ISBN ?????????

EGI Federated Cloud 구축 기술 보고서 II

2015. 11. 30.

한국과학기술정보연구원

저자소개

김상완 (Sangwan Kim)
Korea Institute of Science and Technology Information
Supercomputing Center
Senior Researcher
Email : sangwan@kisti.re.kr

황순욱 (Soonwook Hwang)
Korea Institute of Science and Technology Information
Supercomputing Center
Researcher, PhD
Email : hwang@kisti.re.kr

ISBN ??????????

목 차

참고: Federated Cloud 구축 기술 보고서 (2014.11.30, KISTI, ISBN 978-89-294-0589-2)

제1장 머리말

이 문서는 EGI Federated Cloud에 대한 소개와 Federate Cloud의 resource provider 로 참여하기 위해 필요한 시스템의 준비와 소프트웨어 설치에 관한 경험을 기술하였다. EGI [1] 는 유럽내 참여 NGI(National Grid Initiatives)와 유럽내 국제 연구기관(European International Research Organizations | EIROs)를 위하여 연구자들의 협력과 컴퓨팅 자원의 공동 활용을 위한 비영리 기구로 네덜란드에 본부를 두고 있다. EGI.eu는 2010년 2월 8일에 조직되었으나, 이 이전에 DataGrid Project(2001), EGEE(Enabling Grid form E-sciencE, 2004)에 이어 EGI-InSPIRE(Integrated Sustainable Pan-European Infrastructure for Researchers in Europe, 2010년 5월~2014년 12월)를 추진하였고, EGI-Engate(2015년 3월~)를 추진하고 있다. EGI-InSPIRE에는 30개국이상에서 70개 이상의 기관들이 참여중이다.

EGI Federated Cloud[2]는 EGI에 참여하고 있는 기관들이 각자 운영하고 있는 클라우드 서비스를 연합하여 유럽과 전세계 사용자들을 대상으로 제공하기 위한 목적으로 출발하였다. 표준 기술에 기반은 IaaS 클라우드 서비스로 특정 vendor에 lock-in을 방지하고, 다양한 resource provider를 연합하여, 기존의 Grid 인프라와도 자연스럽게 통합되도록 지향하고 있다.

2014년 5월 EGI Community Forum 2014에서 정식 서비스로 전환되었으며, 현재 14개국가, 21개 기관이 참여하고 있다.

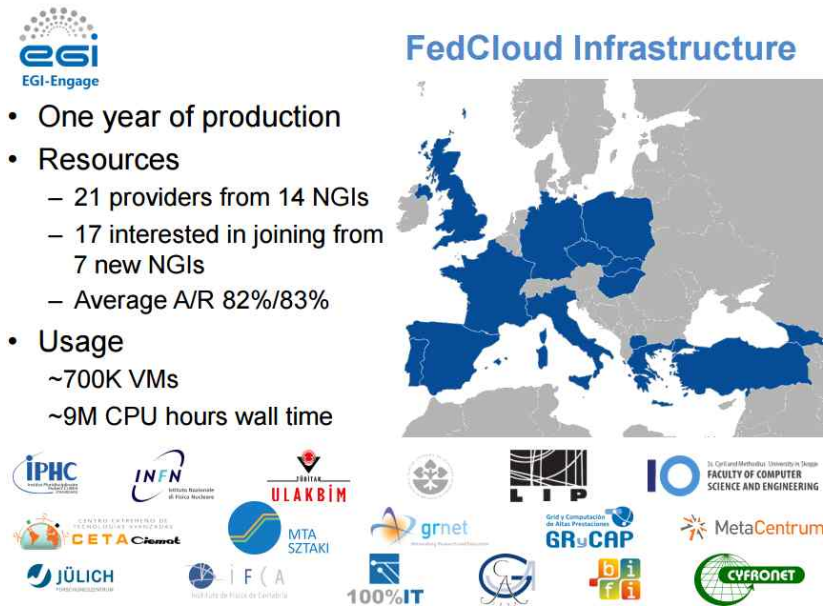


그림 1 Federated Cloud 자원 현황

1) http://www.egi.eu/
2) http://www.egi.eu/infrastructure/cloud/

제2장 EGI Federated Cloud 개요

제1절 개요

European Grid Infrastructure (이하 EGI)의 자원 센터들은 십 여년 넘게 운영되고 있으며, 과학자들의 협업과 계산 및 데이터 집약적인 응용분야를 지원하기 위한 각종 서비스를 제공하고 있다.[1] 이러한 서비스들은 그리드 미들웨어 소프트웨어를 활용하여 제공되고 있다. 몇몇의 NGI(National Grid Infrastructures, 국가별 그리드 인프라)들은 클라우드 서비스(cloud services)를 자국의 연구자들을 위해 제공하고 있는데, 많은 연구자들이 자국의 클라우드 서비스를 국가간 연구 협력안에서 공유할 수 있기를 원하고 있다. 이러한 요구는 십수년전 grid computing 이 출현하게된 동기화 비슷한 것이다. Grid computing 은 각 연구소에 있는 batch computing cluster들을 연합(federation)하여 사용하기 위한 요구에서 출발하였다. EGI Federated Cloud 는 이러한 요구에서 출발하였으며, 유럽내 NGI와 비유럽권의 연구기관의 자원까지도 대상으로 하는 연합된 클라우드 컴퓨팅 자원이라고 할 수 있다.

제2절 자원의 규모

EGI Federated Cloud 인프라는 2014년 5월에 production 으로 선언되어 정식으로 EGI의 서비스의 하나가 되었다. 자원의 규모는 2,000 CPU core와 15TB 의 storage 로 구성되어 있다. 2025년까지 10M core와 1EB(exabyte)이상의 storage 규모로 제공하는 것이 목표이다.

2015년 11월 현재 EGI Federated Cloud의 자원 현황은 다음 표와 같다. (최신 현황[3]을 참고)

3) https://wiki.egi.eu/wiki/Federated_Cloud_Operation

표 3 EGI Federated Cloud Resource Provider (Fully integrated)

Affiliation	CC	Host Site	CMF	Resources			Certified
				cores	RAM (GB)	Storage (TB)	
100 Percent IT Ltd	UK	100IT	OpenStack	120	128	16	certified
UNIZAR/BIFI	ES	BIFI	OpenStack	720	740	36	2014-02-14
Cyfronet (NGI PL)	PL	CYFRONET-CLOUD	OpenStack	200	400	20	2014-05-08
FCTSG	ES	CESGA	OpenNebula	288	592		2014-03-06
CESNET	CZ	CESNET-MetaCloud	OpenNebula	416	741	56.6	2014-02-24
FZ Jülich	DE	FZJ	OpenStack	216	294	95	(planed)
GWDG	DE	GoeGrid	OpenNebula	192	768	40	2014-03-13
CSIC/IFCA	ES	IFCA-LCG2	OpenStack	2288	7616		2014-03-06
GRNET	GR	HG-09-Okeanos-Cloud	Synnefo	20	40	0.4	2014-05-13
II SAS	SK	IISAS-FedCloud	OpenStack	176	528	50	2014-04-03
INFN-Catania	IT	INFN-CATANIA-NEBULA	OpenNebula	32	132	5.4	certified
	IT	INFN-CATANIA-STACK	OpenStack	16	65	16	certified
INFN-Bari	IT	PRISMA-INFN-BARI	OpenStack	300	600	50	certified
INFN-Padova	IT	INFN-PADOVA-STACK	OpenStack	144	288	9.1	2014-06-03
TUBITAK ULAKBIM	TR	TR-FC1-ULAKBIM	OpenStack	336	672	40	
UKIM	MK	MK-04-FINKICLOUD	OpenNebula	100	24	1	certified
CETA-CIEMAT	ES	CETA-GRID	OpenStack	184	224	5.3	
IPHC	FR	IN2P3-IRES	OpenStack	180	1024	5	
NCG-INGRID-PT/LIP	PT	NCG-INGRID-PT	OpenStack	80	192	3	
Polythecnic Univ. of Valencia I3M	ES	UPV-GryCAP	OpenNebula	128	192	5	
	CZ DE ES FR GR IT MK PL SK SK TR UK 12개국			6,136	15,260	453.8	

제3절 워킹 그룹

FedCloud Task Force의 활동은 워킹 그룹으로 나뉘어져 있다.

1. VM Management
   - Leader : Boris Parák (CESNET)
2. Data Management
   - Leader : Kostas Koumantaros (GRNET)
3. Information Discovery
   - Leader : Peter Solagna (EGI.eu)
4. Accounting
   - Leader : Stuart Pullinger (STFC)
5. Monitoring
   - Leader : Emir Imamagic (SRCE)
6. Federated AAI
   - Leader : Paul Millar (DESY)
7. VM Image Management
   - Leader : Marios Chatziangelou (IASA)
8. VM endorsement
   - Leader : Vincenzo Spinoso (EGI.eu)
9. Brokering, APIs, SDKs
   - Leader : Enol Fernandez (EGI.eu)
10. Security
    - Leader : Linda Cornwall (STFC)
11. Intra Cloud Networking
    - Leader : Zdenek Sustr(CESNET)

제4절 Federated Cloud 기술

클라우드 시스템을 상호 연동하기위해서는 몇 가지 공통적인 인터페이스가 정의 되어야 한다.

- 클라우드 인터페이스
  - VM management interface: OCCI
  - Data management interface: CDMI
- VM Image management
- EGI 코어 서비스와 통합
  - Virtual Organisation Management & AAI: VOMS
  - Information discovery: BDII
  - Central service registry: GOCDB
  - Monitoring: SAM
  - Accounting

자세한 내용은 Technology Architecture [4] 를 참조

EGI 를 구성하는 소프트웨어 정책은 open standard와 interoperability 로 요약할 수 있다. 공정하고 공개된 투명한 기준은 자원 제공자나 소비자를 포함한 모든 참여자들에게 활동의 기회를 제공해 준다.

EGI는 다음의 표준들을 클라우드 인프라 연동을 위한 요구사항의 일부로 포함하고 있다.

■ OCCI (Open Cloud Computing Interface)

OCCI (Open Cloud Computing Interface) 는 RESTful 프로토콜로써 클라우드 기반의 자원에 대한 접근과 정보 조회를 위해 설계된 API이다. 공식적인 표준 명세는 OGF의 OCCI-WG 워킹그룹[5]에서 업데이트 되고 있다.

OCCI 표준은 세가진의 기본적인 요소로 구성되어 있다. (1) OCCI Core Model[6]은 인스턴스의 타입을 정의하는데, 이 타입들은 실제로 다루어질 자원에 대한 추상화된 상태로 정의가 되어 있다. (2) OCCI HTTP Rendering 모델[7]은 RESTful OCCI API를 이용하여 코어 모델과 상호작용하는 방법을 정의한다. (3) OCCI Infrastructure[8] 확장 모델은 IaaS 영역에서 OCCI의 확장 방법에 대해 기술하고 있는데, 자원의 종류(type)와 속성(attribute) 및 그 자원에 대한 동작(action)에 대해 기술한다.

OCCI Core 에서는 기본 타입으로 Resource, Link, Action, Mixin 이 있다. Resource 란 예를 들면 자원 제공자가 제공하는 이미지(storage resource), 네트워크(netowrk resource), 가상머신(compute resource) 또는 이용가능한 서비스들이 있다. Link는 2가지 Resource 인스턴스 간의 연결을 말한다. 예를 들면 네트워크 인터페이스 링크나 스토리지 링크는 스토리지와 네트워크 자원을 컴퓨팅 자원에 연결하는 역할을 한다. Action은 특정 Resource 타입이나 Resource 인스턴스의 집합에 대한 동

4) https://wiki.egi.eu/wiki/Fedcloud-tf:Technology:Architecture
5) http://occi-wg.org/
6) https://www.ogf.org/documents/GFD.183.pdf
7) https://www.ogf.org/documents/GFD.185.pdf
8) https://www.ogf.org/documents/GFD.184.pdf

작(operation)을 정의한다. 가상 머신을 재시작 하거나 마이그레이션 하는 것은 해당 Resource의 상태를 변화시키는 Action에 해당한다. Mixin은 확장성을 위한 것으로 프로바이더에 특화된 방법을 제공하기 용이하도록 한다.

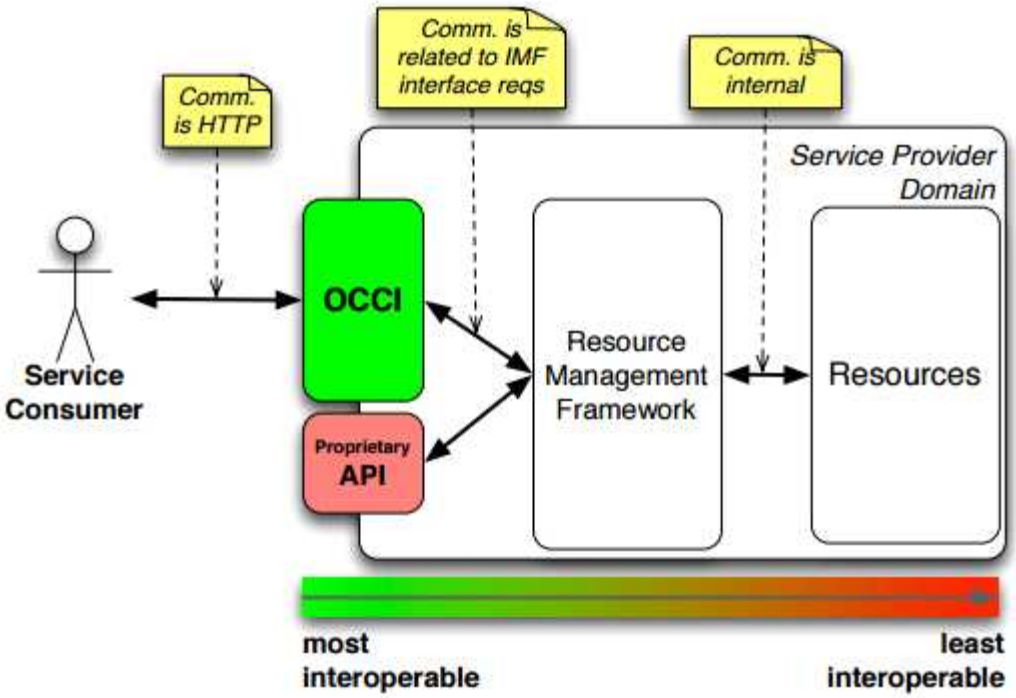


그림 2 서비스 구조상 OCCI의 위치

- FedCloud를 위한 OCCI의 확장

Contextualization 은 준비된 가상 머신 이미지를 이용하여 부팅할 때 필요한 소프트웨어를 설치하고, 설정하는 준비과정을 말한다. (hostname에 대한 설정, IP주소 설정, SSH 키에 대한 설정, 서비스를 시작하는 과정, 특정 어플리케이션을 설치하는 등) OCCI v1.1 에서는 이러한 contextualization 과정에 대한 매카니즘이 부족하기 때문에 FedCloud 에서는 새로운 OCCI mixin을 이용하여 user-provided 데이터를 context 정보로 VM에 제공할 수 있도록 확장 하고 있다. 사용된 mixin 은 다음과 같다.

term	schema	attributes
user_data	http://schemas.openstack.org/compute/instance#	org.openstack.compute.user_data: string that holds base64 encoded data to be available at the VM upon instantiation
public_key	http://schemas.openstack.org/instance/credentials#	org.openstack.credentials.publickey.name: string with the name of the public key (optional) org.openstack.credentials.publickey.data: string with the public key

각각의 Cloud Management stack(OpenNebula, OpenStack, Synnefo)에서는 이 정보를 이용하여 VM에 적용하는 메커니즘이 존재한다. FedCloud 에서는 cloud-init [9]을 이용하는 것을 권장한다.

cloud-init 은 context 정보를 처리할 수 있는 방법을 제공하며, 다양한 OS 버전와 IaaS 클라우드 플랫폼에 적용할 수 있다. 기본적으로 cloud-init 은 (1) ssh-key를 root user(or equivalent)의 ~/.ssh/authorized_keys 파일에 추가한다. (2) user data 가 존재하면 이것이 쉘 스크립트라면 시작시 실행한다.

■ CDMI (Cloud Data Management Interface)

CDMI는 SNIA [10]에서 정의된 클라우드 스토리지를 위한 인터페이스 스토리지 자원에 대한 개방된 표준의 RESTful API이다. CDMI 는 아마존 AWS S3와 MS Azure Blob와 유사하다. CDMI는 스토리 관리 시스템과 단일의 데이터 모두에 대한 공통된 방법을 제공한다. 프로토콜의 설계의 기본 원칙은 확장성(flexibility)과 효율성(efficiency)을 바탕으로 설계 되었다. 몇몇의 전송량이 많은 동작들(예를 들면 bob download)은 pure HTTP 클라이언트를 이용하여 수행될수 있다. CDMI는 Object 라는 개념을 기반으로 설계되었다. Object 종류에 따라 가능한 오퍼레이션과 메타데이터 스키마가 상이하다. 각각의 Object 는 유일한 ID를 가진다.

EGI Federated Cloud에서 사용되는 CDMI object 는 4가지가 존재한다.

- Data object : 파일이나 메타데이터를 위한 추상화 객체
- Container : 폴더를 위한 추상화. Container는 다른 Container를 포함가능. HTTP 프로토콜 이외의 프로토콜로 export 하는 경우 container 레벨에서 수행됨
- Capability : 특정 object에 대한 feture set에 대한 정보. CDMI 클라이언트가 어떤 functionality 가 있는지 알아보기 위해서는 GET 요청으로 /cdmi_capabilities를 요청하면 된다.
- Domain : 특정 정보의 배포와 관련

가상머신에 스토리지 자원을 연결하기 위해 NFS나 iSCSI를 사용하는데, CDMI는 이러한 정보를 container metadata를 통해 클라이언트에게 제공하게 되고, 클라이언트는 이 정보와 OCCI 프로토콜을 이용하여 storage를 VM에 연결한다.

■ VM Image management

분산된 federated cloud 인프라 상에서 사용자는 VM 이미지를 여러 자원에서 분산하여 사용해야 하는 상황에 처하게 된다. VM 이미지 관리 시스템은 새로운 VM이미지가 어떤 자원에 해당 이미지가 지원되는지를 사용자에게 알려준다. 다음과 같은 기능을 제공하는 것이 목표이다.

- VM 이미지 라이프사이클 관리 : EGI 인프라 차원에서 소프트웨어 라이프 사이클 관리가 이루어짐
- 자동화된 VM 이미지 배포 : VM 이미지 정보를 한번만 publish 하면 자동적으로 클라우드 자원 제공자로 배포가 이루어진다.
- 비동기적인 배포 메커니즘 : 이미지를 publishing 하는 것과 pooling 하는 것은 비동기적으로 이

9) http://cloudinit.readthedocs.org/en/latest/
10) http://www.snia.org/

루어짐

- 제공자에 의한 VM 이미지의 위반 정책 설정 : 특정 자원 제공자가 스스로의 인프라를 보호하기 위해 VM 이미지나 인스턴스와 관련된 보안장치를 강제할 수 있음.

■ Virtual Organisation Management & AAI: VOMS

EGI 인프라 내에서 연구를 위한 커뮤니티들은 Virtual Organisations(VO)을 이용하여 인증수단으로 이용한다. EGI Cloud도 VO 정보를 authentication과 authorization을 위해 사용한다. 모든 자원 제공자는 다음의 3가지 VO를 지원해야 한다.

- ops VO : 모니터링을 위한 전용 VO
- dteam VO : 사이트 운영자를 위한 테스트 목적의 VO
- fedcloud.egi.eu VO : 특정 응용 커뮤니티에 속하지 않은 VO로 FedCloud 이용시 prototyping와 validation을 목적으로 사용하는 VO

자원 제공자는 그 외에 추가적으로 VO를 설정하여 해당 커뮤니티의 접근을 허용할 수 있다.

사용자는 VOMS 인증서를 VOMS 서버에 요청하고 로컬 VOMS 프록시 인증서를 생성한다. VOMS 프록시 인증서는 이후의 OCCI 요청에 사용된다. rOCCI 클라이언트는 cloud management framework 의 OCCI 인터페이스에 접근하게 되고, 해당 인증서의 사용자와 VO정보를 이용하여 로컬 계정으로 연결시킨다.

OpenStack에서 rOCCI 클라이언트는 OpenStack Keystone 서비스로부터 token을 요청하여 받는다. 이 과정에서 VOMS 프록시 인증서를 이용하여 사용자에게 투명하게 진행된다.

■ Information discovery: BDII

EGI에서 사용자는 Information discovery 서비스에 접속하여 이용가능한 자원정보를 조회할 수 있다. EGI에서 현재는 Berkeley Database Information Index(BDII)를 이용하여 EGI 전에의 자원정보를 계측적으로 구성하여 제공하고 있다.

정보시스템은 3단계 레벨로 구성되어 있다.

서비스는 자신의 정보를 publish 한다. 이때 사용되는 스키마 표준은 OGF의 표준 형식인 GLUE2 스키마[11]를 사용한다. 서비스에서 퍼블리시된 정보는 Site-BDII에 수집된다. Site-BDII는 EGI의 모든 사이트에 설치되어야 한다. Site-BDII는 Top-BDII에 의해 쿼리된다. Top-BDII는 국가단위 또는 지역단위로 계층적으로 관리된다.

자원 제공자는 이용가능한 자원에 대한 정보를 퍼블리시한 Site-BDII의 주소를 함께 제공해야 한다. 클라우드 자원에 맞도록 GLUE2 스키마를 확장하는 작업이 논의중이며, 현재는 기존 EGI 그리드 자원과 별도로 독립된 subtree인 Glue2GroupID=cloud 로 정보를 퍼블리시 하고 있다. 다음과 같은 정보들이 퍼블리시 된다.

