Network Management for Community Networks

Submitted in fulfilment
of the requirements for the degree of
Master of Science
of Rhodes University

Daniel David Wells

December 2009

Abstract

Community networks (in South Africa and Africa) are often serviced by limited bandwidth network backhauls. Relative to the basic needs of the community, this is an expensive ongoing concern. In many cases the Internet connection is shared among multiple sites. Community networks may also have a lack of technical personnel to maintain a network of this nature. Hence, there is a demand for a system which will monitor and manage bandwidth use, as well as network use.

The proposed solution for community networks and the focus within this dissertation, is a system of two parts. A Community Access Point (CAP) is located at each site within the community network. This provides the hosts and servers at that site with access to services on the community network and the Internet, it is the site's router. The CAP provides a web based interface (CAPgui) which allows configuration of the device and viewing of simple monitoring statistics. The Access Concentrator (AC) is the default router for the CAPs and the gateway to the Internet. It provides authenticated and encrypted communication between the network sites. The AC performs several monitoring functions, both for the individual sites and for the upstream Internet connection. The AC provides a means for centrally managing and effectively allocating Internet bandwidth by using the web based interface (ACgui). Bandwidth use can be allocated per user, per host and per site. The system is maintainable, extendable and customisable for different network architectures.

The system was deployed successfully to two community networks. The Centre of Excellence (CoE) testbed network is a peri-urban network deployment whereas the Siyakhula Living Lab (SLL) network is a rural deployment. The results gathered conclude that the project was successful as the deployed system is more robust and more manageable than the previous systems.

Acknowledgments

I would like to express an immeasurable amount of gratitude to my supervisors, Barry Irwin and Ingrid Siebörger, for their endless support throughout this project.

The Department of Computer Science at Rhodes University has the utmost gratitude from me, for supporting me throughout this process.

I am grateful for the proof reading and unconditional support received from my fantastic family; thank you Kate, Bob and Tom.

Telkom provided me with a Centre of Excellence bursary to read toward this degree without which this research would have not have been completed. I would also like to thank the Centre of Excellence in Distributed Multimedia in the Department of Computer Science and its sponsors (Telkom, Amatole, Business Connexion, Comverse, Mars Technologies, Stortech, Tellabs and THRIP) for equipment and funding.

For Michelle, Rebecca and Roger
Who were charged with the guardianship of my sanity

Contents

Abstract ii

Acknowledgments iii

1 Introduction 1

1.1 Problem Statement 3

1.2 Research Goals and Intended Outcomes 4

1.3 Document Structure 5

2 Related Work 7

2.1 Introduction 7

2.2 Internet and bandwidth in South Africa 8

2.3 Managing Internet Access 9

2.4 Tools for Monitoring and Troubleshooting Networks 11

2.4.1 ping 11

2.4.2 traceroute 13

2.4.3 cURL 14

2.4.4 Simple Network Management Protocol (SNMP) 15

2.4.5 NetFlow 17

2.4.6 Cacti 18

2.4.7 Zenoss Core 21

2.4.8 Nagios 23

2.4.9 Section Summary 24

2.5 Network Address Translation (NAT) 25

2.6 Point-to-Point Protocol over Ethernet (PPPoE) 26

2.7 DNS and BIND 26

2.8 Squid Caching Proxy Server 29

2.9 LightSquid 31

2.10 Firewalls 32

2.10.1 IPFW and PF 33

2.10.2 AltQ 34
2.10.3 Dummynet 35
2.11 Research Networks 35
2.11.1 CoE Testbed Network 35
2.11.2 Siyakhula Living Lab (SLL) 38
2.12 Similar Projects 41
2.12.1 Network Management Station 41
2.12.2 Network Discovery (NeDi) 42
2.12.3 m0n0wall and pfSense 43
2.13 Summary 45

3 System Design 47
3.1 Introduction 47
3.2 Conceptual Design and Feature Specification 47
3.3 Community Access Point (CAP) 50
3.4 CAP Graphical User Interface Design 52
3.5 Access Concentrator (AC) 53
3.6 AC Graphical User Interface Design 55
3.7 Connection Medium and Internet Connection 57
3.8 Summary 57

4 Implementation 59
4.1 Introduction 59
4.2 Implementation Overview 59
4.3 Operating System - FreeBSD 63
4.4 Community Access Point (CAP) 64
4.4.1 CAP Network and Application Configuration 64
4.4.2 CAP Graphical User Interface (CAPgui) 67
4.5 Access Concentrator (AC) 71
4.5.1 Overview of the Access Concentrator 72
4.5.2 Router/Gateway Networking Configuration 73
4.5.3 Applications and Configuration 75
4.5.3.1 Squid Caching Proxy Server 75
4.5.3.2 LightSquid 77
4.5.3.3 Cacti 78
4.5.4 Bandwidth Management 79
4.5.5 Access Concentrator Graphical User Interface (ACgui) 80
4.6 Virtual Development Environment 87

4.7 Summary 89

5 Deployment and Results **91**

5.1 Introduction 91

5.2 Hardware and Installation 91

5.3 Deployment Strategy 92

5.4 Deployment in CoE Testbed Network 93

5.5 Results - CoE Testbed Network 95

5.5.1 Bandwidth Use 95

5.5.2 System Use Graphs 96

5.6 Deployment in SLL Network 98

5.7 Results - SLL Network 102

5.7.1 Squid Use 103

5.7.2 Bandwidth Use 107

5.7.3 System Use Graphs 109

5.7.4 Bandwidth Use Restriction Imposed 110

5.8 Reflections on SLL 111

5.9 Summary 112

6 Conclusion **114**

6.1 Summary 114

6.2 Research Goals Revisited 115

6.3 Future Extensions 116

6.4 Conclusion 117

Reference List **118**

Appendices **126**

A SNMP - Useful OIDs **126**

A.1 General MIBs 126

A.2 Squid Proxy Server OIDs/MIBs 126

B Configuration Files **128**

B.1 Generating Configuration Files 128

B.2 Squid 128

B.2.1 RADIUS Authentication Helper 129

B.2.2 Delay Pools 129

B.2.3 Blacklists 130

C Electronic Appendix **131**

Glossary of Terms **132**

List of Figures

2.1 Example of ping Usage 12
2.2 Example of ping with IP record route option 12
2.3 Example of traceroute usage 14
2.4 Example of cURL usage 15
2.5 Example of Cacti - Screenshot 20
2.6 Example of Zenoss Core - CoE Testbed 22
2.7 Domain Name System (DNS) Tree Structure 27
2.8 Example of LightSquid 32
2.9 CoE Testbed Network - Location of Sites [36] 36
2.10 CoE Testbed - SCW - Logical Network Structure 37
2.11 SLL Network - Location of Sites [36] 39
2.12 SLL - Logical Network Structure 40
2.13 m0n0wall - User Interface 43
2.14 pfSense - User Interface 44

3.1 Generic Community Network 48
3.2 Conceptual System Design 49
3.3 Community Access Point (CAP) 50
3.4 Access Concentrator (AC) 53

4.1 Overview of System Developed 60
4.2 Community Access Point Implementation 64
4.3 CAPgui - System Page 68
4.4 CAPgui - Service Status Page 69
4.5 CAPgui - Disk Usage Status Page 69
4.6 CAPgui - Interfaces Status Page 70
4.7 Perl Regular Expression in PHP to Check a Valid Hostname 70
4.8 CAPgui - DHCPd Configuration 71
4.9 Access Concentrator Implementation 72
4.10 LightSquid - Overall Access Report 77

4.11 LightSquid - Per Month Internet Usage Graph 78
4.12 Cacti - HTTP Service Times Graph 79
4.13 Cacti - Load Averages Graph 79
4.14 ACgui - System Page 81
4.15 ACgui - CAP/CLAN Status Page 81
4.16 ACgui - CAP Services Status Page 82
4.17 ACgui - Squid Status Page 83
4.18 ACgui - Site Quota Status Page 83
4.19 ACgui - Host Quota Status Page 84
4.20 ACgui - User Quota Status Page 85
4.21 ACgui - CAPs Configuration Page 85
4.22 ACgui - Squid Configuration Page - Delay Pools 86
4.23 ACgui - Quota Configuration Page 87
4.24 Virtual Development Network 88

5.1 CoE Testbed - Logical Network Layout - SCW Deployment 94
5.2 SCW - Benjamin Mahlasela - Traffic(tun0) - Nov 2009 97
5.3 SCW - Dan Wells - Traffic(tun0) - Nov 2009 97
5.4 SCW - Access Concentrator - High CPU Use 98
5.5 Overview of SLL Network 99
5.6 SLL - Logical Network Layout - Deployment 100
5.7 SLL - Squid - HTTP Requests - Nov 2009 105
5.8 SLL - Bandwidth Use - Nov 2009 105
5.9 SLL - Squid - HTTP Service Times - Nov 2009 106
5.10 SLL - AC - CPU Use - Nov 2009 109
5.11 SLL - Ngwane - Load Average - Nov 2009 110
5.12 SLL - Mthokwane - Ping Latency - Nov 2009 110

B.1 Squid User Authentication with FreeRADIUS (squid.conf) 129
B.2 Squid Delay Pools (squid.conf) 130
B.3 Squid Blacklists (squid.conf) 130

List of Tables

2.1 SNMP OIDs Mapped to MIBs [70] 16
2.2 Private IP Address Space [78] 25

5.1 SCW - Deployed Sites Information and Requirements 94
5.2 SCW - Top 15 Websites - Sorted By Connects - Nov 2009 96
5.3 SLL - Deployed Sites Information and Requirements 101
5.4 SLL - Squid Proxy - Nov 2009 104
5.5 SLL - Bandwidth Use per Site - Nov 2009 106
5.6 SLL - Top 15 Websites - Sorted By Bytes Transferred - Nov 2009 108
5.7 SLL - Top 15 Websites - Sorted By Connects - Nov 2009 108

A.1 System MIBs 126
A.2 Network MIBs 127
A.3 Host Resources MIBs 127
A.4 Squid Proxy Server MIBs 127

Chapter 1

Introduction

The Internet is a global system of interconnected computer networks that interchange data by packet switching using the TCP/IP protocols. It is a *network of networks* that consists of millions of private and public, academic, business and government networks, local and global in scope, that are linked by copper wire, fiber optic cable, wireless technologies and various other technologies.

Before the invention of the Internet browser the World Wide Web (WWW) existed as a text only based communication medium. The advent of the Internet browser led to the general public being able to access content on the Internet with increased ease. Over the last fifteen years, the Internet has become a ubiquitous communication medium and is employed as a powerful tool for researchers, for personal benefits and for business gains.

South Africans' Internet use has progressed from the days when there were 300 bits per second (bps) dial up modems and users were connecting to local Bulletin Board Systems (BBSs) for online chat and simple file sharing. South Africans can now obtain a variety of broadband Internet connections. Over the last nine years the number of Internet users in South Africa has almost doubled. According to Internet World Stats [47], with data from 2009, South Africa currently has a penetration rate of 9.4% of the country's population using the Internet, this accounts for almost 4.6 million people out of a total population of almost 49.3 million [101]. Between 2000 and 2009, the number of Internet users had increased by 91.3%, from 2.4 million to 4.6 million users. These 4.6 million users in South Africa account for 6.8% of the Internet users in Africa [47].

There are a variety of broadband Internet connection options present, however, the majority of these options can be costly or are not available in all areas of South Africa. Rural areas in South Africa are affected by limited telecommunications infrastructure and these areas are the focus of the study.

Internet usage in South Africa remains expensive when compared to costs in developed countries [12, 95]. In a privately conducted survey, South Africa's telecommunication prices were compared with developed countries and with an international peer group [12, 95]. Its peers were chosen by those countries similar to South Africa (in terms of geographical dispersion of population, income dispersion and market structure) and those considered to have advanced telecommunication structures. An appropriate peer group was identified, by constructing a telecommunications competitiveness index. The countries in the survey included Canada, Hong Kong (China), Sweden, United States and other developed countries as international telecommunications best practice. The survey also included Brazil, India, Morocco, Thailand and other developing countries as peer group telecommunications best practice [12, 95].

The surveys conducted in 2005 [95] and 2007 [12] revealed some relevant Internet price differences between South Africa and 14 other countries. For an international leased line, South African prices had reduced by 44% between 2005 and 2007, however, the price remained 404.7% higher than the surveyed average price. A national leased line in South Africa had reduced in average price compared with the 15 countries, from 102% in 2005 to 25.5% higher in 2007. Business broadband (1 mbps ADSL) was the third most expensive service in its class of the 15 surveyed countries, and it was 127.2% more expensive than the average price. This had improved from 148% more expensive in 2005. Retail broadband (512 kbps ADSL) was the most expensive service in its class when compared to its peer group and was 130.5% more expensive than the average price. This had improved from 139% more expensive than the average in 2005 [12,95]. In July 2009, the Seacom undersea fibre optic cable came online promising 10 times more bandwidth to South Africans [84].

The demand for broadband Internet at the office and at home is increasing. Between 2003 and 2008 in Small to Medium Enterprises (SMEs) ADSL is becoming the primary form of Internet connectivity, with dial up, ISDN and leased lines decreasing (satellite usage remains small and wireless is slowly increasing with more options becoming available) [91]. Although the supply of bandwidth is improving year by year, there still exists a great demand for bandwidth to be managed due to the expense incurred in obtaining a connection.

Connection options which are made available to the South African public are limited by their speed (throughput) and are also generally limited by their bandwidth capacity (downloaded content or quota). For example, one may be able to purchase an ADSL line with a maximum download throughput rate of 512 kbps, a 1 GB maximum bandwidth capacity for international traffic per month and a 3 GB maximum bandwidth capacity for local (South African) traffic per month. In 2007, this form of Internet connection in South Africa would have been 130.5% more expensive when compared to its international peers [12]. As this is a large expense it results in South Africans requiring methods to control what the connection is used for.

Companies typically find that after purchasing an Internet connection that bandwidth intensive sites or applications (for example file sharing and social networking web sites) can saturate the shared Internet link. The solution most often implemented is to upgrade the connection at substantial cost. In many cases this scenario can be averted by effectively managing the connection [105].

Community networks can also suffer from this situation, where a limited capacity Internet connection is utilised for bandwidth intensive Internet content. A community network is a type of network which connects various geographical sites to share services and allow collaboration. A community network can allow the sharing of a single Internet connection amongst multiple sites. In this case a community network would rarely be able to cover the cost of upgrading the connection. Community networks could instead limit throughput to bandwidth intensive content to allow a higher use of informative, educational or knowledge based content.

Due to the many restrictions in place in South Africa, the Internet is a scarce, finite and expensive resource and this resource needs to be managed and monitored to provide a fair service to all that are using it.

1.1 Problem Statement

In 2008, Africa accounted for 14.3% of the world population but only accounted for 3.5% of the worlds total Internet users [47]. However, between 2000 to 2008 the growth rate of Internet users in Africa had exceeded 1000%. It is interesting to note that the rest of the world increased by only 296%. As of 2008 South Africa had the fourth highest number of Internet users on the continent with 4.6 million. Nigeria had the most Internet users on the African continent with 10 million [47], closely followed by Egypt and Morocco.

The years between 2000 to 2008 saw Internet usage in Africa increase impressively. However, in 2008 the price for Internet connectivity still remained high when compared with developed countries. Africa, specifically Sub-Saharan Africa, has among the highest connectivity costs in the world. African universities and schools spend almost 50 times more for their Internet connection than a United States of America equivalent [50]. This high price leads to a digital isolation of African students from the rest of the world.

Community networks (in South Africa and Africa) are often serviced by constrained capacity network backhauls. In relative terms this is an expensive ongoing part of their operation, relative to the basic needs of the community in question (water, electricity, food and shelter). In cases where this backhaul is financed or donated by other parties it is still a bounded resource. It needs to be managed, monitored and allocated equitably between users of the Internet connection.

The focus of the monitoring and management should be on bandwidth and throughput, but also allow for monitoring of network quality. Network quality can be understood as the degree of excellence that the user experiences when making use of the network and its available services. Monitoring will reveal which services are used most often and what those services require in terms of bandwidth and throughput to make the users experience more valuable. Monitoring will also produce statistics which will guide the administrator in how to best apportion the available bandwidth and throughput between services and users, and at what times certain services should get preference over others.

A problem with current network deployments of this type (community networks) is ensuring that there is an equitable distribution of the scarce resource between sites. A simple division of the resources (such as each site only having $1/n$th of the total throughput as a maximum rate and $1/n$th of the total bandwidth) may seem appropriate, but could lead to significant under-performance and under-utilization of resources. This division leads to an inefficient division of bandwidth and throughput, especially when multiple parties are not utilising the shared link. Division needs to be fair for each site as each site may have different requirements of the resources.

Many network tools and open source applications exist to help in monitoring and managing networks but the problem is that many of these tools and applications need to be integrated to form a final solution. By combining and interlinking these already existing applications this project hoped to produce a system that allowed network managers to equitably manage and control access to a scarce resource like bandwidth and the associated throughput for community networks in South Africa and other developing countries.

1.2 Research Goals and Intended Outcomes

The four primary research goals for this project are as follows:

- This project investigates network management and monitoring, to collate applications and tools which can aid a network administrator. Many applications and tools exist which are designed for a specific purpose. Administrators need to choose which of these meet the requirements for their network and implement them separately.

- By researching numerous network applications and tools, this project attempts to integrate these into a single and easily deployable system. The system integrates multiple applications and tools in a client-server architecture to manage and monitor a network, whilst managing bandwidth and users more effectively. This system is designed in such a way to allow easy implementation into an existing network and have a Graphical User Interface (GUI) to abstract the management of the system from the underlying Operating System (OS).

- The system must allow for bandwidth and throughput management and monitoring of a network with a single Internet connection. It must be simple to configure, to deploy and to customise.

- Deployment of the system to two community networks. Success of the deployments will be measured in terms of the reliability and usability of the system.

Two secondary goals have been identified:

- The system should be sufficiently lightweight to run on low-cost Personal Computers (PCs) with minimal performance requirements.

- Documentation of the system components will be made available to explain how the components fit together, allowing for further modifications or for easy replication. The documentation should be understandable by an administrator with limited networking knowledge.

1.3 Document Structure

This document is presented in three broad sections, divided over a number of chapters. The first section (Chapters 1 and 2) defines and describes the problem area that this project seeks to address, highlighting literature relevant to the project's goals. This is followed (Chapters 3 to 5) by a detailed description of the system employed to manage and monitor bandwidth, first presenting the system design then the implementation phase, and finally deployment and results. The last section (Chapter 6) is a review of the work covered and draws conclusions regarding this project.

The remainder of this section presents an overview of each chapter:

- Chapter 2 examines current Internet usage in South Africa and why a need exists to manage and monitor bandwidth use to provide effective and efficient use of the scarce resource. Network applications, tools and hardware are discussed along with their benefits and shortfalls in terms of achieving the goals of this project. This chapter describes two community research networks which were used as test sites for this project. Systems which provide similar outcomes to this project are investigated, with their advantages and disadvantages.

- Chapter 3 describes the design of the system that was implemented to manage and monitor bandwidth use at the two community networks, along with how it fits into an existing network. This chapter gives insight into what type of applications and tools are needed to implement a workable package. This is a high level view of the system and how it was constructed, in terms of concepts used and not the actual technology used.

- Chapter 4 discusses the implementation of the system design. Using the concepts discussed in Chapter 3 applications and tools are chosen to meet the goals of the design requirements. This chapter presents some of the more technical aspects of this project, demonstrating how the applications and tools are integrated to produce a working solution to meet the design and project goals.
- Chapter 5 discusses the deployment of the proposed solution to the two community networks. It explains how the deployment was conducted. This chapter also presents the results of the deployed system and shows how it was used to manage and monitor the two research networks.
- Chapter 6 discusses a number of conclusions that can be reached when considering the results from Chapter 5. Goals stated in Chapter 1 are revisited and conclusions are drawn on them. This chapter summarises and concludes this dissertation and lists future research that could be conducted to extend and further this work.

Chapter 2

Related Work

2.1 Introduction

The Internet has influenced the way that the population of the world stays in touch. Many devices today are *Internet Enabled*, enabling millions of users world wide to get connected. Devices capable of Internet connectivity include computers, laptops, netbooks, mobile phones, smart phones and even WiFi enabled SD cards[1]. Internet use in South Africa is increasing, from 2.4 million Internet users in 2000 to 4.6 million users in 2008. However, South Africa's level of Internet connectivity is not nearly as ubiquitous as it is in developed countries.

The chapter begins with a discussion on the limited availability of Internet bandwidth in South Africa in Section 2.2. In a scenario with a limited bandwidth Internet connection, the resource needs to be adequately managed and this is described in Section 2.3. A number of monitoring and troubleshooting tools are available for use within a network and some pertinent tools and applications are discussed in Section 2.4. Monitoring provides invaluable network usage information to an administrator, however, monitoring itself does not lead to effective management of a network, thus we also investigate a number of tools and applications for bandwidth management.

A widely used solution for a private network with a limited number of real world IP addresses is Network Address Translation (NAT). Reasons for utilising NAT are described in Section 2.5. To facilitate secure tunnels over a shared connection medium is Point-to-Point Protocol over Ethernet (PPPoE) can be implemented, it is discussed further in Section 2.6.

A brief introduction to the Domain Naming System (DNS) and a popular application to implement it, BIND, is presented in Section 2.7. Squid Caching Proxy Server, an

[1]Eye-Fi - http://www.eye.fi/

application to control and reduce the amount of bandwidth used in accessing the World Wide Web (WWW), is discussed in Section 2.8. Section 2.9 presents a web based application called LightSquid, which summarises the Squid access log files. LightSquid allows effectively monitoring the amount and type of content accessed from the Internet via the proxy server. A necessary component for a network of any size is a firewall and two implementations are described in Section 2.10 with their methods for limiting Internet usage.

Two community research networks have been identified which are in need of a system to manage and monitor bandwidth use. The two networks, their layouts and their associated problems are discussed in Section 2.11. Four applications, which produce similar desired results to this research project have been identified in Section 2.12. The similar projects' advantages and disadvantages are also presented, however, neither were found to meet the requirements of the research networks.

2.2 Internet and bandwidth in South Africa

In South Africa, there are many different types of Internet connectivity options available, however, the majority of these options are in urban areas. Internet connectivity options are limited in rural areas due to the scarcity of telecommunications infrastructure. In South Africa, and indeed in most of Africa, using or having an Internet connection is considered a luxury, and it is not as widely used as it is in developed countries. Even though there are a number of Internet connection options, the majority of South Africans cannot afford it due to high prices.

Statistics South Africa[2] conducted a study of South African households and their Income and Expenditure, and the results were made available in 2008 [100]. The study took a sample of over 21 thousand households from a population of over 12 million households in South Africa. Approximately 11 thousand households were in urban areas and 9 thousand were situated in rural areas. The study revealed that 14.7% of all South African households had a computer, however, only 6.5% of all households had some form of Internet service [100]. Only 22% of South African households had a landline telephone, although 70.4% had a cellular telephone. 30.9% of landline telephones were situated in urban households compared to only 5.4% of rural households. The majority of the households which had a computer and an Internet service were in the upper income groups [100].

The high cost of an Internet connection is hampering the uptake of Internet based services in South Africa. This is limiting South Africans from using and creating new

[2]Statistics South Africa: http://www.statssa.gov.za/

web-based technologies. Internet services created for South African users need to take into account that a large majority of South Africans have a very limited Internet connection in terms of bandwidth and throughput [68]. The high cost of bandwidth in South Africa poses a barrier to the country's contribution to international trade and knowledge, also limiting the growth of local markets and education in the country [86].

Seacom has provided a fibre optic cable along the East Coast of Africa to Southern Africa, Europe and Asia [84]. The Seacom cable has a capacity of 1.28 TB/s, which will enable a higher utilisation of services; such as high definition television; IPTV; peer to peer networking and supply the growing demand of Internet bandwidth in Africa. The undersea cable came on-line in South Africa on 23 July 2009 [84]. Before this, South Africa was connected to Europe and Spain through the SAT-3/SAFE (South Atlantic 3/West Africa Submarine Cable) fibre optic cable along the West Coast of Africa. Capacity of the SAT-3/SAFE cable was insufficient to offer fast broadband, however, Seacom is proposed to provide 10 times more bandwidth [57]. The introduction of the Seacom cable is expected to reduce the price of Internet bandwidth [51] and provide new opportunities [1], however, telecommunication infrastructure in many rural areas within South Africa are still being developed and deployed. Therefore, the majority of benefits from the increased bandwidth supply will only be seen in urban areas. Telecommunications in South Africa is improving, however, it is not at a state similar to developed countries.

With limited availability of bandwidth and infrastructure, systems need to be put in place to manage how Internet services are being used. When an Internet connection is used by a single user, they can control and limit their own usage. A shared Internet connection needs specific measures in place to control its use and allow fair distribution among all the users.

2.3 Managing Internet Access

The Internet encapsulates a wealth of information and data which is constantly being changed and updated, and search engines allow for easy access to a wide variety of multimedia. Information that can be found on the Internet can be legitimate and well researched information with reliable sources, or it can simply be a single individual's opinion on a specific subject matter. As information on any subject matter can be easily accessed and taking into account the location of the people using the Internet connection, controls may need to be implemented to prevent or limit access to certain subject matter [81].

For an example, in the scenario of a school computer laboratory, access to certain websites and services could be limited (or denied) depending on their content. As schools are locations in which education takes place, the Internet should generally be used for educational purposes and not primarily recreational purposes. Instead of allowing the students access to only certain sites, it is preferred to limit or deny access to a list of known websites, otherwise known as a blacklist. The blacklists can either be created manually or ready made lists are freely available [85]. It is possible to obtain blacklists in distinct content categories such as illegal, advertising, downloads, gambling, pornographic and violent content [85]. A blacklists could be constructed based on the requirements of the location at which the Internet is accessed. Another popular method of limiting access is to prevent accessing websites which contain one or more predefined keywords [81]. However, keywords, or combinations of keywords can be a rather poor representation of the meaning of texts and therefore may block otherwise acceptable sites [81].

Many websites exist that are bandwidth intensive, for example, the popular video sharing website YouTube and social networking websites Facebook and MySpace [34,61]. In an educational setting, the aforementioned websites are generally considered to be used as recreational websites, however, they can also have educational merit in specific circumstances. As these types of websites are bandwidth intensive, they may be required to be limited or blocked, unless a user has a specific educational reason to have access to them. Without controls on bandwidth intensive websites, you run the risk of a "tragedy of the commons" scenario where a few ruin the experience for the rest by wasting the shared resource [40]. However, this may not be the intentional outcome by the few that are affecting the experience for others.

When bandwidth is a finite resource, it may need to be equitably distributed amongst users to ensure fairness of use. In a limited bandwidth scenario, Internet usage could be logged and a per user quota could be used to limit user access to fair levels. User quotas can be a predefined amount (bandwidth), which could differ per location. When a user reaches their maximum allocated bandwidth, their Internet access can either be limited in speed (throughput) or their access can be denied. Internet usage could be limited by time online or amount transferred in a time period, for example bandwidth used per x days, per week or per month.

Two common methods that are used to limit throughput on a (Internet) connection are policing and shaping of network traffic [16]. Traffic policing is implemented by dropping packets over the maximum throughput rate. Policing network traffic tends to propagate bursts of traffic flows. The result is a throughput rate that, when graphed, appears as a saw-tooth with crests and troughs [16]. Traffic shaping retains the excess packets, which would exceed the maximum throughput rate, in a queue and schedules them for later transmission over a period of time. The result of traffic shaping is a smoothed throughput rate.

To aid in allocating bandwidth fairly and efficiently, monitoring of the network needs to be conducted to investigate how the network is being used. The following section discusses some applications and tools that are available to assist in management and monitoring networks and bandwidth.

2.4 Tools for Monitoring and Troubleshooting Networks

The applications and tools discussed in this section aid administrators in determining the health of an IP based network, providing troubleshooting information and assessing the utilisation of a network. Utilisation of the network includes network traffic within the Local Area Network (LAN) or utilising services outside the LAN (on the Internet). By understanding how the network is being used, it is then possible to implement better controls and limits on network services to ensure an equitable distribution of network resources between the users. A number of these components have been integrated into the project system.

This section discusses some simple tools and network protocols before it proceeds onto more advanced tools, graphing and monitoring applications which can be customised to suit specific network requirements. This section is separated into nine subsections. The first three sections discuss ping, traceroute and cURL. These are basic network troubleshooting and information gathering tools. Section 2.4.4 is a detailed description of SNMP, and NetFlow is discussed in Section 2.4.5. The last three sections present a graphing solution, Cacti, and two monitoring applications, Zenoss Core and Nagios.

2.4.1 ping

The ping [67] program is a common tool included with most modern Operating Systems (OSs). The ping program tests basic connectivity between two systems running TCP/IP and using the ICMP protocol [104]. ICMP is often considered part of the Network Layer (layer three) protocol in the Open Systems Interconnect (OSI) network stack [44], however, ICMP packets are encapsulated within IP datagrams [104]. Ping sends Internet Control Message Protocol (ICMP) echo requests to a host and receives ICMP echo responses [76].

This simple action allows the Round Trip Time (RTT) to be measured which quantifies the latency of the network connection [48]. Ping also measures how many packets were lost during the test; packet loss provides a rough measure of the quality of the network connection between the two hosts. Latency is the delay from requesting data and receiving a response, or in the case of one-way communication from the moment of transmission until the moment the data is received. Different communication technologies have different latencies. By measuring packet loss and RTT, an administrator can determine the quality of the network connection being tested. Ping can be used as a starting point for network

```
dan@g03w0418:~$ ping -c 4 catalyst.scw.ru.ac.za
PING catalyst.scw.ru.ac.za (146.231.118.131) 56(84) bytes of data.
64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=1 ttl=124 time=42.1 ms
64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=2 ttl=124 time=46.6 ms
64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=3 ttl=124 time=48.2 ms
64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=4 ttl=124 time=49.1 ms

--- catalyst.scw.ru.ac.za ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3013ms
rtt min/avg/max/mdev = 42.110/46.555/49.189/2.721 ms
```

Figure 2.1: Example of ping Usage

```
dan@g03w0418:~$ ping -R -c 4 catalyst.scw.ru.ac.za
PING catalyst.scw.ru.ac.za (146.231.118.131) 56(124) bytes of data.
64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=1 ttl=124 time=50.2 ms
NOP
RR:     g03w0418-5.ict.ru.ac.za (146.231.123.9)
        dukat.dsl.ru.ac.za (146.231.113.37)
        dan.gw.ru.ac.za (146.231.118.129)
        catalyst.scw.ru.ac.za (146.231.118.131)
        dan-gw.scw.ru.ac.za (146.231.117.154)
        dukat.dsl.ru.ac.za (146.231.113.37)
        g03w0418-5.ict.ru.ac.za (146.231.123.9)

64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=2 ttl=124 time=46.1 ms
NOP     (same route)
64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=3 ttl=124 time=52.4 ms
NOP     (same route)
64 bytes from catalyst.scw.ru.ac.za (146.231.118.131): icmp_seq=4 ttl=124 time=50.5 ms
NOP     (same route)

--- catalyst.scw.ru.ac.za ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 46.170/49.834/52.434/2.279 ms
```

Figure 2.2: Example of ping with IP record route option

troubleshooting. By using ping, a network user can test whether a specific host on the network is online, if the host is online it will respond to the ICMP echo request (unless the host is configured to not respond to ICMP echo requests or ICMP is blocked by a firewall, which are the default settings on Microsoft Windows XP and newer OSs). By testing a known host, it is possible to test if a network link (or segment) is up.

An example of ping in use can be seen in Figure 2.1. In the example, four ICMP Echo Requests are sent to the host *catalyst.scw.ru.ac.za*. Each line displays the ICMP Echo Responses from the host; it displays how many bytes were received, which ICMP packet it was in the sequence, the packets Time to Live (TTL) and the time in took from sending the request to receiving the response. In this example, of the four ICMP packets that were transmitted four ICMP packets were received. The summary shows that there was a 0% packet loss and the total test took 3013 ms. The last line in the example shows minimum, average, maximum and mean deviation of the RTT [67].

The ping program provides an option to record the network route it took to reach and return from the specified host. The IP record route option (most versions enabled by attaching the -*R* flag) asks each router which handles the ICMP datagram to add its IP address to a list [104]. When ping receives the ICMP echo reply it prints the list of IP addresses. See Figure 2.2 for an example of ping with the IP record route option. However, this feature of ping does not always provide suitable outputs; there is limited space in the ICMP datagram; and some routers might not check the ICMP datagram for this option. Rather, for tracing the route to a specific host on the network, traceroute can be used.

2.4.2 traceroute

The traceroute [49] program is a network tool used to determine the route taken by packets across a TCP/IP network. Although it is possible that two consecutive IP datagrams from the same source to the same destination could follow different routes, it is unlikely. Ping with the *record route* option set (see Figure 2.2) can gather this information, but it relies on the routers supporting the option. Traceroute does not require any special or optional features at any routers along the path.

Traceroute relies on IP and manipulates the TTL field in the IP packet header. The Microsoft Windows implementation of traceroute (*tracert*) [64] uses ICMP packets. FreeBSD's *traceroute* [49] uses UDP packets by default, but can use TCP, GRE and ICMP packets. Each router that handles the IP datagram is required to decrement the TTL by either one or the number of seconds the router held the datagram. Most routers will hold a datagram for less than a second, effectively making the TTL a hop counter, decremented by one by each router. The purpose of the TTL field is to prevent datagrams from traveling in infinite loops, which can occur during times that routers are down [104]. When a router receives a datagram whose TTL is either 0 or 1 it will not forward the datagram. Instead the router discards the datagram and sends back a ICMP "Time Exceeded" response to the originating host.

Traceroute, by default, sends three IP packets (triplets) with consecutively incremented TTLs on the IP datagrams. Traceroute sends an IP datagram with a TTL of 1 to the destination host, the first router decrements the TTL, discards the datagram and sends back a ICMP "Time Exceeded". Traceroute has identified the first router in the path. Traceroute then sends a datagram with a TTL of 2 to the destination host, and identifies the second router in the path.

A number of tracing methods are supported by traceroute, the default (in FreeBSD) is to use UDP datagrams on decrementing unlikely port numbers to return a ICMP "Port Unreachable" on the destination host. Various other methods supported by traceroute include using ICMP datagrams as the probes; constant destination port (default port 80, HTTP) and TCP datagrams, intended to bypass firewalls; and UDP datagrams with

```
dan@g03w0418:~$ traceroute catalyst.scw.ru.ac.za
traceroute to catalyst.scw.ru.ac.za (146.231.118.131), 30 hops max, 40 byte packets
 1  ict.gw.ru.ac.za (146.231.120.1)  0.451 ms  0.601 ms  0.738 ms
 2  rtu02-dsl1.dsl.ru.ac.za (146.231.113.234)  7.742 ms  8.846 ms  9.839 ms
 3  dukat.dsl.ru.ac.za (146.231.113.37)  11.453 ms  12.122 ms  13.446 ms
 4  dan-gw.scw.ru.ac.za (146.231.117.154)  43.352 ms  48.511 ms  55.140 ms
 5  catalyst.scw.ru.ac.za (146.231.118.131)  65.502 ms  68.553 ms  69.559 ms
```

Figure 2.3: Example of traceroute usage

constant destination port (default 53, DNS) [49].

Figure 2.3 shows an example of the traceroute program, tracing the network path to the host *catalyst.scw.ru.ac.za* . The first line of output displays the name and IP address of the destination host. Next, the output states that traceroute will attempt a maximum of 30 increments of the TTL and each datagram sent will be 40 bytes. Each successive line in the output shows the TTL followed by the name of the host or router and its associated IP address. For each TTL, three datagrams are sent and the RTT for each are calculated and printed. By sending three datagrams, an average RTT can be quantified. The traceroute halted at a TTL of 5 when it reached its destination host. Figure 2.3 shows a traceroute in its default form using UDP datagrams and an unlikely port number of the destination host. Firewalls may block connections on high port numbers, effectively stopping a traceroute short of its destination.

Traceroute allows administrators to confirm routing is configured correctly within the network. It provides information such as the RTT for each datagram to reach each router, providing an understanding about the latency of each hop. However, traceroute does not provide the throughput rate for transferring files between the source and destination hosts. The cURL program can be used to quantify the throughput between two hosts.

2.4.3 cURL

The cURL[3] program is a command line tool for transferring files with Uniform Resource Locator (URL) syntax [102]. It is available for download on a number of OSs. A large number of protocols are supported including FTP, HTTP, HTTPS and SCP [103]. The syntax of the URL is protocol dependent, more detailed information on URLs can be found in RFC 3986 [7].

Figure 2.4 shows an example of a transfer using cURL to download a file from an FTP server on the host *catalyst.scw.ru.ac.za*. The URL syntax shows the protocol used (*ftp://*), the user (*test*), the password (*password*), the host name (*catalyst.scw.ru.ac.za*) and the file required (*/test8000k.dat*). The file was not stored on the local machine but

[3]cURL - http://curl.haxx.se/

```
dan@g03w0418:~$ curl ftp://test:password@catalyst.scw.ru.ac.za/test8000k.dat > /dev/null
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 8023k  100 8023k    0     0  88064      0  0:01:33  0:01:33 --:--:-- 88874
```

Figure 2.4: Example of cURL usage

rather the received output was redirected to */dev/null*. The summary provides some basic information about the transfer, including the size of the file, average download speed and duration of the transfer. In this example, the download of a 8023 Kb file took 1 minute 33 seconds at an average throughput rate of 88 Kbps.

The example provided in Figure 2.4 shows cURL as a basic test for measuring throughput across a network or the Internet.

2.4.4 Simple Network Management Protocol (SNMP)

A widely used approach to network monitoring and management is the Simple Network Management Protocol (SNMP) [38,62,72]. This protocol was formulated in 1988 in RFC 1067 [14]. SNMP has undergone many changes since 1988 and is presently in version three (SNMPv3 [41]), however, the two prior versions (SNMPv1 [15] and SNMPv2c [77]) are still supported in most software and devices (as this is best current practice according to RFC 3584 [33]).

A number of applications are available providing a host with an SNMP agent and SNMP tools, three implementations (Net-SNMP [72], BSNMP[4] and Microsoft Windows SNMP Service [63]) are discussed later in this section.

Key network devices (such as servers, hosts, switches and routers) may be equipped with SNMP agent software so that they may be managed from a management station [62]. "The [SNMP] management agent responds to requests for information from a management station, responds to requests for actions from the management station, and may asynchronously provide the management station with important but unsolicited information" [98].

In SNMP, resources on the network are referred to as an object and each object is a data variable that represents one aspect of the managed system. An object is referred to using a numerical Object Identifier (OID) for examples see Table 2.1. In this table, OIDs are presented for the monitored systems description, system uptime and hostname. As OIDs are difficult for people to remember a method of assigning human readable names has been defined.

A collection of these objects (OIDs), in human readable form are referred to as the

[4]BSNMP - Mini SNMP daemon - http://people.freebsd.org/~harti/bsnmp/

Object Identifier (OID)	Management Information Base (MIB)
.1.3.6.1.2.1.1.1.0	.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0
.1.3.6.1.2.1.1.3.0	.iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.sysUpTimeInstance
.1.3.6.1.2.1.1.5.0	.iso.org.dod.internet.mgmt.mib-2.system.sysName.0

Table 2.1: SNMP OIDs Mapped to MIBs [70]

Management Information Base (MIB). Using MIBs and the examples given in Table 2.1, the OID numerical representation changes to more human readable forms. To a large extent, the information contained in the MIBs are standardised, however, some proprietary devices can set aside their own MIBs and these can be obtained by contacting the device manufacturer. A common use for SNMP is measuring how many bytes have been transferred on a network interface by using the MIBs for the InOctet and OutOctet counters [48].

Monitoring the SNMP managed system can be undertaken by retrieving the value of MIB objects. A management station can then cause an action to take place at an agent or can change the configuration settings of an agent by modifying the value of specific MIBs [62,98]. SNMPv1 and SNMPv2 both have limited security features, namely authentication and privacy [99]. The use of SNMPv3 is advised as it provides authentication, privacy and access control when managing the SNMP agents [99]. In addition to responding to requests, SNMP can notify the management station of an occurrence of a preconfigured event, these events are sent as a one way message to the management station and are called traps. SNMP is a useful network management protocol as it provides an administrator with the advantage of being able to manage all devices from a central point.

A number of SNMP agent and management software implementations are available, however, three have been identified for inclusion in this discussion, Net-SNMP, BSNMP and Windows SNMP Service. Net-SNMP [72] is a collection of SNMP applications used to implement SNMP v1/v2c/v3 using both IPv4 and IPv6. Net-SNMP is available for many Unix like OS's and also for Microsoft Windows. The installation of Net-SNMP comes prepackaged with a number of MIBs for a wide variety of devices. An SNMP agent is included for responding to queries for MIBs via the daemon *snmpd*, this includes many built in MIBs and can be extended using dynamically loadable modules. The SNMP agent *snmpd* binds to a specific port and awaits requests from SNMP management software; when it receives a request it processes it, collects the requested information or performs the requested operations and returns the information to the sender [71]. Net-SNMP also provides a number of useful SNMP based applications. The collection of Net-SNMP applications include *snmpget*, *snmpgetnext* and *snmpset* which retrieves or sets a single OID; *snmpwalk* retrieves a subtree of MIBs using multiple *snmpgetnext* requests; *snmpstatus* returns a number of basic statistics of the host such as OS, system uptime and summarises network interface statistics; and *snmptranslate* which converts numerical OIDs to textual forms.

