클라우드 파일/블록/객체 스토리지의 통합사용을 위한 소프트웨어 정의 스토리지 자동 설정 모듈의 설계 및 구현

(Design and Implementation of Software-Defined Storage Autoconfiguration Module for Integrated Use of Cloud File/Block/Object Storage)

박선*, 차병래*, 김종원**

(Sun Park, ByungRae Cha, Jongwon Kim)

요 약

클라우드 컴퓨팅(Cloud Computing)의 경제성과 유연성을 향상시키기 위해 복잡해지는 자원의 운영 및 관리를 자동화하는 추세에 있다. 그러나 클라우드 스토리지에 대한 자동화는 제조업체의 스토리지 하드웨어에 종속되나, 사용자가 필요로 하는 용도에 맞추어 스토리지 유형을 유연하게 지원할 수 없다. 본 논문에서는 클라우드 스토리지의 자동화 추세에 맞추어 사용자의 환경에 연계한 블록/파일/객체 스토리지를 통합으로 지원하는 자동 설정 모듈을 제안한다. 제안방법은 클라우드 스토리지인 ceph을 자동으로 설치하기 위하여 Chef 구성관리도구 기반의 자동설치 및 설정 모듈을 제안하였으며, 사용자들이 ceph 스토리지를 쉽게 사용할 수 있도록 쉘 프로그램 기반의 블록/파일/객체 스토리지 자동설정 모듈을 제안하였다. 제안방법은 하드웨어 종속 없이 가상이나 물리적인 사용자 환경에서도 자동적으로 공유파일 스토리지, 블록 스토리지, 객체 스토리지에 대한 설정 및 관리를 쉽게 할 수 있다.

■ 중심어 : 파일 스토리지; 블록 스토리지; 객체 스토리지; 자동설치 및 설정; 소프트웨어 정의 스토리지

Abstract

In order to improve the economics and flexibility of cloud computing, tendency to automate the operation and management of cloud resources has become complicated. However, while automation for cloud storage depends on the manufacturer's storage hardware, it cannot flexibly support the storage type in accordance with users' needs. In this paper, we propose an automatic configuration module that supports block/file/object storages suitable for user environment. In order to automatically install ceph, a cloud storage, we propose an automatic installation and configuration module based on the Chef configuration management tool. In addition to that, we also propose an automatic configuration module based on a shell programming in pursuit of enabling users to use ceph storage of block/file/object. The proposed method can automatically set up and manage shared file, block, and object storages in a virtual or physical user environment with no hardware dependencies.

■ keywords : File Storage; Block Storage; Object Storage; Auto Installation & Configuration; Software-Defined Storage (SDS)

Ⅰ. 서 론

최근 클라우드 컴퓨팅(Cloud Computing)에 연계된 가상화 기술의 발전에 영향을 받아서, 클라우드 데이터센터들은 서버, 네트워크, 스토리지를 위한 자원 집합들을 하나로 묶는 융합형(hyper-convergence) 구성으로 단순하게 재구성하면서 복잡해진 자원 운영/관리를 자동화하여 경제성과 유연성을 향상시키기 위해 노력하고 있다. 즉 제공하는 자원집합의 유연성과 경제성에 대한 요구가 계속적으로 증대됨에 따라 단순화되고 개방된 융합형 노드들로 만들어내는 자원 인프라와 이 인프라를 사용자의 요구에 맞춰 선택적으로 대응할 수 있는 자동 설치 및 관리 기술들을 요구하고 있다 [1].

클라우드 스토리지는 파일 스토리지, 블록 스토리지, 객체 스토리지의 유형으로 구분된다. 파일 스토리지는 공유 파일 시스템을 통해 서버와 애플리케이션에 데이터에 대한 액세스를 제공하도록 클라우드에 데이터를 저장하는 방법이다. 블록 스토리지는 클라우드 컴퓨팅의 가상머신인 인스턴스에 스토리지를 추

* 정회원: 제노테크(주), 광주과학기술원 전기전자컴퓨터공학부
** 교신저자: 광주과학기술원 전기전자컴퓨터공학부
이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (2016R1D1A1B03934823). This research was one of KOREN projects supported by National Information Society Agency (18-951-00-001).

접수일자 : 2018년 08월 21일
수정일자 : 1차 2018년 09월 13일, 2차 2018년 11월 27일
게재확정일 : 2018년 11월 27일
교신저자 : 김종원, 박선 e-mail : sunpark@smartx.kr

가하여 마치 일반 컴퓨터에 하드디스크를 추가하여 스토리지를 사용하는 것과 같이 사용하는 스토리지이다. 객체 스토리지는 객체들 단독으로 구성될 수 있는 스토리지로 사용자의 계정을 통하여 파일이나 데이터를 저장할 수 있는 스토리지이다. 스토리지는 하드웨어 중심의 스토리지 시스템에서 점차 소프트웨어 정의 스토리지(SDS: Software-Defined Storage)로 이동이 가속화 되고 있다 [2, 3, 4, 5, 6].

