ISBN ????????????????

EGI Federated Cloud 구축 기술 보고서

HTCaaS Team @ KISTI

2014. 11. 30.

한국과학기술정보연구원

저자소개

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

곽재혁 (Jae-Hyuck Kwak)
Korea Institute of Science and Technology Information
Supercomputing Center
Senior Researcher
Email : jhkwak@kisti.re.kr

허태상 (Taesang Huh)
Korea Institute of Science and Technology Information
Supercomputing Center
Senior Researcher
Email : tshuh@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

제1장 머리말 ---- **1**

제2장 EGI Federated Cloud 개요 ---- **2**

제1절 개요 ---- 2

제2절 자원의 규모 ---- 2

제3절 워킹 그룹 ---- 5

제5절 Federated Cloud 기술 ---- 9

제6절 사용자 커뮤니티 ---- 12

제3장 Federated Cloud 사이트 구축 ---- **13**

제1절 설치환경 ---- 13

제2절 OpenStack ---- 14

제3절 VOMS ---- 39

제4절 OCCI ---- 48

제5절 Accounting System ---- 56

제6절 Information System ---- 63

제4장 Federated Cloud 활용 ---- **70**

제1절 VO 회원 가입 ---- 70

제2절 커맨드라인 환경 ---- 71

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 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~)를 추진하고 있다.

EGI Federated Cloud는 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 12개국가, 19개 기관이 참여하고 있으며, KISTI도 비유럽권으로서 최초로 FedCloud 참여를 위해 노력하고 있다.

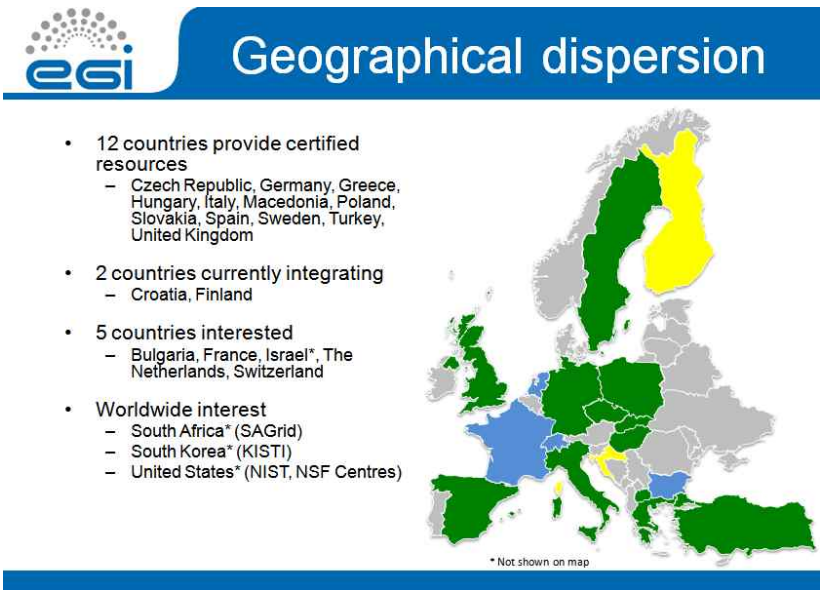


그림 1 EGI Federated Cloud 참여 국가와 기관

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.

EGI Federated Cloud에 대한 3페이지 정도의 position paper 는 EGI Cloud position paper 2014 (Interoperability is the key to freedom in the Cloud) [2] 에서 읽어 볼수 있다.

[1] http://www.egi.eu/services/catalogue/ (EGI service catalogue)

[2] https://documents.egi.eu/public/ShowDocument?docid=2120

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.

2014년 11월 현재 EGI Federated Cloud의 자원 현황은 다음 표와 같다. (최신 현황은 [1]을 참고)

[1] https://wiki.egi.eu/wiki/Fedcloud-tf:ResourceProviders

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		2014-02-14
BSC	ES	BSC-Cloud	Emotive				-
Cyfronet (NGI PL)	PL	CYFRONET-CLOUD	OpenStack	200	400	20	2014-05-08
FCTSG	ES	CESGA	OpenNebula	296	592		2014-03-06
CESNET	CZ	CESNET-MetaCloud	OpenNebula	240	960	44	2014-02-24
FZ Jülich	DE	FZJ	OpenStack	144	294	33	(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	96	288	16	2014-04-03
INFN-Catania	IT	INFN-CATANIA-NEBULA	OpenNebula	120		6	certified
	IT	INFN-CATANIA-STACK	OpenStack				certified
KTH	SE	KTH-CLOUD	OpenNebula	192	384	6	2014-02-27
INFN-Bari	IT	PRISMA-INFN-BARI	OpenStack	300	600	50	certified
INFN-Padova	IT	INFN-PADOVA-STACK	OpenStack/ Havana	48	96	5	2014-06-03
MTA SZTAKI	HU	SZTAKI	OpenNebula	128	128	6	certified
TUBITAK ULAKBIM	TR	TR-FC1-ULAKBIM	OpenStack	224	672	10	certified
UKIM	MK	MK-04-FINKICLOUD	OpenNebula	216	432	17	certified
CETA-CIEMAT	ES	CETA-GRID	OpenStack Havana	112	224	2	
	CZ DE ES GR HU IT MK PL SE SK TR UK 12개국			5,656	14,362	271.4	

표 4 EGI Federated Cloud Resource Provider (Integrating RPs)

Affiliation	CC	Main Contact	Deputies	CMF	Comment
IN2P3-CC	FR	Gilles MATHIEU	Mattieu Puel	OpenStack	
KISTI	KR	Soonwook Hwang	Sangwan Kim, Taesang Huh, Jae-Hyuck Kwak	OpenStack	64 cores with 256 GB RAM and 6TB HDD
NCG-INGRID-PT / LIP	PT	Mario David	Joao Pina, Joao Martins	Openstack	

표 5 EGI Federated Cloud Resource Provider (Interested RPs)

Affiliation	CC	Main Contact	Deputies	CMF	Comment
IICT-BAS	BG	Emanouil Atanassov	Todor Gurov	OpenStack	
INFN - Napoli	IT	Silvio Pardi		OpenStack	
INFN - CNAF	IT	Elisabetta Ronchieri	Davide Salomoni, Andrea Cristofori	OpenStack	
INFN - Torino	IT	Andrea Guarise		OpenStack	
CIEMAT	ES	Rubio Montero	Rafael Mayo García, Manuel Aurelio Rodríguez Pascual	OpenNebula	
SURFsara	NL	Jhon Masschelein	Maurice Bouwhuis, Machiel Jansen	OpenNebula	
CRNS/IPHC	FR	Jérôme Pansanel		OpenStack	
ISRGrid/IUCC	IL	Yossi Baruch			
DESY	DE	Patrick Furhmann	Paul Millar	dCache	Storage only
STFC/RAL	UK	Ian Collier	Frazer Barnsley, Alan Kyffin		
STFC/RAL Harwell Science	UK	Jens Jensen		Castor	Storage only
IFAE / PIC	ES	Victor Mendez			
SAGrid	ZA	Bruce Becker			
CSC	FI	Jura Tarus	Luís Alves, Ulf Tigerstedt, Kalle Happonen	OpenStack	Status: Testing
SRCE	HR	Emir Imamagic	Luko Gjenero	OpenStack	Status: deploying OpenStack cluster
GridPP	UK	Adam Huffman		OpenStack	
Polythecnic University of Valencia, I3M	ES	Ignacio Blanquer		OpenNEbula & OpenStack	OpenNebula 60 cores, 192GB RAM OpenStack 40 cores, 36 GB RAM
In OST: 40 cores, 36 GBs de RAM.					
CRNS/IN2P3-LAL	FR	Michel Jouvin	Mohammed Araj	StratusLab	

제3절 워킹 그룹

FedCloud Task Force의 활동은 워킹 그룹으로 나뉘어져 있다.

1. Work Group Scenario 1: VM Management
   - Scenario Leader : Boris Parák (CESNET)
   - 워킹 그룹 주제
   - Trust level and auditing of the VM
   - Different VMs needed based on underlying infrastructure such as 64 vs 32bits OR VT enabled plus Xen vs KVM
   - Contextualization (how users should login to this vm, how his public key transfer and active to login as root to this vm)
   - Which libraries/versions/compilers will be installed by default?
   - 미리 정의된 VM 이미지에 소프트웨어 설치
   - 설치된 이미지를 저장하여 재사용하기
   - VM에서 persistent local storage 에 접근하기

2. Work Group Scenario 2: Data Management
   - Scenario Leader : Matteo Turilli (Oxford e-Research Centre(OeRC))
   - 워킹 그룹 주제
   - WeNMR use cases :Using VMs prepared with Gromacs and some other software to run MD simulations for educational purpose, possibly on multi-core VMs
   - stoxy(STOrage proXY) - CDMI compliant server [1]
   - dCache[2] CDMI interface

3. Work Group Scenario 3: Information Systems
   - Scenario Leader : David Wallom (Oxford e-Research Centre(OeRC))
   - 워킹 그룹 주제

   정보시스템에서 다루어 지는 정보의 종류는 다음과 같다.
   - What is the name of the resource?
   - What type of interface can I use to manage instances on the resource?
   - What is the endpoint I should contact to interact with the cloud management interface? (E.g. the url of the web-service/portal)
   - What are the AuthN and AuthZ rules that operate on your cloud?
   - What instances are already installed on the resource and am I allowed to upload my own instances?
   - If I am able to upload instances what format of instances does the resource accept?
   - Is there a data interface available and if so what is it?

- What is the overall size of the resource?
- Are instance templates defined that limit the choice of instance scales I am able to run?
- What type of virtual network can I establish on the resource?
- Does the resource support cloud scalability through managed bursting to another external provider?

4. Work Group Scenario 4: Accounting

- Scenario Leader : Alison Packer (Science & Technology Facilities Council | STFC)
- 워킹 그룹 주제

Resource accounting을 위해서 정의되어야 할 것들:

- the elements to be accounted for and how accounting data may be gathered. (This relates to the statement "Resource providers agree and follow the same rules for accounting the resource usage".)
- how accounting data will be published
- what is required from this data (for VM users, Resource Providers, VO Managers, others?)
- Accounting Records [3]

5. Work Group Scenario 5: Monitoring

- Scenario Leader : Emir Imamagic (University of Zagreb University Computing Centre | SRCE)
- 워킹 그룹 주제

- Monitoring in this context is the monitoring of the availability and reliability of the cloud resources provided by the resource providers. What will be tested is the possibility for an hypothetical user to instantiate at least one predefined virtual machine within a given period of time. It consists of an "external" monitoring, no data will be collected from inside the VMs or underlying virtualization systems. Monitoring the capabilities of the cloud resource providers in terms of how many resources are available is beyond the scope of this Scenario, at least in its initial phase. Possible evolution of the FedCloud monitoring will be evaluated when the basic monitoring will be in place.
- Given the experience accumulated with the NAGIOS [4] system within the EMI and EGI projects the monitoring framework will be based on NAGIOS. This has also the advantage to ease the integration of the FedCloud monitoring framework in the SAM monitoring sytem used by the EGI project to monitor the production infrastructure.
- Integration with EGI operational tools (GOCDB)

The following service types can be added to GOCDB:

eu.egi.cloud.accounting (required)

eu.egi.cloud.information.bdii (required)

eu.egi.cloud.storage-management.cdmi
eu.egi.cloud.vm-management.occi (required)
eu.egi.cloud.vm-metadata.marketplace

6. Work Group Scenario 6: Notification

- Scenario Leader : Peter Solagna (EGI.eu)
- 워킹 그룹 주제

The scenario enables:

- Resource providers to notify interested parties about changes in the user's VM state of a proposed state change in the VMM.
- End user to build automated VM management and reactive workflows in user space.

Such notification system must enable automatic notification consumers and so there is the need to identify the standards to be used:

- Standard for the payload in the notification messages
- Standard for the messages transport

If such standards are available in the federated cloud infrastructure, the users will be able to implement monitoring/management tools for their workflows.

