ETRI

2013
Electronics and
Telecommunications
Trends

OpenStack 클라우드 네트워크 기술 분석

Analysis of OpenStack Cloud Networking Technology

박종근 (J.G. Park) 스마트노드플랫폼연구실 선임연구원
최강일 (G.I. Choi) 스마트노드플랫폼연구실 선임연구원
이상민 (S.M. Lee) 스마트노드플랫폼연구실 책임연구원
이정희 (J.H. Lee) 스마트노드플랫폼연구실 책임연구원
이범철 (B.C. Lee) 스마트노드플랫폼연구실 실장

* 본 연구는 미래창조과학부가 지원한 2013년 정보통신·방송(ICT) 연구개발사업의 연구결과로 수행되었음.

오늘날 클라우드 서비스가 확산되면서 공용 클라우드 뿐만 아니라 사설 클라우드나 하이브리드 클라우드에 대한 관심도 높아지고 있다. 특히 다양한 규모와 형태의 클라우드 인프라를 손쉽게 구축하고 관리할 수 있는 클라우드 운영체제에 대한 관심은 최근 급증하고 있는 추세이다. 또한 정보통신 인프라에서 네트워크가 차지하는 비중만큼이나, 클라우드 서비스에서 멀티테넌시 네트워크 구성, 다양한 네트워크 하부 기술의 수용, 가상 네트워크 인프라의 동적 프로비저닝 등을 포함하는 클라우드 네트워크 기술에 대한 관심과 기술개발 노력이 더욱 집중되고 있다. 본고에서는 현재 가장 주목받고 있는 공개 소프트웨어 기반의 클라우드 운영체제인 OpenStack의 구조와 클라우드 네트워크 기술에 대한 주요 특징을 정리한다.

Ⅰ. 서론

정보통신 기술에 대한 비즈니스 의존도가 날로 높아지면서 기업은 정보통신 인프라 구축에 꾸준히 투자해 왔지만, 한편으로 이기종의 복잡한 정보통신 인프라를 유지하고 운용하기 위한 총소유비용(Total Cost of Ownership)의 증가도 불가피했다. 때마침 1960년대에 IBM이 처음으로 개발하였던 가상화 기술이 Xen 하이퍼바이저와 VMware를 중심으로 다시금 주목을 받게 되면서, 결국 기업의 정보통신 인프라에 대한 총소유비용 부담을 줄이고 새로운 서비스 생태계를 창출하려는 노력이 클라우드 컴퓨팅 서비스의 확산으로 이어졌다.

클라우드 컴퓨팅 서비스는 새로운 정보통신 서비스(WHAT)라기 보다는 정보통신 서비스를 제공하는 방식(HOW)이 변화된 것이라 볼 수 있다. 즉, 사용자는 과거에도 호스팅 서비스나 인프라 아웃소싱 서비스 등을 통해서 직접 인프라를 구축하지 않고 서비스 제공자로부터 일정 부분 임대하여 이용하기도 하였다. 다만 클라우드 컴퓨팅 서비스의 도입으로 기존의 서비스가 좀더 유연하고 효율적으로 제공되면서 새로운 비즈니스 모델이나 생태계가 창출되고 있기도 하다.

초기에는 Amazon, Google, Dropbox 등과 같은 공용 클라우드(public cloud) 서비스의 확산과 함께 기업들은 보안 등의 이유로 사설 클라우드(private cloud) 구축에 더 많은 관심을 가져왔다. 그러나, 최근에는 훨씬 더 통제가 필요한 데이터나 워크로드는 사설 클라우드에 두고, 서비스 이용이나 데이터 손실 부담이 적은 것은 공용 클라우드를 이용하는 하이브리드 클라우드(hybrid cloud) 서비스가 점차 주목을 받고 있다.

클라우드 서비스가 더욱 보편화된 서비스로 자리매김하기 위해서는 여러 워크로드와 사용자 그룹별로 가상화된 인프라 자원을 동적으로 프로비저닝하고, 워크로드의 상태나 전체 인프라 자원의 상태 정보를 바탕으로 워크로드의 자동 최적 배치 및 가상 인프라의 생명주기가 체계적으로 관리될 수 있어야 한다. 또한 이러한 기능을 제공하는 클라우드 운영체제 및 시스템관리 소프트웨어의 개발이나 기술 고도화도 반드시 뒷받침되어야 한다. 이에 VMware나 Citrix와 같은 상용 솔루션뿐만 아니라, 다양한 규모와 형태의 클라우드 인프라를 손쉽게 구축하고 관리할 수 있는 클라우드 플랫폼으로써 OpenStack, CloudStack, Eucalyptus 등 공개 소프트웨어 기반의 클라우드 운영체제도 많은 주목을 받고 있다.

본고에서는 공개 소프트웨어 기반의 클라우드 운영체제 중에서 최근 개발이 활발히 진행되고 있는 OpenStack의 Grizzly 배포판을 중심으로 그 구조와 클라우드 네트워크 기술의 주요 특징을 살펴보기로 한다.

Ⅱ. OpenStack 개요

OpenStack은 지난 2010년 7월 Rackspace와 NASA의 주도로 시작된 공개 소프트웨어 프로젝트로써, 다양한 규모와 형태의 클라우드 인프라를 손쉽게 구축할 수 있는 클라우드 플랫폼 프로젝트이다[1]. 컴퓨팅, 저장장치, 네트워크 자원으로 구성된 대규모 자원 풀(pool)을 제어하고 관리하는 클라우드 운영체제로써, 웹 기반의 사용자 인터페이스를 통해 관리자는 손쉽게 자원을 제어하고 서비스를 관리할 수 있으며, 서비스 이용자는 가상 인프라 서비스(Infrastructure as a Service)를 동적으로 제공받고 관리할 수 있다.

