# 헤더 기반 인터넷 응용 트래픽 분석을 위한 시그니쳐 관리 방법에 관한 연구

# Research on Signature Maintenance Method for Internet Application Traffic Identification using Header Signatures

윤 성 호* 김 명 섭**
Sung-Ho Yoon Myung-Sup Kim

## 요 약

효율적인 네트워크 관리를 위한 응용 트래픽 분석의 중요성이 강조되고 있다. 헤더 기반 분석 방법론은 기존 분석 방법론의 한계점들(분석 오버헤드, 페이로드 암호화 등)을 극복하기 위해 응용 트래픽의 헤더 정보를 시그니쳐로 추출(IP address, port number, transport layer protocol (TCP/UDP))하여 트래픽을 분석한다. 헤더 기반 트래픽 분석 방법론은 헤더 정보를 사용하기 때문에 많은 양의 시그니쳐가 추출된다. 따라서 최적의 시그니쳐를 유지할 수 있는 관리 방법이 필요하다. 본 논문에서는 시그니쳐로 분석된 트래픽의 특성과 시그니쳐의 분석이력을 이용하여 최적의 시그니쳐를 관리하는 방법론을 제안한다. 또한, 실험과 검증을 통하여 헤더 시그니쳐 관리 방법의 타당성을 증명한다.

## ABSTRACT

The need for application traffic classification becomes important for the effective use of network resources. The header-based identification method uses the header signature {IP address, port number, transport layer protocol (TCP/UDP)}extracted from Internet application server to overcome some limitations (overhead, payload encryption, etc.) of previous methods. A lots signature is extracted because this method uses header information of server. So, we need a maintenance method to keep essential signatures. In this paper, we represent the signature maintenance method using properties of identified traffic and history of the signature. Also, we prove the feasibility and applicability of our proposed method by an acceptable experimental result.

☞ keyword : Traffic identification; Traffic Classification; Application identification; Application signatures
트래픽 분석, 트래픽 분류, 응용 분석, 응용 시그니쳐

## 1. 서 론

최근 인터넷 사용자의 증가와 고속 네트워크의 보급으로 네트워크 트래픽이 급증하였다. 이는 단순히 WWW, FTP, E-mail 과 같은 전통적인 인터넷 서비스뿐만 아니라 멀티미디어 스트리밍, P2P(peer-to-peer) 파일 공유, 게임과 같은 다양한 멀티미디어 서비스의 대중화에 원인이 있다. 인터넷 트래픽이 급증함에 따라 효율적인 네트워크 관리를 위한 트래픽 모니터링 및 분석의 중요성이 커지고 있다[1,2].

인터넷 트래픽 분석은 다양한 목적으로 이루어질 수 있겠지만 본 논문에서는 분석 대상 네트워크의 트래픽을 수집하여 분석 기준(프로토콜, 응용, 타입 등)에 맞게 트래픽을 분류하고 분류된 트래픽을 수량적으로 측정을 하는 것을 의미한다. 본 논문에서는 트래픽 분류 기준을 트래픽을 발생시킨 응용으로 정하였다. 즉, 인터넷 트래픽을 해당 트래픽을 발생시킨 응용을 기준으로 분류한다. 이렇게 분석된 정보는 네트워크 관리 및 보안에 중요한 자료로 사용될 수 있다. 특히, 효과적인 네트워크 자

* 정 회 원 : 고려대학교 대학원 컴퓨터정보학과 박사과정
sungho_yoon@korea.ac.kr
** 정 회 원 : 고려대학교 컴퓨터정보학과 부교수
tmskim@korea.ac.kr
[2011/07/11 투고 - 2011/07/13 심사 - 2011/09/08 심사완료]

원 관리를 위해 특정 응용에서 발생하는 트래픽의 대역폭을 조절하거나 차단하기 위해서는 대용량의 트래픽을 실시간으로 정확하게 분석하는 트래픽 분석이 반드시 선행되어야 한다.

정확한 트래픽 분석을 위한 많은 연구가 제안되었다. 트래픽을 분석하는 방법 중 가장 원시적인 방법은 IANA에서 지정한 포트 번호를 사용하는 것이다[3]. 이 방법은 단순히 포트 번호를 비교하기 때문에 매우 간단하게 트래픽을 분석할 수 있다. 하지만, 최근 사용되는 응용들은 포트 기반의 방화벽을 원활히 통과하기 위해 동적 혹은 알려진 포트(Well-Known port)를 사용한다. 따라서 포트 번호 기반 분석 방법은 70% 미만의 정확도를 가진다[4].

최근에 발표된 여러 논문에서는 포트 번호와 호스트 정보를 조합한 헤더 기반 분석 방법을 제안한다[4-7]. 포트 번호만을 사용하여 트래픽을 분석 할 경우 낮은 정확도를 가지만, 특정 응용을 제공하는 서버의 헤더 정보{IP address, port number, transport layer protocol (TCP/UDP)}를 함께 사용하면 매우 정확하고 빠르게 트래픽을 분석 할 수 있다.

선행 연구[8]에서는 시그니쳐를 추출하는 방법과 이를 이용하여 트래픽을 분석하는 방법에만 초점을 맞추었다. 하지만 분석의 성능을 높이기 위해서는 최적의 시그니쳐를 관리하는 방법이 필요하다. 인터넷 응용은 사용자의 요구에 따라 매번 수정되고 새로 출현하기 때문에 지속적인 관리가 필요하다. 즉, 급격히 증가하는 시그니쳐 중 일회적으로 사용되는 시그니쳐를 조기에 판단하여 삭제하고 자주 사용되는 시그니쳐를 유지 시킴으로써 분석기의 메모리 오버헤드를 낮추고 분석 성능을 향상시켜야 한다. 특히, P2P 응용의 경우 다수의 호스트와 트래픽을 발생 시키기 때문에 많은 시그니쳐를 발생시키지만, 대부분 일회적인 시그니쳐이기 때문에 분석 성능을 저하시킨다.

헤더 시그니쳐를 관리하는 가장 단순한 방법은 일정 시간 사용하지 않는 시그니쳐를 삭제하는 방법이다[7]. 하지만, 이 방법은 적절한 timeout값을 결정하기 어렵고, 다양한 응용의 특성을 고려하지 않는 동일한 timeout값을 적용하기 때문에 최적의 시그니쳐를 관리하지 못한다. 즉, 비교적 트래픽을 많이 발생하는 P2P 응용의 시그니쳐를 관리하기 위해 지속적으로 사용하지만 사용 주기가 긴 응용의 시그니쳐를 삭제해야 한다.

본 논문에서는 최적의 헤더 시그니쳐를 트래픽 분석에 활용하기 위해 응용이 발생하는 트래픽의 발생 형태와 특성, 그리고 사용이력을 고려한 시그니쳐 관리 방법을 제안한다. 또한, 급변하는 네트워크 환경(응용의 수정, 새로운 응용 출현)에 적용하기 위해 지속적인 시그니쳐 추출 및 관리, 트래픽 분석이 가능한 실시간 헤더 시그니쳐 기반 분석 시스템을 제안한다.

제안하는 관리 방법의 타당성을 검증하기 위해 timeout값을 사용하는 방법[7], 추출된 모든 시그니쳐를 사용하는 방법[8]과 성능을 비교하였다. 비교한 결과, 분석률(completeness) 측면에서 timeout값을 사용하는방법보다 약 18%(byte 기준) 성능이 향상된 것을 확인 할 수 있었다. 또한 시그니쳐 개수 측면에서도 추출된 모든 시그니쳐 중 약 1.58% 의 시그니쳐만 분석에 사용하였다. 즉, 최적의 시그니쳐를 관리하여 최대의 분석 성능을 가지는 것을 확인할 수 있었다.

