경량 가상머신의 네트워크 동작 원리 분석 및 성능 비교

고의진, 최원미, 유연호, 유혁*
*고려대학교

{ejko, ymcui, yhyoo}@os.korea.ac.kr, *chuckyoo@os.korea.ac.kr

A Study on the Light-Weighted Virtual Machine

Eujin Ko, Wonmi Choi, Yeonho Yoo, Chuck Yoo*
*Korea Univ.

요 약

컨테이너 가상화 기술은 용이한 배포와 효율적인 자원 활용으로 클라우드에서 애플리케이션을 구성하는데 많이 사용된다. 그러나 하나의 물리 커널을 공유하기 때문에 보안성이 취약하다. 이를 보완하기 위하여 가상머신과 같은 추가 계층을 도입하는 가상화 기술들이 소개되고 있다. 그러나 추가 계층으로 인해 네트워크와 같은 성능에 영향을 미칠 수 있다. 본 논문에서는 최근에 소개되고 있는 경량 가상화 기술들 중 Amazon 에서 제공하는 Firecracker 와 Xen, Linux Foundation 프로젝트에서 개발중인 Unikraft 에 대해 네트워크 구조 및 성능 차이를 비교한다.

Ⅰ. 서 론

컨테이너 가상화 기술은 일반적인 운영체제 상에서 여러 사용자를 위한 고립된 실행 환경을 제공한다. 애플리케이션의 쉬운 배포와 효율적인 자원 활용의 장점으로 인해 컨테이너 가상화는 클라우드 환경에서 애플리케이션을 구축하는데 많이 사용된다.[1] 그러나 컨테이너는 하나의 운영체제를 공유하기 때문에 성능 및 보안 측면에서 성능 간섭과 보안 위협의 문제가 있다.[2] 반면 독립적인 운영체제를 가진 서버 가상화 기술은 높은 보안성을 가지고 있지만 가상머신의 추가 계층으로 인해 컨테이너 대비 네트워크 성능 저하의 문제가 발생한다. 이는 차량용 시스템과 같은 네트워크 성능이 중요한 애플리케이션을 운영하는데 치명적이다. 따라서 최근에는 보안과 네트워크 성능을 동시에 보장하는 경량 가상머신 기술이 개발되고 있다. 본 논문에서는 최근에 사용되고 있는 경량 가상머신들을 네트워크 측면에서 분석한다. 특히, 사용자 레벨에서 동작하며 최소한의 장치 드라이버만 제공하는 Firecracker 와 단일 애플리케이션을 실행하기 위한 커널을 빌드 하는 Unikraft 의 네트워크 구조를 분석하고 성능 차이를 비교한다.

Ⅱ. 본론

본 장에서는 Firecracker 와 Unikraft 의 구조를 소개하고 네트워크 동작 방식을 비교한다. 또한 구조의 차이로 인해 발생하는 성능 차이를 분석하기 위해 패킷의 크기에 따라 네트워크 처리량 및 자원 사용량을 비교한다.

Ⅱ.Ⅰ. 경량 가상머신

Firecracker 는 Amazon 의 serverless computing service 인 AWS Lambda 에 사용되는 가상화 플랫폼이다.[3] Firecracker 는 애플리케이션이 사용하지 않는 장치 드라이버 (e.g., USB driver)를 제거한 경량화 된 커널 이미지 (microVM)와 이를 실행하는 Virtual Machine Monitor (VMM)를 제공한다. Firecracker VMM 은 사용자 레벨에서 동작하고 REST API 를 통해 microVM 을 제어한다. microVM 에서 발생한 I/O 요청은 virtio 드라이버를 통해 virtqueue 공유 메모리에 저장한다. 이후 Firecracker VMM 이 공유 메모리에 저장한 요청을 물리 커널의 네트워크에 전송한다.

