JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 22, No. 5: 794~804, May 2018

패킷 페이로드 내 특정 패턴 탐지 알고리즘들의 성능 분석에 관한 연구

정구현[1] · 이봉환[1] · 양동민[2*]

Performance Analysis of Detection Algorithms for the Specific Pattern in Packet Payloads

Ku-Hyun Jung[1] · Bong-Hwan Lee[1] · Dongmin Yang[2*]

[1]Department of Electronics, Information and Communications Engineering, Daejeon University, 62, Daehak-ro, Dong-gu, Daejeon, 34520, Republic of Korea
[2*]Graduate School of Archives and Records Management, Chonbuk National University, 567, Baekje-daero, Jeonju-si, Jeollabuk-do, 54896, Republic of Korea

요 약

컴퓨터에서 실행되는 다양한 응용들은 네트워크를 통해 패킷 형태로 정보를 전달하며 대부분의 패킷들은 TCP/IP 또는 UDP/IP 프로토콜을 따른다. 기업 및 기관의 네트워크 관리 담당자는 네트워크 트래픽 측정 및 감시, 네트워크 보안 등을 위해서 네트워크를 통해 전달되는 패킷들을 지속적으로 관리할 수 있어야 한다. 본 논문에서는 실제 전달되는 데이터를 면밀히 조사하는 DPI(Deep Packet Inspection)에서 페이로드의 특정 패턴을 검색하는 패킷 페이로드 분석 알고리즘들의 성능 분석하는 것을 목적으로 하고 있다. 페이로드를 조사하는 가장 기본적인 과정은 특정 패턴을 페이로드에서 신속하게 검색하는 것이다. 본 논문에서는 페이로드에 특정 패턴이 존재하는 경우, 그 패턴을 검출할 수 있는 여러 알고리즘들을 소개하고, 세 가지 관점에서 수학적으로 성능을 분석하고, 응용프로그램의 목적에 적합한 적용 방안을 제시한다.

ABSTRACT

Various applications running in computers exchange information in the form of packets through the network. Most packets are formatted into UDP/IP or TCP/IP standard. Network management administrators of enterprises and organizations should be able to monitor and manage packets transmitted over the network for Internet traffic measurement & monitoring, network security, and so on. The goal of this paper is to analyze the performance of several algorithms which closely examine and analyze payloads in a DPI(Deep Packet Inspection) system. The main procedure of packet payload analysis is to quickly search for a specific pattern in a payload. In this paper, we introduce several algorithms which detect a specific pattern in payloads, analyze the performance of them from three perspectives, and suggest an application method suitable for requirements of a given DPI system.

키워드 : 심층 패킷 분석, 패킷 페이로드, 수학적 성능 분석, 조각화, 세분화

Keyword : DPI(Deep Packet Inspection), packet payload, mathematical performance analysis, fragmentation, segmentation

Received 12 February 2018, **Revised** 22 February 2018, **Accepted** 23 April 2018

*** Corresponding Author** Dongmin Yang(E-mail:dmyang@jbnu.ac.kr, Tel:+82-63-270-3249), Graduate School of Archives and Records Management, Chonbuk National University,567, Baekje-daero, Jeonju-si, Jeollabuk-do, 54896, Republic of Korea

Open Access http://dx.doi.org/10.6109/jkiice.2018.22.5.794 pISSN:2234-4772

Ⅰ. 서 론

인터넷에 광통신 또는 기가비트 통신이 도입되어 높은 전송률로 네트워크를 통해 데이터 교환이 이루어지면서 수많은 응용 프로그램들이 등장하고 있다. 서로 다른 컴퓨터에 실행하고 있는 응용 프로그램들이 데이터를 교환할 때는, 전체 데이터를 한꺼번에 보내지 않고 일정한 크기로 분할해서 전송한다. 일정한 크기의 데이터는 일반적으로 패킷이라고 불리고 있으며, 패킷은 헤더(header)와 페이로드(payload)로 이루어져 있다.

기업 및 기관의 네트워크 관리 담당자는 네트워크 트래픽 감시를 통한 사용자 행동 분석, 네트워크 보안 등을 위해서 데이터를 전달되는 패킷들을 지속적으로 관리할 수 있어야 한다[1-7]. 네트워크 트래픽을 관리하는 방법은 주로 TCP/UDP 및 IP 프로토콜의 헤더 정보(포트번호, IP 주소, 프로토콜 등)를 검색하는 패킷 헤더 기반 분석 방법, 패킷의 통계 정보(패킷 크기, 전송 방향, 발생 시간, 지속 시간, 빈도수 등)를 활용하는 패킷 통계 기반 분석 방법, 패킷 페이로드의 실제 내용을 조사하는 패킷 페이로드 기반 분석 방법이 대표적이다.

첫 번째, 패킷 헤더 기반 분석 방법은 가장 많이 활용되고 있는 방법으로, OSI 7 계층(OSI 7 Layer) 중 전송계층(Transport Layer) 프로토콜인 TCP와 UDP 의 헤더와 네트워크 계층(Network Layer) 프로토콜인 IP의 헤더 정보를 기반으로 트래픽을 분석한다. 플로우(flow)를 정의하는 기본 단위인 5-tuple(source port, destination port, source ip address, destination ip address, protocol)을 기준으로 트래픽을 분류하고 헤더 정보를 분석한다. 헤더만 분석하기 때문에 속도가 빠르다는 장점이 있지만 실제 데이터를 담고 있는 페이로드를 분석하지 않기 때문에 구체성과 정확성이 부족하다. 두 번째, 패킷 통계 기반 분석 방법은 패킷 크기, 패킷간의 발생시간, 윈도우 크기 등의 플로우의 통계 정보를 기반으로 분석한다. 이 방법은 패킷의 페이로드 정보를 분석하지 않기 때문에 분류 속도가 빠르고, 페이로드 정보를 분석하지 않아 암호화된 트래픽에 적합한 방법론이라는 장점이 있다. 하지만 통계 정보를 생성하기 위해서는 플로우가 끝날 때까지 기다려야 하며, 분류 범위가 넓고 정확하지 않다는 단점이 있다[2]. 세 번째, 패킷 페이로드 기반 분석 방법은 응용 프로그램의 패킷 페이로드를 분석한다. 실제 전달되는 데이터를 기반으로 분석하기 때문에 정확도가 매우 높다는 결과를 보여 주지만 정보 추출 과정이 어렵고 시간이 오래 걸리며 페이로드를 저장하는데 드는 유지 및 관리 비용이 크기 때문에 상당히 제한적으로만 사용되었다[5, 6]. 이런 단점은 컴퓨팅 성능 향상 및 하드웨어 가격 하락으로 패킷 페이로드 자체를 분석하는 방법에 관심이 높아지고 있다. 패킷 페이로드를 분석하는 것은 메타데이터 또는 헤더 정보와 같은 부수적인 데이터가 아니라 실제 내용을 조사하기 때문에 가장 확실한 방법이기 때문이다. 본격적으로, 패킷 페이로드 기반 분석 방법을 활용하여 네트워크를 관리하기 위해서는 지금까지 제안되어 온 패킷 분석방법들에 대해 정확히 분석하여 향후 성능 개선을 위한 토대를 마련하는 것은 중요한 이슈이다. 관련 연구들을 살펴 본 결과, 지금까지 패킷 페이로드 기반 분석 방법들에 대한 정량적인 성능 평가가 없는 것으로 조사되었다. 사물인터넷, 빅데이터, 클라우드 컴퓨팅, 모바일 등의 발전으로 네트워크 전송 데이터의 양도 급격하게 증가될 것이기 때문에 향후 풍부한 HW 및 SW의 자원을 바탕으로 패킷 페이로드 기반 분석 방법을 활용한 DPI 시스템이 활발하게 도입될 것으로 예상된다. 최근 보안을 위해 SSL(Secure Socket Layer)를 HTTP 또는 FTP 등에 적용하여 페이로드가 암호화된 경우에도 기업 또는 기관에 소속되어 있는 중간 네트워크 장비에 프락시 서버를 통해 페이로드 기반 분석이 가능하다. 여러 관점에서 패킷 페이로드 기반 분석 방법에 분석 결과가 도출된다면, 다양한 상황마다 적합한 패킷 페이로드 분석 방법을 도입하는데 기준 지표를 제공할 것으로 기대된다.

