and forcing SMTP to use TCP slow start algorithm, improvement on email performance is achieved, as described in table 4.6, during congestion for the case considered in the study.

	Nagle's algorithm	TCP slow start algorithm	Average Email traffic Received (packet/sec)
Node	enabled	disabled	0.0000
Email Server	disabled	disabled	0.02250
	disabled	enabled	0.10889

Table 4.6 Effect of SMTP operation on email response time

4.2 Impacts of End-to-end delay

End to end delay is the sum of packet transmission delay and propagation delay. It is the result of physical distance between WAN nodes, propagation speed of WAN link and the WAN link rate (in bits per sec). Table 4.7 presents average packet transmission delay of Each SRO WAN links (Column 2). In the experiment packet's point-to-point queue delay is found to be variable depending on the application type in the traffic flow. Therefore, packets point to point queue delay and its performance impacts are discussed below.

Demand Name	Average Bits Per Sec	Average Delay (msec)	End to End Jitter (msec)	Maximum Packet Loss (%)	Volume
AA-Router-->SRO-SA-LA	553,704	648.434	180.133	88.886	19.802 MB
AA-Router-->SRO-EA-LA	553,704	641.558	180.133	88.886	19.802 MB
AA-Router-->SRO-WA-LA	553,704	653.873	180.133	88.886	19.802 MB
AA-Router-->SRO-SA-Ro	553,704	648.419	180.133	88.886	19.802 MB
AA-Router-->SRO-EA-Ro	553,704	641.543	180.133	88.886	19.802 MB
AA-Router-->SRO-WA-Ro	553,704	653.858	180.133	88.886	19.802 MB
AA-Router-->SRO-CA-LA	553,704	173.695	45.033	55.544	19.802 MB
AA-Router-->SRO-CA-Ro	553,704	173.68	45.033	55.544	19.802 MB
AA-Router-->SRO-NA-LA	553,704	108.171	23.057	13.172	19.802 MB
AA-Router-->SRO-NA-Ro	553,704	108.156	23.057	13.172	19.802 MB

Table 4.7 End-to-End delay for each WAN links

As shown in Table 4.7 SRO-WA, SRO-SA and SRO-EA has long end-to-end delay respectively and a result in Figure 4.3(A) describes, HQ to SRO-SA link has long point-to-point queue delay when email application is used. Further analysis done on SRO-SA link determines that its router transmit with high rate than others. When the router sends traffic with a high rate, and since the link does not match the rate, the traffic will be queued.

point-to-point queuing delay (sec)

Statistic sampling period is 36 seconds.

Sort By Link	Sorted By Average	Sort By Peak
Head Quarter-Addis Ababa <-> SRO-SA-LUSAKA [0] <--	16.190	26.723
Head Quarter-Addis Ababa <-> SRO-EA-Kigali [0] <--	16.167	25.101
Head Quarter-Addis Ababa <-> SRO-WA-Niamey [0] <--	15.937	24.442
Head Quarter-Addis Ababa <-> SRO-SA-LUSAKA [0] -->	0.081	0.194
Head Quarter-Addis Ababa <-> SRO-WA-Niamey [0] -->	0.079	0.235

A) Point to point queue delay of each SRO WAN links

Client Email Download Response Time (sec)

Statistic sampling period is 36 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-SA-LUSAKA.SRO-SA-LAN	213.96	480.08
SRO-WA-Niamey.-LAN	146.08	256.71
SRO-EA-Kigali.SRO-EA-LAN	1.55	1.55
SRO-CA-Yaounde.SRO-LAN	0.23	0.40
SRO-NA-Rabat.LAN	0.20	0.25

B) Email response time on each SRO WANs

Figure 4.3 Analysis result when email application is used alone in the traffic flow

Figure 4.3(B) shows the email downloading response time for end users in each SROs. SRO-SA and SRO-WA end users experience poor email downloading. The WAN link of SRO-WA and SRO-SA has the highest delay, moreover SRO-SA queue delay is high (Table 4.4 and Figure 4.3(A). This analysis result explains that end-to-end delay has direct impact on email response time.

point-to-point queuing delay (sec)

Statistic sampling period is 36 seconds.

Sort By Link	Sorted By Average	Sort By Peak
Head Quarter-Addis Ababa <-> SRO-EA-Kigali [0] <--	16.339	25.210
Head Quarter-Addis Ababa <-> SRO-WA-Niamey [0] <--	16.166	25.014
Head Quarter-Addis Ababa <-> SRO-SA-LUSAKA [0] <--	15.804	25.188
Head Quarter-Addis Ababa <-> SRO-SA-LUSAKA [0] -->	0.076	0.126
Head Quarter-Addis Ababa <-> SRO-EA-Kigali [0] -->	0.070	0.157

A) Point to point queue delay of each SRO WAN links

Client Http Page Response Time (seconds)

Statistic sampling period is 36 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-SA-LUSAKA.SRO-SA-LAN	98.838	238.85
SRO-EA-Kigali.SRO-EA-LAN	35.722	94.92
SRO-WA-Niamey.-LAN	28.135	55.43
SRO-CA-Yaounde.SRO-LAN	0.235	0.31
SRO-NA-Rabat.LAN	0.194	0.21

B) Http response time on each SRO WANs

Figure 4.4 Analysis result when Http application is used alone in the traffic flow

Figure 4.4 (A) describes SRO-EA has long point-to-point queue delay when http application is used that is different from point to point delay of links when email is used (Figure 4.3 A). For http application point to point queue delay is even better for SRO-SA link, this is explained by referring to section 4.2, which is Http application TCP connection is fast and it makes the link to match, the SRO-SA router sending rate.

Like an email application, end-to-end delay of a WAN link has direct impact on http application response time. Figure 4.4(B), has described the http application response time as seen by WAN end users. SRO-SA end users have shown poor http usage experience. Point to point queue delay (mainly the receiving side) has impact than packet transmission delay in Http response time as shown above, this explains why SRO-EA users has poor http response time than SRO-WA users (Figure 4.4-B).

End-to-end delay and Optimization techniques

End to end delay is a function of the distance traveled, the speed of light and Transmission delay. Device latencies can also add to this bottleneck. If there is high application turn, the effect of End to end delay (latency) will be high (OPNET, 2008).

As per the experimental analysis result, Point to point queue delay (mainly the receiving side) has significant impact than packet transmission delay for web applications performance. Whereas email Performance is affected by end-to-end delay (packet delay and point to point queue delay). The Data Exchange Chart in section 3.2.4 notified that some applications transactions show high number of application. Based on these facts, http and email applications shows high application turns.

Sending more packets in parallel is found out to be a good technique for reducing the effect of latency as by AppDoctor diagnosis. For web application: by caching the dialog information as much as possible in the client side, it is managed to enhance the application performance. For email application, end-to-end delay aggravates TCP segment delay problem, so it is addressed by increasing TCP window size.

4.3 Applications behavior when sharing WAN links

In this section, the analysis result will be interpreted in terms of all critical applications behavior and presented as follows. Detailed measurement results on IP traffic statistics, applications traffic sending and receiving rate, applications response time is presented. Furthermore, the relation of this scenario conditions and their impact on each applications performance is explained and discussed.

Node	IP End-to-end Delay (sec)	IP End-to-end Delay Variation (sec)	IP Multicast Traffic Received (packets/sec)	IP Multicast Traffic Sent (packets/sec)	IP Number of Hops	IP Processing Delay (sec)	IP Traffic Dropped (packets/sec)	IP Traffic Received (packets/sec)	IP Traffic Sent (packets/sec)	Video Conferencing Packet Delay Variation	Video Conferencing Packet End-to-End Delay (sec)
SRO-CA-Yaounde. SRO-LAN	• 297.66 <192.0.5.5 --> 192.0.3.1> • 234.27 <192.0.5.6 --> 192.0.3.1>	111.74 <192.0.5.7 --> 192.0.3.1>	0	0	3 -->	0	0	53.78	5,112	31.60	12.085
SRO-EA-Kigali. SRO-EA-LAN	43.10 <192.0.5.7 --> 192.0.1.1>	21.25 <192.0.5.7 --> 192.0.1.1>	0	0	• 3 --> • 3 <--	0	0	7.50	8,954	469.91	48.988
SRO-NA-Rabat. LAN	• 366.92 <192.0.5.2 --> 192.0.4.1> • 248.76 <192.0.5.6 --> 192.0.4.1>	• 107.97 <192.0.5.7 --> 192.0.4.1> • 95.21 <192.0.5.6 --> 192.0.4.1> • 81.52 <192.0.5.5 --> 192.0.4.1>	0	0	3 -->	0	0	104.46	5,095	8.24	7.058
SRO-SA-LUSAKA. SRO-SA-LAN	241.40 <192.0.5.5 --> 192.0.0.1>	132.33 <192.0.5.5 --> 192.0.0.1>	0	0	• 3 --> • 3 <--	0	0	9.79	12,135	469.30	57.783
SRO-WA-Niamey.-LAN	41.55 <192.0.5.7 --> 192.0.2.1>	20.73 <192.0.5.7 --> 192.0.2.1>	0	0	3 -->	0	0	7.50	5,112	471.98	48.535

Table 4.8 Average layer 3 IP traffic statistics for each SRO WAN links

Table 4.8 summarizes the average IP packet statistics for all SRO WAN links reported by OPNET. The table shows that the WAN link utilization by five SROs is between 82% and 86 %. The Point to point queue delay (in sec) for Traffic IN and Traffic Out is also shown: SRO-NA and SRO-CA has no Traffic out delay reported. Row 9 and 10 of the table represents the IP traffic receiving and sending rate for the links. In this result, sending means when SRO users send a request to HQ data center and receiving is, when the users get a reply from the data center server.

The statistics indicates that, SRO-NA and SRO-CA send IP traffic with low packet/sec rate and receive IP traffic with high packet/sec rate, relative to other WAN links. These links have lower packet loss as indicated in table 4.7. On the other hand, Table 4.8 also presents IP statistics for the other WAN links, which is SRO-SA, SRO-EA and SRO-WA send IP packets with high packet /sec rate and receive IP packet with small IP packet/sec rate. This three WAN links has higher packet loss accordingly as shown in Table 4.7.

The above result clearly describes that if a link sends packet with high rate but receives with low rate, the link has high packet loss/error rate on the contrary if receiving rate is better than sending the link has no or few packet loss/error rate. This phenomenon can be explained by the fact that request packets size used during sending are relatively smaller than the applications data packets size, which are used during receiving (section 2.2). Thus, if application is sending IP traffics with relatively low packet/sec rate and receiving IP traffic with relatively high packet/sec rate, then it is sensible to reflect that the application has encountered low packet loss/error rate when using that link.

4.3.1 Multimedia Applications (Video/Voice)

Here the tables in Figures 4.5 and 4.6 represent the Video and voice traffic analysis result obtained from OPNET when a traffic flow scenario with all application is experimented. Traffic sending rate, traffic receiving rate, and packets end-to-end delay for these applications is described in figures when being used on different SRO WAN links.

Video Conferencing Traffic Sent (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-SA-LUSAKA.SRO-SA-LAN	61.725	90.333
SRO-EA-Kigali.SRO-EA-LAN	44.987	60.167
SRO-CA-Yaounde.SRO-LAN	24.767	30.167
SRO-WA-Niamey.-LAN	24.752	30.167
SRO-NA-Rabat.LAN	24.668	30.167

A) Video conferencing traffic sent from each SRO links

Video Conferencing Traffic Received (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-CA-Yaounde.SRO-LAN	0.0083333	0.33333
SRO-EA-Kigali.SRO-EA-LAN	0.0083333	0.16667
SRO-NA-Rabat.LAN	0.0083333	0.50000
SRO-SA-LUSAKA.SRO-SA-LAN	0.0083333	0.16667
SRO-WA-Niamey.-LAN	0.0083333	0.16667

B) Video conferencing traffic received on each SRO links

Video Conferencing Packet End-to-End Delay (sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-SA-LUSAKA.SRO-SA-LAN	57.783	90.144
SRO-EA-Kigali.SRO-EA-LAN	48.988	81.274
SRO-WA-Niamey.-LAN	48.535	80.858
SRO-CA-Yaounde.SRO-LAN	12.085	18.103
SRO-NA-Rabat.LAN	7.058	9.606

C) Video conferencing packets end to end delay for each SRO links

Figure 4.5 Video traffic analysis when all application is used in the traffic flow

Figure 4.5(A) and 4.6(A) show that video conferencing traffic sending rate ranges from 24 to 62 packets per sec, whereas for voice application it ranges from 405 to 408 packets per sec. Figures 4.5(B) and 4.6(B) present the Video conferencing traffic receiving rate which is 0.008 packets per sec for all WAN links and the voice application receiving rate which ranges from 0 to 35 packets per sec, respectively. Finally Figures 4.5(C) and 4.6(C) are about Video conferencing and voice applications traffics end-to-end delay, which is a metric, used in this study for measuring these multimedia applications performance.

Voice Application Traffic Sent (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-WA-Niamey.-LAN	408.32	500
SRO-NA-Rabat.LAN	406.38	500
SRO-SA-LUSAKA.SRO-SA-LAN	405.68	500
SRO-EA-Kigali.SRO-EA-LAN	405.63	500
SRO-CA-Yaounde.SRO-LAN	404.97	500

A) Voice application traffic sent from each SRO links

Voice Application Traffic Received (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-NA-Rabat.LAN	34.627	56.167
SRO-CA-Yaounde.SRO-LAN	19.498	50.167
SRO-SA-LUSAKA.SRO-SA-LAN	2.595	38.500
SRO-EA-Kigali.SRO-EA-LAN	0.000	0.000
SRO-WA-Niamey.-LAN	0.000	0.000

B) Voice application traffic received on each SRO links

Voice Application Packet End-to-End Delay (sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-CA-Yaounde.SRO-LAN	297.74	430.71
SRO-SA-LUSAKA.SRO-SA-LAN	241.48	439.42
SRO-NA-Rabat.LAN	212.32	430.08

C) Voice applications packets end to end delay for each SRO links

Figure 4.6 Voice traffic analysis when all application is used in the traffic flow

As presented in Figures 4.5(C) and 4.6(C), Video conferencing traffic end-to-end delay ranges from 7 sec to 57 sec whereas for voice application it ranges from 212 sec to 297 sec. These analysis results reflects that Video conferencing is performing well for most SRO users however voice service is unavailable for almost all SRO users, provided that voice packets latency from server to the users is from 4 to 5 minutes.

Referring to the above tables in Figures 4.6 and 4.7, voice application sends and receive applications traffic with high packets/sec rate than video application however, the performance (measured by end to end latency) is poorer than video. These detailed analysis result reveals that video applications performs best with relatively low application packet receiving rate (packet/sec) than voice which needs a high packet per sec rate. To summarize the point, voice demands from the WAN link low IP packet latency than video or voice demands high applications packet receiving rate than video.

