Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.31, NO.5, Oct. 2021 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2021.31.5.901

모바일 엣지 컴퓨팅 환경에서 안전 복사를 활용한 도커 컨테이너 마이그레이션 성능 분석*

변 원 준,[1†] 임 한 울,[1] 윤 주 범[2‡]
[1,2]세종대학교 (대학원생, 교수)

Performance Analysis of Docker Container Migration Using Secure Copy in Mobile Edge Computing*

Wonjun Byeon,[1†] Han-wool Lim,[1] Joobeom Yun[2‡]
[1,2]Sejong University (Graduate student, Professor)

요 약

모바일 기기는 그 자체가 가지고 있는 연산 자원이 제한적이기 때문에 클라우드를 활용하여 컴퓨팅하거나 데이터를 저장하는 경향이 있다. 5G로 인해 실시간성이 중요해 짐에 따라, 중앙 클라우드보다 사용자에게 더 가까운 위치에서 컴퓨팅하는 엣지 클라우드에 관한 많은 연구가 수행되었다. 사용자가 현재 연결된 기지국의 엣지 클라우드와 물리적인 거리가 멀어질수록 네트워크 전송 속도가 느려지게 된다. 따라서 원활한 서비스 이용을 위해서는 가까운 엣지 클라우드로 애플리케이션을 마이그레이션 한 뒤 재실행해야 한다. 우리는 호스트 운영 체제와 독립적이며, 가상 머신에 비해 이미지 크기가 상대적으로 가벼운 도커 컨테이너에서 애플리케이션을 실행한다. 기존의 마이그레이션 연구는 네트워크 시뮬레이터를 사용하여 실험하였다. 시뮬레이터는 고정된 값을 사용하기 때문에 실제 환경에서의 결괏값과는 차이점이 발생한다. 또한, 공유 저장소를 통해 이미지를 마이그레이션 하는 방식을 사용하였는데, 이는 패킷 내용 노출에 대한 위험을 갖는다. 본 논문에서는 실제 환경에서 엣지 컴퓨팅 환경을 구현하여 데이터 암호화 전송방식인 안전 복사(Secure CoPy) 방식으로 컨테이너를 마이그레이션 한다. 공유 저장소 방식 중 하나인 네트워크 파일 시스템(Network File System)과 마이그레이션 시간을 비교하고 안전성 확인을 위해 네트워크 패킷을 분석한다.

ABSTRACT

Since mobile devices have limited computational resources, it tends to use the cloud to compute or store data. As real-time becomes more important due to 5G, many studies have been conducted on edge clouds that computes at locations closer to users than central clouds. The farther the user's physical distance from the edge cloud connected to base station is, the slower the network transmits. So applications should be migrated and re-run to nearby edge cloud for smooth service use. We run applications in docker containers, which is independent of the host operating system and has a relatively light images size compared to the virtual machine. Existing migration studies have been experimented by using network simulators. It uses fixed values, so it is different from the results in the real-world environment. In addition, the method of migrating images through shared storage was used, which poses a risk of packet content exposure. In this paper, Containers

Received(07. 26. 2021), Modified(1st: 09. 10. 2021, 2nd: 10. 05. 2021), Accepted(10. 06. 2021)
* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW컴퓨팅산업원천기술개발사업의 연구결과로 수행되었음 (2020-0-00325)
† 주저자, wonjun100@naver.com
‡ 교신저자, jbyun@sejong.ac.kr(Corresponding author)

are migrated with Secure CoPy(SCP) method, a data encryption transmission, by establishing an edge computing environment in a real-world environment. It compares migration time with Network File System, one of the shared storage methods, and analyzes network packets to verify safety.

Keywords: Edge Computing, Migration, Docker Container, SCP, NFS

I. 서 론

Internet of Things(IoT) 기술의 발달로 인해 여러 센서에 의해서 상당한 양의 데이터가 발생하고 있다[8]. 이러한 센서를 이용해 자율주행 자동차의 경우 Machine To Machine(M2M) 통신을 할 수 있으며, 주차선을 인식하여 자동 주차도 가능하게 되었다. 하지만 더 나아가서 신호등 정보를 분석하여 주행 및 정지하는 주행 방법의 경우 한정된 자원 내에서 빠른 시간 안에 정보를 분석하고 저장하기에는 계산력과 저장공간이 부족할 수 있다. 이를 해결하기 위해 클라우드 서버에서 서비스 공급자가 애플리케이션을 호스팅 하고, 고객이 사용하는 방식인 Software as a Service(SaaS)를 이용할 수 있다[1]. 무선 기기의 센서에서 나온 데이터를 클라우드 서버로 전송하고, 클라우드 서버는 전송받은 데이터를 애플리케이션 내부에서 처리하여 결과를 다시 무선 기기로 전달한다. 하지만 클라우드 서버와 무선 기기에서 사용하는 네트워크 간의 물리적 거리가 멀어지게 되면 지연시간이 발생하며, 이는 원활한 서비스 이용에 지장을 준다. 이에 따라 엣지 클라우드의 필요성이 나날이 높아지고 있다.

