Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.33, NO.6, Dec. 2023 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2023.33.6.961

컨테이너 환경에서의 네트워크 보안 정책 집행 분석*

김 봄,[1†] 이 승 수[2‡]
[1,2]인천대학교 (대학원생, 교수)

Analysis of Network Security Policy Enforcement in Container Environments*

Bom Kim,[1†] Seungsoo Lee[2‡]
[1,2]Incheon National University (Graduate student, Professor)

요 약

현대 컴퓨팅 환경의 변화에 따라 컨테이너화된 워크로드의 보안과 컨테이너 네트워킹의 복잡성 해결이 중요한 쟁점이 되었다. 특히, 네트워크 정책 설정의 복잡성과 클라우드 보안 아키텍처의 부족은 다양한 보안 문제를 야기한다. 본 논문은 컨테이너화된 환경에서 네트워크 보안과 효율성의 중요성에 주목하며, 다양한 컨테이너 네트워크 인터페이스 플러그인들의 보안 기능과 성능을 분석하였다. 특히, Cilium, Calico, Weave Net, Kube-router 별 특징 및 기능들을 비교 및 평가하였으며, 각 플러그인이 제공하는 레이어 3/4 및 레이어 7의 네트워크 정책과 성능 특징에 대한 분석을 진행하였다. 결과적으로, Cilium과 Calico는 레이어 7 프로토콜을 포함한 다양한 보안 정책을 제공하며, Weave Net과 Kube-router는 레이어 3/4에 중점을 둔다. 또한, 레이어 3/4 정책 적용 시 처리량이 감소하는 현상과 레이어 7 정책을 적용할 때 복잡한 처리로 인한 지연 시간의 증가가 확인되었다. 이러한 분석을 통해, 네트워크 보안 정책 및 보안 구성에 대한 이해를 높이고, 향후보다 안전하고 효율적인 컨테이너 네트워킹 환경 구축에 기여할 수 있을 것으로 기대된다.

ABSTRACT

With the changes in the modern computing landscape, securing containerized workloads and addressing the complexities of container networking have become critical issues. In particular, the complexity of network policy settings and the lack of cloud security architecture cause various security issues. This paper focuses on the importance of network security and efficiency in containerized environments, and analyzes the security features and performance of various container network interface plugins. In particular, the features and functions of Cilium, Calico, Weave Net, and Kube-router were compared and evaluated, and the Layer 3/4 and Layer 7 network policies and performance features provided by each plugin were analyzed. We found that Cilium and Calico provide a wide range of security features, including Layer 7 protocols, while Weave Net and Kube-router focus on Layer 3/4. We also found a decrease in throughput when applying Layer 3/4 policies and an increase in latency due to complex processing when applying Layer 7 policies. Through this analysis, we expect to improve our understanding of network policy and security configuration and contribute to building a safer and more efficient container networking environment in the future.

Keywords: Container Security, CNI, Kubernetes, Security Policy

Received(09. 04. 2023), Modified(10. 11. 2023), Accepted(11. 02. 2023)
* 이 논문은 인천대학교 2023년도 자체연구비 지원에 의하여 연구 되었음.
† 주저자, zxx0313@inu.ac.kr
‡ 교신저자, seungsoo@inu.ac.kr(Corresponding author)

I. 서 론

최근 디지털 전환 이니셔티브에 따른 컨테이너화 애플리케이션과 쿠버네티스의 채택은 현대 컴퓨팅 환경의 중요한 변화이다. 컨테이너 기반의 애플리케이션 배포는 서비스의 확장성, 유연성 및 재사용성이라는 강점으로 구글, AWS, Microsoft, IBM을 비롯한 세계적 기업뿐만 아니라 국내 금융권, 전자상거래, 게임사 등 다양한 분야의 기업들도 쿠버네티스와 같은 플랫폼을 채택하고 있다[5][6].

특히, 컨테이너 기반의 네트워킹 환경에서 네트워크 보안 정책 설정은 복잡한 과정이다. 네트워크 정책은 컨테이너 간 통신을 제어하는데, 부적절한 설정은 네트워크 성능 저하와 보안 문제의 위험성이 있다. 이러한 문제를 근본적으로 해결하기 위해, 본 논문은 컨테이너 네트워크 인터페이스(CNI, Container Network Interface)[1]의 네트워크 보안 정책 기능을 분석하고 처리 성능을 비교한다.

본 논문이 기여하는 바는 다음과 같다. :

- 다양한 CNI 별 보안 기능을 체계적으로 분석하고 네트워크 정책 기능에 대해 명확히 이해시켜 준다. 이로써 사용자는 각 CNI의 보안 기능과 정책을 바탕으로 더욱 안전한 컨테이너 환경을 구축할 수 있는 기초 정보를 얻을 수 있다.
- CNI 별 네트워크 정책의 처리 성능을 실제 환경에서 체계적으로 비교 및 분석한다. 특히, 레이어 3/4와 레이어 7에서의 네트워크 정책 적용 시의 성능 차이를 검토함으로써, CNI을 선택하고 네트워크 정책을 구성할 때 참고할 수 있는 중요한 지표를 제공한다.
- CNI 및 네트워크 정책의 효과적인 선택과 확고한 기준을 고려하도록 돕기 위해, 다양한 환경과 요구 사항에 따라 어떤 CNI 및 네트워크 정책이 적합한지에 대한 논의를 제공한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 배경지식을 소개하고, 3장에서는 관련 연구에 대해 언급한다. 4장은 컨테이너 네트워크 인터페이스 플러그인의 네트워킹과 보안 기능을 분석하며, 5장에서 보안 기능 중 네트워크 정책 기능을 중점적으로 제시한다, 6장에서는 특정 시나리오 기반의 처리 성능 분석 실험 결과를 제시한다. 7, 8장에서는 논의와 결론으로 맺는다.

1) 본 논문에서는 앞으로 간략하게 'CNI'라고 표기한다.

II. 배경 지식

2.1 컨테이너화와 쿠버네티스

2.1.1 컨테이너화 (Containerization)

Fig 1의 가상화는 물리적 하드웨어 리소스를 추상화하여 독립적인 가상 머신을 생성하는 기술이다. Fig 1(a)에 따르면, 하이퍼바이저를 이용해 여러 운영체제를 동시에 구동하고 각 가상 머신은 독립된 리소스와 운영체제를 포함한다. 하지만, 가상 머신 환경의 불일치라는 한계가 존재한다.

반면에, 컨테이너화는 애플리케이션과 그 의존성을 패키징하는 기술이다. Fig 1(b)과 같이 하나의 운영체제 위에서 여러 애플리케이션을 독립적으로 실행하며, 이를 '컨테이너'로 관리한다. 컨테이너는 호스트 OS의 커널을 공유하기 때문에 가상 머신보다 경량화되고, 환경의 불일치 문제도 감소한다.

