SaaS OverCloud 기술 문서 #19

DTN을 통한 가상머신 마이그레이션 실험 환경 구축 및 검증

Document No. SaaS OverCloud #19
Version 0.5
Date 2018-12-15
Author(s) GIST Team, KISTI

■ 문서의 연혁

버전	날짜	작성자	비고
초안 - 0.1	2018. 10. 18	Jargalsaikhan Narantuya	
0.2	2018. 11. 01	Jargalsaikhan Narantuya, KISTI	
0.3	2018. 11. 17	Jargalsaikhan Narantuya, KISTI	
0.5	2018. 12. 15	Jargalsaikhan Narantuya	
0.7			
0.8			
0.9			
1.0			

본 문서는 2018년도 정부(미래창조과학부)의 재원으로
정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.R7117-16-0218,
이종 다수 클라우드 간의 자동화된 SaaS 호환성 지원 기술 개발)

The research was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No.R7117-16-0218, Development of Automated SaaS Compatibility Techniques over Hybrid/Multisite Clouds)

Contents

SaaS OverCloud: DTN을 통한 가상머신 마이그레이션 실험 환경 구축 및 검증

표 목차

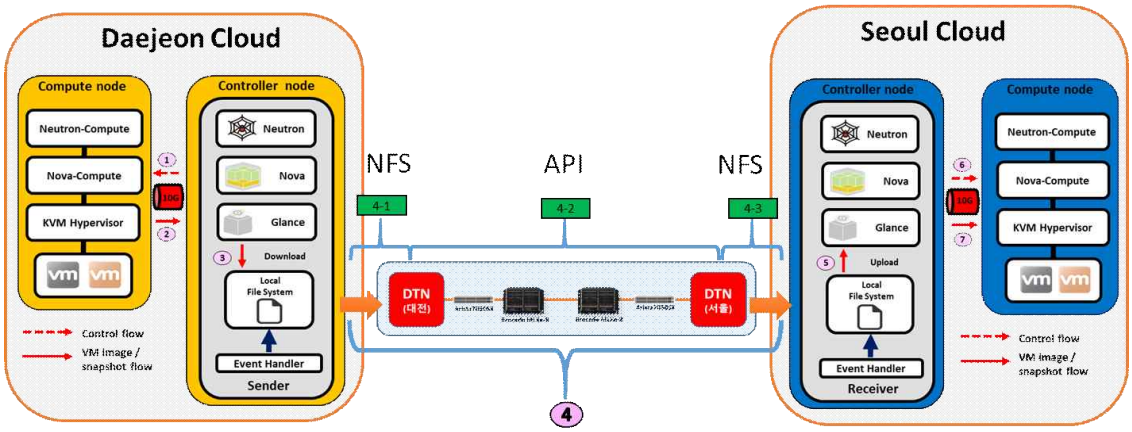
그림 목차

SaaS OverCloud #19.