리눅스 이후 가장 주목 받는 공개 소프트웨어 프로젝트 중 하나인 OpenStack은 컴퓨터, 저장장치, 네트워크 장비 제조사와 소프트웨어 솔루션 업체 그리고 정보통신 서비스 사업자를 아우르는 수많은 정보통신 글로벌 선도 기업들이 회원사로 참여하고 있으며, 사용자 층도 전세계적으로 지속적으로 확대되고 있다.

1. OpenStack 개발 프로세스

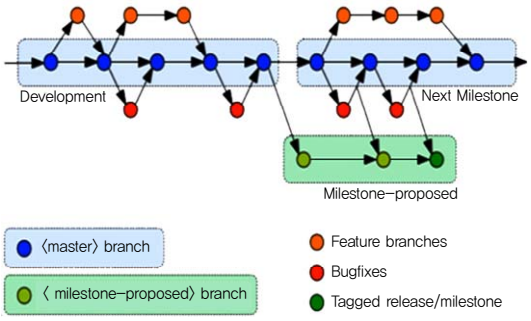
지난 2010년 Austin 배포판을 시작으로 알파벳 순서

(그림 1) OpenStack 배포판 이력

에 따라 공식 발표되던 OpenStack은 Diablo 배포판 이후 매 6개월마다 새로운 배포판을 공식적으로 발표하고 있으며, 2013년 7월 현재까지 지난 4월 발표된 Grizzly 배포판이 최종 공식 배포판이며, 오는 10월에는 Havana 배포판이 새롭게 발표될 예정이다((그림 1) 참조).

OpenStack의 배포주기는 크게 계획, 개발, 사전배포, 공식배포의 4단계로 이루어진다. 매번 새로운 배포판이 공식 발표된 이후에는 차기 배포판에 대한 개발 논의가 Design Summit을 포함해 약 4주간 지속되며, 이 기간 동안 수집된 각종 기능 추가 또는 성능 개선 현안들에 대한 검토를 거쳐 우선순위를 부여하고 전체적인 개발일정 계획을 수립한다.

그 다음에는 (그림 2)와 같이, 수립된 일정 계획에 따라 실제 선별된 기능 추가 또는 성능 개선 현안에 대한 개발이 동시다발적으로 진행되며, 개발 완료된 소스코드는 검토 및 검증 과정을 거쳐 기본 소스코드인 master branch에 병합된다. 새로운 배포판에 대한 개발 일정이 완료되면, 그 시점에서 master branch를 분기시켜 배포를 위한 milestone-proposed branch를 생성한다.



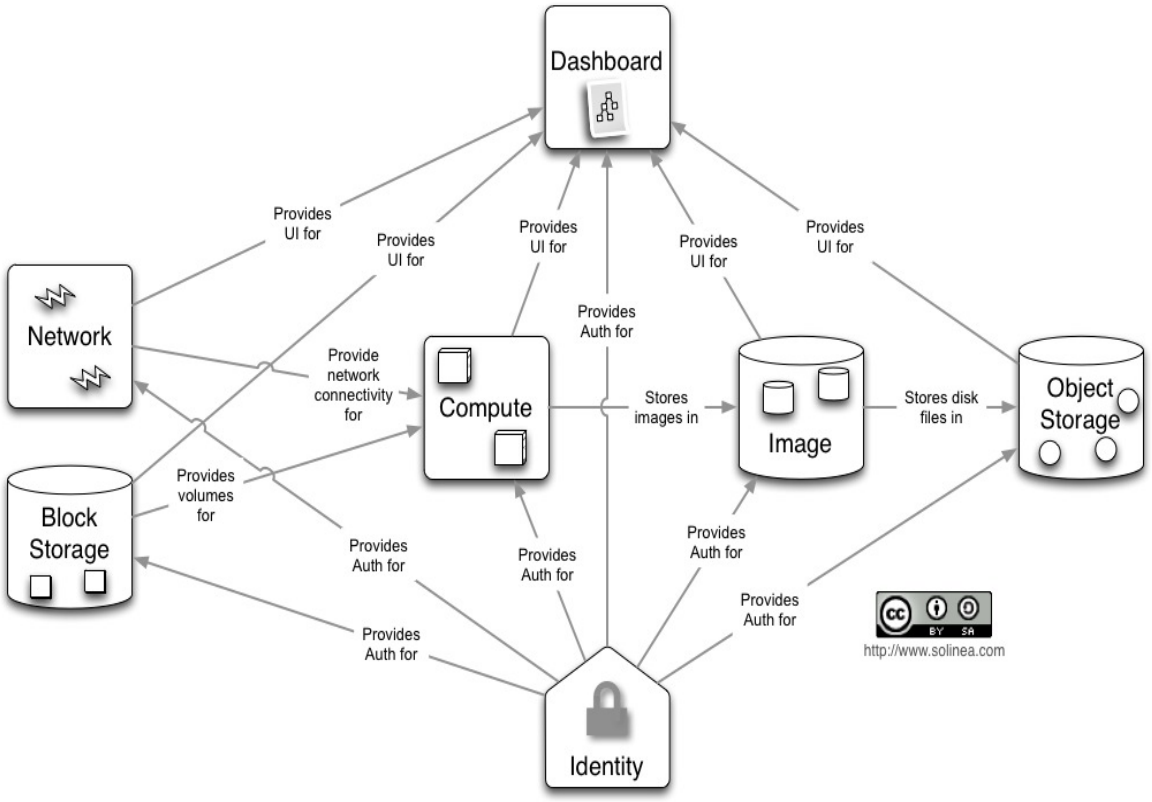
(그림 2) OpenStack 개발 모델

Milestone-proposed branch에는 새로운 기능 추가 또는 성능 개선 현안은 더 이상 반영되지 않으며, 발견된 오류의 해결에 집중하게 되고 해결된 오류는 master branch에도 반영된다. 이러한 과정을 통해 새로운 배포판의 발간과 관련된 주요 오류를 해결한 버전을 배포 후보판(release candidate)으로 생성하며, 더 이상 새로운 오류가 해결되지 않는다면 새로운 배포판의 공식 발표 예정 시점에서의 최종 배포 후보판이 공식 배포판으로 발표된다.