이러한 컨테이너화의 특징을 활용하여 Netflix, Spotify, Twitter와 같은 대기업들은 대규모 서비스를 제공한다[7]. 그중 Spotify는 세계적인 오디오 스트리밍 플랫폼으로, 큰 규모의 서비스를 제공하기 위해서 수백 또는 수천 개의 복잡한 서버, DB, 서비스와 응용 프로그램 등 다양한 컴포넌트가 필요한데, 컨테이너 기술을 활용하여 복잡한 마이크로서비스 구조를 효과적으로 관리하며, 서비스를 배포 및 확장한다. 이 외에도 IoT, 클라우드 마이그레이션 등에서 컨테이너 기술은 다양한 이점을 제공한다.

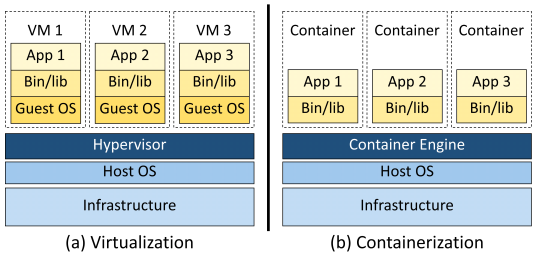


Fig. 1. The Comparison of the Virtualization and the Containerization[8]

2.1.2 쿠버네티스 (Kubernetes)

컨테이너 오케스트레이션은 여러 컨테이너를 효율

적으로 관리하는 기술로, 컨테이너의 배포, 확장 및 안정적인 운영을 지원한다. 여러 오케스트레이션 중에서 Docker Swarm[9], Apache Mesos [10], 특히 Kubernetes[11]가 대표적이다. 쿠버네티스(Kubernetes)는 현대 컨테이너 기반 인프라에서 널리 사용되는 오케스트레이션 플랫폼 중 하나로, 컨테이너화된 워크로드와 서비스 관리에 있어 높은 이식성과 확장성을 제공하는 오픈소스 플랫폼이다. 주요 특징으로는 로드 밸런싱, 스토리지 오케스트레이션, 자동화된 롤 아웃 및 롤백, 구성 관리 등의 기능을 제공한다.

2.2 컨테이너 네트워크 인터페이스 (Container Network Interface)

컨테이너 네트워크 인터페이스(CNI)는 컨테이너 간 통신을 제어하고 관리하기 위한 표준화된 플러그인으로, Cloud Native Computing Foundation (CNCF)[12]에서 관리한다. CNI의 주요 설계 목적은 다양한 컨테이너 런타임과 오케스트레이션 간에 일관된 네트워크 계층을 제공하는 것이며, 네트워크 연결 방식의 차이를 최소화하여 표준화를 위해 도입되었다. 주로 쿠버네티스 클러스터 내의 파드(Pod)[2]의 네트워크 구성과 연결을 담당하고, 쿠버네티스의 가상 네트워크를 구축한다. IP, 서브넷, 라우팅 테이블 등의 네트워크 관련 작업을 처리한다. 또한 ARP 요청에 대한 프록시 응답을 제공하고 다양한 인터페이스를 통해 컨테이너별 네트워크 설정을 관리한다. 파드를 생성할 때 해당 파드의 모든 컨테이너는 동일한 네임스페이스를 공유하며, 이때의 네트워크 설정을 CNI가 담당한다. CNI는 CNI Provider를 통해 다양한 네트워킹 전략을 사용하여 컨테이너 간의 통신 패브릭을 구성하고, 여러 네트워크 인프라 기술을 통합 및 제공한다.

쿠버네티스는 Kubenet라는 CNI를 제공하지만, 그 자체로는 노드 간 교차 네트워킹조차 지원하지 않기 때문에 네트워크의 다양한 요구 사항을 만족시키기 위해 Cilium[13], Calico[14] 과 같은 서드파티 플러그인이 필요하다.

2) 1개 이상의 컨테이너 그룹으로 쿠버네티스에서의 컨테이너를 칭하는 최소 단위

2.3 쿠버네티스의 네트워크 정책 (Kubernetes Network Policy)

Fig 2의 YAML 파일처럼, 쿠버네티스의 네트워크 정책[18]은 레이블, IP, 포트, 프로토콜을 기반으로 설정하여 원하는 네트워크 연결 상태를 구성하거나 특정 트래픽을 허용 및 제한하도록 구성하여 클러스터 내의 네트워크 통신을 세밀하게 제어한다. 기본적으로 클러스터 기본 설정에 정책이 적용되지 않으므로 명시적으로 설정해야 한다.

기본적으로 네트워크 정책은 Allow와 Deny 규칙을 사용하여 트래픽을 제어한다. Allow는 특정 트래픽을 허용하며, Deny는 특정 트래픽을 차단한다. 화이트리스트 원칙에 따라 정책이 적용되지 않은 경우, 해당 네임스페이스의 모든 트래픽이 기본적으로 허용된다. 정책이 적용된 파드는 해당 정책에 부합하는 트래픽만 수용하고 나머지는 차단한다. 또한, 셀렉터를 통해 특정 레이블을 가진 파드를 식별하며 트래픽의 방향, 출발지, 목적지를 설정한다. 실제 패킷의 필터링은 CNI 플러그인을 통해 수행되므로 구현과 적용은 CNI 플러그인의 역할이며, 쿠버네티스는 네트워크 정책의 명세만 제공한다.

