SDN환경에서 네트워크 플러딩 공격 탐지 및 방어 시스템 구현

이윤기, 김승욱, 김경백
전남대학교 전자컴퓨터공학부
negee1564@naver.com, kimseungwoog@gmail.com, kyungbaekkim@jnu.ac.kr

Implementation of Network Flooding Attack Detection and Prevention System for SDN

Yungee Lee, Seungwook Kim, Kyungbaek Kim
Department of Electronics and Computer Engineering,
Chonnam National University

요약

최근 SDN은 데이터센터 네트워크로 활발히 사용되고 있으며 그 사용범위를 점진적으로 늘려나가고 있다. 이러한 새로운 네트워크 환경 변화와 함께, 네트워크 보안 시스템을 SDN환경 상에서 구축하는 연구들이 진행중이다. 본 논문에서는 SDN환경에서 네트워크 플러딩 공격을 효과적으로 탐지하고 이에 따른 발빠른 방어를 수행하기 위한 네트워크 공격 탐지 및 방어 시스템의 구축 방법을 소개한다. 본 시스템은 sFlow, Snort 그리고 OpenDaylight SDN 컨트롤러를 이용하여 구성하였다. 네트워크 트래픽의 주기적인 샘플링을 이용한 모니터링을 수행하고 이 샘플 결과를 분석하여 네트워크 플러딩 공격을 탐지한다. 공격이 탐지되면 네트워크 스위치의 플로우 엔트리 관리를 통해 공격을 유발하는 트래픽을 차단함으로써 능동적 방어를 수행한다. 구축된 탐지 및 방어 시스템은 가상 SDN 네트워크 환경상에 적용한 후, 네트워크 플러딩 공격을 발생시킴으로써 시스템의 탐지 및 방어 기능을 검증하였다.

1. 서 론

기존 네트워크 업계에선 네트워크 장비가 하드웨어 기반으로 기술 관리 측면에서 장비제조사의 의존도가 매우 높았다. 그러나 소프트웨어로 정의하는 네트워크인 SDN(Software Define Network) 기술이 개발, 도입 됨으로써, 소프트웨어 기반으로 트래픽 조절 등 망 최적화 관리가 가능해지고, 특정 제조사가 아닌 다양한 애플리케이션 업체들과 협력하는 등 선택의 폭이 넓어지게 되면서 SDN에 관심이 집중되고 있다.

SDN은 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.[1] 소프트웨어를 기반으로 네트워크를 개방화하고 유연하게 만들 수 있어 네트워크를 최적화 하여 관리에 용이하다. 이를 이용해 기존 네트워크의 문제점을 해결하는 솔루션으로 많은 연구가 진행되고 있으며, 본 논문에서는 보안 관련 방향으로 초점을 맞추어 접근해 보았다.

최근에 보안 문제는 네트워크 분야에서 중요성이 점차 증가하고 있으며, 임의의 공격자로부터 서버와 같은 주요 시스템을 공격하여 보안을 위협하는 DDoS(Distributed Denial of Service)와 같은 형태의 공격기법이 대표적이며 이러한 공격은 그 유형이 나날이 다양해지고 발전하고 있어 탐지와 방어에 어려움이 있다. 하지만 SDN을 사용 할 경우 중앙 집중적인 형태로 분리 되어있기 때문에 데이터 플로우를 제어함 으로써 이러한 공격에 대응할 수 있을 것이라 예상된다.

최근 논문들을 살펴보면 Snort를 이용하여 Snort-IDS을 구축해 네트워크 트래픽을 분석 후 공격을 탐지하는 시스템에 관한 논문들을 볼 수 있다. 그러나 공격을 탐지하는 시스템에 비해 그 공격의 방어를 위한 IPS가 추가된 시스템에 관한 내용은 부족하다.

본 논문에서는 SDN환경에서의 Flooding공격을 탐지하고 방어 할 수 있는 시스템을 제안한다. 이 시스템은 Flooding 공격을 샘플링 하여 네트워크 트래픽 모니터링을 위한 sFlow, 공격을 탐지해 IDS의 역할을 수행 할 Snort, 그리고 SDN 환경에서의 장비의 제어를 위한 OpenDayLight로 구성되어 있다.

본 논문의 순서는 다음과 같다. 2장에서는 시스템에 사용된 SW에 대한 설명을 하고, 3장에서는 제시하는 시스템의 구조와 동작에 대해 설명하고, 4장에서는 테스트베드에서 실험한 결과를 가지고 분석한다. 5장에서는 결론 및 향후 연구에 대해서 기술한다.