A lightweight and simpler implementation of an SNMP agent is BSNMP [9]. BSNMP consists of a small number of tools and *bsnmpd* the SNMP agent daemon. BSNMP supports SNMP v1/v2c and only contains a small number of pre-package MIBs. Although a lightweight implementation of an SNMP agent, BSNMP's functionality is implemented via loadable modules and therefore can be extended to provide functions outside the scope of network management. BSNMP is only available for installation on FreeBSD, it is included in FreeBSDs source tree and is shipped with FreeBSD releases.

Bundled with Microsoft Windows family is the SNMP Service [63]. The SNMP service is not enabled by default, however, it is easy to install and configure via a graphical configuration page. The Microsoft Windows OS's that were investigated were Windows XP, Windows Vista and Windows 2003 Server [63]. Configuration options include selecting which services the SNMP service must monitor, such as physical disks, applications, network interfaces, network routing and end-to-end connections. It is also possible to configure SNMP communities, SNMP traps, and security.

An SNMP agent can only collect information on the machine on which it is installed. To provide more detail of the network utilisation by the hosts, a NetFlow sensor can be employed.

2.4.5 NetFlow

Operating a network without accurate traffic statistics is not desirable [94]. NetFlow can provide network traffic usage statistics. NetFlow provides more fine grained data than an SNMP agent but not as detailed or high volume as packet sniffers [94]. SNMP facilitates capacity planning, however, it does little to characterize traffic applications and patterns. SNMP provides packet and byte counters, which are useful, but understanding which IP addresses are the source and destination of traffic and which applications are generating the traffic is invaluable [17]. NetFlow is a widely used tool for traffic accounting and traffic analysis [75]. Using NetFlow to aggregate traffic information it is possible to understand how the network is being used and how bandwidth should be separated between sites.

NetFlow is an open protocol developed by Cisco Systems Inc. for collecting IP network traffic information. NetFlow can track a wide range of packet information at the Network level (level 3 of the OSI stack [44]), for IPv4 and IPv6 traffic flows. A flow is described as "active as long as observed packets that are meeting the flow specification are observed separated in time by less than a specified timeout value" [19]. NetFlow gathers information such as source and destination MAC addresses, IP address and ports; the number of packets and bytes transmitted; timestamps on the packets; and TCP flags [18].

NetFlow provides a session-level view of network traffic and records information about every TCP/IP transaction that occurs over the network [54]. NetFlow has three major components, a sensor, a collector and a reporting system. The sensor is a daemon which actively listens to network traffic on a specific network interface and captures the TCP/IP session data. The NetFlow sensor exports summarised flow data (NetFlow datagrams) to a collecting host by UDP on a specific port number. The collector is a daemon which listens on a UDP port for NetFlow datagrams and saves them for later evaluation. A reporting system compiles the NetFlow datagrams from the collector and produces human readable reports or graphs.

A range of NetFlow sensor implementations are available for use and three forms of implementation are discussed. These are sensors implemented in hardware, software based sensors and sensors dependent on another application. Cisco Systems Inc. provides a range of NetFlow enabled routers and switches which natively perform flow analysis and exporting at the hardware level. Softflowd [92] is a software implementation of a NetFlow sensor which relies on the *libpcap*[5] library to sniff packets as they pass through the monitored network interface. Softflowd has been developed for use on Linux distributions and BSD type OSs. A sensor application that is dependent on PF Firewall [39] is Pfflowd [73]. Pfflowd converts PF Firewall status messages to NetFlow datagrams which are transmitted to a collector. Utilising PF Firewall's stateful packet filtering infrastructure Pfflowd allows flow tracking that is fast and accurate [73].

SNMP and NetFlow monitoring can provide a substantial amount of data very quickly. Data can be gathered about the system health of servers, the state of network services, bytes or packets transferred in and out of network interfaces and flows of data throughout a network. Without effectively using or viewing this data, the resources needed to collect the data have been consumed unnecessarily. The following section discusses a graphing solution, Cacti, which uses data from a number of sources and represents the data over time via graph plots.

2.4.6 Cacti

Cacti[6] is a web based graphing solution, utilising the customisability of Round Robin Database tool (RRDtool) graphs [82]. Cacti provides quick polling of devices for system data, advanced graph templates, various methods of acquiring data from multiple sources (primarily utilising SNMP) and user management features. Cacti is capable of graphing data from a small number of hosts, or networks with hundreds of devices [107].

The design of RRDtool is such that it allows the collection of status information from

[5]tcpdump/libpcap - http://www.tcpdump.org/
[6]Cacti - http://www.cacti.net/

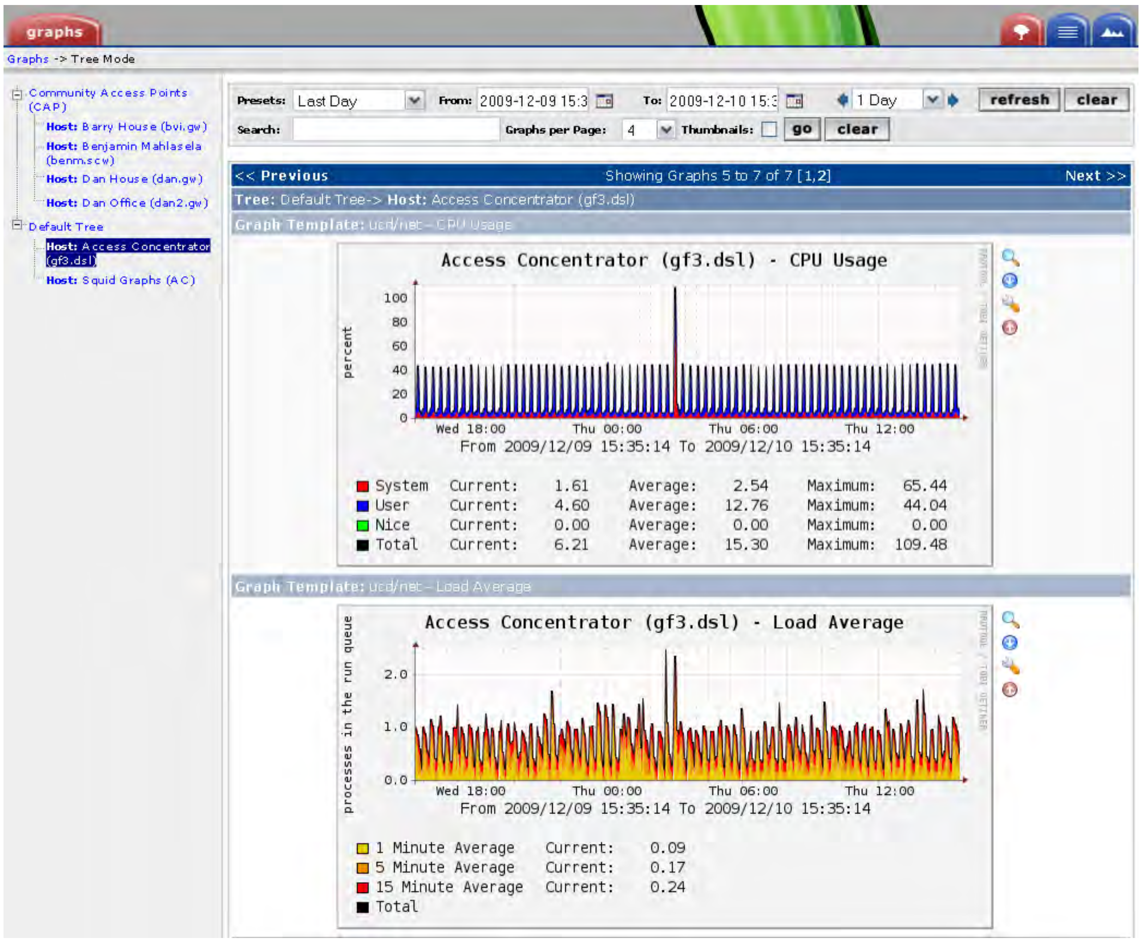
a range of data sources and generate graphical representations of the data collected over a definable time period [82]. Data acquisition is preferred at constant time intervals, however, RRDtool can interpolate missing values when new data is gathered from a source. Data consolidation is performed after specific time periods as storing gathered data for long periods can take up considerable amounts of disk space. RRDtool stores the collected data in Round Robin Archives (RRA) as it is an efficient method to store data for a certain amount of time while using a known and constant amount of disk space [82]. RRAs are pre-configured to store a set number of records and when this value is reached, the data is stored from the beginning of the RRA again. Multiple RRAs can be stored within an RRD, allowing different resolutions of the data for different time frames. For example, finer grained data can be stored for data over the period of a day, and coarser grained data can be stored for the period of a month. RRDtool allows the generation of reports in numerical and graphical forms based on the data stored in one or more RRDs [82]. Graphing using RRDtool is customisable, allowing changes of size, colour and the contents of the graph. Cacti utilises the aforementioned strengths of RRDtool and presents it in an easy to use and intuitive web based interface.

Cacti includes extensive user management features, allowing administrators to create users and assign different levels of permissions to the interface. Permissions can be specific per-graph for each user, allowing only specific users to view graphs related to specific devices.

A number of graphs can be created and viewed in Cacti. Limitations exist on the amount of disk space available to house the RRDs and time to poll the devices for data. RRD files can be populated with data from multiple data sources allowing for highly customisable graphs. Users are able to define custom scripts that can be used to gather data. Cacti has built in SNMP support for a number of SNMP implementations. A PHP-based poller is provided to execute scripts, retrieve SNMP data and update the RRD files. Cacti provides three types of templates to insert new devices and create new graphs easily. Graph templates allow for the easy construction of new graphs. Data source templates enable common data source types to be grouped together. Host templates are a group of graph and data source templates that allows Cacti to define common host types. By using host templates, data sources are populated and graphs are created to that specification.

Figure 2.5 shows a working installation of Cacti. Along the left hand side of Figure 2.5 a list of devices is displayed in a tree structure, upon selecting a device, graphs that are related to the device are shown in the main panel. This example shows the graphs for *CPU Usage* and *Load Average* for the device *Access Concentrator*.

Cacti provides an administrator with a number of graphs about specific variables and devices on a network. The next sections describe two monitoring applications, which ob-


graphs
Graphs -> Tree Mode
Community Access Points (CAP)
Host: Barry House (bvi.gw)
Host: Benjamin Mahlasela (benm.scw)
Host: Dan House (dan.gw)
Host: Dan Office (dan2.gw)
Default Tree
Host: Access Concentrator (gf3.dsl)
Host: Squid Graphs (AC)
refresh
clear
go
clear
<< Previous
Next >>
percent
processes in the run queue
Wed 18:00
Thu 00:00
Thu 06:00
Thu 12:00

Presets: Last Day From: 2009-12-09 15:3 To: 2009-12-10 15:3 1 Day

Search: Graphs per Page: 4 Thumbnails:

Showing Graphs 5 to 7 of 7 [1,2]

Tree: Default Tree-> Host: Access Concentrator (gf3.dsl)

Graph Template: ucd/net - CPU Usage

Access Concentrator (gf3.dsl) - CPU Usage

From 2009/12/09 15:35:14 To 2009/12/10 15:35:14

```
🟥 System   Current:   1.61   Average:   2.54   Maximum:   65.44
🟦 User     Current:   4.60   Average:  12.76   Maximum:   44.04
🟩 Nice     Current:   0.00   Average:   0.00   Maximum:    0.00
⬛ Total    Current:   6.21   Average:  15.30   Maximum:  109.48
```

Graph Template: ucd/net - Load Average

Access Concentrator (gf3.dsl) - Load Average

From 2009/12/09 15:35:14 To 2009/12/10 15:35:14

```
🟨 1 Minute Average    Current:   0.09
🟧 5 Minute Average    Current:   0.17
🟥 15 Minute Average   Current:   0.24
⬛ Total
```

Figure 2.5: Example of Cacti - Screenshot

tain a number of monitoring inputs and provide a succinct summary of all the devices on a network. The applications were researched to understand how monitoring could be performed and which variables should be monitored in a network. The method for displaying status information on the user interfaces was particularly relevant to this project.

2.4.7 Zenoss Core

Zenoss Inc.[7] provide three software packages which can aid the administrator of a network of any size. The software packages available are Zenoss Core, Zenoss Professional and Zenoss Enterprise. The Zenoss Core package is licensed as open source, whereas Professional and Enterprise require a commercial subscription license. The focus of this discussion is on Zenoss Core as it is a free package.

Zenoss Core [121] is a network monitoring package that monitors the configuration, health and performance of networks. Network devices, servers and applications are monitored through a single, integrated software package. Zenoss Core is available for installation on a number of OS's including various Linux distributions, FreeBSD, Solaris 10 and Mac OS X.

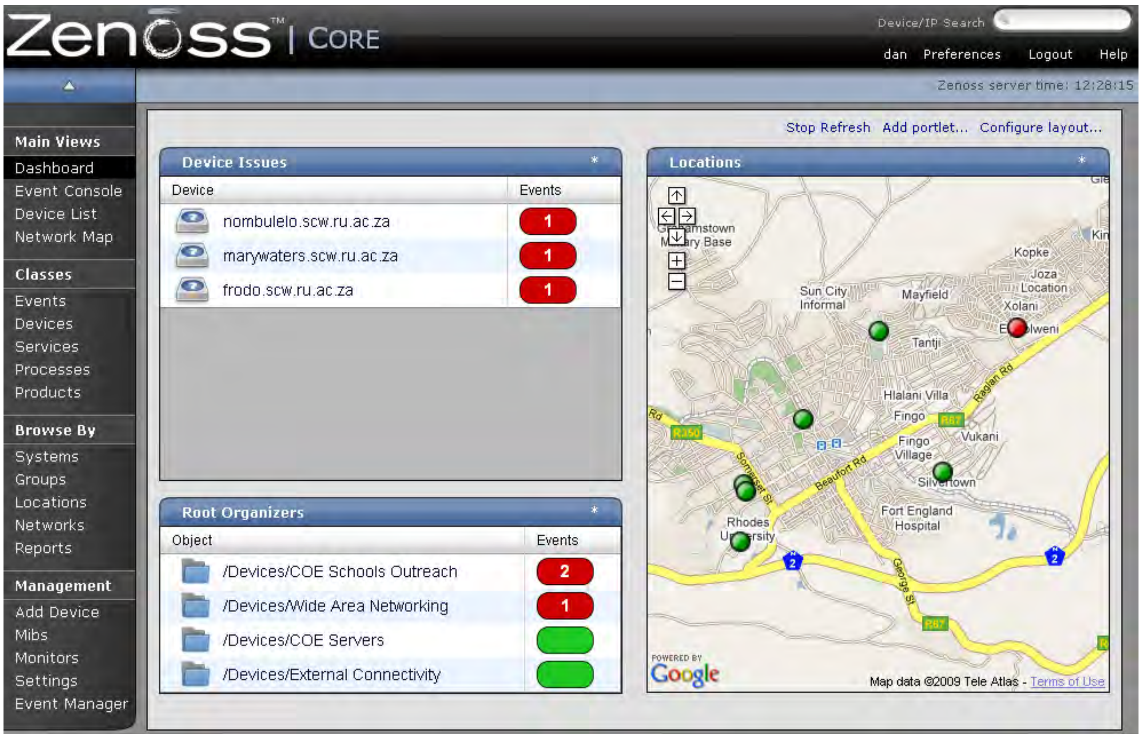
Zenoss Core is a web based application which installs to a central server on the network. It provides administrators with a single interface that allows device management; availability and performance monitoring; event management; system report generation; and user alert management [5].

The user interface of Zenoss Core provides a single access point to the monitoring system and requires no OS specific knowledge to use. The interface features customisable drag-and-drop *portlets* that allow a customised view of a network's health. An example of the interface is provided in Figure 2.6. In this example, there are three *portlets*: Device Issues, Root Organisers and Locations. The Device Issues portlet displays any problems with individual devices. The Root Organiser displays device issues per group of devices, and Locations shows the location of the devices, with their related status.

Device management uses a Configuration Management Database (CMDB) to store a model of the network environment and any changes to the environment [5]. With Zenoss Core, devices can either be added manually or by auto-discovering active devices by searching the network routing tables. Devices, and the variables which are monitored are modeled using SNMP, SSH or port scans. Devices can be organised by locations, groups, classes or systems.

Availability of devices or services can be monitored by using SNMP or ICMP. Availability statuses can be checked for network devices; TCP/IP or UDP services and ports;

[7]Zenoss Inc. - http://www.zenoss.com/



Device	Events
nombulelo.scw.ru.ac.za	1
marywaters.scw.ru.ac.za	1
frodo.scw.ru.ac.za	1

Object	Events
/Devices/COE Schools Outreach	2
/Devices/Wide Area Networking	1
/Devices/COE Servers	🟩
/Devices/External Connectivity	🟩

Figure 2.6: Example of Zenoss Core - CoE Testbed

URL availability and Microsoft Windows or Linux services and processes. Performance monitors collect time series data and provide an administrator with graphical analysis of file system statistics; CPU and memory usage. It also has plugins for Cacti support. Zenoss Core can generate an alert if a device crosses a pre-defined performance threshold (for example a disk running low on storage space).

Zenoss Core monitors a variety of sources for events. Sources monitored include system logs; availability and performance monitors; SNMP traps; and Microsoft Windows event logs. Events can generate a customised response by sending an email, a pager alert or running a script. Configuration of how Zenoss handles events is defined by alerting rules for users or groups of users.

Zenoss Core generates reports that allow an administrator to view the current status of devices, or historic events. The reports integrate with device management, performance monitors, events and user functions. The ability to create custom reports is also present.

Monitoring using Zenoss Core provides the administrator of a network of any size with a fully featured monitoring solution. It provides a number of monitoring methods, generates reports and alerts administrators of specified events. Zenoss Core was deployed for evaluation to monitor devices in the CoE Testbed network (described in Section 2.11.1) and alerted specific individuals of events for devices which they were responsible for. Another open source software package which is similar yet more extendable and customisable than Zenoss Core, is Nagios.

2.4.8 Nagios

Nagios[8] is a system, network and application monitoring software package. It is an open source, Unix based, monitoring package with a web based front-end. Nagios can monitor servers, network devices, applications and any other device that can be accessed via TCP/IP networking. It can monitor hosts running almost any OS and can be configured to work through firewalls, VPN tunnels, across SSH tunnels and the Internet [110]. Nagios monitoring is based on three simple states; green informs that a variable is in a working condition; yellow means a variable is in a questionable or warning state; and red denotes a critical state.

Nagios can monitor a variety of attributes of devices. Attributes can range from system attributes such as CPU, memory and disk usage to the status of applications, files and databases. It can be configured to continuously monitor network services such as SMTP, POP3, HTTP, NNTP, SSH and FTP [6]. It can also monitor environmental factors such as temperature and humidity, given the right devices. Alerts can take many forms, including email, pager alerts and SMS's.

Multiple Nagios installations situated around a network can form a distributed monitoring model, with multiple servers collecting data about devices in a network and reporting them to a central Nagios server. This is ideal for large networks with sites geographically separated from each other.

A strength of Nagios compared to other monitoring packages, such as Zenoss Core, is its modular structure. Nagios relies on external programs for service and host monitoring and these are known as plugins. Nagios is initially installed with many plugins for typical monitoring requirements, however, custom plugins are available and can be personally developed. A plugin is a simple program that outputs one of four possible conditions; OK; WARNING; CRITICAL; or UNKNOWN for unexpected errors [6]. Plugins can be created as a shell script, for example using bash[9], perl[10] or python[11]. By providing a method to create plugins, Nagios is extendable to monitor any device which can be measured electronically.

Nagios has a complex alerting or notification system. The sending side, Nagios, can be configured as to when a user should be notified; for example, only on week days during the hours 8 AM to 5 PM or when a device is in a critical state and not a warning state. The person who receives the notification has the ability to forward the notification to another user, or discard it. Nagios also has the ability to make use of external programs

[8]Nagios - http://www.nagios.org/
[9]GNU Bourne-Again SHell (Bash) - http://www.gnu.org/software/bash/
[10]Perl Programming Language - http://www.perl.org/
[11]Python Programming Language - http://www.python.org/

for notification, to send SMS's or leave voice mail messages on an administrators mobile phone [6]. Notifications can be configured to escalate, for example if a problem has not been addressed in a certain time frame, an email can be sent to the administrators supervisor.

The web front-end provides administrators with a range of information arranged by any issues involved. The front-end can provide summaries of the monitoring in many forms, including an overall summary, problematic services or hosts and group statuses [6]. Custom summaries can be created to show specific devices together. From the front-end, and in a network with more than one administrator, an administrator can inform colleagues upon accepting a particular problem to address, allowing the other administrators to focus on other problems.

Nagios provides a similar function to Zenoss Core, however, it is more extendable to fit into any network. Nagios provides an administrator with the ability to write custom plugins to monitor almost any device.

2.4.9 Section Summary

This section described a number of tools and applications available for monitoring and troubleshooting networks. The section started with some simple tools, ping and traceroute, which can aid an administrator in troubleshooting problems and configuring network routing. The cURL program allows the transfer of files using the URL syntax from a number of different protocols, and generates some simple throughput rates achieved in transferring a file. SNMP was discussed, demonstrating how an SNMP agent can be used to monitor various hosts and services throughout a network. Following this, NetFlow was described allowing for finer grained data than SNMP, capturing information about how the network is being used and how data flows around a network. A graphing solution, Cacti, was discussed which can generate graphs from a number of different input sources including SNMP and NetFlow data. Two open source web based monitoring software packages were discussed, including the advantages they can provide a network administrator. Zenoss Core takes input from a number of different sources to produce a comprehensive summary on the health of a number of devices, services and applications. Nagios, a similar yet more extendable and customisable package than Zenoss Core, allows data input through the use of custom plugins.

Monitoring can produce a large amount of information from the data it collects but this information needs to be used to effectively manage a network. The following sections describe relevant network related tools and solutions.

Subnet	IP Start	IP End
10.0.0.0/8	10.0.0.0	10.255.255.255
172.16.0.0/12	172.16.0.0	172.31.255.255
192.168.0.0/16	192.168.0.0	192.168.255.255

Table 2.2: Private IP Address Space [78]

2.5 Network Address Translation (NAT)

In response to the fast growing IP based Internet, where IP addresses were being depleted rapidly, Network Address Translation (NAT) was seen as a possible solution [22, 97]. NAT was first defined in RFC 1631 in 1994 [22], however, it was later made obsolete and replaced by RFC 3022 in 2001 [97]. NAT was intended to be an interim solution until a new IP standard was developed with a larger address space [22]. A new IP protocol, IP version 6 (IPv6), was introduced providing a much larger address space [21]. However, uptake and support of IPv6 is still underway.

Another solution to the rapidly depleting unique IP addresses was the introduction of private IP ranges as defined in RFC 1918 [78]. Specific IP ranges were allocated for private use, which required no application to an Internet registry for the allocation of a unique range of IP addresses [78]. The IP ranges that were allocated can be seen in Table 2.2. However, hosts assigned with a private IP can only communicate with hosts within the private network, they cannot have IP connectivity to any host outside of their network [78]. Using NAT, external IP connectivity is possible for hosts within a private IP range.

NAT is a service configured on a router to translate internal (LAN) IP addresses to an external IP address(es) which is Internet routable [22]. In some cases there may be more than one external IP address for NAT to utilise. NAT is required when a LAN cannot communicate with external hosts for security reasons or because the LAN uses a private IP range [97]. As an example, an Internet Service Provider (ISP) may assign a single unique IP address for use with an access router, NAT would allow hosts in the LAN to simultaneously use the Internet using the single IP address assigned to their router [97]. An ISP may also provide multiple IP addresses for the access router to utilise.

When NAT receives an outbound IP packet from the LAN, it rewrites the source IP address and the TCP/UDP port number [97]. The source IP address will be rewritten with an Internet routable IP address. The translation process also requires checksums within the IP and TCP/UDP packets to be rewritten to reflect the changes. NAT stores information about the state of each TCP/IP connection and when a packet is received from outside the LAN it checks the state information and correctly rewrites the destination IP address of the packet to the host in the LAN which requested it.

As a result of ISP's providing a limited number of unique IP addresses for networks to use due to their limited availability, NAT and private IP ranges can be used to provide hosts within a LAN with shared Internet connectivity. The next section describes PPPoE, a protocol to encrypt and authenticate connections to securely transport data over an Ethernet link.

2.6 Point-to-Point Protocol over Ethernet (PPPoE)

Point-to-Point Protocol over Ethernet (PPPoE) is a method for encapsulating Point-to-Point Protocol (PPP) frames within Ethernet frames [58]. PPP is described in RFC 1661 [89] and PPPoE in RFC 2516 [58]. A frame is a unit of transmission at the datalink layer, layer 1 of the OSI stack [44]. PPP operates at the datalink layer and can provide authentication, encryption and compression over a point-to-point link [89]. PPP provides a method to encapsulate network layer protocols (such as IP or ICMP) simultaneously over the same link [89].

By encapsulating PPP in Ethernet frames, it is possible to overcome some of the limited security features of Ethernet such as IP address conflicts. PPPoE was initially designed for use by ISPs with broadband access technology over an Ethernet network [58]. However, it can be used over any point-to-point Ethernet network which uses a shared connection medium (such as a wireless network) to enforce encrypted and authenticated links for authorised users.

The PPPoE connection has two stages, the discovery stage and the PPP session stage [58]. The discovery stage is inherently a client-server relationship, with a PPPoE server and PPPoE client. During this stage, the client discovers a server by identifying the Ethernet MAC address of the server. It is possible to have multiple PPPoE servers within a network. After identifying the server, a PPP session is created, involving the client and server allocating a PPP virtual interface for communication with each other [58]. The established connection is commonly referred to as a tunnel. The tunnel can carry network layer protocols over the encrypted, authenticated and compressed PPPoE tunnel [58].

PPPoE can be used to limit network use to authorised users and ensure encrypted IP traffic over an Ethernet network. The next section explains DNS and describes a widely used application that implements it, BIND.

2.7 DNS and BIND

The Internet, and every IP based network, works by assigning a globally or locally unique IP address to each host (server, router, interface) on the network. The Domain

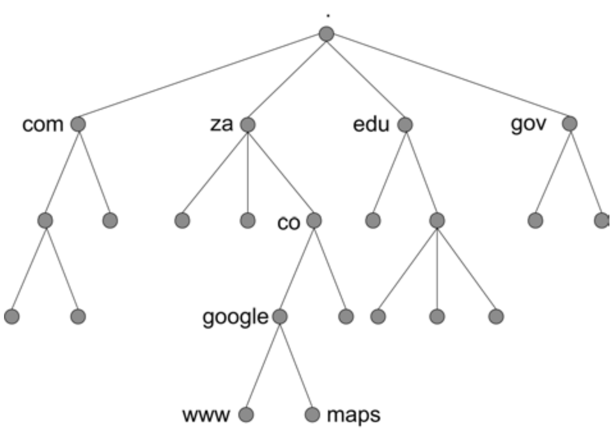


Figure 2.7: Domain Name System (DNS) Tree Structure

Name System (DNS) provides the ability to assign human understandable names (such as *www.example.com*) to those IP addresses (such as *192.0.32.10*) [2]. The standards which were published in 1987, RFC 1034 [65] and RFC 1035 [66], are still the Internet base standards for DNS which are used today.

A name server (DNS server), present in a network, allows any host to query it to resolve a resource name (for example a website) to an IP address. A name server is a specialised database which facilitates the translation of resource names to IP addresses (forward mapping), or IP addresses to resource names (reverse mapping) [2].

Before the discussion on types of name servers, a brief introduction to DNS is provided. Following this the widely used DNS implementation, Berkeley Internet Name Domain (BIND)[12], is discussed.

The structure of the DNS database can be seen in Figure 2.7. It is represented in an inverse tree or hierarchical structure. At the top of the tree is the root node followed by Top-Level Domains (TLDs). Following the TLDs are the Second-Level Domains (SLDs) and beneath these are any number of lower domains. Each level is separated by a dot ('.'). TLDs are further separated into Generic Top-Level Domains (gTLD) and Country Code Top-Level Domains (ccTLD). An example of gTLDs are *.com*, *.net*, *.org*, *.gov* and *.info*. Examples of ccTLDs include *.za* (South Africa), *.us* (United States), *.de* (Germany), and *.nz* (New Zealand).

A domain name is a combination of an SLD name and a TLD name, and is represented from left to right with the lowest level of the hierarchy on the left and the highest level on the right [2]. Each domain may have several subdomains of its own, for example *www.example.com* is a subdomain of the domain *example.com* [3].

The main goal of the design for DNS was to decentralize administration via delegation.

[12]Berkeley Internet Name Domain (BIND) - https://www.isc.org/software/bind

Organisations are delegated administration over particular domains, and may divide that domain into subdomains. Each subdomain can then be delegated to other organisations, and so forth. An organisation becomes responsible for maintaining all the data for that subdomain. The parent of the subdomain retains only pointers to the subdomains data, so that queries can be referred there [3]. For example, the domain *ru.ac.za* is delegated to Rhodes University and they manage all subdomains within their domain.

A name server generally has complete information about some part of the domain namespace, called a zone, which they are provided with from file or from another name server. A name server can then be called authoritative for that zone. A name server can also be authoritative for multiple zones. However, there are different types of name servers [2].

Four basic types of name servers exist: master (primary), slave (secondary), caching and forwarding name servers. A master name server contains one or more zone files for which it is authoritive. As DNS is a critical service for any network, slave name servers replicate the data of the master to be an additional authoritative name server for the zone. Caching name servers obtain requested DNS records from authoritative name servers and store the result locally until the records Time-To-Live (TTL) has expired. A forwarding name server is one that forwards all queries to another DNS server and caches the result until the TTL expires [2].

A master name server (zone master) contains the zone file whereas a slave name server (zone slave) obtains its zone information from the zone master. The zone master can indicate a change to the zone and all zone slaves can update their zone information. It is possible to have multiple zone masters for a zone, however, each zone file will have to be updated if changes in the zone information occur. Both the master and slave name servers are said to be authoritative for their zone [2].

A caching name server obtains information from a zone's authoritative name server in the form of Resource Records (RR) in order to satisfy a host query and the RR is cached. Subsequent requests for the same information results with a response from the name server's cache. The RR is stored until the TTL has expired and at this point the RR is removed from the cache. By using a caching name server DNS request performance can be significantly improved. The RR remains in the cache until the TTL expires or until the name server is restarted [2].

A forwarding name server forwards all queries to another name server, and caches the results. This provides improved response times for frequently requested DNS information and also eliminates additional external traffic [2].

Implementing the various types of DNS servers is the Open Source application BIND [46]. BIND is available for installation on numerous OSs including the BSD family, Linux platforms, Microsoft Windows 2000 and Windows NT 4.0. BIND is widely viewed as the reference implementation of the Internet's DNS. Due to its stability and high quality it is also the most widely deployed DNS software [2, 3].

For any host which is connected to the Internet, DNS is a critical service. Configuring BIND in its simplest form, as a forwarding and caching name server, a network can have improved response times for name lookups for DNS. The next section discusses the Squid proxy server, which can be configured to handle all WWW based requests for hosts in a network. Squid also provides a cache for storing downloaded web content, which in turn improves response times for already cached items.

2.8 Squid Caching Proxy Server

Squid[13] is an open source caching proxy server. Squid supports a number of protocols, including HTTP, HTTPS and FTP. Squid is at version 3.1 at the time of writing, yet the most stable version is currently 2.7. As a proxy, Squid accepts requests from a client, processes the request and then forwards the request to the destination server (on behalf of the client). The client's request may be logged, rejected or modified before it is forwarded. As a cache Squid stores retrieved web content for possible later use. Subsequent client requests for the same web content can then be retrieved from the cache.

Using Squid as a proxy and a cache provides a number of advantages to a network: one can utilise less Internet bandwidth by retrieving regularly used content from a cache; reduces load times for web pages; protects internal hosts by proxying their traffic; collects statistics about Internet traffic; prevents access to inappropriate web content; ensures only authorised users have access to the Internet and enhances user privacy by filtering information from web requests [117].

A Squid installation can provide all the various advantages discussed above, however, if firewall rules are not in place to restrict HTTP traffic, Internet bound connections could simply bypass it. Using necessary firewall rules will force connections to use the proxy to retrieve web content. This may also require the configuration of the users' web browsers (or user agents) to use the proxy. Authentication of users can then be enforced. However, if no authentication is required and it is impractical to modify the configurations of users' browsers, Squid can be configured as a transparent proxy. Squid must be compiled at installation with support for transparent proxying. Appropriate firewall rules must be configured to forward HTTP connections to the Squid service for a transparent proxy.

Squid is configured to listen on a particular IP address and a specific port. If Squid

[13]Squid Caching Proxy Server - http://www.squid-cache.org/

is installed on a firewall, it is usually best to only listen on the internal (LAN) facing interface. Squid can be configured to only allow authorised users access to the proxy and cache. Authentication is aided by an authentication helper, which is an external script which Squid calls. Authentication through Squid is made possible by various Access Control Lists (ACL) and by using an authentication helper thread. Squid can utilise many different types of authentication helpers including checking credentials using RADIUS, LDAP, ident, NTLM and MySQL. As mentioned earlier, authentication can not be used if Squid is configured as a transparent proxy as the client does not send a proxy authentication header [117]. Further details about the Squid configuration is described in Appendix B.2.

Squid has extensive logging capabilities, logging the application itself, access to the proxy by clients and storing content in the cache. Of these logs, the access log is the most important as it logs every request which passes through the proxy. It logs information such as web content retrieved; whether the content was in the cache; source IP address; username; and the number of bytes transferred in retrieving the content. As the access log contains a line for each client request, and with Squid proxying requests for a large number of requests, the log files can become very large. It is advisable to *rotate* the log files after a specific period, this can be achieved with a *cron-job* and the *squid -k rotate* shell command. This command renames the current log file and Squid opens a new file. The specific period after which log files should be rotated is dependant on the amount of requests the Squid proxy handles, as the access log file can become relatively large. A Squid configuration directive can limit how many old log files must be stored [117].

Squid provides a method to monitor the health of the application via SNMP. SNMP must be configured when the Squid package is installed. Access to Squid monitoring via SNMP is configured by setting a community string and a port number to listen on for SNMP requests. SNMP data from Squid can be retrieved by using the following: an SNMP tool such as *snmpwalk* or *snmpget* (see Section 2.4.4); using the correct community string; and walking the MIB tree *iso.org.dod.internet.private.enterprises.nlanr.squid* or *.1.3.6.1.4.1.3495.1.* Further to this some useful MIBs for Squid are discussed in Appendix A.2.

It can make use of extensive ACLs to provide a highly configurable and secure proxy server. Squid can route content requests to servers in a variety of ways to build extensive caches and optimise network throughput. Monitoring of the Squid proxy can be performed using SNMP. It is an invaluable application for networks with limited Internet resources as it reduces bandwidth use and improves response times for requests. It can control access to the Internet by only allowing authenticated clients, and limit those clients' Internet usage to acceptable levels.

Tools and applications can use the data and log files that Squid generates to further provide added functionality. LightSquid uses the Squid access log file to generate reports and summaries of the transactions that Squid has performed. LightSquid is discussed in the next section.

2.9 LightSquid

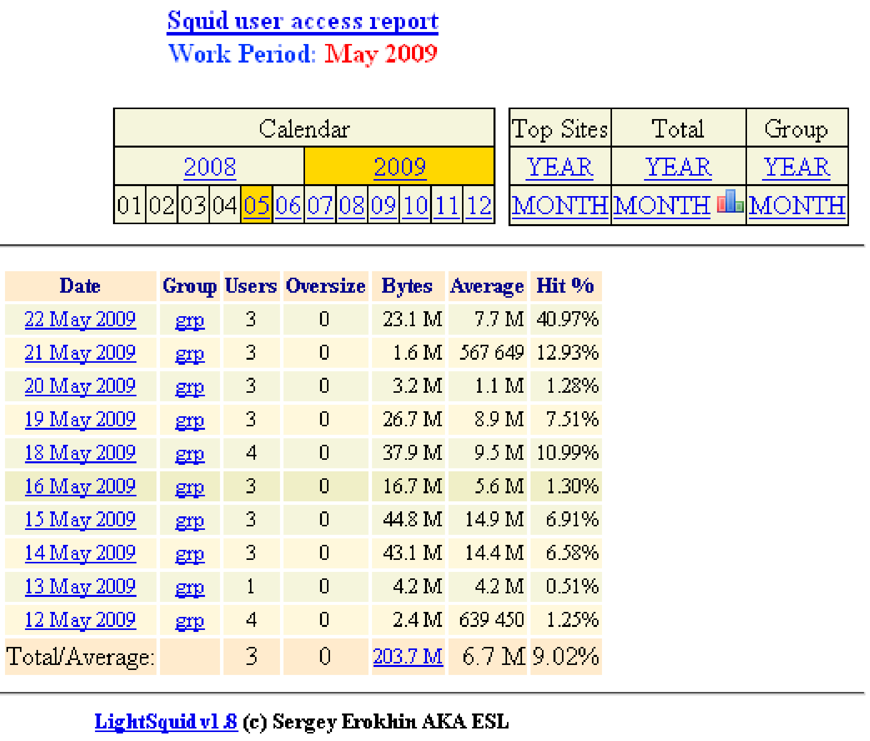
LightSquid[14] provides Internet usage summaries which are presented as a web based application. It processes the Squid access log file to provide useful information about web transactions performed by the proxy server.

LightSquid is separated into two components, the parser and the web front-end [23]. The parser is a perl script which processes the Squid access log file and creates reports. The parser is run automatically every predetermined period, the recommended period is 20 minutes [23]. The reports are separated into individual folders for each day. Within the report folder, three types of summary files exist. The first type of file is a summary for each user (if using Squid proxy authentication) or IP address. It provides a summary of the web pages that were visited by that user, the number of connects to that web page, the total bytes transferred in retrieving the web page and the times at which the web page was retrieved. Each user or IP address has their own file within the report folder. The second type of file is a summary of totals, storing the total bytes transferred for the day, total bytes transferred per user or IP address and the total connects per user or IP address. The last type of file is a features summary storing information about the cache usage: the *cache hit* percentage; bytes retrieved from the cache; *cache misses*, bytes retrieved from the Internet; and total bytes retrieved for that day.

The web front-end displays the reports that have been generated by the parser in a graphical form [23], which can be seen in Figure 2.8. LightSquid provides a number of levels of summarisation including total bytes transferred by the Squid proxy per day, per month and per year. It provides the ability to *drill-down* into this information to see more data. It is possible to ascertain the number of bytes transferred per user and per user group. Reports are compiled ranking popularly accessed websites, either sorted by the number of connects to that website or the number of bytes transferred in retrieving it. These popularly accessed website reports can be per day, per month or per year. Bar graphs can be generated displaying Internet usage per day for the period of a month, either showing all users or a single user.

LightSquid provides a graphical and simple method for analysing the Squid proxy ac-

[14]LightSquid - http://lightsquid.sourceforge.net/


Squid user access report
Work Period: May 2009

Calendar											
2008						2009					
01	02	03	04	05	06	07	08	09	10	11	12

Top Sites	Total	Group
YEAR	YEAR	YEAR
MONTH	MONTH	MONTH

Date	Group	Users	Oversize	Bytes	Average	Hit %
22 May 2009	grp	3	0	23.1 M	7.7 M	40.97%
21 May 2009	grp	3	0	1.6 M	567 649	12.93%
20 May 2009	grp	3	0	3.2 M	1.1 M	1.28%
19 May 2009	grp	3	0	26.7 M	8.9 M	7.51%
18 May 2009	grp	4	0	37.9 M	9.5 M	10.99%
16 May 2009	grp	3	0	16.7 M	5.6 M	1.30%
15 May 2009	grp	3	0	44.8 M	14.9 M	6.91%
14 May 2009	grp	3	0	43.1 M	14.4 M	6.58%
13 May 2009	grp	1	0	4.2 M	4.2 M	0.51%
12 May 2009	grp	4	0	2.4 M	639 450	1.25%
Total/Average:		3	0	203.7 M	6.7 M	9.02%

LightSquid v1.8 (c) Sergey Erokhin AKA ESL

Figure 2.8: Example of LightSquid

cess log. Various other Squid access log file analysers are available including Calamaris[15], Squish[16] and Webalizer[17].

Utilising a Squid log file analyzer, such as LightSquid, it is possible to effectively monitor the amount and type of content which is being accessed from the Internet. Squid can only provide these advantages to a network if web bound traffic is forced to pass through it and this can be achieved by configuring a firewall to forward all web requests to Squid.

2.10 Firewalls

Two firewall implementations are discussed in this section and they are FreeBSD's Internet Protocol Firewall (IPFW) and OpenBSD's Packet Filter (PF). Firewalls play a key role in network security. Used as a choke point at the edge of a network, they enforce fine-grained access control. FreeBSD and OpenBSD have reliably made good platforms for firewall deployments through their stable development process allowing them to be configured in a secured manner. FreeBSD and OpenBSD provide high performance networking that is

[15]Calamaris - http://cord.de/tools/squid/calamaris/
[16]Squish - http://ledge.co.za/software/squint/squish/
[17]Squid Logfile Analysis - http://www.squid-cache.org/scripts/

fundamental to the scalability of a firewall [52]. A firewall is the entry point for many network attacks and an insecure firewall makes for an insecure network. Following the discussion on IPFW and PF two traffic shaping mechanisms are discussed. They are AltQ and Dummynet found in Sections 2.10.2 and 2.10.3 respectively.

