Journal of Knowledge Information Technology and Systems

ISSN 1975-7700

http://www.kkits.or.kr

Establishment of Testbed for Detection of Harmful Traffic in SDN-based Network

Jung Yun KIM*

Department of Computer Science and Engineering, Kyunghee University

ABSTRACT

SDN-based networks have many advantages over existing networks. SDN is different from existing network configuration It is a method of delivering packets based on software functions in network equipment, which has many advantages in terms of cost and operation over existing network structure. However, since it is built around OpenFlow, which is the most widely known interface in SDN, it is also vulnerable to security. So SDN tried to solve this by providing virtualized security function based on NFV(Network Function Virtualization). The advantage of NFV is that it is possible to apply security only to the place where security is needed by using controller without need to set up like existing equipment. In this paper, I tried to investigate a method to filter harmful traffic without the ACL(Access Control List), ZFW(Zone-Based Firewall) and other functions set in the security device in the NFV that operates for security in the SDN environment. The hardware-based security technologies currently used are inevitably subject to delays and can not be used to prevent security attacks that are executed in various ways by changing port numbers. Therefore, the purpose of this paper is to improve the security performance of SDN based on NFV by analyzing traffic pattern and filtering harmful traffic and nontoxic traffic. For this study, a test network based on actual SDN was constructed and the pattern - based harmful traffic detection technology was proved.

© 2017 KKITS All rights reserved

KEYWORDS: SDN-based network, OpenFlow, Network Virtualization, Traffic Filtering, Detecting harmful traffic, NFV(Network function virtualization), SDN security

ARTICLE INFO: Received 31 May 2017, Revised 9 June 2017, Accepted 9 June 2017.

*Corresponding author is with the Department of Computer Science and Engineering, Kyunghee University, 1732, Deogyeong-daero, Giheung-gu, Yongin-si, Gyeonggi-do 17104, Republic of Korea.

E-mail address: inokyuni@khu.ac.kr

www.kci.go.kr

1. 서 론

SDN(Software Defined Networking, SDN)은 기존의 네트워크와는 완전히 다른 개방형 API(Application Programming Interface)를 기반으로 네트워크의 트래픽을 전달하는 방식이다. 라우터와 스위치 같이 OSI 7 Layer 기반의 장치를 통해 트래픽이 전달되는 방식이 아닌 말 그대로 개방형의 API를 기준으로 이와 관련된 오픈소스 기반의 컨트롤러를 통해 트래픽을 전달하는 방식이므로, 크게 두 가지로 나뉘어져 있다. 하나는 트래픽의 경로를 어떻게 설정할 것인가를 결정하는 Control Plane 및 트래픽의 전송을 직접적으로 실행하는 Forwarding Plane으로 나뉘어져 있다.[1]

기존의 장비를 기반으로 네트워크 서비스가 되고 있음에도 불구하고 SDN 기반의 네트워크가 주목을 받는 이유를 알아볼 필요가 있다. 현재 사용하는 인터넷 환경의 변화로 트래픽이 더욱더 많아지는 결과를 가져오게 되었다.

처리하여야 하는 트래픽이 많아지는 상황에서 네트워크 장비를 계속적으로 추가하여 관리하기에는 분명히 물리적인 한계가 있을 수밖에 없고, QoS 등의 네트워크에서 처리하여야 하는 여러 서비스들이 각 장비마다 설정이 되어야 하는 번거로움 및 관리상의 어려움, 그리고 이러한 네트워크가 운영되는데 있어서 들어가는 비용 등의 문제를 해결하는데 SDN 만큼 효율적인 네트워크는 없기 때문이다. SDN은 위의 문제를 해결하기 위하여 SDN Controller를 가지고 있는데, 이는 네트워크를 중앙에서 쉽게 관리할 수 있도록 하는 기능을 제공한다.[2] SDN이 가지고 있는 목적 중에 하나는 하드웨어 기반의 네트워크 관리를 소프트웨어 중심으로 변경하는 것이 목적이며, 이를 기반으로 확장성 및 비용의 절감을 비롯한 가용성 등을 제공하는데 목적이 있고, 세계적으로 유명한 회사들이 이미 SDN을 기반으로 서비스를 구축하고 있다.[11] 그러나 이러한 장점을 가지고 있는 SDN 기반의 네트워크에서도 보안과 관련된 이슈는 피해갈 수 없다. 현재 라우터 및 스위치 기반의 네트워크에서 정상적으로 전달되는 트래픽과 유해한 트래픽을 분류하기 위해서 특정 포트를 막거나 IP 주소 등의 정보를 통해 트래픽을 제어하지만, 이는 기존에 알려진 유해 트래픽을 필터링 하기 위해 사용하는 방법이다. 따라서 본 논문에서는 SDN의 정상적인 트래픽의 패턴을 분석하기 위하여 트래픽을 수집할 테스트베드를 구축하였다. 정상적인 트래픽 패턴을 벗어나는 트래픽을 유해 트래픽으로 규정하기 위함이다.