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test-network-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
    - Ingress
    - Egress
  ingress:
    - from:
        - ipBlock:
            cidr: 172.17.0.0/16
            except:
              - 172.17.1.0/24
        - namespaceSelector:
            matchLabels:
              project: myproject
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 6379
  egress:
    - to:
        - ipBlock:
            cidr: 10.0.0.0/24
      ports:
        - protocol: TCP
          port: 5978
```

Fig. 2. The Example of Network Policy (YAML)

III. 관련 연구

쿠버네티스의 CNI 플러그인에 대한 기존 연구들은 다양한 측면에서의 분석을 중심으로 진행되었다. Jin 등[1]은 패킷 처리의 원리와 성능, 그리고 자원 사용량의 차이를 포괄적으로 조명했다. Cui 등[2]은 CNI 기반 네트워킹 기술과 구조의 차이에 따라 발생하는 성능과 기능의 차이를 분석했다. 또한, Qi 등[3]은 다양한 오픈 소스 CNI의 오버헤드와 병목 현상에 대한 세부 성능을 측정하여 제시했다. 특히, 데이터 패스, iptables, 그리고 호스트 네트워크 스택 간의 상호 작용에 중점을 둔 성능 평가를 통해 CNI 환경 구축의 방향성을 제시했다. 이러한 연구[1]-[3]는 CNI 플러그인에 대한 기본적인 이해를 제공하나, 네트워크 보안 정책의 세부적인 분석에서는 상대적으로 부족함이 있다.

Budigiri 등[4]은 엣지 컴퓨팅 솔루션 배포의 네트워크 보안과 성능에 대한 중요성을 주목하였다. 이 연구는 네트워크 정책이 컨테이너 간의 통신 흐름에 큰 영향을 미치며, 컨테이너화된 애플리케이션의 보안 강화의 핵심적인 역할을 한다는 점을 강조하였다. 그러나, 이 연구는 주로 동적 배포의 측면과 엣지 환경에서의 성능만 분석이 수행되었고, 특정 CNI 플러그인 간의 상세한 성능과 보안 기능 비교에는 한계가 있었다.

본 연구는 이러한 공백을 해소하기 위해, 주요 CNI 플러그인의 보안 기능과 네트워크 정책에 대해 세부적으로 분석하고, 이에 따른 처리 성능 결과를 제시한다. 이로써, CNI 플러그인 선택에 있어 중요한 참고 자료를 제공하고자 한다.

IV. CNI에 따른 기본 및 보안 기능 분석

4.1 쿠버네티스 CNI 플러그인 별 기본 분석

4.1.1 Cilium

Fig 3과 같이, Cilium은 Extended Berkeley Packet Filter (eBPF)를 활용한 CNI 플러그인이다. Cilium의 데이터 흐름은 네트워크 장치에서 시작되어 TCP/IP 스택과 소켓을 거쳐 프로세스에 도달한다. 커널 내에서 eBPF Maps와 syscall을 연결하여 데이터 흐름과 동작을 조정하며 최적화한다. eBPF 프로그램은 Verifier를 통해 안정성을

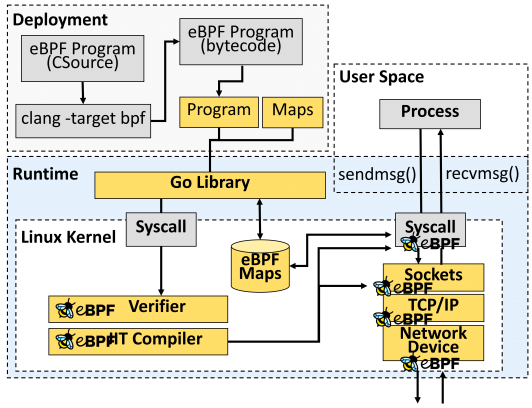


Fig. 3. Cilium Networking with eBPF

확보하고, JIT 컴파일러로 바이트 코드를 네이티브 코드로 변환하여 실행된다. Cilium의 eBPF 프로그램 배포와 맵 관리는 런타임에서 Go 라이브러리에 의해 이루어진다. 이러한 과정을 통해 사용자 공간과 프로세스 간의 통신을 최적화한다. 기존 iptables 의존도가 높은 환경에서는 규칙 증가에 따른 병목 현상과 성능 저하가 발생한다. Cilium은 eBPF를 통해 기존의 한계를 극복하여 커널 수준에서 높은 성능과 보안에 기여한다. Cilium은 모니터링, 패킷 추적과 같은 고급 기능을 제공하고, 오버레이, Geneve 같은 캡슐화 방식도 지원한다. 추가로 VXLAN을 통한 교차 노드 통신, Istio를 활용한 서비스 메시 환경을 지원한다.

Cilium의 다양한 기능과 복잡한 설정은 높은 러닝 커브를 가져올 수 있으나 세분화된 네트워크 제어와 대규모 환경에서의 관리에 이상적이다.

4.1.2 Calico

Fig 4에서 보이듯이, Calico는 IP 라우팅을 통해 서비스 간의 네트워크 트래픽을 제어하는 CNI 플러그인이다. 사용자는 구성 파일 및 네트워크 정책을 Datastore에 저장한다. Typha는 Datastore의 변화를 감지하고, 해당 정보를 Felix에 전달한다. Felix는 노드에 정책을 적용하고, BIRD와 Confd는 라우팅 관리를 담당한다. BIRD는 Calico의 BGP 클라이언트로, 다른 노드나 외부 BGP 피어와의 라우팅 정보 교환을 담당한다. Calico는 기존 레이어 2 네트워킹의 복잡성과 성능 저하 문제를 해결하기 위해 레이어 3 라우팅을 활용한다. BGP를

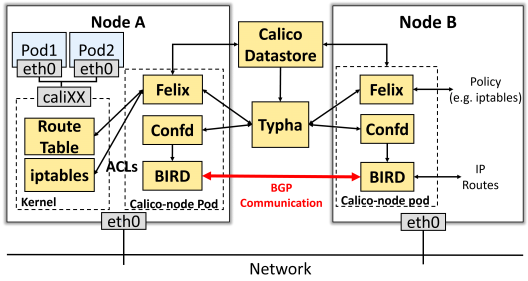


Fig. 4. Calico Networking with BGP

이용하여 라우팅 정보를 공유하는 방식으로, 클러스터 내의 라우팅 테이블 크기를 최소화한다. 또한 동적 라우팅과 다른 네트워크와의 자동 피어링이 가능하게 한다. 이러한 접근 방식 덕분에 소규모 클러스터에서도 가벼운 라우팅이 가능해진다. 특히 Calico는 다양한 환경에서 일관된 네트워킹과 정책을 제공한다. IP 주소를 직접 할당하여 NAT의 필요성을 제거하고, 패킷 전달 오버헤드도 줄여준다.

비록 Calico의 초기 설정과 관리가 복잡할 수 있지만, 다양한 클라우드 및 온프레미스 환경에서 일관적이고 안정성 있게 운영하도록 한다.

4.1.3 Weave Net

