석 사 학 위 논 문

로드 분산 방법을 이용한
네트워크 트래픽 모니터링 및 분석

홍 순 화 (洪 淳 華)

전자컴퓨터 공학부 (컴퓨터공학)

네트워크 전공

포항공과대학교 대학원

2002

로드 분산 방법을 이용한
네트워크 트래픽 모니터링 및 분석

Network Traffic Monitoring and Analysis
using Load Distribution Methodology

Network Traffic Monitoring and Analysis using Load Distribution Methodology

by

Soon-Hwa Hong

Division of Electrical and Computer Engineering

(Computer Science and Engineering)

POSTECH

A thesis submitted to the faculty of POSTECH in partial fulfillment of the requirements for the degree of Master of Science in the Division of Electrical and Computer Engineering (Computer Science and Engineering)

Pohang, Korea

December 17, 2001

Approved by

_____

Major Advisor

로드 분산 방법을 이용한
네트워크 트래픽 모니터링 및 분석

홍 순 화

위 논문은 포항공과대학교 전자컴퓨터공학부 (컴퓨터공학) 석사 학위논문으로 학위논문 심사위원회를 통과하였음을 인정함.

2001 년 12 월 17 일

학위논문심사 위원회 위원장 홍원기 (인)

위 원 서영주 (인)

위 원 박찬익 (인)

MECE 20002096

홍 순 화, Soon-Hwa Hong, Network Traffic Monitoring and Analysis using Load Distribution Methodology, **로드 분산 방법을 이용한 네트워크 트래픽 모니터링 및 분석**, Division of Electrical and Computer Engineering (Computer Science and Engineering), 2002, 67P, Advisor: J. Won-Ki Hong, Text in Korean..

ABSTRACT

Today, network traffic is increasing continuously on the Internet, and the deployment of many network-based multimedia applications, multimedia network traffic continues to increase.

Many network traffic analysis systems have been developed and introduced. Most of these network traffic analysis systems can analyze only real-time packets or short-term traffic, but cannot analyze long-term network traffic. Thus, these network analysis tools are not appropriate for trend analysis or for long-term network plan. And, these tools are not appropriate for analysis of very high network traffic due to packet drop from the system overload and cannot analyze multimedia service.

To analyze high network traffic without packet drop and slow response time, we propose a load distribution network traffic monitoring and analysis architecture. This architecture is constructed with packet capture module, packet analysis module, and web-based traffic analysis module in distributed system environment. From this architecture, packet capture module can prevent packet drop from the monitoring system overload. We can analyze long-term network traffic using packet hash algorithm and database. Further, we can analyze multimedia service traffic using UDP/TCP port numbers.

목 차

그림 목차

표 목차

1. 서론

최근 대용량의 인터넷 회선이 보편화되고, 인터넷의 확산과 Video, Audio 등의 멀티미디어 서비스를 제공하는 네트워크 기반의 응용 프로그램이 다양하게 개발되어 사용자가 급격히 증가하고 있다. 이에 따라 네트워크 트래픽이 대용량화 되고 있으며 계속해서 증가하고 있다.

네트워크 관리에 있어서 네트워크 트래픽 모니터링 및 분석의 중요성으로 인해 오래전부터 여러 네트워크 트래픽 분석 시스템이 개발되어 사용되고 있다. 그러나 기존의 네트워크 트래픽 분석 시스템들은 대용량화 되어 가는 네트워크 트래픽을 정확하게 분석하지 못할 뿐만 아니라 장기간 동안 모니터링하는 데 많은 문제점을 가지고 있다.

서론에서는 네트워크 트래픽 모니터링 및 분석의 문제점을 살펴보고 대처하기 위해 네트워크 트래픽 모니터링 및 분석의 필요성, 최근의 네트워크 트래픽 추세, 기존의 네트워크 트래픽 분석 시스템들의 문제점 등을 살펴보고 문제점 해결을 위한 접근 방법을 제시한다.

1.1 네트워크 트래픽 모니터링 및 분석의 필요성

네트워크 트래픽 모니터링 및 분석은 네트워크를 관리하는 데 있어서 매우 중요한 일이다. 그러나 네트워크 트래픽의 분석 및 모니터링이 등한시 되거나 제대로 이루어 지지 못하는 경우가 대부분이다. 따라서 여기서는 네트워크 트래픽 모니터링 및 분석의 필요성을 정리해 본다. 필요성은 다음과 같다.

첫째, 네트워크 트래픽의 현재 상황 및 문제점을 파악하기 위해 필요하다. 현재 사용되고 있는 대역폭은 어느 정도인지, 어떤 호스트들이 트래픽의 대부분을 차지하는지, 어떤 서비스들이 사용되고 있는지, 총 호스트 수는 어느 정도 인지, 어느 시간 대에 트래픽 발생이 높은

지 등을 네트워크 트래픽 모니터링 및 분석을 통해 알 수 있어야 한다.

둘째, 네트워크 트래픽의 증가 원인을 분석하기 위해 필요하다. 네트워크 트래픽의 증가 원인이 전체 사용자의 증가로 인한 것인지, 몇몇 시스템의 트래픽의 증가로 인한 것인지 네트워크 트래픽 모니터링 및 분석을 통해 알 수 있어야 한다.

셋째, 네트워크 회선 계획을 세우는 데 필요하다. 하루 중 최고 트래픽은 어느 정도인지, 달별, 년별로 어느 정도 트래픽이 증가하고 있는지, 네트워크 트래픽 모니터링 및 분석을 통해 알 수 있어야 한다.

위의 세 가지의 필요성으로 인해 네트워크 트래픽 모니터링 및 분석이 필요하며, 네트워크 트래픽 모니터링 및 분석을 통해 네트워크 관리 및 계획을 최적화 할 수 있다.

1.2 네트워크 트래픽 추세 및 요구 사항

다음으로는 네트워크 트래픽 추세와 추세에 맞는 요구사항을 살펴본다. 네트워크 트래픽의 추세를 알아야 정확한 네트워크 트래픽 모니터링 및 분석을 할 수 있다. 네트워크 트래픽의 추세와 추세에 맞는 요구사항은 다음과 같다.

첫째, 대용량 인터넷 회선이 보편화되는 추세이다. 최근에는 10 Mbps 이상의 대용량 인터넷 회선이 많은 곳에서 기본으로 사용되고 있다. 따라서 이러한 추세에 맞게 대용량 네트워크 트래픽을 정확하고 신속하게 분석 할 수 있어야 한다.

둘째, 네트워크 트래픽은 계속해서 증가하는 추세이다. 따라서 달별, 년 별 등의 장기간 동안의 네트워크 트래픽 모니터링 및 분석을 통해 네트워크 회선 계획을 세울 수 있어야 한다.

셋째, 네트워크 트래픽은 짧은 시간 동안에도 증가와 감소를 반복

하면서 동시다발적으로 변한다. 네트워크 트래픽은 짧은 시간 안에 급속하게 증가할 수 있으며, 트래픽이 적은 시간 대에도 갑자기 ftp같은 서비스가 사용되어 네트워크 트래픽이 급증 할 수 있다. 따라서 분 단위, 시간 단위 등의 단기간 동안의 네트워크 트래픽 모니터링 및 분석을 통해서 문제점에 대한 빠른 분석 및 대처가 필요하다.

넷째, Video, Audio 등의 다양한 네트워크 기반의 응용 서비스가 늘어나는 추세이다. 특히 스트리밍, Peer-to-Peer (P2P) [1] 같은 멀티미디어 서비스가 늘고 있다. 따라서 멀티미디어 서비스 등의 사용되고 있는 서비스에 대한 분석이 필요하며 새로 개발되는 서비스에 대한 빠른 대비가 필요하다.

1.3 네트워크 트래픽 분석 시스템의 문제점

1.1 과 1.2 를 통해 네트워크 트래픽 모니터링 및 분석의 필요성과 네트워크 트래픽 추세 및 요구 사항을 살펴 보았다. 네트워크 트래픽 모니터링 및 분석의 가장 큰 문제점은 기존의 네트워크 트래픽 분석 시스템들이 네트워크 트래픽 모니터링 및 분석의 필요성을 만족시키지 못하고 네트워크 트래픽 추세를 반영하여 분석하지 못하는 점이다.

기존의 네트워크 트래픽 분석 시스템들의 문제점을 정리하면 다음과 같다.

첫째, 패킷 손실로 인해 네트워크 트래픽을 정확하게 분석하지 못한다. 패킷 손실은 CPU 로드에 따라 적은 트래픽에서도 발생하며 대용량 트래픽일 수록 패킷 손실이 높아진다. 무엇보다 패킷 손실은 일정한 비율로 일어나는 것이 아니라 시스템 과부하 시 갑자기 일어난다. 따라서 패킷 손실로 인해 전혀 다른 네트워크 트래픽 분석 결과를 보여 줄 수도 있다.

둘째, 대용량 네트워크 트래픽을 처리하지 못한다. 대용량 트래픽

을 패킷 손실 없이 캡처했다고 해도 분당 40 ~ 100 만개의 패킷이 쏟아지는 대용량 트래픽을 저장하고 분석하는 일은 쉽지 않다. 기존의 네트워크 트래픽 분석 시스템들은 대용량 트래픽 처리를 고려하지 않고 설계 되어 대용량 트래픽 분석에 적당치 않다. 대용량 트래픽 처리 시 분석 시간 및 응답 시간이 길어진다거나 분석을 할 수 없는 것이 그 예이다.

셋째, 장기간 동안의 네트워크 트래픽 모니터링 및 분석을 지원하지 않는다. 따라서 기존의 네트워크 분석 시스템으로는 장기적으로 증가 추세인 네트워크 트래픽을 분석하고 모니터링하여 대처 할 수 없다. 장기간 동안의 네트워크 트래픽 모니터링은 네트워크 회선 계획에 중요한 일이므로 네트워크 트래픽 모니터링 및 분석의 중요한 요구사항이다.

넷째, 잘 알려진 서비스만 분석하므로 네트워크 트래픽의 동향을 파악 할 수 없다. 기존의 네트워크 트래픽 분석 시스템들은 주로 telnet, ftp 등의 잘 알려진 서비스만을 분석하는 것에 그친다. 따라서 최근에 증가하고 있는 멀티미디어 서비스를 분석 할 수 없고 새로 개발되는 서비스를 대처하기 어려운 단점이 있다.

1.4 문제점 접근 방법

기존의 네트워크 분석 시스템들은 패킷 캡처와 패킷 분석, 프리젠테이션 등을 동시에 하는 구조이다. 이러한 집중 구조로 인해 대용량 트래픽의 패킷 캡처 및 분석 시에 패킷 손실이 발생하며, 장기간 동안의 데이터 저장이 어렵고, 분석 시간 및 응답 시간이 길어진다.

본 논문에서는 패킷 손실을 막고 대용량 트래픽을 다양하게 분석하기 위해 네트워크 분석 시스템의 로드를 분산하는 로드 분산 네트워크 트래픽 모니터링 및 분석 시스템 구조를 제시한다. 로드 분산 방법

을 사용하여 패킷 캡처 모듈과 패킷 분석 모듈, 프리젠테이션 모듈을 각각의 시스템으로 나눈다. 그러면 패킷 분석 및 프리젠테이션으로 인한 시스템의 로드를 패킷 캡처 모듈이 영향을 받지 않아 패킷 손실이 적어진다. 또, 분석 시간 및 응답 시간도 빨라진다. 따라서 안정적이고 신속한 네트워크 트래픽 모니터링 및 분석을 할 수 있다.

이러한 로드 분산 구조에 패킷 손실 방지, 대용량 패킷 처리를 위한 패킷 해쉬, 데이터베이스를 이용한 장기간 데이터 저장, 멀티미디어 서비스의 유형 분석 등에 관한 연구 및 실험을 더해서 패킷 손실을 막고, 대용량 트래픽을 정확하게 분석하며, 장기간 동안의 모니터링이 가능하고, 멀티미디어 서비스를 분석 할 수 있는 네트워크 트래픽 모니터링 및 분석 시스템을 설계하고 구현한다.

2. 관련 연구

네트워크 트래픽 분석 시스템들의 문제점은 패킷 캡처 시에 패킷 손실이 발생하고, 대용량 트래픽을 처리하지 못하며, 장기간 동안의 모니터링이 어렵고, 멀티미디어 서비스 등을 확인 할 수 없다는 것이다. 관련 연구에서는 이러한 문제점의 원인을 파악하기 위해 패킷 캡처에 대한 연구를 통해 패킷 손실의 원인을 파악하고 패킷 분석 방법을 분류한 뒤 장단점을 조사한다. 또 기존의 네트워크 분석 시스템들을 조사하여 장단점을 비교해 본다.

2.1 패킷 캡처 (Packet Capture)

패킷을 캡처하는 방법에는 크게 libpcap [2] 라이브러리 등을 사용하여 패킷을 캡처하는 방법과 SNMP [3]의 MIB 정보를 사용하여 캡처하는 방법이 있다. SNMP를 사용하면 링크 계층까지만 분석 할 수 있어 네트워크 트래픽의 총량 등의 간단한 정보의 확인만이 가능하나 libpcap 라이브러리 등의 패킷을 캡처하는 방법을 사용하면 응용 계층까지 분석 할 수 있어 네트워크 트래픽의 총량 뿐만 아니라 호스트 정보, 프로토콜 정보, 서비스 정보 등을 확인 할 수 있다. 따라서 대부분의 네트워크 트래픽 분석 시스템들이 libpcap 라이브러리 등을 사용하는 패킷 캡처 방법을 사용한다. 여기서는 libpcap 라이브러리를 사용한 패킷 캡처 방법의 원리와 패킷 손실이 일어나는 원인 및 패킷 손실에 대한 대처 방법을 알아본다.

2.1.1 패킷 캡처 원리

패킷 캡처는 패킷 캡처 프로그램 혹은 운영체제를 통해 시스템의 디바이스 드라이버를 promiscuous 모드로 설정하는 것으로 시작한다. 이

더넷에서 디바이스 드라이버를 promiscuous 모드로 설정하면 시스템이 위치한 네트워크 포인트의 모든 패킷을 캡처 할 수 있다. 그림 1 은 패킷 캡처의 원리를 설명해 준다.

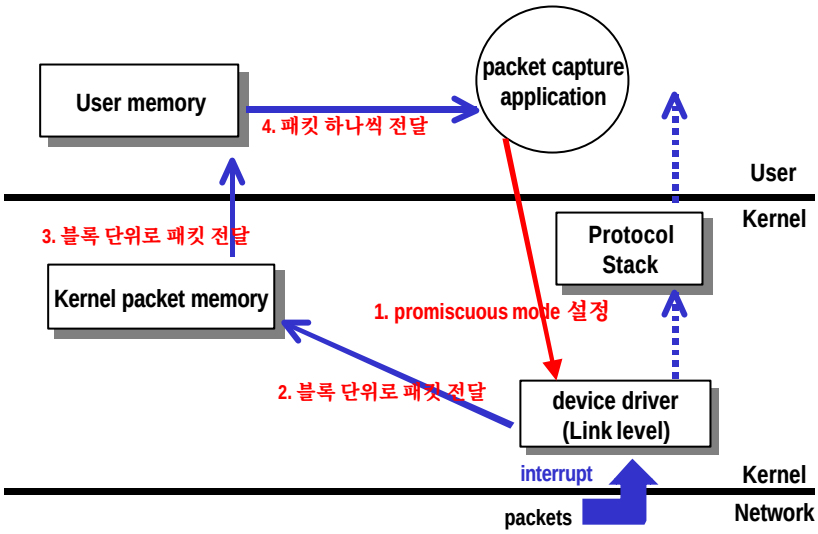


그림 1 패킷 캡처 원리

시스템에 패킷이 들어와 랜카드에서 감지가 되면 랜카드의 디바이스 드라이버에서 인터럽트를 건다. promiscuous 모드가 아닌 경우에는 그림 1 의 디바이스 드라이버를 거쳐 바로 프로토콜 스택으로 전달 된다. 이 경우에는 패킷이 시스템에서 사용하는 패킷인지, 아닌지를 판단하여 시스템에서 사용하는 패킷인 경우 TCP/IP 등의 프로토콜 스택을 통해 사용자 어플리케이션에 전달하여 처리한다. 그런데 그림 1 의 스텝 1 과 같이 패킷 캡처 어플리케이션을 통해 디바이스 드라이버를 promiscuous 모드로 설정하면 디바이스 드라이버에서는 시스템에 감지되는 모든 패킷을 복사해서 커널의 패킷 메모리에 저장한다. 또 이렇게

커널 패킷 메모리에 저장된 패킷들을 패킷 캡처 어플리케이션이 읽을 수 있도록 다시 사용자 메모리에 복사 한다. 마지막으로 사용자 메모리에 저장된 패킷을 패킷 캡처 어플리케이션이 하나씩 읽어서 패킷의 헤더에서 패킷에 대한 정보를 살펴 봄으로써 패킷 캡처가 이루어 진다.

그림 1 에서 promiscuous 모드로 설정 후에 대용량 트래픽이 시스템에 들어오면 디바이스 드라이버에 시스템 과부하가 발생한다. 디바이스 드라이버는 패킷을 감지하면 인터럽트를 걸어 커널에 알리고 들어오는 모든 패킷을 커널의 패킷 메모리에 복사해야 하므로 대용량의 트래픽이 들어오는 경우에 CPU 과부하가 발생하고 커널의 패킷 메모리가 부족해 진다. 디바이스 드라이버의 CPU와 메모리 등의 시스템 자원이 부족하면 커널에서 패킷 손실이 발생한다.

패킷 캡처 어플리케이션에서 대용량 트래픽을 처리 할 경우에도 패킷 손실이 발생한다. 패킷 캡처 어플리케이션이 사용자 메모리에서 패킷을 하나씩 읽어 올 때 시스템 과부하가 발생하기 때문이다. 그림 1 에서 디바이스 드라이버에서 커널의 패킷 메모리로 패킷이 복사 되는 때와 커널의 패킷 메모리에서 사용자 메모리로 패킷이 복사 될 때는 블록 단위로 패킷을 전달한다. 반면 패킷 캡처 어플리케이션이 사용자 메모리의 패킷을 읽어 올 때는 패킷을 하나씩 읽어 온다. 따라서 패킷 캡처 시에 패킷을 하나씩 읽어야 하는 패킷 캡처 어플리케이션에 의해 시스템 과부하가 발생하고, 패킷 캡처 어플리케이션이 사용자 메모리에 쌓인 패킷을 빨리 처리하지 못해 메모리 부족으로 패킷 손실이 발생할 수 있다.

2.1.2 패킷 손실 원인 및 대처 방법

CPU, 메모리 등의 시스템 자원 부족이 패킷 손실의 주 원인이다. 패킷 손실은 2.1.1 에서 살펴 본 것처럼 디바이스 드라이버의 커널에서 일어나는 패킷 손실과 패킷 캡처 어플리케이션의 사용자 레벨에서 일

어나는 패킷 손실로 나눌 수 있다.

