논문 21-46-10-19 The Journal of Korean Institute of Communications and Information Sciences '21-10 Vol.46 No.10
https://doi.org/10.7840/kics.2021.46.10.1709

REST-API 방식의 3자 전송을 이용한 컨테이너 기반의 클라우드 데이터 접근성 향상 기법

이 상 권◆, 석 우 진°, 문 정 훈*, 홍 원 택*, 김 기 현*, 김 천 용*, 구 영 훈*

Container Based Cloud Data Accessibility Improvement Method Using REST-API Based Third Party Transmission

Sang-kwon Lee◆, Woo-jin Seok°, Jung-hoon Moon*, Won-taek Hong*, Ki-hyeon Kim*, Chun-yong Kim*, Young-hoon Goo*

요 약

거대한 사이즈의 과학 데이터를 고속으로 전송하기 위해 빅데이터 전용의 고속도로 역할을 하는 네트워크 망을 구축해 사용하고 있다. 최근에는 데이터 전송 노드에서 전송 역할만을 하는 것을 넘어 노드의 여유 자원을 활용하기 위한 방법으로 컨테이너 기반의 클라우드를 구성한다. 컨테이너 방식은 필요한 만큼의 자원을 할당 받아 사용할 수 있어 자원의 가용성이 높고 애플리케이션 실행에 필요한 라이브러리 같은 파일들이 포함되어 있어 종속성 문제에서도 자유롭다는 장점이 있다. 컨테이너는 격리된 환경에서 실행되며 가상 네트워크를 사용하고 있어 외부와 통신을 위해서는 실제 노드의 아이피와 포트 번호를 매핑해주는 서비스를 별도로 생성해주어야 한다. 이러한 점은 지속적으로 서비스해야 하는 것이 아닌 단순히 데이터 전송을 하려는 사용자에게도 전송 프로그램이 사용하는 포트 번호의 정보를 알고 있어야 하는 등 사용자에게 어려운 점으로 다가온다. 이러한 점을 해결하기 위한 방법으로 REST-API 기반의 3자 전송을 제안한다. 3자 전송 서버는 외부와 통신이 가능하고 클라우드의 네트워크 정보도 알고 있어 양방향의 통신이 가능하다. 사용자는 별도의 프로그램을 설치 없이 REST 형식의 전송 명령어를 서버에 보내 사용할 수 있다. 본 논문에서는 포트 번호를 매핑 하는 서비스를 생성해 전송하는 방법과 REST-API를 이용한 3자 전송 방법을 비교하여 접근성을 분석한다.

Key Words : Third Party Transmission, Cloud, High Bandwidth Network, Data Transfer Node, REST-API

ABSTRACT

To transmit large-sized scientific data at high speed, network that serves as high way dedicated to bigdata has built and used. Recently, container based cloud is configured as method to utilize resources of node beyond just transmission role. Container has advantage of high availability of resources as it can be used by allocating as many resources as necessary and being free from dependency problems as it includes files such as libraries necessary for application execution. Since container runs in isolated environment and uses virtual network, service that maps IP and port number of node must be created separately for communication with

※ 본 연구는 한국과학기술정보연구원 '출연연 중심 AI 융합을 위한 빅데이터 Super Highway 융합클러스터(G-20-GT-CR05-S01)' 과제의 지원을 받아 수행되었음
◆ First Author : UST(University of Science and Technology), sglee@kisti.re.kr, 학생회원
° Corresponding Author : Korea Institute of Science and Technology Information(KISTI), wjseok@kisti.re.kr, 정회원
* 한국과학기술정보연구원
논문번호 : 202106-117-D-RN, Received May 31, 2021; Revised July 16, 2021; Accepted July 26, 2021

outside. These are difficult for users, such as having to know information of port number in which the transmission program is used even for user who simply wants to transmit data rather than continuous service. To solve this problem, we propose third-party transmission based on REST-API. Third-party transmission can communicate with the outside and knows network information of cloud, so it can communicate in both directions. Users can send REST-type transmission commands to server and use them without installing program. we analyze accessibility by comparing method of using service that maps port number and third-party transmission method using REST-API.

Ⅰ. 서 론

과학 탐구를 위해서 우선적으로 과학 데이터를 수집하고 분류하고 저장하고 공유하고 분석하는 과정이 진행되어야 한다. 다양한 IoT 장비들과 거대 천체 망원경이나 입자가속기와 같은 거대 실험 장비들이 대거 등장하면서 과학 데이터는 이전에 비해 큰 폭으로 증가되어 빅데이터라고 불리고 있다. 이러한 빅데이터 환경에서는 데이터를 고속으로 전송할 수 있는 기술이 중요한 역할을 한다[1].

대규모의 데이터를 일반적인 네트워크를 이용해 전송하게 되면 네트워크 대역폭에 비해 네트워크 트래픽 발생량이 많아 오버헤드가 자주 발생된다. 이로 인해 전송 속도가 크게 감소되기 때문에 데이터의 고속 전송이 가능한 데이터 전송 전용 노드 DTN(Data Transfer Node)을 경유지에 배치하여 과학 데이터의 초고속 전송이 가능하도록 고속도로 역할을 하는 네트워크 망을 구축하는 프로젝트를 진행하고 있다[2-5]. DTN은 일반적으로 사용되는 리눅스 환경 시스템을 기반으로 거대한 크기의 과학 데이터를 고속으로 전송 할 수 있도록 고대역폭을 지원하는 네트워크 카드나 고성능의 CPU 및 스토리지 같은 고성능의 하드웨어를 장착하고 성능을 최대한 사용할 수 있도록 튜닝까지 적용되어 있는 데이터 전송 특화 서버를 의미한다.

