KISM Smart Media Journal
Vol. 4, No. 4 pp. 24~32 ISSN:2287-1322

SDN을 위한 샘플링 기반 네트워크 플러딩 공격 탐지/방어 시스템

(Sampling based Network Flooding Attack Detection/Prevention System for SDN)

이윤기*, 김승욱*, 부 둑 티엡*, 김경백**
(Yungee Lee, Seung-uk Kim, Tiep Vu Duc, Kyungbaek Kim)

요 약

최근 SDN은 데이터센터 네트워크로 활발히 사용되고 있으며, 그 사용범위를 점진적으로 늘려나가고 있다. 이러한 새로운 네트워크 환경 변화와 함께, 네트워크 보안시스템을 SDN 환경 상에서 구축하는 연구들이 진행되고 있다. 특히 OpenFlow Switch의 포트를 통과하는 패킷들을 지속적으로 관찰함으로써 네트워크 플러딩 공격 등을 탐지하기 위한 시스템들이 제안되었다. 하지만 다수의 스위치를 중앙집중형 컨트롤러에서 관리하는 SDN의 특성상 지속적인 네트워크 트래픽 관찰은 상당한 오버헤드로 작용할 수 있다. 이 논문에서는 이러한 지속적인 네트워크 트래픽 관찰에 따른 오버헤드를 줄이면서도 네트워크 플러딩 공격을 효과적으로 탐지 및 방어 할 수 있는, 샘플링 기반 네트워크 플러딩 공격 탐지 및 방어 시스템을 제안한다. 제안된 시스템은 네트워크 트래픽을 주어진 샘플링 조건에 맞추어 주기적으로 관찰하고, 샘플링 패킷들을 분석하여 네트워크 플러딩 공격을 탐지하며, 탐지된 공격을 OpenFlow Switch의 플로우 엔트리관리를 통해 능동적으로 차단하다. 네트워크 트래픽 샘플링을 위해 sFlow agent를 활용하고, 샘플링된 패킷 정보를 소프트웨어적으로 분석하여 공격을 탐지하기 위해 오픈소스 기반 IDS인 snort을 사용하였다. 탐지된 공격의 자동화된 방어 기작의 구현을 위해 OpenDaylight SDN 컨트롤러용 어플리케이션을 개발하여 적용하였다. 제안된 시스템은 OVS (Open Virtual Switch)를 활용한 로컬 테스트베드 상에서 그 동작을 검증하였고, 다양한 샘플링 조건에 따른 제안된 시스템의 성능 및 오버헤드를 분석하였다.

■ 중심어 : 소프트웨어 정의 네트워크, 네트워크 플러딩 공격, 공격 탐지 방어 시스템

Abstract

Recently, SDN is actively used as datacenter networks and gradually increase its applied areas. Along with this change of networking environment, research of deploying network security systems on SDN becomes highlighted. Especially, systems for detecting network flooding attacks by monitoring every packets through ports of OpenFlow switches have been proposed. However, because of the centralized management of a SDN controller which manage multiple switches, it may be substantial overhead that the attack detection system continuously monitors all the flows. In this paper, a sampling based network flooding attack detection and prevention system is proposed to reduce the overhead of monitoring packets and to achieve reasonable functionality of attack detection and prevention. The proposed system periodically takes sample packets of network flows with the given sampling conditions, analyzes the sampled packets to detect network flooding attacks, and block the attack flows actively by managing the flow entries in OpenFlow switches. As network traffic sampler, sFlow agent is used, and snort, an opensource IDS, is used to detect network flooding attack from the sampled packets. For active prevention of the detected attacks, an OpenDaylight application is developed and applied. The proposed system is evaluated on the local testbed composed with multiple OVSes (Open Virtual Switch), and the performance and overhead of the proposed system under various sampling condition is analyzed.

■ keywords : Software Defined Network, Network Flooding Attack, Attack Detection Prevention System

* 학생회원, 전남대학교 전자컴퓨터공학부
** 정회원, 전남대학교 전자컴퓨터공학부
이 논문은 한국정보화진흥원(NIA)의 미래네트워크연구시험망(KOREN) 사업 지원과제의 연구결과로 수행되었음(15-951-00-001).

접수일자 : 2015년 12월 02일
수정일자 : 2015년 12월 27일
게재확정일 : 2015년 12월 28일
교신저자 : 김경백, e-mail : kyungbaekkim@jnu.ac.kr

Ⅰ. 서 론

소프트웨어 정의 네트워크, SDN (Software Defined Network)은 소프트웨어를 기반으로 트래픽 조절 등 망 최적화를 수행하고 특정 하드웨어 제조사가 아닌 소프트웨어 애플리케이션 업체들과 협력하여 다양한 네트워크 기능을 제공함으로써, 네트워크 인프라 관리 측면에서 장비 제조사의 의존도를 낮추고 관리 비용을 절감하는 등의 장점을 가진다. 이러한 장점에 힘입어 소프트웨어 정의 네트워크는 최근 네트워크 인프라 효율적 활용 및 새로운 네트워크 응용 서비스를 적용하고자 하는 네트워크 사업자 및 클라우드 사업자들에게 각광받고 있다.