- Cloud computing resources
- Service endpoint

11) http://www.ogf.org/documents/GFD.147.pdf

- Capabilities
- Interface, the type of interface - e.g. OCCI 1.2.0
- User authentication and authorization profiles supported by the services
- Virtual machine images made available by the cloud provider
- Resource templates (number of cores and physical memory) allocable in a VM

■ Central service registry: GOCDB

서비스 카탈로그인 GOCDB에서는 production infrastructure의 정적인 자원정보를 서비스 한다. 자원 제공자는 자신의 클라우드 자원의 정보를 GOCDB에 등록 시켜야 한다. 클라우드와 관련된 service type 은 다음과 같다.

- eu.egi.cloud.accounting
- eu.egi.cloud.storage-management.cdmi
- eu.egi.cloud.vm-management.occi.
- eu.egi.cloud.vm-metadata.marketplace
- eu.egi.cloud.vm-metadata.vmcatcher
- eu.egi.cloud.vm-metadata.appdb-vmcaster

■ Monitoring: SAM

EGI 내에서 자운들은 SAM(Service Availabiligy Monitoring)을 통하여 모니터링을 받는다. 서비스 제공자는 이를 위해 자원을 모니터링 할 수 있는 probe 기능을 제공해야 한다. 클라우드 자원을 모니터링 하기 위해 현재 활용되는 probe들은 다음과 같다.

- OCCI probes (eu.egi.cloud.OCCI-VM and eu.egi.cloud.OCCI-Context): OCCI-VM은 OCCI 요청을 통하여 사전에 설정된 이미지의 인스턴스를 실행한다. OCCI-Context 는 해당 OCCI 인터페이스가 FedCloud extension을 적절히 지원하는지 체크한다.
- Accounting probe (eu.egi.cloud.APEL-Pub): 클라우드 자원이 아카운팅 저장소로 데이터를 보내오고 있는지 체크한다.
- TCP checks (org.nagios.Broker-TCP, org.nagios.CDMI-TCP, org.nagios.OCCI-TCP and org.nagios.CloudBDII-Check): 서비스에 대한 기본적인 TCP 연결 체크
- VM Marketplace probe (eu.egi.cloud.AppDB-Update): 미리 정의된 이미지를 AppDB로부터 가져오며, 업데이트 주기를 체크한다.
- Perun probe (eu.egi.cloud.Perun-Check): 내부적인 Perun 인터페이스를 이용하여 서버에 주기적으로 접속한다.

■ Accounting

OGF Usage record에서 기반한 Cloud Usage Record를 이용하여 어카운팅을 수행한다. 이ㄴㄴ EGI 어카운팅 정보 저장소로 보내져야하는 정보를 정의한다. OpenNebula는 cloudacc를 이용하며,

OpenStack은 CASO를 이용한다. 어카운딩 정보를 생성시킨다음 APEL SSM(Secure STOMP Messenger)를 이용하여 중앙의 어카운팅 저장소로 전송된다. EGI Accounting Potal [12]을 통하여 수집된 어카운팅 정보를 조회할 수 있다.

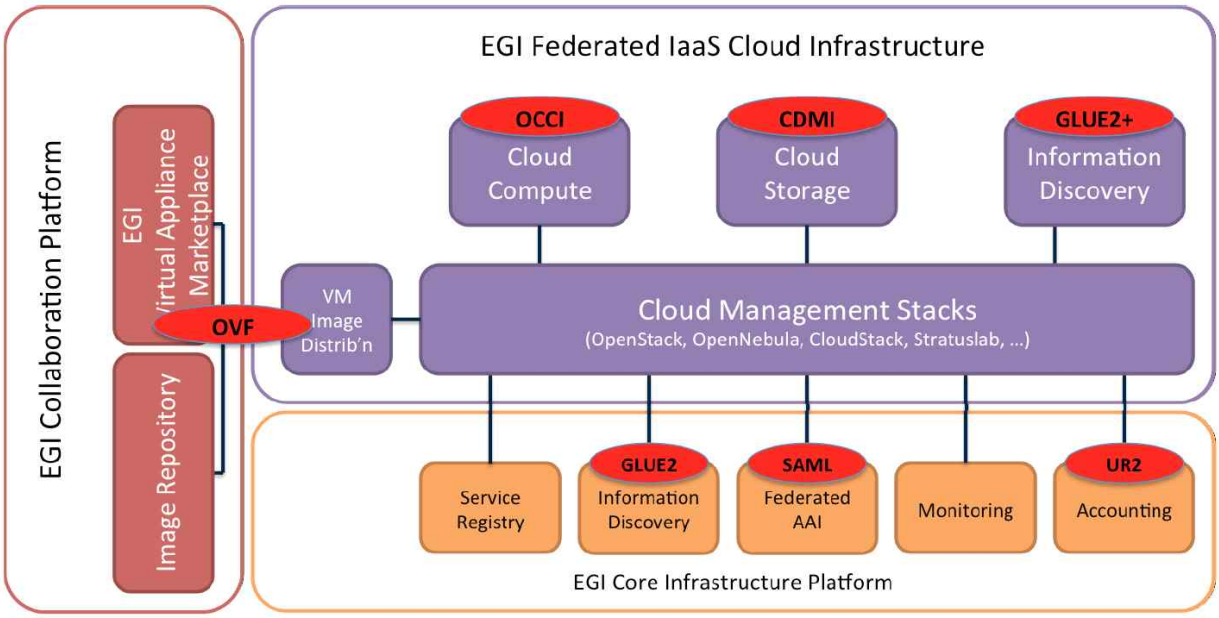


그림 3 EGI Federated Cloud Infrastructure Platform architecture and standards

12) http://accounting-devel.egi.eu/egi.php

제3장 Federated Cloud 사이트 구축

제1절 설치환경

1. 설치 시스템 사양

KISTI 사이트에 설치중인 자원 현황은 다음과 같다.

구분	대수	사양
Controller	1	NEXTSERVER 1120 2Way 1U Rack Server - 2 x Intel Xeon™ Processor E5-2660v2 - 96GB(6x 16GB) Reg. ECC DDR3 SDRAM - 3TB 3.5 SATA 7200rpm DISK x 1ea Intel(R) Xeon(R) CPU E5-2660 v2 @ 2.20GHz 10 cores x 2 sockets = 20 cores
Compute	2	Dell(TM) PowerEdge(TM) R815 Rack Mount Server - 4 x AMD Opteron 6378 2.4GHz 16C Turbo CORE 16M L2/16M L3 1600Mhz Max Mem 115W - 256GB Reg. ECC DDR3 SDRAM, 1333MHz - 6 x 1TB 7.2K RPM, 6Gbps Near Line SAS 2.5인치 Hard Drive AMD Opteron(tm) Processor 6378 cpu MHz : 2.4 GHz 16 cores x 4 sockets = 64 cores

장비 실제 사진 (제2전산실)

그림 4 Federated Cloud Resources in KISTI

제2절 OpenStack (Juno)

1. 설치준비

OpenStack 설치 가이드는 다음 주소를 참고하였다.

http://docs.openstack.org/juno/install-guide/install/yum/content/

Controller 노드로 사용할 장비는 DNS에 등록을 한다.

```
# nslookup  fccont.kisti.re.kr
Name:    fccont.kisti.re.kr
Address: 150.183.###.170
```

계산 노드는 /etc/hosts 등록하여 준다.

```
# vi /etc/hosts
150.###.###.170     fccont.kisti.re.kr
150.###.###.171     fccomp1.kisti.re.kr
150.###.###.172     fccomp2.kisti.re.kr
```

노드간 네트워크 통신을 위해 iptables 규칙을 설정한다. CentOS7 에서는 기본적으로 firewalld 가 사용되지만, iptables을 이용한 패킷 필터링 메커니즘을 이용하는 것이 좋은 것으로 여겨진다.

```
# vi /etc/sysconfig/iptables
# TRUSTED NETWORK
-A INPUT -s 150.###.###.0/24 -j ACCEPT
:wq
```

selinux 설정은 disabled 로 한다.

```
# setenforce 0
# vi /etc/selinux/config
```

계산 노드에서는 KVM hypervisor 가 설치 되어 있어야 한다.

```
[root@fccomp1 ~]$ lsmod | grep kvm
kvm_amd                40665  0
kvm                   333172  1 kvm_amd
```

기본적으로 설치할 패키지를 설치한다.

```
yum -y install      tree  wget  mlocate   bind-utils  ntsysv  man  telnet
yum -y groupinstall "Development Tools"
yum -y install      libxml2-devel openssl-devel lrzsz tree mlocate
yum -y update
```

파이썬 버전을 확인한다.

```
# python -V
Python 2.6.6
```

시간동기화를 위해 ntp 서비스를 설치한다. 필요시 /etc/ntp.conf 에 타임 서버 주소를 추가한다.

```
# yum -y install ntp
# service ntpd start
```

MySQL 데이터베이스 서버 설치

```
# yum  -y  install   mysql   mysql-server   MySQL-python
# chkconfig --level 2345 mysqld on
# service mysqld start
# /usr/bin/mysqladmin -u root password '########'
# alias db='mysql -u root -p########'
# mysql_secure_installation
```

OpenStack 설치를 위한 RDO repository 설치

Repository URL : http://repos.fedorapeople.org/repos/openstack/openstack-juno/epel-7/

```
#  yum install -y  http://rdo.fedorapeople.org/openstack-juno/rdo-release-juno.rpm
```

EPEL 확장 리포지터리 설치

```
# yum install -y http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-5.noarch.rpm
```

OpenStack 패키지 설치

```
# yum -y install openstack-selinux
# yum upgrade
# reboot
```

데이터베이스(MariaDB) 설치

```
# yum install -y   mariadb    mariadb-server    MySQL-python
# vi  /etc/my.cnf
```

```
[mysqld]
...
```

```
bind-address = 0.0.0.0

[mysqld]
...
default-storage-engine = innodb
innodb_file_per_table
collation-server = utf8_general_ci
init-connect = 'SET NAMES utf8'
character-set-server = utf8
max_connections = 1000
wait_timeout = 60

# systemctl enable mariadb.service
# systemctl start  mariadb.service
# mysql_secure_installation
```

(3) 메시징 서버 설치

OpenStack에서 서비스들 간에 상호 동작과 상태정보 전송을 위해 메시지 브로커를 이용한다. 메시지 브로커는 보통 컨트롤러 노드에서 실행된다. 이용가능한 브로커로는 RabbitMQ, Qpid, ZeroMQ 등이 있다.

RabbitMQ 설치

```
# yum install -y   rabbitmq-server
# vi /etc/rabbitmq/rabbitmq.config
[{rabbit, [{loopback_users, []}]}].
:wq
# systemctl enable rabbitmq-server.service
# systemctl start  rabbitmq-server.service
# systemctl status rabbitmq-server.service

# rabbitmqctl status | grep rabbit
Status of node rabbit@localhost ...
 {running_applications,[{rabbit,"RabbitMQ","3.3.5"},

# export RABBIT_PASS=******
# rabbitmqctl change_password guest $RABBIT_PASS
```

메시지 브로커 서비스 포트

서비스	포트
rabbitmq	5672

2. Identity 서비스(keyston) 설치

패키지 설치

```
# yum install -y   openstack-keystone   python-keystoneclient
# id keystone
uid=163(keystone) gid=163(keystone) groups=163(keystone)
```

데이터베이스 설정

```
# echo "CREATE DATABASE keystone;" | db
# export KEYSTONE_DBPASS=**********
# echo "GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY '$KEYSTO
NE_DBPASS';" | db
# echo "GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%'  IDENTIFIED BY '$KEYSTONE_DBPA
SS';" | db
# echo "select * from mysql.user" | db

# openssl rand -hex 10
13334e6a3a349f397351
# export ADMIN_TOKEN=13334e6a3a349f397351
# vi  /etc/keystone/keystone.conf
...
[DEFAULT]
...
admin_token = 13334e6a3a349f397351

# cp  /usr/share/keystone/keystone-dist-paste.ini    /etc/keystone/keystone-paste.ini

# keystone-manage pki_setup --keystone-user keystone --keystone-group keystone
# chown -R keystone:keystone /var/log/keystone
# chown -R keystone:keystone /etc/keystone/ssl
# chmod -R o-rwx /etc/keystone/ssl
# su -s /bin/sh -c "keystone-manage db_sync" keystone

# systemctl enable openstack-keystone.service
# systemctl start  openstack-keystone.service
```

keystone 관련 로그파일 경로

```
# less /var/log/keystone/keystone.log
```

keystone 서비스 포트

서비스	포트
keystone (public)	5000
keystone (admin)	35357

Tenant, User, Role 생성

Create the admin tenant:

```
# keystone tenant-create --name admin --description "Admin Tenant"
```

Create the admin user:

```
# export ADMIN_PASS=adminpass
# export EMAIL_ADDRESS=sangwan@kisti.re.kr
# keystone user-create --name admin --pass $ADMIN_PASS --email $EMAIL_ADDRESS
```

Create the admin role:

```
# keystone role-create --name admin
```

Add the admin role to the admin tenant and user:

```
# keystone user-role-add --user admin --tenant admin --role admin
```

Create the service tenant:

```
# keystone tenant-create --name service --description "Service Tenant"
```

Create the service entity for the Identity service:

```
keystone service-create --name keystone --type identity  --description "OpenStack Identity"
```

Create the Identity service API endpoints:

```
keystone endpoint-create ₩
  --service-id $(keystone service-list | awk '/ identity / {print $2}') ₩
  --publicurl http://controller:5000/v2.0 ₩
  --internalurl http://controller:5000/v2.0 ₩
  --adminurl http://controller:35357/v2.0 ₩
  --region regionOne
```

설치 검증

```
# unset OS_SERVICE_TOKEN OS_SERVICE_ENDPOINT
# keystone --os-tenant-name admin --os-username admin --os-password $ADMIN_PASS ₩
  --os-auth-url http://controller:35357/v2.0 token-get
+-----------+----------------------------------+
|  Property |              Value               |
+-----------+----------------------------------+
|  expires  |       2015-04-27T10:17:01Z       |
|     id    | aece8e2b3d684324864a4422508f3574 |
| tenant_id | 341216285fad4fd095a84a7c3cc7703f |
|  user_id  | 94d4bf10737e42379e15a7001faa7408 |
```

```
+----------+----------------------------------+
```

```
# keystone --os-tenant-name admin --os-username admin --os-password $ADMIN_PASS ₩
  --os-auth-url http://controller:35357/v2.0 tenant-list
+----------------------------------+---------+---------+
|                id                |   name  | enabled |
+----------------------------------+---------+---------+
| 341216285fad4fd095a84a7c3cc7703f |  admin  |   True  |
| 9052fd811bcc465d94f722685e9c24f4 |   demo  |   True  |
| 633beb3d209f48528d260abe4d504a71 | service |   True  |
+----------------------------------+---------+---------+
```

3. Image 서비스(glance) 설치

참고 : http://docs.openstack.org/juno/install-guide/install/yum/content/ch_glance.html

이미지 서비스는 다음 컴포넌트로 구성된다.

- glance-api : API 호출을 담당. image discovery, retrival, and storage
- glance-registry: 이미지의 메타데이터를 저장, 처리. 메타데이터는 size, type, 등등의 정보
- database : 이미지 메타데이터를 저장.
- storage reposotory for image file :

To install and configure the Image Service components

```
# yum install -y  openstack-glance python-glanceclient
# id glance
uid=161(glance) gid=161(glance) groups=161(glance)
```

데이터베이스 권한 설정

```
# echo "CREATE DATABASE glance;" | db
# export GLANCE_DBPASS=glancepass
# echo "GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost'   IDENTIFIED BY '$GLANCE_D
BPASS';" | db
# echo "GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%'   IDENTIFIED BY '$GLANCE_DBPASS';"
| db
```

user, tenant, service 생성

```
# source admin-openrc.sh
# export GLANCE_PASS=glancepass
# keystone user-create --name glance --pass $GLANCE_PASS
# keystone user-role-add --user glance --tenant service --role admin
# keystone service-create --name glance --type image  --description "OpenStack Image Servic
e"
```

endpoint 생성

```
# keystone endpoint-create ₩
  --service-id $(keystone service-list | awk '/ image / {print $2}') ₩
  --publicurl http://controller:9292 ₩
  --internalurl http://controller:9292 ₩
  --adminurl http://controller:9292 ₩
  --region regionOne
```

설정 파일 수정

```
vi  /etc/glance/glance-api.conf

[database]
...
connection=mysql://glance:GLANCE_DBPASS@controller/glance

[keystone_authtoken]
...
auth_uri = http://controller:5000/v2.0
identity_uri = http://controller:35357
admin_tenant_name = service
admin_user = glance
admin_password = GLANCE_PASS

[paste_deploy]
...
flavor = keystone
[glance_store]
...
default_store = file
filesystem_store_datadir = /var/lib/glance/images/

[DEFAULT]
:210
notification_driver = noop

[DEFAULT]
:3
verbose=True
debug=False
```

```
# vi /etc/glance/glance-registry.conf
[database]
```

```
:143
connection = mysql://glance:GLANCE_DBPASS@controller/glance
```

```
[keystone_authtoken]
:227
auth_uri = http://controller:5000/v2.0
identity_uri = http://controller:35357
admin_tenant_name = service
admin_user = glance
admin_password = GLANCE_PASS

[paste_deploy]
:243
flavor = keystone

[DEFAULT]
:85
notification_driver = noop

[DEFAULT]
...
verbose = True
```

데이터 베이스 초기화 및 서비스 시작하기

```
# su -s /bin/sh -c "glance-manage db_sync" glance
# systemctl enable openstack-glance-api.service openstack-glance-registry.service
# systemctl start  openstack-glance-api.service openstack-glance-registry.service
# systemctl status  openstack-glance-api.service
# systemctl status  openstack-glance-registry.service
```

이미지 서비스 사용 포트는 다음과 같다.

서비스	포트
glance-registry	9191
glance-api	9292

설치 검증

```
# mkdir /tmp/images
# wget -P /tmp/images http://cdn.download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.i
mg
# source admin-openrc.sh
# glance image-create --name "cirros-0.3.3-x86_64" --file /tmp/images/cirros-0.3.3-x86_64-d
isk.img ₩
  --disk-format qcow2 --container-format bare --is-public True --progress
```

```
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | 133eae9fb1c98f45894a4e60d8736619     |
| container_format | bare                                 |
| created_at       | 2015-05-07T08:31:16                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | 9eabafdb-b223-4144-aff5-a1279b53e4f4 |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | cirros-0.3.3-x86_64                  |
| owner            | cc5dd73321a8451aa7fe04d2edfde697     |
| protected        | False                                |
| size             | 13200896                             |
| status           | active                               |
| updated_at       | 2015-05-07T08:31:16                  |
| virtual_size     | None                                 |
+------------------+--------------------------------------+

# glance image-list
+--------------------------------------+---------------------+-------------+--------------
---+----------+--------+
| ID                                   | Name                | Disk Format | Container Form
at | Size     | Status |
+--------------------------------------+---------------------+-------------+--------------
---+----------+--------+
| 9eabafdb-b223-4144-aff5-a1279b53e4f4 | cirros-0.3.3-x86_64 | qcow2       | bare
   | 13200896 | active |
| 4f3bdd0e-6acc-40ae-853b-e33d23ed6207 | htcaas_agent        | qcow2       | ovf
   | 1915682816 | active |
+--------------------------------------+---------------------+-------------+--------------
---+----------+--------+
```

4. Compute Service (nova) 설치

참고: http://docs.openstack.org/juno/install-guide/install/yum/content/ch_nova.html

Compute 서비스는 다음과 같은 구성요소로 이루어져 있다.

- API

- nova-api 서비스 : end user 의 compute API 호출에 응답. OpenStack Compute API, Amazon EC2 API와 관리자를 위한 Admin API로 구성.

- nova-api-metadata 서비스 : instance로 부터 metadata 요청에 응답. 일반적으로 nova-network 설치시 multi-host 모드 일때만 사용됨.

■ Compute core
- nova-compute 프로세스 : hypervisor API를 이용하여 가상 머신 인스턴스를 시작하고 종료하는 daemon.
- nova-scheduler 프로세스: queue에서 가상머신 인스턴스 요청을 취하여 어떤 compute server host 에 실행할지를 결정.
- nova-conductor 모듈 : nova-compute 와 database 사이의 중계자 역할.

■ Networing for VMs
- nova-network 데몬: 네트워크 관련 요청을 처리함. (bridging interfaces 설정 또는 iptables 규칙 변경)
- nova-dhcpbridge 스크립트 : IP 주소 leases를 추적하고 데이터베이스에 기족한다. (dnsmasq dhcp-script 기능을 이용한다.)

■ Console interface
- nova-consoleauth 데몬
- nova-novncproxy 데몬
- nova-console 데몬
- nova-xvpnvncproxy 데몬
- nova-cert 데몬

■ Image management (EC2 scenario)
- nova-objectstore 데몬
- euca2ools 클라이언트 :

■ Command-line clients and other interfaces
- nova 클라이언트
- nova-manage 클라이언트

■ Other components
- The queue
- SQL database

가. 컨트롤러 노드 설치

패키지 설치

```
yum install -y openstack-nova-api openstack-nova-cert openstack-nova-conductor ₩
  openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler ₩
  python-novaclient
```

데이터베이스 권한 설정

```
echo "CREATE DATABASE nova;" | db
export NOVA_DBPASS=novapass
echo "GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost'   IDENTIFIED BY '$NOVA_DBPASS';"
| db
echo "GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%'   IDENTIFIED BY '$NOVA_DBPASS';" | db
```

tenant, user, service 등록

```
# source admin-openrc.sh
# export NOVA_PASS=novapass
# keystone user-create --name nova --pass $NOVA_PASS
# keystone user-role-add --user nova --tenant service --role admin
# keystone service-create --name nova --type compute  --description "OpenStack Compute"
# keystone endpoint-create ₩
  --service-id $(keystone service-list | awk '/ compute / {print $2}') ₩
  --publicurl http://controller:8774/v2/%₩(tenant_id₩)s ₩
  --internalurl http://controller:8774/v2/%₩(tenant_id₩)s ₩
  --adminurl http://controller:8774/v2/%₩(tenant_id₩)s ₩
  --region regionOne
```

```
# vi /etc/nova/nova.conf

[database]
...
connection = mysql://nova:NOVA_DBPASS@controller/nova

[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controller
rabbit_password = RABBIT_PASS

[DEFAULT]
:530
auth_strategy = keystone

[keystone_authtoken]
:2614
auth_uri = http://controller:5000/v2.0
identity_uri = http://controller:35357
admin_tenant_name = service
admin_user = nova
admin_password = NOVA_PASS

[DEFAULT]
...
my_ip = 150.###.###.170

[DEFAULT]
```

```
:1970
vncserver_listen = 150.###.###.170
vncserver_proxyclient_address = 150.###.###.170
[glance]
...
host = controller

[DEFAULT]
:1484
verbose = True
```

데이터베이스 초기화 및 서비스 시작

```
# su -s /bin/sh -c "nova-manage db sync" nova
# systemctl enable openstack-nova-api.service openstack-nova-cert.service ₩
  openstack-nova-consoleauth.service openstack-nova-scheduler.service ₩
  openstack-nova-conductor.service openstack-nova-novncproxy.service
# systemctl start openstack-nova-api.service openstack-nova-cert.service ₩
  openstack-nova-consoleauth.service openstack-nova-scheduler.service ₩
  openstack-nova-conductor.service openstack-nova-novncproxy.service
```

Compute 서비스 사용 포트는 다음과 같다.