Furthermore, Voice applications packet delay variation (which is between 1794 and 2942) is high as compared to video applications delay variation (which is between 8 and 472). Since those WAN links with high packet delay variations has low voice application performance the benchmarking result reveals, voice applications also demands very low packet delay variation, in addition to low delay. To improve voice performance it is recommended to change the application less delay variation and less latency requirement or implement optimization to decrease packet latency and packet variation.

4.3.2 Data Applications (Http/Email/Database)

This section presents the traffic analysis result of Http, Email, and Database applications obtained from OPNET when a traffic flow scenario with all application is experimented. Traffic sending rate, traffic receiving rate, and response time for these applications is described in figures below

Client Http Traffic Sent (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-SA-LUSAKA.SRO-SA-LAN	0.011667	0.83333
SRO-EA-Kigali.SRO-EA-LAN	0.010000	1.00000
SRO-CA-Yaounde.SRO-LAN	0.000000	0.00000
SRO-NA-Rabat.LAN	0.000000	0.00000
SRO-WA-Niamey.-LAN	0.000000	0.00000

when being used on different SRO WAN links.

A) Http application traffic sent from each SRO links

Client Http Traffic Received (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-EA-Kigali.SRO-EA-LAN	0.0100000	1.00000
SRO-SA-LUSAKA.SRO-SA-LAN	0.0066667	0.50000
SRO-CA-Yaounde.SRO-LAN	0.0000000	0.00000
SRO-NA-Rabat.LAN	0.0000000	0.00000
SRO-WA-Niamey.-LAN	0.0000000	0.00000

B) Http application traffic received on each SRO links

Client Http Object Response Time (seconds)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-SA-LUSAKA.SRO-SA-LAN	3.6784	5.6333
SRO-EA-Kigali.SRO-EA-LAN	0.3859	0.3859

C) Http object response time for each SRO users

Client Http Page Response Time (seconds)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-EA-LAN	0.88809	0.88809

D) Http page response time for each SRO users

Figure 4.7 Http traffic analysis when all application is used in the traffic flow

Figures 4.7(A), 4.8(A) and 4.9(A) show that client http traffic sending rate ranges from 0.0100 to 0.0117 packets per sec, and client database traffic sending rate ranges from 0.000 to 0.087 whereas for email application no packet flow reported (zero packets per sec). Figures 4.7(B), 4.8(B) and 4.9(B) present the http traffic-receiving rate, which ranges from 0.006 to 0.01 packets per sec whereas the email and database applications receive no packets (0 packets per sec). Finally Figures 4.7(C) and 4.7(D) are about http applications traffics response time, which is a metric, used in this study for measuring these data applications performance.

Client DB Traffic Sent (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-SA-LUSAKA.SRO-SA-LAN	0.086667	0.66667
SRO-CA-Yaounde.SRO-LAN	0.000000	0.00000
SRO-EA-Kigali.SRO-EA-LAN	0.000000	0.00000
SRO-NA-Rabat.LAN	0.000000	0.00000
SRO-WA-Niamey.-LAN	0.000000	0.00000

A) Database application traffic sent from each SRO links

Client DB Traffic Received (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-CA-Yaounde.SRO-LAN	0	0
SRO-EA-Kigali.SRO-EA-LAN	0	0
SRO-NA-Rabat.LAN	0	0
SRO-SA-LUSAKA.SRO-SA-LAN	0	0
SRO-WA-Niamey.-LAN	0	0

B) Database application traffic received on each SRO links

Figure 4.8 Database traffic analysis when all application is used in the traffic flow

As results presented in figures, Out of three tested critical data applications only http application responds to client requests. And, from the five SROs, SRO-EA users get better average page response time which is 0.888 sec and object response time, 0.386 sec which is shown in Figures 4.7(C) and (D). This result also reveals that when Object response time is slow http applications response time will be slow accordingly. These phenomena is explained by applications' TCP connection behavior (Section 2.2) which is, when application object doesn't respond within a given time, application considers the packet is lost and will re-establish the TCP communication again that takes more additional time. This result clearly points out another performance issue that is improving the Http object lost (packet lost) handling mechanism can improve the Http application response time.

Client Email Traffic Sent (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-CA-Yaounde.SRO-LAN	0	0
SRO-EA-Kigali.SRO-EA-LAN	0	0
SRO-NA-Rabat.LAN	0	0
SRO-SA-LUSAKA.SRO-SA-LAN	0	0
SRO-WA-Niamey.-LAN	0	0

A) Email application traffic sent from each SRO links

Client Email Traffic Received (packets/sec)

Statistic sampling period is 6 seconds.

Sort By Node	Sorted By Average	Sort By Peak
SRO-CA-Yaounde.SRO-LAN	0	0
SRO-EA-Kigali.SRO-EA-LAN	0	0
SRO-NA-Rabat.LAN	0	0
SRO-SA-LUSAKA.SRO-SA-LAN	0	0
SRO-WA-Niamey.-LAN	0	0

B) Email application traffic received on each SRO links

Figure 4.9 Email traffic analysis when all application is used in the traffic flow

The analysis result presented in section 4.3.1 and 4.3.2, demonstrate that multimedia traffics (Voice and Video) show better traffic usage in SRO-WAN links than Data traffic. The low end-to-end packet delays for video are evident (see Figure 4.5(C)). From Data applications Http application is better than email and database. This result is explained further as follows.

Since data applications use TCP for layer four communication, its connection delay will affect the performance than video and voice that uses UDP, which needs no connection establishment and connects faster than data, and consequently consumes the available bandwidth.

Since web application's TCP delay is smaller than database and email as discussed in section 4.1 but not faster than UDP, it needs a link with a packet rate not fast enough (like SRO-CA and SRO-NA) because voice/video will use them quickly: and not very much slow (like SRO_WA and SRO-

SA) which makes TCP connection slow. Since SRO-EA has less packet delay than SRO-WA and SRO-SA, Http traffic uses this link during this scenario.

To summarize the experimental result found on applications behavior, applications compute to be connected to low latency links. Furthermore, as the application response result found in the experiment tells, video application consumes the low latency links bandwidth first, and then Data applications with low TCP connection delay (like HTTP) get advantage to use the link. Those applications with high TCP connection delay (Database and Email) and applications that demand low latency like (Voice) will be affected by packet loss and high retransmission count in this scenario.

4.4 Impacts of Background Traffic

In this section, the analysis result that examines the impact of non-critical applications (background traffic) bandwidth usage on end users critical applications response time will be presented. Detailed measurement results on applications response time, applications traffic sending and receiving rate, and applications packets end-to-end delay is presented. How changing the volume of this background traffic is reflected on each applications performance is explained and discussed.

4.4.1 Email Application

Tables 4.9, 4.10 and 4.11 present email download response time result obtained from OPNET when email traffic flow scenario is experimented by changing the volume of the background traffic. Table 4.9 shows email download response time when a traffic flow is without link background load. Table 4.10 shows email download response time when a traffic flow is with link background load (size = 15% of link bandwidth) and Table 4.11 shows email download response time when a traffic flow is with link background load (size = 75% of link bandwidth).

Email

Statistic	Average	Maximum	Minimum
Email Download Response Time (sec)	0.19294	0.22949	0.18880
Email Traffic Received (bytes/sec)	1,989.5	3,425.8	710.2
Email Traffic Received (packets/sec)	1.3228	2.2778	0.4722
Email Traffic Sent (bytes/sec)	1,989.5	3,425.8	710.2
Email Traffic Sent (packets/sec)	1.3228	2.2778	0.4722
Email Upload Response Time (sec)	0.19457	0.22916	0.19059

Table 4.9 Email download response time without link background load

Email			
Statistic	**Average**	**Maximum**	**Minimum**
Email Download Response Time (sec)	0.20041	0.25929	0.19227
Email Traffic Received (bytes/sec)	2,062.6	3,885.3	459.6
Email Traffic Received (packets/sec)	1.3714	2.5833	0.3056
Email Traffic Sent (bytes/sec)	2,062.6	3,885.3	459.6
Email Traffic Sent (packets/sec)	1.3714	2.5833	0.3056
Email Upload Response Time (sec)	0.20092	0.23705	0.19203

Table 4.10 Email download response time with link background load (size = 15% of link bandwidth)

Email			
Statistic	**Average**	**Maximum**	**Minimum**
Email Download Response Time (sec)	1.0136	4.8277	0.3834
Email Traffic Received (bytes/sec)	5,067.6	9,400.0	1,336.9
Email Traffic Received (packets/sec)	3.3694	6.2500	0.8889
Email Traffic Sent (bytes/sec)	5,067.6	9,400.0	1,336.9
Email Traffic Sent (packets/sec)	3.3694	6.2500	0.8889
Email Upload Response Time (sec)	0.9531	4.0457	0.4233

Table 4.11 Email download response time with link background load (size = 75% of link bandwidth)

As presented in the above tables, user's Email Download Response Time is increased from 0.19 sec to 0.2 sec when the link carries additional 15% background load. In the second experiment when the background traffic volume is increased from 15% to 75%, Email Download Response Time rises to 1 sec from 0.2 sec.

4.4.2 Http Application

Tables 4.12, 4.13 and 4.14 present http Page Response Time result obtained from OPNET when http traffic flow scenario with various background traffic is experimented. Table 4.12 shows http page response time when a traffic flow is without link background load. Table 4.13 shows http page response time when a traffic flow is with link background load (size = 15% of link bandwidth) and Table 4.14 shows http page response time when a traffic flow is with link background load (size = 75% of link bandwidth).

HTTP			
Statistic	Average	Maximum	Minimum
HTTP Object Response Time (seconds)	0.080925	0.089803	0.077726
HTTP Page Response Time (seconds)	0.19698	0.22352	0.19098
HTTP Traffic Received (bytes/sec)	9,272	11,121	5,921
HTTP Traffic Received (packets/sec)	11.898	14.083	7.583
HTTP Traffic Sent (bytes/sec)	9,281	11,119	5,921
HTTP Traffic Sent (packets/sec)	11.905	14.076	7.583

Table 4.12 Http page response time without link background load

HTTP			
Statistic	Average	Maximum	Minimum
HTTP Object Response Time (seconds)	0.083685	0.089423	0.079932
HTTP Page Response Time (seconds)	0.20487	0.21980	0.19537
HTTP Traffic Received (bytes/sec)	9,337	11,399	5,656
HTTP Traffic Received (packets/sec)	12.009	14.764	7.333
HTTP Traffic Sent (bytes/sec)	9,345	11,399	5,656
HTTP Traffic Sent (packets/sec)	12.015	14.764	7.333

Table 4.13 Http page response time with link background load (size = 15% of link bandwidth)

HTTP			
Statistic	Average	Maximum	Minimum
HTTP Object Response Time (seconds)	1.571	12.325	0.227
HTTP Page Response Time (seconds)	3.283	15.950	0.555
HTTP Traffic Received (bytes/sec)	21,568	25,467	9,733
HTTP Traffic Received (packets/sec)	27.895	32.965	13.563
HTTP Traffic Sent (bytes/sec)	21,969	25,587	11,510
HTTP Traffic Sent (packets/sec)	28.217	33.035	14.993

Table 4.14 Http page response time with link background load (size = 75% of link bandwidth)

As presented in the above tables, user's http page response time is increased from 0.19 sec to 0.2 sec when the link carries additional 15% background load. In the second experiment when the background traffic volume is increased from 15% to 75%, http page response time rises to 3.2 sec from 0.2 sec. This significant response time change comparing to email application conveys, http application is affected by background traffic than email.

4.4.3 IMIS Application (Database application)

Tables 4.15, 4.16 and 4.17 present Database Query Response Time result obtained from OPNET, when database traffic flow scenario is experimented with various background traffic. Table 4.15 shows Database Query Response Time when a traffic flow is without link background load. Table 4.16 shows database query response time when a traffic flow is with link background load (size =

15% of link bandwidth) and Table 4.17 shows database query response time when a traffic flow is with link background load (size = 75% of link bandwidth).

DB Query			
Statistic	Average	Maximum	Minimum
DB Query Response Time (sec)	1.5501	2.7465	1.1086
DB Query Traffic Received (bytes/sec)	82,139	93,365	26,812
DB Query Traffic Received (packets/sec)	4.9365	5.6042	1.6181
DB Query Traffic Sent (bytes/sec)	82,144	93,138	27,040
DB Query Traffic Sent (packets/sec)	14.810	16.792	4.875

Table 4.15 Database query response time without link background load

DB Query			
Statistic	Average	Maximum	Minimum
DB Query Response Time (sec)	3.4669	9.9790	1.5148
DB Query Traffic Received (bytes/sec)	82,082	93,095	21,511
DB Query Traffic Received (packets/sec)	4.9340	5.5139	1.3264
DB Query Traffic Sent (bytes/sec)	82,123	92,213	22,649
DB Query Traffic Sent (packets/sec)	14.806	16.625	4.083

Table 4.16 Database query response time with link background load (size = 15% of link bandwidth)

DB Query			
Statistic	Average	Maximum	Minimum
DB Query Response Time (sec)	470.5	1,641.7	3.5
DB Query Traffic Received (bytes/sec)	65,296	91,332	21,767
DB Query Traffic Received (packets/sec)	3.9726	5.5694	1.3889
DB Query Traffic Sent (bytes/sec)	67,177	94,091	24,501
DB Query Traffic Sent (packets/sec)	12.533	17.014	4.424

Table 4.17 Database query response time with link background load (size = 75% of link bandwidth)

As presented in the above tables, user's Database Query Response Time is increased from 1.5 sec to 3.4 sec when the link carries additional 15% background load. In the second experiment when the background traffic volume is increased from 15% to 75%, database query response time rises to 470 sec from 3.4 sec. This immense response time change shows how a database application is extremely impacted by high volume of background traffic change comparing to Email and database applications.

4.4.4 Voice Application

Tables 4.18 and 4.19 present selected voice application packets end to end delay result obtained from OPNET, when voice traffic flow scenario is experimented with various background traffic. Table 4.18 shows voice application packets end to end delay when a traffic flow is without link background load and Table 4.19 shows voice application packets end to end delay when a traffic flow is with link background load (size = 50% of link bandwidth).