과학 데이터의 고속 전송을 목적으로 ScienceDMZ라고 하는 네트워크 구조가 적용된다. ScienceDMZ는 패킷 로스가 빈번하게 발생되는 일반 네트워크와 과학 데이터 전송 용도의 네트워크를 분리한다. 이 때 사용되는 방법으로는 물리적인 방법으로 라우터를 이용해 망 분리를 하거나 소프트웨어적인 방법으로 SDN(Software Defined Network) 또는 ACL(Access Control List)과 같은 방법을 이용한다. 또한 초고속 전송을 위해 고성능의 하드웨어를 장착하고 지원 대역폭을 최대한 사용할 수 있도록 시스템 및 네트워크를 튜닝 하는 작업 같은 일련의 과정을 통해서 ScienceDMZ를 구축하게 된다[6,7].

현재 ScienceDMZ 모델에서 데이터를 전송만 하는 것에 그치는 것이 아니라 여기서 더 나아가 계산하는 작업도 진행 할 수 있고 노드 자원의 가용성을 높일 수 있도록 하기 위해 클라우드를 도입하였다. 고성능 하드웨어로 구성된 DTN의 유휴 자원을 모아 하나의 통합된 자원 풀을 만들어서 계산하는 작업 필요한 사용자에게 필요한 만큼의 컴퓨팅 자원을 할당해준다. 클라우드는 도커 컨테이너를 기반으로 하는 쿠버네티스가 사용된다. 컨테이너는 애플리케이션과 애플리케이션 실행 환경을 이미지로 만들어 외부와 독립된 상태로 실행될 수 있도록 하는 기술이다. 컨테이너를 사용하면 프로그램을 설치하거나 종속되어 있는 패키지들을 설치하는 과정 없이 쉽게 소프트웨어를 사용할 수 있다. 컨테이너 기반 클라우드는 기존에 머신 단위로 빌려주었던 클라우드에 비해 컨테이너를 실행할 만큼의 자원을 빌려주면 되기에 동일 자원을 가지고 더 많은 사용자가 쓸 수 있어 가용성이 더 높다는 장점이 있다[8].

클라우드 내부는 가상 네트워크 환경을 사용하고 있어 클라우드에서 실행되는 사용자의 웹서버 같은 애플리케이션을 외부에 서비스하거나 외부의 노드와 데이터를 주고받는 등의 작업을 하려면 별도의 네트워크 서비스 생성이 필요하다. 네트워크 서비스는 호스트 머신의 아이피와 컨테이너의 가상 아이피간의 포트 번호를 매핑 시켜주는 역할을 한다. 예를 들어 컨테이너로 80번 포트를 사용하는 웹서버를 생성하면 호스트 머신의 아이피의 80번 포트와 매핑해서 사용할 수 있다. 이러한 방식을 사용하기에 여기에 다른 컨테이너를 생성하게 되면 80번 포트는 중복이기에 81번과 같이 다른 포트 번호를 할당해야 한다. 통신을 위해 별도의 서비스를 생성해야 하는 점과 기존의 서비스들과의 포트 번호 중복을 고려해야 하는 등의 요소는 접근성을 낮추어 사용을 어렵게 한다. 웹서버 같이 지속적인 서비스를 해야 하는 애플리케이션을 실행하는 게 아닌 단순하게 클라우드에서의 작업을 위해 데이터 전송을 원하는 사용자에게는 이와 같은 과

정이 사용을 어렵게 한다.

본 논문에서는 REST API 기반의 3자 전송 서버 모델을 클라우드 환경에서 적용하여 네트워크 서비스 유무와 관계없이 전송이 가능하도록 해 접근성이 향상 되도록 하였다. 2장에서는 ScienceDMZ와 컨테이너 기반 클라우드에 대해 서술하고 3장에서는 REST API 3자 전송 서버를 소개한다. 4장에서 전송 과정을 비교하는 실험을 하고 5장에서 끝을 맺는다.

Ⅱ. 관련 연구

본 장에서는 과학 데이터의 초고속 전송을 위한 네트워크 모델인 ScienceDMZ와 네트워크 망의 노드의 여유 자원을 활용하기 위해 구성된 컨테이너 기반의 클라우드에 대해 설명한다.

2.1 ScienceDMZ

대용량의 데이터를 빠르게 전송하기 위해 초고속 과학 네트워크망 모델인 ScienceDMZ가 사용된다. ScienceDMZ에서는 전송 속도를 높이기 위해 방화벽을 우회하거나 망분리, 네트워크 튜닝 등의 기술을 적용해 빠르게 데이터를 전송할 수 있도록 해준다. ScienceDMZ라는 용어는 네트워크 보안 분야에서 사용되는 DMZ라는 용어로부터 출발하였다. DMZ가 외부로부터의 공격을 차단하기 위해 네트워크의 구조를 분리한 것과 비슷하게 ScienceDMZ는 패킷 로스가 많이 발생하는 일반 용도의 네트워크와 과학 데이터 전송 전용의 네트워크를 서로 분리하여 사용하도록 하여 전송 성능을 높일 수 있도록 설계되었다.

고대역폭 네트워크망의 구축에 있어 기본적으로 높은 전송성능을 낼 수 있는 서버가 필요하다. 이러한 서버에는 10Gbps이상의 네트워크 속도를 지원하는 네트워크 카드가 장착되고 이러한 네트워크 전송 속도를 처리할 수 있도록 높은 성능의 CPU 장착이 필요하다. 여기에 빠른 속도로 데이터를 읽고 쓸 수 있는 SSD(Solid State Drive)와 같이 고성능 스토리지가 장착된다. 이렇게 초고속 데이터 전송을 위한 목적으로 구성되어 활용되는 서버를 DTN이라고 부른다[9].

