논문 2025-2-9 http://dx.doi.org/10.29056/jsav.2025.06.09

소프트웨어 정의 데이터 센터의 제로 트러스트 네트워크 접근 설계 및 성능 평가

민석홍*†

Zero Trust Network Access Design for Software-defined Data Centers and its Performance Evaluation

Seokhong Min*†

요 약

최근 유연한 시스템 접근 방식의 출현과 함께 ZTA 개념이 네트워크 위협 환경에 대한 대응방안으로 다시 주목받고 있다. 전통적인 네트워크 보안 방법이 외부 및 내부 경계 지점에서 접근 제어에 중점을 두는 반면, ZTA 개념은 항상 검증을 요구하며 어떤 사용자나 장치도 신뢰하지 않는다. 본 논문에서는 SDDC에 유연하게 적용할 수 있는 SDN 기반 ZTNA 방안을 제안하고, 엔터프라이즈급 오픈 소스 SDDC 솔루션을 활용한 테스트베드 구현을 통해 ZTNA-enabled SDDC의 구현 전략을 제안한다. 또한, 구축 테스트베드에서 트래픽 플로우 기반의 접근 제어 성능 평가를 수행하여, SDDC의 외부 및 내부 네트워크 경계뿐만 아니라 내부 네트워크에서도 허용되지 않은 접근을 차단함으로써 SDDC의 보안성을 유연하게 향상시킬 수 있음을 보인다.

Abstract

Recently, with the emergence of flexible system access approaches, the concept of ZTA has regained attention as a response to network threat environments. While traditional network security methods focus on access control at external and internal boundary, the ZTA concept requires verification at all times and does not trust any user or device. This paper proposes an SDN-based ZTNA solution that can be flexibly applied to SDDC and presents an implementation strategy for ZTNA-enabled SDDC through the implementation of a testbed using an enterprise-grade open-source SDDC solution. Additionally, by performing performance evaluations of access control based on traffic flows within the implementation testbed, we show that it is possible to flexibly enhance the security of SDDC by blocking unauthorized access not only at the boundaries of external and internal networks but also within the internal network.

한글키워드 : SDx, 제로 트러스트 구조, 소프트웨어 정의 데이터 센터, 네트워크 트래픽 엔지니어링, 클라우드

keywords : SDx, zero-trust architecture, software-defined data center, network traffic engineering, cloud

1. 서 론

코로나19 팬데믹(COVID-19 pandemic)으로 인하여 원격 근무 및 미래 지향적인 업무환경 제

* 배재대학교 소프트웨어학과
† 교신저자: 민석홍(email: shmin@pcu.ac.kr)
접수일자: 2025.05.31. 심사완료: 2025.06.13.
게재확정: 2025.06.20.

공 필요성이 갑자기 증가함에 따라 기업들은 원격 근무 지원을 위해 클라우드 기반 솔루션으로 신속히 전환하였으며, 전통적인 네트워크 경계가 모호해지고 있다[1][2]. 이에 따라 보안 측면에서 기존의 전통적인 네트워크 보안 방식인 경계 기반 보안 모델을 벗어나 클라우드 시스템 내부 네트워크 또한 위협 요소로 간주해야 할 필요성이 증가하고 있다[3].

기존의 경계 기반 보안 모델은 최근 급증하고 있는 SDDC(Software-Defined Data Center) 기반 클라우드 시스템에서, 내부 가상 머신(Virtual Machine) 간의 접근 제어를 효과적으로 수행하기 어렵다. 이는 해당 모델이 외부와 내부 간 경계 지점에서의 접근 제어에 중점을 두고, 내부 네트워크로의 접속이 허용되지 않은 플로우(Flow)만을 차단하는 방식이기 때문이다. 따라서 일단 내부 네트워크로의 접근이 허용된 플로우는 신뢰 대상으로 간주되며, 내부 시스템 간의 세부적인 접근 제어는 수행되지 않는다. 그러나 원격 근무의 확산과 클라우드 기반 업무 환경의 보편화로 인해 보안 위협이 점차 복잡하고 정교해지고 있어, 이를 효과적으로 대응하기 위한 보다 발전된 네트워크 보안 모델의 도입이 요구되고 있다.