DTN을 통한 가상머신 마이그레이션 실험 환경 구축 및 검증

1. DTN을 통한 가상머신 마이그레이션 실험 환경 구축

1.1. 목적 및 개요



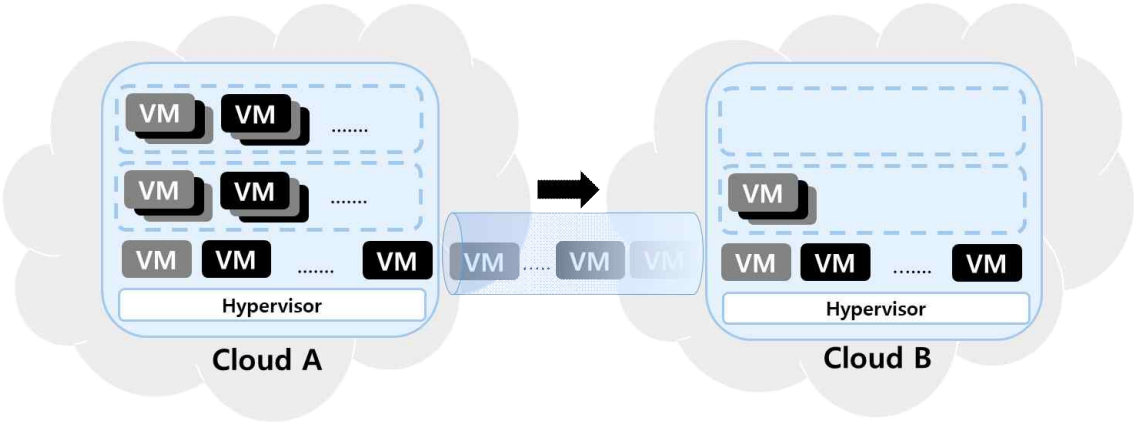
<그림1. DTN을 통한 가상머신 마이그레이션 실험 환경>

o 본 기술문서에서 멀티사이트 클라우드 환경에서의 고속 OverCloud 마이그레이션을 위한 실험 환경 구축 및 검증을 논한다. 멀티 사이트 클라우드 환경을 대전 및 서울에 있는 데이터센터에서 OpenStack [1] 기반으로 구축하고, 구축한 클라우드를 고속 데이터 전송을 지원하는 KREONET-S 네트워크에 Data Transfer Node (DTN)을 통해 연결하였다.

o 멀티사이트 클라우드 관리를 위한 OpenStack Cascading, SlapOS 등 다양한 공개 소프트웨어가 진행되고 있으나, 이는 클라우드 브로커의 역할을 수행할 뿐 멀티사이트 클라우드 환경에 대한 응용의 실행환경 의존성을 근본적으로 해결하지는 못하고 있다. 단일 클라우드에서 가상머신을 마이그레이션할 때 가상머신의 다운 시간을 최소화하기 위한 OpenStack 라이브 마이그레이션, VMware vMotion 등의 기술들이 개발되어 있으나, 멀티사이트 클라우드 환경에서의 가상머신 마이그레이션 관련 기술들은 아직 개발되어 있지 않다.

o 멀티사이트 클라우드 환경에서 자원 관리 및 여러 까지 긴급 상황들이 있을 때 한 사이트에 있는 서비스들을 다른 사이트로 옮겨야 하는 일이 자주 생긴다. 이러한 사이트간의 클라우드 기반 서비스 마이그레이션을 OverCloud 마이그레이션 이라고 한다. 그러나, OverCloud 마이그레이션을 위한 라이브 마이그레이션 기술이 개발되어 있지 않아서, 마이그레이션 동안의 서비스다운 시간이 길다. 본 문서에서 마이그레이션 수행시 발생하는 시간지연 문제를 해결하기 위하여, 멀티사이트 클라우드 환경에서 고속 OverCloud 마이그레이션 실험 환경을 구축 및 검증한다.