7. Work Group Scenario 7: Federated AAI

- Scenario Leader : Paul Millar (Deutsches Elektronen-Synchrotron | DESY)
- 워킹 그룹 주제

- We have already defined that user authentication should be based on X.509 certificates rather than usernames and passwords or other credential material. Nevertheless, depending on the type of federation intended, this may not even be a real requirement. Any service should rely on an identity provider that is in charge of the type of credentials used for authentication.

For the technical implementations of this scenario, please go to Federated AAI Implementation [5].

A quick overview of AAI support in technologies and providers, as well as the specific settings for FCTF can be found at Federated AAI Integration Status.

8. Work Group Scenario 8: VM Image Management

- Scenario Leader : Marios Chatziangelou (Institute of Accelerating Systems | IASA)
- 워킹 그룹 주제

- VM image management open issues:

Provide a mechanism so that a user can upload transparently his own image to the Fedcloud testbed, with a unique global ID.

Provide a common place to add an endorsement to a pertinent VM so that it can be trusted by the resource providers.

9. Work Group Scenario 9: Brokering

- Scenario Leader : Iván Díaz (JR2)
- 워킹 그룹 주제

- This workgroup deals with the issues around cloud brokering. With 10+ resource providers lined up to be federated into the EGI cloud testbed, users need effective ways to access cloud resources. The goal is for a user to have a choice between a unified, abstracted view of the cloud testbed as a whole and the opportunity to target specific providers for their needs. As a consequence, this workgroup is concerned with both brokers and OCCI clients.

10. Work Group Scenario 10: Contextualisation

- Scenario Leader : Enol Fernandez (Instituto de Física de Cantabria | IFCA)
- 워킹 그룹 주제

- VM contextualization open issues, namely being able to pass user-defined data to the VM.

Contextualization is the process of installing, configuring and preparing software upon boot time on a pre-defined virtual machine image (e.g. setting the hostname, IP addresses, SSH aithorized keys, starting services, installing applications, etc.). We have identified as a requirement to contextualize images the possibility of passing user provided data to the VM when they are instantiated. Hence there are two things to be defined:

a. how to pass data upon VM creation (the exact type and format of the data should not be relevant, it should be up to the user)

b. how to retrieve those data from the running VM

For passing the data we have proposed the use of a new OCCI mixin that has an attribute to hold the data to pass to the image. The second part, related to retrieving the data, is more dependent of the back-end implementation. There are different methods in the systems in place in FedCloud, but tools like cloud-init [6] handle those possible differences in a transparent way for the users.

[1] https://github.com/stoxy/stoxy
[2] http://www.dcache.org/
[3] https://code.grnet.gr/documents/18
[4] http://www.nagios.org/
[5] https://wiki.egi.eu/wiki/Federated_AAI_Implementation
[6] https://launchpad.net/cloud-init

제4절 Federated Cloud 기술

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

- 관리를 위한 인터페이스
  - VM management interface: OCCI
  - Data management interface: CDMI
  - Virtual Organisation Management & AAI: VOMS
  - VM Image management
- 서비스를 위한 인터페이스
  - Information discovery: BDII
  - Central service registry: GOCDB
  - Monitoring: SAM
  - Accounting
  - Image metadata publishing & repository

자세한 내용은 Technology Architecture [1] 를 참조

[1] https://wiki.egi.eu/wiki/Fedcloud-tf:Technology:Architecture

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 [1] : IaaS 클라우드에 접근하고 관리하기 위한 명세. OCCI 자체는 IaaS에만 국한 된것이 아니므로 확장가능하다.

CDMI [2] : 클라우드 스토리지를 위한 인터페이스, 메터데이터 관리, 데이터 접근 프로토콜에 대한 명세

GLUE2 [3] , CLUE+: information model 에 대한 명세. 처음에는 그리드 자원에 대한 목적으로 정의되었으나, 클라우드 자원에 대한 확장으로 현재 논의중

SAML [4] : authentication 와 authorization 에 관한 명세

UR2 [5] : GLUE2와 마찬가지로 grid 자원에 대한 목적으로 처음 개발되었으나 클라우드 자원의 accounting record 에 대한 정의 까지도 포함하고 있음.

OVF [6] : virtual machine image와 기본적인 deployment, contextualization 방법에 대한 정보를 저장하는 명세. virtual appliances를 관리하기 위한 출발 점.

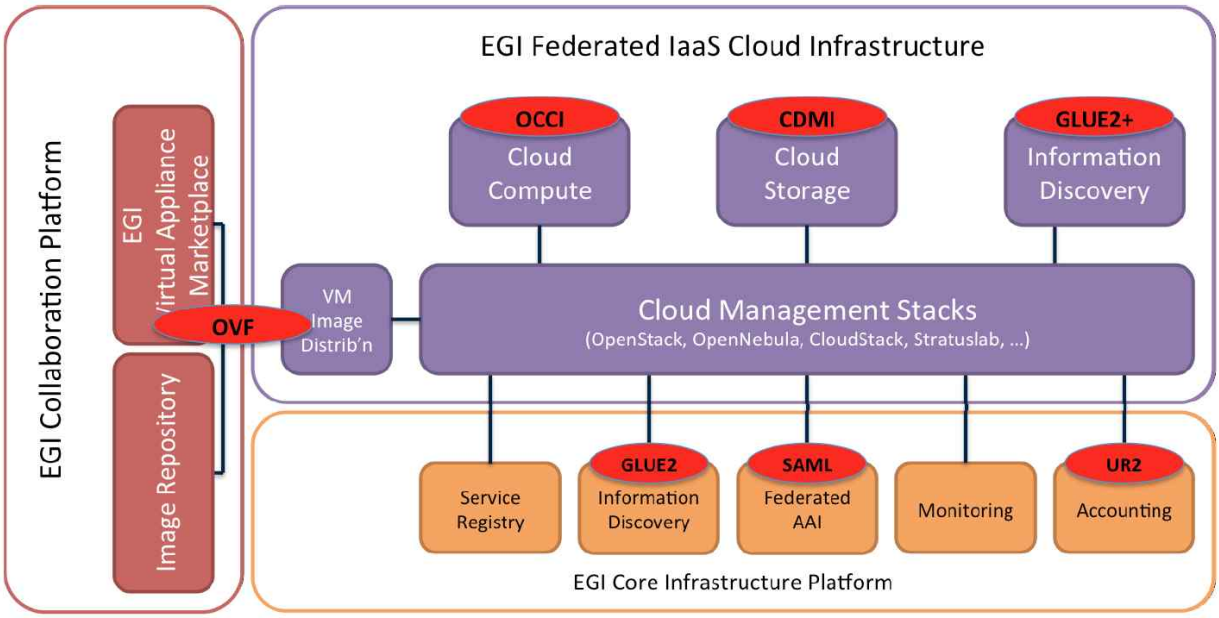


그림 2 EGI Federated Cloud Infrastructure Platform architecture and standards

[1] http://occi-wg.org/about/specification/
[2] http://www.snia.org/cdmi
[3] http://www.ogf.org/documents/GFD.147.pdf
[4] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security
[5] https://www.ogf.org/documents/GFD.98.pdf
[6] http://www.dmtf.org/standards/ovf

1. GLUE Specification v2.0 요약

GFD.147 문서[1]는 grid community 에서 사용되는 GLUE information model 을 기술하고 있다.

[1] http://www.ogf.org/documents/GFD.147.pdf (2009)

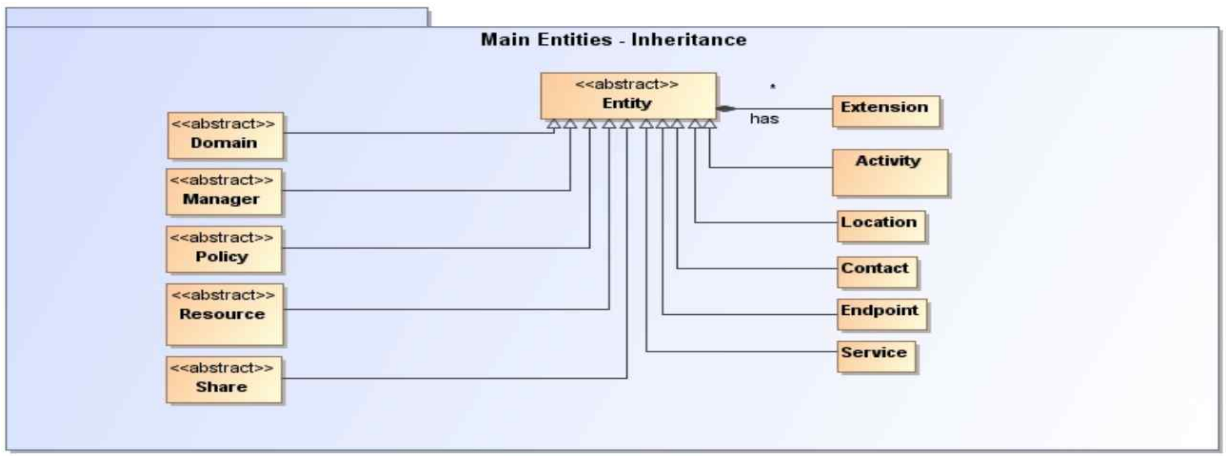


그림 3 GLUE's Main Entities - Inheritance (UML diagram)

표 6 GLUE Specification - Main Entities

Entity		**Extension**	
Attribute	Type	Attribute	Type
Creation Time	DateTime	LocalID	LocalID_t
Validity	UInt64	Key	String
ID	URI	Value	String
Name	String		
OtherInfo	String		
Location		**Contact**	
Attribute	Type	Attribute	Type
Address	String	Detail	URI
Place	String	Type	ContactType_t
Country	String		
PostCode	String		
Latitude	Real32		
Longitude	Real32		
Domain		**Service**	
Attribute	Type	Attribute	Type
Description	String	Capability	Capability_t
WWW	URL	Type	ServiceType_t
		QualityLevel	QualityLevel_t
		StatusInfo	URL
		Complexity	String
Endpoint		**Share**	
Attribute	Type	Attribute	Type
URL	URL	Descriptioin	String
Capability	Capability_t		
Technology	EndpointTechnology_t	**Manager**	
InterfaceName	InterfaceName_t	Attribute	Type
InterfaceVersion	String	ProductName	String
InterfaceExtension	URI	ProductVersion	String
WSDL	URL		
SupportedProfile	URI	**Resource**	
Semantics	URL	Attribute	Type
Implementor	String		
ImplementationName	String	**Activity**	
ImplementationVersion	String	Attribute	Type
QualityLevel	QualityLevel_t		
HealthState	EndpointHealthState_t	**Policy**	
HealthStateInfo	String	Attribute	Type
ServingState	ServingState_t	Scheme	PolicyScheme_t
StartTime	DateTime_t	Rule	String
IssuerCA	DN_t		
TrustedCA	DN_t		
DowntimeAnounce	DateTime_t		
DowntimeStart	DateTime_t		
DowntimeEnd	DateTime_t		
DowntimeInfo	String		

제5절 사용자 커뮤니티

EGI Federated Cloud 자원을 활용하고 있는 사용자 커뮤니티에 대한 정보는 [1][2]에서 찾을 수 있다. 대표적인 사용자 커뮤니터를 정리하면 다음과 같다.

- CERN ATLAS [3] : ATLAS (A Toroidal LHC Apparatus) 실험시 그리드 작업을 처리하기 위한 on-demand 자원 제공
- ENVRI [4] : Data access, catalog and dissemination (EISCAT 3D)
- BioVel [5] : workflows for the processing of data in major areas of biodiversity research: ecological niche modelling, ecosystem functioning, and taxonomy
- DIRAC : VMDIRAC portal as VM scheduler with 3rd party broker (with WeNMR)
- CHAIN-REDS [6] : WRF(weather research forecast) 날씨 예측 모델, R(통계 분석 프로그램)
- VERCE [7] : 웹서비스(Science Gateway 도구)
- SCI-BUS [8] : Test/debug new releases of WS-PGRADE(Workflow)

그림 4 EGI Federated Cloud User Communities

[1] https://wiki.egi.eu/wiki/Federated_Cloud_user_support
[2] https://wiki.egi.eu/wiki/Federated_Cloud_Communitie
[3] http://atlas.ch/
[4] http://envri.eu/
[5] http://www.biovel.eu/
[6] http://www.chain-project.eu/
[7] http://www.verce.eu/
[8] http://www.sci-bus.eu/

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 5 Federated Cloud Resources in KISTI

제2절 OpenStack

1. 설치준비

OpenStack 설치 가이드는 다음 주소를 참고한다.