2. OpenStack 구조 및 구성

OpenStack은 클라우드 운영체제로써의 주요 기능이 각각 독립된 하부 프로젝트로 나뉘어 개발되고 있다. 초기에는 컴퓨팅 서비스인 Nova, 오브젝트 저장장치 서비스인 Swift, 이미지 관리 서비스인 Glance의 세가지 하부 프로젝트로 시작되었으나, 배포판 버전이 업데이트 되면서 새로운 프로젝트가 지속적으로 추가 발표되어 점차 그 영역을 확장해 나가고 있다. 일례로 네트워킹 서비스와 볼륨 저장장치 서비스와 같이 Nova에서 함께 제공되던 서비스가 별도의 프로젝트로 분리되거나 인증 서비스 또는 사용자 인터페이스와 같이 독립된 프로젝트가 새롭게 추가되기도 한다.

(그림 3)은 Grizzly 배포판을 기준으로 OpenStack의 공식 하부 프로젝트간의 상호관계를 도식화 한 것이다 [2]. OpenStack의 여러 하부 프로젝트는 기본적으로 가상머신을 생성하고 제어하는 Nova 프로젝트를 중심으로 상호 연계되어 있다. 가상머신에 대한 네트워킹 서비스는 Neutron 프로젝트를, 가상머신에서 사용하는 추가적인 볼륨 저장장치는 Cinder 프로젝트를 이용하여 서비스를 제공한다. 가상머신 이미지는 Glance 프로젝트를 통해 관리되고, Swift 프로젝트를 통해 저장할 수 있다. 또한 기본적으로 사용자가 OpenStack 서비스를 이용하기 위한 인터페이스인 대쉬보드(dashboard)는



(그림 3) OpenStack 하부 프로젝트 관계 구조도

Horizon 프로젝트로써, 모든 하부 프로젝트의 인터페이스가 대쉬보드를 통해 사용자에게 제공되며, 서비스를 이용하고자 하는 사용자를 인증하고 권한을 검증하는 서비스는 Keystone 프로젝트를 통해 제공한다.

가. 컴퓨팅 서비스 : Nova

Nova 프로젝트는 사용자의 요청에 따라 컴퓨팅 자원을 제공하고 관리한다. OpenStack을 통한 클라우드 서비스 제공에 있어 가장 핵심적인 프로젝트로써, 가상머신의 생성, 갱신, 삭제, 생명주기 관리 기능을 제공한다. Essex 배포판까지는 네트워킹 서비스를 제공하는 nova-network와 볼륨 저장장치 서비스를 제공하는 nova-volume이 Nova 프로젝트에 포함되어 있었으나, Folsom 배포판부터 각각 Quantum과 Cinder 프로젝트로 분리되었다.

나. 네트워킹 서비스 : Neutron

Grizzly 배포판까지 Quantum으로 널리 알려진 네트워킹 서비스는 최근 상표권 문제로 인해 프로젝트 이름을 Neutron으로 공식 변경하였다. Neutron 프로젝트의 특징은 다양한 네트워킹 관련 기술 및 장비를 지원하기 위해 플러그인(plugin) 방식을 채택하고 있으며, 소프트웨어 기반의 네트워킹 서비스를 제공하는 것이다.

다. 인증 서비스 : Keystone

Keystone 프로젝트는 사용자 및 API(Application Programming Interface)에 대한 인증 및 권한설정 서비스를 제공하는 것으로서, 사용자에 대한 인증 서비스, 인증된 사용자에 대한 토큰 검증 및 관리 서비스, 이용 가능한 서비스 정보를 제공하는 카탈로그 서비스, 그리

고 규칙 기반의 권한설정 기능을 제공하는 정책 서비스 등으로 구성된다.

라. 이미지 관리 서비스 : Glance

Glance 프로젝트는 가상머신 이미지 관리 서비스로서, Swift 프로젝트의 오브젝트 저장장치를 포함한 가상머신 이미지 저장소로부터 가상머신 이미지를 탐색, 등록 및 추출하는 기능을 제공한다. 따라서, Nova를 통해 생성되는 가상머신은 Glance를 통해 가상머신 이미지를 제공받고 관리한다.

마. 볼륨 저장장치 서비스 : Cinder

Cinder 프로젝트는 범용의 저장장치를 기반으로 디스크 볼륨을 제공하는 서비스로서, IBM, EMC, HP, RedHat/Gluster, Ceph/RBD, NetApp, SolidFire, Nexenta 등과 같은 저장장치 시스템을 위한 드라이버도 지원한다. Cinder 프로젝트에 의해 생성된 가상 디스크 볼륨은 가상머신에 마운트되어 가상머신 사용자의 데이터 등을 저장하는데 사용될 수 있다.

바. 오브젝트 저장장치 서비스 : Swift

Swift 프로젝트는 대용량 데이터를 저장할 수 있는 고확장성의 오브젝트 저장장치를 제공하는 서비스로써, 가상머신 이미지 파일의 저장장치로 사용되기도 한다.

사. 사용자 인터페이스 : Horizon

Horizon 프로젝트는 시스템 관리자 및 서비스 사용자를 위해 권한별로 OpenStack 서비스를 이용할 수 있는 웹 기반의 사용자 인터페이스 환경을 제공한다.

