멀티 코어 환경에서 실시간 트래픽 분석 시스템 처리속도 향상

준회원 윤 성 호*, 박 준 상*, 종신회원 김 명 섭**°

Performance Improvement of a Real-time Traffic Identification System on a Multi-core CPU Environment

Sung-ho Yoon*, Jun-sang Park* *Associate Members*, Myung-sup Kim**° *Lifelong Member*

요 약

오늘날 네트워크 환경은 응용 프로그램 및 서비스의 변화가 많아 응용탐지에 있어 기존의 단일 분석 알고리즘으로는 모든 트래픽의 응용을 정확하게 탐지하기 어렵다. 최근 이러한 단점을 보완하기 위해 여러 개별 알고리즘을 통합한 멀티 레벨의 트래픽 탐지 알고리즘에 대한 연구가 진행되고 있다. 이러한 멀티 레벨 탐지 알고리즘은 단일 알고리즘 구조에 비해 계산 복잡도가 높은 단점이 있다. 또한, 고속 네트워크에서 실시간으로 트래픽을 분류하기 위해서는 멀티코어 CPU의 장점인 병렬처리를 이용하여 높은 계산 복잡도를 해결해야 할 필요가 있다. 본 논문에서는 요즘 일반화된 멀티 코어 CPU환경에 적합한 실시간 응용 트래픽 탐지 시스템 구조를 제안한다. 먼저 멀티 레벨 트래픽 탐지 알고리즘이 멀티 코어 환경에서 실시간으로 동작하기 위한 고려 사항들을 살펴보고, 이를 통해 시스템을 설계하고 구현한 내용을 기술한다. 본 논문에서 구축한 시스템은 캠퍼스 네트워크와 기숙사 네트워크를 대상으로 구축하여 그 실효성을 검증하였다.

Key Words : traffic analysis, traffic classification, passive monitoring, multi-core, 트래픽 분석, 트래픽 분류, 패시브 모니터링, 멀티 코어 CPU, 병렬처리

ABSTRACT

The application traffic analysis is getting more and more challenging due to the huge amount of traffic from high-speed network link and variety of applications running on wired and wireless Internet devices. Multi-level combination of various analysis methods is desired to achieve high completeness and accuracy of analysis results for a real-time analysis system, while requires much of processing burden on the contrary. This paper proposes a novel architecture for a real-time traffic analysis system which improves the processing performance on multi-core CPU environment. The main contribution of the proposed architecture is an efficient parallel processing mechanism with multiple threads of various analysis methods. The feasibility of the proposed architecture was proved by implementing and deploying it on our campus network.

※ 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단-차세대정보컴퓨팅기술개발사업의 지원을 받아 수행된 연구임 (20100020728).

* 고려대학교 컴퓨터정보학과 박사과정({sungho_yoon, junsang_park}@korea.ac.kr)

** 고려대학교 컴퓨터정보학과 부교수(tmskim@korea.ac.kr). (° : 교신저자)

논문번호 : KICS2011-10-491, 접수일자 : 2011년 10월 21일, 최종논문접수일자 : 2012년 4월 16일

Ⅰ. 서 론

효율적인 네트워크 관리를 위해서 네트워크 관리자들은 CRM(Customer Relationship Management), SLA(Service Level Agreement), QoS(Quality of Service), 트래픽 과금과 같은 다양한 정책을 적용하려고 한다. 네트워크 트래픽의 응용을 탐지하는 트래픽 분류는 이러한 정책들을 적용하기 위해 반드시 필요한 기술이다. 트래픽 분류 방법론 또는 시스템의 최종목표는 분석하고자 하는 대상 네트워크에서 발생하는 트래픽을 모두 정확하게 빠른 속도로 분류하는 것이다.

트래픽 분류 방법론은 그 중요성에 따라 지속적으로 연구가 진행되어 현재에는 다양한 트래픽 분류 방법론이 존재한다. 하지만, 오늘날의 네트워크에는 다양한 응용 프로그램 및 서비스로 인해 복잡 다양한 트래픽이 발생하므로 하나의 단일 트래픽 분류 방법론으로는 네트워크 내의 모든 트래픽을 분류하기 어렵다. 따라서, 최근에는 여러 가지 트래픽 분류 방법론을 통합한 멀티 레벨 트래픽 분류 방법론[1-5]이 제안되고 있다.

멀티 레벨 트래픽 분류 방법론은 기존의 여러 트래픽 분류 방법론들 중 몇 가지를 통합하여 만들어진 방법론이다. 기존의 제안된 포트 기반 분류 방법[6,7], 페이로드 시그니쳐 분류 방법[8,9,10], 머신러닝 분류 방법[11,12] 등은 분류 방법에 따라 정확하게 분류할 수 있는 트래픽의 종류가 다르다. 멀티 레벨 트래픽 분류 방법론은 여러 분류 방법론의 장점을 통합하기 때문에 높은 분석률과 정확도를 가진다. 하지만, 여러 방법론의 통합으로 인해 단일 방법론에 비해 계산 복잡도가 높은 단점을 가지고 있다.