DTN의 하드웨어를 설치한 뒤 하드웨어가 지원하는 성능을 최대한 끌어낼 수 있도록 적합한 운영체제의 설치와 튜닝 작업을 진행해야 한다. DTN의 운영체제로는 리눅스를 많이 사용하고 있다. 설치시에는 되도록 최신 버전을 사용한다. 리눅스의 버전에 따라 성능의 차이가 발생되기도 한다. 리눅스 커널 6.5 버전과 7.2 버전에서 각각 TCP 성능을 테스트 해보면 전송 작업이 시작되고 전송 속도가 0부터 최고 속도까지 올라가는데 걸리는 시간이 6.5버전에서는 오래 걸리는 반면 7.2버전에서는 단 시간 내에 최고 속도까지 도달하며 이로 인해 전송 시간에 차이가 생기게 된다. 이러한 차이는 커널 버전에 따라 파일 시스템과 네트워크 프로토콜에 변경사항으로 인해 발생된다[10].

네트워크 튜닝에서는 MTU(Maximum Transmission Unit)과 Txqueuelen(Transmit Queue Length)를 조절한다. MTU는 한 번에 전송 가능한 데이터의 크기를 말한다. 이 값의 튜닝을 통해 전송되는 패킷의 사이즈를 늘려 고속 전송이 가능하도록 해준다. 이 값이 너무 크면 패킷을 처리하지 못해 재전송 작업을 진행해야 하기 때문에 오히려 역효과가 있을 수 있다. 반대로 MTU 값이 너무 적으면 프레임 수가 늘어나 속도가 감소된다. 따라서 네트워크 상황에 따라 적절한 값으로 설정해주는 작업이 필요하다. MTU 값은 1Gbps의 네트워크 대역폭에서는 기본 값으로 1500으로 설정되어 있다. 이를 100Gbps의 네트워크 대역폭에서는 9000으로 설정하는 것이 적절하다. 다만 중간에 경유하는 라우터의 MTU 값이 이보다 낮으면 다시 분할하는 작업을 거쳐야 하므로 오히려 속도가 감소될 수 있어 경로를 확인해 보아야 한다. MTU 튜닝 다음으로는 Txqueuelen 값을 튜닝해준다. 이 값은 패킷이 전송되기 전에 저장되는 큐의 사이즈를 의미한다. 이 값이 불충분할 경우 패킷 로스가 발생될 수 있기 때문에 기본값으로 1000으로 설정되어 있는 값을 100Gbps의 네트워크 대역폭에서는 10000으로 설정해 준다[11].

네트워크 튜닝 다음으로는 바이오스 튜닝이 있다. 여기에서 CPU의 성능을 향상시키기 위한 튜닝을 진행한다. CPU의 터보부스트 옵션을 활성화하고 논리적 코어 옵션을 비활성화로 설정해준다. 또한 CPU 사용량이 적을 때에는 idle 상태로 전환되어 이후 고성능을 요구하는 작업 실행 시 최고 성능까지 내는데 오버헤드가 발생하게 되는데 이러한 점을 해소하고자 항상 고성능으로 작동하도록 설정해준다.

CPU 설정 이후에는 I/O 스케줄러라고 하는 블록 디바이스 스케줄링 알고리즘을 변경해준다. 기본 설정으로는 다중 작업에 유리한 Fair 스케줄러로 설정되어 있는데 이 값을 Deadline 스케줄러로 변경한다. Deadline 스케줄러는 연속된 쓰기 읽기 작업을 효율적으로 처리한다. 그 다음은 메모리 버퍼를 튜닝한다. 메모리 버퍼는 BDP(Bandwidth x Delay)라고 하는 값을 계산하여 설정한다. BD는 네트워크 대역폭과 지연시간을 곱하여 네트워크에서 전송되는 데이터 사이

즈를 의미한다. 100Gbps의 속도와 100ms의 지연속도를 기준으로 계산한다면 DBP는 100 Gbps * 100ms = 10000 Mb 가 된다. 여기에 비트단위를 바이트로 바꾸면 1250MB가 된다. 이 값을 기준으로 이 보다 높은 값을 메모리 버퍼로 할당해준다. DTN에서는 넉넉하게 2GB로 할당한다.

이로써 ScienceDMZ 구성에 있어서 중요한 요소인 DTN의 하드웨어 설치와 튜닝이 완성된다. 나머지로 네트워크 구성에 대한 설정을 하면 된다. ScienceDMZ에서는 방화벽을 사용하지 않고 우회하는 방식을 사용한다. 방화벽은 1Gbps의 네트워크 대역폭을 기준으로 설계되었기 때문에 고대역폭 네트워크에서 사용하면 오버헤드가 커서 성능에 악영향을 준다. 방화벽을 사용하지 않는 대신 ACL(Access Control List)을 사용한다. ACL설정을 통해서 신뢰성 있다고 판단되는 노드와 전송에 필요한 포트 번호로만 통신이 가능하도록 설정하여 방화벽을 우회하여도 문제가 없도록 하였다.

앞에서 소개한 ScienceDMZ 모델을 각각의 연구기관에 배치된 DTN에 적용한다. ScienceDMZ로 구성된 네트워크의 구조는 아래의 그림 1과 같다. ACL을 적용하여 패킷 로스가 많은 일반 용도의 네트워크와 과학 데이터 전송 용도의 네트워크를 분리해 사용한다. 고속 전송 시 오버헤드가 많이 발생되는 방화벽을 우회하여 통신할 수 있도록 하여 DTN간의 고속 전송이 가능한 환경을 구축한다.