커널에서 발생하는 패킷 손실은 디바이스 드라이버의 빈번한 인터럽트로 인한 CPU 로드와 커널의 패킷 메모리 할당 실패가 주 원인이다. 커널의 패킷 손실을 막기 위해서는 CPU, 메모리 등의 시스템 사양을 높이는 방법과 네트워크 드라이버 모듈을 최적화하는 방법이 있다. 네트워크 드라이버 모듈을 최적화하는 방법은 한계가 분명하기 때문에 시스템 사양을 높이는 것으로 커널의 패킷 손실을 막아야 한다. 약 80 Mbps의 포항공대 인터넷 회선에서 커널의 패킷 손실에 대한 테스트에 의하면 350 MHz, 256 MB의 PC에서는 패킷 손실에 관한 로그 메시지가 0.5 초 간격으로 발생하였으며 시스템 과부하로 인해 시스템이 동작 하기도 어려웠다. 또, 500 MHz, 256 MB의 PC에서는 대역폭 사용량이 높은 시간에 패킷 손실에 관한 로그 메시지가 3 초 간격으로 발생 되었다. 그러나 1.0 GHz, 256 MB의 PC에서는 대역폭 사용량이 높은 시간에도 커널에서의 패킷 손실에 관한 로그 메시지가 전혀 없었으며 시스템이 정상적으로 동작하였다. 커널에서의 패킷 손실은 같은 CPU, 메모리 등의 시스템 사양이라고 해도 보드의 처리 속도 등 여러 조건에 따라 달라지므로 커널에서의 패킷 손실을 막기 위한 표준 시스템 사양을 찾기는 힘들다. 다만, 커널에서의 패킷 손실에 관한 로그 기록을 통해 패킷 손실 여부를 판단할 수 있다. 커널에서의 패킷 손실을 막기 위해서는 시스템을 실제로 네트워크 포인트에 설치한 뒤 커널의 로그 기록을 점검하여 패킷 손실에 대한 로그 기록이 있을 경우 시스템 사양을 높여야 한다.

사용자 레벨에서의 패킷 손실은 패킷 캡처 어플리케이션의 자원 부족으로 발생한다. 2.1.1 에서 살펴 본 것처럼 패킷 캡처 어플리케이션이 사용자 메모리의 패킷을 하나씩 읽을 때 시스템 과부하가 발생한다. 패킷 캡처 어플리케이션이 캡처한 패킷을 어떻게 처리하느냐에 따라 시스템 과부하가 발생 할 수 있다. 캡처한 패킷을 처리하는 방법이 복잡해서 CPU 사용 시간이 길어 질수록 시스템 로드가 높아지고 로드가

높아져서 시스템 과부하가 발생하면 패킷 손실이 늘어 난다. 이뿐 아니라 패킷 캡처 어플리케이션이 다른 프로세스에 의해 CPU 자원을 빼앗길 때도 패킷 손실이 발생한다. 다른 프로세스에 의한 패킷 손실은 트래픽이 적은 곳에서도 발생하므로 심각한 문제를 발생시킨다. 다른 프로세스의 예로는 우선 분석 프로세스를 들 수 있다. 예를 들어 패킷 캡처 프로세스가 패킷 분석 프로세스와 한 시스템에서 함께 CPU를 사용할 경우 패킷 캡처 모듈의 CPU 사용률이 낮아지므로 패킷 손실이 발생할 수 있다.

사용자 레벨에서의 패킷 손실은 시스템 사양을 높이는 것으로 완벽하게 해결 할 수 없다. 시스템 사양을 높이면 패킷 손실이 줄어 드는 것이 당연한 일이나, 시스템 사양이 높다고 해도 트래픽 처리 방법에 따라 패킷 손실이 발생 할 수 있다. 따라서 사용자 레벨의 패킷 손실을 막기 위해서는 패킷 캡처 어플리케이션에서의 패킷 처리 시간을 줄이고 다른 프로세스로부터 패킷 캡처 어플리케이션의 시스템 자원 사용을 보장해야 한다.

2.2 패킷 분석

네트워크 트래픽 분석 시스템의 패킷을 분석하는 방법에 대해 알아본다. 패킷을 분석하는 방법에는 패킷을 하나씩 분석하여 패킷 하나씩의 정보를 보여주는 패킷 단위 분석 방법과 패킷의 정보를 모아 분석한 뒤 트래픽 정보를 보여주는 트래픽 분석 방법이 있다. 또, 트래픽 분석 방법은 실시간 트래픽 분석 방법과 일괄처리 트래픽 분석 방법으로 나눌 수 있다. 여기서는 패킷 단위 분석 방법과 실시간 트래픽 분석 방법과 일괄처리 트래픽 분석 방법의 장단점을 살펴본다.

2.2.1 패킷 단위 분석

패킷을 하나씩 분석하여 패킷 하나씩의 정보를 보여 주는 방법을 말한다. 이 방법은 패킷 하나 하나가 어떻게 동작하는지 확인하기 좋다. 또 패킷을 캡처 할 때 원하는 패킷만을 캡처하여 분석하는 패킷 필터링 기능을 적용 할 수 있는 장점이 있다. 그러나 일정 시간 동안의 네트워크 트래픽 총량 이라든가 어떤 서비스가 어떤 호스트에서 사용되었는지 등을 알아내야 하는 네트워크 트래픽 분석에는 적당치 않다. 쉽게 말해 패킷 단위 분석은 모니터링 개념이 결여되어 있다. 따라서 서론에서 소개한 네트워크 트래픽 추세를 분석하기에는 적당한 방법이 아니다. 단, 특정 네트워크의 패킷 동작을 확인하는 곳에 유용하게 사용 할 수 있다.

2.2.2 트래픽 분석

트래픽 분석 방법에는 실시간 트래픽 분석 방법과 일괄처리 트래픽 분석 방법이 있다. 실시간 트래픽 분석은 패킷 하나를 캡처 할 때마다 바로 분석하여 트래픽 정보를 업데이트 하는 방법이다. 실시간으로 누적되어 증가하는 네트워크 트래픽 정보를 확인하기 좋고 패킷 헤더 정보를 저장하는 것이 아니라 분석된 트래픽 정보를 저장하는 것이 특징이다. 반면 일괄처리 트래픽 분석 방법은 패킷 하나를 캡처 할 때마다 패킷의 헤더 정보를 저장해 놓았다가 일정 시간이 지나면 저장된 패킷 헤더 정보를 가지고 트래픽을 분석하는 방법이다. 시간 간격 별로 증가 혹은 감소를 반복하는 네트워크 트래픽을 시간 간격으로 확인하기 좋으며 트래픽 정보도 저장 할 수 있으나 패킷의 헤더 정보를 일정한 시간 동안 반드시 저장해야 분석 할 수 있다는 특징이 있다.

실시간 트래픽 분석 방법은 패킷을 캡처한 뒤 바로 분석하므로 사용자가 요청한 시간의 트래픽 정보를 바로 확인 할 수 있는 장점과

분석 정보만 저장하므로 저장 공간이 절약된다는 장점이 있다. 그러나 정해진 규칙대로만 분석 할 수 있어 분석에 제한이 있는 단점이 있다. 예를 들어, 네트워크 상에서 브로드캐스트가 발생한다고 해도 다 같은 곳에서 발생한 브로드캐스트는 아니다. 서로 다른 곳에서 다른 서비스를 사용하기 위해 발생한 브로드캐스트이지만 실시간 트래픽 분석 방법은 그 특성상 일단 브로드캐스트면 다 같은 것으로 취급한다. 이럴 경우 브로드캐스트의 총량은 확인 할 수 있으나 어떤 서비스를 위한 브로드캐스트인지 알 수가 없는 단점이 있다. 또, 한 패킷을 처리하는데 CPU 로드가 증가하여 대용량 트래픽 분석 시 패킷 손실이 많은 단점이 있다. 2.1.2 에서 언급했듯이 패킷을 처리하는 시간이 증가하면 패킷 캡처 어플리케이션의 CPU 로드가 증가한다. 패킷 캡처 어플리케이션은 패킷을 하나씩 처리하므로 CPU 로드가 증가 할수록 커널 메모리에서 사용자 메모리로 블록 단위로 복사되는 패킷을 모두 처리하기 힘들다. 따라서 메모리가 부족해 지면 패킷 캡처 어플리케이션이 처리하지 못한 사용자 메모리의 패킷들은 버려지게 된다. 이런 문제로 발생하는 패킷 손실은 대용량 트래픽에서 더욱 심각한 문제를 발생한다. 또 실시간 트래픽 분석 방법은 패킷 캡처 후 바로 분석하므로 장기간 동안의 네트워크 트래픽 모니터링 및 분석 보다는 단기간 동안의 모니터링에 더 적합하다.

일괄처리 분석 방법은 패킷의 헤더 정보를 저장해 놓으므로 분석에 제한이 없는 장점이 있다. 필요한 패킷의 헤더 정보를 타임스탬프와 함께 저장해 놓으면 패킷을 하나 발생시킨 호스트의 특정 시간 대의 서비스까지도 정확하게 분석 할 수 있다. 또, 패킷을 일정 시간 동안 분석해 주므로 매분 별, 매시간 별 등의 시간 간격 별로 과거의 트래픽에서 지금까지의 트래픽을 확인 할 수 있는 장점도 있다. 또, 실시간 트래픽 분석 방법과 달리 장기간 동안의 네트워크 트래픽 분석에 적당한 장점도 있다. 이뿐 아니라 패킷 캡처 시간이 짧아 대용량 트래픽의 패킷 캡처 시에 실시간 트래픽 분석 방법보다 패킷 손실이 적다. 그러

나 패킷 하나 하나의 패킷의 헤더 정보를 계속 저장해야 하므로 저장 공간이 늘어나는 단점이 있다. 또한 분석 시간 동안의 분석 프로세스의 시스템 과부하로 인한 패킷 손실이 발생하는 단점도 있다.

2.2 에서 살펴본 방법 중 패킷 단위 분석 방법은 엄밀히 말해 트래픽 분석이 목적이 아닌 것이라는 것을 알 수 있다. 또, 실시간 트래픽 분석 방법은 분석에 제한이 있고 대용량 패킷 캡처에 부적당하며 장기간 동안의 트래픽 모니터링에도 적당치 않다는 것을 알 수 있다. 반면 일괄처리 트래픽 분석 방법은 대용량 패킷 캡처에 적당하며 분석에 제한이 없고 장기간 동안의 네트워크 트래픽 모니터링에 적당하다. 따라서 일괄처리 트래픽 분석 방법이 서론에서 소개한 네트워크 트래픽 추세를 분석하기에 적당한 방법이다. 일괄처리 트래픽 분석 방법에서 저장 공간 문제와 패킷 분석 시의 시스템 과부하로 인한 패킷 손실 문제점을 보완하면 최근의 네트워크 트래픽 추세에 맞는 네트워크 트래픽 모니터링 및 분석이 가능하다.

2.3 네트워크 분석 시스템 조사

여러 네트워크 분석 시스템들이 사용되고 있다. 이 장에서는 각 네트워크 트래픽 분석 시스템들의 특징을 살펴보고 장단점을 파악한 후 특징있는 시스템들을 비교해 본다.

2.3.1 MRTG

MRTG (Multi-Router Traffic Grapher) [4]는 네트워크 링크 간의 트래픽 부하량을 측정하는 도구로써 5 분 단위에서 1 년 단위까지 네트워크 트래픽의 총량을 확인하는 도구이다. 웹기반으로 동작하여 네트워크 트래픽 정보를 확인하기에 편리하고 C와 Perl로 작성되어 유닉스 플랫폼

뿐 만 아니라 윈도우즈 NT에서도 동작하는 장점이 있다. 또 MRTG는 SNMP MIB 정보를 사용하여 libpcap과 같은 방법으로 패킷 캡처를 하지 않으므로 패킷 손실 없이 정확한 정보를 제공해 준다. 그러나 MRTG는 네트워크 트래픽의 총 양에 대한 정보를 제공해 줄 뿐 어떤 호스트에서 어느 정도의 트래픽을 발생시켰는지, 어떤 어플리케이션이 어느 정도의 트래픽을 발생시켰는지, 어떤 프로토콜이 사용되었는지 등의 정보를 제공해 주지 못하는 단점이 있다.

2.3.2 Ntop

Ntop [5]은 Deri Luca가 1998 년부터 오픈 프로젝트로 개발하고 있는 네트워크 트래픽 모니터링 및 분석 시스템이다. Ntop은 'Network top' 이라는 뜻으로 유닉스의 시스템 자원 상태를 보여주는 top 명령어를 네트워크 모니터링 및 분석 시스템에 응용했다는 뜻이다. Ntop은 웹기반에서 호스트 정보, 프로토콜 정보, 어플리케이션 정보 등을 분석해 준다. 호스트 중심으로 분석 정보를 제공하여 특정 호스트에서 나가고 들어오는 패킷의 양 이라든가 특정 호스트에서 사용되는 프로토콜을 확인하기 편리하다. Ntop은 실시간 트래픽 분석 방법을 사용하는 시스템이다. Ntop은 패킷 하나를 캡처 할 때마다 정해진 분석 규칙에 따라 분석 정보를 업데이트 한다. 예를 들어, 들어온 패킷이 IP 패킷인지 ARP 패킷인지를 판단하여 IP 패킷이면 IP 패킷의 총 트래픽량에 그 패킷의 크기를 합산한다. 누적된 네트워크 트래픽의 총량을 확인하기에 좋은 방법이다. Ntop은 시스템 과부하로 인한 패킷 손실을 막기 위해 패킷 분석 모듈에 스레드를 사용하는 등 시스템 자원을 적게 사용하도록 만들었다. 그러나 대용량 트래픽에서는 트래픽 분석 및 모니터링으로 인해 시스템 과부하가 발생하고 패킷 손실이 발생한다. 또 실시간 트래픽 분석 방법을 사용하여 정해진 규칙으로만 분석 할 수 있고 장기간 동안의 네트워크 트래픽 분석에도 적당치 않다. 호스트 중심의 분석 기능

은 조사한 네트워크 트래픽 분석 시스템들 중 가장 뛰어났으나 분석에 제한이 있고 대용량 트래픽을 처리하지 못하며, 장기간 동안의 네트워크 트래픽 분석을 제공하지 못하는 단점이 있다.

2.3.3 WebTrafMon

WebTrafMon [6]은 일괄 처리 트래픽 분석 방법을 사용하는 시스템이다. WebTrafMon은 패킷의 헤더 정보를 한 시간 동안 저장해 두었다가 과거 한 시간 동안의 네트워크 트래픽 정보를 분석해 준다. 시간 별로 증가와 감소를 반복하는 네트워크 트래픽의 동향을 파악하기에 좋은 방법이다. 매 시간별로 네트워트 총량만을 확인 할 수 있는 Ntop과는 달리 한 시간 동안의 패킷 헤더 정보를 가지고 호스트 정보, 프로토콜 정보 등을 확인 할 수 있다. 그러나 매 시간 별 트래픽 확인이 24 시간 동안만 가능하고 호스트와 프로토콜의 연관 관계를 알 수 없는 단점이 있다.

2.3.4 Ethereal

Ethereal [7]은 패킷 단위 분석 혹은 파일에 저장해 놓은 네트워크 트래픽 정보를 분석해 주는 프로그램이다. 패킷 단위 분석 방법과 트래픽 분석 방법을 동시에 제공해 주는 장점이 있으나 패킷 단위 분석 방법에 더욱 가까워 모니터링 개념은 결여되어 있다. GTK+ 기반으로 유닉스 플랫폼의 X 윈도우 화면에서 동작하며 MS 윈도우즈용도 있다. 여러 다양한 프로토콜 분석을 지원하는 장점이 있다. Gerald Comb가 50 명 이상의 프로그래머들과 함께 오픈 프로젝트로 진행되고 있다. Ethereal은 패킷 단위 분석에 적당한 프로그램으로 장기간 동안의 네트워크 트래픽 분석에는 적당치 않다.

2.3.5 NNStat

NNStat [8]은 Robert Braden과 Annette DeSchon이 1988 년에 발표한 프로그램으로 여러 네트워크 포인트에서의 트래픽을 합쳐서 분석 할 수 있으며 사용자가 지정하는 장기간의 네트워크 트래픽 분석도 지원한다. NNStat은 SAA(Statistics Acquisition Agent)와 SCH(Statistics Collection Host)로 구성되어 있다. SAA는 여러 네트워크 노드에서 네트워크 트래픽을 모으는 모듈이고 SCH는 분석하는 모듈이다. 각 노드에 SAA가 하나씩 있게 되고 각각의 SAA에서 모은 데이타를 SCH에 보내준다. SAA에는 분석 시간을 설정하는 기능이 있어 사용자가 분석 시간을 설정할 수 있다. NNStat은 SunOS 4.0 NIT(Network Interface Tap) 인터페이스를 사용하는 시스템에만 지원되는 단점이 있다.

2.3.6 기타

Tcpdump [9]는 패킷 단위 분석을 제공해 주는 시스템이며 Tcpslice [10]는 Tcpdump의 패킷 캡처 포맷을 파일에 다양하게 저장 할 수 있는 기능을 제공해 준다. snoop [11]은 Etherfind의 새로운 버전으로 SunOS 5.x에 기본적인 패키지로 들어가 있다. snoop은 tcpdump와 기능이 비슷하며 역시 패킷 단위 분석 도구이다. 그외 argus [12], arpwatch [13], nsfwatch [14], drawbridge [15] 등은 네트워크 보안을 목적으로 한 도구로 네트워크 트래픽 모니터링에는 적당치 않다. 또 MS 윈도우즈 전용 프로그램으로 ewatch [16]와 sniffer pro [17] 같은 프로그램이 있다. ewatch와 sniffer pro는 상용 프로그램으로 트래픽 분석 시 편리한 UI를 제공해 주는 것이 장점이나 장기간 동안의 네트워크 트래픽 모니터링 및 분석 기능을 제공해 주지 않는다.

2.3.7 비교

2.3 에서 조사한 네트워크 트래픽 분석 시스템 들 중 특징있는 몇 가지 시스템을 선택하여 비교해 본다. 다음 표 1 은 네트워크 트래픽 분석 시스템의 비교를 나타낸다.