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

openstack-install-guide-yum-havana.pdf

Controller 노드로 사용할 장비는 DNS에 등록을 한다.

```
# nslookup  fccont.kisti.re.kr
Name:    fccont.kisti.re.kr
Address: 150.183.250.170
```

계산 노드는 /etc/hosts 등록하여 준다.

```
# vi /etc/hosts
150.183.250.170     fccont.kisti.re.kr
150.183.250.171     fccomp1.kisti.re.kr
150.183.250.172     fccomp2.kisti.re.kr
```

노드간 네트워크 통신을 위해 iptables 규칙을 설정한다.

```
# vi /etc/sysconfig/iptables
# TRUSTED NETWORK
-A INPUT -s 150.183.250.0/24 -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
```

2. OpenStack Controller Node 설치

(1) 설치 준비

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 packages 설치를 위한 yum repository 설치

Repository URL : http://repos.fedorapeople.org/repos/openstack/openstack-havana/epel-6/

```
# yum -y install http://repos.fedorapeople.org/repos/openstack/openstack-havana/rdo-release
-havana-6.noarch.rpm
```

EPEL 확장 리포지터리 설치

```
# yum -y install http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
```

OpenStack 패키지 설치

```
# yum -y install openstack-utils
# yum -y install openstack-selinux
```

OpenStack 서비스에서 사용되는 포트는 다음과 같다.

Service	TCP port
qpid	5672
keystone-all	35357
keystone-all	5000
glance-registry	9191
glance-api	9292
nova-novncproxy	6080
nova-xvpvncproxy	6081
nova-objectstore	3333
nova-api-metadata	8775
cinder-api	8776

(2) Messaging Server 설치

```
# yum -y install qpid-cpp-server  memcached
# sed --in-place 's/auth=yes/auth=no/'  /etc/qpidd.conf
# service qpidd start
# chkconfig qpidd on
```

메시지 서비스 포트정보

서비스	포트
qpid	5672

(3) Installing and Configuring the Identity Service

```
# yum -y install   openstack-keystone   python-keystoneclient
```

keystone python 패키지 설치 경로:

```
cd /usr/lib/python2.6/site-packages/keystone
```

사용자 keystone 이 추가된다.

```
# id keystone
uid=163(keystone) gid=163(keystone) groups=163(keystone)
```

기존 데이터베이스가 있으면 삭제한다.

```
# openstack-db --drop --service keystone
```

데이터베이스 테이블 생성

```
# openstack-db --init --service keystone --password  keystone  --rootpw 12345678
Verified connectivity to MySQL.
Creating 'keystone' database.
Initializing the keystone database, please wait...
Complete!
# echo "show tables from keystone" | mysql -u root -p######## keystone
```

데이터베이스 정보 설정

```
openstack-config --set /etc/keystone/keystone.conf ₩
   sql connection mysql://keystone:keystone@127.0.0.1/keystone
```

인증 토큰을 생성하기

```
# openssl rand -hex 10
9ddf8555a62b3b11c0c0
# export ADMIN_TOKEN=9ddf8555a62b3b11c0c0
# openstack-config --set /etc/keystone/keystone.conf DEFAULT admin_token $ADMIN_TOKEN
```

keystone 은 PKI token 을 사용하므로 PKI key와 certificate 를 생성한다.

```
# ls -la /etc/keystone/*
# ls -la /etc/keystone/ssl/
# keystone-manage pki_setup  --keystone-user keystone --keystone-group keystone
# ls -la /etc/keystone/ssl/certs/*
# ls -la /etc/keystone/ssl/private/*
# chown -R keystone:keystone /etc/keystone/*
```

인증서의 subject 와 날짜를 확인

```
# openssl x509 -in /etc/keystone/ssl/certs/ca.pem -subject -dates -noout
subject= /C=US/ST=Unset/L=Unset/O=Unset/CN=www.example.com
notBefore=Apr  9 03:13:45 2014 GMT
notAfter=Apr  6 03:13:45 2024 GMT
```

Identity 서비스를 시작한다.

```
# service openstack-keystone start
# chkconfig openstack-keystone  on
```

keystone 서비스 포트 정보

서비스	포트
keystone-all	35357
keystone-all	5000

keystone 관련 로그파일 경로

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

(4) 사용자(user), 테넌트(tenants), 롤(role) 설정

```
# echo $ADMIN_TOKEN
# export OS_SERVICE_TOKEN=$ADMIN_TOKEN
# export OS_SERVICE_ENDPOINT=http://fccont.kisti.re.kr:35357/v2.0
```

관리자를 위한 테넌트 생성, 다른 서비스를 위한 테넌트 생성

```
# keystone tenant-create --name=admin --description="Admin Tenant"
# keystone tenant-create --name=service --description="Service Tenant"
# keystone tenant-create --name=fedcloud   --description="FedCloud Tenant"
```

아이디 생성

```
# keystone user-create --name=admin --pass=admin    --email=sangwan@kisti.re.kr
# keystone user-create --name=sangwan --pass=sangwan    --email=sangwan@kisti.re.kr
```

관리작업을 위한 admin 역할 생성 (policy.json 참고)

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

롤을 사용자에게 추가해야 함

```
# keystone user-role-add --user=admin --tenant=admin --role=admin
# keystone user-role-add --user=sangwan --tenant=fedcloud --role=_member_
# keystone user-role-add --user=admin --tenant=fedcloud --role=admin
# keystone user-role-list --user=admin --tenant=admin
+----------------------------------+-------+----------------------------------+----------------------------------+
|                id                |  name |             user_id              |            tenant_id             |
+----------------------------------+-------+----------------------------------+----------------------------------+
| ec178cf928e14fd29aed86abe0978261 | admin | 3c97866a483e44fc9a5fa35aabebf45d | 8165163a12074a2ea3809dbc63a11822 |
+----------------------------------+-------+----------------------------------+----------------------------------+
# keystone user-role-list --user=sangwan --tenant=fedcloud
+----------------------------------+----------+----------------------------------+----------------------------------+
|                id                |   name   |             user_id              |            tenant_id             |
+----------------------------------+----------+----------------------------------+----------------------------------+
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ | ad52cb62816a498a8eb5070b253c298a | 43a25673743144cb97e259f7d98197b9 |
+----------------------------------+----------+----------------------------------+----------------------------------+
```

Defining Services and API endpoints

identity 서비스 생성

```
# keystone  service-create  --name=keystone --type=identity --description="Identity Service
" | tee a
# export IDENTITY_SERVICE_ID=`grep ' id ' a | awk '{ print $4 }'`
# echo   IDENTITY_SERVICE_ID=$IDENTITY_SERVICE_ID
# keystone endpoint-create ₩
 --service-id=$IDENTITY_SERVICE_ID ₩
 --publicurl=http://fccont.kisti.re.kr:5000/v2.0 ₩
 --internalurl=http://fccont.kisti.re.kr:5000/v2.0 ₩
 --adminurl=http://fccont.kisti.re.kr:35357/v2.0 | tee a
```

Verifying the Identity Service Installation

identity 서비스를 검증하기 위해서 OS_SERVICE_TOKEN and OS_SERVICE_ENDPOINT 환경변수를

해제한다.

```
# unset OS_SERVICE_TOKEN OS_SERVICE_ENDPOINT
```

일반적인 username 기반의 인증을 이용한다.

```
# keystone --os-username=admin --os-password=admin --os-auth-url=http://fccont.kisti.re.kr:35357/v2.0   token-get
```

```
# keystone --os-username=sangwan --os-password=sangwan --os-auth-url=http://fccont.kisti.re.kr:35357/v2.0   token-get
```

암호가 틀린 경우 다음 에러 발생

```
The request you have made requires authentication. (HTTP 401)
```

주어진 테넌트 안에서 인증이 되는지 확인

```
# keystone --os-username=admin --os-password=admin --os-tenant-name=admin ₩
  --os-auth-url=http://fccont.kisti.re.kr:35357/v2.0   token-get
# keystone --os-username=sangwan --os-password=sangwan --os-tenant-name=fedcloud ₩
  --os-auth-url=http://fccont.kisti.re.kr:35357/v2.0   token-get
```

환경설정 방법

```
# vi ~/openrc.sh
export OS_USERNAME=admin
export OS_PASSWORD=admin
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://fccont.kisti.re.kr:35357/v2.0
# source ~/openrc.sh
# keystone   token-get
```

(5) Image Service 설정

참고 : http://docs.openstack.org/havana/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 :

이미지 서비스 설치

```
# yum -y  install openstack-glance
```

데이터베이스 설정

```
# openstack-config --set /etc/glance/glance-api.conf ₩
   DEFAULT sql_connection mysql://glance:glance@127.0.0.1/glance
# openstack-config --set /etc/glance/glance-registry.conf ₩
   DEFAULT sql_connection mysql://glance:glance@127.0.0.1/glance
```

create database and tables

```
# openstack-db --init --service glance --password  glance  --rootpw 12345678
```

create a glance user (password is glance)

```
keystone user-create   --name=glance --pass=glance --email=sangwan@kisti.re.kr
keystone user-role-add --user=glance --tenant=service --role=admin
```

glance 설정파일 (glance-api.conf)

```
# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken ₩
   auth_uri http://fccont.kisti.re.kr:5000
# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken ₩
   auth_host fccont.kisti.re.kr
# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken ₩
   admin_tenant_name service
# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken ₩
   admin_user glance
# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken ₩
   admin_password  glance
# openstack-config --set /etc/glance/glance-api.conf paste_deploy ₩
   flavor keystone
```

glance 설정파일 (glance-registry.conf)

```
# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken ₩
   auth_uri http://fccont.kisti.re.kr:5000
# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken ₩
   auth_host fccont.kisti.re.kr
# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken ₩
   admin_tenant_name service
# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken ₩
   admin_user glance
# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken ₩
   admin_password  glance
# openstack-config --set /etc/glance/glance-registry.conf paste_deploy ₩
   flavor keystone
```

```
# cp /usr/share/glance/glance-api-dist-paste.ini       /etc/glance/glance-api-paste.ini
```

```
# cp /usr/share/glance/glance-registry-dist-paste.ini /etc/glance/glance-registry-paste.ini
```

이미지 서비스를 identity 서비스에 등록

```
# keystone service-create --name=glance --type=image  --description="Glance Image Service"
| tee a
# export SERVICE_ID=`grep ' id ' a | awk '{ print $4 }'`
# echo   SERVICE_ID=$SERVICE_ID
# keystone endpoint-create ₩
  --service-id=$SERVICE_ID ₩
    --publicurl=http://fccont.kisti.re.kr:9292 ₩
  --internalurl=http://fccont.kisti.re.kr:9292 ₩
     --adminurl=http://fccont.kisti.re.kr:9292
```

sheepdog 패키지 설치 (Distributed Object Storage System for KVM/QEMU)

```
# rpm -qi sheepdog
```

이미지 서비스 시작하기

```
# service openstack-glance-api      restart
# service openstack-glance-registry restart
# chkconfig openstack-glance-api on
# chkconfig openstack-glance-registry on
```

이미지 서비스 포트정보

서비스	포트
glance-registry	9191
glance-api	9292

이미지 로그파일 경로

```
# chown  glance.glance  /var/log/glance/*.log
# less    /var/log/glance/registry.log
# less    /var/log/glance/api.log
```

(6) Verifying the Image Service Installation

샘플 이미지 다운로드

```
# mkdir /tmp/images
( cd /tmp/images/ ; wget http://download.cirros-cloud.net/0.3.1/cirros-0.3.1-x86_64-disk.im
g  )
# file  /tmp/images/cirros-0.3.1-x86_64-disk.img
cirros-0.3.1-x86_64-disk.img: Qemu Image, Format: Qcow , Version: 2
```