서비스	포트
nova-novncproxy	6080
nova-xvpvncproxy	6081
nova-objectstore	3333
nova-api	8773,8774
nova-api-metadata	8775

나. 계산 노드 설치

패키지 설치

```
# yum install -y   openstack-nova-compute   sysfsutils
```

환경설정 파일 수정

```
# vi  /etc/nova/nova.conf

[DEFAULT]
...
:180
rpc_backend = rabbit
:79
rabbit_host = controller
:96
```

```
rabbit_password = RABBIT_PASS


[DEFAULT]
:530
auth_strategy = keystone

[keystone_authtoken]
:2614
auth_uri = http://controller:5000/v2.0
identity_uri = http://controller:35357
admin_tenant_name = service
admin_user = nova
admin_password = NOVA_PASS

[DEFAULT]
:245
my_ip = 150.###.###.171
또는
my_ip = 150.###.###.172

[DEFAULT]
...
:1977
vnc_enabled = True
:1971
vncserver_listen = 0.0.0.0
vncserver_proxyclient_address = 150.###.###.171
또는
vncserver_proxyclient_address = 150.###.###.172
novncproxy_base_url = http://controller:6080/vnc_auto.html

[glance]
...
:2413
host = controller

[DEFAULT]
...
verbose = True
```

서비스 시작하기

```
# systemctl enable  libvirtd.service  openstack-nova-compute.service
```

```
# systemctl start   libvirtd.service  openstack-nova-compute.service
# systemctl status   libvirtd.service
# systemctl status   openstack-nova-compute.service
```

서비스 상태 확인하기

```
# source admin-openrc.sh ; nova service-list
```

5. 데쉬보드 (horizon) 설치

참고 http://docs.openstack.org/juno/install-guide/install/yum/content/ch_horizon.html

패키지 설치

```
# yum install -y  openstack-dashboard  httpd mod_wsgi memcached python-memcached
```

설정 파일 수정

```
# cd  /etc/openstack-dashboard/
# vi  /etc/openstack-dashboard/local_settings

OPENSTACK_HOST = "controller"

ALLOWED_HOSTS = ['*']

# Configure the memcached session storage service:
CACHES = {
   'default': {
       'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
       'LOCATION': '127.0.0.1:11211',
   }
}
TIME_ZONE = "UTC"
```

서비스 시작하기

```
# systemctl enable httpd.service  memcached.service
# systemctl start  httpd.service  memcached.service
# systemctl status httpd.service
# systemctl status memcached.service
```

서버 경로의 /dashboard 경로로 접속 http://150.###.###.170/dashboard

로그인은 admin 계정으로 위에서 설정한 암호로 한다.

6. 블록 스토리지 서비스(cinder) 설치

데이터베이스 권한 설정

```
# echo "CREATE DATABASE cinder;" | db
# export CINDER_DBPASS=***********
# echo "GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY '$CINDER_DBPASS';" | db
# echo "GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' IDENTIFIED BY '$CINDER_DBPASS';" | db
```

user, tenant, service 생성

```
# keystone user-create --name cinder --pass $CINDER_PASS
# keystone user-role-add --user cinder --tenant service --role admin
# keystone service-create --name cinder --type volume  --description "OpenStack Block Storage"
# keystone service-create --name cinderv2 --type volumev2 --description "OpenStack Block Storage"
# keystone endpoint-create ₩
  --service-id $(keystone service-list | awk '/ volume / {print $2}') ₩
  --publicurl http://controller:8776/v1/%₩(tenant_id₩)s ₩
  --internalurl http://controller:8776/v1/%₩(tenant_id₩)s ₩
  --adminurl http://controller:8776/v1/%₩(tenant_id₩)s ₩
  --region regionOne
# keystone endpoint-create ₩
  --service-id $(keystone service-list | awk '/ volumev2 / {print $2}') ₩
  --publicurl http://controller:8776/v2/%₩(tenant_id₩)s ₩
```

```
  --internalurl http://controller:8776/v2/%₩(tenant_id₩)s ₩
  --adminurl http://controller:8776/v2/%₩(tenant_id₩)s ₩
  --region regionOne
```

패키지 설치 및 설정

```
# yum install -y openstack-cinder python-cinderclient python-oslo-db
# vi /etc/cinder/cinder.conf

[database]
...
connection = mysql://cinder:cinderpass@controller/cinder

[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controller
rabbit_password=******

[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controller:5000/v2.0
identity_uri = http://controller:35357

admin_tenant_name = service
admin_user = cinder
admin_password=CINDER_PASS

[DEFAULT]
...
my_ip = 10.0.0.11

[DEFAULT]
verbose = True
debug=true
```

데이터베이스 초기화 및 서비스 시작

```
# su -s /bin/sh -c "cinder-manage db sync" cinder
# systemctl enable openstack-cinder-api.service openstack-cinder-scheduler.service
# systemctl start openstack-cinder-api.service openstack-cinder-scheduler.service
```

서비스	포트
cinder-api	8776

7. 네트워크 서비스 (neutron) 설치

참고 http://docs.openstack.org/juno/install-guide/install/yum/content/section_neutron-networking.html

가. 컨트롤러 노드 설치

```
# yum install -y openstack-neutron openstack-neutron-ml2 python-neutronclient
```

데이터베이스 설정

```
mysql>  CREATE DATABASE neutron;
mysql>  GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' ₩
  IDENTIFIED BY 'NEUTRON_DBPASS';
mysql>  GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' ₩
  IDENTIFIED BY 'NEUTRON_DBPASS';
```

서비스 tenant 아이디 구하기

```
# export SERVICE_TENANT_ID=`keystone tenant-get service | grep id | awk '{ print $4 }'`
```

tenant, user, service 생성

```
# keystone user-create --name neutron --pass NEUTRON_PASS
# keystone user-role-add --user neutron --tenant service --role admin
# keystone service-create --name neutron --type network ₩
  --description "OpenStack Networking"
# keystone endpoint-create ₩
  --service-id $(keystone service-list | awk '/ network / {print $2}') ₩
  --publicurl http://controller:9696 ₩
  --adminurl http://controller:9696 ₩
  --internalurl http://controller:9696 ₩
  --region regionOne
```

```
# vi /etc/neutron/neutron.conf

[database]
...
connection = mysql://neutron:NEUTRON_DBPASS@controller/neutron

[DEFAULT]
...
```

```
rpc_backend = rabbit
rabbit_host = controller
rabbit_password = RABBIT_PASS

[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controller:5000/v2.0
identity_uri = http://controller:35357
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS

[DEFAULT]
...
core_plugin = ml2
neutron_plugin_config="/etc/neutron/plugins/ml2/ml2_conf.ini"
service_plugins = router
allow_overlapping_ips = True

[DEFAULT]
...
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
nova_url = http://controller:8774/v2
nova_admin_auth_url = http://controller:35357/v2.0
nova_region_name = regionOne
nova_admin_username = nova
nova_admin_tenant_id = SERVICE_TENANT_ID
nova_admin_password = NOVA_PASS
```

```
# vi /etc/neutron/plugins/ml2/ml2_conf.ini

[ml2]
...
type_drivers = gre
tenant_network_types = gre
mechanism_drivers = openvswitch

[ml2_type_gre]
```

```
...
tunnel_id_ranges = 1:1000

[securitygroup]
...
enable_security_group = True
enable_ipset = True
firewall_driver = neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver

[ovs]
local_ip = 10.0.0.11
enable_tunneling = True
bridge_mappings = external:br-ex

[agent]
tunnel_types = gre
```

```
# vi /etc/nova/nova.conf

[DEFAULT]
...
network_api_class = nova.network.neutronv2.api.API
security_group_api = neutron
linuxnet_interface_driver = nova.network.linux_net.LinuxOVSInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver

[neutron]
...
url = http://controller:9696
auth_strategy = keystone
admin_auth_url = http://controller:35357/v2.0
admin_tenant_name = service
admin_username = neutron
admin_password = NEUTRON_PASS
```

```
# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
# su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf ₩
  --config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade juno" neutron
```

```
# systemctl restart ₩
  openstack-nova-api.service ₩
  openstack-nova-scheduler.service ₩
  openstack-nova-conductor.service
```

```
# systemctl enable neutron-server.service
# systemctl start  neutron-server.service
```

서비스	포트
neutron	9696

나. 네트워크 노드 설정

```
# vi /etc/sysctl.conf
net.ipv4.ip_forward=1
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
# sysctl -p
# sysctl -w "net.ipv4.conf.all.rp_filter=0"
```

패키지 설치하기

```
# yum install -y  openstack-neutron  openstack-neutron-ml2  openstack-neutron-openvswitch
# vi /etc/neutron/neutron.conf

[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controller
rabbit_password = RABBIT_PASS

[DEFAULT]
auth_strategy = keystone

[keystone_authtoken]
auth_uri = http://controller:5000/v2.0
identity_uri = http://controller:35357
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS

[DEFAULT]
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True

[DEFAULT]
...
verbose = True
```

```
# vi /etc/neutron/plugins/ml2/ml2_conf.ini

[ml2]
...
type_drivers = gre
tenant_network_types = gre
mechanism_drivers = openvswitch

[ml2_type_gre]
...
tunnel_id_ranges = 1:1000

[securitygroup]
...
enable_security_group = True
enable_ipset = True
firewall_driver = neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver

[ovs]
...
local_ip = 10.0.0.11
enable_tunneling = True
bridge_mappings = external:br-ex

[agent]
...
tunnel_types = gre
```

```
# vi /etc/neutron/l3_agent.ini
[DEFAULT]
...
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
use_namespaces = True
external_network_bridge = br-ex
router_delete_namespaces = True

[DEFAULT]
...
debug = True
```

```
# vi /etc/neutron/dhcp_agent.ini
```

```
[DEFAULT]
...
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
use_namespaces = True
dhcp_delete_namespaces = True

[DEFAULT]
debug = True
```

```
# vi /etc/neutron/metadata_agent.ini
[DEFAULT]
auth_url = http://controller:5000/v2.0
auth_region = regionOne
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS

[DEFAULT]
nova_metadata_ip = controller

[DEFAULT]
...
metadata_proxy_shared_secret = METADATA_SECRET

[DEFAULT]
...
debug = True
```

```
# systemctl enable openvswitch.service
# systemctl start  openvswitch.service
# ovs-vsctl add-br br-ex
# ovs-vsctl  list-br
# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
```

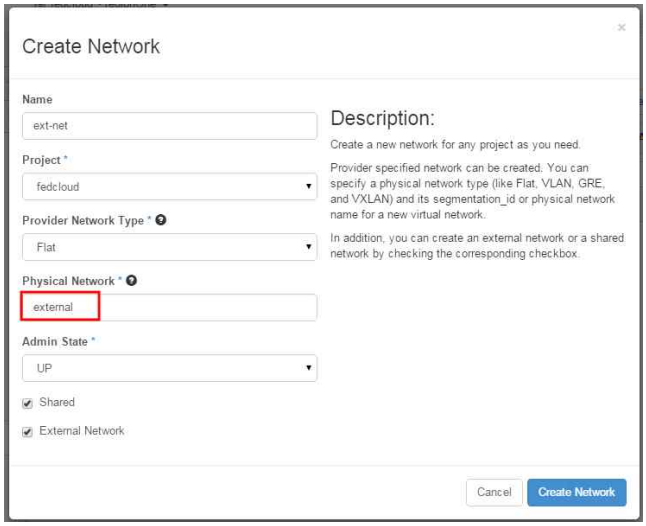
```
# cp /usr/lib/systemd/system/neutron-openvswitch-agent.service ₩
  /usr/lib/systemd/system/neutron-openvswitch-agent.service.orig
# sed -i 's,plugins/openvswitch/ovs_neutron_plugin.ini,plugin.ini,g' ₩
  /usr/lib/systemd/system/neutron-openvswitch-agent.service

# neutron agent-list
+--------------------------------------+--------------------+--------------------+-------+----------------+---------------------------+
| id                                   | agent_type         | host               | alive | admin_state_up | binary                    |
+--------------------------------------+--------------------+--------------------+-------+----------------+---------------------------+
```

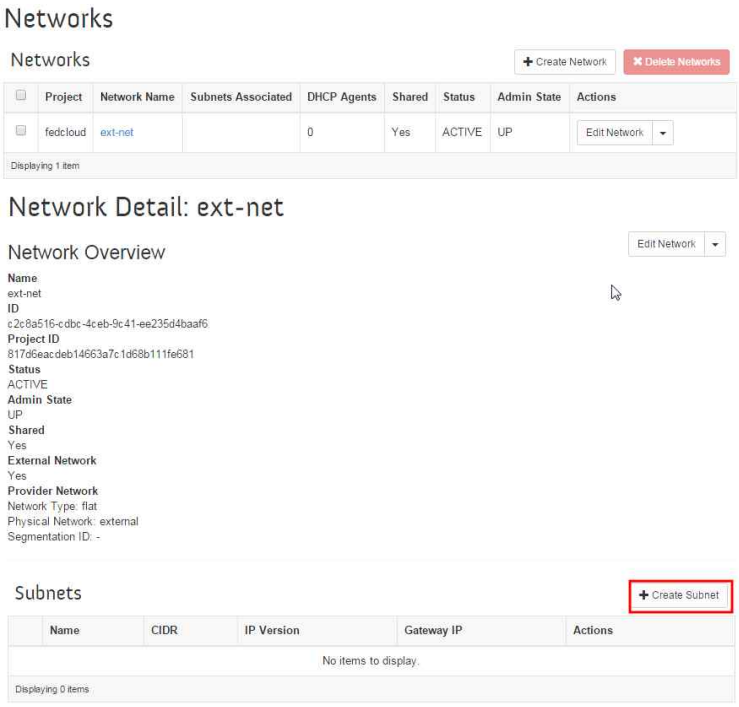
```
| 06642e03-4d4a-43ce-8d0e-84ab43d9d75a | DHCP agent         | fccont.kisti.re.kr  | :-)   | True          | neutron-dhcp-agent        |
| 37719016-2db2-4301-a35f-51248c0fd3c3 | Open vSwitch agent | fccont.kisti.re.kr  | :-)   | True          | neutron-openvswitch-agent |
| 9cec429c-0e1f-4dad-9e37-ff34363cd816 | Metadata agent     | fccont.kisti.re.kr  | :-)   | True          | neutron-metadata-agent    |
| df83dcd3-0318-4e65-bfe7-11bd9310e417 | Open vSwitch agent | fccomp2.kisti.re.kr | :-)   | True          | neutron-openvswitch-agent |
| fbf1fa1d-0c0f-4cb4-8b80-6c687637425f | Open vSwitch agent | fccomp1.kisti.re.kr | :-)   | True          | neutron-openvswitch-agent |
| fd20873b-5330-42da-9b3d-4c7b36227828 | L3 agent           | fccont.kisti.re.kr  | :-)   | True          | neutron-l3-agent          |
+--------------------------------------+--------------------+---------------------+-------+---------------+---------------------------+
```

다. 네트워크 구성하기

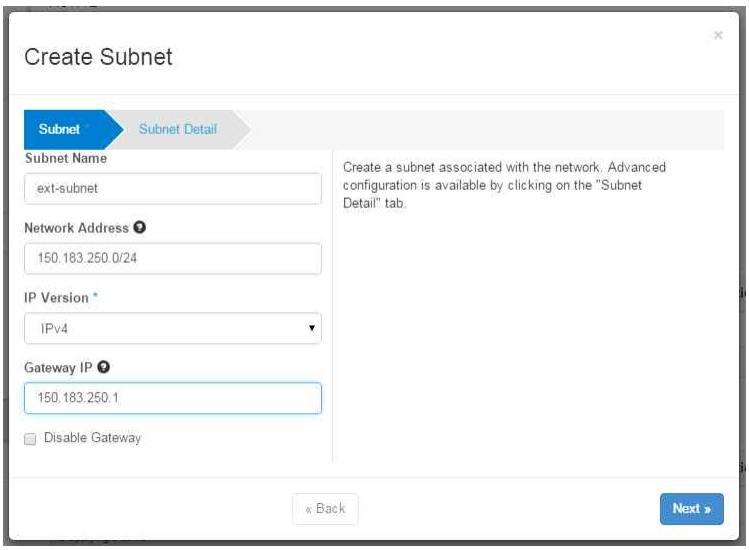
네트워크 구성은 데쉬보드를 이용하여 하였다.



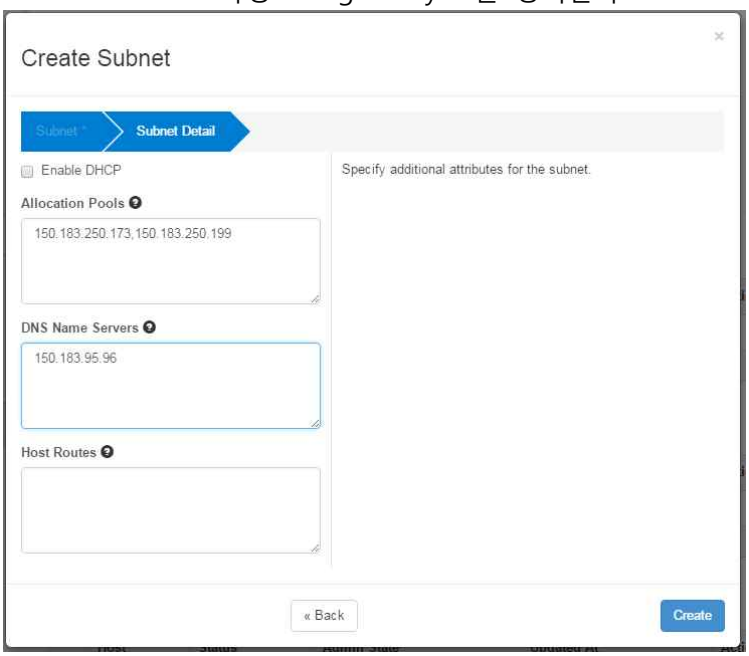
외부 네트워크를 만들어준다. ext-net



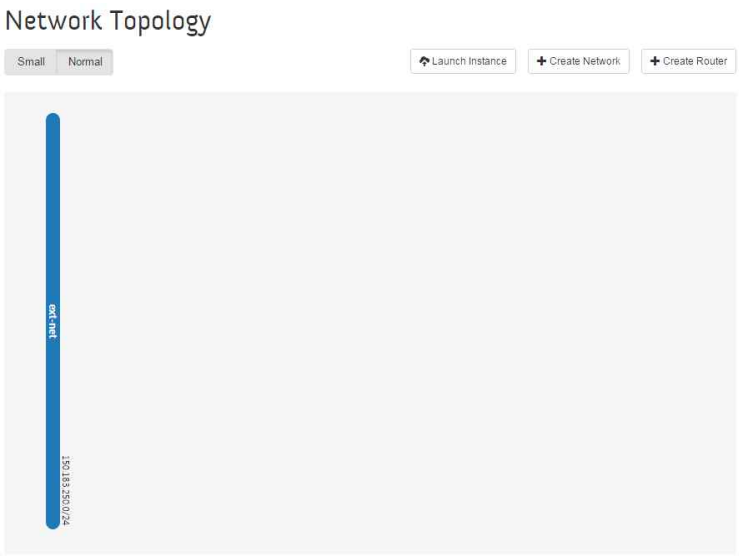
만들어진 ext-net에 subnet을 만들어 준다.



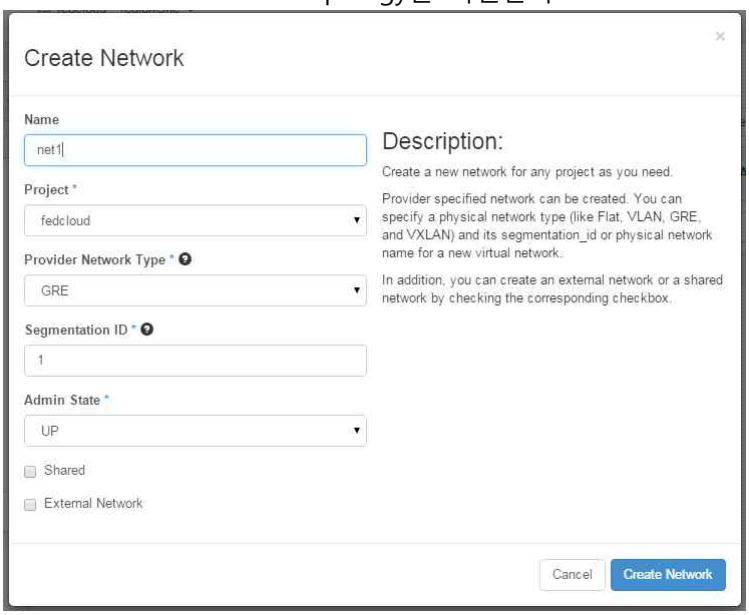
subnet 속성으로 gateway IP를 명시한다.