본 논문에서는 알고리즘들의 성능을 세 가지 관점에서 수학적으로 분석한다. 첫 번째는 '패턴검출확률'이다. 패턴검출확률은 5-tuple로 구분되는 플로우 내에 특정 패턴이 존재할 경우 그 패턴을 검출하는 확률이다. 대부분 데이터들은 패킷이라는 작은 단위의 조각으로 나뉘어져서 전달된다. 그리고 패킷에 담을 수 있는 데이터 크기가 작기 때문에 큰 파일 또는 대용량 콘텐츠를 전송할 때에는 조각화(Fragmentation)이나 세분화(Segmentation)로 패킷 크기만큼 분리되어 전송된다. 이 때 그 특정 패턴이 두 개의 패킷에 분리되어 전송된다면, 하나의 패킷 페이로드마다 패턴을 조사하는 방식은 검출에 실패할 것이다. 특정 패턴을 검출여부는 DPI의 효용성을 결정짓는 중요한 평가요소로 반드시 분석되어야 한다. 두 번째는 '평균패턴검출시간'이다. 평균

패턴검출시간은 5-tuple로 구분되는 플로우에 특정 패턴이 존재할 경우 그 패턴을 검출하는데 걸리는 평균 시간을 의미한다. 패턴검출시간은 DPI가 실시간 분석을 제공하는지 여부를 판단하는 기준으로 매우 중요하지만, 이 값은 컴퓨터의 성능에 의해서 크게 달라진다. 합리적으로 성능을 분석하기 위해서 패턴검출시간과 직접적인 연관이 있는 패턴 비교 횟수를 이용한다. 특정 패턴을 한 번 비교하는 횟수를 1로 하고, 각 알고리즘이 플로우를 조사하면서 특정 패턴을 검출하는데 까지 패턴을 비교하는 총 횟수를 분석한다. 특정 패턴은 플로우에 정해져 있는 위치에 존재하지 않고 골고루 분산되어 있다고 가정하여 패턴검출시간의 평균값을 계산한다. 세 번째는 '패킷검색완료시간'이다. 패킷검색완료시간은 특정 패턴을 플로우의 처음부터 끝까지 비교하는데 걸리는 총시간으로, 패킷 페이로드 기반 분석 방식의 전체적인 성능을 알 수 있다. 평균패턴검출시간에서와 마찬가지로 특정 패턴의 한 번 비교 횟수를 1로 하고, 플로우의 처음부터 끝까지 특정 패턴을 비교하는 총횟수로 분석한다. 그리고 이를 바탕으로 시스템 환경 또는 요구사항에 따라 어떠한 패킷 페이로드 기반 분석 알고리즘이 적용되어야 하는지에 대해 제안한다.

2장에서는 대표 알고리즘들을 소개한다. 3장에서는 세 가지 관점에서의 수학적인 성능 분석을 통해 비교하고 시스템 환경과 요구사항에 따라 적용되어야 하는 패킷 페이로드 기반 분석 방법을 제안한다. 4장에서는 결론 및 향후 연구에 대해서 서술한다.

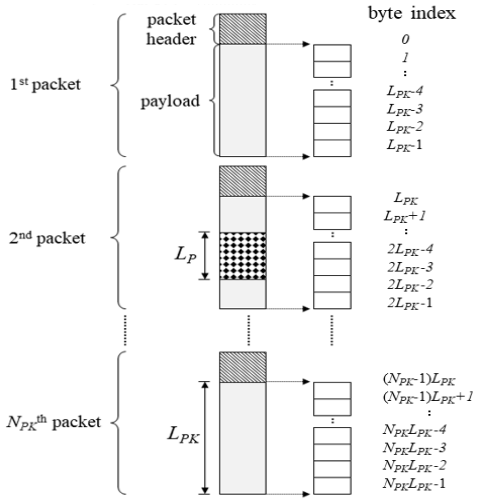

Fig. 1 An example of a flow consisting of multiple packets, and related notations

Ⅱ. 패킷 페이로드 기반 분석 방법

패킷 페이로드 기반 분석 방법의 수학적 분석을 위한 기호, 용어 및 환경 등에 대해서 설명한다.

그림 1은 여러 개의 패킷들로 구성되어 있는 플로우와 관련 기호들을 보여 준다. 하나의 플로우는 동일한 5-tuple을 가진 N_{PK} 개의 패킷들로 구성되어 있다. 각 패킷은 헤더와 페이로드로 이루어져 있으며, 패킷 페이로드 기반 분석 방법에서는 헤더를 제거하고 페이로드만 분석한다. 패킷의 페이로드의 최대 길이는 N_{PK} 바이트이며, 플로우를 구성하는 모든 패킷들은 N_{PK} 바이트의 페이로드를 포함하고 있다고 가정한다. 첫 번째 패킷(1st packet) 페이로드의 첫 번째 바이트를 '0'으로 하고, 각 바이트마다 1만큼 증가시키면서 인덱스를 부여한다. 마지막 패킷(N_{PK}th packet) 페이로드의 마지막 바이트의 인덱스는 N_{PK}*L_{PK}-1이다. 패킷 페이로드 기반 분석 방식에서 찾고자 하는 특정 패턴은 패킷의 페이로드 내에 포함되어 있으며, 그 길이는 L_P 바이트이다.