Weave Net은 오버레이 네트워크를 기반으로 한 CNI로, 컨테이너화된 애플리케이션의 배포 및 관리를 물리 네트워크 구조에 상관없이 효율적으로 수행한다. 이 오버레이 네트워크는 클러스터의 모든 노드를 연결하여 컨테이너 간 통신을 가능하게 한다. 클러스터의 각 노드에 설치된 Weave Net 에이전트는 IP 주소를 할당하고 패킷 라우팅을 담당한다. Weave Net은 레이어 2에서 MAC 주소를, 레이어 3에서는 IP 주소를 활용해 통신한다. 별도의 외부 저장소나 복잡한 설정 없이 안정적인 네트워크 환경을 제공하며 네트워크 파티션 처리, 장애 복구 및 보안 기능도 포함한다. 추가로 DNS를 이용해 컨테이너 간의 서비스 검색 기능을 제공하는데, 컨테이너 간에 이름을 통해 서로를 찾아 통신하므로 동적 스케일링이 필요한 환경에서 유용하다.

Weave Net은 간결하고 사용하기 쉽지만, 복잡한 네트워크 구조를 가지므로 소규모 환경에 적합하다.

4.1.4 Kube-router

Kube-router는 BGP와 레이어 3 기반의 Flat 네트워크 구조를 사용하는 CNI이다. 오버레이를 사용하지 않으므로 구조가 단순하며, 모든 노드를 통합된 네트워크의 일부로 간주한다. BGP 라우팅을 활용하여 클러스터와 외부 네트워크 간의 연결을 효율적으로 관리한다. 클러스터 내의 모든 노드를 Kube-router에서 자동으로 감지하므로 직접적인 통신을 가능하게 한다. kube-proxy를 대체하는 IPVS와 LVS 기반의 서비스 프록시를 제공하는데, IPVS의 해시 기반 방식은 iptables의 체인 방식에 비해 빠른 성능을 보인다. 또한 여러 기능이 독립적인 컴포넌트로 구성되어 있어 사용자는 필요에 따라 특정 기능만 선택적으로 활성화할 수 있다.

Kube-router는 기본적인 기능에 초점을 맞췄기 때문에 가볍지만, 고급 기능이나 커스터마이징 측면에서 제한적이다.

4.2 CNI에서 지원하는 보안 기능 분석

다양한 CNI 솔루션들은 보안 기능 측면에서 서로 다른 특징과 기능을 제공하므로, Table 1을 참조하여 지원하는 보안 기능에 대해 분석한다.

4.2.1 Cilium

Cilium은 네트워크 정책 기능을 통해 특정 파드, 서비스, 혹은 엔드포인트 간의 통신을 허용하거나 제한하는 세분화된 액세스 제어가 가능하다. 워크로드 간의 미세 조정된 트래픽 흐름을 허용하는 이 기능은 보안 및 네트워크 접근 제어의 강화에 중요하다. 쿠버네티스 레이블 기반의 정책 적용을 통해, 파드의 레이블을 활용한 트래픽 제어가 가능하다. 더불어, DNS 기반의 네트워크 정책도 지원하며 FQDN을 사용하여 외부 서비스 접근을 허용하거나 제한할 수 있다. 더불어 CIDR 기반의 정책을 지원하기 때문에 IP 주소 범위에 따른 트래픽 제어도 가능하다.

레이어 7의 트래픽을 검사하고 제어할 수 있는 자체 API 기반의 보안 기능도 제공한다. 이를 통해 HTTP/HTTPS 요청이나 gRPC 호출 같은 애플리케이션 레벨의 통신을 세밀하게 제어할 수 있으며, 악성 트래픽이나 특정 API 호출을 차단하는 것이 가능하다. IP 주소가 아닌 엔드포인트의 실제 신원

Table. 1. Security Capabilities Comparison of CNI

CNI	Network Policy	Identity Based Security	TLS Encryption	Support eBPF	Host Security	Multi-Cluster
Cilium	Yes	Yes (Endpoint Identity)	Yes (Wireguard, IPsec)	Yes	Yes (Host-based policies)	Yes (ClusterMesh)
Calico	Yes	Yes (Endpoint Identity)	Yes (Wireguard)	Partial	Yes (Host Endpoint security)	Yes (Federation)
Weave Net	Yes	No	Yes (IPSec)	No	No	No
Kube-router	Yes	No	No	No	No	No

(예: 파드의 메타데이터나 레이블)을 기반으로 통신을 제어하므로 네트워크 주소가 변경되더라도 통신 정책은 유효하게 유지된다. 더불어, Wireguard 및 IPsec을 지원하여 트래픽을 암호화하여 중간자 공격을 방지하고, 데이터의 기밀성을 보장한다. eBPF를 활용한 빠른 패킷 처리와 복잡한 네트워크 정책 실행, 호스트 레벨의 네트워크 트래픽 제어 기능은 클러스터 내외부 통신을 안전하게 관리한다. 마지막으로 ClusterMesh 기능을 통해 여러 클러스터 간의 네트워크 통신을 제어하고 관리할 수 있다. 이는 대규모 환경에서의 워크로드 통신을 효율적으로 관리하며, 보안을 강화하는 데 도움을 준다.

4.2.2 Calico

Calico는 강력한 네트워크 정책 기능을 제공하여 파드 간 통신을 세밀하게 제어한다. Ingress 및 Egress 규칙을 정의하여 특정 트래픽을 허용하거나 차단할 수 있으며, 이로써 보안 그룹과 유사한 기능을 구현한다. 네트워크 세그먼테이션과 보안의 세부 적용 역시 가능하다. Calico는 Kubernetes 레이블을 활용하여 정책을 적용한다. 비록 Calico는 DNS 정책과 FQDN 기반의 정책을 무료로 지원하지 않지만, GlobalNetworkSet 리소스를 이용하면 도메인 이름을 IP 주소로 변환하는 것이 가능하다. 또한, CIDR 기반의 정책을 지원하여 특정 IP 범위에 대한 트래픽을 제어할 수 있다.

Calico의 자체 API를 통해 네트워크 정책을 동적으로 구성하고 관리할 수 있다. 사용자가 정의한 네트워크 정책이 Calico API에 의해 적용되어, 보안 요구 사항에 따라 런타임에서 네트워크 트래픽을 조절하게 된다. 더불어, 엔드포인트 기반의 보안 기능을 지원하며, 각 파드는 고유의 엔드포인트 ID를 가지고 있다. Calico는 이 ID를 기반으로 트래픽을 검증하고, 파드 간 트래픽의 정합성을 확인한다. Wireguard를 통한 TLS 암호화도 지원되어, 네트워크 트래픽의 기밀성과 무결성을 보장한다.

Calico는 eBPF의 일부 기능을 지원하며, eBPF를 사용하면 커널 수준에서 높은 성능의 패킷 처리 및 네트워크 관측이 가능하다. 더불어, 호스트 수준에서의 보안을 강화하는 Host Endpoint security를 지원하여, 호스트 자체의 네트워크 정책을 적용하고 워크로드와 네트워크 자원을 보호한다. 마지막으로, Calico Federation을 통해 멀티 클러스터 환경에서도 라우팅 정보 교환과 로깅 기능을 강화하여, 네트워크 확장성과 관리의 효율성을 높인다.

4.2.3 Weave Net