엣지 클라우드는 네트워크의 가장자리에서 사용자와 가까운 위치에 있는 클라우드를 의미하고, 이러한 엣지 클라우드를 이용하여 상대적으로 연산 자원이 부족한 모바일 기기에 서비스를 제공하는 형태를 모바일 엣지 컴퓨팅이라고 한다[5]. 사용자가 정지 상태에서 엣지 컴퓨팅을 이용한다면 일정한 네트워크 속도를 유지하며 애플리케이션을 이용할 수 있지만, 이동성을 갖는 모바일 기기의 경우 기지국에서 멀어짐에 따라 네트워크 속도가 느려지게 되어 서비스 지연시간이 발생하여, 엣지 컴퓨팅을 사용하는 의미가 퇴색된다. 따라서 사용자의 이동에 따라 이전 엣지 클라우드보다 가까이에 있는 다른 엣지 클라우드에서 데이터 처리가 가능해야 하는데, 이를 위해선 이전 엣지 클라우드로부터 목적지 엣지 클라우드까지 사용중이던 애플리케이션을 마이그레이션 하는 과정이 필요하다. 엣지 컴퓨팅의 장점 중 하나인 실시간성을 위해 본 논문에서는 신속한 애플리케이션 사용을 위한 마이그레이션 속도에 중점을 두고 연구하였다.

기존의 많은 연구는 네트워크 시뮬레이터 3(Network Simulator-3,NS3)등을 이용하여 엣지 클라우드 환경을 가상으로 구축해 실험하였으나, 이는 실제 환경에서 발생하는 다양한 변수에 대한 고려가 되어있지 않기 때문에 실제 환경의 결괏값과 차이가 발생한다. 또한, 각 엣지 클라우드 내의 가상 머신 또는 컨테이너가 공유 저장소의 데이터를 참조하여 실행하는 방식을 이용했다. 이러한 방식은 동일한 네트워크의 안전성이 보장된 환경에서 사용하기에는 편리하지만 신뢰할 수 없는 외부 네트워크 환경에서 사용하기에는 패킷 내용 노출에 대한 위험이 있다.

본 논문에서는 시뮬레이터가 아닌 실제 환경에서 물리적으로 분리된 호스트와 각각 다른 공인 IP에 연결된 공유기의 Wi-Fi를 이용한다. 또한, 기존의 공유 저장소를 이용하여 마이그레이션 하는 방식 중 하나인 네트워크 파일 시스템(Network File System, NFS)과 본 논문에서 제안하는 방법인 시큐어 쉘(Secure SHell, SSH) 프로토콜 기반의 안전 복사(Secure Copy, SCP)를 이용하여 도커 컨테이너의 마이그레이션 시간을 비교하고 와이어샤크를 이용해 네트워크 패킷을 관찰하여 보안수준을 확인한다.

II. 관련 연구

2.1 엣지 클라우드

엣지 클라우드는 사용자에 가까이 위치하여 더 빠르고 안전한 컴퓨팅 능력과 저장공간을 지닌 네트워크의 가장자리에 있는 클라우드를 의미한다. 클라우드의 세 가지 주요 서비스를 제공하며 다음과 같다:

- Infrastructure as a Service(IaaS) 서버 또는 하드웨어 자원을 제공하는 서비스.
- Platform as a Service(PaaS) 소프트웨어 개발에 필요한 플랫폼을 제공하는 서비스.
- Software as a Service(SaaS) 제공하는 소프트웨어를 사용할 수 있는 서비스.

중앙 클라우드와 비교하면 계산력과 저장공간 등은 제한적이지만, 사용자와 가까운 거리에서 서비스를 제공하므로 중앙 클라우드와의 통신 시 발생하는 지연시간 등을 줄여주어 실시간성을 요구하는 작업에는 많은 도움이 된다.

Fig. 1은 여러 기기의 센서로부터 생성된 데이터를 가까운 엣지 클라우드로 전송하여 애플리케이션 내에서 데이터를 분석하거나 계산함을 보여준다. 필요에 따라 분석결과를 다시 기기에 알려주거나 저장이 필요한 데이터를 엣지 클라우드 내부에 저장할 수 있다. 예를 들어 엣지 클라우드에서 동작하고 있는 '카메라를 이용한 머신러닝 기반 얼굴 분석 프로그램(Openface)'의 경우 사용자 기기의 카메라에서 수집한 이미지를 엣지 클라우드 내에서 학습시킨 후에 다시 기기로 분석결과를 전달한다[2]. 저장공간 및 계산자원이 부족할 경우 중앙 클라우드에 전송하여 데이터를 저장할 수도 있으며, 추가적인 계산을 하도록 할 수 있다.

