오픈소스 SW를 이용한 IDS NFV 구축 및 평가

*정경승, *뉘엔트리 투안 힙, *김경백
*전남대학교 전자컴퓨터공학부
*wwwjkscom@naver.com, tuanhiep1232@gmail.com, kyungbaekkim@jnu.ac.kr

Deployment and Assessment of OpenSource SW based IDS NFV

Kyung Seung Jung* Hiep Tuan Nguyen Tri* and Kyungbaek Kim*
*Department of Electronics and Computer Engineering,
Chonnam National University

요 약

NFV(Network Function Virtualization) 기술은 통신사업자들이 사용하고 있는 네트워크 장비 내의 여러 기능들을 분리시켜 소프트웨어적으로 제어 및 관리가 가능하도록 가상화하는 기술이다. 결과적으로 NAT, 방화벽, 침입 탐지, DNS 서비스와 같은 기능은 소프트웨어로 범용 기기들에게 전달 될 수 있다. 또한 통신서비스 설계 방식에 더 많은 유연성을 제공하게 된다. 본 논문에서는 널리 사용되는 오픈소스 기반 침입 탐지 시스템인 Snort를 이용해 침입 탐지 시스템(IDS : Intrusion Detection System)기능을 NFV로 구축하고, 해당 기능의 실용성을 이해하기 위해 IDS NFV 사용 유무에 따른 패킷 전송 지연 시간을 비교한다. 비교 결과, IDS NFV사용 시 최대 약 0.1ms의 패킷 전송 시간 증가를 관찰하였고, 이는 평균 패킷 전송 시간이 큰 광대역 네트워크에 오픈소스 기반 IDS로 해당 시스템을 사용하는 것이 알맞다고 할 수 있다.

1. 서 론

현대의 네트워크는 다양한 종류의 전용 하드웨어 장비들로 구성되어 있다. 새로운 응용 서비스를 제공하려면 이를 위한 전용 장비를 설치해야 하는 경우가 있을 수 있으며 이 경우 공간 부족이나 전력 문제가 발생한다. 또한 트래픽 양이 급격히 증가하거나 감소할 경우 혹은 수명이 다한 장비를 새로운 장비로 교체 시 시간이 많이 소요되며 신속하게 대응하는데 어려움이 있다. 그리고 기술 발전과 응용 서비스에 대한 변화가 급격히 빨라지면서 장비의 수명이 갈수록 짧아지고 있고 이러한 현상은 더욱 가속화될 것이며 이는 IT 사업자에게 새로운 서비스를 이용한 수익 창출을 막는 걸림돌이 될 수가 있다. 위와 같은 문제를 해결하기 위해서 다양한 네트워크 장비들을 고성능 서버, 스토리지와 스위치를 이용하여 구현하여, 비용을 절감하고 효율을 높이며 서비스 대응 및 트래픽 변화 등에 신속하게 대처하고자 하는 것이 네트워크 기능 가상화(이하 NFV : Network Function Virtualization)이다[1].

NFV는 유무선 네트워크 장비에 IT가상화 기술을 적용해 하드웨어와 소프트웨어를 분리하는 기술이다. NFV를 적용하면 표준화된 하드웨어 기반으로 네트워크 기능을 가상화해 제공할 수 있다. 이를 통해 라우팅, 파이어월, 로드 밸런서 및 침입 탐지 시스템과 같은 네트워크 기능들을 소프트웨어로 제공할 수 있게 된다.