Voice			
Statistic	**Average**	**Maximum**	**Minimum**
Voice Jitter (sec)	0.00000403	0.00039487	-0.00000000
Voice MOS Value	3.4537	3.4970	3.4528
Voice Packet Delay Variation	0.002575	0.042756	0.000000
Voice Packet End-to-End Delay (sec)	0.52567	0.53073	0.09281
Voice Traffic Received (bytes/sec)	147,054	151,800	0
Voice Traffic Received (packets/sec)	4,456.2	4,600.0	0.0
Voice Traffic Sent (bytes/sec)	188,572	194,700	0
Voice Traffic Sent (packets/sec)	5,714.3	5,900.0	0.0

Table 4.18 Voice application packets end to end delay without link background load

Voice			
Statistic	**Average**	**Maximum**	**Minimum**
Voice Jitter (sec)	0.00001053	0.00096365	-0.00008674
Voice MOS Value	1.0510	3.4970	1.0000
Voice Packet Delay Variation	0.01162	0.10568	0.00001
Voice Packet End-to-End Delay (sec)	1.1817	1.2059	0.0966
Voice Traffic Received (bytes/sec)	116,056	120,781	0
Voice Traffic Received (packets/sec)	3,516.9	3,660.0	0.0
Voice Traffic Sent (bytes/sec)	191,758	198,000	0
Voice Traffic Sent (packets/sec)	5,810.8	6,000.0	0.0

Table 4.19 Voice application packets end to end delay with link background load

As presented in the above tables, voice application packets end-to-end delay is increased from 0.5 sec to 1.1 sec, when the link carries additional 50 % background load in the voice application's traffic flow. This application's packet end-to-end delay change shows that voice application is impacted by background traffic and needs more throughput over WAN links.

4.4.5 Video Application

Tables 4.20 and 4.21 present selected video conferencing packets end-to-end delay result obtained from OPNET when video traffic flow scenario is experimented with various background traffic. Table 4.20 shows video conferencing packets end to end delay when a traffic flow is without link background load and Table 4.21 shows video conferencing packets end to end delay when a traffic flow is with link background load (size = 50% of link bandwidth).

Video Conferencing			
Statistic	Average	Maximum	Minimum
Video Conferencing Packet Delay Variation	15.752	32.174	0.000
Video Conferencing Packet End-to-End Delay (sec)	11.017	20.099	3.074
Video Conferencing Traffic Received (bytes/sec)	7,231	361,541	0
Video Conferencing Traffic Received (packets/sec)	0.0285	1.4265	0.0000
Video Conferencing Traffic Sent (bytes/sec)	51,007,602	60,991,909	0
Video Conferencing Traffic Sent (packets/sec)	201.27	240.66	0.00

Table 4.20 Video conferencing packets end to end delay without link background load

Video Conferencing			
Statistic	Average	Maximum	Minimum
Video Conferencing Packet Delay Variation	32.492	32.492	32.492
Video Conferencing Packet End-to-End Delay (sec)	9.1602	9.1602	9.1602
Video Conferencing Traffic Received (bytes/sec)	7,411	140,800	0
Video Conferencing Traffic Received (packets/sec)	0.02924	0.55556	0.00000
Video Conferencing Traffic Sent (bytes/sec)	31,753,735	38,065,280	0
Video Conferencing Traffic Sent (packets/sec)	125.30	150.19	0.00

Table 4.21 Video conferencing packets end to end delay with link background load

As presented in the above tables, video conferencing packets end-to-end delay is changed from 11 sec to 9 sec, when the link carries additional 50 % background load in the Video conferencing traffic flow. This result reveals an interesting relationship found on a video: video conferencing packets end-to-end delay is decreased when the background load is increased to some level. This is explained by a phenomenon resulted in the experiment which is regarding LAN delay. When a Video conferencing traffic flow is used in the experiment without background load, then end users LAN delay becomes high on the contrary LAN delay becomes low, when it is used with Back ground load. If there is a high delay on the LAN, it is becoming a bottleneck for communication, which results in packet loss and poor performance. This can tell us video can use the bandwidth efficiently even if there is a background load, since it requires less packet/sec rate to work efficiently (section 4.3.1).

Background Traffic impact and Optimization Techniques

From different scenarios experimentation, it is found out that; background traffic load has different level of impact on critical applications performance. As far as data applications are concerned, Database applications are highly impacted by large volume of background traffic change comparing to email and web applications. Web applications are considerably affected by background traffic than email. Considering multimedia applications, voice application performance is affected by background traffic, which implies that voice application needs more throughput over WAN links.

In this study, as shown in Tables 4.22 and 4.23, considerable critical applications performance improvement is exhibited by rescheduling non-critical applications (for organization business) when link is over utilized. Non- critical applications traffic flow is controlled in a congested link using two approaches 1) by rescheduling applications in an order starting from domain applications, Monitoring traffics, and replication traffics and 2) by dropping application traffics using unknown TCP ports.

	Email Download Response Time (sec)	Http Page Response Time (sec)	DB Query Response Time (sec)
When non-critical applications are using the WAN link	1.01	3.28	470.50
After non-critical applications traffic flow is rescheduled	0.19	0.19	1.50

Table 4.22 Effect of background traffic on Data Applications Response time

	Voice Packet End-to-End Delay (sec)	Video Packet End-to-End Delay (sec)
When non-critical applications are using the WAN link	1.18	9.16
After non-critical applications traffic flow is rescheduled	0.52	11.01

Table 4.23 Effect of background traffic on Multimedia Applications performance

Table 4.23 describes that a Video Packet End-to-End Delay becomes higher after a non-critical applications traffic flow using the WAN link is rescheduled. This is explained by LAN delay as briefly described above. When Video conferencing traffic is used without background load, end users LAN delay becomes high on the contrary when it is used with Back ground load, LAN delay becomes low. From this experimental analysis result, it is clear that video applications needs relatively low throughput than voice applications.

4.5 Impacts of Protocol Overhead

The analysis results in sections 4.31 and 4.3.2 explain that when there is a link congestion, multimedia traffics (Voice and Video) shows better traffic usage than Data traffic. Voice application utilizes the WAN link bandwidth than video, it also demands high applications packet receiving rate and very low packet delay variation than video. Although voice utilize the bandwidth with high traffic volume and high packet receiving rate than video, its performance is poorer than video. Protocol overhead is indicated a source of performance problem in the data analysis phase. As investigated in the experiment, the performance behavior of voice tells that it is highly affected by protocol overhead.

Protocol headers add overhead to each application message (TCP and IP protocol headers). Large amounts of protocol overhead are increasing the utilization of your network. This overhead also introduces delays by increasing congestion in the network; the delays can be especially significant if you are sending a large number of small application messages. Make your application send fewer, larger application messages. This will utilize network resources more efficiently (OPNET, 2008).

Hence, Table 4.24 show that; better voice application performance is achieved in this experimental study by increasing its application segment size, which results in less packet receiving rate demand and efficient utilization of bandwidth. Application segment size is the maximum size in bits of any segment that will be passed on to the TPAL layer from the application.

	Voice application traffic received (packets/sec)	Voice Packet End-to-End Delay (sec)
When Voice Application segment size is 32kbits	1,452.8	0.89
When Voice Application segment size is 64kbits	1,203.0	0.09

Table 4.24 Effect of protocol overhead on Voice Applications performance

4.6 Analysis Result Summary for UNECA WAN Applications Performance

4.6.1 Performance Bottlenecks

- Email segment delay is one of the factor for email applications performance problem. IMIS (Database application) on the other hand has high TCP retransmission count which results in poor Database application performance (Section 4.1).
- End to end delay (packet delay and point-to-point queue delay) has direct impact on email Performance. For web applications performance, point-to-point queue delay (mainly the receiving side) has significant impact than packet transmission delay (Section 4.2).
- Voice demands low IP packet latency from the WAN link than video, in other words voice demands high applications packet receiving rate than video. Voice applications also demands very low packet delay variation, in addition to low delay (Section 4.2).
- When the router sends traffic with a high rate, and if the link does not match that rate the traffic will be queued and this is found to be one of the causes for high point to point queue delay (Section 4.2).
- Http protocol chattiness is a bottleneck (this bottleneck is explained in section 5.1.2) for better Web application performance (Section 4.1).
- When there is a link congestion multimedia traffics (Voice and Video) shows better traffic usage than Data traffic. Although, voice utilizes the bandwidth well it experiences high end-to-end packet delay (Section 4.3).
- During congestion among data applications: Http application performs better, but email and database applications become totally unresponsive. As the analysis result explains, when link is congested, email and IMIS applications' TCP connection rate becomes slow and applications using UDP protocol, which needs no connection establishment, performs better than TCP dependent applications (Section 4.3).
- Voice application, which demands low latency, email and IMIS database, can get impacted highly by packet loss and high retransmission count than others (Section 4.3).
- Web application is affected by background traffic than email. Database application is considerably impacted by high volume of background traffic change comparing to Email and

database applications. Voice application performance is affected by background traffic, which implies that voice application needs more throughput over WAN links (Section 4.4).

- When a Video conferencing traffic flow is used in the experiment without background load, end users LAN delay becomes high on the contrary when it is used with back ground load, LAN delay becomes low. From this experimental analysis result, it is clear that video applications needs relatively low throughput than voice applications (Section 4.4).

4.6.2 Optimization Techniques

In this study, the techniques listed below finds out to be the appropriate application optimization techniques for various situations (bottlenecks) encountered in an enterprise WAN environments under the study.

- Modifying TCP window size on the application server is found to be effective for reducing TCP segment delay.
- By Enabling TCP extensions such as “Fast Retransmit” and “Fast Recover it is possible to reduce the impact of high retransmission bottleneck.
- Enabling “TCP NODELAY” (disabling Nagle’s algorithm) is found to be effective to make TCP transfer small segments of the application and reduce protocol delay bottleneck.
- Forcing SMTP to use TCP slow start algorithm is appropriate for SMTP protocol bottleneck
- Increasing application segment size an effective technique to reduce the protocol overhead, for links with high propagation delay.
- By increasing TCP window size of chatty application, it is possible to improve its performance in high bandwidth and high latency WAN links.
- By rescheduling applications in an order starting from domain applications, monitoring traffics, and replication traffics and by dropping application traffics using unknown TCP ports, it is possible to reduce the impact of congestion and bandwidth bottleneck
- Decreasing the application segment size on the application server is effective for congestion and bandwidth bottleneck.
- Applying server based compression techniques on applications traffic for reducing the effect of bandwidth and congestion bottleneck.

CHAPTER FIVE

DEVELOPING WAN OPTIMIZATION MODEL

Detail experimental effort has been made to understand the limitation of the data communication through the current UNECA WAN and to explore ways that enhances the application traffic flow of the enterprise. Based on the experimental result, this thesis develops a model for optimizing applications communications over the WAN, so that the organization will be benefited from increased efficiency and effectiveness of their enterprise applications.

The objective of this study is developing WAN optimization model that will improve the performance of business critical applications when used over the WAN. To achieve the goal, the study uses an approach that benchmarks the applications performance in terms of various WAN bottlenecks and then based on benchmarking result develops the optimization algorithms. Therefore, benchmarking the performance of critical applications based on various WAN bottlenecks is the primary task for developing the optimization algorithms.

In this chapter the benchmarking result and the developed WAN optimization algorithm is discussed.

5.1 Application Performance benchmarking

According to Nahum, Ros, Seshan & Almeida (2001) performance benchmarks are used to evaluate applications or server performances, and thus have a large impact on what performance optimizations are applied to application infrastructure and servers. A key challenge is to predict precisely, the impacts of WAN performance factors on applications performance. Hence, in this study, applications performance benchmarking is done to define the impact of various bottlenecks on application performance.

This section presents application performance benchmarking results, which are built based on the experimental analysis findings. To begin with, the meanings of various WAN applications bottlenecks that have been identified in the experiment are explained briefly as defined by OPNET. Then, major business critical applications' performance bottlenecks encountered in the enterprise WAN is described and finally applications performance is benchmarked in terms of the reported bottlenecks.

5.1.1 WAN applications bottleneck

Generally, the analysis result shows that the enterprise WANs introduce the following applications bottlenecks, which are explained briefly as defined by OPNET (OPNET, 2008). The WAN factors that cause each application bottlenecks are also described.

Protocol overhead bottleneck is the total protocol overhead expressed as a percentage of the total amount of data transferred. Each protocol adds overhead to an application message in the form of headers.

Chattiness bottleneck is the number of application bytes per application turn. If an application is "chatty", the data sent in each application turn is small. This may cause significant network delays and also processing delays at each tier since each tier now has to handle many little messages. Applications that send many small packets back-and-forth incur a network delay. This delay becomes significant if you have a high latency link.

Propagation delay bottleneck is the time taken by the packets to propagate across the network represented as a percentage of the total application response time. Propagation delay is a function of the distance traveled and the speed of light. Device latencies can also add to this bottleneck.

Transmission delay bottleneck is the transmission delay caused by line speeds expressed as a percentage of the total application response time. The transmission delay is a function of the total bytes transmitted and the line speed.

Protocol delay bottleneck is the total delay due to protocol effects represented as a percentage of the total application response time. Examples of protocol effects are TCP flow control, congestion control, delay due to retransmissions, and collisions.

Connection resets bottleneck is the total number of connection resets. Connection resets are caused due to delayed or duplicated packets, application ports not being available or when applications close connections. By connection, we imply a transport connection.

Retransmissions bottleneck is the total percentage of packets that were retransmitted. Protocols such as TCP retransmit a packet if they detect a long latency or a packet loss. Retransmission causes delays and additional protocol overhead. TCP also reduces the rate at which applications can send traffic when a retransmission occurs as a means of congestion control. This causes additional throttling of application traffic. Packet loss or unusual delays that trigger retransmissions can occur as a result of “bursty” application traffic, overflowing queues, misbehaving devices and link or node failures.

TCP windowing bottleneck is the bandwidth-delay product used by the TCP connection. When an application sends bulk data over a TCP connection, the TCP window size should be large enough to permit TCP to send many packets in a row without having to wait for TCP acknowledgements.

TCP frozen window bottleneck is the advertised TCP Receive Window that has dropped to a value smaller than the Maximum Segment Size (MSS). When this occurs, the sender cannot send any data until the receive window is one MSS or larger.

TCP Nagle’s algorithm bottleneck indicates that Nagle's algorithm is present and is slowing application response times. Nagle's algorithm is a sending-side algorithm that reduces the number of small packets on the network, thereby increasing router efficiency. Nagle's algorithm causes excessive numbers of delayed ACKs and slows down the application.

5.1.2 Performance Bottlenecks

As described in chapter 4, the WAN factors that primarily caused the applications bottlenecks are network devices processing delay, link latency, inefficiencies of existing transport and application protocols, link propagation delay, link bandwidth, duplicate or lost packets, link congestion. Thus, based on OPNET (2008) definition, bottlenecks that are caused by the above-mentioned factors are summarized in to protocol bottleneck, congestion bottleneck, bandwidth bottleneck and latency bottleneck.

In general, this study finds out that the bottlenecks listed below are a major source of problem for poor performance of critical applications in enterprise WAN environments particularly for remote and branch office users. Please refer chapter 4, for experimental result that leads to conclusions, which are mentioned below.

- Protocol bottleneck: The problem is caused by TCP and application protocols inefficiency specifically for low resourced WAN environment. It is characterized by WAN conditions like TCP retransmission, MAC collusions and TCP delay (OPNET, 2008).
- Latency bottleneck: The problem is caused by long propagation delay and characterized by WAN conditions like long round trip time (OPNET, 2008).
- Congestion bottleneck: This problem is due to over utilized link and characterized by WAN conditions like unusual queue delay, MAC collusions and TCP retransmission (OPNET, 2008).
- Bandwidth bottleneck: Due to slow link transmission rate and characterized by WAN conditions like very small packet transmission rate in bps (OPNET, 2008).