아. 기타 프로젝트

현재까지 공식 배포판에 포함된 상기 프로젝트 외에도 다양한 실험적 프로젝트가 진행 중에 있다. 일례로 과금을 위한 사용자의 서비스 사용 정보를 계측하고 모니터링하기 위한 Ceilometer 프로젝트와, 템플릿 기반으로 클라우드 서비스를 제공하기 위해 필요한 다양한 자원을 조율(orchestration)하여 제공하는 Heat 프로젝트를 꼽을 수 있다. 이 두 프로젝트는 이미 실험적으로 Grizzly 배포판에 포함되었으며, Hanava 배포판부터는 공식 OpenStack 프로젝트에 포함될 것으로 기대된다.

Ⅲ. OpenStack 네트워크 서비스 기술

OpenStack은 클라우드 컴퓨팅 환경에서 가상머신에 대한 멀티테넌트(multi-tenant) 네트워크 연결 서비스뿐만 아니라 방화벽, 부하분산 등과 같은 네트워크 기반의 다양한 부가 서비스를 제공한다. OpenStack의 초기 배포판에서는 Nova 프로젝트의 nova-network을 통해 사용자 또는 사용자 그룹에게 기본적인 L2 가상 네트워크를 제공해 왔다. 그러나, 여러 가지 기술적 제약사항과 확장 가능성을 고려하여 독립된 네트워킹 서비스의 필요성이 제기되었고, Diablo 배포판부터 Quantum이라는 코드명의 네트워킹 서비스 프로젝트가 실험적으로 배포되었으며, Folsom 배포판부터 공식 프로젝트로 포함되었다. 아울러 오는 10월 발표될 예정인 Havana 배포판부터는 Quantum 대신 Neutron이라는 새로운 코드명으로 배포될 예정이다. 본고에서는 OpenStack 네트워크 서비스 프로젝트의 코드명으로써 Quantum과 Neutron을 혼용해서 사용하기로 한다.

1. Neutron 프로젝트 출현 배경

Neutron 프로젝트의 출현 배경은 초기 네트워크 서비스를 제공한 Nova 프로젝트의 nova-network이 갖는 다음과 같은 제약사항으로 귀결된다[3][4].

가. 기술적 제약

클라우드 컴퓨팅 서비스가 확산되면서 초기의 클라우드 컴퓨팅 서비스에 비해 네트워크에 대한 중요성이 점차 부각되기 시작하였다. 이러한 추세에 따라 다양한 요구가 도출되었는데, 대표적인 것들을 꼽으면 다음과 같다.

- 멀티테넌시 지원
- 동적인 네트워크 프로비저닝
- 워크로드의 자유로운 배치 및 이전

먼저 데이터센터 내에서 사용자 또는 사용자 그룹인 테넌트(tenant)가 생성한 가상의 네트워크는 마치 독립된 네트워크인 것처럼 다른 테넌트가 생성한 가상 네트워크와 논리적으로 구분되고 상호 격리되어야 한다. 이러한 멀티테넌트 네트워크 환경은 VLAN(Virtual Local Area Network) 기술, 터널링 기술, 그리고 MPLS(Multi-Protocol Label Switching) 기술 등을 통해 제공될 수 있다. Nova 프로젝트의 nova-network에서도 VLAN 기술을 통해 최대 4094개의 멀티테넌트 네트워크를 구성할 수 있는 환경을 제공해 왔다. 그러나, 데이터센터의 규모가 점차 확대되고 가상머신의 집적도가 증가함에 따라, 단일 클라우드 인프라 내에서 더욱 많은 테넌트 네트워크를 구성할 수 있는 환경이 요구되고 있다.

또한, 클라우드 컴퓨팅 서비스에서 워크로드의 수요에 맞춰 동적으로 자원을 제공하는 기능은 클라우드 인프라 자원을 효율적으로 사용하고 서비스의 품질을 제고하기 위해 실현되어야 할 기술 중의 하나이다. 그러나, 기존 네트워크 환경에서는 가상머신이 생성되는 서버 내의 가상 스위치 또는 브릿지를 제어할 수는 있었지만, 완전한 가상의 테넌트 네트워크를 구성하기 위해 관리자가 일일이 수작업으로 네트워크 장비를 설정해야 한다. 따라서, 동적인 네트워크 프로비저닝을 위해서는 테넌트 네트워크를 위한 자원을 할당하고 설정하는 일련의 프로세스가 자동화될 수 있어야 한다.

더욱이 클라우드 컴퓨팅 환경에서 제공되는 서비스의 품질을 보장하기 위해서는 필요에 따라 가상머신을 여유자원이 충분한 서버에 생성하거나 또는 이전(migration)할 수 있어야 하며, 서비스의 연속성을 위해서는 가상머신이 생성되거나 이전되는 서버와 무관하게 IP(Internet Protocol) 주소 등의 네트워크 상태가 그대로 유지되어야 한다.

나. 테넌트 제어 한계

클라우드 컴퓨팅 서비스는 새로운 정보통신 인프라 환경의 구축이라기 보다는, 기존의 인프라 환경을 보다 유연하고 효율적인 방식으로 활용하고자 하는 것이다. 따라서, 기업 등에서 클라우드 컴퓨팅 환경을 도입할 때에는 이미 운용 중이던 복잡한 인프라를 가급적 그대로 복제하기를 원한다. 이를 위해서는 다음과 같은 요구사항들이 충족될 수 있어야 한다.

