논문 22-47-08-10 The Journal of Korean Institute of Communications and Information Sciences '22-08 Vol.47 No.08
https://doi.org/10.7840/kics.2022.47.8.1114

엣지 컴퓨팅에서 딥러닝 기반의 침입 탐지 시스템 설계 및 구현

김 종 욱◆, 최 미 정°

Design and Implementation of a Deep Learning-Based Intrusion Detection System in Edge Computing

Jong-Wouk Kim◆, Mi-Jung Choi°

요 약

엣지 컴퓨팅은 클라우드 계층과 단말 계층 사이에 엣지 계층을 추가한 새로운 분산 컴퓨팅 기술이다. 엣지 컴퓨팅은 클라우드 컴퓨팅보다 상대적으로 더 많은 표적을 제공하기 때문에 공격자는 취약점 악용, DoS/DDoS, 중간자 공격, 그리고 인증 우회 등을 사용한다. 다양한 위협을 탐지하기 위해 침입 탐지 시스템, 방화벽, 안티 바이러스 소프트웨어와 같은 보안 시스템들이 사용되지만 낮은 정확도, 높은 오탐으로 인해 엣지 컴퓨팅에는 부적합하다. 또한, 침입 탐지를 위해 필요한 전문 인력과 대응할 수 있는 솔루션의 부족과 같은 한계점도 드러났다. 본 논문에서는 엣지 컴퓨팅에서 한계점을 극복하기 위해 딥러닝 기반의 침입 탐지 시스템을 제안한다. 제안하는 딥러닝 기반의 침입 탐지 시스템은 엣지 컴퓨팅 플랫폼인 KubeEdge를 사용하여 엣지 컴퓨팅을 구성하였다. 침입 탐지 모델을 생성하기 위해 희소성 제약을 사용하여 중요한 특징들을 추출 및 학습했다. 학습된 침입 탐지 모델을 엣지 컴퓨팅에 배포 및 운영하였을 때 평균 98.96%의 정확도, 99.41%의 F1-Score, 2.270%의 오탐률, 0.4990%의 미탐률을 달성했으며 사용자에게 침입이 발생했음을 보고하고 공격자 IP를 차단하는 적절한 대응을 수행했다.

키워드 : IDS, 엣지 컴퓨팅, 희소성 제약, KubeEdge, 딥러닝
Key Words : IDS, Edge Computing, Sparsity Constratins, KubeEdge Deep Learning

ABSTRACT

Edge computing is a new distributed computing technology that adds an edge layer between the cloud and device layer. Edge computing offers more targets than cloud computing, and attackers exploit vulnerabilities, DoS/DDoS, man-in-the-middle attacks, and authentication bypasses to threaten them. Security systems such as intrusion detection systems (IDS), firewalls, and anti-virus software are unsuitable for edge computing due to low accuracy and high false positives. It also revealed limitations such as a lack of security personnel and solutions to respond. This paper proposes a deep learning-based IDS to overcome the limitations of edge computing. We implemented a deep learning-based IDS in KubeEdge that a highly scalable edge computing platform and extracted important features using sparsity constraints to train an intrusion detection model. The

※ 본 연구는 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2020R1A2C1012117).
◆ First Author : Kangwon National University Department of Computer Science jw.kim@kangwon.ac.kr.korean.ac.kr, 정회원
° Corresponding Author : Kangwon National University Department of Computer Science and Kangwon National University IGP. in Medical Bigdata Convergence, mjchoi@kangwon.ac.kr, 종신회원
논문번호 : 202203-043-B-RN, Received March 28, 2022; Revised May 21, 2022; Accepted May 25, 2022

model deployed in edge computing achieved 98.96% accuracy, 99.41% F1-Score, 2.270% false-positive, and 0.4990% undetected rate. The system reported to the user that an intrusion had occurred and took appropriate actions to black the attackers' IP.

Ⅰ. 서 론

엣지 컴퓨팅(edge computing)은 통신, 자율 주행, 스마트 팩토리, 보건 의료 등의 Internet of Things (IoT) 분야에서 대량의 데이터와 서비스를 처리하는 중요한 기술로 사용되고 있다. 엣지 서버는 그림 1처럼 기존 클라우드 컴퓨팅에서 클라우드와 단말 장치(device) 사이에 엣지 서버를 추가하여 데이터나 서비스를 클라우드 서버 대신 처리한다. 엣지 컴퓨팅은 중앙 집중형에서 벗어난 분산형 컴퓨팅 방식으로 디바이스와 비교적 가까운 곳에 위치하여 처리하기 때문에 효율적으로 대역폭을 이용할 수 있고, 실시간으로 처리할 수 있으며, 초저지연, QoS(Quality of Service) 등을 보장한다[1-3]. 이러한 장점들로 인해 많은 수의 기업이 엣지 컴퓨팅을 채택하고 컴퓨팅 환경을 전환하고 있다. Statista 社의 통계에 따르면 전 세계의 엣지 컴퓨팅 시장 규모는 2019년 29억 4,000만 달러였으며 2030년까지 687억 1,000만 달러까지 성장할 것으로 예상했다[4]. 또한, IoT 디바이스의 수는 2020년 87억 4,000만개에서 2030년까지 254억개로 늘어날 것이라고 예상했다[5,6].

엣지 컴퓨팅 시장은 계속해서 성장하고, 기존의 컴퓨팅 기술에 비해 많은 장점이 있지만 이와 동시에 다양한 한계점도 존재한다. 엣지 컴퓨팅이 가지고 있는 대표적인 한계점은 클라우드 서버에 비해 낮은 연산 능력, 디바이스에서 사용자 인터페이스 부재로 인한 공격의 미인지, 운영체제와 프로토콜의 다양성, 엣지 컴퓨팅에 적합하지 않은 접속 및 인증 시스템 등이 있다. 이와 같은 한계점들로 인해 엣지 컴퓨팅은 그림 2와 같은 다양한 공격에 노출되어 있으며, 침입을 탐지 및 차단할 수 있는 솔루션/제품과 지속적인 감시와 시스템을 관리할 수 있는 전문 인력이 부족한 상황이다. 그 뿐만 아니라, 사이버 공격 및 침입의 형태가 점점 다양화되고 지능화되면서 이를 탐지할 수 있는 보안 기술의 진화가 필요한 시점이다.

최근 국내외에서 IoT 해킹, 랜섬웨어, 서비스 거부(denial of service, DoS)/분산 서비스 거부 공격(distributed denial of service, DDoS) 등과 같은 형태의 사이버 침해 사고들이 발생하고 보고되고 있다. 대표적인 사례는 Mirai 바이러스에 의한 DDoS 공격이다. Mirai 바이러스는 IoT 디바이스들의 인증 취약점을 이용하여 2016년 8월에 65,000여개의 디바이스들을 감염시켰다. 감염된 디바이스들은 봇넷(botnet)으로 전환되어 연결된 수많은 네트워크에 DDoS 공격을 수행했다. Mirai 바이러스의 공격을 받은 Dyn 社의 178,000여개 도메인을 분석한 결과, Mirai 바이러스가 유도한 DDoS 공격으로 인해 14,000개 이상의 도메인이 무력화됐다[7,8]. 이후 IoTReaper, Hajime 등과 같은 Mirai 바이러스의 변종들이 등장하였고 2017년

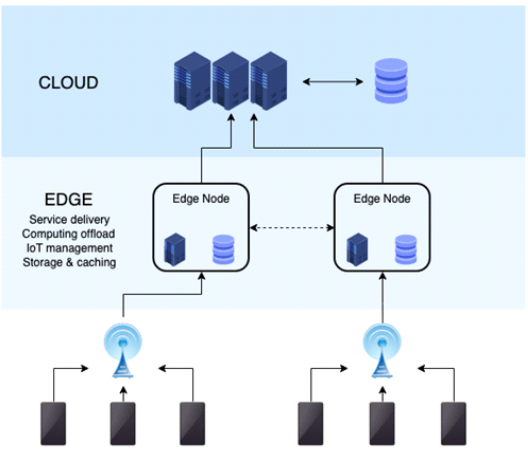


그림 1. 엣지 컴퓨팅의 구조
Fig. 1. Structure of edge computing

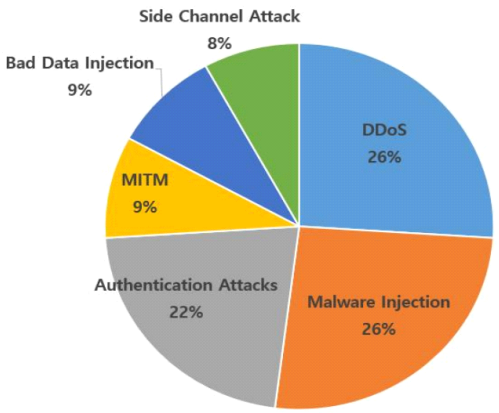


그림 2. 엣지 컴퓨팅 대상 공격 유형
Fig. 2. Types of attacks targeting edge computing

에는 3억 78,000만개 이상의 디바이스들을 감염시켰다. 이러한 사이버 공격들은 엣지 컴퓨팅을 포함한 네트워크에 막대한 피해를 주기 때문에 다양한 사이버 공격들과 침입을 효율적으로 탐지할 수 있는 보안 시스템의 개발 및 연구가 필요하다.