본 논문은 다음과 같이 구성되었다. 2 장에서는 관련 연구를 살펴보며 3장에서는 헤더 시그니쳐 기반 분석시스템과 제안하는 시그니쳐 관리 방법을 설명한다. 4장에서는 실험 및 결과를 5장에서는 결론 및 향후 연구를 기술한다.

## 2. 관련 연구

포트 번호 기반 분석 방법의 낮은 정확도를 보완하기 위해 헤더 기반 분석 방법이 제안되었다[4-7]. 이들 연구들의 공통적인 아이디어는 특정 응용의 서버는 일정 기간 동안 지속적으로 동일한 응용을 서비스하고 이러한 헤더 정보를 이용하여 트래픽을 분석하면 많은 이점이 있다는 점이다. 헤더 정보를 이용하여 트래픽을 분석 할 경우 얻을 수 있는 이점은 다음과 같다[7].

(표 1) 헤더 정보 기반 분석 방법

| Paper | Header format | Purpose | Maintain method |
|---|---|---|---|
| Toward the Accurate Identification of Network Application[4] | {host/port} | Verification of Identification Result | Only active host |
| GTVS: Boosting the Collection of Application Traffic Ground Truth[5] | {dst IP, dst port} {src IP, dst IP} | Additional Identification | Not mentioned |
| Transport Layer Identification of P2P Traffic[6] | {TCP/UDP IP pairs} {IP, port}pairs | Identification | Accumulated |
| Service-based Traffic Classification: Principles and Validation[7] | {IP address, TCP/UDP port} | Additional Identification | Inactivity timeout |

- 트래픽 수집 시 발생 할 수 있는 패킷 손실 및 TCP/IP fragmentation 문제를 고려하지 않는다.
- 헤더 정보를 통해 트래픽을 분석하기 때문에 페이로드가 없거나 암호화된 경우에도 분석할 수 있다.
- 트래픽의 헤더 정보는 고정된 위치에 존재하기 때문에 추출과 분석이 용이하다.

(표 1)은 각 논문에서 사용한 헤더 정보의 형태와 목적, 그리고 관리 방법을 보여준다.

[4]는 수행 복잡도(Port < Signature < Protocol)와 입력 데이터 형태(Packet< 1stKbyte < selected Flow < all Flow)를 기준으로 여러 분석 방법론을 분류하고 이들을 순차적으로 적용하여 트래픽을 분석한다.

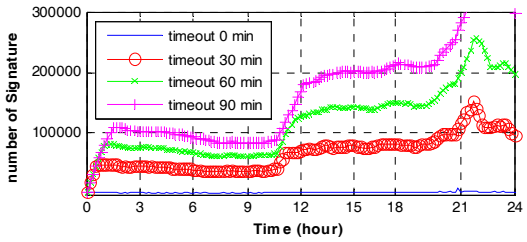
이 논문에서는 port scanning, streaming audio 등과 같은 독특한 발생 형태의 응용 트래픽을 검증하기 위하여 분석된 결과의 host/port 정보를 활용한다. 헤더 정보는 해당 호스트가 활성화되어 있는 동안만 유지한다.

[5]는 정답지(Ground-truth) 트래픽 생성 방법을 소개한다. 페이로드 기반으로 분석된 결과에 응용 트래픽의 발생 형태와 특성을 고려한 다섯 가지 분석 단계를 반복적으로 적용하여 정답지 트래픽을 생성한다. 이 논문에서는 동일한 sub-tuple({dst IP, dst port}, {src IP, dst IP})을 포함하는 모든 flow는 동일한 서비스 혹은 응용에서 발생한 것이라 가정한다. 따라서 선행 분석 방법으로 분석된 트래픽에서 특정 응용으로 분석된 트래픽의 sub-tuple을 추출하고 이를 사용하여 분석되지 않은 트래픽을 분석한다. 헤더 정보의 관리 방법은 언급하지 않았다.

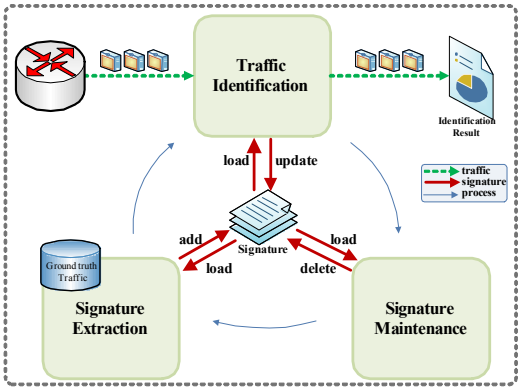
[6]은 P2P 응용의 트래픽 발생 형태와 통계적 특성을 기반으로 P2P 트래픽을 분석한다. 즉, P2P 응용 트래픽을 구분하는 규칙을 통해 특정 {TCP/UDP IP pairs}과 {IP, port}를 발생 시킨 호스트의 헤더 정보를 P2PIP 목록에 일정 기간 유지시켜 해당 호스트에서 발생한 트래픽을 분석한다. 알고리즘 소개에 초점을 맞추었기 때문에, 헤더 정보의 특별한 관리 방법을 제안하지 않고, 단순히 누적시키는 방법을 사용하였다.

[7]은 특정 서버에서 제공하는 서비스(응용)을 분석하고, 해당 서버의 “network coordinates”(IP address and TCP/UDP port) 을 service table에 일정 기간 유지 시킴으로써, 해당 서버에서 발생한 트래픽을 분석한다. 이 분석 방법은 다른 분석 방법(페이 로드 시그니쳐 기반)으로 분석된 결과에서 특정 응용을 서비스하는 서버 정보를 추출하여 선행 분석 방법으로 분석되지 않은 트래픽(페이로드가 없는)을 추가적으로 분석한다. 즉, 선행 분석방법의 성능향상(completeness)에 초점을 맞추었다. 또한, 적정 크기의 service table를 유지하고 일시적으로 사용되는 응용 서버로 인한 부정확한 결과를 방지하기 위해 service activity timeout값을 사용하여 일정 기간 사용하지 않는 헤더 정보를 삭제한다.

(그림 1)은 시그니쳐 관리 방법 중 가장 단순한 방법인 일정 timeout값을 사용하여 시그니쳐를 관리[7]하였을 때 변화하는 시그니쳐의 개수를 보여




(그림 1) Timeout 값에 따른 시그니쳐 개수 변화




(그림 2) 분석 시스템 구조

준다. 즉, 실제 발생되는 트래픽을 대상으로 추출된 시그니쳐를 가지고 분석 하였을 때, 특정 시그니쳐가 일정 timeout 값 이상 분석에 사용되지 않으면 시그니쳐 목록에서 삭제한다.

(그림 1)과 같이 시그니쳐의 개수는 시그니쳐 추출의 원천이 되는 정답지 트래픽의 양에 따라 변화한다. 즉, 특정 응용을 대표하는 헤더 시그니쳐일지라도 지속적으로 정답지 트래픽에서 일정 timeout값 동안 나타나지 않으면 시그니쳐 목록에서 삭제된다. 또한, 동일한 timeout 값을 적용하였기 때문에 상대적으로 긴 사용 주기를 가지는 응용의 시그니쳐는 목록에서 지속적으로 삭제된다. 따라서 응용의 특성을 반영한 관리 방법이 필요하다.

## 3. 헤더 시그니쳐 기반 트래픽 분석

헤더 시그니쳐 기반 트래픽 분석 방법은 특정 응용을 서비스하는 서버의 헤더 정보를 시그니쳐로 추출하여 트래픽을 분석한다. 본 논문에서 제안하는 헤더 시그니쳐는 특정 응용을 서비스하는 서버의 3-tuple{IP address, port number, transport layer protocol (TCP/UDP)}이다. 분석 시스템의 구조는 (그림 2)와 같다.