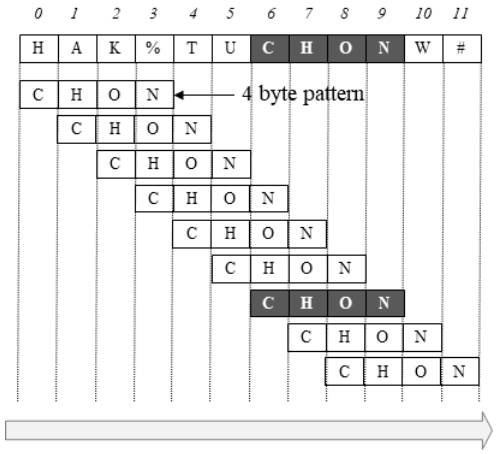

Fig. 2 Sliding windows pattern matching

그림 2는 페이로드를 한 바이트씩 시프트하여 "CHON"이라는 4바이트 비트패턴을 검색하는 슬라이딩 윈도우 방식의 패턴 매칭 기법이다. 여기서 L_{PK}는 12바이트 이고 L_P는 4바이트이다. 조각화나 세분화의 상황이 존재할 때, 특정 패턴을 검색하는 것에 초점을 맞추고 있다. 본 논문에서는 TCAM (Tenary Contents Addressable Memory)[8], GPU(Graphics Processing Unit) 등을 하드웨어 활용하거나 효과적인 FSM(Finite State Machine)

을 적용하여 소프트웨어적으로 패턴 매칭 속도를 높이는 것을 배제한 가장 기본적인 슬라이딩 윈도우 패턴 매칭 기법을 가정한다.

패킷-1, 패킷-2, 패킷-(1+a) 분석 방식은 패킷이 수신될 때마다 분석할 수 있으며, 패킷-1, 패킷-2, 패킷-(1+a) 순으로 개선되었다. 플로우 분석 방식은 모두 패킷들을 저장하고 페이로드를 연결하여 분석하는 방식으로 DPI가 본격적으로 등장하기 전 네트워크 패킷 분석 초창기에 사용되는 방식이다. 다음 2.1~2.4 장에서 이 4가지 분석 방법에 대해서 구체적으로 설명한다.

2.1. 패킷-1 분석 방식 [4,6,8]

패킷-1 분석 방식은 패킷이 도착하면 헤더를 제거하고 패킷 페이로드의 첫 번째 바이트부터 마지막 바이트까지 슬라이딩 윈도우 방식으로 패턴 매칭을 수행하면서 특정 패턴을 검출한다.

기가비트 네트워크 등이 도입되면서 데이터 전송이 고속화되었다. 수신되는 패킷들의 페이로드를 소프트웨어적으로 분석하기에는, 엄청난 부하가 발생하였고 이로 인해 심각한 속도 저하가 발생하였다. 또한, 하드웨어 자원의 제약으로 고속 패턴 매칭에 어려움을 겪고 있다. 그래서 대부분 분석 방식들은 조각화나 세분화를 고려하지 않고 하나의 패킷 페이로드를 대상으로 성능을 높이는 것에만 집중하였다[4, 6, 8]. 그래서 패킷이 NIC(Network Interface Card)로 들어올 때마다 분석하는 패킷-1 분석 방식이 사용되었다.

각 패킷의 페이로드의 길이가 L_{PK} 바이트이고, 특정 패턴의 길이가 L_P 바이트이라면, 패킷이 도착할 때마다 $(L_{PK} - L_P+1)$번의 비교를 수행한다. 이러한 과정은 N_{PK} 개 패킷을 모두 수신할 때까지 계속된다.

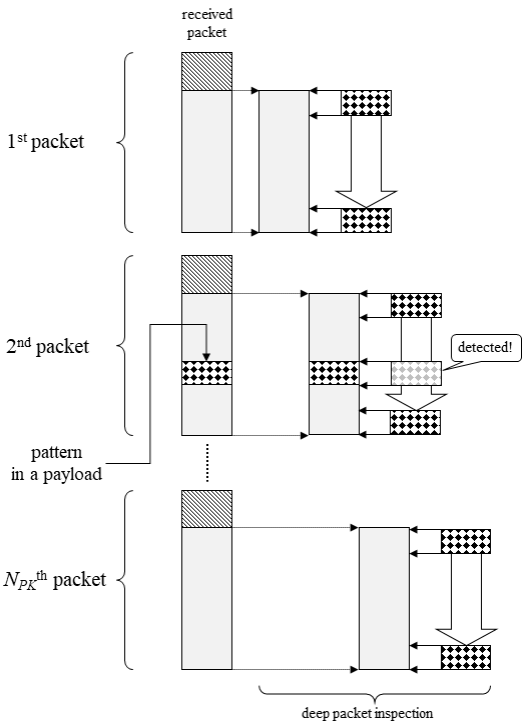

Fig. 3 Procedure of packet-1 analysis

그림 3은 패킷-1 방식이 동작하는 예제로 두 번째 패킷에 특정 패턴이 존재하는 경우이다. 패킷이 도착할 때마다 헤더를 제외한 패킷의 페이로드를 슬라이딩 윈도우 방식으로 패턴매칭하며 두 번째 패킷의 페이로드에 존재하는 특정 패턴을 발견하는 것을 보여 준다.

패킷-1 방식은 하나의 페이로드 내에 특정 패턴이 존재한다면 100% 검출할 수 있다. 그러나 두 개의 페이로드에 분리되어 전송되는 경우에는 검출에 실패한다.

그림 4는 패킷-1 방식이 특정 패턴 검출하는데 실패하는 경우를 보여 준다. 원본 데이터(original data)은 하나의 패킷 페이로드에 탑재할 수 없는 경우 여러 개의 페이로드로 나누고 각 페이로드마다 패킷 헤더를 붙여 전송한다. 이 때 원본 데이터에 있는 특정 패턴이 두 개의 패킷으로 나뉘어져 전송될 확률이 존재한다. 그림 4에서처럼 특정 패턴이 두 개의 패킷으로 나뉘어져 전송될 때, 하나의 패킷 페이로드마다 검사하는 패킷-1 방식이 사용된다면 특정 패턴 검출에 실패한다.