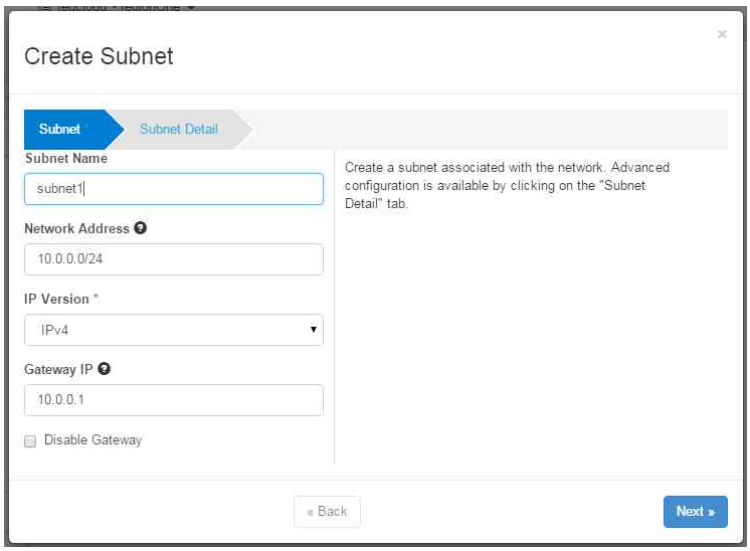
subnet 속성으로 allocation pool을 지정하여 준다.



network topology를 확인한다.



가상의 내부 네트워크를 만들어준다.

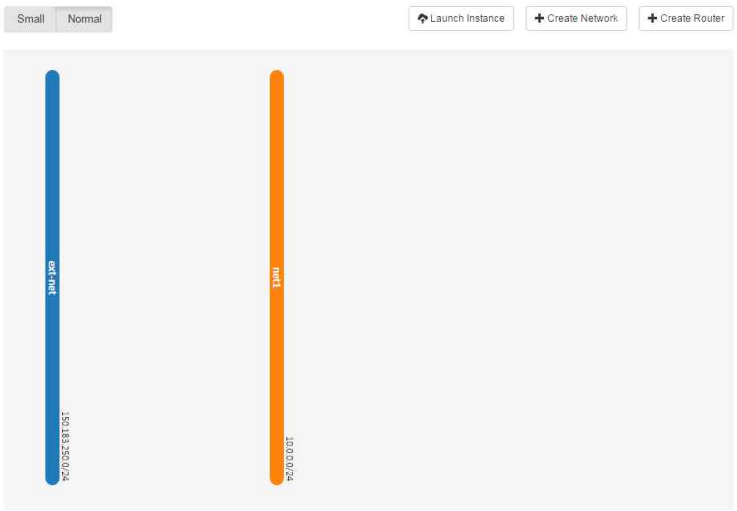


내부 네트워크의 network address와 gateway IP를 명시한다.

Networks

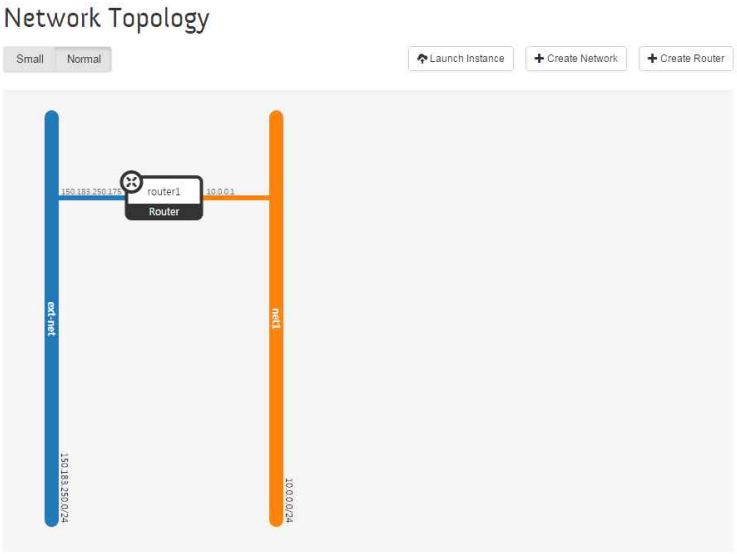
생성된 네트워크는 외부와 내부 2개가 만들어 짐

Network Topology

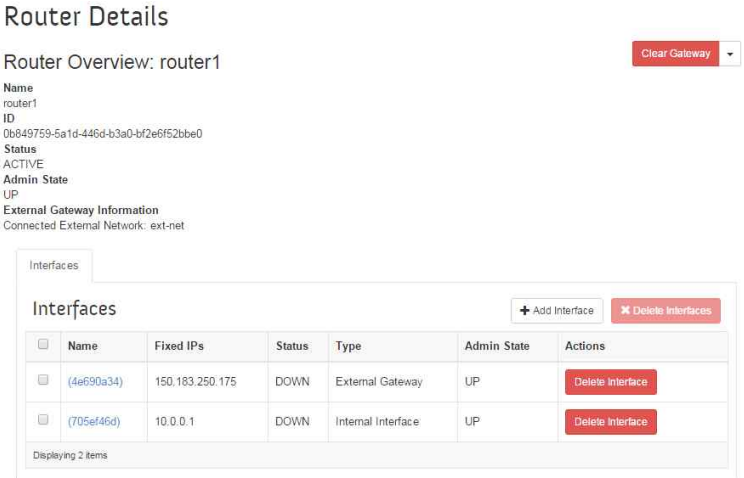


network topology를 확인한다.

라우터를 만들어 준다.



네트워크 토폴러지를 확인한다.



라우터 속성 확인

8. 인스턴스 실행하기

가. 커맨드 라인으로 인스턴스 생성하기

ssh key 생성하기

```
# ssh-keygen -t rsa
# nova keypair-add --pub_key   ~/.ssh/id_rsa.pub   demo-key
# nova keypair-list
+----------+-------------------------------------------------+
| Name     | Fingerprint                                     |
+----------+-------------------------------------------------+
| demo-key | dd:c2:72:32:f3:d1:90:50:55:2e:20:70:06:38:80:ba |
+----------+-------------------------------------------------+
```

flavor 조회하기 (가상머신 인스턴스의 CPU개수, RAM 크기를 정의)

```
# nova flavor-list
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0         | True      |
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0         | True      |
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0         | True      |
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0         | True      |
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0         | True      |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
```

인스턴스에서 사용할 이미지 ID 구하기

```
# nova image-list | tee output
+--------------------------------------+--------------+--------+--------+
| ID                                   | Name         | Status | Server |
+--------------------------------------+--------------+--------+--------+
| e0ded5b7-b533-4a2c-a77a-63c101f77583 | CirrOS 0.3.1 | ACTIVE |        |
+--------------------------------------+--------------+--------+--------+
# export ID=`grep CirrOS outpu | awk '{ print $2 }'`
```

인스턴스로 SSH와 ping을 위해서 security group 규칙을 설정하기

```
# nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
# nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
+-------------+-----------+---------+-----------+--------------+
| IP Protocol | From Port | To Port | IP Range  | Source Group |
+-------------+-----------+---------+-----------+--------------+
| tcp         | 22        | 22      | 0.0.0.0/0 |              |
+-------------+-----------+---------+-----------+--------------+
+-------------+-----------+---------+-----------+--------------+
| IP Protocol | From Port | To Port | IP Range  | Source Group |
+-------------+-----------+---------+-----------+--------------+
| icmp        | -1        | -1      | 0.0.0.0/0 |              |
+-------------+-----------+---------+-----------+--------------+
```

```
# nova secgroup-list
+----+---------+-------------+
| Id | Name    | Description |
+----+---------+-------------+
| 1  | default | default     |
+----+---------+-------------+
```

```
# nova   secgroup-list-rules   default
+-------------+-----------+---------+-----------+--------------+
| IP Protocol | From Port | To Port | IP Range  | Source Group |
+-------------+-----------+---------+-----------+--------------+
| tcp         | 22        | 22      | 0.0.0.0/0 |              |
| icmp        | -1        | -1      | 0.0.0.0/0 |              |
+-------------+-----------+---------+-----------+--------------+
```

인스턴스 실행하기

형식: nova boot --flavor flavorType --key_name keypairName --image ID newInstanceName

```
# nova boot --flavor m1.small  ₩
   --image $IMAGE_ID  --nic net-id=$NET_ID ₩
   --security-group default --key-name demo-key   demo-instance1 > output ; cat output
+--------------------------------------+----------------------------------------------------------+
| Property                             | Value                                                    |
+--------------------------------------+----------------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL                                                   |
| OS-EXT-AZ:availability_zone          | nova                                                     |
| OS-EXT-SRV-ATTR:host                 | -                                                        |
| OS-EXT-SRV-ATTR:hypervisor_hostname  | -                                                        |
| OS-EXT-SRV-ATTR:instance_name        | instance-00000919                                        |
| OS-EXT-STS:power_state               | 0                                                        |
| OS-EXT-STS:task_state                | scheduling                                               |
| OS-EXT-STS:vm_state                  | building                                                 |
| OS-SRV-USG:launched_at               | -                                                        |
| OS-SRV-USG:terminated_at             | -                                                        |
| accessIPv4                           |                                                          |
| accessIPv6                           |                                                          |
| adminPass                            | S4sDeMFWeaWZ                                             |
| config_drive                         |                                                          |
| created                              | 2015-10-05T02:20:30Z                                     |
| flavor                               | m1.small (2)                                             |
| hostId                               |                                                          |
| id                                   | 9763a64e-9cf7-4004-8593-a9bf5e417e2b                     |
| image                                | cirros-0.3.3-x86_64 (9eabafdb-b223-4144-aff5-a1279b53e4f4) |
| key_name                             | demo-key                                                 |
| metadata                             | {}                                                       |
| name                                 | demo-instance1                                           |
| os-extended-volumes:volumes_attached | []                                                       |
| progress                             | 0                                                        |
| security_groups                      | default                                                  |
| status                               | BUILD                                                    |
```

```
| tenant_id                            | 817d6eacdeb14663a7c1d68b111fe681                                 |
| updated                              | 2015-10-05T02:20:30Z                                             |
| user_id                              | 32df0da96be34486bdb039c587088717                                 |
+--------------------------------------+------------------------------------------------------------------+
```

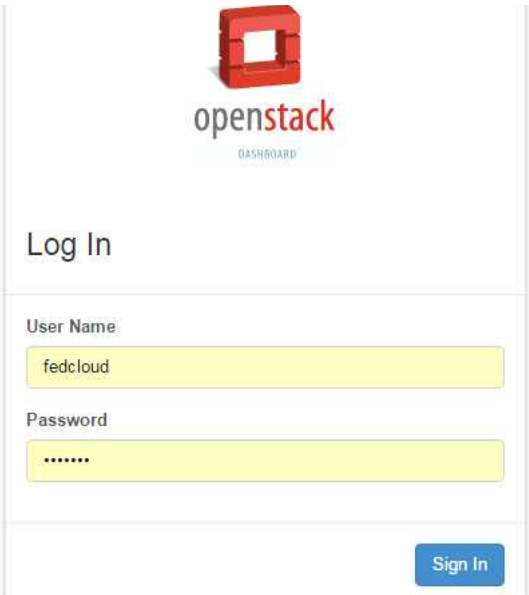
가상머신 인스턴스 리스트

```
# nova list
+--------------------------------------+---------------+--------+------------+-------------+---------------+
| ID                                   | Name          | Status | Task State | Power State | Networks      |
+--------------------------------------+---------------+--------+------------+-------------+---------------+
| 87be4ced-c566-4257-b945-277c23e13397 | demo-instance1 | BUILD  | spawning   | NOSTATE     | net1=10.0.0.9 |
.........
| 87be4ced-c566-4257-b945-277c23e13397 | demo-instance1 | ACTIVE | -          | Running     | net1=10.0.0.9 |
+--------------------------------------+---------------+--------+------------+-------------+---------------+
```

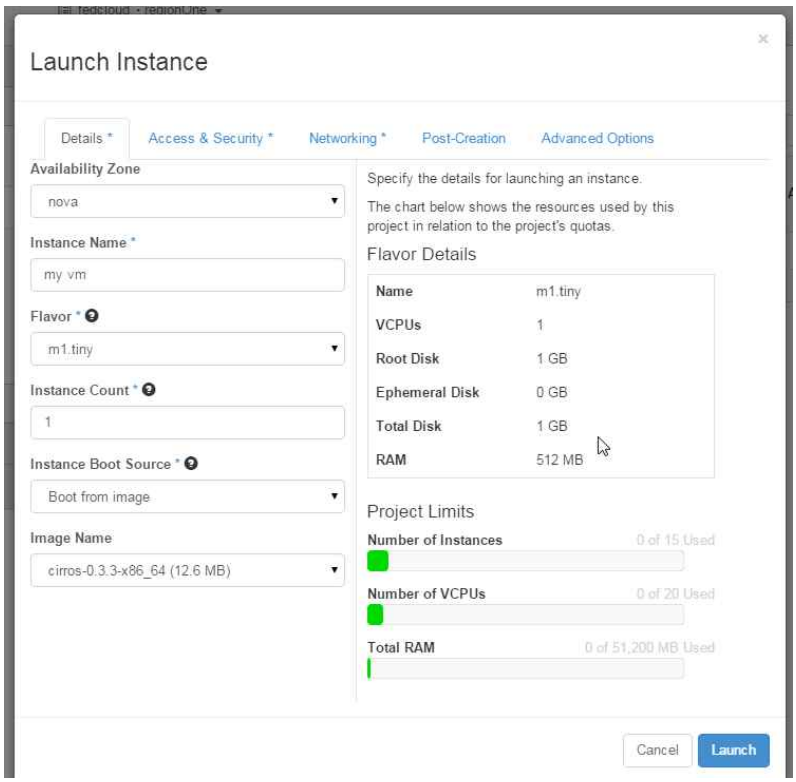
계산 노드에서 virsh list 로 생성된 인스턴스를 확인

```
# virsh list
 Id    Name                           State
----------------------------------------------------
 1     instance-00000001              running
```

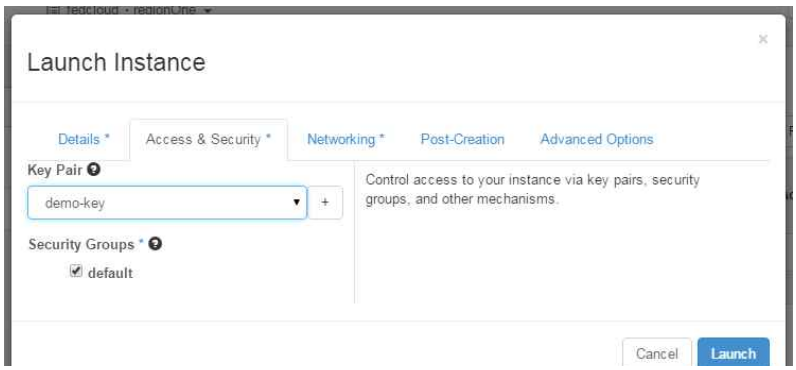
나. 데쉬보드를 이용하여 인스턴스 생성하기



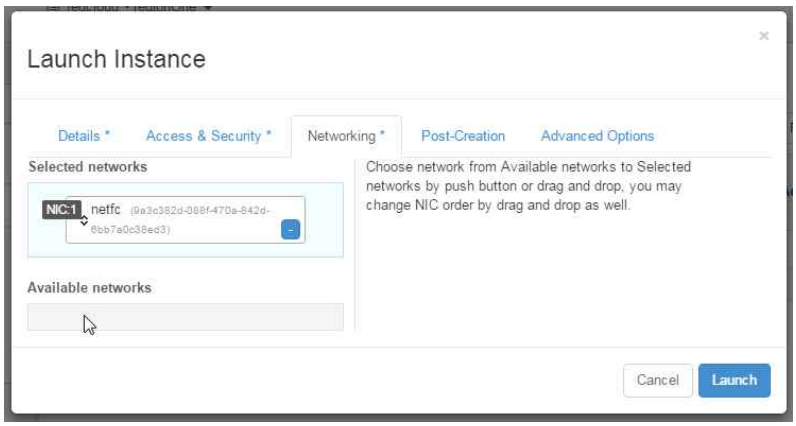
데쉬보드 로그인



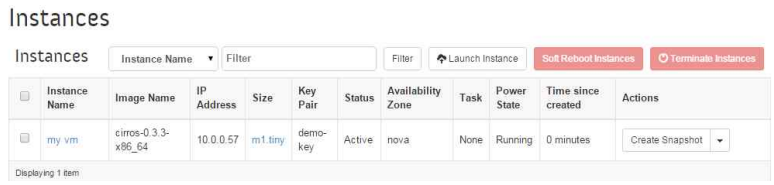
인스턴스 실행하기



키 패어 설정

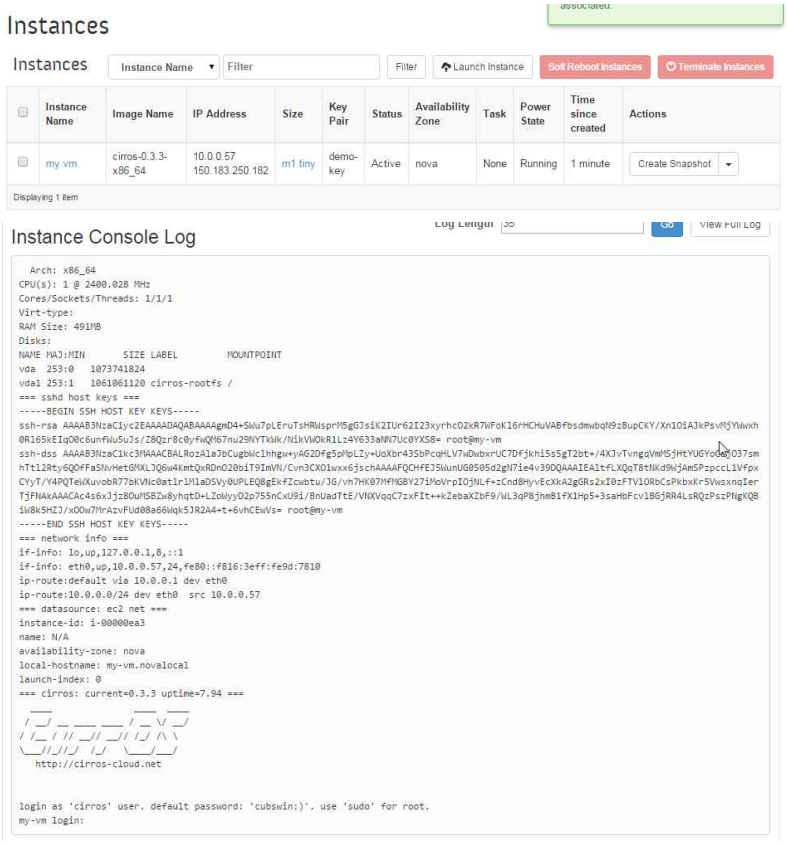


네트워크 선택

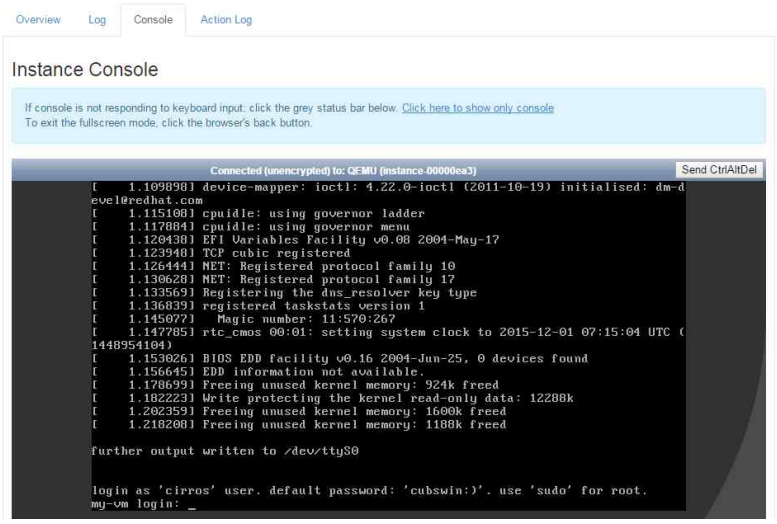


생성된 인스턴스 목록

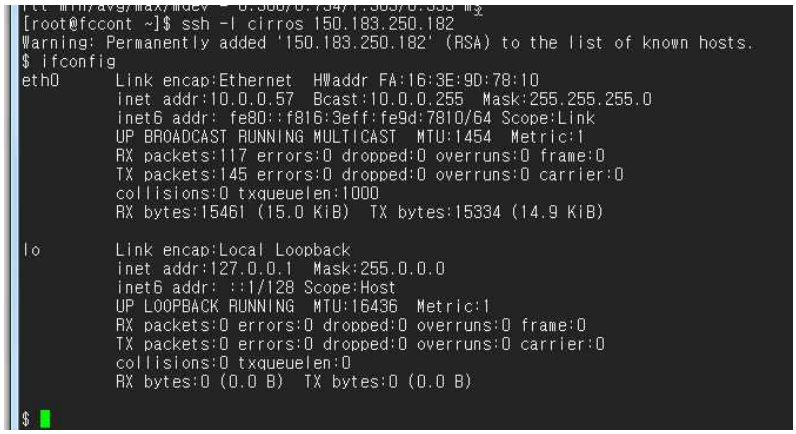
플로팅 IP 할당



인스턴스 콘솔 로그 조회



인트턴스 콘솔 접속



외부에서 네트워크를 통하여 인스턴스 접속 성공

제3절 Federated Cloud 소프트웨어 설치

FedCloud를 위한 CMS(Cloud Management Frameworks) 설치를 위해서는 타스크 포스 사이트의 문서[13]를 참고한다. 현재 지원하는 오픈스택 버전은 Icehouse와 Juno이며 Havana 버전은 향후 지원되지 않을 예정이다. keystone 서비스는 필수이며, OCCI를 이용한 VM 관리 기능을 위해서 nova, cinder, glance, neutron을 설치해야한다. nova-network 는 neutron 대신 사용될수 있지만 권장사항은 아니다. CDMI 접근을 이용하고자 한다면 swift를 설치해야 한다.