표 1 네트워크 분석 시스템 비교

	Capture Method	Analysis Method	Analysis Interval	Analysis Scope	Load Distribution	User Interface
Tcpdump	libpcap	packet by packet	current	layer 4	no	Text
Ntop	libpcap	real-time traffic analysis	5 second, hourly	layer 2-7	no	Web
MRTG	snmp agent	batch traffic analysis	5 minute, hourly, daily, weekly, monthly	layer 2	yes	Web
WebTrafMon II	libpcap	batch traffic analysis	1 minute, hourly, daily, monthly, yearly	layer 2-7	yes	Web

표 1 에서 제일 밑에 있는 WebTrafMon II는 새로 설계 및 구현할 네트워크 트래픽 모니터링 및 분석 시스템의 이름이다. 표 1 에서 'Capture Method'는 패킷을 캡처 하는 방식을 말한다. libpcap으로 캡처하여 'Analysis Scope'을 layer 2 에서 layer 7 까지 분석 할 수 있는 방법과 SNMP를 사용해서 layer 2 만 분석 할 수 있는 방법이 있다. Tcpdump는 libpcap을 사용하지만 layer 4까지만 분석 할 수 있다. 'Analysis Method'는

‘packet by packet’, ‘real time traffic analysis’, ‘batch traffic analysis’ 로 나누었다. 2.2 에서 소개한 패킷 단위 분석 방법과 실시간 트래픽 분석 방법과 일괄처리 분석 방법을 각각 의미한다. ‘Analysis Interval’은 분석하여 보여 줄 수 있는 시간 범위를 의미하는 데 current는 프로그램이 동작하는 시간의 정보를 보여주는 것으로 Tcpdump가 그 예이다. 1 minute, hourly 등은 1 분 간격, 한 시간 간격 단위로 트래픽 정보를 보여주는 것을 의미하나, Ntop의 5 second 만은 5 초 간격으로 누적된 네트워크 트래픽 정보를 보여주는 것을 말한다.

또, ‘Load Distribution’은 네트워크 트래픽 분석 시스템의 로드를 분산할 수 있는지의 여부를 의미하는 것으로 MRTG의 경우에는 SNMP Agent가 설치된 시스템과 패킷 정보를 분석하는 시스템이 나누어져 있다. 또 새로 설계하고 구현할 WebTrafMon II도 로드 분산 구조로 갖는다. 마지막으로 ‘User Interface’는 네트워크 분석 시스템의 정보를 확인할 수 있는 환경을 말하는 것으로 웹기반이 필수이다.

표 1 에서 보면 분석 범위가 장기간이면서 응용 계층까지 분석해 주는 프로그램이 없다는 것을 발견 할 수 있다. 일정한 간격의 장기간 동안의 네트워크 트래픽 정보를 제공하는 것은 MRTG밖에 없는데 MRTG는 호스트 정보 혹은 프로토콜 정보 등을 제공하지 못한다. 따라서 장기간 동안의 네트워크 트래픽 정보를 제공해 주면서 호스트 정보 혹은 프로토콜 정보를 분석해 줄 수 있는 프로그램이 필요하다. 또한 libpcap을 사용하여 응용 계층까지 분석 할 수 있으나 로드 분산 구조를 사용하여 대용량 트래픽을 처리하도록 하는 네트워크 트래픽 분석 시스템도 없다. 따라서 이러한 네트워크 트래픽 분석 시스템들의 장단점을 파악하여 WebTrafMon II의 설계 및 구현에 반영한다.

3. 요구사항

서론 및 관련 연구를 바탕으로 해서 최근의 네트워크 트래픽 추세에 맞는 네트워크 트래픽 모니터링 및 분석 시스템의 요구사항을 정리한다. 3.1 에서 일반적인 네트워크 트래픽 분석 시스템의 기본 요구사항을 살펴 본뒤 3.2 부터 기존 네트워크 트래픽 분석 시스템의 문제점 해결을 위한 요구사항을 제시한다.

3.1 기본 요구사항

3.1 에서는 네트워크 트래픽 모니터링 및 분석 시스템의 기본 요구 사항을 살펴본다. 이는 서론에서 제시한 문제점 해결을 위한 요구사항이 아니라 일반적인 네트워크 트래픽 분석 시스템의 요구 사항이다.

네트워크 트래픽을 정확히 분석하기 위해서는 네트워크 트래픽 분석 시스템이 패킷 헤더 정보를 분석하여 네트워크 트래픽의 주된 양을 어떤 호스트가 차지하는지, 사용되고 있는 호스트의 수는 어느 정도인지, 호스트가 주로 사용하고 있는 서비스는 무엇인지 등의 정보를 알려 줄 수 있어야 한다. 이것을 정리하면 다음과 같다.

1) 네트워크 트래픽의 총량을 확인 할 수 있어야 한다. MRTG 등에서 제공하는 기능으로 네트워크 밴드위스의 사용을 확인 할 수 있는 가장 기본 기능이다.

2) 호스트 정보를 분석 할 수 있어야 한다. 특정 시간 동안 어떤 호스트들이 네트워크 트래픽을 발생시켰고 그 중 가장 많은 트래픽을 발생시킨 호스트는 무엇인지 알 수 있어야 네트워크의 문제점을 파악할 수 있다.

3) 사용되고 있는 전송 프로토콜(예: arp, ip, udp, tcp) 및 어플리케이

션 계층의 프로토콜(예: ftp, http, mms, rtp, snmp, telnet)을 분석 할 수 있어야 한다. 프로토콜을 분석해야 어떤 서비스가 어느 정도 사용되는 지 알 수 있다.

4) 호스트와 프로토콜과의 관계를 분석 할 수 있어야 한다. Ntop을 제외한 대부분의 네트워크 트래픽 분석 시스템들이 이 기능을 제공해 주지 못한다. 어떤 호스트가 어떤 서비스를 사용했는지, 어떤 서비스를 가장 많이 이용한 호스트는 무엇인지 등을 알 수 있어야 한다.

5) 웹기반의 사용자 인터페이스를 제공해야 한다. 웹기반의 사용자 인터페이스는 여러 사람이 언제 어디서나 정보를 편리하게 확인 할 수 있는 장점이 있다.

3.2 분 단위와 시 단위의 단기간 모니터링

대부분의 네트워크 트래픽 분석 시스템들이 단기간 모니터링을 지원하는 것처럼 보이지만 실제로 찾아보면 의외로 단기간 동안의 모니터링을 제공하는 네트워크 트래픽 분석 시스템을 찾기 힘들다. Ntop이 한 시간 단위의 모니터링을 제공하나 네트워크 트래픽의 총량을 확인 할 수 있을 뿐이며 하루가 지나면 모든 정보가 없어진다. WebTrafMon도 한 시간 단위의 모니터링을 제공하나 24 시간 범위에서 이루어 지고, 호스트와 프로토콜의 관계는 확인 할 수 없다. 실시간에 가까운 모니터링에서부터 최소한 하루 동안의 트래픽 동향을 파악해야 하는 네트워크 트래픽 모니터링의 특성을 고려한다면 분 단위와 시 단위의 단기간 모니터링이 적당하다.

3.3 일, 월, 년 단위의 장기간 모니터링

장기간 네트워크 트래픽 모니터링은 계속해서 증가하는 네트워크

트래픽의 특성상 네트워크 회선 계획에 중요한 일이므로 달 별과 년 별의 네트워크 트래픽 모니터링을 통해 증가 추세를 파악하는 것이 중요하다. 여기에 일 별의 네트워크 트래픽 모니터링을 추가하면 구체적으로 언제부터 어떤 문제로 인해 네트워크 트래픽이 증가했는지를 파악할 수 있다. 장기간 동안의 네트워크 트래픽 모니터링 기능을 제공해 주는 네트워크 분석 시스템이 없는 것과 급격히 증가하는 네트워크 추세를 고려할 때 일, 월, 년 단위의 장기간 모니터링은 중요한 요구사항이다.

3.4 패킷 손실 방지

패킷 손실이 일어나면 정확한 네트워트 트래픽 분석이 이루어 질 수 없음에도 대부분의 네트워크 트래픽 분석 시스템들이 이에 대한 대비가 이루어 지지 않고 있다. 패킷 손실 방지는 네트워크 트래픽을 정확하게 분석하기 위한 중요한 요구사항으로 로드를 분산하고 패킷을 캡처하는 시간을 줄임으로 패킷 손실을 막을 수 있다.

3.5 대용량 트래픽 분석

10 Mbps ~ 100 Mbps의 대용량 트래픽을 분석 할 수 있어야 한다. 많은 네트워크 트래픽 분석 시스템들이 대용량 트래픽을 처리하지 못해 무용지물이 되고 만다. 로드 분산 구조에서 대용량 트래픽을 저장한 뒤 패킷 해쉬를 통해 데이터베이스에 저장하면 대용량 트래픽을 적당한 시간 안에 처리 할 수 있다.

3.6 멀티미디어 서비스 등의 새로운 서비스 분석

최근에는 스트리밍, P2P 서비스 등의 멀티미디어 트래픽이 증가하고 있다. 이러한 멀티미디어 트래픽을 분석 할 수 있어야 한다. 또, 멀티미디어 서비스 뿐만 아니라 새로 개발되는 네트워크 기반의 응용 프로그램의 트래픽 분석도 반영 할 수 있어야 한다.

4. 설계를 위한 문제점 해결 방법

4 장에서는 요구 사항을 만족하는 설계를 하기 전에 서론에서 살펴본 문제점을 해결하기 위한 방법을 제시한다. 대용량 트래픽 처리, 패킷 손실 방지, 장기간 동안의 모니터링 등의 문제점을 해결할 수 있는 방법이 명확해야 요구사항에 맞는 설계를 할 수 있다.

대용량 네트워크 트래픽을 장기간 동안 분석하고 모니터링 하기 위해서는 일괄처리 트래픽 분석 방법이 적당하다. 일괄처리 트래픽 분석 방법은 대용량 네트워크 트래픽을 헤더 정보만 간단히 저장하므로 패킷 손실이 적고 장기간 동안의 모니터링을 위한 트래픽 정보를 연속적으로 저장해 나갈 수 있기 때문이다. 그러나 일괄처리 트래픽 분석 방법은 트래픽 분석으로 인한 시스템 과부하로 패킷 손실이 발생하고 대용량 네트워크 트래픽을 장기간 동안 저장하기에는 저장 공간의 문제점이 있다. 따라서 4 장에서는 일괄처리 트래픽 분석 방법을 개선해서 패킷 손실을 막고 대용량 네트워크 트래픽을 장기간 동안 모니터링 및 분석 할 수 있는 방법을 연구한다. 로드 분산 방법을 통해서 패킷 캡처 모듈 만의 시스템 자원 사용을 보장하고 패킷 캡처 시간과 패킷 손실의 관계, 대용량 패킷 처리를 위한 패킷 해쉬, 데이터베이스를 이용한 장기간 동안의 데이터 저장, 멀티미디어 서비스의 포트 번호 조사 등을 연구한다.

4.1 로드 분산 방법

네트워크 트래픽 분석 시스템의 가장 큰 문제점은 패킷 손실이다. 대용량 트래픽에서의 패킷 손실은 물론이고 트래픽이 적은 곳에서도 패킷 캡처 모듈이 시스템 자원을 보장 받지 못하면 패킷 손실이 일어난다. 일괄처리 트래픽 분석 방법에서는 패킷 분석 모듈이 패킷 캡처에 영향을 미치고 패킷 캡처 모듈도 패킷 분석 모듈에 영향을 미치기 때

문에 패킷 손실도 발생하고 패킷을 분석하는 시간도 늦어진다. 따라서 이러한 네트워크 트래픽 분석 시스템의 문제점을 해결하기 위하여 패킷 캡처 모듈, 패킷 분석 모듈, 프리젠테이션 모듈을 각각의 시스템으로 나눈 로드 분산 네트워크 트래픽 모니터링 및 분석 구조를 제시한다. 다음 그림 2 는 기존의 로드 집중 네트워크 트래픽 모니터링 및 분석 시스템 구조와 로드 분산 네트워크 트래픽 모니터링 및 분석 구조를 비교한 것이다.

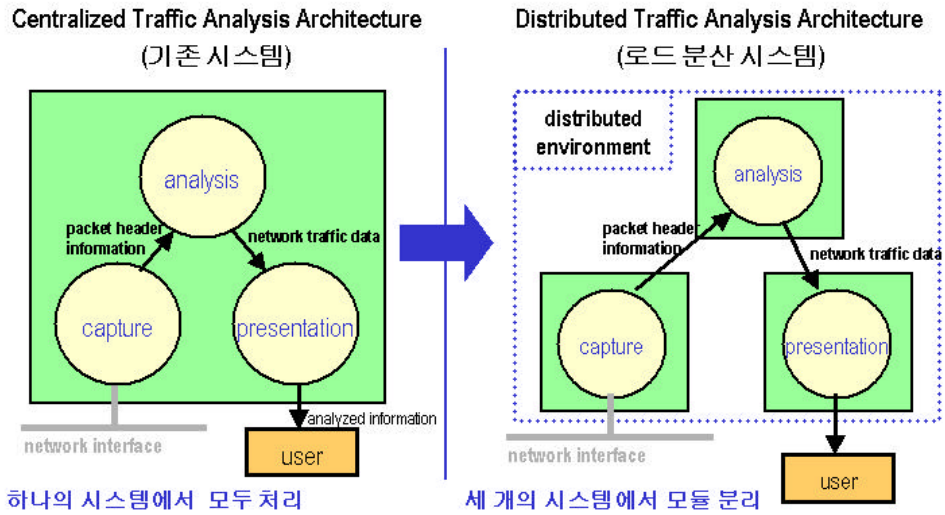


그림 2 로드 집중 방법 Vs. 로드 분산 방법

기존의 네트워크 트래픽 분석 시스템에서는 하나의 시스템에서 패킷 캡처, 패킷 분석, 프리젠테이션 등을 동시에 처리한다. 따라서 시스템 로드로 인한 패킷 손실이 발생하고 시스템 로드로 인해 분석 및 응답 시간이 지연된다. 이러한 문제점을 해결하기 위해 로드 분산 네트워크 트래픽 모니터링 및 분석 구조에서는 패킷 캡처 모듈, 패킷 분석 모듈, 프리젠테이션 모듈 등을 각각의 시스템에서 처리하도록 분산 환

경을 만들었다. 로드 분산 방법을 사용하면 패킷 캡처 모듈이 독립된 시스템에서 동작하므로 트래픽 분석 및 프리젠테이션의 로드로 인한 패킷 손실을 막을 수 있다. 또 트래픽 분석 및 프리젠테이션 모듈도 각각의 시스템의 자원을 보장 받으므로 네트워크 트래픽을 로드 집중 시스템에서 보다 빠르게 분석하여 사용자에게 보여 줄 수 있다.

4.2 패킷 캡처 시간과 패킷 손실

로드 분산 방법을 사용해서 패킷 캡처 모듈의 시스템 자원 사용을 보장해 주었다고 패킷 손실이 없는 것은 아니다. 패킷 캡처 모듈의 패킷 처리 시간에 따라 패킷 손실이 발생 할 수 있다. 다음 그림 3 은 패킷 캡처 모듈에서 하나의 패킷을 처리하는 시간에 따른 패킷 손실률을 측정하기 위한 테스트 베드이다.

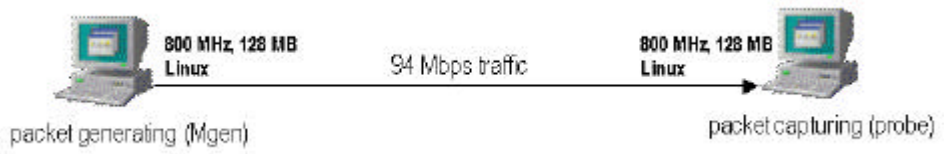


그림 3 패킷 손실 테스트 베드

펜티엄 800 MHz, 128 MB 의 PC 두 대를 연결하여 그림 3 의 왼쪽 위의 PC에서는 패킷을 발생시키고 오른쪽 위의 PC에서는 패킷을 캡처한다. 두 대 모두 각각 패킷을 발생시키고 패킷을 캡처 하는 일에만 CPU 등의 시스템 자원을 사용한다. 패킷을 발생시키는 프로그램으로는 Mgen [18]을 사용한다. Mgen은 패킷의 크기와 초당 발생하는 UDP 패킷의 수를 정해서 일정 시간 동안 원하는 시스템으로 보낼 수 있는 프로

그램이다. 이 실험에서는 1036 bytes의 프레임을 40 분 동안 11,400 pkt/s (초당 패킷 수)로 보내어 94 Mbps의 트래픽을 만들었다. 패킷을 캡처 하는 프로그램으로는 직접 구현한 probe를 사용하였다. probe는 libpcap 라이브러리를 사용하여 패킷을 캡처한 뒤 패킷 하나씩의 헤더 정보를 저장하는 프로그램이다.

Mgen을 사용하여 94 Mbps의 트래픽을 probe가 실행되고 있는 시스템에 보내고 probe는 패킷을 캡처 한다. Mgen을 이용한 94 Mbps의 트래픽은 고정시키고 probe를 이용하여 하나의 패킷 처리 시간을 조절하여 패킷 손실률을 측정한다.

패킷 하나의 처리 시간을 변경해 가면서 probe가 동작하고 있는 시스템의 패킷 손실률을 측정한 결과는 다음 표 2와 같다.

표 2 패킷 하나 처리 시간과 손실률

패킷 하나 처리 시간 (usec)	손실률 (%)
0.5	0.01
2	0.56
10	0.9
100	46.72
200	65.21
600	91.19

표 2 의 패킷 하나 처리 시간에서 0.5 usec은 패킷을 캡처한 뒤 패킷 헤더 정보를 저장하지 않을 때 걸리는 시간이며, 2 usec은 바이너리 파일에 패킷 헤더 정보를 저장 하면서 곱셈 연산으로 약간의 오버헤드를 붙였을 때의 시간이다. 10 usec은 텍스트 파일에 저장 할 때, 100 usec은 화면으로 출력할 때, 200 usec은 데이터베이스에 저장 할 때의 시간이다. 또, 600 usec은 임의의 오버헤드를 추가해서 만든 처리 시간이다.

표 2 의 손실률은 패킷 이 손실된 정도를 말해 준다. Mgen에서 보낸 패킷의 수를 확인하고 probe에서 캡처한 패킷을 확인한 뒤 손실된 만큼을 %로 계산한다. 표 2 를 그래프로 그리면 다음 그림 4 와 같다.