항상 NFV와 함께 거론되는 개념으로는 SDN(Software Defined Networking)개념이 있다. 몇몇 IT 전문가들이 이 두 개의 개념을 상반되게 생각하는 경향이 있기는 하지만, 이 두 개의 혁신적인 네트워킹 기술들은 서로 상충되는 관계에 있지 않다. 오히려 이 둘은 상부상조하여 SDN과 NFV가 현대 기업 네트워크와 통신사 인프라에 적용될 가능성이 높다. SDN을 간단하게 설명하자면 SDN 제어 하드웨어가 네트워킹 장치들로 구성된 물리적인 인프라 계층위에 위치하고 있으며, 이를 이용해 오픈플로우(OpenFlow) 같은 제어면 인터페이스를 통해 통신한다는 의미다. 목표는 네트워크를 유연하고 프로그램 가능한 플랫폼으로 변모시켜 자원 활용을 최적화함으로써 더욱 비용 효율적이고 확장 가능하도록 한다는 것이다[2].

문제는 이러한 기술들을 이용하여 사용하였을 때 현재의 하드웨어 기반의 서비스와 상응하는 서비스가 제공될 수 있냐는 것이다. 최근 인포네틱스의 조사에서는 사업자의 82%가 NFV 및 SDN 평가를 시작하고 있는 결과를 확인하였다고 보도하였다. 사업자들 또한 기존의 기술에 상응하는 서비스가 제공되는 것에 확신을 가지기 위함이다. 앞으로 가상 기술의 상용화에 있어서 성능 평가는 꼭 필요하다. 가상 기능의 성능을 평가할 때 4가지 주요한 데이터 영역(입출력 운용과 읽기쓰기 메모리 운용을 포함하는

패킷 처리와 관련된 업무), 제어 영역, 신호 처리, 스토리지 유형의 작업 부하가 있다[3].

본 논문 에서는 이 중의 데이터 영역 테스트 특히 네트워크 패킷 처리에 따른 지연 시간을 통하여 구축된 IDS NFV의 성능을 평가한다. 성능 평가를 위해 Mininet을 통해 설정된 SDN 네트워크 토폴로지 상에서 오픈 소스 기반의 침입 탐지 시스템인 Snort기반의 IDS NFV를 구축하고, 해당 IDS NFV의 사용 여부에 따른 TCP/UDP 트래픽의 전송 지연 시간을 비교하여 해당 시스템의 실용성을 분석하였다.

본 논문의 구성은 다음과 같다. 2장에서는 오픈소스 SW인 Snort기반의 IDS NFV를 SDN상에서 구축하기 위한 배경 지식에 대해서 기술하고, 3장에서는 오픈소스 기반 IDS의 성능측정 및 평가에 관련된 연구를 기술하며, 4장에서는 오픈소스 기반 IDS의 구축 및 평가 결과 및 그 분석에 대하여 기술한다. 5장에서는 결론 및 향후 연구 수행 방향에 대해 기술한다.

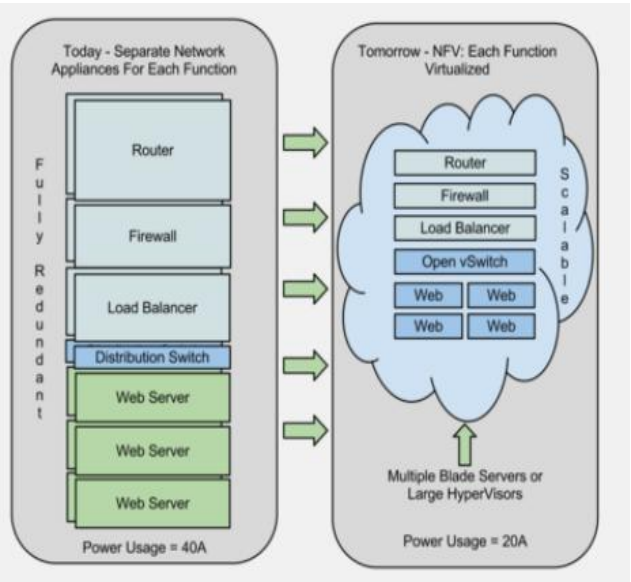

그림1. NFV 개념도[11]