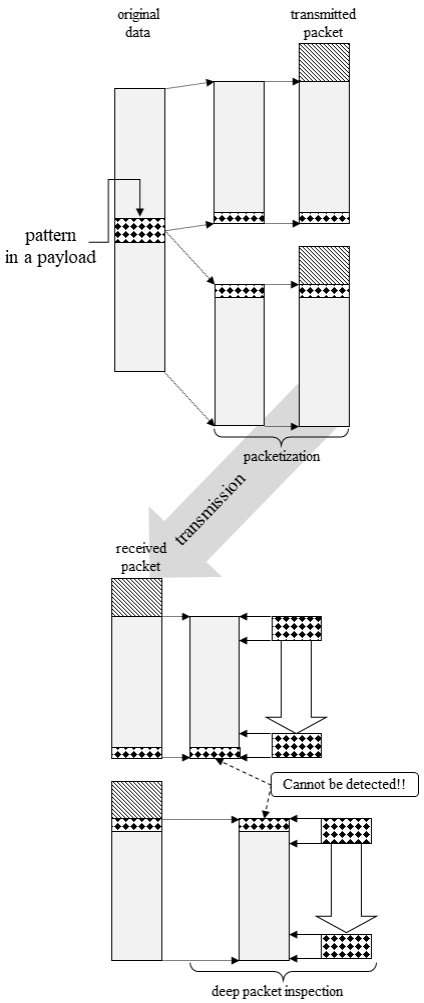

Fig. 4 A example of failure to find a specific pattern

2.2. 패킷-2 분석 방식 [9]

패킷-2 분석 방식은 같은 플로우의 패킷이 도착할 때마다 이전 패킷의 페이로드를 연결하여 특정 패턴을 검출한다.

악성 코드임을 감추기 위해 악성 코드의 특정 패턴을 조각화 또는 세분화하여 여러 개의 패킷으로 나누어서 전송하는 공격 형태가 등장하게 되었다. 이러한 경우에 그 특정 패턴을 검출하기 위해 패킷-1 방식을 사용한다면 검출되지 않을 확률이 있다. 반대로 조각화된 패킷들을 모두 수신한 이후에 한다면, 실시간으로 탐지할 수 없으며, 재조립(Reassembly)을 위해서 얼마나 패킷 버퍼의 크기를 설정해야 하는지 예상이 어렵고, 버퍼 오버플로우(Overflow)가 발생할 수도 있다. 이러한 단점을 보완하기 위해서 패킷-2 분석 방식이 제안되었다. 바로 직전에 수신한 패킷을 패킷 버퍼에 저장하고 있다가 새로운 패킷이 수신되면 두 패킷의 페이로드를 연결하여 패턴 매칭을 수행한다. 특정 패턴이 존재하면 100% 검출할 수 있으며 패턴 매칭을 수행하는데 걸리는 시간을 단축시켜 고속의 침입탐지가 가능하다.

1^{st} packet 이 도착하면 버퍼에 저장하고 있다가 2^{nd} packet 이 도착하면 1^{st} packet와 2^{nd} packet의 페이로드를 연결하여 특정 패턴 검출을 수행한다. 3^{rd} packet이 도착하면 이전 패킷인 2^{nd} packet의 페이로드와 연결하여 조사한다. 정리하면, k^{th} packet (k=2, ..., N_{PK})을 수신하였을 경우 $(k\text{-}1)^{th}$ packet의 페이로드와 k^{th} packet의 페이로드를 연결하여 특정 패턴 검출을 수행한다.

패킷의 페이로드의 길이와 특정 패턴의 길이가 각각 L_{PK} 바이트와 L_P 바이트라면, 2^{nd} packet부터 최대 $N_{PK}{}^{th}$ packet을 수신할 때까지 패킷마다 $(2L_{PK} - L_P+1)$번의 비교를 수행한다. 패킷-1 분석 방식보다 약 2배의 비교 횟수를 수행하지만 특정 패턴이 두 개의 페이로드에 분리되어 전달되던지 하나의 페이로드에 전달되던지 상관없이 100% 특정 패턴을 검출할 수 있다.

그림 5는 특정 패턴이 2^{nd} packet에 포함되어 있을 때 패킷-2 분석 방식이 수행되는 과정을 보여 준다. 1^{st} packet과 2^{nd} packet이 도착하면 각각의 헤더를 제거하고 2개의 페이로드를 연결하여 특정 패턴에 대한 패턴 매칭을 수행한다. 2^{nd} packet 패킷을 버퍼에 잠시 보관하였다가 3^{rd} packet이 도착하면, 2^{nd} packet의 페이로드와 3^{rd} packet의 페이로드를 연결하여 패턴매칭을 수행한다. 이러한 과정을 $N_{PK}{}^{th}$ packet이 도착할 때까지 계속 수행한다.

패킷-2 방식은 특정 패턴이 존재할 경우에 100% 검출할 수 있지만, 그림 5에서 확인할 수 있듯이 두 번 검출하게 된다. 이러한 현상은 패턴 매칭 시 검색 대상 페이로드마다 바이트 오프셋(byte offset)을 조정하는 방식 등 소프트웨어적으로 해결할 수 있다. 그러나 본 논문에서는 성능 분석할 때에 소프트웨어적인 영향의 최소화를 위해서 각 방식에서 슬라이딩 윈도우 방식으로 패턴 매칭을 수행하는 것 이외에 소프트웨어적 수정이나 개선은 배제한다.

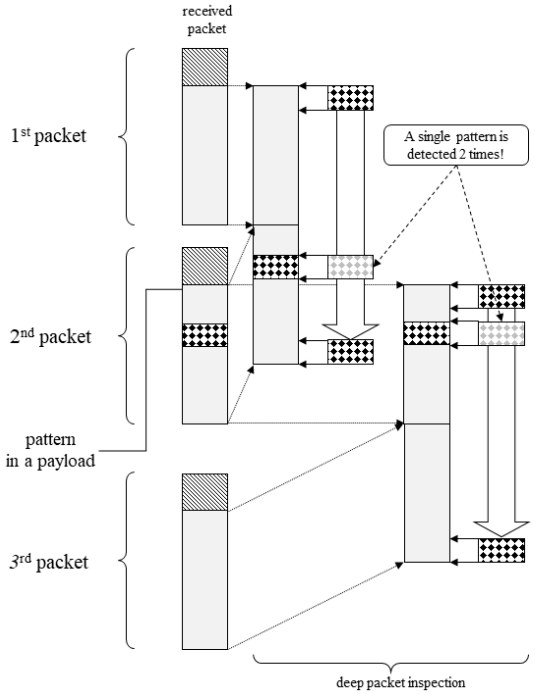

Fig. 5 Procedure of packet-2 analysis

2.3. 패킷-(1+α) 분석 방식 [9]

패킷-(1+*a*) 분석 방식은 이전 페이로드 일부를 버퍼에 보관하고 있다가, 같은 플로우의 새로운 패킷이 수신되면 이전 페이로드 일부와 새로운 패킷 페이로드를 연결하여 패턴 매칭을 수행하는 방식이다.