이러한 배경 속에서, 시스템 접근 요청에 대해 신뢰를 전제로 하지 않고 지속적인 검증을 요구하는 ZTA(Zero Trust Architecture) 개념이 다시 주목받고 있다. ZTA는 모든 사용자와 장치에 대해 끊임없는 인증 및 검증을 수행함으로써, 내부와 외부에서 발생할 수 있는 다양한 보안 위협을 효과적으로 차단할 수 있다. 또한 사용자는 최소 권한만을 부여받기 때문에 불필요한 데이터 접근이 줄어들며, 그에 따라 데이터 유출 위험도 감소한다. 더불어, 다양한 장치와 위치에서의 안전한 접근을 보장하여, 원격 근무 및 클라우드 환경에서도 높은 유연성과 보안성을 동시에 제공한다.

본 논문에서는 SDDC 환경에서의 유연한 네트워크 보안 패러다임 전환을 위해, SDN(Software-Defined Networking) 기반의 플로우 단위 접근 제어, 특히 SDDC 내부 가상 머신의 상호 연결에 대한 제어를 중심으로 한 기초 설계 방안을 제안한다. 제안 설계는 클라우드 내부와 외부 간의 상호 연결을 제어하기 위해 오픈 소스 기반의 SDDC 및 SDN 솔루션을 통합하고, 플로우 단위 접근 제어 정책이 적용된 ZTNA(Zero Trust Network Access) 에이전트(Agent)를 추가하여 테스트베드(Testbed)를 구축한다. 이후, 테스트베드에서 다양한 플로우에 상이한 접근 권한을 부여하고, 접근 제어 실험을 수행함으로써, SDDC 환경 내에서 발생할 수 있는 다양한 보안 위협에 대해 유연하고 정밀한 대응이 가능함을 보여 제안 설계가 SDDC의 보안성을 효과적으로 향상시킬 수 있음을 확인한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서 ZTA 및 ZTNA에 대한 선행 연구들을 살펴보고, Ⅲ장에서는 제안 설계를 상세히 설명한다. Ⅳ장에서는 제안 설계 기반의 구축 테스트베드에서 실험을 통해 SDDC 내부에서 시스템 접근에 대한 세분화된 제어 가능성을 분석한다. 마지막으로 Ⅴ장에서는 ZTNA와 SDDC의 통합 설계에 대한 결론 및 향후 연구 계획에 대하여 기술한다.

2. 관련 연구

2.1 제로 트러스트 구조(ZTA)

최근 진화하는 악의적인 시스템 공격에 대응하기 위해, 접속 제어 모델과 인증 프로토콜의 장단점을 비교 분석하여 신뢰성 있는 시스템 구축 방안을 모색하는 연구[4] 등이 활발히 진행되고 있다. 미국 NIST(National Institute of Stand-

ards and Technology)에서는 현대 사이버 보안 접근 방식으로서 “신뢰하지 말고 항상 검증하라(Zero Trust)”는 원칙을 제시하며, 내부와 외부를 불문하고 모든 네트워크 트래픽을 기본적으로 신뢰하지 않고 모든 요청에 대해 철저한 검증과 인증을 수행하는 접근 방식을 제로 트러스트로 정의하고 있다[5].

또한, [6]에서는 사용자가 데이터에 접근하는 경로 대신 데이터 객체 자체에 제로 트러스트 개념을 적용하고 있다. ZTA에서는 사용자에게 업무 수행에 필요한 최소한의 권한만을 부여하며, 네트워크를 세분화하여 공격자가 일부 영역에 침투하더라도 전체 시스템에 접근하지 못하도록 한다. 아울러, 사용자와 장치의 신뢰성을 정기적으로 평가하는 것을 원칙으로 한다. 이러한 ZTA는 다양한 기술과 솔루션을 통합하여 구축되며, 클라우드 환경뿐만 아니라 모든 온프레미스(On-premise) 환경에도 적용 가능하다. 이를 통해 데이터 유출 방지, 공격 표면 축소, 신속한 위협 대응 등 다양한 보안 이점을 제공하며, 그 구조는 그림 1과 같다.