Weave Net은 쿠버네티스 네트워크 정책 API를 활용하여 쿠버네티스 레이블 기반의 네트워크 정책 기능을 지원한다. 이 기능을 통해, 특정 파드나 서비스 간의 통신을 허용하거나 제한하는 설정을 구성할 수 있다. 이를 통해 관리자는 의도치 않은 워크로드 간 통신을 제한하거나, 특정 서비스만을 허용하는 등 세부적인 네트워크 제어가 가능하다. Weave Net은 FQDN 기반의 네트워크 정책은 지원하지 않지만, CIDR를 활용한 트래픽 제어는 가능하다.

또한, Weave Net의 특징 중 하나는 웹 UI를 통해 사용자에게 정책의 영향과 전체 네트워크 상태를 직관적으로 확인할 수 있게 제공한다는 점이다. 이외에도, IPsec를 기반으로 한 TLS 암호화 기능

을 지원하여 데이터의 기밀성, 무결성 및 인증을 보장하며, 트래픽의 도청이나 조작을 방지한다.

4.2.4 Kube-router

Kube-router도 Weave Net과 같이, 쿠버네티스 네트워크 정책 API를 완전히 지원하고, 쿠버네티스 레이블을 활용한 네트워크 정책 구성을 가능하게 한다. FQDN 기반의 네트워크 정책은 지원하지 않지만, CIDR를 통한 트래픽 제어는 가능하다.

Kube-router의 특성상, 쿠버네티스 기본 네트워크 정책을 사용한다고 해도 Ingress 트래픽 제어만 가능하며 Egress 제어는 제공하지 않는 특징이 있다. Kube-router는 네트워크 정책을 중심으로 한 기본 보안 기능을 제공하고, 고급 보안 기능은 다른 도구와의 통합이 필요하다.

V. CNI에 따른 네트워크 정책 분석

CNI는 네트워크 정책 기능을 제공함으로써 클러스터의 안정성과 효율성 향상에 기여한다. 본 섹션에서는 Table 2를 기반으로 네트워크 정책 기능에 사용되는 네트워크 정책을 세부적으로 분석한다.

5.1 쿠버네티스 기본 네트워크 정책

쿠버네티스 기본 네트워크 정책은 레이어 3, 4를 지원하며, 네임스페이스 내에서 Ingress와 Egress 트래픽 규칙을 설정하여 트래픽 흐름을 제어하고 파드 간의 통신 규칙을 설정한다. 이를 통해 네트워크에 대한 보안을 제공하고, 워크로드를 격리하여 네트워크 제어를 세분화한다. Pod Selector, From/To Selector를 통해 대상 파드를 상세히 지정하여 특정 파드 그룹의 통신을 제어한다.

하지만 이 정책에는 명확한 한계가 있다. 트래픽 허용 정책만 명시할 수 있기 때문에 트래픽 거부 방식이 제한적이다. 그리고 특정 보안 이벤트 로깅, 공통 게이트웨이의 트래픽 라우팅 지원, LocalHost 트래픽 차단 기능이 없으며 네임스페이스에 국한된 범위로, 전체 클러스터에 대한 정책 설정이 불가하다. 반면, 다른 CNI 플러그인의 자체 네트워크 정책은 복잡한 환경에서 로깅, 모니터링, 라우팅, 멀티 클러스터 지원 등의 다양한 기능을 제공한다. 따라서 보안적인 측면에서는 세부적으로 제어할 수 있도록 확장된 네트워크 정책을 가진 CNI 사용을 고려해 볼 필요가 있다.

5.2 Cilium 네트워크 정책

Cilium 네트워크 정책은 레이어 3 및 4에서의 IP와 포트를 활용한 ACL 기반 트래픽 제어와 함께 레이어 7에서의 고급 프로토콜 인식 기능을 제공한다. 이러한 레이어 7 프로토콜 인식은 HTTP, gRPC, Kafka 외에도 DNS, MQTT, 메시징 프로토콜 등 다양한 프로토콜에 대한 트래픽 제어와 분석을 가능하게 한다. 이를 통해 사용자는 서비스 간의 세밀한 트래픽 제어를 수행할 수 있다. Endpoint Selector와 Node Selector는 파드 뿐만 아니라 다른 엔드포인트나 노드를 정책의 대상으로 지정하는 데 사용된다. 트래픽 제어 메커니즘은 Allow와 Deny를 활용하여 특정 트래픽을 허용하거나 제한하며, 이를 통해 다양한 프로토콜 및 서비스에 대한 정밀한 정책 설정이 가능하다.

Cilium의 정책 체계는 크게 두 가지로 구분된다.

Table 2. Summary of the Details and Support for the Network Policies by CNI

CNI	Controll Traffic		Support for Independent Network Policy	About Network Policies			
	Ingress	Egress		Layer	Type	Endpoint	Action
Cilium	Yes	Yes	**Yes** (CiliumNetworkPolicies)	L3,L4 ,L7	Ingress, Egress	Pod,Service, Node	Allow, Deny
Calico	Yes	Yes	**Yes** (CalicoNetworkPolicies)	L3,L4	Ingress, Egress	Pod,VM, Host Interface	Allow, Deny, Pass,Log
Weave Net	Yes	Yes	**No** (K8sDefaultNetworkPolicy)	L3,L4	Ingress, Egress	Pod	Allow
Kube-router	Yes	No	**No** (K8sDefaultNetworkPolicy)				

CiliumNetworkPolicy (CNP)는 특정 네임스페이스 범위 내에서의 파드 통신 정책을 정의하고 관리한다. CNP를 사용하여 특정 네임스페이스의 파드 간의 세밀한 통신 규칙을 지정할 수 있다. CiliumClusterWideNetworkPolicy (CCNP)는 클러스터 전체의 통신을 제어하는 정책으로, 네임스페이스의 경계를 초월하여 트래픽을 제어한다. 이는 클러스터 전체에 걸쳐 공통된 정책을 적용하거나, 특정 네임스페이스를 제외한 클러스터 전체를 적용하는 등의 활용이 가능하다.

이러한 체계를 통해 Cilium 사용자는 네임스페이스별로 세분화된 통신 정책을 설정하거나, 클러스터 전체에 적용되는 통합된 정책을 구성할 수 있다. 이는 높은 수준의 유연성과 보안을 동시에 제공하며, 복잡한 마이크로 서비스 아키텍처에서도 효과적인 네트워크 관리가 가능해진다.

5.3 Calico 네트워크 정책