최근 연구자들은 엣지 컴퓨팅의 보안 문제점을 해결하기 위해 인공 지능을 활용한 침입 탐지 시스템(intrusion detection system, IDS)을 연구하고 있다. IDS는 보안을 위해 대표적으로 사용되는 시스템으로써 네트워크나 시스템을 감시하여 외부의 침입을 탐지하는 시스템을 의미한다. 인공 지능과 결합한 IDS는 대량의 데이터를 학습하여 데이터를 분류하거나 특정한 값을 예측할 수 있으며, 데이터의 양이 많을수록 높은 성능을 보여준다[9]. IT의 많은 분야에서 다양하게 활용되고 우수한 성능을 보여주는 딥러닝은 대량의 데이터를 처리해야 하는 엣지 컴퓨팅에서 활용할 수 있는 적합한 기술이다. 특히, 딥러닝은 기존의 공격 탐지 기술들과 비교할 때 데이터 셋에서 다양한 공격 패턴을 학습함으로써 정확한 탐지가 가능하다. 반면에, 엣지 컴퓨팅에서 디바이스들은 클라우드 노드와 엣지 노드에 비해 사용할 수 있는 자원이 적어 딥러닝을 포함한 보안 시스템의 적용이 어렵다. 따라서 상대적으로 자원이 한정된 디바이스에서도 보안 향상을 위한 시스템의 연구 및 개발이 필요하다.

본 논문에서는 딥러닝과 확장성이 높은 엣지 플랫폼을 사용하여 한정된 자원을 가진 이기종 장치들에서도 호환성이 높은 IDS 운영을 목표로 한다. 또한, 기존의 IDS보다 정확한 침입 탐지 및 대응을 자동화할 수 있는 딥러닝 기반의 IDS를 설계 및 구현한다. 실험을 통해 실제 엣지 컴퓨팅을 구성하고 희소성 제약을 사용하여 추출한 중요한 특징들만 학습한 침입 탐지 모델을 배포 및 운영하여 98.96%의 평균 정확도와 99.41%의 F1-Score, 2.270%의 오탐률과 0.4990%의 미탐률을 달성했다. 침입이 탐지됐을 경우 침입 탐지 시스템이 사용자에게 탐지 결과를 보고하고 공격자 IP를 차단하는 대응을 수행하는 것을 확인했다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로 딥러닝, 딥러닝 기반의 침입 탐지 시스템, 엣지 컴퓨팅의 보안을 위한 IDS에 관한 연구들을 소개한다. 3장에서는 본 연구에서 제안하는 딥러닝 기반의 IDS를 설계 및 구현한다. 4장에서는 설계한 침입 탐지 모델의 성능을 평가하고 IDS의 가동을 확인한다. 마지막으로 5장에서 논문의 결론을 맺는다.

Ⅱ. 관련 연구

2.1 딥러닝

딥러닝은 기계 학습의 한 분야로 여러 층의 신경망을 구성하여 특징을 추출하거나 고차원의 데이터를 학습하는 방법이다. 딥러닝의 종류에는 크게 지도 학습, 비지도 학습, 강화 학습으로 나눌 수 있다[10]. 지도 학습은 데이터 셋의 정답이 표시된 출력(output) 데이터를 이용하여 데이터를 분류하거나 특정 값을 분류한다. 비지도 학습은 지도 학습과 다르게 정답이 표시되어 있지 않은 데이터를 학습하여 새로운 데이터를 생성하거나 상관관계를 분석한다. 강화 학습은 인간의 판단 과정을 구현한 학습 방법으로 어떠한 행위를 이행하였을 때 최대한의 보상을 제공받는 방향으로 다음 판단을 결정하는 학습 방법이다. 딥러닝의 신경망은 하나 이상의 계층으로 구성되며, 각 계층은 다수의 노드를 포함한다. 각 계층의 노드들은 가중합(weighted sum)된 하위 계층 노드의 출력값을 전달받는다. 수식 (1)처럼 n번째 계층의 j번째 노드의 출력값(X^{n}_{j})은 이전 계층(n-1)의 출력값(X^{n-1}_{j}), 가중치(w^{n}_{ij}), 그리고 편향(bias, θ^{n}_{j})으로 계산할 수 있다.

$$X^{n}_{j} = f(\sum_{i} w^{n}_{ij} \times X^{n-1}_{j} + \theta^{n}_{j}) \quad (1)$$

딥러닝의 서로 연결된 계층적인 구조는 데이터를 재표현하는 가장 좋은 방법이다. 각 계층의 가중치(w^{n}_{ij})가 적절히 학습된 경우 각 계층은 입력 계층과 가까운 하위 계층이 출력한 낮은 수준(low-level)의 특징으로부터 좀 더 높은 수준(high-level)의 특징을 추출한다[11]. 최하위 계층에서는 가장 낮은 수준의 특징을 추출하며, 출력 계층과 가까운 상위 계층으로 갈수록 더 높은 수준의 특징으로 가중합되어 나타낼 수 있다. 그 결과, 최상단의 계층은 가장 높은 수준의 특징을 추출한다. 따라서, 딥러닝 모델은 높은 수준의 특징을 추출하여 복잡한 연산을 요구한다. 이를 통해 기존의 기계 학습 방법보다 많은 수의 데이터를 효과적으로 학습할 수 있다. 딥러닝은 이러한 특성으로 인해 많은 수의 가중치와 학습 파라미터(parameter)를 포함하고 있다. 학습 파라미터가 많아지면 과대적합(overfitting) 문제가 발생하여 새로운 데이터에 대한 분류, 인식, 그리고 생성 성능이 떨어진다. 반면에, 학습 파라미터가 부족한 경우 과소적합(underfitting) 문제가 발생하여 데이터를 분류, 인식, 및 생성할 수 있

는 특징을 학습하지 못한다. 따라서 모델을 학습할 때 성능 저하 문제가 발생하지 않도록 적절한 수의 계층, 하이퍼 파라미터 조절과 같은 방법을 고려해야 한다.

2.2 딥러닝 기반의 침입 탐지 시스템

IDS는 네트워크나 호스트 시스템을 감시함으로써 외부의 침입을 탐지하는 시스템을 의미한다. IDS는 크게 탐지 방법과 데이터의 출처에 따라 분류할 수 있다. 탐지 방법에 따라 이상 기반(Anomaly-based) IDS와 시그니처 기반(Signature-based) IDS로 분류한다.

이상 기반 IDS는 정상적인 행위와 비정상적인 행위를 구분하여 정상적인 행위에서 벗어난 정도에 따라 침입 행위라고 정의하는 방법이다. 이상 기반 IDS는 잘 알려지지 않은 공격들도 탐지할 수 있지만, 높은 오탐률을 가지고 있으며 공격 행위라고 판단한 이유를 증명하기 쉽지 않다. 시그니처 기반 IDS는 공격 행위들의 시그니처를 데이터베이스에 저장하고 탐지 프로세스가 실행되면 데이터베이스를 참조하여 시그니처가 일치하는 경우 침입으로 보고한다. 시그니처 기반 IDS는 잘 알려진 공격들은 매우 잘 탐지하고 오탐률이 낮다는 장점이 있지만, 시그니처가 존재하지 않는 새로운 공격들은 탐지하기 어렵다.