5.1.3 Performance Benchmarking

AppDoctor's Summary of Delays and Diagnosis have revealed application bottlenecks and their threshold value for a better application performance. For WAN end users to get the required service from critical applications that are deployed in a WAN environment: application bottlenecks should be reduced below their respective threshold value as described below. These benchmarks are the result of the experimental study on the target organization WAN and applies for low resourced WAN environments (please refer section 3.2.4 and chapter 4 for information on how these benchmarks are derived).

- **Protocol Overhead**: Protocol overhead should be kept below 25.6% of the total traffic volume in the WAN link.
- **Propagation Delay**: The total delay incurred due to propagation should not be greater than 20.3% of the overall application response time.
- **Transmission Delay**: The total delay incurred due to transmission speed (Bandwidth) should not be greater than 30.0% of the overall application response time.

- **Protocol Delay:** The total delay incurred due to protocol effects should not be greater than 3.0% of the overall application response time.
- **Congestion Delay**: The total delay incurred due to congestion effects should not be greater than 27.0% of the overall application response time**.**
- **Retransmissions**: the number of packets that are retransmitted should be kept below 3.0% of the total packet transmitted.

The benchmarks shown above are the standards required from the WAN environment as a whole, so that better performance is achieved by all critical applications. Additional required benchmarks by each critical application for an improved performance are:

- Voice application demands a protocol overhead of less than 20 % of its total application payload to perform as required by end user.
- Database application requires link utilization by background traffics to be less than 15%, for end users to use the application with response time of less than 14 sec.
- Database application also demands a protocol overhead lower than the threshold value recommended for the other data applications.
- Email application requires very low TCP segment delay from the WAN environment
- Web application requires very low percentage of packet retransmission for an improved performance in a low resourced environment.
- Email application requires an improved TCP Nagle's algorithm to be efficient in a low resourced WAN environment.
- No of application turns should be kept to a minimum value so that voice, email and http applications performance is enhanced to the required level.
- Video Conferencing application demands low LAN delay unlike the other four applications.

5.2 Developing WAN Optimization Model

The causes of applications performance problem and the appropriate optimization technique for each causes is determined using experimentation on UNECA WAN environment. This final phase of the study develop the proposed WAN Optimization Model, which is capable of determining performance problem in WAN link at real-time and accordingly apply the appropriate optimization techniques until performance is improved. This section discusses this algorithm in detail including the approach used to develop algorithm.

The analysis result (chapter 3 and 4) and the benchmarking result from section 5.1 is used to narrow down the elements of the WAN Optimization algorithm model.

5.2.1 Proposed WAN Optimization Model

The first three research question of the study have been answered by experimental analysis and benchmarking task, which determine application bottlenecks, performance behavior of each critical applications under this bottlenecks and benchmarks. The last research question of the study "How is it possible to develop a WAN optimization model that can identify and improve the performance problem of WAN critical applications in real-time?" is answered by developing the Optimization Model with the proposed features.

As indicated in the research question, the proposed features are to identify application bottlenecks in real time and apply appropriate techniques for improving the performance of each critical application. Hence, the features of the proposed model are further elaborated below so that an algorithm will be designed for each feature.

- Monitors and identifies the real-time WAN conditions of the link, and initiate optimization actions when the conditions changes in any enterprise's WAN link.
- From the identified conditions of the WAN, determine the category of the applications bottleneck of the link. Hence, optimization technique will be selected based on these categories of the application bottleneck.
- Identify the new applications connection requests from any enterprise WAN link at real-time (SYN packets), when WAN conditions changes.
- Select the appropriate optimization techniques based on the determined applications bottleneck of the link and apply it to the application server, which is accepting new applications connection requests at that moment. Control optimization techniques are also applied on non-critical application traffic flows that are sharing the bandwidth of the WAN link. The optimization techniques are prioritized on their effectiveness to reduce the performance impact of the real time WANs conditions.

Appropriate technique in this context means: the techniques applied should control the effect of the real time WAN factors to a level that is required by the application that is currently using the link. The applications' demand for various WAN factors (bottlenecks) has been determined by the experimentation as described in chapter 3 and 4, and used to benchmark the performance of applications as discussed in section 5.1 of this chapter. The optimization techniques used to control each bottlenecks category are also selected based on the application performance result achieved by each technique in the experiments conducted on simulated UNECA WAN environment (please refer chapter 3 and 4).

Desirable Features of optimization Model

One of the main feature of the proposed Model is to recognize the real-time WAN conditions of the link before initiating the appropriate optimization action. The model is desired to have an algorithm that use an intelligent approach to spot applications bottlenecks and provide the optimization solutions, accordingly. Moreover, the model needs an algorithm that can adapt quickly to network changes. An intelligent approach to detect and solve applications bottlenecks means that the proposed algorithm is able to decide for the best decision on the way information is exchanged between the client and the server.

The algorithm is designed to determine the presence of applications bottlenecks listed below which are selected based on the experimentation and benchmarking task done as described in section 5.1:

- Protocol bottleneck
- Latency bottleneck
- Congestion bottleneck
- Bandwidth bottleneck

The algorithm is also needed be equipped with various predefined rules. Once the application bottlenecks are determined at real time in any given WAN link, these rules are used to pick the decision that should be taken by the algorithm. The rules will decide on issues like what optimization technique to apply and how to apply it based on the behavior of the application currently using the link. The next section discusses the architecture of the model and the components incorporated to provide the above-mentioned necessary features.

5.2.2 WAN Optimization Model Architectures

The optimization model is designed with four major building blocks to improve the application performance problem over a WAN link, which lacks sufficient resource for providing services for remote office and branch office users. As per the study, such kinds of enterprise WANs introduce various nature of bottlenecks when users try to access and use enterprise applications through these WAN links. Thus, this study proposes an optimization model that can solve these application performance issues particularly for enterprise WAN that lacks sufficient network resources to provide resource access for their branch office users. The major components of the model are listed below:

- Network Condition Recognizer (NCR)
- WAN Bottleneck Determiner (WBD)
- Optimization Solution Provider (OSP)
- Event Reporter (ER)

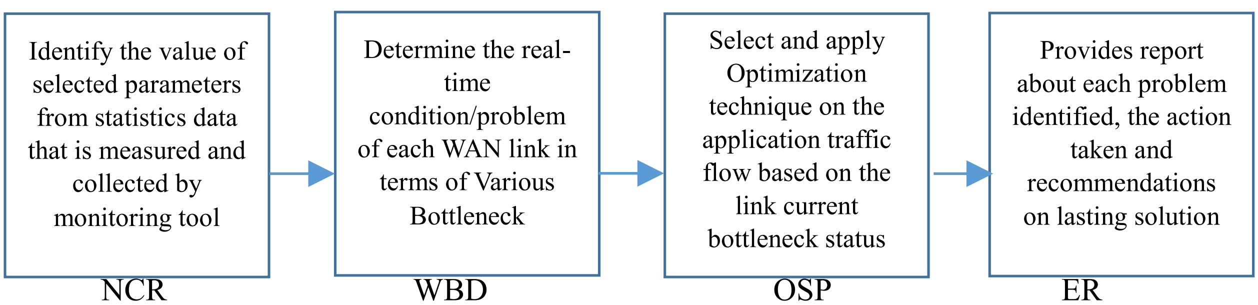


Figure 5.1 Building Blocks of the optimization Model

Network condition recognizer

This component is responsible for monitoring the enterprise WAN environment to detect and recognize the presence of any WAN conditions that affects business critical applications performance. It also recognizes if there is a change in any of the WAN conditions on each links. The Network condition recognizer periodically monitors the status of the conditions and when it recognizes any change, it provides the information to WAN bottleneck determiner. This component is capable of recognizing any change on various WAN conditions (factors, selected from the WAN environment under the study) in any enterprise WAN link, which includes TCP retransmission count, Mac layers collusion, link transmission rate, packets receiving rate, queue delay, Link utilization level, TCP delay and propagation delay.

Monitoring link statistics including TCP retransmission count, MAC collusions, propagation delay, TCP delay, link rate and queue delay helps to recognize the real time status of WAN factors in each WAN link. And from these real-time link statistics it is possible to determine which performance bottleneck category should be solved to optimize the performance of the application traffic flow which is using the WAN link at that point of time.

This component uses link Statistics collector and Timer algorithms to accomplish its task, which will be explained later in section 5.2.3.

WAN bottleneck determiner:
This component is responsible for determining the bottleneck status of each Branch offices WAN links. Which means it determines the real time bottlenecks status of each WAN link as per the WAN conditions reported by network condition recognizer. Then, it uses predefined rule that are described below for interpreting WAN conditions to bottleneck problem categories. The rules are defined based on the experimental results found from the study regarding the factors and their effect on application bottlenecks categories.

- .If TCP delay and TCP retransmission count is high but no queue delay, then Protocol bottleneck
- If link propagation delay is high (based on link RTT), then Latency bottleneck
- If queue delay, TCP retransmission count and mac collusion is high then congestion bottleneck
- If link transmission rate is slow, then Bandwidth bottleneck

This component uses bottleneck determiner algorithm to accomplish its task and is initialized by link-statistics-collector algorithm. The bottleneck-determiner-algorithm detail will be discussed later.

Optimization solution provider:
The task of this component is selecting and applying the optimization solution technique, as request is received from any of the WAN links. Its operation is based on the real time bottleneck status of the WAN link that is determined by the pervious components. Like bottleneck determiner,

it uses predefined rule but in this case, it is for selecting the optimization solution to be applied on the link, which is appropriate for the link bottleneck problem. This component has an element that can trace the incoming client application SYN requests from any WAN link, so that optimization technique will be applied to the application server before it starts sending applications object to the client. The action rules are predefined by the researcher based on the application performance result achieved by each actions in the experimental study. Some of the predefined rules are described below:

- If protocol bottleneck is true and there is no congestion bottleneck in any WAN link, then increase TCP window size of email and database applications traffic flow as new SYN request comes over the WAN link for these applications.
- If protocol and congestion bottlenecks are true in any WAN link, then enable "TCP NODELAY" (disable Nagle's algorithm) on email and database applications traffic flow as new SYN request comes over the WAN link for these applications.
- If protocol and congestion bottlenecks are true in any WAN link then enable TCP slow start on SMTP protocol as new email SYN request comes over the WAN link.
- If protocol and congestion bottlenecks are true in any WAN link then enable TCP extensions such as "Fast Retransmit" and "Fast Recover for Database application as new SYN request comes over the WAN link for these applications.
- If latency bottleneck is true and there is no bandwidth bottleneck in any WAN link, then increase the TCP window size of web application (to send more data in parallel) as new SYN request comes over the WAN link.
- If latency bottleneck is true in any WAN link, then enable caching on clients for web application.
- If latency bottleneck is true and there is no bandwidth bottleneck in any WAN link, then increase application segment size on the server as new SYN request comes over the WAN link (for voice application). Since voice shows no protocol bottleneck in the experiment, TCP Window size will be adjusted automatically.
- If bandwidth and congestion bottlenecks are true in any WAN link, then decrease the data applications segment size on the server: as new SYN requests comes over the WAN link. (email, database and web)

- If congestion bottleneck is true in any WAN link and if non critical enterprise applications are actively using the link, then call method to reschedule these applications in an order starting from domain applications, Monitoring traffics, replication traffics and drop application traffics using unknown ports.
- If bandwidth, latency and congestion bottlenecks are true in any WAN link, then apply server based compression techniques on applications traffic as new SYN request comes over the WAN link.
- If congestion bottleneck is true in any WAN link and if video and voice applications are actively using the link then decrease the segment size of critical data applications (email/web/IMIS database) for low bandwidth links and increase segment size of voice applications for high latency and high bandwidth links.

This component uses Packet tracer, Packet classifier and optimization selection algorithms to accomplish its task and it is initialized by packet tracer algorithm. The next section will explain how these algorithms integrated to accomplish the tasks of the component.

Event Reporter:

Event reporter provides a report about each problem identified and the action taken to solve it. Moreover, it periodically recommends a lasting solution for the most seen problems like deploying VPN, IP-tunneling or QOS and New version of TCP protocol (SACK).

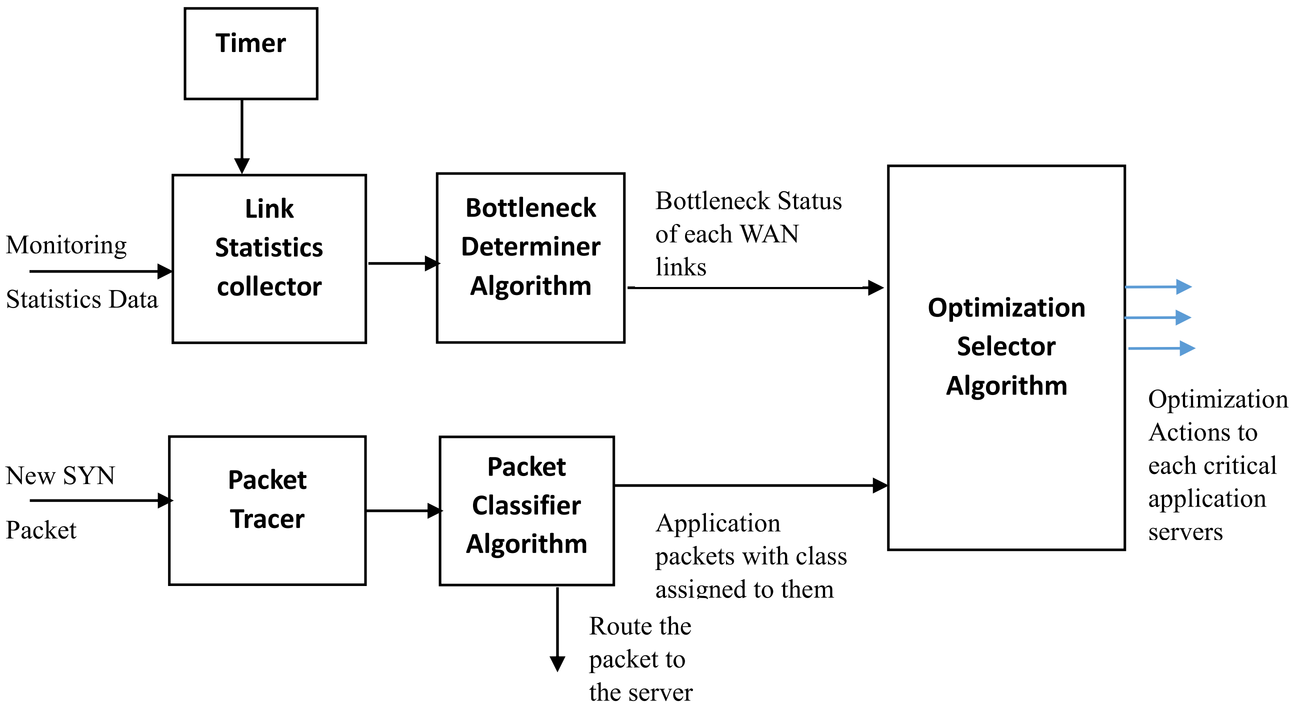


Figure 5.2 the system architecture of the optimization Model

5.2.3 Developing WAN Optimization Algorithms

The WAN Optimization system is designed to be deployed before the gateway router in the data center. The system has two stages, and it uses different algorithms for each stage to accomplish the optimization tasks. Basically, the algorithm is designed based on the required operations by the system in both stages as described below:

In stage 1, the reports derived from the monitoring tool about each WAN links is used to determine the real time status of each WAN links in terms of various application bottlenecks mentioned in section 5.2.1. In other words, the information provided by Network condition recognizer is used by WAN bottleneck determiner component to label the status of each WAN links in terms of bottleneck.