OpenStack 과 함께 다음 소프트웨어를 설치해야 한다.

- OCCI-OS : OpenStack을 위한 OCCI API
- keystone-voms : VOMS proxy를 이용한 인증
- cASO : OpenStack의 어카운팅 정보를 수집하는 스크립트
- BDII : 사이트 정보와 설설을 EGI 정보 시스템에 등록
- vmcatcher : EGI App DB를 체크하여 업데이트된 이미지가 존재하는지 체크하고 다운로드하여 OpenStack에서 이용할 수 있도록 함

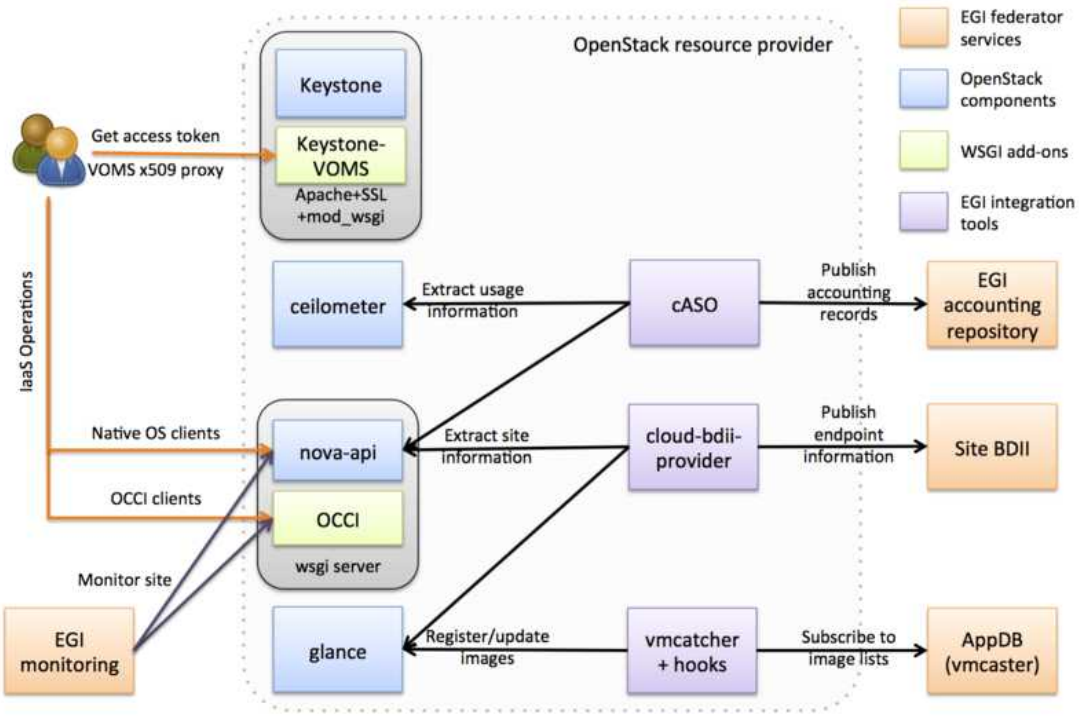


그림 29. OpenStack 기반의 FedCloud 서비스 구성도

1. VOMS 설정

VOMS [14] 서비스는 RFC 3820 에 기반은 X.509 프록시를 만들수 있는데, -rfc 옵션을 커맨드 라

13) https://wiki.egi.eu/wiki/Fedcloud-tf:ResourceProviders:OpenStack#OpenStack
14) http://en.wikipedia.org/wiki/VOMS

인에 사용해야 한다. 보통의 X.509 인증서대신 이 프록시는 설정된 keystone 서버에 인증용도로 사용될 수 가 있다. keystone VOMS authentication module 소스는 github [15]에 올려져 있다.

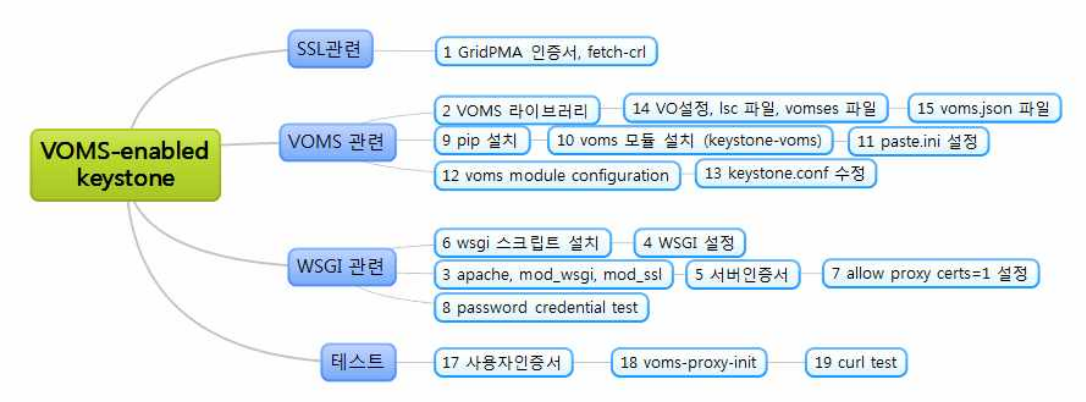


그림 30 keysont-voms 설치 과정

EUgridPMA CA certificates 설치

```
# cd   /etc/yum.repos.d
# wget http://repository.egi.eu/sw/production/cas/1/current/repo-files/EGI-trustanchors.repo
# cat EGI-trustanchors.repo
# yum install -y   ca-policy-egi-core
```

fetch-crl 설치

```
# yum install -y fetch-crl
# /usr/sbin/fetch-crl -h
# /sbin/chkconfig fetch-crl-cron on
# /sbin/service fetch-crl-cron start
# touch   /etc/sysconfig/fetch-crl
# time /usr/sbin/fetch-crl
```

VOMS 라이브러리 설치

```
# yum -y install voms
# rpm -qf /usr/lib64/libvomsapi.so.1
voms-2.0.12-3.el7.x86_64
```

keystone 설정 수정

```
# vi /etc/keystone/keystone.conf

[paste_deploy]
# Name of the paste configuration file that defines the available pipelines
```

15) http://ifca.github.io/keystone-voms/

```
# config_file = /usr/share/keystone/keystone-dist-paste.ini
config_file = /etc/keystone/keystone-paste.ini
:wq
# cp /usr/share/keystone/keystone-dist-paste.ini /etc/keystone/keystone-paste.ini
```

아파치 웹서버 설치와 설정

Apache 에 WSGI 와 mod_ssl 을 설정한다.

```
# yum -y install  mod_wsgi  mod_ssl  httpd   php
```

아파치 서버를 다음과 같이 설정한다.

포트번호는 기존 keystone 과 중복을 피하기 위해 5001, 35358 로 한다.

```
# cp   /etc/httpd/conf/httpd.conf   /etc/httpd/conf/httpd.conf.orig
# vi /etc/httpd/conf/httpd.conf
<Directory />
    AllowOverride none
    Require all granted
</Directory>

WSGISocketPrefix   /var/log/httpd

Listen 5001
<VirtualHost _default_:5001>
    LogLevel     warn
    ErrorLog    /var/log/httpd/5001_error.log
    CustomLog   /var/log/httpd/5001_ssl_access.log combined

    SSLEngine               on
    SSLCertificateFile      /etc/ssl/certs/hostcert.pem
    SSLCertificateKeyFile   /etc/ssl/private/hostkey.pem
    SSLCACertificatePath    /etc/grid-security/certificates
    SSLCARevocationPath     /etc/grid-security/certificates
    SSLVerifyClient         optional
    SSLVerifyDepth          10
    SSLProtocol             all -SSLv2
    SSLCipherSuite          ALL:!ADH:!EXPORT:!SSLv2:RC4+RSA:+HIGH:+MEDIUM:+LOW
    SSLOptions              +StdEnvVars +ExportCertData

    WSGIDaemonProcess keystone user=keystone group=keystone processes=1 threads=1
    WSGIScriptAlias /  /usr/lib/cgi-bin/keystone/main
    WSGIProcessGroup keystone
</VirtualHost>

Listen 35358
<VirtualHost _default_:35358>
    LogLevel    warn
    ErrorLog    /var/log/httpd/35358_error.log
    CustomLog   /var/log/httpd/35358_ssl_access.log combined
```

```
    SSLEngine               on
    SSLCertificateFile      /etc/ssl/certs/hostcert.pem
    SSLCertificateKeyFile   /etc/ssl/private/hostkey.pem
    SSLCACertificatePath    /etc/grid-security/certificates
    SSLCARevocationPath     /etc/grid-security/certificates
    SSLVerifyClient         optional
    SSLVerifyDepth          10
    SSLProtocol             all -SSLv2
    SSLCipherSuite          ALL:!ADH:!EXPORT:!SSLv2:RC4+RSA:+HIGH:+MEDIUM:+LOW
    SSLOptions              +StdEnvVars +ExportCertData

    WSGIDaemonProcess   keystoneapi user=keystone group=keystone processes=1 threads=1
    WSGIScriptAlias     / /usr/lib/cgi-bin/keystone/admin
    WSGIProcessGroup    keystoneapi
</VirtualHost>
```

SSLVerifyClient 옵션은 optional 로 했기 때문에 VOMS 프록시가 없는 사람은 keystone credential 로 인증이 가능하다.

Keystone 을 WSGI application 으로 실행하기 위해 WSGI 스크립트가 필요하다.
이것은 keystone 레포지터리에 포함되어 있다.
https://github.com/openstack/keystone/blob/stable/havana/httpd/keystone.py
이 스크립트를 저장하고 다음과 같이 링크를 만들어 준다.

```
# mkdir -p /usr/lib/cgi-bin/keystone/
# cd /usr/lib/cgi-bin/keystone/
#  wget https://raw.githubusercontent.com/openstack/keystone/stable/juno/httpd/keystone.py
#  ln -s /usr/lib/cgi-bin/keystone/keystone.py /usr/lib/cgi-bin/keystone/main
#  ln -s /usr/lib/cgi-bin/keystone/keystone.py /usr/lib/cgi-bin/keystone/admin
```

/etc/grid-security 에 인증서를 복사해 준다.

```
# cd   /etc/grid-security
# chmod 644 hostcert.pem
# chmod 644 hostkey.pem
# cd /etc/ssl ; ln -s ../pki/tls/private
# cp  /etc/grid-security/hostcert.pem  /etc/ssl/certs/hostcert.pem
# cp  /etc/grid-security/hostkey.pem   /etc/ssl/private/hostkey.pem
```

X.509 프록시 인증서를 아파치 서버에서 이용하기 위해서 환경 변수를 설정한다.

```
# vi /etc/sysconfig/httpd
...
OPENSSL_ALLOW_PROXY_CERTS=1
:wq
```

httpd 서버를 시작한다.

```
# /etc/init.d/httpd  restart
```

curl 명령을 이용하여 설정 확인 방법

```
# export CURL_OPTION="--capath /etc/grid-security/certificates"
# curl $CURL_OPTION -i -X POST https://fccont.kisti.re.kr:35358/v2.0/tokens -H "Content-Type: application/json" -H "User-Agent: python-keystoneclient" -d '{"auth": {"tenantName": "admin", "passwordCredentials": {"username": "admin", "password": "admin"}}}'
```

SQL Token driver 설정

```
# grep "driver = " /etc/keystone/keystone.conf | grep Token
# vi /etc/keystone/keystone.conf
[token]
driver = keystone.token.backends.sql.Token
```

VOMS module 설치

python-pip 설치

```
# yum install -y python-pip.noarch   git   swig
# yum install -y  openssl-devel   python-devel
# yum -y groupinstall "Development Tools"
```

Keystone VOMS module 설치

```
# pip uninstall keystone-voms
# cd /root
# git clone git://github.com/IFCA/keystone-voms.git -b stable/juno
# cd keystone-voms
# pip install . 2>&1 | tee i.log
...
Successfully installed python-keystone-voms
Cleaning up...
```

설치 확인

```
# pip  list | grep keystone
keystone (2014.2.2)
keystone-voms (2014.2.0)
keystonemiddleware (1.2.0)
python-keystoneclient (0.11.1)
```

설치경로

```
/usr/lib/python2.7/site-packages/keystone_voms
/usr/lib/python2.7/site-packages/python_keystoneclient-0.11.1-py2.7.egg-info
```

Keystone VOMS module 활성화 하기

```
# vi /etc/keystone/keystone.conf
config_file = /etc/keystone/keystone-paste.ini
:wq
```

```
# vi /etc/keystone/keystone-paste.ini

[filter:voms]
paste.filter_factory = keystone_voms.core:VomsAuthNMiddleware.factory
:wq

# systemctl  restart  openstack-keystone.service
```

VOMS module 설정

- vomsdir_path: Path storing the .lsc files.
- ca_path: Path where the CAs and CRLs are stored.
- voms_policy: JSON file containing the VO/tenant/role mapping.
- vomsapi_lib: Path to the voms library to use.
- autocreate_users: Whether a user should be autocreated if it does not exist.
- add_roles: Whether roles should be added to users or not.
- user_roles: list of role names to add to the users (if add_roles is True).

```
# vi /etc/keystone/keystone.conf

[voms]
vomsdir_path = /etc/grid-security/vomsdir
ca_path = /etc/grid-security/certificates
voms_policy = /etc/keystone/voms.json
vomsapi_lib = libvomsapi.so.1
autocreate_users = True
add_roles = False
user_roles = _member_
```

Allowed VOs 설정하기

FedCloud voms setting

```
# mkdir -p /etc/grid-security/vomsdir/fedcloud.egi.eu
# cd       /etc/grid-security/vomsdir/fedcloud.egi.eu
# cat >voms1.egee.cesnet.cz.lsc <<EOF
/DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=voms1.egee.cesnet.cz
/C=NL/O=TERENA/CN=TERENA eScience SSL CA
EOF
```

```
# cat >voms2.grid.cesnet.cz.lsc <<EOF
/DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=voms2.grid.cesnet.cz
/C=NL/ST=Noord-Holland/L=Amsterdam/O=TERENA/CN=TERENA eScience SSL CA 2
EOF
```

VO to local tenant mapping 설정

```
# vi /etc/keystone/voms.json
{
    "fedcloud.egi.eu": {
        "tenant": "fedcloud"
    },
    "dteam": {
        "tenant": "ops"
    },
    "ops": {
        "tenant": "ops"
    }
}
```

테스트

VOMS 클라이언트 설치

```
# yum install -y  voms-clients
```

VOMS proxy 생성

```
# su - sangwan
# mkdir .globus
# cd .globus  # 사용자 인증서 복사
# chmod 644 usercert.pem
# chmod 400 userkey.pem
```

VOMS 프록시 생성

```
# voms-proxy-init -voms fedcloud.egi.eu --rfc -dont-verify-ac
Enter GRID pass phrase:
Your identity: /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
Creating temporary proxy ..................................... Done
Contacting  voms2.grid.cesnet.cz:15002 [/DC=org/DC=terena/DC=tcs/OU=Domain Control Validate
d/CN=voms2.grid.cesnet.cz] "fedcloud.egi.eu" Done
Creating proxy ............................................. Done

Your proxy is valid until Thu May 14 05:05:45 2015
```

프록시 정보 확인

```
# voms-proxy-info -file /tmp/x509up_u1000 -all
subject   : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim/CN=105066999
```

```
issuer    : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
identity  : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
type      : RFC compliant proxy
strength  : 1024 bits
path      : /tmp/x509up_u1000
timeleft  : 11:58:46
key usage : Digital Signature, Key Encipherment, Data Encipherment
=== VO fedcloud.egi.eu extension information ===
VO        : fedcloud.egi.eu
subject   : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
issuer    : /DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=voms2.grid.cesnet.cz
attribute : /fedcloud.egi.eu/Role=NULL/Capability=NULL
timeleft  : 11:58:46
uri       : voms2.grid.cesnet.cz:15002
```

VOMS 인증 테스트

```
# export X509_USER_PROXY=/tmp/x509up_u1000
# echo '{"auth":{"voms":true}}' > data_voms
# cat /etc/grid-security/certificates/*.0 > /etc/grid-security/certificates/ca-bundle.crt
# export ENDPOINT=https://fccont.kisti.re.kr:5001/v2.0/tokens
# export CURL=/usr/local/curl/bin/curl
# $CURL  --cert $X509_USER_PROXY -X POST $ENDPOINT ₩
  $CURL_OPTS -H "Content-type: application/json" ₩
  -d @data_voms ₩
  --cacert  /etc/grid-security/certificates/ca-bundle.crt ₩
  --capath /etc/grid-security/certificates > output
# cat output | python -m json.tool
{
    "access": {
        "metadata": {
            "is_admin": 0,
            "roles": []
        },
        "serviceCatalog": [],
        "token": {
            "audit_ids": [
                "jBdpSUs0Sb6K0i4a5MqQjw"
            ],
            "expires": "2015-05-20T03:01:50Z",
            "id": "7bd3785e64ae4d6aa780a383daa28a43",
            "issued_at": "2015-05-20T02:01:50.671625"
        },
        "user": {
```

```
            "id": "aa874aa706bd435abfdb782123615a1a",
            "name": "/C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim",
            "roles": [],
            "roles_links": [],
            "username": "/C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim"
        }
    }
}
```

2. OCCI

The Open Cloud Computing Interface (OCCI) is a RESTful Protocol and API designed to facilitate interoperable access to, and query of, cloud-based resources across multiple resource providers and heterogeneous environments.

OCCI 서비스 설치

패키지 설치

```
$ yum install -y python-pip.noarch git
```

pyssf 파이썬 패키지 설치

```
# pip install pyssf
# pip list | grep pyssf
pyssf (0.4.7)
```

occi-os 소스 설치 [16]

```
# git clone git://github.com/EGI-FCTF/occi-os  -b stable/juno
# cd occi-os/
# python setup.py install
```

OpenStack 설정

```
# vi /etc/nova/api-paste.ini

########
# OCCI #
########
[composite:occiapi]
use = egg:Paste#urlmap
/: occiapppipe
[pipeline:occiapppipe]
pipeline = authtoken keystonecontext occiapp
```

16) https://github.com/IFCA/occi-os

```
[app:occiapp]
use = egg:openstackocci-havana#occi_app
```

```
# vi /etc/nova/nova.conf
...
enabled_apis=ec2,occiapi,osapi_compute,metadata
...
:wq
```

OCCI 서비스를 keystone 에 등록하기

```
# keystone service-create --name occi_api --type occi --description 'Nova OCCI Service'
# keystone service-list
# keystone endpoint-create --service_id 527a66524bba441ca13750a9b787b096 --region RegionOne ₩
  --publicurl    http://controller:8787/ ₩
  --internalurl  http://controller:8787/ ₩
  --adminurl     http://controller:8787/
```

수정후 nova-api 서비스를 재시작

```
# /etc/init.d/openstack-nova-api restart
```

OCCI 서비스 포트를 수정

```
# vi /etc/nova/nova.conf
occiapi_listen_port=9999
```

서비스를 재시작

```
# systemctl  restart openstack-nova-api.service
```

nova-api 서비스가 9999 포트를 사용중 임을 확인

```
# netstat -antp | grep ":9999" | grep LISTEN
tcp        0      0 0.0.0.0:9999        0.0.0.0:*    LISTEN      3910/python
```

사용하는 서비스 포트. (아래 포트를 iptables 방화벽에서 열어준다.)