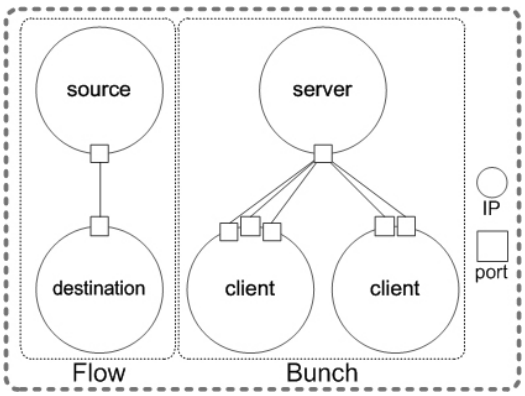
기존에 제안된 시그니쳐 기반 분석 방법(payload, statistic 기반)들은 특정 기간 동안(off-line) 수집한 트래픽에서 응용 시그니쳐를 추출하여 사용한다. 하지만, 인터넷 응용은 사용자의 요구에 따라 매번 수정되고 새로 출현하기 때문에 실시간 시그니쳐 추출 및 관리가 필수적이다[9]. 따라서, 그림 2와 같이 정답지 트래픽(Ground truth Traffic)을 실시간으로 수집하여 시그니쳐를 추출하고, 트래픽을 분석한다. 또한, 시그니쳐 관리 방법을 통해 최적의 시그니쳐로 관리한다.

본 시스템은 총 세 모듈로 구성된다. (1) 시그니쳐추출(Signature Extraction), (2) 트래픽 분석(Traffic Identification), 그리고 (3) 시그니쳐 관리(Signature Maintenance) 모듈이다.

### 3.1 시그니쳐 추출 모듈

시그니쳐 추출 모듈에서는 정답지 트래픽에서 특정 응용을 서비스하는 서버의 헤더 정보를 시그니쳐로 추출한다. 정답지 트래픽을 생성하는 방법은 이미 많은 연구를 통해 제안되었다[5][10]. 본 연구에서는 실제 인터넷 트래픽을 발생 시키는 호스트에 Agent를 설치하여 정답지 트래픽을 수집하였다[10].

실제 트래픽에서 응용 서버의 3-tuple을 효과적으로 추출하기 위해 새로운 트래픽 단위를 제안한다. 기존 트래픽 분석에서는 트래픽(패킷)을 flow 단위로 조합하여 사용하였다. flow는 같은5-tuple (source and destination IP address, source and destination port, transport layer protocol)을 가지는 단방향 패킷들의 집합이다[11]. 실제 네트워크에서 발생하는 인터넷 응용 트래픽은 특정 서버를 중심으로 발생되기 때문에 flow를 서버 기준으로 재 조합




(그림 3) Flow와 Bunch의 구조

하면 응용의 발생 형태와 flow간의 관계를 효과적으로 파악할 수 있다. 따라서 헤더 시그니쳐 추출을 위해 응용 서버 기준으로 flow를 재 조합한 bunch 를 제안한다. Bunch는 특정 서버의 3-tuple(IP address, port, transport layer protocol)를 포함하는 모든 flow들의 집합이다. Bunch는 하나의 server 노드와 하나 이상의 client 노드로 구성된다. 즉, server 노드의 3-tuple이 헤더 시그니쳐로 추출된다. (그림 3)은 flow와 bunch의 구조를 보여준다.

대부분의 응용 서버는 하나의 서버 포트와 다수의 클라인언트 포트를 통해 트래픽을 발생시킨다. 이러한 관점에서 볼 때, bunch 단위의 트래픽은 특정 서버에서 발생한 응용 트래픽을 더욱 명확하게 나타낸다.

Bunch는 flow를 서버 3-tuple을 기준으로 재 조합한 것이기 때문에 bunch를 생성하기 위해서는 flow의 서버를 구분해야 한다. TCP flow의 경우 SYN, ACK flag를 통해 서버를 쉽게 판별할 수 있지만, UDP flow는 서버 판별이 모호하다. 본 논문에서는 UDP flow인 경우 두 종단 호스트에 발생한 첫 번째 패킷을 서비스 요청(request)으로 가정하고 첫 패킷의 destination 호스트를 서버로 판별한다.

알고리즘 1은 시그니쳐를 추출하는 과정을 보인다.

알고리즘 1은 flow를 입력 받아 헤더 시그니쳐를 생성한다. 본 추출 방법은 flow를 bunch로 재조합하는 부분(line:4~6)과 생성된 bunch의 server 노드에서

*Algorithm 1. extractSignature*

```
1:  Input: Flow={f1,...,fn}, HS= ø
2: Output:Bunch={b1,...,bk }, HS={hs1,...,hsm}
3:  void extractSignature (...){
4:    for( i=1; i≤n ; i++){
5:      makeBeunch(fi);
6:    }
7:   for( i=1; i≤k ; i++){
8:     HS=HS∪{getServer(bi)}
9:   }
10: }
```

시그니쳐를 추출하는 부분(line:7~9)으로 구성된다. 만약 추출된 시그니쳐와 기존에 추출된 시그니쳐가 동일한 3-tuple을 가지고 서로 다른 응용에서 추출된 경우 충돌이라 정의한다. 본 논문에서는 시그니쳐 추출 시 충돌이 발생한 경우, 기존의 시그니쳐와 새로 추출된 시그니쳐 모두 삭제한다.

## 3.2 트래픽 분석 모듈

트래픽 분석 모듈에서는 시그니쳐 추출 모듈에서 생성된 시그니쳐를 사용하여 트래픽을 분석한다. 헤더 시그니쳐를 사용하여 트래픽을 분석 할 때에는 트래픽의 고정된 위치에 존재하는 헤더 정보와 시그니쳐를 비교하기 때문에 다른 분석 방법론 보다 빠르게 분석할 수 있다.

알고리즘 2는 추출된 시그니쳐를 사용하여 트래픽을 분석하는 과정을 보인다.

*Algorithm 2. identifyTraffic*

```
1:  Input: Traw={t1,...,tk}, HS={hs1,...,hsm}
2: Output: Tidentified={t1,...,tk}
3:  void identifyTraffic (...){
4:    for( i=1; i≤k ; i++){
5:      hstemp=getServer(ti);
6:      if(code =foundHS(hstemp))
7:        ti.setCode(code);
8:    }
9:  }
```

트래픽 분석 알고리즘은 트래픽의 server 3-tuple을 추출(line:5)하여 동일한 3-tuple을 가지는 헤더 시그니쳐를 찾는다(line:6). Server 3-tuple을 추출하는 방법은 앞서 설명한 bunch 생성 시 server을 구분하는 방법과 동일하다. 만약 동일한 시그니쳐를 찾았을 경우, 해당 시그니쳐의 응용으로 트래픽을 분석한다(line:7). 실제 분석기를 구현할 때에는 3-tuple 정보를 key로 사용하는Hash 자료 구조를 사용하기 때문에 빠르게 동일한 시그니쳐를 찾을 수 있다. 또한, 시그니쳐의 분석 이력(해당 시그니쳐가 분석한 트래픽의 size, duration, host, usage time 등)을 기록하여 관리 모듈에서 활용한다.

## 3.3 시그니쳐 관리 모듈

시그니쳐 관리 모듈은 무한히 증가하는 헤더 시그니쳐 중 최적의 시그니쳐를 유지하기 위해 분석된 트래픽의 특성과 시그니쳐의 사용 이력을 이용하여 최적의 시그니쳐로 관리한다.