소프트웨어 정의 네트워크는 네트워크 장비의 데이터 영역(Data Plane)과 제어 영역(Control Plane)을 분리하고, 네트워크 데이터 영역의 동작 및 기능을 정의할 수 있는 제어 영역의 인터페이스를 오픈 API 형태로 외부에 제공하여, 이러한 오픈 API를 활용해 작성된 소프트웨어로 다양한 네트워크 경로 설정 및 제어 등을 수행 할 수 있다 [1]. 이러한 능동적인 네트워크 장비의 동작 및 기능 설정 등을 통해 새로운 네트워크 활용법들이 연구되고 있다. 본 논문에서는 소프트웨어 정의 네트워크를 활용해 보안 시스템을 구축하는 연구 방향에 초점을 맞추어 접근하였다.

최근 보안 문제는 네트워크 분야에서 그 중요성이 점차 증가하고 있으며, 임의의 공격자로부터 서버와 같은 주요 시스템을 공격하여 보안을 위협하는 문제해결을 위한 연구들이 진행되고 있다. 이러한 네트워크 공격 중 가장 단순하면서도 대표적인 공격으로 서비스 거부 공격, DoS (Denial of Service)을 생각할 수 있다. 서비스 거부 공격은 방대한 양의 네트워크 패킷을 임의로 생성하여 공격하고자 하는 목표 머신에 전송함으로서, 목표 머신의 통신 프로토콜의 동작을 마비시키거나, 목표 머신의 컴퓨팅 리소스 및 네트워크 리소스를 고갈시켜, 공격을 당하는 머신 및 네트워크가 제대로 된 동작을 할 수 없게 만든다. 이러한 공격은 그 유형이 나날이 다양해지고, 기법이 발전하고 있어 탐지와 방어에 어려움이 있다 [7].

소프트웨어 정의 네트워크를 활용할 경우 중앙 집중적인 형태로 스위치를 지나가는 패킷의 정보를 수집하고, 수집된 패킷정보를 기반으로 공격을 탐지하면, 보다 능동적인 대응이 가능하다. 최근 소프트웨어 정의 네트워크를 활용한 서비스 거부 공격 탐지 및 방어 시스템에 대한 연구가 진행되어 왔다 [2,3,4,5,6]. 하지만, 초기 연구에서는 스위치의 트래픽 또는 패킷의 정보를 지속적으로 관찰하여 네트워크 공격을 탐지하는 연구들이 있었으나[2,3] 이러한 연구들의 문제는 중앙 집중 제어 소프트웨어에 무리한 부담이 될 수 있다는 것이다. 이러한 지속적 모니터링에 따른 부담을 줄이기 위해 sflow를 활용한 패킷 샘플링 기반의 네트워크 공격 탐지 및 방어 기법들이 제안되었다 [4,5,6]. 이들 연구는 주로 sFlow를 활용해 스위치를 통과하는 TCP SYN 패킷을 샘플링하고, 샘플링된 TCP SYN 패킷 정보를 기반으로 네트워크 플로우의 초당 TCP SYN 패킷 개수를 세어 임의의 임계값을 넘으면 TCP SYN 플러딩 공격으로 탐지한다. 하지만 이들 연구는 주로 TCP SYN 플러딩 공격 탐지 기법에 초점을 맞추고 있어, 다른 형태의 네트워크 플러딩 공격 탐지 및 탐지된 공격에 대한 자동화된 방어 기법에 대한 설명이 부족하다. 또한, 샘플링 기법의 설정에 따른 공격 탐지 및 방어 시스템의 성능 분석이 부족하다.

이 논문에서는 SDN 환경에서 다양한 형태의 네트워크 플러딩 공격을 탐지하고 자동으로 방어하기 위한 샘플링 기반 네트워크 플러딩 공격 탐지 및 방어 시스템을 제안한다. 제안된 시스템은 네트워크 트래픽을 주어진 샘플링 조건에 맞추어 주기적으로 관찰하고, 샘플링 패킷들을 분석하여 네트워크 플러딩 공격을 탐지하며, 탐지된 공격을 OpenFlow Switch의 플로우 엔트리관리를 통해 능동적으로 차단하다. 네트워크 트래픽 샘플링을 위해 sFlow agent를 활용하고, 샘플링된 패킷 정보를 소프트웨어적으로 분석하여 다양한 형태의 네트워크 플러딩 공격을 탐지하기 위해 오픈소스 기반 소프트웨어 IDS(Intrusion Detection System)인 snort을 사용하였다. 탐지된 공격의 자동화된 방어 기작의 구현을 위해 OpenDaylight SDN 컨트롤러용 어플리케이션을 개발하여 적용하였다. 또한 탐지된 공격들의 정보는 데이터베이스에 저장하여 공격 방어 어플리케이션에 활용할 수 있도록 하고, 또한 향후 네트워크 플러딩 공격의 유형 분석 및 동향 분석 등에 활용하기 쉽도록 하였다.

제안된 시스템의 성능 검증을 위해 OVS(Open Virtual Switch)로 구성된 로컬 테스트베드를 구축하고 제안된 시스템을 적용한 후, 임의의 노드 간에 TCP SYN, UDP, ICMP를 악용한 네트워크 플러딩 공격을 발생시켰을 때의 시스템 동작 및 네트워크 상황을 확인하였다. 로컬 테스트베드상에서의 검증을 통해 제안된 시스템이 성공적으로 동작하는 것을 확인하였다. 또한, Mininet을

이용한 가상 네트워크 상에서 제안된 시스템을 적용하여, sFlow agent의 샘플링 조건 및 네트워크 토폴로지에 따른 제안된 시스템의 동작여부 및 오버헤드를 분석하였다.