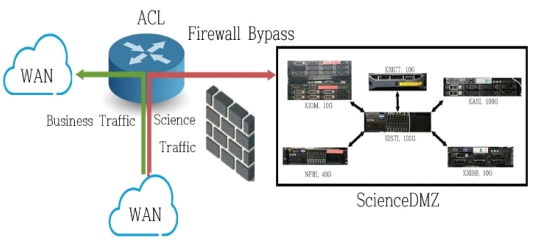

그림 1. ScineceDMZ 네트워크 구성도[14]
Fig. 1. ScienceDMZ Network Diagram

2.2 컨테이너 기반 클라우드

고대역폭 네트워크 전송망의 노드를 데이터 전송 용도 이외에 대규모 또는 복잡한 계산 작업을 수행할 수 있도록 클라우드를 설치한다. 클라우드 설치에는 컨테이너 오케스트레이션인 쿠버네티스를 사용한다. 컨테이너는 애플리케이션과 애플리케이션의 실행에 필요한 환경을 격리한 공간을 의미한다. 컨테이너 방식을 사용하는 클라우드는 사용자에게 노드 단위로 할당할 필요없이 컨테이너 실행에 필요한 자원의 크기 만큼을 할당해 사용할 수 있어 클라우드의 자원을 효율적으로 사용할 수 있다. 또한 컨테이너 내부에는 애플리케이션 실행에 필요한 모든 것이 내장되어 있어 별도의 설치 과정을 요구하지 않기 때문에 컨테이너를 빠르게 생성하고 확장하는 것이 가능하다.

쿠버네티스를 기반으로 구성한 클라우드의 모델은 아래의 그림 2와 같다. 쿠버네티스 내부에는 마스터와 노드로 구분된다. 노드는 실질적으로 클라우드 사용자에게 CPU, GPU 그리고 RAM 과 같은 자원을 할당해주는 역할을 하고 마스터는 이들 노드를 관리하며 스케줄러에 따라 가장 최적의 노드를 선택해 작업을 할당해주는 역할을 한다. 여기에 Ceph 노드가 추가된다. Ceph은 오픈소스 분산 파일시스템으로 블록 스토리지와 오브젝트 스토리지를 제공해주고 확장성과 안정성이 높다는 장점을 가지고 있다. 클라우드에서의 작업이 완료된 후 작업을 삭제하고자 할 때 할당 받았던 자원을 반환하게 되는데 이때 작업으로 생성된 데이터들 또한 사전에 다른 곳에 저장해 두지 않으면 디스크가 반환되면서 같이 소멸되게 된다. 이러한 이유로 작업이 생성되거나 삭제되는 것에 영향을 받지 않고 영구적으로 데이터를 보관할 수 있도록 하기 위한 목적으로 Ceph노드를 추가해 영구적인 스토리지의 할당이 가능하도록 하였다. 또한 Ceph노드는 대용량의 디스크 자원을 보유하고 있어 영구 저장 이외에도 대용량의 디스크 자원을 필요로 할 때 할당 받아 사용할 수 있다.

쿠버네티스에서 작업을 생성하고 삭제하는 등의 과정들이 CLI(Command Line Interface)로 되어 있어 익숙지 않은 사용자들에게는 사용이 어려울 수 있어 여기에 웹 포탈을 추가해 GUI(Graphical User Interface)환경에서 작업할 수 있도록 하였다. 웹 포탈을 통해 접근성을 높이고 사용자는 클라우드 전체의

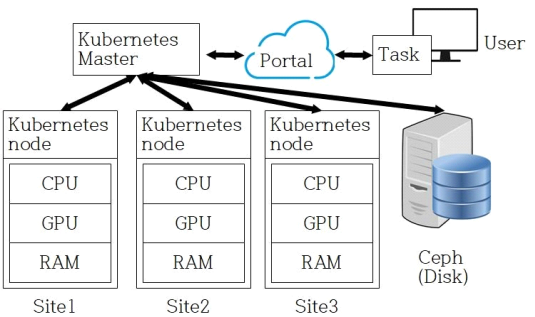

그림 2. 클라우드 플랫폼 구성도[14]
Fig. 2. Cloud Platform Diagram

가용 가능한 상황과 실행되는 작업들의 상황을 실시간으로 확인 할 수 있다.

파드는 쿠버네티스의 기본적인 배포 단위로 단일 또는 다수의 컨테이너로 구성된다. 쿠버네티스는 파드가 외부와 통신으로 가능하도록 네트워크 서비스 기능을 지원하고 있다. 네트워크 서비스를 이용한 파드의 통신과정은 그림 3과 같이 진행된다. 그림 3은 쿠버네티스 네트워크 가이드의 내용을 확장하여 나타내었다. 파드 는 가상 네트워크 인터페이스인 veth0을 사용하며 컨테이너 브릿지 cbr0를 통해 실제 네트워크 인터페이스인 eth0를 거쳐서 외부와 통신할 수 있다. 이러한 과정에서 쿠버네티스 서비스는 컨테이너 브릿지와 실제 네트워크 인터페이스 사이에서 주소를 변환해주는 NAT 역할을 해주게 된다. 쿠버네티스 서비스에 대한 설정들은 쿠버 프록시를 통해 변경 할 수 있다.

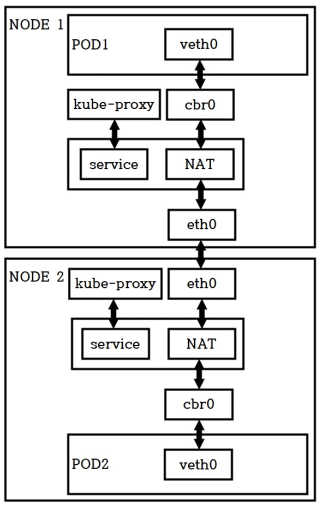

그림 3. 네트워크 서비스를 이용한 파드 통신 과정[15]
Fig. 3. Pod Communication Process using Network Service

Ⅲ. REST API 기반 3자 전송 모델

3자 전송 방법은 데이터를 전송 할 때 출발지나 목적의 노드 이외에 제 3자의 노드를 추가하여 3자 서버를 통해 전송 경로를 제어할 수 있다[9]. 3자 전송이 REST API로 동작하도록 하여 특정 프로그램에 종속되지 않고 REST 형식의 명령어 지원하기만 하면 사용할 수 있어 다양한 프로그램과 연동하여 사용할 수 있어 활용도가 높다는 장점이 있다.