데이터의 출처에 따라서는 호스트 기반(Host-based) IDS와 네트워크 기반(Network-based) IDS가 있다. 호스트 기반 IDS는 운영체제나 애플리케이션 프로그램들의 로그를 추적하여 침입을 정확히 탐지할 수 있지만, 상대적으로 네트워크 공격에 대한 탐지는 어렵다. 네트워크 기반 IDS는 일반적으로 대표적인 호스트나 스위치에 적용하여 운영체제에 독립적이다. 그러므로 특정한 형태의 프로토콜이나 네트워크 공격을 탐지할 수 있지만, 특정 네트워크를 통과하는 트래픽만 감시할 수 있다[12].

Ahmim et al.은 최근 필수적인 정보와 서비스를 다루는 중요한 국가 기반 시설들이 사이버 공격에 목표가 되고 있다고 설명했다[13]. 이러한 시스템들을 사이버 공격들로부터 방어하기 위해 접근 제어 및 권한 인증과 같은 보안 메커니즘들과 IDS와 같은 보안 요소들을 적용해야 한다. 따라서 저자는 다양한 공격들을 정확하게 탐지하는 것이 가장 중요하다고 주장했으며, 이것을 위해 서로 다른 분류 모델들을 포함한 계층적인 IDS를 제안했다. 첫 번째 분류기는 'Attack'과 'Benign'으로 분류된 데이터 셋을 학습했고, 두 번째 분류기는 'Benign'과 각 공격으로 분류된 데이터 셋을 학습한 모델이다. 그리고 특징 벡터(feature vectors)들의 정규화를 진행했다. 세 번째 분류기는 첫 번째와 두 번째 분류기를 통해 계산된 특징 벡터들과 순수한 데이터 셋을 합쳐서 학습했다. 저자는 CICIDS2017 데이터 셋에서 2,830,743개의 데이터를 분리하였고 학습 데이터 셋과 실험 데이터 셋을 각각 40,000개와 20,000개를 추출하여 사용했다. 저자는 분류기로 각각 REP tree, JRip 알고리즘과 Forest PA를 사용했으며 실험을 통해 96.67%의 정확도를 보여주었으며 학습 소모된 학습 시간은 195.5초이고 탐지 속도는 2.27초가 걸렸다.

2.3 엣지 컴퓨팅 보안을 위한 침입탐지 시스템

Eskandari et al.은 개인의 데이터를 인터넷으로 전송할 수 있는 디바이스의 수가 기하급수적으로 증가함에 따라 사이버 위협 탐지가 오늘날의 기술 중 가장 중요한 연구 분야라고 설명했다[14]. IoT 및 엣지 컴퓨팅에서의 통신은 사용자, 애플리케이션, 장비들과 상호 작용하는 여러 IoT 기기들의 중요한 정보들을 포함하기 때문에 반드시 보안을 설정해야 한다. 하지만, 현대의 IoT 디바이스들은 소형화 및 저렴함을 이유로 제한된 보안 기능을 제공하기 때문에 공격에 더욱 취약하다. 따라서, 저자는 제한된 IoT 환경에서도 효율적으로 침입을 탐지할 수 있는 IDS를 제안했다. 저자는 시그니처 기반의 IDS는 세 가지 제약 사항이 있다고 설명하고 있다. 첫 번째는 잘 알려진 공격들만 탐지할 수 있으며 제로데이(Zero-day) 공격이나 미확인된 공격은 탐지하지 못하는 것이다. 두 번째는 식별할 수 있는 공격의 수가 증가할수록 시그니처의 수도 증가하기 때문에 IDS의 응답 시간이 지연된다는 것이다. 이것은 실시간성이 중요한 IDS에 있어서 치명적인 문제이다. 마지막은 시그니처를 확보하려면 전문가가 연구하고 분석해야만 한다. 이러한 문제점들을 해결하기 위해 시그니처 기반의 IDS보다 이상 기반의 IDS가 더욱 적절하다고 설명했다.

이를 극복하기 위해 저자는 일반적인 IoT 환경에 적합한 Passban IDS를 제안했다. 저자가 제안한 Passban IDS는 경량화된 탐지 모델이자 one-class 분류 모델로 디바이스에 필요한 하드웨어 요구 사항과 비용을 감소시켰다. Passban IDS는 네트워크 패킷을 캡처하고 특징을 추출함으로써 네트워크 트래픽을 감시한다. 캡처한 패킷으로부터 학습에 사용할 특징 집합을 만들고 정상인 패킷만 학습한다. 학습한 모델을 게이트웨이에 배포하여 위협을 탐지한다. 저자는 이를 증명하기 위해 Raspberry Pi 3을 무선 네트워크를 사용하여 연결하고 실험 환경을 구축했다. 저자는 IoT 디바이스들의 데이터를 보호하기 위해 디바이스들과

가까운 곳에 IDS를 적용함으로써 전체 네트워크의 부하를 줄이고 디바이스의 보안을 보장할 수 있다고 설명했다.

저자는 새로운 위협을 발견했을 때 디바이스를 유지함과 동시에 관리가 용이하도록 학습 모델만 업데이트하는 방식을 채용했다. 학습 모델의 업데이트는 웹 기반의 관리 인터페이스를 이용하여 손쉽게 모델의 업데이트가 가능하다. 예측 단계에서 위험하다고 판단될 경우 목표 디바이스의 트래픽 출입을 막는다. 저자는 네트워크 공격들이 일반적으로 '정상'이라고 알려진 경우와 큰 차이를 보이므로 one-class 분류 방법이 효율적이라고 설명했다. 저자가 제안한 머신러닝 모델은 정상으로 분류된 경우만 학습했으며, 다른 행위들을 비정상 또는 침입이라고 분류한다. 저자는 12시간 동안 일반적인 상황에서 1,730만개의 패킷들을 수집했으며, 포트 스캐닝, HTTP Brute Force, SSH Brute Force, SYN Flood의 4가지 공격을 직접 수행한 상황의 네트워크 패킷을 수집했다. 수집한 패킷들에서 저자가 제안한 Passban IDS의 성능 측정을 위해 정밀도, 재현율, F1-Score를 측정했고, Passban IDS가 실행 중일 때 네트워크의 대역폭 사용률, CPU 사용률, 메모리 사용률을 측정했다. 저자들은 실험을 통해 최고 99.0% 및 최저 79.0%의 F1-Score, 최고 98.0% 및 최저 92.0%의 정밀도, 그리고 최고 100.0% 및 최저 69.0%의 재현율을 달성했다. 또한, Passban IDS를 사용했을 경우 CPU를 28.75% 더 사용했으며 5.27%의 메모리와 17.2 Mbit/s의 네트워크 대역폭을 절감했다.

Ⅲ. 딥러닝 기반의 침입 탐지 시스템

보안의 중요한 고려 사항은 네트워크 아키텍처의 다양함으로 인한 통합된 침입 탐지의 어려움이다[15]. 기존의 IDS들은 클라우드 계층과 단말 계층에 적용되어 운영되었다. 하지만, 엣지 컴퓨팅의 등장으로 보안 메커니즘이 관점이 달라졌다. 많은 수의 디바이스 연결로 복잡해진 네트워크는 사이버 공격에 노출되어 있다. 또한, 엣지 노드는 다양한 위치에 분산되어 있기 때문에 통합하여 관리하기 어렵다[16]. 따라서 분산된 엣지 노드들과 서로 다른 운영체제를 가진 이기종 장치들에서도 침입 탐지가 가능하고 효율적으로 관리할 수 있는 호환성이 높은 IDS의 설계 방법이 필요하다.

3.1 침입 탐지 시스템 설계

3.1.1 KubeEdge

본 논문에서 제안하는 IDS 설계를 위해 오픈소스(open-source) 기반의 엣지 컴퓨팅 플랫폼을 조사하고 본 연구에서 가장 적합한 플랫폼을 활용하고자 한다. 오픈소스 기반의 엣지 컴퓨팅 플랫폼의 종류는 StarlingX 2.0, AKRAINO, EdgeX Foundary, Home Edge, KubeEdge 등이 있다. 본 연구에서 사용하는 엣지 컴퓨팅 플랫폼은 CNCF(Cloud Native Computing Foundation)의 대표적인 프로젝트인 Kube-Edge이다. KubeEdge는 쿠버네티스(Kubernetes) 기반의 경량화된 엣지 컴퓨팅 플랫폼으로 컨테이너화된 애플리케이션을 오케스트레이션(orchestration)하고, 엣지 계층의 호스트까지 관리할 수 있도록 확장하는 오픈소스 플랫폼이다[17]. 클라우드에 비종속적이기 때문에 퍼블릭, 프라이빗, 하이브리드 등의 다양한 환경에서 동작이 가능하고, 쿠버네티스 기반으로 설계되었기 때문에 업그레이드, 롤백, 모니터링가 가능하며 유지 및 보수가 쉽다는 장점이 있다.