소프트웨어 정의 스토리지에 대한 제품들로는 델 EMC, HPE, 뉴타닉스 등이 있다. 델 EMC의 스토리지 제품군으로는 VxRail, IsilonSD Edge, ECS 등이 있으며, VxRail은 객체 스토리지를 지원하고, IsilonSd Edge는 하둡파일 스토리지 및 파일 스토리지를 지원하며, ECS는 객체 스토리지, 하둡파일 스토리지, 파일 스토지리를 지원한다. 이들 제품군은 델 EMC 스토리지 하드웨어에 종속되어 있다 [7]. HPE (Hewlett Packard Enterprise)에서는 가상화 환경에 스토지리를 지원하기 위하여 HPE StoreVirtual VSA 소프트웨어 및 스토리지 서버군을 보유하고 있다. HPE StoreVirtual VSA는 서버의 내부 또는 직접 연결된 스토리지를 공유 스토리지 배열로 사용할 수 있도록 지원하나 단순히 블록스토리지만을 지원하고 있다 [8]. 뉴타닉스(Nutanix)는 서버, 스토리지, 네트워크를 하나의 노드에 융합한 가상 컴퓨팅 플랫폼을 보유하고 있다. 뉴타닉스 게스트 툴을 이용하여 파일 스토리지 및 블록 스트로지를 지원하나 객체 스토리지를 지원하지 않는다 [9]. VxRail, IsilonSD Edge, ECS, HPE StoreVirtual VSA, Nutanix 등 클라우드 스토리지 제품들은 판매되는 스토리지 하드웨어에 종속 지원되어 다양한 종류의 하드웨어를 지원할 수 없거나, 사용자가 필요로 하는 용도에 맞추어 파일 스토리지, 블록 스토리지, 객체 스토리지를 유연하게 지원할 수 없는 제약을 가지고 있다.

본 논문은 다양한 가상환경 및 물리적인 환경에서 사용자의 필요에 맞추어 유연하게 파일 스토리지, 블록 스토리지, 객체 스토리지를 자동으로 설정하여 사용할 있도록 지원하는 자동 설정 모듈을 제안한다. 본 논문에서 제안한 방법은 하드웨어 종속 없이 가상이나 물리적인 사용자 환경에서도 자동적으로 공유파일 스토리지, 블록 스토리지, 객체 스토리지에 대한 설정 및 관리를 사용자들이 쉽게 할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구로 자동설정 모듈에서 사용되는 프레임워크 및 도구에 대하여 알아보고, 3장에서는 제안방법과 각 모듈의 적용례를 알아보고, 4장에서는 실험 및 평가를, 5장에서는 결론을 제시한다.

Ⅱ. 관련도구

1. ceph

ceph는 분산 클라우드 스토리지로 인텔 프로세서 기반의 범용 하드웨어를 사용해 대규모 소프트웨어 정의 스토리지를 구현하기 위해 시작된 오픈소스 프로젝트이다. ceph은 세이지 웨일이 캘리포니아 대학에서 박사 논문을 준비하면서 개발한 스토리지 시스템으로 2007년 그가 졸업한 이후에 집중적으로 개발하여 현재에 이르렀다. 세이지 웨일이 2012년 Inktank Storage라는 회사를 설립하였으며, 2014년 레드햇에 팔렸으나, 여전히 오픈소스 프로젝트로 커뮤니티의 참여 하에 개발이 진행되고 있다. ceph에서 지원하는 스토리지 유형으로 파일 스토리지, 블록 스토리지, 객체 스토리지를 지원하고 있다. 파일 스토리지는 ceph-fuse를 이용하여 사용자가 직접 접근할 수 있으며, libcephfs를 통하여 NFS, CIFS, SMB에 접근할 수 있다. 블록 스토리지는 librados를 통하여 OpenStack의 가상머신이나 운용체제에 블록 스토리지를 지원한다. 객체 스토리지는 아마존의 S3 API나 객체 스토리지인 Swift API와 librgw를 통하여 ceph 스토리지에 접근하여 이용할 수 있다 [10].

2. Chef

Chef 는 Ruby 와 Erlang으로 작성된 구성 관리 도구로 사용자는 Chef가 서버 응용 프로그램 및 유틸리티를 관리하는 방법과 이를 구성 하는 방법을 설명하는 recipes를 작성하여 다양한 리소스를 실행할 특정 소프트웨어 버전으로 구성 할 수 있으며 소프트웨어가 종속성을 기반으로 올바른 순서로 설치되도록 한다. Chef는 클라이언트/서버 모드 또는 chef-solo라는 독립 실행 형 구성으로 실행하여 응용 프로그램을 관리한다. 클라이언트/서버 모드에서 Chef 클라이언트는 Chef 서버 노드에 대한 다양한 속성을 보내 속성을 색인화하고 클라이언트가 이 정보를 조회 할 수 있는 API를 제공한다. Chef는 CFEngine , Ansible 및 Puppet 과 함께 Linux의 주요 구성 관리 시스템 중 하나이며, Puppet 및 Ansible과 함께 가장 많이 사용하는 IAC (Infrastructure as Code) 도구이다 [11].