REST API의 기반의 3자 전송 방법을 초고속 전송을 위해 네트워크가 분리되어 있는 ScienceDMZ에서 사용하면 전송 속도가 감소되는 않으면서도 외부 네트워크와 데이터 전송이 가능하도록 해준다[12].

2장에서 소개한 컨테이너 기반의 클라우드와 같이 ScienceDMZ와 같은 과학 데이터 전용 고속도로를 단지 전송 용도로만 사용하는 것에 그치는 것이 아닌 컴퓨팅 자원도 같이 활용 할 수 있는 쪽으로 변화되고 있다. 클라우드를 통해 자원을 할당 받아 머신러닝 같은 계산 작업을 수행하려면 컨테이너로 과학 데이터를 전송하는 과정이 요구된다. 이처럼 클라우드 내부로의 과학 데이터 전송이 필요하지만 실제 데이터를 전송하기 위해서는 사용하는 전송 프로그램의 데이터 채널, 제어 채널, 인증 채널의 포트 번호를 숙지하고 호스트 머신의 포트 번호와 매핑해주는 네트워크 서비스를 생성해 주어야 한다. 또한 네트워크 서비스 생성 시 호스트 머신 또는 다른 네트워크 서비스가 이미 해당 포트 번호를 사용하고 있는지를 확인하고 이미 사용 중이라면 다른 포트 번호로 변경해주는 작업까지 필요하다. 이 같은 과정은 클라우드에서 데이터를 전송하고자 하는 사용자의 접근성을 낮추는 요인이 된다.

제안한 REST API 기반의 3자 전송은 위와 같은 과정의 생략을 통해 접근성을 향상시키도록 한다. 3자 전송 서버는 클라우드가 설치되어 클라우드의 가상 네트워크 인터페이스가 있는 노드에 설치하여 실제 네트워크와 클라우드의 가상 네트워크 간의 전송을 위한 데이터 전송 채널을 오픈할 수 있도록 구성하였다.

3자 전송 서버는 REST API를 기반으로 하기 때문에 종속성 관련 없이 다른 프로그램을 사용하더라도 REST 형식만 따르는 명령어만 전송해주면 사용 할 수 있다. 이러한 점 덕분에 다양한 방식으로 전송 명령어를 보내는 것이 가능하다. REST 명령어를 Curl 같은 기본 유틸 프로그램을 통해 콘솔 환경에서 전송을 할 수도 있고 REST API를 따르는 웹서버를 개설해 인터넷 브라우저를 통해 명령어를 보내 데이터를 전송시킬 수도 있어 활용도가 매우 높다는 장점이 있다.

REST API 기반 3자 전송 서버에서 사용하는 명령어는 아래의 표 1과 같이 디자인하였다. 3자 서버의 IP/info url로 GET 요청을 보내면 데이터베이스에 저장되어 있는 DTN의 이름과 아이피 주소 목록을 보여준다. 여기에 POST 요청에 DTN의 이름과 아이피 정보를 같이 보내는 방법으로 DTN 정보를 추가하거나 갱신하는 것이 가능하다. 반대로 삭제를 원하면 DELETE 요청을 보내면 된다. IP/log url로 GET 요청을 보내면 이전 전송된 결과들을 보여준다. 데이터

표 1. REST API 설계[12]
Table 1. Design of REST API

Method	URL	Payload	Action
GET	IP/info	N/A	Get DTN information
POST	IP/info	DTN_name, DTN_IP	Add DTN
DELETE	IP/info	DTN_name, DTN_IP	Delete DTN
GET	IP/log	N/A	Get transfer result log
GET	IP/transfer	Source_IP, Destination_IP	Directory lookup
POST	IP/transfer	Source_IP_File, Destination_IP_Directory	File transfer

전송 관련은 IP/transfer url에 GET 요청시 해당 노드의 디렉토리 정보를 얻을 수 있고 POST 요청시 데이터 전송을 진행한다.

클라우드 환경에서의 3자 전송은 아래의 그림 4와 같은 방법으로 진행된다. 데이터 전송의 출발지와 목적지로 사용할 Site1과 Site2 노드의 자원을 할당 받아 데이터 전송용 파드를 생성한다. 파드의 이미지는 컨테이너 이미지들을 관리하는 웹사이트인 도커 허브를 통해 받아오거나 이미 노드에 저장되어 있다면 바로 사용할 수 있다. 컨테이너의 이미지에는 과학 데이터 전송에 많이 사용되는 GridFTP[13]가 내장되어 있다. 3자 전송 진행하기 위해 Site3의 노드에는 3자 전송 서버가 설치된다. 3자 전송 서버는 테이블 1과 같은 REST API에 대응하여 GridFTP의 컨트롤 채널을 제어하는 역할을 한다. 사용자는 REST API 명령어를 3자 전송 서버에 보내는 것만으로 Site 1과 Site 2 간의 데이터 전송을 진행할 수 있다.

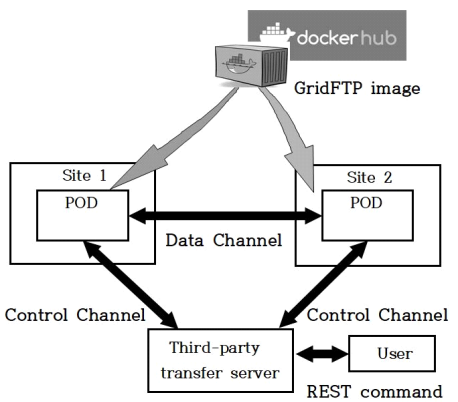

그림 4. 3자 전송 모델
Fig. 4. Third Party Transmission Model

Ⅳ. 클라우드에서의 과학 데이터 전송 비교 실험

본 장에서는 실제 서버인 DTN 과 클라우드로 생성된 파드 간 데이터 전송을 비교하는 실험을 진행한다. 클라우드에서 머신러닝과 같은 작업을 수행하기 위해 클라우드 외부로부터 과학 데이터를 전송 받아야 하는데 이 때 기존의 방법인 네트워크 서비스를 생성하여 진행하는 방법과 제안한 REST API 기반의 3자 전송 방법을 사용하여 테스트를 진행한다. 여기에 비교를 위해 클라우드가 아닌 DTN 간 전송을 추가한다. DTN 간 전송은 클라우드를 거지치 않기에 네트워크 서비스나 REST API 같은 방법을 사용하지 않을 수 있어 서버 본연의 전송 성능을 보여준다.