3.1.2 IDS 설계

본 논문에서 제안하는 딥러닝 기반의 IDS는 그림 3처럼 클라우드, 엣지, 그리고 디바이스 노드로 이루어져 있다. 클라우드 노드는 데이터 셋을 사용하여 학습이 완료된 침입 탐지 모델을 엣지 노드로 배포한다. 엣지 노드는 학습이 완료된 침입 탐지 모델을 실행, 운영하여 디바이스 노드 및 엣지 노드의 침입을 실제 탐지하는 역할을 수행하고 이를 각 디바이스에 전달한다. 디바이스 노드 및 엣지 노드의 대응 모듈은 전달받은 탐지 결과를 사용자에게 보고하고 운영체제에 따라 적절한 대응을 수행한다. 해당 시스템은 많은 시스템 자원을 요구하는 딥러닝 모델을 엣지 노드에서 운영함으로써 한정된 자원을 가진 단말 디바이스는 간접적으로 딥러닝 모델을 사용할 수 있다. 또한, 확장성이 높은 엣지 컴퓨팅 플랫폼을 사용하여 침입 탐지 모델의 배포 및 관리가 용이하다는 장점이 있다. 따라서 딥러닝과 엣지 컴퓨팅 플랫폼을 활용한 IDS 설계로 한정된 자원을 가진 이기종 디바이스에서도 보다 정확한 IDS 운영 및 적용이 가능하다.

그림 4는 클라우드 노드와 엣지 노드의 구조도를 나타낸 것이다. 그림 4에서 클라우드 노드의 모델 생성 모듈은 데이터 셋을 사용하여 학습이 완료된 침입 탐지 모델을 생성하고, 모델 갱신 모듈이 학습이 완료된 침입 탐지 모델을 도커(docker) 이미지로 만들어

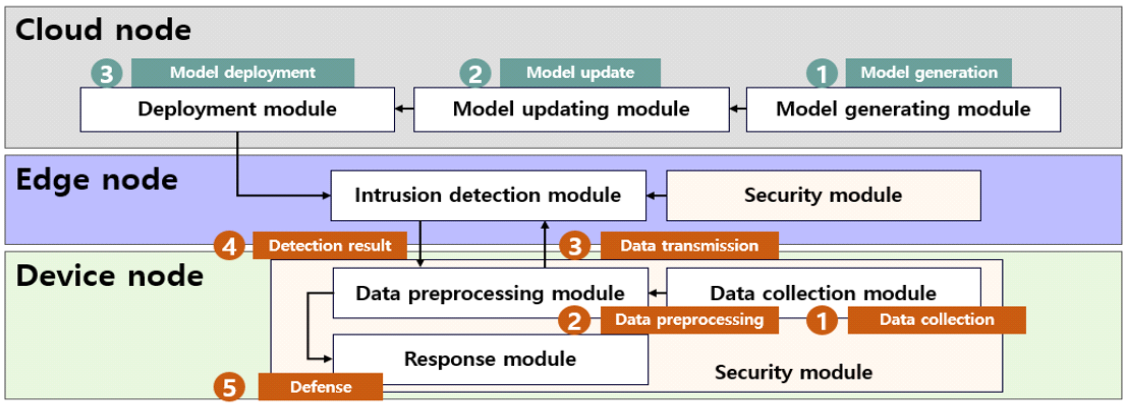


그림 3. 침입 탐지 시스템의 전체 구조도
Fig. 3. Overall structure of IDS

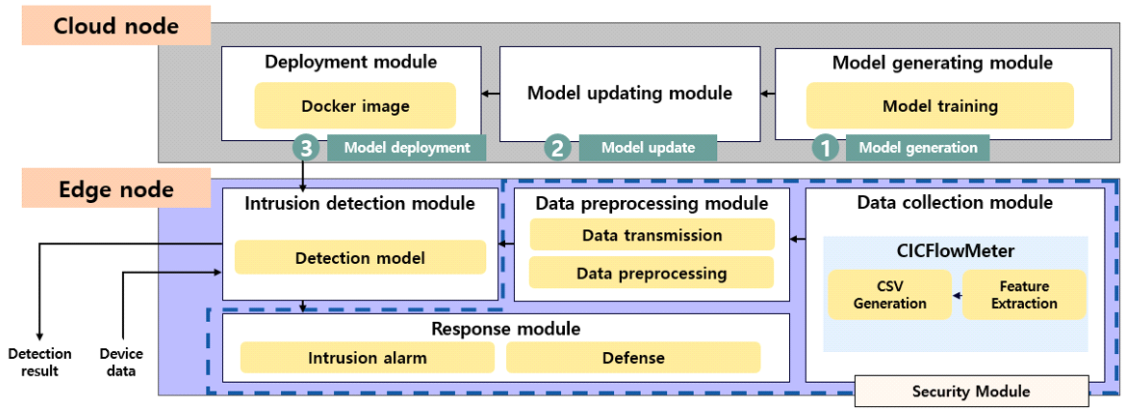


그림 4. 클라우드 노드와 엣지 노드의 구조도
Fig. 4. Structure of cloud and edge node

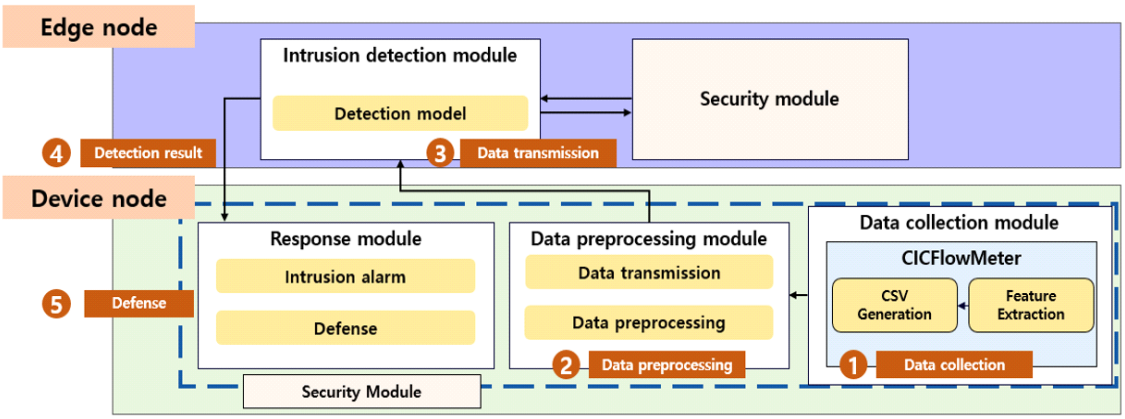


그림 5. 엣지 노드와 디바이스 노드의 구조도
Fig. 5. Structure of edge and device node

도커 내부 저장소에 저장한다. 배포 모듈은 쿠버네티스를 통해 침입 탐지 모델이 포함된 도커 이미지를 엣지 노드로 배포한다. 엣지 노드의 침입 탐지 모듈은 배포 받은 도커 이미지를 실행하여 침입 탐지 모듈을 운영하고 엣지 노드와 디바이스 노드의 보안 모듈로부터 전달받은 데이터를 통해 침입 여부를 판단하고 결과를 각 디바이스로 전송한다.

그림 5는 엣지 노드와 디바이스 노드의 구조도를 나타낸 것이다. 그림 5에서 데이터 수집 모듈은 CICFlowMeter 소프트웨어를 사용하여 패킷을 캡쳐 및 특징을 추출하여 CSV 파일로 저장한다. 데이터 전처리 모듈은 CSV 파일의 데이터를 탐지 모델이 사용할 수 있도록 적절한 전처리를 수행하여 엣지 노드의 침입 탐지 모듈로 전송한다. 엣지 노드의 침입 탐지 모듈은 전달받은 데이터가 위협인지 아닌지 판단하여 그 결과를 각 디바이스로 전송한다. 이때, 엣지 노드의 보안 모듈은 엣지 노드의 보안을 위해 디바이스 노드의 보안 모듈과 동일하게 데이터 수집 모듈, 데이터 전처리 모듈, 대응 모듈을 통해 노드 내부의 침입 탐지를 수행한다. 엣지 노드와 디바이스 노드의 대응 모듈은 엣지 노드의 침입 탐지 모듈이 전송한 탐지 결과를 사용자에게 보고하고 침입이 탐지되었을 경우 공격자의 IP를 차단하고 운영체제가 윈도우즈(Windows) 계열일 경우 윈도우즈에서 제공하는 방화벽 소프트웨어인 윈도우즈 디펜더(Windows Defender)로 규칙을 생성하여 IP를 차단한다. 운영체제가 리눅스(Linux) 계열일 경우 패킷 필터링 도구인 iptables를 사용하여 규칙을 추가하여 공격자의 IP를 차단한다.