1.2. 멀티사이트 클라우드 환경 구축



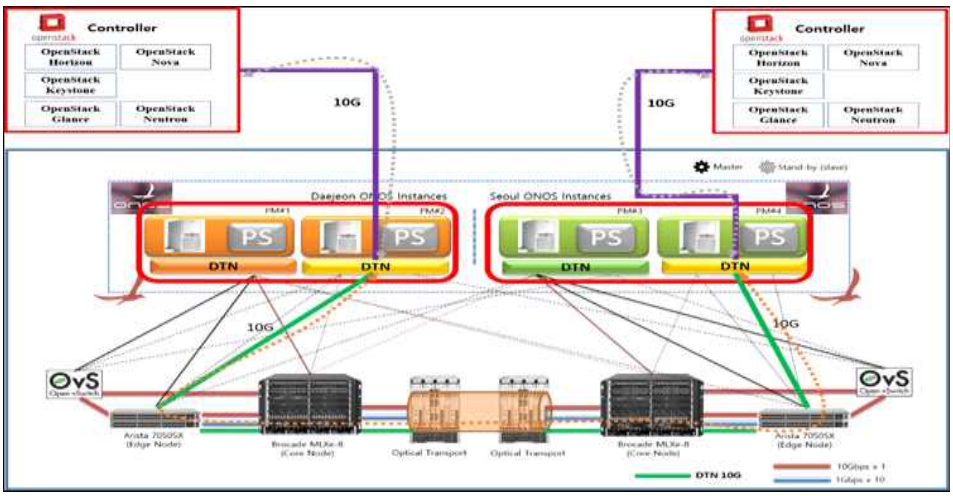
<그림 2 사이트간 가상머신 마이그레이션>

o 멀티사이트 클라우드는 다수의 단일 클라우드를 통합하여 하나의 시스템으로 구축하는 기술이다. 이러한 멀티사이트 클라우드 환경에서 가상 머신을 하나의 물리머신에서 다른 물리머신으로 이동시키는 가상 머신 마이그레이션 기술을 적용하여 클라우드간 자원의 효율적인 관리를 가능하게 할 수 있다. 본 문서에서는 OpenStack 기반 멀티사이트 클라우드 환경에서 가상 머신 마이그레이션을 지원하는 클라우드 환경을 구축 및 검증한다.

o 서로 다른 사이트에 있는 2개의 OpenStack 클라우드 환경을 연동하여 멀티사이트 클라우드 환경을 구축하였다. 각 OpenStack 클라우드는 하나의 컨트롤러 노드와 컴퓨트 노드로 이루어져있다. 컨트롤러 노드는 관리 네트워크 (Management Network)를 통해 사용자로부터 가상 머신의 생성 혹은 삭제 명령, 마이그레이션 명령을 전달한다. 컴퓨트 노드는 컨트롤러 노드로부터 명령을 받아 하이퍼바이저 상에 가상 머신의 생성, 삭제, 마이그레이션 등을 수행한다.

o 컨트롤러 노드와 컴퓨트 노드는 총 3개의 네트워크 인터페이스를 통해 필요한 데이터나 명령을 주고받는다. 관리 네트워크 (Management Network)를 통해 컨트롤러 노드의 명령이나 컴퓨트 노드의 응답이 전달되며, 데이터 네트워크 (Data Network)를 통해 컴퓨트 노드의 가상 머신이 다른 subnet에 있는 가상 머신과 데이터를 주고받을 수 있다. 컨트롤러 노드와 컴퓨트 노드가 설치되어있는 호스트 컴퓨터와 컴퓨트 노드의 가상 머신은 외부 네트워크 (External Network)를 통해 인터넷에 접근한다.

1.3. 사이트간의 네트워크 연결도



<그림 3 대전-서울 DTN 구성도>

o 그림 3과 같이 고속 OverCloud 마이그레이션을 지원하는 멀티사이트 클라우드 환경 구축을 위하여 대전 및 서울에 있는 데이터센터에서 각 사이트의 클라우드 환경을 OpenStack 기반으로 구축하고 고속 데이터 전송용 네트워크인 KREONET-S에 연결하였다.

o KREONET-S는 KISTI에서 개발한 SD-WAN 기반의 프로그래머블/가상화 코어 및 에지/엑세스 네트워크 인프라를 지원한다. KREONET-S의 데이터 평면 네트워크 인프라는 크게 하드웨어 스위치와 소프트웨어 스위치로 구성되며 이들은 대전과 서울의 KREONET 지역망 센터에 구축되어 있다. 지역망 센터에는 하드웨어 스위치 장비로 코어 노드 및 에지 노드가 구축되어 있으며 이와 함께 소프트웨어 스위치로써 다수의 Open vSwitch (OVS)들이 범용 서버 상에 구축되어 있다. 한편, 하드웨어 스위치들 사이의 링크는 다중 링크로 구성되며 이는 10개의 1Gbps와 1개의 10Gbps 링크들로 이뤄진다. 또한 하드웨어 스위치와 OVS 스위치 혹은 OVS 스위치들 사이의 링크의 경우 10Gbps의 대역폭을 지닌다.

o 서로 다른 지역 간의 하드웨어 스위치 사이의 네트워크 구성은 패킷 광 전송(Packet Optical Transport) 기반의 다중 계층(Multi Layer) 구조로 구성되었으며 이는 효율적인 네트워크 운용 및 이에 수반하는 비용 절감의 측면에서 기존에 구축되어 운영 중인 KREONET 네트워크를 활용한다 [2].