본 논문의 구성은 다음과 같다. 2장에서는 SDN의 구조를 알아보고, 기존 네트워크와 비교하여 어떠한 점이 더 효율적인지 알아본다. 3장은 정상적인 트래픽 패턴을 분석하기 위해 Testbed를 구축하고, 4장에서는 SDN 네트워크에서 트래픽을 분류할 수 있는 방법에 대해 설명하고 실제 데이터를 수집하며, 5장에서는 결론을 기술한다.

2. SDN 구조

2.1 SDN 구조개요

현재 SDN는 표준화가 되지 않았다. 다만 ONF(Open Networking Foundation)에서 표준화를 진행하고 있으며, ONF에서는 SDN을 Application Plane, Control Plane, Data Plane으로 나뉘었고, 각각의 역할을 정의하였다. SDN Application은 네트워크에서 요구되는 정책 사항을 프로그래밍을 통해 구현하는 일종의 솔루션 역할을 한다. [8]

<그림 1>에서 SDN Application에 대한 구조를 볼 수 있으며, NBI(Northbound Interfaces) Driver를 통해 Control Plane 즉, SDN Controller와 연결된다.

www.kci.go.kr

그림 1. SDN Application 구조
Figure 1. SDN Application Architecture

Control Plane은 SDN의 전신이라고 해도 과언이 아닐 정도로 SDN에서 핵심적인 역할을 수행하고 있고, 하나 이상의 NBI Agent와 SDN Control Logic 및 CDPI(Control to Data-Plane Interface) Driver로 구성되어 있다. SDN 초기에 가장 많은 사용자를 확보했던 Controller는 NOX 이었으나, 최근들에 다양한 종류의 Controller가 공개되면서 NOX의 사용자들이 분산되고 있는 추세이다.[12][13]

SDN 기반으로 많이 사용되고 있는 Open Source 기반의 Controller를 <표 1>에서 정의하였다.

표 1. Open Source Controller 종류
Table 1. Open Source Controller Type

Controller Type	Language	OS
Beacon	Java	Win, MAC, Linux, Android
Floodlight	Java	Win, MAC, Linux, Android
Maestro	Java	Win, MAC, Linux
Opendaylight	Java	Win, MAC, Linux
RoutFlow	C++, Python	Linux
OpenFlow	C	Linux
NOX	Python, C++	Linux
POX	Python	Win, MAC, Linux, Android
Trema	C, Ruby	Linux

<그림 2>는 SDN의 Controller를 나타내고 있으며, 이 Controller를 통해 네트워크의 설계 및 운용을 기존의 네트워크보다 단순하게 구성할 수 있게 된다. 또한, 네트워크를 관리하면서 논리적인 스위치로 간주되므로, 특정 회사의 장비에 영향을 받지 않고 네트워크 설계가 가능하다. [9]

NBI Agent를 기반으로 Application Plane의 NBI Driver로 Northbound 통신을 통해 외부 관리 시스템이나 네트워크와 관련된 정보 등을 추출하고 정책을 제어할 수 있다.

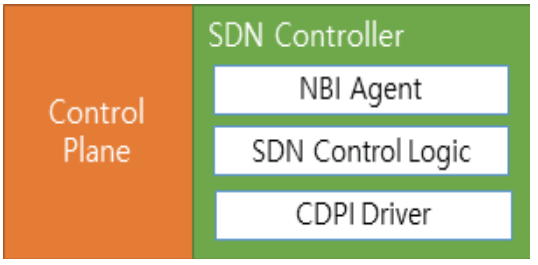

그림 2. SDN Control Plane 구조
Figure 2. SDN Control Plane Architecture

SDN의 Data Plane은 트래픽을 전송하려는 개체로 보면 무난하겠다. 이 영역은 CDPI Agent와 Forwarding Engine 및 Processing Function으로 구성되어 있으며 <그림 3>에서 Data Plane의 구조를 확인할 수 있다.