In stage 2, the real time status of each WAN link that is determined in step1 is used to decide the list of actions that should be implemented to optimize the application traffic flow between data center and each branch office. The algorithm uses predefined rules to decide the list of appropriate actions for every application's TCP session. Finally implements these selected actions on all WAN links accordingly to optimize the performance of organizations' Business critical applications.

WAN Optimization Algorithm High level Flow Charts

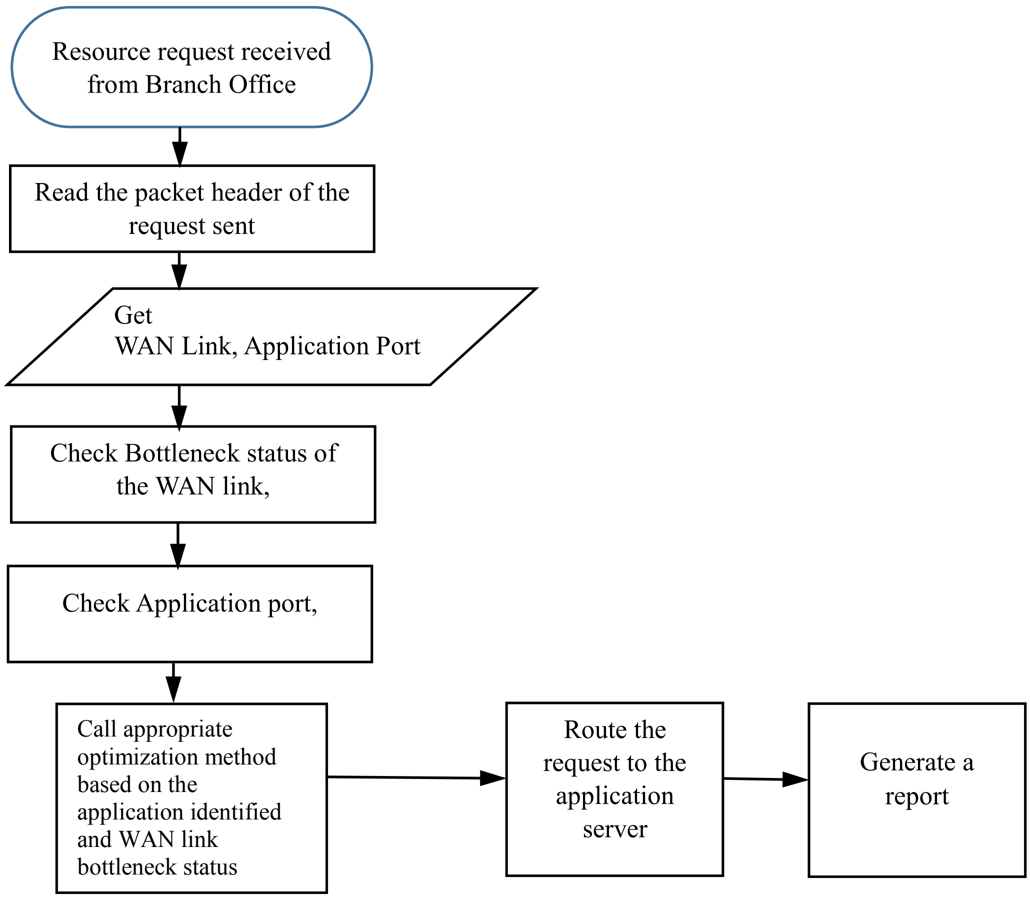


Figure 5.3 WAN optimization algorithm high level flow chart

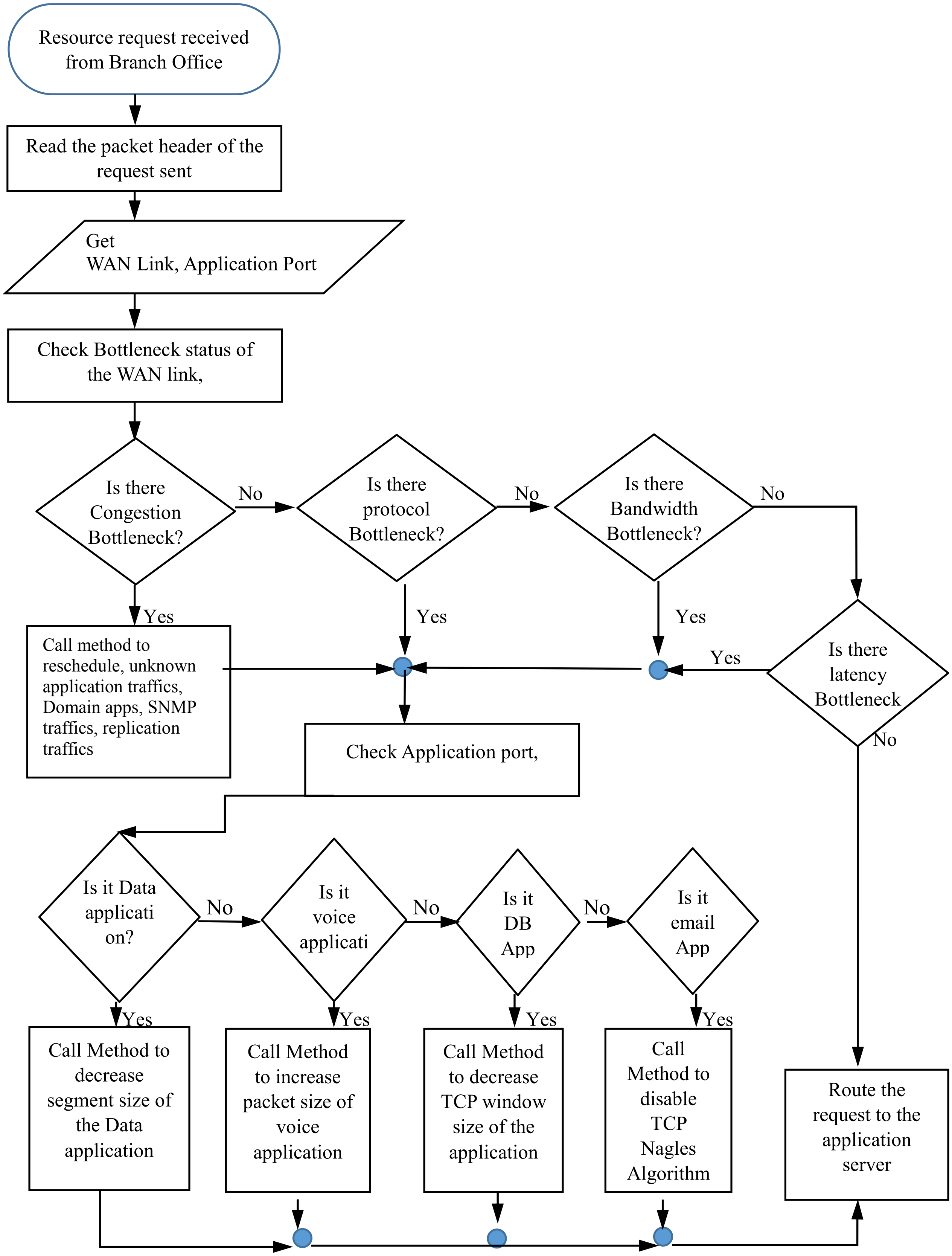


Figure 5.4 WAN optimization algorithm low-level flow chart

The low-level flowchart provides a detailed picture of a process by mapping all of the steps and activities that occur in the process. It indicates the steps or activities of a process and includes such things as decision points, waiting periods, tasks that frequently must be redone (rework), and feedback loops. These flowcharts are useful for examining areas of the process in detail and for looking problems or areas of inefficiency.

WAN Optimization Algorithm Pseudo-code

The Pseudo-code of the algorithm developed for two major stages of the system is presented below

Stage 1: Determining the bottleneck status of each Branch offices WAN links.

In this stage, the system uses three elements for determining the bottleneck status of each WAN link, which are Timer, Link statistics collector and Bottleneck determiner. The pseudo code for each element is presented below.

Timer: To initiate the monitoring statistics collection every 2 minutes so that any change in WAN factors is detected in 2 minutes time.

```
BEGIN
    LABEL Start:
    INIT T1=currentSystemTime();
    FLAG time=1;
    WHILE(time)
        DO READ NextCurrentTime as T2
        IF(T2-T1==2 minutes) THEN
            CALL Link_Statstics_Collector( )
            FLAG time=0; //false
            GOTO Start;
        ENDIF
        NEXT WHILE
    ENDWHILE;
END
```

Link statistics collector: Extracts information from monitoring tools about selected WAN conditions for each WAN links and organize the information in a file. This function is initiated by a timer every two minutes. This function also calls bottleneck determiner and packet identifier functions, if there is any change in any of WAN factors. First, to initiate the process of determining, the real-time bottleneck status of the links, then to identify the new packet SYN requests, and optimize them.

Let file ***"Link Statistics"*** holds all WAN links monitoring information, in a format shown below. The algorithm function ***Link_Statstics_Collector*** writes monitoring statistic of each WAN links, in one line per each link

destIP,	macColln,	tcpReCount,	qDelay,	pDelay,	linkRate
IP address	####	####	###	###	###

```
Function: Link_Statstics_Collector( )
BEGIN
        Check whether file "Link Statistics" exists or not THEN
                IF yes THEN
                        OPEN file "Monitoring log" for INPUT (reading)
                        OPEN file "Link Statistics-2" for OUTPUT (writing)
                        For each line in file "Monitoring log"
                                WRITE destIP, macColln, tcpReCount, qDelay, pDelay, linkRate
                        END FOR
                        OPEN file "Link Statistics"
                        COMPARE "Link Statistics" and "Link Statistics-2"
                                IF different THEN
                                        RENAME  "Link Statistics-2" to "Link Statistics"
                                        CALL BottleneckDeterminer ( )
                                        CALL PacketIdentifier( )
                                ELSE
                                        REMOVE file   "Link Statistics-2"
```

```
                    END IF
        ELSE
                OPEN file "Monitoring log" for INPUT (reading)
                OPEN file "Link Statistics" for OUTPUT (writing)
                For each line in file "Monitoring log"
                        WRITE destIP, macColln, tcpReCount, qDelay, pDelay, linkRate
                END FOR
                CALL BottleneckDeterminer( )
                CALL PacketIdentifier( )
        END IF
END
```

Bottleneck Determiner: to determine the bottleneck status of each WAN links. The algorithm stores information about bottleneck status of all WAN links in list ***all_WAN_Links_Status[]*** and the algorithm populates the list in a format shown below.

WANLinkAddress; CongestionBottleneck; Bandwidthbottleneck; latencybottleneck; protocolbottleneck.

- Let the variable **macColln** denotes the presence of MAC collusions in the in the WAN link
- Let the variable PDelay denotes the presence of link propagation delay in the WAN link
- Let the variable qDelay denotes the presence of point to point queue delay in WAN link
- Let the variable linkRate denotes the presence of slow transmission rate in the WAN link
- Let the variable tcpReCount denotes the presence of slow transmission rate in the WAN link
- Let the variable destIP denotes the IP address of the Gateway router in the branch office
- Let the flag congestedBottleneck denotes the presence of congestion Bottleneck problem in a WAN link
- Let the flag **bandwidthBottleneck** denotes the presence of transmission rate Bottleneck problem in a WAN link
- Let the flag **latencyBottleneck** denotes the presence of propagation delay Bottleneck problem in a WAN link
- Let the flag **protocolBottleneck** denotes the presence of TCP protocol Bottleneck problem in a WAN link

- Let the string **WANLinkAddress** denotes the IP address of the Router located in each Branch offices LAN
- Functions: **Baseline_macCollusion(), Baseline_tcpRetranCount(), Baseline_queueDelay(), Baseline_propagationDelay()** and **Baseline_linkRate()** is deigned to determine the average value of each conditions for WAN links (3 month to 1 year period is chosen based on the WAN environment.

```
Function: BottleneckDeterminer( )
BEGIN
    DECLARE: Global variables
    congestedBottleneck, protocolBottleneck latencyBottleneck, bandwidthBottleneck
    OPEN file "Link Statistics" for input(reading)
    READ destIP, macColln, tcpReCount, qDelay, pDelay, linkRate
    all_WAN_Links_Status = [None]  *  5   ## Initializing the List that holds bottleneck status of each WAN link
    i = 0
    DOWHILE(NOT EOF)
        IF macColln > Baseline_ macCollusion() AND tcpReCount >
           Baseline_tcpRetranCount() THEN
            IF qDelay > Baseline_queueDelay() THEN
                congestedBottleneck = TRUE
                protocolBottleneck = FALSE
            ELSE
                congestedBottleneck = FALSE
                protocolBottleneck = TRUE
            END IF
        ELSE
            congestedBottleneck = FALSE
            protocolBottleneck = FALSE
        END IF
```

```
            IF pDelay > Baseline_ propagationDelay() THEN
                latencyBottleneck = TRUE
            ELSE
                latencyBottleneck = FALSE
            END IF
            IF linkRate > Baseline_linkRate() THEN
                bandwidthBottleneck = TRUE
            ELSE
                bandwidthBottleneck = FALSE
            END IF
            WAN_Link_status = {}
            WAN_Link_status [“WAN Link Address”] = destIP
            WAN_Link_status [“Congestion Bottleneck Status”] = congestionBottleneck
            WAN_Link_status [“Protocol Bottleneck Status”] = protocolBottleneck
            WAN_Link_status [“Latency Bottleneck Status”] = latencyBottleneck
            WAN_Link_status [“Bandwidth Bottleneck Status”] = bandwidthBottleneck
            all_WAN_Links_Status [i] = WAN_Link_status
            i = i + 1
            READ Next record
        ENDWHILE
        RETURN all_WAN_Links_Status
END
```

Stage 2: Selecting and applying the optimization solution technique, as request is received from any of the WAN links, based on the real time bottleneck status of the WAN link that is determined in the first phase.