전송 테스트에 사용한 DTN의 하드웨어 스펙은 아래의 표 2와 같다. Site 1과 Site 2는 서로 다른 기관에 위치한 DTN 이다. Site 1을 DTN으로 사용하고 Site 2의 머신을 클라우드를 통해 파드로 생성하여 사용하였다. 파드 생성시에 사용하려는 컴퓨팅 자원을 명시하거나 사용량 제한을 걸어 파드가 최대 사용할 수 있는 자원량을 정할 수 있다. 이러한 옵션에서 자원 사용량의 제한을 두지 않으면 파드는 호스트 머신과 동일한 컴퓨팅 자원을 사용한다. 즉, Site 2의 하드웨어 스펙이 파드의 스펙이 된다.

네트워크 인터페이스에서는 스펙이 달라진다. Site1의 네트워크 인터페이스가 100 Gbps이고 Site2의 네트워크 인터페이스가 40 Gbps로 되어 있으나 파드에서는 가상 네트워크 인터페이스를 사용하기 때문에 실제로는 10 Gbps로 작동된다. 그 이유는 40 또는 100 Gbps의 경우 포트 타입으로 광케이블을 사용하는데 가상 네트워크 인터페이스는 기본 포트 타입으로 일반적으로 많이 사용되는 랜케이블인 트위스트

표 2. DTN 하드웨어 스펙
Table 2. DTN Hardware Specification

Classification	Site 1	Site 2
Product name	Gigabyte G291-280	Dell PowerEdge R730
CPU	Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz	Intel(R) Xeon(R) CPU E5-2660 v3 @ 2.60GHz
Memory	256 GB	128 GB
Storage	Nvme, 1024 GB	SSD, 512 GB
Network	100Gb Network Adapter	40Gb Network Adapter
Kernel	3.10.0-957.12.2.el7.x86_64	3.10.0-957.12.2.el7.x86_64

페어를 사용하고 있어 최대 속도가 10 Gbps로 제한되어 있다. 이 문제는 가상화가 아닌 호스트의 네트워크 인터페이스를 직접 사용하도록 하거나 다른 컨테이너 네트워크 인터페이스를 사용하여 해결하는 방법도 있으나 스토리지의 읽기 쓰기 속도가 10 Gbps를 넘지 않기 때문에 그러한 방법은 고려하지 않았다.

파드는 가상 네트워크를 사용하고 있어 실제 네트워크를 사용하는 DTN으로부터 직접적으로 데이터를 전송 받을 수가 없어 네트워크 서비스와 같은 기능을 사용해야 한다. 쿠버네티스에는 파드가 외부 네트워크와 통신 할 수 있도록 네트워크 서비스 기능을 제공한다. 이 기능을 사용하여 전송 작업을 진행하는 과정은 그림 5와 같다. 우선 네트워크 서비스를 생성하기 위해 컨테이너 내부의 전송 프로그램에서 사용되는 포트 번호를 알아야 할 필요가 있다. 테스트에서 전송프로그램은 과학 데이터 전송에 많이 활용되는 GridFTP를 사용한다. GridFTP 는 제어 채널로 2811 포트를 데이터 채널로 50000 ~ 51000 번대 포트를 사용하고 인증용 포트로 7512를 사용한다. 해당 포트는 고정된 것이 아닌 기본 값으로 GridFTP 설정 파일을 통해 필요에 따라 변경해 사용할 수 있다. 해당 포트들에 대한 정보를 가지고 네트워크 서비스를 생성하기 위한 Yaml 파일을 작성한다. 컨테이너가 실행되고 있는 실제 머신의 네트워크 아이피가 externalIP 설정되고 해당 머신의 포트 번호를 컨테이너의 포드 번호와 매칭된다. GridFTP 클라이언트를 이용해 전송 명령어를 보낼 때는 파드의 가상 아이피를 사용하는 게 아닌 파드가 실행되고 있는 호스트의 아이피를 사용한다. 매핑 되어 있는 포트 번호로 통신이 오면 파드와 연결되는 방식으로 진행된다.

이와 같은 방식은 사용자가 전송 프로그램에서 사용되는 포트 번호 목록과 호스트 머신의 아이피 주소

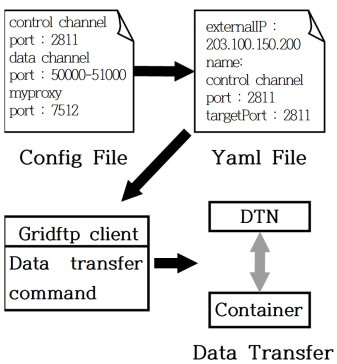

그림 5. 네트워크 서비스를 이용한 전송 예시
Fig. 5. Example of transmission using network service

그리고 네트워크 서비스를 생성하기 위한 정보들을 알아야하기에 사용이 어려워 접근성이 낮다. 제안한 REST API 기반의 3자 전송을 이용한 방법은 그림 6과 같은 과정을 거친다. REST API를 지원한다는 점 덕분에 사용 방법이 한 가지에 국한 되는 게 아닌 REST 형식을 지원하는 요청을 보낼 수 있다면 어떠한 방식이라도 사용할 수 있다. 따라서 콘솔 형태로 사용하거나 웹 브라우저를 통한 전송 작업도 가능하여 활용도가 높다. 3자 전송 서버에 REST API 메시지를 보내는 것으로 전송 작업이 가능하다. 전송 시 출발지와 목적지 아이피에는 실제 아이피 주소가 아닌 클라우드 컨테이너에서 사용하는 가상 아이피를 사용한다. 3자 전송 서버는 클라우드가 설치된 머신에 같이 설치되어 있어 같은 서브넷을 사용하기에 가상 아이피임에도 전송 경로를 설정해 주는 게 가능하다.