서비스	포트
OCCI (http)	9999
OCCI (https)	8787

아파치 설정 수정

```
# yum -y install mod_proxy_html
# vi /etc/httpd/conf/httpd.conf
LoadModule rewrite_module modules/mod_rewrite.so
```

```
...
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule substitute_module /usr/lib64/httpd/modules/mod_substitute.so
LoadModule filter_module     /usr/lib64/httpd/modules/mod_filter.so
...
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so

.........

# OCCI
Listen 8787
<VirtualHost _default_:8787>
    LogLevel warn
    ErrorLog  /var/log/httpd/8787_error.log
    CustomLog /var/log/httpd/8787_ssl_access.log combined

    SSLEngine               on
    SSLCertificateFile      /etc/ssl/certs/hostcert.pem
    SSLCertificateKeyFile   /etc/ssl/private/hostkey.pem
    SSLCACertificatePath    /etc/grid-security/certificates
    SSLCARevocationPath     /etc/grid-security/certificates
    #SSLCACertificateFile    /etc/pki/tls/certs/ca-bundle.crt

    SSLVerifyClient optional
    SSLVerifyDepth 10
    SSLProtocol all -SSLv2
    SSLCipherSuite ALL:!ADH:!EXPORT:!SSLv2:RC4+RSA:+HIGH:+MEDIUM:+LOW
    SSLOptions +StdEnvVars +ExportCertData
<IfModule mod_proxy.c>
 # Do not enable proxying with ProxyRequests until you have secured your
 # server. Open proxy servers are dangerous both to your network and to the
 # Internet at large.
 ProxyRequests Off

 <Proxy *>
 Order deny,allow
 Deny from all
 #Allow from .example.com
 </Proxy>

 ProxyPass / http://fccont.kisti.re.kr:9999/ connectiontimeout=600 timeout=600
 ProxyPassReverse / http://fccont.kisti.re.kr:9999/
 FilterDeclare OCCIFILTER
 FilterProvider OCCIFILTER SUBSTITUTE resp=Content-Type $text/
 FilterProvider OCCIFILTER SUBSTITUTE resp=Content-Type $application/
 <Location />
 #AddOutputFilterByType SUBSTITUTE text/plain

 FilterChain OCCIFILTER
 Substitute s|http://fccont.kisti.re.kr:9999|https://fccont.kisti.re.kr:8787|n
 Order allow,deny
 Allow from all
 </Location>
```

```
</IfModule>
</VirtualHost>
```

rOCCI 클라이언트 설치

occi 설치방법[17]을 참고 한다.

yum repository 설정

```
# wget -O /etc/yum.repos.d/rocci-cli.repo ₩
  http://repository.egi.eu/community/software/rocci.cli/4.2.x/releases/repofiles/sl-6-x86_64.repo
```

패키지 설치

```
# yum install -y occi-cli
# ln -s  /opt/occi-cli/bin/occi  /usr/bin/occi
```

사용법

```
# occi --version
4.2.5

# occi
Missing required arguments: resource, action
Usage: occi [OPTIONS]

Options:
    -e, --endpoint URI               OCCI server URI, defults to "http://localhost:3000"
    -n, --auth METHOD                Authentication method, only: [x509|basic|digest|none], defaults to "none"
    -k, --timeout SEC                Default timeout for all HTTP connections, in seconds
    -u, --username USER              Username for basic or digest authentication, defaults to "anonymous"
    -p, --password PASSWORD          Password for basic, digest and x509 authentication
    -c, --ca-path PATH               Path to CA certificates directory, defaults to "/etc/grid-security/certificates"
    -f, --ca-file PATH               Path to CA certificates in a file
    -s, --skip-ca-check              Skip server certificate verification [NOT recommended]
    -F, --filter CATEGORY            Category type identifier to filter categories from model, must be used together with the -m option
    -x, --user-cred FILE             Path to user's x509 credentials, defaults to "/root/.globus/usercred.pem"
    -X, --voms                       Using VOMS credentials; modifies behavior of the X509 authN module
    -y, --media-type MEDIA_TYPE      Media type for client <-> server communication, only: [application/occi+json|text/plain,text/occi|text/plain|text/occi], defaults to "text/plain,text/occi"
    -r, --resource RESOURCE          Term, identifier or URI of a resource to be queried, required
    -t, --attribute ATTR             An "attribute='value'" pair, mandatory attrs for creating new resource instances: [occi.core.title]
    -T, --context CTX_VAR            A "context_variable='value'" pair for new 'compute' resource instances, only: [public_key, user_data]
    -a, --action ACTION              Action to be performed on a resource instance, required
```

17) https://wiki.egi.eu/wiki/Fedcloud-tf:CLI_Environment

```
    -M, --mixin IDENTIFIER           Identifier of a mixin, formatted as SCHEME#TERM or SHORT_SCHEME
#TERM
    -j, --link URI                   URI of an instance to be linked with the given resource, applica
ble only for action 'link'
    -g, --trigger-action ACTION      Action to be triggered on the resource, formatted as SCHEME#TER
M or TERM
    -l, --log-to OUTPUT              Log to the specified device, only: [stdout|stderr], defaults to
'stderr'
    -o, --output-format FORMAT       Output format, only: [json|plain|json_pretty|json_extended|json
_extended_pretty], defaults to "plain"
    -b, --log-level LEVEL            Set the specified logging level, only: [debug|error|fatal|info|
unknown|warn]
    -z, --examples                   Show usage examples
    -m, --dump-model                 Contact the endpoint and dump its model
    -d, --debug                      Enable debugging messages
    -h, --help                       Show this message
    -v, --version                    Show version
```

사용 예제를 보려면

```
# occi -z
# Examples
## Quick reference guide
occi --endpoint http://localhost:3000/ --action list --resource os_tpl
occi --endpoint http://localhost:3000/ --action list --resource resource_tpl
occi --endpoint http://localhost:3000/ --action describe --resource os_tpl#debian6
occi --endpoint http://localhost:3000/ --action create --resource compute --mixin os_tpl#de
bian6 --mixin resource_tpl#small --attribute occi.core.title="My rOCCI VM"
occi --endpoint http://localhost:3000/ --action delete --resource /compute/65sd4f654sf65g4-
s5fg65sfg465sfg-sf65g46sf5g4sdfg

## Listing resources
occi --endpoint http://localhost:3000/ --action list --resource compute
occi --endpoint http://localhost:3000/ --action list --resource network
occi --endpoint http://localhost:3000/ --action list --resource storage
occi --endpoint http://localhost:3000/ --action list --resource os_tpl
occi --endpoint http://localhost:3000/ --action list --resource resource_tpl

## Describing resources
occi --endpoint http://localhost:3000/ --action describe --resource compute
occi --endpoint http://localhost:3000/ --action describe --resource network
occi --endpoint http://localhost:3000/ --action describe --resource storage
occi --endpoint http://localhost:3000/ --action describe --resource os_tpl
occi --endpoint http://localhost:3000/ --action describe --resource resource_tpl

## Creating resources
occi --endpoint http://localhost:3000/ --action create [ --attribute attribute_name='attrib
ute_value' ]+ [ --mixin mixin_type#mixin_term ]+ --resource compute
occi --endpoint http://localhost:3000/ --action create [ --attribute attribute_name='attrib
ute_value' ]+ [ --mixin mixin_type#mixin_term ]+ --resource network
occi --endpoint http://localhost:3000/ --action create [ --attribute attribute_name='attrib
ute_value' ]+ [ --mixin mixin_type#mixin_term ]+ --resource storage
.....
```

OCCI 클라이언트 사용법

OCCI 클라이언트를 이용하여 VM관리 하는 방법

OS 템플릿 리스트 조회

CESNET (OpenNebula) 사이트

```
# occi --endpoint https://carach5.ics.muni.cz:11443/ --action list --resource os_tpl ₩
 --auth x509 --user-cred /tmp/x509up_u500 --voms
http://occi.carach5.ics.muni.cz/occi/infrastructure/os_tpl#uuid_monitoring_20
http://occi.carach5.ics.muni.cz/occi/infrastructure/os_tpl#uuid_egi_sl6goldenimage_cesnet_50
http://occi.carach5.ics.muni.cz/occi/infrastructure/os_tpl#uuid_generic_vm_54
....
http://occi.carach5.ics.muni.cz/occi/infrastructure/os_tpl#uuid_ubuntu_server_14_04_lts_fedcloud_dukan_84
```

사이트내의 활용할 수 있는 자원 조회하기

```
# INFN-Bari (OpenStack) https://prisma-cloud.ba.infn.it:8787/
# occi --endpoint https://prisma-cloud.ba.infn.it:8787/  --action list --resource os_tpl ₩
   --auth x509 --user-cred /tmp/x509up_u500 --voms
http://schemas.openstack.org/template/os#ec4bb03e-d6df-4964-a490-ae0ef57536e7
http://schemas.openstack.org/template/os#7440eb4b-ff1e-4059-a3fa-78112362282e
http://schemas.openstack.org/template/os#36378598-7f42-4fc5-806b-9a6df2791f20
...
http://schemas.openstack.org/template/os#c0a2f9e0-081a-419c-b9a5-8cb03b1decb5
http://schemas.openstack.org/template/os#7cfba655-f692-406f-a659-79b0224290cc
```

특정 OS templete의 정보를 얻기

```
# occi --endpoint https://prisma-cloud.ba.infn.it:8787/ --action describe ₩
  --auth x509 --user-cred /tmp/x509up_u500 --voms ₩
  --resource os_tpl#72ada03a-5694-4a79-8e7e-069516a31a59
##############################################################################################
[[ http://schemas.openstack.org/template/os#72ada03a-5694-4a79-8e7e-069516a31a59 ]]
title:        Image: Ubuntu-14.04-amd64
term:         72ada03a-5694-4a79-8e7e-069516a31a59
location:     /72ada03a-5694-4a79-8e7e-069516a31a59/
##############################################################################################
```

리소스 템플릿 정보 얻기

```
# occi --endpoint https://prisma-cloud.ba.infn.it:8787/ --action list ₩
  --resource resource_tpl --auth x509 --user-cred /tmp/x509up_u500 --voms
http://schemas.openstack.org/template/resource#1cpu-1gb-10dsk
http://schemas.openstack.org/template/resource#m1-xlarge
http://schemas.openstack.org/template/resource#2cpu-4gb-20dsk
http://schemas.openstack.org/template/resource#2cpu-4gb-50dsk
...
http://schemas.openstack.org/template/resource#8cpu-16gb-40dsk
http://schemas.openstack.org/template/resource#8cpu-8gb-10dsk
http://schemas.openstack.org/template/resource#m1-medium
```

리소스 템플릿 정보 얻기

```
# occi --endpoint https://prisma-cloud.ba.infn.it:8787/ --action describe ₩
  --resource resource_tpl#2cpu-4gb-20dsk ₩
  --auth x509 --user-cred /tmp/x509up_u500 --voms
##############################################################################################
[[ http://schemas.openstack.org/template/resource#2cpu-4gb-20dsk ]]
```

```
title:        Flavor: 2cpu-4GB-20dsk
term:         2cpu-4gb-20dsk
location:     /2cpu-4gb-20dsk/
################################################################################################
```

keypair 생성하기

```
# ssh-keygen -t rsa -b 2048 -f tmpfedcloud
```

contextualization을 위한 설정 스크립트 작성

```
# cat > tmpfedcloud.login << EOF
#cloud-config
users:
  - name: cloudadm
    sudo: ALL=(ALL) NOPASSWD:ALL
    lock-passwd: true
    ssh-import-id: cloudadm
    ssh-authorized-keys:
      - `cat tmpfedcloud.pub`
EOF

# cat tmpfedcloud.login
#cloud-config
users:
  - name: cloudadm
    sudo: ALL=(ALL) NOPASSWD:ALL
    lock-passwd: true
    ssh-import-id: cloudadm
    ssh-authorized-keys:
      - ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAv8hrTXYVKNepmwYxF8ah0miV0uB0L9lij3jM02NCrZrFUh
.....
V+BGd7n3wlQtDvCjGxyXmw== sangwan@fccont.kisti.re.kr
```

가상머신 인스턴스 생성하기

```
# occi --endpoint https://prisma-cloud.ba.infn.it:8787/ --action create --resource compute ₩
  --attribute occi.core.title="MyFirstVM" ₩
  --mixin os_tpl#72ada03a-5694-4a79-8e7e-069516a31a59 ₩
  --mixin resource_tpl#2cpu-4gb-20dsk ₩
  --context user_data="file://$PWD/tmpfedcloud.login" ₩
  --auth x509 --user-cred /tmp/x509up_u500 --voms
https://prisma-cloud.ba.infn.it:8787/compute/e302331e-665c-4ca1-8047-97ea1a9d02d2
```

인스턴스 상세 정보

```
# occi --endpoint https://prisma-cloud.ba.infn.it:8787 --action describe ₩
  --resource https://prisma-cloud.ba.infn.it:8787/compute/e302331e-665c-4ca1-8047-97ea1a9d02d2 ₩
  --voms  --auth x509 --user-cred /tmp/x509up_u500
################################################################################################
[[ http://schemas.ogf.org/occi/infrastructure#compute ]]
>> location: /compute/e302331e-665c-4ca1-8047-97ea1a9d02d2
occi.core.id = e302331e-665c-4ca1-8047-97ea1a9d02d2
occi.compute.architecture = x86
occi.compute.cores = 2
occi.compute.hostname = myfirstvm
occi.compute.memory = 4.0
```

```
occi.compute.speed = 0.0
occi.compute.state = active
org.openstack.compute.console.vnc = http://prisma-cloud.ba.infn.it:6080/vnc_auto.html?token=8141567a
-187a-4912-9767-ca46ceefe86f
org.openstack.compute.state = active

Links:
    [[ http://schemas.ogf.org/occi/infrastructure#networkinterface ]]
    >> location: /network/interface/9dd33d82-d387-4578-8ee1-b086f65ef3c4
    occi.networkinterface.gateway = 90.147.102.1
    occi.networkinterface.mac = fa:16:3e:7e:6d:29
    occi.networkinterface.interface = eth0
    occi.networkinterface.state = active
    occi.networkinterface.allocation = static
    occi.networkinterface.address = 90.147.102.243
    occi.core.source = /compute/e302331e-665c-4ca1-8047-97ea1a9d02d2
    occi.core.target = /network/admin
    occi.core.id = /network/interface/9dd33d82-d387-4578-8ee1-b086f65ef3c4

Mixins:
    [[ http://schemas.openstack.org/compute/instance#os_vms ]]
    title:
    term:         os_vms
    location:     /os_vms/

    [[ http://schemas.openstack.org/template/os#72ada03a-5694-4a79-8e7e-069516a31a59 ]]
    title:        Image: Ubuntu-14.04-amd64
    term:         72ada03a-5694-4a79-8e7e-069516a31a59
    location:     /72ada03a-5694-4a79-8e7e-069516a31a59/

Actions:
    [[ http://schemas.ogf.org/occi/infrastructure/compute/action#stop ]]
    [[ http://schemas.ogf.org/occi/infrastructure/compute/action#suspend ]]
    [[ http://schemas.ogf.org/occi/infrastructure/compute/action#restart ]]
##############################################################################################
```

SSH 로 인스턴스에 로그인하기

```
$ ssh -i tmpfedcloud  -l  cloudadm  90.147.102.243

The authenticity of host '90.147.102.243 (90.147.102.243)' can't be established.
RSA key fingerprint is 84:98:f9:18:f2:80:cc:4e:41:ee:1d:52:5b:32:a9:50.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '90.147.102.243' (RSA) to the list of known hosts.
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)
...
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
$
```

인스턴스 삭제하기

```
# occi --endpoint https://prisma-cloud.ba.infn.it:8787/ --action delete ₩
  --resource https://prisma-cloud.ba.infn.it:8787/compute/e302331e-665c-4ca1-8047-97ea1a9d0
2d2 ₩
  --auth x509 --user-cred /tmp/x509up_u500 --voms
```

3. Accounting System

EGI 자원 제공자는 어카운팅 서비스인 APEL을 설치하여 해당 사이트의 사용량 정보를 FedCloud 어카운팅 서버로 전송하여 수집될 수있도록 해야 한다. 어카운팅 서비스에 대한 자세한 내용은 타스크포트 위키[18]을 참고한다.

현재 어카운팅 정보가 수집되고 있는 사이트에 대한 정보는 [19]를 참고한다.

Sites publishing new cloud accounting records (SSM 2.0)

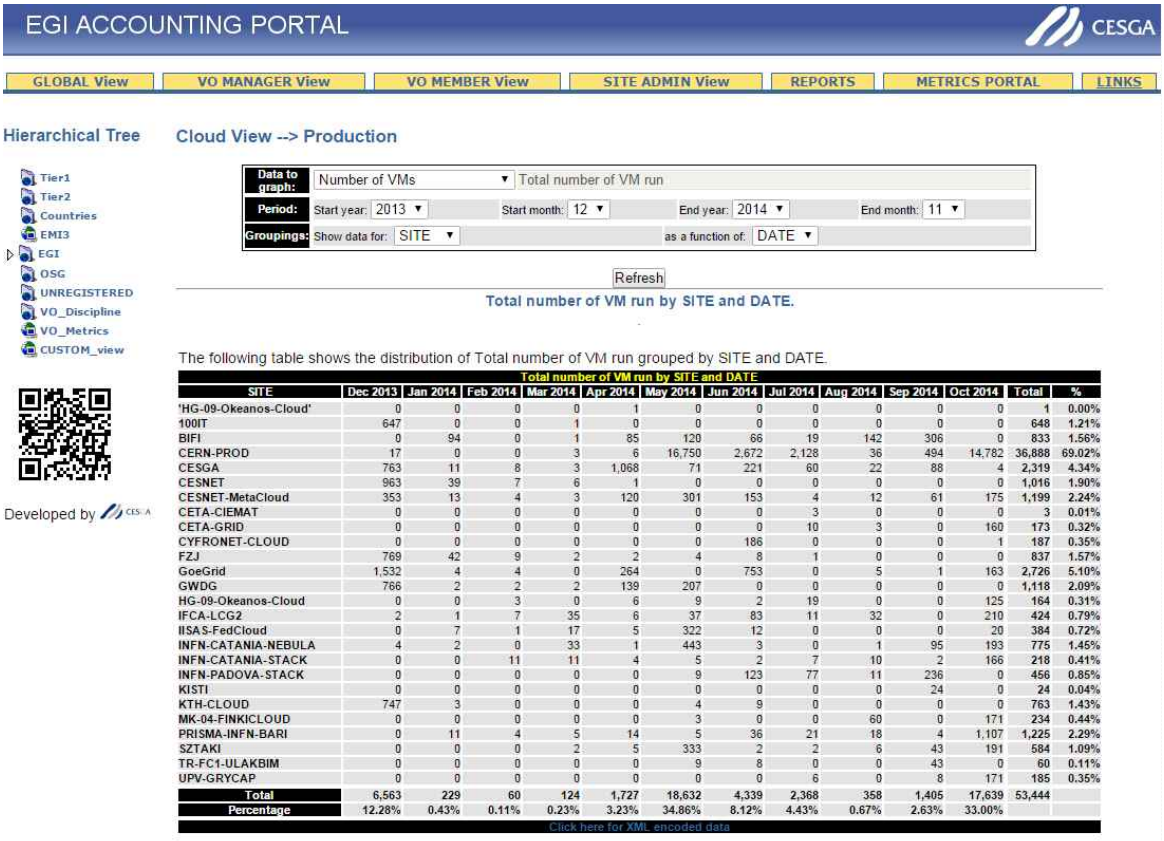
Page last updated: 2015-12-01 07:30:08.621071

Site	NumberOfMachines	CloudType	LastUpdated
100IT	17920	OpenStack	2015-12-01 00:09:28
BIFI	229906	OpenStack	2015-11-30 09:55:56
CERN-PROD	472371	OpenStack	2015-11-30 22:49:55
CESGA	44251	OpenNebula	2015-12-01 06:00:07
CESNET-MetaCloud	62352	OpenNebula	2015-12-01 06:15:17
CETA-GRID	18267	OpenStack	2015-12-01 00:00:25
CSIC-EBD-LW	3	OpenStack	2015-10-29 15:45:42
CYFRONET-CLOUD	8285	OpenStack	2015-12-01 07:00:03
FZJ	71594	OpenStack	2015-06-15 11:05:08
GoeGrid	52635	OpenNebula	2015-12-01 06:15:06
GRIF	47	StratusLab	2013-09-10 09:27:05
HG-09-Okeanos-Cloud	5412	Synnefo	2015-12-01 07:00:05
IFCA-LCG2	30166	OpenStack	2015-12-01 05:30:06
IISAS-Bratislava	46	Openstack	2013-04-09 12:23:19
IISAS-FedCloud	36643	OpenStack	2015-12-01 07:18:06
IISAS-GPUCloud	1492	OpenStack	2015-12-01 05:40:44
IN2P3-CC	11	Openstack	2013-03-19 13:47:58
IN2P3-IRES	8610	OpenStack	2015-11-30 23:30:16
INFN-CATANIA-NEBULA	16583	OpenNebula	2015-12-01 07:03:57
INFN-CATANIA-STACK	11561	OpenStack	2015-12-01 02:14:04
INFN-PADOVA-STACK	97694	OpenStack	2015-11-30 23:30:06
KISTI	43	Openstack	2015-04-29 10:00:07
KR-KISTI-CLOUD	230	OpenStack	2015-11-28 21:50:06
KTH-CLOUD	14198	OpenNebula	2015-02-03 21:59:47
MK-04-FINKICLOUD	13956	OpenNebula	2015-10-22 10:48:37
NCG-INGRID-PT	8992	OpenStack	2015-11-30 08:20:10
PRISMA-INFN-BARI	32687	Openstack	2015-11-27 17:00:07
SZTAKI	9186	OpenNebula	2015-03-02 07:30:38
TR-FC1-ULAKBIM	10090	OpenStack	2015-11-28 08:10:08
TW-EMI-PPS	213	OpenStack	2013-09-12 14:11:42
UA-BITP	600	OpenStack	2015-12-01 07:10:06
UPV-GRyCAP	8241	OpenNebula	2015-12-01 06:15:09

그림 31 EGI FedCloud 어카운팅 수집 사이트 정보

EGI Accounting Portal [20]에서는 FedCloud 연동 사이트들의 어카운팅 정보 통계를 조회할 수 있다. VM의 개수, CPU시간, 총시간, 메모리 사용량, 네트워크 사용량, 디스크 사용량을 SITE와 VO와 날짜 별로 조회할 수 있다.

18) https://wiki.egi.eu/wiki/Fedcloud-tf:WorkGroups:Scenario4
19) http://goc-accounting.grid-support.ac.uk/cloudtest/cloudsites2.html
20) http://accounting-devel.egi.eu/cloud.php



SITE	Dec 2013	Jan 2014	Feb 2014	Mar 2014	Apr 2014	May 2014	Jun 2014	Jul 2014	Aug 2014	Sep 2014	Oct 2014	Total	%
'HG-09-Okeanos-Cloud'	0	0	0	0	1	0	0	0	0	0	0	1	0.00%
100IT	647	0	0	1	0	0	0	0	0	0	0	648	1.21%
BIFI	0	94	0	1	85	120	66	19	142	306	0	833	1.56%
CERN-PROD	17	0	0	3	6	16,750	2,672	2,128	36	494	14,782	36,888	69.02%
CESGA	763	11	8	3	1,068	71	221	60	22	88	4	2,319	4.34%
CESNET	963	39	7	6	1	0	0	0	0	0	0	1,016	1.90%
CESNET-MetaCloud	353	13	4	3	120	301	153	4	12	61	175	1,199	2.24%
CETA-CIEMAT	0	0	0	0	0	0	0	3	0	0	0	3	0.01%
CETA-GRID	0	0	0	0	0	0	0	10	3	0	160	173	0.32%
CYFRONET-CLOUD	0	0	0	0	0	0	186	0	0	0	1	187	0.35%
FZJ	769	42	9	2	2	4	8	1	0	0	0	837	1.57%
GoeGrid	1,532	4	4	0	264	0	753	0	5	1	163	2,726	5.10%
GWDG	766	2	2	2	139	207	0	0	0	0	0	1,118	2.09%
HG-09-Okeanos-Cloud	0	0	3	0	6	9	2	19	0	0	125	164	0.31%
IFCA-LCG2	2	1	7	35	6	37	83	11	32	0	210	424	0.79%
IISAS-FedCloud	0	7	1	17	5	322	12	0	0	0	20	384	0.72%
INFN-CATANIA-NEBULA	4	2	0	33	1	443	3	0	1	95	193	775	1.45%
INFN-CATANIA-STACK	0	0	11	11	4	5	2	7	10	2	166	218	0.41%
INFN-PADOVA-STACK	0	0	0	0	0	9	123	77	11	236	0	456	0.85%
KISTI	0	0	0	0	0	0	0	0	0	24	0	24	0.04%
KTH-CLOUD	747	3	0	0	0	4	9	0	0	0	0	763	1.43%
MK-04-FINKICLOUD	0	0	0	0	0	3	0	0	60	0	171	234	0.44%
PRISMA-INFN-BARI	0	11	4	5	14	5	36	21	18	4	1,107	1,225	2.29%
SZTAKI	0	0	0	2	5	333	2	2	6	43	191	584	1.09%
TR-FC1-ULAKBIM	0	0	0	0	0	9	8	0	0	43	0	60	0.11%
UPV-GRYCAP	0	0	0	0	0	0	0	6	0	8	171	185	0.35%
Total	6,563	229	60	124	1,727	18,632	4,339	2,368	358	1,405	17,639	53,444	
Percentage	12.28%	0.43%	0.11%	0.23%	3.23%	34.86%	8.12%	4.43%	0.67%	2.63%	33.00%		

그림 32 EGI Accounting Portal (Cloud View)

cASO 설치

cASO는 Cloud Accounting Usage Records 정보를 오픈스택으로부터 추출해 내기 위한 스크립트이다. nova또는 ceilometer API를 이용하여 정보를 얻는다. APEL SSL에 전송하기위한 형태로 출력한다.