The optimization model has three components for accomplishing this phase. The first one is Packet tracer for detecting the incoming packet from branch office WAN and determine whether it is a new request or not. Packet classifier is the second module, which classifies the SYN packets with the application request type that it represents. Finally, optimization selection algorithm, which selects and applies the optimization solution technique appropriate for the enterprise WAN real time situation.

Input = packet of resource request

Source IP address; Destination application port=http or email or database or voice or video

Output = actions

Packet Identifier: To receive SYN request from any WAN link and identify the source address and application type of the SYN request so that optimization actions is initiated. Example SYN packet header is shown below:

```
Connection
server_ip=123.38.194.79
server_port=443
client_ip=52.57.188.150
client_port=1224
protocol=6
conn_rtt=0.301852
conn_starttime=1148515201.006697
```

The algorithm stores information about the source address, destination address and application type of the SYN request in a list ***syn_packet_header[]*** and the algorithm populates the list in a format shown below.

srcIP; ApplicationType; desIP

```
Function: PacketIdentifier( )
BEGIN
    DECLARE Global Varibales
    syn_Packet_Header = [None] * 10  ## for holding the Source IP address and application type of the request from WAN
    i = 0
    FOR(each incoming SYN packet from WAN link in 2 min) # because within 2 minute new call comes
        READ Packet header
        GET srcIP, dstPort, dstip
        IF Packet_classifier(dstport) <> non_critical_apps THEN
```

```
                FOR (j =0 TO j = i -1)  # not to process the same source SYN packet
                    IF syn_packet_Header[j][0] <> srcIP AND syn_packet_Header[j][1] <>
                      dstport
                        syn_packet_Header[i] = [srcIP, Packet_classifier(dstport),
                        dstIP]
                        RETURN syn_packet_Header[i]
                        i = i + 1
                        CALL OptimizationSelector( )
                    END IF
            END IF
        END FOR
END
```

Packet classifier: To classify SYN packets to the type of requested application that it represents

```
Function: Packet_classifier(i )
BEGIN
        DECLARE Global Varibales
        IF i = 443 OR 80 OR 25 OR 1352 OR imis OR video OR voice
            IF i = 443 OR 80 THEN
                RETURN web
            IF i = 25 OR 1352 THEN
                RETURN email
            IF i = #### THEN
                RETURN imis
            IF i = #### THEN
                RETURN voice
            IF i = #### THEN
                RETURN video
            END IF
            END IF
            END IF
            END IF
            END IF
        ELSE
            RETURN non_critical_apps
        END IF
END
```

Optimization Selector: To select and apply appropriate optimization solution. The reporting module is also included here to log each optimization action.

Let the file ***"Event log"*** holds each reported bottleneck problem and the associated action taken.

```
Function: OptimizationSelector( )

BEGIN

    FOR (i=0 to count(all_WAN_Links_Status[]))

        IF syn_packet_Header[0] = all_WAN_Links_status[i] ["WAN Link Address"] THEN

            IF all_WAN_Links_status[i] ["Congestion Bottleneck Status"] = TRUE THEN

                FOR the wan link = all_WAN_Links_status[i] ["WAN Link Address"]
                CALL method to reschedule Domain apps
                CALL method to reschedule Monitoring traffics
                CALL method to reschedule replication traffics
                CALL method to drop unknown application traffics

                WRITE Bottleneck status and the called action method to file "Event log"

                END FOR

            IF all_WAN_Links_status[i] ["Congestion Bottleneck Status"] = TRUE AND
               all_WAN_Links_status[i] ["Bandwidth Bottleneck Status"] = TRUE THEN

                FOR server with IP address = syn_packet_Header[2]

                    IF syn_packet_Header[1] = email THEN
                        CALL method to decrease segment size of email application

                    IF syn_packet_Header[1] = web THEN
                        CALL method to decrease segment size of web application

                    IF syn_packet_Header[1] = database THEN
                        CALL method to decrease segment size of DB application

                WRITE bottleneck status and the called action method to file "Event log"

                END FOR

                    END IF
                    END IF
                    END IF

            IF all_WAN_Links_status[i] ["Protocol Bottleneck Status"] = TRUE AND
              all_WAN_Links_status[i] ["Congestion Bottleneck Status"] = FALSE THEN
                FOR server with IP address = syn_packet_Header[2]
```

```
                IF syn_packet_Header[1] = database THEN
                        CALL method to increase TCP Window size for DB APP

                IF syn_packet_Header[1] = email THEN
                        CALL method to increase TCP Window size for email APP

        WRITE bottleneck status and the called action method to file "Event log"

        END FOR

                END IF
                END IF

IF all_WAN_Links_status[i] ["Protocol Bottleneck Status"] = TRUE AND
   all_WAN_Links_status[i] ["Congestion Bottleneck Status"] = TRUE THEN
        FOR server with IP address = syn_packet_Header[2]

                IF syn_packet_Header[1] = database THEN
                        CALL method to enable TCP "Fast Retransmit" for DB APP

                IF syn_packet_Header[1] = email THEN
                        CALL method to enable TCP slow start algorithm for the APP

                IF syn_packet_Header[1] = email OR database THEN
                        CALL method to disable Nagle's algorithm on the APP server

        WRITE Bottleneck status and the called action method to file "Event log"

        END FOR

                END IF
                END IF
                END IF

IF all_WAN_Links_status[i] ["Latency Bottleneck Status"] = TRUE AND
   all_WAN_Links_status[i] ["Bandwidth Bottleneck Status"] = FALSE THEN
        FOR server with IP address= syn_packet_Header[2]

                IF syn_packet_Header[1] = web THEN
                        CALL method to increase TCP window size for web APP

                IF syn_packet_Header[1] = voice THEN
                        CALL method to increase segment size of voice application

        WRITE Bottleneck status and the called action method to file "Event log"

        END FOR

                END IF
                END IF
```

```
IF all_WAN_Links_status[i] [“Bandwidth Bottleneck Status”] = TRUE AND
   all_WAN_Links_status[i] [“Latency Bottleneck Status”] = TRUE AND
   all_WAN_Links_status[i] [“Congestion Bottleneck Status”] = TRUE THEN
        FOR server with IP address = syn_packet_Header[2]

                Call method that apply compression techniques for each applications

        WRITE Bottleneck status and the called action method to file “Event log”

        END FOR

END IF
END IF
END IF
END IF
END IF
END IF

        END IF
    END FOR
END
```

CHAPTER SIX
CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions

Centralizing enterprise applications reduces the cost and complexity of branch office IT infrastructure in addition, it improves security and data protection practices. Organizations can gain these advantages by deploying WAN and consolidating their distributed applications in the data center. While the benefits are apparent, application centralization also presents a performance challenge, particularly for low resourced WAN environments.

This thesis was set out to investigate the performance problems of business-critical applications and propose an optimization solution. The research has identified the type and nature of various network factors in UNECA WAN environment. The study has also sought to determine the appropriate optimization technique for reducing the impact of each factor, particularly for low resourced WAN environment. In general, the study required to answer the following vital questions:

- How does the enterprise WAN network performance bottleneck manifest itself?
- What is the impact level of different WAN factors on end users response time for each critical application? And
- How is it possible to reduce their impact and improve application performance over the WAN in real-time?

Experimental analysis has been conducted on cases taken from the UNECA WAN environment using the test bed environment and the performance bottlenecks revealed in the enterprise WAN have been pinpointed. Some factors, which have not been considered in previous research attempts, are also unveiled in this study. The main empirical finding that answers the study's first-two research questions from the enterprises' WAN perspective is synthesized as follows:

- Email segment delay is one of the factors for the email application performance problem. UNECA ERP application on the other hand has high TCP retransmission count which results in poor database application performance.
- End to end delay (packet delay and point-to-point queue delay) has a direct impact on email Performance. For web application performance, point-to-point queue delay (mainly the receiving side) has significant impact than packet transmission delay.
- Voice demands low IP packet latency from the WAN link than video, in other words voice demands high application packet receiving rate than video. Voice applications also demand very low packet delay variation, in addition to low delay.
- Http protocol chattiness is a bottleneck for better Web application performance.
- When there is a link congestion multimedia traffics (Voice and Video) shows better traffic usage than Data traffic. Voice utilizes the bandwidth well but with high end-to-end packet delay.
- During congestion among data applications: Http application performs better, but Email and database applications become unresponsive. As the analysis result explains, when the link is congested email and database applications' TCP connection rate becomes slow and applications using UDP protocol, which needs no connection establishment, performs better than TCP dependent applications.
- Voice application, which demands low latency, email and database applications, can get impacted highly by packet loss and high retransmission count than others.
- Web applications affected by background traffic than email. Database applications are considerably impacted by the high volume of background traffic change comparing to Email and IMIS (database) applications. Voice application performance is affected by background traffic, which implies that the voice application needs more throughput over WAN links.
- When a Video conferencing traffic flow is used in the experiment without background load end users LAN delay becomes high, on the contrary when it is used with background load, LAN delay becomes low. From this experimental analysis result, it is clear that video applications need a relatively lower throughput than voice applications.

In regards to performance challenges, this study finds out that, WAN bottlenecks, such as congestion, bandwidth, latency and protocols inefficiency are the prime causes of the application performance problem. These findings explain how network conditions and application protocols itself have an impact on the way applications work when used in WAN. And also justifies, how evaluating and identifying application specific performance issues is a key task for developing optimization solutions, since each application behave differently in different situations.

To answer the third research question various optimization techniques have been evaluated in the test bed environment and the techniques listed below have shown good performance result, for different WAN conditions.

- Modifying the TCP window size on the application server to reduce TCP segment delay: for protocol bottleneck.
- Enabling TCP extensions such as “Fast Retransmit” and “Fast Recover for protocol bottleneck
- Enabling “TCP NODELAY” (disabling Nagle’s algorithm) to make TCP transfer small segments of the application: for protocol bottleneck.
- Modifying the TCP protocol version or type on the application server: for protocol bottleneck
- Forcing SMTP to use a TCP slow start algorithm for SMTP protocol bottleneck
- Caching the dialog information of chatty applications as much as possible on the client side, for latency bottleneck.
- Increasing application segment size, to reduce the protocol overhead, for latency bottleneck.
- Increasing TCP window size of chatty applications using a high bandwidth and high latency WAN link, for latency bottleneck.
- Controlling Non- critical application traffic flow by rescheduling applications in an order starting from domain applications, Monitoring traffics, and replication traffics and by dropping application traffics using unknown TCP ports, for congestion and bandwidth bottleneck
- Decreasing the application segment size on the application server: for congestion and bandwidth bottleneck.
- Applying server based compression techniques on application traffic for bandwidth and congestion bottleneck.

The study developed the WAN optimization Model, based on the experimental findings on the issue of performance bottlenecks and proved optimization techniques. The developed model has an algorithm with features listed below:

- The optimization technique is applied based on the real time WAN conditions. The model has a Network Condition Recognizer element, which monitors the real-time status of each WAN link and notify for the application optimizer element. As soon as this element notified about the presence of the problem, it applies the appropriate optimization technique for each active application on the WAN links and the optimization process continues until the problem is solved.
- The optimization Model is application dependent, because it applies different techniques to different applications based on their behavior/reaction for the real-time WAN conditions. The optimization Model uses its predefined rules to determine the appropriate optimization technique for each type of application based on their behavior, which is identified in the experiment.
- The optimization Model is ideal for enterprise WANs with network resource issues. The major bottlenecks are determined based on the experimental study done on poor performing WAN links due to low bandwidth and significant packet delay. The network conditions introduced by such networks are analyzed in this thesis and various optimization techniques are experimented to select the best techniques to minimize the performance effect of these conditions. As a result, this optimization model can be a better solution to improve the application performance particularity for enterprise WANs with low bandwidth and poor infrastructure resource.
- The Optimization Model can be tuned easily to optimize any newly developed applications and enhanced existing applications. Since the solution is application dependent, the new introduced application can be integrated into the model by only updating the solution's rule with the new application behavior.

This study has used empirical findings to develop the WAN optimization Model. The performance result suggests that developing an optimization solution, after investigating the reaction of each application for various WAN bottlenecks is an effective approach. Moreover, the approach makes the solution, application specific and capable of integrating new as well as enhanced applications easily.

The output of this research has proven that, when server based applications, accessed and used through enterprise WANs, they behave and react differently to different conditions of the WAN. Hence, WAN optimization solutions that apply optimization techniques on application traffic, based on the real time conditions of the WAN is a viable solution particularly for organizations with under-resourced WAN environment. In other words, the optimization solution that is designed after analyzing the real-time behavior of applications provides better performance improvement. Since the algorithm of the developed WAN optimization Model is designed to select and apply optimization technique based on the real time conditions of the WAN, it shows a better response on improving the performance of critical applications, which are used over a low resourced enterprise WAN.

6.2 Recommendations

The study has been conducted in UNECA WAN environment and the performance result is promising. However, further work has to be done to improve the optimization algorithm so that it can include more real time conditions and to improve the efficiency of the algorithm.

One immediate possible future work that can be commenced from this research is to codify, apply and test the prototype Model in the enterprise WAN environment. Since the Model created is a prototype and the model validation is done on simulated environment, it has to be tested and validated on the actual WAN environment.

Another recommendation in the future attempts in this area is applying the model for optimizing others applications and application protocol's performance that are not considered in the experimentation process of this study. The attempt can also consider introducing some additional bottlenecks and WAN factors to measure the existence of bottlenecks other than queue delay, Mac collusion, TCP retransmission count, TCP delay and packet rate, which are used in this research.

Furthermore, improving the real-time operations of the model can be an important future attempt considering its advantage for improving the performance of enterprise applications and as a whole productivity of the branch office workforce for an organization having a centralized resource but decentralized employees in different geographical locations.

The efficiency issue has two sides; the first one is, enhancing the efficiency of the bottleneck determiner algorithm specially for improving the WAN Optimization Model capability of giving a real time solution. The other side is, enhancing the efficiency of the Optimization selector algorithm particularly for minimizing the time taken by the algorithm for processing the packet.

Finally, application performance issues have been there since the advent of Wide Area Networking technologies. Since there is always an effort by application designers for developing new enterprise applications and enhancing the existing ones, applications are getting robust and complex. Hence, there should be a consistent research attempts for improving the performance problem of these new and complex applications when being used over the WAN environment.

REFERENCES

Aberdeen Group. (2007, October). *Optimizing WAN for Application Acceleration.* Retrieved from www.aberdeen com: http://www.riverbed.com/docs/WhitePaper-Riverbed-ApplicationDeliveryOverTheWAN.pdf

Aberdeen Group. (2009, March). *A Smarter Approach to WAN Optimization.* Retrieved from http://www1.exinda.com

Abrams, M., Standridge, C. R., Abdulla, G., Fox, E. A., & Williams, S. (1996). Removal Policies in Network Caches for World-Wide Web Documents. *Proc. SIGCOMM,* , (pp. 293–305). California, USA.