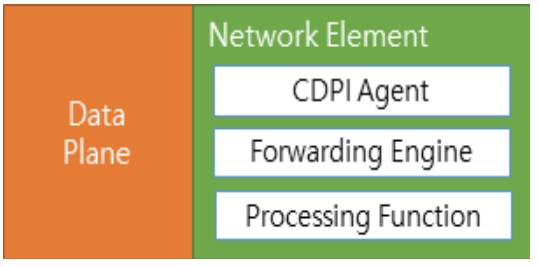

그림 3. SDN Data Plane 구조
Figure 3. SDN Data Plane Architecture

www.kci.go.kr

SDN에서 Southbound 통신은 제어 자원과 데이터 자원의 연결을 위해 OpenFlow 프로토콜을 사용하여 Controller 및 스위칭 Hardware와의 통신을 정의하고 있다. 또한 Controller 및 Switch 사이에 암호화된 전송 계층 보안 인증을 기반으로 하는 통신도 지원하고 있다. [10]

위와 같이 SDN은 세 개의 영역으로 나뉘어 동작하고, 이 세 개의 영역이 상호간에 어떻게 동작하는지는 <그림 4>에서 확인할 수 있듯이 SDN Controller가 주된 역할을 한다. [7]

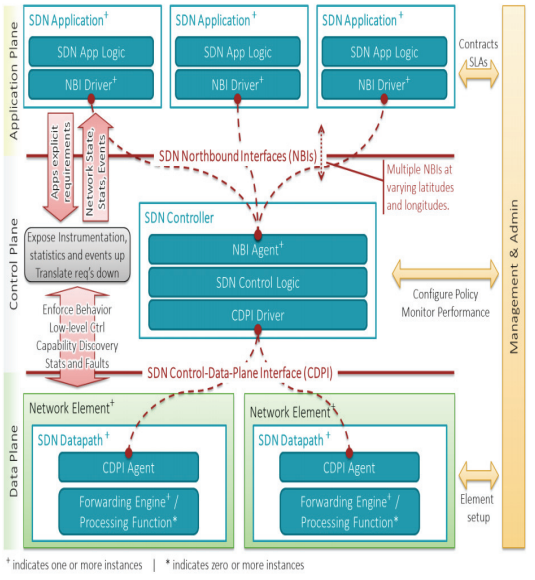

그림 4. SDN 네트워크 구조
Figure 4. SDN Architecture

3. SDN 테스트 베드 구축

3.1 Hybrid 방식의 스위치

ONF를 중심으로 하여 활발하게 논의되고 있는 방안 중에 하나인 Hybrid 방식의 SDN 스위치는 OpenFlow 기반의 데이터와 현재 장비 위주로 되어 있는 라우팅을 동시에 처리하는 스위치이다.[14]

OpenFlow 1.5.1에서는 Egress Table을 도입하여 Output 포트에서도 Processing을 할 수 있도록 하였고, 이는 <그림 5>을 통해서도 잘 보여주고 있다.[3]