2. DTN을 통한 가상머신 마이그레이션 실험 환경 검증

2.1. 클라우드간의 가상머신 마이그레이션 자동화 소프트웨어

o 가상머신 마이그레이션 자동화를 위해 Migrator 및 Receiver 파이썬 프로그램들을 개발하였다. Migrator는 Multi-site A의 컨트롤러 노드에서 동작하며, OpenStack API의 스냅샷 기능을 통해 컴퓨트 노드에 있는 가상 머신의 현재 상태를 image 파일 형태로 로컬 시스템에 저장한다. 그 다음에 로컬 시스템에 저장된 image 파일을 KREONET-S를 통해 Multi-site B의 컨트롤러 노드에 있는 특정한 디렉터리로 전송한다.

o Receiver는 Multi-site B의 컨트롤러 노드에서 동작하며, 리눅스 커널 서브시스템 중 하나인 inotify 이벤트 통보 기능을 통해 Multi-site A에서 Migrator가 보낸 image 파일들이 저장되는 디렉터리를 listen한다. 해당 디렉터리에 새로운 파일이 도착할 때 리눅스 커널에서 도착한 파일에 대한 notify 메시지를 Receiver에게 보낸다. Receiver는 notify 메시지를 통해 도착한 image 파일에 대한 정보를 확인하고, 그 image 파일을 OpenStack API를 통해 Multi-site B 컨트롤러 노드의 Glance에 올린다. 마지막으로, Receiver는 Glance에 올린 image를 통해 Multi-site B의 컴퓨트 노드에서 가상 머신을 생성한다.

2.1.1. 마이그레이션 소프트웨어 사용 방법

o The program works in OpenStack based cloud environment. It enables automated virtual machine migration between two different Openstack based cloud environments [3].

o **Current Support**

- Ubuntu Operating System 16.04.01 LTS
- OpenStack Ocata release

o **Prerequisites:**

1. Two OpenStack based clouds (Source and Destination clouds)
2. Install figlet and git on controller nodes (# apt-get install figlet, git)
3. Set SSH connection without password between the controller nodes
4. Flavor name, and its related resources should be exactly same on both source and destination clouds.
5. Additinional IP address for setting external network gateway on destination cloud. It is required for step 7 of following user guide.

o **User guide:**

1. Load admin-openrc file for Openstack identification on both source and destination clouds
2. Download Cloud-to-Cloud-Migration folder on the contreller nodes of both source and destination clouds
   git clone https://github.com/K-OverCloud/Cloud-to-Cloud-Migration.git
3. On the source cloud, move to Cloud-to-Cloud-Migration/SourceCloud/ directory
4. On the destination cloud, move to Cloud-to-Cloud-Migration/DestinationCloud/ directory
5. If ownership of shared folder is root, change it to user
   On the source cloud:
   #chown username:username -R ~/Cloud-to-Cloud-Migration/SourceCloud/shared/
   On the destination cloud:
   #chown username:username -R ~/Cloud-to-Cloud-Migration/DestinationCloud/shared/
6. Execute receiver.py on the destination cloud ($ python receiver.py)
   Enter shared directory at source cloud: "sourceUser@SourceIP:/home/sourceUser/Cloud-to-Cloud-Migration/SourceCloud/shared/"
   For convenience, you can set the input manually as sample input in receiver.py file
7. Execute sender.py on the source cloud ($ python sender.py)
   Enter shared directory at destination cloud: "destinationUser@DestinationIP:/home/destinationUser/Cloud-to-Cloud-Migration/DestinationCloud/shared/"
   Enter external gateway IP for destination cloud: "IP for external network gateway"
   Enter VM migration order: ["VM1", "VM2", "VM3"]
   For convenience, you can set the inputs manually as sample input in sender.py file

For re-using the migration tool, do following:

1. On the destination, delete configurations.json, completed, and all image files in /Cloud-to-Cloud-Migration/DestinationCloud/shared/ folder
2. On the source, delete configurationCompleted, and all image files in /Cloud-to-Cloud-Migration/SourceCloud/shared/ folder
3. Remove all network configurations, instances, and images on the destination

2.2. 클라우드간 파일 전송

2.2.1. NFS 기반의 DTN-Cloud 연결 및 설정

o OpenStack Controller 내의 Glance 서비스에서 이미지 파일들이 저장되는 위치는 "[glance_store]" 부분의 filesystem_store_datadir에서 설정되고 그림 1에서는 /var/lib/glance/images/위치에 해당된다. 실제로 /var/lib/glance/images 위치에는 아래 그림 2처럼 VM 이미지 파일들이 저장된다.

```
[glance_store]
...
stores = file,http
default_store = file
filesystem_store_datadir = /var/lib/glance/images/
```