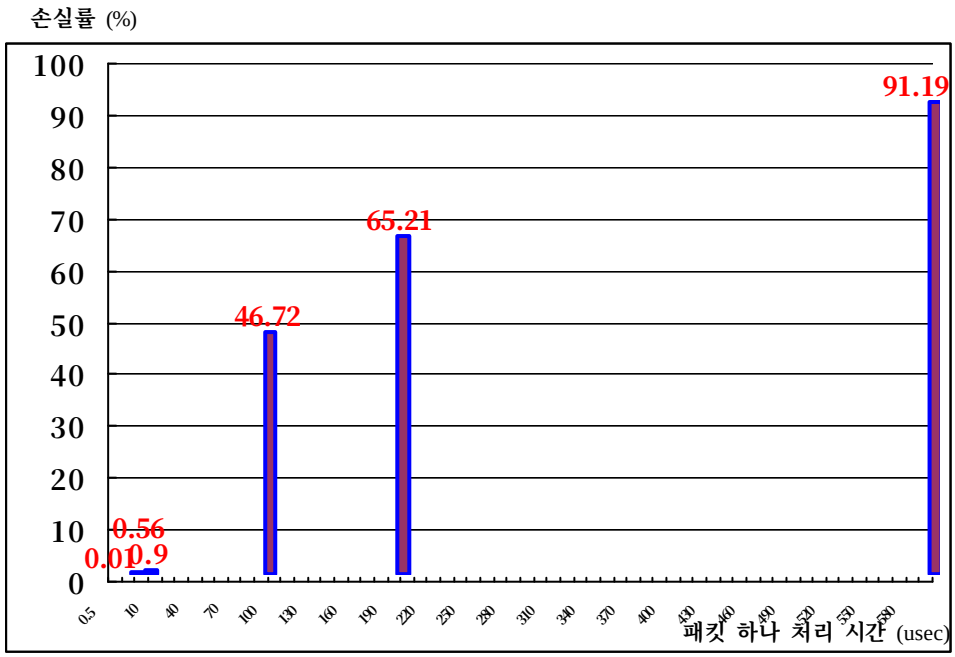


그림 4 패킷 하나 처리 시간과 패킷 손실률

그림 4 와 표 2 에서 알 수 있듯이 패킷 처리 시간이 2 usec 이내일 경우에는 패킷 손실이 아주 미미하다. 약 0.5 % 정도의 패킷 손실은 대용량 트래픽을 정확히 분석하는 데 큰 문제가 되지 않는다. 그러나 패킷 처리 시간이 100 usec을 지나면서 패킷 손실이 50 % 이상으로 급증하는 것을 알 수 있다. 패킷 처리 시간이 데이터베이스에 저장하는 시간인 200 usec에서는 약 65 %에 이르러 약 1/3 이 손실됨을 알 수 있다. 이실험을 통해 800 MHz, 128 MB 보다 사양이 높은 PC 에서 패킷

캡처 어플리케이션이 패킷을 2 usec 내로 처리하면 94 Mbps의 정도의 대용량 트래픽에서도 패킷 손실이 거의 없음을 알 수 있다.

패킷 손실을 막으려면 패킷 캡처 모듈이 패킷 헤더 정보를 직접 데이터베이스에 저장하지 않고, 바이너리 로그 파일 등을 사용해야 한다는 결론을 내릴 수 있다.

4.3 패킷 손실 예

포항공대 인터넷 회선에서 MRTG와 Ntop을 동시에 비교 조사했다. Ntop이 사용된 시스템은 1.0 GHz, 256 MB의 PC로 커널의 로그 파일을 통하여 커널에서 일어나는 패킷 손실이 없음을 확인했다.

MRTG는 SNMP를 이용하여 네트워크 총량을 확인 할 수 있도록 제공해 주는 네트워크 트래픽 분석 시스템이다. SNMP를 사용하기 때문에 패킷을 캡처할 때 로드가 적어 패킷 손실 없이 정확한 정보를 확인할 수 있다. Ntop은 libpcap 을 사용하여 패킷을 캡처하는 네트워크 트래픽 분석 시스템이다.

포항공대 인터넷 회선의 미러링 회선에서 MRTG와 Ntop에서 보여주는 네트워크 총량을 5 분 동안 측정하여 비교하였다. 결과는 다음 표 3 과 같다.

표 3 Ntop의 패킷 손실 테스트

	5 분 평균 (Mbps)	손실률 (%)
MRTG	58.2	0 %
Ntop	9.3	84 %

MRTG는 5 분 동안의 총량을 58.2 Mbsp를 표시하는 데 비해 Ntop 은 5 분 동안 9.3 Mbps를 표시하여 패킷 손실률이 약 84 %에 이르는 것을 알 수 있다. Ntop의 경우, 패킷 캡처, 패킷 분석, 웹 프리젠테이션 등을 한 시스템에서 동시에 하므로 대용량 트래픽에서 시스템 과부하로 인해 표 3 과 같은 패킷 손실이 발생한다. 따라서 Ntop으로는 대용량 트래픽을 정확하게 분석 할 수 없음을 알 수 있다.

4.4 패킷 해쉬와 데이터베이스

패킷 손실이 없다고 해도 대용량 트래픽을 정확하게 분석 할 수 있는 것은 아니다. 40 ~ 90 Mbps 의 대용량 트래픽에서는 분당 40 ~ 100 만개의 패킷이 발생한다. 이렇게 많은 패킷들을 사용자 요구 시간에 안에 분석해야 하며, 장기간 동안 저장할 수 있어야 한다. 그런데, 기존 네트워크 분석 시스템들의 대부분이 이러한 대용량 네트워크 트래픽을 처리하도록 설계 되어 있지 못하고 장기간 동안의 트래픽 정보를 저장하지 못한다.

대용량 트래픽을 처리하기 위해서는 먼저 트래픽의 유형을 파악해야 한다. 대용량 트래픽은 단순히 총량이 높은 트래픽만을 의미하지 않는다. 대용량 인터넷 회선에는 많은 호스트 들이 물려 있고 많은 서비스 들이 사용된다. 예를 들어, 2 만개의 호스트 중에서 http 서비스를 사용한 호스트를 찾아내어 사용자에게 보여 준 다거나 1 년간의 트래픽 정보를 모아 가장 많이 트래픽을 발생시킨 호스트를 찾아내는 일이 있다. 많은 호스트 들이 사용한 각 서비스 들을 모두 확인하고 장기간 동안의 정보를 저장하기 위해서는 데이터베이스의 사용이 필수이다. Ntop 과 같이 해쉬 테이블을 사용할 경우에는 해쉬 테이블의 한계가 분명하기 때문에 모니터링 할 수 있는 호스트 수와 분석하는 서비스에 제한을 둘 수 밖에 없다.

데이터베이스를 사용할 경우에는 대용량 트래픽의 패킷 헤더 정보를 데이터베이스에 넣을 때 시스템 과부하가 발생하는 문제점이 있다. 또, 장기간 동안 트래픽 정보를 저장 할 경우 저장 공간이 급증하는 문제점도 있다. 또, 대용량 트래픽을 그대로 데이터베이스에 저장할 경우에는 정보를 검색하는 데도 응답 시간이 길어 진다. 이러한 문제점을 해결하기 위해 해쉬 알고리듬을 이용하여 같은 패킷을 합치는 방법을 사용한다. 다음 그림 5는 패킷 해쉬에 대해서 설명한다.

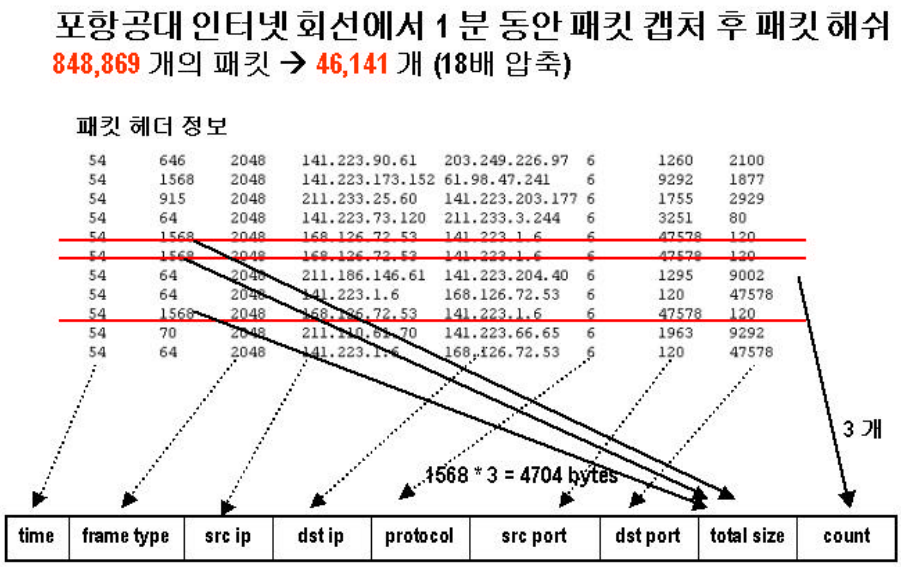


그림 5 패킷 해쉬

그림 5는 포항공대 인터넷 회선에서 패킷을 캡처 했을 때를 예로 해서 패킷을 해쉬 하는 방법을 설명한다. 패킷의 타임 스탬프를 분 단위로 정한 뒤 패킷의 헤더 정보를 프린트 해보면 패킷의 크기만 조금씩 다를 뿐 패킷의 송, 수신지 및 사용된 서비스가 같은 패킷이 있는

것을 알 수 있다. 이렇게 같은 패킷들을 해쉬 알고리듬을 통해 패킷 크기를 더하고 패킷 수를 계산해서 모아 하나의 패킷 포맷으로 모은다.

그림 5 의 밑줄 친 패킷이 같은 송, 수신지 및 서비스를 갖는 패킷이다. 밑줄 친 세 개의 패킷의 크기를 합쳐서 4704 로 계산하고 count 를 3 으로 표시하면 밑줄 친 패킷 세 개를 저장하는 것과 같은 의미를 갖는다.

포항공대 인터넷 회선에서 1 분 동안 848,869 개의 패킷을 캡처한 뒤 같은 종류의 패킷을 합친 결과 46,141 개로 줄었다. 약 18 배가 압축된 것으로 같은 종류의 패킷이 짧은 시간 동안 몰려 있음을 알 수 있다. 패킷을 이렇게 해쉬를 통해 합쳐 놓으면 데이터베이스에 저장하는 시간이 30 배 정도 단축된다. 대용량 인터넷 회선에서 1 분 동안 패킷을 캡처 하여 패킷 하나씩을 데이터베이스에 저장하는 데는 Pentium 800 MHz, 256 PC에서 약 8 분 정도가 걸렸다. 8 분 이상이 걸린 다는 말은 1 분 동안 캡처한 내용을 8 분 뒤에야 보여 줄 수 있다는 뜻으로 응답 시간의 심각한 지연 문제가 발생한다. 그러나 패킷을 해쉬 하여 저장하면 Pentium 800 MHz, 256 PC에서 약 20 초 정도면 데이터베이스에 저장 할 수 있다. 따라서 대용량 트래픽을 처리해야 하는 문제점이 해결되며 장기간 동안 저장해도 저장 공간이 10 배 이상으로 줄며, 트래픽 분석 시간도 상당히 절약된다. 또, 패킷을 해쉬 하지 않는다면 데이터베이스에 848,869 개의 칼럼에서 정보를 검색해야 하나, 패킷을 해쉬 하면 46,141 개의 칼럼에서 정보를 검색하므로 응답 시간이 향상된다.

4.5 멀티미디어 서비스 포트 번호 조사

기존의 네트워크 트래픽 분석 시스템은 멀티미디어 서비스를 확인 할 수 없다. 패킷 분석 방법을 사용하는 일부 네트워크 분석 시스템에서 RTP [19] 프로토콜을 분석하려고 시도하였으나 RTP 프로토콜을 사

용하는 서비스의 UDP/TCP 포트 번호를 알지 못해 분석하지 못하였다.

2.3.4 에서는 스트리밍 서비스 혹은 P2P 서비스 등의 멀티미디어 서비스의 고정된 UDP/TCP 포트 번호가 있는 지 조사한다. 고정된 UDP/TCP 포트 번호가 있다면 멀티미디어 서비스를 분석 할 수 있기 때문이다.

Real Media [20]의 UDP/TCP 포트 번호 조사를 예로 들면, Real Server와 Real Player에서 오고 가는 패킷을 Windump [21]와 Netmon [22]을 사용하여 조사하였다. Windump는 Tcpdump처럼 패킷 하나씩의 동작을 확인하는 도구이며 Netmon은 사용되고 있는 호스트의 UDP/TCP 포토 번호를 알려 주는 도구이다. Windump와 Netmon을 이용해서 정밀하게 패킷 분석을 해본 결과에 의하면 Real Media에서 RTP 프로토콜을 사용해서 데이터를 전송할 경우에 클라이언트 포트가 6972 번으로 고정되어 있었다. 이와 같은 방법으로 Windows Media [23]의 Windows Media Server와 Windows Media Player에서 오고 가는 패킷을 조사한 결과 Windows Media 에서는 MMST [24] 프로토콜 사용되었으며 서버측 포트 번호가 1775 번으로 고정되어 있었다. 또, P2P 서비스의 FastTrack [25], Gnutella [26], FreeNet [27] 등의 프로토콜을 Windump와 Netmon을 이용해서 확인해 본 바에 의하면 각각이 고정된 UDP/TCP 포트 번호가 있었다. 다음 표 4 는 조사한 멀티미디어 서비스의 UDP/TCP 포트 번호 이다.

표 4 멀티미디어 서비스의 포트 번호 조사

서비스	프로토콜	UDP/TCP 포트 번호
Real Media	RTP (UDP)	클라이언트 6972
Windows Media	MMST (TCP)	서버 1755
Quick Time	RTP(UDP)	?
FastTrack	fasttrack (TCP)	서버 1214
Gnutella	gnutella (TCP)	6347
FreeNet	freenet (TCP)	19114

표 4 를 통해 멀티미디어 서비스도 telnet, ftp 등과 같이 UDP/TCP 포트 번호를 통해 분석 할 수 있음을 알 수 있다. 이러한 UDP/TCP 포트 정보를 사용자를 통해서 수시로 입력 받아 분석 시에 이 정보들이 반영하여 멀티미디어 서비스 등의 네트워크 기반의 새로운 서비스들을 분석 할 수 있다.

5. WebTrafMon II 설계

이 장에서는 네트워크 트래픽 모니터링 및 분석 기본 요구 사항과 단기간 모니터링, 장기간 모니터링, 패킷 손실 방지, 대용량 트래픽 분석, 멀티미디어 서비스 등의 새로운 서비스 분석 등의 요구사항을 만족하는 네트워크 트래픽 모니터링 및 분석 시스템의 설계에 대하여 설명한다. 로드 분산 방법을 사용한 시스템 설계와 패킷 캡처 모듈, 로그 포맷, 패킷 분석 모듈, 데이터베이스 스키마, 웹기반 트래픽 분석 모듈 등에 대하여 설명한다.

5.1 시스템 구조

로드 분산 방법을 사용한 네트워크 트래픽 모니터링 및 분석 시스템 구조는 그림 6과 같다.

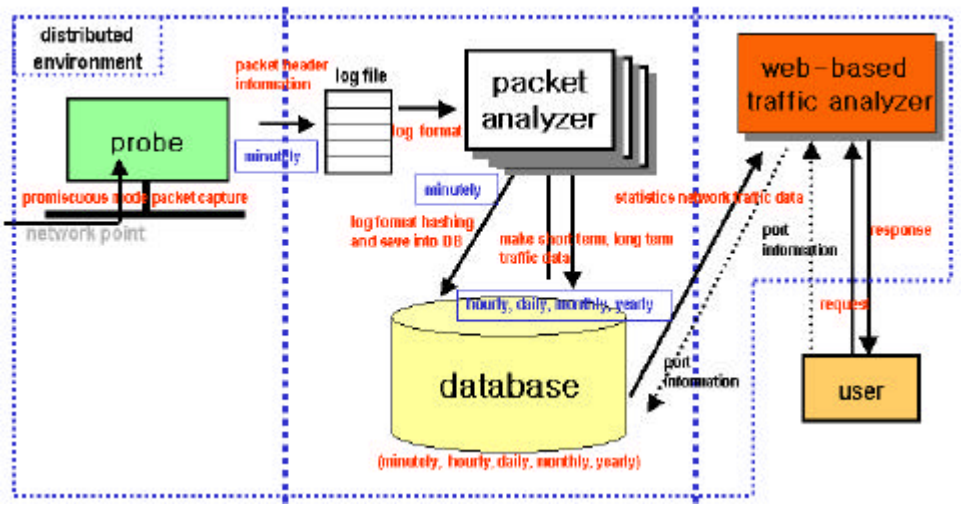


그림 6 WebTrafMon II 시스템 구조

그림 6 와 같이 시스템 구조는 분산 환경에서의 로드 분산 방법을 기본으로 한다. 분산 환경에서 네트워크 트래픽 분석 시스템의 중요한 모듈 세 가지를 각각의 시스템으로 나누었다. 패킷 캡처 모듈(probe), 패킷 분석 모듈(analyzer), 웹기반 트래픽 분석(web-based traffic analyzer) 모듈이 그 세 가지 모듈이다. 패킷 캡처 모듈은 패킷을 캡처한 뒤 패킷 헤더 정보를 로그 파일을 저장한다. 이렇게 저장한 로그 파일을 패킷 분석 모듈이 분 단위로 읽어 패킷을 해쉬 한 뒤 데이터베이스에 저장한다. 또, 패킷 분석 모듈은 분 단위로 저장되는 트래픽 정보를 시 단위로, 일 단위로, 월 단위로, 년 단위로 분석하여 장기간 동안의 트래픽 정보를 만들어 간다. 웹기반 트래픽 분석 모듈은 패킷 분석 모듈이 장기간 동안 분석하여 데이터베이스에 저장한 패킷 헤더 정보를 가공하여 사용자 요구에 맞는 트래픽 정보를 제공해 준다. 즉, 패킷 분석 모듈은 단기간 동안의 패킷 헤더 정보를 분석하여 장기간 동안의 패킷 헤더 정보를 만들어 내는 역할을 하며, 이렇게 장단기에 맞게 분석된 패킷 헤더 정보를 가지고 웹기반 트래픽 분석 모듈이 호스트 혹은 서비스 정보를 요청하는 사용자 요구에 맞게 트래픽 정보를 제공해 준다.

5.2 패킷 캡처 모듈 (Probe)

패킷 캡처 모듈은 네트워크 포인트 상의 패킷을 캡처한 뒤 패킷 헤더로부터 원하는 정보를 추출하여 분 단위로 로그 파일에 저장하는 모듈이다. 패킷 캡처 모듈의 구조는 그림 7과 같다.