트래픽 분류 결과는 주로 트래픽 제어에 활용되는데, 올바른 제어를 위해서는 정확한 분류뿐만 아니라 실시간 분류가 가능해야 한다. 실시간 분류는 트래픽 분류 속도에 영향을 받으며, 멀티 레벨 트래픽 분류 시스템에서는 빠른 분류 속도를 위해 높은 계산 복잡도를 해결해야 한다. 높은 계산 복잡도를 해결하기 위해서는 알고리즘의 개선도 필요하지만 트래픽 분류 시스템이 동작하는 하드웨어의 성능이 매우 중요하다. 하드웨어 중 연산에 가장 큰 영향을 미치는 하드웨어는 중앙처리장치(CPU) 이다. 오늘날의 CPU는 단일 코어의 성능 개선의 한계점으로 인해 하나의 CPU 안에 여러 개의 코어가 동작하는 멀티 코어 CPU가 개발되고 있다.

본 논문에서는 멀티코어 CPU 환경에서 여러 개의 트래픽 분류 알고리즘이 조합되어 있는 멀티 레벨 트래픽 분류 시스템의 처리속도 향상에 관한 방법론을 제안한다. 본 논문에서 구축한 멀티레벨 트래픽 분석 시스템은 헤더 시그니쳐[13], 통계 시그니쳐[14], 페이로드 시그니쳐[9], 행동양식 기반 알고리즘[15]의 트래픽 분류 알고리즘의 조합으로 구성된다. 먼저 트래픽 분류 시스템이 멀티 코어 환경에서 실시간으로 동작하기 위한 고려 사항들을 살펴보고, 이를 통해 시스템을 설계하고 구현한 내용을 기술한다. 본 논문에서 구축한 시스템은 캠퍼스 네트워크와 기숙사 네트워크를 대상으로 구축하여 그 실효성을 검증하였다.

본 논문은 다음과 같은 순서로 기술한다. 2장에서는 기존 트래픽 분류 방법론들에 대해 살펴보고, 3장에서는 멀티 코어 환경에서 실시간 트래픽 분류 시스템이 고려해야 할 사항들을 살펴본다. 4장에서 시스템을 설계한 내용을, 5장에서는 설계한 내용을 바탕으로 시스템을 구현한 내용과 성능평가 결과를 기술하고, 마지막으로 6장에서는 결론과 향후 연구를 언급한다.

Ⅱ. 관련 연구

트래픽이 변화함에 따라 트래픽 분류 방법 또한 지속적으로 변화해 왔다. 인터넷 초창기에는 잘 알려진 포트를 사용하는 HTTP, telnet, e-mail, FTP, SMTP의 응용들이 대부분의 트래픽을 차지하였다. 따라서, IANA[6]에 정의된 포트 정보 기반의 분류 방법을 통해 신뢰성과 정확성이 높은 트래픽 분류 결과를 도출할 수 있었다. 하지만, 현재 인터넷 트래픽의 상당수를 차지하고 있는 P2P 프로그램을 비롯한 많은 응용 프로그램들은 동적 포트 할당과 같은 기술들로 포트 기반 분류 방법론은 더 이상 정확하지 않다[10].

페이로드 시그니쳐 기반 트래픽 분류 방법론[8-10]은 패킷의 페이로드 내에서 응용마다 가지는 특정한 스트링의 포함 유무를 통해 트래픽을 분류하는 방법으로써 높은 분석률과 분류 정확도를 나타낸다. 하지만, 암호화된 트래픽에 대해서는 분석을 하지 못하며, 높은 계산 복잡도, 사생활 침해 문제 등과 같은 문제점을 가지고 있다.

트래픽 암호화 및 사생활 침해 문제를 해결하기 위해 패킷 및 윈도우 크기, 패킷 간 시간 간격 등과 같은 트래픽의 통계적 특징을 이용한 분류 방법론[14,16]이 많이 개발되고 있다. 이 방법론은 패킷의 헤

더 정보를 통해 통계 정보를 생성하므로 기존 트래픽 분류 방법론들의 단점들을 보완할 수 있다. 하지만, 같은 엔진 기반의 응용이거나 같은 응용 레벨 프로토콜을 사용하는 경우 동일한 통계적 특징을 가지기 때문에 상세한 응용 별 분석이 어렵다.

이와 같이 전통적인 시그니쳐 기반의 분석 방법의 한계점을 보완하기 위해 특정 응용을 제공하는 응용 서버의 헤더 정보를 이용하는 헤더 기반 방법론[13], 응용 트래픽의 발생 형태를 사용하여 트래픽을 분석하는 행동 기반 방법론[15], 트래픽 상이의 연관성을 사용한 상관 관계 기반 방법론[17,18]등 다양한 분석 방법론이 제안되고 있다.

표 1은 다양한 분석 방법과 각 분석 방법의 특징을 정리한 것이다. 최근에는 각 분석 방법의 한계점을 보완하고 분석 성능을 향상시키기 위해 여러 분석 방법을 결합한 형태의 멀티 레벨 분석 방법론이 제안되고 있다[1-5]. 하지만 대부분의 연구들은 다양한 분석 알고리즘의 통합구조에 주로 집중하고 있을 뿐 멀티코어 CPU 환경에서의 병렬처리 및 분석속도 향상에 대한 고려는 아직 미흡하다.