시그니쳐를 관리한다는 의미는 각 시그니쳐의 특성을 파악하여 분석에 유용한 시그니쳐는 목록에 유지 시키고 그렇지 않은 시그니쳐는 목록에서 삭제하는 것을 의미한다. 시그니쳐 관리(Signature Maintenance) 모듈은 총 4가지 세부 관리 방법들로 구성된다.

- 발생 형태를 고려한 관리 방법
- 사용 주기를 고려한 관리 방법
- 트래픽 통계적 특성을 고려한 관리 방법
- 사용 빈도를 고려한 관리방법

알고리즘 3은 본 논문에서 제안하는 시그니쳐 관리 모듈의 전체적인 과정을 보인다.

본 알고리즘의 입력 데이터는 현재까지 추출된 시그니쳐의 목록(HSbefore)이고 출력 데이터는 관리 방법을 사용하여 최적화된 시그니쳐의 목록(HSafter)이다. 추출된 시그니쳐의 특성을 파악하여 비정상(단방향 통신, P2P Peer)트래픽을 분석하는 시그니쳐(isAbnormal)와 해당 응용을 기준으로 일

*Algorithm 3. maintainSignature*

```
1:  Input: HS_before={hs_1,...,hs_m}
2: Output: HS_after={hs_1,...,hs_n}, m ≥n
3:  void maintainSignature (...){
4:    for( i=1; i≤m ; i++){
5:      if(isAbnormal(hs_i)||isTimeout(hs_i))
6:        if(!isDominant(hs_i)&&!isPopular(hs_i))
7:          HS=HS-{hs_i}
8:    }
9: }
```

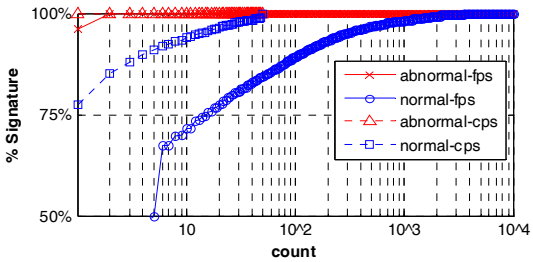
정 기간 사용되지 않은 시그니쳐(isTimeout)를 삭제한다(line:5). 단, 해당 시그니쳐로 분석된 트래픽이 우세한 특징을 가지고(isDominant), 여러 호스트에 의해 사용된 경우(isPopular), 삭제 대상에서 제외된다(line:6).

### 3.3.1 발생 형태를 고려한 관리 방법

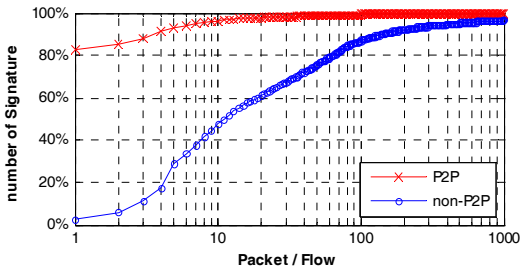
비정상적인 트래픽(port scan, NetBIOS-SMB)과 P2P응용에서 발생한 트래픽은 발생 형태의 특성 때문에 많은 시그니쳐가 추출된다. 하지만, 추출된 시그니쳐의 대부분은 일회적으로 분석에 사용되기 때문에 분석기의 메모리 오버헤드를 증가 시킨다. 따라서 본 관리방법에서는 시그니쳐가 분석한 트래픽의 특성을 분석하여 비정상 트래픽과 P2P응용에서 발생한 트래픽에서 추출 된 시그니쳐를 판별한다.

비 정상 트래픽과 P2P응용의 특성을 조사하기 위하여 트래픽을 수집하였다. 비 정상 트래픽은 학내 망에서 빈번히 발생하는 NetBIOS-SMB(port 445/TCP)[12] 트래픽을 수집하였고, P2P 트래픽은 Bittorent[13], Fileguri[14] 응용에서 발생한 트래픽을 수집하였다.

(그림 4)는 비정상과 정상 트래픽의 시그니쳐 당 분석된 flow수(fps)와 시그니쳐 당 분석된 client 수(cps)를 CDF(Cumulative Distribution Function)로 나타낸다. 그림에서와 같이 비정상 트래픽에서 추출한 대부분의 시그니쳐(96.16%)는 오직 하나의 flow




(그림 4) 비정상 트래픽의 특성




(그림 5) P2P트래픽의 특성

만 분석하였다. 즉, 정상적인 양방향 통신이 아니단 방향 통신에서 발생된 트래픽을 분석하였다. 이에 반해 정상 트래픽에서 추출한 시그니쳐는 소수의 시그니쳐(10.01%)만이 하나의 flow를 분석하였다. 이러한 특성을 이용하여 비정상 트래픽에서 추출한 시그니쳐를 판별할 수 있지만, 좀 더 정확한 판별을 위해 추가적인 특성을 사용한다.

비정상 트래픽에서 추출한 시그니쳐로 분석된 트래픽의 고유한 클라이언트 수(cps)를 조사한 결과 시그니쳐의 cps가 대부분1이었다. 즉, 해당 시그니쳐로 분석된 flow를 앞서 설명한 bunch로 재구성하였을 때, 오직 하나의 client 노드로 구성된다. 따라서 특정 시그니쳐가 분석한 flow의 개수가 1이고, 해당 시그니쳐로 분석된 client 수가 1 이면, 비정상 시그니쳐로 판별한다. 즉, 단일 호스트에서 발생된 단방향 플로우로 추출한 비정상 시그니쳐로 판별한다.

(그림 5)는 P2P응용과 non-P2P응용에서 추출된 시그니쳐가 분석한 트래픽의 flow 당 packet 개수(ppf)를 CDF로 나타낸다. 그림에서와 같이 P2P 시그니쳐로 분석된 트래픽의 대부분(80%)의 flow는 오직 하나의 packet으로 구성되었다. 이러한 트래픽은 P2P응용의 검색 또는 파일 전송에 관련된 것이므로 해당 시그니쳐를 P2P 시그니쳐로 판별한다.

본 관리 방법에서는 비정상 및 P2P 시그니쳐를 판별하기 위해 고유한 트래픽 발생 형태에 따른 특성을 사용한다. 알고리즘 3-1는 자세한 관리 방법을 나타낸다.

알고리즘 3-2는 입력된 시그니쳐를 분석하여 비

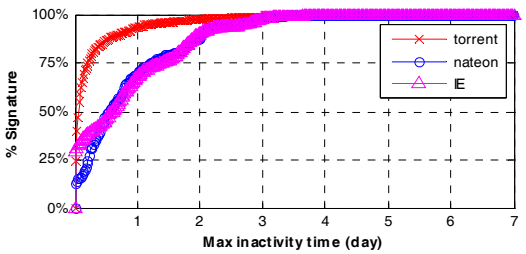
*Algorithm 3-1 isAbnormal()*

```
1:  Input: hs
2: Output:True or False
3:  bool isAbnormal (...){
4:    // check abnormal
5:  if(getIdentfiedFlow(hs)==1)    //one-way flow
6:      if (getIdentifiedClient(hs) ==1)   //exclusive
7:    return True;
8:    // check P2P
9:    if (getIdentifiedPacket(hs)
                    /getIdentifiedFlow(hs)==1);
10:   return True;
11:  return False;
12: }
```

정상 시그니쳐(line:4~7), P2P 시그니쳐(line:8~10)를 판별한다. 비 정상 시그니쳐를 판별하기 위해 해당 시그니쳐로 분석된 flow 개수를 확인(line:5)하고 단방향 flow 이면 해당 트래픽을 발생 시킨 클라이언트 개수를 확인(line:6)한다. 해당 트래픽이 특정 호스트에서 독점적으로 발생 한 것이면 비정상 시그니쳐로 판별한다. P2P 시그니쳐는 해당 시그니쳐로 분석된 트래픽의 flow 당 packet 개수를 확인(line:9)하여 1이면 P2P 시그니쳐로 판별한다.