패킷-2방식과 마찬가지로 특정패턴이 두 개의 패킷에 분리되어 전송하는 경우에도 100% 검출을 할 수 있다. 패킷-2 분석 방식보다 버퍼에 복사되는 데이터 크기를 줄이기 위해 패킷 전체를 저장하지 않고, 페이로드에서 특정 패턴의 크기만 저장한다[9]. 버퍼의 부하를 줄일 수 있으며 패킷-2 분석 방식에처럼 똑같은 검사를 중복해서 하는 경우가 없다.

1st packet 이 도착하면 페이로드를 헤더와 분리하여 패턴매칭을 수행하고, 1st packet 페이로드의 마지막 L_P 바이트를 버퍼에 저장하고 다음 패킷 수신을 대기한다. 2nd packet 이 도착하면 1st packet의 마지막 L_P 바이트와 2nd packet의 페이로드를 연결하여 윈도우 슬라이딩 방식으로 특정 패턴 검출을 수행하고, 2nd packet 페이로드의 마지막 L_P 바이트를 버퍼에 저장하여 다음 패킷 수신을 대기한다. 이와 같은 과정을 N_{PK}th packet을 수신할 때까지 반복한다. 정리하자면, 1st packet 수신시 페이로드 패턴매칭을 수행하고, 페이로드의 마지막 L_P 바이트를 버퍼에 저장하고 패킷 수신을 대기한다. 이후, kth packet (k=2, ..., N_{PK})을 수신하면 (k-1)th packet의 페이로드의 마지막 L_P 바이트와 kth packet의 페이로드를 연결하여 특정 패턴 검출을 수행한다.

각 패킷 페이로드의 길이와 특정 패턴의 길이가 각각 L_{PK}와 L_P 바이트라면, 1st packet 이 수신시에는 (L_{PK} – L_P+1)번, 2nd packet부터 최대 N_{PK}th packet까지 수신할 때마다 (L_{PK} +1)번의 비교를 수행한다. 패킷-1 분석 방식과는 비교 회수는 비슷하고, 언제든지 특정 패턴을 100% 검출할 수 있다.

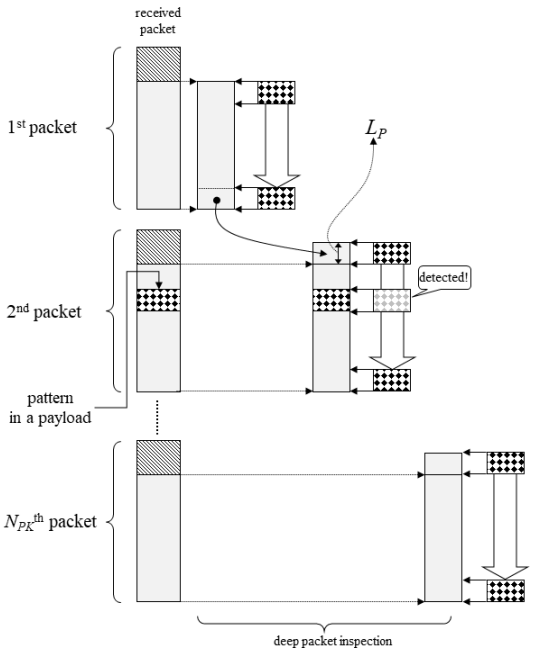

Fig. 6 Procedure of packet-(1+α) analysis

그림 6은 2nd packet에 특정 패턴이 존재하는 경우 패킷-(1+*a*) 분석 방식으로 특정 패턴을 검출하는 수행 과정의 예를 보여 준다. 1st packet이 도착하면 패턴 매칭을 수행하고 1st packet 페이로드의 마지막 L_P 바이트를 버퍼에 보관한다. 같은 플로우의 2nd packet이 도착하면 버퍼에 보관되어 있는 L_P 바이트와 2nd packet 페이로드를 연결하여 패턴 매칭을 수행하고, 2nd packet 페이로드의 마지막 L_P 바이트를 버퍼에 보관한다. 3rd packet이

도착하면 2nd packet도착했을 때와 동일한 과정으로 패턴 매칭을 수행한다.

패킷-(1+α) 분석 방식은 패킷-1 분석 방식이 100%의 패턴검출확률을 제공하지 못하는 점, 패킷-2 분석 방식이 특정 패턴을 2회 검출하는 단점을 보완하였다. 이 방식에서 이전 패킷 페이로드의 LP 바이트를 버퍼에 보관하는 대신에 LP-1 바이트를 보관하면 검색 속도를 소폭 개선할 수 있다.

2.4. 플로우 분석 방식 [10, 11]

플로우 분석 방식은 플로우 내 전체 패킷의 페이로드 전부를 한꺼번에 모아서 분석하는 방법이다.

패킷-1, 패킷-2, 패킷-(1+*a*) 방식은 NIC(Network Interface Card)로 들어온 패킷을 커널에서 분석하거나 응용프로그램에게 전달되기 전 프로토콜 드라이버 계층에서 실시간으로 분석하는 방식이다. 반면, 플로우 분석 방식은 프로토콜 드라이버 계층에 수신된 패킷들을 플로우 별로 저장하거나 응용프로그램까지 전달된 데이터들을 저장해 놓았다가 전체를 한꺼번에 패턴을 분석하는 방법이다. 네트워크 패킷을 분석하기 시작한 초기 방식으로 패킷 페이로드 기반 분석 방식 분석을 위한 기준을 제시할 수 있는 분석 방식이다.

1st packet부터 플로우의 마지막 패킷인 N_{PK}th packet까지 도착하고 난 후에 모든 패킷 페이로드를 연결하여 패턴 매칭을 수행한다. 각 패킷 페이로드의 길이와 특정 패턴의 길이가 각각 L_{PK}와 L_P 바이트이고 플로우가 N_{PK}개의 패킷으로 구성되어 있다면, ($N_{PK}L_{PK}$ - L_P+1)번의 비교를 수행한다.