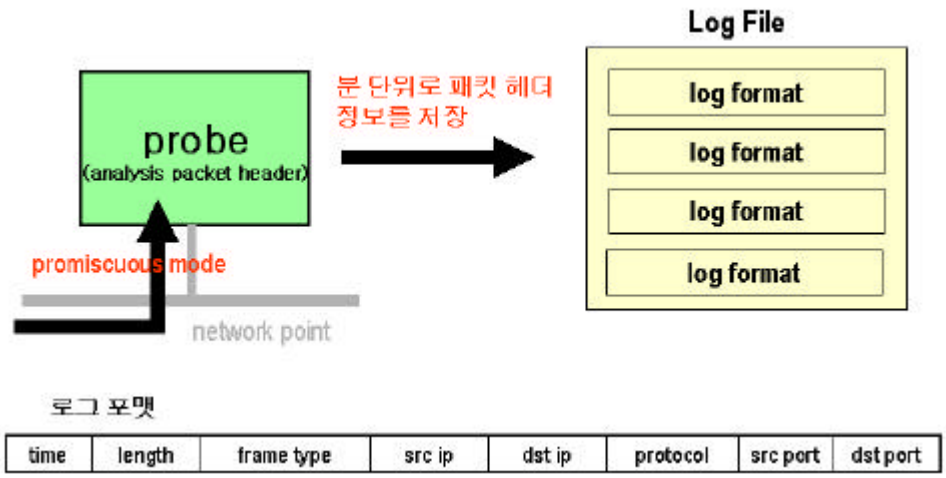


그림 7 패킷 캡처 모듈

패킷 캡처 모듈은 로드 분산 구조에서 독립적인 시스템에서 동작하므로 CPU 등의 시스템 자원 사용이 보장된다. 이렇게 패킷 캡처 모듈만의 일정한 시스템 자원 사용을 보장하면 다른 프로세스의 시스템 과부하로 인한 패킷 손실은 근본적으로 막을 수 있다. 로드 분산 구조에서 시스템 자원 사용을 보장한 후에는 패킷 캡처 모듈 자체의 시스템 로드로 인한 패킷 손실을 막기 위해 패킷 처리를 빨리 해야 한다. 그래서 패킷 캡처 모듈의 패킷 헤더 저장 방법을 바이너리 로그 파일로 설계하였다. 바로 데이터베이스에 저장하는 것이 효과적인 방법 같으나 패킷을 데이터베이스에 넣을 때의 시스템 과부하로 인해 패킷 손실이 발생한다. 바이너리 로그 파일을 사용하면 패킷 하나를 펜티엄 800 MHz, 256 MB 정도의 PC에서 1.5 ~ 2 usec안에 저장 할 수 있어 패킷 손실을 막을 수 있고 데이터베이스에 바이너리 상태로 저장 할 수 있으며 데이터베이스의 공간도 절약된다.

5.3 로그 포맷 (Log Format)

패킷을 캡처한 뒤 패킷의 헤더 정보를 우선 로그 파일에 저장한다. 패킷 헤더에서 필요한 정보만을 골라 로그 포맷으로 로그 파일에 저장한다. 로그 포맷의 구조는 그림 8과 같다.

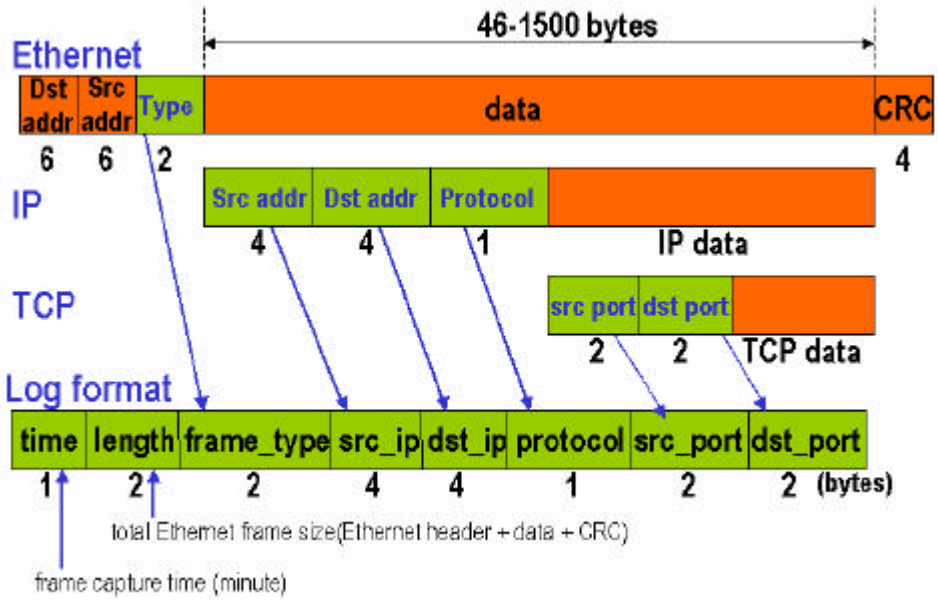


그림 8 로그 포맷

그림 8은 TCP 패킷일 경우를 예로 든 것이다. 로그 포맷의 'time'은 패킷을 캡처해서 로그 파일에 저장 할 때의 시간으로 분 단위로만 표시한다. 예를 들어 캡처할 때의 시간이 3시 5분일라면 '05'라고만 표시한다. 년, 월, 일, 시는 데이터베이스 테이블에 표시되므로 단지 분만 표시해도 패킷의 타임스탬프를 정확히 알 수 있다. time을 1 byte로 줄여서 데이터베이스 공간을 절약할 수 있다. 'length'는 프레임 크기에 CRC 값 (4 바이트)을 더한 값이다. 프레임 크기라는 것은 이더넷 헤더 크기에에 데이터 크기를 더한 값이다.

time과 length를 제외한 나머지 정보는 모두 패킷의 필드에서 가져온다. ‘frame_type’은 이더넷 헤더에서 가져오는 정보로 IP 일 경우는 십진수로 2048 을, ARP 일 경우에는 십진수로 2054 의 값을 갖는다 [28]. ‘src_ip’, ‘dst_ip’, ‘protocol’ 등은 IP 프로토콜 헤더에서 가져오는 정보이다. src_ip와 dst_ip는 송신지 IP 주소와 수신지 IP 주소를 저장한다. 4 byte가 표준으로 정해져 있고 바이너리 포맷 그대로 저장한다. protocol 은 IP 패킷 위에 올라가는 전송 프로토콜을 표시한 정보로 TCP는 십진수로 6 번, UDP는 17 번, ICMP는 1 번이다. ‘src_port’, ‘dst_port’ 등은 UDP/TCP 헤더에서 정보를 가져온다. 송신지 포트와 수신지 포트 번호는 오직 UDP와 TCP 패킷에서만 갖는 정보이면서 UDP와 TCP가 네트워크 서비스의 대부분을 차지하기 때문에 서비스 분석에 중요한 정보이다. UDP/TCP 포트는 telnet일 경우 십진수로 23 번, http일 경우에 십진수로 80 번, FastTrack이 사용하는 kazaa는 1214 번이다.

로그 포맷의 일정 필드를 사용하지 않는 프로토콜은 사용하지 않는 필드를 0 으로 채운다. 예를 들어, ARP같이 IP기반이 아닌 프로토콜은 time, length, ether_type 등의 값만 저장한뒤 사용하지 않는 src_ip, dst_ip, protocol, src_port, dst_port 등은 모두 0 으로 저장한다. 또, ICMP같이 IP기반이지만 포트 정보를 사용하지 않는 프로토콜은 time, length, ether_type, src_ip, dst_ip 등의 값만 저장한뒤 src_port, dst_port 등을 0 으로 저장한다. 포트 번호 중 0 번이 있으나 분석 할 때 UDP/TCP 프로토콜 여부를 먼저 체크한 뒤 포트 분석을 하므로 UDP/TCP 의 포트 번호 0 번과 빈값을 채워넣는 0 값과 구분 할 수 있다.

5.4 패킷 분석 모듈 (Packet Analyzer)

패킷 분석 모듈은 바이너리 로그 파일에 저장된 패킷의 헤더 정보를 읽은 뒤 같은 패킷을 해쉬 알고리듬을 사용하여 패킷들을 압축한 뒤 데이터베이스 저장하고 패킷 헤어 정보를 분석해서 장기간 동안의

네트워크 트래픽 정보를 만드는 모듈이다. 패킷 분석 모듈의 구조는 그림 9와 같다.

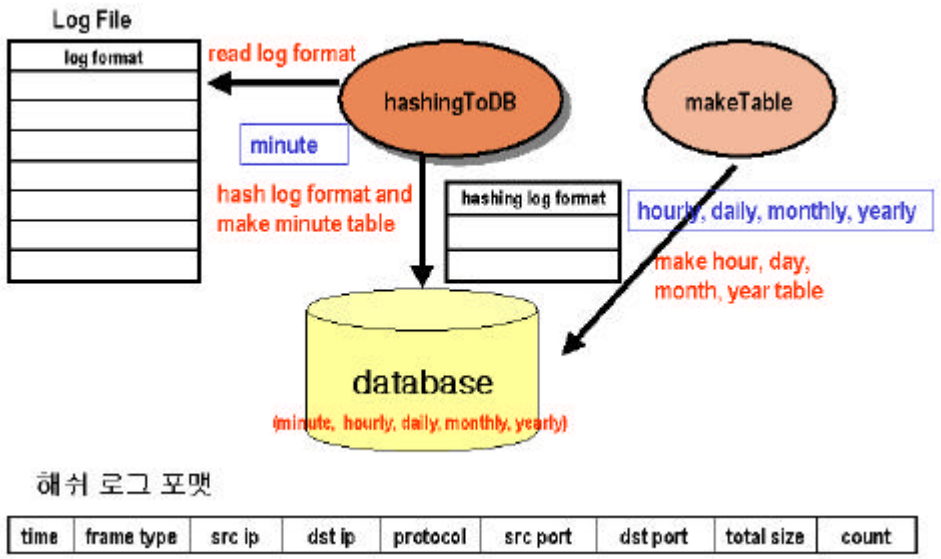


그림 9 패킷 분석 모듈 구조

그림 9 에서 패킷 분석 모듈은 'hashingToDB' 모듈과 'makeTable' 모듈로 구성된 것을 알 수 있다.

hashingToDB 모듈은 1 분 전의 로그 파일을 읽어 로그 포맷 정보를 데이터베이스의 minute 테이블에 저장하는 일을 한다. 대용량 트래픽을 데이터베이스에 빨리 저장하고, 트래픽 분석 시간을 줄이고, 데이터베이스의 저장 공간을 줄이기 위해서 로그 포맷을 2.3.3 에서 소개한 패킷 해쉬 방법을 사용하여 같은 패킷을 합친 뒤 저장한다. 그림 9 에 해쉬 로그 포맷을 표시하였다. 'count'는 합친 패킷의 개수이며 'total size'는 합친 패킷들의 총 트래픽 크기이다. 이런 방법으로 hashingToDB 는 1 분 간격으로 계속해서 실행된다. 즉 저장 공간에 문제가 없는 한

1 분 간격의 네트워크 트래픽 정보를 1 년 이상 저장 할 수 있다.

makeTable 모듈은 hashingToDB 모듈이 만든 minute 테이블을 가지고 한 시간 마다 hour 테이블을 만든다. minute 테이블을 분석하여 1 시간 범위 내에서 같은 포맷을 갖는 칼럼들의 합치어 total size와 count를 계산한 뒤 hour 테이블을 만든다. 즉 hashingToDB의 패킷 해쉬 알고리듬과 결과가 비슷한 방법이다. 또 이렇게 많든 hour 테이블을 가지고 매 일마다 day 테이블을 만들고 같은 방법으로 month 테이블과 year 테이블을 만든다. 이렇게 minute, hour, day, month, year 테이블을 따로 만들어 놓음으로써 사용자 요청을 빨리 처리 할 수 있다. minute 테이블 하나만으로도 year 트래픽 정보를 보여 줄 수 있으나 응답 시간이 엄청나게 지연된다. 따라서 응답 시간을 고려해서 저장 공간이 늘어남에도 테이블을 각각 만들었다. 이렇게 설계함으로써 1 년 동안 http 서비스를 가장 많이 발생시킨 호스트를 찾아 내는 등의 일을 빠른 시간 내에 처리 할 수 있다.

5.5 데이터베이스 스키마 (Database Schema)

데이터베이스 스키마는 해쉬 로그 포맷을 그대로 저장 할 수 있는 구조를 갖는다. 대용량 트래픽에서의 저장 공간을 줄이기 위해 패킷 캡처 상태의 바이너리 포맷 그대로 저장한다. 데이터베이스의 정보를 웹기반 트래픽 분석 모듈에서 접근하기 전에 미리 사용자에게 맞는 정보로 가공해 놓는 방법을 연구하였으나 공간이 너무 늘어나고 대용량 트래픽을 분석하는 시간이 오래 걸려 데이터베이스는 단순히 해쉬된 로그 포맷을 장기간 동안 바이너리 포맷으로 저장한다. 다음 표 5 는 minute 정보를 저장하는 데이터베이스 스키마를 나타낸 것이다.

표 5 minute 데이터베이스 스키마

Field	Type	Null	Key	Default	Field Description
minute	TINYINT UNSIGNED	NO	PRI		0 ~ 59
ether_type	SMALLINT UNSIGNED	NO	PRI		0 ~ 65,535
src_ip	INT UNSIGNED	NO	PRI		0 ~ 4,294,967,295
dst_ip	INT UNSIGNED	NO	PRI		0 ~ 4,294,967,295
protocol	TINYINT UNSIGNED	NO	PRI		0 ~ 255
src_port	SMALLINT UNSIGNED	NO	PRI		0 ~ 65,535
dst_port	SMALLINT UNSIGNED	NO	PRI		0 ~ 65,535
length	BIGINT UNSIGNED	NO			0 ~ 18,446,744,073,709,551,615
count	INT UNSIGNED	NO			0 ~ 4,294,967,295

minute table은 한 시간 동안의 네트워크 트래픽 정보를 저장하는 테이블이다. minute 필드를 0 에서 59 까지 사용하여 각 분 단위의 정보로 구분 할 수 있다. minute 테이블을 만드는 방법에는 minute 테이블 59 개를 만드는 방법과 표 5 와 같이 한 시간 단위의 테이블을 만든 뒤 minute 필드로 구분하는 방법 등이 있다. minute 테이블을 59 개 만들 경

우에는 사용자에게 정보를 조금 더 빨리 보여 줄 수 있는 장점은 있으나 minute 테이블을 hour 테이블로 만들 때 하나로 만든는 것보다 복잡하고 시간이 많이 걸린다. 따라서 표 5 와 같이 minute 테이블을 시간 단위로 만들어 hour 테이블을 간단하게 만들 수 있도록 설계했다. 이런 방법으로hour 테이블도 하루 단위로 만들어 hour 필드로 시간을 구분하였고, day 테이블은 한 달 단위로, month 테이블은 1 년 단위로 만들었다. year 테이블은 하나의 테이블로 만들어 year 필드로 정보를 구분한다.

5.6 웹기반 트래픽 분석 모듈 (Web-based Traffic Analyzer)

웹기반 트래픽 분석 모듈은 패킷 분석 모듈이 분 단위에서 년 단위까지 단기간에서 장기간 동안의 분석해 놓은 정보를 사용자 요청에 맞게 가공해서 보여 주는 일을 한다. 그림 10 은 웹기반 트래픽 분석 모듈의 구조이다.

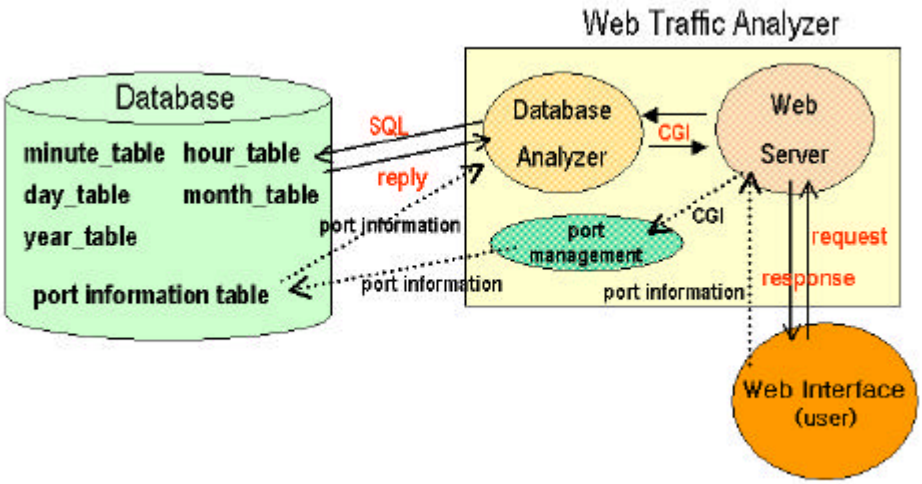


그림 10 웹기반 트래픽 분석 모듈

그림 10과 같이 웹기반 트래픽 분석 모듈은 'Database Analyzer' 모듈과 'port management' 모듈로 구성되어 있다. 그리고 이 두 모듈은 모두 웹 서버의 도움을 받아 웹에서 동작하게 되어 있다.

Database Analyzer는 사용자의 요청을 받으면 그 요청을 SQL로 만들어 데이터베이스에서 검색을 한다. 예를 들어, 2001년 11월 28일 15시 8분에 가장 많은 트래픽을 발생한 호스트 10개를 요청하면 minute_2001_11_28_15_table에서 src_ip, dst_ip, total size, count 등을 참고하여 정보를 검색한 뒤 사용자에게 보여 준다. 예로 든 단순한 정보 뿐만 아니라 어떤 호스트가 어떤 서비스를 사용했는지, 어떤 트랜스포트 계층의 프로토콜이 사용되었고 그런 프로토콜을 가장 많이 사용한 호스트는 무엇인지, 어떤 서비스가 주로 사용되었고 그런 서비스를 가장 많이 이용한 호스트는 무엇인지 알아 낼 수 있다.

port management 모듈은 사용자가 UDP/TCP 포트 정보를 데이터베이스에 입력하고 수정 할 수 있는 기능을 제공한다. 이 모듈은 멀티미디어 서비스 등의 새로운 서비스를 분석 할 수 있도록 도와 준다. 사용자가 Windows Media 서비스가 1775 번의 UDP/TCP 포트를 사용한다는 것을 알고 이러한 정보를 port management 모듈을 이용해 데이터베이스에 입력하면 트래픽 분석 시 즉시 반영된다. 과거의 트래픽 정보도 새로 입력한 정보가 반영되어 분석하므로 서비스의 증가 추세도 확인 할 수 있다.

6. WebTrafMon II 구현

이 장에서는 5 장에서 설계한 네트워크 트래픽 모니터링 및 분석 시스템의 구현에 대해서 설명한다.

6.1 시스템 구조

Pentium III 800, 256 MB의 PC 세 대에 리눅스를 설치한 후 NFS [29]를 이용해 로드 분산 환경을 만들었다. 다음 그림 11 은 NFS를 사용하여 로드 분산 환경을 만든 구현 시스템 구조이다.