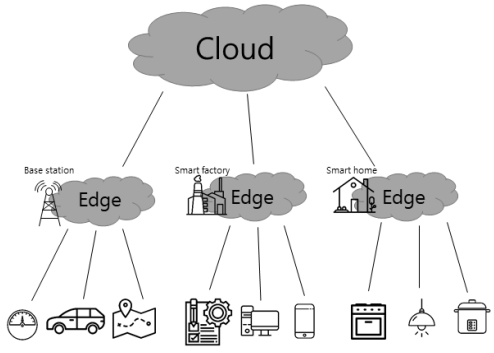


Fig. 1. Edge Cloud Architecture

2.2 도커 컨테이너

도커는 OpenVZ, cri-o, 리눅스 컨테이너(LXC)와 같은 컨테이너 가상화 기술을 지원하는 다양한 플랫폼 중 하나이다. Fig. 2는 도커 컨테이너의 생명주기를 보여준다. 먼저, 저장소에서 PULL 명령어로 이미지를 로컬로 가져온다. 그 후 CREATE 명령어로 컨테이너를 생성하고 START 명령어로 컨테이너를 메모리에 올려 사용할 수 있도록 한다. 반대로 실행 중인 컨테이너에 STOP 명령어를 사용하면 실행을 멈추고 대기 상태로 존재한다. 대기 상태인 컨테이너는 RM 명령어로 삭제할 수 있고 COMMIT 명령어로 다시 이미지 형태로 만들 수 있다. 저장된 이미지는 RMI 명령어로 삭제할 수 있고 PUSH 명령어로 저장소에 저장할 수 있다. 이미지 상태에서 RUN 명령어를 사용하면 컨테이너를 생성함과 동시에 메모리에 올려 실행시킨다. 현재 실행 중인 컨테이너를 삭제하기 위해선 먼저 STOP 명령어로 정지하여야 한다. 만약 실행 중이던 컨테이너를 정지와 동시에 삭제하고 싶으면 RUN 명령어를 입력할 때 -rm 명령어를 함께 입력함으로써 가능하게 한다.

도커는 읽기 전용 이미지로부터 컨테이너를 생성하여 애플리케이션을 실행한다. 컨테이너 내부에 운영체제, 웹 서버, 데이터베이스를 모두 설치하여 사용할 수도 있고 각각 다른 컨테이너에 설치하여 연동할 수도 있다. 컨테이너가 변경되면 컨테이너의 변경사항만 레이어 형태로 스택에 쌓여 저장된다. 따라서 컨테이너가 삭제되면 내부에 저장된 데이터베이스 내용과 파일들이 사라지게 된다. 이를 방지하기 위해 로컬 내부의 폴더와 컨테이너 내부의 폴더를 공유할 수 있도록 마운트 하는 방법인 볼륨을 활용한다.

컨테이너 가상화 기술 이외에 가상 머신(Virtual Machines, VMs)을 이용하는 방법이 있다. Fig. 3에서 도커는 도커 엔진 위에서 컨테이너를 생성한다. 생성된 컨테이너는 독립된 파일시스템을 가지며, 게스트 운영체제를 요구하지 않는다. 가상 머신은 컨테이너 기술과 비교하면 웹 서비스의 응답 시간 측면에서 125%의 성능 향상을 보인다[3]. 하지만 가상 머신의 경우 하이퍼바이저 위에 또 다른 게스트 운영체제가 필요하므로 컨테이너 가상화 기술에 비해 무거운 접이 있다.

본 논문은 사용자 이동에 따라 애플리케이션을 마이그레이션 하여 가까운 서버에서 컴퓨팅하는 것이 목적이다. 따라서 새로운 엣지 클라우드에 도착할 때

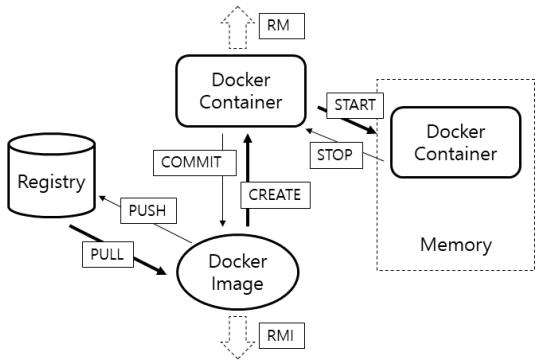


Fig. 2. Docker Container Life Cycle

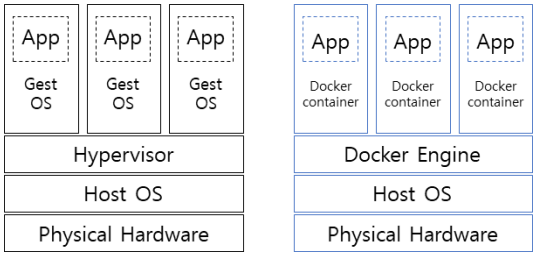


Fig. 3. Compare VM to Docker container

마다 기존에 애플리케이션을 구동하였던 운영체제를 매번 설치하는 것은 비효율적이다. 그러므로, 다양한 운영체제 위에서 구동이 가능한 도커 컨테이너를 마이그레이션하는 것이 더욱 도움 될 것으로 예상한다.

2.3 라이브 마이그레이션

모바일 기기가 현재 연결되어있는 엣지 클라우드로부터 서비스를 받는 도중에 다른 지역으로 이동하게 되면, 엣지 클라우드의 중앙 기지국에서 멀어짐에 따라 통신이 원활히 이루어지지 못한다. 이는 송신과 수신 안테나 사이의 거리에 따른 전력 감소를 나타내는 자유 공간 경로 손실이 원인 중 하나가 될 수 있다[4]. 따라서 현재 연결되어있는 엣지 클라우드와 거리가 멀어진다면 최적의 통신속도를 보장하며 원활한 서비스를 제공해줄 수 있는 엣지 클라우드로 데이터를 마이그레이션 해주어야 한다.