3. OpenStack

오픈스택(OpenStack)은 클라우드 컴퓨팅 오픈 소스 프로젝트로 프로세싱, 저장 공간, 네트워킹의 가용자원을 제어하는 목적의 여러 개의 하위 프로젝트로 이루어져 있으며, 대시 보드 프로젝트인 Horizon을 이용하여 다른 하위 프로젝트의 운영 제어를 웹 인터페이스를 통해 담당한다. 오픈스택은 다음과 같은 하위 프로젝트로 구성된다. Nova는 컴퓨터 자원의 풀을 관리하고 자동화하도록 설계되어 있으며 베어 메탈, 고성능 컴퓨팅, 가상화 기술을 지원하다. Neutron은 네트워크와 IP 주소들을 관리하기 위한 시스템이다. Cinder는 오픈스택 컴퓨터 인스턴스

에 사용할 지속적인 블록 레벨 스토리지 장치들을 제공한다. Keystone는 클라우드 운영 체제를 통하는 공통 인증 시스템을 지원한다. Glance는 디스크 및 서버 이미지를 위한 검색, 등록, 배급 서비스를 제공한다. Heat는 오픈스택 네이티브 REST API를 통해 여러 개의 복합 클라우드 애플리케이션들을 관리하는 서비스이다. Mistral은 워크플로를 관리하는 서비스이다. Ceilometer는 모든 오픈스택 구성요소를 통해 고객 청구가 필요한 모든 계정을 제공한다. Trove는 관계형 및 비관계형 데이터베이스 엔진을 제공한다. Sahara는 하둡 클러스터를 지원한다. Ironic은 가상 머신 대신 베어 메탈 머신을 이용할 수 있도록 지원한다. Zaqar는 웹 개발자들을 위한 멀티테넌트 클라우드 메시징 서비스이다. Manila는 오픈스택 공유 파일 시스템으로 오픈 API를 제공한다. Designate은 DNS를 관리하는 멀티테넌트 REST API이다. Searchlight는 다양한 오픈스택 클라우드 서비스를 통해 고급 및 일정한 검색 기능을 제공한다. Barbican은 기밀 정보의 스토리지에 보안을 제공하고 준비하고 관리하는 REST API이다 [12].

III. 소프트웨어 정의 스토리지 자동 설정 모듈

본 논문에서 제안한 소프트웨어 정의 스토리지 자동 설정 모듈은 그림1과 같이 ACD(Atomatic Ceph Deploy), FSA(File Storage Agent), BSA(Block Storage Agent), OSA(Object Storage Agent)로 구성된다. ACD는 스토리지 클러스터 서버에 소프트웨어 정의 스토리지인 ceph[10]을 자동으로 설치 및 설정하는 모듈이다. FSA는 클라이언트의 가상머신이나 물리적 머신에 상관없이 ceph의 공유 파일 시스템을 사용할 수 있도록 자동으로 설정하는 모듈이다. BSA는 클라이언트의 가상머신이나 물리적 머신에 상관없이 ceph의 블록 스토리지를 사용자의 머신에 추가하여 사용할 수 있도록 자동으로 설정하는 모듈이다. OSA는 클라이언트의 가상머신이나 물리적 머신에 상관없이 ceph의 객체 스토리지를 사용할 수 있도록 자동으로 설정하는 모듈이다.

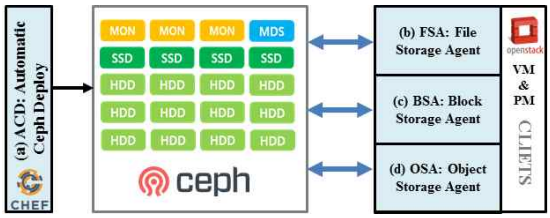

그림 1. 소프트웨어 정의 스토리지 자동 설정 모듈도

1. Automatic Ceph Deploy (ACD) 모듈

그림1(a)의 ACD은 소프트웨어 정의 소토리지(SDS)인 ceph을 자동으로 설치 및 설정하는 모듈로 Chef 도구를 이용하여 구현하였다. Chef[11]는 서버 환경의 메타데이터를 관리하고 노드의 역할 및 상태를 조정하는 운영 프레임워크로 서버 설정이나 갱신을 자동화하는 도구이다. 그림1(a)의 ACD모듈에 Chef를 Server를 설치하여 자동으로 Ceph을 설치하여 클러스터를 구성할 수 있도록 하였다. Chef의 버전은 11.1.6-1로 우분투 리눅스 14.04 LTS 버전의 운영체제를 사용하여 클러스터 노드에 설치하였다. 그림2는 다음의 명령어를 이용하여 node01에서 node03에 Chef 클라이언트 노드를 생성한 상태를 보여준다. 그림3은 Ceph 설치를 위한 관련 패키지의 Cookbooks을 보여준다.

```
$ sudo knife bootstrap node01 -u chef -P chef --sudo
$ sudo knife bootstrap node02 -u chef -P chef --sudo
$ sudo knife bootstrap node03 -u chef -P chef --sudo
```