Calico 네트워크 정책은 레이어 3, 4의 트래픽 제어를 지원하고 Ingress와 Egress 규칙으로 세부적인 트래픽을 관리한다. Label Selector, Name space Selector 등 여러 셀렉터를 통해 정책의 적용 대상을 상세하게 지정한다. Calico는 레이어 7 프로토콜인 HTTP/HTTPS 트래픽 관리를 부분적으로 지원하는데, 이를 통해 웹 애플리케이션의 경로, 메서드, 헤더를 기반으로 하는 정책을 설정할 수 있다. 즉, Calico 정책을 사용하면, 예를 들어, GET 요청만을 허용하거나 특정 경로에 대한 접근을 제한하는 것이 가능하다. 트래픽 허용 및 차단은 Allow와 Deny 기능을 통해 이루어지며, Pass 기능은 조건에 맞는 트래픽을 문제없이 허용하고 나머지 트래픽을 다음 정책으로 넘긴다. 또한, Log 기능을 통해 트래픽 정보를 syslog에 기록하며, 이를 통해 네트워크 모니터링과 디버깅에 유용하다.

Calico는 쿠버네티스의 기본 네트워크 정책 외에도 CalicoNetworkPolicies를 제공하여, 트래픽을 제어하는 규칙 집합을 정의하고, 네트워크 피어와의 통신을 제어한다. 'order' 필드를 통해 정책 간의 우선순위를 지정할 수 있다.

Calico 네트워크 정책 체계는 크게 두 가지 범위를 제공함으로써, 클러스터 내에서 간단한 네트워크 정책부터 복잡한 전체 클러스터 범위의 정책까지 다양한 요구사항을 충족시킬 수 있다. Calico의 NetworkPolicy는 특정 네임스페이스 내에서만 작동한다. 이를 통해, 네임스페이스별로 독립적인 네트워크 정책을 구성하고 관리할 수 있다. 파드 레이블뿐만 아니라 서비스나 네임스페이스 레이블을 기반으로 설정하는 것도 가능하다. 이에 따라 유연한 정책 구성이 가능하다. GlobalNetworkPolicy는 클러스터 전체에 걸쳐 적용된다. 이는 모든 네임스페이스의 워크로드와 호스트에 대한 엔드포인트에 적용되며 Calico의 엔드포인트 그룹과 호스트 엔드포인트 간의 네트워크 연결 규칙을 정의한다.

VI. CNI에 따른 네트워크 정책 처리 성능 분석

6.1 실험 목적 및 방법

본 실험은 CNI 플러그인별로 레이어 3/4 및 레이어 7 네트워크 정책이 쿠버네티스 클러스터 내 트래픽에 미치는 영향을 측정하였다. 실험 환경은 Intel(R) Xeon(R) Silver 4210R CPU 2.40 GHz, 256GB RAM, 2TB HDD를 탑재한 서버에서 Ubuntu 22.04와 5.15.0-76-generic x86-64

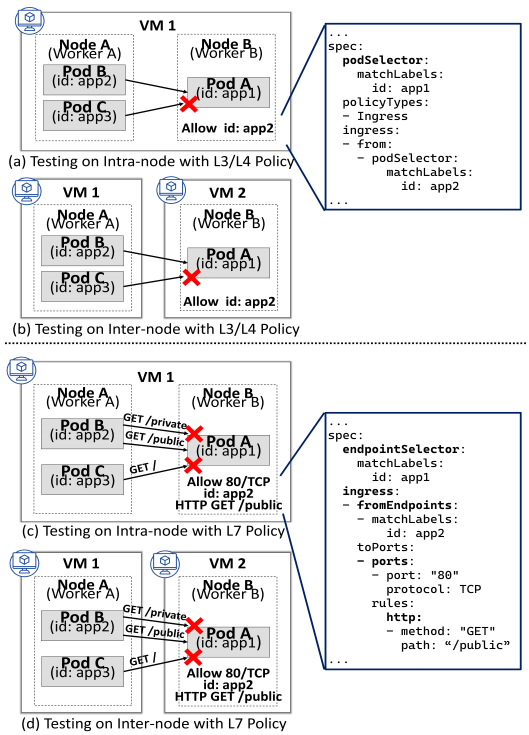


Fig. 5. Test Scenarios for Enforcing L3/L4 Network Policies

리눅스 커널 기반으로, 3대의 가상 머신을 사용하여 각 CNI로 쿠버네티스 클러스터를 설정하였다. 가상 머신 중 하나는 쿠버네티스 마스터 노드로 작동하고 나머지 가상 머신들은 워커 노드의 역할을 하도록 구성하였다. 네트워크 정책의 영향을 파악하기 위해 아래와 같은 시나리오로 처리량과 지연시간을 측정하였고, 레이어 3/4에서는 iperf와 netperf, 레이어 7에서는 Apache Benchmark를 사용하였다.

Fig 5의 (a)와 (b)는 레이어 3/4 정책 적용 시의 통신 시나리오를, (c)와 (d)는 레이어 7 정책 적용 시의 통신 시나리오를 각각 나타낸다. 레이어 3/4에서는 'id: app1' 레이블의 파드로의 인바운드 트래픽 중 'id: app2' 레이블의 파드로부터 오는 트래픽만을 허용하였다. 레이어 7 정책은 'id: app1' 레이블의 엔드포인트로의 트래픽을 제어하며, 오직 'id: app2' 레이블의 엔드포인트로부터 80번 포트의 TCP 트래픽 중 HTTP 'GET' 메서드와 '/public' 경로로 시작하는 요청만을 허용하였다.

6.2 레이어별 네트워크 정책 적용에 따른 성능 비교

Fig 6은 레이어 3/4 정책 적용 전후의 처리량 차이를 보여준다. 노드 내 통신에서는 모든 CNI 플러그인에서 처리량이 감소하지만, 노드 간 통신에서는 일부 증가하는 추세가 나타났다. Fig 7은 지연 시간 변화를 나타낸다. 노드 내 통신에서 Cilium이, 노드 간 통신에서는 Kube-router가 가장 낮은 지연 시간을 보였다. 레이어 7 정책 적용 전후의 성능 영향을 평가하기 위해 이를 지원하는 Cilium 자체 정책을 이용하기 위해 Cilium이 설치된 환경에서만 실험하였다. Fig 8은 레이어 7 정책 적용 전후의 처리량을 비교한다. 정책 적용 후 CiliumNetwork-Policy에서 '/public' 경로의 처리량이 예상과 다르게 상승하지 않았는데, 이는 Cilium 네트워크 정책

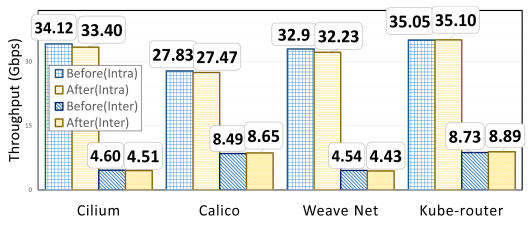


Fig. 6. Throughput Analysis of CNI Before and After Applying L3/L4 Network Policies on Intra/Inter-Node

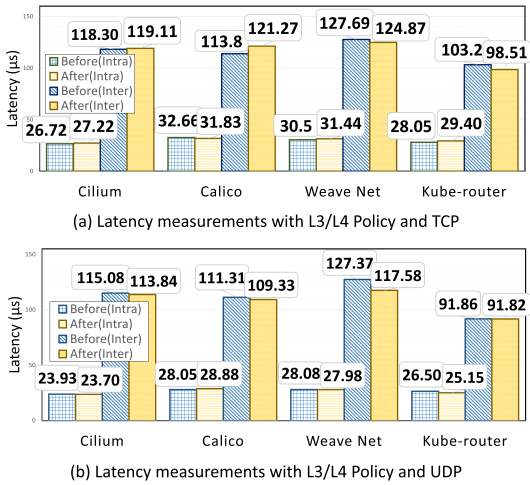


Fig. 7. Latency Impact of L3/L4 Network Policies on Intra/Inter-Node

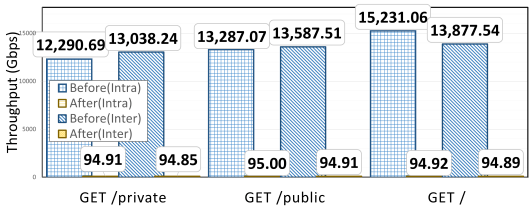


Fig. 8. Throughput Analysis before and after applying Intra/Inter-node L7 network policies in Cilium

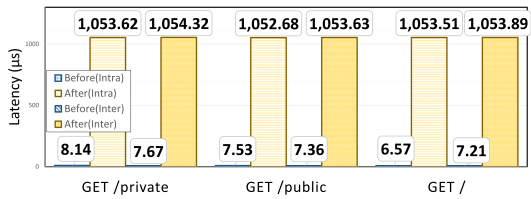


Fig. 9. Latency Impact of L7 Network Policies on Intra/Inter-Node in Cilium

의 오버헤드 때문으로 해석된다. Fig 9에서는 레이어 7 정책 적용 후의 지연 시간이 대략 140배 증가하였으며 허용된 경로 '/public'에서도 확인되었다. 이에 대한 원인 분석은 섹션 6.4에서 다룬다.

전반적으로 노드 내 통신은 트래픽이 로컬로 전송되기 때문에 최적화된 지연 시간과 처리량을 보인다. 노드 간 통신에서는 네트워크 전송의 영향으로 지연 시간이 증가하고 처리량이 감소하는 경향이 있다.

6.3 네트워크 정책들의 우선순위 분석

정책 간 우선순위 확인을 위해 Nginx와 Busybox 이미지를 통해 임의로 생성한 파드 A와 B를 이용하여 2가지 시나리오를 모두 실험하였다. 첫 번째 시나리오(시나리오 1)에서는 쿠버네티스 기본 네트워크 정책을 사용하여 모든 통신을 차단하고, Cilium이나 Calico 정책으로 일부 통신만을 허용했다. 두 번째 시나리오(시나리오 2)에서는 쿠버네티스 정책으로 파드 A의 트래픽을 파드 B에 허용하게 설정하고, Cilium 또는 Calico 정책으로 이 트래픽을 차단했다.