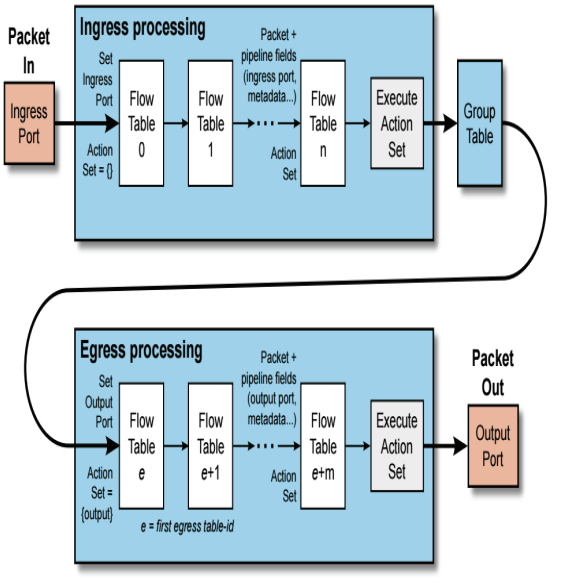

그림 5. Pipeline을 통하는 패킷 흐름 구조
Figure 5. Packet flow through the pipeline

이전 버전에서는 Ethernet 패킷만 처리하였으나 1.5.1에서는 다양한 패킷 타입 들을 처리할 수 있으며, Statistic trigger 방식을 도입하여 기존의 높은 오버헤드의 문제점을 해결하고, Threshold 값을 기반을 컨트롤러에게 통계정보를 보내는 점이 달라졌다. 1.5.1 이전의 OpenFlow에서의 L2스위칭과 라우팅 및 VLAN을 포함하는 QoS와 ACL을 지원하며, 이는 OpenFlow FIB에 기존의 FIB를 추가 하는 것으로 구현될 수 있다.[4]

Packet Register Pipeline Fields는 이전의 버전에는 없었던 것으로, 이 필드를 사용하여 Pipeline Processing에서 Packet 정보를 임시로 저장한다. 또한, TCP flag matching을 통해 TCP 헤더의 flag

www.kci.go.kr

bits를 식별할 수 있으며, TCP 연결의 시작 및 연결이 끝나는 것을 감지할 수 있게 되었다. 그리고 Bucket에 Bucket_id가 추가 되므로 인하여 하나의 Group에서 지정한 Group Bucket 만을 삭제하고 삽입할 수 있으며, Controller Connection status를 기반으로 Controller가 스위치와 연결된 모든 Controller의 상태를 파악할 수 있도록 하였다.

3.2 OpenDaylight SDN Controller

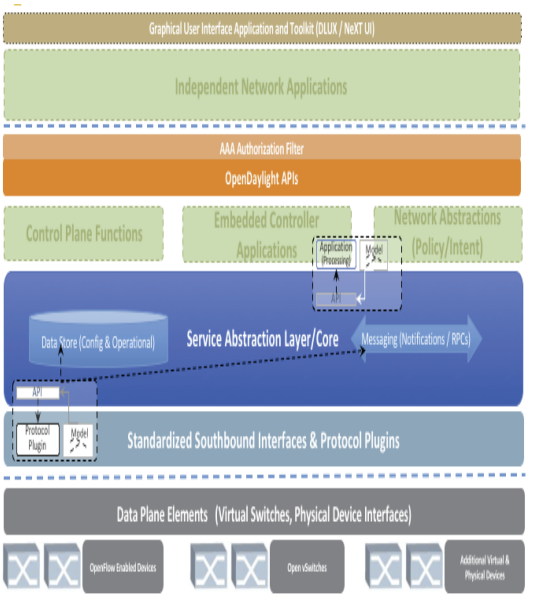

그림 6. ODL(OpenDaylight) 구조
Figure 6. ODL(OpenDaylight) Architecture

OpenDaylight(ODL)은 다양한 크기의 네트워크를 위한 모듈형 Open SDN 플랫폼이다. ODL는 멀티 벤더 환경에서 다양한 하드웨어의 네트워크 서비스를 가능하가 하며, 마이크로 서비스 아키텍처를 통한 Application 및 Protocol, 그리고 Plug-in을 제어할 수 있으며, 개방형 API를 통합하여 네트워크를 보다 지능적이고 다양하게 구성하기 위한 SDN 플랫폼을 지원한다. 또한 모듈형 설계로 인하여 이미 만들어진 다른 서비스를 활용할 수 있으며, 여러 서비스와 프로토콜의 결합이 가능하다. ODL의 구조는 <그림 6>과 같다.[6][15]

3.3 Topology

SDN 기반의 테스트 베드를 구축하여 TCP와 UDP Traffic이 SDN 기반의 네트워크에서 어떻게 수신이 되고 송신이 되는지 알아보고자 했다. 다만, 실제 장비를 사용하기에는 한계가 있으므로 PC 기반의 가상화 프로그램을 사용하여 테스트 베드를 구축하고, 상호 Traffic을 전송하여 Traffic이 잘 전달되는지 확인하여 유해하지 않은 TCP와 UDP의 트래픽이 어떠한 패턴을 가지는지 확인하고자 했다. 테스트 베드의 토폴로지는 <그림 7>과 같다.