그림 1. 제로 트러스트 접속
Fig. 1. zero-trust access

ZTA는 보안성과 관련하여 인상적인 비즈니스 결과를 제공하여, 고객 데이터와 비즈니스를 보호하는 더 큰 엔터프라이즈 가시성과 같은 상당한 비즈니스 가치를 제공한다[7].

2.2 제로 트러스트 네트워크 접근(ZTNA)

ZTA와 ZTNA는 모두 제로 트러스트 보안 모델의 원칙을 기반으로 하지만, 적용 범위와 초점에서 차이를 보인다. ZTA는 사용자와 기기의 접근을 전반적으로 관리하는 포괄적인 보안 프레임워크로, 모든 사용자와 기기를 신뢰하지 않는 것을 전제로 하며, 모든 접근 요청에 대해 철저한 검증과 인증을 요구한다. 이 모델은 클라우드 서비스, 애플리케이션, 데이터 등 다양한 자원에 대한 접근 제어에 중점을 둔다.

반면, ZTNA는 네트워크 접근 제어에 특화된 기술로, 주로 네트워크 트래픽 보호에 초점을 맞춘다. 사용자가 네트워크에 접근할 때, ZTNA는 사용자의 신원과 장치를 먼저 검증한 후, 최소 권한 원칙에 따라 필요한 수준의 접근 권한만을 부여한다. ZTNA는 기존의 가상 사설망(Virtual Private Network)과 같은 전통적인 네트워크 접근 방식의 대안으로, 보다 강화된 보안성과 세분화된 애플리케이션 접근 제어를 제공한다.

2.3 제로 트러스트 네트워크 접근 연구 동향

기존 연구들은 주로 ZTNA 구조를 단일 사용자 인증 중심으로 구현하거나, 전통적 네트워크 환경에서의 외부 접근 제어 강화에 집중해 왔다. 예컨대, 대부분은 가상 사설망 및 다단계 인증(Multi-Factor Authentication) 중심의 인증 체계를 통해 외부 경계에서 요청을 검증하며, 내부 네트워크의 세부 흐름 제어나 가상화 환경 내에서의 접근 통제에 대해서는 탐구나 실험이 제한적이었다[8].

또한, SDN 기반 접근 제어에 대한 선행 연구들도 존재하지만, 이는 주로 정책 수립 수준에 머무르고 있으며, 실제 SDDC 클라우드 환경에서 플로우 단위의 접근 제어, 특히 가상 머신 간 상호 연결에 대한 접근 제어 연구는 시도되지 않았다[9][10].

최근에는 엔터프라이즈 네트워크에 대하여 저비용으로 접속에 대한 제어, 성능, 확장성 및 보안을 달성하는 구조에 대한 방안[11]들도 등장하고 있다.

3. ZTNA-enabled SDDC 구현

제안 ZTNA-enabled SDDC 설계는 SDDC 환경에서 ZTNA의 유연한 제어를 구현하기 위해, SDDC 클라우드 인프라(Cloud Infrastructure)에 오픈플로우 프로토콜(OpenFlow Protocol) 기반 SDN(Software-Defined Networking) 기술을 적용한다. 클라우드 인프라는 범용 x86 서버에 Proxmox[13]를 설치하여 구축하며, ZTNA 제어 기능은 오픈플로우 스위치(OpenFlow Switch)와 이를 제어하는 ZTNA 에이전트로 구성하여 그림 2와 같은 ZTNA-enabled SDDC 환경을 구현한다. 또한, 그림 3은 오픈플로우 프로토콜을 활용한 트래픽 플로우 제어 구조[14]를 나타낸다.