### 3.3.2 사용 주기를 고려한 관리 방법

시그니쳐 목록의 폭발적인 증가를 막기 위해서는 오랜 기간 사용하지 않는 시그니쳐를 목록에서 제거해야 한다. 하지만, 모든 응용에 동일한 유휴시간 기준값(inactivity timeout)을 적용하면 사용 주기




(그림 6) 응용별 시그니쳐 최대 유휴 시간

가 긴 응용의 시그니쳐는 지속적으로 목록에서 삭제된다[7]. 따라서, 응용의 사용 주기를 반영하여 기준값(timeout)을 설정하고 시그니쳐를 관리해야 한다.

응용별 시그니쳐의 사용 주기를 확인하기 위해 학내망에서 자주 사용하는 응용(web: Internet Explore[15], messenger: Nateon[16], P2P: Bittorrent [13])을 선정하여 시그니쳐를 추출하고 실제 학내망에서 발생하는 트래픽을 분석하였다. 분석에 사용된 시그니쳐에 사용 이력을 기록하여 시그니쳐별 최대 유휴 시간(max inactivity time)을 분석하였다. 정확한 실험을 위해 2회 이상 사용된 시그니쳐를 대상으로 inactivity time을 측정하였다. 즉, 일회성 시그니쳐는 실험 대상에서 제외시켰다.

(그림 6)과 같이 web이나 메신져 트래픽을 분석한 시그니쳐의 유휴 시간은 P2P 응용에서 발생한 트래픽을 분석한 시그니쳐는 유휴 시간보다 상대적으로 길었다. 즉, 응용마다 상이한 유휴 시간을 고려하여 일정 시간 분석에 사용되지 않는 시그니쳐를 목록에서 제거해야 한다.

하지만, 네트워크의 환경에 따라 특정 응용에 적합한 timeout값이 다르기 때문에 유동적인 응용 별 timeout 값 생성 방법을 고려해야 한다. 본 논문에서는 특정 응용의 timeout 값을 유동적으로 구하기 위해 해당 응용 시그니쳐 중 가장 긴 유휴시간의 α배한 값을 사용한다. 본 논문에서 사용한 α는 1.1이다.

본 관리 방법은 최적의 시그니쳐를 관리하기 위해 응용별 최대 유휴 시간을 사용한다. 알고리즘 3-2은 자세한 관리 방법을 나타낸다.

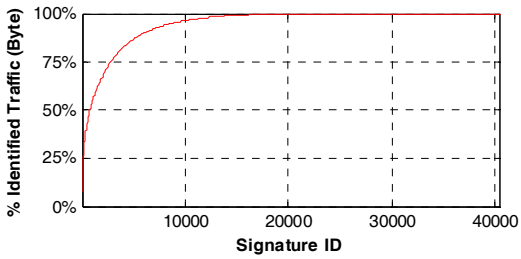
*Algorithm 3-2. isTimeout()*

```
1:  Input: hs
2: Output:True or False
3:  bool isTimeout (...){
4:     application = getApp(hs));
5:     inactivityTime=currentTime-getLastTime(hs);
6:     if(inactivityTime
           >getMaxInactivityTime(application)* α)
7:     return True;
8:   return False;
9: }
```

알고리즘 3-2는 입력된 시그니쳐의 최대 유휴시간과 해당 응용의 최대 유휴 시간을 비교하여 일정 기간 이상 사용 하지 않은 시그니쳐를 판별한다. 입력된 시그니쳐의 응용을 확인(line:4)하고, 현재 시간을 기준으로 유휴 시간을 계산한다(line:5). 입력된 시그니쳐의 유휴 시간이 해당 응용에서 추출된 시 그니쳐들의 최대 유휴 시간 1.1배와 비교(line:6)하여 크면 참을 반환(line:7)하고 그렇지 않으면 거짓을 반환(line:8)한다. 즉, 해당 응용의 최대 유휴 시간 보다 오랜 기간 사용하지 않은 시그니쳐를 삭제 대상 시그니쳐로 판별한다. 1.1배를 사용한 이유는 응용 별 유휴 시간 기준을 응용의 사용량에 따라 유동적으로 변화시키기 위함이다. 즉, 현재의 최대 유휴시간보다 더 긴 기준값(1.1배)을 적용함으로써, 응용의 특성에 유동적으로 변화하는 timeout 값을 설정할 수 있다.

### 3.3.3 트래픽 통계적 특성을 고려한 관리

트래픽 특성을 연구한 선행 논문[17,18]에 따르면 소량의 우세한 몇몇 트래픽이 전체 트래픽의 큰 영향을 미친다. 즉, 전체 트래픽에서 소수의 heavy hitter flow가 트래픽의 많은 byte 비율을 차지한다. 이와 마찬가지로 헤더 시그니쳐에도 heavy hitter가 존재 한다. 즉, 트래픽 분석에 있어 우세한 트래픽을 분석하는 우세한 헤더 시그니쳐가 존재한다. 따라서 이러한 우세한 시그니쳐들을 판별하고 이를




(그림 7) 개별 시그니쳐 분석 성능

시그니쳐 테이블에 오래 유지 시킴으로써 분석 성능을 향상시킬 수 있다.

우세한 시그니쳐의 존재를 확인하기 위하여 일정 기간 동안 시그니쳐를 추출하고 해당 시그니쳐로 분석된 트래픽의 비율을 조사하였다.

(그림 7)은 시그니쳐 별 분석 트래픽의 비율을 누적한 결과를 보여준다. 우세한 시그니쳐를 확인하기 위하여 분석한 트래픽의 비율 높은 순서로 시그니쳐 아이디를 부여하였다. (그림 7)에서와 같이 일부 시그니쳐(73.56%) 만이 트래픽 분석에 사용되었고, 또한 몇몇 소량의 시그니쳐(15.17%)가 분석된 트래픽의 대부분(90%)을 분석 하였다. 즉, 우세한 트래픽을 분석하는 우세한 시그니쳐가 존재함을 확인 할 수 있다.

본 관리 방법에서는 우세한 시그니쳐를 판별하기 위해 해당 시그니쳐가 분석한 트래픽의 다양한 특성(average size of flow, average duration of flow)를 사용한다. 또한, 응용 별로 상이한 특성을 반영하기 위해 응용 별 평균 트래픽 특성을 기준으로 트래픽 특성을 파악한다. 우세한 특성을 판단하기 위해 각 특성의 평균(mean)과 표준편차(standard deviation)을 사용한다[18]. 알고리즘 3-3은 자세한 관리 방법을 나타낸다.

알고리즘 3-3은 시그니쳐가 분석한 트래픽의 특성을 파악하여 우세한 트래픽을 분석하는 시그니쳐를 판별하는 과정을 보인다. 본 관리 방법은 2가지 측면의 트래픽 특성을 파악하기 위해 반복 구조(line:4~11)를 사용하며, 하나 이상의 측면에서 우세한 특성을 가지면 해당 시그니쳐를 우세한 시그니쳐로 판별한다. 우선, 시그니쳐가 입력되면, 해당 시그니쳐가 추출된 응용을 확인한다(line:6). 확인된 응용으로 분석된 전체 트래픽의 평균(mean) 과 표준 편차(standard deviation)을 구한다(line:7~8). 해당 시그니쳐의 특성과 해당 시그니쳐가 속한 응용의 전체 특성을 비교하여 우세한 시그니쳐를 분석한다(line:9~10). 우세한 트래픽을 판단하기 위해 다양한 방법이 제안되었다. 본 논문에서는 시그니쳐 관리 방법에 초점을 맞추었기 때문에 Lan, K 등[18]이 제안한 판별 방법을 사용하였다. 본 논문에서는 사용한 수식은 다음과 같다.