앞에서 언급된 전송 방식들을 사용해 1, 10, 25, 50, 100GB 사이즈의 데이터로 전송 테스트를 10회씩 진행하여 얻은 평균 전송 속도를 그림 7을 통해 그래프로 나타내었다. 전송 방식은 3가지로 DTN과 DTN 간의 베어메탈 전송, DTN과 파드 간의 네트워크 서비스를 이용한 전송 그리고 DTN과 파드 간의 REST API 기반 3자 전송으로 진행하였다. 베어메탈 기준으로 1GB 파일이 700 MB/s 로 전송되고 10GB 파일 전송 시에는 735 MB/s 로 속도가 향상되다가 25, 50 ,100GB 같이 사이즈가 큰 파일을 전송할수록 속도가 감소되는 경향을 보인다. 1GB 파일의 경우 전송 속도가 0부터 증가하다가 최고 속도에 도달하기 전에 전송이 완료되기에 10GB에 비해 속도가 약간 낮은 결과를 보여주고 25 GB 이상의 큰 파일은 버퍼보다 커지기 때문에 전송 속도가 감소된다. 반면 네트워크 서비스와 REST API와 같이 DTN과 파드간 전송하는 방법에서는 그러한 경향을 보이지 않았다. 이는 네트워크 카드나 스토리지 같은 하드웨어 장치의 드라이버가 지원하는 기능들이 클라우드 상에서는 제한되어 작동되기 때문이다. 네트워크 서비스를 이용한 방법과 REST API 기반 3자 전송을 이용한 방법이 약간의 오

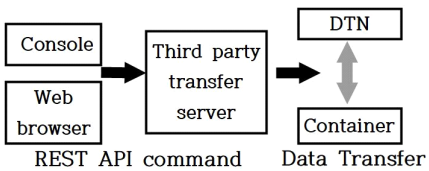

그림 6. REST API 기반 3자 전송 서버를 이용한 전송 예시
Fig. 6. Example of transmission using a third party transmission server based on REST API

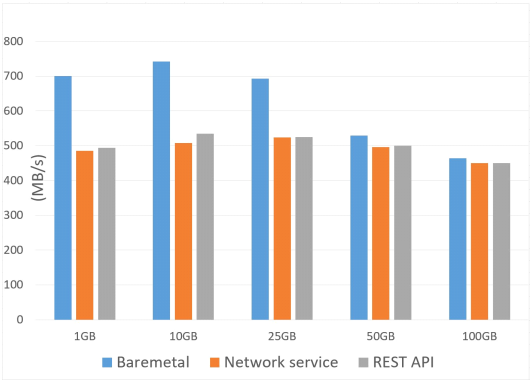

그림 7. 전송 방식 차이에 따른 site1에서 site2로의 전송 속도
Fig. 7. Transmission speed of site1 to site2 by Transmission Method difference

차를 제외하고는 평균 500 MB/s 정도로 전송 속도에는 차이가 거의 없었다.

REST API를 이용한 전송 방식이 네트워크 서비스 방식과 비교 시 속도의 저하가 없었으며 접근성 및 활용도 측면에서 우수하였다.

V. 결 론

본 논문에서는 과학 데이터의 초고속 전송을 위해 구성된 네트워크 망과 여유 컴퓨팅 자원의 활용하기 위한 컨테이너 기반 클라우드에 대해 설명하고 파드와 DTN 간 데이터 전송 방법으로 REST API 기반의 3자 전송을 제안한다.

고대역폭 네트워크망은 대용량의 과학데이터를 전송하는데 있어 방화벽 우회와 일반용도의 트래픽과의 분리 그리고 네트워크 튜닝을 통해 기존의 네트워크 망에 대비 빠른 속도로 전송이 가능하다. 여기에 고대역폭 네트워크망 노드의 여유 자원을 활용해 클라우드를 구성할 수 있다. 클라우드 구축을 통해서 기존에는 단순히 데이터의 전송만을 진행했다면 이제는 전송된 데이터를 활용하여 머신러닝 같은 계산 작업을 진행할 수 있는 환경을 구축한다.

컨테이너 기반의 클라우드는 가상 네트워크를 사용하고 있어 외부 네트워크와 통신시 네트워크 서비스를 사용해야 한다. 네트워크 서비스는 전송 프로그램의 포트 정보, 호스트 머신의 아이피를 사용자가 알아내야 하고 네트워크 서비스에서 포트 번호가 중복되는지도 확인해야해 접근성이 낮다. 제안한 REST API 기반 3자 전송 모델을 이러한 과정을 생략할 수 있어 접근성을 높이고 REST 형식만 따른다면 다양한 매체를 사용할 수 있어 활용도 또한 우수하다.

References

[1] Q. Lu, L. Zhang, S. Sasidharan, W. wu, P. DeMar, C. Guok, J. Macauley, and I. Monga, "BigData express: Toward schedulable, predictable, and high-performance data transfer," in *2018 IEEE/ACM Innovating the Netw. for Data-Intensive Sci.*, pp. 75-84, 2018.

[2] J. Cruchigno, E. Bou-Harb, and N. Ghani, "A comprehensive tutorial on ScienceDMZ," *IEEE Commun. Surv. & Tuts.*, vol. 21, no. 2, pp. 2041-2078, 2018.

[3] E. H. Trussell, L. Rotman, B. Tierney, H. Hester, and J. Zuraski, "The ScienceDMZ: A network design pattern for data-intensive Science," *Scientific Programming*, vol. 22, no. 2, pp. 173-185, 2014.