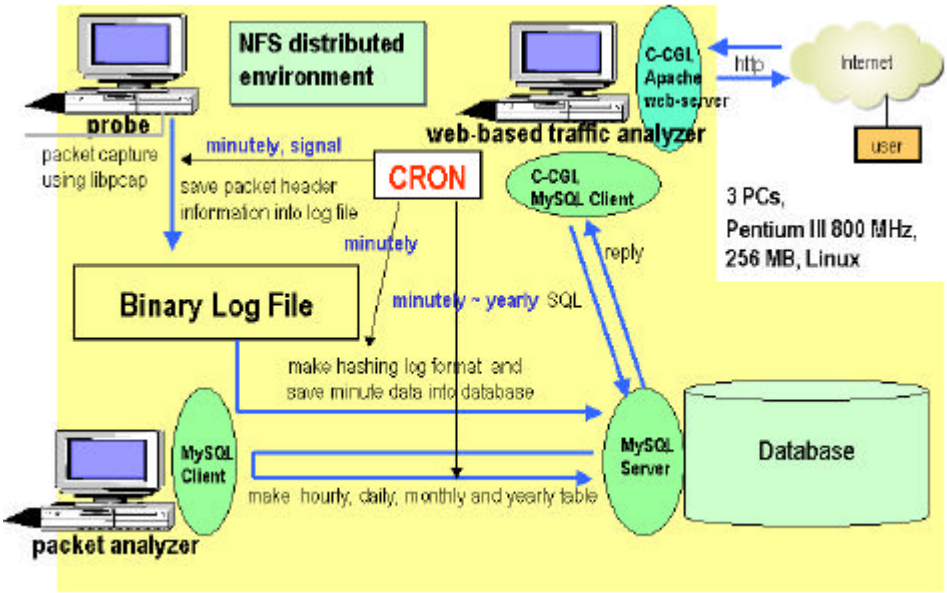


그림 11 WebTrafMon II 구현 시스템 구조

그림 11 에서 'probe'는 패킷 캡처 모듈이고 'packet analyzer'는 패킷 캡처 모듈이며 'web traffic analyzer'는 웹기반 트래픽 분석 모듈이다.

NFS 분산 환경에서 probe, packet analyzer, web traffic analyzer가 각각의 독립된 시스템에서 동작한다.

NFS는 로드 분산 환경을 만들 수 있도록 하며 NFS는 오버헤드도 크지 않다. 리눅스의 crontab [30] 명령을 사용하여 장단기 동안의 패킷 분석을 자동으로 실행되게 설정해 놓았다.

데이터베이스 서버 및 클라이언트 프로그램으로는 Mysql 3.22.32 [31]을 웹 서버로는 Apache Web Server 1.3.14 [32]를 사용하여 구현하였다.

6.2 패킷 캡처 모듈

패킷 캡처 모듈을 구현하기 위해서 C 언어와 libpcap 라이브러리를 사용했다. libpcap 라이브러리에서 제공해 주는 함수들을 이용하면 패킷 캡처 모듈을 promiscuous 모드로 설정하고, 사용자 메모리에서 패킷을 하나씩 읽어 올 수 있다. libpcap 라이브러리는 유닉스 계열의 다양한 운영체제에서 쓸 수 있도록 운영체제에 독립적인 API를 제공한다. 따라서 libpcap 라이브러리를 사용하여 구현하면 운영체제에 상관없이 패킷 캡처 모듈을 실행시킬 수 있다.

사용자 메모리에서 패킷을 하나씩 읽어 올 때 패킷의 38 byte만 읽어 오도록 구현했다. 38 byte는 이더넷 프레임에서 패킷 헤더의 최대 길이이다. 패킷 캡처 모듈에서 읽어 오는 패킷의 길이가 작을수록 좀 더 빠른 패킷 캡처를 할 수 있다.

패킷을 캡처한 뒤 캡처한 헤더 정보를 데이터베이스에 바로 저장하면 패킷 손실이 발생한다. 이는 데이터베이스의 한 칼럼을 저장하는 CPU 로드 때문이다. 그래서 캡처한 헤더 정보를 로그 파일에 저장하였다. 텍스트 로그 파일에 기록하는 것은 데이터베이스에 저장하는 것보다 빠르다. 또, 바이너리 파일은 텍스트 파일에 기록하는 것보다 빠르다.

텍스트 로그 파일을 사용하면 로그 정보를 눈으로 확인 할 수 있는 장점을 갖지만 데이터베이스에 저장하는 방법이 복잡하고 데이터베이스의 공간이 늘어나며 바이너리 파일에 비해 시간이 많이 걸린다. 따라서 텍스트 로그 파일보다 더 빠르게 저장 할 수 있고 데이터베이스에 바로 저장 할 수 있는 바이너리 로그 파일을 사용하였다.

6.3 패킷 분석 모듈

패킷 분석 모듈은 C 언어와 Mysql 3.22.32 데이터베이스를 사용해서 구현했다. 패킷 분석 모듈은 매 분마다 패킷 캡처 모듈에서 저장한 바이너리 로그 파일을 읽어 패킷을 해쉬해서 같은 패킷을 합친 뒤 데이터베이스에 저장한다.

대용량 트래픽을 분, 시, 일, 달 ,년 별로 분석하여 사용자 요구시 바로 웹기반 트래픽 분석 모듈이 정보를 참고 할 수 있도록 해야 한다. 대용량 트래픽을 빨리 저장하기 위하여 패킷 해쉬 후 데이터베이스에 저장 시 해쉬 로그 포맷을 30 개씩 저장하였다. 데이터베이스에서는 테이블의 상태를 항상 재점검 (table complete)하기 때문에 해쉬 로그 포맷을 하나씩 데이터베이스에 넣으면 재점검 시간이 늘어난다. 해쉬 로그 포맷을 30 개 정도씩 모아 입력하면 재점검 시간이 줄어 저장 시간이 반 정도로 줄어든다. 실험에 의하면 30 개를 모아서 입력할 때가 저장 시간이 가장 적었다. 30 개 이상은 다시 시간이 증가하였다.

패킷 분석 모듈은 때가 되면 매분, 매시, 매일, 매달, 매년의 네트워크 트래픽 정보를 자동으로 분석해서 데이터베이스에 저장해야 하므로 Unix의 cron명령어를 이용해서 매시, 매일, 매달, 매년마다 패킷 분석 모듈이 자동으로 동작하도록 했다.

6.4 웹기반 트래픽 분석 모듈

웹기반 트래픽 분석 모듈은 Apache Web Server 1.3.14 를 사용해서 웹기반으로 동작한다. 사용자의 요구에 따라 데이터베이스의 내용을 분석하여 사용자에게 전달하기 위해 C-CGI를 사용하여 웹과 데이터베이스를 연동하였다. 네트워크 트래픽 정보를 사용자에게 편리하게 제공하기 위해 C 언어와 gd [33]라이브러리를 사용하여 네트워크 트래픽의 총 양을 보여 주는 그래프를 그렸다.

또한 웹 브라우저에서 사용자에게 UDP/TCP 포트 번호와 포트 이름을 입력 받을 수 있도록 구현 했다. 이렇게 구현한 이유는 다양한 어플리케이션이 개발되는 것을 고려해서 어플리케이션 계층 분석의 확장성과 정확성을 높이기 위함이다. 사용자가 특정한 포트 번호와 어플리케이션 이름을 입력해 주면 분석 할 때 이것이 반영되어 잘 알려진 포트 번호 외에 최근 개발된 어플리케이션까지 분석 할 수 있다. Windows Media의 UDP/TCP 1775 번 포트 번호 등의 멀티미디어 서비스가 그 예이다. 웹기반 트래픽 분석 모듈이 데이터베이스의 정보를 가공해서 사용자에게 보여주는 방식이므로 입력한 UDP/TCP 포트 정보는 과거의 트래픽 정보에도 반영되어 사용자에게 보여주는 장점이 있다.

7. 시스템 적용

시스템 적용에서는 패킷 손실 없이 대용량 네트워크 트래픽을 다양하게 분석한 결과와 장기간 모니터링 및 분석, 멀티미디어 서비스 분석 등의 구현 예를 다룬다.

7.1 대용량 트래픽의 분 단위의 트래픽 총량 확인

포항공대 인터넷 회선에서의 1 분 단위의 네트워크 트래픽 분석 예를 보인다. 다음 그림 12 는 분 단위의 트래픽을 확인 할 수 있는 'Minute View' 이다.

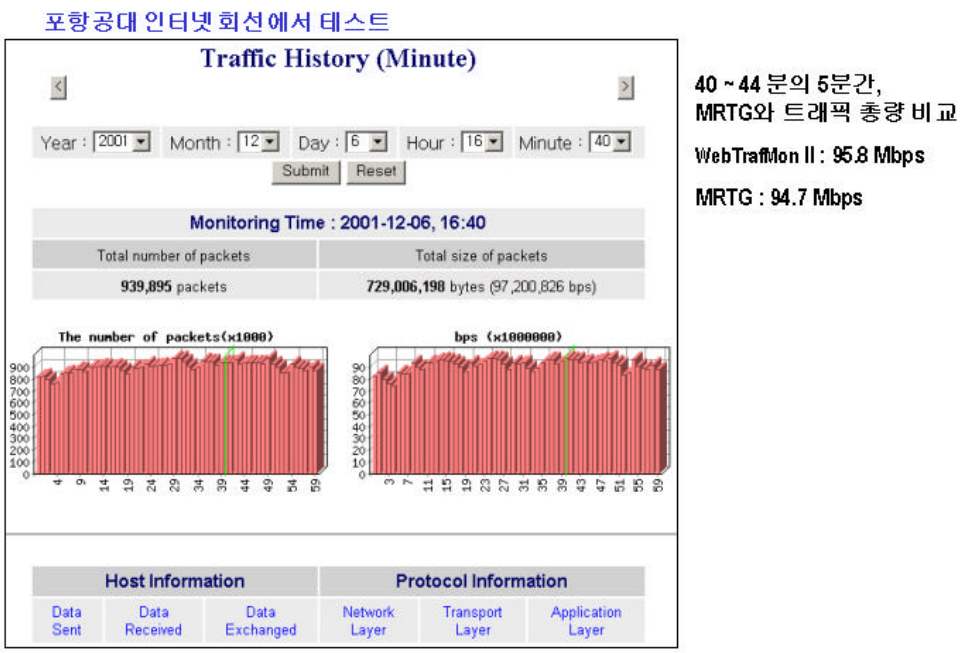


그림 12 Minute View

그림 12 의 제일 위의 '시간 선택 창'에서 2001 년 12 월 6 일 16 시 40 분을 선택 하였다. 분 단위의 모니터링을 제공하는 기능이므로 시간 선택 창을 통해서 년, 월, 일, 시, 분을 선택 할 수 있다. 또, 시간 선택 창 위에 있는 화살표 모양의 버튼을 통해서 현재의 시간을 기준으로 1 분 전과 1 분 후로 이동 할 수 있다.

시간 선택 창 밑에는 '2001-12-06, 16:40'과 같이 모니터링 시간을 표시해 주고 그 밑에 1 분 동안의 총 패킷 수와 총 네트워크 트래픽 양을 표시해 준다. 포항공대의 인터넷 회선에서 오후 4 시 40 분에서 41 분 사이의 1 분 동안 729,006,198 bytes의 약 97.2 Mbps의 트래픽이 발생했음을 알 수 있다.

트래픽이 많이 발생된 40 분에서 44 분 사이의 트래픽 총량의 평균을 구해서 MRTG와 비교해 보았다. WebTrafMon II에서는 95.8 Mbps가 계산되었고, MRTG에서는 94.7 Mbps가 계산되었다. MRTG와 비교하여 큰 차이가 없으므로, WebTrafMon II도 MRTG와 같이 패킷 손실없이 트래픽을 분석해 주고 있음을 알 수 있다.

모니터링 시간 밑의 두 그래프는 한 시간 동안의 분 단위의 총 패킷 수와 bps를 보여 준다. 이 두 그래프를 통해 2001 년 12 월 6 일 16 시의 한 시간 동안의 1 분 단위의 네트워크 트래픽을 총량을 패킷 수와 bps로 확인 할 수 있다.

그림 12 의 제일 밑의 'Host Information'과 'Protocol Information'을 통해 2001 년 12 월 6 일 16 시 40 분의 트래픽 정보를 자세하게 확인 할 수 있다. Host Information은 'Data Sent'와 'Data Received', 'Data Exchanged'로 나뉘고 'Protocol Information'은 'Network Layer', 'Transport Layer', 'Application Layer'로 나뉜다.

7.2 분 단위에서 호스트 정보 확인

그림 12 에서 호스트 정보 중에서 송신 호스트에 해당하는 정보를 확인하기 위해 Data Sent를 선택하면 그림 13 과 같이 2001 년 12 월 6 일 16 시 40 분에 트래픽을 가장 많이 발생시킨 송신 호스트 10 개를 기본으로 확인 할 수 있다.

Monitoring Time : 2001-12-06-16-40

Host Information : Data Sent (TOP 10)

Order by Byte	Source	Packets	Bytes
1	211.172.226.50 unknown	27,914 (2.97 %)	43,769,152 (6.00 %)
2	211.218.209.115 unknown	21,058 (2.24 %)	33,006,007 (4.53 %)
3	141.223.163.50 jupiter.postech.ac.kr	19,827 (2.11 %)	26,737,284 (3.67 %)
4	141.223.5.20 home.postech.ac.kr	17,508 (1.86 %)	25,332,536 (3.47 %)
5	255.255.255.127 unknown	31,606 (3.36 %)	23,465,864 (3.22 %)
6	141.223.180.1 Postech-Internet.postech.ac.kr	12,641 (1.34 %)	19,076,944 (2.62 %)
7	141.223.66.65 unknown	14,222 (1.51 %)	18,862,124 (2.59 %)
8	141.223.87.124 belief.postech.ac.kr	12,899 (1.37 %)	18,836,137 (2.58 %)
9	141.223.208.109 MDOM12402-1.postech.ac.kr	12,767 (1.36 %)	17,420,092 (2.39 %)
10	210.219.251.124 unknown	10,417 (1.11 %)	16,312,126 (2.24 %)

[Top 100] [Show All]

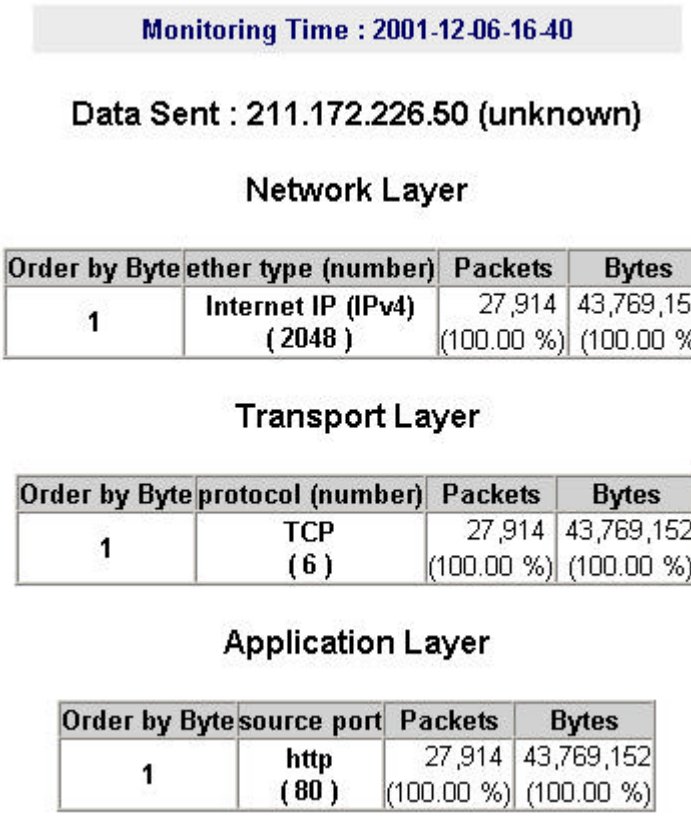
그림 13 2001 년 12 월 6 일 16 시 40 분의 Data Sent 정보

그림 13 에서 ‘Order by Byte’는 트래픽을 많이 발생시킨 순으로 정렬을 하는 것이고, ‘Source’는 송신 호스트라는 뜻이며, ‘Packets’는 개별 호스트가 발생한 패킷 총 개수와 전체 패킷 수에서 차지하는 비율을, ‘Bytes’는 개별 호스트가 발생한 트래픽 총량과 전체 트래픽 총량에서 차지하는 비율을 나타낸다.

그림 13 에서 211.172.226.50 의 이름을 알 수 없는 호스트가 2001 년 12 월 6 일 16 시 40 분의 1 분 동안 43,769,152 bytes로 가장 많은 트래픽을 발생시켰음을 알 수 있다. 또, 발생한 트래픽이 전체의 6 %를 차지한 것도 확인 할 수 있다.

두번째로 트래픽을 발생시킨 호스트는 4.53 %의 트래픽을 발생시킨 211.218.209.115 의 이름을 알 수 없는 호스트이며, 세번째와 네번째는 jupiter.postech.ac.kr과 home.postech.ac.kr 호스트이다.

가장 많은 트래픽을 발생시킨 211.172.226.50 호스트의 사용 정보를 확인하기 위해 211.172.226.50 호스트를 선택하면 다음 그림 14 와 같은 정보를 볼 수 있다.



Monitoring Time : 2001-12-06-16-40

Data Sent : 211.172.226.50 (unknown)

Network Layer

Order by Byte	ether type (number)	Packets	Bytes
1	Internet IP (IPv4) (2048)	27,914 (100.00 %)	43,769,152 (100.00 %)

Transport Layer

Order by Byte	protocol (number)	Packets	Bytes
1	TCP (6)	27,914 (100.00 %)	43,769,152 (100.00 %)

Application Layer

Order by Byte	source port	Packets	Bytes
1	http (80)	27,914 (100.00 %)	43,769,152 (100.00 %)

그림 14 211.172.226.50 호스트 정보 확인

그림 14 에서 211.172.226.50 호스트가 TCP/IP 프로토콜 스택 위에서 http 서비스를 43,769,152 bytes 만큼 사용했음을 알 수 있다. 이와 마찬가지로 jupiter.postech.ac.kr 등의 호스트를 선택하면 사용된 프로토콜 및 서비스를 확인 할 수 있다.

7.3 분 단위에서 전송 계층 정보 확인

그림 12 에서 'Protocol Information'의 'Transport Layer'를 선택하면

그림 15 와 같은 전송 계층에서 사용된 프로토콜 정보를 확인 할 수 있다.