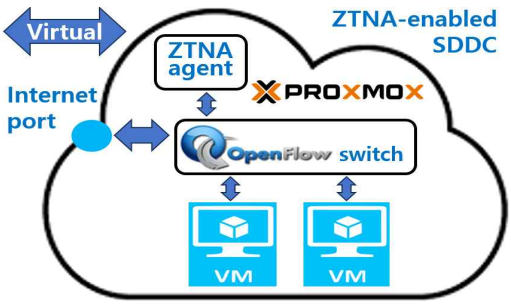

그림 2. 제안 ZTNA-enabled SDDC 설계
Fig. 2. Design for Proposed ZTNA-enabled SDDC

제안 설계에서의 ZTNA 제어는 오픈플로우 스위치의 플로우 테이블(Flow Table)에 트래픽 플로우(Traffic Flow)의 다양한 속성(Attribute)을 조합하여 플로우 엔트리(Flow Entry)를 설정함으로써 원하는 접근 제어를 수행한다. 플로우 엔트리는 자유롭게 설정할 수 있으므로, Proxmox 기반 클라우드 인프라와 오픈플로우 스위치의 결합을 통해 ZTNA 기능을 효과적으로 구현할 수 있다. 플로우 테이블에 대한 제어 명령은 오픈플로우 스위치의 명령어 셋(Command Set)을 기반으로 ZTNA 에이전트에서 수행된다.

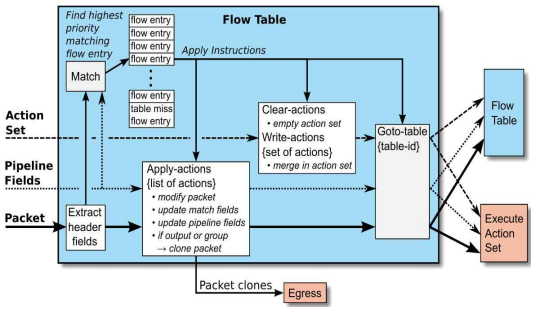

그림 3. 오픈플로우의 트래픽 플로우 제어 구조[11]
Fig. 3. Traffic Flow Control Architecture of OpenFlow

3.1 오픈플로우 프로토콜

오픈플로우(OpenFlow) 프로토콜은 오픈플로우 컨트롤러(OpenFlow Controller)와 오픈플로우 스위치 간의 통신 및 네트워크 트래픽 제어를 위한 표준화된 프로토콜로, 프로그래머블(Programmable) 특성을 갖는다. 이 프로토콜은 플로우 기반의 포워딩 테이블(Forwarding Table)을 생성하며, L2(2계층)부터 L4(4계층)까지 다양한 프로토콜의 속성들을 조합하여 플로우 단위의 세밀한 제어를 가능하게 한다. 이러한 플로우 기반 테이블 구성과 표준화된 프로토콜의 활용을 통해, 시스템 요구에 맞는 NFV(Network Function Virtualization) 프로그램 작성이 가능하며[15], 네트워크 트래픽 엔지니어링 기술의 적용 및 확장이 용이하다. 이에 따라 추가적인 기능 확장이 유연해지며, 소프트웨어 정의 시스템의 통합 구현에 있어 폭넓은 활용이 가능하다.

3.2 프록스목스

프록스목스는 엔터프라이즈급 오픈 소프트웨어 SDDC 솔루션으로, 페도라 리눅스(Fedora Linux) 기반 가상화 관리 플랫폼이다. VMWare[16]와 거의 동일한 기능을 가지며, 커널 기반 가상머신(Kernel-based Virtual Machine)과 리눅스 컨테이너(Linux Container)를 기반으로 컴퓨터 서버를 가상화한다. 또한, 서버 가상화 및 컨테이

너화를 쉽게 관리할 수 있는 웹 기반 인터페이스인 하이퍼바이저(Hypervisor)를 제공한다. 국내 대학에서 여러 학과가 컴퓨터 실습실을 공유하여 팬데믹 대비 및 여러모로 효율적인 컴퓨터 실습 환경 운영을 도모하기 위해 도입한 사례가 있다.