이미지 생성하기

```
# glance image-create --name="CirrOS 0.3.1" --disk-format=qcow2 ₩
  --container-format=bare --is-public=true <  /tmp/images/cirros-0.3.1-x86_64-disk.img
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | d972013792949d0d3ba628fbe8685bce     |
| container_format | bare                                 |
| created_at       | 2014-04-09T04:08:33                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | 30035163-a17d-4747-8969-f95f360d5ce3 |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | CirrOS 0.3.1                         |
| owner            | c42a3b24b4fe46178deb9730d4cc8fc9     |
| protected        | False                                |
| size             | 13147648                             |
| status           | active                               |
| updated_at       | 2014-04-09T04:08:34                  |
+------------------+--------------------------------------+

# glance image-list
+--------------------------------------+-------------+------------+------------------+----------+--------+
| ID                                   | Name        | Disk Format | Container Format | Size     | Status |
+--------------------------------------+-------------+------------+------------------+----------+--------+
| 30035163-a17d-4747-8969-f95f360d5ce3 | CirrOS 0.3.1 | qcow2       | bare             | 13147648 | active |
+--------------------------------------+-------------+------------+------------------+----------+--------+
```

(7) OpenStack Compute Service (nova)

참고: http://docs.openstack.org/havana/install-guide/install/yum/content/compute-service.html

Compute interacts with the Identity Service for authentication, Image Service for images, and the Dashboard for the user and administrative interface.

nova 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 기록한다. (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

Install Compute controller services

```
# yum install -y  openstack-nova   python-novaclient
```

database setting

```
# openstack-config --set /etc/nova/nova.conf  database connection mysql://nova:nova@127.0.
0.1/nova
```

configure Compute to use Qpid message broker

```
# openstack-config --set /etc/nova/nova.conf ₩
  DEFAULT rpc_backend nova.openstack.common.rpc.impl_qpid
# openstack-config --set /etc/nova/nova.conf DEFAULT qpid_hostname  fccont.kisti.re.kr
```

Create database and tables

```
# openstack-db --init --service nova --password  nova  --rootpw ########
```

Set the my_ip, vncserver_listen, and vncserver_proxyclient_address configuration options to the

internal IP address of the controller node:

```
# openstack-config --set /etc/nova/nova.conf DEFAULT my_ip       10.1.1.1
# openstack-config --set /etc/nova/nova.conf DEFAULT vncserver_listen   10.1.1.1   0.0.0.0
# openstack-config --set /etc/nova/nova.conf DEFAULT vncserver_proxyclient_address   10.1.1.1
```

Create a nova user that Compute uses to authenticate with the Identity Service. Use the service tenant and give the user the admin role:

```
# keystone user-create   --name=nova --pass=nova --email=sangwan@kisti.re.kr
# keystone user-role-add --user=nova --tenant=service --role=admin
# openstack-config --set /etc/nova/nova.conf DEFAULT auth_strategy keystone
# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_host fccont.kisti.re.kr
# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_protocol http
# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_port 35357
# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_user nova
# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_tenant_name service
# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_password   nova
# openstack-config --set /etc/nova/nova.conf DEFAULT  api_paste_config  /etc/nova/api-paste.ini
```

서비스 생성

```
# keystone service-create --name=nova --type=compute --description="Nova Compute service" | tee a
# export SERVICE_ID=`grep ' id ' a | awk '{ print $4 }'`
# echo   SERVICE_ID=$SERVICE_ID
# keystone endpoint-create ₩
  --service-id=$SERVICE_ID ₩
    --publicurl=http://fccont.kisti.re.kr:8774/v2/%₩(tenant_id₩)s ₩
  --internalurl=http://fccont.kisti.re.kr:8774/v2/%₩(tenant_id₩)s ₩
     --adminurl=http://fccont.kisti.re.kr:8774/v2/%₩(tenant_id₩)s
```

서비스 시작하기

```
# service  openstack-nova-api           restart
# service  openstack-nova-metadata-api  restart
# service  openstack-nova-cert         restart
# service  openstack-nova-cells        restart
# service  openstack-nova-conductor    restart
# service  openstack-nova-network      restart
# service  openstack-nova-scheduler    restart
# service  openstack-nova-xvpvncproxy  restart
# service  openstack-nova-novncproxy   restart
# service  openstack-nova-console      restart
# service  openstack-nova-consoleauth  restart
```

nova 서비스 포트

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

3. OpenStack Compute Node 설치

(1) compute node 설치

mysql 클라이언트 라이브러리 설치

```
# yum  -y  install    mysql  MySQL-python
```

Enable the OpenStack packages for the distribution

```
# yum -y install http://repos.fedorapeople.org/repos/openstack/openstack-havana/rdo-release-havana-6.noarch.rpm
# yum -y install http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
# yum -y install openstack-utils
```

compute 서비스 설치

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

/etc/nova/nova.conf 수정

```
# openstack-config --set /etc/nova/nova.conf database connection mysql://nova:nova@fccont.kisti.re.kr/nova
# openstack-config --set /etc/nova/nova.conf DEFAULT auth_strategy keystone
# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_host  fccont.kisti.re.kr
# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_protocol http
# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_port 35357
# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_user nova
# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_tenant_name service
# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_password   nova
```

Qpid message broker 설정

```
# openstack-config --set /etc/nova/nova.conf ₩
  DEFAULT rpc_backend nova.openstack.common.rpc.impl_qpid
# openstack-config --set /etc/nova/nova.conf DEFAULT qpid_hostname  fccont.kisti.re.kr
```

Configure Compute to provide remote console access to instances.

```
# openstack-config --set /etc/nova/nova.conf DEFAULT my_ip  10.1.1.2
# openstack-config --set /etc/nova/nova.conf DEFAULT vnc_enabled True
# openstack-config --set /etc/nova/nova.conf DEFAULT vncserver_listen 0.0.0.0
# openstack-config --set /etc/nova/nova.conf DEFAULT vncserver_proxyclient_address  10.1.1.2
# openstack-config --set /etc/nova/nova.conf ₩
  DEFAULT novncproxy_base_url http://fccont.kisti.re.kr:6080/vnc_auto.html
```

Specify the host that runs the Image Service.

```
# openstack-config --set /etc/nova/nova.conf DEFAULT glance_host fccont.kisti.re.kr
```

```
# openstack-config --set /etc/nova/nova.conf DEFAULT api_paste_config  /etc/nova/api-paste.ini
```

```
# vi /etc/nova/api-paste.ini
[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
auth_host = fccont.kisti.re.kr
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = nova
```

Start the Compute service and configure it to start when the system boots.

```
# service libvirtd start
# service messagebus start
# chkconfig libvirtd on
# chkconfig messagebus on
# service openstack-nova-compute start
# chkconfig openstack-nova-compute on
```

Enable Networking

```
# yum install -y     openstack-nova-network
```

```
# openstack-config --set /etc/nova/nova.conf DEFAULT network_manager nova.network.manager.F
latDHCPManager
# openstack-config --set /etc/nova/nova.conf DEFAULT ₩
  firewall_driver nova.virt.libvirt.firewall.IptablesFirewallDriver
# openstack-config --set /etc/nova/nova.conf DEFAULT network_size 254
# openstack-config --set /etc/nova/nova.conf DEFAULT allow_same_net_traffic False
# openstack-config --set /etc/nova/nova.conf DEFAULT multi_host True
# openstack-config --set /etc/nova/nova.conf DEFAULT send_arp_for_ha True
# openstack-config --set /etc/nova/nova.conf DEFAULT share_dhcp_address True
# openstack-config --set /etc/nova/nova.conf DEFAULT force_dhcp_release True
# openstack-config --set /etc/nova/nova.conf DEFAULT flat_interface  em2
# openstack-config --set /etc/nova/nova.conf DEFAULT flat_network_bridge br100
# openstack-config --set /etc/nova/nova.conf DEFAULT public_interface  em1
```

```
# vi /etc/nova/nova.conf
[database]
connection = mysql://nova:nova@fccont.kisti.re.kr/nova
```

…

Provide a local metadata service that is reachable from instances on this compute node. Perform this step only on compute nodes that do not run the nova-api service.

```
# yum install -y  openstack-nova-api
# service openstack-nova-metadata-api restart
# chkconfig openstack-nova-metadata-api on
```

Start the network service and configure it to start when the system boots

```
# service openstack-nova-network restart
# chkconfig openstack-nova-network on
```

nova 서비스 포트

서비스	포트
nova-api-metadata	8775

```
# service  openstack-nova-api           restart
# service  openstack-nova-metadata-api  restart
# service  openstack-nova-cert          restart
# service  openstack-nova-cells         restart
# service  openstack-nova-conductor     restart
# service  openstack-nova-network       restart
# service  openstack-nova-scheduler     restart
```

로그파일 경로

```
# vi /var/log/nova/compute.log
# vi /var/log/nova/metadata-api.log
# vi /var/log/nova/network.log
```

4. Network 설정

네트워크 생성

```
# nova network-create vmnet --fixed-range-v4=10.1.2.0/24  --bridge=br100 --multi-host=T
# nova network-list
+--------------------------------------+-------+-------------+
| ID                                   | Label | Cidr        |
+--------------------------------------+-------+-------------+
| d4eb558c-71f1-47b4-ad78-cd947644de5c | vmnet | 10.1.2.0/24 |
+--------------------------------------+-------+-------------+
```

삭제하려면 다음과 같이 하면 됨

```
# nova   net-delete   d4eb558c-71f1-47b4-ad78-cd947644de5c
```

상세정보 조회

```
# nova network-show vmnet
+---------------------+--------------------------------------+
| Property            | Value                                |
+---------------------+--------------------------------------+
| bridge              | br100                                |
| bridge_interface    | -                                    |
| broadcast           | 10.1.2.255                           |
| cidr                | 10.1.2.0/24                          |
| cidr_v6             | -                                    |
| created_at          | 2014-07-04T07:20:07.000000           |
| deleted             | 0                                    |
| deleted_at          | -                                    |
| dhcp_start          | 10.1.2.2                             |
| dns1                | 8.8.4.4                              |
| dns2                | -                                    |
| gateway             | 10.1.2.1                             |
| gateway_v6          | -                                    |
| host                | -                                    |
| id                  | ccf834ad-3817-4916-b6c0-799fe889f5c0 |
| injected            | False                                |
| label               | vmnet                                |
| multi_host          | True                                 |
| netmask             | 255.255.255.0                        |
| netmask_v6          | -                                    |
| priority            | -                                    |
| project_id          | -                                    |
| rxtx_base           | -                                    |
| updated_at          | -                                    |
| vlan                | -                                    |
| vpn_private_address | -                                    |
| vpn_public_address  | -                                    |
| vpn_public_port     | -                                    |
+---------------------+--------------------------------------+
```

호스트 정보 보기

```
# nova  host-list
+--------------------+-------------+----------+
| host_name          | service     | zone     |
+--------------------+-------------+----------+
| fccont.kisti.re.kr | conductor   | internal |
| fccont.kisti.re.kr | cert        | internal |
| fccont.kisti.re.kr | cells       | internal |
| fccont.kisti.re.kr | scheduler   | internal |
| fccont.kisti.re.kr | consoleauth | internal |
| fccont.kisti.re.kr | console     | internal |
| fccont.kisti.re.kr | network     | internal |
| fccomp1.kisti.re.kr | compute     | nova     |
| fccomp1.kisti.re.kr | network     | internal |
+--------------------+-------------+----------+
```

```
# nova hypervisor-list
```

```
+----+---------------------+
| ID | Hypervisor hostname |
+----+---------------------+
| 1  | fccomp1.kisti.re.kr |
+----+---------------------+
```

```
# nova hypervisor-show 1
+--------------------------+----------------------------------+
| Property                 | Value                            |
+--------------------------+----------------------------------+
| cpu_info_arch            | x86_64                           |
| cpu_info_features        | ["bmi1", "perfctr_nb", "perfctr_core", "topoext", "nodeid_msr
", "tce", "lwp", "wdt", "skinit", "ibs", "osvw", "cr8legacy", "extapic", "cmp_legacy", "fxs
r_opt", "mmxext", "osxsave", "monitor", "ht", "vme"] |
| cpu_info_model           | Opteron_G5                       |
| cpu_info_topology_cores  | 64                               |
| cpu_info_topology_sockets | 1                               |
| cpu_info_topology_threads | 1                               |
| cpu_info_vendor          | AMD                              |
| current_workload         | 0                                |
| disk_available_least     | 35                               |
| free_disk_gb             | 49                               |
| free_ram_mb              | 257787                           |
| hypervisor_hostname      | fccomp1.kisti.re.kr              |
| hypervisor_type          | QEMU                             |
| hypervisor_version       | 12001                            |
| id                       | 1                                |
| local_gb                 | 49                               |
| local_gb_used            | 0                                |
| memory_mb                | 258299                           |
| memory_mb_used           | 512                              |
| running_vms              | 0                                |
| service_host             | fccomp1.kisti.re.kr              |
| service_id               | 5                                |
| vcpus                    | 64                               |
| vcpus_used               | 0                                |
+--------------------------+----------------------------------+
```