이 논문의 순서는 다음과 같다. 2장에서는 제안된 시스템과 관련된 배경 지식인 네트워크 플러딩 공격, sflow, snort, 그리고 OpenDayLight에 대해 기술하고, 3장에서는 제안된 시스템의 구조 및 동작 원리에 대해 기술한다. 4장에서는 제안된 시스템의 동작 검증을 위한 로컬 테스트베드 구축에 대한 설명 및 검증 결과를 기술하고, 샘플링 조건에 따른 시스템 오버헤드 및 성능에 대한 분석을 수행한다. 마지막으로 5장에서 본 논문의 결론을 기술한다.

Ⅱ. 관련 배경 지식

1. 네트워크 플러딩 공격

최근 인터넷 보안 체계 공격 경향은 시스템 또는 네트워크 자원을 공격 대상으로 하여, 사용 가능한 자원을 모두 소비하여 사용자가 실제 사용해야 하는 자원을 사용할 수 없게 하는 서비스 거부 공격(Denial of Service)이다. 서비스 거부 공격 중 네트워크 리소스를 고갈하기 위한 네트워크 플러딩 공격은 대량의 의미 없는 패킷을 생성하여 네트워크의 대역폭을 고갈 시켜 정상적인 패킷들이 제대로 전달되는 것을 막는다. 이러한 네트워크 플러딩 공격은 다수의 공격자가 공격을 수행할 수 있고 공격자마다 TCP, UDP, ICMP등의 다양한 형태의 패킷을 공격에 이용하기 때문에, 공격자의 추적은 물론이고 공격 탐지와 방어에 어려움이 있다. 대표적인 네트워크 플러딩 공격으로는 TCP SYN Flooding 공격, UDP Flooding 공격, ICMP Flooding 공격이 있다.

TCP SYN Flooding 공격은 TCP (Transmission Control Protocol)를 이용하는 신뢰성 연결을 맺기 위해 수행되는 3 way handshaking 해진 규칙에서 필요한 SYN 패킷을 임의로 생성하여 공격 대상 호스트에 대량으로 전송하는 공격이다. 이 공격은 TCP 취약점을 악용하는 것으로, TCP SYN 패킷을 받은 호스트는 SYN+ACK 패킷으로 응답 후 ACK를 기다리는 “Half Open” 상태가 되어 대기 상태에 머무른 후 일정 시간(75초) 후에 다음 요청이 오지 않으면 해당 연결을 초기화 하게 되는데, 초기화하기 전의 Half Open상태의 연결 정보는 메모리 공간인 백 로그 큐 (Backlog Queue)에 계속 쌓여 호스트의 자원을 낭비하게 되고, 백 로그 큐가 다 찰 경우 추가적인 연결 요청을 거부하게 된다.

UDP Flooding 공격은 대량의 UDP(User Diagram Protocol) 패킷을 공격 대상 호스트의 네트워크 자원을 소모시키는 공격이다. UDP는 비연결형 서비스로 대표적인 응용 서비스는 TFTP, SNMP, 실시간 인터넷 방송들이 있다. 비 연결성 및 비 신뢰성 특성 때문에 UDP 패킷은 source address와 source port를 변조하기 쉬워서 임의의 호스트에 과다한 트래픽을 전송하기 용이하다.

ICMP Flooding 공격은 IETF RFC 792에 정의된 ICMP (Internet Control Message Protocol)의 특징을 악용하여 대량의 패킷을 공격 대상에게 전송하여 네트워크 과부하를 초래하는 공격이다. ICMP는 호스트간 혹은 호스트와 라우터간의 에러상태 혹은 상태 변화를 알려주고 요청에 응답을 하는 기능을 담당하고, 활성화된 서비스나 포트가 필요하지 않는 유일한 프로토콜이다. 이러한 특징을 악용하여 ICMP echo request등을 통해 대량의 패킷들을 발생시킬 수 있다.

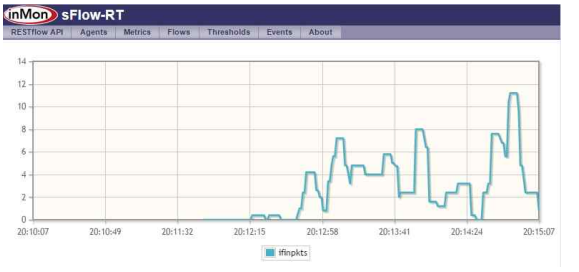

그림 1. sFlow-RT를 이용한 트래픽 확인

2. sFlow

네트워크 장비에서 모든 패킷을 dump하여 일일이 모니터링 하는 것은 사실상 불가능 하다. 이를 위해 패킷의 헤더 정보만을 분석하는 방법을 고안하였다. InMon사에 의해 개발된 sFlow는 보다 빠르게 네트워크를 모니터링 하는 도구로, 다양한 종류의 네트워크 장비에 설치될 수 있는 sFlow agent를 이용하여 패킷을 샘플링 및 가공하여 sFlow 프로토콜 형태로 지정된 sFlow Collector에 전송 한다. sFlow Collector는 수신한 패킷을 처리하여 사용자에게 정보를 제공하는데, 이 때 sFlow Collector에 위치한 SW에 따라 패킷의 이용이 달라진다. 예를 들어 sFlow Collector에 모니터링 SW인 sFlow-RT를 위치시키면 네트워크를 모니터링 할 수 있으며, sflow toolkit을 위치시키면 샘플링 된 패킷정보를 요약하고

정리하여 다른 프로그램들이 사용할 수 있게 변환하고 전달하는 역할을 할 수 있다. 그림 1은 Collector에 sFlow-RT를 위치 시켜 모니터링 한 모습으로 시간에 따라 변하는 트래픽의 양을 그래프를 통해 보여주고 있다.

sFlow 설정을 통해 sFlow Collector를 지정하거나, 샘플링 주기, 비율 등을 조절할 수 있다. sFlow는 트래픽을 샘플링하여 모니터링 하기 때문에 cpu에 걸리는 부하를 줄여주는 장점이 있어 네트워크 전체의 흐름이나 DDos와 같은 대규모 네트워크 공격을 파악하는데 효과적으로 사용 될 수 있다.