3.2 침입 탐지 모델

엣지 컴퓨팅에서 침입 탐지를 위해 사용한 딥러닝 모델은 SSAE-DeepCNN으로 그림 6과 같다. 본 모델은 stacked sparse autoencoder (SSAE)와 DeepCNN을 결합한 하이브리드 모델이다. SSAE는 오토인코더의 한 종류로 수많은 뉴런 중 일부가 비활성화하는 것을 구현한 것으로 희소성 제약(sparsity constraints)을 사용하여 중요한 특징 벡터를 추출한다. SSAE는 은닉층(hidden layer)의 손실 함수(loss function)에 희소성 제약을 추가하여 중요도가 낮은 특징 벡터들을 학습하는 유닛을 비활성화한다. SSAE를 통해 높은 중요도의 특징 벡터를 추출하고 학습하면 학습 모델의 편향과 분산을 줄일 수 있다. 입력 데이터가 재구성되었으므로 학습 모델의 분류 성능을 향상시킬 수 있다. 또한 감소된 수의 특징 벡터를 학습하기 때문에 학습

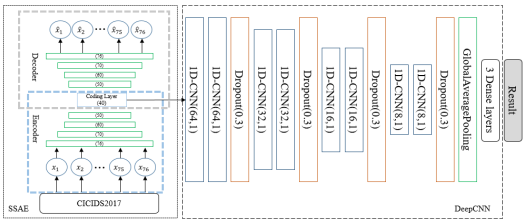


그림 6. SSAE-DeepCNN 모델 구성도.
Fig. 6. Structure of SSAE-DeepCNN model

모델의 파라미터도 줄어든다는 장점이 있다.

SSAE의 가장 가운데에 존재하는 은닉층을 'coding layer' 또는 'bottleneck'이라고 하며 이 은닉층의 손실 함수에 희소성 제약을 추가한다[18]. Ng는 SSAE에 희소성 제약을 추가하기 위해 Kullback-Leibler 함수를 이용한다. 본 연구에서는 SSAE의 희소성 제약을 추가하기 위해 L1 정규화 방법을 사용한다. L1 정규화 방법은 일반적으로 과대적합 문제를 방지하기 위해 사용되지만, 유닛의 손실 함수에 정규화 계수가 포함된 특정 값을 빼줌으로써 유닛의 가중치를 0으로 만들어 유닛을 비활성화한다. 이러한 과정을 통해 상대적으로 중요도가 낮은 특징 벡터들을 제거하고 중요한 특징 벡터들을 추출한다. 따라서 본 논문에서는 SSAE의 은닉층에 L1 정규화 방법으로 희소성 제약을 추가하여 중요도가 높은 특징 벡터를 추출한다.

DeepCNN은 SSAE가 추출한 높은 중요도의 특징 벡터들을 학습한다. DeepCNN은 두 개의 1D-CNN과 Dropout 계층이 한 개의 블록으로 총 네 개의 블록이 연결되고 한 개의 GlobalAverage- Pooling(GAP) 계층과 세 개의 심층 계층(dense layer)이 연결된 구조이다. 1D-CNN 계층을 통해 입력 데이터를 합성곱 연산을 수행 후 특징 맵을 생성한다. 드롭아웃 계층(dropout layer)에서 임의 유닛의 가중치를 0으로 만들어 과대적합 문제를 방지한다. GAP 계층에서 모든 특징 맵의 가중치의 평균을 계산하고 네 개의 블록을 통해 계산된 데이터의 형태를 1차원의 형태로 변환한다. 세 개의 심층 계층이 가중치를 학습하고 데이터가 침입인지 아닌지 이진 분류한다. 즉, SSAE를 통해 추출한 높은 중요도의 특징 벡터들만 학습하여 학습 시간을 줄이고 정확도가 높은 침입 탐지 모델을 생성한다. 그러므로 SSAE-DeepCNN 모델은 중요한 특징들만 학습하므로 빠르게 침입 모델을 생성할 수 있으며 판단 오류를 줄여 정확도가 높기 때문에 속도와 정확도가 중요한 IDS에 적합한 모델이라고 할 수 있다.

3.3 데이터 수집 및 모델 업데이트

IDS의 가동 확인 실험에서 사용할 데이터 수집을 위해 그림 7과 같이 한 개의 공격자 노드와 세 개의 희생자 노드를 구성했다. 공격자 노드를 제외한 희생자 노드에 CICFlowMeter 소프트웨어를 설치했으며 해당 노드의 모든 플로우 및 패킷 데이터를 수집하고 학습에 사용할 데이터를 추출하여 CSV 파일로 저장한다.

CICIDS2017 데이터 셋을 학습한 탐지 모델의 가중치 업데이트를 위해 실제 공격을 통해 수집하고 전처리한 추가 데이터 셋을 사용한다. 이때, 탐지 모델에서 특징을 추출하는 SSAE의 학습 가능(trainable) 변수를 False로 설정하여 가중치를 고정했고, 데이터를 분류하는 DeepCNN의 학습 가능 변수만 True로 설정하여 가중치를 갱신한다. 가중치를 업데이트하는 이유는 CICIDS2017 데이터 셋의 수집 환경과 실험을 진행하는 환경에 차이가 있어 모델의 오탐과 미탐이 발생하는데 이를 방지하기 위해 실제 환경에서 수집한 데이터 셋으로 갱신한다.

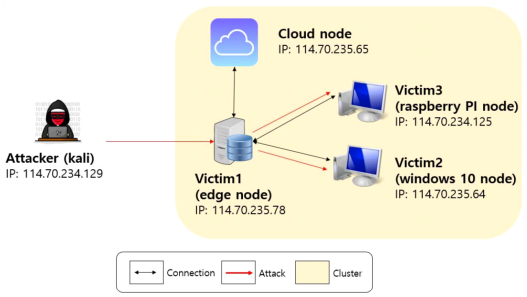


그림 7. 데이터 수집을 위한 공격 시나리오
Fig. 7. Attack scenario for data collection

Ⅳ. 실 험

실험을 위해 클라우드 노드와 엣지 노드의 하드웨어 및 소프트웨어 정보는 표 1과 같다. 클라우드 노드에서 모델을 생성 및 엣지 노드를 관리하며, 생성한 모델을 엣지 노드로 전달한다. 엣지 노드는 다양한 디바이스의 관리할 수 있으며, 클라우드 노드로부터 전달받은 모델을 사용하여 위협 여부를 판단할 수 있다. 다음 절에서는 본 실험을 위한 설계와 실험 결과에 관하여 설명한다.

4.1 실험 설계

본 절에서는 최적의 침입 탐지 모델 생성 실험과

표 1. 실험 환경
Table 1. Environment of experiments

Node	Category		Specification
Cloud	H/W	CPU	Intel® CoreTM i7-8700 CPU 3.20GHz
		RAM	64GB
		GPU	GeForce GTX 2080 Ti
	S/W	OS	Ubuntu 18.04.2 LTS
		Python	Python 3.6.9
		Keras	Keras 2.3.1
Edge	H/W	CPU	Intel® CoreTM i7-8700 CPU 3.20GHz
		RAM	32GB
		GPU	GeForce GTX 1080 Ti
	S/W	OS	Ubuntu 18.04.2 LTS
		Python	Python 3.6.9
		Keras	Keras 2.3.1

침입 탐지 시스템 가동 확인 실험을 설명한다. 최적의 침입 탐지 모델 생성 실험은 네트워크 침입 탐지를 위한 최적의 모델을 찾는 것이며, 시스템 가동 확인 실험은 최적의 모델을 엣지 컴퓨팅에 배포하고 IDS를 운영하여 실제로 침입을 정상적으로 탐지하는지 확인한다. 표 2는 최적의 탐지 모델 생성 실험에서 각각의 모델에 설정한 희소성 계수를 정리한 것이다. 희소성 제약을 추가하는 방법으로 L1 정규화 방법과 Kullback-Leibler 함수를 사용한 경우를 비교한다. L1 정규화 방법의 희소성 계수는 λ(lambda)이고, Kullback-Leibler 함수의 희소성 계수는 ρ(rho)로 표기한다. 또한, SSAE가 추출한 특징 벡터를 학습할 경우 최적의 성능을 달성하는지 확인하기 위해 모든 특징 벡터를 학습한 DeepCNN과도 성능을 비교한다.