마이그레이션은 크게 사전 복사, 사후 복사 두 가지로 나뉘게 된다[5]. 사전 복사는 먼저 가상 머신의 모든 메모리 페이지를 복사하여 목적지 엣지 클라우드로 마이그레이션 한다. 목적지 엣지 클라우드로 데이터 전송이 완료됐다면 해당 클라우드에서 애플리케이션을 실행한다. 그 후 사용자는 새로운 엣지 클라우드에서 서비스를 이용하기 위해 마이그레이션된 애플리케이션을 이용한다. 이때 이전 엣지 클라우드에서 애플리케이션 실행을 위해 먼저 전송된 메모리 페이지를 제외한, 새롭게 생성된 메모리 페이지를 모두 복사하여 다시 한번 전송하는 방법으로 마이그레이션 한다. 만약 복사 도중에 데이터의 변경이 생기면 재복사하여 새롭게 복사된 부분 만큼 다시 전송한다. 이러한 방법은 서비스 중단 시간을 최소화한다. 사후 복사는 가상 머신의 프로세스가 멈춰있는 상태에서 목적지 엣지 클라우드로 CPU 상태 정보, 레지스터 정보 등을 전송한다. 사후 복사의 경우, 메모리 페이지를 한 번만 복사하여 전송하기 때문에 아직 복사되지 않은 정보를 목적지 엣지 클라우드에서 이용하려는 경우 오류가 발생할 수 있다.

본 논문에서는 도커 컨테이너를 이미지화한 뒤 Tar 압축파일 형태로 전송한다. 따라서 다른 액세스 포인트 즉, 목적지 엣지 클라우드를 알고 있는 경우에 먼저 압축파일을 전송한 뒤 볼륨 폴더 안의 파일들은 나중에 전송함으로써 사전 복사한다고 정의한다. 사후 복사는 유저 기기가 목적지 엣지 클라우드에 도착하고 난 뒤 이전 엣지 클라우드로부터 압축파일 및 볼륨 폴더 안의 파일들을 한 번에 전송받는 것으로 정의한다.

III. 컨테이너 마이그레이션 기술

3.1 체크포인트/복원 기술

체크포인트/복원(Checkpoint/Restore In Userspace, CRIU)기술은 동작 중인 컨테이너를 일시 정지한 뒤 현재 상태에 대한 정보를 저장한다. 저장한 파일을 목적지 엣지 클라우드로 마이그레이션하여 새로운 컨테이너에서 복원한다. 상태 파일은 컨테이너 자체보다 용량이 가벼우며, 복원 후 일시 정지된 상태를 이어서 서비스받을 수 있으므로 유용하게 이용될 수 있다. 하지만 목적지 엣지 클라우드에서 이전에 사용하던 것과 동일한 컨테이너가 존재하고 있어야 하므로, 어떤 엣지 클라우드로 전송될지 모르는 상황에서는 사용하기 어렵다. 또한, 도커 컨테이너는 체크포인트/복원 기술을 정식적으로 지원하고 있지 않으며, 실험모드로 설정해야만 체크포인트 상태 파일을 생성해 낼 수 있다. 하지만 최적화된 버전이 아니라면 상태 파일이 다른 컨테이너에서 정상적으로 복원되지 않는다.

3.2 네트워크 파일 시스템

네트워크 파일 시스템(Network File System, NFS)은 서버/클라이언트로 구성된 공유 저장소 기술 중 하나이다. 네트워크 파일 시스템 서버에 마운트한 클라이언트는 자신의 파일을 업로드할 수 있고, 다른 사용자가 공유한 파일을 다운로드 하거나 실행, 수정, 삭제할 수 있다. 공유 폴더 안의 데이터는 삭제하기 전까지 저장되어 있다. 이는 사용자의 이동 경로가 불분명하여 목적지 엣지 클라우드를 예측할

수 없는 경우에도 도커 컨테이너 이미지가 이미 복사된 덕분에 마운트 되어있는 어느 서버에서도 빠르게 실행할 수 있다. 새로운 액세스 포인트에 도착한 뒤 해당 엣지 클라우드에서 도커 컨테이너 이미지를 로드 후 실행시키면 되므로, 사전 복사 방식에 유리하게 작용할 수 있다.

공유 폴더를 이용하는 네트워크 파일 시스템은 임의의 클라이언트에서 악의적인 접근을 제한하기 위해 서버는 신뢰할 수 있는 클라이언트의 IP를 설정할 수 있다. 하지만 하나의 공인 IP를 신뢰할 수 있다고 설정하였을 때, 네트워크 주소 변환(Network Address Translation, NAT)에 의해 생성된 사설 IP를 모두 제한하기는 어려우므로 서로 다른 액세스 포인트를 갖는 모바일 엣지 컴퓨팅 내에서 사용하기에는 아직 보안 적으로 해결해야 할 문제가 많다.

3.3 안전 복사

안전 복사(Secure Copy, SCP)는 텔넷, rlogin와 같은 프로토콜을 대체하기 위한 시큐어 셸(Secure SHell, SSH) 프로토콜 기반 원격 암호화 파일 전송 방법이다. 시큐어 셸 프로토콜은 공개키 알고리즘 기반 암호화 방식을 사용한다. 각 서버에는 호스트 키가 있어야 하며, 데이터를 전송하려는 원격 서버의 공개 호스트 키를 사전에 알고 있어야 한다[6]. 키 교환은 최초 한 번만 이루어지고 데이터 전송 전에 한 번의 확인 후에 해시값으로 암호화가 이루어진다.