- 다계층(multi-tier) 네트워크 생성
- IP 주소 관리
- 네트워크 서비스 장비의 추가 및 설정
- 원격의 서버 또는 장비와의 VPN(Virtual Private Network) 연결

일반적으로 웹 서비스를 제공할 때, 부하분산을 통한 성능 향상은 물론 보안성을 높이기 위해 웹 서버, 응용 서버, 데이터베이스 서버를 각각 분리하여 다계층으로 구성할 수 있다. 만일 임의의 테넌트가 다계층 구조의 네트워크를 생성하고자 할 때에는 격리된 L2 가상 네트워크의 제공은 물론, 허용된 IP 주소를 가진 가상머신의 허용된 포트를 통해서만 상호 접근할 수 있도록 제어할 수 있어야 하며, 필요한 경우 데이터센터 내에서 동일한 IP 주소를 중복해서 사용하도록 제어할 수 있어야 한다.

또한 방화벽, L4 또는 L7 기반의 부하분산 장비, 침입탐지시스템, 침입차단시스템 등 다양한 네트워크 서비

스 장비를 테넌트별로 손쉽게 추가하고 제어할 수 있어야 완전한 가상 네트워크 환경을 구축할 수 있다.

더 나아가 클라우드 인프라 밖의 원격 서버나 장비와의 VPN 연결 등을 통해 안전한 가상의 사설 네트워크 환경이 구축될 수 있어야 한다.

요컨대, Neutron 프로젝트는 이상과 같은 종전의 기술적 제약사항을 해결하고 테넌트별 다양한 네트워크 서비스의 제어를 통해 완전한 멀티테넌트 네트워크 환경을 제공하는 것을 목표로 한다. 이를 위해 필요에 따라 다양한 네트워크 기술이 클라우드 환경에서 사용될 수 있도록 확장 가능한 플러그인 기반의 네트워크 제어 구조를 채택하고 있다.

2. OpenStack 배포판별 Neutron 특징

본 절에서는 Neutron 프로젝트가 Quantum 코드명으로 처음 소개된 Diablo 배포판부터 최근의 Grizzly 배포판까지의 주요 특징들을 살펴보고, Grizzly 배포판의 제약사항에 대해서도 함께 기술한다.

가. Diablo 배포판의 특징

2011년 9월, OpenStack의 네 번째 공식 배포판인 Diablo 배포판에 처음으로 Quantum 프로젝트가 실험적으로 포함되기 시작하였다.

주요 특징으로는 먼저 L2 네트워크와 포트의 생성, 갱신, 삭제 및 정보 추출은 물론 생성된 포트와 가상머신 인터페이스의 상호 연결을 제어할 수 있는 Quantum REST(Representational State Transfer) API 1.0이 정의되었다. 또한 Nova API를 통해서도 Quantum 서비스를 이용할 수 있도록 QuantumManager가 포함되었으며, Quantum의 가장 큰 특징 중 하나인 플러그인으로는 Open vSwitch와 Cisco UCS/Nexus 플러그인이 이때 함께 배포되었다[5].

나. Essex 배포판의 특징

Diablo 배포판에 이어 지난 2012년 4월에 발표된 Essex 배포판에서도 Quantum은 인큐베이티드 프로젝트로서 공식 프로젝트로 포함되진 않았지만, DevStack과의 통합은 물론 Ubuntu 12.04, Fedora 17, 그리고 Debian 등의 리눅스 배포판에 함께 패키징되었다. 그리고 Linux Bridge, Nicira Network Virtualization Platform (NVP), Ryu OpenFlow Controller 플러그인이 새롭게 추가되었다[3].

다. Folsom 배포판의 특징

OpenStack 프로젝트에 Quantum이 공식 프로젝트로서 등장한 것은 2012년 9월의 Folsom 배포판이다. 이때 볼륨 저장장치 서비스를 제공하는 Cinder 프로젝트도 Quantum과 함께 공식 프로젝트로 포함되었다.

Essex 배포판까지와는 달리, Folsom 배포판에서는 다양한 네트워크 서비스와 제어 기능이 추가되었다. 가장 눈에 띄는 사항으로는 REST API가 2.0으로 대폭 변경되면서, 종전의 가상 L2 네트워크 및 포트 제어뿐만 아니라 L3 서브넷의 생성, 갱신, 삭제 및 생성 정보를 추출할 수 있는 API가 새롭게 추가되었다. 또한 nova-network의 제약사항 중 하나인 IP 주소관리와 관련하여 서로 다른 L2 네트워크에서는 DHCP(Dynamic Host Configuration Protocol) 에이전트를 통해 부여되는 Fixed IP 주소를 중복하여 사용할 수 있게 되었으며, 확장 기능으로서 프로바이더 네트워크(provider network) 뿐만 아니라 복수의 가상 라우터 생성, 송신측 주소변환(SNAT: Source Network Address Translation), Floating IP 관리 등 L3 네트워크 제어를 위한 확장 API가 정의되었다. 그리고, 인증 서비스를 제공하는 Keystone과의 통합을 통해 일반 클라우드 서비스 이용자 또는 관리자와 같은 사용자의 유형에 따라 API를 이용할 수 있는 권한을 달리 제공한다[6].

라. Grizzly 배포판의 특징

가장 최근에 발표된 Grizzly 배포판에서는 이전 Folsom 배포판의 버그 수정과 더불어 새로운 기능이 추가되었는데, 이 중 가장 대표적인 것은 네트워크 노드의 멀티 호스트 기능과 부하분산 서비스 기능이다.

먼저 멀티 호스트 기능은 OpenStack을 통한 클라우드 인프라를 구축할 때 복수의 네트워크 노드를 배치하여 운용하는 것이다. 이때 네트워크 노드란, L3 에이전트 및 DHCP 에이전트 등과 같은 네트워크 제어 에이전트가 설치된 노드로써, 하나의 네트워크 노드만 구성된 싱글 호스트의 경우 단일 장애점(single point of failure)으로서 네트워크 노드에서 장애가 발생할 경우 전체 네트워크 서비스가 중단되며 외부망과의 네트워크 단절이 초래될 위험을 갖고 있다. 따라서, 멀티 호스트 기능을 통해 네트워크 서비스의 장애 위험을 최소화하여 서비스 가용성을 제고할 수 있다.