5. 인스턴스 실행하기

ssh key 생성하기

```
# ssh-keygen -t rsa
# nova keypair-add --pub_key   ~/.ssh/id_rsa.pub   mykey
```

```
# nova keypair-list
+-------+-------------------------------------------------+
| Name  | Fingerprint                                     |
+-------+-------------------------------------------------+
| mykey | 33:64:4f:29:61:d9:31:84:63:35:86:f1:37:01:77:84 |
+-------+-------------------------------------------------+
```

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 a
+--------------------------------------+--------------+--------+--------+
| ID                                   | Name         | Status | Server |
+--------------------------------------+--------------+--------+--------+
| e0ded5b7-b533-4a2c-a77a-63c101f77583 | CirrOS 0.3.1 | ACTIVE |        |
+--------------------------------------+--------------+--------+--------+
# export ID=`grep CirrOS a | awk '{ print $2 }'`
# echo $ID
```

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 |              |
+-------------+-----------+---------+-----------+--------------+
```

floating ip 범위 대역 생성하기

```
# nova-manage  floating  create --pool nova --ip_range 150.183.250.176/29
# nova-manage floating list
None    150.183.250.177 None    nova    eth0
None    150.183.250.178 None    nova    eth0
None    150.183.250.179 None    nova    eth0
None    150.183.250.180 None    nova    eth0
None    150.183.250.181 None    nova    eth0
None    150.183.250.182 None    nova    eth0
```

삭제할 때는 다음과 같이 한다.

```
# nova-manage floating delete 150.183.250.176/29
```

인스턴스 실행하기

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

```
# export flavor=3
# nova boot --flavor $flavor  --key_name mykey  --image $ID  --security_group default test
+--------------------------------------+-----------------------------------------------------+
| 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-00000001                                   |
| 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                            | YC2irZBXYHNq                                        |
| config_drive                         |                                                     |
| created                              | 2014-07-04T07:22:55Z                                |
| flavor                               | m1.medium (3)                                       |
| hostId                               |                                                     |
| id                                   | e3705164-7fb7-41d2-a68c-6c0b396aba16                |
| image                                | CirrOS 0.3.1 (06a773b6-d904-4084-962f-32874dd2336a) |
| key_name                             | mykey                                               |
| metadata                             | {}                                                  |
| name                                 | cirrOS                                              |
| os-extended-volumes:volumes_attached | []                                                  |
| progress                             | 0                                                   |
| security_groups                      | default                                             |
| status                               | BUILD                                               |
| tenant_id                            | e4c9ba1b147d4cab8ab5bd5e0c8c9524                    |
| updated                              | 2014-07-04T07:22:55Z                                |
| user_id                              | 7798cd02b3d04017bc611f6a0230a365                    |
+--------------------------------------+-----------------------------------------------------+
```

가상머신 인스턴스 리스트

```
# nova list
+--------------------------------------+--------+--------+------------+-------------+----------------+
| ID                                   | Name   | Status | Task State | Power State | Networks       |
+--------------------------------------+--------+--------+------------+-------------+----------------+
| 3a92cc6f-319f-4030-8e48-37c6e5066305 | cirr0S | BUILD  | spawning   | NOSTATE     | vmnet=10.1.1.2 |
.........
| 3a92cc6f-319f-4030-8e48-37c6e5066305 | cirr0S | ACTIVE | -          | Running     | vmnet=10.1.1.2 |
.........
```

계산 노드에서 virsh list 로 생성된 인스턴스를 확인

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

nova-manage 사용법

```
# nova-manage
usage: nova-manage [-h] [--config-dir DIR] [--config-file PATH] [--debug]
                   [--log-config PATH] [--log-date-format DATE_FORMAT]
                   [--log-dir LOG_DIR] [--log-file PATH] [--log-format FORMAT]
                   [--nodebug] [--nouse-syslog] [--noverbose]
                   [--syslog-log-facility SYSLOG_LOG_FACILITY] [--use-syslog]
                   [--verbose] [--version]


# nova-manage  service  list
Binary           Host                                 Zone             Status     State Updated_At
nova-conductor   fccont.kisti.re.kr                   internal         enabled    :-)   2014-09-01 06:05:23
nova-cert        fccont.kisti.re.kr                   internal         enabled    XXX   2014-08-28 10:33:39
nova-cells       fccont.kisti.re.kr                   internal         enabled    XXX   2014-08-28 10:33:39
nova-scheduler   fccont.kisti.re.kr                   internal         enabled    :-)   2014-09-01 06:05:24
nova-consoleauth fccont.kisti.re.kr                   internal         enabled    XXX   2014-08-28 10:33:40
nova-console     fccont.kisti.re.kr                   internal         enabled    XXX   2014-08-28 10:33:40
nova-network     fccont.kisti.re.kr                   internal         enabled    :-)   2014-09-01 06:05:25
nova-compute     fccomp1.kisti.re.kr                  nova             enabled    :-)   2014-09-01 06:05:24
nova-network     fccomp1.kisti.re.kr                  internal         enabled    :-)   2014-09-01 06:05:25
```

호스트 리스트

```
# nova-manage host list
host                            zone
fccont.kisti.re.kr              internal
fccomp1.kisti.re.kr             nova
fccomp2.kisti.re.kr             nova
```

VM 인스턴스의 콘솔로그 보기

```
# nova console-log  cirr0S
[    0.000000] Initializing cgroup subsys cpuset
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 3.2.0-37-virtual (buildd@allspice) (gcc version 4.6.3 (Ubuntu/Linaro 4.
6.3-1ubuntu5) ) #58-Ubuntu SMP Thu Jan 24 15:48:03 UTC 2013 (Ubuntu 3.2.0-37.58-virtual 3.2.35)
[    0.000000] Command line: LABEL=cirros-rootfs ro console=tty1 console=ttyS0
```

```
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   Centaur CentaurHauls
...
info: initramfs: up at 1.26
GROWROOT: CHANGED: partition=1 start=16065 old: size=64260 end=80325 new: size=83859300,end=83875365
info: initramfs loading root from /dev/vda1
info: /etc/init.d/rc.sysinit: up at 1.46
Starting logging: OK
Initializing random number generator... done.
Starting acpid: OK
cirros-ds 'local' up at 1.57
no results found for mode=local. up 1.60. searched: nocloud configdrive ec2
Starting network...
udhcpc (v1.20.1) started
Sending discover...
Sending select for 10.1.2.2...
Lease of 10.1.2.2 obtained, lease time 120
deleting routers
route: SIOCDELRT: No such process
adding dns 10.1.2.1
cirros-ds 'net' up at 1.74
checking http://169.254.169.254/2009-04-04/instance-id
successful after 1/20 tries: up 1.75. iid=i-00000004
found datasource (ec2, net)
Starting dropbear sshd: generating rsa key... generating dsa key... OK
/run/cirros/datasource/data/user-data was not '#!' or executable
=== network info ===
if-info: lo,up,127.0.0.1,8,::1
if-info: eth0,up,10.1.2.2,24,fe80::f816:3eff:fe6e:5543
ip-route:default via 10.1.2.1 dev eth0
ip-route:10.1.2.0/24 dev eth0  src 10.1.2.2
=== datasource: ec2 net ===
instance-id: i-00000004
name: N/A
availability-zone: nova
local-hostname: cirros.novalocal
launch-index: 0
=== cirros: current=0.3.1 uptime=4.11 ===
  ____                ____  ____
 / __/ __ ____ ____ / __ \/ __/
/ /__ / // __// __// /_/ /\ \
\___//_//_/  /_/   \____/___/
   http://cirros-cloud.net


login as 'cirros' user. default password: 'cubswin:)'. use 'sudo' for root.
cirros login:
```

ping 으로 인스턴스로 네트워크 연결 테스트

```
[fccomp1] ping 10.1.2.2
PING 10.1.2.2 (10.1.2.2) 56(84) bytes of data.
64 bytes from 10.1.2.2: icmp_seq=1 ttl=64 time=0.175 ms
64 bytes from 10.1.2.2: icmp_seq=2 ttl=64 time=0.122 ms
^C
--- 10.1.2.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1966ms
```

```
rtt min/avg/max/mdev = 0.122/0.148/0.175/0.029 ms
```

인스턴스로 ssh 로그인 하기

```
[root@fccomp1 ~]$ ssh -o StrictHostKeyChecking=no  -l cirros   10.1.2.2
cirros@10.1.2.2's password: cubswin:)
$ hostname
cirros
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:64:28:67
          inet addr:10.1.2.2  Bcast:10.1.2.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe64:2867/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:271 errors:0 dropped:0 overruns:0 frame:0
          TX packets:273 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:45166 (44.1 KiB)  TX bytes:45653 (44.5 KiB)
```

인스턴스 삭제 방법

```
# nova delete 3a92cc6f-319f-4030-8e48-37c6e5066305
```

6. Block Storage Service (cinder) 설치

참고:

http://docs.openstack.org/havana/install-guide/install/yum/content/block-storage-service.html

블록스토리지 서비스는 volume, volume snapshot, volume type을 관리한다. 다음 구성요소가 있다.

- cinder-api.
- cinder-volume
- cinder-scheduler daemon
- Messaging queue

패키지 설치

```
# yum   install -y   openstack-cinder openstack-utils openstack-selinux
```

MySQL 데이터베이스 설정.

```
# openstack-config --set /etc/cinder/cinder.conf ₩
  database connection mysql://cinder:cinder@fccont.kisti.re.kr/cinder
```

서비스 생성

```
# openstack-db --init --service cinder --password  cinder --rootpw ######
```

사용자 생성

```
# keystone user-create --name=cinder --pass=cinder  --email=cinder@example.com
# keystone user-role-add --user=cinder --tenant=service --role=admin
```

인증관련 설정

```
# vi /etc/cinder/api-paste.ini
[filter:authtoken]
paste.filter_factory=keystoneclient.middleware.auth_token:filter_factory
auth_host=fccont.kisti.re.kr
auth_port = 35357
auth_protocol = http
auth_uri = http://fccont.kisti.re.kr:5000
admin_tenant_name=service
admin_user=cinder
admin_password=cinder
```

메시지 브로커(qpid) 설정

```
# openstack-config --set /etc/cinder/cinder.conf  DEFAULT rpc_backend cinder.openstack.comm
on.rpc.impl_qpid
# openstack-config --set /etc/cinder/cinder.conf  DEFAULT qpid_hostname  fccont.kisti.re.kr
```

서비스 등록 작업

```
# keystone service-create --name=cinder --type=volume  --description="Cinder Volume Service
" | tee a
# export SERVICE_ID=`grep ' id ' a | awk '{ print $4 }'`
# echo   SERVICE_ID=$SERVICE_ID
# keystone endpoint-create ₩
  --service-id=$SERVICE_ID    ₩
  --publicurl=http://fccont.kisti.re.kr:8776/v1/%₩(tenant_id₩)s ₩
  --internalurl=http://fccont.kisti.re.kr:8776/v1/%₩(tenant_id₩)s ₩
  --adminurl=http://fccont.kisti.re.kr:8776/v1/%₩(tenant_id₩)s | tee a
```