6.3.1 쿠버네티스 기본 네트워크 정책과 Cilium 네트워크 정책의 비교

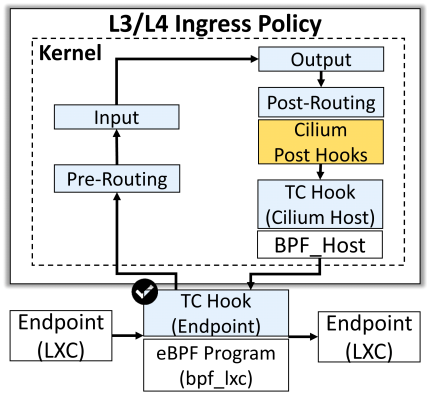
시나리오 2 실험에서 쿠버네티스의 기본 정책이 설정된 상태에서 Cilium의 정책만으로는 통신 허용이 불가능했지만, 쿠버네티스의 정책을 제거하면 통신이 가능하였다. 즉, 쿠버네티스 기본 네트워크 정책이 Cilium 정책보다 우선 적용된다. 이는 쿠버네티스와 Cilium의 정책 적용 시점의 차이로 인해 패킷의 라우팅 방식에 따라 정책의 우선순위가 달라지는 것을 의미한다. 쿠버네티스의 기본 정책이 노드 레벨에서 패킷 필터링 전에 먼저 적용되므로 패킷이 Cilium의 eBPF Hook에 도달하기 전에 차단될 수 있다. 즉, 이러한 현상은 쿠버네티스와 Cilium의 정책 적용 시점의 차이 때문에 발생한다고 보인다. 추가로, Cilium의 자체 네트워크 정책 간에도 스코프 차이가 있어 우선순위가 다르게 적용되는 것을 확인하였다. CiliumClusterwideNetworkPolicy가 CilumNetworkPolicy보다 더 넓은 범위로 우선 적용된다.

6.3.2 쿠버네티스 기본 네트워크 정책과 Calico 네트워크 정책의 비교

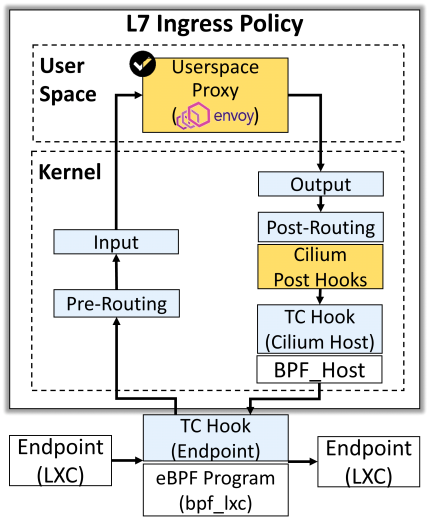
시나리오 1 실험에서 쿠버네티스 기본 정책과 GlobalNetworkPolicy를 함께 적용하면, 파드 간 모든 트래픽이 차단되었다. 쿠버네티스 정책을 제거하면 Calico 정책만 남으므로 파드 간 통신이 원활하게 이루어졌다. 이 결과로, 쿠버네티스의 기본 네트워크 정책이 Calico 정책보다도 우선 적용되는 것을 확인할 수 있다. 더 나아가, Calico의 자체 네트워크 정책인 CalicoNetworkPolicy와 클러스터 전체 범위의 GlobalNetworkPolicy 간에 우선순위 차이가 있으며, GlobalNetworkPolicy가 넓은 범위로 우선 적용된다.

6.4 eBPF와 레이어 7 네트워크 정책 적용의 성능 영향 분석

Fig 10는 Cilium에서의 레이어 3/4와 레이어 7의 패킷 처리 과정을 비교한다. eBPF는 커널 수준에서 트래픽을 빠르게 처리하는 장점이 있지만, Fig 8, 9에서 확인된 바와 같이, 레이어 7의 네트워크



(a) Traffic Flow when Applying L3/L4 Ingress Policy



(b) Traffic Flow when Applying L7 Ingress Policy

Fig. 10. Life of a Packet in Cilium: L3/L4 vs. L7 Ingress Policies[19]

정책 적용 시 성능 저하가 발생한다.

Fig 10(a)과 같이 레이어 3/4에서 네트워크 정책이 적용될 때는 엔드포인트에서 시작해 eBPF를 이용하여 커널 공간에서 패킷 처리가 진행되며, 사용자 공간으로의 이동 없이 효율적으로 처리된다. 레이어 7 정책이 적용될 때의 패킷 흐름은 Fig 10(b)처럼, 패킷이 eBPF를 지나 사용자 공간의 User Space proxy로 이동된다. 이곳에서 HTTP, DNS와 같은 레이어 7 정책 기준으로 패킷이 처리된 후 커널로 반환되어 처리가 완료된다.

패킷 처리가 커널에 있는 eBPF 내에서 수행할 수 없으므로 User Space로 가져가서 처리해야 하는데, 이동하는 과정에서 오버헤드와 지연 시간이 발생한다. 즉 레이어 7 정책 적용으로 패킷 처리가 IP나 포트 기반 필터링보다 복잡하므로 성능 저하가 발생한다. 따라서, eBPF의 높은 성능을 유지하려면 레이어 7의 복잡한 패킷 처리와 관련된 패킷 이동을 최소화해야 한다.

VII. 논 의

성능과 보안 간의 균형. 성능 저하 없이 보안 기능을 최대한 활용하는 것은 항상 도전적이다. 본 논문에서 레이어 3/4의 경우 eBPF의 빠른 패킷 처리를 보여준 반면 레이어 7은 페이로드 검증을 위한 복잡한 네트워크 정책 처리로 인해 성능 저하가 관찰되었다. 따라서, 레이어 7의 세밀한 트래픽 제어가 필요한 경우 해당 트래픽을 최소화하는 방향으로 네트워크 정책을 디자인하거나 하드웨어(예: Smart-NIC)를 활용하여 성능 저하를 줄일 수 있다.

정책 우선순위와 네트워크 관리의 복잡성. 다양한 CNI 솔루션들 간의 정책 적용 우선순위가 다르다는 점은 클러스터 관리자에게 중요한 고려사항이다. 이를 고려하여 여러 정책을 동시에 사용할 경우, 충돌이나 예기치 않은 행동이 발생하지 않도록 주의해야 한다. 그리고 쿠버네티스 기본 네트워크 정책과 각 CNI의 자체 네트워크 정책 간의 상호작용이 예상과 다를 수 있으므로, 이러한 특징을 충분히 이해하고 정책을 구성해야 한다.

다양한 워크로드와 환경에 따른 CNI 선택. CNI 솔루션의 선택은 워크로드의 특성, 네트워크 트래픽 패턴, 보안 요구 사항 등 다양한 요소를 고려하여 결정해야 한다. 예를 들어, 높은 네트워크 성능이 요구되는 워크로드의 경우 eBPF를 지원하는 Calico, Cilium 같은 CNI를 선택하는 것이 적합할 수 있다. 반면, 고급 보안 기능과 세밀한 트래픽 제어가 중요한 워크로드의 경우 Cilium과 같은 CNI를 고려할 수 있다.

VIII. 결 론

본 연구에서는 Cilium, Calico, Weavenet, Kube-router의 보안 기능과 네트워크 정책 처리 성능을 상세히 분석하였다. 컨테이너 환경에서 레이어 3/4와 레이어 7에서의 네트워크 정책을 적용했을 때의 성능 차이는 CNI 플러그인의 특성에 따라 좌우되었다. Cilium과 Calico에서 제공하는 네트워크 정책은 쿠버네티스 기본 정책과 잘 호환되며, 쿠버네티스 기본 정책이 CNI 정책보다 먼저 적용된다. 또한, 레이어 7 정책을 처리할 때 커널 단에서 해주던 필터링이 불가하기 때문에 사용자 공간으로 이동해서 처리하는 과정 때문에 딜레이가 발생하므로 성능 저하를 확인하였다. 따라서, 쿠버네티스 환경에서 정책을 효과적으로 적용을 위해서는 CNI 플러그인의 선택과 정책 구성에 주의가 필요하다는 결론을 얻었다.

향후 연구는 네트워크 정책의 자동 생성 및 검증 메커니즘 개발에 초점을 맞출 예정이며, 이를 통해 네트워크 정책 설정의 복잡성을 줄이고 효율적인 컨테이너 환경을 제공하고자 한다.