4. ZTNA-enabled SDDC 제어 평가

트래픽 플로우의 네트워크 접근 제어 실험을 위해, 프록스목스와 오픈플로우 스위치, 그리고 네트워크 접근 제어를 담당하는 ZTNA 에이전트로 구성된 제안 ZTNA-enabled SDDC 설계를 기반으로 그림 4와 같은 구조의 로컬 테스트베드(Local Testbed)를 구축한다. SDDC 내부 가상 머신 간 상호 연결을 위해, 가상 오픈플로우 스위치인 OVSbr0를 생성하여 SDDC 내부 가상 네트워크를 구성하고, SDDC의 물리 네트워크 인터페이스인 enp4s0와 OVSbr0는 표 1과 같이 설정하여, 내부 가상 머신과 외부 호스트 간의 상호 연결이 가능하게 한다.

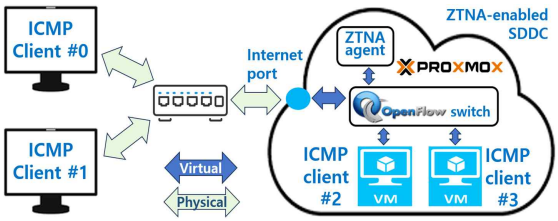

그림 4. 테스트베드 구조
Fig. 4. Testbed Architecture

ICMP(Internet Control Message Protocol) 클라이언트(Client) #2의 IP(Internet Protocol) 주소는 192.168.0.101/24로 설정하고, ICMP 클라이언트 #3의 IP 주소는 192.168.0.102/24로 설정한다. 제어 성능의 확인 실험은 ICMP 클라이언트 #2와 #3의 IP 주소만 이용하여 가능하다.

표 1. SDDC의 네트워크 설정
Table. 1. Network Configuration for SDDC

```
auto lo
iface lo inet manual
iface lo inet loopback ovsbr0
auto enp4s0
iface enp4s0 inet static
        ovs_type OVSPort
        ovs_bridge ovsbr0

auto ovsbr0
iface ovsbr0 inet static
        address 192.168.0.100/24
        gateway 192.168.0.1
        ovs_type OVSBridge
        ovs_ports enp4s0
        network 192.168.0.0
```

4.1 SDDC 외부 플로우 제어 성능 평가

그림 4의 테스트베드에서 SDDC 외부 호스트들과 내부 가상 머신 간 상호 연결에 대한 접근 제어 비교를 위하여 SDDC 외부의 ICMP 클라이언트 #0과 #1에서 SDDC 내부의 ICMP 클라이언트 #2와 #3으로 접근하는 각각의 플로우 #0와 #1을 생성한다. 이때, ZTNA 에이전트에서 표 2와 같이 오픈플로우 스위치 플로우 테이블의 플로우 엔트리를 설정하여, 플로우 #0의 연결은 허용하고, 플로우 #1의 연결은 차단한다.

표 2. SDDC 외부 플로우들의 내부 포워딩 설정
Table. 2. Internal Forwarding Configuration for External Flows of SDDC

```
ovs-ofctl add-flow ovsbr0 ip, nw_dst=192.168.0.101,
        actions=output:fwln101o0
ovs-ofctl add-flow ovsbr0 ip, nw_src=192.168.0.101,
        actions=output:1
```

오픈플로우 스위치는 초기 실행 시 플로우 테이블에 어떠한 플로우 엔트리도 정의되어 있지 않으므로, 표 2와 같이 트래픽 전송을 허용하는 플로우 #0에 대한 플로우 엔트리만 설정하면 된다. 첫 번째 엔트리는 OVSbr0를 경유하는 플로우의 목적지 IP 주소가 192.168.0.101일 경우, 해당 트래픽을 ICMP 클라이언트 #2가 연결된 OVSbr0의 포트 fwln101o0으로 포워딩의 허용을 의미한다. 두 번째 엔트리는 OVSbr0를 경유하는 모든 플로우를 SDDC 외부와 연결된 OVSbr0의 1번 포트로 포워딩의 허용을 의미한다.

그림 5에서는 OVSbr0의 플로우 테이블에 플로우 #0에 대한 플로우 엔트리가 설정되어 있기 때문에, ICMP 메시지가 정상적으로 전송되는 모습을 확인할 수 있다. 반면, 그림 6에서는 플로우 #1에 대한 포워딩 엔트리가 설정되어 있지 않기 때문에, OVSbr0는 해당 패킷을 다음 노드로 전달하지 않으며, 이로 인해 ICMP 메시지가 전송되지 않는 결과를 확인할 수 있다. 표 3은 그림 5와 그림 6의 실험 결과를 비교한 내용을 정리한 것이다.