2. NFV, SDN 그리고 Snort

2.1 NFV : Network Function Virtualization

2012년 10월에 독일 다름슈타트에서 열린 SDN World Congress에서 AT&T, BT, 도이치텔레콤 등을 포함한 13개의 통신 사업자들이 NFV(Network Function Virtualization) 결성을 발표하였다. 이는 통신 시장의 주도권을 가져옴으로써 새로운 형태의 네트워크 장비 시장 생태계를 조성하고, 이를 통해 비용 절감 및 효율성 증대 그리고 신속한 서비스대응력 강화 등을 가치로 내건바 있다. 지금까지 이런 형태의 시도들은 많았지만 SDN의 흐름과 같이 벤더들은 저마다 NFV에 최적화된 솔루션을 발표하고, 통신사업자들도 매우 적극적으로 NFV를 검토하고 있기 때문에 최근에 느껴지는 분위기는 이 NFV가 대세로 굳어지는 듯한 분위기 이다. NFV가 추구하는 것은 말 그대로 흩어져있던 네트워크 기능들을 고집적 장비에 몰아넣어 비용 절감 및 효율성 극대화에 목적이 있다. 기존의 네트워크 개념은 각 요소를 하드웨어 단위로 구분하였다. 그래서 개별 단위로 분류하는 경향이 강했다. 하지만 이로 인해 자원의 낭비가 심해졌고, 전력 사용량도 매우 높았다. 이러한 문제를 해결하기 위해 NFV는 개별 성능에 초점을 둔 것이라기보다는 오히려 잘 설계된 아키텍처에 의해 물리적 자원을 최소화하여 전체의 효율성을 향상시키고 시스템의 복잡성을 감소시키는 것에 초점을 두고 있다[6].

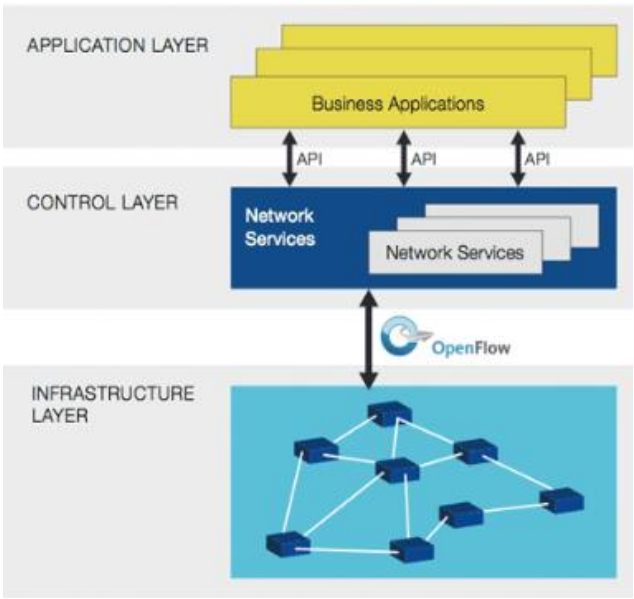

그림2. SDN Architecture [5]