3. snort

snort는 오픈소스 기반의 네트워크 침임 탐지 및 방어 시스템으로, IP network위에서 전송과정중의 패킷을 검사하여 공격여부를 판단하는 signature, protocol and anomaly-base 탐지 방식으로 동작한다. snort는 스니퍼, 패킷로거, 네트워크 공격 탐지 세가지 모드로 운영될 수 있다. 스니퍼 모드에서 snort는 네트워크 패킷을 읽어 들이고 콘솔 창에 해당 정보를 나타낸다. 패킷로거 모드에서 snort는 패킷의 정보를 로깅해 디스크에 남긴다. 네트워크 공격 탐지 모드에서는 네트워크 트래픽을 모니터링하고 사용자가 정의한 룰에 따라 네트워크 패킷을 분석하여 대부분의 네트워크 공격을 탐지할 수 있다. 네트워크 공격 탐지 모드에서는 사용자가 정의한 룰에 해당하는 패킷이 있을 경우 경고 (Alert)를 발생 시켜 공격이 탐지되었음을 판단한다. 여기서 룰(Rule)은 규칙 문법이 존재하여, 환경에 맞게 커스터마이징 할 수 있어 다양한 공격을 탐지할 수 있는 Rule 공유가 활발히 진행되고 있다. 이를 통해 새롭게 발생되는 네트워크 공격에 대해서도 신속한 대응이 가능하다.

네트워크 공격 탐지 모드에서 동작할 때, 분석할 패킷 정보는 꼭 네트워크 장비에서 읽어 들일 필요는 없다. 즉, 저장된 패킷정보를 파일로 받아도 해당 패킷이 네트워크 공격의 원인이 되는지는 판단이 가능하다. 이러한 파일 기반의 패킷 정보 분석 기능은, sFlow에서 수집된 패킷 정보를 파일로 넘겨받아 공격 징후를 탐지하는 것을 가능하게 한다.

4. OpenDayLight SDN Controller

OpenDayLight는 Open SDN의 환경에서 실행되는 확장성, 모듈화가 뛰어난 멀티 프로토콜 제어기이다. OpenDayLight에서는 모델 기반의 추상적인 서비스 플랫폼과 South-bound-API를 제공해 사용자가 직접 어플리케이션을 작성하여 네트워크 장치를 조절할 수 있도록 한다. OpenDayLight는 표준화된 모델을 사용함으로써 플랫폼간 이식성이 높다. OpenDayLight는 다양한 네트워크 장비 제어 프로토콜을 플러그인 형태로 지원하기 때문에 OpenFlow 표준 장비 뿐 만이 아닌 다른 제조업체 장비 또한 제어가 가능하다. 또한 재부팅 없이 동적으로 어플리케이션 추가/삭제가 가능한 OSGI 프레임워크를 사용함으로써 모듈화와 확장성을 갖추었으며, 필요한 어플리케이션을 직접 개발하여 추가함으로써 사용자가 원하는 형태로 스위치를 제어할 수 있다. 현재 Opendaylight 프로젝트에는 시스코를 비롯해 IBM, 시트릭스, 에릭슨, 주니퍼네트웍스, 레드햇, 델, 마이크로소프트등 IT업계의 쟁쟁한 선두주자들이 참여하고 있다.

Ⅲ. 샘플링 기반 공격 탐지 방어 시스템

제안하는 시스템은 SDN환경에서 네트워크 플러딩 공격 탐지 방어 시스템으로, 네트워크 장비를 통과하는 네트워크 패킷을 샘플링하기 위한 샘플링 에이전트(Sampling Agent), 샘플링되는 패킷정보를 수집하기 위한 샘플링 컬렉터(Sampling Collector), 수집된 정보를 기반으로 네트워크 플러딩 공격을 판별하고 탐지하기 위한 공격 탐지기(Attack Detector), 탐지된 공격 이벤트 정보를 저장 및 관리하기 위한 이벤트 데이터베이스(Event Database), 그리고 탐지된 공격 이벤트 정보를 기반으로 네트워크 장비를 동적으로 조절하여 네트워크 플러딩 공격을 차단 하는 공격 방어기(Attack Defender)로 구성된다. 전체적인 시스템의 구조는 그림 2와 같다. 이러한 구성 요소들은 모두 하나의 머신에 위치하여 운용할 수 도 있지만, 시스템 확장성 및 규모를 고려해 분산 환경에서 운용하는 것도 가능하다.

샘플링 에이전트는 주어진 샘플링 조건에 따라 네트워크 스위치를 통과 하는 패킷을 샘플링 한다. 이를 위해 우리는 sFlow agent를 활용하였다. sFlow agent 설치 시 sampling rate와 polling rate를 설정함으로써 얼마나 자주 패킷을 샘플링 할지를 설정할 수 있다. sFlow agent는 주어신 샘플링 조건에 따라 패킷의 헤더를 수집하고 이를 샘플링 컬렉터로 전달 한다.