Monitoring Time : 2001-12-06-16-40

Protocol Information : Transport (TOP 10)

Order by Byte	Protocol	Packets	Bytes
1	TCP (6)	887,455 (94.42 %)	694,026,835(95.20 %)
2	UDP (17)	49,040 (5.22 %)	34,655,028(4.75 %)
3	ICMP (1)	3,225 (0.34 %)	305,605(0.04 %)
4	ESP (50)	58 (0.01 %)	6,644(0.00 %)
5	EIGRP (88)	31 (0.00 %)	2,418(0.00 %)
6	IGP (9)	4 (0.00 %)	2,104(0.00 %)

그림 15 2001 년 12 월 6 일 16 시 40 분의 Transport Layer 정보

그림 15 에서 TCP, UDP, ICMP, ESP, EIGRP, IGP 순으로 전송 프로토콜이 사용되었고 TCP 프로토콜이 거의 대부분을 차지했다는 것도 알 수 있다. TCP를 선택하면 TCP 프로토콜을 사용한 호스트 정보 및 TCP 프로토콜을 이용한 서비스 정보를 확인 할 수 있다.

7.4 분 단위에서 서비스 정보 확인

그림 12 에서 'Application Layer'를 선택하면 다음 그림 16 을 볼 수 있다.

Monitoring Time : 2001-12-06-16-40

Protocol Information : Application (TOP 10)

Order by Byte	Source Port (number)	Destination Port (number)	Packets	Bytes
1	http (80)	netview-aix-4 (1664)	27,935 (2.97 %)	43,791,908 (6.01 %)
2	http (80)	cplscrambler-in (1087)	21,122 (2.25 %)	33,072,334 (4.54 %)
3	unknown (54765)	unknown (2055)	12,633 (1.34 %)	19,075,830 (2.62 %)
4	armtechdaemon (9292)	cecsvc (2571)	13,373 (1.42 %)	18,563,980 (2.55 %)
5	http (80)	tripwire (1169)	7,498 (0.80 %)	11,719,895 (1.61 %)
6	unknown (51914)	armtechdaemon (9292)	8,683 (0.92 %)	11,677,611 (1.60 %)
7	xs-openstorage (1619)	cyaserv (2584)	12,160 (1.29 %)	10,813,348 (1.48 %)
8	novation (1322)	unknown (4662)	6,478 (0.69 %)	9,404,917 (1.29 %)
9	http (80)	qnxnetman (3385)	5,973 (0.64 %)	9,346,211 (1.28 %)
10	armtechdaemon (9292)	isoipsigport-2 (1107)	6,556 (0.70 %)	8,852,746 (1.21 %)

[Top 100] [Top 1000] [Show All]

그림 16 2001 년 12 월 6 일 16 시 40 분의 Application Layer 정보

그림 16 에서 'Source Port'는 송신측에서 사용된 UDP/TCP 포트 정보 이며, 'Destination Port'는 수신측에서 사용된 UDP/TCP 포트 정보이다.

그림 16 에서 수신측의 http 서비스와 송신측의 netview-aix-4 서비스가 6.01 %로 1 분 동안 가장 많은 트래픽을 발생시켰음을 알 수 있다. 두 서비스를 사용한 호스트 정보를 확인 하려면 1 번을 선택한다. 다음 그림 17 은 그림 16 에서 1 번을 선택하였을 때의 정보이다.

Monitoring Time : 2001-12-06-16-40

Application : http(80) to netview-aix-4(1664)

Data Exchanged (TOP 10)

Order by Byte	Source	Destination	Packets	Bytes
1	211.172.226.50 unknown	255.255.255.127 unknown	27,914 (99.92 %)	43,769,152 (99.95 %)
2	211.32.117.39 www19.hanmail.net	255.255.255.127 unknown	18 (0.06 %)	22,562 (0.05 %)
3	211.169.240.71 unknown	141.223.93.59 wind104-2.postech.ac.kr	3 (0.01 %)	194 (0.00 %)

그림 17 http 와 netview-aix-4 서비스를 사용한 호스트

그림 17 에서 수신측 http 서비스는 211.172.226.50 호스트가 99% 이상을 발생시켰고 송신측의 netview-aix-4 서비스는 브로드캐스트 메시지가 99% 이상 발생시켰음을 알 수 있다.

그림 16 에서 수신측에서 사용된 http 서비스의 정보를 확인하려면 그림 16 의 1 번에서 'http(80)'을 선택한다. 다음 그림 18 은 그림 16 에서 http를 선택한 정보이다.

Monitoring Time : 2001-12-06-16-40

This is http(80) traffic of source port during one minute

Total http usage(packets) : 277,771 packets (29.55 %)

Total http usage(bytes) : 340,738,852 bytes (46.74 %)

Data Sent using http (TOP 10)

Order by Byte	Source	Packets	Bytes
1	211.172.226.50 unknown	27,914 (10.05 %)	43,769,152 (12.85 %)
2	211.218.209.115 unknown	21,058 (7.58 %)	33,006,007 (9.69 %)
3	141.223.5.20 home.postech.ac.kr	17,502 (6.30 %)	25,331,712 (7.43 %)
4	255.255.255.127 unknown	31,606 (11.38 %)	23,465,864 (6.89 %)
5	210.219.251.124 unknown	10,417 (3.75 %)	16,312,126 (4.79 %)
6	202.239.172.95 a202-239-172-95.deploy.akamaitechnologies.com	7,411 (2.67 %)	11,619,300 (3.41 %)
7	255.255.255.127 unknown	7,271 (2.62 %)	11,399,468 (3.35 %)
8	255.255.255.127 unknown	5,576 (2.01 %)	8,333,492 (2.45 %)
9	141.223.95.13 lunar.postech.ac.kr	5,906 (2.13 %)	7,298,601 (2.14 %)
10	209.114.91.43 sunwww4v3.ca1.breakaway.com	4,231 (1.52 %)	6,491,116 (1.91 %)

그림 18 1 분 동안의 수신측 http 서비스 확인

그림 18 에서 http 서비스는 340,738,852 bytes가 사용되어 전체 트래픽의 46.74 %를 차지했음을 알 수 있다. 또 http 서비스를 사용한 호스트 10 개를 기본으로 확인 할 수 있다. 211.172.226.50, 211.218.209.115 순으로 http 서비스를 사용했음을 알 수 있다.

7.5 시 단위의 트래픽 분석 예

시 단위의 트래픽 분석 예는 다음 그림 19와 같다.

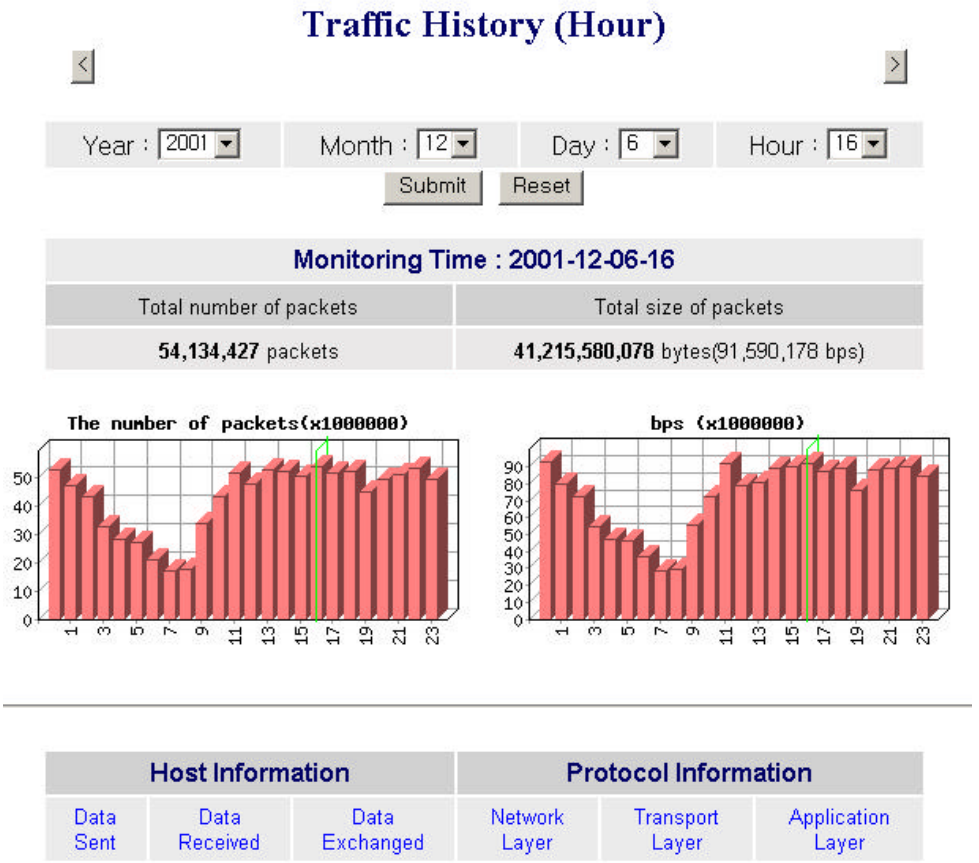


그림 19 Hour View

그림 19은 2001년 12월 6일의 16시에서 17시 사이의 한 시간 동안의 트래픽을 분석한 예이다. 포항공대 인터넷 회선의 16시에서 17시의 한 시간 동안의 평균 트래픽이 91 Mbps라는 것을 알 수 있으며 0

시, 11 시, 16, 22 시에 트래픽이 높다는 것을 알 수 있다. 또, 7 시와 8 시에 트래픽이 낮은 것을 확인 할 수 있다. 호스트와 프로토콜 정보를 확인하는 방법은 7.3 에서 소개한 것과 같다.

7.6 일 단위의 트래픽 분석 예

하루 단위의 트래픽 분석 예는 다음 그림 20 과 같다.

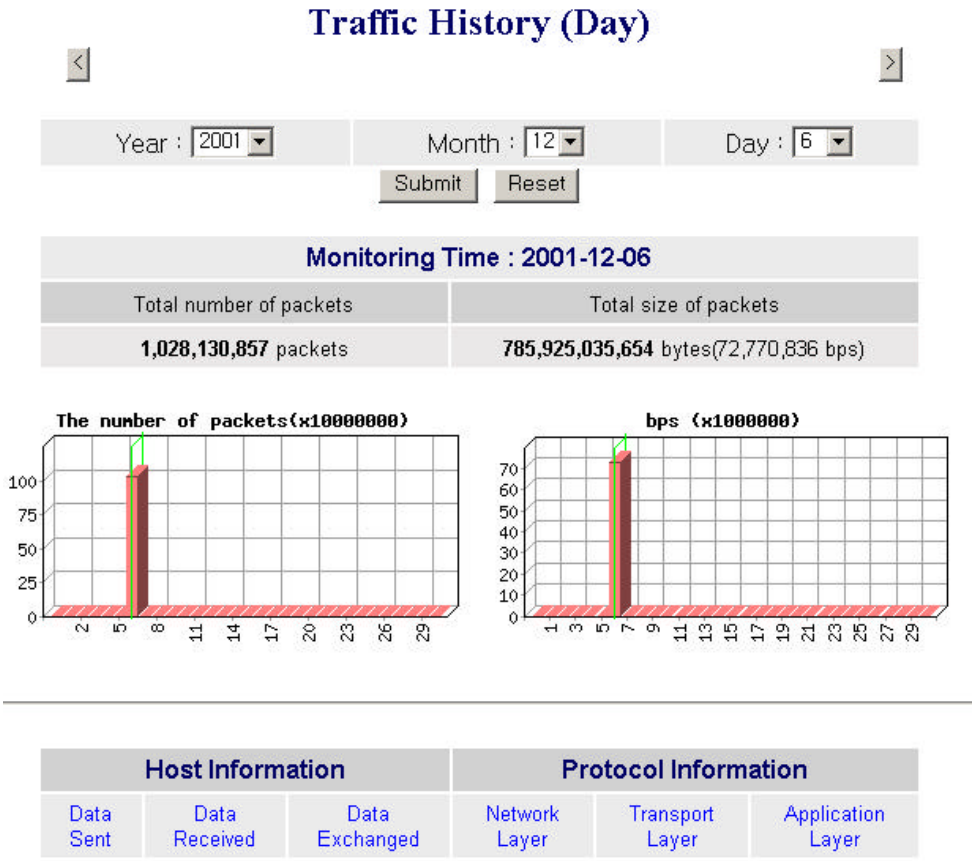


그림 20 Day View

그림 21 은 2001 년 12 월 6 일 하루 동안의 트래픽을 분석한 예이다. 6 일 하루 동안 평균 72 Mbps의 트래픽이 발생한 것을 알 수 있다. 호스트와 프로토콜 정보를 확인하는 방법은 7.3 에서 소개한 것과 같다.

7.7 월 단위의 트래픽 분석 예

월 단위의 트래픽 분석 예는 다음 그림 21 과 같다.

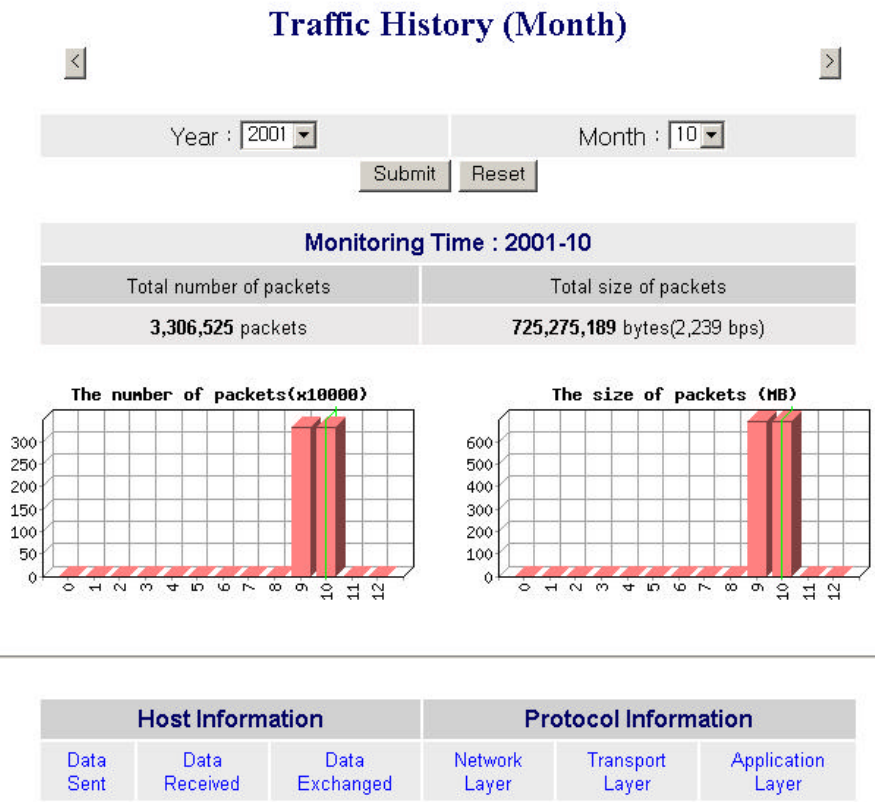


그림 21 Month View

그림 21 은 2001 년 10 월 한 달 동안의 트래픽을 분석한 예이다. 호스트와 프로토콜 정보를 확인하는 방법은 7.3 에서 소개한 것과 같다. 월과 년 단위의 분석은 포항공대 인터넷 회선의 트래픽 데이터가 아닌 실험 데이터를 사용한다.

7.8 년 단위의 트래픽 분석 예

년 단위의 트래픽 분석 예는 다음 그림 22 과 같다.

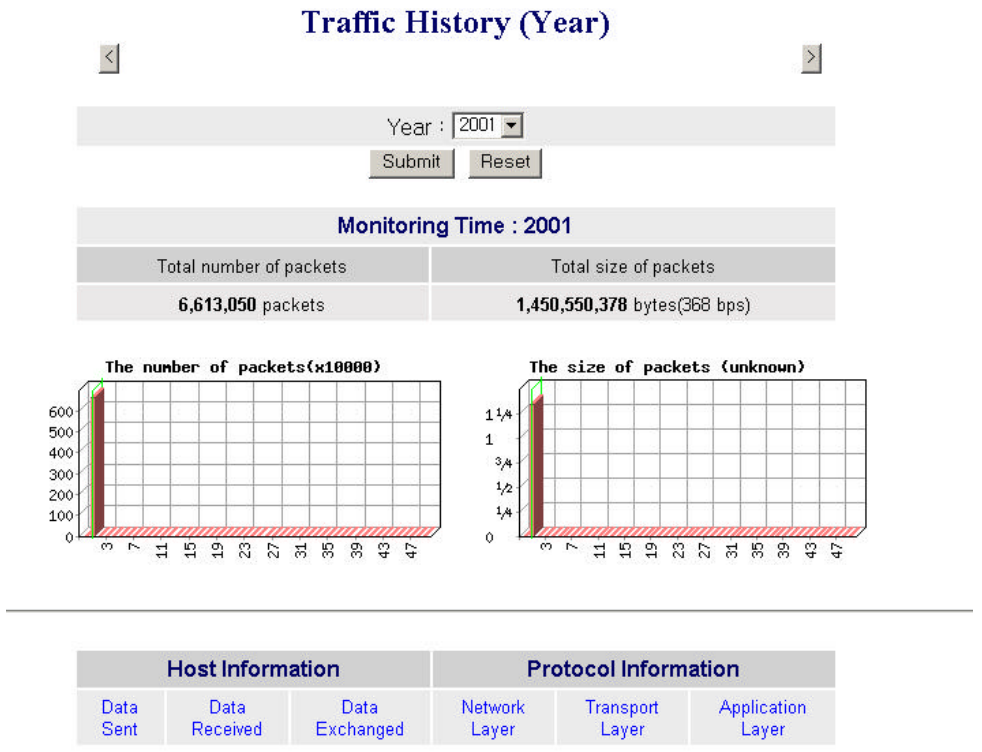


그림 22 Year View

그림 22 은 2001 년 1 년 동안의 트래픽을 분석한 예이다. 호스트와 프로토콜 정보를 확인하는 방법은 7.3 에서 소개한 것과 같다. 년 단위의 분석은 실제 데이터를 구할 수 없으므로 실험 데이터를 사용해서 구현 예만을 소개한다.

7.9 멀티미디어 서비스의 트래픽 분석

Widows Media 를 예로 든다. 그림 23 과 같이 UDP/TCP 정보를 입력한다.