2.10.1 IPFW and PF

IPFW is FreeBSD's primary firewall and has been included in the OS since FreeBSD 2.0 [4]. In 2002 IPFW was rewritten as part of the FreeBSD 5 development. It was aptly renamed to IPFW2, however, the FreeBSD community continues to refer to it as IPFW. OpenBSD prior to 2001 utilised the firewall IPFilter, however, a licensing dispute forced the maintainers of OpenBSD to remove it. PF was created to fill the gap in OpenBSD [39]. It was designed to integrate smoothly with OpenBSD and as such is usable and flexible. PF is included in a base install of FreeBSD as of 2004 [108], however, it is primarily an OpenBSD project.

Both IPFW and PF can be configured with either *stateless* or *stateful* rules for their packet filtering [4, 39]. A stateless ruleset filters packets on a *per-packet* level, restricting or blocking packets based on source/destination IP address or other static values. Stateless firewalls are not aware of traffic patterns or data flows and therefore could allow packets through which meet certain criteria but were never requested. A stateful firewall observes communication paths from start of the flow until the end, remembering the state of the flow. Both types of firewalls have their merits. Stateless firewalls can typically perform faster under heavy traffic flows whereas a stateful firewall is better at identifying unauthorised or unrequested communication.

IPFW and PF are composed of two parts, a kernel-level packet filter engine and a userland utility for controlling the firewall. However, PF provides a device node for further control (*/dev/pf*) [39]. Both firewalls support stateful and stateless processing of connections. Both firewalls contain customisable rulesets which configure the packet filtering.

IPFW has a list-based ruleset while PF is more object-oriented. PF's configuration is split into many parts, while IPFW configurations are scripts with rules processed in a specific (each rule is numbered) order. In IPFW the first rule in a ruleset that matches a packet is the rule that is chosen [4]. For example, if a ruleset allows traffic to port 22 (SSH) before a rule that denies it, the packet will be allowed. PF operates in an opposite manner to this and the last rule that matches a packet is the rule that is chosen [39]. In the same example SSH would be denied by PF.

In IPFW, denied packets are logged through the *syslog* facility [4]. In PF denied packets are logged to a pseudo interface called *pflog0* [39]. This interface makes it easy to use tools such as *tcpdump* to capture the logged packets allowing applications to monitor and analyse the firewall's activity without having to directly interact with the PF firewall. An example of such a system is an Intrusion Detection System (IDS) such as Snort[18] or Bro[19], however, these systems work inline with IPFW as well.

PF directly implements NAT and Quality of Service (QoS) via AltQ [55], whereas IPFW provides NAT by a userland program [29], however, QoS is integrated with Dummynet [79]. This provides no functional difference but the PF approach provides a single file for all configuration.

Two FreeBSD systems are provided for traffic shaping and QoS, AltQ and Dummynet. AltQ is closely tied with PF and Dummynet with IPFW. Both systems can impose limits on throughput via a particular network interface and therefore can be used to limit throughput over an Internet connection.

2.10.2 AltQ

ALTernate Queuing (AltQ) is a flexible traffic shaping mechanism which is integrated into PF [55]. In a default networking environment with no AltQ-type queuing, the TCP/IP stack processes the packets in the order that they arrive on the interface. The TCP/IP stack follows the First In, First Out (FIFO) principle when filtering packets. AltQ can modify this behaviour.

The core concept to AltQ is the queue. Queues are a form of buffer for network packets. Packets are held in queues until they are either dropped or sent to their destination according to the criteria that forms the queue. Multiple queues can be defined with AltQ. Queues are attached to particular network interfaces, defined with specific amounts of bandwidth and have a priority. Priority is a preference of which queue should be serviced with the least delay.

AltQ defines several types of queues including Priority-based queue (*priq*), Class-based queue (*cbq*) and a Hierarchical Fair Service Curve (*hfsc*) [39]. A priority-based queue is defined in terms of priority within the total bandwidth and a packet in a higher priority queue results in that packet being serviced before ones in a lower priority queue. A class-based queue is defined as a having a constant-size bandwidth allocation and this type of queue can also have an attached priority. Packets in a class-based queue are kept in the queue until bandwidth is available to transmit it. Hierarchical Fair Service Curve is a complex algorithm which ensures a fair allocation of bandwidth among queues in a hierarchy.

[18]Snort - http://www.snort.org/
[19]Bro Intrusion Detection System - http://www.bro-ids.org/

AltQ is an effective method for traffic shaping on networks of any size using PF. A similar mechanism provided by the IPFW firewall is Dummynet.

2.10.3 Dummynet

Dummynet was originally designed for testing network protocols over a simulated network [80]. However, its uses have extended into providing a comprehensive traffic shaping application built on IPFW [79].

Dummynet works by intercepting packets, using IPFW, and passing them through one or more objects called queues and pipes. The queues and pipes simulate effects such as bandwidth limitations, propagation delays, bounded-size queues, packet losses and multipath effects. Pipes are fixed bandwidth channels which carry packets. Queues are queues of packets with an associated weight (priority) which share the bandwidth of the pipe they are connected to. Each pipe or queue can be configured separately, allowing the application of different limitations to different traffic according to an IPFW ruleset. Dummynet is configured by the addition of pipes and queues using IPFW commands or an IPFW ruleset [4].

Implementation of the IPFW and PF firewall, including their rulesets and associated traffic shaping, can become overwhelming to configure for an administrator. However, a firewall is highly recommended in securing a network from unwanted traffic and efficiently utilising available bandwidth.

2.11 Research Networks

In this section two research networks are identified and discussed. Further to this, their purpose and connectivity are described. Issues have been identified and there is a need for a system to manage and monitor bandwidth usage within the networks. The Centre of Excellence (CoE) testbed network in Grahamstown provides a number of schools, Rhodes University students and staff members with connectivity to the Internet via a number of access technologies. The Siyakhula Living Lab (SLL) network, situated in the Dwesa-Cwebe region in rural Eastern Cape, allows the sharing of a single Internet connection via the use of a WiMAX test network within a community where there is no fixed line telecommunication infrastructure in place.

2.11.1 CoE Testbed Network

The Telkom Centre of Excellence (CoE) in the Department of Computer Science at Rhodes University has been conducting research into cost effective last mile Internet access so-

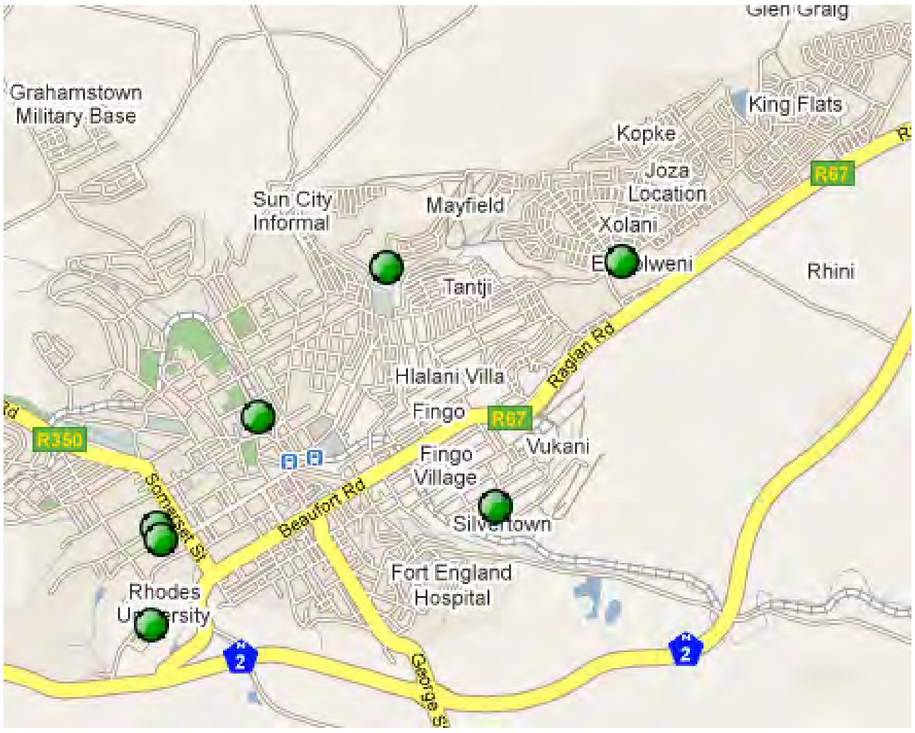


Figure 2.9: CoE Testbed Network - Location of Sites [36]

lutions since its inception in 1998. This research has led to the construction of the CoE testbed network and the ongoing testing and experimentation with technologies within the Grahamstown area[20]. Figure 2.9 shows the locations of some of the CoE network devices around Grahamstown. The network is used by various previously disadvantaged schools as well as a handful of Rhodes University staff members and post-graduate students. The network provides a method for connecting to the Rhodes University network and the Internet for staff and students, and provides only Internet access for the schools. The CoE network has two parts, the DSL portion and the wireless portion. The wireless portion is referred to as the Settler City Wireless (SCW) network.

One of the aims of the CoE was the identification of affordable telecommunication solutions for previously disadvantaged schools. The research over the years has included Digital Subscriber Lines (DSL) [10, 38], WiFi [11, 118] and WiMAX [8, 87] type connections. Subsequently, the SCW portion of the network consists of both WiFi and WiMAX technologies.

DSL has limited use within the CoE network in that it cannot be used over a maximum distance of 5 km (point-to-point via the copper) and many disadvantaged schools in Grahamstown are further than this limit [38]. Over time WiFi was introduced as it could be used to reach those individuals further away than 5 km and had benefits over DSL in

[20]Grahamstown GPS Coordinates - S33 18' 37.6" E26 31' 31.2"

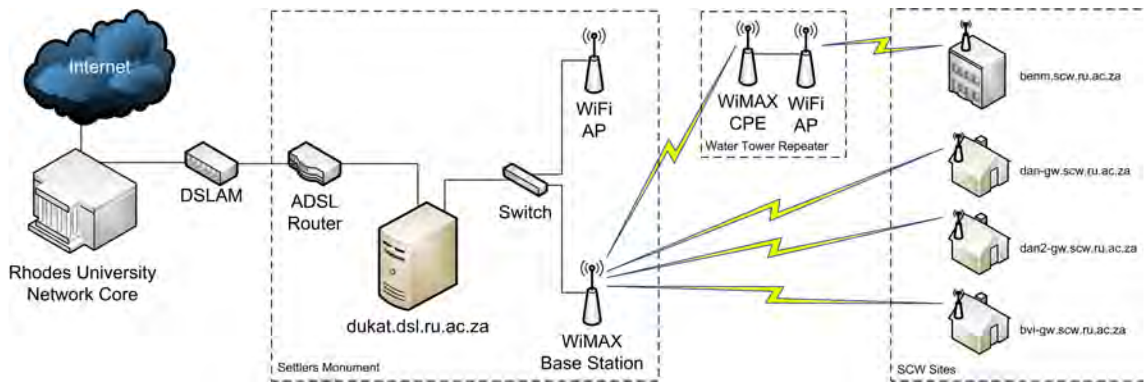


Figure 2.10: CoE Testbed - SCW - Logical Network Structure

that it was quick and easy to install (and so the SCW component of the CoE network was born). WiFi equipment was inexpensive and did not rely on any previous infrastructure. However, WiFi had limitations in that it was not designed to connect computers at distances over 100 meters [11]. It also requires line of sight between devices and is very susceptible to interference [10,116]. Progressing from the disadvantages of WiFi, the CoE research progressed to deploying WiMAX in the SCW network. WiMAX bypassed most of the problems brought about when using WiFi by providing better transmission rates, greater resistance to interference and connectivity to locations that do not have direct line of sight [88]. In addition, WiMAX can provide better transmission rates than DSL as it is generally available as a symmetric technology. The downlink and uplink throughput rates are often the same. In a simple experiment (using cURL; see Section 2.4.3) from one of the network sites throughput rates of 7 Mbps were obtained over the WiMAX link at a straight line distance of 2 km.

Figure 2.10 shows the logical layout of the SCW portion of the CoE network. The Access Router (*dukat.dsl.ru.ac.za*) connects to the Rhodes University network via an ADSL backhaul which terminates in the CoE DSLAM. The ADSL link provides a throughput of 6 Mbps download from the DSLAM and 800 Kbps upload. The Access Router routes connections from the WiFi Access Point and the WiMAX base station to and from the Rhodes University network. The water tower repeater extends the reach of the SCW network. It houses a WiMAX Customer Premise Equipment (CPE) and a WiFi Access Point for clients to connect to. The WiMAX CPE provides network backhaul to the WiMAX base station and therefore the Rhodes University network.

Connecting to the WiMAX base station and WiFi Access Point are the clients, either by WiMAX CPE or some form of WiFi network interface. Beyond the client's connections are the routers or individual PCs, which route packets from their LAN through the SCW network to the Internet, via the Rhodes University network.

The Access Router is installed with the FreeBSD 6.2 OS. A PPPoE service is provided by the Access Router, providing secure tunnels and ensuring encrypted and authenticated communication over the WiFi and WiMAX network. The Access Router implements the IPFW firewall, mainly for the use of Dummynet to provide Quality of Service over the ADSL backhaul. However, it is also used to secure the *open* wireless network from unwanted traffic and only allow traffic over the PPPoE sessions. A simple web interface provides monitoring data in the form of RRD graphs. This data is collected by the IPFW firewall using packet counters. Graphs are available for the total in/out Internet traffic; total in/out Rhodes traffic; total in/out Internet and Rhodes traffic per client router; and Access Router system health, namely CPU use, memory use and file system use.

Although there is a monitoring system in place a number of variables are not monitored. The system health of the client routers is not monitored, only their traffic inbound or outbound through the Access Router. The status of devices is not provided. The monitoring system does not provide information such as the type of traffic being generated, for example the type of protocols being used. In addition, no data is gathered regarding flow of packets in order to better understand how the network is being used.

Management of the network and the devices is done either by console access or SSH. Management of the Access Router requires an administrator skilled in using FreeBSD. Adding new PPPoE subscribers is performed by editing a text file and running custom scripts.

A need for a more substantial monitoring system has been identified with a comprehensive management interface which allows the addition and modification of new PPPoE subscribers.

2.11.2 Siyakhula Living Lab (SLL)

The Dwesa-Cwebe region is a rural community located in the Mbashe Municipality of the Eastern Cape (South Africa)[21] [60, 90]. This area generally lacks technical personnel, has a low level of economic activity, low income per capita, limited and intermittent electricity availability, limited telecommunications infrastructure (mainly cellular networks) and is geographically mountainous. The schools involved in this project and their geographical locations can be seen in Figure 2.11. These conditions have led to the deployment of a WiMAX network to connect a number of schools together to provide access to the Internet via a Satellite (VSAT) backhaul [60]. The community network is named Siyakhula Living Lab (SLL) [22].

This community network is in need of a system to manage and monitor bandwidth

[21]Dwesa-Cwebe GPS Coordinates - S32 17' 60" E28 49' 60"

[22]Siyakhula Living Lab - http://www.dwesa.org/

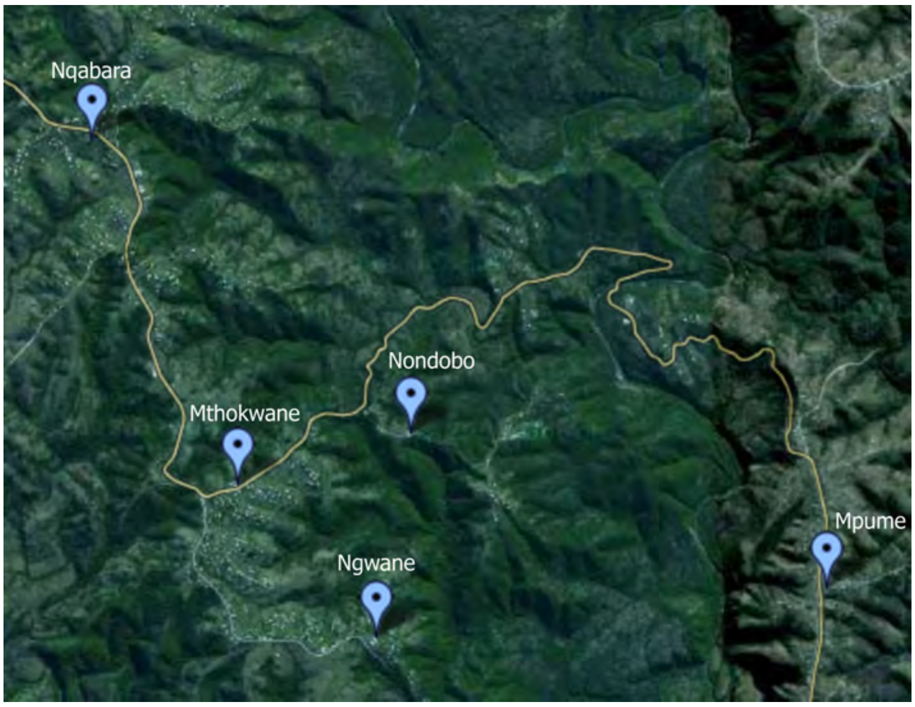


Figure 2.11: SLL Network - Location of Sites [36]

usage. Figure 2.12 shows the logical layout of the community network. The Community Wide Area Network (C-WAN) of SLL incorporates five schools; Mpume, Ngwane, Mthokwane, Nondobo and Nqabara. Nqabara has not been connected to the SLL network yet, however, future projects plan to incorporate the school. Nqabara seldom use their computer lab, as they have limited funds to incorporate ICTs into their curriculum.

Each school has its own Community Local Area Network (C-LAN) which includes its own servers and hosts. Each school is connected to a central point (at Ngwane) which houses the WiMAX (IEEE 802.16) base station technology linking the C-WAN together [60]. The C-WAN has a single Internet connection (via Telkom VSAT; satellite Internet backhaul) which is shared amongst the five schools. The Internet connection is located at the Mpume school.

Each of the schools have a customised installation of FreeBSD on a computer acting as their C-LAN router. The FreeBSD routers are low end Intel Pentium III computers. Each router has two network interface cards (an internal interface and an external interface). The internal interface is connected to the C-LAN via a switch at the school and the external interface binds an IP address on the open WiMAX IP network. The external interface of each router is connected to the WiMAX CPE, with the exception that Ngwane connects to the base station directly and that the router at Mpume is also an access concentrator terminating all PPPoE connections from the other four schools. The routers at each of the schools route packets intended for other C-LANs or the Internet to the

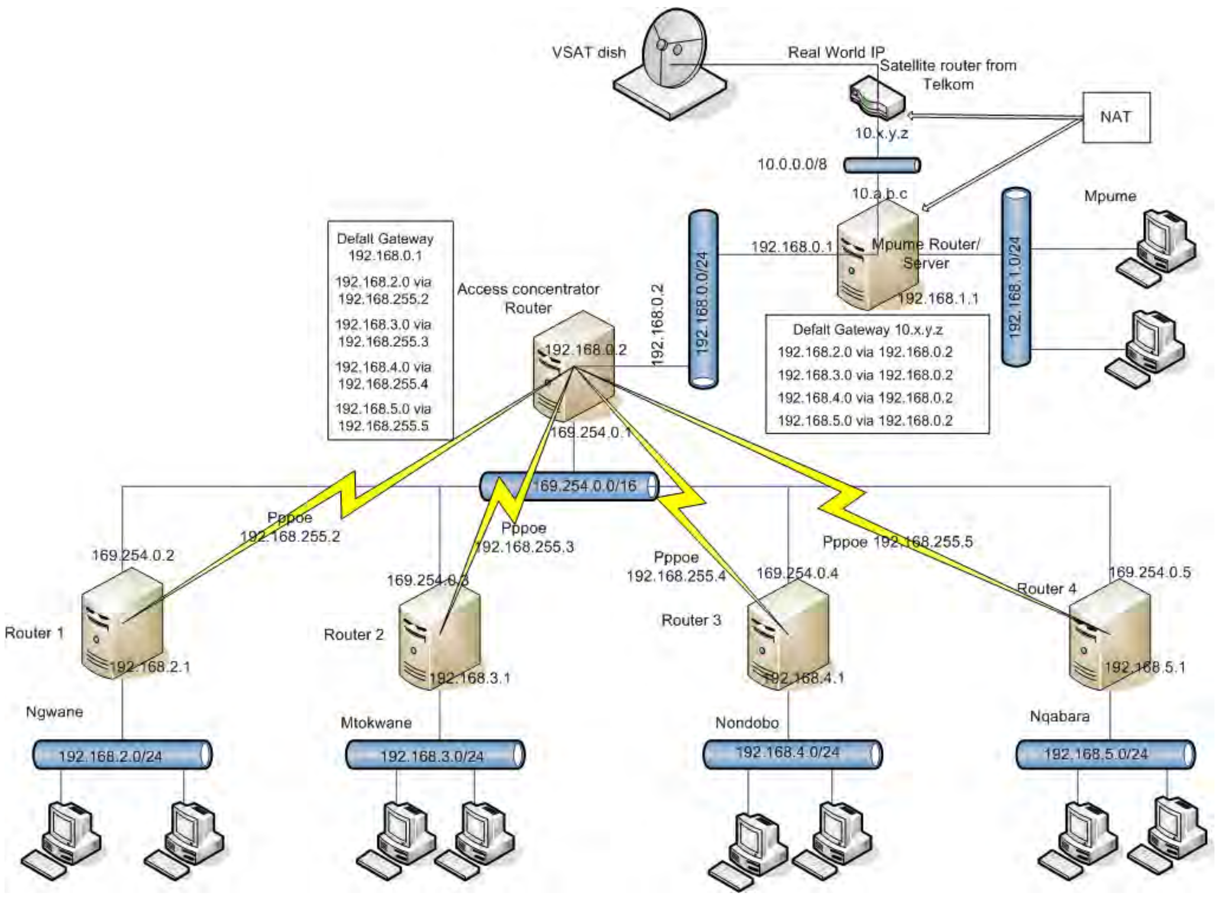


Figure 2.12: SLL - Logical Network Structure

access concentrator at Mpume. The access concentrator then routes the packets to the intended C-LAN over the WiMAX network, or routes the packets to the Internet via the VSAT link.

As the schools are connected using a shared medium, authentication and encryption of the traffic which is transmitted between them was required. A PPPoE service runs on the Mpume access concentrator to establish secure and encrypted tunnels over the WiMAX link between the school routers and the access concentrator.

The network is not without its flaws. The community network has a single point of failure at the Mpume school and if the access concentrator is down (or off), the other schools lose connectivity with each other and with the Internet. In addition, the VSAT connection has limited bandwidth available for the community and latency is very high. The community network required measures to be put in place to manage and monitor the use of bandwidth throughout the community, and detect network errors as and when they occur.

The two research networks were identified as in need of a system to monitor and manage bandwidth usage, hopefully making both networks more robust to failure. The next section describes two *ready made* projects which can aid in monitoring and managing a network.

2.12 Similar Projects

The project problem statement, in Section 1.1, describes the need for a custom system to manage and monitor bandwidth use in a community network. This section identifies existing systems which provide similar functionality, however, none match the requirements within this project's context. Four systems are presented along with their advantages and disadvantages. The four systems are Network Management Station (NMS), Network Discovery (NeDi), m0n0wall and pfSense.

NeDi and NMS primarily provide monitoring data, whereas m0n0wall and pfSense are firewall solutions which are customisable.

2.12.1 Network Management Station

The Network Management Station (NMS) [119] is available as a VMware virtual appliance[23]. As a virtual appliance it allows easy deployment into an existing network. It is based on FreeBSD 6.1 and provides a number of applications, accessible via an Apache web server. It uses open source applications and is a free appliance.

The applications, of most importance, that are installed include Cacti, Syslog-NG, MySQL, phpMyAdmin and Webmin [119]. Cacti is a graphing application and more information about this application is discussed in Section 2.4.6. Syslog-NG is a logging application which transfers and stores log messages in an encrypted channel. Further to this, it provides flexible sorting and management of system events and alerts can be generated based on the occurrence of system events. phpMyAdmin is a web based tool for use in management of MySQL databases. Webmin is a web based interface for system administration of Unix based machines.

The NMS is based on Cacti for graphing of data from devices on a network. Devices are added into Cacti which creates graphs and polls devices for SNMP monitoring information. An extension of Cacti provides scanning of a network (or subnet) for easy addition of SNMP enabled devices to Cacti. Syslog messages and SNMP traps can be sent to the NMS and alerts can be triggered and sent via email.

phpMyAdmin and Webmin are provided for troubleshooting purposes and do not require regular use. Webmin is a very powerful application which provides administration

[23]Network Management Station - http://www.vmware.com/appliances/directory/310

from a web based interface, however, it has a number of known security issues[24]. As it provides administration on system configuration files, the application has to be run as *root* (the superuser or administrator user account). A security breach could allow an attacker to gain root access and have full access to all services on machine.

The highlight of this system is the Cacti graphing application, providing extensive graphs on a wide range of devices.

2.12.2 Network Discovery (NeDi)

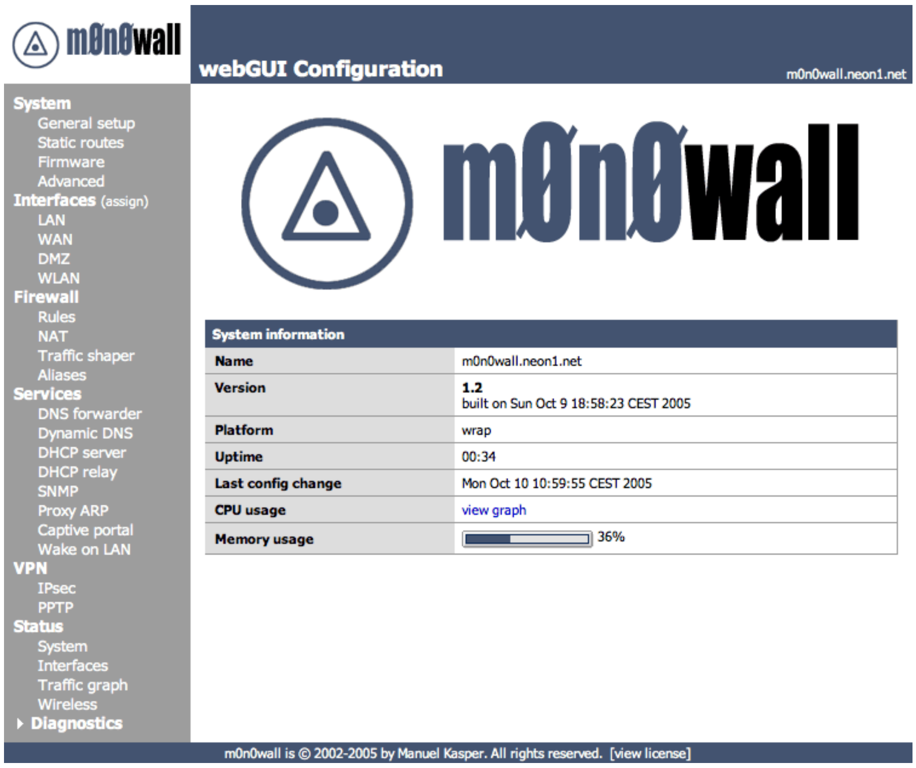
The Network Discovery (NeDi) [69] system is also available as a VMware Virtual Appliance called NeDi Virtual Appliance (NeDiVA[25]). NeDiVA is based on the Ubuntu OS and provides a fully functional NeDi deployment. NeDi is available for installation on a number of Linux distributions, OpenBSD and Microsoft Windows 2003 Server; NeDi is an open source application.

NeDi provides a full featured and simple to use web based interface which summarises gathered network monitoring data. It provides automated discovery of Cisco network devices and the network topology. Comprehensive diagrams can be generated displaying the network topology and it is possible to append this network topology to an architectural building plan. Network connections can be displayed at the switches port level, demonstrating how the physical connections are made.

Device management is provided allowing modification of configuration from a single point and many devices can be managed simultaneously. In terms of reporting it provides a method for locating users on the network. It is possible to drill down by DNS entries, to IP addresses, to MAC addresses, to which switch they are connected and to which port. IP address usage is calculated showing which IPs in the range are used and which are available. Performance of devices is monitored including traffic statistics on each network interface and CPU and memory utilisation. Performance data that is gathered is also graphed using RRD graphs. The status of network devices are monitored and alerts can be generated, either by email or SMS alerts. It also listens for SNMP traps and Syslog messages.

A highlight of this system is the automatic discovery of networked devices, however, this assumes that all the network devices are manufactured by Cisco. An automated discovery of network devices utilises a network scan and an Intrusion Detection System (IDS) would trigger alerts. The generated network topology diagrams, from the automatic discovery, are useful for visualising the network links and displaying which ports on which

[24]Webmin Security Alerts - http://www.webmin.com/security.html
[25]NeDiVa - http://www.vmware.com/appliances/directory/184



System information	
Name	m0n0wall.neon1.net
Version	1.2 built on Sun Oct 9 18:58:23 CEST 2005
Platform	wrap
Uptime	00:34
Last config change	Mon Oct 10 10:59:55 CEST 2005
CPU usage	view graph
Memory usage	36%

m0n0wall is © 2002-2005 by Manuel Kasper. All rights reserved. [view license]

Figure 2.13: m0n0wall - User Interface

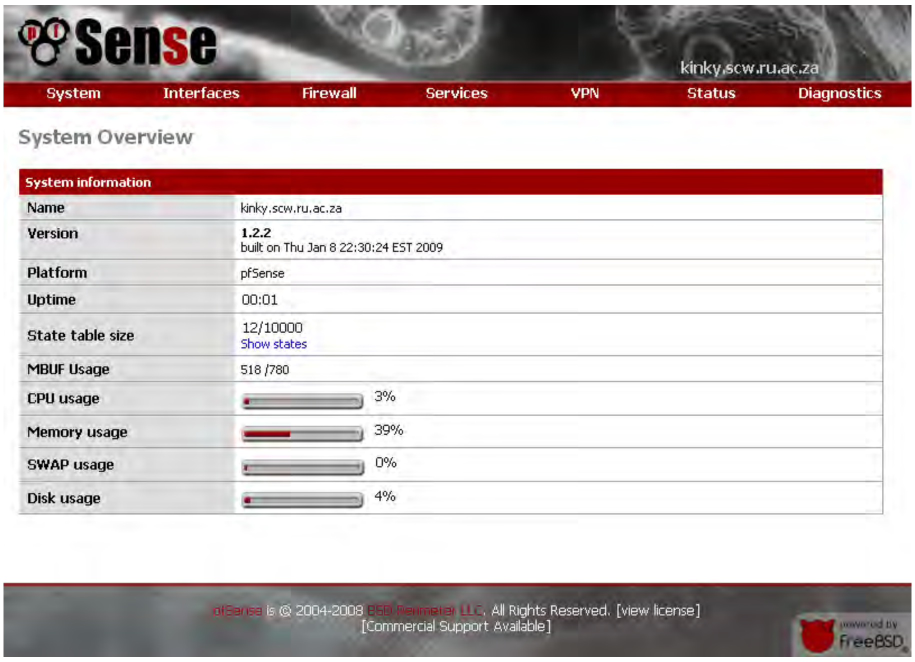
switches connect to which devices. However, a large network with many thousands of devices and a diagram of this nature could become very cluttered.

2.12.3 m0n0wall and pfSense

This section describes two firewall solutions, m0n0wall [56] and pfSense [74]. Both packages are free and open source and are based on a custom distribution of FreeBSD which has been tailored as a firewall and router. m0n0wall is a complete embedded firewall software package (based on IPFilter) that is intended to be used on an embedded computer. pfSense was created from m0n0wall, however, it is based on the PF firewall and intended towards full computer installations rather than embedded hardware. Both software packages are discussed further in this section.

m0n0wall[26] provides a full featured firewall which is based on a customized distribution of FreeBSD. It is possible to run off a Compact Flash card, hard drive or CD. A working installation of m0n0wall uses less than 5 MB of disk space. The requirements are an x86 type processor [56] with a suggested minimum of 64 MB of RAM and two network interfaces. Initial configuration is performed via a serial console and subsequent configuration is performed by a web front-end built in PHP. An example of the web front-end can be

[26]m0n0wall - http://m0n0.ch/wall/



System Overview

System information	
Name	kinky.scw.ru.ac.za
Version	1.2.2 built on Thu Jan 8 22:30:24 EST 2009
Platform	pfSense
Uptime	00:01
State table size	12/10000 Show states
MBUF Usage	518 /780
CPU usage	3%
Memory usage	39%
SWAP usage	0%
Disk usage	4%

pfSense is © 2004-2008 BSD Perimeter LLC. All Rights Reserved. [view license]
[Commercial Support Available]

Figure 2.14: pfSense - User Interface

seen in Figure 2.13. The figure displays the m0n0wall features on the left hand side and the system status in the main web frame. The configuration is stored in a single XML file. The sole purpose of m0n0wall is to provide security as a firewall and with limited functionality it has limited vulnerability to attack [56]. As m0n0wall is intended to be deployed on an embedded computer the services which are CPU and memory intensive have been limited.

m0n0wall provides a number of network pertinent features, including a web interface for easy configuration; serial console for recovery purposes; ability to be configured as a wireless access point; stateful packet filtering using IPFilter; NAT; DHCP server and client; PPPoE client; routing tables; caching and forwarding DNS; Virtual Private Network (VPN) support and traffic shaping.

pfSense[27] is an extension of the m0n0wall project by focusing on full computer installations rather than embedded computers. However, pfSense does offer an image for Compact Flash based installations but it is not their primary focus. Initial configuration for the network interfaces is performed via the console and subsequent configuration is performed using the web front-end. An example of the web front-end is seen in Figure 2.14. At the top of the figure the services it provides are listed in drop-down menus. pfSense uses PF firewall instead of IPFilter, and provides all the functionality of PF (see Section 2.10) including stateful packet filtering and comprehensive rulesets. pfSense pro-

[27]pfSense - http://www.pfsense.com/

vides a number of functions including NAT; redundancy of multiple pfSense firewalls using CARP; traffic shaping using AltQ; VPN support; PPPoE server and client; a variety of RRD graphs; DHCP server and client and SNMP monitoring.

m0n0wall and pfSense are free and open source and utilising them as a base they can be customised to provide additional functionality. However, the web front-end only has a single user account which provides administrators access to configuration. Users cannot view statistics or monitoring variables without being authorised. This allows only authorised administrators with access to troubleshoot network problems. In a network with multiple administrators or support staff, it would be preferential to have two types of views on the interface: initially presenting monitoring data and graphs which would be available for all users; and secondly providing access to the system configuration only after an administrator has authenticated.

2.13 Summary

The chapter began with a discussion of Internet bandwidth availability in South Africa. There is a limited availability of infrastructure as well as limited bandwidth specifically in rural South Africa. There is a demand for a system to manage and monitor a shared Internet connection. In terms of managing Internet Access, controls could be put in place such as blacklists and quota systems. Before putting these controls in place, further understanding of the tools and applications available to aid this project were discussed.

Numerous tools for monitoring and for troubleshooting a network were presented. The most important tools are identified here. Host and network segment status can be checked using *ping*. The tool *traceroute* affords confirmation of network routing configuration. *cURL* allowed throughput rate testing over a network segment, or download throughput rate testing from the Internet. An SNMP agent provides a method to monitor a host's system health, service status and network interface statistics. NetFlow provides more fine grained data than SNMP, allowing an administrator to understand how the network (and the Internet) is being used and by which hosts on the network. The graphing application Cacti, allows the construction of many types of graphs and can graph data from a variety of sources. Zenoss Core and Nagios provide customisable monitoring applications, however, the complexity of these applications suggests that the average user could find these difficult to maintain.

A solution to limited availability of unique IP addresses is NAT, which allows numerous hosts in a private network to communicate with the Internet by using a limited number of IP addresses. PPPoE was discussed, describing its use in encrypting and authenticating communications over a shared medium.

DNS is a critical service for any network and by providing a forwarding and caching DNS server it can improve response times for name lookups. The Squid proxy server provides a method for all hosts within a network to access the Internet. As Squid can be configured to store (cache) accessed web content, subsequent requests for the same web content can be retrieved from the cache, reducing Internet usage. Squid can be configured to provide blacklists, user authentication, dynamic caching and delay pools. LightSquid provides Internet usage summaries via the Squid proxy server.

A firewall is necessary for a network of any size and IPFW and PF provide methods to control what type of traffic is allowed out of and into a network. Both firewalls provide methods for implementing NAT and both provide methods for traffic shaping.

The CoE testbed and SLL networks were described. The CoE network is in an peri-urban setting whereas the SLL network is situated in a rural community. In both networks, a demand was established for a system to monitor and manage bandwidth use.

Before concluding the chapter, four similar projects were presented. They were NMS, NeDi, m0n0wall and pfSense. NMS and NeDi were evaluated and found to only provide suitable network monitoring. Both lacked bandwidth management features. m0n0wall and pfSense are very customisable applications, however, they lacked views of monitoring statistics for users who were not authenticated. This would prevent users from being able to troubleshoot network connectivity issues without help from an authorised administrator.

The next chapter describes the design of a solution for the research networks based on survey of literature that was conducted in this chapter.

Chapter 3

System Design

3.1 Introduction

This chapter describes the project system that was designed to meet the requirements and features highlighted in Chapter 1; a system is required to aid in monitoring and managing bandwidth in community networks with a shared Internet connection. The conceptual design of the system is described based on the evaluation of past research and literature in Chapter 2. It includes the application features, how they tie into the final system solution, the Graphical User Interface (GUI) design and the underlying connections between the various parts of the system. The discussion begins with the overall conceptual design of the system presenting the numerous features that each part of the system provides. The chapter then moves into greater detail of each part of the system and highlights the finer points of the design.

3.2 Conceptual Design and Feature Specification

The goal of this research project is to integrate multiple useful network applications and tools into a single and easily deployable system. The integration of the applications and tools is designed specifically to aid in managing and monitoring a community network, but also to manage the bandwidth use of the shared Internet connection. The design must allow the developed system to be easily implemented within the two testbed networks identified in Section 2.11, however, care must also be taken to allow for further configurations, extensions and customisations for alternative networks.

The design of the system is inherently that of a client-server architecture and as such is composed of two main parts. Figure 3.1 demonstrates a generic community network based on the project's testbed networks (described in Section 2.11). The Community Wide Area Network (C-WAN) has a single Community Network Core (CNC) with multiple Commu-

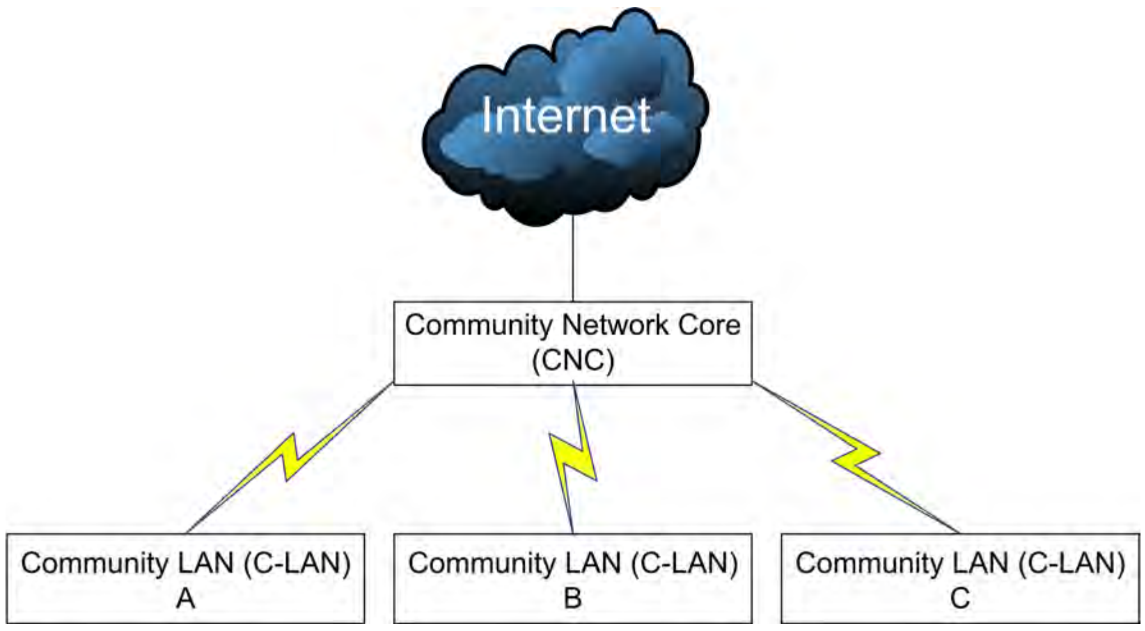


Figure 3.1: Generic Community Network

nity Local Area Networks (C-LANs) connecting through the CNC to obtain connectivity to other C-LANs or to have access to the shared Internet connection. The connection between the C-LANs and the CNC can be achieved via various access technologies, however in this system it is assumed that the connection medium provides a TCP/IP network. Access technologies between the CNC and C-LANs can include, but are not limited to, WiMAX, WiFi, xDSL and Ethernet (Section 3.7 provides a more detailed explanation on the types of connection mediums).

As previously stated, the design of the system is composed of two components forming a client-server architecture; the Access Concentrator (AC) which is the server component in this architecture, and the Community Access Point (CAP) which is the client. A C-WAN has a single AC at the CNC and multiple CAPs in communication with it from the various C-LANs. Figure 3.2 shows two CAPs in communication with the AC, however, there is no upper limit on the number of CAPs that can be in the C-WAN. Neither the AC nor the CAP are required to have a screen or monitor as they are both accessed via a web based GUI, discussed later in this chapter (Sections 3.4 and 3.6). A fully featured shell environment is also provided for administrators for troubleshooting purposes.