표 1. 다양한 트래픽 분류 방법론 및 그 특징
Table 1. Features of various traffic classification methods

IIdentification Method	Accu-racy	Complete-ness	Vulnerable to payload encryption	Cost of Operation	Input format
Header -based	High	Low	No	Low	Flow
Statistic -based	Medium	Medium	No	Medium	Packet
Payload -based	Medium	High	Yes	High	Packet
Behavior -based	High	Low	No	Medium	Packet
Correlation -based	Depen dent	Dependent	No	Low	Flow

그림 1은 본 연구진이 개발한 멀티 레벨 분석기의 구조[5]이다. 그림 1에 제시된 멀티 레벨 분석기는 총 4 레벨로 구성된다. 1 레벨에서는 트래픽을 수집하여 분석기의 입력 데이터를 생성하고, 2 레벨에서는 각 세부 분석기가 각자의 분석 방법으로 트래픽을 분석한다. 3 레벨에서는 각 세부 분석기가 분석한 결과를 통합하고, 마지막 4 레벨에서는 상관관계 기반의 분석을 통해 추가적인 분석을 한다. 제시된 분석 구조는 멀티 레벨 분석기의 여러 구현 방법 중 하나를 나타내며, 다양한 방법으로 구현될 수 있다.

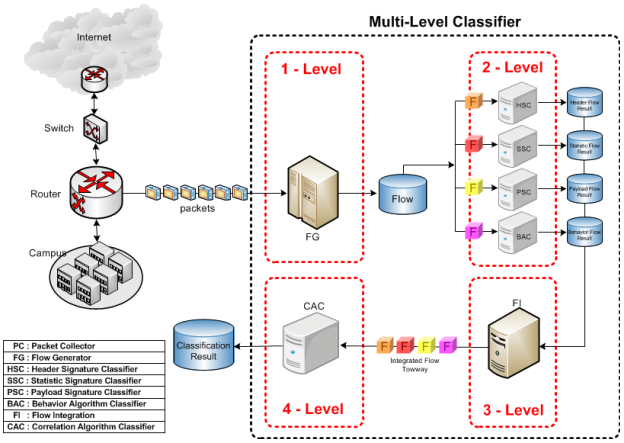

그림 1. 멀티 레벨 분석기의 구조
Fig. 1. Structure of the multi-level classification system

본 논문에서는 본 연구팀이 기 개발한 멀티 레벨 기반 분석 방법론을 기반으로 멀티 코어 환경에서 트래픽 처리속도 향상을 위한 효과적인 트래픽 분석 방법에 대해 제안한다.

Ⅲ. 시스템 구축 고려사항

본 시스템은 멀티 코어 CPU 환경에서의 멀티 레벨 기반 실시간 트래픽 분석을 목적으로 한다. 이를 위해 다음과 같은 사항들을 고려해야 한다.

3.1 트래픽 처리 단위

본 논문에서 제안하는 분석기는 다양한 분석 방법론을 통합한 멀티 레벨 분석 방법을 기반으로 한다. 표 1과 같이 각 세부 분석기는 분석 방법에 따라 서로 다른 형태의 트래픽(flow/packet)을 입력 데이터로 사용한다. 따라서 멀티 레벨 기반 분석기의 입력 데이터는 모든 세부 분석기가 사용 할 수 있는 형태로 재구성하여야 한다. 본 시스템에서는 패킷과 플로우를 조합한 flow_with_packet 단위를 정의하여 사용한다. 트래픽 처리 단위에 대한 자세한 내용은 4.2 절에서 설명한다.

3.2 비정상 트래픽 처리

최근 인터넷이 대중화됨에 따라 악의적인 목적을 가진 네트워크 공격 또한 활발히 발생하고 있다. 네트워크 공격은 공격 대상 호스트에 대량의 비정상적인 트래픽을 보냄으로써 정상적인 인터넷 사용을 방해한다. 이렇게 발생된 트래픽은 분석 시스템의 입력 데이터의 양을 증가시켜 분석기의 오버헤드를 증가시킨다. 따라서 실시간 트래픽 분석을 위해 비정상적인 트래픽을 사전에 판별하여 분석에 반영해야 한다.

비정상 트래픽 처리에 대한 자세한 내용은 4.3절에서 설명한다.

3.3 멀티 코어 환경에 적합한 시스템 구조

멀티 코어 환경에서는 단일 코어 환경과 달리 여러 가지 사항을 고려하여야 한다. 첫째, 단위 프로세스 내부의 여러 쓰레드들이 동일한 메모리를 참조하여 트래픽을 분석하기 때문에 각 분석 쓰레드가 사용하는 메모리의 범위를 정확히 할당해야 한다. 둘째, 정확한 분석을 위하여 분석 쓰레드간 사용하는 메모리의 동기화 작업을 해야 한다.

본 시스템에서는 이를 위해 각 분석 쓰레드들을 관리하는 Buffer Manager를 두었고, 쓰레드들 간의 동기화를 위해 세마포어를 사용하였다. 자세한 내용은 4.4절, 4.5절에서 설명한다.

Ⅳ. 실시간 트래픽 분류 시스템 설계

4.1 시스템 전체 구조

그림 2는 멀티코어 환경에 적합한 트래픽 분류 시스템의 전체적인 구조를 나타낸 것이다. 트래픽 분류 시스템은 설치하고자 하는 대상 네트워크에 모든 패킷을 캡쳐할 수 있는 지점에 설치되어야 한다. 특정 네트워크에서 외부 네트워크로 나가는 경로가 여러 개인 경우에는 모든 경로에서 패킷을 수집하는 것이 필요하다.