References

[1] Hyeon-Du Jin, Byung-Seo Kim, “Types and Comparative Analysis of Kubernetes CNI in Cloud Systems.”, Journal of the Institute of Electronics and Information Engineers, 56(8), pp. 21 - 27, Aug. 2019.

[2] Yuan-Mei Cui, Kyung-Woon Lee, Ji-You Lee, and Chuck Yoo, “An analysis on the network plugins of Kubernetes”, Proceedings of the Korean Society of Computer Information Conference, 48(1), pp. 250-252, Jun. 2021.

[3] Qi, S., Kulkarni, S. G., & Ramakrishnan, K. K. Assessing container network interface plugins:

Functionality, performance, and scalability. IEEE Transactions on Network and Service Management, vol. 18, pp. 656-671, Dec 2020.

〔4〕 Budigiri, G., Baumann, C., Mühlberg, J. T., Truyen, E., & Joosen, W.. Network policies in kubernetes: Performance evaluation and security analysis. In 2021 Joint European Conference on Networks and Communications & 6G Summit, pp. 407-412, Jun. 2021,

〔5〕 IT Daily, "〔Kubernetes①〕 Rise while resolving container management complexity", http://www.itdaily.kr/news/articleView.html?idxno=212049. accessed Jul. 2023,

〔6〕 ITWorld, "Tmax Cloud supplies Kubernetes-based container cloud platform to ShinhanBank", https://www.itworld.co.kr/news/279481., accessed Jul. 2023.

〔7〕 Tatum Hunter, "What Is Containerization in DevOps?",https://builtin.com/software-engineering-perspectives/containerization, accessed Jul. 2023

〔8〕 Aniruddh, M., Dinkar, A., Mouli, S. C., Sahana, B., & Deshpande, A. A. . Comparison of containerization and virtualization in cloud architectures. In 2021 IEEE International Conference on Electronics, Computing and Communication Technologies, pp. 1-5, Jul. 2021

〔9〕 Marathe, N., Gandhi, A., & Shah, J. M. Docker swarm and kubernetes in cloud computing environment. In 2019 3rd International Conference on Trends in Electronics and Informatics, pp. 179-184. Apr. 2019,

〔10〕 Frampton, M., & Frampton, M.. Apache mesos. Complete Guide to Open Source Big Data Stack, 97-137.

〔11〕 Kubernetes,T."Kubernetes." Kubernetes. accessed Jul. 2023,

〔12〕 "Cloud Native Computing Foundation" cncf docs., https://www.cncf.io/. accessed Jul. 2023,

〔13〕 "Cilium - Cloud Native, eBPF-based Networking, Observability, Security" Cilium Docs. https://cilium.io/ accessed Jul, 2023,

〔14〕 "Project Calico", Tigera project-calico, https://www.tigera.io/project-calico/. accessed Jul. 2023,

〔15〕 "Weave Net: Network Containers Across Environments" weave.works., https://www.weave.works/oss/net/. accessed Jul. 2023,

〔16〕 "Kube-router: Turnkey Kubernetes networking solution" kube-router. https://www.kube-router.io/. accessed Jul. 2023

〔17〕 "Network Policy" Kubernetes docs., https://kubernetes.io/docs/concepts/services-networking/network-policies/.accessed Jul. 2023

〔18〕 "Life of a Packet", Cilium Docs, https://docs.cilium.io/en/stable/network/ebpf/lifeofapacket/, accessed Jul . 2023,

〈저자소개〉

김 봄 (Bom Kim) 학생회원
2023년 8월: 인천대학교 컴퓨터공학부 학사
2023년 9월~현재: 인천대학교 일반대학원 컴퓨터공학과 석사과정
〈관심분야〉 클라우드 보안, 네트워크 보안

이 승 수 (Seungsoo Lee) 종신회원
2014년 2월: 숭실대학교 컴퓨터학부 학사
2016년 2월: KAIST 정보보호대학원 석사
2020년 8월: KAIST 정보보호대학원 박사
〈관심분야〉 클라우드 보안, 네트워크 보안