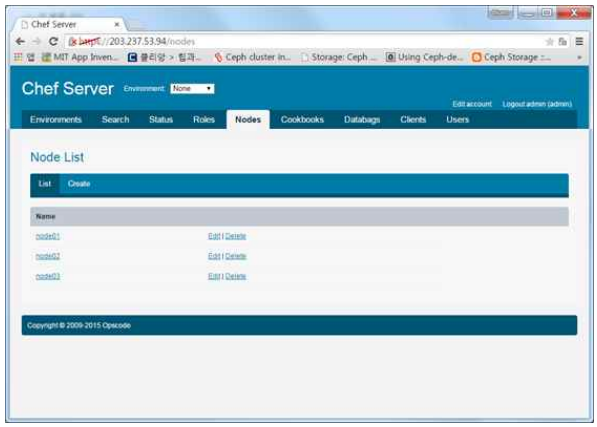

그림 2. Chef Server 및 Chef 클라이언트 노드

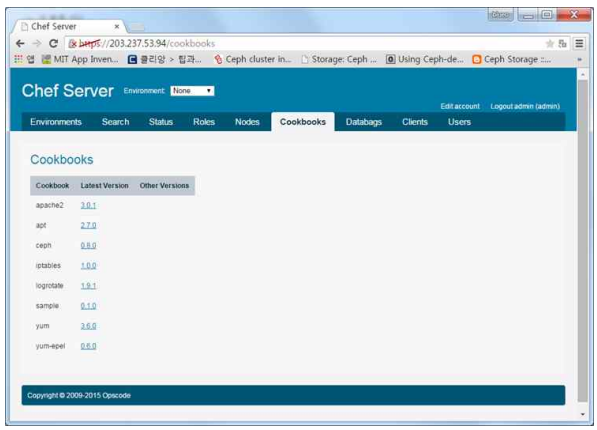

그림 3. Ceph 설치를 위한 관련 패키지의 Cookbooks

Chef를 이용하여 ceph을 자동으로 설치 및 설정하기 위해서는 Chef 서버에 ceph의 cookbook과 role이 있어야 한다. 이를

위하여 다음 그림4와 같이 ceph 설치를 위한 환경 및 설정에 관련된 정보를 구축하여 ACD 모듈에서 자동으로 ceph 클러스터 스토리지를 설치 및 설정할 수 있도록 하였다. Chef 클러스터를 쉽게 설치 및 설정할 수 있도록 ACD 모듈에 Bash 스크립트로 프로그래밍 하여 구성하였다. 다음 그림4는 ACD 모듈의 클라우드 스토리지 ceph의 자동 설치 및 설정에 대한 진행 알고리즘의 순서도를 보여준다.