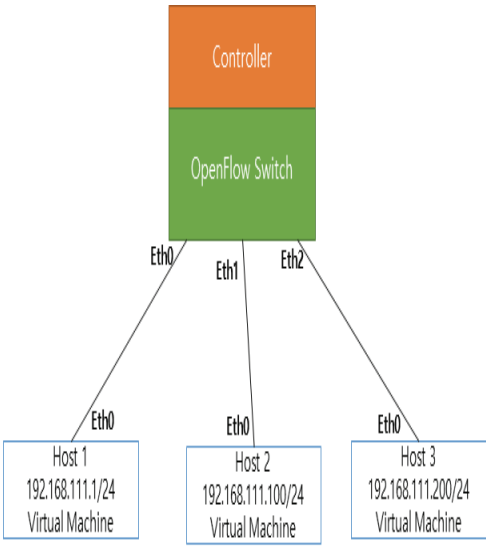

그림 7. 테스트베드
Figure 7. Testbed

Host1의 운영체재는 윈도우 10을, Host2의 운영체재는 Centos, Host3의 운영체제는 우분투 리눅스

www.kci.go.kr

를 사용하여 구성하였다. 모든 방화벽은 해제하였으며, Traffic의 원활한 흐름을 위하 Traffic 제너레이터도 같이 설치하여 사용하였다.

4. SDN Traffic 수집 및 모니터링

4.1 iperf3

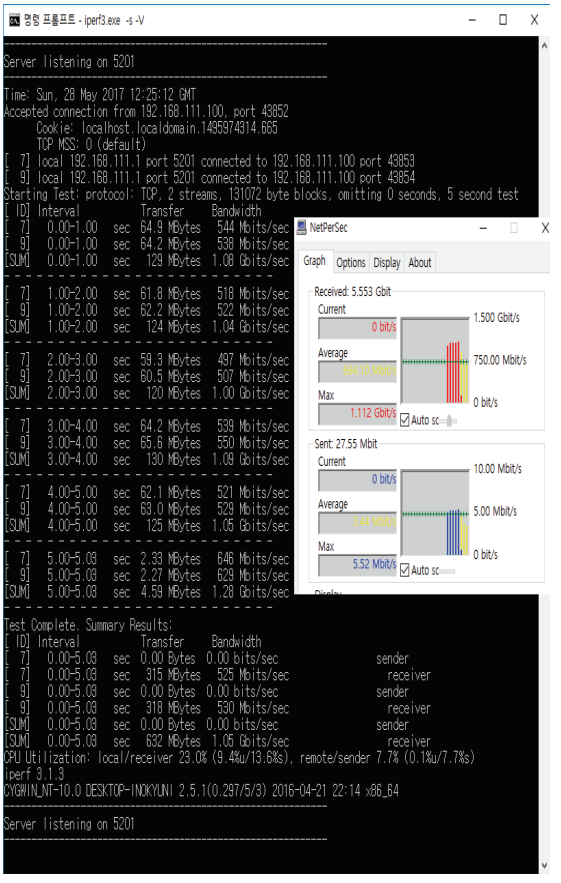

그림 8. Traffic 생성 및 모니터링
Figure 8. Traffic generation and Monitoring

iperf3는 유선 및 무선 네트워크 통신 구간에서 최대전송능력 즉 대역폭을 측정할 수 있는 프로그램이다. TCP와 UDP, 그리고 IPv4/IPv6 환경을 지원하고 멀티캐스트 등을 지원하며, GUI 형태로 그 결과도 보여주며, 가장 큰 장점은 리눅스나 안드로이드는 물론이고 윈도우 환경에서도 잘 동작한다.[5]

또한 iperf3를 이용하여 TCP와 UDP의 트래픽을 생성하여 SDN 기반의 네트워크에서 이를 전송하고 트래픽을 확인 하고자 한다.

Host1에서 Host 2로 5초 동안 2개의 Stream을 동시에 서버로 전송하고 이를 측정한 결과가 <그림 8>과 같으며, UDP Traffic을 생성한 화면과 UDP Traffic의 수신 결과는 <그림 9>와 같다.