The CAP design requirements are that they are easy to deploy and configure, and provide a number of networking, monitoring and management functions for a C-LAN. The CAP is the default router for the C-LAN, all IP packets destined for outside the C-LAN are routed via the CAP to the AC, the AC then routes the packets to their destination, whether that is within the C-WAN or to the Internet. In addition, the CAP collects network monitoring data for the C-LAN. The monitoring data collected by the CAP is

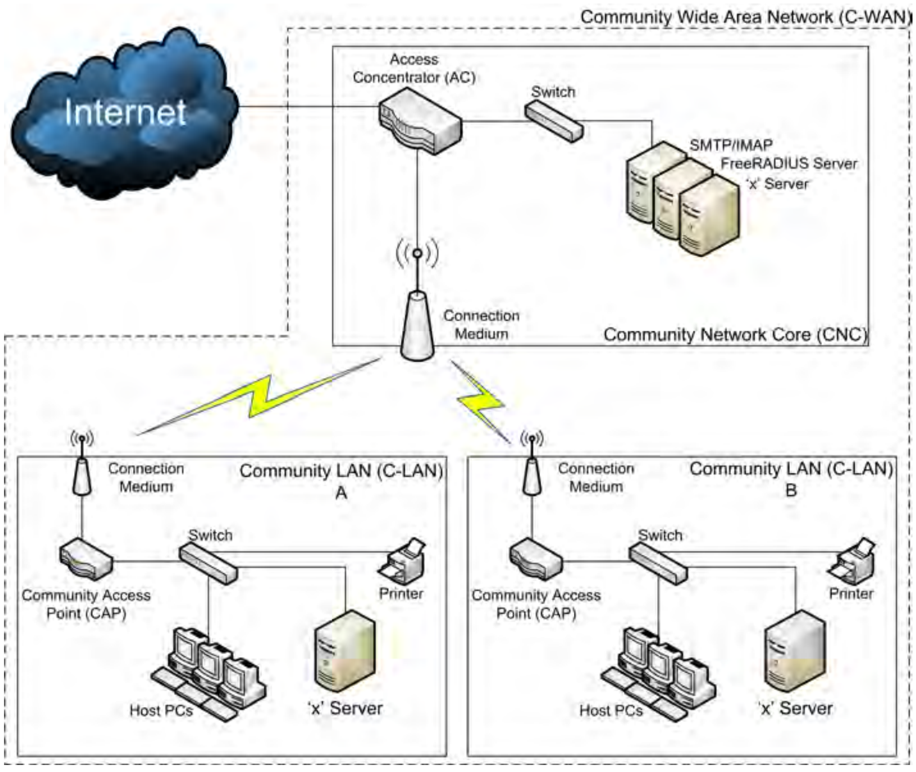


Figure 3.2: Conceptual System Design

communicated with the AC for aggregation and summarisation. CAP related monitoring data is critical in evaluating Internet usage per C-LAN (site) and per host. The CAP has a web based GUI to enter the system configuration and display monitoring information such as system uptime, upstream connectivity status, whether specific local services are running, and statistics gathered from the network interfaces. More detailed monitoring information regarding the status of the C-WAN and bandwidth usage is available on the Access Concentrators GUI.

The AC design requirements are more complex, although it is required to be easy to install, configure and use. The AC is dual purposed, first it routes IP packets between the CAPs and is the gateway to the Internet connection. Secondly, the AC performs a number of network management and monitoring tasks. The AC is at the core of the community network and provides network administrators with a number of tools to aid in everyday tasks. These applications and tools include monitoring and managing of critical services (and servers) within the C-WAN, monitoring and managing bandwidth usage throughout the C-WAN and providing network administrators with network wide access to a web based GUI to effectively manage the C-WAN. The ACs web based GUI allows administrators to configure various system settings on the AC and also provide a means to set bandwidth management related thresholds. The ACs GUI also displays a number of summarised monitoring variables, including the system health of the AC (CPU use,

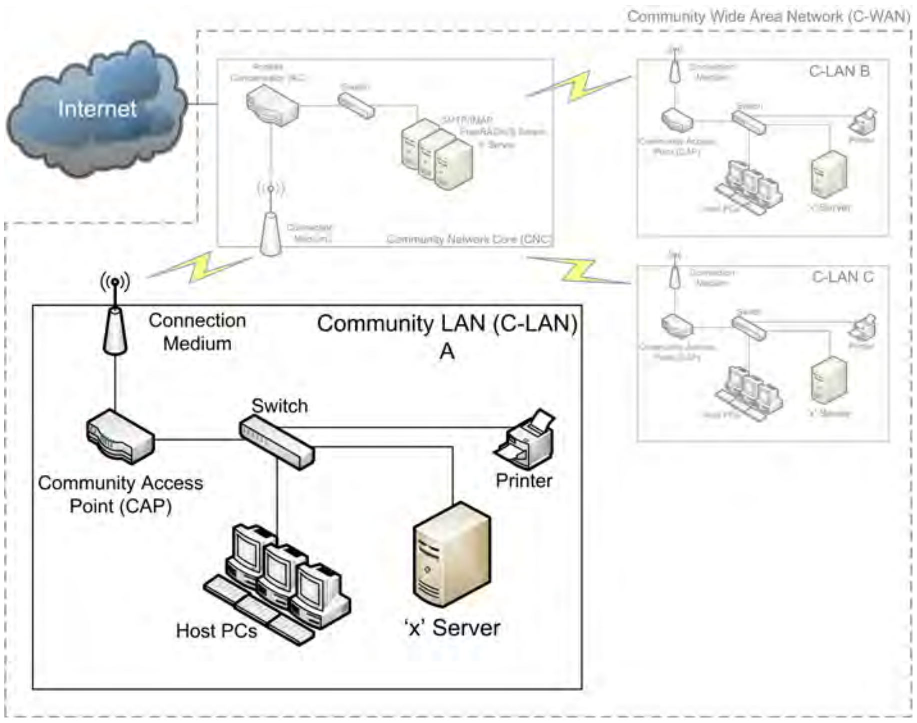


Figure 3.3: Community Access Point (CAP)

memory use and system uptime); the status of the CAPs in the C-WAN, their associated system uptime; the Internet usage per C-LAN (site), per host and per user; and the viewing of a list of the most accessed web sites in a specified period.

Various parts of the system design are discussed in detail in the following sections, highlighting how the parts are integrated and function together. First in Section 3.3 the CAP is detailed demonstrating how it provides the AC with critical monitoring information. The web based GUI for the CAP is also designed in this section. Following this, in Section 3.5, a discussion on the AC describes how the various applications work together to provide effective network management and monitoring and how the web based GUI is designed to link these applications together. Connecting the two parts is a discussion addressing the connection medium between the AC and the CAPs in Section 3.7. This section also details how the community network connects to the Internet.

3.3 Community Access Point (CAP)

The CAP is depicted in Figure 3.3 with the rest of the C-WAN greyed out. It is effectively a PC configured as a network router with custom modules installed, collects critical monitoring data for the C-LAN in which it is situated. The PC configured as the CAP is required to have a reliable, tried-and-tested network Operating System (OS). The OS is required to permit extensions and customisations of the applications and tools that are, and can be, installed. The OS is required to be open source and allow the installation of Open Source Software (OSS). By using a widely implemented open source networking OS, not only is an expense removed (in purchasing a proprietary OS) but generally large amounts of documentation exist which aid in the implementation process.

The OS is required to be able to run on a device with minimal hardware specifications. This is an important requirement as the CAP is either installed on low-end PCs or low-cost PCs which have minimal performance ratings. Adequate network performance, via the Network Interface Card (NIC), may be a requirement, as it could be possible that many hosts at each C-LAN will route their C-WAN or Internet bound traffic through the CAP.

The CAP is configured to communicate with the AC using an encrypted network tunnel. This is an essential requirement as the two testbed networks (Section 2.11) both use WiFi or WiMAX technologies to transmit data between the C-LANs and the CNC. Using encrypted and authenticated connections over wireless networks prevents unauthorised network access and network traffic is secured. In any communication over a shared medium where unauthorised users could get access to private data, the connection should be authenticated and encrypted.

The CAP is required to have a number of applications installed to facilitate network configuration and communicate network usage data to the AC. It is required to monitor the health of the system on which it is hosted. Monitoring system health provides a number of variables that can be collected and graphed such as secondary storage usage; RAM usage; CPU usage; and bytes in and out of the network interfaces. By monitoring these variables issues can be identified without the need of physically being at the PC being monitored. Issues that could arise include secondary storage space running low or RAM and CPU usage being very high. Variables such as bytes in and out of the network interfaces are also collected. By comparing bytes in and out with system variables such as RAM or CPU usage, it is possible to identify any hardware bottlenecks. By monitoring the system health of the CAP, it allows for the identification of hardware issues and the quick resolution of any problems that may arise.

As the CAP is the default router for the C-LAN and all packets entering or leaving the C-LAN are routed via it, information is collected about how many bytes are transmitted and received from outside the C-LAN. This information allows the modeling or graphing of total network usage patterns per C-LAN.

The CAP is not only required to monitor the flow of data in and out of the C-LAN, but also monitor flows of data between hosts in the C-LAN to hosts in the C-WAN or the Internet. For example, the CAP can provide data suggesting the external network interface is at 85% capacity. In addition to this information, the CAP can provide data suggesting which hosts within the C-LAN are responsible for this traffic.

The monitoring data collected by the CAP is sent on to the AC, which creates summaries and graphs of the data (more information on how this data is used is discussed in Section 3.5). The CAPs have been designed to require minimal management. They are required to be configured once (however subsequent configurations are possible) when they are installed at the C-LAN. They are required to be installed once and should need no further modification to configuration.

3.4 CAP Graphical User Interface Design

The CAP provides a standard Unix shell environment for administrators to troubleshoot the device, however, this is not the preferred interface for everyday use. In this section, the CAP GUI and its design requirements are discussed. The design goal for the GUI is primarily to allow system configuration of the CAP and secondly display monitoring variables in a meaningful way for users of varying levels of networking knowledge to understand.

It was decided that a web based GUI is preferred as any PC within the C-WAN with an Internet browser can access the information that it provides. Only authenticated users should be able to modify system configuration, however, the monitoring data is available to anyone within the C-WAN.

The GUI should be easy to navigate and to discover the monitoring statistics. A number of monitoring variables are viewable on the GUI by any user and this includes: upstream connectivity statuses; the system uptime; checking local services status; disk usage; and statistics from the network interfaces.

The device has three network interfaces, the internal, external and tunnel interfaces. The internal interface connects to the C-LAN, the external interface connects to the C-WAN and the tunnel interface provides a secure and encrypted tunnel over the external interface to the AC. Monitoring variables collected on the interfaces include link status, IP address, netmask, MAC address, media type and bytes in/out. A page within the GUI provides these collected network interface variables in a table.

Another page in the GUI provides a simple table for local services' status. If a local service or a network interface on the CAP is not up, the item is highlighted in red with the text *DOWN* otherwise if it is running it is highlighted in green with the text *UP*. A simple colour difference can allow novice support staff to confirm and resolve problems within the C-LAN.

The user has to be authenticated to modify the system configuration of the CAP. Configuration of the device entails giving the CAP a meaningful hostname; setting net-

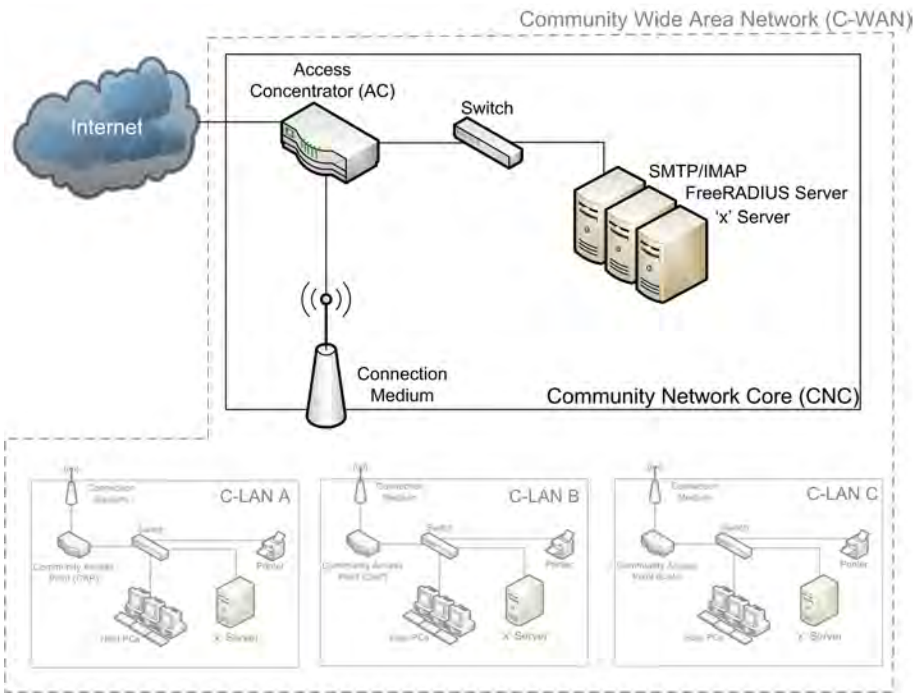


Figure 3.4: Access Concentrator (AC)

work related variables, including IP addresses and DNS servers; setting a username and password for encrypted and authenticated communication with the AC; and configuring the monitoring software and other services for the C-LAN.

The GUI for the CAP displays information regarding the C-LAN at which it is situated. For further monitoring information regarding the rest of the C-WAN, the ACs GUI should be visited. A hyperlink from the CAPs GUI transfers the user to the ACs GUI (see Section 3.6) to view C-WAN information. The subsequent sections discuss the design of the AC and its GUI.

3.5 Access Concentrator (AC)

In Figure 3.4, the AC is depicted with the other parts of the C-WAN greyed out as the AC is the focus of this section. The AC has a dual function as the C-WAN gateway router (for the shared Internet connection) and it also hosts many applications and tools, which are discussed further in this section. The primary function of the AC is to route traffic between the C-LANs, and to the Internet; and to facilitate encrypted and authorised network tunnels to the C-LANs over the C-WAN.

The ACs secondary function is to host monitoring and management applications which provide an administrator with a single interface to manage the C-WAN. These applications allow effective use of the shared Internet connection, and also provide Internet usage statistics for all users, hosts and C-LANs (sites) within the C-WAN.

The OS chosen has similar requirements to those discussed in Section 3.3. A reliable, well documented and widely used networking OS is required. The hardware specifications are similar to the CAP, however, the AC requires additional disk space for the storing of the proxy cache and network flow data.

A number of applications are required to be hosted on a web server running on the AC. Therefore making these applications accessible from anywhere within the C-WAN, using an Internet browser. User authentication is required for users to gain access to certain portions of these applications (for example their relevant configuration settings).

The AC has a caching proxy server which all clients on the C-WAN should use to connect to the Internet. Firewall directives prevent direct connections to the Internet by forwarding all Internet bound connections to pass through the proxy server. The proxy server's cache is configured to store regularly requested web content. By retrieving requested web content from the cache, Internet usage is decreased and response times are improved. The proxy server is configured to cache dynamic content specifically when identical content is hosted off multiple web servers. By recognising that this content is hosted on multiple web servers, the content is stored in the cache as a single item instead of multiple items for each URL. By caching dynamic content, requests to the Internet can be reduced when the same item is requested from a different web server.

The proxy server can be configured to require user authentication by matching credentials with those stored on an AAA (Authentication, Authorisation and Accounting) server. However, the proxy server can also be configured as a transparent proxy, requiring no user authentication. If no user authentication is present, user based Internet usage limits cannot be imposed. However, host and C-LAN based limits can still be imposed.

The proxy server is configured with the option to *blacklist* or deny access to certain content, allowing administrators to control the content that the C-WAN users can view. The blacklists are required to be customisable to block different forms of content, including specific web pages, domains, IP ranges, web page sizes or web server types.

The caching proxy server is configured to log to file all attempts to request information from the Internet. The log files are comprehensive, providing information about the user who requested the information, the time and date, the source IP address, the URL and the bytes transferred in retrieving the web page. By using these log files, summaries of bandwidth use per user (or per IP address) are compiled. By investigating the log files further, it is possible to identify popularly accessed websites (ordered by either bytes transferred or connects to that website).

Host based Internet usage is monitored by retrieving information about the network flows from the CAPs. Any flows that are recorded with a destination or source IP outside the C-WAN communicating with an IP within the C-WAN is considered to be Internet based traffic. Combined with this, all traffic which has a source or destination IP for the proxy server on the proxy port number will be considered web traffic as well. C-LAN (site) based Internet usage is determined by calculating the sum of all the hosts Internet usage within a particular C-LAN.

Delay pools are configured to limit throughput for particular users (or hosts) or to particular web sites (or domains). For this system, it was decided to configure four delay pools on the proxy server. The pools are *no delay*, *slight delay*, *significant delay* and *no access*. The pools have an effect over a particular period of time. Thresholds are configured for a particular period, for example, over x day(s) or per month. Different thresholds can be set for user based Internet usage, host based Internet usage and C-LAN (site) based Internet usage. Host and site limits on throughput are imposed by queues configured within firewall rules.

As a user's (or host's or site's) Internet usage increases, they move into a higher delay pool and when they reach a predefined threshold of Internet usage, their Internet access is denied (or their throughput is severely limited). The period and thresholds of Internet usage for moving between pools are configured via the ACs web based GUI (see Section 3.6).

The AC is required to graph all monitoring data received from the CAPs, the proxy server and the system health of the AC. By graphing the monitoring data, the data is displayed in a more meaningful way than interpreting an array of numbers. Numerous variables are graphed, including the system health data of each CAP and the AC. The graphs are grouped per device.

In the following section, the interface to the AC is designed, discussing how the previously identified requirements are integrated into a single and meaningful interface.

3.6 AC Graphical User Interface Design

The AC hosts a number of applications to aid in monitoring and managing a community network. An interface was required to link these applications together in a meaningful way to both the user and the administrator. It was decided that a web based GUI should be created for the interface, as the AC is not required to have a screen or a monitor and a web GUI can be accessed throughout the C-WAN. Any user within the C-WAN can view the monitored information, however, only authenticated users may modify system configuration, including thresholds and periods on Internet use. The use of colour was required, as colour allows for differentiation of a healthy network/system from an unhealthy one.

The GUI was designed to easily interpret the health of the AC, including the status of upstream connections. But more importantly, the GUI provides a number of statistics on the health of the CAPs in the C-WAN, and the usage of the shared Internet connection distinguished by each user, host and C-LAN (site). Once the user has been authenticated, links are provided for system configuration and bandwidth management.

Similar to the CAP, the GUI describes the system health status for the AC, the status on the services that it provides, disk usage and statistics about the network interfaces. The GUI also provides a summary of the CAPs situated within the network. The CAPs are listed in a table with their relevant status information. Information that is displayed, includes status (*UP/DOWN*), system uptime, date and time when it was last seen, IP address, subnet information and status information about the services it provides. Any item which has the status of *DOWN* is highlighted in red and a status of *UP* is highlighted in green. By clicking on the CAPs name, the user is redirected to the GUI hosted on that particular CAP.

A page is available for the status of the proxy's cache, providing a summary of the cache use. This includes statistics about whether objects were served from the cache (*HITS*), or whether they had to be downloaded from the Internet (*MISSES*). The status of the cache is also available, providing information about how much disk space is still available for caching web content. Another page provides a list of the most accessed websites, ordered by the total bytes transferred or total connects to that web site.

Following these pages, a user quota page lists all the users (or IP addresses) which have accessed the Internet via the proxy, with the total bytes transferred by that user. The user quota page also provides delay information about whether a user is in a delay pool or not. A page for C-LAN (site) and host quotas is available, listing the C-LANs and their specific Internet usage and whether they are in a delay pool or not. The Internet usage (per host and per C-LAN) is displayed as the total bytes transferred and also as a percentage of the total bandwidth available to the C-WAN.

Numerous configuration pages are available once the user has been authenticated, allowing changes to network settings, addition/removal of CAPs and adjustments to their site quotas, routing settings, proxy settings, blacklists and quota configuration for user and host quotas.

Two hyperlinks are available to the user from the AC's GUI. First, a link is available to view more detailed summaries of Internet usage via the proxy than is shown on the AC's GUI. The second link provides a number of graphs which have been generated by data received from the CAPs.

3.7 Connection Medium and Internet Connection

The connection medium could include the technologies Ethernet, DSL, WiFi or WiMAX. The technology used to connect the CAPs and AC does not have to be the same used throughout the C-WAN, some CAPs may connect using WiFi while others connect over WiMAX or DSL as long as they all provide TCP/IP.

Whichever connection medium is used to connect the CAPs to the AC, the connection is regarded as insecure and therefore traffic that is transmitted is authenticated and encrypted. The connection medium provides an open network on which an authenticated and encrypted connection can be configured. This configuration allows for standard endpoints to be utilised on which the traffic will be encrypted. Using an authenticated and encrypted connection ensures that only users who have access to the network are allowed to utilise the services it provides. A secured connection also protects network traffic from being intercepted by potentially malicious third parties, and restricts access to the shared services to recognized members of the community.

The connection to the Internet can be provided by a number of different technologies. This system does not require or rely on a specific type of technology to connect to the Internet.

3.8 Summary

A solution has been designed to meet the goals of the project's problem statement (stated in Section 1.1). The architecture that was chosen is inherently that of a client-server architecture.

The CAP, the client in the system design, is required to route packets via the AC to other parts of the C-WAN or the Internet, and collect monitoring information for its local network (C-LAN). System health of the CAP is monitored, and flows of data into and out of the C-LAN are logged. Information that is gathered by the CAP is communicated with the AC, which is aggregated and summarised. A web based GUI is accessible by any Internet browser within the C-WAN, however, only authenticated users may access the configuration portions of the interface.

The AC, the server in the design, is the default gateway to the shared Internet connection for the C-WAN. It also ensures the connections to the CAPs are encrypted and authenticated. The AC hosts a number of applications which will aid administrators in managing and monitoring the network. An administrator is able to set thresholds which control how much the users can utilise the Internet. As thresholds are exceeded users are moved into higher delay pools to limit their Internet usage to acceptable levels. Thresholds should not only be set for users, but for hosts and C-LANs (sites) on the C-WAN as well. Host and C-LAN based Internet usage is calculated by retrieving flow data from the CAPs. Regularly accessed web pages are cached using a caching proxy server to reduce Internet usage and improve response times. Users' Internet usage is summarised using the proxy server's log files and a list of popular web pages that consume the most bandwidth is produced. Bandwidth intensive web sites or domains can be placed within delay pools to limit their throughput, or blacklists to deny access to them. Graphs are generated to display information received from the CAPs and to monitor critical servers and services on the network.

Linking the various AC applications together is a web based GUI, facilitating configuration and also displaying monitoring variables for the C-WAN. Only authenticated users should be able to modify the configuration, however monitoring data may be viewed by all users within the C-WAN. The AC GUI should provide information regarding the status of the various CAPs in the C-WAN and display information gathered about the specific C-LANs. Using the ACs GUI, limits can be imposed on user, host and C-LAN (site) based Internet usage.

This chapter focused on the design requirements of the solution. Chapter 4 delves into the finer details of the CAPs' and the AC's implementation.

Chapter 4

Implementation

4.1 Introduction

Chapter 3 described the design requirements of the project solution. This chapter delves into the implementation decisions made to meet these requirements. The primary design goal was to integrate multiple network applications and tools together to aid in managing and monitoring the users use of bandwidth in a community network with a shared Internet connection.

The chapter begins with an overview of the implementation and highlights the most important points. This is followed by a brief look at the Operating System (OS) which was chosen, FreeBSD. FreeBSD provides a number of advantages to this system which are discussed further in Section 4.3. Revisiting the system design stipulated in Chapter 3, the project system is composed of a Community Access Point (CAP) and an Access Concentrator (AC). These components are discussed in detail including the integrated applications and tools chosen to meet the requirements. The Graphical User Interfaces (GUI) development of the CAP (CAPgui) and the AC (ACgui) is based on their design requirements which were discussed in Sections 3.4 and 3.6 respectively, and implementations of the GUIs is presented in Sections 4.4.2 and 4.5.5 respectively. An isolated testbed network was needed for the development process, and thus a discussion follows in Section 4.6 on a virtual network that was created within which the system was implemented and deployed, before physical deployment.

4.2 Implementation Overview

The implementation of this system was divided into two components, the AC and the CAP. The implementation phase of this project sought to develop the AC and CAP based on the specifications in Chapter 3. Initial testing and development was performed using a virtual

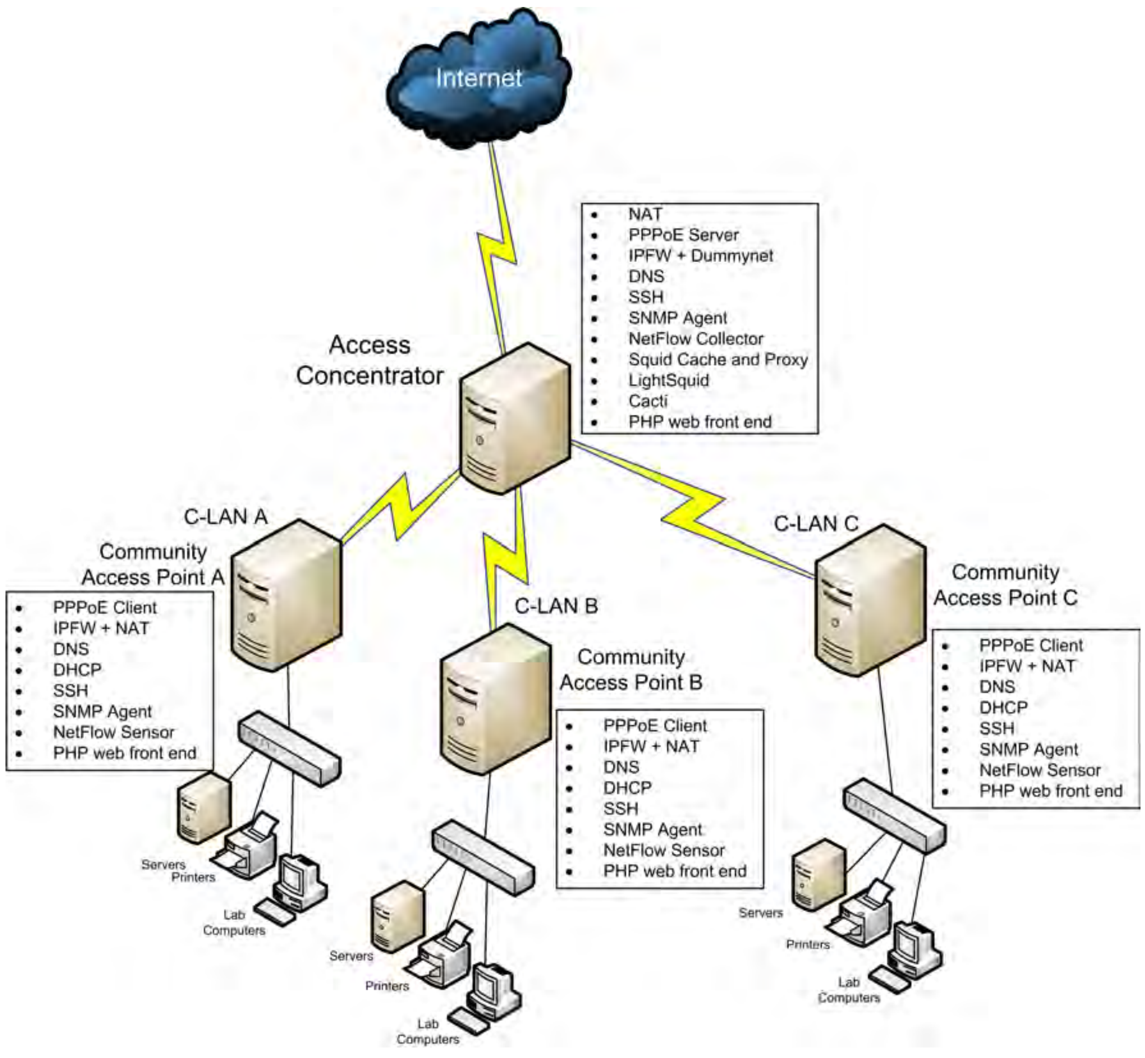


Figure 4.1: Overview of System Developed

network which is discussed in Section 4.6. This allowed for rapid turnaround cycles, and scalability testing without having to implement on a physical network or devices.

Figure 4.1 shows a typical community network configuration, based on the testbed networks, with the developed project system in place. The AC and the CAP have been developed using FreeBSD[1] as the base operating system (more detail in Section 4.3). Both devices have a number of Open Source Software (OSS) applications and tools installed in order to provide the functionality stipulated in the design. The AC and CAP have PHP[2] web front-ends, ACgui and CAPgui, to integrate the various applications, to facilitate system configuration and to display monitoring statistics. The applications and tools used in the project solution are highlighted in Figure 4.1 and are discussed in detail in this and subsequent sections.

The CAP is configured as the default router for the Community Local Area Network (C-LAN) at which it is situated. It also provides a PPPoE [58] tunnel to the AC to ensure encrypted and authenticated traffic over the Community Wide Area Network (C-WAN). PPPoE is detailed in Section 2.6. The CAP utilises IPFW as its firewall limiting IP traffic to only allow certain protocols. All outgoing traffic is passed through Network Address Translation (NAT) [97] which translates the IP address to that of the PPPoE tunnel. NAT is explained in further detail in Section 2.5. However, HTTP traffic is automatically forwarded by IPFW to the Squid proxy situated on the AC.

The CAP provides a forwarding and caching Domain Name System (DNS) server for the C-LAN, forwarding DNS requests to the AC. DNS lookups are cached, allowing the C-LAN to have improved response times for recent DNS lookups. The CAP provides a simple Dynamic Host Control Protocol (DHCP) server for providing clients with IP addresses in a particular IP range.

The CAP also collects monitoring data for the C-LAN, using an SNMP agent [33, 72] and a NetFlow collector [20,92]. The AC requests data from the SNMP agent on the CAP and also receives data from the NetFlow sensor, generating aggregated and summarised C-LAN Internet bandwidth usage data.

The CAP has a custom PHP web front-end (CAPgui) which facilitates initial and subsequent configuration. The CAPgui also provides a number of monitoring statistics about the system health of the CAP and status of upstream connectivity. The CAPgui can be accessed from any location within the C-WAN via the use of an Internet browser, to allow configuration changes to take place from any location. Monitoring statistics are available to any user, however, the user must be authenticated to make configuration changes.

[1]The FreeBSD Project - http://www.freebsd.org/

[2]PHP: Hypertext Preprocessor - http://www.php.net/

The AC, configured as the default router for the C-WAN, terminates the PPPoE tunnels from the various C-LANs in the community network by running a PPPoE service to which each CAP connects. As the community network will most likely have RFC 1918 [78] IP addresses (private IP addresses), which are not Internet routable, NAT [97] is configured on the AC to translate the internal IP addresses to an Internet routable address (the external IP on the AC). The AC has a number of firewall rules implemented, using an IPFW Firewall [4], to ensure only acceptable network traffic enters and leaves the C-WAN via the Internet connection. The IPFW firewall has been configured with Dummynet [79] pipes and queues, to enforce host and C-LAN (site) Internet usage quotas.

The AC provides a forwarding and caching DNS server to serve all name lookups for the C-WAN. DNS requests are forwarded from the AC to the Internet connection service provider's (ISP) DNS servers. With all the C-LANs requesting DNS lookups through a single point, a comprehensive cache of recent DNS entries can be stored. The cached DNS entries provide improved response times to all hosts within the C-WAN compared to retrieving the DNS entry from the ISP's DNS servers.

An SNMP agent is configured to gather monitoring data on the AC. The AC collects NetFlow data from its external network interface and the CAPs within the C-WAN for use within other applications.

The AC provides a number of useful applications, aiding in monitoring and managing bandwidth use, monitoring critical servers and services, and managing users throughout the C-WAN. Applications installed on the AC include a Squid Caching Proxy Server [96], LightSquid [23] and Cacti [107]. Squid provides a proxy server through which all Internet based traffic passes and caches regularly used web content. Direct connections to the Internet from within the C-WAN are not allowed and this has been enforced with appropriate IPFW rules to forward all HTTP traffic to Squid. Squid has also been configured to provide dynamic caching and user quotas using delay pools. Squid has been configured with the option of communicating with an Authentication, Authorisation and Accounting (AAA) server within the C-WAN to only allow authenticated users access to the Internet. LightSquid and Cacti are hosted on an Apache [106] web server on the AC. LightSquid utilises Squid log files to provide useful summaries of user Internet usage, popular websites and total Internet traffic. Cacti takes a number of inputs, including SNMP data, NetFlow data and Squid data to provide graphical representations of various C-WAN usage variables over time.

The AC has a PHP web based front-end, the ACgui. The ACgui can be accessed from any location within the C-WAN by using an Internet browser, and allows the viewing of monitoring statistics and modification of system and bandwidth management configuration. The statistics are available to any user, however, changes to configuration are only available to authenticated users. Limits on bandwidth use can be imposed per user, per host and/or per C-LAN (site).

Detailed information about how the CAP is configured is discussed in Section 4.4. The AC, and specifics about its configuration is described in Section 4.5. A rationale behind using FreeBSD as the base OS for the CAP and AC is presented in the next section.

4.3 Operating System - FreeBSD

FreeBSD provides a number of advantages which this project utilises to achieve its goals. "FreeBSD is a freely available Unix-like operating system, used widely by Internet service providers, in appliances and embedded systems, and anywhere that reliability on commodity hardware is paramount" [55]. FreeBSD is used in some of the most critical services and also visible Internet-oriented companies, including Yahoo!. Companies such as IBM, Nokia, Juniper and NetApp rely on FreeBSD for their embedded systems to provide mixed functionality. Darwin, the core OS of Apple Mac OS X, borrows heavily from FreeBSD, including the virtual file system, network stack and many other components [55].

An additional advantage of FreeBSD is the world wide community of developers which provide OSS applications which work well on this OS. Extensive documentation exists about FreeBSD [108] and the *ports* (software source code) collection [26], which can be built onto the OS. FreeBSD is reliable, extendable and adaptable in many situations and support structures are available for user assistance. There is an extensive knowledge base of FreeBSD expertise at Rhodes University, and this project has been built in conjunction and collaboration with this experience. The various advantages and support structures in place led to the use of FreeBSD as the base OS for the CAP and AC. The implemented system could potentially be migrated to a Linux OS with only minor modifications.

The CAPs and AC are running an installation of FreeBSD 6.3 [28], which only includes the functionality that is required (reducing the installation size) and is sufficiently lightweight to run on lightweight computers (low cost or low performance computers) with memory (primary and secondary) and processor limitations. For example the X Window System is not required to be installed as the devices are not required to have screens or monitors.

The next section describes how FreeBSD has been utilised and configured together with the 3rd party application software in the implementation of the CAP.

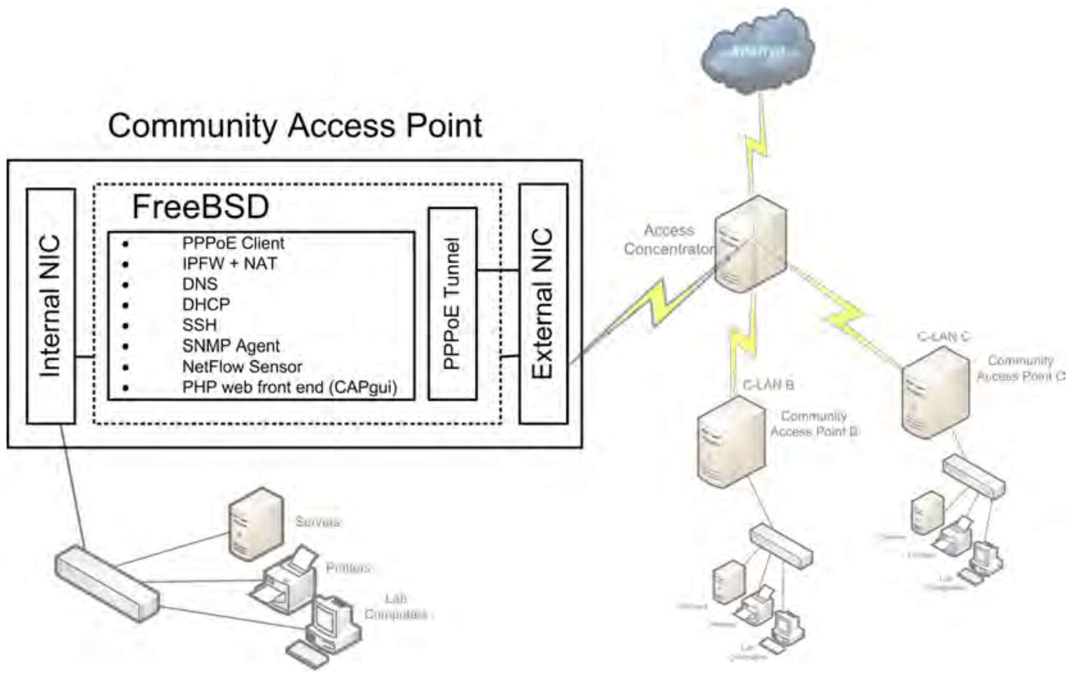


Figure 4.2: Community Access Point Implementation

4.4 Community Access Point (CAP)

Figure 4.2 shows a CAP and the services that it provides for its C-LAN. The CAP is the default router (or gateway) for the C-LAN at which it is situated. The CAPs default gateway is the PPPoE tunnel to the AC. Situated at the various C-LANs, the CAPs provide a secure connection to the AC and the Internet over the C-WAN. Monitoring is performed using an SNMP agent and a NetFlow sensor. SNMP monitoring data is collected by the AC and NetFlow data is transmitted to the NetFlow collector running on the AC, for aggregation and analysis. A PHP web based GUI, CAPgui allows system configuration and details the monitoring statistics that have been collected. An administrator can also use Secure SHell (SSH) [120] to connect to the CAP for a shell type environment. The CAPs network and application configuration is described in Section 4.4.1, and the CAPgui is discussed in Section 4.4.2.

4.4.1 CAP Network and Application Configuration

The CAP has two network interfaces. This can either be achieved by using two physical Network Interface Cards (NIC) or a single NIC with two IP aliases [27] which reside on separate Virtual LANs (VLANs). One network interface is connected to the C-WAN (external interface) and the other is connected to the C-LAN (internal interface). The CAPs external interface connects to a shared and an assumed to be insecure network, the C-WAN. The CAPs internal interface connects to the C-LAN via switching infrastructure to connect to the computers and servers within the C-LAN. The external interface is used to establish a PPPoE tunnel to the AC over the C-WAN, providing authenticated and encrypted network traffic over the C-WAN.

By using PPPoE to facilitate secure connections between the CAPs and the AC, it ensures that only authorised hosts may use the C-WAN and the services it provides. Secured connections are a necessary requirement when using a shared medium for the network link, as it prevents the potential of network abuse or attacks. Technologies which use shared mediums can include WiFi or WiMAX. WiFi has well documented exploits [37]: passive attacks, listening or eavesdropping on network traffic to obtain information; active attacks to gain unauthorised access or perform Denial of Service (DoS); Man-in-the-Middle attacks in which a legitimate user is *tricked* into talking to a *rogue* access point; and jamming attacks which involve introducing interference on the radio frequency. PPPoE can aid in preventing active, passive and man-in-the-middle attacks.

The CAP is configured as a forwarding DNS server using the application BIND [45] and all DNS lookups by the hosts within the C-LAN must use the CAP as their DNS server. DNS lookups are cached for a set period of time called the Time To Live (TTL) which is set by an authoritative DNS server for a specific DNS entry. If the DNS cache on the CAP does not have the entry, the request is forwarded to the AC which in turn will forward the request to the ISP's DNS servers. By providing a DNS server on the CAP, response times for regularly accessed websites can be improved, as DNS entries can be retrieved from the cache instead of requesting lookups from the upstream DNS servers.

The CAP can be configured as a DHCP server for the C-LAN, providing IP addresses to hosts within the C-LAN in a specific IP range. DHCP allows automatic configuration of IP addresses for hosts within the C-LAN, providing automatic TCP/IP configuration for hosts such as IP address, subnet mask, default gateway, DNS servers and Windows Internet Naming (WIN) servers. By providing this service hosts within the C-LAN do not require manual configuration for their TCP/IP settings. The configuration for DHCP on the CAP is modified by using the CAPgui.

SNMP agent software is installed on each CAP. Net-SNMP [72] was chosen as the SNMP agent for the CAPs as the installation comes pre-packaged with a number of MIBs for a wide variety of devices. Net-SNMP on the CAP provides a method to monitor system health; variables such as system uptime, secondary storage usage, RAM usage, CPU usage, load averages, services running and bytes in and out of the network interfaces. SNMP data is retrieved from the CAP by querying a number of MIBs related to the monitoring data required. For example, the MIB for system uptime is "HOST-RESOURCES-MIB::hrSystemUptime.0" [115]. In another example, to discover how many bytes have passed through any particular network interface, we look at "IF-MIB::ifInOctets" for the number of bytes that have been received or "IF-MIB::ifOutOctets" for the number of bytes transmitted [25]. Appendix A identifies some useful OIDs that have been monitored via SNMP within this system.