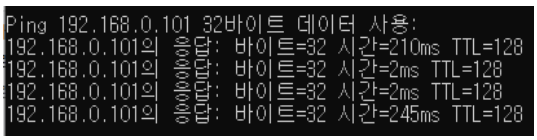

그림 5. 플로우 #0의 실험 결과
Fig. 5. Experimental Results for Flow #0

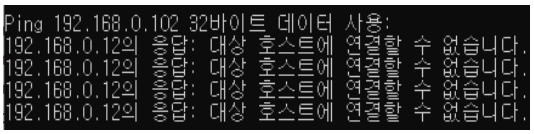

그림 6. 플로우 #1의 실험 결과
Fig. 6. Experimental Results for Flow #1

표 3의 제어 성공률은 ICMP 메시지 전송 시도에 대한 네트워크 접근 제어의 결과를 나타낸다. 플로우 #0의 경우, 총 10,000회의 ICMP 메시지 전송 시도가 모두 성공하여 100%의 제어 성공률을 보인다. 반면, 접근이 허용되지 않은 플로우 #1의 경우, 10,000회의 ICMP 메시지 전송 시도가 모두 차단되어 마찬가지로 100%의 제어 성공률을 보인다. 이는 ZTNA 에이전트가 오픈플로우 스위치의 플로우 테이블에 플로우 #1에 대한 플로우 엔트리를 설정하지 않았기 때문이며, 결과적으로 플로우 #1은 OVSbr0를 통해 다음 노드(Node)로 전송될 수 없게 된다.

표 3. SDDC 외부 플로우들의 실험 결과 비교
Table. 3. Comparison of Experimental Results for External Flows within SDDC

비교 항목	플로우 #0	플로우 #1
전송 시도 횟 수 (회)	10,000	10,000
전송 횟 수 (회)	10,000	0
제어 성공률	100%	100%

4.2 SDDC 내부 플로우 제어 성능 평가

그림 4의 테스트베드에서 SDDC 내부 가상 머신 간 상호 연결에 대한 접근 제어 비교를 위해, 두 개의 가상 머신(ICMP 클라이언트 #2와 #3) 간 상호 연결을 시도하는 플로우 #2와 플로우 #3을 각각 생성한다. ZTNA 에이전트는 표 4와 같이 오픈플로우 스위치의 플로우 테이블에 플로우 엔트리를 설정하여, 플로우 #2의 연결은 허용하고 플로우 #3의 연결은 차단하도록 구성한다.

표 4의 플로우 테이블 설정을 살펴보면, 첫 번째와 두 번째 엔트리는 각각 목적지 IP 주소가 192.168.0.101 및 192.168.0.102일 경우, ICMP 클라이언트 #2와 #3으로의 포워딩을 허용하는 의미이다. 이로 인해 내부 가상 머신 간 상호 연결이 가능하여 플로우 #2는 정상적으로 목적지에 도달할 수 있다. 반면, 세 번째 엔트리는 목적지 IP 주소가 192.168.0.102인 ARP(Address Resolution

Protocol) 요청 트래픽에 대해 포워딩을 허용하지 않는다는 의미이다. 이 설정으로 인해 ICMP 클라이언트 #3에서 클라이언트 #2로의 ARP 요청이 전송되지 않으며, 결과적으로 플로우 #3는 목적지로 전송되지 못한다.