버전2 서비스 API 등록

```
# keystone service-create --name=cinderv2 --type=volumev2 ₩
  --description="Cinder Volume Service V2"  | tee a
export IDENTITY_SERVICE_ID=`grep ' id ' a | awk '{ print $4 }'`
echo   IDENTITY_SERVICE_ID=$IDENTITY_SERVICE_ID
```

endpoint 생성

```
# keystone endpoint-create ₩
  --service-id=$SERVICE_ID ₩
  --publicurl=http://fccont.kisti.re.kr:8776/v2/%₩(tenant_id₩)s ₩
```

```
  --internalurl=http://fccont.kisti.re.kr:8776/v2/%₩(tenant_id₩)s ₩
  --adminurl=http://fccont.kisti.re.kr:8776/v2/%₩(tenant_id₩)s
```

cinder 서비스 시작

```
# service openstack-cinder-api start
# service openstack-cinder-scheduler start
# chkconfig openstack-cinder-api on
# chkconfig openstack-cinder-scheduler on
```

nova 설정에 추가

```
# vi /etc/nova/nova.conf
...
cinder_catalog_info=volume:cinder:http://fccont.kisti.re.kr:8776/
:wq
```

7. Dashboard 설치

패키지 설치

```
# yum install -y memcached python-memcached mod_wsgi openstack-dashboard
# rpm -ql mod_wsgi
# rpm -ql openstack-dashboard
```

dashboard 설정

```
# vi /etc/httpd/conf.d/openstack-dashboard.conf

WSGIDaemonProcess dashboard
WSGIProcessGroup dashboard
WSGISocketPrefix run/wsgi

WSGIScriptAlias /dashboard /usr/share/openstack-dashboard/openstack_dashboard/wsgi/django.w
sgi
Alias /static /usr/share/openstack-dashboard/static

<Directory /usr/share/openstack-dashboard/openstack_dashboard/wsgi>
  <IfModule mod_deflate.c>
    SetOutputFilter DEFLATE
    <IfModule mod_headers.c>
      # Make sure proxies don't deliver the wrong content
      Header append Vary User-Agent env=!dont-vary
    </IfModule>
  </IfModule>

  Order allow,deny
  Allow from all
</Directory>

<Directory /usr/share/openstack-dashboard/static>
```

```
  <IfModule mod_expires.c>
    ExpiresActive On
    ExpiresDefault "access 6 month"
  </IfModule>
  <IfModule mod_deflate.c>
    SetOutputFilter DEFLATE
  </IfModule>

  Order allow,deny
  Allow from all
</Directory>
```

memcached 포트 확인

```
# grep PORT /etc/sysconfig/memcached
PORT="11211"
```

local_settings 수정

/etc/openstack-dashboard/local_settings 수정후에 httpd를 재시작 해야 함

```
# vi /etc/openstack-dashboard/local_settings
   (/usr/share/openstack-dashboard/openstack_dashboard/local/local_settings.py)
:8
#DEBUG = False
DEBUG = True
:103
CACHES = {
    'default': {
        'BACKEND' : 'django.core.cache.backends.locmem.LocMemCache',
        'LOCATION': '127.0.0.1:11211'
    }
}
...
# https://docs.djangoproject.com/en/dev/ref/settings/#allowed-hosts
:15
ALLOWED_HOSTS = ['*']
...
:129
OPENSTACK_HOST = "127.0.0.1"
...
:wq
```

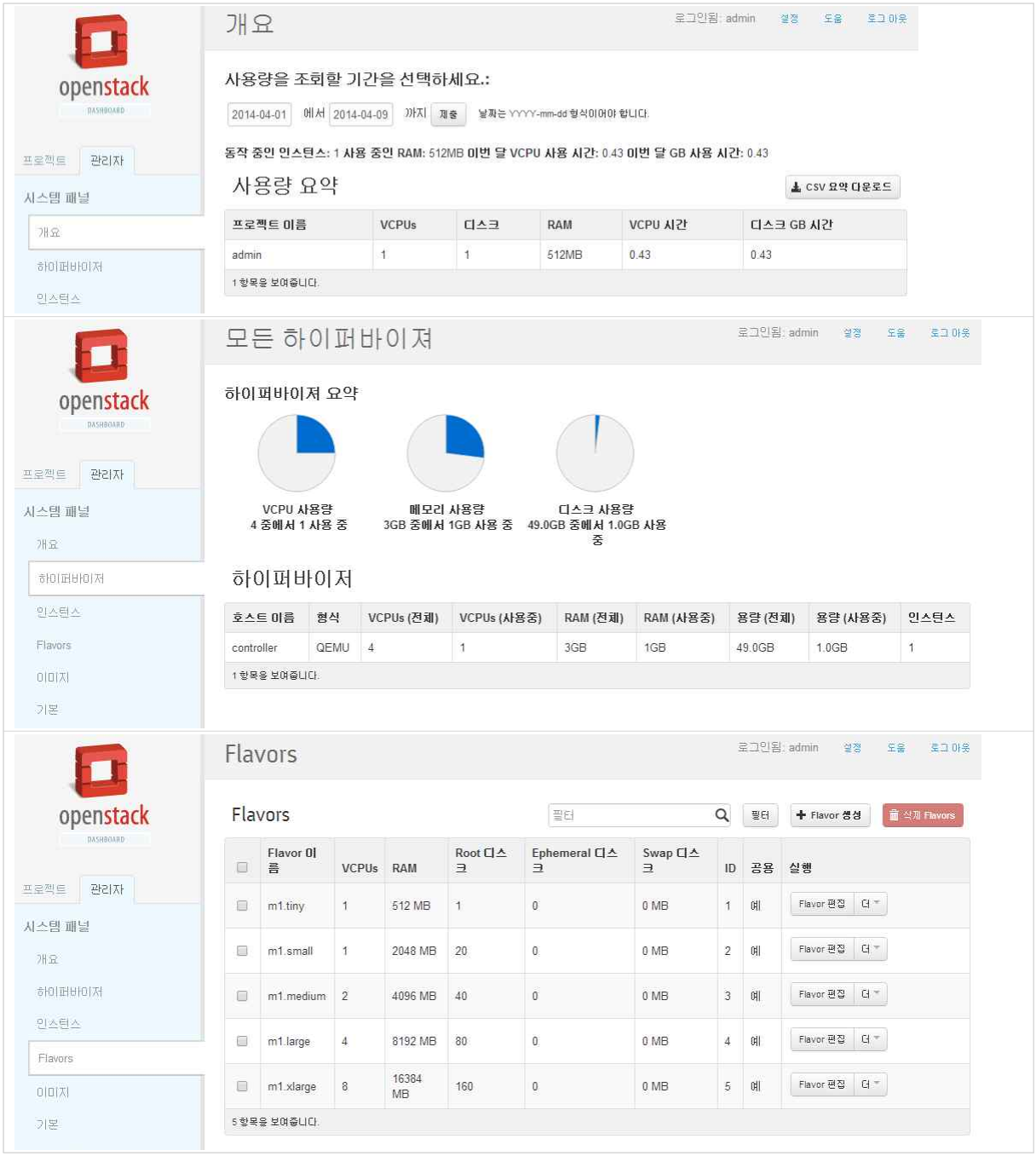
서비스 재시작

```
# service httpd     restart
# service memcached restart
# chkconfig httpd on
# chkconfig memcached on
```

dashboard 접속. 브라우저로 다음 주소로 접속한다.

http://fccont.kisti.re.kr/dashboard/

dashboard 접속 화면



제3절 VOMS

VOMS [1] 서비스는 RFC 3820 에 기반은 X.509 프록시를 만들수 있는데, -rfc 옵션을 커맨드 라인에 사용해야 한다. 보통의 X.509 인증서대신 이 프록시는 설정된 keystone 서버에 인증용도로 사용될 수 가 있다.

본 문서에서는 OpenStack Havana 버전의 Keystone이 VOMS 인증을 할 수 있도록 하는 방법을 설명한다. DB에 수정은 필요없으며, 외부의 플러그인 형태로 설치가 된다. 그러므로 다른 종류의 인증방법도 함께 이용이 가능하다.

VOMS 인증 모듈은 Keystone 이 http 서버의 WSGI application 으로 실행되어야 한다. http 서버의 SSL 은 enable 되어야 한다. 현재로서는 Keystone V2 API 에만 동작한다.

[1] http://en.wikipedia.org/wiki/VOMS

요구사항

Keystone VOMS 인증 모듈은 몇개의 추가적인 패키지를 요구한다. 또 keystone 을 WSGI application 으로 http 서버에서 실행해야 한다.

- Keystone >= Havana.
- EUgridPMA CA certificates at the latest version.
- fetch-crl package.
- VOMS libraries.
- HTTP server with WSGI enabled.

(1) 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
```

(2) VOMS 라이브러리 설치

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

keystone 설정 수정

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

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

(3) 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
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/havana/httpd/keystone.py
# ln  /usr/lib/cgi-bin/keystone/keystone.py /usr/lib/cgi-bin/keystone/main
# ln  /usr/lib/cgi-bin/keystone/keystone.py /usr/lib/cgi-bin/keystone/admin
# yum install -y python-paste-deploy
```

/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
```

httpd 서버를 시작한다.

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

아파치 로그에서 mod_ssl 과 mod_wsgi 의 동작을 확인한다.

```
# tail -f /var/log/httpd/error_log
[Mon Apr 28 14:47:09 2014] [notice] Apache/2.2.15 (Unix) DAV/2 mod_ssl/2.2.15 OpenSSL/1.0.1
```

```
e-fips mod_wsgi/3.2 Python/2.6.6 configured -- resuming normal operations
```

WSGI Socket 을 확인

```
# cd  /var/log/ ; ll *.sock
srwx------. 1 apache root 0 Jul 27 03:06 httpd.194515.1.1.sock
srwx------. 1 apache root 0 Jul 27 03:06 httpd.194515.1.2.sock
srwx------. 1 apache root 0 Jul 27 03:06 httpd.194515.1.3.sock
```

아파치 서버의 환경설정으로 OPENSSL_ALLOW_PROXY_CERTS 을 1 로 설정해야 한다. 이것은 X.509 프록시 인증서를 OpenSSL 에서 인식할 수 있게 해준다.

```
# vi /etc/init.d/httpd
...
export OPENSSL_ALLOW_PROXY_CERTS=1
...
:wq
# /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"}}}'
```

(4) SQL Token driver 설정

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

더 많은 디버깅 메시지를 원하면 keystone.conf 에 추가한다.

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

(5) 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/havana
# yum install -y  python-m2ext
# pip install . 2>&1 | tee i.log
...
Successfully installed python-keystone-voms
Cleaning up...
```

설치 확인

```
# pip  list | grep keystone
keystone (2013.2.3)
python-keystone-voms (2013.2)
python-keystoneclient (0.7.1)
```

설치경로

```
/usr/lib/python2.6/site-packages/keystone_voms
/usr/lib/python2.6/site-packages/python_keystone_voms-8749337-py2.6.egg-info
```

(6) Keystone VOMS module 활성화하기

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

```
# vi /etc/keystone/keystone-paste.ini
:51
[filter:voms]
paste.filter_factory = keystone_voms:VomsAuthNMiddleware.factory

[pipeline:public_api]
pipeline = access_log sizelimit url_normalize token_auth admin_token_auth xml_body json_bod
y ldap_ro_ifca ldap_ro_lip voms debug ec2_extension user_crud_extension public_service

[pipeline:public_api]
pipeline = access_log sizelimit url_normalize token_auth admin_token_auth xml_body json_bod
y  voms debug ec2_extension user_crud_extension public_service

[pipeline:admin_api]
pipeline = access_log sizelimit url_normalize token_auth admin_token_auth xml_body json_bod
y  voms debug ec2_extension s3_extension crud_extension admin_service
```

(7) 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 <<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
# cd /etc
# cat >>vomses <<EOF
"fedcloud.egi.eu" "voms1.egee.cesnet.cz" "15002" "/DC=org/DC=terena/DC=tcs/OU=Domain Contro
l Validated/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 Contro
l Validated/CN=voms2.grid.cesnet.cz" "fedcloud.egi.eu" "24"
EOF
```

VO to local tenant mapping 설정

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

(8) 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  voms1.egee.cesnet.cz:15002 [/DC=org/DC=terena/DC=tcs/OU=Domain Control Validate
d/CN=voms1.egee.cesnet.cz] "fedcloud.egi.eu" Done
Creating proxy ..............................................................................
..................................... Done
Your proxy is valid until Tue May 13 06:21:08 2014
```

프록시 정보 확인

```
# voms-proxy-info -file /tmp/x509up_u500 -all
subject   : /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim/CN=2119799137
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_u500.2
timeleft  : 11:59:45
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  : 11:59:34
uri       : voms1.egee.cesnet.cz:15002
```

VOMS 인증 테스트

```
# export X509_USER_PROXY=/tmp/x509up_u500
# /usr/local/curl/bin/curl  --cert $X509_USER_PROXY  -d '{"auth":{"voms": true}}' ₩
  --capath /etc/grid-security/certificates ₩
```

```
  -v ₩
  -H "Content-type: application/json" https://fccont.kisti.re.kr:5001/v2.0/tokens
{
    "access": {
        "token": {
            "expires": "2011-08-10T17:45:22.838440",
            "id": "0eed0ced-4667-4221-a0b2-24c91f242b0b"
        }
    }
}
```

(9) VOMS authentication client plugin

VOMS 인증 방식을 OpenStack 에서 활용하려면 nova client 의 authentication plugin 을 설치해야 한다.

