A REAL TIME DISTRIBUTED NETWORK MONITORING PLATFORM (RTDNM)

AHMED MANSOUR MOHAMMED MANASRAH

UNIVERSITI SAINS MALAYSIA
Thesis Submitted in fulfilment of the Requirements for the degree
Of Doctor of philosophy in
the Graduate School of Computer Science

January 2009

ACKNOWLEDGMENTS

Working as a Ph.D. student in Univeristy Sains Malayisa (USM) was a significant as well as challenging experience to me. Numerous people were helpful directly or indirectly in shaping up my academic career. It was hardly possible for me to succeed in my research work without this precious support. Here is a small acknowledgment to all those people.

First of all, all praises and thanks goes to almighty ALLAH for giving me the patience, the health as well as giving me the chance to work in such an environment in Malaysia and in USM in particular. Second, I wish to deeply thank my supervisor Prof. Dr. Sureswaran Ramadass for introducing me to the world of research and giving me the golden chance to work in this field. It was only due to his valuable guidance, cheerful enthusiasm and ever-friendly nature that I was able to complete my research work. I am also grateful to my co-supervisor Assoc. Prof. Dr. Rahmat Budiarto, for his help and comments.

Last but not least, I would like to thank those whom are in my heart; my father, for his endless and continuous encouragement and constant support, my mother, for her continuous prayers and inspirations, my wife, for her great and deep understanding and support, and my dearest brothers and sisters for keeping me smiling and motivated. I dedicated this work to all of them as without whose support and understanding, this thesis would not have been completed. Thank you very much.

Table of content

List of Tables

List of Figures

List of Abbreviations

ANOVA	Analysis of Variances
API	Application Programming Interface
BPF	Berkeley Packet Filter
bps	Bit per second
CPU	Control Processing Unit
eth	Ethernet hardware description
FCFS	First Come First Serve
FIFO	First In First Out
HLCB	High Level Circular Buffer
HLCBP	High Level Circular Buffer for Packet Capturing
HLCBR	High Level Circular Buffer for Packet Reading
HTML	Hyber text mark up language
ICMP	Internet Control Message Protocol
IO	Input Output
IP	Internet protocol
IPC	Inter Process Communication
IPFIX	Internet Protocol Flow Information eXport
IT	Information Technology
JPCAP	Java Packet Capturing
LAN	Local Area Network
MAC	Media Access Control
MB	Mega byte

MD5	Message-Digest algorithme 5
MIB	Management Information Base.
NASTY	Network Analysis and Statistics Yielding
NAv6	National Advanced IPv6 Centre of Excellence
NIC	Network Interface Card.
NMS	Network Management System.
OS	Operating System
PMH	Passive Measuring Host.
PSAMP	Packet SAMPling
pps	Packet per Second
RCP	Rich Client Platform
RMON	Remote Network MONitoring
RPC Bind()	Remote Procedure Call Bind request
RTDNM	Real Time Distributed Network Monitoring Platform
SDLC	Software Development Life Cycle
SNMP	Simple Network Management Protocol.
SQL	System Query Language
TCP	Transmission Control Protocol
TTL	Time to Live
UDP	User Datagram Protocol
VERMONT	VERsatile MONitoring Toolkit
WAN	Wide Area Network

PLATFORM TERAGIH MASA NYATA UNTUK PEMANTAUAN RANGKAIAN

ABSTRAK

Perkembangan geografi dan peningkatan saiz dalam rangkaian-rangkaian komputer menjadikan keperluan pemantauan terhadapnya menjadi semakin penting. Hasilnya, teknik-teknik dan mekanisme-mekanisme baru bagi pemantauan rangkaian teragih adalah diperlukan bagi menampung keadaan tersebut.

Tesis ini bertujuan untuk membangunkan sebuah platform bagi menyokong aplikasi teragih masa nyata pasif untuk pemantauan rangkaian dan keselamatan. Platform tersebut menggunakan konsep seni bina terbuka bagi pemaju aplikasi lain untuk menambah kefungsian atau melanjutkan kefungsian yang sedia ada.

Platform ini juga mencadangkan tiga konsep baru dalam pemantauan rangkaian masa nyata pasif. Pertama ialah 3-peringkat seni bina untuk memastikan satu liputan lengkap rangkaian teragih dan muatan yang seimbang di kalangan semua unsur pemantauan rangkaian teragih. Konsep kedua adalah teknik penimbalan inovatif yang digunakan untuk menangani lalu lintas rangkaian berkelajuan tinggi tanpa risiko kehilangan paket serta mengurangkan kos pemprosesan. Konsep ketiga adalah pengumpulan data teragih pasif dengan satu pelayan penyelarasan pusat. Setiap unsur mempunyai kecerdasan terbina sebelum proses pengumpulan data sekaligus mengurangkan penggunaan jalur lebar antara pelayan penyelarasan pusat dan unsur-unsur pemantauan teragih.