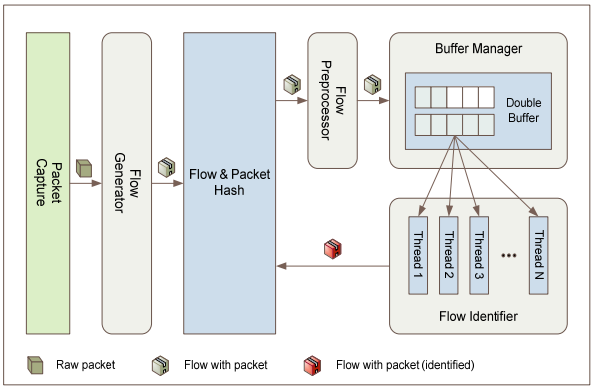

그림 2. 트래픽 분류 시스템의 전체 구조
Fig. 2. Overall architecture of the classification system

트래픽 분류 시스템은 Flow Generator, Flow Preprocessor, Buffer Manager, Flow Identifier의 4개의 모듈로 구성된다. Flow Generator는 패킷 캡쳐 라이브러리를 통해 패킷을 실시간으로 수집하고 수집된 패킷을 통해 플로우를 생성한다. 이 때, 플로우는 양방향 플로우로써 5-tuple(source IP, source port, destination IP, destination port, protocol)을 기준으로 패킷 수, 바이트 수 등의 정보뿐만 아니라 트래픽 분류 알고리즘에 필요한 패킷의 페이로드 정보까지 포함될 수 있다. 본 시스템에서는 이를 flow_with_packet 이라고 지칭하며, 해쉬 형태의 메모리 구조로 저장된다. 저장된 flow_with_packet은 Flow Preprocessor에 의해서 트래픽 분류 관점에서의 정상과 비정상 플로우로 먼저 분류되고, 응용 별 분류를 위해서 Buffer Manager에 의해 쓰기 버퍼에 분류 대상 플로우가 저장된다. Buffer Manager는 버퍼의 분석 조건이 만족되면 쓰기 버퍼를 읽기 버퍼로 전환하고, Flow Identifier에 해당하는 다수의 쓰레드 모듈을 동작시켜 읽기 버퍼에 저장된 플로우들을 멀티레벨 트래픽 분류 알고리즘을 통해 분류한다. 읽기버퍼에 저장된 플로우를 분석하는 동안 Flow Generator, Flow Preprocessor, Buffer Manager는 지속적으로 쓰기 버퍼에 다음에 분석할 플로우를 저장한다. 트래픽 분류 시스템은 위의 과정이 지속적으로 반복하는 구조이다.

4.2 Flow Generator

Flow Generator는 네트워크 링크로부터 패킷을 캡쳐하고 플로우를 생성 및 저장하는 것이 주된 역할이다. 본 시스템에서 사용하는 플로우는 양방향 형태의 플로우이며, 그림 3과 같은 데이터 구조를 가지고 있다. 하나의 플로우는 5-tuple로 구성된 주소 정보(Address Info.)를 가지고 있으며, 두 단말간의 통신에서 단방향의 트래픽에 대한 통계정보(Statistical Info.)를 가진다. 통계정보에는 플로우의 시작 시간, 종료 시간, 패킷의 개수, 바이트 크기, TCP 플래그 패킷의 개수, 데이터 패킷의 개수와 기타 정보가 포함된 플래그 정보가 포함된다.

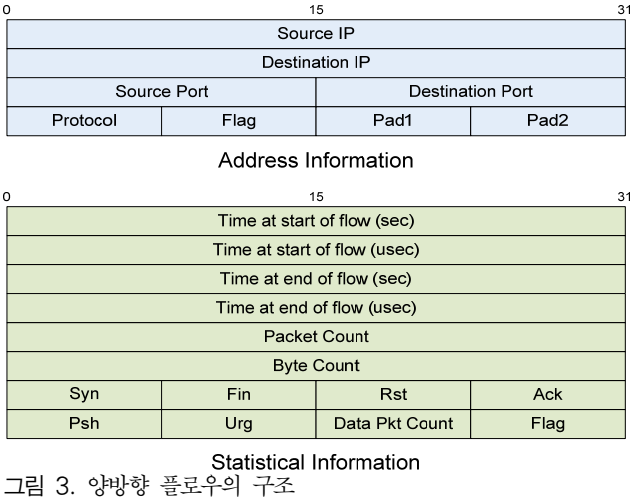

그림 3. 양방향 플로우의 구조
Fig. 3. The structure of the bidirectional traffic flow

플로우에는 초기에 발생하는 데이터 패킷에 대한 헤더와 페이로드 정보가 포함되어 있다. 이러한 정보는 트래픽 분류 알고리즘에 사용되기 위해 저장되며, 본 시스템에서는 플로우의 양방향으로 각각 최대 5개의 데이터 패킷을 저장한다. 이러한 데이터 형태를 flow_with_packet이라 지칭한다. 개수의 제한은 메모리 공간의 한계와 분류 알고리즘에서 필요한 패킷의 개수를 반영한 것이다.