2.2 SDN : Software Defined Networking

SDN이란 스위치와 같은 네트워크 장비의 제어 부분을 데이터 전송 부분과 분리하고, 네트워크 장비의 기능을 정의할 수 있는 오픈 API를 외부에 제공하여 이를 통해 프로그램된 소프트웨어로 다양한 네트워크 경로 설정 및 제어 등을 할 수 있도록 하는 기술이다. 기존 네트워크장비에서는 불가능 했던 소프트웨어로 네트워크를 프로그램하고 정의한다. SDN 기술의 시작은 스탠퍼드 대학의 오픈플로우 기술에서 비롯되었다. 처음 오픈플로우 기술은 초기 네트워크 보안을 위해 다양한 정책을 네트워크에 손쉽게 적용 할 수 있도록, 스위치의 제어 부문을 중앙 집중적인 구조로 분리하고, 플로우 기반으로 네트워크를 관리하는 기술로부터 시작된 것으로 2008년 미국 대학의 캠퍼스에서 기존 인터넷 트래픽에 영향을 주지 않고 구내망의 다양한 실험을 지원하기 위해 인터넷 패킷 플로우 제어를 위한 스위치용 오픈 인터페이스와 이를 위한 간단한 프로토콜 규격만을 정의한 것으로 출발한다. 이후 오픈플로우 기술은 2010년부터 구글의 데이터 센터 연결을 위한 G-스케일 프로젝트에 사용되기 시작하여 2012년 4월에는 글로벌 IDC 연결에 운영되는 모든 내부 네트워크에 오픈플로우를 사용할 것이라고 발표했다. 2011년에는 구글, 페이스북, 야후와 같은 서비스 사업자들이 모여, 일본의 NTT, NEC등과 함께 오픈 플로우 상용화를 위한 컨소시엄인

ONF(Open Networking Foundation)을 구성하여 공격적인 SDN 표준화 및 기술 보급에 나서는 등, 최근 국내외로 네트워크 산업의 가장 중요한 핵심적인 기술로 인정받고 있다[4].

2.3 NFV와 SDN의 연관 관계

NFV가 반드시 SDN을 사용하는 것은 아니다. 기존에 나와 있는 수많은 기술을 이용하여 본래의 취지에만 맞으면 된다. 하지만 NFV의 취지를 위해서는 NFV와 SDN이 매우 상호 보완적인 관계인 것만은 분명하다. 현재 SDN에 적극적인 주요 벤더들은 동시에 NFV의 주요 벤더들이다. NFV와 SDN을 통해 오픈된 경쟁 시장을 조성함으로써 다양한 선택 및 비용 절감 그리고 효율적 운영을 가능하게 할 분만 아니라 '고객이 주도권을 가지는 자신만의 네트워크' 구성으로 원하는 서비스를 개발하여 새로운 수익 창출의 기회까지 도모해 나갈 수 있게 된다[6].

2.4 Mininet

Mininet[13]은 스탠퍼드 대학에서 배포하였으며 개인 pc나 노트북에서 쉽고 빠르게 Virtual OpenFlow Network를 구성하여 테스트 할 수 있는 OpenSource Project이다. Mininet은 soft Switch 기반으로 OpenFlow Specification 1.0을 지원한다. 2013년 9월 기준 2.1 버전이 출시 되었다[6].

Mininet은 단일 시스템에서 간단한 명령으로 실제커널, 스위치 및 애플레이케이션 코드를 실행하고 에서 현실적인 가상 네트워크를 만든다. OpenFlow와 SDN 시스템의 개발, 공유, 실험에 있어서 좋은 툴이다. Mininet으로 네트워크 토폴로지를 구성할 때, 사용 하는 스위치는 일반스위치가 아닌 OVS(Open Virtual Switch)이기 때문에 Controller를 스위치에 연결해주어야 한다. 실제로 이번 실험에서 Controller를 사용하진 않았지만 OVS에 연결하기 위해 토폴로지에 Controller를 사용하였다.

2.4 Snort

Snort는 "sniffer and more"라는 말에서 유래되었는데, 처음 공개되었을 때는 코드도 얼마 되지 않은 단순한 패킷 스니퍼 프로그램이었다. 그러나 이후 현재의 IDS와 같이 rule을 이용한 분석 기능이 추가되고, 커뮤니티를 통하여 계속적인 기능 보완과 향상을 통해 지금과 같이 다양한 기능과 탁월한 성능을 갖춘 프로그램이 되었다. snort는 공식 홈페이지인 http://www.snort.org를 통해 지속적인 업그레이드가 가능하다.