그림 7은 2nd packet에 특정 패턴이 존재하는 경우 플로우 분석 방식이 수행되는 과정을 보여 준다. 플로우의 마지막 패킷인 N_{PK}th packet이 도착하면 패턴 매칭을 수행하여 특정 패턴을 검출한다.

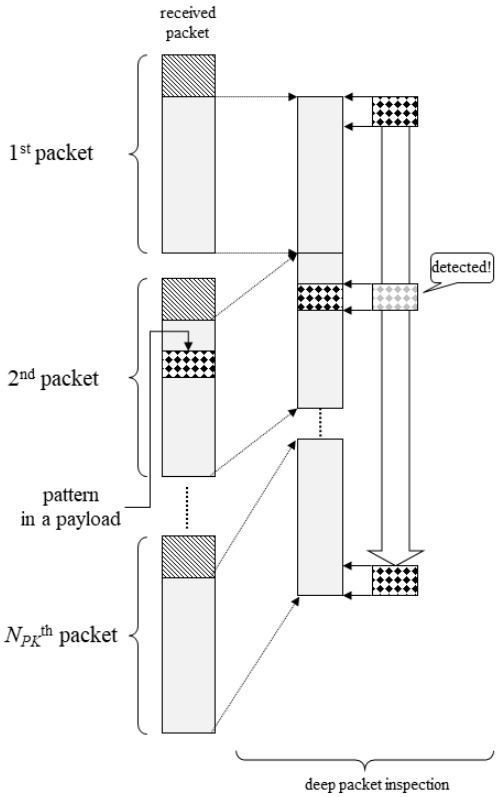

Fig. 7 Procedure of flow analysis

플로우 분석 방식은 100%의 패턴검출확률을 제공하고 특정 패턴이 존재하는 경우 1회만 검출하지만 모든 패킷들이 도착하고 난 후에 작업을 진행할 수 있기 때문에 실시간 검색에는 사용할 수 없다는 단점이 있다. 패킷 페이로드 기반 분석 방식이 실시간성을 요구하지 않는다면 플로우 분석 방법으로 페이로드를 분석하는 것이 합리적이다.

Ⅲ. 분석 알고리즘 성능 평가

본 장에서는 앞에서 소개한 패킷 페이로드 기반 분석 방법들을 3가지 관점에서 수학적으로 분석한다. L_{PK}는 대부분 TCP 표준 환경에서 사용되는 1460 바이트로 고정하였고, L_P와 N_{PK}는 성능 평가 관점에 따라 고정하거나 변경하면서 성능평가를 실시하였다. 각 성능 평가 관점마다 4개의 분석 방식들에 대해 수식을 도출하여 표로 정리하고, 성능 비교를 시각적으로 명확하게 나타

낼 필요한 있는 경우에 수식을 그래프로 나타낸다.

3.1. 패턴검출확률

패턴검출확률은 플로우를 구성하는 패킷의 페이로드에 특정 패턴이 존재할 경우 그 패턴을 검출하는 확률이다. 패킷-1 분석 방식은 그림 4의 경우처럼 패턴검출에 실패할 수 있는 경우가 존재한다. 이러한 경우들을 고려하여 도출된 패킷-1 분석 방식의 패턴검출확률은 표 1의 packet-1에서 처럼 N_{PK}, L_{PK}, L_P의 함수로 나타난다. 패킷-1의 패턴검출확률은 그림 1의 환경에서 특정 패턴이 존재할 수 있는 모든 경우에서 특정 패턴이 k^{th} packet과 $(k+1)^{th}$ packet (k=1, ..., N_{PK}-1)사이에 존재할 경우를 고려하면 계산할 수 있다. 패킷-1 분석 방식이 특정 패턴을 검출 못하는 단점을 보완하여 설계된 패킷-2, 패킷-(1-a), 플로우 분석 방식은 패턴검출확률은 표 1와 같이 모두 1이다.

Table. 1 Probabilities to detect a specific pattern

Method	Probability
packet-1	$1-\frac{(N_{PK}-1)L_P}{N_{PK}L_{PK}-L_P+1}$
packet-2	1
packet-(1+α)	1
flow	1

패킷-1 분석 방식의 패턴검출확률을 수치로 계산하기 위해서 L_P는 10바이트로 가정하고, 패킷의 개수 N_{PK}를 10~10^{20}개로 변경했을 때 대부분 0.99 이상이다. 다른 분석 방식처럼 100% 검출은 못하지만 상당히 높은 확률로 특정 패턴을 검출할 수 있다.

3.2. 평균패턴검출시간

평균패턴검출시간은 플로우 내에 특정 패턴이 존재할 경우 플로우의 첫 번째 패킷(1^{st} packet)을 받은 시점에서부터 특정 패턴을 검출해내는 시점까지의 측정시간의 평균값이다. 패턴검출시간은 네트워크로부터 패킷을 수신하는 시점부터 측정되기 때문에 이 패킷 페이로드 기반 분석 알고리즘을 활용하는 DPI 시스템의 실시간성 제공여부와 직접적으로 연관된다. 모든 패킷이 도착한 후에 검색할 수 있는 플로우 기반 분석 방식은 평균패턴검출시간 관점에서 성능을 비교할 때는 제외한다.

패턴검출시간의 값은 컴퓨터의 성능에 의해서 달라지기 때문에 패턴검출시간과 직접적인 연관이 있는 패턴 비교 횟수로 성능을 평가하였다. 특정 패턴을 한 번 비교하는 횟수를 1로 하고, 각 알고리즘이 특정 패턴을 검출하는데 까지 비교하는 총 횟수를 분석한다.

특정 패턴은 플로우에 골고루 분산되어 있다고 가정한다. 평균값을 구하기 위해서 그림 1의 byte index i (i=0, ..., $N_{PK}L_{PK}$-L_P) 위치에서 특정 패턴이 시작하는 경우마다, 각각의 분석 방식이 byte index 0부터 검색을 시작하여 패턴을 검출할 때까지의 패턴 매칭 횟수를 계산하고 평균을 구한다. 이 때 전체 경우의 수는 특정 패턴이 존재할 수 있는 경우로 ($N_{PK}L_{PK}$-L_P-1) 이다. 패킷-1, 패킷-2, 패킷-(1+a)의 평균패턴검출시간은 표 2과 같이 도출되며 모두 N_{PK}, L_{PK}, L_P의 함수로 표현된다.