The AC requests SNMP data from the CAP and creates status summaries and graphs. As the CAP is the default router for the C-LAN and all packets leaving or entering the network will route through it, the SNMP agent will collect data regarding the network traffic that passes through the C-LAN interfaces (from within the local C-LAN and the community wide C-WAN). However, the SNMP agent can only collect information on the host on which it is installed.An SNMP agent cannot provide details regarding which hosts are using the network and what they are using the network for. For example, the SNMP agent will provide information suggesting that the external network interface, of the host on which it is running, is at 90% capacity. In this example, SNMP does not provide information regarding the origin of the traffic, be it the local host or another host on the local network. To provide finer grain network utilisation data, NetFlow [20] is used.

The CAP has Softflowd [92] installed, a software implementation of NetFlow to monitor flows within the C-LAN. Softflowd is configured to listen on the PPPoE tunnel interface and to transmit the summarised flow data to the AC, which has a NetFlow collector, via UDP flow exports on a specific port.

As the CAP is not required to have a screen or monitor to access it, OpenSSH [120] is used to provide a shell environment via Secure SHell (SSH). However, SSH should only be used by administrators who are troubleshooting problems with the CAP, upgrading applications or installing new applications. For this reason the login process is secured using cryptographic keys rather than plain-text passwords. For typical use, the CAPgui is provided for configuration and viewing of monitoring data and this is discussed in the subsequent section.

The CAP is configured with IPFW firewall. IPFW has been configured with an *inclusive* firewall ruleset, only allowing traffic which matches the rules and with a default to deny all other incoming and outgoing traffic. The C-LANs are only required to be able to access the WWW, via HTTP and HTTPS. HTTP traffic is automatically forwarded by IPFW to the Squid proxy server situated on the AC. DNS traffic is allowed on the internal interface (from the C-LAN) to the CAP, where the DNS requests will be forwarded upstream.

ICMP and SSH are allowed in and out on any interface. Traffic destined to the CAP from the AC for the SNMP port is allowed. Softflowd transmits UDP packets on a specified port to the AC and this is allowed. NAT is implemented by using an IPFW *divert* rule and all outgoing traffic has its IP address translated to that of the PPPoE tunnel interface.

The IPFW firewall rules prevent unencrypted traffic from being directly transmitted using the external network interface and packets are routed over the PPPoE tunnel carried by the external interface. However, SSH and ICMP are allowed over the external interface for troubleshooting purposes.

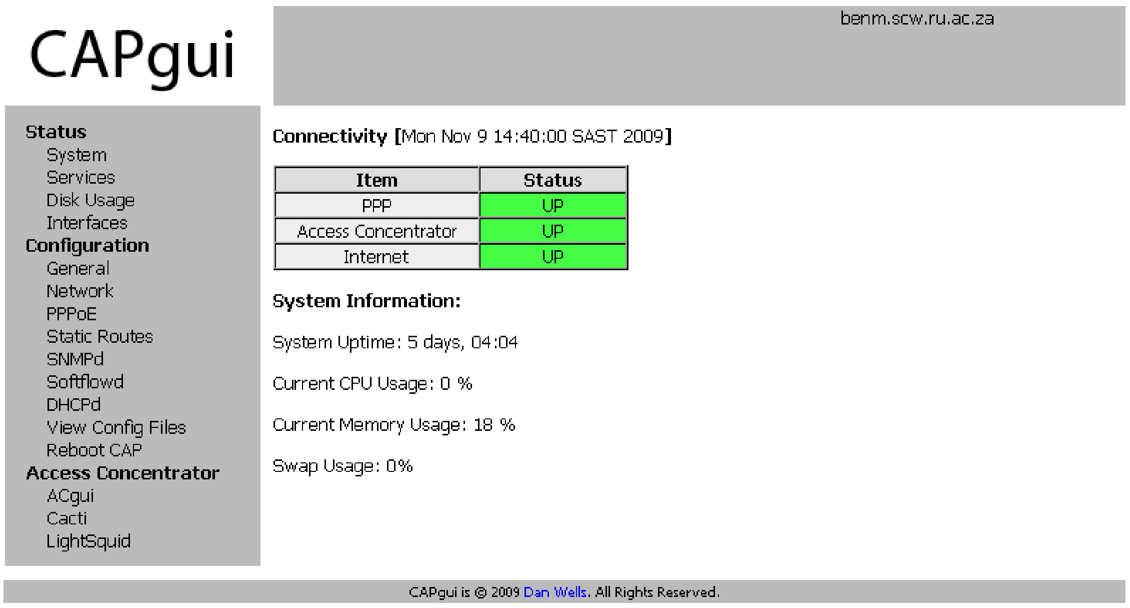
4.4.2 CAP Graphical User Interface (CAPgui)

A PHP [109] based web front-end is provided for initial and subsequent configurations of the CAP and it has been named CAPgui. The CAPgui also provides the user with connectivity and system health variables to monitor the status of upstream links and the CAP.

The CAPgui is hosted on the CAP using the Lighttpd [53] web server. Lighttpd was chosen over other web server implementations as it is designed and optimised for any server (or PC) that is either already under heavy load or has limited memory or processing ability [53]. As the CAPs are deployed to low cost or low performance machines, using an application which is efficient and has a small memory footprint is advantageous.

The CAPgui is required to display dynamic content as the content is dependent on monitoring information and configuration pages. As the content is not static the pages are generated by server-side PHP running on the CAP. The PHP generates Extensible HyperText Markup Language (XHTML) [113] pages for the Internet Browser to display. XHTML is an application of Extensible Markup Language (XML) [42], which requires the document to be well-formed and able to be parsed by a standard XML parser. Cascading Style Sheets (CSS) [114] have been used to separate the presentation of the page from the information contained in the XHTML. Configuration changes within the CAPgui modify a single XML file. The XML file is parsed, when the CAP is rebooted via the CAPgui, to generate and update FreeBSD configuration files for various system settings and service configuration. More detail about how these configuration files were generated is described in Appendix B.1. The CAPgui User Manual, source code, background scripts and screenshots of all the pages can be viewed in Appendix C.

Figure 4.3 demonstrates the CAPgui which is the front-end for the CAP. This particular CAP is deployed in the Settler City Wireless (SCW) testbed network. Figure 4.3 shows the first page seen when visiting the CAPgui, the *System Status* page. In the top left of the figure, a logo displaying *CAPgui* informs the user where they are. The top-right of the CAPgui provides the hostname of the CAP, in this example *benm.scw.ru.ac.za*. The menu (on the left-hand side of the page) is separated according to content that can be viewed (*Status*), configuration modification (*Configuration*) and external links to the *Access Concentrator*. Items listed beneath *Configuration* are only visible to the user once they have been authenticated, if they are not authenticated, the only item in this list is



Connectivity [Mon Nov 9 14:40:00 SAST 2009]

Item	Status
PPP	UP
Access Concentrator	UP
Internet	UP

System Information:

System Uptime: 5 days, 04:04

Current CPU Usage: 0 %

Current Memory Usage: 18 %

Swap Usage: 0%

CAPgui is © 2009 Dan Wells. All Rights Reserved.

Figure 4.3: CAPgui - System Page

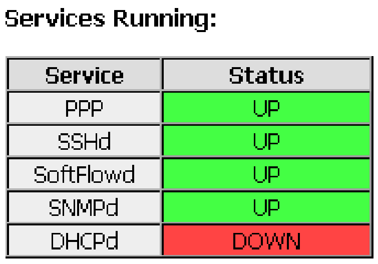
Log In. The central area of the CAPgui interface displays the content that is relevant to the specific page that the user is currently viewing. Lastly, the bottom of the page provides a footer to denote the bottom of the page.

The items under *Status* are *System*, *Services*, *Disk Usage* and *Interfaces*. These pages display system health variables for the CAP. The various *Status* pages display information from log files which are generated by PHP scripts that run at five minute intervals (unless otherwise specified) on the CAP. By separating the gathering of status information from displaying it, the *Status* pages have improved loading times as they do not have to wait for information to be gathered.

The *System Status* page, seen in Figure 4.3, provides upstream 'Connectivity' status, displaying information about the status of upstream links. If any items within this list are *down* they are highlighted in the colour red, otherwise if they are *up* they are highlighted in the colour green. The connectivity table relies on a background *bash* [35] script which checks the status of the PPP process and uses *ping* [67] to test upstream network hosts every five minutes.

Also in the *System Status* page, 'System Information' is displayed showing 'System Uptime', 'Current CPU Usage', 'Current Memory Usage' and 'Swap Usage'. These variables, specifically CPU and memory usage, are constantly changing and are obtained every time the page is visited or refreshed. The 'System Information' is obtained by using PHP scripts which execute specific commands on the AC to obtain the necessary information.

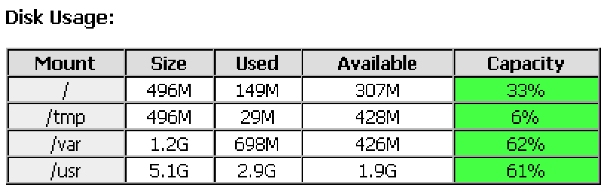
The *System Status* page is the first point of reference for administrators wishing to troubleshoot network issues or check the status of the CAP. If any of the 'Connectivity'



Services Running:

Service	Status
PPP	UP
SSHd	UP
SoftFlowd	UP
SNMPd	UP
DHCPd	DOWN

Figure 4.4: CAPgui - Service Status Page



Disk Usage:

Mount	Size	Used	Available	Capacity
/	496M	149M	307M	33%
/tmp	496M	29M	428M	6%
/var	1.2G	698M	426M	62%
/usr	5.1G	2.9G	1.9G	61%

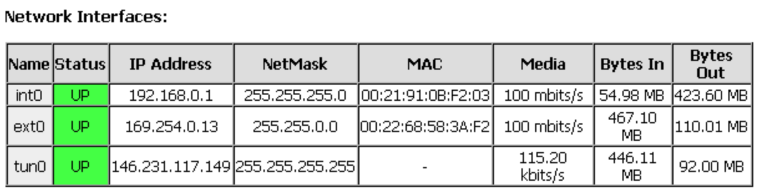
Figure 4.5: CAPgui - Disk Usage Status Page

items are *down*, administrators can take the necessary steps to rectify the root cause of the problem. A flowchart was produced for administrators (or network support staff) to follow in the case of no or limited access to the Internet from their C-LAN, see Appendix C.

The *Services Status* page, highlights various important network services, such as PPP, SSHd, Softflowd, SNMPd and DHCPd in a table. Figure 4.4 displays such a table. In this example, all the services are *up* (and highlighted in the colour green) except for DHCPd which is *down* (highlighted in the colour red). The service status information is gathered by using the SNMP service on the CAP (see Appendix A.1 for details of the MIBs requested). However, if SNMP is unavailable then the services are checked by using a PHP method which executes *ps* (process status) [31] on the CAP. In this case, SNMPd would be marked as down.

The *Disk Usage Status* page provides a table with the FreeBSD mount points and per disk partition the total size, the amount used, amount available and percentage used are displayed. An example can be seen in Figure 4.5. In this example, the total disk size was 8 GB. The *Disk Usage Status* page allows administrators to resolve issues such as running out of disk space on the CAP. The table relies on a PHP method which executes *df* [30] on the CAP to provide details about each partition and their related usage status.

The last of the *Status* pages, the *Interfaces* page, provides a table of the network interfaces with various fields relating to their configuration and monitored statistics. In Figure 4.6, three network interfaces are present: ‘int0’, the internal interface; ‘ext0’, the external interface; and ‘tun0’, the PPPoE tunnel interface to the AC. The interface table provides information for each one’s status (*UP* or *DOWN*), IP Address, Netmask, MAC



Network Interfaces:

Name	Status	IP Address	NetMask	MAC	Media	Bytes In	Bytes Out
int0	UP	192.168.0.1	255.255.255.0	00:21:91:0B:F2:03	100 mbits/s	54.98 MB	423.60 MB
ext0	UP	169.254.0.13	255.255.0.0	00:22:68:58:3A:F2	100 mbits/s	467.10 MB	110.01 MB
tun0	UP	146.231.117.149	255.255.255.255	-	115.20 kbits/s	446.11 MB	92.00 MB

Figure 4.6: CAPgui - Interfaces Status Page

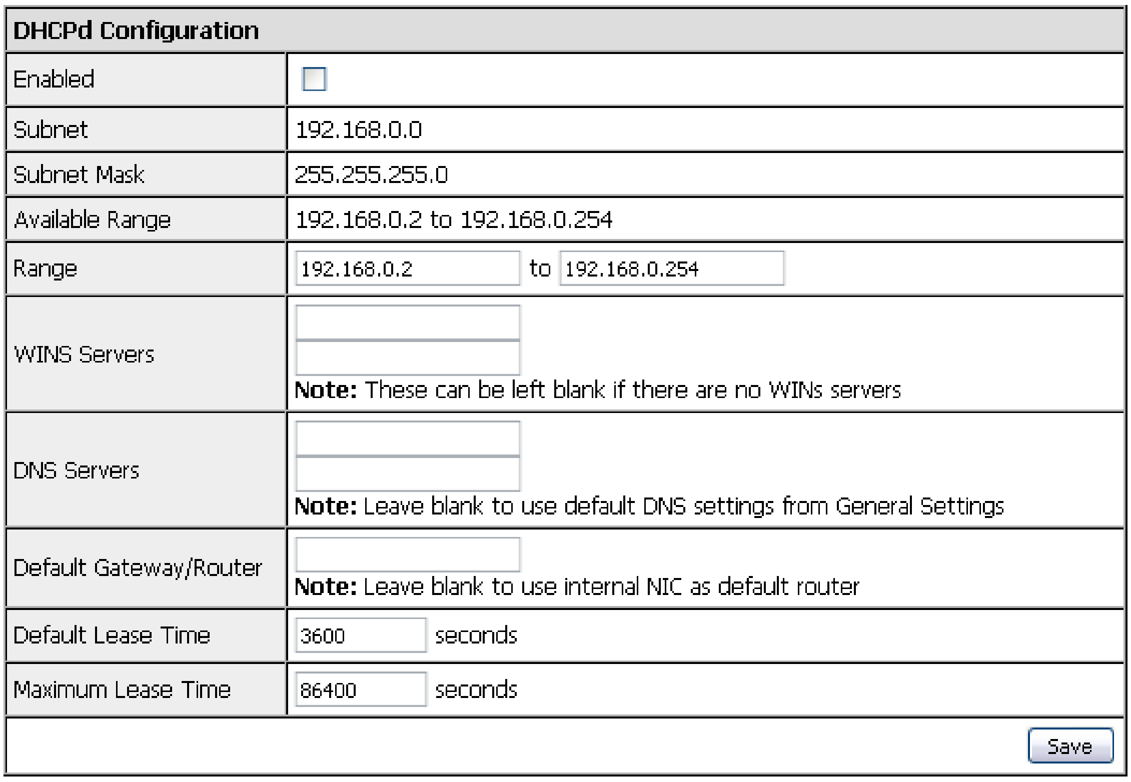
```
function is_hostname($hostname)
{
  if (!is_string($hostname))
    return false;

  if (preg_match("/^([_a-z0-9\-]+\.?)+$/i", $hostname))
    return true;
  else
    return false;
}
```

Figure 4.7: Perl Regular Expression in PHP to Check a Valid Hostname

Address, Media Type, Bytes In and Bytes Out. An administrator can visit this page to view the network configuration of the CAP, see what type of network the interfaces are connected to (Media) and view the number of bytes transferred in and out of the network interfaces. The interfaces table obtains its data using SNMP on the CAP (see Appendix A.1 for details of the MIBs requested).

Following the *Status* pages are the *Configuration* pages. The *Configuration* pages allow the modification of system and service settings, and therefore are only available to a user who has been authenticated. As the pages rely on input by the user to update configuration, all input is validated for the specific field that is being updated. Input is validated using a variety of methods, but the majority of the methods rely on custom Perl Regular Expressions [32] in PHP. For an example of such a Perl Regular Expression, namely one checking whether a hostname is correct, see Figure 4.7. By validating the input, checks can be put in place to make sure the CAP is not misconfigured. If the input fails the validation check, the user is presented with a description of what type of input is expected and the XML configuration is not updated. When the input has been validated, the user is shown a list of changes, the XML configuration file is updated with the new settings and new configuration files relevant to that page are generated from the XML.

Configuration pages are available for *General*, *Network*, *PPPoE*, *Static Routes*, *SNMPd*, *Softflowd* and *DHCPd*. The *View Config Files* page allows the administrator to view the configuration files that have been generated or updated by the CAPgui. Additionally, the *Reboot CAP* page allows the CAP to be rebooted to load the new configuration files.



DHCPd Configuration	
Enabled	☐
Subnet	192.168.0.0
Subnet Mask	255.255.255.0
Available Range	192.168.0.2 to 192.168.0.254
Range	192.168.0.2 to 192.168.0.254
WINS Servers	**Note:** These can be left blank if there are no WINs servers
DNS Servers	**Note:** Leave blank to use default DNS settings from General Settings
Default Gateway/Router	**Note:** Leave blank to use internal NIC as default router
Default Lease Time	3600 seconds
Maximum Lease Time	86400 seconds
	Save

Figure 4.8: CAPgui - DHCPd Configuration

For an added level of security the user has to confirm their username and password to reboot the CAP.

In Figure 4.8, the *Configuration* page for *DHCPd* is presented. It displays the common layout of all *Configuration* pages, with fields for the relevant configuration in a table layout and a 'Save' button at the bottom right.

Under the *Access Concentrator* heading, there are links to the *ACgui*, *Cacti* and *LightSquid*. Visiting the ACgui provides more detailed information about the status of the C-WAN, including the statuses of all the CAPs within the network. Cacti is a graphing application which displays graphs for all devices in the C-WAN, LightSquid provides detailed reports on all Internet access by the C-WAN via the Squid proxy.

The CAPs connect via PPPoE tunnels to the AC which provides the C-LANs with connectivity to the rest of the C-WAN or the Internet. The AC is discussed in detail in the next section.

4.5 Access Concentrator (AC)

This section discusses the implementation of the AC. First, an overview of the AC is presented in Section 4.5.1 describing the various parts of the implementation. Section

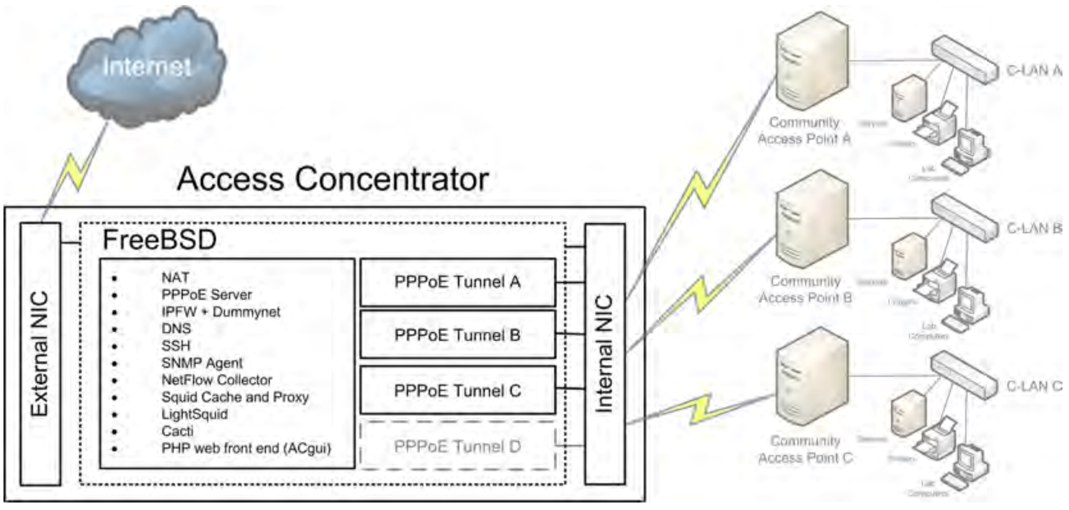


Figure 4.9: Access Concentrator Implementation

4.5.2 discusses the networking and routing related configuration. In Section 4.5.3 the applications hosted on the AC are detailed. Section 4.5.4 discusses the methods utilised for managing bandwidth by imposing Internet usage quotas in the C-WAN. Lastly, Section 4.5.5 presents the PHP web based front-end of the AC, the ACgui, and how it was implemented.

4.5.1 Overview of the Access Concentrator

The AC provides two main functions; to route traffic securely throughout the C-WAN and to the Internet, and to perform a number of network monitoring, management, logging and accounting tasks. Figure 4.9 shows the AC and the services that have been implemented. The AC is the default router for the CAPs and all traffic originating from or destined to the C-LANs pass through the AC. Secure and authenticated connections are facilitated between the AC and the CAPs by using PPPoE. As the C-WAN could use an RFC 1918 [78] address space for internal IP addresses, these are translated to an Internet routable IP, by using NAT. Internet in and out bound traffic is restricted by the use of an IPFW firewall [4]. Monitoring of the AC is performed using an SNMP agent. Aggregated flow data is collected from the CAPs and AC with the use of a NetFlow collector on the AC.

Three applications, which were pertinent in achieving the goals of this project, are the Squid Caching Proxy Server, LightSquid and Cacti. The Squid proxy caches regularly accessed web content and can therefore reduce the web content that needs to be retrieved from the Internet by serving it from the cache, reducing bandwidth use. Necessary IPFW firewall directives are in place to prevent direct connections to the Internet, forwarding all Internet bound connections to pass through the proxy server. User based Internet quotas were implemented by summarising the Squid proxy log files and utilising delay pools in Squid. Host based quotas are implemented by summarising NetFlow data received from the CAPs and AC, and utilising Dummynet [79] in IPFW. LightSquid processes the Squid proxy log files and presents neat summaries of Internet usage per user, usage per month, usage per year and popular web pages (calculated using the number of hits or number of bytes transferred). A graphing solution, Cacti, retrieves monitoring information from multiple sources and generates graphs. Cacti generates graphs displaying information about the AC and the CAPs within the C-WAN. Information that is graphed by Cacti includes secondary storage; traffic in and out of the various network interfaces; CPU load; load averages; memory usage; and various graphs related to the Squid Proxy and Cache.

Monitoring data that has been collected is summarised and displayed on the ACgui. Using this information, administrators can impose user, host and C-LAN (site) Internet usage quotas.

Tying the network configuration, bandwidth management and other applications together is the ACgui. It provides a single portal for administrators to view monitoring variables and modify system configuration. The front-end displays a number of monitoring variables: including the statuses of the CAPs, and their related services, located within the C-WAN; the status of the Squid proxy; a list of the popularly accessed websites; C-LAN (site) and host quotas; and user quotas.

Administrators can modify user, host and C-LAN (site) based Internet quotas; add or remove web content to blacklist; add or remove CAPs from the network; and various other system and service configuration. The front-end provides a number of services to aid in bandwidth monitoring and management for an administrator.

4.5.2 Router/Gateway Networking Configuration

The AC has two network interfaces. This can either be achieved by using two physical NICs or a single NIC with two IP aliases [27] which reside on separate Virtual LANs (VLANs). From this point, the network interfaces will be referred to as the *internal* and *external* network interfaces.

The internal network interface communicates with the C-WAN and the external network interface allows communication with the Internet. The AC controls the way packets are routed between the CAPs and the AC.

The internal network interface facilitates the PPPoE tunnels with the CAPs. The unencrypted C-WAN network uses an IP address range of 169.254.0.0/16: this is to accommodate native windows clients on the network as its the default IP range that windows machines assign when there is no DHCP server present or no fixed IP configured [43]. The PPPoE service is provided by the PPPoEd [93] daemon and configured to only allow authorised CAPs to connect to the AC. When a CAP attempts to establish a PPPoE connection over the C-WAN, the AC confirms that the CAP is in the list of subscribers and the given password is correct. The CAP and the AC create a PPPoE tunnel binding an IP address provided by the AC. The AC then creates a network route to the CAPs C-LAN via the tunnel interface's IP address to ensure traffic destined for the C-LAN is routed correctly. When the PPPoE link from the AC to the CAP terminates, the route to the C-LAN is removed and the tunnel interface on the AC is removed.

The community network could use an RFC 1918 [78] IP address space. These private IP ranges are not Internet routable, as packets with a source or destination in an RFC 1918 address space are often discarded by routers, unless the router is providing a NAT service for that IP range. NAT translates an outbound packet's IP address to an Internet routable IP address. When inbound IP packets are received from outside the C-WAN they are translated, by the AC, to internal IP addresses to find the host that initiated the connection. NAT is provided on the AC by using the NATd daemon [29] in conjunction with *divert* rules in the IPFW firewall [4].

The AC has been configured as a forwarding and caching DNS server similarly to the configuration of the CAP. It differs in that it forwards DNS requests to the ISP's DNS servers. By using a central caching DNS server, requests from all hosts within the C-WAN are cached, allowing improved response times for DNS name lookups for regularly accessed web content.

The AC is not required to have a screen or monitor to access it, therefore OpenSSH [120] is configured to allow remote logins to a shell environment using cryptographic keys to encrypt connections. For general purpose configuration and interaction with the AC, the ACgui should be used, however when more complex problems or requirements arise, such as needing to upgrade applications or troubleshoot network problems, then SSH access could be used.

IPFW firewall on the AC limits any incoming and outgoing connections from the C-WAN by only allowing connections using certain ports. The ruleset is an *inclusive* one, which allows connections to a set range of services with the default to deny for any service which is not specified in the rules. Services which are allowed out include ICMP, HTTPS, SSH, SMTP and FTP. DNS is only allowed out by the AC, as it is running a DNS server. HTTP traffic is also only allowed out by the AC, as all C-WAN HTTP traffic is forwarded by the Squid proxy server.

An SNMP agent is installed and configured on the AC similarly to that of the CAP. Net-SNMP is used as the SNMP agent, which provides a method to monitor critical information about the health of the AC, network interfaces and local services status.

This section discussed the networking configuration of the AC and how the various network applications have been configured. The next section discusses applications which have been installed and configured to provide a C-WAN with management, monitoring, logging and accounting.

4.5.3 Applications and Configuration

Three applications, pivotal to the goals of the project, have been installed and configured on the AC, the Squid Caching Proxy Server, LightSquid and Cacti. These applications, why they were selected, their associated configuration and their implications are discussed in the following sections (Sections 4.5.3.1 - 4.5.3.3). LightSquid and Cacti are hosted on the AC via an Apache [106] web server. Apache was chosen as the web server for the AC as it is widely used, well maintained, secure and has a large community providing online support. Squid, LightSquid and Cacti are configured to achieve effective management and monitoring of the network.

4.5.3.1 Squid Caching Proxy Server

Squid [96] was chosen for use in this system as it is a widely used, extensively documented and customisable application (see Section 2.8). Squid has been configured to provide a number of functions specific to this project which are discussed in this section. The functionality, which Squid provides, includes caching regularly accessed web content; ensuring authenticated and authorised users have access to the Internet; limiting Internet bound connections to specific ports; collecting statistics on Internet usage by extensive logging; providing delay pools to limit users' Internet throughput and limit throughput to specific web content; caching dynamic web content; providing SNMP monitoring for its status; and peering with other caches.

Hosts within the C-WAN cannot directly access the Internet and all their connections are channeled to pass through the proxy server. This is enforced with forwarding firewall rules on the AC. Squid, within this system, can either be configured as a transparent proxy which automatically proxies Internet bound connections or as a proxy which requires client authentication and client browser configuration. However, configuring Squid as a transparent proxy has its trade-offs. An advantage is that clients will not require any configuration of their web browsers to access the Internet as requests are transparently received and proxied to the Internet. A disadvantage is that authentication of users is not possible when using Squid as a transparent proxy.

If Squid is not configured as a transparent proxy, authentication of users can be enabled to only allow authorised users access to the Internet. This system has utilised proxy authentication in conjunction with a FreeRADIUS server (for more information in regards to the configuration of Squid and FreeRADIUS see Appendix B.2.1). Using user authentication with Squid provides a number of benefits; more detailed reports can be generated in terms of how specific users are using the Internet; greater control over bandwidth use (per user) can be achieved; and stricter control over who has authorisation to access the Internet is allowed. Users would have a community wide username and password which would allow them access to the Internet at any of the C-LANs and this is achieved by using a federated authentication scheme for the community network. Squid is configured to provide a cache in conjunction with its proxy.

The Squid cache stores regularly retrieved web content. If a client request is received for an item which is stored in the cache, it is served from the cache instead of retrieving the item from the Internet a second time. Squid has been configured to cache dynamic content, by rewriting URLs that are stored in the cache of identical content, hosted off multiple web servers. This frees up cache space for more content to be stored. This has been achieved by using a store URL rewrite helper configured by the use of multiple Access Control Lists (ACL) which refer to a number of destination domains (or match regular expressions) which use multiple web servers. This list of web servers includes popularly used services which are hosted on multiple web servers such as Google, Google Maps, YouTube and Facebook. All access to the Squid proxy and cache is logged for further analysis.

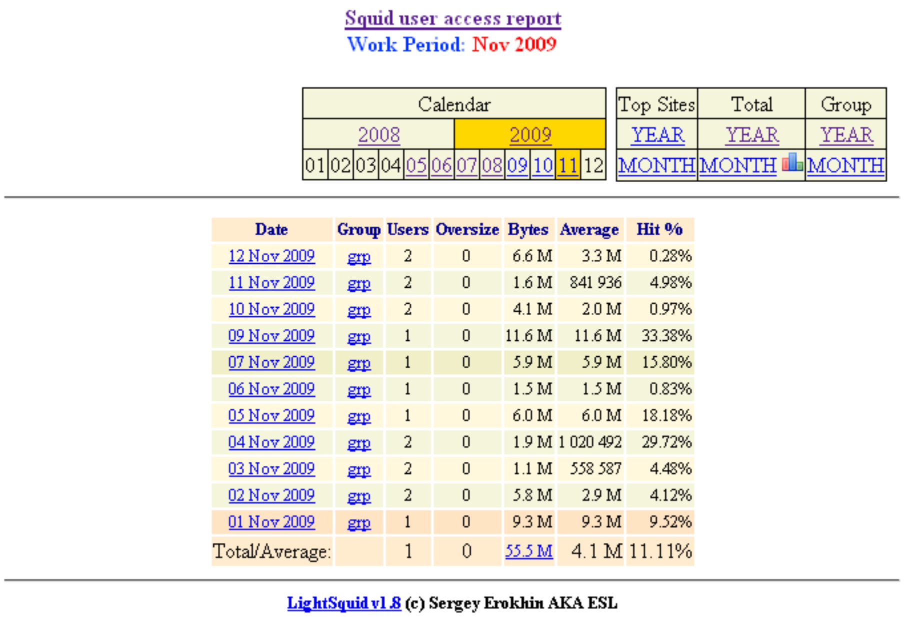
The Squid proxy logs every connection that passes through it, saving information such as; the web content that was retrieved; the date and time of the transaction; whether the web content was in the Squid cache, a cache *HIT* or *MISS*; the IP address for the host that requested the content; username (if user authentication required); and the number of bytes transferred to retrieve the content. By processing the Squid proxy access log, the Internet usage per user can be calculated.

Squid has been configured with delay pools to impose throughput limits on the C-WAN users' Internet usage. These delay pools form the basis for user based Internet quotas, and are discussed in more detail in Section 4.5.4. In conjunction with limits imposed on the quantity of Internet bandwidth used the C-WAN users can also be limited to the extent of type of content they can view.

Blacklists are lists of specific Internet content that administrators do not want their users to have access to. Blacklists can include specific web pages, domains (ie: ".facebook.com"; or ".co.za"), IP ranges, web page sizes, file types, web server types, MIME types[3], Autonomous System Numbers (ASN)[4] or regular expressions. An example of content which can be blocked is advertising content. Advertising can sometimes be un-

[3]MIME Media Types - http://www.iana.org/assignments/media-types/

[4]Autonomous System (AS) Numbers - http://www.iana.org/assignments/as-numbers/as-numbers.xml


Squid user access report
Work Period: Nov 2009

Calendar		Top Sites	Total	Group
2008	2009	YEAR	YEAR	YEAR
01 02 03 04 05 06	07 08 09 10 11 12	MONTH	MONTH	MONTH

Date	Group	Users	Oversize	Bytes	Average	Hit %
12 Nov 2009	grp	2	0	6.6 M	3.3 M	0.28%
11 Nov 2009	grp	2	0	1.6 M	841 936	4.98%
10 Nov 2009	grp	2	0	4.1 M	2.0 M	0.97%
09 Nov 2009	grp	1	0	11.6 M	11.6 M	33.38%
07 Nov 2009	grp	1	0	5.9 M	5.9 M	15.80%
06 Nov 2009	grp	1	0	1.5 M	1.5 M	0.83%
05 Nov 2009	grp	1	0	6.0 M	6.0 M	18.18%
04 Nov 2009	grp	2	0	1.9 M	1 020 492	29.72%
03 Nov 2009	grp	2	0	1.1 M	558 587	4.48%
02 Nov 2009	grp	2	0	5.8 M	2.9 M	4.12%
01 Nov 2009	grp	1	0	9.3 M	9.3 M	9.52%
Total/Average:		1	0	55.5 M	4.1 M	11.11%

LightSquid v1.8 (c) Sergey Erokhin AKA ESL

Figure 4.10: LightSquid - Overall Access Report

wanted with browser pop-ups and flashing images on web pages. By blocking a known list of advertisement web servers, bandwidth use while browsing can decrease and web page loading times can improve. More information about how Squid ACLs have been configured to deny access to these blacklists can be seen in Appendix B.2.3. These lists can be modified manually via the ACgui (see Section 4.5.5) or updated using ready-made lists that are freely available on the Internet [85].

As Squid logs every transaction which passes through it and generates a comprehensive access log file for these transactions, this data requires summarisation for analysis. By analysing the log files Internet bandwidth use information can be gathered. An application which summarises these log files is LightSquid, it provides various views of the Squid log files.

4.5.3.2 LightSquid

LightSquid [23] parses the Squid access log file to summarise data into different views and display them via a web front-end. Figure 4.10 displays a view of LightSquid's web front-end. In this figure, an overall summary is provided showing information for the current month and displaying data such as total data transferred, average data transferred per day and the percentage of data that was served from the Squid cache (a cache *HIT*). Summaries can also be viewed for previous years and months. From the overall summaries LightSquid provides the option to *drill-down* to find more information. The options to view reports organised per user are provided, showing information such as the total data transferred, the websites they have visited and the times that they accessed those websites.

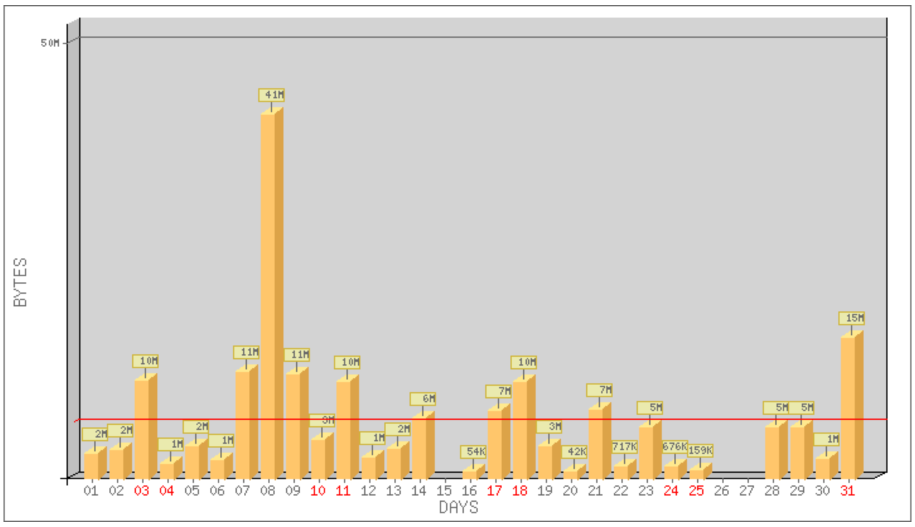


Figure 4.11: LightSquid - Per Month Internet Usage Graph

LightSquid also provides a list of the most popular websites, viewed by all the users, sorted either by the most data transferred or the most viewed site (connects to that website).

LightSquid provides a simple graph displaying Internet usage per month, either for all users or per user. Figure 4.11 shows an Internet usage bar graph for the month of October in 2009 for all users, the y-axis displays the amount of bytes transferred and the x-axis displays each day in the month. Data is displayed per day showing how many bytes were transferred in that day. The mean is displayed by a horizontal line, in this example, at approximately 4 MB.

Using the extensive information about Internet usage via the Squid proxy, the popularly visited websites that have been identified by LightSquid can be placed into a delay pool by an administrator, to limit access to that website or be placed in a blacklist to prevent all access.

LightSquid provides a single type of graph for Internet usage per month or per user utilising the Squid access logs. Cacti is a graphing application which takes in many input data sources and displays graphs for that data over various periods and is able to produce many types of graphs. Cacti and its implementation on the AC is described in the next section.

4.5.3.3 Cacti

Cacti [107] is a graphing application used to graphically display data received from the SNMP agents on the CAPs and AC. An example of a graph generated by RRDTool and Cacti can be seen in Figure 4.12 which is graphing 'HTTP Service Times' for the Squid proxy. This graph is measuring the duration of time that it took to perform a request. The graph displays *Hits*, when the object was found in the cache; *Misses*, when the object

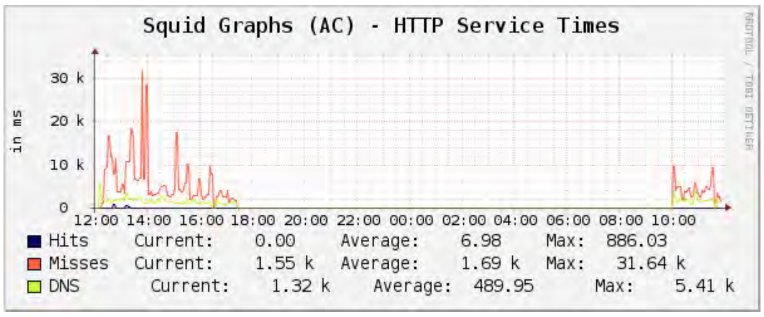


Figure 4.12: Cacti - HTTP Service Times Graph

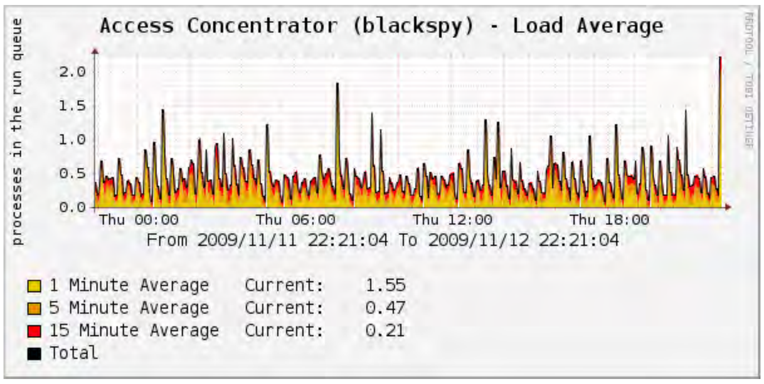


Figure 4.13: Cacti - Load Averages Graph

was not in the cache and had to be retrieved from the Internet; and *DNS* lookups.

Cacti provides the means to create templates for specific types of devices and two host templates have been implemented for Cacti. The templates are for the AC/CAP and for the Squid proxy. The AC/CAP host graph template will poll the devices and create graphs for available disk space; retrieve information about the network interfaces and create graphs for them displaying bytes inbound and outbound; CPU usage; load averages (see example in Figure 4.13); and memory (RAM or primary memory) usage. The Squid proxy host graph template will poll the Squid proxy for data and create graphs on the state of the cache (total size, amount used and number of objects); file descriptors (reserved and remaining); HTTP requests (the amount of requests and how many hits in the cache); HTTP Service Times (discussed earlier and seen in Figure 4.12); and HTTP Data (data received, data sent, data saved and percentage of data saved).

By utilising summarised data and graphs, limits on throughput and bandwidth use can be imposed by various methods discussed in the next section.