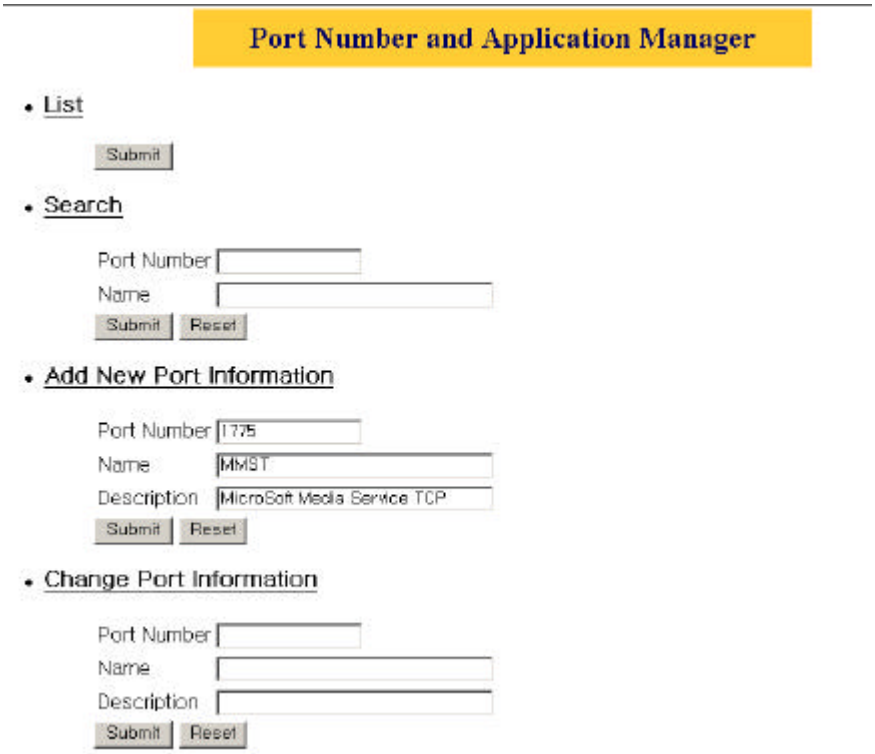


그림 23 MMST 의 UDP/TCP 포트 정보 입력

그림 23 과 같이 정보를 입력하면 과거의 트래픽 정보부터 앞으로의 정보까지 MMST 트래픽을 확인 할 수 있다. 포항공대 인터넷 회선에서 확인한 것은 그림 24 와 같다.

561	http (80)	qip-msgd (2468)	60 (0.01 %)	85,058 (0.02 %)
562	http (80)	nessus (1241)	153 (0.02 %)	84,753 (0.02 %)
563	http (80)	MMST (1775)	64 (0.01 %)	84,383 (0.02 %)
564	http (80)	unknown (1983)	105 (0.01 %)	84,327 (0.02 %)
	http	newheights	106	84,129

그림 24 MMST 트래픽 정보 확인

그림 24 에서 563 번째로 수신측 http와 송신측 MMST 서비스가 사용되었다는 알 수 있다. 또, 전체 트래픽의 0.02 %를 차지 했다는 것을 알 수 있다. 그림 24 에서 MMST 정보를 더욱 자세히 확인하고 싶으면 MMST를 선택한다. 선택한 정보는 그림 25 와 같다.

Total MMST usage(packets) : 64 packets (0.01 %)

Total MMST usage(bytes) : 84,383 bytes (0.02 %)

Data Sent using MMST (TOP 10)

Order by Byte	Destination	Packets	Bytes
1	255.255.255.127 unknown	46 (71.88 %)	65,475 (77.59 %)
2	141.223.175.68 sk-ttl18.postech.ac.kr	14 (21.88 %)	18,201 (21.57 %)
3	141.223.170.58 unknown	4 (6.25 %)	707 (0.84 %)

그림 25 MMST 서비스 확인

그림 25 **에서** MMST **프로토콜을 브로드캐스트와** 141.223.175.68 **호스트가 발생시켰음을 알 수 있다. 그림** 24 **와** 25 **를 통해서** Windows Media**를 사용하고 있는 호스트 정보를 확인 할 수 있다.**

이와 같은 방법으로 UDP/TCP **고정 포트 번호를 가지고 있는 모든 서비스를 분석 할 수 있다.** MMST **프로토콜 뿐만 아니라** P2P **등에서 사용되는 트래픽도 확인 할 수 있다. 즉,** UDP/TCP **포트를 이용해서 새로운 서비스 분석을 대비 할 수 있다.**

8. 결론 및 향후 연구

본 논문에서는 최근의 네트워크 트래픽 추세를 반영하기 위해 네트워크 트래픽 모니터링 및 분석을 위한 '로드 분산 네트워크 트래픽 모니터링 및 분석 구조'를 제시하였다. 로드 분산 네트워크 트래픽 모니터링 및 분석 구조를 사용하면 패킷 분석 모듈과 프리젠테이션 모듈 등으로 인한 시스템 로드가 패킷 캡처 모듈에 영향을 주지 않는다. 따라서 분석 모듈과 프리젠테이션 모듈의 시스템 과부하로 인한 패킷 손실을 근본적으로 막을 수 있고, 패킷 분석 모듈과 프리젠테이션 모듈도 각각 독립되어 있어, 분석 시간 및 응답 시간이 향상된다. 또, 로드 분산 네트워크 트래픽 모니터링 및 분석 구조에서 대용량, 장단기 동안의 네트워크 트래픽 모니터링 및 분석을 위한 일괄처리 트래픽 분석 방법을 연구하였다. 일괄처리 트래픽 분석 방법을 이용하여 패킷 캡처 모듈에서 트래픽 분석 없이, 패킷 헤더 정보만을 바이너리 로그파일에 바로 저장하여 패킷 캡처 시간을 줄였다. 이로 인해 패킷 캡처 모듈 자체의 시스템 과부하로 인한 패킷 손실을 막을 수 있었다. 또, 패킷 해쉬 알고리듬과 데이터베이스를 사용하여 대용량 트래픽 분석 및 장기간 모니터링을 가능하도록 했다. 멀티미디어 서비스 등을 UDP/TCP 포트 번호로 분석 할 수 있는 방법을 제시하고, UDP/TCP 포트 정보의 사용자 입력을 통해 새로운 서비스의 대처 방법도 제시하였다.

로드 분산 방법과 문제점 해결 방법을 적용하여 설계하고 구현한 WebTrafMon II는 포항공대 인터넷 회선에서 패킷 손실없이 장단기 동안의 네트워크 트래픽 모니터링 및 분석이 가능하였다. 40 ~ 100 Mbps의 대용량 트래픽을 다양하고 정확하게 분석 할 수 있는 기존의 네트워크 트래픽 분석 시스템을 찾기 어려운 것을 고려할 때 WebTrafMon II는 대용량 트래픽을 정확하고 다양하게 분석하는 데 큰 의의가 있다. 또 1분 단위부터 시, 일, 월, 년 단위의 장단기 동안의 네트워크 트래픽 모니터링 및 분석을 제공하는 것도 WebTrafMon II의 장점이다. 이외에도

멀티미디어 서비스 등의 새로운 서비스를 UDP/TCP 포트 번호를 이용하여 분석 할 수 있다는 것도 WebTrafMon II의 장점이다.

향후 연구로는 1 Gbps 이상의 초고속 통신망에서의 네트워크 트래픽 모니터링 및 분석에 관한 연구가 필요하다. 앞으로는 1 Gbps 이상의 초고속 통신망이 더욱더 보편화 될 것이기 때문이다.

또, 이더넷에서 구현한 WebTrafMon II를 IPoATM 등의 다양한 플랫폼에서 사용 할 수 있도록 하는 연구와 멀티미디어 트래픽을 정확하게 분석하기 위한 서비스 내용 분석에 관한 연구가 필요하다. 지금의 분석 방법으로는 단지 서비스의 트래픽 총량 만을 확인 할 수 있을 뿐, 사용된 서비스가 어떤 일을 하기 위해 트래픽을 발생시켰는지 알 수 없다. 서비스 내용 분석을 하려면 패킷의 데이터 필드를 분석해야 한다. 패킷의 데이터 필드를 분석을 하기 위해서는 우선 UDP/TCP 포트 번호를 알아야 한다. 따라서 본 논문의 멀티미디어 서비스의 UDP/TCP 포트 번호 조사는 서비스 내용 분석의 기초 연구라 할 수 있다. 특정 서비스의 UDP/TCP 포트 번호를 파악한 후 패킷의 데이터 필드 분석을 통해 서비스의 구체적인 동작과 기능을 파악 하는 연구가 필요하다.

마지막으로 SLA (Service Level Agreement)를 위한 네트워크 트래픽 모니터링 및 분석이 필요하다. 최는 다양한 SP (Service Provider)들이 등장하면서 SP와 사용자간에 서비스 질에 대한 계약을 맺고 있다. 따라서 계약을 맺은 서비스가 잘 이행되고 있는 지 등을 확인하기 위한 SLA가 중요한 이슈로 등장하고 있다. 사용자와 SP 간에 맺어진 계약을 확인할 수 있고 다양한 계약을 맺을 수 있도록 지원하는 네트워크 트래픽 모니터링 및 분석이 필요하다.

참고 문헌

[1] Peer-to-Peer Service, “http://www.peer-to-peerwg.org/”.

[2] Lawrence Berkley National Laboratory, “libpcap 0.6.2”, http://www.tcpdump.org.

[3] SNMP Research International, Inc., “http://www.snmp.org”.

[4] Tobias Oetiker and Dave Rand, “MRTG: Multi-Router Traffic Grapher”, http://www.mrtg.org.

[5] L. Deri and R. Carbone, “Monitoring Networks Using Ntop”, Released paper in http://luca.ntop.org, Jan 29th 2001.

[6] J. Won-Ki Hong, Soon-Sun Kwon and Jae-Young Kim, WebTrafMon: Web-based Internet/Intranet Network Traffic Monitoring and Analysis System, Computer Communications, Elsevier Science, Vol. 22, No. 14, September 1999, pp. 1333-1342. (SCIE).

[7] Ethereal, “http://www.ethereal.com”.

[8] Robert T. Braden and Annette L. DeSchon, “NNStat: Internet Statistics Collection Package”, USC/Information Sciences Institute Marina del Rey, California, November 28, 1988.

[9] Lawrence Berkley National Laboratory, “tcpdump 3.6”, http://www.tcpdump.org.

[10] Lawrence Berkeley National Laboratory, “tcpslice-1.1a3”, ftp://ftp.ee.lbl.gov/tcpslice.tar.Z.

[11] Snoop, “http://www.sun.com/products/sunray1/ts-sysmon.html”.

[12] Carter Bullard, “argus-1.7.beta.1b”, ftp://ftp.sei.cmu.edu/pub/argus.

[13] Lawrence Berkley National Laboratory, “arpwatch 2.0”, ftp://ftp.ee.lbl.gov/arpwatch.tar.Z.

[14] Dave Curry and Jeff Mogul, “nfswatch-4.3”, ftp://ftp.lip6.fr/pub2/networking/nfs.

[15] David K. Hess and Douglas Lee Schales, David R. Safford, “drawbridge 2.0”, http://www.certcc.or.kr/tools/index.html.

[16] ewatch, “http://ewatch.hangkong.ac.kr”.

[17] sniffer pro, “http://www.softseek.com”.

[18] Mgen, "Multi-Generator", The Naval Research Laboratory.

[19] Audio-Video Transport Working Group, “RTP: A Transport Protocol for Real-Time Applications”, IETF RFC 1889, January 1996.

[20] Real Media, “http://www.real.com”.

[21] Windump, “http://netgroup-serv.polito.it/windump”.

[22] Netmon, “http://w1.132.telia.com/~u13200034/netmon.html”.

[23] Windows Media Player, “http://www.Microsoft.com”.

[24] MMST, “http://www.microsoft.com/windows2000/en/server/help”.

[25] FastTrack, “http://www.fasttrack.nu”.

[26] Gnutella, “http://www.gnutellaworld.net/gw/stories.php”.

[27] FreeNet, “http://freenet.sourceforge.net/”.

[28] IANA, “Ethernet Numbers”, “http://www.iana.org/assignments/ethernet-numbers”.

[29] NFS, “http://www.sun.com/software/white-papers/wp-nfs.sw”.

[30] crontab, “http://tenet2.knu.ac.kr/~gjback/xcu/crontab.html”.

[31] Mysql 3.22.32, “http://www.mysql.com”.

[32] Apache Web Server, “http://www.apache.org”.

[33] BOUTELL.COM, “gd 1.8.2”, http://www.boutell.com/gd.

[34] R. Enger and J. Reynolds, “FYL on a Network Management Tool Catalog”, IETF RFC 1470, June 1993.

[35] Raw IP Networking FAQ, “http://www.whitefang.com/rin/rawfaq.txt”.

[36] IANA, “Protocol Numbers”, “http://www.iana.org/assignments/protocol-numbers”.

[37] IANA, “Port Numbers”, “http://www.iana.org/assignments/port-numbers”.

[38] J. Raynolds, J. Postel, “Assigned Numbers”, RFC 1700, Network WG, October 1994.

[39] W. Richard Stevens, *TCP/IP Illustrated, Volume 1*, Addison-Wesley, 1994.

[40] W. Richard Stevens, *Unix Network Programming Volume 1*, Prentice-Hall, 1998.

감사의 글

포항공대 컴퓨터 공학과 DPNM 연구실에서 생활하며 참으로 많은 것을 배운 것 같습니다. 20 대 중반을 이곳에서 보내게 된 것이 지금 돌아 보면 참 신기한 일이였고, 이 곳 생활이 앞으로의 저의 삶에 어떤 영향을 미칠 지 하나님의 뜻이 참으로 궁금합니다.

포항에서도 나의 잘못됨을 늘 꾸짖어 주시고 바쁘고 힘들어도 성경 읽는 것만큼은 손 놓지 않게 인도해 주신 하나님께 진심으로 감사드립니다. 2 년 동안 死活(사활)의 代贖(대속)의 은혜를 주신 예수 믿는 가치를 더욱 깨달았습니다.

지도 교수님이신 홍원기 교수님께 감사 드립니다. 지난 2 년간 교수님께 죄송한 일이 많았음을 고백하지 않을 수 없습니다. 부족한 저를 받아주시고 잘 지도해 주셔서 진심으로 감사드립니다. 앞으로는 사회에 나가 더욱 좋은 사람이 되겠습니다. 논문을 심사해 주신 박찬익 교수님과 서영주 교수님께도 감사의 말씀 올립니다.

학비를 보태 주시고 늘 걱정해 주신 부모님께 감사 드립니다. 누나와 형에게도 감사 드립니다. 우리 가족이 있었기에 제가 이 자리에 와서 이렇게 석사 학위를 받을 수 있었습니다.

포항대광교회의 교역자분들과 권사님, 집사님, 그리고 청년들에게 감사합니다. 포항대광교회에 늘 하나님의 축복 있기를 기도합니다.

학부 선배인 정원형님, 신이형, 진민형, 민선누나, 남이누나, 자영누나에게도 감사하며 멀리서 응원해 주신 중재형 등의 숭실대 오즈 가족들에게도 감사드립니다.

우리 연구실을 이끄셨던 홍택형님과 재영형님의 박사 졸업을 다시 한번 축하드립니다. 두분 다 앞으로 DPNM연구실을 빛내 주실 것이라 믿습니다. 논문 주제로 세미나 할 때 많은 조언을 주신 홍택형님,

그리고 저의 부족한 논문을 여러 번 손봐주신 재영형님, 감사합니다.

늘 연구실 사람들을 챙기는 미정누나에게도 고맙습니다. 열심히 하셔서 원하는 바를 이루시고 DPNM의 훌륭한 박사가 되길 바랍니다. 그리고 개인적으로 고마운 것이 너무 많은 명섭형님, 앞으로 우리 연구실의 미래를 미정 누나와 함께 잘 이끌어 주세요. 명섭형이 패킷 해쉬에 대한 이야기를 해주지 않았더라면 WebTrafMon II의 프로젝트는 실패했을 지도 모릅니다. 늘 긍정적인 마음으로 프로젝트를 진행하는 명섭형의 모습은 꼭 본 받겠습니다. 명섭형의 박사 취득을 축하해 드리는 날을 기다리지요.

같은 2 년차 동기인 허인형님, 정혁형님, 범래형님께도 참으로 감사드립니다. 여러모로 부족한 저를 이해해 주시고 많은 도움 주신 형님들께 진심으로 감사드립니다. 허인형님의 포철에서의 건승을 빌며, 곧 애 아빠가 되는 범래형께도 미리 축하의 말씀 드립니다. 정혁형의 재승이와 예진이가 반듯하고 아름답게 자라길 바라며 정혁형께 여러모로 너무나 고맙다는 말씀 드리고 싶습니다. 형님과의 우정을 앞으로 잘 간직하길 원하며 앞으로 교회 꾸준히 다니세요.

늘 열심히 학생과 엄마의 역할을 하여 연구실의 귀감이 되는 윤정 누나, 누나의 밝은 모습 늘 기억하겠습니다. 우리랩의 희망인 1 년차 훈정형, 세희, 효진이에게 정말로 고맙다는 말을 전합니다. 1 년차 분들을 통해 동료애가 무엇인지 배우고 저의 부족함을 많이 깨닫게 되었습니다. 유학을 목표로 하는 훈정형의 건승을 빌며, 나의 친구인 세희가 하고자 하는 것 잘 이루길 바랍니다. 이쁜 효진이도 열심히 공부해서 훌륭한 '이석사'가 되길 바랍니다. 내년에 후배님들께 축하의 소식 보내겠습니다. 교수님 비서로 연구실 일을 꼼꼼히 챙겨주시는 선희씨에게도 고맙다는 말 전합니다. 선희씨 덕분에 많은 부분이 편했던 것 같습니다.

마지막으로, 이 논문을 작성하는데 편집에서 많은 도움을 주신 정혁형님의 노고에 깊이 감사드립니다. MS Word의 사용법을 잘 몰라 정혁

형이 없었다면 졸업하기 힘들었을 것입니다. 정혁형도 바쁜데 손수 논문과 발표자료 편집에 도움을 주셨습니다.

이 력 서

성 명 : **홍 순 화**
생 년 월 일 : 1976 **년** 5 **월** 17 **일**
출 생 지 : **대전 광역시**
주 소 : **경북 포항시 남구 효자동 산** 31 **번지 포항공대 기숙사** 19 **동** 403 **호**

학 력

1996 – 2000 : **숭실대학교 컴퓨터학부 졸업** (**학사**)
2000 – 2002 : **포항공과대학교 전자컴퓨터공학부**(**컴퓨터공학**) **졸업** (**석사**)

학 술 활 동

? **Conference Papers**

- S. H. Hong, J. Y. Kim, B. R. Cho, and J. W. Hong, "Distributed Network Traffic Monitoring and Analysis using Load Balancing Technology", Proc. of the Asian-Pacific Network Operation and Management Symposium, Sydney, Australia, September 2001, pp. 172-183.
- **홍순화**, **김재영**, **조범래**, **홍원기**, "**분산 시스템 환경에서의 로드 밸런싱을 통한 웹기반 네트워크 트래픽 모니터링 및 분석**", Proc. of KNOM 2001 Conference, Daejeon, Korea, May 24-25, 2001, pp. 198-205.

? **Projects**

- "Linux-based Real-Time Operating System", POSCO project, 2000/1 ~ 2000/12."
- "WebTrafMon II : **인터넷 서비스 트래픽 모니터링 및 분석**", **포항공대 정보통신연구소 프로젝트**, 2001/1 ~ 2001/12.