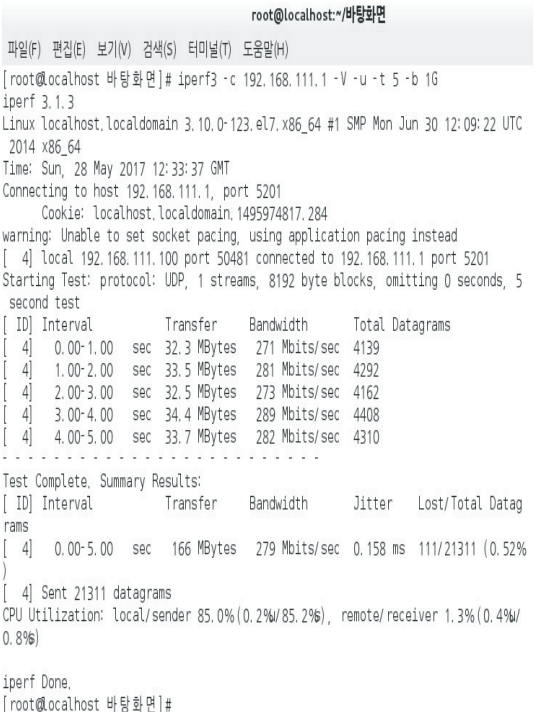

그림 9. UDP Traffic 생성 및 모니터링
Figure 9. UDP Traffic generation and Monitoring

위의 결과를 보면 21311개의 UDP 패킷을 보냈으나, 111개의 패킷이 Lost 되었다는 것을 Traffic을 생성하여 전송한 쪽과 UDP Traffic을 받은 쪽의 모니터링을 통해 확인할 수 있었다. 이렇듯 트래픽이 상호 전송되는 환경을 구축함으로 인하여, 구축

www.kci.go.kr

된 SDN 네트워크를 기반으로 안정적으로 동작하는 네트워크 트래픽을 모니터링 하여 안정적으로 동작하는 네트워크의 패턴을 분석한다.

그 후, 안정적으로 동작하는 네트워크의 패턴 분석과 일치 하지 않는 비정상적 패턴을 가지는 트래픽을 생성하여 SDN 네트워크에 전송한 후, SDN 네트워크에서 비정상적인 트래픽의 패턴을 분석하여 정상적이지 않은 트래픽이라고 판단한 후, 이를 차단함으로써 SDN 기반 네트워크에서 보안과 관련된 이슈들을 해결한다.

4. 결 론

SDN 기반의 네트워크는 현재도 진화하고 있고, 앞으로도 현재 네트워크의 문제점을 해결하고 장비에 들어가는 비용을 낮추는 등의 여러 가지 장점으로 인해 많이 사용될 것으로 예측된다.

기존의 장비 벤더들도 SDN을 지원하고 있으며, 표준화도 꾸준히 진행되는 등, 여러 면에서 SDN이 구체화 되고 상용화되기 위해 많은 기술들이 개발되고 적용되고 있다.

본 논문에서는 이러한 SDN 네트워크에서 유해 트래픽을 검출하는데 있어서 기존의 여러 방법들과 달리 SDN 에서 실제 동작하는 정상적인 트래픽의 패턴을 기반으로 유해 트래픽이 들어올 경우, 정상적인 트래픽의 패턴과 다른 것을 감지해 자동으로 트래픽을 전송하지 않으므로 알려지지 않은 네트워크 공격 등에 대비하고자 하였다. 위와 같은 연구를 진행하면서 안정적인 SDN 네트워크 구축이 필요하게 되었고, 이를 구축한 후, iperf3를 이용하여 Traffic을 주고받음으로써 테스트베드가 구축되었음을 증명하였다.

향후, 이 테스트베드를 기반으로 하여 정상 적인 트래픽의 패턴을 분석하고, 유해 트래픽 패턴을 분석 한 후, 이를 비교하여 트래픽을 전달하지 않는 방법 등의 연구가 필요하다.

References

[1] W. Xia, Y. Wen, C. H. Foh, and H. xie, *A survey on software-defined networking,* Communications Surveys & Tutorials, IEEE Vol. 17, No. 1, pp. 27-51, 2014.

[2] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, *Scalable flow-based networking with DIFANE,* ACM SIGCOMM Computer Communication Review Vol. 41, No. 4, pp. 351-362, 2011.

[3] OpenFlow Networking Foundation, *OpenFlow Switch Specification Version 1.5.1(Protocol version 0x06*, pp. 18-21, 2015.