*Algorithm 3-3. isDominant()*

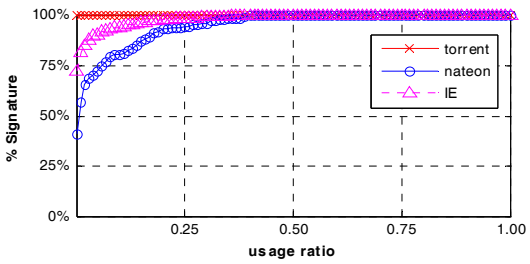
```
1:  Input: hs
2: Output:True or False
3:  bool isDominant (...){
4:   for( i=0 i<2 i++){
5:        // i means 0:size 1:duration
6:        application = getApp(hs));
7:    mean = getMean(application, i);
8:      std = getStd(application, i);
9:      if (getProperty(hs,i) >mean+std*3)
10:    return True;
11:   }
12:   return False;
13: }
```

$$Sig_{dominant} = Sig_{(flow,duration,bye)} > (mean_{app} + 3*std_{app}) \quad (1)$$

해당 응용으로 분석된 전체 트래픽의 평균과 표준편차 3배의 합 보다 큰 경우 우세한 시그니쳐로 판별한다.

### 3.3.4 사용 빈도를 고려한 관리 방법

시그니쳐는 해당 응용 서버를 대표하는 것이기 때문에 해당 응용을 사용 할때 자주 사용되는 것이어야 한다. 따라서, 해당 시그니쳐로 분석된 트래픽을 발생시킨 호스트의 개수와 해당 응용으로 분석




(그림 8) 응용별 시그니쳐 사용률

된 트래픽을 발생시킨 호스트 개수의 비율을 사용하여 시그니쳐의 사용률을 계산하고 이를 이용하여 시그니쳐를 관리한다. 시그니쳐 사용률을 구하는 수식은 다음과 같다.

$$\text{usage ratio}=\frac{\#\text{ of Host using the Sig}}{\#\text{ of Host using the Application}} \quad (2)$$

응용 별 시그니쳐의 사용률을 확인하기 위해 학내망에서 자주 사용하는 응용을 선정하고 사용률의 분포를 분석하였다.

(그림 8)과 같이 단일 호스트에서 트래픽이 발생하는 P2P응용의 경우 대부분 시그니쳐의 사용률이 0인 것에 반해, 특정 서버를 중심으로 트래픽이 발생되는 메신저 응용의 경우 사용률이 상대적으로 높은 시그니쳐가 존재함을 확인할 수 있다. 따라서 본 논문에서는 0이상의 β값을 설정하여 해당 값 이상의 usage ratio을 가지는 시그니쳐를 삭제 대상에서 제외한다.

본 관리 방법은 최적의 시그니쳐를 관리하기 위해 시그니쳐의 사용률을 이용한다. 알고리즘 3-4는 자세한 관리 방법을 나타낸다.

알고리즘 3-4는 시그니쳐로 분석된 트래픽을 발생시키는 호스트의 비율, 즉 사용률을 기준으로 시그니쳐의 인기도를 판별한다. 즉, 사용률이 일정 기준값을 초과하였을 경우, 해당 시그니쳐를 해당 응용의 핵심적인 시그니쳐라 판별한다. 우선, 시그니쳐가 입력되면, 해당 시그니쳐가 추출된 응용을 확인한다(line:4). 확인된 응용을 통해 해당 응용을 사용한 호스트의 개수를 구한다(line:5). 앞서 계산한 응용을 사용한 총 호스트의 개수와 해당 시그니쳐를 사용한 호스트의 개수의 비율(line:6)이 특정 기준값(0.1)이 넘으면 해당 시그니쳐를 핵심적인 시그니쳐로 판별(line:7)한다.

*Algorithm 3-4. isPopular()*

```
1:  Input: hs
2: Output:True or False
3:  bool isPopular (...){
4:     application = getApp(hs));
5:     appClient=getAppClient(application);
6:  if(getClient(hs)/appClient > β)
7:       return True;
8:  return False
9: }
```

## 4. 실험 및 결과

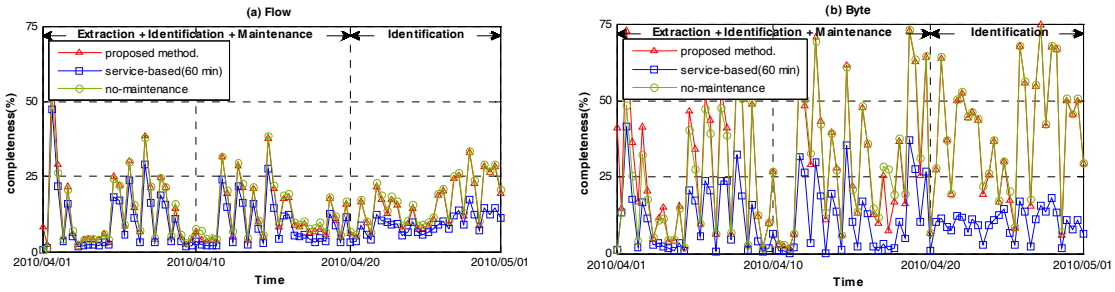
본 논문에서 제안한 헤더 시그니쳐 기반 응용 트래픽 분석 시스템의 성능을 확인하기 위해 학내 망에 분석 시스템을 설치하고 실험하였다.

객관적 분석을 위해 동일한 시간(30일) 동안 서로 다른 두 개의 모집단에서 TMA[10]를 이용하여 정답지 트래픽을 수집하였다. (표 2)는 수집한 정답지 트래픽의 정보를 보여준다.

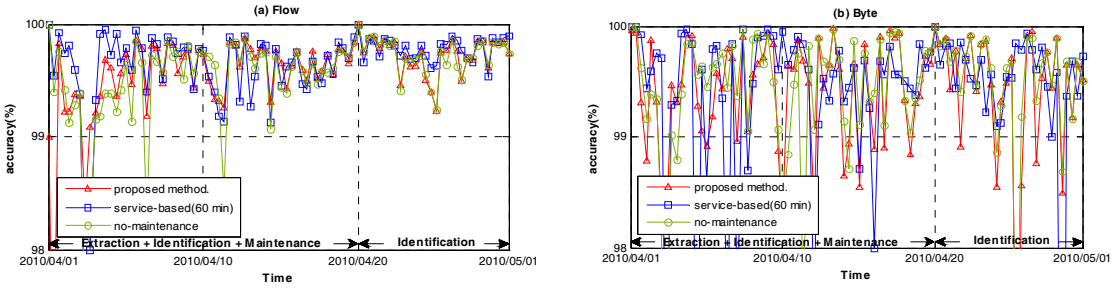
GT-KU201004-#01를 사용하여 시그니쳐를 추출하고 GT-KU201004-#02를 대상으로 분석을 수행하

(표 2) 트래픽 트래이스 정보

| Name | Start time | Duration | Flow | Packet | Byte | # of host |
|---|---|---|---|---|---|---|
| KU201004-#01 | 2010.04.01 00:00 | 30 days | 83,978 X $10^3$ | 1,380 X $10^6$ | 1,046 X $10^9$ | 108 |
| KU201004-#02 | 2010.04.01 00:00 | 30 days | 116,818 X $10^3$ | 1,456 X $10^6$ | 1,008 X $10^9$ | 105 |




(그림 9) 분석률(completeness) 비교




(그림 10) 정확도(accuracy) 비교

였다. 즉, 추출 대상 호스트와 분석 대상 호스트를 달리하여 헤더 시그니쳐의 성능을 객관적으로 측정하였다. 또한, 최초 20일 동안은 헤더 시그니쳐의 추출, 분석 그리고 관리를 동시에 수행하였고, 마지막 10일 동안은 분석만 수행하였다.