부하분산 기능은 공개 소프트웨어인 HAProxy[7]를 기반으로 클라우드 네트워크 내에 부하분산 서비스(Load Balancer as a Service)를 제공하는 것이다. 따라서, OpenStack의 네트워크 서비스 제어를 통해 사용자는 필요에 따라 테넌트 네트워크에 부하분산 서비스를 자유롭게 추가하고 수정하며 제거할 수 있다.

이 외에도 Big Switch, Brocade, Hyper-V, Plum Grid, Midonet 플러그인이 새롭게 추가되어 공개되었으며, L3/L4 패킷 필터링 기능과 메타데이터 서비스 기능 등이 Grizzly 배포판에서 개선되었다[8].

마. Grizzly 배포판의 제약사항

클라우드 컴퓨팅 서비스에서 네트워크의 중요성이 부각되는 만큼, OpenStack에서 네트워크 서비스를 제공하는 Quantum에 점차 다양한 기능들이 추가 확대되고 있다. 그러나, 여전히 많은 네트워크 서비스 기능이 새롭게 추가되어야 하고, 또한 이미 제공되는 기능들의 오류 수정이나 개선도 병행되어야 한다.

가장 최근에 공식 발표된 Grizzly 배포판의 주요 제약사항을 살펴보면 다음과 같다[9].

먼저 Grizzly 배포판의 가장 큰 특징 중 하나인 멀티 호스트 기능이 완전하지 못한 것이다. 기존 nova-network의 경우 가상머신이 생성되는 컴퓨팅 노드 상에 네트워크 제어를 담당하는 nova-network이 함께 설치 운용되어 자연스레 멀티 호스트 기능이 제공되었으나, Quantum 프로젝트는 네트워크 서비스 제어 기능을 담당하는 독립된 네트워크 노드를 구축하여 운용함에 따라 멀티 호스트 기능을 제공하는데 제약이 따른다. 물론 Grizzly 배포판에서 이러한 문제를 해결하기 위해 모든 네트워크 서비스 에이전트에게 Heartbeat 메시지를 주기적으로 전송하여 상태를 점검하고, 가용한 네트워크 서비스 에이전트 중 하나를 선정하여 네트워크 서비스를 제공하는 Quantum Scheduler 기능 등이 추가되기도 하였다. 그러나, 실제 네트워크 노드를 멀티 호스트 환경으로 구축했을 때 Nova를 통해 생성된 가상머신이 어떤 물리노드에 생성되어 있는지와 이 가상머신에게 네트워크 서비스를 제공하는 특정 네트워크 에이전트의 관계 정보가 엄격하게 관리되지 못함에 따라 현재로는 멀티 호스트 기능이 완전하게 제공되지 않는 상태이다.

두 번째 제약사항으로써, 서로 다른 L2 네트워크에서 IP 주소를 중복하여 사용하기 위해, L3 에이전트 또는 DHCP 에이전트가 설치된 네트워크 노드에는 반드시 Linux network namespace가 지원되어야만 한다.

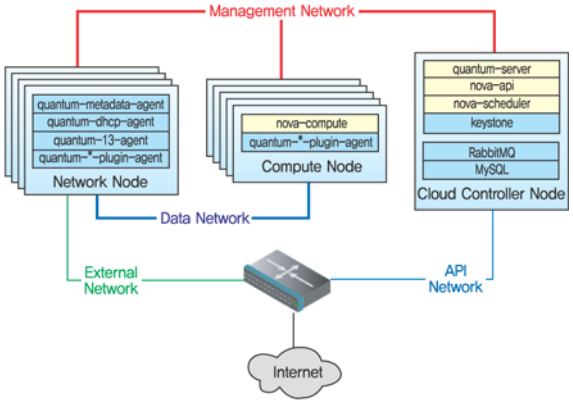
그 다음 제약사항으로는 MetaPlugin 기능이 실험적으로 추가되었으나 아직 폭넓은 검토나 시험이 진행되지 못한 상태이다. 현재 Quantum의 플러그인은 한번에 하나의 플러그인만 사용할 수 밖에 없는 제약을 갖고 있다. 그러나, 클라우드 인프라 내에서 다양한 네트워크 하부 기술을 사용하기 위해서는 동시에 서로 다른 플러

그인을 함께 사용할 수 있어야 하는데, 이를 실현하는 방법 중의 하나가 바로 MetaPlugin 기능이다. MetaPlugin은 다수의 Quantum 플러그인을 서브 플러그인으로써 감쌀 수 있는 상위의 플러그인을 뜻한다.

이외에도 L3 에이전트에서 IPv6가 지원되지 않거나, 네트워크 에이전트간 원격 메시지 통신(RPC: Remote Procedure Call)을 위해 기존의 RabbitMQ[10] 이외에 ZeroMQ[11]를 실험적으로 포함하고 있으나 아직 검증이 완료되지 않은 상태이다.

3. OpenStack 네트워크 구축 모델

OpenStack 환경 하에서 물리적인 서버는 (그림 4)와 같이 크게 클라우드 제어 노드, 컴퓨팅 노드, 네트워크 노드의 세 가지 유형으로 나뉘어 운용될 수 있다[9]. 클라우드 제어 노드는 사용자 또는 관리자의 요청에 의해 OpenStack 클라우드 컴퓨팅 서비스를 제어하는 소프트웨어가 설치 및 운용되는 물리노드로써, quantum-server를 비롯해 nova-api, nova-scheduler, Keystone 등이 설치된다. 컴퓨팅 노드는 가상머신을 생성하여 클라우드 컴퓨팅 서비스를 제공하는 물리노드로써, 하이퍼바이저와의 인터페이스를 통해 가상머신을 제어하는 nova-compute와 물리노드 내의 가상머신 간 또는 다른 물리노드에 있는 가상머신과의 네트워크 연결을 위



(그림 4) OpenStack 네트워크 구축 모델

한 가상스위치를 제어하는 플러그인 에이전트가 설치된다. 마지막으로 네트워크 노드는 앞서 기술한 바와 같이 L3 에이전트 또는 DHCP 에이전트 등이 설치되어 클라우드 네트워크 서비스를 제공한다.