[4] L. Smarr, C. Crittenden, T. DeFani, J. Graham, D. Mishin, R. Moore, P. Papadopoulos, and F. Wurthwein, "The pacific research platform: Making high-speed networking a reality for the scientist," in *Proc. ACM*, p. 29, 2018.

[5] W. Hong, D. An, J. Lee, J. Moon, and W. Seok, "Deployment and performance analysis of data transfer node cluster for hpc environment," *KIPS Trans. Comput. and Commun. Syst.*, vol. 9, no. 9, pp. 197-206, 2020.

[6] W. Seok, J. Moon, W. Hong, J. Kwak, and M. Lee, "Store-and-forward data transfer using optimized intermediate node," in *2019 20th Asia-Pacific Netw. Oper. and Manag. Symp.*, pp. 1-4, 2019.

[7] J. S. Park, J. H. Park, S. H. Kim, and M. K. Noh, "Performance enhancement method through science DMZ data transfer node tuning parameters," *KIPS Trans. Comput. and Commun. Syst.*, vol. 7, no. 2, pp. 33-40, 2018.

[8] J. Park, S. Lee, K. Jeong, and T. Hong, "KI cloud: Design and implementation of bigdata analysis and machine learning applications on supercomputer," *KIPS Review*, vol. 27, pp. 80-82, 2020.

[9] W. Seok, W. Hong, J. Kwak, and J. Moon,

"DTN-aware store-and-forward data transfer for exabyte scale science big-data," *J. KICS*, vol. 42, no. 10, pp. 1991-1998, 2017.

[10] N. Hanford and B. Tierney, "Recent linux TCP updates, and how to tune your 100G host," Retrieved 2016 [Online], from: https://www.es.net/assets/Uploads/100G-Tuning-TechEx2016.tierney.pdf

[11] K. Kulkarni, S. Badhe, and G. Gadre, "Fine tuning network MTU for HPC cluster: Correlating MTU with PCIe parameters and HCA architecture," in *2015 Int. Conf. Comput. and Netw. Commun. IEEE*, pp. 838-841, 2015.

[12] S. Lee, W. Seok, and J. Moon, "REST API based 3rd-party access model for high-speed transmission in ScienceDMZ," *J. KICS,* vol. 44, no. 7, pp. 1411-1421, 2019.

[13] Globus Connect Server, Retrieved 2021 [Online], from: https://docs.globus.org/globus-connect-server-installation-guide/

[14] K. Kim, et al., "Connecting method research of distributed computing for AI research based on ScienceDMZ," *J. KICS,* vol. 46, no. 6, pp. 1006-1022, 2021.

[15] Kubernetes network overview, Retrieved 2021 [Online], from: https://cloud.google.com/kubernetes-engine/docs/concepts/network-overview

이 상 권 (Sang-kwon Lee)

2015년 : 충북대학교 컴퓨터공학과 학사

2017년 : 충북대학교 컴퓨터공학과 석사

2017년~현재 : 과학기술연합대학원대학교 과학기술정보과학학과 박사과정

<관심분야> 네트워크 성능 테스트, 네트워크 튜닝, 클러스터 컴퓨팅, IoT 네트워크

석 우 진 (Woo-jin Seok)

1996년 2월 : 경북대학교 컴퓨터공학과 학사

2001년 1월 : UNC Chapel Hill Computer Science M.S.

2008년 8월 : 충남대학교 컴퓨터공학과 박사

1998년~현재 : KISTI 근무, (현) 과학기술연구망센터장

2021년~현재 : 한국통신학회 KNOM 연구회 위원장

2020년~현재 : SDN/NFV 포럼 PoC 분과장

2012년~2020년 : UST 겸임교원

<관심분야> TCP 프로토콜, 양자암호통신, 비면허대역 무선통신

문 정 훈 (Jeong-hoon Moon)

1997년 : 경일대학교 컴퓨터공학과 학사

1999년 : 경북대학교 컴퓨터공학과 석사

2000년~현재 : 한국과학기술정보연구원 책임연구원

<관심분야> ScienceDMZ, 네트워크 QoS, 가상화, SDN, 클라우드 컴퓨팅, TCP 성능향상

홍 원 택 (Won-taek Hong)

1998년 : 성균관대학교 정보공학과 학사
2000년 : 성균관대학교 전기전자 및 컴퓨터공학과 석사
2019년 : 성균관대학교 전기전자컴퓨터공학과 박사
2000년~2002년 : ㈜콤텍시스템 기술연구소 연구원
2002년~현재 : 한국과학기술정보연구원 책임연구원
<관심분야> 망 성능 분석, 대용량 데이터 전송 프로토콜, 고성능 네트워크 구조, 클라우드 네트워킹

김 천 용 (Chun-yong Kim)

2013년 2월 : 충남대학교 컴퓨터공학과 학사
2015년 2월 : 충남대학교 컴퓨터공학과 석사
2019년 2월 : 충남대학교 컴퓨터공학과 박사
2019년 7월~현재 : 한국과학기술정보연구원 박사후연구원
<관심분야> Internet of Things, Mobile Edge Computing, Resource Allocation 등

김 기 현 (Ki-hyeon Kim)

2016년 2월 : 강원대학교 컴퓨터과학과 석사
2017년 2월~현재 : 강원대학교 컴퓨터과학과 박사
2016년 2월~현재 : 한국과학기술정보연구원(KISTI) 연구원
<관심분야> ScienceDMZ, SDN/NFV, Network Virtualixation

구 영 훈 (Young-hoon Goo)

2016년 : 고려대학교, 컴퓨터정보학과 학사
2020년 : 고려대학교, 컴퓨터정보학과 박사
2020년~현재 : 한국과학기술정보연구원 박사후연구원
<관심분야> 네트워크 관리 및 보안, IoT 플랫폼, 트래픽 모니터링 및 분석