2. sFlow 와 Snort, OpenDayLight

2.1 InMon사 sFlow

네트워크 장비에서 모든 패킷을 dump하여 일일이 모니터

링 하는 것은 사실상 불가능 하다. 이를 위해 패킷의 헤더 정보만을 분석하는 방법을 고안하였다. InMon사에 의해 개발된 sFlow는 보다 빠르게 네트워크를 모니터링 하기 위한 기술로, 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, Snort toolkit을 위치시키면 Snort가 처리할 수 있게 변환하여 넘겨주는 역할을 하게 할 수 있다. 그림 1은 Collector에 sFlow-RT를 위치 시켜 모니터링 한 모습으로 시간에 따라 변하는 트래픽의 양을 그래프를 통해 보여주고 있다.

(그림 1) Snort-RT를 이용한 트래픽 확인.

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

2.2 Snort

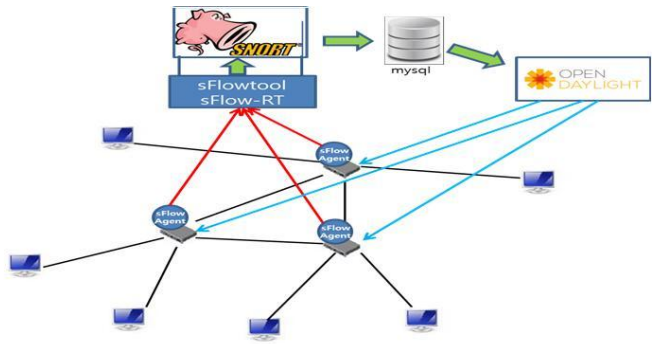
Snort는 네트워크 상에서 패킷의 전송 과정의 패킷을 검사하여 공격 여부를 판별하는 signature, protocol and anomaly-base 탐지방식으로 IDS의 역할을 수행한다.
먼저 네트워크를 돌아다니는 패킷을 잡아내고(sniffing) 재조합 및 가공 후 이를 미리 저장된 네트워크 공격 패턴 규칙(Rule)과 비교를 통해 패킷에 해당 패턴이 있을 경우 경고(Alert)을 발생시켜 공격을 판단한다. 여기서 Rule은 규칙 문법이 존재하여, 환경에 맞게 커스터마이징 할 수

있어 널리 사용되고 있는 NIDS인 만큼 Rule 공유 또한 활발하다. 이를 이용해 새롭게 발생되는 네트워크 공격에 대해서도 신속한 대응이 가능하다.

2.3 OpenDayLight

OpenDayLight는 다양한 네트워크 장비 제어 protocol을 플러그인 형태로 지원 하기 때문에 OpenFlow 표준 장비 뿐만이 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.

3. 네트워크 플러딩 탐지 및 방어 시스템



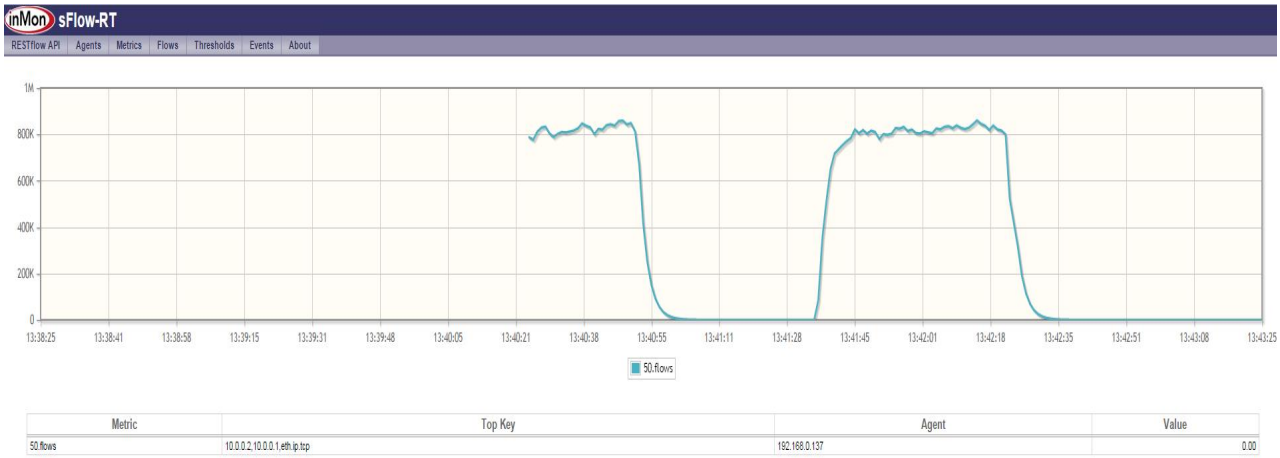
(그림 2) sFlow, Snort, OpenDaylight를 활용한 네트워크 플러딩 탐지 방어 시스템 구조

제안하는 네트워크 플러딩 탐지 및 방어시스템의 전체 구조는 그림 2와 같다. 네트워크 장비에서 sFlow Agent에 의해 패킷을 샘플링 하고 샘플링 된 패킷을 Collector에 보낸다. 이때 Collector에는 Snort-toolkit이 위치하고 있어 들어오는 정보를 Snort가 처리할 수 있는 pcap파일로 포맷을 변환하여 전달하게 된다. Snort는 미리 정의된 탐지 규칙(Rule)을 기반으로 공격 여부를 판별하고, 이를 DB에 저장하게 되는데 Snort가 공격을 탐지하는 기능과 DB에 저장하는 기능을 모두 하게 되면 성능의 저하가 발생하기 때문에 Snort의 출력기능을 대신 출력 해주는 Barnyard를 이용하여 DB에 저장하게 하고, Snort는 탐지기능에 전념하게 하였다.
OpenDayLight에 만들어진 application을 추가하여 Snort-Barnyard에 의해 저장되는 DB를 바라보게 한다. 이후 공격을 알리는 Alert이 들어오게 되면 네트워크 장비의 플로우 테이블을 변경함으로써 Flooding 공격 패킷을 차단하게 된다.
sFlow-RT 모니터링 툴을 이용하여 트래픽량을 실시간으로 관찰하고, application이 동작 했을 때 Flooding 공격 패킷을 차단하는지 여부를 확인할 수 있다.

4. 실험 및 결과 분석

4.1 실험환경

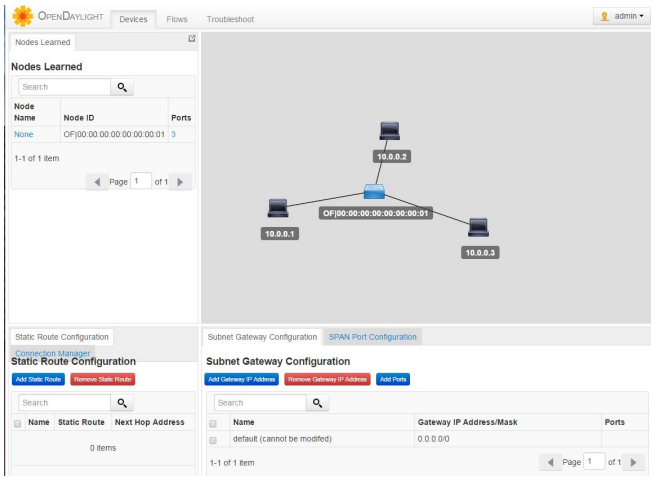
본 논문에서 제안된 시스템을 구현하기위해 다음과 같은 테스트베드를 구성하였다. 시뮬레이션을 위한 데스크탑은 두 개의 VMware muchine VM1과 VM2가 있으며 각각에 Ubuntu 14.04 LTS 운영체제가 설치되어 있다. VM1에 mininet을 이용하여 1개의 스위치와 3개의 호스트를 생성하였다. 1개의 호스트를 공격자로 설정하고 다른 하나의 호스트를 공격을 받는 서버로 설정, 나머지 하나는 일반 사용자



(그림 5) Controller의 Application에 의한 SYN Flooding 차단 확인.

로 설정하였다. 가상의 스위치는 OpenFlow와 sFlow를 지원하는 OpenvSwitch를 사용하였고 VM2에 스위치를 제어하는 컨트롤러로 OpenDayLight의 Hydrogen 1.0 Base 버전을 설치하여 스위치와 연결하였다.

OpenDayLight를 통해 그림 3과 같이 현재 컨트롤러와 연결된 네트워크 토폴로지를 확인 할 수 있다. 스위치에는 sFlow agent를 두고 지나가는 트래픽을 샘플링하여 sFlow collector가 전송받게 설정하였다. sFlow collector로는 VM2의 Snort-toolkit을 위치하였고, collector로 두 SW가 모두 올 수 없어 모니터링 툴 sFlow-RT와 wireshark를 병행하여 사용하였다. VM1의 공격 호스트는 패킷생성 툴인 hping3를 이용하여 서버를 향해 SYN, UDP, ICMP Flooding 공격을 행한다.



(그림 3) OpenDayLight로 확인한 가상네트워크 토폴로지

2) 실험 및 분석

VM1에서 mininet을 이용하여 생성한 스위치와 VM2의 OpenDayLight 컨트롤러를 연결하였다. sFlow agent의 샘플링 비율은 100, 주기는 10으로 하여 지나가는 트래픽의 약 1%를 sFlow collector로 전송하였다. 공격자 호스트에서는 hping3를 이용하여 서버를 향해 SYN, UDP, ICMP Flooding 공격을 수행한다.

Device	Description	IP	Packets	Packets/s
eth0		192.168.0.137	2538	33
s1-eth1		fe80::98e8:66ff:fe94:3d7d	7944271	94621
s1-eth2		fe80::c5e:ddff:feb3:7af0	2506318	30132
s1-eth3		fe80::a83c:1ff:fe9e:bfb5	7	0
s1		fe80::c49:b5ff:feff:d18	1	0

(그림 4) SYN Flood 공격 시 전송 패킷 개수.

그림 4를 보면 알 수 있듯 hping3를 이용하여 SYN Flood 공격 시 초당 약 10만개의 패킷을 전송하는데 스위치 Queuing에 의한 패킷 Drop 현상으로 3만개의 패킷만 전송되었고, 그 중 약 1%를 Collector에 전송하게 된다. (다량의 패킷이 들어올 경우, agent는 패킷 정보를 UDP 패킷으로 한 번에 9개까지 보낼 수 있다.) 이와 같은 SYN Flooding 공격을 탐지하기 위해 Snort에 flag s의 tcp 패킷이 한 포트로 10초안에 70개 이상 들어오면 Alert을 출력하는 네트워크 공격 패턴 규칙(Rule)을 추가하였다. Collector에 위치한 Snort-tookit은 이 패킷을 받아 Snort로 넘겨주게 되고 Rule과 비교하여 공격 여부를 판단하고 출력된 Alert은 DB에 저장된다.

OpenDayLight Controller에는 application을 설치하여 탐지된 공격에 대한 정보를 DB에서 가져와 분석하고 해당 공격에 대해 미리 정의된 방어기작을 수행한다. 그림 5는 Application이 동작을 수행했을 때의 트래픽을 sFlow-RT를 이용하여 모니터링한 결과이다. SYN Flooding 공격 시 트래픽이 800K 이상 발생하였고, 13:40:51경에 Application을 동작 시키자 패킷의 흐름이 Drop됨에 따라 트래픽이 현저히 줄어드는 것을 확인 하였으며, Application 작동 중에 VM1의 일반 사용자 호스트와 서버 호스트간에는 원활한 통신이 가능함을 확인하였다. 13:41:33경 어플리케이션의 작동을 중지 시키자 다시 트래픽이 증가하는 것을 볼 수 있었고 13:42:22경 Application을 다시 동작 시켜 트래픽의 변화를 보았다. 이를 통해 Snort-IDS를 이용하여 공격을 탐지하고,

Controller를 이용하여 플로우 엔트리를 설정하여 탐지된 특정 호스트 만을 차단하여 방어 할 수 있음을 검증하였다.

5. 결론 및 향후 연구

본 논문에서는 sFlow의 샘플링 기술과 트래픽 모니터링 기능, Snort-IDS의 탐지 기능, OpenDayLight Controller의 플로우 제어 기능을 이용하여 DDoS Flooding 공격에 대한 신속한 방어가 가능한 시스템을 제안하고 프로토 타입을 제작해 보였다. 그리고 이 시스템을 이용해 가상의 네트워크에서 Flooding 공격을 수행했고 이를 탐지하고 플로우 엔트리를 설정하여 방어하고, 트래픽 모니터링 툴을 이용하여 검증함으로써 이 시스템의 가능성을 보였다.

향후 보다 복잡하고 다양한 네트워크 토폴로지를 구성하여 실험을 진행 할 것이며, 같은 Snort Rule에 대해서도 sFlow의 샘플링 주기와 비율에 따라 결과가 다르기 때문에 최적의 샘플링 주기와 비율을 찾아내는데 중점을 두고 연구를 진행 할 것이다.

Acknowledgements

본 연구는 한국정보화진흥원(NIA)의 미래네트워크연구시험망(KOREN) 사업 지원과제의 연구결과로 수행되었음 (15-951-00-001).

참 고 문 헌

[1] 신명기. “ICT 전문가 인터뷰”, 『한국정보통신기술협회 TTA저널』 제 151호, 2014. pp.14-15

[2] 하태진, 정치욱, Jargalsaikhan Narantuya, 안남원, 임혁, 김종원 “sFlow를 이용한 네트워크 공격 탐지 시스템”, 『2014년 한국통신학회 하계학술대회』 2014.

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

[4] Zhengyang Xiong. “An SDN-based IPS Development Framework in Cloud Networking Environment”, 『2014 mster thesis』. 2014

[5] Martin Andreoni Lopez, Otto Carlos M. B. Duarte. “Providing Elasticity to Intrusion Detection Systems in Virtualized Software Defined Networks.” 『2015 ICC』 . 2015

[6] Nhu-Ngoc Dao, Junho Park, Minho Park, Sungrae Cho. “A Feasible Method to combat against DDoS Attack in SDN Network.” 『2015 ICOIN』. 2015

[7]InMon. (http://www.inmon.com/).

[8]Snort. (http://www.snort.org/).