네트워크 링크에서 수많은 패킷이 발생하는 상황에서 실시간으로 플로우를 만들고 분석하기 위해서는 플로우를 효율적으로 저장할 수 있는 메모리 구조가 필요하다. 본 시스템에서는 이를 위해 해쉬 구조를 사용하여 이를 통해 플로우 생성시간을 최소화한다. 해싱 함수는 shift folding function을 사용하고, 해쉬 테이블 구조는 같은 해쉬 키를 가진 플로우들을 링크드 리스트로 연결하는 오픈 해쉬 테이블을 사용한다.

4.3 Flow Preprocessor

네트워크에는 정상적인 응용에서 발생한 트래픽 뿐만 아니라 악의적이거나 잘못된 현상(Port scan, Brute-force, 등)에 의해 비정상적인 트래픽이 발생할 수 있다. 본 논문에서는 분류 속도의 향상과 분류 정확도의 향상을 위해 비정상 플로우를 정상 플로우와 구분하여 다르게 처리한다. 응용 별 트래픽 분류 관점에서 정상/비정상 플로우는 그림 4와 같이 정의한다.

If there is a vector $f \in F_T$, where $F_T = \{f \mid \text{the protocol of } f \text{ is TCP}\}$,
and $f = (P_S, P_{SA}, P_A, \cdots, P_F, P_F, P_A, P_A)$ or $f = (P_S, P_{SA}, P_A, \cdots, P_F, P_{FA}, P_A)$,
P_i is a packet with flag i, then vector f is called *Normal* in traffic classification.

If there is a $f \in F^T$ and f doesn't follow the sequence of packets in *Normal*,
then f is called *Abnormal* in traffic classification.

그림 4. 정상/비정상 플로우 정의
Fig. 4. The definition of normal/abnormal traffic flow

정상/비정상 플로우의 구분은 TCP 플로우에 대해서만 고려하고 UDP 플로우는 모두 정상 플로우로 간주한다. TCP 정상 플로우는 3-핸드셰이크(SYN-SYN/ACK-SYN)로 시작하고 4-핸드셰이크(FIN-FIN-ACK-ACK)로 끝나거나 3-핸드셰이크(FIN-FIN/ACK-ACK)로 끝나는 플로우이며, 비정상 플로우는 정상 플로우에 속하지 않는 모든 TCP 플로우를 뜻한다.

4.4 Buffer Manager

트래픽 분류 시스템이 멀티 코어 환경을 활용하기 위해서는 쓰레드 단위의 작업 분담이 필요하다. 그림 5는 트래픽 분류 시스템의 모듈과 쓰레드 간의 관계를 나타내고 있다.

시스템은 플로우를 생성하는 메인 쓰레드와 생성된 플로우를 분류하는 여러 개의 분석 쓰레드들로 구성된다.

메인 쓰레드는 패킷 캡쳐에서부터 플로우 생성과 플로우를 분석하기 위해 Buffer Manager를 통해 쓰기 버퍼에 플로우를 저장한다. 버퍼에 저장되는 플로우의 조건은 TCP/UDP 플로우, 분석되지 않은 플로우, 데이터 패킷의 개수가 10개 이하인 플로우이다.

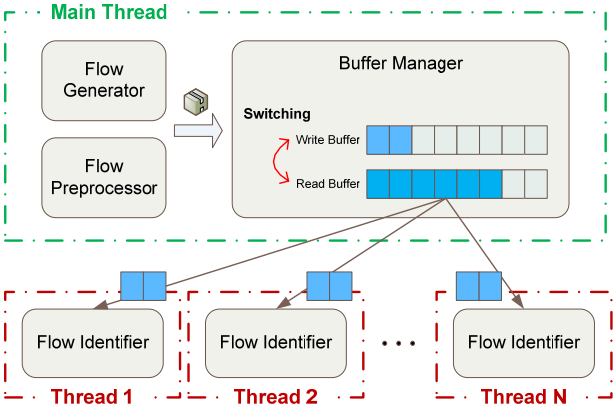

그림 5. 모듈과 쓰레드 간의 관계
Fig. 5. Relationship among processing modules and threads

버퍼의 각 원소에는 해쉬 테이블에 저장된 플로우의 주소와 해당 플로우의 현재 데이터 패킷의 개수가 저장되어 있다. 버퍼에 분석하고자 하는 플로우의 복사본이 아닌 포인터 값을 지님으로써 복사에 의한 시간낭비를 줄이고, 현재 데이터 패킷의 개수를 기록함으로써 분석 쓰레드가 플로우를 분류하는 동안 해당 플로우의 패킷이 메인 쓰레드에 의해서 추가되더라도 분석 쓰레드는 버퍼에 기록된 패킷의 수만큼만 검사한다. 플로우 내에서 트래픽 분류 알고리즘에 의해 이용되는 정보는 주소 정보와 각 패킷 정보이며, 플로우의 패킷이 추가되면서 변하는 통계 정보는 이용되지 않는다. 따라서, 메인 쓰레드와 분석 쓰레드 간의 데이터 동기화 문제를 피할 수 있다. 또한, 현재 쓰고 있는 버퍼 내에서는 중복된 플로우가 존재하지 않도록 최신의 플로우만 저장하고 같은 플로우를 중복해서 분류하지 않게 한다.