<그림 4. [glance_store]에서의 주요 설정>

```
root@RL2-controller:/var/lib/glance/images# la -la
total 208060
drwxr-xr-x 2 glance glance     4096 Oct  8 15:06 .
drwxr-xr-x 4 glance glance     4096 Jul 19  2016 ..
-rw-r----- 1 glance glance 22151168 Oct  8 15:04 1ba294a7-13bf-4770-8367-04d5c2df5b63
-rw-r----- 1 glance glance 22085632 Oct  8 15:04 1c0c5f40-4f44-420b-8d22-5c7c7062338b
-rw-r----- 1 glance glance 22085632 Oct  8 15:04 3a4fd6a5-ffd8-4780-b83a-cc7228b85730
-rw-r----- 1 glance glance 22151168 Oct  8 15:04 4cf77395-281a-4523-a383-12504e9bdc79
-rw-r----- 1 glance glance 22872064 Oct  8 15:04 58ff183b-eacd-4628-912c-a92d0d8e0a9a
-rw-r----- 1 glance glance 22085632 Oct  8 15:04 59d815ee-e59d-4d67-a2fc-d6da80cfa6fa
-rw-r----- 1 glance glance 13287936 Jul 31  2016 67996954-629e-452f-9f04-93c391f8a017
-rw-r----- 1 glance glance 22085632 Oct  8 15:04 91472e31-20c9-452f-8f3e-edf403605d2f
-rw-r----- 1 glance glance 22151168 Oct  8 15:04 9234cf12-8362-42d6-9a0a-7d2f44b9a54c
-rw-r----- 1 glance glance 22085632 Oct  8 15:04 d00e6b4b-ea92-423d-a923-912648625c63
```

<그림 5. 가상머신 이미지 파일 목록>

o CentOS 7이 기본 OS로 설치된 DTN 노드에 NFS 서버를 설치하기 위해서 아래와 같이 nfs-utils 패키지를 설치한다.

```
# yum install nfs-utils
```

o 패키지 설치가 완료된 후에는 /etc/exports를 수정한다. 작성 내용으로는 export 할 디렉토리(예, /home/nfsTest), 접근 가능한 네트워크 대역(예, client_ip_address) 및 접근허가 옵션들(예, rw, sync, no_root_squash)이다. 설정이 완료된 후에는 NFS 서버를 재구동 한다.

```
# vi /etc/exports
...
/home/nfsTest   client_ip_address(rw, sync, no_root_squash)
...
```

o 아래 표에서는 NFS 서버에서 구동되는 주요 데몬들의 포트 정보를 보여준다. portmap 데몬은 RPC(Remote Procedure Call)을 이용하는 프로그램을 지정된 포트에 매핑시키는 역할을 한다. mountd 데몬은 /etc/exports 파일의 설정된 내용에 따라 마운트 요청을 처리한다. nfs데몬은 마운트가 이뤄진 후, 마운트된 디렉토리에 읽고 쓰는 등의 작업을 수행한다. nlockmgr 데몬은 파일 잠금을 통해 여러 사용자가 동시에 한 파일을 수정하는 것을 방지한다.

표 1. NFS 서버에서의 주요 데몬 포트 정보

포트번호	프로토콜	데몬
111	tcp/dup	portmap
2049	tcp/dup	nfs
random	tcp/dup	mountd
random	tcp/dup	nlockmgr

o Ubuntu 16.04이 기본 OS로 설치된 OpenStack Cloud Controller 노드에 NFS 클라이언트(NFS version 4)를 설치하기 위해서 아래와 같이 nfs-utils 패키지를 설치한다.

```
# apt-get install nfs-common
```

o DTN의 NFS 서버에 대한 마운트 포인트를 설정하기 위해 디렉토리(예, /home/nfsClient)를 생성한다.

```
# mkdir /home/nfsClient
```