Unikraft 는 Xen 과 Linux Foundation 프로젝트에서 개발 중인 경량 커널 플랫폼으로서 단일 주소 공간을 가지는 라이브러리 운영체제의 일종인 Unikernel 을 기반으로 한다.[4] Unikernel 은 특정 애플리케이션을 구동하기 위해 한정적인 모듈 및 라이브러리로 구성된 운영체제를 제공한다. Unikraft 는 애플리케이션의 네트워크 통신을 제공하기 위해 lwip 라는 경량화 TCP/IP 스택 라이브러리를 사용한다. Firecracker 와 유사한 방식으로 가상머신에서 실행하는 애플리케이션이 보낸 I/O 요청은 virtio 드라이브를 통해 virtqueue 에 저장한다. 단, virtqueue 에 저장한 요청은 QEMU emulator 를 통해 처리된다.

Ⅱ.Ⅱ. 성능 분석

Ⅱ.Ⅱ.Ⅰ. 실험 환경

본 실험은 10Gb 이더넷으로 연결된 두 개의 X86_64 기반의 Intel(R) Xeon(R) CPU E5-2650 v3 2.30GHz CPU 를 사용한다. 호스트 서버는 우분투 20.04 그리고 클라이언트 서버는 우분투 18.04 버전을 사용하며 두 서버 모두 리눅스 커널 5.4 버전을 사용한다. Firecracker 는 1.0.0 버전 그리고 Unikraft 는 Tethys 버전에 X86_64 아키텍처 기반에 KVM

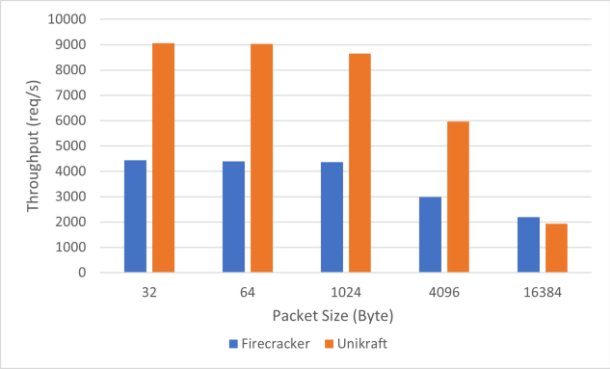

그림 1. Firecracker & Unikraft 네트워크 성능 비교

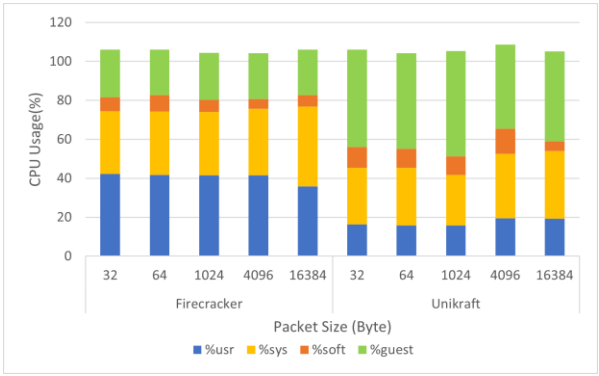

그림 2. Firecracker & Unikraft CPU 사용량 비교

하이퍼바이저를 적용하도록 빌드 한 바이너리 이미지를 사용한다. 네트워크 성능 측정을 위해 호스트 서버에서 Redis v5.0.6 데이터베이스[5] 서버를 실행하고, 클라이언트 서버에서 Redis-benchmark 툴을 사용하여 Redis 서버에 요청을 보낸다. 이때, 각각 32B, 64B, 1KB, 4KB, 그리고 16KB 크기의 SET 요청을 100,000 번씩 보내며 발생하는 네트워크 처리량을 측정하고 그에 발생하는 CPU 사용량을 mpstat 성능 모니터링 툴[6]을 사용하여 측정한다. 추가로, 동일한 양의 자원에서의 성능을 측정하기 위해 하나의 CPU core 만을 사용하도록 프로세스를 제한하였다.