4.5.4 Bandwidth Management

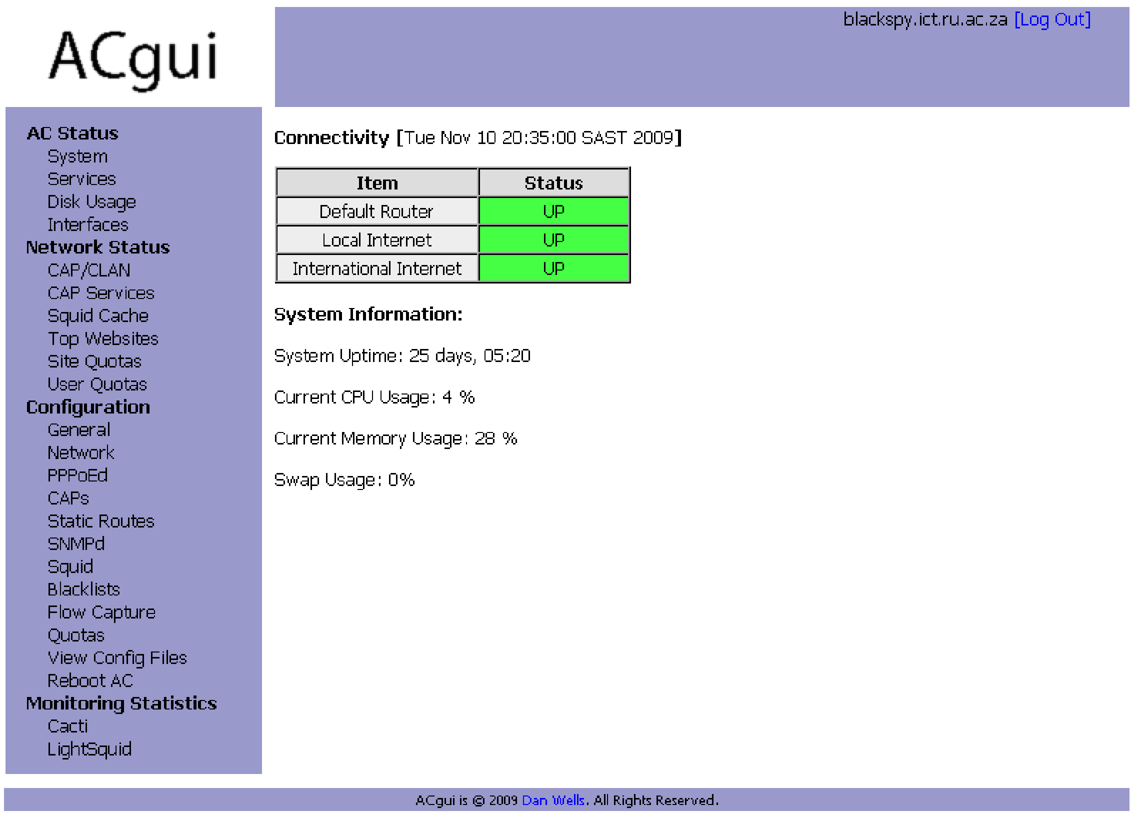
The Squid proxy server provides a comprehensive access log which can be processed to summarise Internet usage per user. By calculating Internet usage per user for a specified period, when a specified threshold of Internet usage is reached, users are automatically placed into a delay pool (limiting their throughput) until their Internet usage has decreased. The AC has been configured with four types of delay pools that a user can be in and these have been configured by the necessary ACLs in the Squid configuration (see Appendix B.2.2 for a detailed configuration). The pools that have been configured are *no delay*, *slight delay*, *significant delay* and *no access*. For example, when a user is in the *no delay* pool, their Internet throughput is not restricted, but when they have exceeded the threshold set on the *no delay* pool, they move into the *slight delay* pool. The per user Internet usage is calculated over a particular period of days using a sliding window, or total bandwidth used per month. A user is allocated a set amount of bandwidth they can utilise within that period. If a user uses all their pre-allocated bandwidth, their Internet access is denied by moving them into the *no access* pool. The thresholds and periods at which these delays come into effect can be set via the ACgui (see Section 4.5.5).

Data received from the NetFlow sensors running on the AC and CAPs are collected and aggregated by the AC to provide accounting of how the hosts and C-LANs are using the Internet connection. The collected NetFlow data is manipulated to provide a list of network hosts and C-LANs with their Internet usage. The NetFlow data is used in conjunction with IPFW and Dummynet on the AC to limit Internet usage originating from a particular host or C-LAN. As with user based delay pools, hosts have delay pools as well, although implemented as IPFW tables. Four tables have been configured for host and C-LAN (site) based quotas; they are *host slight delay*, *host significant delay*, *site slight delay* and *site significant delay*. Dummynet pipes and queues have been implemented which effect the hosts or sites (C-LAN subnets) according to the specified delay. These delays work in a similar fashion to user based quotas, however they can be configured with different thresholds and periods. Thresholds and periods for which the hosts and C-LANs fall into these delays can be configured via the ACgui (see Section 4.5.5).

The ACgui provides comprehensive summaries of user, host and site based Internet usage. By viewing these usage summaries, an administrator can allocate acceptable levels of bandwidth for each type of quota. The ACgui is discussed in the next section.

4.5.5 Access Concentrator Graphical User Interface (ACgui)

The AC has a PHP based web front-end, named the ACgui, which provides a C-WAN with a number of monitoring and management functions. System health of the AC is displayed as well as the status of the CAPs and their services. The ACgui integrates monitoring data received from the CAPs (SNMP and NetFlow data), Squid and LightSquid. The combined data is displayed meaningfully for an administrator to make informed decisions when setting thresholds for bandwidth use.



Connectivity [Tue Nov 10 20:35:00 SAST 2009]

Item	Status
Default Router	UP
Local Internet	UP
International Internet	UP

System Information:

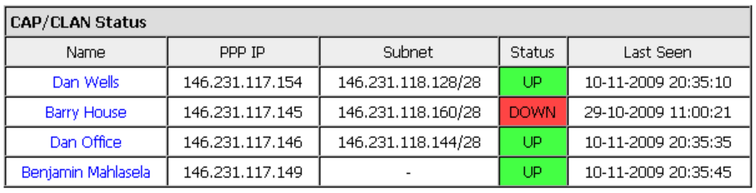
System Uptime: 25 days, 05:20

Current CPU Usage: 4 %

Current Memory Usage: 28 %

Swap Usage: 0%

Figure 4.14: ACgui - System Page



CAP/CLAN Status

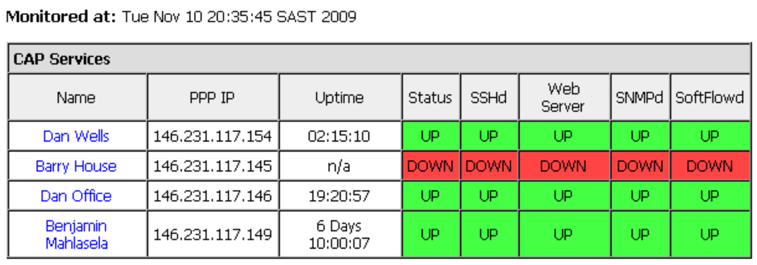
Name	PPP IP	Subnet	Status	Last Seen
Dan Wells	146.231.117.154	146.231.118.128/28	UP	10-11-2009 20:35:10
Barry House	146.231.117.145	146.231.118.160/28	DOWN	29-10-2009 11:00:21
Dan Office	146.231.117.146	146.231.118.144/28	UP	10-11-2009 20:35:35
Benjamin Mahlasela	146.231.117.149	-	UP	10-11-2009 20:35:45

Figure 4.15: ACgui - CAP/CLAN Status Page

Figure 4.14 shows the *System Status* page of the ACgui and the general layout of the ACgui. The ACgui is similar in visual design and structure to the CAPgui. It has been constructed using PHP which generates XHTML and the style is defined using CSS. As with the CAPgui, all configuration changes modify a single XML file, which is parsed to generate FreeBSD configuration files. The ACgui User Manual, source code, background scripts and screenshots of all the pages can be viewed in Appendix C.

In Figure 4.14, on the left-hand side of the interface is a menu, which is separated into the headings *AC Status*, *Network Status*, *Configuration* and *Monitoring Statistics*. Under the heading *AC Status*, information is available about the system health of the AC and the status of the Internet connection. The information contained in the *AC Status* sub-headings is obtained in a similar fashion to the CAPgui (see Section 4.4.2).

The heading *Network Status* provides status information for each CAP in the C-WAN,


Monitored at: Tue Nov 10 20:35:45 SAST 2009

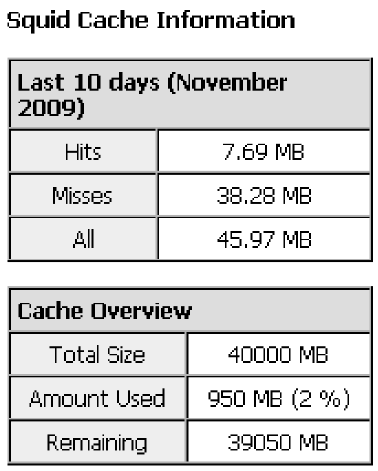
CAP Services							
Name	PPP IP	Uptime	Status	SSHd	Web Server	SNMPd	SoftFlowd
Dan Wells	146.231.117.154	02:15:10	UP	UP	UP	UP	UP
Barry House	146.231.117.145	n/a	DOWN	DOWN	DOWN	DOWN	DOWN
Dan Office	146.231.117.146	19:20:57	UP	UP	UP	UP	UP
Benjamin Mahlasela	146.231.117.149	6 Days 10:00:07	UP	UP	UP	UP	UP

Figure 4.16: ACgui - CAP Services Status Page

status of the Squid cache, most popularly accessed websites, site (C-LAN) quotas and user quotas. First, *CAP/CLAN Status* (see Figure 4.15) provides information in a table for each CAP in the network, its PPP IP Address, the subnet it serves, its current status (*UP* or *DOWN*) and the date and time of when it was last seen. This page provides a simple overview of the CAPs in the C-WAN and whether they are available or not. The use of colour, green for *up* and red for *down* is used to denote whether a CAP is available or not. The status information displayed in this table relies on a PHP script which uses *ping* [67] to attempt to reach the CAP on its PPP IP address. The status script is run every five minutes. By clicking on the name of the CAP, the Internet browser is directed to the relevant CAPgui.

The *CAP Services Status* page extends the information in the *CAP/CLAN Status* page by checking the status, uptime and services running on the various CAPs in the C-WAN (see Figure 4.16). Information is displayed in a table showing: the PPP IP address, system uptime, general status and service status of SSH, the web server, SNMPd and SoftFlowd. The general status is retrieved from the previously mentioned status script. System uptime is retrieved when checking the SNMP service. SSH and the web server status are checked by connecting to the relevant port for the service (port 22 for SSH and port 80 for the web server).

Figure 4.17 provides an example of the table that can be viewed on the *Squid Cache Status* page. This page provides a high-level summary of the cache of web content that the Squid proxy provides. Information for the table is gathered by processing the reports that LightSquid generates. The information in this table is updated every 20 minutes. The first table seen in Figure 4.17, provides a summary of the Squid cache for the current quota period (November 2009). First, it shows the *Hits*, the amount of web content served from the cache. Secondly, *Misses*, the amount of web content which had to be retrieved from the Internet. And lastly, *All*, the sum of both *Hits* and *Misses*. From the data in this example, analysis reveals that 16% of web content was retrieved from the cache in the current quota period (November 2009), resulting a 7.69 MB decrease in use of the Internet connection. The 'Cache Overview' table provides an overall status of the cache,

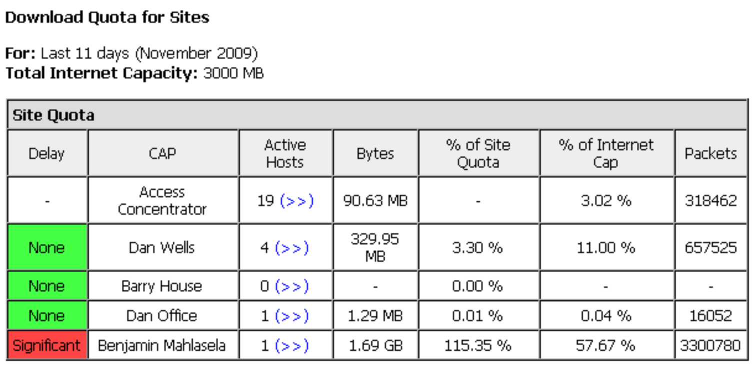


Squid Cache Information

Last 10 days (November 2009)	
Hits	7.69 MB
Misses	38.28 MB
All	45.97 MB

Cache Overview	
Total Size	40000 MB
Amount Used	950 MB (2 %)
Remaining	39050 MB

Figure 4.17: ACgui - Squid Status Page



Download Quota for Sites

For: Last 11 days (November 2009)
Total Internet Capacity: 3000 MB

Site Quota

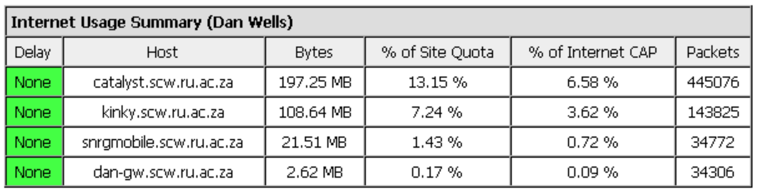
Delay	CAP	Active Hosts	Bytes	% of Site Quota	% of Internet Cap	Packets
-	Access Concentrator	19 (>>)	90.63 MB	-	3.02 %	318462
None	Dan Wells	4 (>>)	329.95 MB	3.30 %	11.00 %	657525
None	Barry House	0 (>>)	-	0.00 %	-	-
None	Dan Office	1 (>>)	1.29 MB	0.01 %	0.04 %	16052
Significant	Benjamin Mahlasela	1 (>>)	1.69 GB	115.35 %	57.67 %	3300780

Figure 4.18: ACgui - Site Quota Status Page

with the total size of the cache, how much of the cache is used and how much disk space is still remaining.

The next subheading under *Network Status* is *Top Websites*, which provides a listing of the most popular websites (domains) for the current quota period. The websites can either be sorted by total bytes transferred or total connects (visits to that domain name). Data is gathered by processing the LightSquid reports and summing up the total bytes transferred and the total connects to the various domain names for the current quota period. Viewing this information, an administrator can assess which websites are using the most amount of bandwidth and whether acceptable sites are being viewed by users. Unacceptable websites can then be placed in a blacklist to prevent access to that site.

The *Site Quotas Status* page provides a summary of bandwidth use gathered from the NetFlow sensors on the AC and the CAPs. Bandwidth use is displayed in a table, with a row for each CAP. Data displayed shows the amount of 'Active Hosts' behind the NetFlow sensor (AC and CAPs), the total amount of bytes transferred to the Internet from that site, the percentage of Site Quota used by that C-LAN, the percentage of total Internet



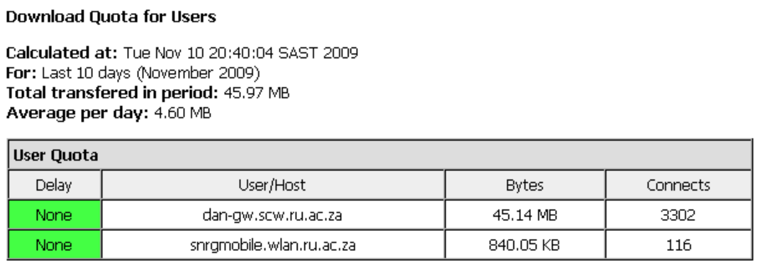
Internet Usage Summary (Dan Wells)					
Delay	Host	Bytes	% of Site Quota	% of Internet CAP	Packets
None	catalyst.scw.ru.ac.za	197.25 MB	13.15 %	6.58 %	445076
None	kinky.scw.ru.ac.za	108.64 MB	7.24 %	3.62 %	143825
None	snrgmobile.scw.ru.ac.za	21.51 MB	1.43 %	0.72 %	34772
None	dan-gw.scw.ru.ac.za	2.62 MB	0.17 %	0.09 %	34306

Figure 4.19: ACgui - Host Quota Status Page

bandwidth (of the C-WAN) used and the total number of packets that were transmitted to or from the Internet. In Figure 4.18, the first column of the table shows what type of delay the site (C-LAN) is experiencing when accessing the Internet. The limits for which the delays come into effect for each site are set in the *CAPs Configuration* page which updates the XML configuration file. The PHP script which enforces the delays compiles reports from the NetFlow flow data files, obtains site bandwidth use thresholds from the XML configuration file and updates the delay tables in IPFW. In Figure 4.18, 'Benjamin Mahlasela' is in a *significant* delay as they have used 115.35% of their allocated site quota (1500 MB) for the current quota period (November 2009). Colour has been used as follows for the delays: green denotes *no delay*, orange denotes *slight delay* and red denotes *significant delay*. An administrator can consult this table to check that bandwidth usage thresholds have been set at fair levels for each site.

By clicking the '(>>)' link in the 'Active Hosts' column on the *Site Quota Status* page, it is possible to view the Internet usage per host (host quota) for all hosts behind a specific CAP (in a specific C-LAN). An example, seen in Figure 4.19, is for all the hosts in the 'Dan Wells' C-LAN. In this example, there are four hosts in the 'Dan Wells' C-LAN, and bandwidth use for each is shown. This page extends the overall C-LAN information seen in the *Site Quota Status* page to separate bandwidth use per host. In the first column of the table, delay levels are shown, in this example no hosts are in a delay as they have not exceeded their allocated bandwidth. Colour is used in this column, in the same manner as the *Site Quota Status* page. Each host has information for the total number of bytes transferred to the Internet, percentage of site quota used, percentage of total Internet bandwidth used and packets transmitted. An administrator can consult this table to check whether Internet bandwidth used per host is set at fair levels, and whether particular hosts are in a delay.

Figure 4.20 demonstrates the *User Quotas Status* page. The summary at the top of the page provides details about when the user quotas were last calculated, the period that they were calculated for, the total bytes transferred in that period and an average bandwidth used per day. A table is generated for user quotas, displaying the delay, the user (or host), total bytes transferred (bandwidth used) and connects via the Squid proxy. In the example of Figure 4.20, Squid has not been used in conjunction with user authentication

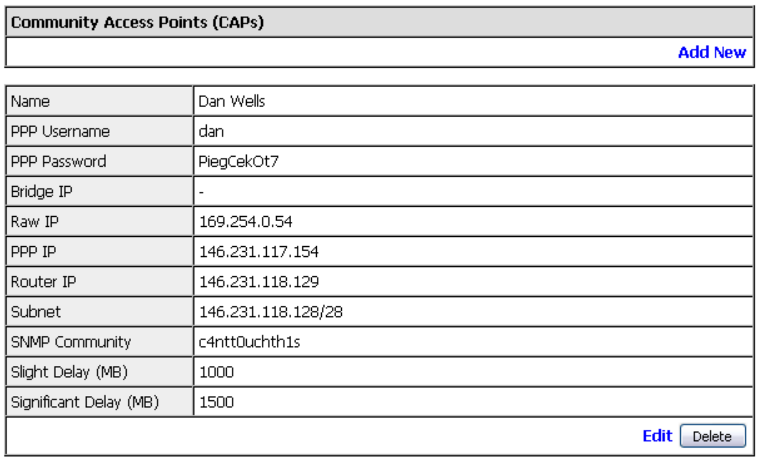


Download Quota for Users

Calculated at: Tue Nov 10 20:40:04 SAST 2009
For: Last 10 days (November 2009)
Total transfered in period: 45.97 MB
Average per day: 4.60 MB

User Quota			
Delay	User/Host	Bytes	Connects
None	dan-gw.scw.ru.ac.za	45.14 MB	3302
None	snrgmobile.wlan.ru.ac.za	840.05 KB	116

Figure 4.20: ACgui - User Quota Status Page



Community Access Points (CAPs)

Add New

Name	Dan Wells
PPP Username	dan
PPP Password	PiegCekOt7
Bridge IP	-
Raw IP	169.254.0.54
PPP IP	146.231.117.154
Router IP	146.231.118.129
Subnet	146.231.118.128/28
SNMP Community	c4ntt0uchth1s
Slight Delay (MB)	1000
Significant Delay (MB)	1500

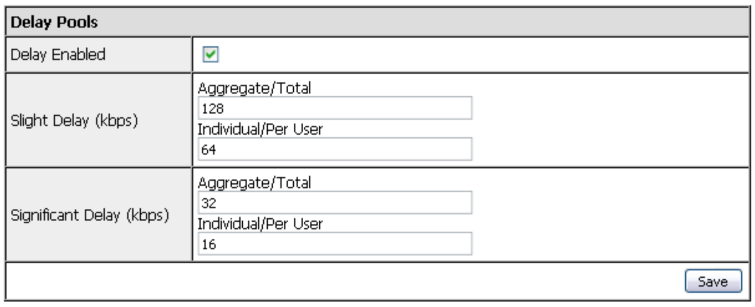
Figure 4.21: ACgui - CAPs Configuration Page

and therefore the *users* are in fact the hosts that have accessed the Internet via the Squid proxy.

Following the *Status* pages are the *Configuration* pages. *Configuration* pages are available for *General* settings, *Network*, *PPPoEd*, *CAPs*, *Static Routes*, *SNMPd*, *Squid*, *Blacklists*, *Flow Capture*, *Quotas*, *View Config Files* and *Reboot AC*. For more detailed information about each *Configuration* page, consult Appendix C. Pages explained in more detail in this section are *CAPs*, *Squid* and *Quotas*.

The *CAPs Configuration* page allows the addition, removal and configuration changes of CAPs within the C-WAN. See Figure 4.21 for an example of the *CAPs Configuration* page in the ACgui. Each CAP is presented in a separate table showing the configuration of each. Importantly, configuration is here for the *Slight Delay* and *Significant Delay* fields which affect the bandwidth use thresholds of the C-LAN. The fields update the XML configuration file, which in turn is parsed to generate configuration files for PPPoE. The delay fields are checked when calculating Site Quotas.

The *Squid Configuration* page is fairly comprehensive, after which a Squid configuration file is generated. It allows the modification of configuration for general squid settings, cache settings, delay pools and SNMP monitoring for Squid. General settings include IP



Delay Pools	
Delay Enabled	✓
Slight Delay (kbps)	Aggregate/Total 128 Individual/Per User 64
Significant Delay (kbps)	Aggregate/Total 32 Individual/Per User 16
	Save

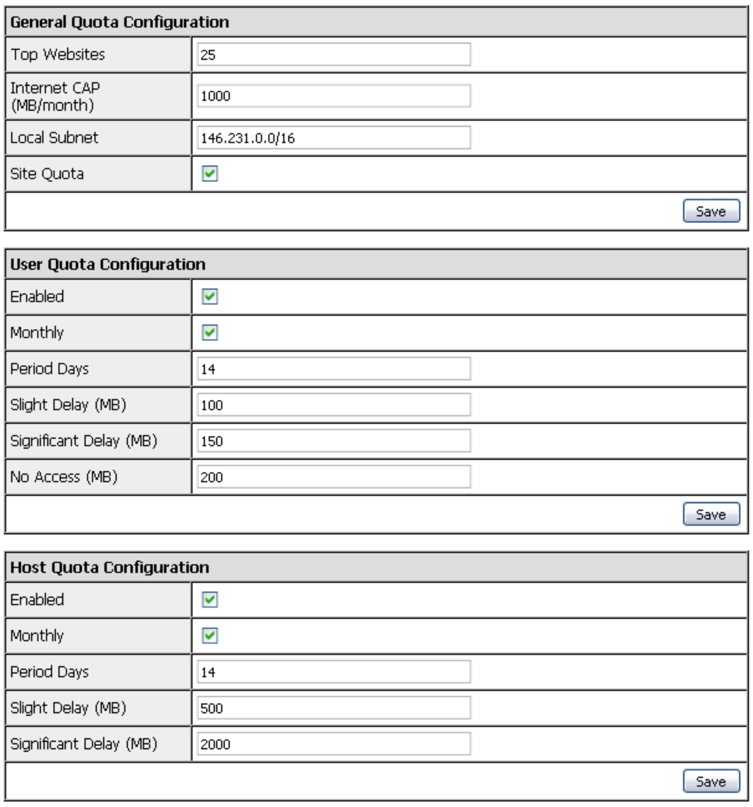
Figure 4.22: ACgui - Squid Configuration Page - Delay Pools

address and port to listen on, setting of transparent proxy, user authentication, enabling blacklists and the subnet allowed to use Squid. Cache settings include the size of the cache and configuration for a cache parent. A cache parent is an upstream cache to which all Internet requests are forwarded. Figure 4.22 shows the delay pools configuration for Squid. Delays pools can be disabled here if they are not required. Throughput delays can be set for *slight delay* and *significant delay* in kbps. Aggregate/Total is the total throughput provided to all users in that delay pool, the combined delay pool users will never reach a throughput higher than the value set here. Individual/Per User is the throughput allocated to each user in the delay pool. Thresholds for when these delay pools come into effect are set in the *Quotas Configuration* page.

Figure 4.23 shows an example of the *Quota Configuration* page. First, in the top table, general settings can be modified: this includes the number of top websites to show in the Top Websites Status page, total Internet Capacity of the shared Internet connection per month, the local subnet of all hosts within the C-WAN and whether Site Quotas are enabled or not. Secondly, thresholds and periods can be modified for User Quotas. The period can either be set as *Monthly* where bandwidth usage is calculated for the current month, or per *Period Days* where bandwidth usage is calculated for a specific number of days using a sliding window. Host Quotas have a similar configuration.

When configuration has been updated, the AC must be rebooted to enforce the changes to permanent configuration files. Changes that do not require the AC to be rebooted are modifications to *CAPs*, *Squid*, *Blacklists* and *Quotas*. Before rebooting the AC, the administrator should consult *View Config Files* to make sure that the new configuration is correct.

From the ACgui, links to Cacti and LightSquid are available to access more information and graphs for the AC and CAPs in the network. Before the CAPs and the AC were deployed to the testbed networks, a testing environment was required. As a result, VMware was used to construct a virtual network similar to those of the testbed networks.



General Quota Configuration	
Top Websites	25
Internet CAP (MB/month)	1000
Local Subnet	146.231.0.0/16
Site Quota	✓

User Quota Configuration	
Enabled	✓
Monthly	✓
Period Days	14
Slight Delay (MB)	100
Significant Delay (MB)	150
No Access (MB)	200

Host Quota Configuration	
Enabled	✓
Monthly	✓
Period Days	14
Slight Delay (MB)	500
Significant Delay (MB)	2000

Figure 4.23: ACgui - Quota Configuration Page

4.6 Virtual Development Environment

This system was ultimately deployed at the testbed networks which were described in Section 2.11, and the deployment discussed in Chapter 5. However, a development environment was required during the implementation phase. In lieu of building a development network with physical connections, devices and hardware, a virtual network was constructed using *VMware Workstation* [112].

The main advantage of using a virtual network over a production network for the development of the system is that modifications can be easily made without disrupting users on the network. By using snapshots of running virtual machines, a known working state can be saved. If additions or modifications to the virtual machine are not performing as they should (or have been misconfigured), the virtual machine can be rolled back to the previously stored working state. Snapshots also allowed experimental development and testing and if the experiments failed no work or time was lost in the process.

The virtual network was configured and installed using *VMware Workstation 6.5.1* on *Ubuntu 8.10* [111]. The private virtual networks (each subnet) were created by using the *Virtual Network Editor* tool which is bundled with *VMware Workstation*. Virtual interfaces on the virtual machines are bound to the private virtual networks and can only

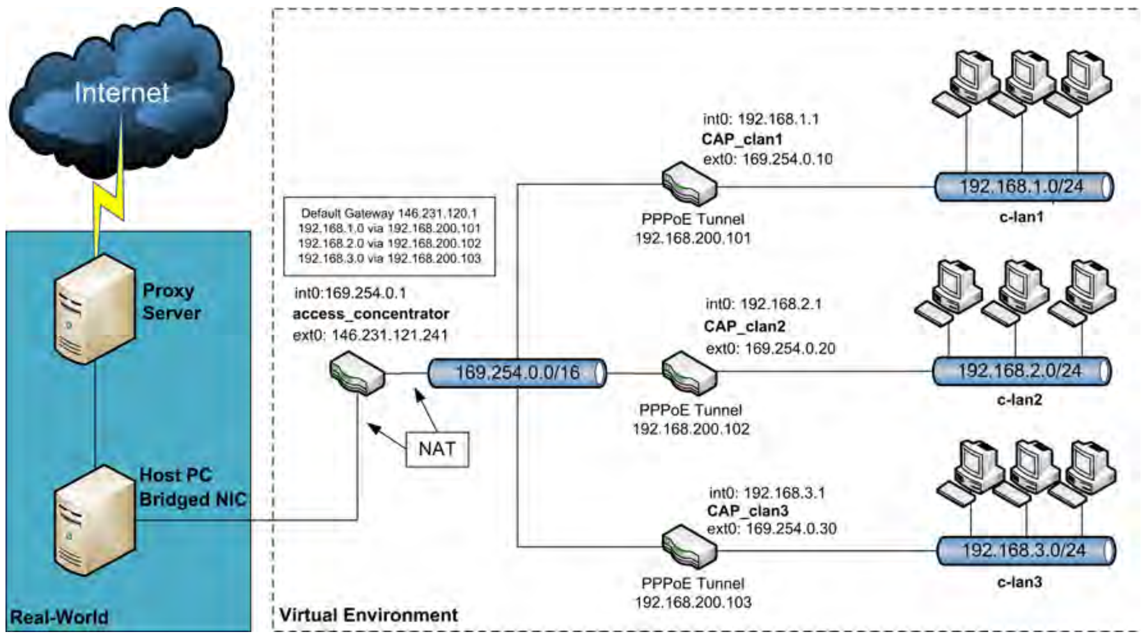


Figure 4.24: Virtual Development Network

communicate via the virtual network that they are bound to. This is similar to a physical connection in that a computer can only communicate via the network it is attached to.

The virtual development network was based on the two testbed networks and is similar in structure. Figure 4.24 provides an overview of how the virtual network was constructed. On the right of Figure 4.24 are three C-LANs, and each C-LAN has a specific subnet. In *c-lan1*, networked devices have an IP in the range of 192.168.1.0/24 and their default router is *CAP_clan1*. The CAP labeled *CAP_clan1* has two virtual NICs, the 'internal' NIC connected to the virtual network *c-lan1* (subnet 192.168.1.0/24) and the 'external' NIC connected to the C-WAN (subnet 169.254.0.0/16). The C-WAN is used for link testing and troubleshooting but primarily carries PPPoE tunnels which are established on the links to ensure encrypted and authenticated communication between the *access_concentrator* and the CAPs.

All traffic entering or leaving *c-lan1* passes through *CAP_clan1*. The CAPs (*CAP-clan[1-3]*) route traffic to the *access_concentrator* via their PPPoE tunnel interface. The *access_concentrator* then routes the traffic to the Internet or to another CAP in the C-WAN. As the virtual network IP ranges are in the RFC 1918 [78] address space, the *access_concentrator* uses NAT to translate their IP addresses to an IP address which is Internet routable. The *access_concentrator* has two virtual NICs, the first is attached to a virtual network (in the IP range 169.254.0.0/16) and the second is bridged onto the hosts physical NIC. In this virtual network, traffic is routed from the *access_concentrator* over the Rhodes University network (with a Rhodes University IP address) to the Internet.

By implementing the AC and CAP in this virtual network, it allowed problems to be identified and resolved before deploying the system to real world networks. A number of applications within the system depend on each other for the project goals to be met, therefore evaluating whether these goals had first been achieved in a virtual environment led to a smoother deployment to the two research networks. By identifying problems before deploying the system to the two research networks, it allowed the construction of a more robust and ultimately better system.

4.7 Summary

This chapter built on the design requirements of the system that were described in Chapter 3. A number of applications have been identified to meet the design requirements and have been implemented into a two part solution. The two components are the AC and the CAP, and both are based and built on the FreeBSD OS.

The CAP is the default router for a C-LAN, and connects to the AC via an encrypted and authenticated PPPoE tunnel. It collects monitoring data for the C-LAN by using an SNMP agent and a NetFlow sensor. The SNMP agent provides a method to monitor the system health of the CAP and gather statistics on the network interfaces. The NetFlow sensor gathers data on the flows of data entering and leaving the C-LAN. The SNMP data is polled from the CAP by the AC and NetFlow data is transmitted to the NetFlow collector on the AC. The CAPgui was constructed for initial and subsequent network configuration, and provides a number of monitoring statistics that have been gathered on the CAP. The CAPgui is the first point-of-reference for troubleshooting network issues at the C-LAN, providing the status of the CAP itself and the upstream connectivity.

The AC is the C-WANs default gateway router to the Internet and terminates PPPoE tunnels from the various CAPs within the community network to ensure all traffic is encrypted and authenticated. All Internet bound traffic is translated using NAT, providing internal network hosts with Internet connectivity. An IPFW firewall is configured to limit incoming and outgoing connections to specific ports, providing security to the C-WAN hosts. An SNMP agent is configured to provide a method to monitor the ACs system health. And a NetFlow collector receives NetFlow data from all the CAPs.

The AC utilised three important applications in terms of reaching the project goals. First, the Squid caching proxy server presents a single method for all hosts within the C-WAN to access the Internet. Squid also reduces bandwidth usage by caching regularly accessed web content. Squid logs every connection and by processing these logs it was possible to implement user based Internet quotas by using delay pools. Blacklists were integrated with Squid to prevent access to specific types of web content. The second application, LightSquid, provided a summary of the Squid logs, displaying Internet usage per user; Internet usage per day, month and year; and the most popularly accessed websites. By analysing the popularly accessed websites, they can be placed in a black-lists to deny access to them via the use of the ACgui and Squid. The third application, Cacti, retrieved monitoring data from the CAPs, the AC and Squid to generate graphs for monitoring purposes.

Internet usage summaries can be viewed on the ACgui, summarised per user, host and C-LAN (site). Utilising this bandwidth monitoring, administrators can impose limits on Internet usage. User based quotas are imposed by summarising the Squid access log for each user and using Squid delay pools with predefined throughput limits. Users have higher delays imposed on them as their bandwidth usage increases. The delay pools are *no delay*, *slight delay*, *significant delay* and *no access*. Configuration of the period of Internet usage, throughput limits and thresholds for moving between the delay pools can be configured using the ACgui.

Host and C-LAN (site) quotas are calculated by processing collected NetFlow flow data from the AC and CAPs. Limits on bandwidth use are then imposed by using IPFW tables and Dummynet pipes and queues. Delays that are enforced are *host slight delay*, *host significant delay*, *site slight delay* and *site significant delay*. The thresholds and periods for moving between these delays can be configured using the ACgui.

The ACgui provides administrators with a single portal for managing and monitoring the C-WAN. Summaries of C-LAN use are provided, including their Internet usage and the status of services on their CAP. The ACgui also provides many configuration pages which allow administrators to configure the AC; modify the CAPs authorised on the C-WAN; and modify thresholds for user, host and site based quotas.

The system, the AC and the CAP combined, allow community networks to be managed and monitored from a single portal, which is the ACgui. The system provides a number of variables which aid the administrator in resolving network problems within the C-WAN.

This chapter described a working system and deployed it in a virtual environment for development and reliability testing. Chapter 5 discusses the deployment of the system to the research networks that were identified in Section 2.11.

Chapter 5

Deployment and Results

5.1 Introduction

Once the system had been deployed and tested in a virtual environment, as discussed in Section 4.6, it was deployed to the two research networks which were presented in Section 2.11.

This chapter begins with an overview of the hardware used for the deployment of the Community Access Points (CAPs) and the Access Concentrator (AC). A deployment strategy was created to aid the process and this is discussed in Section 5.3. Next, the deployment to the Centre of Excellence (CoE) network is discussed in Section 5.4, with its related results in Section 5.5. Following this, deployment for the Siyakhula Living Lab (SLL) network is described in Section 5.6 and the results are presented in Section 5.7.

The deployed system in the two networks and their related results are reflected upon in Section 5.8.

5.2 Hardware and Installation

Revisiting the design specifications (seen in Section 3.3 and 3.5), the CAPs and AC had specific hardware requirements. The CAPs and AC were required to be able to run on a device with minimal hardware performance specifications.

The platform that was chosen for the deployments was a small-form factor (Mini-ITX case) Personal Computer (PC). The PC contained an Intel Atom N270 1.6 GHz processor and 1 GB of RAM. It had built in video and a 100 Mbit/s Ethernet Network Interface Card (NIC). It had a single PCI expansion slot which was used to house an additional NIC. The PCs were ordered with the Ubuntu Operating System (OS) pre-installed and therefore no additional cost was incurred for purchasing a proprietary OS. The PCs contained no CD/DVD-drive or floppy drive and as such installation was performed using a bootable

USB flash stick with g4u ("ghosting for unix") [24]. Details regarding how g4u was used for installation is contained in Appendix C.

g4u[1] is a NetBSD-based harddisk cloning tool which allows the deployment of a harddisk image to a number of PCs using FTP [24]. Harddisk images of the CAP and AC were created from the virtual implementation environment (discussed in Section 4.6) and uploaded to an FTP server. This allowed CAPs and ACs to be deployed to physical PCs concurrently.

Only minor modifications to the system configuration of the physically deployed AC and CAPs had to be made: the network interface names[2] had to be updated from the virtual to physical NIC; and the harddisk device name[3] for the physical harddisk. These changes were trivial to update in FreeBSD as the system is flexible to cope with these changes. Once this had been completed, the CAP's and AC's harddisks were imaged to make a final, easily deployable image which required no FreeBSD system configuration updates.

The CAP and AC were deployed with pre-configured IP addresses for the two network interfaces. The CAP had an external IP address of '169.254.0.2' and an internal IP of '192.168.0.1'. The AC had an external IP of '10.0.0.1' and an internal IP of '169.254.0.1'. This aided in easy configuration of the devices, from another networked PC, via their respective web based interfaces.

Following the deployment of the ACs and CAPs to physical PCs from the virtual network, further testing was done to make sure all components were functioning correctly. After the testing, a strategy for deployment was designed to ensure a smooth deployment to the two research networks.

5.3 Deployment Strategy

To minimise the risks of deploying this system to the two research networks, a deployment strategy was constructed. The core of this strategy was to fully understanding the existing networks and their topologies. Having accurate knowledge of the network allowed the quick resolution of potential problems during deployment. The strategy involved undertaking research into three specific areas: the type of Internet connection present, connection medium used between sites and services offered on the network.

First, the strategy involved understanding how the network is provided with Internet

[1]g4u - http://www.feyrer.de/g4u/

[2]Setting Up Network Interface Cards - http://www.freebsd.org/doc/en/books/handbook/config-network-setup.html

[3]Storage Device Names - http://www.freebsd.org/doc/en/books/handbook/disks-naming.html

connectivity along with the IP address(es) that it was allocated by the ISP. This step also involved determining the IP addresses for the ISP's DNS servers. Secondly, the connection medium was researched, to understand how the various hosts and sites within the network were interconnected. This step also observed how IP packets were routed to the rest of the network, or to the Internet. The third step, sought to document what existing services the network provided or required. This final step also gathered information about which additional network services were required for connectivity (such as a DHCP server) on a site by site basis.

Beyond understanding the network and developing a strategy, a deployment procedure had to be planned for each of the research networks.

5.4 Deployment in CoE Testbed Network

The Centre of Excellence (CoE) testbed network extends the Rhodes University network and is situated in Grahamstown (see Section 2.11.1). The CoE network provides connectivity to the Rhodes University network and the Internet for a handful of Rhodes University staff members and post-graduate students. It also provides Internet connectivity for a number of previously disadvantaged schools situated throughout Grahamstown. This system was deployed to the Settler City Wireless (SCW) portion of this network incrementally over the period of a year. An overview of the SCW network can be seen in Figure 5.1.

The CoE network is dependent on the Rhodes University network and relies on its Internet connection to obtain Internet connectivity. The Rhodes University network provides a cluster of proxy servers which must be used to access the World Wide Web (WWW), any host which does not use the proxy servers are denied Internet access. Rhodes University also provide a cluster of DNS servers for which all name lookups within the network can be performed. All hosts within the CoE network are dependent on the Rhodes University proxy servers and DNS servers for external Internet access.

In Figure 5.1, the SCW portion of the CoE network is displayed, demonstrating how it is linked to the Rhodes University network. The deployment procedure involved replacing the existing site routers at four sites, each were replaced with a CAP to initially connect with the previous SCW access router (*dukat.dsl.ru.ac.za*). Once this step was complete, the system's AC was deployed (*gf3.dsl.ru.ac.za*) in parallel with the previous access router, see Figure 5.1. The four deployed CAPs were then modified to route traffic through the new AC. Other sites would still route their traffic through the existing access router (*dukat.dsl.ru.ac.za*).

On the right-hand side of Figure 5.1 are the four deployed sites. WiFi and WiMAX

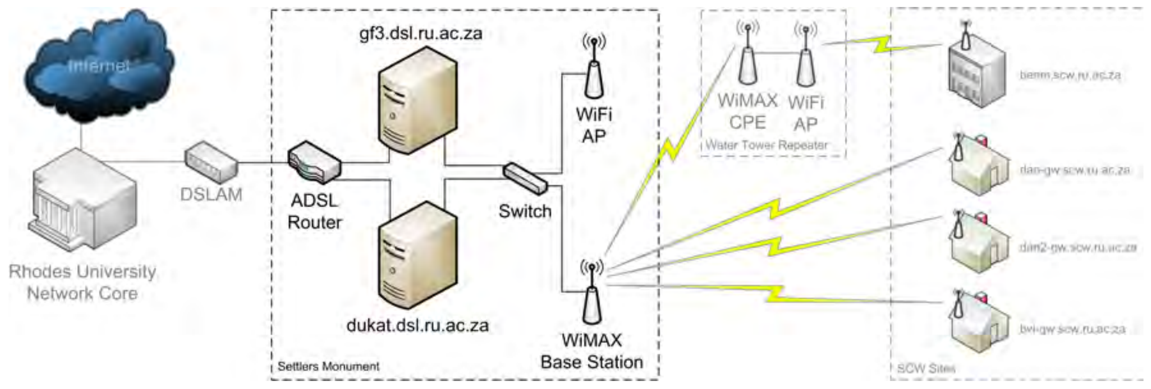


Figure 5.1: CoE Testbed - Logical Network Layout - SCW Deployment

Site Name	Type of User	PPP Address	Subnet	Connection Medium
Benjamin Mahlasela	School	benm.scw.ru.ac.za	192.168.0.0/24	WiFi via WiMAX Repeater
Dan Wells	Post-Grad Student	dan-gw.scw.ru.ac.za	146.231.118.128/28	WiMAX
Dan Wells 2	Test Site	dan2-gw.scw.ru.ac.za	146.231.118.144/28	WiMAX
Barry Irwin	RU Staff Member	bvi-gw.scw.ru.ac.za	146.231.118.160/28	WiMAX

Table 5.1: SCW - Deployed Sites Information and Requirements

technologies are used to connect the remote sites to the AC. Each site has a CAP which facilitates this connection for the hosts at the site. From each site, packets are routed over the WiFi or WiMAX network via PPPoE to the AC, which either routes packets to other hosts within the SCW network or to the Rhodes University network via the ADSL connection. Beyond the ADSL connection is out of the scope of this project, however, packets can be routed to any host in the Rhodes University network, to the proxy servers, to the DNS servers or to the Internet.