Platform ini digunakan untuk melaksanakan aplikasi rangkaian pemantauan teragih masa nyata (jEnterprise) dengan merggunakan platform RTDNM. Penggunaan ini dilaksanakan dalam aplikasi perusahaan. Pelaksanaan ini membuktikan kebolehan platform RTDNM untuk membenarkan aplikasi lain memperkembangkan teras platform melalui Aplikasi Pengaturcaraan Antara Muka (APIs) dan mata sambungan sedia ada. Penggunaan aplikasi yang dilaksanakan juga membuktikan kebolehan platform RTDNM berdasarkan aplikasi yang disepadukan ke dalam persekitaran rangkaian yang berbeza.

A REAL TIME DISTRIBUTED NETWORK MONITORING PLATFORM

ABSTRACT

As computer networks increase in size and expand geographically, the necessity to monitor them becomes increasingly important. Thus, new techniques and mechanisms for distributed network monitoring are needed to accommodate the above situation.

The intent of this thesis is to develop a platform to support real time passive distributed network monitoring applications. The platform will use the open architecture concept for other application developers to add functionalities or extend existing functionalities.

The platform also proposes three new concepts in real time passive network monitoring. The first is the 3-tier architecture to ensure a complete coverage of the distributed network as well as balancing the load among all distributed network monitoring elements. The second concept is the innovative buffering technique that is used to handle the high speed network traffic with minimal packet loss as well as minimizing the processing and network overhead. Finally, the third concept is the distributed passive data gathering with a central coordination server. Each element has intelligence built-in to pre-process the passive gathered data and reduce the bandwidth consumption between the central coordination server and the distributed monitoring elements.

Finally, we implement a distributed real time network monitoring application (jEnterprise) utilizing the RTDNM platform. We deploy the implemented application into an enterprise network (Nav6). However, this implementation proves the RTDNM platform ability to allow other applications to extend the platform cores through its available APIs and extension points. The deployment of the implemented application further proves the ability of the RTDNM platform based application to be integrated into heterogeneous network environments.

CHAPTER ONE

INTRODUCTION

Network Monitoring has been around since networks existed. Due to the increasing complexity and importance of data networks as well as the distributed nature of the networks, there has been a steady demand for advanced network monitoring systems. The demand for such monitoring is not only in terms of the number of communicating hosts, links, switches, routing nodes and new protocols but also new security threats and network performance issues (Sihan Qing and Wen 2005). Thus, there is a need to provide network administrators with advanced monitoring tools to enable them to better understand and monitor their networks. The monitoring task will ensure that the monitored networks and network applications experience high availability and good end-to-end performance.

In this thesis, we propose a novel distributed passive real time network monitoring platform to enable network administrators monitor their networks in real time. This thesis also proposes a new monitoring architecture to provide a richer set of data that will contribute to enhance understanding network application and distributed network performance as a whole. The monitoring architecture will have a distributed passive real time monitoring elements that cover the whole network, from the end hosts to the network devices, as well as a central monitoring server for repository and coordination. The distributed monitoring elements will pre-process real time traffic passively with minimal packets loss utilizing the proposed innovative buffering technique.

1.1. Background

Generally, most routers, switches, and intelligent hubs collect some levels of network traffic statistics in a form of SNMP or RMON data. This information is important to network administrators who are responsible for keeping small networks operating at peak performance. SNMP or RMON data requires a network management (Curley 2000) or monitoring application (Subramanian 2000) to query the required information. While SNMP or RMON provides excellent statistics on the large-scale level, but it does not provide the level of details that is often required to completely resolve many network issues. For example, SNMP or RMON may show high utilization on the router's Internet interface, but it will not show in-depth details what kind of traffic is using up the bandwidth. This leaves the administrators knowing what the problem is (*high bandwidth consumption to the Internet*), but not knowing the cause, and therefore, lacking the ability to quickly resolve the issue.

1.2.1 Network Monitoring

Network monitoring basically aims to give a clear view of the monitored network and assist in dealing with network faults and performance issues proactively (K. G. Anagnostakis, S. Ioannidis et al. 2002). A differentiation between network management and network monitoring must be clear, even though the two concepts are interchangeably used. Thus, network monitoring is defined as a bouquet of techniques, practices, and technologies that lead to obtaining information regarding network performance (Subramanian 2000). Whereas, network management is defined as a group of operation, maintenance, and administration of network devices (Subramanian 2000).

Network monitoring describes a set of techniques to observe and quantify what is happening in the network. Network monitoring provides important input towards:

- Performance tuning: identifying and reducing bottlenecks.
- Troubleshooting: identifying, diagnosing and isolating faults.
- Planning: predicting the scale and nature of additional resources.
- Development and design of new technologies
- Understanding and controlling complexity: to understand and observe the interaction between components of the network.
- Identification and correction of odd behaviour.