먼저 로컬 서버와 원격 서버의 키 교환이 이루어지고 나면 로컬 서버에서 아래와 같은 입력 방식으로 파일을 전송할 수 있게 된다.

#:scp[옵션][파일명][서버 내 사용자 ID][서버 IP]:[파일 전송 경로]

하지만 위 방법으로 파일을 전송하면 매번 원격 서버 내 사용자 ID의 비밀번호를 직접 입력해야 하므로 본 논문에서는 비밀번호 자동 입력 방법인 sshpass툴을 사용한다. sshpass는 쉘 명령어를 작성할 때 미리 서버 내 사용자 ID의 암호를 입력하거나 암호가 적혀있는 파일을 입력한다. 이처럼 작성하면 비밀번호를 입력하는 절차를 요구받지 않고 데이터를 전송할 수 있다.

IV. 실험 방법

Fig. 4는 안전 복사를 이용한 마이그레이션 구조를 나타낸다. 각각 다른 공인 IP 회선으로 연결된 공유기를 액세스 포인트로 두고, 파일 전송에 필요한 포트를 포워딩한다. Wi-Fi 1에 무선으로 연결된 노트북과 Wi-Fi 2에 유선으로 연결된 데스크톱을 각각 엣지 클라우드 1과 엣지 클라우드 2로 가정하며, 전송 속도는 각각 59Mbps, 29Mbps을 갖는다. 사용자 기기는 엣지 클라우드 1을 통해서 웹 서비스를 이용하거나, Openface 애플리케이션을 사용할 수 있다. 사용자 기기가 이동함으로 인해 Wi-Fi 2로 연결됐다고 가정할 때 엣지 클라우드 1은 안전 복사를 사용하여 엣지 클라우드 2로 도커 컨테이너를 마이그레이션 한다. 애플리케이션 크기에 따른 마이그레이션 시간 비교를 위해 각각 1.24MB, 72.2MB, 667MB, 2.51GB의 컨테이너 이미지를 마이그레이션 한다.

먼저 실행 중인 컨테이너를 COMMIT 명령어로 이미지화한 뒤 SAVE 명령어를 사용하여 Tar 압축파일 형태로 저장한다. 압축파일을 안전 복사의 비밀번호 자동 입력 도구인 sshpass를 이용하여 엣지 클라우드 2로 전송한다. 전송이 완료되면 엣지 클라우드 1에 있는 도커 컨테이너를 삭제하고, 동시에 엣지 클라우드 2에서 압축파일을 LOAD 명령어로 이미지화한 뒤, RUN 명령어로 실행시킨다. 만약 클라우드 1에서 볼륨 폴더 안에 저장된 파일이 있으면, 클라우드 2에서 컨테이너를 실행하기 전에 안전 복사를 이용하여 볼륨 파일을 마이그레이션 한 뒤에 RUN 명령어로 컨테이너를 실행한다.

본 논문에서는 위 과정에서 COMMIT을 시작한

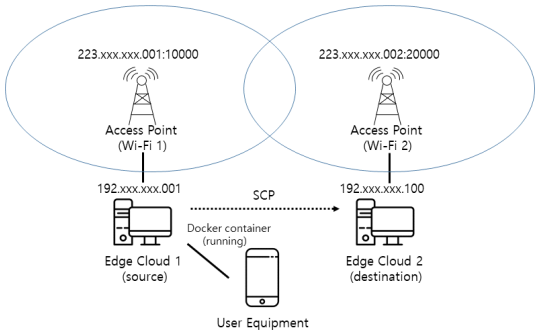


Fig. 4. Migration Using Secure Copy

시간부터 엣지 클라우드 2로 전송이 완료된 시간까지를 마이그레이션 시간으로 정의하고, 엣지 클라우드 2에서 LOAD를 시작한 시간부터 RUN 하여 실행이 된 상태까지를 실행 시간으로 정의하며, 마이그레이션 시간과 실행 시간을 합한 시간을 총 마이그레이션 시간이라고 정의한다. 또한, 사용자 기기에서 엣지 클라우드로 데이터를 오프로딩 하는 것은 고려하지 않으며, 사용자 이동으로 인해 두 엣지 클라우드 간에 마이그레이션 돼야 하는 상황으로 가정한다.

Fig. 5는 또 다른 공인 IP와 연결된 공유기에 Wi-Fi 3으로 연결한 노트북을 이용하여 네트워크 파일 시스템의 서버를 구축한다. 엣지 클라우드 1과 2는 Wi-Fi 3에 포트 포워딩 된 공인 IP를 통해 공유 저장소에 마운트한다. 마이그레이션 과정은 안전 복사와 같지만, 압축파일을 공유 폴더에 저장하는 방식에서 차이가 난다. 마이그레이션 속도뿐만 아니라, 데이터를 안전 복사 및 네트워크 파일 시스템 전송 시 안전성 검사를 위해 와이어샤크를 통한 패킷 내용을 확인한다.