Al-laham, M., & Emary, I. M. (April 2007). Comparative study between various algorithms of data compression techniques. *International Journal of Computer Science and Network Security, vol. 7, no. 4*.

Anand, A., Muthukrishnan, C., Akella, A., & Ramjee, R. (2009). Redundancy in network traffic: findings and implications,. *Proc. 11th International Joint Conference on Measurement and Modeling of Computer Systems*, (pp. 37–48). Seattle, WA, USA.

Aron, M., & Druschel, P. (1998). Tcp: Improving start-up dynamics by adaptive timers and congestion control. *Rice University, Tech. Rep.TR98-318*.

Ashton, M. &. (2002, October). *The three components of Optimizing WAN Bandwidth.* Retrieved from http://www.ashtonmetzler.com: http://www.ashtonmetzler.com/BW_Optimization.pdf

Bai, G., Oladosu, K., & Williamson, C. (2007). Performance benchmarking of wireless Web servers. *Science Direct*.

Balakrishnan, B. (2011, September 11). *Case study and analysis of WAN Optimization pre-requirements.* Retrieved from Cornell University Library: http://arxiv.org/abs/1109.2285

Bartlett, J., Sevick, P., & Moore, S. (2008). Economics of QOS on WAN Access Lines. *EBSCO*.

Besson, E. (2000). Performance of TCP in a Wide-Area Network: Influence of Successive Bottlenecks and Exogenous Traffic. *Proc. Global Telecommunications Conference, vol. 3*, (pp. 1798 –1804.).

Bestavros, A., & Kim, G. (June 1997.). Implementation and Performance Evaluation of TCP. *The IEEE International Conference on Communications,.* Montreal, Canada,: IEEE.

Blue Coat. (2007). *Application Performance Brief: Lotus Notes.* Retrieved from bluecoat: http://www.bluecoat.com/sites/default/files/documents/files/Lotus_Notes_Applications.pdf

Bolosky, W. J., Corbin, S., Goebel, D., & Douceur, J. R. (Aug. 2000). Single Instance Storage in Windows 2000. *Proc. 4thconference on USENIX Windows Systems Symposium*, (pp. 13–24). Seattle, Washington.

Brakmo, L., & Peterson, L. (Oct. 1995). Tcp vegas: end to end congestion avoidance on a global internet. *IEEE J. Sel. Areas Commun., vol. 13, no. 8*, 1465 –1480.

Brassil, J., McGeer, R., Rajagopalan, R., Sharma, P., & Yalagandula, P. (Jan. 2009). High-Performance, Fair Transport Architecture Based on Explicit-Rate Signaling. *Operating Systems Review, vol. 43*, 26–35.

Cecchet, E., Udayabhanu, V., & Wood, T. (2011). BenchLab: An Open Testbed for Realistic Benchmarking of Web Applications. *2nd USENIX Conference on Web Application Development (WebApps '11)* (pp. 37-58). Portland, OR, USA: The USENIX Association.

Chakravorty, R., Katti, S., J, C., & I, P. (April 2003). Flow Aggregation for Enhanced TCP over Wide-Area Wireless. (pp. 1754 – 1764). Proc. INFOCOM.

Chou, W. (2002). *Inside SSL: The Secure Sockets Layer Protocol.* Retrieved from IEEE: www.ieee.org

Chou, W. (2009). Optimizing the WAN between Branch Offices and the Data Center. *IEEE C omputer S ociet y*, 1-4.

Christner, J., Seils, Z., & Jin, N. (2010). *Deploying Cisco Wide Area Application Services.* Indianapolis, IN 46240 USA: Cisco Press.

Cisco. (2008, August 22). *Internetworking Technology Handbook.* Retrieved from Cisco: http://www.cisco.com/en/US/docs/internetworking/technology/handbook/Intro-to-LAN.html

Clarke, R. J. (1999). Image and video compression: a survey. *International Journal of Imaging Systems and Technology, vol. 10*, 20–32.

Cole, R., & Ramaswamy, R. (2000). *Wide-Area Data Network Performance Engineering.* Boston . London: Artech House.

Dierks, T., & Allen, C. (1999). *The TLS Protocol Version 1.0.* Retrieved from IETF: http://www.ietf.org/rfc/rfc2246.txt

Dierks, T., & Rescorla, E. (2008). *The Trasport Layer Security (TSL) Protocol Version 1.2.* Retrieved from IETF: http://www.ietf.org/rfc/rfc5246.txt

Edwards, J., & Bramante, R. (2009). *Networking Self-Teaching Guide: OSI, TCP/IP, LAN's, MAN's,.* John Wiley & Sons (US), Wiley Publishing, Inc.

Feng, W., Balaji, P., Baron, C., Bhuyan, L., & Panda, D. (Aug. 2005). Performance characterization of a 10-Gigabit Ethernet TOE. *Proc. 13th Symp.on High Performance Interconnects*, (pp. 58 – 63).

Floyd, S. (March 2004). *Limited Slow-Start for TCP with Large Congestion Windows.* RFC 3742.

Forrester Consulting. (2010, October). *Managing the performance of Critical applications.* Retrieved from http://www.opnet.com/whitepapers/Managing_The_Performance_Of_Critical_Applications31-1.pdf

Grevers, T., & Christner, J. (2007). *Application Acceleration and WAN Optimization Fundamentals.* Cisco Press.

Hoe, J. C. (1996). Improving the start-up behavior of a congestion control scheme for tcp. *Proc. SIGCOMM, Palo Alto*, (pp. 270–280). California, United States.

Hu, N., & Steenkiste, P. (Nov. 2003). Improving tcp startup performance using active measurements: algorithm and evaluation. *Proc. 11th IEEE International Conference on Network Protocols*, (pp. 107 – 118).

Hunt, J. J., Vo, K.-P., & Tichy, W. F. (1996). An Empirical Study of Delta Algorithms. *Proc. SCM-6 Workshop on System Configuration Management*, (pp. 49–66).

IXIA. (2013, January). *Network Applications: Ensuring Performance and QoE with Network Emulation.* Retrieved from Anue White Paper Series: www.ixiacom.com/pdfs/library/white_papers/Network-Applications.pdf

Iyer, S., Rowstron, A., & Druschel, P. (2002). Squirrel: a decentralized peerto-peer web cache. *Proc. 21th annual symposium on Principles of distributed computing* , (pp. 213–222). Monterey, California .

Jenson, S. (2009). Consolidated Tactical Network Analysis for Optimizing Bandwidth: Marine Corps Support Wide Area Network (Swan) and TCP Accelerators. *Master's thesis, Naval Postgraduate School,*. MONTEREY, CALIFORNIA.

Juniper Networks, Inc. (2005). *Accelerating Application Performance Across the WAN.* Retrieved from www.juniper.net: www.juniper.net/us/en/products-services/

Kansanen, M. (2009). Wide Area Network Acceleration in Corporate Networks. *Master's Thesis, Lappeenranta University of Technology*.

Kelly, T. (2003). Scalable TCP: Improving Performance in Highspeed Wide Area Networks. *ACM SIGCOMM Computer Communications Review. Volume 33, Number 2 [retrieved May 20, 2008]. From: ACM at:http://portal.acm.org/portal.cfm*.

Kerravala, Z. (2009). *Mobile WAN Optimization Can Help Cut Costs and Avoid Future Upgrades.* Retrieved from Yankee Group Research, Inc.

Kozierok, C. (2005). *The TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference.* No Starch Press.

Kroeger, T. M., Long, D. D., & Mogul, J. C. (Dec. 1997). Exploring the Bounds of Web Latency Reduction from Caching and Prefetching. *Proc.USENIX Symp. on Internet Technologies and Systems.*

Ladiwala, S., Ramaswamy, R., & Wolf, T. (March 2009). Transparent TCP Acceleration. *Computer Communication, vol. 32*, 691–702.

Li, Z., Zhang, M., Zhu, Z., Chen, Y., Greenbergy, A., & Wang, Y.-M. (2010). WebProphet: Automating Performance Prediction for Web Services. *NSDI 2010,* (pp. 143-158). San Jose, California, USA: USENIX Association 2010.

Lu, X., Zhang, K., Fu, C. P., & Foh, C. H. (2010). A sender-side tcp enhancement for startup performance in high-speed long-delay networks. *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, (pp. 1 –5).

Mandagere, N., Zhou, P., Smith, M. A., & Uttamchandani, S. (2008). Demystifying Data Deduplication. *Proc. Middleware Conference Companion,* , (pp. 12–17). Leuven, Belgium.

Mirza, M., Sommers, J., Barford, P., & Zhu, X. (2010). A Machine Learning Approach to TCP Throughput Prediction. *IEEE/ACM Transactions on Networking, Volume 18, Issue 4*.

Nagle, J. (1984). *Congestion Control in IP/TCP Internetworks.* Network Working Group: RFC 896.

Nahum, E. M., Rosu, M.-C., Seshan, S., & Almeida, J. (2001). The effects of wide-area conditions on WWW server performance. *SIGMETRICS*.

Newman, D. (Aug 2007). WAN acceleration offers huge payoff. *Network World*.

Nishiyama, H., Ansari, N., & Kato, N. (April 2010). Wireless loss-tolerant congestion control protocol based on dynamic aimd theory. *IEEE Wireless Commun., vol. 17, no. 2*, 7 –14.

Noble, T. J. (2002). The End-to-End Performance Effects of Parallel TCP Sockets on a Lossy Wide-Area Network. *IEEE*.

Norton, T. R. (1999). End-To-End Response Time: Where to Measure? *CMG99 Session 423,* . Reno, Nevada: Computer Measurement Group.

OPNET. (2008). *OPNET Modeler Documentation Set.* Bethesda, Maryland: OPNET Technologies, Inc., 7255 Woodmont Avenue, Bethesda, Maryland 20814.

Paxson, V. (1994). Growth Trends in Wide-Area TCP Connections. *IEEE Network , vol. 8, no. 4*, 8 –17.

Peng, A. S., & Lilja, D. J. (2006). Performance Evaluation of Navy's Tactical Network using OPNET. *Military communication conference, MILCOM* (pp. 1- 7). Washington, DC: IEEE.

Postel, J. B. (1981). *Internet Control Message Protocol.* Network Working Group: RFC 792.

Puri, S. (2008). *Recommendations for Performance Benchmarking .* InfoSys.

Rabinovich, M., Chase, J., & Gadde, S. (November 1998). Not all hits are created equal: cooperative proxy caching over a wide-area network. *Computer Networks and ISDN Systems, vol. 30*, 2253–2259.

Rao, N., Poole, S., Hicks, S., Kemper, C., Hodson, S., & Lothian, G. H. (Oct. 2009). Experimental Study of Wide-Area 10 Gbps IP Transport. *Proc. Military Communications Conference (MILCOM)*, (pp. 1 – 7).

Riverbed. (2013). *WAN Optimisation vs. More Bandwidth – why increasing bandwidth is not always the answer.* Retrieved from http://www.cxo.eu.com/article/

Rizwana Mehboob, S. A. (2010). Multigig Lossless Data Compression Device. *IEEE*.

Rodriguez, P., Spanner, C., & Biersack, E. (Aug. 2001). Analysis of web caching architectures: hierarchical and distributed caching. *IEEE/ACM Trans. Netw., vol. 9, no. 4*, 404 –418.

Saha, S., Lukyanenko, A., & Yla-Jaaski, A. (April 2011). Combiheader: Minimizing the number of shim headers in redundancy elimination systems. *Proc. INFOCOM workshops on Computer Communications*, (pp. 798 –803).

Sakr, S. (2009). Xml compression techniques: A survey and comparison. *Journal of Computer and System Sciences, vol. 75, no. 5*, 303 –322.

Sevcik, P., & Wetzel, R. (2008, Novemeber). *Improving Effective WAN Throughput for Large Data Flows.* Retrieved from www.silver-peak.com: http://www.silver-peak.com/ assets/download/pdf/Netforecast wp EffectiveThroughput.pdf

Shukla, J., Alwani, M., & Tiwari, A. (April 2010). A survey on lossless image compression methods . *Proc. 2nd International Conference on Computer Engineering and Technology*, (pp. 136–141).

Shunra. (2009). *Best Practices for Selecting WAN Optimization Solutions: Benchmarking Performance ROI.* Retrieved from http://www.shunra.com/resources/white-papers

Shunra. (2009). *Best Practices for Selecting WAN Optimization Solutions: Benchmarking Performance ROI.* Retrieved from http://www.shunra.com/resources/white-papers

Siegel, E. (2006). *Optimizing WAN Performance: Accelerating Market Growth .* Retrieved from Burton Group. ISSN 1048-4620: www.burtongroup.com

Spring, N. T., & Wetherall, D. (2000). A protocol-independent technique for eliminating redundant network traffic. *Proc. SIGCOMM*, (pp. 87–95). Stockholm, Sweden.

Spurgeon, C. E. (2000). *Ethernet: The definite guide.* CA, USA : O'Reilly & Associates, Inc. Sebastopol, .

Stevens, W. R. (2001). *TCP/IP Illustrated, Volume 1.* Addison-Wesley Professional; US ed edition.

Sundararaj, A., & Duchamp, D. (2003). *Analytical Characterization of the Throughput of a Split TCP Connection.* Department of Computer Science, Stevens Institute of Technology, Tech. Rep.

Taneja Group. (2005). *Wide Area Data Services: Optimizing the Branch,.* Retrieved from Techworld: http://www.techworld.com/whitepapers/index.cfm?whitepaperid=4053

Tekala, M., & Szabo, R. (2005). Evaluation of Scalable TCP. *Proc. 3rd ACS/IEEE International Conference on Computer Systems and Applications.*

Tsykin, M., & Langshaw, C. D. (1999). Measurement of Service Levels. *Computer Measurement Group Transactions (95)*.

UNECA. (2013). *About ECA*. Retrieved from UNECA Official Website: http://www.uneca.org/pages/overview

Vakali, A. (2001). Proxy Cache Replacement Algorithms: A History-Based Approach,. *World Wide Web, vol. 4*, pp. 277–297.

Wang, H., & Williamson, C. (July 1998). A new scheme for tcp congestion control:smooth-start and dynamic recovery. *Proc. Sixth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, (pp. 69 –76).

Wang, R., Pau, G., Yamada, K., Sanadidi, M., & Gerla, M. (March 2004). Tcp startup performance in large bandwidth networks. *Proc. INFOCOM, vol. 2,*, (pp. 796 – 805).

Wikipedia. (2010). Retrieved from http://en.wikipedia.org/wiki/Benchmark

Willis, P. (2005). An introduction to quality of service. *BT Technology BT Technology Journal, Vol 23 No2*.

Xu, K., & Ansari, N. (April 2005). Stability and fairness of rate estimation-based AIAD congestion control in TCP. *IEEE Commun. Lett., vol. 9, no. 4,* , 378 – 380.