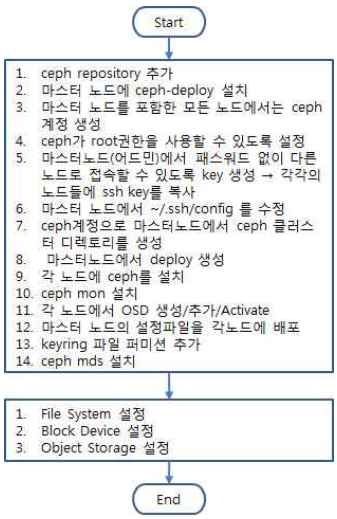

그림 4. ACD 모듈의 ceph 자동설치 및 설정 진행 알고리즘 구성도

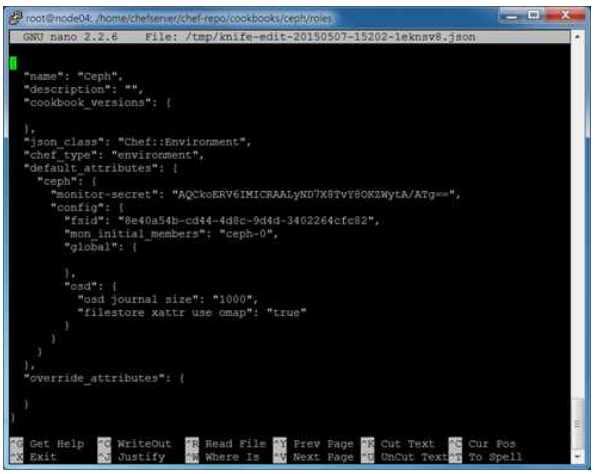

그림 5. Ceph 환경 생성 및 설정

ACD 모듈에는 ceph 스토리지의 성능향상을 위한 튜닝을 포함하여 성능을 향상시켰다. ceph 스토리지가 주로 사용하는 저장 매체로 하드디스크(HDD)를 이용하고 있다. HDD 대신에 SSD (Solid State Disk)를 사용하면 높은 수준의 성능을 향상시킬 수 있다. 그러나 비용문제 때문에 스토리지 시스템들은 주로 HDD가 사용되고 있다. 비용과 성능적인을 부분을 타협하여 SSD와 HDD를 하이브리드로 구성하면 HDD만 사용하는 경우에 비하여 더 좋은 성능향상을 이룰 수 있다. 즉, 그림5와 같이 SSD를 캐쉬로 이용하여 HDD에 접근할 수 있도록 튜닝하면 성능을 높일 수 있으며 소프트웨어 설정에 의하여 구성할 수 있다.

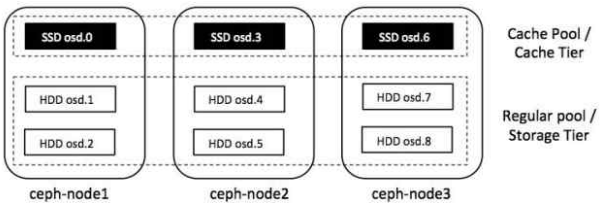

그림 6. Cache Tier를 이용한 Ceph의 성능 향상

2. File Storage Agent (FSA) 모듈

구축된 ceph 스토리지의 공유 파일 스토리지를 사용자들이 이용하기 위해서는 수작업으로 리눅스의 커널 드라이브를 설정하거나 FUSE (File system in User Space)를 설치 및 설정하여 이용해야 한다. 자동으로 사용자들이 공유 파일 스토리지를 사용할 수 있도록 그림1(b)의 FSA 모듈을 Bash 스크립트로 프로그래밍 하였다. 다음 그림7은 FSA 모듈의 파일 스토리지 자동 설치 진행 알고리즘의 순서도를 보여준다.

그림 7. FSA 모듈의 파일스토리지 자동 설정 진행 알고리즘 구성도

자동으로 파일 스토리지를 사용할 수 있도록 사용자의 머신 환경에 연결하는 스크립트는 make-cephfs-v2.sh이며, 만약 파일 스토리지를 필요하지 않으면 해지하는 스크립트는 umount-cephfs.sh이며, 다시 이전의 공유 파일 스토리지가 필요시 기존 설정된 파일 스토리지에 재연결하는 스크립트는 mount-cephfs.sh이다. 다음과 같이 각각의 스크립트를 실행시킬 수 있다.

$ ./make-cephfs-v2.sh // Install ceph-fuse

package and mounting ceph

```
$ ./umount-cephfs.sh        // Umounting CephFS
$ ./mount-cephfs.sh         // Remounting CephFS
```

다음은 자동 파일 스토리지 설정 스크립트의 사용 예를 나타낸다.

```
$ ./make-cephfs-v2.sh
$ cd /mnt/id@cephfs         // root permission
$ sudo mkdir test           // make working directory
$ sudo chown ceph:ceph test
$ sudo ln -s /mnt/id@cephfs/test /home/id/id@cephfs
```

다음 그림8과 그림9는 자동 파일 스토리지 설정 스크립트의 실행과 실행결과를 보여주고 있다. 그림6에서는 make-cephfs.sh를 이용하여 사용자가 ceph 스토리지의 공유 파일을 /home/ceph/cephfs에 마운트 하여 35TB를 사용할 수 있는 것을 보여준다. 그림7에서는 umount-cephfs.sh를 이용하여 35TB의 공유 파일 스토리지를 해지한 것과 mount-cephfs.sh를 이용하여 파일 스토리지를 재공유하는 것을 보여준다.