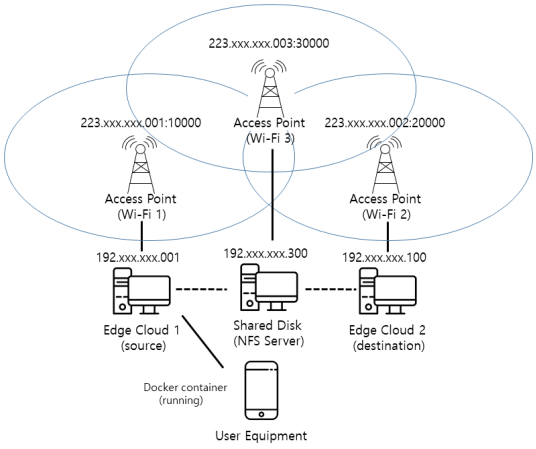


Fig. 5. Migration Using Network File System

V. 실험 결과

5.1 마이그레이션 속도 측정 비교

사용자가 현재 엣지 클라우드의 범위에서 벗어나면 사용자 기기와 엣지 클라우드간의 데이터 전송이 불안정하게 되므로, 가장 가까운 위치에 있는 엣지클라우드로 신속하게 애플리케이션을 마이그레이션 해주어야 한다. 따라서, 마이그레이션 성능은 마이그레이션 하는 시간과 목적지 엣지 클라우드에서 애플리케이션이 실행되는 시간이 적을수록 더 좋은 평가를 얻는다.

Fig. 6~9는 Table 1에서 종류와 이미지 크기가 다른 컨테이너에 대한 마이그레이션 시간을 비교한 그래프를 나타낸다. 그래프의 x축은 안전 복사와 네트워크 파일 시스템의 마이그레이션 시간, 실행 시간, 총 마이그레이션 시간을 나타내고 y축은 시간을 초 단위로 나타낸다.

안전 복사의 경우 컨테이너를 압축하고 전송하는데 대부분 시간이 소요되며, 목적지 엣지 클라우드에

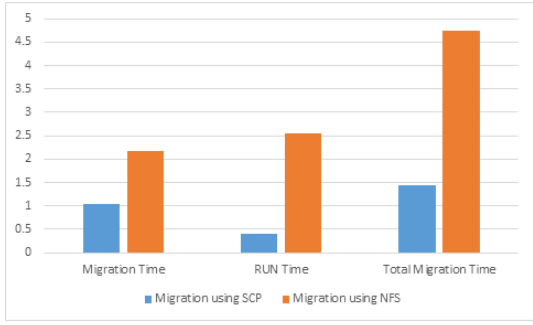


Fig. 6. Busybox(1.24MB) Migration

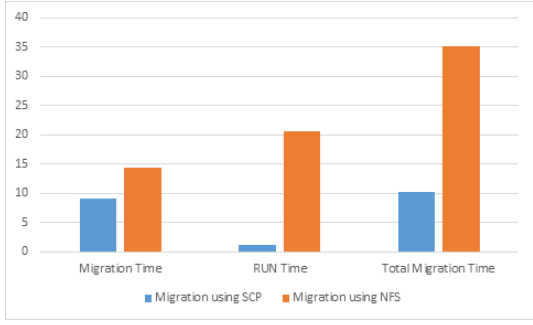


Fig. 7. Ubuntu(72.2MB) Migration

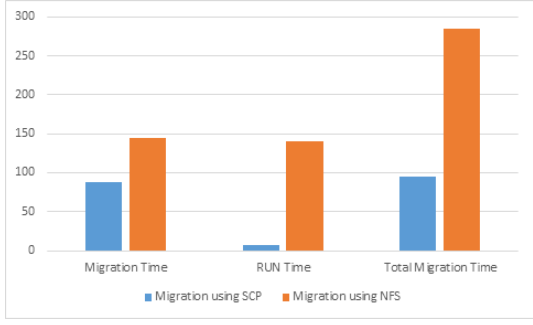


Fig. 8. Tomcat(667MB) Migration

Table 1. Comparison of NFS and SCP Migration Time

		Migration Time		Run Time		Total Migration Time	
Container	Image Size	NFS	SCP	NFS	SCP	NFS	SCP
Busybox	1.24MB	2.179s	1.041s	2.557s	0.394s	4.736s	1.435s
Ubuntu	72.7MB	14.402s	9.117s	20.657s	1.161s	35.059s	10.278s
Tomcat	667MB	144.209s	87.23s	140.416s	7.361s	284.625s	94.591s
Openface	2.51GB	491.411s	357.626s	579.097s	33.341s	1070.508s	390.967s