표 4. SDDC 내부 플로우들의 내부 포워딩 설정
Table. 4. Internal Forwarding Configuration for Internal Flows of SDDC

```
ovs-ofctl add-flow ovsbr0 nw_dst=192.168.0.101,
          actions=output:fwln101o0
ovs-ofctl add-flow ovsbr0 nw_dst=192.168.0.102,
          actions=output:fwln102o0
ovs-ofctl add-flow ovsbr0 dl_dst=ff:ff:ff:ff:ff:ff,
          nw_src=192.168.0.102, actions=drop
```

그림 7에서는 OVSbr0의 플로우 테이블에 플로우 #2에 대한 포워딩을 허용하는 엔트리가 설정되어 있어, ICMP 메시지가 성공적으로 전송되는 모습을 확인할 수 있다. 반면, 그림 8에서는 플로우 #3에 대한 포워딩을 허용하지 않는 엔트리가 적용되어 있어, OVSbr0가 해당 트래픽을 다음 노드로 전송하지 않음으로써 ICMP 메시지가 전송되지 않는 것을 확인할 수 있다.

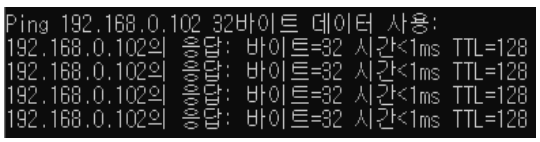

그림 7. 플로우 #2의 실험 결과
Fig. 7. Experimental Results for Flow #2

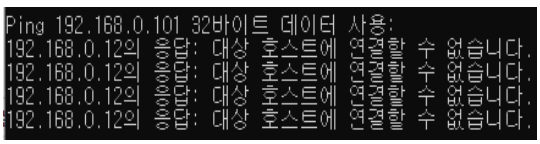

그림 8. 플로우 #3의 실험 결과
Fig. 8. Experimental Results for Flow #3

표 5는 그림 7과 그림 8의 실험 결과를 비교한 내용을 제시한다. 표 5의 제어 성공률은 각 플로우의 전송 시도에 대한 제어 결과를 나타낸다. 플로우 #2의 경우, 총 10,000회의 ICMP 메시지 전송 시도가 모두 성공하여 100%의 제어 성공률을 보인다. 반면, 접근이 허용되지 않은 플로우 #3은 10,000회의 ICMP 메시지 전송 시도가 모두 차단되어 마찬가지로 100%의 제어 성공률을 보인다. 이는 앞선 실험과 마찬가지로, ZTNA 에이전트가 오픈플로우 스위치의 플로우 테이블에 플로우 #3의 ARP 요청 트래픽을 차단하는 플로우 엔트리를 설정하였기 때문이며, 그 결과 OVSbr0를 통해 해당 플로우가 다음 노드로 전송되는 것이 원천적으로 차단되었음을 의미한다.

표 5. SDDC 내부 플로우들의 실험 결과 비교
Table. 5. Comparison of Experimental Results for Internal Flows within SDDC

비교 항목	플로우 #2	플로우 #3
전송 시도 횟 수 (회)	10,000	10,000
전송 횟 수 (회)	10,000	0
제어 성공률	100%	100%

5. 결 론

본 논문에서는 기존 ZTNA 연구들이 주로 정책 수립 수준에 머물렀던 한계를 넘어, 실제 SDDC 기반 클라우드 환경에서 플로우 단위의 접근 제어를 구현하고, 특히 가상 머신 간 상호 연결에 초점을 맞춘 ZTNA-enabled SDDC 설계 방안을 제안하였다. 실험을 통해 제안 구조가 네트워크 트래픽 플로우를 정밀하고 유연하게 제어할 수 있음을 확인하였다.

제안 설계는 SDDC 솔루션과 SDN 솔루션을 통합하여, 오픈플로우 기반 SDN 기술을 통해 트

래픽 플로우의 다양한 속성을 조합함으로써 정밀한 IntServ(Integrated Services)를 구현할 수 있다. 이를 통해 운영 중 발생할 수 있는 다양한 상황에 대해 실시간 대응이 가능하며, ZTNA 에이전트를 통한 중앙 집중식 접근 제어는 사용자 및 장치에 대한 보안 정책의 일관된 적용과 관리 효율성을 크게 향상시킨다. 특히, 이러한 중앙 집중적 제어는 향후 AI(Artificial Intelligence) 기반 환경과의 연계를 통해 지능형 보안 시스템으로의 진화를 가능하게 한다.