1.2.2 Network Monitoring Techniques

With regards to the techniques used in gathering network data, there are two techniques known in quantifying exactly what is happening in the network, they can be classified as passive techniques or active techniques (Forouzan 2003). Using the passive technique, data is gathered from the network in a non intrusive manner without injecting a single packet into the network. Using active technique, data is gathered from the network in an intrusive manner by observing the affect of injecting traffic into the network. The injected traffic usually takes the form suitable to the issue of the problem being investigated (Landfeldt, Sookavatana et al. 2000).

1.2.2.1 Passive Monitoring

Passive data is gathered from the network in a non intrusive manner without injecting a single packet to the network. It can involve sending out results from the passive monitoring station to the monitoring application utilizing standard TCP connections to a known IP address rather than sending traffic to certain hosts to stimulate the host to reply.

The advantage of the passive monitoring is the stealth nature of the monitoring itself without interfering with the normal traffic being monitored. The disadvantage of such monitoring is that, as network bandwidth and the volume of traffic increase it would be difficult to manage and collect data using normal processing power computers.

1.2.2.2 Active Monitoring

Active data is gathered from the network in an intrusive manner by observing the effect of injecting traffic into the network, which takes the form suitable to the issue of investigation. Data gathered by such approach is relevant to the subject of investigation, and maybe stored for later analysis. The injected traffic acts as a stimulus for a specific issue or factor to be measured i.e. OS fingerprinting on a remote host, open ports...etc. Moreover, active technique involves sending queries in a known format or using known management protocol to the monitoring station requesting for certain level of information i.e. SNMP queries. These queries will benefit only the monitoring application requesting certain level of information. Thus, this contributes extra unwanted traffic to the network.

There is always a risk that the injected traffic may cause interfering leading to incorrect results or misleading data.

1.2.3 Real time Network Monitoring

Real time network monitoring applications are able to aggregate, quantify, and analyze network traffic for a clearer view. Thus, without network monitoring systems, it would be difficult to identify and resolve network problems (Zeltserman 2000; Du 2004). To ensure 100% system availability, the organization's IT department not only needs to reduce the time required to resolve a problem, but also needs to prevent problems from occurring. The network administrators are usually the first line of defence whenever abnormality is detected in the network (Kozierok 2005). Even when there is no clear-cut culprit, the network administrator has to ensure that the network does not crash or suffer degradation (Augusto Ciuffoletti and Polychronakis 2006; Trimintzios, M. Papadogiannakis et al. 2006).

Real time network monitoring includes the monitoring of network performance in real time. What is fast for one application can be slow for another (Marcia Zangrilli and Lowekamp 2003). It is the network administrator's job to ensure that their network and network applications are performing properly for their environment. However, advanced tools are needed to help network administrators to intensively and continuously monitor their network performance. Where real time network monitoring must be accurate in detecting network disruptions and the cause of the disruptions (Lee, Chen et al. 2006).

The needs for network monitoring and traffic analysis along with the increasing link speeds have exposed limitations in the existing network monitoring architectures. The most widely used abstraction of network traffic monitoring has been that of flow-level traffic summaries (Lifler 1997; Claise 2004; Sadasivan, Brownlee et al. 2005). The flow-level summaries first demonstrated in software prototype such as NeTraMet (Brownlee 1997) and later integrated as a built-in functionality in most of the existing routers (Claise 2004). This approach has been adequate in supporting monitoring functions such as accounting. However, the information contained in flow-level summaries is usually not detailed enough for the emerging monitoring applications. For instance, it is not possible to identify the network usage of the new applications that are using random ports, such as peer-to-peer file sharing, multimedia streaming or conferencing.

In the absence of any better abstraction, many network operators are using the full packet capturing (Fraleigh, Moon et al. Sprint,2000) or case specific solutions that are supported by a specialized hardware (Endace 2002; Endace 2002; Endace 2003). These approaches have high hardware cost, significant processing needs, and producing vast amount of data that might be unmanageable by the monitoring applications. Therefore, such limitation by providing too little information (flow-level traffic summary) against the vast amount of data provided by fully packet capturing, demonstrate a need for a general purpose environment for running network monitoring applications. This environment would provide monitoring applications with the right amount of information they need, or even provide the result of monitoring the network performance in a form of alerts / events to the monitoring application with minimal overhead on the network.

1.2.4 Distributed Real time Network Monitoring

Nowadays, computer networks are growing rapidly, and they tend to be large heterogeneous collection of computers. The job of managing and monitoring networks of this size has become increasingly difficult. Fortunately, computers are good in watching each other. This opens the door to automate the task of monitoring networks (Han Hee Song and Yalagandula 2007). While an intrusion detection system monitors a network for threats from the outside, the real time distributed network monitoring system monitors the enterprise network from the inside. Figure 1.1 depicts the distributed network monitoring architecture overview.