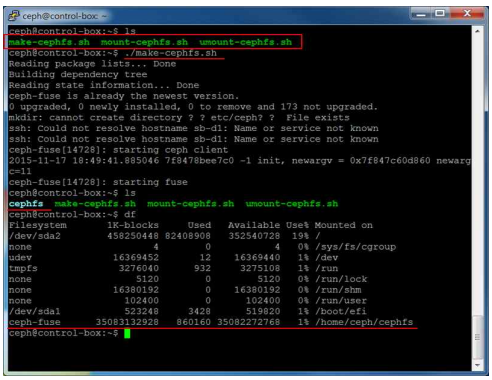

그림 8. 자동 파일 스토리지 연결 스크립트

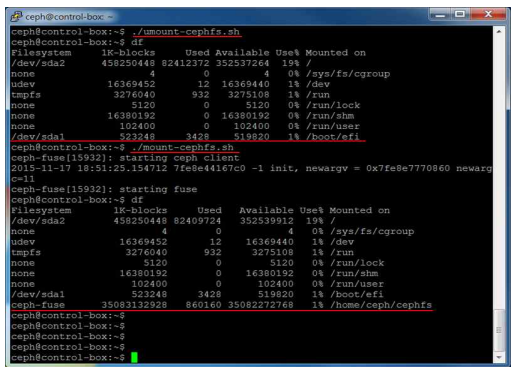

그림 9. 자동 파일 스토리지 설정 스크립트 실행 결과

3. Block Storage Agent (BSA) 모듈

그림1(c)의 BSD 모듈은 사용자의 가상환경이나 물리적인 환경에서 블록 스토리지를 사용할 수 있도록 자동으로 설정한다. 다음 그림10은 BSA 모듈의 블록 스토리지 자동 설치 진행 알고리즘의 순서도를 보여준다.

그림 10. BSA 모듈의 블록 스토리지 자동 설정 진행 알고리즘 구성도

make-cephvolumn-v1.sh는 가상 환경에서 블록 스토리지를 사용할 수 있도록 지원하는 BSD 모듈이며, make-cephvolumn-v2.sh는 물리적인 환경에서 블록 스토리지를 사용할 수 있도록 지원하는 BSD 모듈로 각각 Bash 스크립트로 프로그래밍 하였다. 다음 그림8은 make-cephvolumn-v1.sh를 이용하여 오픈스택의 가상머신에 블록 스토리지를 사용할 수 있도록 데이터 풀을 구성한 것을 보여준다.

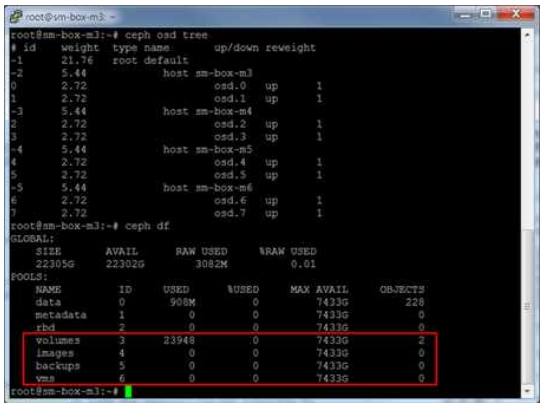

그림 11. 오픈스택의 가상머신에서 블록스토리지를 사용하기 위한 데이터 풀

4. Object Storage Agent (OSA) 모듈

일반적으로 객체 스토리지에 접근하기 위해서는 Raddos gateway의 API를 통하여 RESTful 아마존 S3 호환 API나 RESTful Swift 호환 API 인터페이스를 통하여 웹에서 객체 스토리지에 사용자가 접근할 수 있다. 이 때문에 객체 스토리지를 사용하기 위해서는 그림1(d)의 OSA모듈을 이용하여 ceph 클러스터 노드의 OSD(Object Storage Deamon) 모듈에 아파치 웹서버와 Rados gateway를 설치해야 한다. 이를 위하여 make-cephobject-v1.sh를 이용하여 사용자는 자동으로 ceph 객체 스토리지를 사용할 수 있는 환경은 설정할 수 있도록 Bash 스크립트로 프로그래밍 하였다. 이후 사용자가 아마존의 S3 호환 API를 이용하여 객체 스토리지를 사용하려면 boto 프레임워크를 사용해야 하며, Swift를 통하여 객체 스토리지를 사용하려면 python-swiftclient 패키지를 설치해야 한다. 다음 swift API를 이용하여 웹을 통한 객체 스토리지를 사용하는 예를 보여준다. 다음 그림12는 BSA 모듈의 객체 스토리지 자동 설치 진행 알고리즘의 순서도를 보여준다.

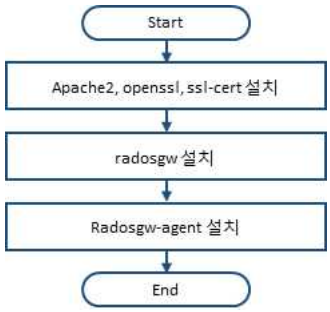

그림 12. OSA 모듈의 객체 스토리지 자동 설정 진행 알고리즘 구성도

$ swift -A http://{IP ADDRESS}/auth/1.0 -U testuser:swift -K '{swift_secret_key}' list