Snort는 먼저 스니퍼를 통해 snort IDS를 통과하는 모든 패킷을 수집하게 된다. 여기에서 수집된 데이터는 바로 룰 기반의 탐지 엔진을 거치지 않고 그 전에 preprocessor를 통해 보다 효율적인 공격 탐지를 위해 HTTP 인코딩이나 플러그인이나 포트스캔 등 몇 가지 플러그인을 먼저 거치면서 매칭이 되는지 확인하게 된다. 물론 preprocessor 역시 모듈화 되어 있어 각자의 환경에 불필요하다면 disable 할 수 있다. 그리고 preprocessor를 통과한 패킷은 snort IDS의 룰 기반의 탐지엔진을 거치면서 사전에 정의된 탐지룰과 매칭되는지 확인하게 된다. 만약 룰에 매칭되었을 경우에는 사전에 정의된 정책에 따라 로그에 남게 되고, 그렇지 않은 경우 통과를 하게 된다[7].

2.5 Snort Rules 시그니쳐 구조

Snort의 시그니쳐는 룰 헤더와 룰 옵션의 2가지 섹션으로 분류된다. 룰 헤더에는 처리 방법, 프로토콜, IP주소, 포트 번호 등의 처리 대상으로서의 판단 기준을 기술한다. 룰 옵션에는 alert 메시지나 패킷 내부의 조사 내용을 기술한다. 시그니쳐는 다음 그림으로 분류 된다.

Rule header							Rule option
action	protocol	IP address	port	->	IP address	Port	(option)
1	2	3	4	5	6	7	8

그림3. snort rules Architecture [8]

action에는 패킷 처리 방법을 alert, drop, sdrop등 중에서 1개를 지정해야 한다. 각각의 action은 의미가 각각 다르다. protocol에는 패킷의 프로토콜을 지정한다. 이번 논문에서는 udp, tcp 프로토콜만 사용하였다. IP address는 첫 번째 에는 송신자 IP address를 기술하고 두 번째 에는 수신자 IP address를 기술한다. 임의의 IP주소인 any를 지정할 수도 있다. IP address와 마찬가지로 port도 같다. 마지막 부분인 옵션부분은 지정할 수 있는 룰 옵션은 매우 많다.

옵션	형식
msg	msg: "<메시지 텍스트>";
dsize	dsize:[>|<]<한계치>[<><한계치>];
content	content: [!]"<검색 문자열>";
offset	offset: <오프셋 번호>;
depth	depth: <페이로드 깊이>;
nocase	nocase;
flags	flags: <플래그>[,마스크];

그림4. snort rules Architecture [8]

3. 관련 연구

NFV의 관심과 침입 탐지 시스템으로 유명한 Snort와 관련된 연구주제는 다양하게 존재한다. 성능 평가 부분과 연관되는 주제의 연구로는 Suricata와 Snort의 비교와, Snort 사용 시 IPv4 패킷과 IPv6 패킷 구분에 따른 처리 효율을 알아보는 실험이 있었다[9]. Suricata와 Snort의 비교 실험에서는 Snort는 싱글 스레딩을 사용하는 반면 Suricata는 멀티 스레딩을

도입으로 인해 패킷을 스니핑하여 수집하는 작업부터 규칙과 비교하는 탐지 작업등 세부적인 작업들을 각각 하나의 스레드로 나누어 각각의 스레드에 대하여 해당 스레드의 작업이 동시에 여러 CPU Core에서 이루어질 수 있도록 설정함으로서 Suricata가 Snort보다 CPU Core 개수에 비례하여 월등하다는 것이다.

침입 탐지 시스템의 IPv4 및 IPv6 패킷 처리 성능에 관한 연구는 패킷 간 간격이 임계치를 넘어서면 패킷 검사를 수행하지 못한 채 대상 프로세스로 전송하는 패킷이 발생하는데, 이러한 임계치가 IPv4 환경에서는 259μs, IPv6 환경에서는 39μs로 나타난다. 이러한 결과는 IPv4에 비해 IPv6의 프로토콜 헤더가 단순해서 보다 효율적인 검사를 수행하기 때문이라는 실험내용이다[10].