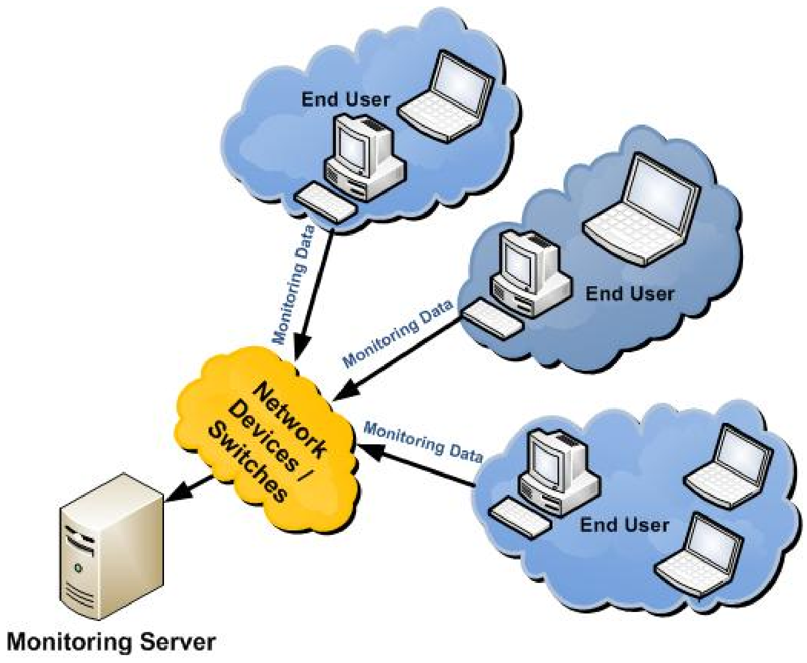

Figure 1.1: Distributed network monitoring architecture overview

Currently, most distributed network monitoring systems are active in nature, depending on SNMP or RMON agents (K. G. Anagnostakis, S. Ioannidis et al. 2002; K. G. Anagnostakis, M. B. Greenwald et al. 2006). These approaches normally consist of agents and managers. Agents are server processes running on each involved manageable network entity that collect network devices data and store them in a specific format with the support of a management protocol such as SNMP (Stalling 1993; Stalling 1996). On the other hand, the manager application is in charge of retrieving the agent's stored data by sending requests using the management protocol. Figure 1.2 illustrates the basic RMON/SNMP network monitoring approach.