또한, SDN의 프로그래머블 특성은 다양한 네트워크 트래픽 엔지니어링 기술의 적용을 유연하게 하며, 전통적으로 하드웨어에 의존하던 네트워크 필수 기능들을 소프트웨어적으로 통합할 수 있게 한다. 이는 클라우드 인프라의 물리적 구성 단순화뿐만 아니라, 효율적이고 미래 지향적인 클라우드 인프라 환경 구축을 가능하게 한다.

향후, 제안 설계를 기반으로 AI 기반으로 자가 학습을 수행하여 이상 행위 탐지 기능과 정책 자동 생성 기능을 포함하는 지능형 ZTNA-enabled SDDC 구조로의 확장을 계획하고 있다. 아울러, 네트워크 트래픽 엔지니어링 기술의 SDDC 적용 방안에 대한 연구를 병행하여, 클라우드 컴퓨팅 인프라의 고도화를 위한 기반 기술 연구를 지속적으로 진행할 것이다.

참 고 문 헌

[1] Paganini, P. et al, “The Impact of Remote Work and Cloud Migrations on Security Perimeters”, Security Affairs, 2024, https://securityaffairs.com/163742/security/remote-work-and-cloud-migrations-impact.html

[2] Mayank G. et al, “Reasons behind growing adoption of Cloud after Covid 19 Pandemic and Challenges ahead”, arXiv, 2021, DOI: 10.48550/arXiv.2103.00176.

[3] Cisco, “Zero Trust: Going Beyond Perimeter Security”, Cisco White Paper, 2021, https://www.cisco.com/c/en/us/solutions/enterprise-networks/zero-trust-security.html

[4] He, Y. et al, “Survey on Zero Trust Architecture: Challenges and Future Trends”, Journal of the Wireless Communications and Mobile Computing, Jan. 2022, DOI: 10.1155/2022/6476274

[5] D. Scott Rose et al, “Zero Trust Architecture”, NIST Special Publication, pp.4-8, Aug. 2020, DOI: 10.6028/NIST.SP.800-207

[6] Pittman, J.M. et al, “Towards a Model for Zero Trust Data”, American Journal of Systems Engineering, pp.18-24, Sep. 2022, DOI: 10.15864/ajse.3103

[7] SLARP (Serial Line Address Resolution Protocol), https://Link4securenetwork.blogspot.com

[8] Syed Muhammad Zohaib. et al, “Zero Trust VPN(ZT VPN): A Systematic Literature Review and Cybersecurity Framework for Hybrid and Remote Work”, Information, Vol. 15, Nov. 2024, DOI: 10.3390/info15110734

[9] Picolae Paladi. et al, “SDN Access Control for the Masses”, arXiv, 2018, DOI: 10.48550/arXiv.1811.08094

[10] Vijay Varadharajan et al, “A Policy based Security Architecture for Software Defined Networks”, IEEE Transactions on Information Forensics and Security, pp. 897-912, Aug. 2018. DOI: 10.1109/TIFS.2018.2868220

[11] N. Ghate, et al, “Advanced zero trust architecture for automating fine-grained access control with generalized attribute relation extraction”, IEICE Proceedings Series, vol. 68, Oct. 2021, DOI:10.34385/proc.68.C1-5

[12] OpenFlow, https://www.opennetworking.org

[13] Proxmox, https://www.proxmox.com

[14] Anders Nygren et al, “OpenFlow Switch Specification ver. 1.5.1”, Open Networking Foundation, pp. 25, March 26, 2015, https://opennetworking.org/wp-content/uplo

ads/2014/10/openflow-switch-v1.5.1.pdf,
[15] Seok Hong Min et al, “Implementation of a Programmable Service Composition Network using NetFPGA-based OpenFlow Switches”, proceedings of the 1st Asia NetFPGA Developers Workshop, Korea, June 2010, http://fif.kr/AsiaNetFPGAws/paper/1-1.pdf
[16] VMWare, https://www.vmware.com

저 자 소 개

민석홍(Seokhong Min)

2015. 8 충남대학교 전자전파정보통신공학과 박사
2024. 9-현재 : 배재대학교 겸임교수
<주관심분야> SDx, DSP, 네트워크 트래픽 엔지니어링