이 논문에서는 오픈소스인 Snort기반의 IDS NFV를 구축하고 해당 서비스의 실제 활용 가능성을 평가 하기 위해 IDS NFV 서비스 사용 유무에 따른 패킷 지연 시간을 비교하는 실험을 수행하고 그 결과를 분석 하였다.

4. 오픈소스 기반 IDS NFV 구축 및 평가

4.1 IDS NFV 구축 및 SYN Flooding 탐지 기능 검증

Mininet을 이용하여 SDN기반의 가상 네트워크 토폴로지를 구성하고, 해당 네트워크 상에서 IDS NFV를 구축하여 그 성능을 테스트 하였다. 구성된 가상 네트워크 토폴로지는 그림 5와 같다. 총 2개의 OVS와 3개의 가상 호스트로 네트워크를 구성하고, SDN 컨트롤러는 OVS들과 연결된다. Snort(version 2.9.7.2)는 타겟 호스트와 연결된 OVS의 network interface를 감지할 수 있는 곳에 위치한다. Mininet, Controller그리고 Snort은 Ubuntu 12.04로 운영되는 머신 상에서 구동하였다.

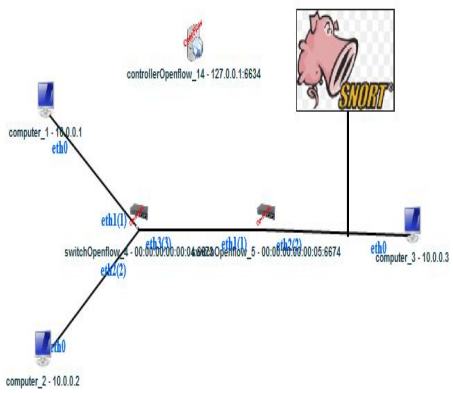

그림5. Network Topology

구축된 오픈소스 기반 IDS NFV의 기능을 테스트하기 위해 SYN Flooding Attack을 탐지 시나리오를 만들고 검증하였다. 검증을 위한 시나리오는 타겟 호스트인 host3(compuer_3 : 10.0.03)에서 http서버를 구동시키고, 공격 호스트인 host1(computer_1 : 10.0.01)에서 host3에 설치된 http서버에게 SYN Flooding공격을 수행한다. SYN Flooding공격은 잘 알려진 네트워크 테스팅 툴중 하나인 hping3을 사용하였다[12]. 사용한 Snort Rule은 10초안에 70개가 넘는 tcp 패킷이 들어온다면 SYN Flooding 어택을 탐지하도록 작성하였다.

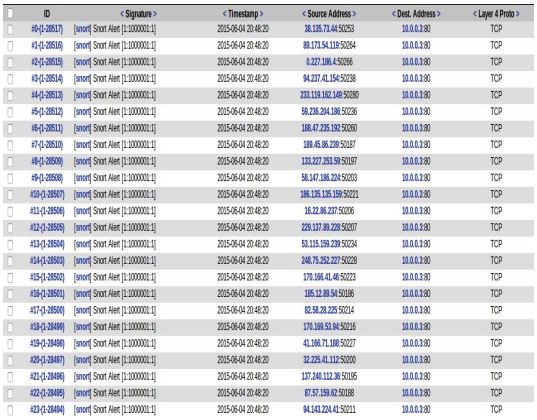

그림6. Snort를 이용한 SYN Flooding 검출 화면

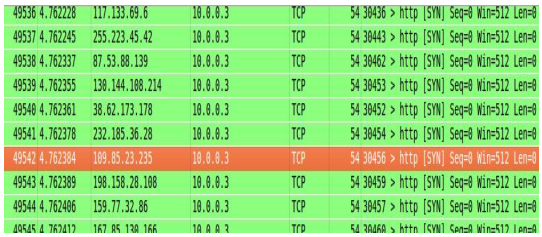

그림7. wireshark를 이용한 SYN Flooding 검출 화면

구축된 IDS NFV의 기능 검증은 그림 6 및 그림 7과 같이 Snort의 로그 확인 및 Wireshark를 통한 네트워크 인터페이스 탭핑 결과 확인을 통해 수행하였다.