플러그인 배포 주소 : https://github.com/IFCA/voms-auth-system-openstack

이 플러그인은 OpenStack 클라이언트가 VOMS 인증을 가능하도록 한다.

```
# pip install voms-auth-system-openstack
# git clone git://github.com/IFCA/voms-auth-system-openstack.git -b master
# pip install .
```

설치경로

```
 /usr/lib/python2.6/site-packages/voms_auth_system_openstack
```

설치확인

```
# pip list | grep voms-auth
voms-auth-system-openstack (1.0)
```

제거방법:

```
# pip uninstall voms-auth-system-openstack
```

CLI 에서 활용하는 방법

옵션 --os_auth-system 과 --x509-user-proxy 을 추가한다.

```
# nova --os-auth-system voms --x509-user-proxy /tmp/x509up_u1000 credentials
```

```
# nova --os-auth-url=$URL  --os-cacert /etc/grid-security/ca-bundle.crt  ₩
  --os-auth-system voms   --x509-user-proxy $X509_USER_PROXY ₩
  --os-tenant-name=$TENANT  list
+--------------------------------------+--------+--------+------------+-------------+----------------+
| ID                                   | Name   | Status | Task State | Power State | Networks       |
+--------------------------------------+--------+--------+------------+-------------+----------------+
| 7ad8ea88-12b8-4dc7-8490-e475708cfb5c | test1 | ACTIVE | -          | Running     | vmnet=10.1.2.2 |
```

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

(10) API 에서 활용하는 방법

```
# cat nova-client-test.py

import novaclient
import novaclient.auth_plugin
import novaclient.client as nvclient

url = "https://fccont.kisti.re.kr:35358/v2.0/"
username = password = None
tenant = "admin"
version = 2

auth_system = "voms"

novaclient.auth_plugin.discover_auth_systems()
auth_plugin = novaclient.auth_plugin.load_plugin(auth_system)
auth_plugin.opts["x509_user_proxy"] = "/tmp/x509up_u500"

novaclient =  nvclient.Client(version, username, password, tenant, url,
                           auth_plugin=auth_plugin, auth_system=auth_system)

print novaclient
```

```
>>> dir(novaclient)
['__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__getattribute__', '__ha
sh__', '__init__', '__module__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__s
etattr__', '__sizeof__', '__str__', '__subclasshook__', '__weakref__', 'agents', 'aggregate
s', 'authenticate', 'availability_zones', 'certs', 'client', 'cloudpipe', 'dns_domains', 'd
ns_entries', 'fixed_ips', 'flavor_access', 'flavors', 'floating_ip_pools', 'floating_ips',
'floating_ips_bulk', 'fping', 'get_timings', 'hosts', 'hypervisors', 'images', 'keypairs',
'limits', 'networks', 'os_cache', 'projectid', 'quota_classes', 'quotas', 'reset_timings',
'security_group_rules', 'security_groups', 'servers', 'services', 'set_management_url', 'te
nant_id', 'usage', 'virtual_interfaces', 'volume_snapshots', 'volume_types', 'volumes']
```

참고: OpenStack Python SDK (http://docs.openstack.org/user-guide/content/ch_sdk.html)

제4절 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. The formal specification is maintained and actively worked on by OGF's OCCI-WG, for details see [1].

[1] http://occi-wg.org

(1) 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 [2] 설치

[2] https://github.com/IFCA/occi-os

```
# git clone git://github.com/EGI-FCTF/occi-os  -b stable/havana
# cd occi-os/
# python setup.py install
...
Using /usr/lib/python2.6/site-packages
Finished processing dependencies for openstackocci-havana==1.0
```

OpenStack 설정

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

########
# OCCI #
########
[composite:occiapi]
use = egg:Paste#urlmap
/: occiapppipe
[pipeline:occiapppipe]
pipeline = authtoken keystonecontext occiapp
# with request body size limiting and rate limiting
# pipeline = sizelimit authtoken keystonecontext ratelimit occiapp
[app:occiapp]
use = egg:openstackocci-havana#occi_app
```

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

```
....
auth_protocol = https
auth_uri = http://fccont.kisti.re.kr:5000/v2.0
....
:wq
```

수정후 nova-api 서비스를 재시작

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

nova 설정 수정

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

enabled_apis=ec2,occiapi,osapi_compute
occiapi_listen_port=9999
```

openstack-nova-* 서비스를 재시작

```
# for i in $(cd /etc/init.d/ ; ls openstack-nova-*); do sudo service $i status; done
```

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
```

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_module      /usr/lib64/httpd/modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
#LoadModule proxy_http_module /usr/lib64/httpd/modules/mod_proxy_html.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>
```

OCCI 를 keystone 에 서비스 등록하기

```
# keystone service-create --name nova --type occi --description 'Nova OCCI Service' | tee a
# export ID=`grep id a | awk '{print $4}'`
# echo $ID
# export HOSTNAME=fccont.kisti.re.kr
# keystone endpoint-create ₩
  --service_id  $ID  ₩
  --region RegionOne --publicurl https://$HOSTNAME:8787/ ₩
  --internalurl https://$HOSTNAME:8787/ ₩
  --adminurl https://$HOSTNAME:8787/
```

(2) OCCI 클라이언트 설치

occi 설치방법은 [1]을 참고 한다.

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

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, defaults 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
    -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, applicable 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
.....
```

(3) 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
```

제5절 Accounting System

EGI resource provider는 어카운팅 서비스인 APEL을 설치하여 해당 사이트의 사용량 정보를 FedCloud accounting server로 전송하여 수집될 수있도록 해야 한다. 어카운팅 서비스에 대한 자세한 내용은 [1]을 참고한다.

현재 어카운팅 정보가 수집되고 있는 사이트에 대한 정보는 [2]를 참고한다. 2014년 현재 28개의 사이트의 어카운팅 정보가 수집되고 있다. 수집된 어카운팅 정보의 상세 내역은 [3]에서 확인할 수 있다. 최근 3시간까지 정보가 표시된다.

Sites publishing new cloud accounting records (SSM 2.0)

Page last updated: 2014-11-18 01:30:04.668772

Site	NumberOfMachines	CloudType	LastUpdated
100IT	9341	Openstack	2014-11-17 23:59:09
BIFI	7232	Openstack	2014-11-17 08:09:30
CERN-PROD	162589	OpenStack	2014-11-17 22:04:17
CESGA	21934	OpenNebula	2014-11-17 08:42:51
CESNET	44662	OpenNebula	2014-03-03 10:58:39
CESNET-MetaCloud	9070	OpenNebula	2014-11-18 00:57:20
CETA-GRID	2297	Openstack	2014-11-15 00:00:23
CYFRONET-CLOUD	1694	Openstack	2014-11-18 01:00:04
FZJ	13896	Openstack	2014-11-18 01:05:06
GoeGrid	23242	OpenNebula	2014-11-17 21:00:58
GRIF	47	StratusLab	2013-09-10 09:27:05
HG-09-Okeanos-Cloud	2672	Synnefo	2014-11-18 01:00:10
IFCA-LCG2	14269	OpenStack	2014-11-17 23:30:08
IISAS-Bratislava	46	Openstack	2013-04-09 12:23:19
IISAS-FedCloud	3083	Openstack	2014-11-18 01:11:07
IN2P3-CC	11	Openstack	2013-03-19 13:47:58
INFN-CATANIA-NEBULA	6566	OpenNebula	2014-11-17 14:42:55
INFN-CATANIA-STACK	6192	Openstack	2014-11-17 02:07:55
INFN-PADOVA-STACK	4192	Openstack	2014-11-16 23:30:05
KISTI	28	Openstack	2014-11-18 01:00:10
KTH-CLOUD	11735	OpenNebula	2014-10-24 20:55:33
MK-04-FINKICLOUD	2472	OpenNebula	2014-11-18 01:08:31
NCG-INGRID-PT	311	Openstack	2014-11-17 08:20:08
PRISMA-INFN-BARI	14341	Openstack	2014-11-17 17:00:18
SZTAKI	7242	OpenNebula	2014-11-17 02:35:23
TR-FC1-ULAKBIM	3487	Openstack	2014-11-12 15:23:32
TW-EMI-PPS	213	OpenStack	2013-09-12 14:11:42
UPV-GRyCAP	2124	OpenNebula	2014-11-17 01:39:41

그림 9 EGI FedCloud sites publishing new accounting records

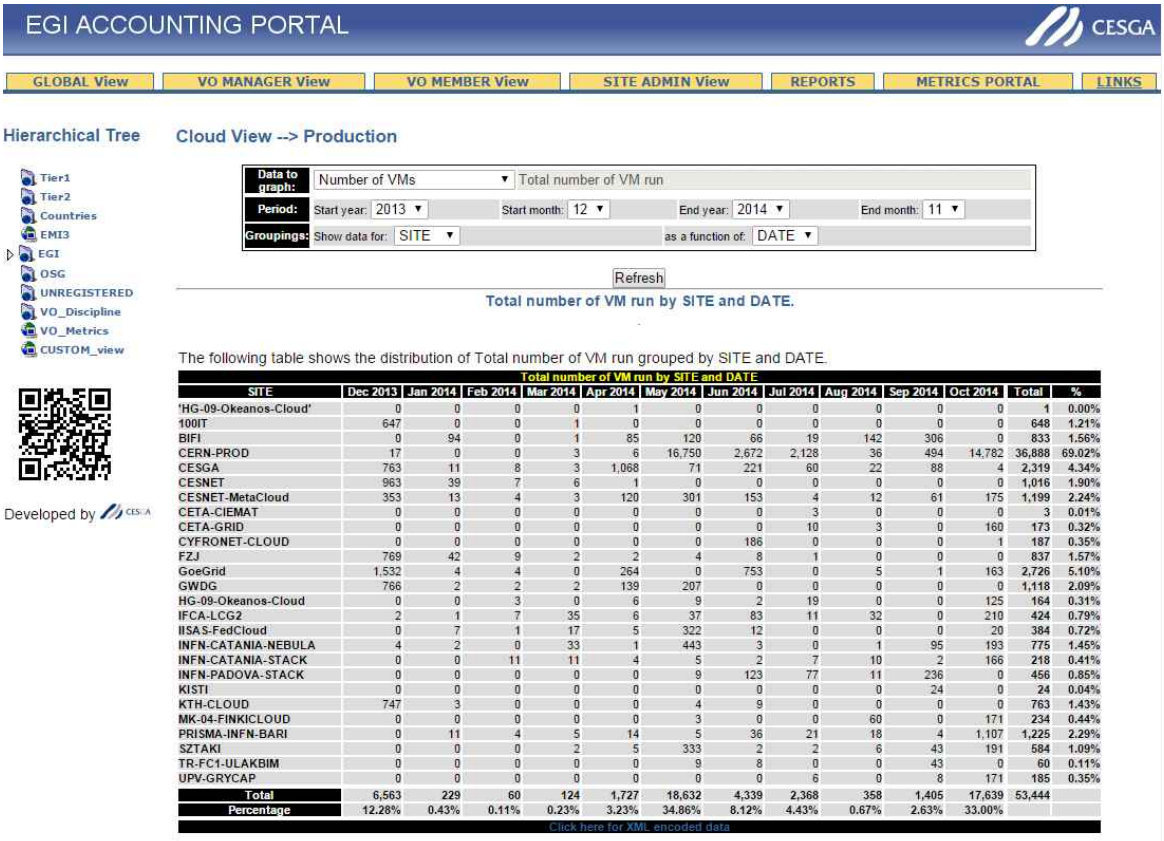
EGI Accounting Portal [4] 에서는 FedCloud 연동 사이트들의 어카운팅 정보 통계를 조회할 수 있다. VM의 개수, CPU시간, 총시간, 메모리 사용량, 네트워크 사용량, 디스크 사용량을 SITE와 VO와 날짜 별로 조회할 수 있다.