Xu, K., Tian, Y., & Ansari, N. (2004). TCP-Jersey for wireless IP communications. *IEEE J. Sel. Areas Commun., vol. 22, no. 4*, 747 – 756.

Xu, K., Tian, Y., & Ansari, N. (February 2005). Improving TCP Performance in Integrated Wireless Communications Networks. *Computer Networks, vol. 47, no. 2*, 219–237.

Xu, Z., Hu, Y., & Bhuyan, L. (2004). Exploiting client cache: a scalable and efficient approach to build large web cache. *18th International Parallel and Distributed Processing Symposium (IPDPS)*, (pp. 55–64).

Zeng, W. G. (2006). *TCP PACKET CONTROL FOR WIRELESS NETWORKS.*

Zhang, Y., Ansari, N., & Wu, M. (2011). On Wide Area Network Optimization. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 14, NO. 4, FOURTH QUARTER 2012*.

APPENDICES

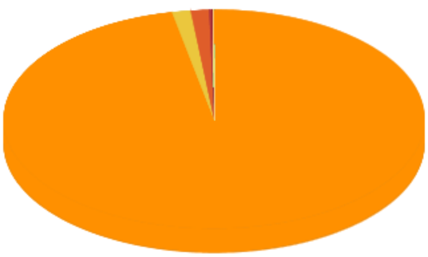
Appendix I: Consolidated Traffic Report for WAN link-1

Router Name: ECA-SRO-WAN
Interface Name: Serial0/1/1.1

Report Start time: 2013-05-11 14:00
Report End time: 2013-06-10 14:00

Top reports

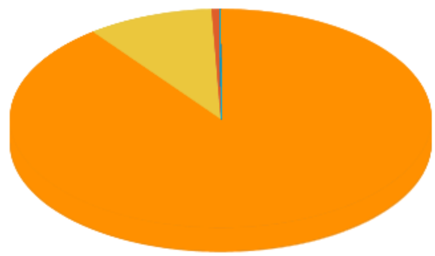
ApplicationIN



ApplicationIN Traffic % Traffic

h323hostcall	25.77 GB	97%
Unknown_App	377.31 MB	1%
palace-5	370.5 MB	1%
snmp	47.54 MB	<1%
EIGRP_App	33.25 MB	<1%
icmp	7.45 MB	<1%
snmptrap	4.58 MB	<1%
telnet	1.18 MB	<1%
isakmp	65.12 KB	<1%
http	30.72 KB	<1%
domain	29.66 KB	<1%
capmux	21.03 KB	<1%
ESP_App	15.84 KB	<1%
http-alt	9.91 KB	<1%
esro-gen	5.22 KB	<1%
https	3.7 KB	<1%
set	3.2 KB	<1%
ftp	432.00 Bytes	<1%
ipfixs	144.00 Bytes	<1%
remcap	144.00 Bytes	<1%
Others	19.79 MB	<1%
Total	26.63 GB	100%

ApplicationOUT



ApplicationOUT Traffic % Traffic

Unknown_App	558.89 GB	90%
h323hostcall	60.13 GB	10%
ncp	4.29 GB	1%
snmp	22.03 MB	<1%
telnet	563.9 KB	<1%
icmp	428.17 KB	<1%
lotusnote	149.66 KB	<1%
netbios-ns	111.61 KB	<1%
domain	7.72 KB	<1%
ftp	3.88 KB	<1%
https	2.49 KB	<1%
http	1.7 KB	<1%
capmux	1.28 KB	<1%
eoss	384.00 Bytes	<1%
ams	288.00 Bytes	<1%
excw	288.00 Bytes	<1%
qsm-gui	288.00 Bytes	<1%
watilapp	192.00 Bytes	<1%
tgp	192.00 Bytes	<1%
ssslic-mgr	192.00 Bytes	<1%
Others	18.93 MB	<1%
Total	623.36 GB	100%

Appendix II: Consolidated Traffic Report for WAN link-2

Router Name : ECA-SRO-WAN

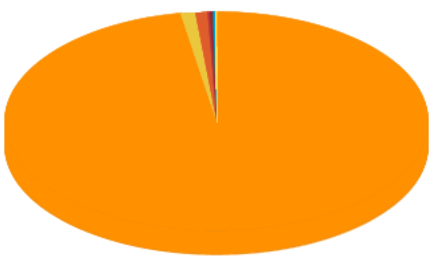
Interface Name : Serial0/0/0.1

Report Start time : 2013-05-11 14:50

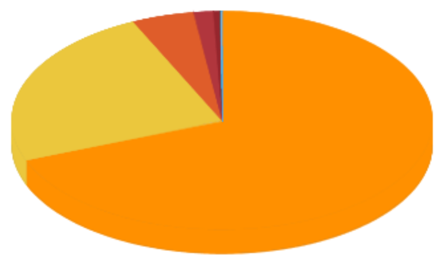
Report End time : 2013-06-10 14:50

Traffic Graph speed

Top reports ApplicationIN



ApplicationOUT



bps MAX : 9.54 Mbps 5th Per : 1.88 Kbps

bps MAX : 13.11 Kbps 5th Per : 1.85 Kbps

ApplicationIN Traffic % Traffic

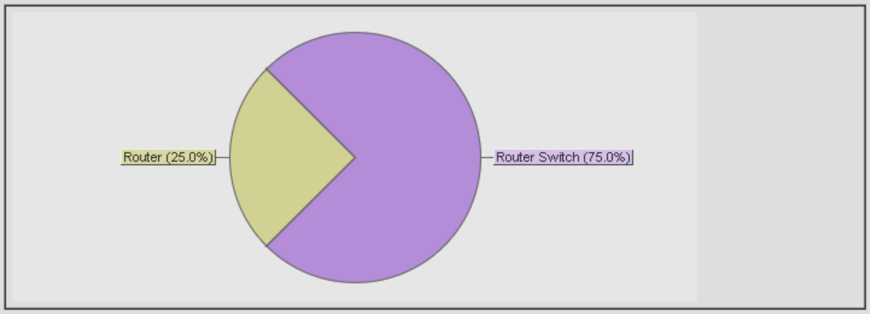
ncp	4.3 GB	97%
Unknown_App	49.46 MB	1%
snmp	41.54 MB	1%
EIGRP_App	8.79 MB	<1%
printer	7.5 MB	<1%
icmp	4.38 MB	<1%
snmptrap	3.32 MB	<1%
notify_srvr	1.42 MB	<1%
resource_mgr	1.42 MB	<1%
hp-pdl-datastr	751.48 KB	<1%
h323hostcall	610.16 KB	<1%
svrloc	227.04 KB	<1%
netbios-ns	128.44 KB	<1%
http	73.56 KB	<1%
ssh	28.83 KB	<1%
telnet	27.86 KB	<1%
spytechphone	14.11 KB	<1%
domain	4.83 KB	<1%
bootpc	1.96 KB	<1%
microsoft-ds	160.00 Bytes	<1%
Others	9.31 MB	<1%
Total	4.42 GB	100%

ApplicationOUT Traffic % Traffic

Unknown_App	70.3 MB	63%
snmp	24.44 MB	22%
ncp	4.9 MB	4%
icmp	1.53 MB	1%
h323hostcall	515.29 KB	<1%
netbios-ns	43.05 KB	<1%
http	32.4 KB	<1%
ssh	16.95 KB	<1%
telnet	11.32 KB	<1%
ftp	3.88 KB	<1%
Others	9.41 MB	8%
Total	111.22 MB	100%

Appendix III: Network Model Summary-OPNET

Device Type (total devices: 4)



Device Type	Count	% of Total
Router Switch	3	75.0%
Router	1	25.0%

Vendor (total vendors: 1)

Vendor	Count	% of Total
Cisco Systems	4	100.0%

Operating System (total OSs: 2)

Operating System	Count	% of Total
Native IOS	3	75.0%
Router IOS	1	25.0%

Ports (total ports: 64)

Port Type	Count	% of Total
Ethernet	34	53.1%
Serial (Frame Relay)	27	42.2%
Serial (IP)	3	4.7%

Other Devices (total devices: 34)

Device Type	Count	% of Total
Inferred Lan	21	61.8%
Inferred Workstation	10	29.4%
Inferred Switch	3	8.8%

Appendix IV: Users' Application Failure report

These sample reported problems are extracted from users request handling system:

"",88,"","",12/06/2012,ECAA-92QARJ,SRO-CA,"Bertrand Tachago","EMAIL REQUEST: Subject - link degardation between ECAHQ and SRO-CA. DETAILS: Dear colleagues, we're facing a degradation of link between ECAHQ and SRO-CA. The link is unstable and the the response time is very high. At the same time, the Internet link is OK. Kindly have a look. Thanks and warm regards === Bertrand TACHAGO IT Team leader United Nations Economic Commission for Africa Sub-Regional Office for Central Africa Tel: +237 22 22 08 61/ 22 22 08 56 Direct: +237 22 01 28 40 Fax: +237 22 23 31 85 Cell: +237 77 69 50 25 Email: btachago@uneca.org / tachago@un.org P.O. Box 14935 Yaounde, Cameroon === s.d"

"",90,"","",11/14/2012,ECAA-922KP6,"","Jean Baptiste Eken","IMIS - Open Call Template EMAIL REQUEST: Subject - Fw: Need for help for IMIS issues. DETAILS: ----- Forwarded by Jean Baptiste Eken/West/ECA on 11/14/2012 04:19 PM ----- From: Jean Baptiste Eken/West/ECA To: IMIShelpdeskECA/ECA@ECA, Frehiwot Getinet/ECA@ECA Date: 11/06/2012 10:53 AM Subject: Need for help for IMIS issues Good day to you. Just to inform you that i am not able to go through Report-report facility to down load report. The message i received is ""UNABLE TO RETRIEVE""; Please, can you help me ? Thank for your support. Jean Baptiste EKEN Administrative and Finance Officer United Nations Economic Commission for Africa (UNECA) Sub-Regional Office for West Africa (ECA/SRO-WA) P.O. BOX 744, Niamey/Niger Tel. (227) 20 72 29 61 / Ext 21524 Direct Line: (227) 20 72 36 22 /(227) 20 72 73 24 Cell: (227) 98 61 16 02 Fax: (227) 20 72 28 94 Official Email: jbeken@uneca.org Private Email: jbeken2002@yahoo.fr "

"",90,"","",07/26/2011,SDEK-8K5CU8,SRO-SA,"Ruth Kananda","We have not been able to connect to CTRIX since 10:00hrs this morning. Kindly note even telephone lines are not clear and internet connection is down too. AB"

"",90,"","",11/28/2012,ECAA-92GFFM,"","Jean Claude Umugaba","EMAIL REQUEST: Subject - Fw: Emails are delayed. DETAILS: Please check this problem. Thanks -- Jean Claude UMUGABA UN Economic Commission for Africa(UNECA) IT Focal Point SRO-EA Kigali Mob :(250)788353047, Ext.number :21614 ----- Forwarded by Jean Claude Umugaba/EAST/ECA on 11/28/2012 01:40 PM ----- From: Jean Claude Umugaba/EAST/ECA To: Service Desk/ECA@ECA Cc: Yacob Gobena/ECA@ECA, Zewdalem Shitaye/ECA@ECA Date: 11/21/2012 04:27 PM Subject: Emails are delayed Dear colleagues, please kindly check this issue. Emails are delayed. Thanks -- Jean Claude UMUGABA UN Economic Commission for Africa(UNECA) IT Focal Point SRO-EA Kigali Mob :(250)788353047, Ext.number :21614 "

"",90,"","",06/20/2012,ECAA-8VFCGL,"","Tommy Bannister","EMAIL REQUEST: Subject - Re: Fw: eLeave in the SROs. DETAILS: Dear Frehiwot, Good morning, Balkissa is having problem completing her leave request using the e-leave application. Balkissa please copy the screen shot to ITSS . _____________________________________ Tommy Dauda Bannister Associate Administrative and Finance Officer ECA Sub Regional Office for West Africa Niamey, Niger e-mail: tbannister@uneca.org Ext no: 21524 Office Tel : (227) 20723622 From: Frehiwot Yabowork/ECA To: SRO-NA All Staff, SRO-WA All Staff Cc: Lamin Fatty-ECA/ECA@ECA, Catherine Bascon/ECA@ECA, Edom Worku/ECA@ECA, Mariel Carrasco/ECA@ECA, Wondwossen Truneh/ECA@ECA Date: 06/18/2012 10:57 AM Subject: Fw: eLeave in the SROs Dear Colleagues, Further to the email exchange below, this is to advise SRO staff members that you can now use e-leave system which comprises the e-lump sum for Home Leave, Family Visit and Education Grant travels. Please let ITSS colleagues know if you encounter any problems. Thank you .... Frehiwot Yabowork (Ms) UNECA/DoA/Human Resource Services Section P. O. Box 3001 Addis Ababa, Ethiopia Ext. 33193 Tel. (251)-11-5443193 Fax (251)-11-5510489 E-mail: yabowork@uneca.org .... ----- Forwarded by Frehiwot Yabowork/ECA on 06/18/2012 10:07 AM ----- From: Catherine Bascon/ECA To: Jose German/ECA@ECA, Frehiwot Yabowork/EC"

"",90,"","",12/20/2011,ECAA-8PQEZ5,SRO-CA,"Michel Fogang","EMAIL REQUEST: Subject - access to eIOV. DETAILS: good morning dear colleagues, Since yesterday, we do not have access to eIOV, please could you look what wrong with that Thanks == Michel FOGANG IT Assistant United Nations Economic Commission for Africa Sub-Regional Office for Central Africa Tel : (+237) 22 22 08 61 / (+237) 22 22 08 56 ext : 21435 Fax : (+237) 22 23 31 85 Cells: +237 75639013 +237 99651469 Email: mfogang@uneca.org B.P. 14935 Yaounde, Cameroon"

"",102,"","",06/14/2012,ECAA-8V9H2W,SRO-SA,"Ronald Nkhoma","EMAIL REQUEST: Subject - Re: Degradation of Internet Services through Ethio Telecoms. DETAILS: Dear Service desk, We are unable to log into IMIS.

"",90,"","",03/07/2012,ECAA-8S6CPZ,"","Denise Guce","EMAIL REQUEST: Subject - SRO-EA Not Receiving Emails from Addis HQ. DETAILS: Dear Sir or Madam, SRO-EA has not received any emails from Addis HQ since this morning. Kindly check if the email server is working properly. Thanks, Denise Ms. Denise Guce| Associate Administrative and Finance Officer| Sub-Regional Office for Eastern Africa (SRO-EA) of the United Nations Economic Commission for Africa (UNECA)|P.O. Box 4654| Kigali, Rwanda| Extension: 21612| Phone: +252 58 65 45/48/49 (Inside Rwanda) or +250 788 389 696 (Dialing from Outside Rwanda)|Email: dguce@uneca.org"