침입 탐지 모델의 학습을 위해 CICIDS2017 데이터 셋의 2,827,876개의 데이터를 사용하여 학습과 실

표 2. 최적의 침입 탐지 모델 생성을 위한 희소성 계수
Table 2. Sparsity coefficients for generating optimal IDS

Model	Coefficient
DeepCNN	N/A
SSAE-DeepCNN	$\lambda = 1e-3$
	$\lambda = 1e-3$
	$\lambda = 1e-3$
	$\rho = 5e-2$
	$\rho = 5e-3$

험을 위해 데이터를 7 대 3으로 나눈다. 데이터 셋은 정답을 포함한 총 79개의 특징을 가지고 있으며 학습에 불필요한 'Destination Port'와 중복된 특징인 'Fwd Header Length'를 제거하고 정답이 표시된 'Label'을 분리한다.

따라서, 학습에 사용한 총 특징의 수는 76개이며 SSAE는 40개의 특징을 추출한다. 딥러닝 모델의 학습을 위해 전처리한 데이터 셋의 구성은 표 3과 같다. CICIDS2017 학습 데이터 셋의 '정상(benign)'으로 라벨링된 데이터는 1,589,924개로 전체 데이터 셋에 56%를 차지하고, '비정상(malicious)'이라고 지정된 데이터는 389,589개로 전체 데이터 셋에 14%를 차지한다. 실험 데이터 셋에서 정상으로 지정된 데이터는 681,396개로 24%를 차지하며, 비정상으로 라벨링된 데이터는 166,967개로 6%를 차지한다. 따라서 각 그룹의 데이터 셋을 7 대 3로 나누어 학습 및 실험 데이터 셋을 구성했으며, 학습 데이터의 수는 1,979,513개, 실험 데이터는 848,363개를 사용하여 실험을 진행한다.

탐지 시스템 가동 실험을 위해 수집한 데이터 셋은 표 4와 같다. 침입 탐지 모델의 업데이트를 위해 수집한 학습 데이터 셋은 총 60,854개이며 정상으로 지정된 데이터는 42,602개로 70%를 차지하고, 비정상으로 지정된 데이터는 18,252개로 30%를 차지한다. 노드별 공격 탐지를 위한 실험 데이터 셋은 총 3개이며 각각 26,083개로 정상으로 지정된 데이터는 18,258개로 70%를 차지하며 비정상으로 지정된 데이터는 7,825개로 30%를 차지한다. 마찬가지로 학습 데이터 셋과 실험 데이터 셋의 비율은 7 대 3으로 분리한다.

표 3. 학습을 위해 전처리된 데이터 셋의 구성
Table 3. Preprocessed dataset for training

Data	Benign	Malicious	Total
Train Set	1,589,924 (56%)	389,589 (14%)	1,979,513 (70%)
Test Set	681,396 (24%)	166,967 (6%)	848,363 (30%)

표 4. 업데이트 및 공격 탐지를 위해 수집한 데이터 셋
Table 4. Dataset for updating and attack detection

Data	Benign	Malicious	Total
Updating Data	42,602 (70%)	18,252 (30%)	60,854 (100%)
Testing data for Windows	18,258 (70%)	7,825 (30%)	26,083 (100%)
Testing data for Raspberry PI	18,258 (70%)	7,825 (30%)	26,083 (100%)
Testing data for Ubuntu	18,258 (70%)	7,825 (30%)	26,083 (100%)

본 연구에서 침입 탐지 모델의 성능 평가를 위해 정확도와 F1-Score를 측정한다. F1-Score는 정밀도와 재현율의 조화 평균이며 데이터의 불균형이 있을 경우에도 모델의 정확도를 측정할 수 있는 평가 지표이다. 또한, 정상 및 공격 데이터 셋의 불균형을 이루므로 F1-Score의 micro 및 macro를 같이 측정한다. F1-Score(macro)는 클래스 별 데이터 수에 상관없이 모든 클래스를 같은 비중으로 다루며, 클래스 별 가중치를 주지 않는다. F1-Score(micro)는 F1-Score(macro)와는 다르게 각 클래스 별 가중치 주어 F1-Score를 측정한다. 침입 탐지 시스템 가동 실험을 위해 정확도, F1-Score, 오탐률(false positive rate, FPR) 그리고 미탐률(false negative rate, FNR)을 측정했다. 위의 성능 지표를 계산하기 위해 수식 (2) - (13)를 사용한다. True positive(TP)는 실제 양성인 것을 양성이라고 바르게 예측한 것이고, false positive(FP)는 실제 음성인 정답을 틀리게 예측한 것이다. True negative(TN)은 실제 음성인 것을 음성이라고 바르게 예측한 것이고, false-negative(FN)는 실제 양성인 것을 음성이라고 틀리게 예측한 것이다.

$$`Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \tag{2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$Recall_{macro} = \frac{Recall_{benign} + Recall_{malicious}}{2} \tag{4}$$

$$Recall_{micro} = \frac{TP_{benign} + TP_{malicious}}{TP_{benign} + FN_{benign} + TP_{malicious} + FN_{malicious}} \tag{5}$$

$$Precision = \frac{TP}{TP+FP} \tag{6}$$

$$Precision_{macro} = \frac{\mathrm{Precison}_{benign} + \mathrm{Precison}_{malicious}}{2} \tag{7}$$

$$Precision_{micro} = \frac{TP_{benign} + TP_{malicious}}{TP_{benign} + FP_{benign} + TP_{malicious} + FP_{malicious}} \tag{8}$$

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

$$F1\text{-}Score_{macro} = 2 \times \frac{Precision_{macro} \times Recall_{macro}}{\mathrm{Precision}_{macro} + Recall_{macro}} \tag{10}$$

$$F1\text{-}Score_{micro} = 2 \times \frac{Precision_{micro} \times Recall_{micro}}{Precision_{micro} + Recall_{micro}} \quad (11)$$

$$`FPR = \frac{FP}{FP + TN} \quad (12)$$

$$FNR = \frac{FN}{TP + FN} \quad (13)$$

4.2 실험

본 논문에서 침입 탐지 모델 생성 실험의 경우 CICIDS2017 데이터 셋을 사용하여 모델의 성능을 측정했으며, IDS 가동 실험에서는 직접 수집한 데이터 셋의 학습 데이터로 탐지 모델을 업데이트하고 실험 데이터로 IDS의 성능을 측정했다.

4.2.1 침입 탐지 모델 생성

표 5는 이전 연구와 본 연구의 성능을 비교한 표이며, 그림 (8) - (9)는 침입 탐지 모델의 희소성 계수를 조절하여 정확도, F1-Score, 학습 및 예측 소요 시간을 측정한 결과이다. 그림 8에서 정확도는 L1 정규화 방법의 희소성 계수(λ)가 $1e-3$일 경우 96.9%로 가장 높았으며 SSAE를 사용하지 않고 모든 특징을 학습한 DeepCNN이 93.2%로 가장 낮았다. F1-Score는 L1 정규화 방법의 희소성 계수(λ)가 $1e-3$일 경우 96.9%로 가장 높았으며 SSAE를 사용하지 않은 DeepCNN이 92.7%로 가장 낮았다. F1-Score(macro) 및 F1-Score(micro) 또한, L1 정규화 방법의 희소성 계수(λ)가 $1e-3$일 때 가장 높았다. 따라서 SSAE를 통해 추출한 중요 특징들을 학습한 경우가 3.7%만큼 더 정확도가 높았다. 왜냐하면 중요하지 않은 특징 벡터까지 학습할 경우 그 특징 벡터들에 대한 가중치가 학습 모델이 예측할 때 판단의 오류를 일으켜 학습을 부적절하게 유도하기 때문이다. 이러한 학습 오류는 학습이 반복될 때마다 모델의 계산 복잡도를 증가시키고 잘못된 값을 예측하게 만든다. 따라서 모든 특징을 학습하는 것보다 올바른 답을 유도할 수 있는 중요한 특징들을 학습하는 것이 중요하다는 것을 알 수 있다.