Ⅳ. 스토리지 성능튜닝 비교평가

다음 표1은 ACD 모듈을 이용한 성능튜닝을 위하여 PC의 하드웨어 구성 및 각 노드들에 대한 ceph 데몬 구성요소를 보여준다. 표1의 환경1(E1)인 경우 운영체제 및 ceph mon (monitor) 데몬을 1TB의 HDD에 설치하여 운영하였으며, OSD를 120GB와 256GB의 SSD를 이용하여 구성하였다. 이중 120GB의 SSD는 캐시 계층으로 사용하였으며, 256GB의 SSD는 저장 계층으로 사용하였다. 다음 그림13은 PC를 이용하여 ceph 클러스터를 구축한 실험환경을 보여준다. 그림14는 E1 Ceph 클러스터 환경에 Ceph에서 제공하는 벤치마크 도구인 Rados Bench[10]를 이용하여 벤치마크 테스트 100번 수행하여 평균을 낸 결과이다. 그림에서 D2는 Public Network를 나타내며, S는 Cluster Network를 나타낸다. 적갈색은 실제 데이터가 저장되는 OSD를 SSD로만 구성하고 캐시 계층을 적용하지 않은 것이고, 녹색은 OSD를 SSD와 캐시 계층을 적용한 결과를 나타낸다. OSD를 SSD로 구성하는 경우에는 SSD로 다시 캐시로 구성하는 것보다 성능이 높은 것을 볼 수 있다. 이것은 SSD에 직접 데이터를 저장하는 것에 비하여 SSD 캐시 계층을 통하여 다시 SSD에 저장하는 것이 저장 성능이 감소하는 것을 알 수 있다. 데이터의 순차읽기와 랜덤읽기 역시 마찬가지로 성능이 감소하는 것 알 수 있다.

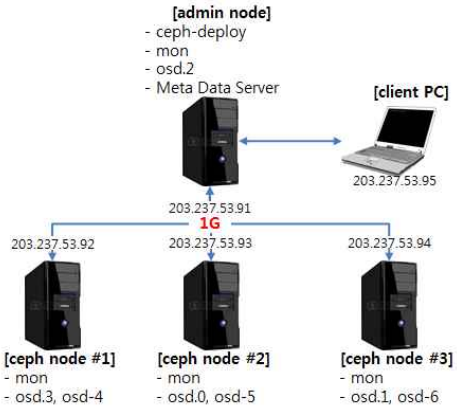

그림 13. PC를 이용한 Ceph 클러스터 구성도

표 1. Ceph Cluster Environment 1 (E1) for Performance Turning

node	HDD-1T	SSD-120G (cache)	SSD-256G
node01	OS/mon.1	osd.0	osd.1
node02	OS/mon.2	osd.2	osd.3
node03	OS/mon.3	osd.4	osd.5

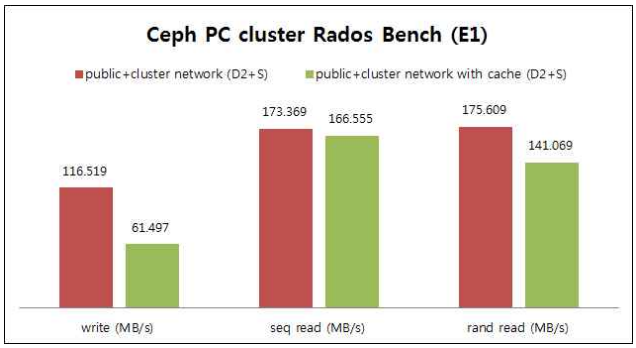

그림 14. Ceph PC 클러스터 환경1의 Rados 벤치마크 결과

표2의 환경2(E2)인 경우 운영체제 및 mon 데몬을 256GB의 SSD에 설치하여 운영하며, OSD 데몬을 120GB와 256GB의

SSD를 이용하여 구성하였다. 이중 120GB의 SSD는 캐시 계층으로 사용하였으며, 1TB의 HDDD는 저장 계층으로 사용하였다. 다음 그림은 E2 Ceph 클러스터 환경에 Ceph에서 제공하는 벤치마크 도구인 Rados Bench를 이용하여 벤치마크 테스트 100번 수행하여 평균을 낸 결과이다. 그림15에서 D2는 Public Network를 나타내며, S는 Cluster Network를 나타낸다. 적갈색은 캐시 계층을 적용하지 않은 것이고, 녹색은 캐시 계층을 적용한 결과를 나타낸다. OSD를 HDD와 SSD로 구성하는 경우에는 SSD로 캐시를 구성하는 것이 저장시 성능 높아지는 것을 볼 수 있다. 이것은 HDD에 직접 데이터를 저장하는 것에 비하여 SSD 캐시 계층을 통하여 다시 SSD에 저장하는 것이 저장 성능이 높아지는 것을 알 수 있다. 그러나 SSD 캐시를 사용하여 데이터를 읽는 경우에는 한계층을 더 거쳐서 읽기 때문에 데이터의 순차읽기와 랜덤읽기 시 성능이 감소하는 것 알 수 있다.