패키지 설치

```
# pip install caso
# pip list | grep caso
caso (0.3)
```

설정 파일 확인

```
# cp  /etc/caso/caso.conf.sample   /etc/caso/caso.conf
# cat /etc/caso/caso.conf | grep -v "^$" | grep -v "^#"
[DEFAULT]
debug=true
verbose=true
spooldir=/var/spool/caso
extractor=nova
site_name=KR-KISTI-CLOUD
tenants=fedcloud
```

```
messengers=caso.messenger.ssm.SsmMessager
[extractor]
user=accounting
password=accountingpass
endpoint=http://fccont.kisti.re.kr:35357/v2.0
insecure=true
mapping_file=/etc/keystone/voms.json
[logstash]
[ssm]
output_path=/var/spool/apel/outgoing/openstack
```

어카운팅을 위한 role을 추가한다.

```
# keystone role-create --name accounting
# keystone user-create --name accounting --pass ********
# keystone user-role-add --user accounting --role accounting --tenant fedcloud
```

어카운팅을 위해 policy 파일을 수정

```
# vi /etc/keystone/policy.json
{
    "admin_required": "role:admin or is_admin:1 or role:accounting",
    "service_role": "role:service",
...
:wq
# cp  /etc/keystone/voms.json  /etc/caso/voms.json
```

어카운팅 정보 추출하기

```
#  caso-extract
2015-06-19 15:49:09.558 53635 DEBUG keystoneclient.session [-] RESP: [200] {'date': 'Fri, 1
9 Jun 2015 06:49:09 GM 'application/json', 'content-length': '459', 'vary': 'X-Auth-Token'}
RESP BODY: {"users": [{"username": "/C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim",
"enabled": true, "name=GRID/O=KISTI/CN=80864141 Sangwan Kim", "id": "aa874aa706bd435abfdb78
2123615a1a"}, {"username": "admin", "name":  true, "email": "sangwan@kisti.re.kr", "id": "d
0edee57b85d4611822ae1c44ad5ccc1"}, {"username": "accounting", "namenabled": true, "email":
null, "id": "e1fbd5e15a8e4e6c85670359c9fe6e62"}]}
 _http_log_response /usr/lib/python2.7/site-packages/keystoneclient/session.py:182
2015-06-19 15:49:12.306 53635 INFO caso.extract.manager [-] Extracted 3 records for tenant
'admin' from 2015-06-10:00 to now

# tree -if /var/spool/apel/outgoing/openstack
/var/spool/apel/outgoing/openstack
/var/spool/apel/outgoing/openstack/5583bb5c
/var/spool/apel/outgoing/openstack/5583bb5c/5583bb684b4134
```

```
1 directory, 1 file

# cat /var/spool/apel/outgoing/openstack/5583bb5c/5583bb684b4134
APEL-cloud-message: v0.2
Status: completed
SiteName: KR-KISTI-CLOUD
MachineName: sangwan_20150618165346
ImageId: 9eabafdb-b223-4144-aff5-a1279b53e4f4
LocalUserId: aa874aa706bd435abfdb782123615a1a
FQAN: fedcloud.egi.eu
LocalGroupId: cc5dd73321a8451aa7fe04d2edfde697
GlobalUserName: /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
VMUUID: 7a2f2efe-2bfa-46ef-8301-613203669bc3
CloudType: OpenStack
%%
CpuCount: 2
Memory: 4096
..........
CloudType: OpenStack
```

어카운팅 정보를 EGI 서버로 전송하기 위해 APEL SSM을 이용한다.

EMI3 레포지터리 설치

```
# yum -y install  http://emisoft.web.cern.ch/emisoft/dist/EMI/3/sl6/x86_64/base/emi-release-3.0.0-2.el6.noarch.rpm
# yum repolist
# rpm -ql emi-release-3.0.0-2.el6.noarch
```

apel-ssm 패키지 설치

```
# yum install -y apel-ssm
# yum install  python-daemon  python-dirq  stomppy
```

rpm을 직접 설치하는 방법

```
# rpm -ivh --nodeps   http://emisoft.web.cern.ch/emisoft/dist/EMI/3/sl6/x86_64/updates/apel-ssm-2.1.5-1.el6.noarch.rpm
```

설정 파일 수정

```
# vi /etc/apel/logging.cfg
[loggers]
keys=root,SSM,Crypto,stomp.py

[handlers]
keys=fileHandler
```

```
[formatters]
keys=simpleFormatter

[logger_root]
level=DEBUG
handlers=fileHandler

# Pick up the logging from the stomppy package
[logger_stomp.py]
level=INFO
handlers=fileHandler
qualname=stomp.py
propagate=0

# We use the SSM key so we get the fileHandler
[logger_SSM]
level=INFO
handlers=fileHandler
qualname=SSM
propagate=0

[logger_Crypto]
level=DEBUG
handlers=fileHandler
qualname=Crypto
propagate=0

[handler_fileHandler]
class=FileHandler
level=DEBUG
formatter=simpleFormatter
args=('/var/log/apel/ssm.log', 'a')

[formatter_simpleFormatter]
format=%(asctime)s - %(name)s - %(levelname)s - %(message)s
datefmt=
```

```
# vi /etc/apel/sender.cfg
[broker]
host: mq.cro-ngi.hr
port: 6163
```

```
use_ssl: true

[certificates]
certificate: /etc/grid-security/hostcert.pem
key: /etc/grid-security/hostkey.pem
capath: /etc/grid-security/certificates

[messaging]
destination: /queue/global.accounting.test.cloud.central
path: /var/spool/apel/outgoing/openstack

[logging]
logfile: /var/log/apel/ssmsend.log
level: INFO
console: true
```

브로커를 명시하는 방법은 2가지가 있는데 BDII에 쿼리를 하여 사용가능한 브로커를 찾는 방법과 특정 브로커의 호스트와 포트를 명시하는 방법이다. 이 포트는 stomp 프로토콜을 위해 사용된다.

실행 방법은 다음과 같다.

```
# /usr/bin/ssmsend
2015-06-19 17:43:10,154 - ssmsend - INFO - ========================================
2015-06-19 17:43:10,155 - ssmsend - INFO - Starting sending SSM version 2.1.5.
2015-06-19 17:43:10,155 - ssmsend - INFO - No server certificate supplied.  Will not encryp
t messages.
2015-06-19 17:43:10,507 - stomp.py - INFO - Established connection to host mq.cro-ngi.hr, p
ort 6163
2015-06-19 17:43:10,834 - ssm.ssm2 - INFO - Connected.
2015-06-19 17:43:10,834 - ssm.ssm2 - INFO - Will send messages to: /queue/global.accountin
g.test.cloud.central
2015-06-19 17:43:10,835 - ssm.ssm2 - INFO - Found 1 messages.
2015-06-19 17:43:10,835 - ssm.ssm2 - INFO - Sending message: 5583d5d8/5583d5e53d1521
2015-06-19 17:43:10,849 - ssm.ssm2 - INFO - Waiting for broker to accept message.
2015-06-19 17:43:11,178 - ssm.ssm2 - INFO - Broker received message: 5583d5d8/5583d5e53d152
1
2015-06-19 17:43:11,250 - ssm.ssm2 - INFO - Tidying message directory.
2015-06-19 17:43:11,251 - ssmsend - INFO - SSM run has finished.
2015-06-19 17:43:11,252 - ssm.ssm2 - INFO - SSM connection ended.
2015-06-19 17:43:11,252 - ssmsend - INFO - SSM has shut down.
2015-06-19 17:43:11,252 - ssmsend - INFO - ========================================
```

위 명령을 주기적으로 실행한다. 크론 작업으로 추가하기 (매6시간마다)

```
# vi /usr/bin/caso-run.sh
#!/bin/sh
/usr/bin/caso-extract
/usr/bin/ssmsend

# chmod 755 /usr/bin/caso-run.sh
# crontab -e
50 */6 * * * /usr/bin/caso-run.sh
```

SSM 2.0 에서 수집되는 어카운팅 정보 형식은 다음과 같다.

표 15 SSM 2.0 Accounting Information

Key	Value	Description	Mandatory
VMUUID	string	Virtual Machine's Universally Unique IDentifier	Yes
SiteName	string	Sitename, e.g. GOCDB Sitename	Yes
MachineName	string	VM Id	
LocalUserId	string	Local username	
LocalGroupId	string	Local groupname	
GlobalUserName	string	User's X509 DN	
FQAN	string	User's VOMS attributes	
Status	string	Completion status - started, completed, suspended	
StartTime	int	Must be set if Status = Started (epoch time)	
EndTime	int	Must be set if Status = completed (epoch time)	
SuspendDuration	int	Set when Status = suspended (seconds)	
WallDuration	int	Wallclock - actual time used (seconds)	
CpuDuration	int	CPU time consumed (seconds)	
CpuCount	int	Number of CPUs allocated	
NetworkType	string	Description	
NetworkInbound	int	GB received	
NetworkOutbound	int	GB sent	
Memory	int	Memory allocated to the VM (MB)	
Disk	int	Disk allocated to the VM (GB)	
StorageRecordId	string	Link to associated storage record	
ImageId	string	Image ID	
CloudType	string	e.g. OpenNebula, Openstack	

4. Information System

BDII와 연동 작업은 Openstack 과 OpenNebula 가 모두 동일하다. 자세한 방법은 위키[21]을 참고한다. cloud-info-provider 패키지는 EGI's AppDB[22]에서 찾을 수 있다. 본 절에서는 RHEL/CentOS/ScientificLinux 를 기준으로 하였다.

BDII 설정

repository 설치

```
# wget  http://repository.egi.eu/community/software/cloud.info.provider/0.x/releases/repofiles/sl-6-x86_64.repo  -O /etc/yum.repos.d/cloud-info-provider.repo
```

패키지 설치

```
# yum install cloud-info-provider-service
또는
# rpm -ivh --nodeps ₩
   http://repository.egi.eu/community/software/cloud.info.provider/0.x/releases/sl/6/x86_64/RPMS/cloud-info-provider-service-0.5-1.el6.noarch.rpm
```

설치된 파이썬 버전 문제로 라이브러리를 수동으로 옮겨 줌

```
# mv /usr/lib/python2.6/site-packages/cloud_bdii-0.5.dev146.gb0877a4-py2.6.egg-info ₩
   /usr/lib/python2.7/site-packages/
# mv /usr/lib/python2.6/site-packages/cloud_bdii    /usr/lib/python2.7/site-packages/
```

/etc/cloud-info-provider 에 있는 설정파일들을 참고하여 작성한다. YAML 파일을 작성

```
# cp  /etc/cloud-info-provider/sample.openstack.yaml  /etc/cloud-info-provider/bdii.yaml
# vi /etc/cloud-info-provider/bdii.yaml
site:
    # Your site name, as in GODCB (if omitted or set to None, this value is
    # retreived from /etc/glite-info-static/site/site.cfg )
    name: KR-KISTI-CLOUD

    # Site url
    url: http://www.kisti.re.kr
    # Production level
    production_level: production
    # Two digit country code
    country: KR
    # Site Longitude
    longitude: 127.35
    # Site Latitude
    latitude: 36.36
    # Your affiliated NGI
    ngi: AsiaPacific
    # Contact email
    general_contact: fedcloud-list@kisti.re.kr
```

21) https://wiki.egi.eu/wiki/Fedclouds_BDII_instructions

22) https://appdb.egi.eu/store/software/cloud.info.provider

```
    # User support email
    user_support_contact: fedcloud-list@kisti.re.kr
    # Sysadmin contact email
    sysadmin_contact: fedcloud-list@kisti.re.kr
    # Security contacts email email
    security_contact: fedcloud-list@kisti.re.krorg
    # User support email
    bdii_host: fccont.kisti.re.kr
    # User support email
    bdii_port: 2170

compute:
    # Total number of cores available
    total_cores: 64
    # Total RAM available (GB)
    total_ram: 512
    # Hypervisor name
    hypervisor: qemu-kvm
    # Hypervisor version
    hypervisor_version: 0.12.1.2
    # OpenStack version
    middleware_version: havana
    # Service Production level (testing, candidate, production...)
    service_production_level: candidate
    # Service capabilities
    capabilities:
        - cloud.managementSystem
        - cloud.vm.uploadImage
............
```

provider 설정을 위해 python-novaclient를 이용하게 되는데, --os-username, --os-password, --auth-tenant-name, --os-auth-url, --os-cacert 와 같은 옵션이 이용된다.

keystone 에 OCCI 타입의 서비스가 등록되어 이어야 한다.

```
# keystone service-list
+----------------------------------+------------+----------+-----------------------------+
|                id                |    name    |   type   |         description         |
+----------------------------------+------------+----------+-----------------------------+
| 2c5c89375bf748afbef3dea720e0208e | ceilometer | metering | Ceilometer Telemetry Service |
| cab0766ae08745e5b553eb215e9ca474 |   cinder   |  volume  |    Cinder Volume Service     |
| 6be960dc450c42769cbef82eba216b92 |  cinderv2  | volumev2 |   Cinder Volume Service V2   |
| 0f05e889c916453daaeb4e5a677e59cb |   glance   |  image   |     Glance Image Service     |
| 17f443ad21c5401f84ac4231bd8fb55c |  keystone  | identity |       Identity Service       |
| 2b0937c081cb484294884898f8ae5d42 |    nova    | compute  |     Nova Compute service     |
| 8418e8373eec45a38832a0b4b9ae8d73 |    nova    |   occi   |      Nova OCCI Service       |
+----------------------------------+------------+----------+-----------------------------+
```

provider 설정

/var/lib/bdii/gip/provider/cloud-info-provider 파일을 생성한다.

```
# vi /var/lib/bdii/gip/provider/cloud-info-provider
#!/bin/sh


cloud-info-provider-service --yaml /etc/cloud-info-provider/bdii.yaml ₩
  --middleware openstack ₩
```

```
  --os-username admin --os-password adminpass ₩
  --os-tenant-name admin --os-auth-url http://controller:35357/v2.0 ₩
  --full-bdii-ldif
```

실행 테스트

```
# chmod +x /var/lib/bdii/gip/provider/cloud-info-provider
# /var/lib/bdii/gip/provider/cloud-info-provider

INFO:urllib3.connectionpool:Starting new HTTP connection (1): fccont.kisti.re.kr
DEBUG:urllib3.connectionpool:"POST /v2.0/tokens HTTP/1.1" 200 6267
dn: o=glue
objectClass: organization
o: glue

dn: GLUE2GroupID=cloud,o=glue
objectClass: GLUE2Group
GLUE2GroupID: cloud

INFO:urllib3.connectionpool:Starting new HTTP connection (1): fccont.kisti.re.kr
DEBUG:urllib3.connectionpool:"POST /v2.0/tokens HTTP/1.1" 200 6267
INFO:urllib3.connectionpool:Starting new HTTP connection (1): fccont.kisti.re.kr
DEBUG:urllib3.connectionpool:"GET /v2/46196f54ae8d4edbac6bb4a7d7871849/flavors/detail HTTP/1.1" 200
2149
DEBUG:urllib3.connectionpool:"GET /v2/46196f54ae8d4edbac6bb4a7d7871849/images/detail HTTP/1.1" 200 2
200
dn: GLUE2ServiceID=http://fccont.kisti.re.kr:35357/v2.0_cloud.compute,GLUE2GroupID=cloud,o=glue
objectClass: GLUE2Entity
objectClass: GLUE2Service
objectClass: GLUE2ComputingService
GLUE2ServiceAdminDomainForeignKey: KR-KISTI-CLOUD
GLUE2ServiceID: http://fccont.kisti.re.kr:35357/v2.0_cloud.compute
GLUE2ServiceQualityLevel: candidate
GLUE2ServiceType: IaaS
GLUE2ServiceCapability: ['cloud.managementSystem', 'cloud.vm.uploadImage']
.................

dn: GLUE2StorageServiceCapacityID=fccont.kisti.re.kr_cloud.storage_capacity,GLUE2ServiceID=fccont.ki
sti.re.kr_cloud.storage,GLUE2GroupID=cloud,o=glue
objectClass: GLUE2Entity
objectClass: GLUE2StorageServiceCapacity
GLUE2StorageServiceCapacityID: fccont.kisti.re.kr_cloud.storage_capacity
GLUE2StorageServiceCapacityType: online
GLUE2StorageServiceCapacityStorageServiceForeignKey: fccont.kisti.re.kr_cloud.storage
GLUE2StorageServiceCapacityTotalSize: 0
```

위와 같이 LDIF 형식으로 나오면 성공이다. 이제 bdii 서비스를 시작한다.

```
# systemctl  start bdii
```

LDAP 검색으로 확인

```
# ldapsearch -x -h localhost -p 2170 -b o=glue
# extended LDIF
#
# LDAPv3
# base <o=glue> with scope subtree
```

```
# filter: (objectclass=*)
# requesting: ALL
#

# glue
dn: o=glue
objectClass: organization
o: glue

# grid, glue
dn: GLUE2GroupID=grid,o=glue
objectClass: GLUE2Group
GLUE2GroupID: grid

# cloud, glue
dn: GLUE2GroupID=cloud,o=glue
objectClass: GLUE2Group
GLUE2GroupID: cloud

# resource, glue
dn: GLUE2GroupID=resource,o=glue
objectClass: GLUE2Group
GLUE2GroupID: resource

# KR-KISTI-CLOUD, glue
dn: GLUE2DomainID=KR-KISTI-CLOUD,o=glue
objectClass: GLUE2Domain
objectClass: GLUE2AdminDomain
GLUE2DomainDescription: Korea Institute of Science and Technology Information,
  Republic of Korea
GLUE2DomainWWW: http://www.kisti.re.kr
GLUE2EntityOtherInfo: GRID=KOREA
GLUE2DomainID: KR-KISTI-CLOUD

# fccont.kisti.re.kr_cloud.storage, cloud, glue
dn: GLUE2ServiceID=fccont.kisti.re.kr_cloud.storage,GLUE2GroupID=cloud,o=glue
GLUE2ServiceAdminDomainForeignKey: KR-KISTI-CLOUD
objectClass: GLUE2Entity
objectClass: GLUE2Service
objectClass: GLUE2StorageService
GLUE2ServiceQualityLevel: None
GLUE2ServiceCapability: ['cloud.data.upload']
GLUE2ServiceType: STaaS
GLUE2ServiceID: fccont.kisti.re.kr_cloud.storage

# location.KR-KISTI-CLOUD, KR-KISTI-CLOUD, glue
dn: GLUE2LocationID=location.KR-KISTI-CLOUD,GLUE2DomainID=KR-KISTI-CLOUD,o=glu
 e
objectClass: GLUE2Location
GLUE2LocationLongitude: 127.359308
GLUE2LocationCountry: South Korea
GLUE2LocationDomainForeignKey: KR-KISTI-CLOUD
GLUE2LocationID: location.KR-KISTI-CLOUD
GLUE2LocationLatitude: 36.365830
.............
# numResponses: 28
# numEntries: 27
```

이후에 resource BDII 를 site-BDII 에 등록해야 한다. Site-BDII 설정에 관해서는 위키문서[23]를 참

고한다.