쓰기 버퍼에 저장된 플로우들은 분석 시점이 되면

읽기 버퍼로 전환되고 여러 개의 분석 쓰레드들에게 공평하게 분석해야 될 플로우의 양을 분배한다. 분석 쓰레드에 의해 읽기 버퍼의 플로우들이 분석되는 동안, Buffer Manager는 쓰기 버퍼에 플로우를 저장한다. 트래픽의 분석 시점은 저장된 플로우의 수, 캡쳐된 패킷 수, 경과 시간 등 다양할 수 있지만, 본 시스템에서는 네트워크에서 발생하는 패킷의 양을 감안하여 캡쳐 된 패킷 수로 결정하였다. 읽기 버퍼의 플로우를 모두 분석하기 전에 쓰기 버퍼의 공간이 다 차지 않도록 버퍼의 크기를 적절히 조절해야 한다.

4.5 Flow Identifier

본 절에서는 읽기 버퍼에 저장된 플로우들을 분석하는 분석기에 대해서 설명한다. 읽기 버퍼의 저장된 n개의 플로우는 m개의 분석 쓰레드에 의해 분석된다. 그림 6은 Flow Identifier의 슈도코드이다.

Buffer Manager에 의해 분석 쓰레드가 시작되며, 본 시스템에서는 쓰레드 간의 동기화에 세마포어를 사용한다. 가장 먼저 m 개의 분석 쓰레드는 하나의 읽기 버퍼에서 자신이 분석해야 될 플로우들의 위치를 계산한다. 여러 개의 쓰레드가 공평한 개수의 플로우를 분석할 수 있도록 한다.

플로우의 분류 방법은 멀티레벨 분류 방법론[5]을 사용하며 다음과 같은 분류 과정을 거친다. 헤더, 통계, 페이로드의 3개의 시그니쳐 기반 분류 방법론과 1개의 행동양식 기반 방법론을 통해 트래픽을 분류한다. 하나의 플로우는 4가지의 분류 방법에 의해 분류 결과를 도출하고, 이를 우선순위 기반 통합 알고리즘에 의해 하나의 결과로 도출한다. 통합 알고리즘에 의해 도출된 분류 결과가 응용으로 구분되지 않을 경우에는 상관관계 기반의 분류 방법을 통해 추가적인 분석을 행한다.

V. 시스템 구현 및 성능 평가

본 장에서는 트래픽 분석 시스템의 구현 내용과 캠퍼스 네트워크에 적용한 성능평가 결과에 대하여 기술한다.

5.1 시스템 구현

4장의 시스템 설계를 바탕으로 트래픽 분류 시스템을 캠퍼스 네트워크와 기숙사 네트워크를 대상으로 개발하였다.

```
void* flowIdentifierThread(void *arg)
{
    while ( TRUE )
    {
        sem_wait(&call);

        set start_index & end_index in read buffer;

        for ( i = start_index;  i <= end_index;  i++ )
        {
            Header(read_buf[i]);
            Statistic(read_buf[i]);
            Payload(read_buf[i]);
            Behavior(read_buf[i]);

            Integration();

            if ( flow is unknown ) Correlation(read_buf[i]);

            set code to flow;
        }

        sem_post(&end);
    }
}
```

그림 6. Flow Identifier의 슈도코드
Fig. 6. Pseudo-code for the Flow Identifier

2개의 네트워크 각각에서 모든 트래픽을 캡쳐할 수 있는 최상단 링크에서 미러링을 통해 패킷을 캡쳐하였다. 캠퍼스 네트워크는 400Mbps, 기숙사 네트워크는 1.4Gbps 네트워크로 구성되며, 개발환경은 표 2와 같다.

표 2. 시스템 개발환경
Table 2. System Development Environment

	Campus	Dormitory
CPU	Intel Core2 Quad Q6600 2.4G	Intel Core i7 930 2.8G
# of cores	4	8
Memory	4.0G	12.0G
OS (Kernel)	CentOS 5.5 (2.6.18)	CentOS 5.5 (2.6.18)

학교 네트워크에 설치된 시스템은 Intel Core2 Quad Q660 2.4G의 CPU로 4개의 코어와 4기가의 메인 메모리로 구성되어 있으며, 기숙사 네트워크에 설치된 시스템은 Intel Core i7 930 2.8G의 CPU로 하이퍼스레딩 기술로 8개의 코어가 동작하고 12기가의 메인 메모리로 구성된다. 그리고 CentOS 5.5의 리눅스 운영체제에서 C++ 언어로 개발되었다. 트래픽 분류시스템에서 메인 쓰레드와 분석 쓰레드들의 총 개수는 CPU 코어 개수로 설정하여 각각의 쓰레드가 하나의 코어를 개별적으로 사용할 수 있도록 구현하였다.

5.2 시스템 성능 평가

개발된 멀티레벨 기반 실시간 트래픽 분류 시스템이 얼마만큼의 성능으로 동작하는지 테스트하기 위해 학교 네트워크와 기숙사 네트워크에서 구동 후 CPU 사용량을 중점으로 분석하였다.