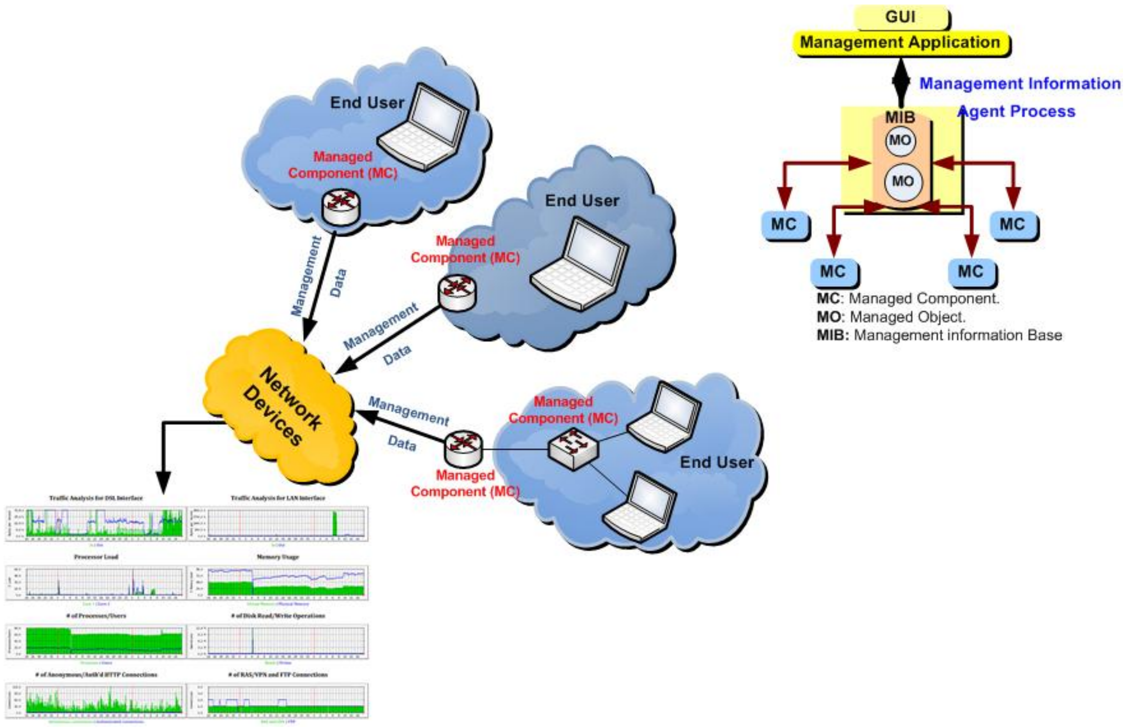

Figure 1.2: Active distributed network monitoring overview

1.3 Problem Statement

Since network traffic monitoring is becoming increasingly important for the operation of the networks, several passive monitoring infrastructures have been proposed for efficient network monitoring. Gigascope (Cranor, Johnson et al. 2003) is a stream database for storing the captured network data in a central repository for further analysis. Sprint's passive monitoring system is another network monitoring tool which collects data from different monitoring points into a central repository for analysis (Fraleigh, Diot et al. 2001). Arlos, Fiedler et al. (2005) proposes a distributed passive measurements infrastructure that supports various monitoring equipment within the same administrative domain. These approaches lack the ability to cover the whole distributed network due to the limitation within the switching environment. They are deployed on strategic points across the networks but none is deployed at the end hosts. This leaves some hosts invisible to the monitoring applications. Thus, they might cause harm to the network by introducing extra unwanted traffic to the network (*worm scanning traffic*) that is not detectable/noticeable by the monitoring elements due to the switching environment. However, one of the main design choices to be considered is where to deploy the monitoring system? What range of distributed traffic is to cover? What monitoring architecture is to adopt?

A monitoring service that is deployed only at the endpoints of the network is easier to maintain and deploy. But at the same time it is limited to the traffic coming or leaving the endpoints. This approach will not cover the whole network traffic. Thus, it will lead to incorrect performance reporting as it is not necessary that all nodes traffic will pass this monitored point.

Some applications rely on collecting traffic at strategic points such as NG-MON (Hong, Kim et al. 2002; Se-Hee Han, Myung-Sup Kim et al. 2002) and EarlyBird (S. Singh, C. Estan et al. 2003). These approaches are feasible for small distributed sub-networks but not for large distributed networks, because they are limited in viewing the whole sub-network traffic due to the switching environments constraints. Other applications support a distributed network monitoring architecture, but these applications monitor the traffic leaving the distributed sub-network perimeter without inspecting the internal traffic that might be suspicious such as Hard-LANs (N. Weaver, D. Ellis et al. 2004; Talks and Research 2005,2006). Similarly, IMS (Michigan 2005) distributes a number of monitoring sensors across the distributed sub-network boundary, leaving the internal network open for internal security breaches, such as worms propagating from mobile nodes like notebooks. The majority of distributed network monitoring applications such as Zenoss (Zenoss 2005), Nagios (Harlan 2003 ; Galstad 2007; Nagios Enterprises 2008) and OpManager (AdventNet 2008) rely on probing certain node's SNMP or RMON 1 objects. This probing will increase the traffic overhead into the distributed network. Furthermore, it is limited to the point of investigation as each SNMP query will result in only a certain level of information regarding the point of the investigation.

Another factor to consider is the additional traffic to be introduced by the distributed network monitoring application to the distributed network. This is important because the additional traffic introduced to the network is not utilized by any network application, and it will affect the overall performance of the distributed network, especially if more than one host (*any network device*) is contributing to the normal network traffic (Antonios Danalis and Dovrolis 2003).

Many applications introduce extra traffic into the distributed network such as Zenoss (Zenoss 2005), Nagios (Harlan 2003 ; Galstad 2007; Nagios Enterprises 2008) and OpManager (AdventNet 2008). Other applications are sniffing the network traffic without introducing any additional traffic to the network, such as Nprobe (Andrew W. Moore, Rolf Neugebauer et al. 2002; Cambridge 2002), NG-MON. These applications do not support a distributed real time passive data gathering with central monitoring architecture. HISTORY (Falko Dressler and Carle 2005), for instance uses a combination of passive and active techniques for distributed data gathering.

Buffering technique is essential in real time application to handle network traffic bursts. In the case of distributed networks, the traffic is voluminous. Thus, the traffic burst can be significant and must be handled properly. Different distributed network monitoring applications use different techniques of buffering in handling the traffic bursts in real time. For instance, PickPacket (Pande 2002; S.K.Jain 2003; Pande, Gupta et al. 2005) depends on the normal processing power of a cluster of machines utilizing the pipelining and parallel processing for traffic capturing and storing. Nprobe shows a very good and fast memory management technique based on a modified OS kernel buffering. However, this buffering technique resides within the OS kernel space without giving any level of flexibility to the users to control or manage the buffer contents. Another type of buffering mechanism is carried out by implementing a set of threads and queues with fixed or flexible size such as Pandora (Patarin and Makpangou 2000), HISTORY, EarlyBird and Hard-LANs. The main problem of such buffering mechanisms is when the buffer gets full, packets will start dropping / overwriting in addition to the thread race condition that might occur. Using a cluster of machines to utilize the processing power of the cluster is another way to handle the traffic bursts such as NG-MON.