OpenStack에서는 클라우드 제어 노드와 다수의 컴퓨팅 노드들, 그리고 하나 이상의 네트워크 노드 등으로 기본적인 클라우드 컴퓨팅 서비스를 제공할 수 있으며, 이 외에 볼륨 저장장치 서비스나 오브젝트 저장장치 서비스 등을 제공하기 위해서는 별도의 저장장치를 필요로 한다. 이때 각 물리 노드 사이의 네트워크 연결은 속성에 따라 네 가지의 서로 다른 네트워크로 구성된다.

관리 네트워크(management network)는 OpenStack 컴포넌트 사이의 내부 통신을 위한 네트워크로써, 관리 네트워크에 연결된 네트워크 인터페이스는 반드시 데이터센터 내에서만 접근이 가능한 IP 주소를 가져야 한다.

데이터 네트워크(data network)는 L2 테넌트 네트워크로서 테넌트가 생성한 가상 머신들 간의 데이터 교환은 물론 네트워크 노드의 가상 라우터를 통한 외부 망과의 연결을 위해 사용된다. 데이터 네트워크에서 사용되는 IP 주소는 OpenStack 네트워크 서비스를 위해 사용하는 플러그인의 종류에 따라 달라질 수 있다. 그 이유는 플러그인마다 IP 주소관리와 테넌트별 가상 네트워크 구성을 위해 각기 다른 기술이나 방식을 채택할 수도 있기 때문이다[12].

외부 네트워크(external network)는 L3 에이전트를 통해 테넌트 네트워크 밖의 외부망으로 연결되는 네트워크로서, 인터넷 상의 임의 사용자로부터 접근이 가능한 IP 주소를 가져야 한다.

API 네트워크(API network)는 원격에서 모든 OpenStack API를 이용할 수 있어야 하므로, 외부 네트워크와 같이 인터넷 상의 임의 사용자로부터 접근이 가능한 IP 주소를 가져야 한다.

이와 같은 서로 다른 속성의 네트워크가 OpenStack 기반의 클라우드 네트워크에서 모두 구성이 되어야 하

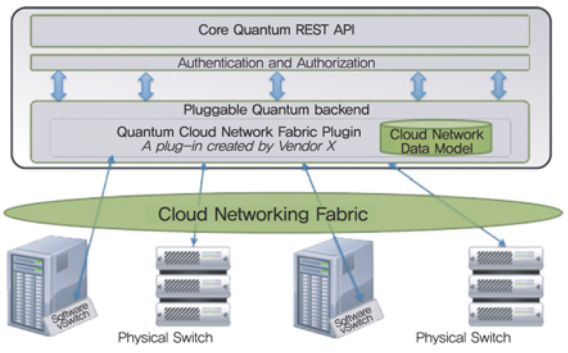
므로, 클라우드 제어 노드와 컴퓨팅 노드들은 최소한 2개의 네트워크 인터페이스를, 네트워크 노드는 3개의 네트워크 인터페이스를 갖도록 권고하고 있다.

4. Neutron 구조

Neutron 프로젝트의 가장 큰 특징 중 하나가 바로 다양한 네트워크 하부 기술을 클라우드 환경에서 수용하여 제어할 수 있도록 플러그인 구조를 채택한 것이다. Neutron 플러그인의 참조 구조[13]를 도식화한 (그림 5)을 기준으로 살펴보면, 가장 상위에 Neutron REST API를 처리하는 Neutron server가 위치한다. Neutron server에서는 L2 네트워크, L3 서브넷 그리고 포트를 제어하는 기본(core) REST API 외에도 확장 API를 지원하여 새롭게 추가되는 다양한 네트워크 서비스의 API도 함께 지원할 수 있다.

Neutron server를 통해 전달된 사용자의 제어 요청은 Keystone과 연결된 인증 및 권한 설정 블록을 통해 사용자의 인증과 접근권한을 확인하며, 인증된 메시지는 설정된 Neutron 플러그인으로 전달된다.

플러그인에서는 생성한 가상 네트워크에 대한 모든 정보를 자체 데이터베이스에서 관리하는데, L2 가상 네트워크의 생성 등과 같이 API의 종류에 따라 실제 네트워크 장치 또는 가상 네트워크 장치와의 연동없이 데이터베이스에 정보를 추가, 갱신 또는 삭제하는 것으로서



(그림 5) Neutron Plugin 참조 구조

완료되기도 한다. 만일 네트워크 장치와의 연동이 필요한 경우, 컴퓨팅 노드 또는 네트워크 노드 등에 설치된 에이전트들과 원격 메시지 호출(RPC)을 통해 제어 메시지를 전달한다.

5. Havana 버전에서의 주요 개선사항

오는 10월 공식 발표될 예정인 차기 OpenStack 배포판인 Havana 버전에서는 세 단계의 마일스톤을 각각 거치면서 Neutron의 많은 기능들이 추가되거나 완성도를 높여나갈 예정이다[14].