o NFS 서버와 클라이언트 간 마운트을 아래와 같이 수행하고, “df” 명령어를 통해 마운트된 것을 확인한다.

```
# mount server_ip_address:/home/nfsTest /home/nfsClient
# df -h
Filesystem                          Size  Used Avail Use% Mounted on
/dev/sda1                           269G  1.4G  254G   1% /
none                                4.0K     0  4.0K   0% /sys/fs/cgroup
udev                                2.9G  4.0K  2.9G   1% /dev
tmpfs                               596M  716K  595M   1% /run
none                                5.0M     0  5.0M   0% /run/lock
none                                3.0G     0  3.0G   0% /run/shm
none                                100M     0  100M   0% /run/user
server_ip_address:/home/nfsTest     173G   11G  162G   6% /home/nfsClient
```

o 재부팅시 자동 마운트를 위해 다음과 같은 방식으로 /etc/fstab을 변경한다.

```
# vi /etc/fstab
...
[NFS Server]:/home/nfsTest /home/nfsClient nfs rsize=8192, wsize=8192, timeo=14, intr
...
```

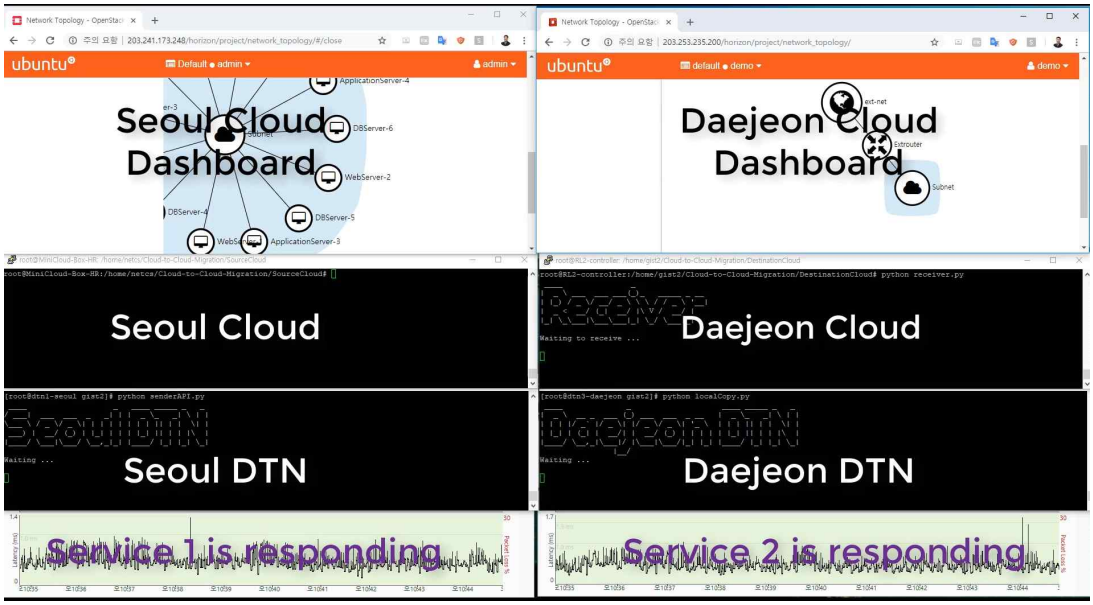
2.2.2. DTN 노드간의 파일 전송용 API 사용 방법

- o 디렉토리 조회 : curl -H "Content-Type: application/json" -X GET -d '{"Source":"Source-IP:Source-port/Source-directory","Dest":"Dest-IP:Dest-port/Dest-directory"}' "APIServerIP":22641/ftp
- o 파일 전송 : curl -H "Content-Type: application/json" -X POST -d '{"Source":"Source-IP:Source-port/Source-directory","Dest":"Dest-IP:Dest-port/Dest-directory"}' "APIServerIP":22641/ftp
- o DTN 데이터베이스 조회 : curl -X GET "APIServer-IP":22641
- o 전송로그 데이터베이스 조회 : curl "APIServer-IP":22641/log