(표 3)은 본 논문에서 제안하는 시그니쳐 관리 방법과, 일정 timeout 값(60분)을 기준으로 관리하는 service기반 관리 방법[7], 그리고 시그니쳐 관리를 적용하지 않고 추출된 모든 시그니쳐를 누적시키는 방법[8]의 분석률(completeness)과 정확도(accuracy)를 나타낸다. (그림 9)와 (그림 10)은 자세한 추이를 나타낸다. 분석률과 정확도를 구하는 수식은 다음과 같다.

$$\text{completeness} = \frac{\text{Identified Traffic}}{\text{Total Traffic}} \quad (3)$$

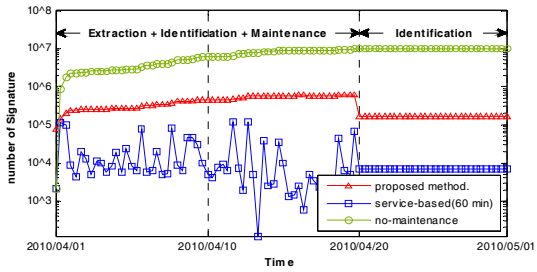
$$\text{accuracy} = \frac{\text{Correctly Identified Traffic}}{\text{Identified Traffic}} \quad (4)$$

본 논문에서 제안한 관리 방법(proposed method)의 분석률은 해당 기간 추출된 모든 시그니쳐를 누적한 방법(no-maintenance)과 비슷한 성능을 보였다. Timeout 값을 이용한 관리 방법 보다 분석율 측면에서 성능이 향상된 것을 확인 할 수 있다. 정확도는 세가지 방법 모두 100%에 가까운 성능을 보였다. (표 3)에서 주목해야 하는 점은 시그니쳐 개수이다. 아무런 관리를 하지 않은 경우 10,421,135개 시그니쳐가 분석에 사용되었지만, 제안한 관리 방법은 비슷한 성능을 가지면서 단지 165,346개 시그니쳐 만을 사용하였다. 비율로 보면 약 1.58% 정도이다. 즉, 본 논문에서 제안한 관리 방법을 적용하면, 적은 양의 시그니쳐를 목록에 유지하면서 분석 성능을 극대화 할 수 있다. 또한 시그니쳐 추출과 관리 없이 분석만 수행한 마지막 10일간의 분석결과에서도 비슷한 성능을 보이는 것을 확인 할 수 있었다.

본 논문에서 제안하는 시그니쳐 관리를 위해서는 시그니쳐로 분석된 트래픽의 통계적 정보와 시

(표 3) 분석 결과 비교

| | Proposed method | Service-based (60 min)[7] | No-Maintenance[8] |
|---|---|---|---|
| Completeness (flow/byte) | 12.40% 30.92% | 8.82% 12.98% | 12.84% 29.89% |
| Accuracy (flow/byte) | 99.69% 99.35% | 99.74% 99.37% | 99.63% 99.56% |
| # of Signature | 165,346 | 6,885 | 10,421,135 |




(그림 11) 시그니쳐 개수 비교

그니쳐의 분석이력을 시그니쳐에 저장해야 하기 때문에 추가적인 저장 공간(시그니쳐 당 72bytes)이 필요하다. 하지만, 제안된 관리 방법에서는 불필요한 시그니쳐를 사전에 판단하여 삭제함으로써 추가적 저장 공간이 상당량 줄어든 것을 확인 할 수 있었다. 본 실험에서 사용한 추가적 저장 공간은 15Mbytes 불가하다.

응용의 종류에 따라 헤더 시그니쳐의 성능을 확인하기 위해 분석 결과의 각 응용 별 precision과 recall 값을 측정하였다.

$$\text{precision}=\frac{\text{True Positive}}{\text{True Positive+False Positive}} \quad (5)$$

$$\text{recall}=\frac{\text{True Positive}}{\text{True Positive+False Negative}} \quad (6)$$

Precision은 분석한 트래픽 중 정확히 분석한 비율을 의미하고 recall은 분석 대상 트래픽 중 정확히 분석한 비율을 의미한다. (표 4)는 분석 된 응용의 precision과 recall을 나타낸다.

(표 4)과 같이 제안된 관리 방법은 다른 두 방법보다 좋은 성능을 보였다. 대부분 응용의 precision은 1에 이다. 즉, 분석된 트래픽은 정확하게 분석되었다. 하지만, recall은 응용의 종류에 따라 상이한 결과를 보인다. Server-client 기반의 응용은 높은 recall값을 가지는 반면, peer-to-peer 기반의 응용은 낮은 recall값을 가진다. 응용의 서버 정보를 사용하여 시그니쳐를 생성하기 때문에 정답지 트래픽에 나타나지 않는peer-to-peer 응용의 peer 정보는 시그니쳐로 추출되지 못한다.

(그림 11)은 실험 기간 동안 변화 하는 시그니쳐의 개수를 로그 스케일로 표현한 것이다. 아무런 관리 방법을 적용하지 않았을 경우, 시그니쳐의 개수가 지속적으로 증가하였다. 그리고 timeout 값을 이용하여 시그니쳐를 관리 하였을 경우, 시그니쳐의 추출 대상이 되는 정답지 트래픽의 양에 따라 급격히 변화하는 것을 확인 할 수 있었다. 본 논문에서 제안한 관리 방법은 일정 양의 개수를 유지하는 것을 확인 할 수 있다.

(표 5)는 각 관리 방법 마다 추출된 시그니쳐의 응용 구성을 보여준다. 시그니쳐 관리를 하지 않은 경우 시그니쳐의 대부분(99%이상)이 비정상 트래픽과 P2P응용에서 추출되었다. 하지만, 본 논문에서 제안한 관리 방법을 적용한 결과, 약 80%만이 비정상 트래픽과 P2P 응용에 관련된 시그니쳐이다. 많은 헤더 정보를 포함하는 응용의 특성을 감안하였을 때, 시그니쳐가 잘 관리되고 있음을 확인할 수 있다.

## 5. 결론 및 향후 연구

본 논문에서는 헤더 시그니쳐 기반 인터넷 응용 트래픽 분석 방법론 중 관리 방법론을 제시 하였다. 헤더 시그니쳐는 특정 응용에서만 사용되는 서버의 3-tuple {IP address, port number, transport layer protocol (TCP/UDP)}이다. 분석 시스템은 시그니쳐

(표 4) 분석 결과(precision, recall)

| Application (type) | Proposed method | | Service-based(60 min) | | No-maintenance | |
|---|---|---|---|---|---|---|
| | Precision (flow/byte) | Recall (flow/byte) | Precision (flow/byte) | Recall (flow/byte) | Precision (flow/byte) | Recall (flow/byte) |
| Internet explorer (Web Browser, SC) | 1.00/1.00 | 0.97/0.90 | 1.00/1.00 | 0.77/0.63 | 1.00/1.00 | 0.95/0.85 |
| Nateon (instance messenger, SC+P2P) | 0.99/0.98 | 0.97/0.08 | 0.97/0.97 | 0.50/0.04 | 0.98/0.99 | 0.95/0.19 |
| Skype (instance messenger, P2P) | 1.00/1.00 | 0.04/0.03 | 1.00/1.00 | 0.02/0.01 | 1.00/1.00 | 0.07/0.05 |
| Melon (music broadcasting, SC+P2P) | 1.00/1.00 | 0.04/0.23 | 1.00/1.00 | 0.02/0.09 | 1.00/1.00 | 0.03/0.17 |
| Bugs music (music broadcasting, SC+P2P) | 0.74/1.00 | 0.21/0.21 | 0.77/0.99 | 0.11/0.02 | 0.84/1.00 | 0.25/0.30 |
| Kdisk (file hosting service, SC) | 0.70/0.52 | 1.00/1.00 | 0.66/0.56 | 0.64/0.88 | 0.64/0.52 | 0.73/0.96 |
| Torrent (file sharing, P2P) | 1.00/1.00 | 0.02/0.04 | 1.00/1.00 | 0.01/0.00 | 1.00/1.00 | 0.03/0.05 |
| Fileguri (file sharing, P2P) | 1.00/1.00 | 0.00/0.00 | 1.00/0.98 | 0.00/0.00 | 1.00/1.00 | 0.01/0.01 |
| Avast (anti-virus, SC) | 1.00/1.00 | 0.94/0.92 | 1.00/1.00 | 0.61/0.40 | 1.00/1.00 | 0.89/0.87 |