표 2. Ceph Cluster Environment 2 (E2) for Performance Turning

node	SSD-256G	HDD-1T	SSD-120G (cache)
node01	OS/mon.1	osd.0	osd.1
node02	OS/mon.2	osd.2	osd.3
node03	OS/mon.3	osd.4	osd.5

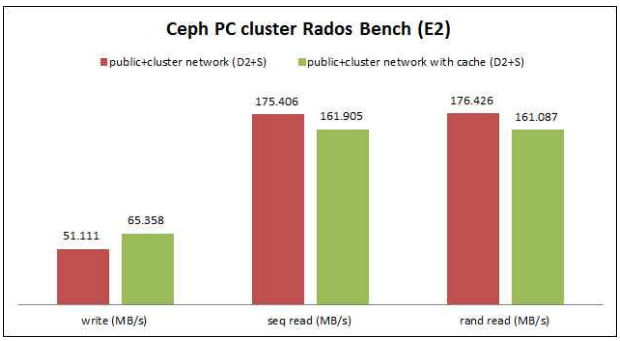

그림 15. Ceph PC 클러스터 환경2의 Rados 벤치마크 결과

V. 결 론

본 논문에서는 세계적인 마이크로 클라우드 스토리지의 자동화 추세에 맞추어 사용자의 환경에 연계한 블록/파일/객체의 통합 SDS 자동 설정 모듈을 제안하였다. 제안방법은 템플릿 기반 자동화 도구를 클라우드 스토리지로 확장하여 자동 설정/관리를 위한 방법론을 적용하였다. 개발 모듈은 하드웨어 종속 없이 가상이나 물리적인 사용자 환경에 맞춰서 자동으로 공유 파일 스토리지, 블록 스토리지, 객체 스토리지에 대한 설정 및 관리를 쉽게 할 수 있도록 지원한다.

REFERENCES

[1] 김종원, "오픈소스 융합형 화이트 박스에 기반한 소프트웨어-정의 인프라 환경," *한국컴퓨터통신연구회*, 제28권, 제1호, 26-35쪽, 2015년 3월
[2] 차병래, 최명수, 박선, 김종원, "Software-Defined RAID 기반 장애복구 기법과 실증 테스트," *스마트미디어저널*, 제5권, 제1호, 69-77쪽, 2016년 3월
[3] 차병래, 차윤석, 최명수, 박선, 김종원, "대용량 Abyss Storage의 KOREN 네트워크 기반 국내 및 해외 실증 테스트," *스마트미디어저널*, 제6권, 제1호, 9-15쪽, 2017년 3월
[4] 차병래, 박선, 신병춘, 김종원, "Abyss Storage Cluster 기반의 DataLake Framework의 설계," *스마트미디어저널*, 제7권, 제1호, 9-15쪽, 2018년 3월
[5] 클라우드 파일 스토리지, https://aws.amazon.com/ko/what-is-cloud-file-storage/ (accessed Aug., 4, 2018).
[6] 블록 스토리지 및 객체 스토리지, http://brownbears.tistory.com/258 (accessed Aug. 15, 2018).
[7] DELL EMC Software-Defined Storage, https://www.dellemc.com/ko-kr/storage/data-storage/software-defined-storage.htm#compare0=0 (accessed Aug., 15, 2018).
[8] HPE StoreVirtual VSA Software, https://h20195.www2.hpe.com/v2/GetPDF.aspx%2Fc04111621.pdf (accessed Aug., 16 2018).
[9] 뉴타닉스, http://www.virtual-space.co.kr/nutanix-works.html (accessed Aug., 16, 2018).
[10] ceph, https://ceph.com/ (accessed Aug., 18, 2018).
[11] Chef, https://blog.chef.io/ (accessed Aug., 19 2018).
[12] OpenStack, https://ko.wikipedia.org/wiki/%EC%98%A4%ED%94%88%EC%8A%A4%ED%83%9D (accessed Aug., 20, 2018).

저 자 소 개

박 선

2007년 인하대학교 컴퓨터정보공학과 공학박사
2008년 호남대학교 컴퓨터공학과 전임강사
2010년 전북대학교 인력양성사업단 박사후 과정
2010년 목포대학교 정보산업연구소 연구전임교수
2013년 ~ 현재 광주과학기술원 연구조교수
2017년 ~ 현재 제노테크(주) 연구소장
<주관심분야 : 정보검색, 데이터마이닝, 해양IT정보융합, 클라우드 컴퓨팅, IoT, 스토리지 시스템>

차병래

2004년 목포대학교 대학원 컴퓨터공학과 졸업(공학박사)
2005년 호남대학교 컴퓨터공학과 전임강사
2009년 ~ 현재 광주과학기술원 정보통신공학부 연구조교수
2012년 ~ 현재 제노테크(주) 대표
<주관심분야 : 정보보안, IDS, Neural Network, Cloud Computing, VoIP, NFC 등>

김종원

1997년 University of Southern California 연구 조교수
1999년 Technology Consultant for VProtect Systems Inc.
2000년 Technology Consultant for Southern California Division of InterVideo Inc.
2001년 광주과학기술원 정보기전공학부 부교수
2008년 ~ 현재 광주과학기술원 정보통신공학부 교수
<주관심분야 : Future Internet, SDN & NFV, SDI>