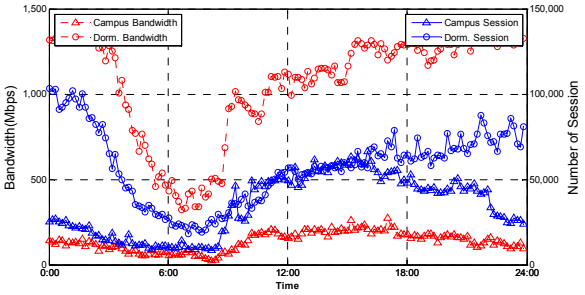

그림 7. 학내/기숙사 망의 트래픽 추이
Fig. 7. Traffic in campus/dormitory network

그림 7은 각각 학내 망과 기숙사 망에 대한 하루 동안의 트래픽 추이를 그래프로 나타내고 있다. 트래픽의 양은 대역폭과 세션 수로 나타내고 있으며, 기숙사 망이 학내 망보다 평균 약 8배 정도의 대역폭을 가지며, 세션의 수는 평균 약 2배가 많다. 특히, 기숙사 망은 저녁시간부터 새벽시간까지 할당된 대역폭을 모두 사용하는 것을 보였다.

표 3. 학내 망에서의 시스템 CPU 사용률
Table 3. CPU usage of the system in campus network

(단위 : %)

Total Usage			
	User	System	User + System
Min	4.58	3.10	9.52
Max	119.93	25.88	143.95
Avg	50.77	13.08	63.85
Core Usage			
	Core0	Core1	Core2	Core3
Min	2.45	2.45	2.22	5.70
Max	42.97	55.07	40.65	37.88
Avg	18.74	20.43	18.05	20.26

표 3과 표4는 학교와 기숙사 네트워크에 대한 각각의 분류 시스템의 CPU 사용률을 나타낸 표이다. CPU 사용률은 모든 코어의 합을 나타내는 전체 사용률뿐만 아니라 각 코어 별 사용률을 같이 측정하였으며, 전체 사용률은 사용자 모드와 시스템 모드에 대해 각각 측정하였다.

표 4. 기숙사 망에서의 시스템 CPU 사용률
Table 4. CPU usage of the system in dormitory network

(단위 : %)

Total Usage			
	User	System	User + System
Min	20.30	3.51	26.87
Max	209.03	14.99	223.42
Avg	96.48	8.87	105.56
Core Usage								
	Core0	Core1	Core2	Core3	Core4	Core5	Core6	Core7
Min	1.50	1.28	1.15	7.98	0.97	8.93	1.88	3.60
Max	32.42	26.43	26.07	37.65	25.53	37.98	23.43	35.08
Avg	11.73	11.36	12.50	24.16	12.47	24.38	11.48	17.32

분류 시스템의 전체적인 CPU 사용률은 기숙사에서 가장 많은 트래픽을 발생시키는 시간대에 약 223%의 최대 부하를 나타내었다. 이는 코어가 8개인 시스템(최대 800%)에서 수용할 수 있는 부하이며, 기숙사 망에서 평균적으로 105%의 CPU 사용률을 보인다. CPU의 시스템 모드 사용률은 대부분 패킷을 캡쳐하는 데 사용되며, 학내 망과 기숙사 망을 비교했을 때, 트래픽 양이 많은 차이를 나타내더라도 사용률은 이에 비례하지 않은 것으로 나타났다. 각 코어 별 사용률을 조사해 본 결과, 고른 형태의 사용률을 나타내고 있는 것으로 보아 분류 시스템이 코어들을 적절히 잘 분배해서 사용하는 것으로 볼 수 있다.

Ⅵ. 결론 및 향후 과제

네트워크에서 발생하는 트래픽이 복잡 다양해지고 있고, 그에 따라 단일 트래픽 분류 알고리즘으로는 모든 트래픽을 분류하기 어려워졌다. 따라서, 최근에는 여러 가지 분류 방법론을 통합한 멀티레벨 분류 방법론이 많이 연구되고 있다. 멀티레벨 분류 방법론은 다양한 분류 방법을 합쳐놓은 형태이기 때문에 다른 방법론에 비해 계산 복잡도가 높다.

본 논문에서는 계산 복잡도가 높은 트래픽 분류 시스템을 실시간으로 동작시키기 위해 멀티 코어를 이용한 시스템 구조를 설계하고 구현한 내용을 기술하였다. 개발된 시스템은 실제 학내 망과 기숙사 망으로 통해 그 성능을 평가하였으며, 실시간 트래픽 처리를 허용할 수 있는 CPU 부하 내에서 처리 가능함을 보였다.

향후 연구로 본 논문에서 실험한 네트워크보다 큰 대용량의 네트워크에서 동작시키고 문제점을 파악하여 시스템을 개선하고자 한다. 또한, 트래픽 분류 시스템의 시스템적인 고려뿐만 아니라 성능 개선을 위한 분석 알고리즘의 개선, 시그니쳐 탐색 공간의 최

적화와 같은 연구를 진행할 계획이다.

참 고 문 헌

[1] G. Zhang, G. Xie, J. Yang, Y. Min, Z. Zhou, X. Duan, “Accurate online traffic classification with multi-phases identification methodology,” *Proc. of CCNC2008*, pp. 141–146, Jan. 2008.