샘플링 컬렉터는 샘플링되는 패킷들을 수집하여 하나의 파일 및 데이터 구조로 정리하는 역할을 한다. 이

를 위해 우리는 sFlow toolkit을 활용하였다. sFlow tool kit은 다수의 sFlow agent에서 보내오는 샘플 패킷들을 정리하여 하나의 데이터 구조로 만들고 이를 다양한 형태로 출력할 수 있다. 제안된 시스템에서 sFlow toolkit은 정리된 샘플 패킷 정보를 TCP dump스타일로 정리해서 출력하고 이를 공격 탐지기가 사용할 수 있도록 전달한다.

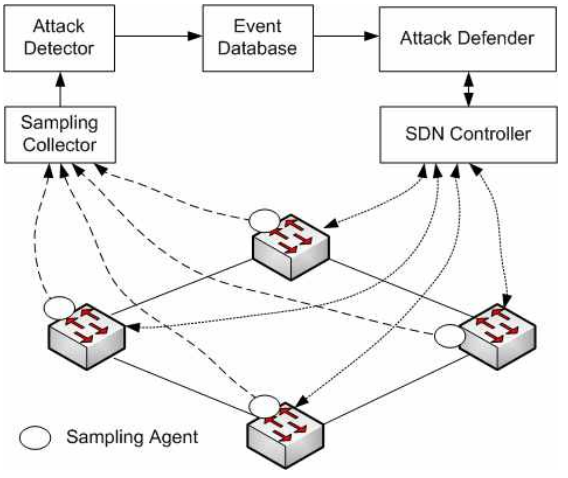

그림 2. 샘플링 기반 공격 탐지 방어 시스템 구조

공격 탐지기는 수집된 샘플 패킷 정보를 기반으로 네트워크 플러딩 공격이 있는지를 탐지하고, 공격이 탐지될 경우 해당 이벤트를 기록하도록 한다. 이를 위해 우리는 snort를 활용하였다. snort는 네트워크 패킷 정보를 분석하여 지정된 룰에 해당하는 네트워크 플러딩 공격 패턴이 발견될 경우 네트워크 플러딩 공격 이벤트가 발생하였음을 기록한다. 이때, snort가 공격 탐지 기능과 이벤트 저장 기능을 동시에 할 경우 성능 저하가 발생하기 때문에, snort의 출력 및 저장 기능을 대신하는 이벤트 데이터베이스 역할을 하기 위한 Barnyard를 활용하였다. 이벤트 데이터베이스는 탐지되는 네트워크 플러딩 공격을 저장하고 관리하며, 공격 이벤트 정보를 공격 방어기에게 넘겨주는 역할을 한다.

네트워크 플러딩 공격을 능동적으로 차단하기 위한 공격 방어기는 OpenDayLight 어플리케이션으로 구현되었다. 이 공격 방어기는 이벤트 데이터베이스에 저장된 공격 정보를 분석하여 현재 네트워크 토폴로지에서 공격자의 위치를 파악하고 해당 공격을 막기 위한 네트워크 제어 룰을 해당 네트워크 장치에 알리는 역할을 한다. 공격 방어기가 수행하는 공격 자동 방어를 위한 룰 생성 알고리즘은 그림 3과 같다.

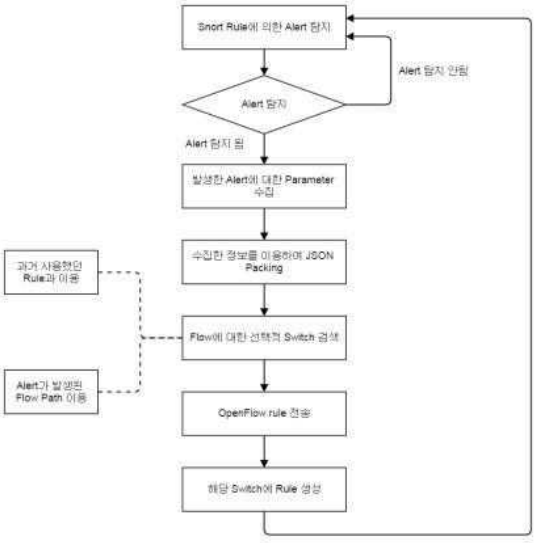

그림 3. 공격 방어기의 룰 생성 알고리즘

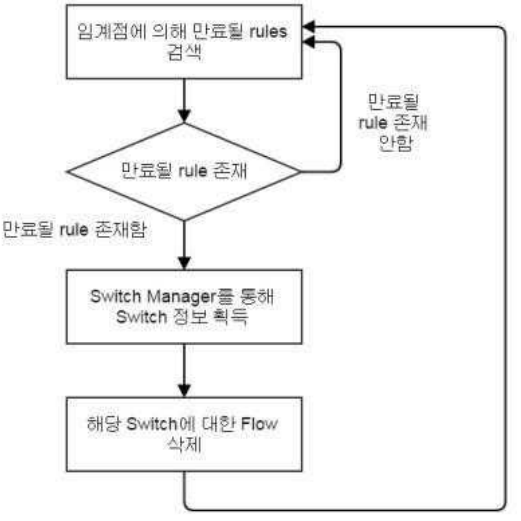

그림 4. 공격 방어기의 룰 삭제 알고리즘

공격 방어기는 네트워크 플러딩 공격 방어를 위한 룰을 생성하고 스위치에 설치하게 되는데, 이러한 공격 방어용 룰은 스위치의 메모리 자원을 사용하게 된다. 만약 사용하지 않는 공격 방어용 룰이 많아지면 스위치의 메모리 자원을 쓸데없이 소모하게 된다. 이러한 문제를 미연에 방지하기 위해, 공격 방어기는 생성된 공격 방어 룰을 관리하고 해당 룰의 유효기간을 설정해 만료된 룰은 네트워크 장치에서 삭제하여야 한다. 이러한 룰 삭제를 위한 알고리즘은 그림 4와 같다.