P2P: peer-to-peer SC: server-client

(표 5) 시그니쳐 구성 비교

| Rank | Proposed method | | Service-based(60 min) | | No-maintenance | |
|---|---|---|---|---|---|---|
| | Application (type) | Count Ratio | Application (type) | Count Ratio | Application (type) | Count Ratio |
| 1 | Torrent (file sharing, P2P) | 126,791 76.68% | Torrent (file sharing, P2P) | 6,419 93.23% | NetBIOS-SMB (abnormal, P2P) | 8,857,806 85.00% |
| 2 | Internet explorer (Web Browser, SC) | 22,515 13.62% | Internet explorer (Web Browser, SC) | 356 5.17% | Torrent (file sharing, P2P) | 1,324,381 12.71% |
| 3 | Donkey (file sharing, P2P) | 10,591 6.41% | Nateon (instance messenger, SC+P2P) | 63 0.92% | Donkey (file sharing, P2P) | 122,240 1.17% |
| 4 | NetBIOS-SMB (abnormal, P2P) | 1,487 0.90% | Avast (anti-virus, SC) | 14 0.20% | Fileguri (file sharing, P2P) | 77,111 0.74% |
| 5 | Skype (instance messenger, P2P) | 670 0.41% | NetBIOS-SMB (abnormal, P2P) | 6 0.09% | Internet explorer (Web Browser, SC) | 27,904 0.27% |
| Total | - | 165,346 100.00% | - | 6,885 100.00% | | 10,421,135 |

P2P: peer-to-peer SC: server-client

생성 모듈, 트래픽 분석 모듈, 시그니쳐 관리 모듈로 구성된다. 급격히 늘어나는 헤더 시그니쳐 중 분석 성능을 극대화 시키는 최적의 시그니쳐를 관리하기 위해 시그니쳐로 분석된 트래픽의 특성과 시그니쳐의 분석이력을 사용하는 4가지 세부 관리 방법을 제안하였다.

제안한 관리 방법의 타당성을 증명하기 위해 학내망에 시스템을 설치하여 성능을 분석한 결과, 최

적의 시그니쳐로 잘 관리하는 것을 확인 할 수 있었다. 헤더 시그니쳐는 peer-to-peer 기반의 응용보다는server-client 기반의 응용에서 좋은 성능을 보였다. 비록 몇몇 응용의 모든 트래픽을 분석하지는 못하였지만, 빠르고 정확하게 트래픽을 분석 할 수 있었다. 따라서 여러 분석 방법을 결합시킨 멀티레벨 분석 방법에 첫 분석 모듈로 사용한다면 분석 시간 측면에서 큰 성능 향상이 기대된다.

앞으로 본 논문에서 제안한 bunch에 대한 자세한 연구와 여러 분석 방법을 결합시킨 멀티레벨 분석 방법을 연구 할 계획이다.

## 참 고 문 헌


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

[2] S. Sen, J. Wang, “Analyzing peer-to-peer traffic across large networks”, Internet Measurement Conference (IMC), Proc. Of the 2nd ACM SIGCOMM Workshop on Internet measurement, pp. 137-150, 2002.

[3] Internet Assigned Numbers Authority list, http://www.iana.org/assignments /port-numbers

[4] A. Moore, K. Papagiannaki, “Toward the Accurate Identification of Network Applications,” Proc. PAM 2005, Boston, USA, 2005.

[5] F. Gringoli, L. Salgarelli, M. Dusi, N. Cascarano, F. Risso, K. Claffy, “GT: picking up the truth from the ground for Internet traffic,” ACM SIGCOMM Computer Communication Review, 39(4), Oct. 2009.

[6] T. Karagiannis, A.Broido, M. Faloutsos, and kc claffy.Transport layer identification of P2P traffic. InACM/SIGCOMM IMC, 2004.

[7] M. Baldi, A. Baldini, N. Cascarano, and F. Risso, “Service-based traffic classification: Principles and validation”, Proc. of the IEEE 2009 Sarnoff Symposium, Princeton, NJ, USA, Mar. 2009.

[8] 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 the Asia-Pacific Network Operations and Management Symposium(APNOMS) 2009, LNCS5787, Jeju, Korea, Sep. 23-25, pp. 21-30, 2009.

[9] V. Carela-Español, P. Barlet-Ros, M. Solé-Simó, A. Dainotti, W. de Donato, and A. Pescapé, “K-Dimensional Trees for Continuous Traffic Classification,” in Traffic Monitoring and Analysis: Second International Workshop, TMA 2010, Zurich, Switzerland, pp. 141, 2010.

[10] Byung-Chul Park, Young J. Won, Myung-Sup Kim, James W. Hong, “Towards Automated Application Signature Generation for Traffic Identification,” Proc. of the IEEE/IFIP Network Operations and Management Symposium (NOMS) 2008, Salvador, Bahia, Brazil, Apri. 7-11, pp. 160-167, 2008.

[11] Cisco, NetFlow Services and Applications, White Paper, http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.html.

[12] Chen, B.C., Yegneswaran, V., Barford, P., Ramakrishnan, R., “Toward a Query Language for Network Attack Data”, 22nd International Conference on Data Engineering Workshops (ICDEW’06), pp. 28-36. IEEE Press, New York, 2006

[13] Bittorrent on http://www.bittorrent.com/

[14] Fileguri on http://www.fileguri.com/

[15] Internet Explorer on http://windows.microsoft.com/ko-KR/internet-explorer/downloads/ie

[16] Nateon on http://nateonweb.nate.com/
[17] K. Xu, Z. Zhang, and S. Bhattacharya, “Profiling Internet Backbone Traffic: Behavior Models and Applications”, ACM SIGCOMM, pp. 169-180, 2005.
[18] Lan, K and Heidemann, J, “A measurement study of correlations of internet flow characteristics”, Elsevier Computer Networks, 50(1), pp. 46-62, 2006.

## ◐ 저 자 소 개 ◑

**윤 성 호**
2009년 고려대학교 컴퓨터정보학과 졸업(학사)
2011년 고려대학교 대학원 컴퓨터정보학과 졸업(석사)
2011년~현재 고려대학교 대학원 컴퓨터정보학과 박사과정
관심분야 : 네트워크 관리 및 보안, 트래픽 모니터링 및 분석.
E-mail : sungho_yoon@korea.ac.kr

**김 명 섭**
1998년 포항공과대학교 전자계산학과 졸업(학사)
2000년 포항공과대학교 컴퓨터공학과 졸업(석사)
2004년 포항공과대학교 컴퓨터공학과 졸업(박사)
2006년 Post-Doc. Dept. of ECE, Univ. og Totonto, Canada
2006년~현재 고려대학교 컴퓨터정보학과 부교수
관심분야 : 네트워크 관리 및 보안, 트래픽 모니터링 및 분석, 멀티미디어 네트워크
E-mail : tmskim@korea.ac.kr