지난 5월을 목표로 개발된 Havana-1에서는 멀티 호스트 기능을 지원하기 위해 nova-compute에서 전달된 가상 포트에 대한 호스트 ID를 저장하는 데이터베이스를 Neutron에 추가할 예정이며, sudo 명령과 같이 권한이 없는 일반 사용자가 root 사용자로써 명령을 실행할 수 있도록 rootwrap 기능을 Neutron에 도입하는 것이다. 이 외에도 Open vSwitch 플러그인이 기존의 KVM/QEMU 하이퍼바이저 뿐만 아니라 XenServer도 지원하도록 개선되는 등 플러그인에 대한 추가나 개선이 포함되었다.

다음 단계인 Havana-2에서는 방화벽 서비스(Firewall as a Service)를 위해 에이전트와 플러그인, 데이터베이스 모델 등이 개발될 계획이며, 부하분산 서비스와 L3 서비스 제어와 같은 확장 API를 기본(core) API로 포함시키는 등 기본 API의 확대개편뿐만 아니라, QoS 제어를 위한 API도 확장 API로서 새롭게 추가될 예정이다.

또한 복수의 플러그인을 동시에 지원하기 위한 방법으로서 Grizzly 배포판에서 추가된 MetaPlugin 외에 Modular L2 Plugin과 에이전트를 개발할 계획이다. Modular L2 Plugin 방식은 다양한 하부 네트워크 기술을 플러그인이 아닌 드라이버로 개발하고, 복수의 드라이버를 동시에 Modular L2 Plugin에 로딩하는 방식이다. 그리고, 부하분산 서비스도 드라이버 방식을 채택하

여 하나의 부하분산 서비스 플러그인으로 여러 부하분산 장비나 솔루션을 이용할 수 있도록 지원할 예정이다.

멀티테넌트 네트워크를 구성하는 방안으로서 종전의 VLAN과 GRE(General Routing Encapsulation) 터널링 방식 외에도 VxLAN 기술이 Open vSwitch와 Linux Bridge 플러그인에서 지원될 예정이며, Open Daylight 플러그인 등의 추가는 물론 기존 플러그인의 기능 개선이 함께 이루어질 것으로 기대된다.

마지막으로 Havana-3 단계에서는 멀티 호스트 기능을 지원하도록 DHCP와 L3 기능을 개선하고, 과금 기능을 지원할 Ceilometer 프로젝트와의 연동을 위한 기능 개선 등이 추가될 예정이다.

Ⅳ. 결론

지난 2011년 HP가 OpenStack 기반의 HP 클라우드 서비스를 출시하면서 공용 클라우드 서비스 시장에 진출하는 등 OpenStack을 이용한 클라우드 서비스가 점차 확산되고 있으며, OpenStack도 범용 클라우드 운영 체제로서 새로운 기능을 지속적으로 추가하고 있다. 특히 클라우드 서비스에서 멀티테넌트 네트워크의 구성, 다양한 네트워크 하부 기술의 수용, 그리고 서비스 요구사항에 적합한 가상 네트워크 인프라의 동적 프로비저닝 등 네트워크 서비스의 중요성만큼이나 클라우드 네트워크 기술에 대한 관심과 기술개발 노력이 집중되고 있다.

용어해설

Tenant (테넌트) 클라우드 컴퓨팅 서비스를 위한 컴퓨팅, 네트워크, 저장장치 등의 자원에 대한 접근이 상호 격리된 서비스 사용자 그룹

Fixed IP OpenStack Neutron의 DHCP 에이전트를 통해 관리되는 IP 주소로서, 테넌트 네트워크 내에서 가상머신에게 동적으로 할당되는 IP 주소

Floating IP OpenStack Neutron의 L3 에이전트 통해 관리되는 IP 주소로서, 테넌트 네트워크의 외부망에서 접근이 가능한 IP 주소

약어 정리

API	Application Programming Interface
DHCP	Dynamic Host Configuration Protocol
GRE	General Routing Encapsulation
IP	Internet Protocol
MPLS	Multi-Protocol Label Switching
REST	Representational State Transfer
RPC	Remote Procedure Call
VLAN	Virtual Local Area Network
VPN	Virtual Private Network
SNAT	Source Network Address Translation

참고문헌

[1] OpenStack Project, http://www.openstack.org/

[2] OpenStack Compute Administration Guide – Grizzly, 2013.1, http://docs.openstack.org/grizzly/openstack-compute/admin/content

[3] D. Wendlandt, Intro to OpenStack Quantum, OpenStack Folsom Summit, 2012, http://www.slideshare. net/dan went/quantum-folsom-summit-developer-overview

[4] D. Wendlandt, OpenStack Quantum: Taking OpenStack Networking to New Heights, OpenStack Grizzly Summit, 2012, http://www.slideshare.net/dan went/quantum-grizzly-summit

[5] OpenStack Quantum 2011.3 “diablo” Release Notes, http://launchpad.net/quantum/diablo/2011.3

[6] OpenStack 2012.2 (Folsom) Release Notes, http://wiki.openstack.org/wiki/ReleaseNotes/Folsom

[7] HAProxy, http://haproxy.1wt.eu

[8] OpenStack 2013.1 (Grizzly) Release Notes, http://wiki.openstack.org/wiki/ReleaseNotes/Grizzly

[9] OpenStack Networking Administration Guide – Grizzly, 2013.1, http://docs.openstack.org/grizzly/openstack-network/admin/content

[10] RabbitMQ, http://www.rabbitmq.com

[11] ZeroMQ, http://www.zeromq.org

[12] Quantum APIv2 Specification, https://wiki.openstack.org/wiki/Quantum/APIv2-specification

[13] OpenStack, Quantum and Open vSwitch – Part II, http://openvswitch.org/openstack/2011/08/01/openstack-quantum-and-open-vswitch-part-ii/

[14] OpenStack Quantum Timeline, http://launchpad.net/quantum/+ serie