그림 5. 로컬 테스트 베드 구축 환경

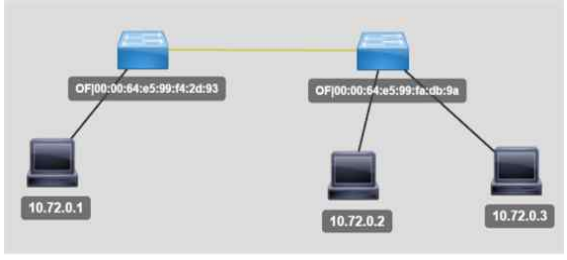

그림 6 로컬 테스트베드 네트워크 구성도

Ⅳ. 성능 평가

1. 네트워크 플러딩 공격 탐지 방어 기능 검증

제안된 샘플링 기반 네트워크 플러딩 공격 탐지 방어 시스템의 기능을 검증하기 위해, 로컬 테스트 베드를 구축하고 네트워크 플러딩 공격을 에뮬레이션하여 제안된 시스템의 기능을 검증하였다.

로컬 테스트베드는 두 개의 OVS (Open Virtual Switch)와 공격 탐지 방어 시스템 운용을 위한 머신, 그리고 세 개의 호스트 머신들로 구성된다.

로컬 테스트 베드 구성환경은 그림 5와 같고, 로컬 테스트베드를 SDN 컨트롤러에서 바라보는 네트워크 구성도는 그림 6과 같다. 호스트 머신들은 각각 Windows XP, Windows 8, Ubuntu 14.04를 운영체제로 사용한다. OVS 머신 및 공격 탐지 방어 시스템 운용머신은 Ubuntu 14.04를 운영체제로 사용하고, OpenDayLight는 Hygrogen 1.0 Base를 사용하였다. 가상의 네트워크 플러딩 공격을 수행하기 위해 hping3 툴을 활용하여 TCP SYN, UDP, ICMP 패킷을 대량으로 임의의 호스트에 전송하도록 한다.

그림 7. 스위치에 설치된 네트워크 공격 방어 룰

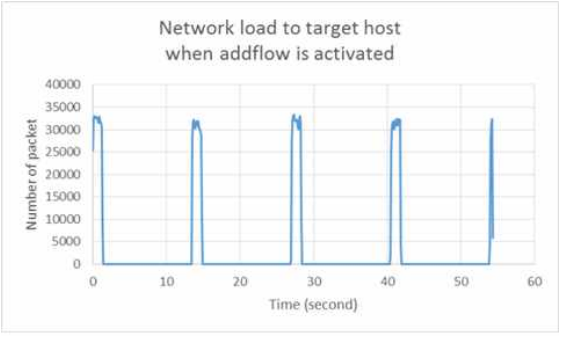

그림 8. 공격대상 호스트 네트워크 트래픽

네트워크 플러딩 공격 탐지를 위한 snort Rule은 다음과 같이 설정하여, 대략 1초에 7번 이상의 동일한 패턴을 가지는 패킷이 들어올 경우 네트워크 플러딩 공격으로 감지하여 해당 공격에 대한 방어 기작을 수하도록 한다.

```
#TCP - SYN
alert tcp any any -> any 80 (flags: S; msg:"Alert SYN flood Attack"; flow: stateless; threshold: type both, track by_src, count 70, seconds 10; sid:10001;rev:1;)
# UDP
alert udp any any -> any 80 (msg:"UDP Attack"; threshold: type both, track by_src, count 70, seconds 10; sid:2345702; rev:1; )
#ICMP
alert icmp any any -> $HOME_NET any (itype: 8; msg:"ICMP Attack"; threshold: type both, track by_src, count 70, seconds 10; sid:10000001; rev:001;)
```

위와 같은 설정을 가진 네트워크 플러딩 공격 탐지 방어 시스템의 동작 검증을 위해, 네트워크 플러딩 공격이 탐지되면 공격 방어 룰을 스위치에 설치하고, 10초 후 스위치에서 룰을 제거하도록 공격 방어기 코드를 변경하여 테스트를 수행하였다. 네트워크 공격은 10.72.0.2에서 시작하고 10.72.0.1을 공격 대상으로 선정하고, TCP SYN 공격을 수행하였다. 네트워크 플러딩 공격이 탐지될 경우 공격 방어기는 OpenDayLight에 생성된 공격 방어 룰을 전달하고, 이는 해당 스위치에 설치된다. 설치된 방어 룰에 해당하는 Flow Table정보는 그림 7과 같다.

설치되는 방어 룰이 정상적으로 동작하는 지를 확인

하기 위해 공격대상인 10.72.0.1 머신의 네트워크 트래픽를 하였고, 그 결과는 그림 8과 같다. 네트워크 플러딩 공격이 시작된 후 약 2초 후 네트워크 공격 방어 룰이 스위치에 설치되고 10.72.0.2에서 전송되는 네트워크 플러딩 공격 패킷들이 스위치에서 모두 버려진다.