그림 9에서 학습 시간은 Kullback-Leibler 함수의 희소성 계수(ρ)가 $5e-3$일 경우 11301.6초로 가장 오래 걸렸으며 Kullback-Leibler 함수의 희소성 계수(ρ)가 $5e-2$일 경우 8310.4초로 가장 빨랐다. 상대적으로 희소성 계수가 큰 경우가 희소성 계수가 작은

표 5. 기존 연구와 성능 비교
Table 5. Comparison of performance with other studies

Research	Dataset	Approach	Accuracy (%)	F1-Score (%)
[13]	CICIDS2017	REP Tree, JRip, Forest PA	96.6	N/A
[14]	Real data	Isolation Forest, local outlier forest	N/A	94.0
[19]	DDoS	Gaussian mixture models	99.0	N/A
[20]	CICIDS 2017	Negative selection algorithm and classifiers	98.0	97.0
Ours (λ=$1e-3$)	CICIDS 2017	Sparse stacked autoencoder with Depp CNN	96.9	96.9

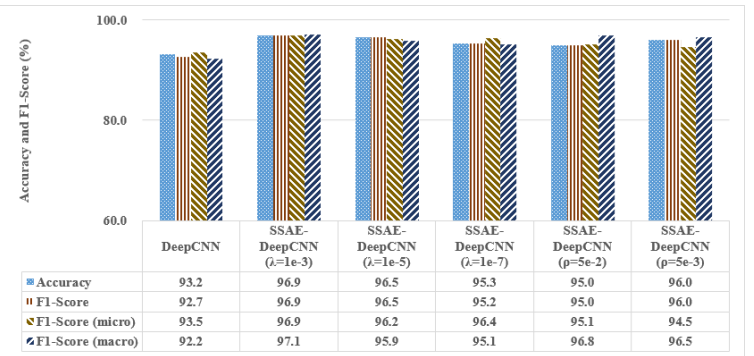


그림 8. 희소성 계수에 따른 성능 결과
Fig. 8. Performance results according to the sparsity coefficients

경우보다 학습 시간이 더 빨랐으며 SSAE를 사용하지 않은 DeepCNN 모델보다 더 빠르게 학습했다. 그 이유는 희소성 계수가 클 경우 학습에 사용되는 특징 벡터가 줄어들어 학습이 빠르게 이루어졌기 때문이다. 그에 반해 희소성 계수가 너무 작을 경우 특징 벡터가 제거되지 않아서 SSAE를 사용하지 않고 모든 특징을 학습한 DeepCNN과 비슷하게 경과했다. 예측 시간의 경우 모든 모델이 비슷하게 경과한 것으로 보아 희소성 계수와 예측 시간의 관련성은 찾기 어렵다는 것을 알 수 있다.

따라서 모델 생성 실험의 결과들을 통해 L1 정규화 방법을 사용하여 L1 정규화 방법의 희소성 계수(λ)가 $1e-3$일 경우 정확도와 F1-Score가 가장 높았다. 또한 정확도 및 F1-Score 대비 학습 시간이 다른 경우들보다 짧았다. 따라서 정확도가 높고 학습 시간이 짧기 때문에 침입 탐지 모델로 가장 적합하다.

그림 12는 IDS가 윈도우즈 10에서 침입을 탐지하고 사용자에게 보고한 것이다. 보고 내용에는 트래픽 발생 시간, 공격자의 IP, 희생자의 IP, 공격에 사용한 프로토콜, 탐지 결과를 포함한다. 그림 12의 (a)에서 공격자의 IP(114.70.234.129) 주소를 확인할 수 있으며 그림 12의 (b)와 (c)는 윈도우즈 계열의 방화벽 소

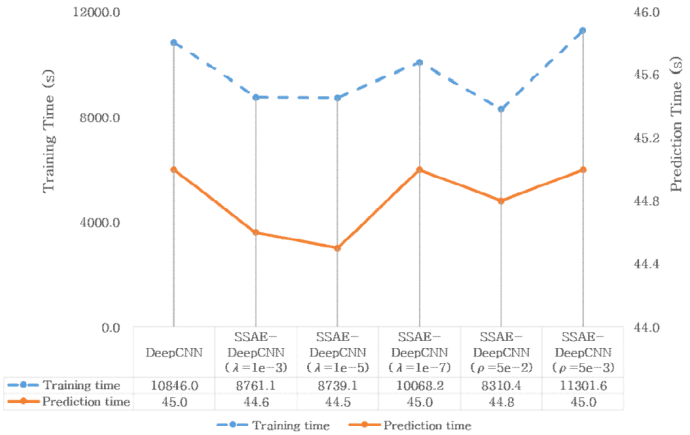
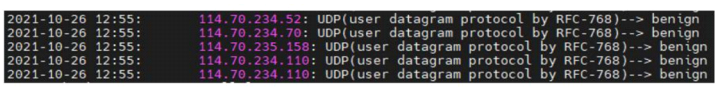
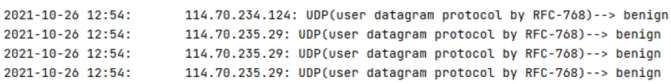


그림 9. 희소성 계수에 따른 학습 시간과 예측 시간
Fig. 9. Training and prediction time as a sparsity coefficients

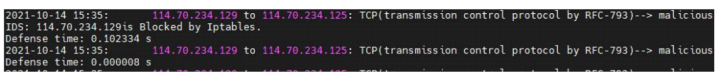


(a) Raspberry PI 노드

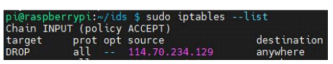


(b) Windows 10 노드

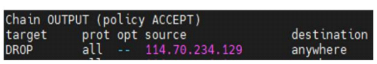
그림 10. IDS가 정상이라고 보고한 경우
Fig. 10. Report cases of 'benign'



(a) 침입 탐지 결과



(b) 공격 IP 차단(chain INPUT)



(c) 공격 IP 차단(chain OUTPUT)

그림 11. 라즈베리 파이에서 침입이 탐지된 경우
Fig. 11. Report cases of 'malicious' in Raspberry Pi

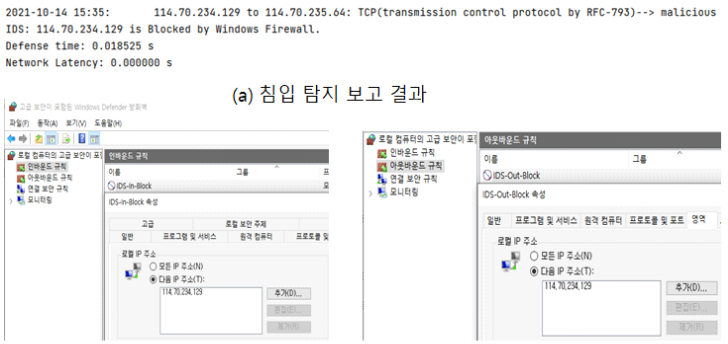


그림 12. 윈도우즈 10에서 침입이 탐지된 경우
Fig. 12. Report case of 'malicious' in Windows 10

표 6. IDS의 성능 측정 결과
Table 6. Results of IDS performance

Node	Accuracy (%)	F1-Score (%)	FPR (%)	FNR (%)
Windows 10	98.94	99.73	2.347	0.496
Raspberry Pi	99.03	99.27%	2.136	0.462
Edge	98.91	99.22	2.326	0.539

프트웨어인 윈도우즈 디펜더에 인바운드 규칙과 아웃바운드 규칙을 추가하여 공격자의 IP를 차단한 것을 나타낸 것이다.

표 6은 공격 시나리오를 통해 수집한 데이터 셋의 실험 데이터로 IDS의 성능을 측정한 결과이다. 실험 결과로 98.96%의 정확도, 99.41%의 F1-Score, 2.270%의 오탐률과 0.4990%의 미탐률을 달성했다. 따라서, 본 연구에서 설계한 딥러닝 기반의 침입 탐지 시스템은 높은 정확도와 F1-Score, 낮은 오탐률과 미탐률을 달성했다. 또한, 침입이 탐지되었을 경우 정확하게 침입을 탐지 및 보고하고 공격자의 IP를 차단하는 적절한 대응을 함으로써 시스템이 정상적으로 작동하는 것을 검증했다.