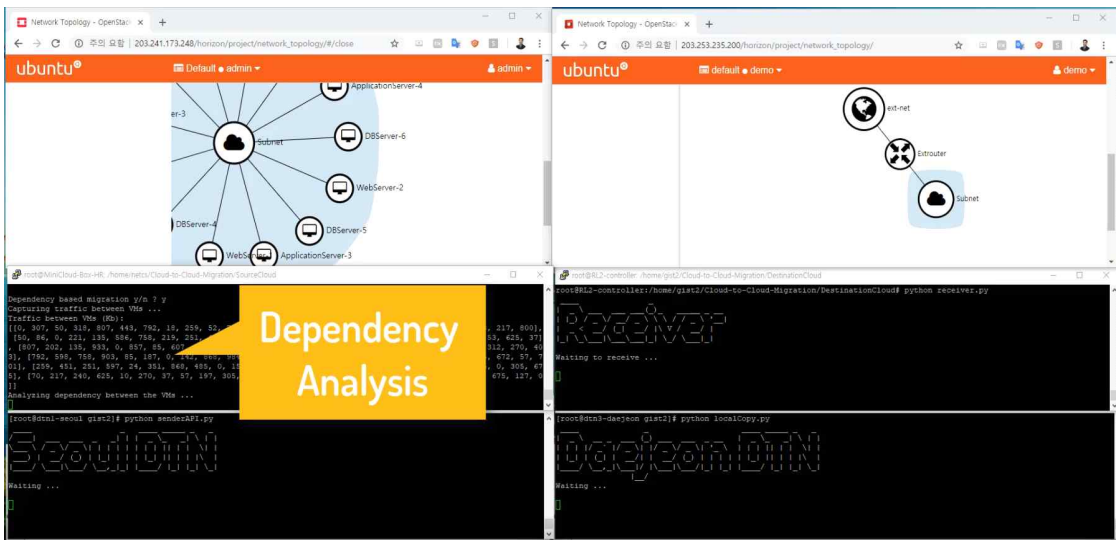
```
[kisti@localhost ~]$ curl -H "Content-Type: application/json" -X GET -d '{"Sourc
e":"210.119.43.34:2811/export/data/","Dest":"203.241.173.215:2811/export/data/"}
' 203.253.235.158:22641/ftp
ftp://210.119.43.34:2811/export/data/
    10G.dat
    1G.dat
    1G_file
    1Gtestfile.dat
    backup/

ftp://203.241.173.215:2811/export/data/
    10G.dat
    1G
    1G.dat
    backup/

[kisti@localhost ~]$ curl -H "Content-Type: application/json" -X POST -d '{"Sour
ce":"210.119.43.34:2811/export/data/1Gtestfile.dat","Dest":"203.241.173.215:2811
/export/data/1Gtestfile.dat"}' 203.253.235.158:22641/ftp
Source: ftp://210.119.43.34:2811/export/data/
Dest:   ftp://203.241.173.215:2811/export/data/
  1Gtestfile.dat

   1000000000 bytes       112.20 MB/sec avg       112.62 MB/sec inst
[kisti@localhost ~]$ curl -H "Content-Type: application/json" -X GET -d '{"Sourc
e":"210.119.43.34:2811/export/data/","Dest":"203.241.173.215:2811/export/data/"}
' 203.253.235.158:22641/ftp
ftp://210.119.43.34:2811/export/data/
    10G.dat
    1G.dat
    1G_file
    1Gtestfile.dat
    backup/

ftp://203.241.173.215:2811/export/data/
    10G.dat
    1G
    1G.dat
    1Gtestfile.dat
    backup/
```