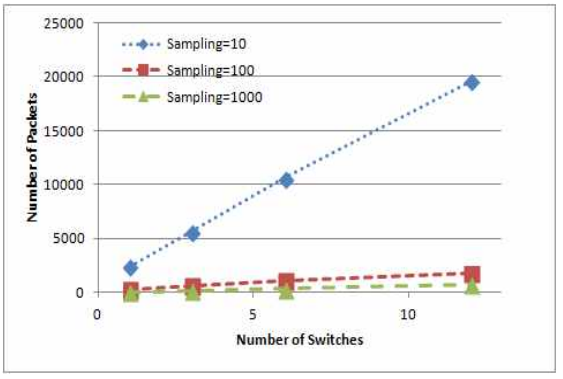

그림 9. Sampling Rate와 스위치 개수에 따른 샘플 패킷량

이에 따라, 네트워크 플러딩 공격 패킷이 더 이상 10.72.0.1로 흘러 들어오지 않는 것을 확인 할 수 있다.

또한, 테스트를 위해 네트워크 방어 룰은 10초 뒤에 만료되어 스위치에서 삭제된다. 스위치에서 룰이 삭제되면, 다시 네트워크 플러딩 공격에 공격대상 호스트는 영향을 받는 것을 확인 할 수 있고, 이는 새로운 공격으로 감지되어 다시 공격 방어 기작이 동작함을 확인할 수 있다.

이러한 네트워크 플러딩 공격 탐지 방어 기능은 TCP SYN flooding 공격, UCP flooding 공격, ICMP flooding 공격에 모두 잘 적용되는 것을 확인 하였다.

2. 샘플링 조건에 따른 성능 분석

제안된 시스템은 스위치를 통과하는 패킷을 샘플링하는 조건에 따라 그 성능이 달라질 수 있다. 샘플링 주기가 짧을수록 샘플링 에이전트는 보다 많은 패킷을 수지하게 되고, 샘플링 컬렉터에 전송되는 데이터의 양이 많아지게 된다. 하지만 샘플링 주기를 늘릴 경우 일정 수준 이하의 패킷량을 가지는 네트워크 플러딩 공격을 탐지하지 못할 수도 있다. 이러한 샘플링 조건에 따른 제안된 시스템의 성능 분석을 위해 스위치에 추가되는 sFlow agent의 sampling rate와 polling rate와 가상 공격의 패킷량을 변경시켜가며 제안된 시스템의 성능을 확인하였다.

그림 9는 Sampling rate 및 스위치의 개수를 변경시키면서 확인한 샘플링 컬렉터에 전달되는 패킷의 수를 나타낸다. 이때 네트워크 플러딩 공격은 1초에 1000번씩 패킷을 발송하도록 설정하였다. Sampling rate가 작을수록 보다 많은 양의 패킷이 샘플링 컬렉터로 전달되는 것을 알 수 있다. 그리고 네트워크 스위치의 개수가 증가함에 따라 샘플링 컬렉터로 전달되는 패킷의 수는 선형적으로 증가하는 것을 확인할 수 있다.

그림 10은 Sampling Rate를 100으로 고정시키고, Polling rate와 스위치의 개수를 변경시키면서 확인한 샘플링 컬렉터에 전달되는 패킷의 수를 나타낸다. 이때 네트워크 플러딩 공격은 1초에 1000번씩 패킷을 발송하도록 설정하였다. Polling rate가 작아질수록 보다 많은 양의 패킷이 샘플링 컬렉터로 전달되는 것을 확인 할 수 있다.

표1. Sampling rate와 attack rate의 변화에 따른 공격 탐지 기능 가능 여부

Sampling Rate / Attack Rate	10	100	1000	10000	20000
10	X	X	X	X	X
100	O	X	X	X	X
1000	O	O	X	X	X
33333	O	O	O	X	X
100000	O	O	O	O	X
200000	O	O	O	O	O

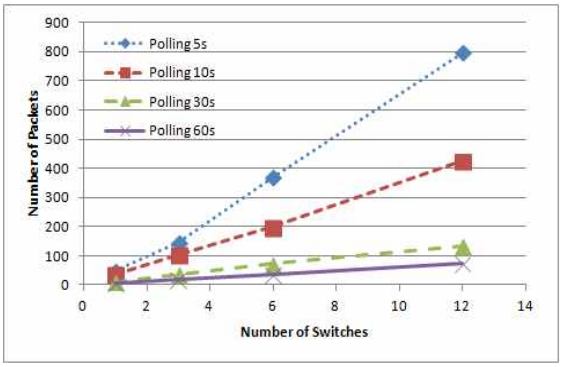

그림 10. Polling과 스위치 개수에 따른 샘플 패킷량

샘플링 에이전트의 Sampling rate 및 Polling rate를 늘임으로써 샘플링 컬렉터로 보내지는 네트워크 오버헤드는 줄일 수 있으나, 이는 네트워크 공격 탐지 기능에 문제를 일으킬 수 있다. 이점의 확인을 위해, Sampling

rate와 네트워크 플러딩 공격 rate를 변경시켜가며 네트워크 공격 탐지 기능의 동작을 확인하고 그 결과를 표 1에서 나타낸다. Sampling rate가 100일 경우 일초에 1000번의 패킷을 보내는 공격부터 확인이 가능하고, 이보다 더 느리게 패킷을 보내는 공격에 대해서는 탐지가 되지 않는 것을 확인할 수 있었다.