Papadogiannakis and Antoniades et al (2007) propose a locality buffer mechanism to enhance network monitoring applications and minimize the packets loss ratio. The locality buffer concept summarizes as reordering the captured traffic by clustering the packets with the same destination port, before they are delivered to the monitoring application. The locality buffering was implemented within the famous Libpcap packet capturing library. The Libpcap single-packet-sized buffer was increased to hold larger number of packets rather than only one packet. Moreover, the implementation was also based on using indexing table with a link list for each port. The index consists of a table of 64K entries, one for each port number. The indexing structure is to maintain the order of the packet arrival without the need to reorder the packet arrival within the buffer. The packets are delivered to the application sorted by their ports during the delivery phase. The Locality buffer optimal size was chosen to be 4000 packets. The author clearly states that the buffering structure will lose packets after the buffer is full without any processing on the captured packets.

Some of the IP traffic analysis tools that are capable of monitoring multi-protocols and live capture analysis are deployed into low bandwidth networks due to the buffering limitation that causes the captured packets to be dropped (Bussiere and But 2005). For instance, But and Bussiere (2005) designed a buffering mechanisms in their NetSniff tool part of JPCAP that is by default set to 32K in size. This size is relatively small and leads to packets lose during high capturing rates. The author evaluated the buffer size under FreeBSD operating system and how the buffer size affects the overall performance of the real time packet capturing of NetSniff. The buffer size was changed based on the operating system functions (*sysctl*) and not within the NetSniff application. This leaves the Netsniff and the new buffering mechanism valid only for FreeBSD operating system

operating system

Finally, different applications have different packets' sizes (Nordqvist and Liu 2003). These packets can easily fill up the large FIFO-based packets buffer. To reduce the packets loss ratio, there are various buffer management algorithms that exist to reduce the amount of the dropped packets (Irland 1978; Foschini and Gopinath 1983; Tobagi 1990). The algorithms mainly focus on reducing the packet loss ratio utilizing the power of the hardware rather than commodity hardware. The popular algorithms for packets buffering managements are: Completely partitioned; the entire buffer is partitioned based on a fixed size of queues (Kamoun and Kleinrock 1980). Packet loss for the queues occurs when a particular queue allocated size is full. Completely shared; the entire buffer is shared among the various queues (Kamoun and Kleinrock 1980). Packets loss only occurs if the entire buffer space is full. Dynamic buffer; with various queues sizes that depend on the remaining space in the allocated buffer space (Choudhury and Hahne 1996). Packets will start dropping only if the current queue length is less than a certain threshold.

As a result, the growth of distributed computing creates new challenges in term of providing accurate and efficient network monitoring in large and distributed enterprise networks without increasing the network overhead and with minimal packet loss. However, current enterprise traffic tools and systems lack the ability to provide the following:

- A distributed and scalable real time network monitoring architecture.
- Passive real time data gathering by all distributed nodes without causing any form of significant processing overhead with minimal packet loss.
- Centralized coordination with distributed intelligent analysis of data. This must be achieved with a minimal increase of processing overheads and bandwidth consumption.

1.4 Goal and Research Objectives

This research intends to propose, design, and develop a generic real time passive distributed network monitoring platform to support real time distributed network monitoring applications. The proposed platform focuses on resolving the limited view of the monitored network by proposing a monitoring architecture that ensures the full coverage of the network. The proposed platform utilizes the passive monitoring technique to avoid injecting extra traffic to the network and also to avoid stimulating the network to reply for certain requests. However, being passive this requires a buffering mechanism to accommodate the different packet arrival rates with minimal packet loss which is another factor the proposed platform considers and address.

However, the main objectives of this research are to:

- Study existing real time network and distributed network monitoring techniques and architectures and highlighting there advantages and outline their limitations.

- Study the affect of the unmanaged network devices on the network coverage, and propose a new architecture that ensures the complete coverage of the distributed network that may consists of heterogeneous (*managed and unmanaged*) set of network devices without injecting extra unwanted traffic to the monitored network and with minimal overhead on the network itself.

- Experimentally study and design a new buffering mechanism to minimize the amount of the packet loss under various traffic arrival rates for on-line passive traffic monitoring utilizing commodity hardware's.

- Propose, design, and develop a generic real time passive distributed network monitoring platform for distributed data collection and analysis. The platform also should be tailored to any real time passive distributed network monitoring applications through a set of distributed monitoring APIs and extension points for other applications to even extend existing functionality or add new functionality to the platform. And compare the proposed platform to the existing approaches or solutions in network and distributed network monitoring in order to evaluate its effectiveness.

1.5 Research Scope

Since the platform proposes to support real time passive distributed network monitoring applications, the platform only considers real time and passive traffic gathering and analysis with minimal packet loss and processing overhead. Moreover, since the platform also proposes to support various network monitoring application development life cycles (SDLC), the platform considers providing a set of extension points and APIs for other network monitoring application development.