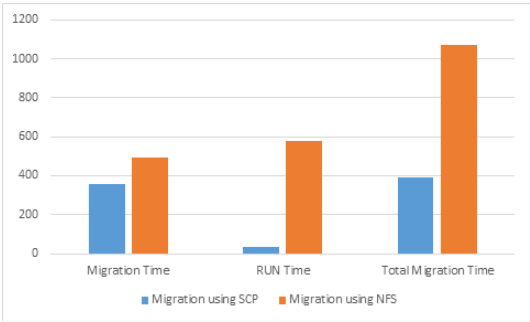


Fig. 9. Openface(2.51GB) Migration

서 압축파일을 실행하는 데에는 오히려 시간이 적게 걸려서 총 마이그레이션 시간 대부분이 이전 엣지 클라우드에서 발생한다. 네트워크 파일 시스템의 경우 컨테이너를 압축하고 공유 폴더에 저장하는 시간이 안전 복사에 비해 오래 걸리며 실행 시간은 마이그레이션 시간과 비슷하거나 오히려 더 오래 걸리는 경향을 보인다. [9]의 안전 복사(SCP), 네트워크 파일 시스템(NFS)의 파일 전송시간 측정 실험 결과에 따르면, 10GB의 단일 파일 하나를 전송할 때 로컬 네트워크, 와이드 네트워크 모두 네트워크 파일 전송방식이 안전 복사 방식보다 약 90초 정도 더 빨랐지만, 평균 13.7kB를 갖는 30000개 이상의 리눅스 소스 파일을 전송할 때에는 오히려 안전 복사 전송방식이 더 빠름을 볼 수 있다. 컨테이너는 레이어를 스택으로 쌓아 데이터를 추가하며, 각 레이어에는 파일 시스템이 포함되기 때문에 많은 폴더와 파일을 전송하게 된다. 컨테이너 내부 애플리케이션의 종류와 네트워크 환경에 따라 조금씩 다른 결과가 나타날 수 있겠지만 총 마이그레이션 시간은 네트워크 파일시스템 방식보다 본 논문에서 제안하는 안전 복사 방식이 더 적은 시간이 걸리는 것을 알 수 있다.

5.2 패킷 안전성 검사

Fig. 10은 실험을 위해 만든 테스트 파일을 와이어샤크를 이용하여 안전 복사와 네트워크 파일 시스템으로 전송되는 패킷을 확인한다. 왼쪽은 안전 복사의 패킷 일부를 나타내며, 모든 데이터가 해시 암호화되어 전송된다. 반면에 네트워크 파일 시스템 전송 방식은 전송되는 패킷에 testfile.txt 파일 이름이 그대로 드러나며 심지어 ID와 PW를 적어놓은 데이터 내용마저 확인된다. 모바일 엣지 컴퓨팅 환경은 외부 네트워크로 컨테이너를 마이그레이션하기 때문에 암호화되지 않은 전송방식은 사용하기에 많은 위험이 있다.

5.3 안전 복사의 CPU 사용률

Fig. 11은 데이터 전송 시 발생하는 시큐어 쉘(SSH), 안전 복사(SCP)의 CPU 사용률과 도커

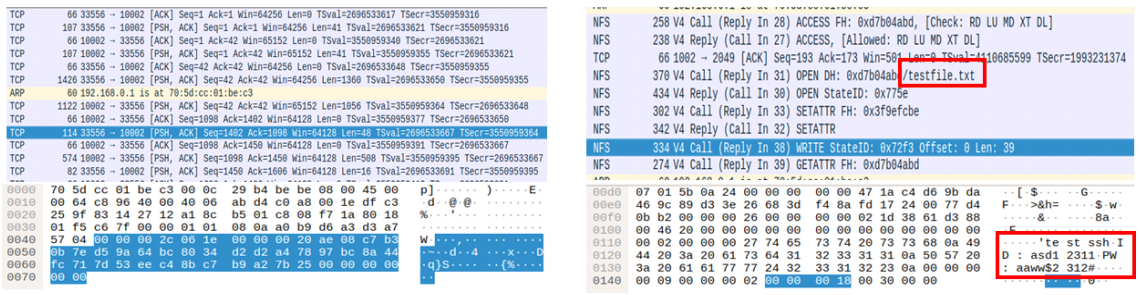


Fig. 10. Check the packet with WireShark. (Left - SCP, Right - NFS)

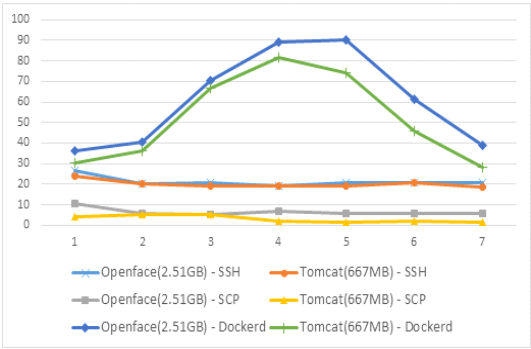


Fig. 11. CPU usage rate according to image size.

명령어를 통해 도커 허브(Docker Hub)에서 이미지를 불러올 때 사용되는 도커 데몬(Dockerd)의 CPU 사용률을 나타낸다. 용량이 작은 이미지 전송은 CPU 사용률이 너무 낮으므로 각각 2.51GB, 667MB의 용량을 갖는 이미지로 실험을 하였다.

y축은 CPU 사용률(%)을 나타내며, x축은 시간을 초 단위로 나타낸다. 실험은 7초 동안의 변화량을 기록하였으며, 7초 이후는 전과 거의 동일하다. 데이터 전송 방법의 경우, 두 이미지 모두 초기 SSH 연결 시에만 26.8%, 23.8%로 높은 CPU 사용률을 보이며, 이후 20.9%, 18.9%로 안정화 된다. SCP의 경우, 초기 암호화를 진행하는 부분에서만 CPU 사용률이 10.6%, 5.3%이며, 이후에는 6%, 2%로 안정화 된다. 도커 허브에서 이미지를 불러올 경우, SCP 데이터 전송 방식보다 약 10% 정도 높은 CPU 사용률을 보이며, 가져온 이미지의 압축을 해제하는 부분에서는 최대 90.1%, 81.8%의 높은 CPU 사용률을 갖는다.