따라서, 제안된 샘플링 기반 네트워크 플러딩 공격 탐지 방어 시스템을 운용할 때는, 방지하고자 하는 네트워크 플러딩 공격의 특성과 샘플링 조건에 따른 오버헤드와 성능의 문제를 동시에 고려하여 샘플링 에이전트의 Sampling rate와 Poling rate를 적절한 값으로 설정하는 것이 중요하다.

Ⅴ. 결 론

SDN환경은 네트워크 서비스들의 네트워크 장비에 대한 의존도를 낮추고, 네트워크 운영비용을 줄일 수 있는 장점을 가지고 점진적으로 그 영향력을 늘여가고 있다. 이러한 SDN환경에서 효과적인 보안 시스템의 구축은 흥미로운 주제이다. 이 논문에서는 SDN환경에서 네트워크 플러딩 공격을 자동으로 탐지 및 방어하기 위한 샘플링 기반 네트워크 플러딩 공격 탐지 방어 시스템을 제안하였다. 제안된 시스템의 검증을 위해 sFlow, sFlow Toolkit, Snort, OpenDayLight로 구성된 시스템 프로토타입을 구현하였고, OVS로 구성된 로컬 테스트베드에서 네트워크 플러딩 공격 탐지 및 방어 기능을 검증하였다. 또한 샘플링 조건에 따른 제안된 시스템의 성능을 분석하였다.

현재, 제안된 시스템은 sFlow agent를 이용해 스위치를 통과하는 패킷의 헤더정보를 사용하여 네트워크 플러딩 공격을 탐지하도록 하고 있다. 따라서, TCP/IP 레이어에 해당하는 프로토콜을 악용하는 네트워크 플러딩 공격은 탐지 및 방어가 가능하다. 이보다 상위 레이어에 위치하는 어플리케이션 프로토콜을 악용하는 플러딩 공격에 대한 탐지 및 방어를 하기 위해서는 패킷의 헤더 뿐만 아니라 패킷의 데이터를 동시에 샘플링할 수 있는 샘플링 에이전트를 활용해야 한다. 또한, 제안된 시스템에서 생성되는 공경 방지 룰을 보다 효과적으로 관리하여 스위치에 기록하는 방어 룰의 개수를 줄일 수 있다.

References

[1] 신명기, “ICT 전문가 인터뷰”, *한국정보통신기술협회 TTA 저널*, 제 15호, pp.14-15, 2014

[2] 이광수, 김광조, “SDN에서 Flow 기반 침입 탐지 시스템의 탐지 성능 개선 방법”, *2014년 한국정보보호학회 동계학술대회*, 2014.

[3] R. Braga, E. Mota, and A. Passito, “Lightweight DDoS Flooding Attack Detection using NOX/OpenFlow”, Proceeding of the 35th Annual IEEE Conference on Local Computer Networks, pp. 408-415, 2010.

[4] 하태진, 정치욱, Jargalsaikhan Narantuya, 안남원, 임 혁, 김종원, “sFlow를 이용한 네트워크 공격 탐지 시스템”, *2014년 한국통신학회 하계학술대회*, 2014.

[5] M. Nugraha, I. Paramita, A. Musa, D. Choi, and B. Cho, “Utilizing OpenFlow and sFlow to Detect and Mitigates SYN Flooding Attack”, Journal of Korea Multimedia Society, Vol. 17, No. 8, pp. 988-994, 2014.

[6] 방기현, 최덕재, 방상원, “SDN 환경에서의 목적지 주소별 패킷 샘플링을 이용한 SYN Flooding 공격 방어기법”, 멀티미디어학회 논문지, 18권, 1호, pp. 35-41, 2015

[7] 이종엽, 윤미선, 이 훈, "DoS 공격의 유형 분석 및 탐지 방법“, in KNOM Review, 2004.

[8] Nhu-Ngoc Dao, Junho Park, Minho Park, and Sungrae Cho, "A Feasible Method to combat against DDoS Attack in SDN Network", in Proceedings of ICON 2015, 2015

[9] Zhengyang Xiong. "An SDN-based IPS Development Framework in Cloud Networking Environment", master thesis, 2014

[10] Martin Andreoni Lopez, Otto Carlos M. B. Duarte. "Providing Elasticity to Intrusion Detection Systems in Virtualized Software Defined Networks." ICC 2015, 2015

[11] 이윤기, 김승욱, 김경백, “SDN환경에서 네트워크 플러딩 공격 탐지 및 방어 시스템 구현”, 2015년 한국스마트미디어학회 추계학술대회, pp. 294-297, 2015.

저 자 소 개

이윤기

2011년~현재 전남대학교 전자컴퓨터공학부 학사과정
〈주관심분야 : 소프트웨어 정의 네트워크, 클라우드 시스템〉

김승욱

2011년~현재 전남대학교 전자컴퓨터공학부 학사과정
〈주관심분야 : 소프트웨어 정의 네트워크, 클라우드 시스템〉

Vu Duc Tiep

2012년~현재 전남대학교 전자컴퓨터공학부 석사과정
〈주관심분야 : 소프트웨어 정의 네트워크, 네트워크 기능 가상화〉

김경백 (정회원)

1999년 한국과학기술원 전기 및 전자공학과 학사 졸업
2001년 한국과학기술원 전기 및 전자공학과 석사 졸업
2007년 한국과학기술원 전기 및 전자공학과 박사 졸업
2007년~2011년 University of California Irvine, 박사후 연구원
2012년~현재 전남대학교 전자컴퓨터공학부 조교수
〈주관심분야 : 분산시스템, 미들웨어, 피어투피어/오버레이 네트워크, 소셜 네트워크, SDN〉