4.2 IDS NFV 사용에 따른 성능 비교

IDS NFV를 사용함에 따른 네트워크 성능을 이해하기 위해, 그림 5와 같이 구축된 가상의 토폴로지 상에서 TCP/UDP 네트워크 프로토콜을 사용한 네트워크 트래픽 전송 지연 시간을 측정하였다. Host3 (computer_3 : 10.0.0.3)에 TCP/UDP 서버를 두고 Host2 (computer_2 : 10.0.0.2)에 클라이언트를 설치하고, 해당 클라이언트는 서버에 사이즈가 500Byte인 패킷을 1초에 500개씩 전달하도록 하였다. 이는 약 250KByte의 대역폭을 가지는 네트워크 플로우를 생성하고, 이는 유투브 비디오를 고화질로 플레이하는 정도의 데이터를 사용하는 네트워크 플로우를 표현한다고 할 수 있다. 각 패킷의 네트워크 지연 시간은 패킷이 클라이언트에서 전송했을 때의 Timestamp와 서버에서 수신되었을 때의 Timestamp의 차이로 구하였다. 네트워크 평균 지연시간의 측정은 네트워크 패킷을 10000, 30000, 50000, 100000, 150000개 보냈을 때의 각 지연시간의 평균을 이용해 측정하였다.

Snort 검출 룰은 프로토콜, 포트, IP주소, 옵션부분의 변화에 따라 작성하였고, 약 100개 정도의 룰을 사용하였다. 이러한 조건에서 Snort의 사용 유무에 따른 평균 네트워크 지연 시간의 결과는 그림8과 같다.

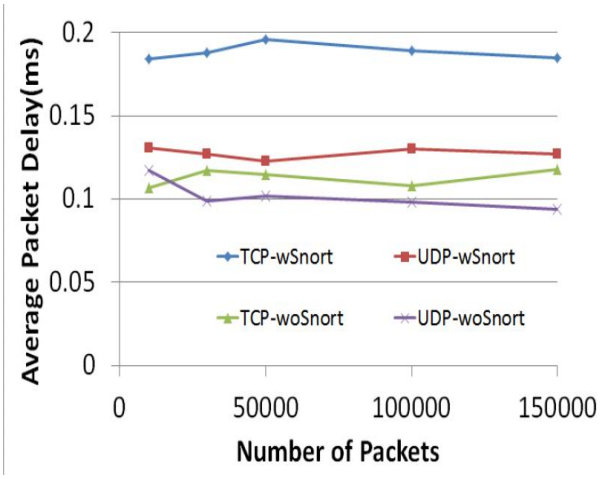

그림8. Snort를 유/무에 따른 TCP, UDP패킷 평균 지연 시간

그림 8에서 TCP-wSnort와 UDP-wSnort는 Snort이 사용되었을 때 TCP와 UDP 패킷의 평균 지연 시간을 각각 나타내고, TCP-woSnort와 UDP-woSnort는 Snort을 사용하지 않았을 때의 TCP와 UDP 패킷의 평균 지연 시간을 각각 나타낸다. 평균 지연 시간의 단위는 ms(밀리세컨드)이다. 그림 8과 같이 Snort을 사용할 때 네트워크 지연 시간이 증가함을 알 수 있었다. UDP 프로토콜의 경우 약 0.02ms의 네트워크 지연시간의 증가가 발생하고, TCP 프로토콜의 경우 약 0.08ms의 네트워크 지연시간 증가가 발생함을 알 수 있었다. TCP의 경우 UDP에 비해 약 4배정도의 네트워크 지연이 더 발생하는데, 이는 TCP프로토콜의 복잡도가 UDP에 비해 더 높은데서 기인하는 것으로 분석 된다. 이 결과에 따라, 일반적인 광대역 네트워크 상에서의 네트워크 지연 시간이 수백 ms인점을 감안한다면, Snort을 사용하더라도 추가적인 네트워크 지연시간이 미치는 영향은 아주 적을 것으로 분석된다.