[2] L. Jun, Z. Shunyi, L. Yanqing, Y. Junrong, “Hybrid Internet Traffic Classification Technique,” *Journal of Electronics*, vol. 26 No. 1, pp. 101-112, Jan. 2009.

[3] Z. Chen, B. Yang, Y. Chen, A. Abraham, C. Grosan, and L. Peng, “Online hybrid traffic classifier for Peer-to-Peer systems based on network processors,” *Applied Soft Computing*, vol. 9, pp. 685-694, Mar. 2009.

[4] C. Gu, S. Zhuang, Y. Sun, J. Yan, “Multi-levels traffic classification technique,” *Future Computer and Communication (ICFCC)*, pp. 448-452, May. 2010.

[5] Young-Suk Oh, Jun-Sang Park, Sung-Ho Yoon, Jin-Wan Park, Sang-Woo Lee, Myung-Sup Kim, “Multi-Level basd Application Traffic Classification Method”, KICS Journal vol.35 no.8, pp.1170-1178, Aug. 2010.

[6] IANA port number list, IANA, http://www.iana.org/ assignments/port-numbers.

[7] Jian Zhang and Andrew Moore, ″Traffic Trace Artifacts due to Monitoring Via Port Mirroring,″ *Proc. of E2EMON2007*, May. 21, 2007.

[8] Risso, F. Baldi, M. Morandi, O. Baldini, A. Monclus, P. Lightweight, “Payload-Based Traffic Classification: An Experimental Evaluation,” *Proc. of ICC2008*, 2008.

[9] Jun-Sang Park, Sung-Ho Yoon, and Myung-Sup Kim, ″Software Architecture for a Lightweight Payload Signature-based Traffic Classification System,″ *Proc. of TMA 2011*, LNCS6613, Vienna, Austria, Apr. 27, pp. 136-149, 2011.

[10] Subhabrata Sen , Oliver Spatscheck , Dongmei Wang, “Accurate, scalable in-network identification of p2p traffic using application signatures” *World Wide Web 2004*, May 17-20, 2004, New York, USA.

[11] Jeffrey Erman, Martin Arlitt, Anirban Mahanti, ″Traffic Classification Using Clustering Algorithms,″ *Proc. of SIGCOMM Workshop on Mining network data*, Pisa, Italy, pp. 281-286, Sep. 2006.

[12] Andrew W. Moore and Denis Zuev, ″Internet Traffic Classification Using Bayesian Analysis Techniques,″ *Proc. of the ACM SIGMETRICS*, Banff, Canada, Jun. 2005.

[13] Sung-Ho Yoon, Jin-Wan Park, Young-Seok Oh, Jun-Sang Park, and Myung-Sup Kim, ″Internet Application Traffic Classification Using Fixed IP-port,″ *Proc. of APNOMS 2009*, LNCS5787, Jeju, Korea, Sep. 23-25, pp. 21-30, 2009.

[14] Jin-Wan Park, Sung-Ho Yoon, Jun-Sang Park, Sang-Woo Lee, Myung-Sup Kim, ″Statistic Signature based Application Traffic Classification″, KICS Journal vol.34 no.11, pp.1234-1244, Nov. 2009.

[15] Sang-Woo Lee, Jin-Wan Park, Sung-Ho Yoon, Hyun-Shin Lee, Myung-Sup Kim, ″Research on Skype Traffic Classification using Behavior Analysis″, Proc. of JCCI 2010, pp. 15, Apr. 28-30, 2010.

[16] Rentao Gu, Minhuo Hong, Hongxiang Wang, and Yuefeng Ji, ″Fast Traffic Classification in High Speed Networks,″ *Proc. of APNOMS2008*, LNCS 5297, pp. 429–432, Beijing, China, Oct. 22-24, 2008.

[17] Myung-Sup Kim, Young J. Won, and James Won-Ki Hong, ″Application-Level Traffic Monitoring and an Analysis on IP Networks,″ *ETRI Journal*, Vol.27, No.1, pp.22-42, Feb. 2005.

[18] Thomas Karagiannis, Konstantina Papagiannaki, and Michalis Faloutsos. ″BLINC: Multilevel Traffic Classification in the Dark,″ *Proc. of SIGCOMM 2005*, Philadelphia, PA, Aug. 22-26, 2005.

윤 성 호 (Sung-ho Yoon) 준회원

2009년 고려대학교 컴퓨터정보학과 학사

2009년~2011년 고려대학교 컴퓨터정보학과 석사졸업

2011년~현재 고려대학교 대학원 컴퓨터정보학과 박사과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

박 준 상 (Jun-sang Park) 준회원

2008년 고려대학교 컴퓨터정보학과 학사

2008년~2010년 고려대학교 컴퓨터정보학과 석사졸업

2010년~현재 고려대학교 대학원 컴퓨터정보학과 박사과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

김 명 섭 (Myung-sup Kim) 종신회원

1998년 포항공과대학교 전자계산학과 학사

1998년~2000년 포항공과대학교 컴퓨터공학과 석사

2000년~2004년 포항공과대학교 컴퓨터공학과 박사

2004년~2006년 Post-Doc., Dept. of ECE, Univ. of Toronto, Canada

2006년~현재 고려대학교 컴퓨터정보학과 부교수

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석, 멀티미디어 네트워크