<그림 6. DTN 노드간의 파일 전송용 API 실행 결과>

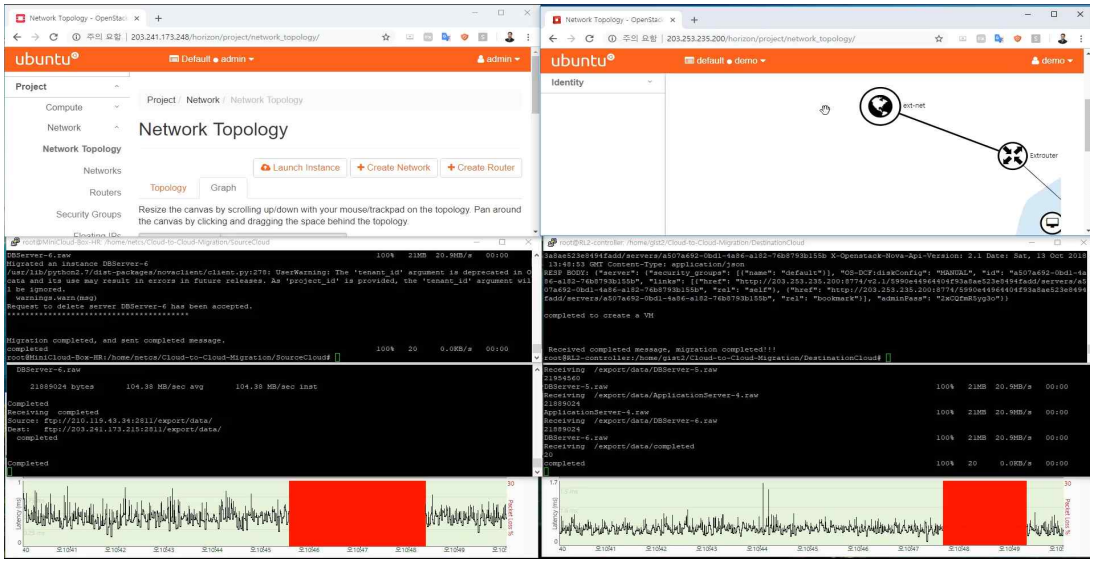
2.3. DTN을 통한 가상머신 마이그레이션 실험 환경 검증



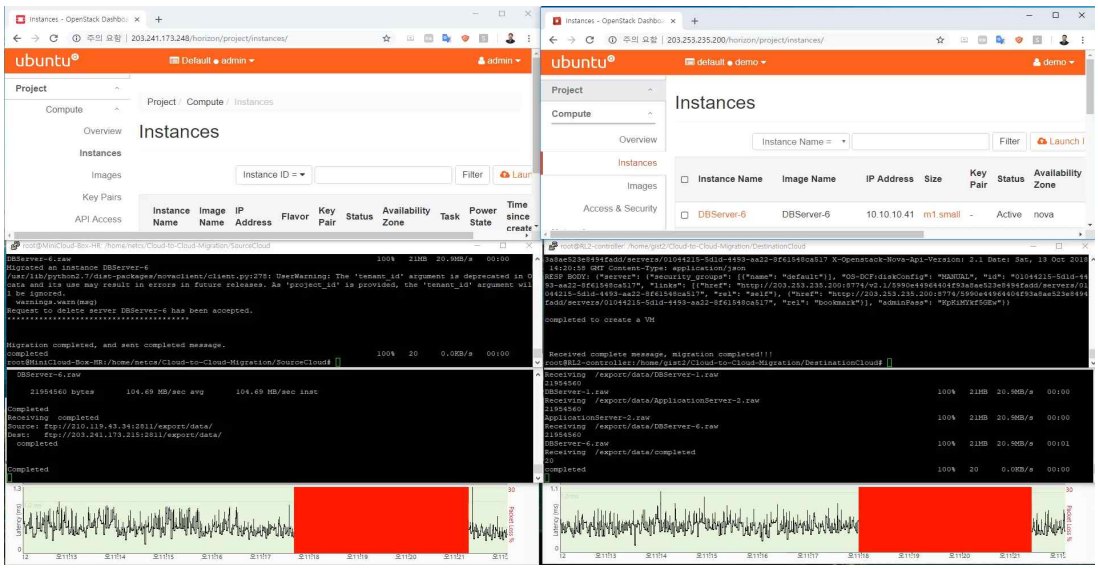
<그림 7. 서울-대전 DTN을 통한 가성머신 마이그레이션 전>



<그림 8. 가상머신간의 종속 분석 과정>



<그림 9. 종속성 분석 기반 사이트간의 가상머신 마이그레이션>



<그림 10 종속성 분석 없는 사이트간의 가상머신 마이그레이션>

References

[1] https://www.openstack.org/
[2] 김용환, 김동균 (2016). SD-WAN 기반의 사용자 중심 가상 전용 네트워크 시스템 설계 및 구현. 한국 통신학회 논문지, 41(9), 1081-1094.
[3] https://github.com/K-OverCloud/Cloud-to-Cloud-Migration

SaaS OverCloud 기술 문서

작성기관: 광주과학기술원
작성년월: 2018/12