VI. 결 론

본 논문은 사용자가 더 가까운 클라우드 서버에서 컴퓨팅 자원을 받을 수 있도록 하는 엣지 컴퓨팅환경을 실제 환경에서 구성하였다. 구성된 환경에서 엣지 클라우드의 주요 도전과제 중 하나인 마이그레이션시 발생하는 서비스 지연시간을 줄이기 위해 가상 머신이 아닌 상대적으로 가벼운 도커 컨테이너를 마이그레이션 하였다. 엣지 컴퓨팅의 가장 큰 장점인 실시간성을 강조하기 위해 마이그레이션 속도를 중점으로 연구를 진행하였으며, 안전 복사를 통한 도커 컨테이너 마이그레이션을 제시하였고 기존의 공유 저장소 방식과 비교하여 성능을 분석하였다.

안전 복사 방식은 공유 저장소를 이용하는 방식에 비해 마이그레이션 시간은 더 적게 소요됐으며, 목적지 엣지 클라우드에서 압축파일을 로드하고 실행하는 시간은 월등히 빠른 실행 속도를 보여주었다. 또한, 데이터를 암호화하기 때문에 더 안전하게 보낼 수 있음을 확인하였다. 공유 저장소 방식의 경우 로드하기 위한 도커 명령어를 입력할 때, 먼저 외부 저장소에 접근해야 하므로 이때 발생하는 대기시간에 의해 실행하기까지 시간이 길어지게 된다. 또한, 서버 쪽에 일시적으로 사용 불가능한 문제가 생기면 마운트한 클라이언트에서도 공유 폴더 내부의 파일을 사용할 수 없는 단점이 있다.

끝으로 마이그레이션 해야 하는 시기를 결정하는 연구로 확대함으로써 실제 환경에서 더욱 엣지 컴퓨팅에 가까운 환경을 구축할 수 있다고 생각한다. 이러한 연구는 T. Taleb 등[7]의 연구에서 제시하였던 마르코프 결정 과정 등을 통해서 예측 가능할 것으로 기대한다.

References

[1] S. Satyanarayana, "Cloud computing: SAAS," Computer Sciences and Telecommunications 2012 vol 4, no. 36, pp.76-79, Dec. 2011.

[2] T. Baltrušaitis, P. Robinson and L. Morency, "OpenFace: An open source facial behavior analysis toolkit," 2016 IEEE Winter Conference on Applications of Computer Vision, pp. 1-10, Mar. 2016.

[3] T. Salah, M. J. Zemerly, C. Y. Teun, M. Al-Qutayri and Y. Al-Hammadi, "Performance comparison between container-based and VM-based services", 2017 20th Conference on Innovations in Clouds, Internet and Networks, pp. 185-190, Mar. 2017.

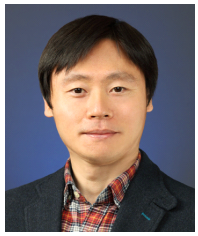
[4] L. Klozar and J. Prokopec, "Propagation path loss models for mobile communication", Proceedings of 21st International Conference Radioelektronika 2011, pp. 1-4, Apr. 2011.

[5] S. Wang, J. Xu, N. Zhang and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing", in IEEE Access, vol. 6, pp. 235511-23528, Apr. 2018.
[6] T. Ylonen, C. Lonvick, Ed. "The Secure Sehll (SSH) Protocol Architecture" Internet Engineering Task Force, RFC4251, Jan. 2006.
[7] T. Taleb and A. Ksentini, "Follow me cloud: interworking federated clouds and distributed mobile networks," in IEEE Network, vol. 27, no. 5, pp. 12-19, Oct. 2013.
[8] W. Yu et al. "A Survey on the Edge Computing for the Internet of Things," in IEEE Access, vol. 6, pp. 6900-6919, Nov. 2017.
[9] Hejtmánek, Lukáš, David Antoš, and Luboš Kopecký. "Choice of Data Transfer Protocols in Remote Storage Applications." CESNET Technical Report 1/2013, Jun. 2013.

〈저자소개〉

변 원 준 (Wonjun Byeon) 학생회원
2018년 2월: 선문대학교 컴퓨터공학과 졸업
2020년 9월~현재: 세종대학교 정보보호학과 석사과정
〈관심분야〉 정보보호, 엣지 컴퓨팅, IoT 보안

임 한 울 (Han-wool Lim) 학생회원
2020년 2월: 세종대학교 정보보호학과 졸업
2020년 3월~현재: 세종대학교 정보보호학과 석사과정
〈관심분야〉 정보보호, 엣지 컴퓨팅, IoT 보안



윤 주 범 (Joobeom Yun) 종신회원
1999년 2월: 고려대학교 컴퓨터학과 학사
2001년 2월: 서울대학교 컴퓨터공학과 석사
2012년 2월: KAIST 전산학과 박사
2001년 3월~2015년 2월: ETRI부설연구소 선임연구원
2015년 3월~현재: 세종대학교 정보보호학과 부교수
〈관심분야〉 네트워크 보안, 시스템 보안, 인공지능 보안