[4] OpenFlow deployment, https://www.opennetworking.org/technical-communities/areas/specification, Nov. 2016.

[5] iPerf - The ultimate speed test tool for TCP, UDP and SCTP, https://iperf.fr/, Dec. 2016.

[6] Open Networking Foundation, https://www.opennetworking.org/technical-communities/areas/specification Resource & Publications, Jan. 2017.

[7] OpenDaylight, https://www.opendaylight.org/ Jul. 2016.

[8] Open Networking Foundation, Nick McKeown, *Software-Defined Networking: The New Norm for Networks,* ONF White Paper pp. 2-12, Apr. 2012.

[9] S. Sezer, S. Scott-Hayward, P. Kaur Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Ra0, *Are we ready for SDN? Implementation challenges*

www.kci.go.kr

for software-defined networks, Communications Magazine, IEEE Vol. 51, No. 7, pp. 36-43, 2013.

[10] D. Kreutz, F. M. V. Ramos, P. Esteves Verissiom, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, *Software-defined networking: A comprehensive survey,* proceedings of the IEEE Vol. 103, No. 1, pp. 14-76, 2015.

[11] OpenDaylight, https://www.opendaylight.org/platform-overview/, Jul. 2016.

[12] Cisco OpenDaylight, https://communities.cisco.com/community/developer/opendaylight/blog, Mar. 2017.

[13] OpenFlow deployment, https://www.opennetworking.org/sdn-resources/opensource-sdn, Dec. 2016.

[14] Global Environment for Network Innovations, http://groups.geni.net/geni/wiki/GENIExperimenter/Tutorials, Jun. 2016.

[15] Wikipedia, OpenFlow, https://en.wikipedia.org/wiki/OpenFlow, Mar. 2016.

SDN 기반 네트워크에서의 유해트래픽 검출을 위한 테스트베드 구축

김정윤

경희대학교 컴퓨터공학과

요 약

SDN은 기존의 네트워크의 구성과 달리 네트워크 장비에서 소프트웨어 기능을 기반으로 패킷을 전달하는 방식으로, 기존의 네트워크 구조보다 비용 및 운용적인 측면에서 많은 장점을 가지고 있다. 그러나 SDN은 OpenFlow 인터페이스 중심으로 구축되기 때문에 기존의 네트워크와 마찬가지로 보안에 취약점을 가지고 있기는 SDN도 마찬가지이다. 이러한 보안 취약점을 보완하고자, SDN은 NFV(Network Function Virtualization)를 기반으로 가상화된 보안기능을 제공하여 보안 문제를 해결 하고자 하였다. NFV의 장점은 기존의 장비처럼 네트워크 보안 설정 등을 모든 장치에 다 하지 않고, 컨트롤러를 사용하여 한 번에 해결한다는 것이 큰 장점이다. 현재 사용되고 있는 하드웨어 기반의 보안 기술은 Delay가 발생할 수밖에 없으며, 포트 번호를 바꿔 다양하게 실행되는 보안 공격을 막는데 역부족이다. 따라서 본 논문은 SDN 환경에서 보안을 위해 동작하는 NFV에서 보안 장치에 설정되는 ACL, ZFW등의 기능 없이도 유해 트래픽을 필터링 할 수 있는 방법에 대해서 연구하고자 하였다. 이를 위하여 SDN으로 유입되는 트래픽을 정상적인 트래픽과 유해 트래픽으로 분류하고, 유해 트래픽은 전달되지 않도록 하여 NFV 기반의 SDN 보안 성능을 향상시키고, 정상적인 트래픽과 유해 트래픽을 분류하는데 패턴을 적용하여 트래픽을 구분한다. 본 논문은 이러한 연구를 위하여 SDN을 기반의 테스트 네트워크를 구축하였다.

Jung Yun Kim received the bachelor's degree in the Department of Mechanical Engineering from the Hoseo University in 2001. He received the M.S. degree in the Department of Information Network Management Engineering form Kyunghee University. He completed the Ph.D. course in the Department of Computer Science and Engineering from Kyunghee university in 2008. He has been a senior researcher at ncodi since 2016. His current research interests include IoT, internet technology, network QoS/QoE, traffic management, wireless communication, and network security. He is a member of the KKITS

E-mail address: inokyuni@khu.ac.kr

www.kci.go.kr