1.6 Contribution of this thesis

The contribution of this thesis is the Real time Distributed Network Monitoring Platform (RTDMN Platform). The RTDNM platform will provide a set of distributed monitoring elements and API's to do real time distributed network monitoring analysis efficiently and transparently. Traditional distributed network monitoring applications are no more efficient in monitoring the distributed networks because of their limited view on the network due to the complexity of the switching environments as well as the switching cascading limitations. However, the RTDNM platform will ensure a complete monitored distributed network environment through:

3-tier distributed architecture

Traditional distributed network monitoring applications have a limited view on the distributed network. The 3-tier distributed architecture will ensure the full coverage of the distributed network by deploying the distributed monitoring elements across the distributed network. These elements will cover the distributed network end elements as well as the core network devices. The 3-tier architecture coordinated and managed by a central coordination server. The distributed element's main function is to collect and process the distributed information. Figure 1.3 depicts the concept of the 3-tier distributed monitoring architecture for distributed network monitoring.

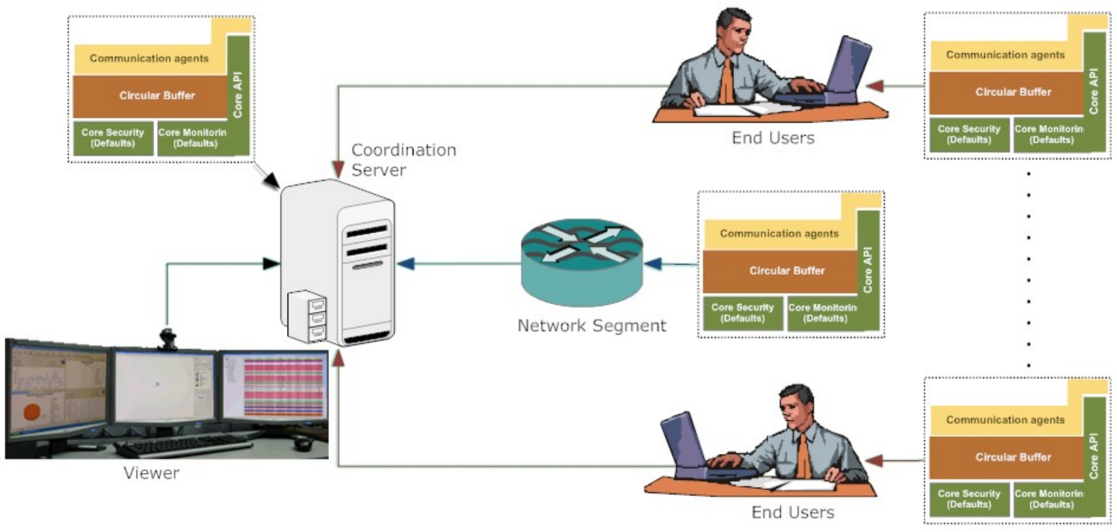

Figure 1.3: The 3-tier distributed network monitoring architecture

Distributed Passive Data Gathering

In a distributed network environment, data are gathered by deploying the distributed monitoring elements across the network to reduce the bandwidth consumed and to balance the load among each other by pre-processing the distributed information within each distributed monitoring element passively. These distributed monitoring elements will utilise the concept of the 3-tier monitoring architecture by deploying a monitoring elements at the end hosts as well as the network devices to ensure the full coverage of the network and to overcome the limited view of the distributed network due to the switching environment.

Innovative Buffering Technique

In a distributed network environments the set of distributed network traffic is potentially available, but it might become unmanageable (Sihan Qing and Wen 2005).

This might occur if the distributed data arrives from the distributed network in higher speed than capturing, which causes memory buffer to get full very fast and force incoming packets to be dropped and loss. Thus, a high speed memory buffering technique is required to accommodate fast distributed network data capturing and processing with minimal packet loss to avoid the memory buffer full issue. The RTDNM supports an innovative buffering technique to minimize the real time packet capturing loss and system overhead by reducing the calls between the kernel level and the user level memory.

On the other hand, several emerging real time distributed applications can use the RTDNM platform. The RTDNM platform API's are tailored to contribute towards any real time passive distributed network monitoring applications. Since the distributed network performance is of main concern, a distributed network monitoring application needs to be developed to ensure that the distributed network operating at an optimal condition. The RTDNM platform provides a set of APIs to ease the task of developing such distributed network monitoring applications, as Figure 1.4 depicts.