Table. 2 Averaged time to find a specific pattern

Method
Averaged time to find a specific pattern(# of times)
packet-1
$\frac{\sum_{k=0}^{L_{PK}-L_P}(k+1)+\sum_{m=1}^{N_{PK}-1}\left(\sum_{k=mL_{PK}}^{(m+1)L_{PK}-L_P}\{(k+1)-m(L_P-1)\}\right)}{N_{PK}L_{PK}-L_P+1}$
packet-2
$\frac{\sum_{k=0}^{2L_{PK}-L_P}(k+1)+\sum_{m=2}^{N_{PK}-1}\left(\sum_{k=mL_{PK}-L_P+1}^{(m+1)L_{PK}-L_P}\{(m-1)(L_{PK}-L_P+1)+k+1)\}\right)}{N_{PK}L_{PK}-L_P+1}$
packet-(1+α)
$\frac{\sum_{m=1}^{N_{PK}-1}\left\{\sum_{k=mL_{PK}-L_P+1}^{(m+1)L_{PK}-L_P}(k+m+1)\right\}}{N_{PK}L_{PK}-L_P+1}$

그림 8은 알고리즘 성능 비교를 명확하게 하기 위해 표 2의 수식의 계산 결과 추이를 보여 준다. 패킷 페이로드의 길이 L_{PK}는 1046바이트로, 패킷의 개수 N_{PK}는 1,000개로 설정하였다. 그림 8에서는 L_P를 5에서 200개로 변화시켰을 때의 평균패턴검출시간을 보여준다. 패킷-1과 패킷-(1+a)의 평균패턴검출시간은 비슷하지만, 패킷-1의 패턴검출확률은 1보다 작다는 단점이 있다. 패킷-2의 시간은 패킷-1 및 패킷-(1+a)의 시간보다 2배 정도 더 걸린다.

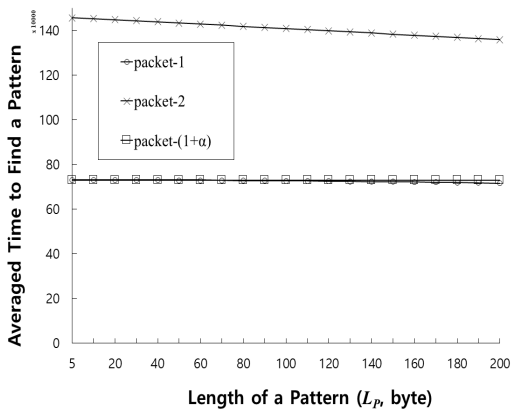

Fig. 8 Averaged time to find a specific pattern of packet-1, packet-2 and packet-α

3.3. 패킷검색완료시간

패킷검색완료시간은 주어진 플로우를 구성하고 있는 패킷들의 페이로드를 처음부터 끝까지 검색을 완료하는데 걸리는 시간이다. 실시간으로 패킷 페이로드들을 검색하는 DPI 시스템 보다는, 패킷들을 계속 수집하다가 문제가 발생하였을 경우 수집된 패킷의 페이로드들을 대상으로 빠르게 분석할 수 있는 오프라인 방식의 DPI 시스템에 적용할 패킷 페이로드 기반 분석 방식을 선정할 때 중요한 기준을 제공할 수 있다. 특정 패턴을 100% 검출할 수 없는 패킷-1 분석은 성능 평가 대상에서 제외한다.

플로우 분석 방식은 페이로드를 모두 연결된 상태에서 byte index 0부터 byte index ($N_{PK}L_{PK}$-L_P)까지 검색한다. 패킷-2와 패킷-(1+a) 분석 방식은 각 패킷마다 페이로드를 분리하고, 바로 전에 수신한 패킷의 전체 또는 일부를 함께 검색한다. 그래서 패킷-2 분석 방식의 경우는 추가적으로 약 $N_{PK}L_{PK}$의 비교 횟수가 발생하여 2배의 시간이 걸리고, 패킷-(1+a) 분석 방식은 N_{PK} 정도의 추가 비교 횟수가 발생하지만 비슷한 수준이다.

Table. 3 Time to complete pattern matching from 1st packet to N_{PK}^{th} packet

Method	Time to complete pattern matching (# of times)
packet-2	$(N_{PK}-1)(2L_{PK}-L_P+1)$
packet-(1+α)	$\{N_{PK}(L_{PK}+1)-L_P\}$
flow	$(N_{PK}L_{PK}-L_P+1)$

패킷-2, 패킷-(1+a), 플로우의 평균패턴검출시간은 표 3과 같이 도출되며, 모두 N_{PK}, L_{PK}, L_P의 함수로 표현된다. 패킷-2의 패턴검색완료시간은 패킷-(1+a)과 패킷-1의 2배 정도 걸린다. 그리고 세 개의 알고리즘 중에서 플로우 분석 방식이 가장 빠른 패킷검색완료시간을 제공한다. 실시간성을 요구하는 DPI 시스템에서 플로우 분석 방식은 도입할 수 없지만, 패킷들을 수집한 이후에 분석하는 DPI 시스템에서는 플로우 분석 방식이 가장 적합하다.

표 4는 세 가지 관점의 성능 비교 결과를 기반으로 패킷 페이로드 기반 분석 방식을 실시간성(realtime), 정확성(accuracy), 속도(offline-speed) 면에서 4단계(best, good, not bad, bad)로 간단히 비교하였다. 비교 분석한 내용을 토대로 3.4.에서 패킷 페이로드 기반 분석 방식의 적용 방안을 도출한다.

Table. 4 performance comparison of packet payload based analysis methods

Method	real time	accuracy	offline -speed
packet-1	best	good	good
packet-2	good	best	not bad
packet-(1+a)	best	best	good
flow	bad	best	best

3.4. 패킷 페이로드 기반 분석 방식 적용 방안

지금까지 DPI 시스템에서 적용 가능한 4가지의 패킷 페이로드 기반 분석 방식에 대한 동작과정을 자세히 소개하고 3가지 관점에서 성능을 수학적으로 분석하였다.

DPI 시스템의 요구사항에 따라 어떤 패킷 페이로드 분석 방식을 도입할지 달라 질 수 있다. DPI시스템의 요구사항은 크게 두 가지로 구분될 수 있다. 첫 번째로 네트워크로 유입되는 패킷들을 실시간으로 분석하는 형태이고, 두 번째로는 일정 기간 동안 패킷을 저장하고 있다가 특정 이벤트가 발생하는 경우 수집된 패킷을 대상으로 분석하는 형태이다. 3.1.~3.3.의 성능 분석결과를 토대로 2가지의 DPI 요구사항에 따라 어떤 분석 방식을 도입할지 제안한다.