The four deployed sites were a school, a post-graduate student, a test site and a Rhodes University staff member. More detail about each site can be seen in Table 5.1.

The school site (*benm.scw.ru.ac.za*) only required access to the Internet, web servers hosted within Rhodes University and Rhodes University DNS servers. The school was not permitted access to other network services which the Rhodes University network provides. As external Internet connectivity is only granted via the Rhodes University proxy servers, all HTTP traffic from the school was configured to be automatically forwarded to the proxy servers. Automatic forwarding was configured using *forward* rules in the IPFW firewall to redirect HTTP traffic to *tproxy*[4] [83] (transparent proxy) on the CAP. ICMP packets (for use with *ping* [67]) were allowed in and out, and Secure SHell (SSH) was allowed to the CAP and out of the school subnet. Besides the HTTP traffic which was automatically forwarded to Rhodes University by the CAP, ICMP and SSH packets had their source IP

[4]Transproxy (*tproxy*) - http://transproxy.sourceforge.net/

addresses translated to the PPP IP address of the CAP using NAT [29]. NAT was required as the internal subnet of the school was in a private IP range (192.168.0.0/24) [78].

The other three sites (*dan-gw.scw.ru.ac.za*, *dan2-gw.ru.ac.za* and *bvi-gw.scw.ru.ac.za*) had no limitations imposed on their use of the Rhodes University network. They were provided with a subnet in the Rhodes University Class-B network (146.231.0.0/16) and therefore required no NAT. Hosts behind the CAPs would require manual configuration of applications to use the Rhodes University proxy servers to access the Internet. One site, *dan-gw.ru.ac.za*, had a special requirement for a DHCP server, to allow automatic configuration of TCP/IP settings for hosts within its subnet. The DHCP server at this site was configured to automatically provide TCP/IP settings for hosts in the subnet such as IP address, subnet mask, default gateway, DNS servers and WINS servers.

The CAPs were configured using their CAPgui, which allowed easy deployment and configuration. The school had an additional requirement of *tproxy* which required the manual modification of the FreeBSD system configuration file */etc/rc.conf*.

The AC was not required to have a proxy server, as all sites would access the Internet via the Rhodes University proxy servers. Rhodes University imposes limits on Internet use via their proxy servers (User Quotas), and also Host Quotas per IP address. Therefore, it was decided to disable all quotas on the AC as enforcing two levels of Internet usage quotas would be redundant and possibly confusing for users. Monitoring, however, was still performed to provide information on how the Internet was being used by each site.

5.5 Results - CoE Testbed Network

The SCW network, albeit similar to the logical network topology of the SLL network, had fewer requirements. The results discussed in this section were used to create a more efficient and robust system for the deployment in the SLL.

As previously stated in Section 5.4, all quotas were disabled in the deployment of this system to the SCW network. Quotas were only enabled intermittently to test their accuracy and effectiveness. For this reason, there is no discussion on quotas in this section.

In Section 5.5.1 bandwidth use at the school site is investigated along with some of the popular websites which were accessed over a month. Following this is a discussion on a number of Cacti graphs for devices in the SCW network.

5.5.1 Bandwidth Use

A school at which one of the CAPs was deployed is Benjamin Mahlasela, and results from this site are presented in this section. The CAP was configured to automatically forward their Internet traffic to the Rhodes University proxy servers. The application which was

Website	Connects
www.theherald.co.za	6431
ads.avusa.co.za	4684
pic.classistatic.com	3927
www.timeslive.co.za	3829
www.google-analytics.com	3662
ad.yieldmanager.com	3545
l.yimg.com	2903
mail.google.com	2429
us.mg4.mail.yahoo.com	2087
us.bc.yahoo.com	2047
ad.za.doubleclick.net	2029
www.dispatch.co.za	1959
cdn.24.com	1940
www.google.com	1927
us.mg3.mail.yahoo.com	1616

Table 5.2: SCW - Top 15 Websites - Sorted By Connects - Nov 2009

used to automatically forward the traffic was *tproxy*. This section investigates the log files which were generated from *tproxy*. The log files only provide simple data such as: the source IP; the URL which was requested; and the number of requests which were forwarded.

The listing in Table 5.2 highlights the websites which generated the highest number of connects. Due to the nature of the *tproxy* log files, the bytes to retrieve the requests were not recorded. Studying the top 15 websites in Table 5.2, it appears that bandwidth use at this site included: news websites; advertising websites; Yahoo! Image Search; Yahoo! Mail; Google Mail; and Google search.

This type of Internet traffic could be expected from an educational setting. The website *www.theherald.co.za* is an up to date and online version of an Eastern Cape newspaper. Accessing this website would allow staff and students of the school to be knowledgeable about current affairs in the region. The use of online mail services can be attributed to users utilising email for communication outside of the school, as there is no school mail server. Beyond these, typical requests include Internet searches for content and images.

5.5.2 System Use Graphs

This section presents, analyses and discusses a number of graphs from the Cacti graphing application. The graphs aided the project in assessing usage patterns, the quality of network links and the efficiency of services which the project system provided.

Figure 5.2 displays a graph of the PPPoE tunnel interface on the Benjamin Mahlasela

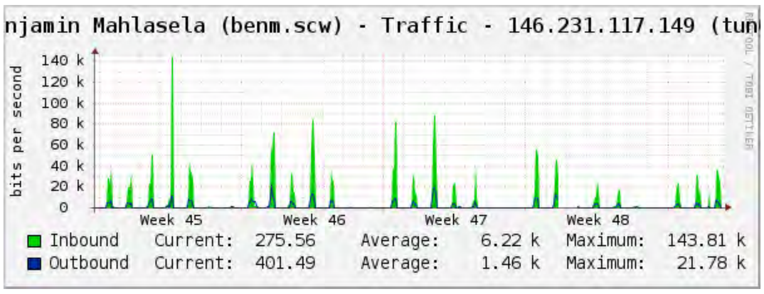


Figure 5.2: SCW - Benjamin Mahlasela - Traffic(tun0) - Nov 2009

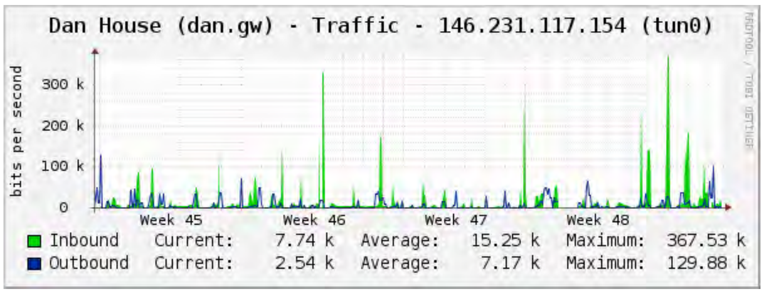


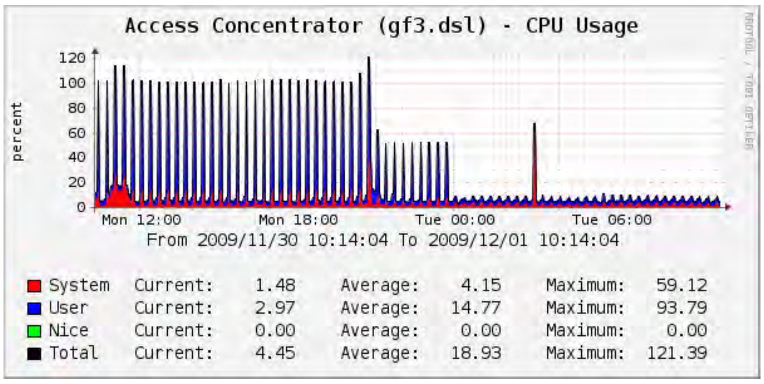
Figure 5.3: SCW - Dan Wells - Traffic(tun0) - Nov 2009

CAP for the period of November 2009. This graph revealed that students and staff at the school typically use the network during weekdays. The school is generally only open during weekdays and traffic during this time should be expected. The majority of traffic shown in this graph is inbound traffic as the school is downloading more content than they are uploading. It appears they are using the school network to search for information on the Internet and this relates with the results obtained in Table 5.2.

Figure 5.3, when compared to Figure 5.2, shows a more erratic use pattern. The deployed site in this example was that of a Rhodes University post-graduate student who was not restricted to specific Internet usage times by using a computer lab. Figure 5.3 shows more outbound traffic compared to the previous figure, as this student was transferring data between their site and hosts on the Rhodes University network.

In Figure 5.4, the CPU usage of the AC is presented. The first half of the graph displays a regular (20 minute intervals) high load on the CPU. The cause of the high CPU load was investigated and it was found that the PHP script which analyses NetFlow data was the cause. The script was parsing all stored NetFlow data instead of only the current and most relevant NetFlow data. This led to a more efficient NetFlow parsing script which required less CPU time (seen in the decrease on the right-hand side of the graph).

Learning from the experimentation, results and deployment in the SCW portion of the CoE testbed network led to the deployment of a more robust system for the SLL network.



🟥 System	Current:	1.48	Average:	4.15	Maximum:	59.12
🟦 User	Current:	2.97	Average:	14.77	Maximum:	93.79
🟩 Nice	Current:	0.00	Average:	0.00	Maximum:	0.00
⬛ Total	Current:	4.45	Average:	18.93	Maximum:	121.39

Figure 5.4: SCW - Access Concentrator - High CPU Use

5.6 Deployment in SLL Network

The SLL testbed network is situated in rural Eastern Cape, the network is discussed in more detail in Section 2.11.2. Staff and students at the schools use the PCs in the computer labs. The schools are geographically dispersed and connected to each other via WiMAX technology. This system was successfully deployed to the SLL network and an overview of the network connecting the schools can be seen in Figure 5.5.

Figure 5.5 shows the single Internet connection, provided via VSAT, a satellite backhaul. The Internet connection is housed at Mpume, which is connected via a WiMAX network to the other schools. The WiMAX base station is located at Ngwane, while the other schools have WiMAX Customer Premise Equipment (CPE). DNS name resolution is provided by querying the ISP's (Telkom) DNS servers.

The VSAT provides a 3 GB monthly bandwidth capacity, with a maximum download throughput rate of 512 Kbps and upload rate of 128 Kbps [59]. Sharing this single Internet connection equitably between the schools was one of the goals of this project.

Before deployment, a procedure was created. The deployment procedure involved researching the layout of the SLL network (see Figure 5.6) and constructing a mock network in a lab environment. This mock network involved using the physical devices (two CAPs and the AC) which were to be deployed to the SLL. The SSL deployment is a six hour car trip from Rhodes University to the Dwesa-Cwebe region, and by confirming the devices were operating correctly before departing, the deployment process ran smoothly.

The AC was deployed at Mpume, and the CAPs were deployed to Ngwane and Mthokwane. Two other schools, Nondobo and Nqabara were not included in the deployment. Reasons for their exclusion from the deployment are further described in Section 2.11.2, in summary, Nondobo seldom use their computer lab and Nqabara has yet to be connected to the WiMAX network. The deployment procedure involved confirming configuration details of the existing AC at Mpume and then replacing it with the new system's AC.

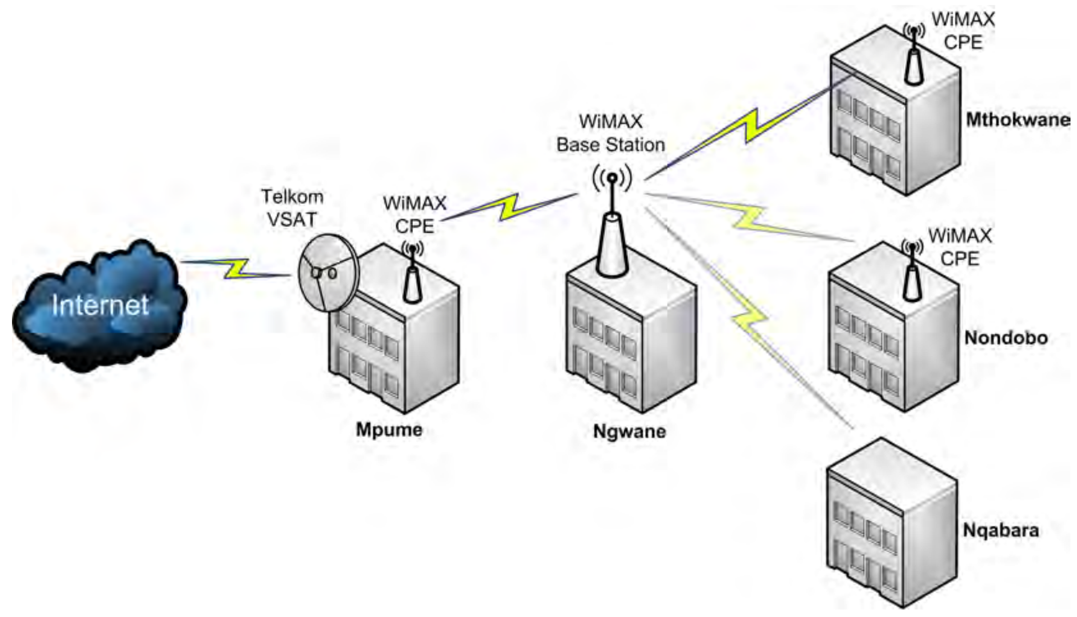


Figure 5.5: Overview of SLL Network

After this, the procedure was to replace the the previous routers with the CAPs at Ngwane and then Mthokwane. The procedure also included labeling the various cables and how they plugged into the CAPs and AC. Labeling of the cables and the devices was to aid local support staff to reconnect a CAP or AC, in the event that they had become disconnected. It can help the SLL staff in ensuring that devices are connected correctly when troubleshooting network issues. Most importantly, labeling would ensure that network cables were connected to the correct network interface. A flow-chart was prepared, see Appendix C, to aid Ngwane and Mthokwane in troubleshooting network connectivity problems.

Each of the schools have a school server, which is an Edubuntu Linux Terminal Server Project (LTSP) server which boots thin client PCs [13]. The thin client PCs are low-cost or low-performance hardware and all processing is performed on the Edubuntu server. LTSP enables the use of older machines as thin clients, reducing hardware and administration costs, as administrators only have to maintain software on the server [13]. Using LTSP and thin clients also prolongs the life and usefulness of the thin clients as all applications are processed on the server and as such older PCs can be used as thin clients.

The physical layout of the SLL network is somewhat different to the logical layout and is described for clarity. The physical connections are described first. Each school is connected via a WiMAX CPE to the WiMAX base station located at Ngwane. This provides TCP/IP connectivity between each of the schools via the WiMAX base station. The CAPs and AC's network interface which is facing the WiMAX network is in the range 169.254.0.0/16. This network provides a method to troubleshoot connectivity issues

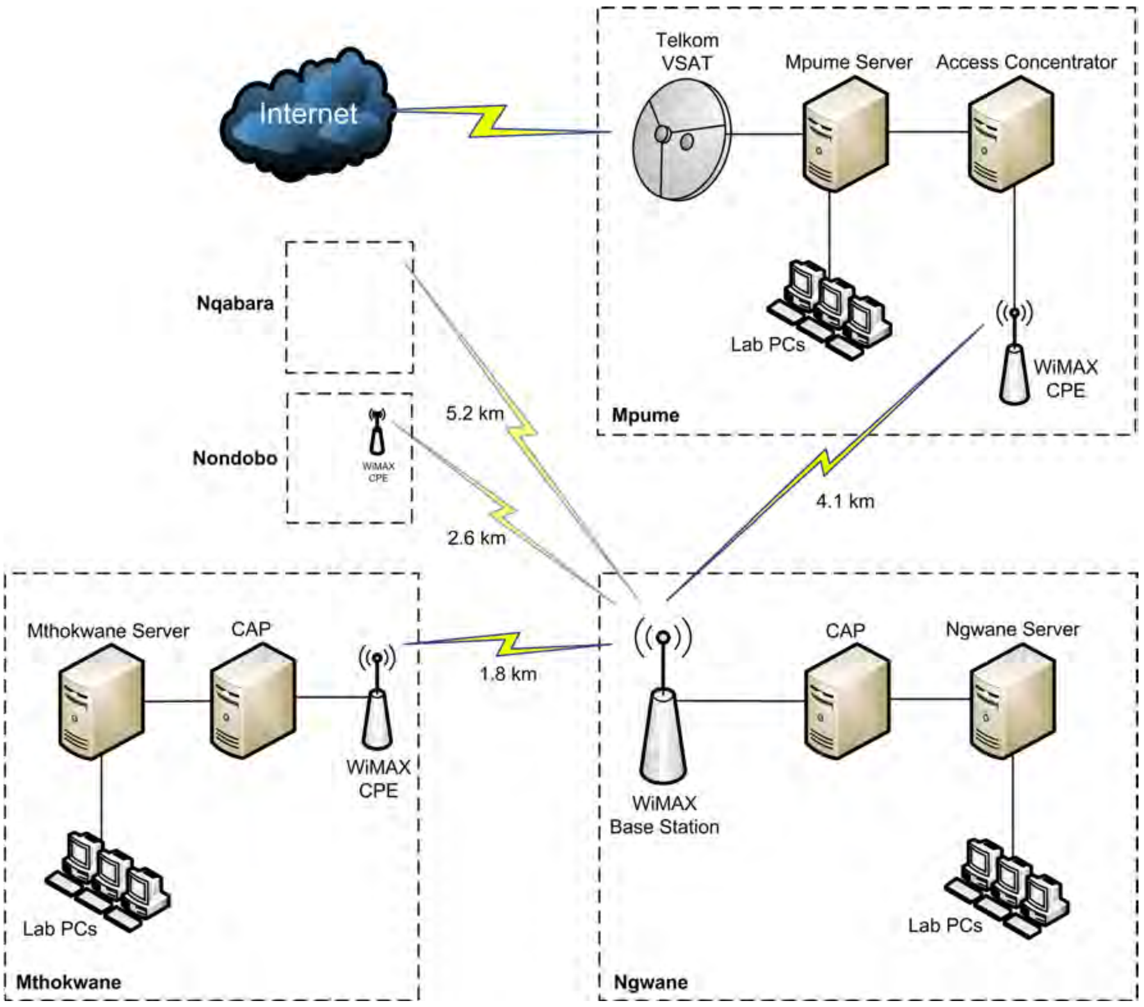


Figure 5.6: SLL - Logical Network Layout - Deployment

Site Name	PPP Address	Subnet	Connection Medium
Mpume	n/a	192.168.1.0/24	Ethernet
Ngwane	192.168.255.2	192.168.2.0/24	WiMAX Base Station
Mthokwane	192.168.255.3	192.168.3.0/24	WiMAX CPE

Table 5.3: SLL - Deployed Sites Information and Requirements

between the schools and to facilitate PPPoE tunnels between the CAPs and AC. All communication from the CAPs, whether it is destined for the Internet or another school, is routed via the AC over their secure PPPoE tunnel. The AC routes traffic destined for the Internet to the Mpume Server.

The network layout provided in Figure 5.6 shows how the network was logically structured and what devices are present. The Mpume Edubuntu server boots the thin clients in their computer lab, and provides the gateway to the Telkom VSAT. The Mpume lab does not need to communicate with the AC for access to the Internet, as all their Internet based traffic is routed directly to the VSAT. This is because there is no need for the connection to be encrypted using PPPoE as Mpume's Internet bound traffic is not carried via the WiMAX network. Connected to the Mpume server is the AC, which provides connectivity to the Ngwane and Mthokwane schools. The AC, located at Mpume, connects to the WiMAX base station via its WiMAX CPE. Located at Ngwane is the WiMAX base station, terminating the WiMAX connections from the other two schools. Connected to the WiMAX base station is Ngwane's CAP and beyond that is the Ngwane server and computer lab. The Ngwane Edubuntu server boots thin clients for their computer lab. Lastly, Mthokwane is similar to Ngwane however the Mthokwane CAP connects to a WiMAX CPE.

The SLL network is dependent on other sites due to its physical layout and inherently has several points of failure. For example, Ngwane cannot access the Internet if the AC is down, if the Mpume Server is down or if the Telkom VSAT is down. While Mthokwane cannot access the Internet if the WiMAX base station at Ngwane is down, if the AC is down, if the Mpume Server is down or if the Telkom VSAT is down. Section 5.8 provides two recommendations to modify the structure of the SLL network to resolve these points of failure. The term *down* can refer to powered off, unreachable via the network or in a halted state (locked up or crashed).

Each of the three sites have a simple requirement, they require access to the Internet. As a result the CAPs and AC allow HTTP, HTTPS, SSH, ICMP and DNS traffic only. This was achieved using an IPFW firewall ruleset, consult Appendix C. IPFW can be modified, if required, to allow additional services.

The system was deployed successfully in three days to the SLL network and encountered only minimal setbacks during deployment which were quickly resolved. The first setback was one which affected the connectivity test script which populated the upstream connectivity table on the CAPgui and ACgui. ICMP packets were being dropped by the Telkom VSAT, therefore no host on the Internet could be reached by a *ping* [67]. The VSAT Indoor Unit (IDU) was not responding to ICMP Echo Requests either. This setback was resolved by sending ICMP Echo Requests to the external interface on the Mpume server, the interface which connected to the VSAT. This had the trade-off of checking that the VSAT device was up and that the external interface on the Mpume server had connectivity to it. However, this test did not provide information as to whether one could indeed access hosts on the Internet. A solution was attempted with *traceroute [49]* which relies on UDP, however, these packets were being dropped as well. This setback could be resolved in future by modification of the script to fetch a web page, although this will add overhead (albeit minimal) bandwidth use to the already scarce resource.

The second setback was related to quotas. The Edubuntu LTSP servers, to which the thin clients connect, performs the processing and execution of applications for all the thin clients. As a result implementing host quotas would be ineffective, as all network traffic from each site would only originate from the Edubuntu server. In addition, the users understanding of quotas is limited. If user quotas were enabled, each user would have to be provided a unique username and password. This would require further training of support staff to add, edit or remove users from a federated authentication server. From first hand experience during deployment at the schools, users seldom use the same Edubuntu account and linking user quotas to the Edubuntu accounts would also be infeasible. This left the system deployment with site quotas, which were disabled for the first month of deployment in order to gather necessary bandwidth use results.

The third setback that was encountered was during the final testing phase of the deployed system, when Mpume run out of pre-paid electricity. This problem was quickly resolved by a number of cellular telephone calls and a short drive to purchase more pre-paid electricity for Mpume

The deployed system was deployed initially without any limits on bandwidth for each site. A month of monitoring data was then collected, and analysis of this data allowed the introduction of limits per site.

5.7 Results - SLL Network

The system was successfully deployed to the SLL network in 2009, and since the deployment the network has been actively used by staff and students at the three schools. The network has since been reliable, with no system-related reported network down time from the local support staff.

The results were gathered from the SLL network by using SSH to access the Mpume server. This allowed the collection of results without having to physically visit the network. The majority of the results presented, analysed and discussed are for the month of November 2009. The monthly bandwidth capacity (3 GB) was reached on 27 November 2009, which prevented further collection of results until 1 December 2009 when more bandwidth was available. The bandwidth capacity is renewed on the first day of each month by the ISP. Before the deployment of this system, there had never been a recorded incident of this nature. This fact lead to the conclusion that this deployment has provided a more robust and stable network. As this was the first recorded month of the SLL utilising its full bandwidth capacity since the VSAT was deployed in 2006, the month of November 2009 was chosen to report on.

No bandwidth was used in the first week of November 2009 as the VSAT Internet connection required repair. Due to the physical layout of the network, the results presented exclude bandwidth use by Mpume, however, it has been estimated. The Mpume server connects to the VSAT unit and therefore their Internet usage is not directed to the Squid proxy on the AC. NetFlow data was not collected from Mpume and as such no bandwidth restrictions could be enforced on this site via the AC.

As previously stated in Section 5.6, no bandwidth quotas were imposed for the first month of deployment. However, monitoring data was gathered and collected for this period to allow analysis of Internet use by the sites. Analysis also revealed which websites were responsible for the majority of bandwidth use.

Section 5.7.1 discusses the results gathered regarding Internet usage via the Squid proxy on the AC. By contrasting cache *hits* and *misses*, it was possible to determine the bandwidth saved when compared to retrieving all web content from the Internet. This section also describes how the bandwidth was used at each site. Following this, in Section 5.7.2, an investigation into the top websites for the month of November 2009 is presented. This data highlights web content which was responsible for the majority of Internet bandwidth use.

Graphs are presented in Section 5.7.3, which show network usage patterns. Lastly, in Section 5.7.4, taking the results for November 2009 into consideration, site quotas were imposed and blacklists were constructed and implemented in early December 2009.

5.7.1 Squid Use

This section provides detail regarding how the Squid caching proxy server was used, including *hits*, *misses* and bandwidth use per site. Table 5.4 depicts the use of the Squid proxy on the AC. This table presents data for November 2009 and shows the *hits*, *misses* and *all* requests via the Squid proxy. The *hits* represent web content which was served

Item	Value
Hits	833 MB
Misses	1489 MB
All	2322 MB

Table 5.4: SLL - Squid Proxy - Nov 2009

to the client from the Squid cache. The term *misses* being the web content which was retrieved from the Internet. *All* refers to the sum of all *hits* and *misses*.

Table 5.4 demonstrates that, during November 2009, there was a total of 2322 MB of web content served from the Squid proxy. 1489 MB of traffic was retrieved from the Internet, while 833 MB were served from the cache. The *hits* provided a 27.8% saving on bandwidth which would have been used to download the requested content additional times. This large number of *hits* is as a result of the browsing habits of the community members, for example, visiting the same website regularly. These bandwidth use results exclude Mpume, but considering that the total bandwidth available is 3 GB per month it can be estimated that Mpume retrieved approximately 1511 MB of web content directly from the Internet.

Figure 5.7 shows a graph from the Cacti application, this graph illustrates the usage of the Squid proxy and therefore the Internet. The figure demonstrates the times that the Internet was accessed in the month of November 2009. Figure 5.7 shows the time and requests per second, of the *hits* and *requests* via the Squid proxy. In the figure, Week 45 has no activity as the VSAT Internet connection was down, however, when the VSAT was repaired, activity increased. As seen in the figure, the majority of requests had a large component of *hits*. Also noted from this figure are the Internet usage times, all spikes of activity are present during mid-day when the schools are open. Typically schools are open between 08:00 and 15:00 on weekdays. The majority of spikes are during weekdays, with the exception of the two spikes at the end of Week 46 which were over a weekend. Figure 5.7 provides summaries of *hits* and *requests*. Over the period of a month, there were an average of 0.075 requests per second and an average of 0.022 requests being *hits* from the Squid cache.

Extending the information provided in Figure 5.7, Figure 5.8 provides the number of bytes transferred per day via the Squid proxy. Both figures show graphs for the month of November 2009. By comparing Figure 5.7 with Figure 5.8, it is possible to determine the number of bytes transferred per day in processing the Squid requests.

In addition, Figure 5.8 demonstrates the days in the week in which the Squid proxy, and thus, the Internet is used. In this figure, the highest amount of activity was on 14 November 2009, when 675 MB was transferred from the Squid proxy. This high usage correlates with the spike (third from the left) in the 6th day of Week 46 in Figure 5.7.

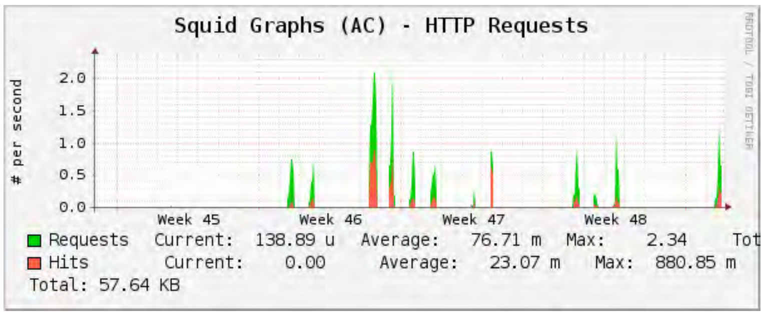


Figure 5.7: SLL - Squid - HTTP Requests - Nov 2009

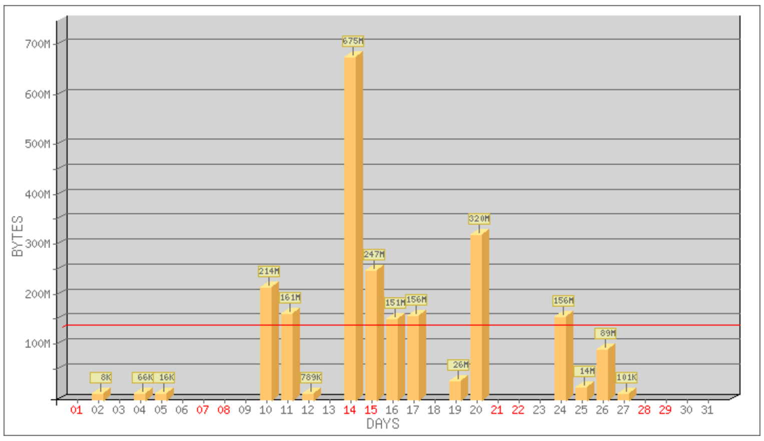


Figure 5.8: SLL - Bandwidth Use - Nov 2009

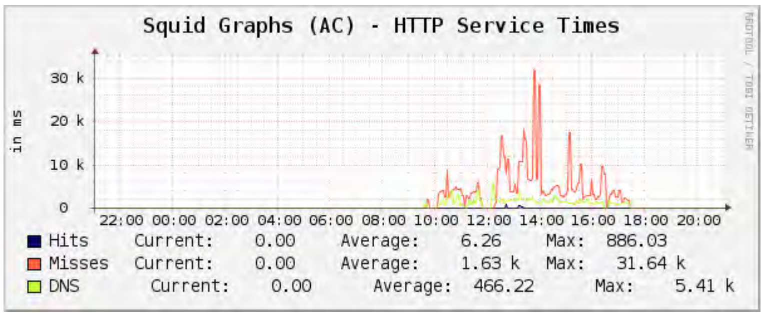


Figure 5.9: SLL - Squid - HTTP Service Times - Nov 2009

Site	Bytes	Connects
Ngwane	763 MB	83889
Mthokwane	1555 MB	87532

Table 5.5: SLL - Bandwidth Use per Site - Nov 2009

Figure 5.9 highlights 24 hours of the Squid proxy use. The graph, from Cacti, in the figure shows the time that requests took to complete. This graph is a closer look at the last spike on Figure 5.7 in Week 49. Figure 5.9 demonstrates a typical daily bandwidth use pattern for a single day, with activity between the morning and afternoon and peaking around mid-day. This graph demonstrates that *hits* were served faster from the cache than requests from the Internet. The average *hit* was retrieved from the cache in 6.26 ms whereas the average *miss* was retrieved from the Internet in 1630 ms. This shows that the experience of the user is improved when a request is retrieved from the cache because the website loads faster. The graph also displays DNS activity, where the Squid proxy would have to resolve host names to IP addresses, the average DNS request was resolved in 466.22 ms.

The results in Table 5.5 separate bandwidth use via the Squid proxy per school in the SLL network. These results take into account cache *hits*. The majority of bandwidth use originates from the Mthokwane school, with a total of 1555 MB for the month of November 2009. Ngwane used roughly half of this amount with 763 MB accounted for the month, although they generated a similar number of connection requests (“Connects”). The variance in connects between the schools is less than 5%, suggesting that the Mthokwane users visited more bandwidth intensive websites.

The following section extends these results by investigating the type of websites and web content that were responsible for bandwidth use.

5.7.2 Bandwidth Use

In this section, popularly accessed websites for the month of November 2009 are discussed. The data has been sorted by two different fields: first by the number of bytes transferred in retrieving content from the website; and secondly by the number of connects to a website. The number of connects refers to the number of requests that the Squid proxy generated to fetch content from a website.

The data has been sorted using two methods as the number of connects is not proportional to the number of bytes transferred. Table 5.6 highlights which websites used the most bandwidth, and Table 5.7 highlights which websites generated the most connects. Each table shows the top 15 websites for each category, and each table is analysed and discussed.

The results seen in Table 5.6 for the month of November 2009 are interesting, as they appear to highlight a “tragedy of the commons” scenario [40] which was discussed in Section 2.3. The web content which appears to be consuming the majority of bandwidth is pornography, The results in Table 5.6 identified that approximately 17% of the monthly bandwidth (3 GB) was utilised on pornographic material which accounted for 25% of the number of bytes via the Squid proxy. Further analysis of the raw data which generated Table 5.6 suggested that the pornographic material was being requested from Mthokwane. In Table 5.6, 7 of the top 15 websites are pornographic in nature. The other 8 websites were: a Google service used in Mozilla Firefox to aid in preventing phishing attacks; Yahoo! Image Search; an online game portal; Google Mail; Wikipedia image server; Yahoo! Mail; and Facebook. The top 15 websites, which used the majority of bandwidth for November 2009 account for 50% of the bandwidth use via the Squid proxy. It is interesting to note that the majority of these top websites are not educational.

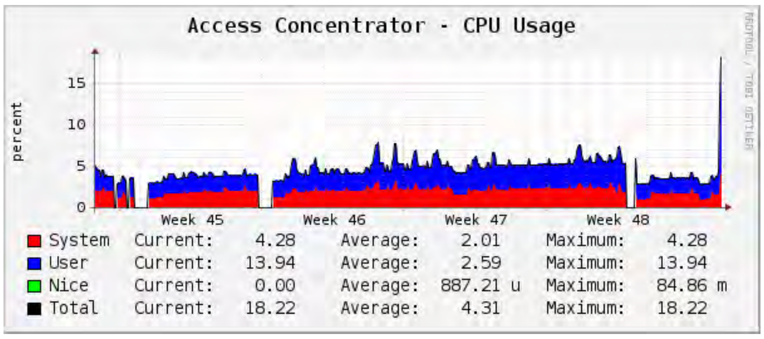
Table 5.7 provides the number of connects to a particular website from the Squid proxy for the month of November 2009. The website that generated the highest number of connects was Yahoo! Image Search. This website generated many Squid requests as it retrieved many images. Of the websites in this table, two contain pornographic material. Compared with the previous table one is lead to infer that the viewing of pornographic material was a series of isolated incidents during this reporting period and not an ongoing activity. The other websites listed in the table include: Google Mail; Facebook; Google search; the Mozilla home page; Yahoo! Mail; and Wikipedia. The regular connects to the Mozilla home page could be attributed to the default home page set in Mozilla Firefox browsers. The top 15 websites, when sorted by number of connects that are listed in Table 5.7 account for 38% of the month’s bandwidth use via the Squid proxy.

Website	Megabytes	Connects
www.gotgayporn.com	294.79 MB	6743
safebrowsing-cache.google.com	284.66 MB	8105
www.yimg.com	118.41 MB	14089
samples.blackicepass.com	78.03 MB	49
t5.cv7.org	71.70 MB	173
www.miniclip.com	52.62 MB	891
mediap.naked.com	46.77 MB	126
mail.google.com	42.29 MB	8238
realgallery.com	40.83 MB	100
upload.wikimedia.org	26.28 MB	2195
galleries.thugorgy.com	23.54 MB	40
galleries.40ozbounce.com	23.07 MB	144
m.www.yahoo.com	22.15 MB	2672
profile.ak.fbcdn.net	21.40 MB	6757
www.facebook.com	17.62 MB	4447
Other	1157.67 MB	116652

Table 5.6: SLL - Top 15 Websites - Sorted By Bytes Transferred - Nov 2009

Website	Bytes	Connects
www.yimg.com	118.41 MB	14089
mail.google.com	42.29 MB	8238
safebrowsing-cache.google.com	284.66 MB	8105
profile.ak.fbcdn.net	21.40 MB	6757
www.gotgayporn.com	294.79 MB	6743
www.facebook.com	17.62 MB	4447
clients1.google.co.za	2.23 MB	3108
www.mozilla.com	11.93 MB	2757
m.www.yahoo.com	22.15 MB	2672
us.mg4.mail.yahoo.com	6.87 MB	2383
static.ak.fbcdn.net	10.40 MB	2325
upload.wikimedia.org	26.28 MB	2195
www.google.co.za	9.32 MB	1936
b.static.ak.fbcdn.net	8.36 MB	1840
213.174.152.12	14.52 MB	1606
Other	1430 MB	102220

Table 5.7: SLL - Top 15 Websites - Sorted By Connects - Nov 2009



🟥 System	Current:	4.28	Average:	2.01	Maximum:	4.28
🟦 User	Current:	13.94	Average:	2.59	Maximum:	13.94
🟩 Nice	Current:	0.00	Average:	887.21 u	Maximum:	84.86 m
⬛ Total	Current:	18.22	Average:	4.31	Maximum:	18.22

Figure 5.10: SLL - AC - CPU Use - Nov 2009

5.7.3 System Use Graphs

This section presents three Cacti graphs from the deployed devices in the SLL network. The graphs have areas where data is missing and this is attributed to either those devices being powered off or unreachable. By looking at the data between the gaps in the graphs, it was possible to understand the times when the network or Internet was probably being used.

Figure 5.10 displays a graph for the AC's CPU Usage over the month of November 2009. First, note that the device is seldom switched off. CPU Usage of the AC is minimal, with an average usage of 4.31% and a maximum peak value of 18.22%. These results indicate that the AC has sufficient capacity for additional CAPs.

In Figure 5.11, the load averages for the Ngwane CAP have been graphed. Unlike the AC, this graph has many areas where the load averages were zero as the device was powered on in the morning and off in the late afternoon each day. It is interesting to note that the majority of use is during the week, with the exception of Week 46 where the device was powered on over the weekend. The graph highlights the Unix load averages in intervals of 1, 5 and 15 minutes. A load average is the number of processes waiting for the CPU averaged over a set time period. In Figure 5.11, the summaries show that the load averages are low, concluding that the CPU is underloaded. This indicates that the hardware is able to adequately meet the demands of the users, the various applications and the background scripts and has the capacity for expansion.

Figure 5.12 is a graph of the ping latency from the AC to the Mthokwane CAP. For this graph, it can be noted that there are periods where the ping latency was infinite as a result of CAP being powered on and off each day. The data for this graph is obtained by using ICMP Echo Requests and measuring the Round Trip Time (RTT). The RTT is a measure of the latency on the network segment, and the lower the value the more responsive the network appears. The latency shown in this graph is relatively low for a wireless link, with no indication of much variation. The average RTT was 66.42 ms.

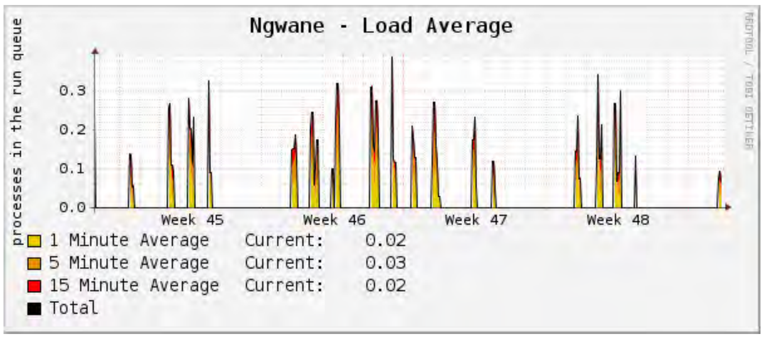


Figure 5.11: SLL - Ngwane - Load Average - Nov 2009

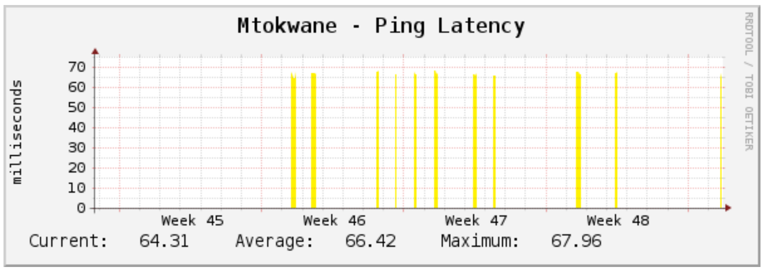


Figure 5.12: SLL - Mthokwane - Ping Latency - Nov 2009

5.7.4 Bandwidth Use Restriction Imposed

Taking into account the monitoring and graphing data that was presented and analysed, restrictions on bandwidth use for the SLL network were imposed. It was decided that site quotas should be implemented and these restrictions are in place as of early December 2009. Both sites, Ngwane and Mthokwane, were provided with the same bandwidth restrictions over the same period. The threshold for *Slight Delay* was set at 900 MB and *Significant Delay* at 1250 MB. These values allowed the schools to continue to use the Internet after they had depleted 1/3rd of the total monthly bandwidth of the VSAT, albeit at a limited throughput rate. The period was set at one month, as the bandwidth capacity of the VSAT Internet connection renews on the first day of each month. By imposing limits of use on the shared Internet connection, future months will hopefully not reach the monthly bandwidth capacity imposed by the ISP.

In an educational setting, access to pornographic websites is questionable and a policy should be formed regarding access to that kind of material. It was decided to blacklist the pornographic websites that were seen in the log files, however, this is a very small portion of the known pornographic content available on the Internet. Freely available blacklists in specific categories are available, however, the pornographic categories can include over 750,000 known websites [85]. Implementing this blacklist into Squid would increase CPU and memory usage on the AC and decrease response times for requests.