5. 결론

SDN 및 NFV는 향후 새로운 인터넷 구조를 이끌어나갈 기술들로, 해당 기술들을 상용화 및 실용화에 대한 고찰은 꼭 필요하다. 본 논문에서는 실제 현재의 네트워크 장비들이 NFV를 이용하여 구현된다면 기존의 하드웨어 기반의 장비들 보다는 소프트웨어가 장비위에 올라가게 되므로 시간 지연이 발생할 것 이라는 예상을 가지고 오픈소스 SW기반의 IDS NFV를 구축하고 평가를 수행하였다. 그 결과 IDS NFV를 사용할 경우 네트워크 지연시간은 작은 시간이지만 존재함을 알 수 있었다. 하지만 이는 광대역 네트워크 상의 네트워크 플로우 관점에서 바라보면 아주 작은 양으로 분석된다.

이 논문에서는 서버와 클라이언트의 하나의 세션만 맺어서 전송하는 네트워크 플로우에 대한 네트워크 지연시간을 측정하였다. 향후 네트워크 플로우의 복잡도에 따른 성능 평가 및 TCP/UDP외 기타 네트워크 프로토콜들에 대한 Rule set의 다양성에 따른 성능 평가를 진행할 계획이다.

감사의 글

이 논문은 2015년 정부(교육부/미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2014R1A1A1007734).

참 고 문 헌

[1] 이정희,이상민,최강일,이범철 “NFV(Network Functions Virtualisation)” 한국통신학회지(정보와통신) 제30권 제3호 51.page 2013.

[2] Ed tittel | CIO. “상충? 보완? SDN과 NFV의 공존에 대한 이해” (http://www.ciokorea.com/news/19770?page=0,1) 2014.

[3] IXIA “캐리어급 전달 보장을 위한 네트워크 기능 가상화 (NFV) 구현 및 테스팅” (www.ixiacom.com) April, 2014.

[4] 신명기 “ICT EXPERT INTERVIEW SDN에 대한 소개” TTA Jounal vol.151 pp. 15 January, 2014

[5] ONF“Software-Defined Networking (SDN) Definition” https://www.opennetworking.org/images/stories/sdn-resources/meet-sdn/sdn-3layers.gif) SDN Architecture.

[6] 서영석, 이미주, "오픈소스를 활용한 OpenFlow 이해하기 SDN 입문" 영진닷컴 , pp. 43-45, 2014.

[7] 인프라보호단/보안관리팀, “Snort를 이용한 IDS구축” KISA 한국정보보호 진흥원 pp 2-3 May, 2005

[8] Ei Jang “Snort 시그네처의 구조” IT,컴퓨터 http://blog.naver.com/misman95/80041665451

[9] 정명기,안성진,박원영 “Snort와 Suricata의 탐지 기능과 성능에 대한 비교연구” 융합보안 논문지 제 14권 제5호 (2014 ,09)

[10] 윤주환, 임을규 “침입 탐지 시스템의 IPv4 및 IPv6 패킷 처리 성능에 관한 연구” 한국통신학회 동계종합학술발표회 (2012)

[11] Steve Noble “Network Function Vritualization or NFV Explained” (http://wikibon.org/) Apr, 2015

[12] Dos Using Hping3 with spoofed IP in KALI Linux, http://www.blackmoreops.com/2015/04/21/denial-of-service-attack-dos-using-hping3-with-spoofed-ip-in-kali-linux/

[13] Mininet, http://mininet.org