실시간성을 요구하는 DPI 시스템에 도입할 수 있는 분석 방식은 패킷-1과 패킷-(1+a)이다. 컴퓨팅 자원이 풍부하지만 그에 따라 네트워크도 고속으로 발전되면서 같은 시간에 처리해야 할 패킷들은 훨씬 더 증가하

고 있다. 그러므로 실시간성 요구하는 DPI시스템을 소프트웨어적으로 구현할 때 패킷-(1+*a*)의 경우에는 중간에 페이로드 일부를 버퍼에 저장하고 다음 패킷의 페이로드와 연결하는 작업들로 인해 시스템에 부하를 줄 수 있다. 작은 수의 플로우를 대상으로 한다면 패킷-(1+*a*)의 방식이 적용 가능하지만, 다수 또는 전수 플로우를 대상으로 한다면 패킷-1(패턴검출확률: 0.99 이상)을 활용하는 것이 바람직하다.

일정 기간 동안 패킷을 저장하고 있다가 추후 패킷들을 분석하는 DPI 시스템에서는 플로우 분석 방식이 가장 적합하다. 패킷들이 유입될 때 플로우의 패킷들을 헤더와 분리하여 페이로드를 별도로 저장한다면 플로우 분석 방식의 효율성을 크게 제고할 수 있다.

Ⅳ. 결론 및 향후 연구 계획

본 논문에서는 페이로드에 특정 패턴이 존재하는 경우, 그 패턴을 검출하는 4가지 패킷 페이로드 기반 분석 방식에 대해서 자세히 설명하고, 세 가지 관점에서 성능을 수학적으로 분석하였다. 그리고 그 분석 결과를 바탕으로 DPI 시스템의 요구사항에 적합한 적용 방안을 제시하였다.

현재는 본 연구의 분석 결과를 토대로 Snort[12]와 WinPcap(Windows Packet Capture Library)[13]를 활용하여 하나의 DPI 시스템에 실시간성을 제공하는 분석 방식을 도입하였으며, 향후에는 수집된 패킷들의 페이로드를 조사하는 분석 방식을 모두 구현하고 비교 분석할 계획이 있다.

ACKNOWLEDGEMENT

This research was supported by Disaster-Safety Platform Technology Development Program of the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT &Future Planning (No.NRF-2016M3D7A1912703). This paper was supported by research funds for newly appointed professors of Chonbuk National University in 2017.

REFERENCES

[1] S.-C. Seo, N.-Y. Ko, “A traffic analysis of Gigabit Ethernet high-speed network design,” *Journal of the Korea Institute of Information and Communication Engineering*, vol. 6, no. 1, pp.48-54, Feb. 2002.

[2] J.-H. Kim, M.-S Kim, “Research on Traffic Classification based on DNS Packet Analysis,” *Korean Network Operations and Management Review*, vol. 13, no. 2, Oct. 2010

[3] Y.-H. Goo, S.-O. Choi, S.-K. Lee, S.-M. Kim, M.-S. Kim, “A Method for Tracking the Source of Cascading Cyber Attack Traffic Using Network Traffic Analysis,” *The Journal of Korean Institute of Communications and Information Sciences* '16-12, vol. 41 no.12, Dec. 2016.

[4] K.-S. Shim, S.-H. Yoon, M.-S. Kim, “The Payload Signature Management System for Network Management on Real-Time,” *in Proceedings of Korean Information and Communications Society, the summer conference 2015*, Ramada Plaza, Jeju, Jun. 23-25, 2015.

[5] A. Hashmi, H. Berry, O. Temam, and M. Lipasti, “IP traceback based on packet marking and logging,” *in Proceedings of 2005 IEEE International Conference on Communications*, Seoul, South Korea, 2005.

[6] B. K. Kim, S. Y. Yoon, J. T. Oh, and J. S. Jang, “High-Performance Intrusion Detection Technology in FPGA-Based Reconfiguration Hardware,” *ETRI Electronics and Telecommunications Trends*, vol. 22, no. 1, pp. 51-58, Feb. 2007.

[7] Vaddempudi Srinidhi, "Classification of User Behaviour in Mobile Internet", Asia-pacific Journal of Convergent Research Interchange, *Asia-pacific Journal of Convergent Research Interchange*, vol. 2, no. 2, June (2016), pp. 9-18

[8] J.-H. Sung, K.-H. Kim, T.-G. Kwon, B.-T. Kim, “Efficient Contents Filtering Algorithm with TCAM,” *in Proceedings of Joint Conference on Communications and Information* 2005.

[9] B.-H. Chung, S.-H Ryu, J.-D. Lim, Y.-H. Kim, K.-Y Kim, Intrusion detection method in network system, KR100656403B1, 2006.

[10] Y.-H. Goo, K.-S. Shim, S.-H. Lee, Baraka D. Sjia, M.-S. Kim, "Traffic-Classification Method Using the Correlation of the Network Flow," *Journal of Korean Institute of Information Scientists and Engineers*, vol. 44, no. 4, pp. 433-438, Apr. 2017.

[11] Y.-H. Goo, S.-H. Lee, K.-S. Shim, W.-S. Jung, S.-M. Kim, M.-S. Kim, "Multi-demensional Application Traffic Analysis using Flow Characteristic," *in Proceedings of Korean Information and Communications Society Winter*

Conference 2017, High1 Resort, Kangwon, Jan. 18-20, 2017.
[12] Snort [Internet]. Available: https://www.snort.org/.
[13] The industry-standard windows packet capture library [Internet].Available: https://www.winpcap.org/.

정구현(Ku-Hyun Jung)

2018년 대전대학교 정보통신공학과졸업(학사)
※ 관심분야 : 네트워크보안, 빅 데이터, AI 등

이봉환(Bong-Hwan Lee)

1985년 서강대학교 전자공학과졸업(학사)
1987년 연세대학교 대학원 전자공학과 졸업(석사)
1993년 Texas A&M 대학교 대학원 전기 및 컴퓨터공학과 졸업(박사)
현재 대전대학교 전자정보통신공학과 교수
※ 관심분야 : 클라우드 컴퓨팅, 사물인터넷, 네트워크보안 등

양동민(Dongmin Yang)

2000년 POSTECH 컴퓨터공학과(학사)
2003년 POSTECH 컴퓨터공학과(석사)
2009년 9월 ~2011년 9월 삼성전자 Senior Engineer
2011년 POSTECH 컴퓨터공학과(박사)
2011년 9월 ~2017년 9월 대전대학교 정보통신공학과 조교수
2017년 9월 ~ 현재 전북대학교 대학원 기록물 관리학과 조교수
※ 관심분야 : Archives & Records Information Security, IoT, Manet