5.8 Reflections on SLL

In the month of November 2009, the bandwidth capacity of 3 GB was reached in the SLL network. This incident was the first of its nature since the deployment of the VSAT at Mpume in 2006 [59]. As this incident occurred after the system deployment, it leads one to conclude that the Internet is being better utilised as the network is more reliable than it was in the past. The results confirm that the network and the Internet have been used regularly by the three schools in the Dwesa-Cwebe community. In addition, telephone and email conversations with staff at the schools confirm that the community is pleased to have a reliable system to access the Internet whenever they require it.

This system facilitates a network and Internet connection which can be reliably used by community members. Building on this reliability, educators at the schools can structure courses for students to gain experience in using the Internet. Educators and students can further their knowledge by accessing the plethora of information available on the Internet. The Internet also provides methods to contact or collaborate with people who are elsewhere in South Africa, in Africa or in the rest of the world.

In terms of bandwidth management, it was decided that quotas would be the best method of managing bandwidth use and allocation. However, a number of difficulties arose during its execution. For example, user quotas were disabled due to limited user education. Implementing user quotas would have involved providing every user who required access to the Internet with a unique username and password. This would require training of support staff to add, remove and modify user accounts. It would also require training of the users to enter their unique username and password each time they wished to access the Internet. Experience obtained during deployment in the community aided in understanding how the computer labs were used, and users would seldom utilise the same Edubuntu LTSP account. The passwords for the LTSP user accounts were typically the same as the username to prevent users forgetting their passwords. User quotas would provide the community with better use of their Internet bandwidth, however, it would add an extra level of complexity for support staff and users.

Furthermore, the ability to impose host quotas was impaired due to each school using an Edubuntu LTSP server to boot their thin clients. As all applications and processes are performed on the LTSP server on behalf of the thin clients, all external network traffic originates from the LTSP server. In the case of many thin clients simultaneously using the Internet, all traffic will appear to originate only from the LTSP server. LTSP in community networks such as SLL, place an emphasis on providing usable services at a reduced cost, as the thin clients can be low cost or low performance PCs. However, this method of providing a school PC lab results in a limitation of not being able to implement host based quotas.

As such, site quotas that were enforced to limit each site to acceptable levels of bandwidth use. It is hoped that these measures will help prevent the bandwidth capacity from being reached before the end of the month, allowing users access to the Internet whenever it is required. In the example of the bandwidth capacity being exhausted before the end of the month, this lead to a period of time where legitimate Internet access was curbed. In this case, the educators in the schools were unable to structure computer learning courses or rely on the Internet connection for communication.

The technology and controls which have been put in place attempt to try and reduce Internet access to material which is deemed inappropriate. However, even with these controls in place users could still access this material by using websites which will proxy the connection for them[5]. A long term solution would be to implement an Acceptable Use Policy (AUP) for the PC labs and all users which require Internet access. The social effects of implementing an AUP would require users to understand what constitutes acceptable Internet content and recognise this before they are granted access to the network.

This research project recommends that the SLL network layout is restructured, two recommendations are provided. First, the SLL network is currently provided its Internet connection via the Mpume server which is connected to the VSAT. The AC provides Ngwane and Mthokwane with access to the Internet which is routed via the Mpume server. Ideally, the AC should connect to the VSAT and provide all the schools with Internet access, this will provide a single method for all schools to gain Internet access. This recommended network restructuring would allow bandwidth restrictions to be imposed on Mpume's Internet usage. Currently, Mpume do not route their traffic via the AC to gain Internet access, and in this case this could lead to Ngwane and Mthokwane being without Internet connectivity should Mpume use too much bandwidth.

The second recommendation would be to migrate the VSAT Internet connection and AC from Mpume to Ngwane which hosts the WiMAX base station. The migration would lead to a single site, Ngwane, which would provide the WiMAX connectivity and Internet connectivity to Mpume and Mthokwane. Thereby merging the physical and logical layouts of the network.

5.9 Summary

This chapter discussed how the virtual deployment of the system was extracted and deployed to physical PCs. Using g4u, images of the virtual machines were created, deployed and finalised for further deployments to a number of physical PCs. The PC which was

[5]A Bypass - https://www.abypass.info/

chosen for use as the AC and CAP were low-cost PCs, with an Intel Atom N270 1.6 GHz processor and 1 GB of RAM. The results revealed that the PCs performed sufficiently for the services they provided and had adequate room for expansion.

A deployment strategy was constructed before deployments to aid in gaining further understanding of the (then) existing deployment network. The strategy involved understanding: how the network was provided with Internet access; what type of connection medium was used to connect the various sites; and further research into which services the network provided and the future services it required. The strategy also called for the creation of a deployment procedure, which was constructed before physical deployment at the two research networks.

Each of the deployment networks were discussed according to the deployment strategy, along with their deployment procedure and the system deployment. First, the system was successfully deployed to the SCW portion of the CoE testbed network. From the technical and network troubleshooting knowledge gained from the system's deployment to SCW, the system was successfully deployed to the SLL network.

Results were discussed and analysed for each network deployment. The results enabled a better understanding of how the Internet was being used on each network. Results obtained from the SCW deployment refined the system for the SLL network.

The SLL network had no system in place to allow each school to troubleshoot its connectivity, external assistance was always required. Results obtained from the SLL network led to the conclusion that this system has made the network more robust and less likely to encounter failure. The monthly Internet bandwidth capacity for the VSAT Internet connection at the SLL is 3 GB. Before this system was deployed, it had never been recorded that the bandwidth limit had been reached. Analysis of the SLL top websites for the month of November 2009 was alarming, as the majority of bandwidth had been expended on pornographic web content. This led to the construction and implementation of a blacklist to prevent access to web content of that nature in an educational setting. Site quotas were imposed to prevent a situation where Internet bandwidth capacity reaches its limit before the end of the month.

The system was deployed successfully to the two research networks. The results that were obtained lead to the conclusion that the project was a success and was more robust than the previous systems. The next chapter concludes the dissertation.

Chapter 6

Conclusion

This chapter concludes the dissertation and provides a concise summary of each chapter. Highlighted in this chapter are the key points which attributed to the conclusions that were drawn during the research. After this summation is a revisit of the project goals that were formulated in Chapter 1. The outcomes of those goals are discussed. Following from this is a section on future research projects which could extend this work. Finally, a conclusion to the dissertation is presented.

This project combined numerous open source applications and tools to construct a two part system with custom web based Graphical User Interfaces (GUI). The GUIs allowed easy management of the two system devices, the Community Access Point (CAP) and the Access Concentrator (AC). The system was successfully deployed to two networks, peri-urban and rural, and results were gathered. The results that were gathered led to the conclusion that the system deployments were a success and that the system was robust. The system could be used by administrators to equitably manage and control access to the Internet on a number of levels. By using this system to manage bandwidth, an administrator could ensure fair use of the shared Internet connection.

6.1 Summary

This section presents an overview of each chapter and the conclusions that were drawn from them.

The dissertation began with an introduction to the problem domain in Chapter 1. Community networks in South Africa and other African countries are often serviced by limited bandwidth capacity Internet connections. The chapter presented the problem statement which highlighted that a need for a system to manage and monitor bandwidth use was required in community networks.

Chapter 2 discussed Internet usage in South Africa along with reasoning as to why a demand existed for fair management and monitoring of community networks and their bandwidth. Network applications, tools and relevant hardware were identified and discussed and these items were presented alongside each other to aid the reader in understanding how the different items could be integrated together to form a final package. This chapter introduced the two community research networks that were used as system deployment sites. Previous work in the domain was discussed to gain a greater understanding of what was required for a network management and monitoring package.

Chapter 3 described the design of the system that was used to manage and monitor bandwidth use at the previously identified community networks. The design is a high level view of the system in terms of requirements and components needed. The design called for a two-part solution in the form of client-server architecture. The server, the AC, was designed to be the Internet gateway router for the community network and act as a centralised management station collecting data from all the CAPs. The CAPs (the clients) were designed as routers for each network site, not only routing traffic but also managing their site and collecting monitoring data to pass onto the AC. Both devices required a GUI in order to configure the device, view monitoring data and enforce bandwidth restrictions.

Chapter 4 used the design discussed in Chapter 3 to implement the CAP and AC. The FreeBSD OS was chosen as the basis for the CAP and AC. FreeBSD is a robust and widely used OS for networked devices. The devices implemented numerous free and open source applications and tools, including Squid Caching Proxy Server, Cacti, SNMP, NetFlow and LightSquid. Each device had a custom PHP based web front-end for management and viewing of monitoring data. Before the system was deployed to physical networks a virtual development environment was constructed. This allowed quick prototyping and experimentation to further the construction of a more robust system.

Chapter 5 discussed the successful deployment of the system to the two community networks. This chapter explains the results which were obtained. The results were analysed and bandwidth restrictions were imposed for one of the research networks. The results revealed the project system deployed in the research networks improved network usability as they were more robust than the previous system. No system related network down time was reported during the data gathering for this project.

6.2 Research Goals Revisited

In Chapter 1 four primary and two secondary goals for the project were listed. This section discusses how those goals were met and reviews the outcomes of achieving those goals. The outcomes of the primary goals are as follows:

- Network management and monitoring was investigated and numerous applications and tools were identified and discussed. These applications and tools were presented in Chapter 2.

- Further details of specific applications are discussed in Chapters 4 and 5 as they were chosen for inclusion in the system's implementation. The system was successfully designed, implemented and deployed to meet the requirements of the problem statement. The system was composed of two devices, the AC and the CAP, with their relevant GUIs: the ACgui and CAPgui. The GUIs allowed a simple method to manage the devices without having specific knowledge of the underlying OS, FreeBSD.
- The system employed various methods to monitor bandwidth use by the different sites within a community network. Using this monitoring information, administrators could enforce bandwidth use restrictions per user, per host and per site. Restrictions could impose limits on throughput as bandwidth use increases. Configuration of the devices in the system was performed via their GUIs. Restrictions could also be imposed on types of Internet content that users can view or download. Websites can be added to a blacklist to prevent users from viewing specific websites.
- Initial implementation and experimentation of the system was conducted in a virtual development environment, using VMware Workstation. The system was then successfully deployed on physical PCs to the two research networks and monitoring data was collected, analysed and discussed. Analysis of the data confirmed that the system was more reliable and usable than the previous system.

Along with the success of the primary goals being achieved, the secondary goals outcomes were as follows:

- The low-cost PCs which were used in the system deployment had an Intel Atom 1.6 GHz CPU and 1 GB of RAM. The hardware specifications of the PCs were more than adequate at performing the tasks and services of the AC and CAP.
- Two user manuals were produced to aid administrators using the CAP and AC. The user manuals were written in simplified terms to allow an administrator of limited networking knowledge to understand how to perform everyday tasks.

6.3 Future Extensions

The system employed a manual method for an administrator to allocate bandwidth to users, hosts and sites. Extensions on this manual method for setting limits could include a dynamic bandwidth allocator. A dynamic system could take inputs from the monitored data and provide administrators with suggestions for thresholds on limits. The thresholds could be suggested by taking into account the previous months bandwidth use and by forming a usage pattern.

Taking into account the results that were gathered in Chapter 5 the majority of bandwidth was being utilised at the Siyakhula Living Lab (SLL) network by community members viewing pornographic material. The SLL network is composed of schools, with staff and students using the Internet, therefore it is an educational location. An Acceptable Use Policy (AUP) could be constructed for the network which would have to be acknowledged by the users of the network before they are granted access to any PCs or Internet access. The AUP can clearly state what the PCs may or may not be used for. The implementation of such a policy could allow more effective use of bandwidth and allow further restrictions on access to material which is found to be inappropriate in such a location.

6.4 Conclusion

This project investigated the methods and procedures available in managing and monitoring the limited quantity of Internet bandwidth available to community networks. A simple solution would be to upgrade the Internet connection to allow more bandwidth, however, the costs of Internet connections in South Africa are relatively expensive when compared to developed countries. The cost of an Internet connection is also relatively expensive when compared to the basic needs of a community.

Two community research networks (with limited and shared Internet connections) were found to lack effective management and monitoring systems, specifically in terms of bandwidth use restrictions. The primary outcome of this project was to integrate multiple tools and applications to effectively manage and monitor bandwidth in community networks. The system that was constructed allowed an equitable distribution of bandwidth. Restrictions on bandwidth use could be imposed per user, host and sites over a specific period of time.

The results that were gathered concluded that the system is a workable solution to the limited capacity of bandwidth in community networks with a shared Internet connection.

Reference List

[1] Africa, S. Seacom, Interoute team up. Online: http://www.itweb.co.za/sections/telecoms/2009/0904291040.asp?A=COV&S=Cover, Accessed: 05/05/2009, April 2009.

[2] Aitchison, R. *Pro DNS and BIND.* Apress, 2005.

[3] Albitz, P., and Liu, C. *DNS and BIND.* O'Reilly, 2006.

[4] Antsilevich, U. J., Kamp, P.-H., Nash, A., Cobbs, A., and Rizzo, L. *ipfw(8) - IP firewall and traffic shaper control program.* FreeBSD 6.2, 2007.

[5] Badger, M. *Zenoss Core - Network and System Monitoring.* Packt Publishing, 2008.

[6] Barth, W. *Nagios - System and Network Monitoring.* No Starch Press, 2008.

[7] Berners-Lee, T., Fielding, R., and Masinter, L. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986 (Standard), Jan. 2005.

[8] Beyleveld, P. Implementation and testing of WiMAX wireless network technology. Computer Science Honours thesis, Rhodes University, Grahamstown, South Africa, 2006.

[9] Brandt, H. *bsnmpd(1) - simple and extensible SNMP daemon.* FreeBSD 6.2, 2006.

[10] Brandt, I. Models of Internet connectivity for secondary schools in the Grahamstown Circuit. Master's thesis, Rhodes University, Jan 2006.

[11] Brandt, I., Terzoli, A., and Hodgkinson-Williams, C. Wireless Communication for Previously Disadvantaged Secondary Schools in Grahamstown, South Africa. in SATNAC 2005, Convergence - Can technology deliver?, Sept 2005.

[12] Business Leadership South Africa. South African telecommunications prices: An updated international price comparison, with regulatory recommendations. Occasional Paper No 3, Nov 2007.

[13] Canonical Ltd. UsingEdubuntu. Online: http://edubuntu.org/UsingEdubuntu, Accessed: 27/11/2009, 2009.

[14] Case, J., Fedor, M., Schoffstall, M., and Davin, J. Simple Network Management Protocol (SNMP). RFC 1098, Apr. 1989. Obsoleted by RFC 1157.

[15] Case, J., Fedor, M., Schoffstall, M., and Davin, J. Simple Network Management Protocol (SNMP). RFC 1157 (Historic), May 1990.

[16] Cisco Systems Inc. Comparing Traffic Policing and Traffic Shaping for Bandwidth Limiting. Online: `http://www.cisco.com/application/pdf/paws/19645/policevsshape.pdf`, Accessed: 29/05/2009, 2005.

[17] Cisco Systems, Inc. Introduction to Cisco IOS NetFlow: Technical Overview. Online: `http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.pdf`, Accessed: 26/05/2009, 2007.

[18] Cisco Systems, Inc. Cisco IOS Flexible NetFlow Technology: Data Sheet. Online: `http://www.cisco.com/en/US/products/ps6965/products_ios_protocol_option_home.html`, Accessed: 26/05/2009, 2008.

[19] Claffy, K. C., Braun, H., and Polyzos, G. C. A Parameterizable Methodology for Internet Traffic Flow Profiling. *IEEE Journal on Selected Areas in Communications 13* (1995), 1481–1494.

[20] Claise, B. Cisco Systems NetFlow Services Export Version 9. RFC 3954 (Informational), Oct. 2004.

[21] Deering, S., and Hinden, R. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460 (Draft Standard), Dec. 1998. Updated by RFC 5095.

[22] Egevang, K., and Francis, P. The IP Network Address Translator (NAT). RFC 1631 (Informational), May 1994. Obsoleted by RFC 3022.

[23] Erokhin, S. LightSquid Home Site. Online: `http://lightsquid.sourceforge.net/`, Accessed: 12/05/2008, 2009.

[24] Feyrer, H. g4u - Harddisk Image Cloning for PCs. Online: `http://www.feyrer.de/g4u/`, Accessed: 19/11/2009, 2009.

[25] Flick, J. IEEE 802.12 Interface MIB. RFC 2020 (Proposed Standard), Oct. 1996.

[26] FreeBSD. About FreeBSD Ports. Online: `http://www.freebsd.org/ports/`, Accessed: 26/04/2009, 2006.

[27] FreeBSD. *ifconfig(8) - configure network interface parameters.* FreeBSD 6.2, 2007.

[28] FreeBSD. FreeBSD 6.3-RELEASE. Online: `http://www.freebsd.org/releases/6.3R/announce.html`, Accessed: 26/04/2009, 2008.

[29] FreeBSD 6.2. *natd(8) - Network Address Translation daemon*, 2003. Archie Cobbs, Charles Mott, Eivind Eklund, Ari Suutari, Dru Nelson, Brian Somers and Ruslan Ermilov.

[30] FreeBSD 6.2. *df(1) - display free disk space*, April 2004.

[31] FreeBSD 6.2. *ps(1) - process status*, August 2006.

[32] Friedl, J. E. F. *Mastering Regular Expressions, 3rd Edition.* O'Reilly, 2006.

[33] Frye, R., Levi, D., Routhier, S., and Wijnen, B. Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework. RFC 3584 (Best Current Practice), Aug. 2003.

[34] Gill, P., Arlitt, M., Li, Z., and Mahanti, A. YouTube Traffic Characterization: A View From the Edge. Internet Measurement Conference, 2007.

[35] GNU Bash-4.0. *bash(1) - GNU Bourne-Again SHell*, February 2009.

[36] Google. Google Maps. Online: `http://maps.google.co.za`, Accessed: 09/12/2009, 2009.

[37] Gregg, M., and Watkins, S. *Hack the Stack: Using Snort and Ethereal to Master the 8 Layers of an Insecure Network.* Syngress, 2006.

[38] Halse, G. A., and Terzoli, A. Open Source in South African Schools: Two Case Studies. Online: `http://eprints.ru.ac.za/100/01/HALSE-Highway-Africa-2002.pdf`, Accessed: 12/06/2008, 2002.

[39] Hansteen, P. N. *The Book Of PF.* No Starch Press, Inc., 2008.

[40] Hardin, G. The Tradegy of the Commons. *Science Vol. 162. no 3859.* (1968), Pages 1243–1248.

[41] Harrington, D., Presuhn, R., and Wijnen, B. An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. RFC 3411 (Standard), Dec. 2002. Updated by RFC 5343.

[42] Hunter, D., Rafter, J., Fawcett, J., van der Vlist, E., Ayers, D., Duckett, J., Watt, A., and McKinnon, L. *Beginning XML.* Wiley Publishing, Inc., 2007.

[43] IANA. Special-Use IPv4 Addresses. RFC 3330 (Informational), Sept. 2002.

[44] International Telecommunication Union (ITU). Information Technology - Open Systems Interconnection - Basic Refernce Model: The Basic Model. Online: `http://www.itu.int/rec/T-REC-X.200-199407-I/en`, Accessed: 14/11/2009, August 1994.

[45] Internet Systems Consortium (ISC). *named(8) - Internet domain name server.* BIND 9, 2000.

[46] Internet Systems Consortium (ISC). ISC BIND. Online: `https://www.isc.org/software/bind`, Accessed: 13/11/2009, 2009.

[47] Internet World Stats. Internet Usage Statistics for Africa. Online: `http://www.internetworldstats.com/stats1.htm`, Accessed: 20/11/2008, January 2009.

[48] Irwin, B. V. W. Bandwidth Management and Monitoring for IP Network Traffic: An Investigation. Master's thesis, Rhodes University, 2001.

[49] Jacobson, V. *traceroute(8) - print the route packets take to network host.* 4.3 Berkeley Distribution, 2000.

[50] Juma, C., and Moyer, E. Broadband Internet for Africa. *Science Vol. 320. no 5881.* (2008), Page 1261.

[51] Kayle, A. Seacom to lower African bandwidth costs. Online: `http://www.itweb.co.za/sections/computing/2009/0903250905.asp?S=All%20Africa%20News&A=AFN&O=google`, Accessed: 05/05/2009, 2009.

[52] Korff, Y., Hope, P., and Potter, B. *Mastering FreeBSD and OpenBSD Security.* O'Reilly, 2005.

[53] Lighttpd. Lighttpd - Fly Light. Online: `http://www.lighttpd.net/`, Accessed: 11/08/2009, 2009.

[54] Lucas, M. W. Monitoring Network Traffic with Netflow. Online: `http://www.onlamp.com/pub/a/bsd/2005/08/18/Big_Scary_Daemons.html`, Accessed: 21/02/2009, 2005.

[55] Lucas, M. W. *Absolute FreeBSD.* No Starch Press, 2007.

[56] m0n0wall. m0n0wall. Online: `http://m0n0.ch/wall/`, Accessed: 05/06/2009, 2009.

[57] Makeni, J. Will Africa join broadband revolution? Online: `http://news.bbc.co.uk/2/hi/africa/7987812.stm`, Accessed: 24/05/2009, April 2009.

[58] Mamakos, L., Lidl, K., Evarts, J., Carrel, D., Simone, D., and Wheeler, R. A Method for Transmitting PPP Over Ethernet (PPPoE). RFC 2516 (Informational), Feb. 1999.

[59] Mandioma, M. Rural Internet Connectivity: A Deployment in Dwesa-Cwebe, Eastern Cape, South Africa. Master's thesis, University of Fort Hare, 2007.

[60] Mandioma, M., T., Rao, G., Terzoli, A., and Muyingi, H. Deployment of WiMAX for Telecommunication and Internet Access in Dwesa-Cwebe Rural Areas, 2007.

[61] Marsan, C. D. MySpace Threatening Net Bandwidth. Online: `http://www.pcadvisor.co.uk/news/index.cfm?newsid=9839`, Accessed: 07/04/2008, 2007.

[62] Mauro, D., and Schmidt, K. *Essential SNMP, 2nd Edition.* O'Reilly, 2006.

[63] Microsoft Corporation. SNMP Service. Online: `http://msdn.microsoft.com/en-us/library/aa379100(VS.85).aspx`, Accessed: 13/11/2009, March 2009.

[64] Microsoft Corporation. *Tracert*, 2009. http://www.microsoft.com/resources/documentati
us/tracert.mspx.

[65] Mockapetris, P. Domain names - concepts and facilities. RFC 1034 (Standard), Nov. 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 2065, 2181, 2308, 2535, 4033, 4034, 4035, 4343, 4035, 4592.

[66] Mockapetris, P. Domain names - implementation and specification. RFC 1035 (Standard), Nov. 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 1995, 1996, 2065, 2136, 2181, 2137, 2308, 2535, 2845, 3425, 3658, 4033, 4034, 4035, 4343.

[67] Muuss, M. *ping(8) - send ICMP ECHO_REQUEST packets to network hosts.* FreeBSD 6.2, 2006.

[68] MyBroadBand. South African bandwidth pricing kills broadband joy. Online: `http://mybroadband.co.za/nephp/?m=show&id=4435`, Accessed: 20/05/2009, 2006.

[69] NeDi. NeDi - Network Discovery and Monitoring. Online: `http://www.nedi.ch/about`, Accessed: 10/06/2009, 2009.

[70] Net-SNMP. *variables(5) - Format of specifying variable names to SNMP tools.* 4th Berkeley Distribution, 1999.

[71] Net-SNMP. *snmpd(8) - daemon to respond to SNMP request packets.* 4th Berkeley Distribution, 2005.

[72] Net-SNMP. Net-SNMP. Online: `http://www.net-snmp.org/`, Accessed: 03/02/2009, 2007.

[73] pfflowd. NetFlow probe for PF packet filter. Online: `http://www.mindrot.org/projects/pfflowd/`, Accessed: 26/05/2009, 2006.

[74] pfSense. pfSense - Open Source Firewall Distribution. Online: `http://www.pfsense.org/`, Accessed: 05/06/2009, 2009.

[75] Plonka, D. FlowScan: A Network Traffic Flow Reporting and Visualization Tool. LISA '00: Proceedings of the 14th USENIX conference on System administration, 2000.

[76] Postel, J. Internet Control Message Protocol. RFC 792 (Standard), Sept. 1981. Updated by RFCs 950, 4884.

[77] Presuhn, R. Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP). RFC 3416 (Standard), Dec. 2002.

[78] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G. J., and Lear, E. Address Allocation for Private Internets. RFC 1918 (Best Current Practice), Feb. 1996.

[79] Rizzo, L. *dummynet(4) - Traffic shaper, bandwidth manager and delay enumerator.* FreeBSD 6.2, 2002.

[80] Rizzo, L. Dummynet: A Simple Approach to the Evaluation of Network Protocols. *ACM Computer Communication Review* (Jan, 1997).

[81] Rosenberg, R. S. Controlling Access to the Internet: The Role of Filtering. *Ethics and Information Technology 3*, 1 (2001), 35–54.

[82] RRDTool. RRDTool Logging & Graphing. Online: `http://www.rrdtool.org/`, Accessed: 11/06/2008, 2008.

[83] Saunders, J. *tproxy(8) - transparently re-direct HTTP requests to a HTTP cache.* Northlink Communications Pty Ltd., 2000.

[84] SEACOM. Connecting Africa to the World. Online: `http://www.seacom.mu/`, Accessed: 22/01/2009, 2009.

[85] Shalla Secure Services. Shalla's blacklists. Online: `http://www.shallalist.de/`, Accessed: 10/03/2008, 2008.

[86] Shuttleworth Foundation. Telecommunications. Online: `http://www.shuttleworthfoundation.org/our-work/telecommunications`, Accessed: 20/05/2009, 2009.

[87] Siebörger, I., and Terzoli, A. Field testing the Alvarion BreezeMAX as a last mile access technology. 10th Annual Southern African Telecommunication Networks and Applications Conference (SATNAC) 2007, Sept 2007.

[88] Siebörger, I., Terzoli, A., and Hodgkinson-Williams, C. The development of ICT networks for South African schools: Two pilot studies in disadvantaged areas. Learning to Live in the Knowledge Society, the TC3 Conference in WCC 2008, Milan, Sept 2008.

[89] Simpson, W. The Point-to-Point Protocol (PPP). RFC 1661 (Standard), July 1994. Updated by RFC 2153.

[90] Siyakhula Living Lab. We are growing together. Online: `http://www.dwesa.org/`, Accessed: 05/05/2009, 2009.

[91] SME Survey 2008. The Findings. Online: `http://www.bizassist.co.za/images/SME_%20Survey_Findings.ppt`, Accessed: 18/11/2008, Oct 2008.

[92] Softflowd. Fast Software NetFlow Probe. Online: `http://www.mindrot.org/projects/softflowd/`, Accessed: 5/5/2009, 2006.

[93] Somers, B. *pppoed(8) - handle incoming PPP over Ethernet connections.* FreeBSD 6.2, 1999.

[94] Sommer, R., and Feldmann, A. NetFlow: Information loss or win? Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement, 2002.

[95] South African Foundation. Telecommunication prices in South Africa: An international peer group comparison. Occasional Paper No 1, April 2005.

[96] Squid. squid: Optimising Web Delivery. Online: `http://www.squid-cache.org/`, Accessed: 10/02/2008, 2007.

[97] Srisuresh, P., and Egevang, K. Traditional IP Network Address Translator (Traditional NAT). RFC 3022 (Informational), Jan. 2001.

[98] Stallings, W. SNMP and SNMPv2: The Infrastructure for Network Management. IEEE Communications Magazine, Mar 1998.

[99] Stallings, W. SNMPv3: A Security Enhancement for SNMP. IEEE Communications Surveys & Tutorials, 1998.

[100] Statistics South Africa. Income and Expenditure of Households 2005/2006. Online: `http://www.statssa.gov.za/Publications/P0100/P01002005.pdf`, Accessed: 20/05/2009, 2008.

[101] Statistics South Africa. Mid-year Population Estimates. Online: `http://www.statssa.gov.za/publications/P0302/P03022009.pdf`, Accessed: 13/11/2009, July 2009.

[102] Stenberg, D. *curl(1) - transfer a URL.* Curl 7.18.0, 2008.

[103] Stenberg, D. cURL and libcurl. Online: `http://curl.haxx.se/`, Accessed: 25/05/2009, 2009.

[104] Stevens, W. R. *TCP/IP Illustrated, Volume 1: The Protocols.* Addison-Wesley, 1996.

[105] Technology Concepts. Bandwidth Management. Online: `http://www.techconcepts.co.za/solutions.php?solutionsid=4`, Accessed: 20/11/2008, 2008.

[106] The Apache Software Foundation. Apache - HTTP Server Project. Online: `http://httpd.apache.org/`, Accessed: 08/11/2009, 2009.

[107] The Cacti Group. Cacti: The Complete RRDTool-Based Graphing Solution. Online: `http://cacti.net/`, Accessed: 10/02/2008, 2007.

[108] The FreeBSD Documentation Project. FreeBSD Handbook. Online: `http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/`, Accessed: 26/04/2009, 2009.

[109] The PHP Group. PHP: Hypertext Preprocessor. Online: `http://php.net/`, Accessed: 09/11/2009, 2009.

[110] Turnbull, J. *Pro Nagios 2.0.* Apress, 2006.

[111] Ubuntu. Ubuntu Homepage. Online: `http://www.ubuntu.com/`, Accessed: 26/04/2009, 2009.

[112] VMware, Inc. VMware Workstation 6 - Product Datasheet. Online: `http://www.vmware.com/files/pdf/ws_datasheet.pdf`, Accessed: 26/04/2009, 2007.

[113] W3C. XHTML 1.0 The Extensible HyperText Markup Language (Second Edition). Online: `http://www.w3.org/TR/xhtml1/`, Accessed: 10/11/2009, August 2002.

[114] W3C. Cascading Style Sheets(CSS). Online: `http://www.w3.org/Style/CSS/`, Accessed: 10/11/2009, 2009.

[115] Waldbusser, S., and Grillo, P. Host Resources MIB. RFC 2790 (Draft Standard), Mar. 2000.

[116] Wells, D. D. IEEE 802.11 Signal Source Mapping using Low Cost Spectrum Analysers. Rhodes University, Department of Computer Science, 2007.

[117] Wessels, D. *Squid: The Definitive Guide*. O'Reilly, 2004. ISBN: 0-596-00162-2.

[118] Whittington, B., Halse, G., and Terzoli, A. Secure, extensible and heterogenic wireless networks: A model for community orientated wireless Internet in South Africa. Computer Science Honours thesis, Rhodes University, Grahamstown, South Africa, 2003.

[119] Wilson, E. G. Network Management Station. Online: `http://www.nmsworld.com/`, Accessed: 10/12/2009, 2007.

[120] Ylonen, T. *sshd(8) - OpenSSH SSH daemon*. FreeBSD 6.2, 1999.

[121] Zenoss Inc. Zenoss Core Product Overview. Online: `http://www.zenoss.com/product/network-monitoring`, Accessed: 28/05/2009, 2009.

Appendix A

SNMP - Useful OIDs

This appendix breifly desribes some SNMP OIDs and MIBs that were used in the implementation of the project system.

A.1 General MIBs

In this section, general MIBs and how they were used in the project system are briefly described. Sample output has also been provided. In Table A.1, two MIBs are presented, for monitoring system uptime and the host name.

In Table A.2, a number of MIBs are presented for collecting data from the various network interfaces. Monitored data includes: description, media type, MAC address, operating status, bytes in, bytes out, IP address and netmask. This data was retrieved to populate the table on the *Interfaces Status* page on the CAPgui and ACgui.

Table A.3 provides the MIBs for identifying the status of processes running on the monitored device. This data was retrieved from the CAPs to populate the table in the *CAP Services* page on the ACgui.

A.2 Squid Proxy Server OIDs/MIBs

This section describes two SNMP MIBs used by the ACgui to populate the Squid Status page. Two MIBs were retrieved: one providing the total size of the cache; and the other providing the amount of cache space used.

OID	MIB	Return Type
.1.3.6.1.2.1.1.3.0	DISMAN-EVENT-MIB::sysUpTimeInstance	Timeticks: (70427) 0:11:44.27
.1.3.6.1.2.1.1.5.0	SNMPv2-MIB::sysName.0	STRING: blackspy.ict.ru.ac.za

Table A.1: System MIBs

OID	MIB	Return Type
.1.3.6.1.2.1.2.2.1.2.2	IF-MIB::ifDescr.2	STRING: ext0
.1.3.6.1.2.1.2.2.1.5.2	IF-MIB::ifSpeed.2	Gauge32: 100000000
.1.3.6.1.2.1.2.2.1.6.2	IF-MIB::ifPhysAddress.2	STRING: 0:22:68:52:cb:12
.1.3.6.1.2.1.2.2.1.8.2	IF-MIB::ifOperStatus.2	INTEGER: up(1)
.1.3.6.1.2.1.2.2.1.10.2	IF-MIB::ifInOctets.2	Counter32: 387881969
.1.3.6.1.2.1.2.2.1.16.2	IF-MIB::ifOutOctets.2	Counter32: 40775870
.1.3.6.1.2.1.4.20.1.2.146.231.121.241	IP-MIB::ipAdEntIfIndex.146.231.121.241	INTEGER: 2
.1.3.6.1.2.1.4.20.1.3.146.231.121.241	IP-MIB::ipAdEntNetMask.146.231.121.241	IpAddress: 255.255.248.0

Table A.2: Network MIBs

OID	MIB	Return Type
.1.3.6.1.2.1.25.4.2.1.2.926	HOST-RESOURCES-MIB::hrSWRunName.926	STRING: softflowd
.1.3.6.1.2.1.25.4.2.1.5.926	HOST-RESOURCES-MIB::hrSWRunParameters.926	STRING: -i ext0 -n localhost:8818
.1.3.6.1.2.1.25.6.3.1.2.229	HOST-RESOURCES-MIB::hrSWInstalledName.229	STRING: squid-2.7.7
.1.3.6.1.2.1.25.6.3.1.5.229	HOST-RESOURCES-MIB::hrSWInstalledDate.229	STRING: 2009-9-30,15:8:42.0,-2:0

Table A.3: Host Resources MIBs

The two fields, their OIDs and MIBs, can be seen in Table A.4. The MIB *cacheSwapMaxSize* returns an *INTEGER* value of the number of Megabytes (MB). The MIB *cacheSysStorage* returns an *INTEGER* value of the number of Kilobytes (KB) of cache utilised.

OID	MIB	Return Type
.1.3.6.1.4.1.3495.1.2.5.2.0	SQUID-MIB::cacheSwapMaxSize.0	INTEGER
.1.3.6.1.4.1.3495.1.1.2.0	SQUID-MIB::cacheSysStorage.0	INTEGER

Table A.4: Squid Proxy Server MIBs

Appendix B

Configuration Files

This appendix describes how configuration files on the CAP and AC were generated. Details of the Squid caching proxy server's configuration file is presented and discussed.

B.1 Generating Configuration Files

The CAPgui and ACgui, as explained in Chapter 4, modified an XML configuration file. Further to this, the XML configuration file was parsed to generate a number of system and application specific configuration files.

Configuration can be modified via the GUI which will generate configuration files for DHCPd, PPPoEd, SNMPd and the Squid proxy server. The application configuration files were generated by editting a default file with specific tags to replace. For example, a unique tag such as *<hostname>* in the configuration file could be replaced with the hostname value stored in the XML configuration file.

Configuration via the CAPgui could also modify */etc/rc.conf* and */etc/resolv.conf*. The FreeBSD system configuration file */etc/rc.conf* was parsed and either updated the previous fields or added new fields for the modified configuration. The file */etc/resolv.conf* stores the IP addresses of nameservers, domains and search domains that the FreeBSD system relies on. Generating this file relied on creating a new file each time.

After modifying settings via the CAPgui or ACgui, the user must reboot the device. Before the device is rebooted, the generated configuration files overwrite the previous configuration files.

B.2 Squid

This section serves to discuss the finer points of the Squid configuration on the AC. Sections of the 'squid.conf' file are presented and discussed.

```
auth_param basic program ...
      /usr/local/libexec/squid/squid_radius_auth ...
      -f /usr/local/etc/squid/squid_radius_auth.conf

auth_param basic children 5
auth_param basic realm Access Limited
auth_param basic credentialsttl 2 hours
auth_param basic casesensitive off
...
acl radius-auth proxy_auth REQUIRED
```

Figure B.1: Squid User Authentication with FreeRADIUS (squid.conf)

B.2.1 RADIUS Authentication Helper

If user authentication is required for access to the Internet via the Squid proxy, a number of authentication methods can be used. The example in Figure B.1 demonstrates the changes to the Squid configuration to utilise a RADIUS authentication server.

Using FreeBSD, the package *squid_radius_auth* can be installed from the ports tree. Configuration of the authentication helper involves setting a server IP address and a shared key.

B.2.2 Delay Pools

Squid delay pools can be implemented to restrict throughput rates for specific users or hosts wishing to access the Internet. An example of the Squid configuration file changes can be seen in Figure B.2. Changes include configuring an ACL which contains a list of users or source IP addresses which should be restricted. The example in Figure B.2, provides two delay pools which are named *sli_delay* and *sig_delay*. In this case, source IP addresses are read from files.

The *delay_parameters* configuration directive takes the arguments: delay pool number; aggregate bucket; and individual bucket. The aggregate bucket and individual bucket arguments are split by: total number of bits available in the bucket at any given time; and the number of bits to replenish every second.

The *sli_delay* pool restricts the maximum throughput rate for all users to a maximum 128 KBps and each user to 64 KBps. The *sig_delay* pool restricts all users to a maximum throughput rate of 32 KBps and each user to 16 KBps.

```
acl sig_delay src "/usr/local/etc/squid/lists/sig_delay"
acl sli_delay src "/usr/local/etc/squid/lists/sli_delay"
...
delay_pools 2
delay_class 1 2
delay_parameters 1 16000/16000 8000/8000
delay_access 1 allow sli_delay
delay_access 1 deny all

delay_class 2 2
delay_parameters 2 4000/4000 2000/2000
delay_access 2 allow sig_delay
delay_access 2 deny all
```

Figure B.2: Squid Delay Pools (squid.conf)

```
acl blacklist dstdomain "/usr/local/etc/squid/lists/blacklist"
...
http_access deny blacklist
```

Figure B.3: Squid Blacklists (squid.conf)

B.2.3 Blacklists

Figure B.3 provides a simple method for implementing a list of domains which are to be blacklisted. The file contains a list of destination domains each on a seperate line.

Appendix C

Electronic Appendix

Various items have been provided electronically on the attached DVD:

- g4u Installation Media (including USB bootable image) and Installation Guide
- g4u Images
  - Access Concentrator (AC)
  - Community Access Point (CAP)
- Source Code
  - ACgui (including background scripts)
  - CAPgui (including background scripts and IPFW rules)
- User Manuals and Screenshots of User Interfaces
  - CAPgui and Connectivity Troubleshooting Flowchart
  - ACgui
- Photographs of SLL Deployment

Glossary of Terms

AC - Access Concentrator

ACL - Access Control List

Bandwidth - Amount of data that can or is downloaded/uploaded

Bandwidth Capacity - Maximum international and local Internet bandwidth provided from an ISP

BIND - Berkeley Internet Name Domain

CAP - Community Access Point

C-LAN - Community Local Area Network

CNC - Community Network Core

CoE - Telkom Centre Of Excellence - Grahamstown

CPE - Customer Premise Equipment

CPU - Central Processing Unit

C-WAN - Community Wide Area Network

DHCP - Dynamic Host Configuration Protocol

DNS - Domain Naming System

DSL - Digital Subscriber Line

DSLAM - DSL Access Multiplexer

FTP - File Transfer Protocol

GRE - Generic Routing Encapsulation

GUI - Graphical User Interface

HTTP - Hyper Text Transfer Protocol

HTTPS - Hyper Text Transfer Protocol Secure - an encrypted form of HTTP

ICMP - Internet Control Message Protocol

IDS - Intrusion Detection System

IP - Internet Protocol

IPFW - IP Firewall

ISDN - Integrated Services Digital Network

ISP - Internet Service Provider

Kbps - Kilobits per second - 1000 bits per second

KBps - Kilobytes per second - 1000 bytes per second

KiBps - Kibibytes per second - 1024 bytes per second

LAN - Local Area Network

MIB - Management Information Base

NAT - Network Address Translation

NIC - Network Interface Card

OID - Object Identifier

OS - Operating System

OSS - Open Source Software

PC - Personal Computer

PF - Packet Filter

PHP - Hypertext Preprocessor - A server side scripting language

PPPoE - Point-to-Point Protocol over Ethernet

QoS - Quality of Service

RAM - Random Access Memory

RFC - Request For Comment

RRA/RRD - Round-Robin Archive/Database

RTT - Round-Trip-Time - A measure of latency

SCP - Secure Copy Protocol

SCW - Settler City Wireless - a component of the CoE Network

SD - Secure Digital

SLD - Second Level Domain

SLL - Siyakhula Living Lab

SMTP - Simple Mail Transfer Protocol

SNMP - Simple Network Management Protocol

SSH - Secure SHell

TCP - Transmission Control Protocol

Throughput - A measure of the amount of data transferred in a specific amount of time

TLD - Top Level Domain

TTL - Time-To-Live

UDP - User Datagram Protocol

URL - Uniform Resource Locator

WAN - Wide Area Network

WiFi - Wireless Fidelity

WiMAX - Worldwide Interoperability for Microwave Access

WLAN - Wireless Local Area Network

WWW - World Wide Web