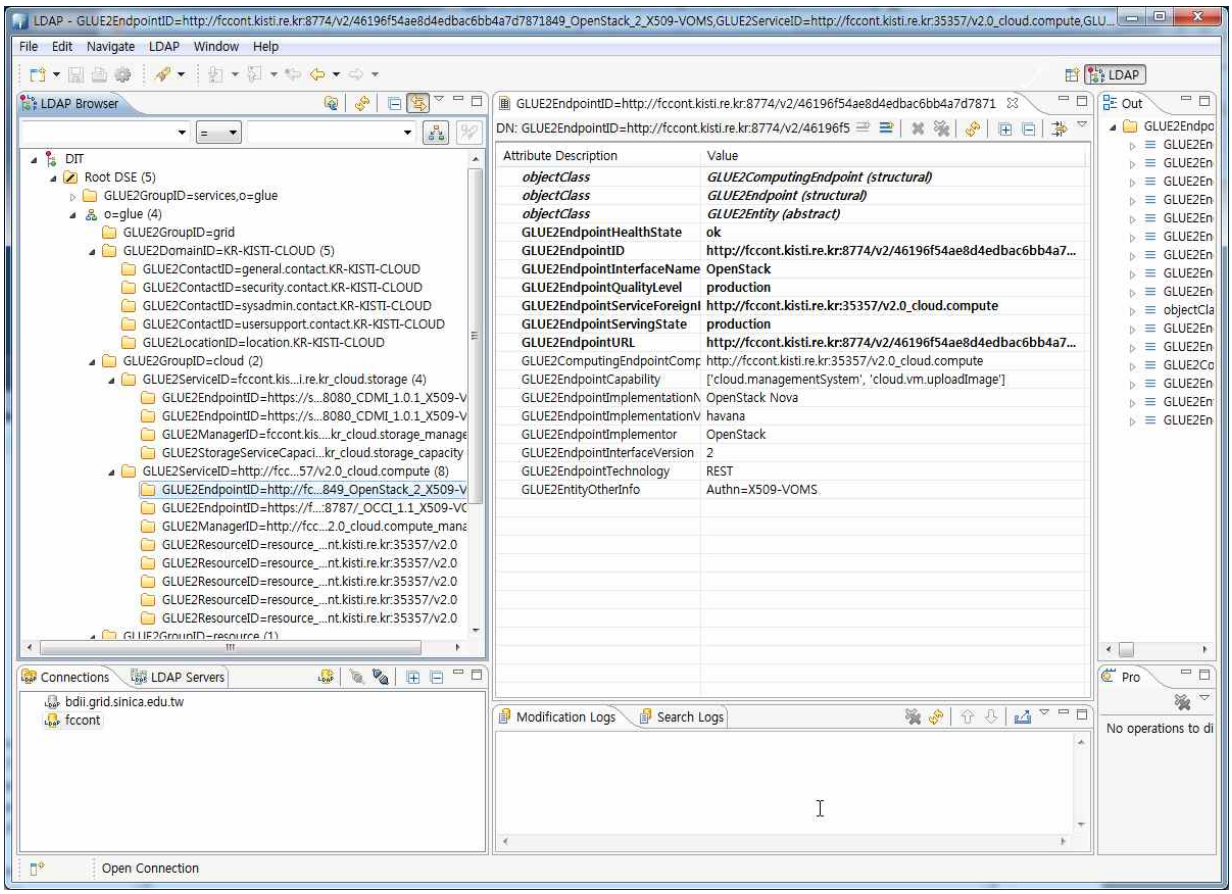


그림 33 Apache Directory Studio를 이용한 LDAP 브라우징

(2) 모니터링을 위해 GOCDB에 등록하기

클라우드 자원에 대한 모니터링은 availability 와 reliability를 말한다. 가상의 사용자가 적어도 한 개의 미리 정의된 VM을 일정 시간 동안 작동시켜 봄으로써 availability 와 reliability 를 테스트 한다. EGI FedCloud의 모니터링 시스템에서는 NAGIOS 를 바탕으로 이루어진다.

클라우드 사이를 GOCDB [1]에 추가 하여 모니터링 서비스가 찾을 수 있도록 해야 한다.

다음과 같은 서비스 타입이 GOCDB에 등록되어야 한다.

- eu.egi.cloud.accounting (required)
- eu.egi.cloud.information.bdii (required) --> 필요없게 됨
- eu.egi.cloud.storage-management.cdmi
- eu.egi.cloud.vm-management.occi (required)
- eu.egi.cloud.vm-metadata.marketplace

23) https://wiki.egi.eu/wiki/MAN01_How_to_publish_Site_Information

먼저는 endpoint 가 속하게될 SITE를 결정하는 것이다. 여기에는 2가지 방법이 있다.

a. 기존의 EGI 사이트에 등록하는 방법

이 경우 해당 사이트의 상태는 "Certified" 이다.

b. 새로운 사이트에 추가하는 방법

새로운 사이트는 다음과 같은 설정이 필요하다.

Infrastructure: 'Production' / Certification Status: 'Candidate'

두 경우 모두 서비스 endpoints 는 다음 플래그를 설정해야 한다.

Production: 'N' / Beta: 'N' / Monitored: 'Y'

다음의 서비스 타입에 대해서는 특별한 규칙이 적용된다.

- eu.egi.cloud.storage-management.cdmi: URL 에 다음 정보가 들어가야 한다.

http[s]://hostname:port

- eu.egi.cloud.vm-management.occi: URL 에 다음 정보가 들어가야 한다.

https://hostname:port/?image=<image_name>&resource=<resource_name>

<image_name>, <resource_name>에는 공백문자가 들어갈 수 없으면 이것은 각각 os_tpl 과 resource_tpl 에 해당된다.

GOCDB 입력 방법은 [2]를 참고한다.

[1] http://goc.egi.eu/

[2] https://wiki.egi.eu/wiki/GOCDB/Input_System_User_Documentation

Service Availability Monitoring (SAM) [3] 은 인프라안에서 resource들을 모니터링 하기 위해 사용되는 도구이다. grid/cloud 사이트의 availability와 reliability를 모니터링한다.

SAM instance 는 [4]에 설치되어 있다.

SAM에서 테스트하는 항목은 [5]에 기술되어 있다.

[3] https://wiki.egi.eu/wiki/SAM

[4] https://cloudmon.egi.eu/nagios

[5] https://wiki.egi.eu/wiki/Cloud_SAM_tests

표 16 SAM Tests

Nagios 테스트	주기	설명
eu.egi.cloud.APEL-Pub	12 hours	APEL publishing 모니터링 체크
eu.egi.cloud.AppDB-Update	15 minutes	image list 로 부터 hv:imagelist.dc:date:created 정보를 조사 error(>12 hours), warning (6~12 hours), ok(<6 hours)
eu.egi.cloud.OCCI-VM	1 h	사용자가 OCCI interface 로 VM을 생성할 수 있는지 체크함

eu.egi.cloud.Perun-Check	5 minutes	perun 에 접속하여 상태를 모니터링함
org.nagios.Broker-TCP	15 min	broker 포트를 채크함
org.nagios.CDMI-TCP	15 min	CDMI 포트를 채크함
org.nagios.CloudBDII-Check	15 min	BDII가 실행중인지 체크함 (using -b o=glue and LDAP version 3).
org.nagios.OCCI-TCP	15 min	OCCI 포트를 채크함

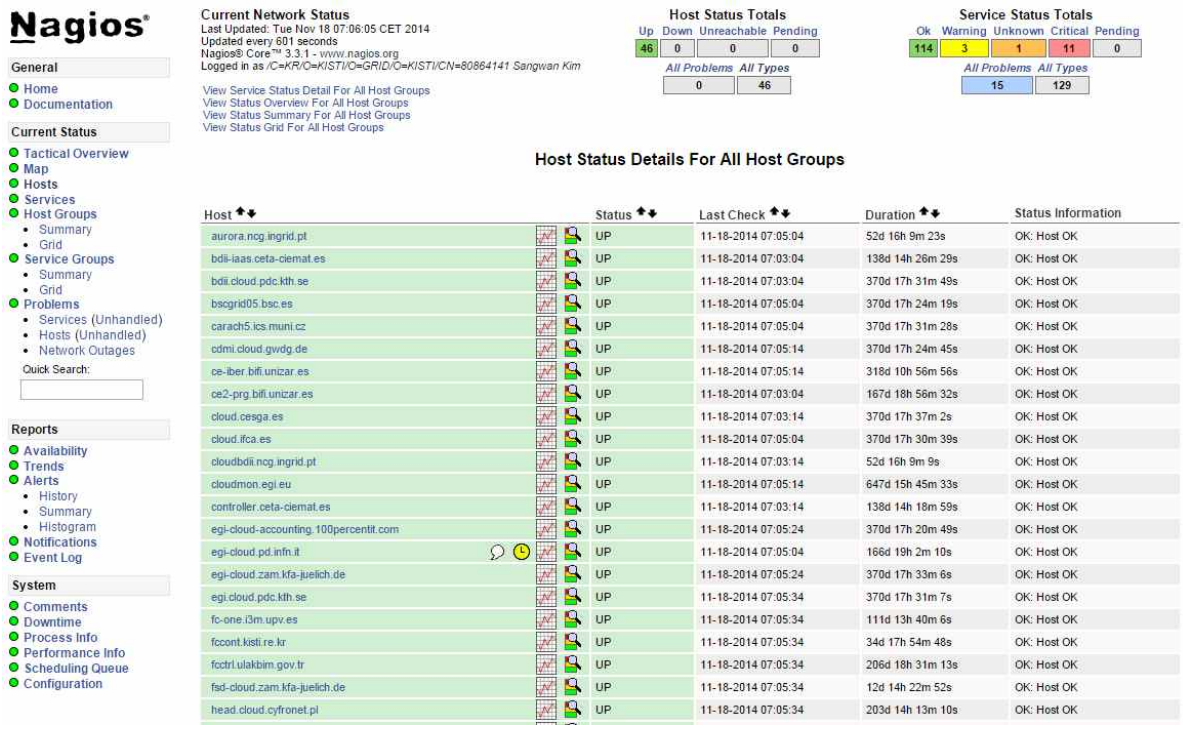


그림 34 FedCloud Nagios SAM 모니터링

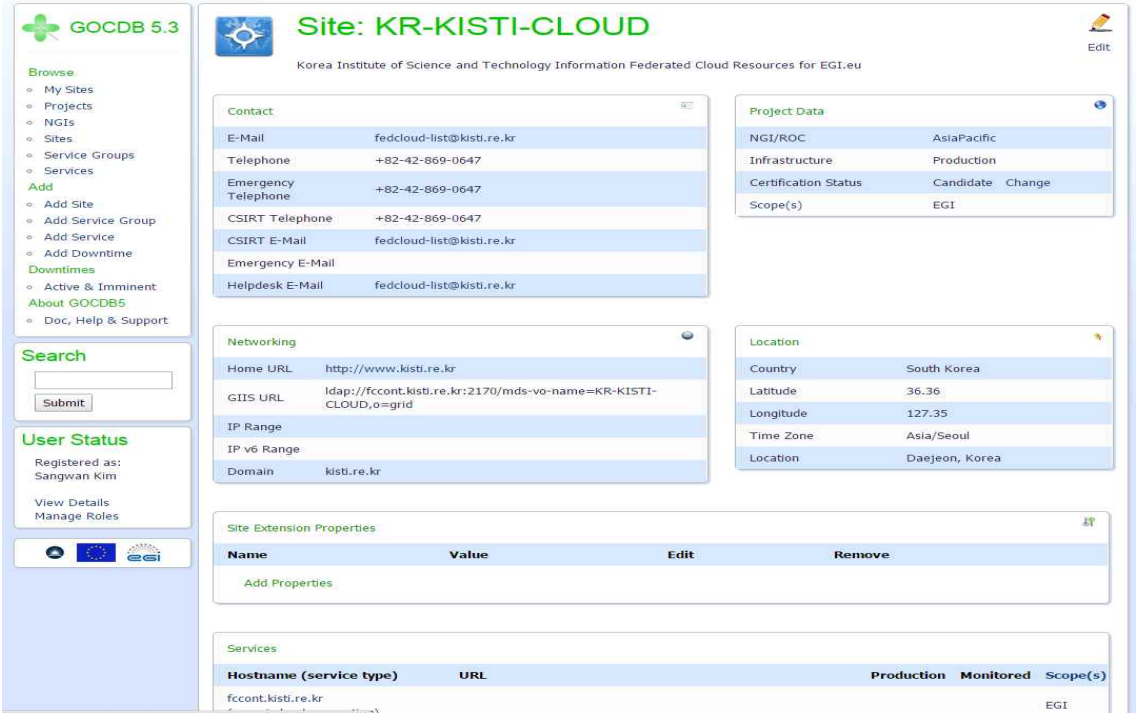


그림 35 GOC DB Site Infomation : KR-KISTI-CLOUD

제4장 Federated Cloud 활용

Federated Cloud 를 활용하기 위해서는 user support 페이지 [1]를 참고한다. 사용자는 다음과 같은 절차를 따라야 한다.

1) 그리드 인증서 발급
2) fedcloud.egi.eu VO(virtual organization) 가입
3) AppDB Cloud Marketplace [2]에서 이미지를 선택하여 활용
4) 커맨드 라인 도구나 high-level brokering tool을 이용하여 FedCloud를 활용할 수 있다.

[1] https://wiki.egi.eu/wiki/Federated_Cloud_user_support

[2] https://appdb.egi.eu/browse/cloud

제1절 VO 회원가입

FedCloud의 VO명은 fedcloud.egi.eu으로 가입을 위해서는 [3]을 방문하면 된다. 처음 접속시 그리드 인증서를 선택해야 하며, 선택한 그리드 인증서 사용자로 VO를 가입하게 된다. 그리드 인증서는 사용자의 브라우저 내에 설치되어 있어야 한다.

[3] https://perun.metacentrum.cz/perun-registrar-cert/?vo=fedcloud.egi.eu

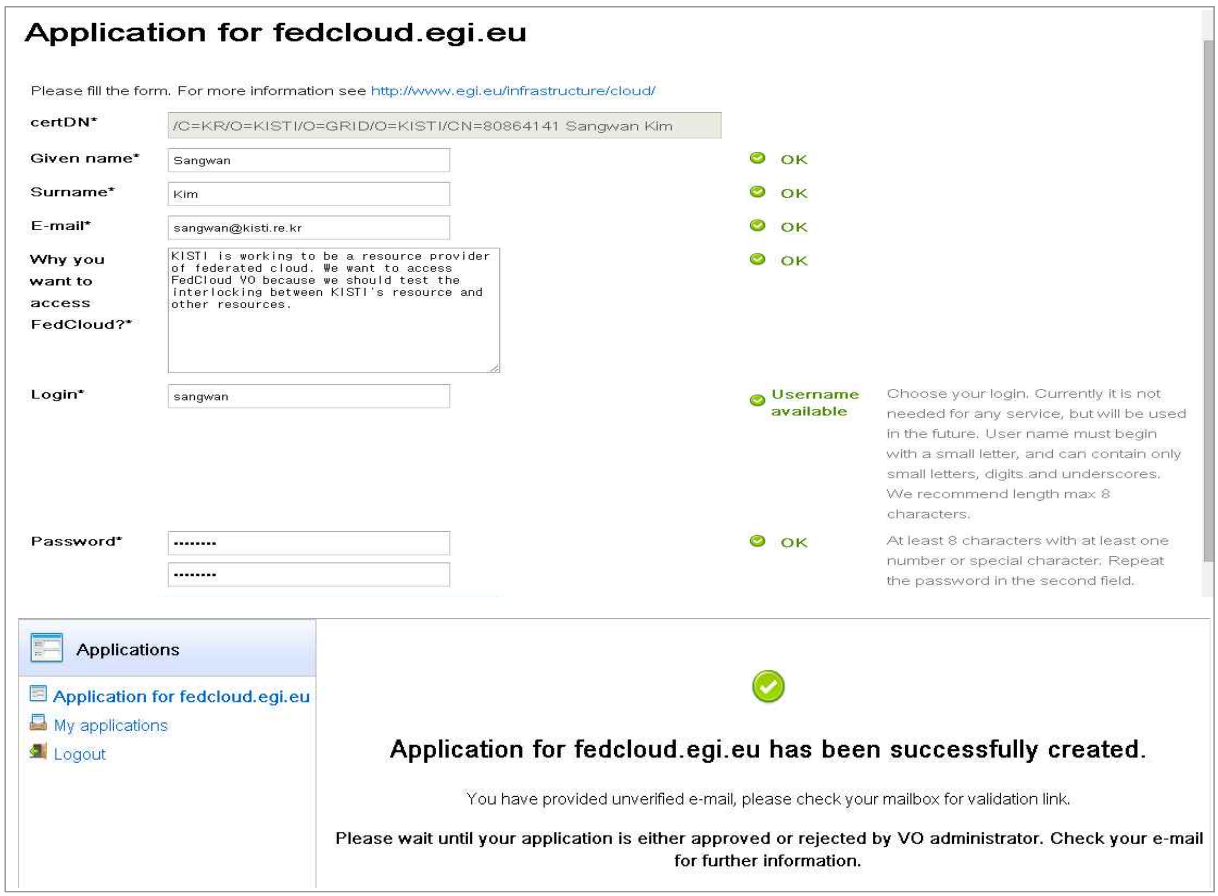


그림 36 FedCloud VO 가입화면

제2절 커맨드라인 환경 설정

FedCloud를 활용하기 위해서는 현재 커맨드라인 환경에서 활용이 가능하다. Ruby 언어로 작성된 OCCI rOCCI [1] 를 설치하면 된다. [2] 참고

[1] https://github.com/gwdg/rOCCI-cli

[2] https://wiki.egi.eu/wiki/Fedcloud-tf:CLI_Environment

설치를 위해 64비트 리눅스 시스템이 필요하다.요

설치가능 OS: RedHat 6.x (x86_64) / Scientific Linux 6.x (x86_64) / CentOS 6.x (x86_64) / Mac OS X > 10.6 / Debian >= 6 (amd64) / Ubuntu 12.04 (amd64)

다음과 같은 순서를 따른다.

1. 그리드 사용자 인증서 발급
2. 인증기관 인증서 설정
3. FedCoud VO 가입
4. VO 클라이언트 도구 서치
5. VO membership 설정
6. 프록시 인증서 생성
7. rOCCI 클라이언트 설치

1) 그리드 사용자 인증서 발급

그리드 인증기관 (Grid Certificate Authority)를 통하여 사용자의 인증서를 발급 받는다. 발급 절차는 인증기관마다 약간씩 다르므로 해당 발급 기관의 발급 절차를 따라야 한다. 인증기관으로 부터 발급받은 사용자 인증서와 개인키를 가지고 클라이언트의 적절한 위치에 복사해준다.

홈디렉터리의 .globus 폴더아래 usercert.pem과 userkey.pem에 인증서와 개인키를 복사한다.

2) 인증기관 인증서 설정

신뢰할 수 있는 인증기관의 인증서를 시스템에 설치해 주어야한다. EGI 인프라에서 운영되는 서비스들은 다양한 인증기관(CA)로 부터 받은 인증서를 사용하고 있기 때문에 EGI 인프라에서 다른 기관의 자원을 활용하기 위해서는 상대방 인증기관의 인증서도 설치해 주어야 한다. EU Grid PMA initiative [3] 에서 관리되고 있는 CA들의 인증서를 설치해 준다.

[3] http://www.eugridpma.org/

EGI trust anchors (인증기관 인증서) 설치 방법

```
# cd   /etc/yum.repos.d
# wget http://repository.egi.eu/sw/production/cas/1/current/repo-files/EGI-trustanchors.repo
# yum install -y   ca-policy-egi-core
```

3) FedCoud VO 가입

EGI 인프라내에서 특정 VO(Virtual Organization)에 가입함으로써 특정 VO내의 자원을 활용할 수 있다. 사용자가 어느 VO에 속해 있는지에 대한 정보를 담고 있으며, 서비스에 접근할때 자격을 증명하기 위해 프록시 인증서(proxy certificate)를 사용한다. 프록시 인증서는 사용자 인증서로 서명하여 만들 수 있으며, 적절한 VO 속성 정보를 담고 있다. FedCloud VO에 가입 방법은 앞 절에서 설명하였으므로 생략한다.

4) VO 클라이언트 도구 설치

프록시 인증서를 만들기 위한 VOMS client tools를 설치한다.

```
# yum install voms-clients3
```

5) VO membership service 설정

VOMS client tool이 프록시 인증서를 만들어내기 위해서는 약간의 설정 정보가 필요하다. 가입한 VO의 membership service(VOMS)에 대한 정보를 설정해 주어야 한다.

VO정보는 다음 경로에 저장된다.

- /etc/grid-security/vomsdir 디렉터리
- /etc/vomses 파일

VOMS 서비스 정보 설정

```
# mkdir -p /etc/grid-security/vomsdir/fedcloud.egi.eu
# cd /etc/grid-security/vomsdir/fedcloud.egi.eu
# cat >voms1.egee.cesnet.cz.lsc <<EOF
/DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=voms1.egee.cesnet.cz
/C=NL/O=TERENA/CN=TERENA eScience SSL CA
EOF
# cat >voms2.grid.cesnet.cz <<EOF
/DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=voms2.grid.cesnet.cz
/C=NL/O=TERENA/CN=TERENA eScience SSL CA
EOF
```

VOMS 서비스 주소 설정 (voms1.egee.cesnet.cz:15002)

```
# cd /etc
# cat >>vomses <<EOF
"fedcloud.egi.eu" "voms1.egee.cesnet.cz" "15002" "/DC=org/DC=terena/DC=tcs/OU=Domain Control Validat
ed/CN=voms1.egee.cesnet.cz" "fedcloud.egi.eu" "24"
"fedcloud.egi.eu" "voms2.grid.cesnet.cz" "15002" "/DC=org/DC=terena/DC=tcs/OU=Domain Control Validat
ed/CN=voms2.grid.cesnet.cz" "fedcloud.egi.eu" "24"
EOF
```

6) 프록시 인증서 생성

프록시 인증서를 생성하기 위해 voms-proxy-init 명령을 이용한다.

```
$ voms-proxy-init -voms fedcloud.egi.eu --rfc -dont-verify-ac
Enter GRID pass phrase:
Your identity: /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
Creating temporary proxy ..................................... Done
Contacting  voms1.egee.cesnet.cz:15002 [/DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=voms
```

```
1.egee.cesnet.cz] "fedcloud.egi.eu" Done
Creating proxy ..................................... Done
Your proxy is valid until Thu Nov 27 01:20:37 2014
```

생성된 프록시 인증서의 정보를 확인하는 방법

```
$ voms-proxy-info -all
subject   : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim/CN=1288283198
issuer    : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
identity  : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
type      : RFC compliant proxy
strength  : 1024 bits
path      : /tmp/x509up_u500
timeleft  : 11:59:17
key usage : Digital Signature, Key Encipherment, Data Encipherment
=== VO fedcloud.egi.eu extension information ===
VO        : fedcloud.egi.eu
subject   : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
issuer    : /DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=voms1.egee.cesnet.cz
attribute : /fedcloud.egi.eu/Role=NULL/Capability=NULL
timeleft  : 11:59:18
uri       : voms1.egee.cesnet.cz:15002
```

7) rOCCI 클라이언트 설치

```
# wget -O /etc/yum.repos.d/rocci-cli.repo http://repository.egi.eu/community/software/rocci.cli/4.3.x/releases/repofiles/sl-6-x86_64.repo
# yum install -y occi-cli
# ln -s /opt/occi-cli/bin/occi /usr/bin/occi
```

이제 설치된 OCCI 클라이언트를 이용하여 FedCloud 자원에 접근할 수 있다.