Ⅱ.Ⅱ.Ⅱ. 실험 결과 및 성능 분석

그림 1 은 32B 부터 16KB 까지 크기의 패킷을 가상머신이 초당 처리하는 요청을 나타낸다. 32B 부터 4KB 크기의 패킷을 처리하는 경우 Unikraft 가 Firecracker 보다 뛰어난 네트워크 처리량을 보여준다. 특히, 1KB 패킷의 Unikraft 의 네트워크 처리량이 Firecracker 에 비해 1.98 배 크다. 그러나 패킷이 최대 전송 단위(MTU) (1500B) 크기보다 클 경우 Unikraft 와 Firecracker 의 네트워크 성능 차이는 0.88 배까지 줄어든다. 심지어, 16KB 의 패킷을 전송 시 Firecracker 가 Unikraft 의 처리량보다 12% 높은 것으로 나타난다. 이는 Unikraft 에서 가상 이미지를 빌드 할 때 사용한 lwip 라이브러리가 TCP 세분화 오프로드 (TSO) 기능을 제공하지 않기 때문에 큰 패킷을 전송할 때 TCP/IP layer 에서 MTU 사이즈에 맞게 세분화 후 전송해야 하기 때문이다. 따라서, 외부로 패킷을 전송하기 전 하드웨어의 NIC 에서 패킷을 세분화할 수 있는 Firecracker 보다 처리할 수 있는 요청의 개수가 줄어드는 결과로 이어진다.

다음으로 구체적으로 성능 차이가 발생하는 원인을 분석하기 위해 CPU 사용량을 추가 분석한다. 그림 2 는 요청을 처리할 때 각각 user, system, softirq 및 guest 로 세분화한 CPU 사용량을 보여준다. 그중 Firecracker 의 user level CPU 자원 사용량이 Unikraft 에 비해 23% 높은 반면 guest level CPU 자원은 25% 낮은 것을 확인할 수 있다. 이는 가상머신을 실행하는 Firecracker VMM 이 Unikraft 가 사용하는 QEMU 에 비해 높은 오버헤드가 발생하고, 이로 인해 Firecracker microVM 에서 I/O 요청을 처리하는 데 사용할 수 있는 CPU 자원이 Unikraft 가상머신에 비해 적기 때문에 그림 1 에서의 네트워크 처리량 감소를 초래한 것을 알 수 있다.

Ⅲ. 결론

본 논문에서는 경량화 가상머신인 Firecracker 와 Unikraft 의 패킷 사이즈별 네트워크 성능에 대해 비교 및 분석한다. 같은 자원이 주어졌을 때, Unikraft 는 Firecracker 에 비해 훨씬 높은 성능을 보여주는데 그 이유는 QEMU 가 Firecracker VMM 보다 낮은 CPU 오버헤드를 발생시켜 네트워크 I/O 를 처리하는데 더 많은 CPU 자원을 사용할 수 있기 때문이다. 그러나 큰 사이즈의 요청을 처리하는 경우 TSO 기능을 지원하는 Firecracker 의 네트워크 성능이 더 좋은 것을 확인할 수 있었다. 따라서 향후 연구로 Firecracker 와 Unikraft 의 장점을 모두 갖춘 새로운 경량 가상화 기술을 개발할 계획이다.

ACKNOWLEDGMENT

이 논문은 2022 년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(No. NRF-2021R1A6A1A13044830)과 2022 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No. 2015-0-00280, (SW 스타랩) 성능 및 보안 SLA 보장이 가능한 차세대 클라우드 인프라 SW 개발)을 받아 수행된 연구임.

참 고 문 헌

[1] Hussein, Mohamed K, et al. "A placement architecture for a container as a service (CaaS) in a cloud environment," Journal of Cloud Computing, 8.1: pp. 1-15. 2019.

[2] Sierra-Arriaga, Federico, et al. "Security issues and challenges for virtualization technologies," CSUR, 53.2: pp. 1-37. 2020.

[3] Agache, Alexandru, et al. "Firecracker: Lightweight virtualization for serverless applications," NSDI '20, pp. 419-434. 2020.

[4] Kuenzer, Simon, et al. "Unikraft: fast, specialized unikernels the easy way," EuroSys '21. pp. 376-394. 2021.

[5] "Redis," (http//www.redis.io/).

[6] "Mpstat," (https://linux.die.net/man/1/mpstat).