V. 결 론

클라우드 컴퓨팅의 제한된 제어 및 확장성, 보안 및 개인정보보호 등의 다양한 한계점을 극복하고자 개발된 엣지 컴팅은 공격자에게 더 많은 표적을 제공한다는 단점이 존재한다. 공격자들은 이러한 취약점을 이용하여 다양한 사이버 공격을 수행하고 있지만, 이전 세대의 침입 탐지 시스템은 이러한 환경에 적용하였을 때 낮은 정확도와 높은 오탐률 및 미탐률로 인해 부적절했다. 이에 많은 연구자들은 AI를 접목하여 침입 탐지의 정확도를 높이고 탐지 및 대응 프로세스를 자동화하는 방안을 연구하고 있다.

본 논문에서는 엣지 컴퓨팅에서 딥러닝 기반의 침입 탐지 시스템을 설계하고 구현함으로써 한정된 자원을 가진 IoT 디바이스들을 포함하는 엣지 컴퓨팅에서 정확한 침입 탐지, 침입 보고 및 대응을 자동화하는 것을 목표로 한다. 딥러닝 모델의 탐지 정확도를 향상시키기 위해 학습 데이터에서 중요도가 높은 특징들을 추출했으며, 추출한 특징들을 학습할 경우 96.9%로 가장 높은 탐지 정확도를 달성했다. 또한, 감소한 수의 특징들을 학습하기 때문에 학습에 소모되는 시간을 절약했다. 중요도가 높은 특징을 추출하기 위해 은닉 층에 희소성 제약을 추가하는 SSAE를 사용했으며, 실험을 통해 L1 정규화 방법으로 희소성 제약을 추가할 경우 가장 효과적인 특징을 추출하는 것을 확인했다.

엣지 컴퓨팅을 구성하고 침입 탐지 모델을 배포 및 운영하면서 실제 공격을 수행하였을 때 IDS는 평균 98.96%의 정확도, 99.41%의 F1-Score, 2.270%의 오탐률, 0.4990%의 미탐률을 달성했으며 사용자에게 침입이 발생했음을 보고하고 공격자 IP를 차단하는 적절한 대응을 수행했다. 따라서 본 논문에서 설계 및 구현한 딥러닝 기반의 IDS는 침입을 탐지하고 적절히 조치하는 것을 확인했다.

향후 연구로 CICFlowMeter 소프트웨어의 알고리즘을 분석하고 데이터 수집 모듈 스크립트에 적용하여 실시간으로 패킷 데이터를 수집 및 추출할 계획이다. 사용자 모니터링 인터페이스를 추가하여 탐지 결과를 도식화하여 제공할 예정이다. 또한, 침입이 탐지될 경우 침입의 여부뿐 만이 아니라 다중 분류를 통해 다

양한 공격의 식별이 가능하도록 IDS를 개선할 것이다.

References

[1] H. Park and T. Hwang, "Changes and trends in edge computing technology," *J. KICS*, vol, 36, no. 2, pp. 41-47, Jan. 2019.

[2] S. Shin, et al., "Edge computing market trends and application scenarios," *J. Electron. and Telecommun. Trends*, vol. 34, no. 2, pp. 51-59, 2019. (https://doi.org/10.22648/ETRI.2019.J.340206)

[3] B. H. Husain and S. Askar, "Survey on edge computing security," *J. Int. J. Sci. and Busin.*, vol. 5, no. 2, pp. 52-60, Feb. 2021. (https://doi.org/10.5281/zenodo.4496939)

[4] Y. Xiao, et al., "Edge computing security: State of the art and challenges," in *Proc. IEEE*, vol. 107, no. 8, pp. 1608-1631, Aug. 2019. (https://doi.org/10.1109/JPROC.2019.2918437)

[5] Statista, *Size of edge computing market worldwide in 2019 and 2030*, Retrieved Aug. 29, 2021, from https://www.statista.com/statistics/1256351/worldwide-edge-computing-market-revenues/

[6] Statista, *Number of internet of things (IoT) connected devices worldwide from 2019 to 2030*, Retrieved Aug. 29, 2021, from https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

[7] M. Antonakais, et al., "Understanding the mirai botnet," in *Proc. 26th USENIX Secur. Symp.,* pp. 1093-1110, Vancouver, Canada, Aug. 2017.

[8] Corero, *Financial impact of mirai ddos attack on dyn revealed in new data*, Retrieved Sep. 5, 2021, from https://www.corero.com/blog/financial-impact-of-mirai-ddos-attack-on-dyn-revealed-in-new-data/

[9] Z. Xu, et al., "Artificial intelligence for securing IoT services in edge computing: A survey," *J. Secur. and Commun. Netw.*, vol. 2020, pp. 1-13, Sep. 2020. (https://doi.org/10.1155/2020/8872586)

[10] J. Van Engelen and H. Hoos, "A survey on semi-supervised learning," *J. Mach. Learn.,* vol. 109, no. 2, pp. 373-440, Nov. 2019. (https://doi.org/10.1007/s10994-019-05855-6)

[11] I.-J. Kim, "Deep learning: A new trend in machine learning," *J. KICS*, vol. 31, no. 11, pp. 52-57, Oct. 2014.

[12] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," *J. Appl. Sci.*, vol. 9, no. 20, pp. 4396-4424, Oct. 2019. (https://doi.org/10.3390/app9204396)

[13] A. Ahmim, et al., "A novel hierarchical intrusion detection system based on decision tree and rules-based models," in *Proc. 15th Int. Conf. Distrib. Comput. in Sensor Syst.*, pp. 228-233, Santorini, Greece, Aug. 2019. (https://doi.org/10.1109/DCOSS.2019.00059)

[14] M. Eskandari, et al., "Passban IDS: An intelligent anomaly-based intrusion detection system for IoT edge devices," *IEEE Internet of Things J.*, vol. 7, no. 8, pp. 6882-6897, Aug. 2020. (https://doi.org/10.1109/JIOT.2020.2970501)

[15] P. Spadaccino and F. Cuomo, "Intrusion detection systems for IoT: Opportunities and challenges offered by edge computing," *arXiv preprint arXiv:2021.01174*, Dec. 2020. (https://doi.org/10.48550/arXiv.2012.01174)

[16] Z. Xu, et al., "Artificial intelligence for securing IoT services in edge computing: A survey," *J. Secur. and Commun. Netw.*, vol. 2020, pp. 1-13, Sep. 2020. (https://doi.org/10.1155/2020/8872586)

[17] H. Yang, J. Oh, and Y. Kim, "Openstack and open edge computing platform of CNCF," *J. KICS*, vol. 36, no. 9, pp. 55-62, Aug. 2019.

[18] A. Ng, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, pp. 1-19, 2011.

[19] Ö. Cepheli, S. Büyükçorak, and G. K. Kurt, "Hybrid intrusion detection system for DDoS attacks," *J. Electr. and Comput. Eng.*, vol. 2016, Article ID 1075648, pp. 1-8, Apr. 2016. (https://doi.org/10.1155/2016/1075648)

[20] S. Hosseini and H. Seilani, "Anomaly process

detection using negative selection algorithm and classification techniques," *Evolving Syst.,* vol. 12, pp. 769-778, Dec. 2019. (https://doi.org/10.1007/s12530-019-09317-1)

김 종 욱 (Jong-Wouk Kim)

2019년 2월 : 강원대학교 컴퓨터과학 졸업
2021년 2월 : 강원대학교 컴퓨터과학 석사
2021년 3월~현재 : 강원대학교 컴퓨터과학 박사과정
<관심분야> 네트워크 보안, 악성코드 분석, 딥러닝
[ORCID:0000-0002-8180-2376]

최 미 정 (Mi-Jung Choi)

1998년 2월 : 이화여자대학교 공학사 졸업
2000년 2월 : 포항공과대학교 공학석사
2004년 2월 : 포항공과대학교 공학박사
2004년~2005년 : 프랑스 INRIA 연구소 박사 후 연구원
2005년~2006년 : 캐나다 워터루대학 박사 후 연구원
2008년~현재 : 강원대학교 컴퓨터학부 컴퓨터공학 전공 교수
<관심분야> 네트워크 관리, 정보보안, 네트워크 보안
[ORCID:0000-0002-9062-4604]