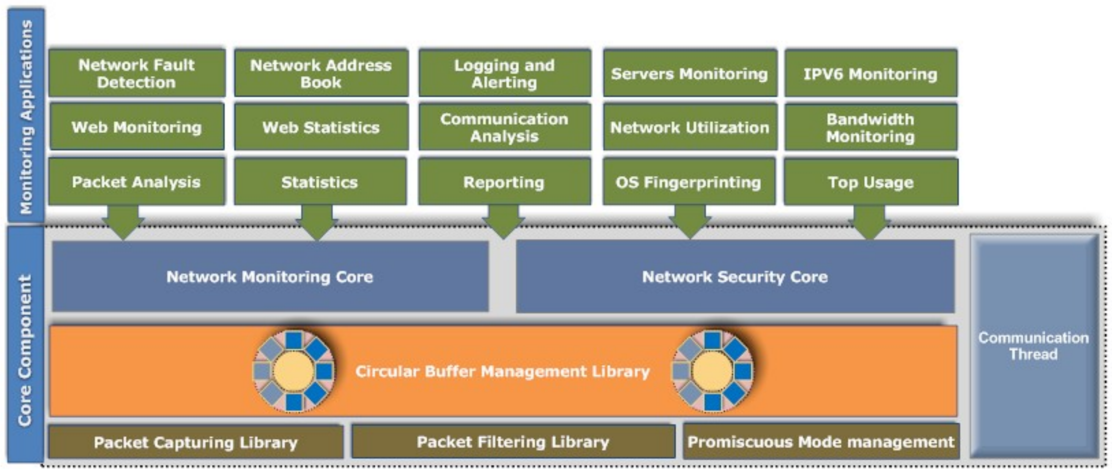

Figure 1.4: Distributed network monitoring applications

For instance, bandwidth is the most critical issue that faces the recent application performance. Monitoring the bandwidth consumption is one of the main factors that need to be continuously monitored.

Finally, the network monitoring APIs (*Application Programmers Interface*) will allow further development of distributed enterprise applications. Examples of such distributed enterprise applications include:

- *A passive mode high end, real-time distributed network monitoring system.*
- *An intelligent distributed real-time security monitoring system.*
- *A versatile and advanced GRID monitoring system.*
- *A distributed Network Assists Management System.*
- *A distributed Fault detection and Isolation system.*

1.7 Thesis Outline

This thesis is organized into six chapters. The contents are arranged such that each previous chapter provides a basic idea to further proceed to the next chapter. Firstly, this chapter introduces the background principles of real time network monitoring along with our research objectives and contributions.

In Chapter 2, we review literature and fundamental concepts related to our work and issues surrounding it. The reasons why we choose the methodologies for our system are discussed.

Chapter 3 covers the methodology of how the proposed platform was designed. The innovative buffering technique for real time network traffic analysis is described. The platform architecture is also introduced in this chapter along with the passive nature of the data gathering elements. Lastly, the general distributed monitoring element design and the distributed scheme are also described.

Chapter 4 consists of implementation details and issues with our proposed platform. We also illustrate the experimentation direction and the implementation of a real time distributed network monitoring application utilizing the RTDNM platform. The experiment details are described.

In depth analysis and discussion of the results from the experiments described in Chapter 4 are the primary content of Chapter 5. This chapter is divided into two parts. The first part reports the results from the first experiment using benchmark data and the second part reports the result from the second experiment by deploying the developed distributed network monitoring system into the national advanced IPv6 Centre of Excellence. A discussion is presented at the end of both parts.

Chapter 6 summarizes this thesis. We revisit our research contributions with regard to methods and strategy we proposed in Chapter 3 and its results in Chapter 5. Finally, a discussion and suggestion for future work pertaining to this thesis is presented.

CHAPTER TWO

BACKGROUND AND RELATED WORK

This chapter will provide a brief overview of the current state of the art in the area of real time network monitoring systems and platforms.

2.1 Network Monitoring; State of the Art Overview

A comprehensive and up-to date listing of network measurements and monitoring tools can be found at the CAID website (Caida 1999). The following sections present representative examples of these tools in analyzing the following questions: how their required data are gathered? What techniques are used? What architectures are supported? Finally, how high speed traffic is handled?

2.1.1 Passive Network monitoring

Passive network monitoring is the process of examining the network traffic silently at a single or multiple points. By this, different aspects of the network, such as traffic, and behaviour can be further characterized. Monitoring the network packets and their magnitude can provide useful information on the link utilization, which can be further used for route planning, and network provisioning. Also, network monitoring is addressed to be essential for network performance improvements. Unfortunately, passive measurement systems are yet surviving with means of filters to minimize the overhead processing.

In recent years, several researches in the field of passive network monitoring have been conducted, Such as:

2.1.1.1 PickPacket

PickPacket is a passive network monitoring tool that focuses on capturing and filtering traffic across several layers of the protocol stack for certain applications and dumping the filtered traffic into a permanent storage media for further analysis (Pande, Gupta et al. 2005). PickPacket also supports real time searching for text strings within the content of the packet. The packets which are dumped to the disk are then analyzed in an offline mode using existing third party software to render the captured data.

The BSD packet filter (BPF) (Steven McCanne and Jacobson 1992) and PickPacket are both a combination of kernel level and user-level filtering. Thus, it is important to highlight the overhead of the dynamic changes in the in-kernel filtering especially if it is combined with user level filtering. This change has to be considered due to the fact that it could turn-off the in-kernel filtering entirely (Pande, Gupta et al. 2005). Figure 2.1 below depicts the ***Pickpacket*** architectural view: