# SDN에서 Flow 기반 침입 탐지 시스템의 탐지 성능 개선 방법

이동수*, 김광조*

*카이스트 전산학과

# Improving Detection Capability of Flow-based IDS in SDN

Dongsoo Lee*, Kwangjo Kim*

*Department of Computer Science, KAIST.

**요 약**

일반적인 침입탐지 시스템은 네트워크 경계에 설치되어 외부망과 내부망 사이의 패킷 데이터를 모두 검사하는 방식을 사용하고 있으나, 내부망의 침입 탐지를 동일한 방법으로 하고자 할 경우 네트워크의 과도한 오버헤드를 유발하기 때문에 어려운 점이 있다. 이 경우 적은 데이터를 사용하는 Flow 기반 탐지 방법이 대안이 될 수 있으나 Flow 기반 탐지 방법은 공격의 상세 내용을 알기 어렵기 때문에 상세 분석 및 추가 대처에 한계가 있다. 본 논문에서는 내부망 침입 탐지 성능을 개선하기 위해 먼저 Flow 기반 침입 탐지를 수행한 뒤, 공격으로 탐지된 패킷에 대해 패킷 기반 검사를 추가로 수행하여 상세한 탐지 결과를 생성하는 방법을 제안한다. 해당 기법의 구현 방법으로써 SDN을 이용하며, Flow 기반 침입 탐지 후, 패킷 라우팅 경로를 조절하여 패킷 기반 탐지 시스템으로 공격 패킷을 전달하는 방식을 제안한다. 이를 통해 세부 검사 및 상세 결과의 보관이 가능하므로, 단기간의 침입 탐지뿐만이 아니라 이후의 보안 정책, 탐지 알고리즘을 보완하는 용도로 또한 활용 가능하다.

## I. 서론

침입 탐지 시스템(IDS, Intrusion Detection System)은 네트워크 내의 패킷이나 통신 정보를 기반으로 침입 여부를 판단하는 시스템으로, 방화벽의 비교적 단순한 필터링 방식에 비해 공격을 심도 있게 탐지 가능하므로, 많은 네트워크에서 방화벽과 함께 필수적인 보안 설비로 장비하고 있다. snort를 비롯한 일반적인 IDS는 패킷의 헤더, 패킷 내용을 기준으로 탐지하기 때문에 탐지하고자 하는 모든 네트워크 패킷에 접근할 수 있어야 한다. 따라서 일반적인 침입 탐지 시스템은 방화벽과 가까운 위치에 설치되어 침입 차단 시스템으로써 활용되거나, 네트워크 통신을 수집하는 여러 개의 IDS 센서가 있고 센서들의 전송 데이터를 받을 수 있는 위치에 설치된다[1]. 이러한 경우 네트워크의 외부망과 내부망 사이에 오가는 데이터는 효율적으로 검사할 수 있으나, 내부망 내에서 오가는 모든 통신을 검사하는 데에는, 데이터 흐름이 IDS를 사용하지 않을 때에 비해 경로가 복잡해지므로, 많은 오버헤드가 발생하게 된다. 따라서 내부망의 침입 탐지가 필요하면서, 내부망의 통신량이 많을 경우 기존의 탐지 방법인 패킷의 시그니쳐 기반 탐지 방식이 아니라 다른 방법을 사용해야 할 필요성이 있다.

* (letrhee, kkj)@kaist.ac.kr
* 본 연구는 미래부가 지원한 2014년 정보통신·방송(ICT) 연구개발사업의 연구결과로 수행되었음 [1391104001, 생체모방 알고리즘(Bio-inspired Algorithm)을 활용한 통신기술 연구]
* This research was supported by the KUSTAR-KAIST Institute, KAIST, Korea.

## II. 관련 연구

내부망에 대한 침입 탐지에 대해서 이미 다양한 관련 연구가 있으며, 본 논문과 관련한 연구들을 소개한다.

### 2.1 Flow 기반 IDS

네트워크의 패킷 전체를 검사하는 것이 아니라, 네트워크의 흐름을 보고 침입 탐지 여부를 판단하는 Flow 기반 IDS가 있다. sFlow[2]와 같은 툴에서는 네트워크 상태를 Flow의 형태로 제공하며, 구분키로써 출발지 IP, 도착지 IP, 출발지 Port, 도착지 Port, 프로토콜을 사용하며, Flow 정보로써, Flow당 패킷 수, Flow당 전송량, 연결 기간 등을 이용한다[3]. Flow 기반 IDS를 이용할 경우, Flow는 여러 패킷들의 합으로 나타내어 포트, 프로토콜 외의 상세 패킷 내용을 다루지 않으므로, IDS가 탐지하는데 필요한 데이터의 양은 대폭 감소하는 장점이 있으나, 공격 탐지를 대략적으로 할 수 있을 뿐 상세 공격에 대한 확인은 어려운 문제가 있다.

### 2.2 SDN

SDN(Software Defined Network)[4]는 이전부터 사용 중인 네트워크 구조에서 라우터 또는 스위치가 데이터 입, 출력 관리, 라우팅 테이블 관리 등을 각각 담당하여 중앙 처리가 어려운 점을 보완하기 위한 새로운 네트워크 방식이다. SDN에서는 스위치의 역할을 데이터 전달 디바이스로만 한정하고, 네트워크 제어를 위해 Flow를 기반으로 처리하는 SDN Controller를 따로 두어 관리한다. SDN Controller는 Switch에서 받은 Flow 정보를 이용해 네트워크 모니터, 로드 밸런서 등의 각종 네트워크 애플리케이션을 운영할 수 있다. Braga 등[5]은 SDN의 Flow 자료들을 활용하여 오버헤드를 최소화한 Flow 기반 IDS를 제안한 바 있다.

## III. 요구사항

Flow 기반 IDS는 비교적 적은 오버헤드로 침입 탐지를 수행 가능하지만 공격의 상세 정보는 알기 어려운 문제가 있으며, 패킷 기반 IDS는 탐지 방법에 따라 탐지한 공격에 대해 자세한 정보를 알 수 있으나, 탐지하기 위해 모든 패킷을 받아야 하는 문제가 있다.

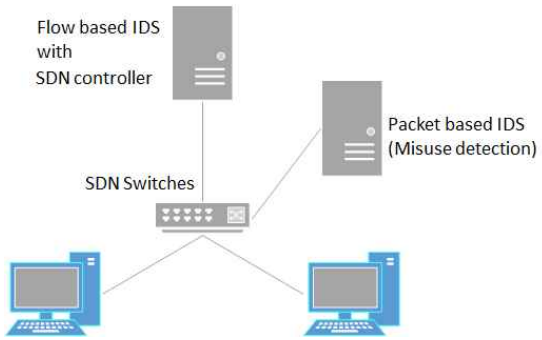
따라서 본 논문에서는 두가지 IDS의 결과물을 보완하여, Flow 기반 IDS와 비슷한 수준의 오버헤드로 침입 탐지가 가능하면서, 침입에 대해서 상세한 결과를 얻어, 이후에 새로운 방어 전략을 수립할 수 있도록 한다.

또한 가능하면 수동적인 대응만 할 수 있는 IDS 역할 뿐만 아니라 차단이 가능한 IPS로서도 운용 가능하도록 하는 것 또한 목표로 한다.

## IV. 제안방식

위의 요구사항에 따라 가능한 두 IDS의 장점을 가져올 수 있는 IDS 방식을 제안한다. SDN은 네트워크의 라우팅 테이블을 동적으로 관리할 수 있기 때문에, 패킷 기반 IDS로 보낼 데이터 또한 동적으로 관리할 수 있으며, 침입 탐지 후 차단 또한 가능한 이점이 있다. 이러한 구성을 위해 Flow 기반 IDS와 Packet 기반 IDS는 [그림 1] 과 같이 설치된다. Flow 기반 IDS는 Braga 등[5]의 구성과 같이 SDN Application으로서 동작하고, 패킷 기반 IDS는 구성에 따라 SDN Controller와 통합하거나, 분리될 수 있다. 각각의 경우 IDS 관리의 용이성, 자원의 분리 등의 이점이 있다.

이 중 실질적인 침입 탐지로는 Flow 기반 IDS를 사용하며, 패킷 기반 IDS는 Flow 기반 IDS의 탐지 결과를 보조하는데 사용한다.




[그림 1] 제안한 방식의 기본 구조

### 4.1 동작 방식

제안한 IDS의 상세 동작 방식은 [코드 1]과 같고, 각 단계에 따른 Packet의 구분 상태 변화는 [그림 2]와 같다.

#### 4.1.1 초기화 단계

먼저 SDN 스위치는 요청에 따라 네트워크 Flow를 주기적으로 보고한다. 이 때 악의적인 사용자가 네트워크의 다른 사용자에게 공격을 시도하면, 해당 행위 또한 Flow Statistics의 형태로 SDN Controller로 전달된다.

#### 4.1.2 Flow 기반 IDS 단계

SDN Controller는 Flow 통계를 받을 경우 SDN Application으로 동작중인 Flow 기반 IDS로 Flow 정보를 보낸다. Flow 기반 IDS는 Anomaly detection을 이용하여 공격 패킷인지, 일반 패킷인지 여부를 확인한다.

공격으로 확인된 경우, SDN Controller는 Flow의 IP, 포트 등을 확인하고, SDN Switch에게 해당하는 정보를 갖는 패킷을 Packet 기반 IDS로 보내도록 지시한다.

#### 4.1.3 Packet 기반 IDS 단계

Packet 기반 IDS에게 전달된 패킷은 Packet 기반 IDS를 이용하여 알려진 공격 여부, 사용된 공격 툴 등의 정보로 상세 분류한 뒤, 패킷 Payload와 두 IDS의 결과를 PCAP 포맷 등의 분석 가능한 형태로 저장한다. 이후 충분한 정보가 전달 받은 경우, 포워딩을 중지하고, 해당 패턴의 패킷을 일정기간 차단한다.

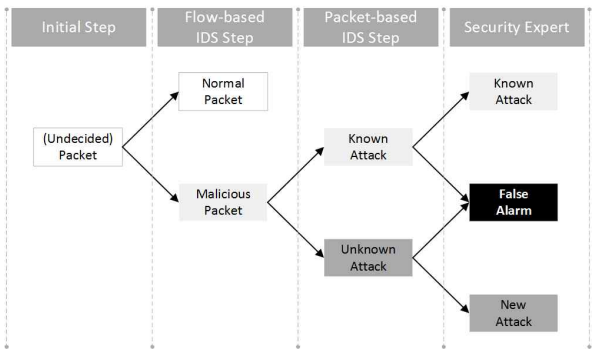
#### 4.1.4 정리 단계

이후 보관된 탐지 결과는 보안 전문가에 의해 상세 분석이 가능하며, 분석 결과의 오탐 여부 확인, 새로운 공격의 분석 등을 수행하고, 결과에 따라 해당 노드의 연구 차단, 분석 결과에 따른 IDS의 재학습 등의 작업이 가능하다.

위 과정을 통해, 제안한 IDS가 낮은 오버헤드로 Flow 기반 IDS의 결과물을 보완하고 탐지 결과를 보관하면서, IPS의 역할 또한 수행할 수 있다.

```
#Request flow statistics periodically
TimerSet(SendStatRequest, FlowStat, 5 sec, repeat)

#Add OpenFlow event handler (Asynchronous)
AddHandler("Flow_Stat", handlerFlowStat)
AddHandler("Flow_Removed", handlerFlowRemoved)
AddHandler("Packet_In", handlerPacketIn)

function handlerFlowStat(flows)
  foreach flow in flows
    checkFlow(flow)

function handlerFlowRemoved(flow)
  #for not grabbed by handlerFlowStat
  checkFlow(flow)

function checkFlow(flow)
  #Flow based IDS Step
  if flow.key not in monitoringList
    resultF = Detect_by_FlowIDS(flow)
    if resultF.malicious is True
      monitoringList.add(flow.key)
      StartForwardPacket(flow.key, ToPacketIDS)
      #Packet will come by PacketIn event

      TimerSet(StopDropOrForwardPacket, flow.key, 1 day, once)
    else
      #Normal Flow, do nothing
  else
    Detect_by_FlowIDS(flow) #Refresh result
    TimerRefresh(StopDropOrForwardPacket, flow.key, 1 day, once)

function handlerPacketIn(event)
  if event.key in monitoringList
    #Packet based IDS Step
    resultP = Detect_by_PacketIDS(event)
    resultF = Result_by_FlowIDS(event)
    SaveStorage(event,resultP,resultF)

    if resultP.knownAttack is True
      StartDropPacket(flow.key)
      StopForwardPacket(flow.key)

    TimerRefresh(StopDropOrForwardPacket, event.key, 1 day, once)
  else
    #Do SDN/OpenFlow routing stuff
```

[코드 1] 제안하는 아이디어의 의사 코드




[그림 2] 각 단계별 패킷 분류

4.2 구현

제안한 아이디어를 다음과 같이 구현하였다.

먼저 Mininet[6] 이용하여 가상 SDN 네트워크를 구성하고, POX[7]이나 다른 SDN Controller 툴을 이용하면 기본적인 테스트가 가능한 SDN 가상 네트워크를 구성하였다. 단 이번 구현에 사용한 POX는 Logical Port 기능이 없는 OpenFlow 1.0버전만 지원하므로[8], 실제 구현시에는 Flow 기반 IDS, SDN Controller, Packet 기반 IDS를 같은 위치에 설치하였다.

또한 IDS의 탐지 알고리즘은 Flow 기반 IDS와 Packet 기반 IDS에 각각 다른 알고리즘을 적용하였다. Flow 기반 IDS에서는 scikit-learn[10]에서 제공하는 기계 학습 알고리즘 중 생체 모방 알고리즘[8]인 PSO를 적용하였고, Packet 기반 IDS에서는 테스트에 사용할 Dataset을 수작업으로 signature를 입력하였다.

## V. 비교

각 IDS의 비교는 [표 1]과 같다. 제안한 IDS는 오버헤드는 Flow 기반 IDS에 바탕을 두므로 적은 오버헤드로 운용이 가능하며, Packet 기반 IDS를 추가 적용하여 탐지 성능을 높일 수 있다.

[표 1] 각 IDS간의 기능 비교

| | Packet 기반 IDS | Flow 기반 IDS | **제안한 IDS** |
|---|---|---|---|
| 오버헤드 | 매우 많음 | 적음 | **비교적 적음** |
| 탐지율 | 높음 | 높음 | **높음** |
| 결과 | 공격 툴 등 상세 분류 | 대략적인 공격 분류 | **공격 툴 등 상세 분류** |

## VI. 결론

본 논문에서는 Flow 기반 IDS의 결과물에 Packet 기반 IDS를 추가 적용하여 탐지 결과를 상세하게 제공해 이후의 보안 대책을 수립할 수 있도록 하였고, 해당 IDS의 오버헤드를 낮게 유지하기 위해 SDN 상에서 복합 동작하는 IDS를 제안하였다.

추후 연구로써 IDS의 네트워크 오버헤드, 탐지 결과물, 및 개선 가능성 등 상세 지표에 대한 분석을 통해 본 IDS의 효용성을 더욱 높일 수 있을 것이다.

## [참고문헌]


[1] Scarfone, Karen, and Peter Mell. "Guide to intrusion detection and prevention systems (idps)." NIST special publication 800.2007, 2007.

[2] sFlow, http://www.sflow.org/about/index.php .

[3] Anna Sperotto, et al. "An overview of IP flow-based intrusion detection." Communications Surveys & Tutorials, IEEE 12.3 (2010): 343-356.

[4] Open Networking Foundation, “Software-Defined Networking: The New Norm for Networks.” ONF White Paper, 2012.

[5] Rodrigo Braga, Edjard Mota, and Alexandre Passito. "Lightweight DDoS flooding attack detection using NOX/OpenFlow." Local Computer Networks (LCN), 2010 IEEE 35th Conference on. IEEE, 2010.

[6] Bob Lantz, Brandon Heller, and Nick McKeown. "A network in a laptop: rapid prototyping for software-defined networks." Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks. ACM, 2010.

[7] POX, http://www.noxrepo.org/pox/about-pox/ .

[8] Open Networking Foundation, “OpenFlow Switch Specification Version 1.0.0.” 2009/

[9] Dario Floreano, Claudio Mattiussi, and Bio-Inspired Artificial Intelligence. "Theories, Methods, and Technologies." 2008.

[10] scikit-learn, http://scikit-learn.org/