[1] https://wiki.egi.eu/wiki/Fedcloud-tf:WorkGroups:Scenario4

[2] http://goc-accounting.grid-support.ac.uk/cloudtest/cloudsites2.html

[3] http://goc-accounting.grid-support.ac.uk/cloudtest/vmshour2.html

[4] http://accounting-devel.egi.eu/cloud.php



Total number of VM run by SITE and DATE

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%		

Click here for XML encoded data

그림 10 EGI Accounting Portal (Cloud View)

(1) APEL SSM 설치

APEL SSM은 EMI [2] 소프트웨어중 하나로 컴퓨터간에 메시지를 전송하기 위한 소프트웨어로, python 으로 작성되어 있으며, STOMP 프로토콜[1]을 이용한다. SSM 은 메시지 전송과 수신을 다 지원하지만, EMI에서는 메시지 전송에 대한 부분만 사용된다. 이 문서는 SSM을 메시지 전송자로써 설정하는 방법을 설명한다.

APEL SSM은 현재 64-bit SL5 또는 SL6만을 지원하며 설치를 위해 EPEL repository가 활성화 되어야 한다. 호스트 인증서는 /etc/grid-security 경로에 설치되어 있어야 한다. 인증서는 SSL connection과 전송 메시지 서명에 사용된다.

Administrator guide 와 user guide 는 [3],[4]를 참고한다.

[1] http://en.wikipedia.org/wiki/Streaming_Text_Oriented_Messaging_Protocol

[2] http://www.eu-emi.eu/

[3] https://twiki.cern.ch/twiki/pub/EMI/EMI3APELClient/APEL_SSM_System_Administrator_Guide.pdf

[4] https://twiki.cern.ch/twiki/pub/EMI/EMI3APELClient/APEL_SSM_User_Guide.pdf

패키지 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 install -y apel-ssm
```

```
Downloading Packages:
(1/4): apel-ssm-2.1.3-1.el6.noarch.rpm                    |  20 kB     00:00
(2/4): python-daemon-1.5.2-1.el6.noarch.rpm               |  27 kB     00:00
(3/4): python-dirq-1.6.1-1.el6.noarch.rpm                 |  49 kB     00:00
(4/4): stomppy-3.1.6-1.el6.noarch.rpm                     |  48 kB     00:00
```

설정 파일 수정

broker 서버의 주소와 포트를 명시하며, 메시지는 encrypt 하지 않도록 해야 한다.

```
# vi /etc/apel/sender.cfg

################################################################################
# Required: broker configuration options
[broker]
# The SSM will query a BDII to find brokers available.  These details are for the
# EGI production broker network
#bdii: ldap://lcg-bdii.cern.ch:2170
#network: PROD
# OR (these details will only be used if the broker network settings aren't used)
# 위의 broker network 셋팅을 하지 않을 경우 호스트명을 직접 명시하여 준다.
host: mq.cro-ngi.hr
port: 6163

# broker authentication.  If use_ssl is set, the certificates configured
# in the mandatory [certificates] section will be used.
use_ssl: true


################################################################################
# Required: Certificate configuration

[certificates]
certificate: /etc/grid-security/hostcert.pem
key: /etc/grid-security/hostkey.pem
capath: /etc/grid-security/certificates
# If this is supplied, outgoing messages will be encrypted
# using this certificate
# server_cert 를 주면 메시지가 encrypt 된다. encrypt 하지 말것.
#server_cert: /etc/grid-security/servercert.pem
.......
```

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
```

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	

(2) OSSSM 설치

osssm [1] 은 모니터링 중인 tenants로 부터 어카운팅 사용량 정보를 뽑아내어 APEL/SSM 어카운팅 시스템으로 전달하는 역할을 한다. osssm은 Openstack의 Nova API를 이용하여 정보를 추출한다.

osssm은 다음과 같이 2가지 서비스로 구성된다.

- **osssm.extract** : 사용량 레코드 정보를 추출하여 캐시한다. 이는 cron 작업으로 주기적으로 실행된다.
- **osssm.push** : push 는 캐시된 레코드 정보를 APEL/SSM 으로 전송하는 역할을 한다. 스풀되어 있는 모든 정보를 전송하며, extract 보다는 실행 주기가 길어야 한다.

[1] https://github.com/EGI-FCTF/osssm/wiki

요구사항

- APEL/SSM 이 올바르게 설정되어 있어야 한다.
- 설정된 user/tenant 가 usage records 를 조회할 수 있는 권한이 있어야 한다.
- VO/tenant mapping 이 되어 있어야 한다. 이것은 보통 /etc/keystone/voms.json 에 정의된다.

설치방법 : 패키지 다운로드와 설치

```
$ wget ftp://ftp.in2p3.fr/ccin2p3/egi-acct-osdriver/apel-ssm-openstack/apel-ssm-openstack-latest.noarch.rpm
$ yum localinstall -y apel-ssm-openstack-current.noarch.rpm
```

설정하기

```
$ man osssmrc
# sed --in-place 's/###KEYSTONE_HOSTNAME###/fccont.kisti.re.kr/'  /etc/osssmrc
# sed --in-place 's/###PORT###/5000/'  /etc/osssmrc
# sed --in-place 's/###USER###/keystone/'  /etc/osssmrc
# sed --in-place 's/###PASSWORD###/keystone/'  /etc/osssmrc
# sed --in-place 's/###TENANT_NAME_LIST###/fedcloud/'  /etc/osssmrc
# sed --in-place 's/###SITE_NAME###/KISTI/'  /etc/osssmrc


# vim /etc/osssmrc
[Main]
keystone_api_url = http://fccont.kisti.re.kr:5000/v2.0
user = keystone
password = keystone
logfile_path = /var/log/apel/osssm.log
spooldir_path = /var/spool/osssm

# may be DEBUG|INFO
#debug_level = INFO
debug_level = DEBUG

# tenants names for which VM accounting should be extracted (comma separated list)
tenants = fedcloud

gocdb_sitename = KISTI

# cloud management platform
cloud_type = Openstack

# SSM related
ssm_input_header = APEL-cloud-message: v0.2
ssm_input_sep = %%
ssm_input_path = /opt/apel/ssm/messages/outgoing/openstack

# path to the VOMS/keystone auth configuration file
voms_tenants_mapping = /etc/keystone/voms.json
```

cron 스캐쥴링 설정하기

```
$ vim /var/lib/osssm/cron

# extract and buffer usage records
* * * * * apel /usr/bin/osssm.extract

# send buffered usage records to APEL/SSM
0 * * * * apel /usr/bin/osssm.push
```

서비스로 시작하기

```
$ chkconfig osssm on
$ service osssm start
```

현재까지 알려진 한계

- Does not report network rx/tx bandwith.
- Does not handle paused Vms.
- Does not implement "SuspendDuration".
- Does not report "VOGroup", nor "VORole" fields
- cron timings should be configuration parameters.
- the accounted disk space should be the addition of the image size and the ephemeral storage. It only accounts the ephemeral for now.

스풀된 어카운팅 메시지는 다음과 같이 기록됨

```
# cat /opt/apel/ssm/messages/outgoing/openstack/542036dc/542036fc511268
APEL-cloud-message: v0.2
VMUUID: 1778d8a5-899b-4635-bd54-0ea1a8da5653
SiteName: KISTI
MachineName: MyFirstVM
LocalUserId: fb5d70d90142464e94fae8013f762548
LocalGroupId: 46196f54ae8d4edbac6bb4a7d7871849
GlobalUserName: /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
FQAN: NULL
Status: NULL
StartTime: NULL
EndTime: NULL
SuspendDuration: NULL
WallDuration: NULL
CpuDuration: NULL
CpuCount: NULL
NetworkType: NULL
NetworkInbound: NULL
NetworkOutbound: NULL
Memory: NULL
Disk: NULL
StorageRecordId: NULL
ImageId: CirrOS 0.3.1
CloudType: Openstack
%%
VMUUID: dbae5127-9c18-4091-a622-db587e68ea89
SiteName: KISTI
MachineName: MyFirstVM
LocalUserId: fb5d70d90142464e94fae8013f762548
LocalGroupId: 46196f54ae8d4edbac6bb4a7d7871849
GlobalUserName: /C=KR/O=KISTI/O=GRID/O=KISTI/CN=80864141 Sangwan Kim
......
ImageId: CirrOS 0.3.1
CloudType: Openstack
%%
```

APEL SSM 메시지 전송 로그

```
# ssmsend
2014-09-22 23:57:36,043 - ssmsend - INFO - =======================================
```

```
2014-09-22 23:57:36,044 - ssmsend - INFO - Starting sending SSM version 2.1.2.
2014-09-22 23:57:36,044 - ssmsend - INFO - Retrieving broker details from ldap://lcg-bdii.cern.ch:21
70 ...
2014-09-22 23:57:37,635 - ssmsend - INFO - Found 2 brokers.
2014-09-22 23:57:37,674 - ssm.ssm2 - INFO - Messages will be encrypted using /etc/grid-security/host
cert.pem
2014-09-22 23:57:37,706 - ssm.crypto - INFO - Certificate verification: stdin: OK
2014-09-22 23:57:38,047 - stomp.py - INFO - Established connection to host mq.cro-ngi.hr, port 6163
2014-09-22 23:57:38,389 - ssm.ssm2 - INFO - Connected.
2014-09-22 23:57:38,389 - ssm.ssm2 - INFO - Will send messages to: /queue/global.accounting.test.clo
ud.central
2014-09-22 23:57:38,390 - ssm.ssm2 - INFO - Found 1 messages.
2014-09-22 23:57:38,390 - ssm.ssm2 - INFO - Sending message: 542036dc/542036fc511268
2014-09-22 23:57:38,427 - ssm.ssm2 - INFO - Waiting for broker to accept message.
2014-09-22 23:57:38,769 - ssm.ssm2 - INFO - Broker received message: 542036dc/542036fc511268
2014-09-22 23:57:38,828 - ssm.ssm2 - INFO - Tidying message directory.
2014-09-22 23:57:38,829 - ssmsend - INFO - SSM run has finished.
2014-09-22 23:57:38,829 - ssm.ssm2 - INFO - SSM connection ended.
2014-09-22 23:57:38,829 - ssmsend - INFO - SSM has shut down.
2014-09-22 23:57:38,830 - ssmsend - INFO - ========================================
```

제6절 Information System

BDII와 연동 작업은 Openstack 과 OpenNebula 가 모두 동일하다. 자세한 방법은 위키[1]을 참고한다. cloud-info-provider 패키지는 EGI's AppDB [2] 에서 찾을 수 있다. 본 절에서는 RHEL/CentOS/ScientificLinux 를 기준으로 하였다.

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

[2] https://appdb.egi.eu/store/software/cloud.info.provider

(1) 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
```

/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 GOCDB (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
    # 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 admin ₩
              --os-tenant-name fedcloud --os-auth-url http://fccont.kisti.re.kr:35357/v2.0
```

```
# cat /usr/bin/cloud-info-provider-service
#!/usr/bin/python
# PBR Generated from u'console_scripts'
import sys
from cloud_bdii.core import main

if __name__ == "__main__":
    sys.exit(main())
```

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 를 시작한다.

```
# service bdii start
```

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 [3]을 참고한다.

[3] https://wiki.egi.eu/wiki/MAN01_How_to_publish_Site_Information

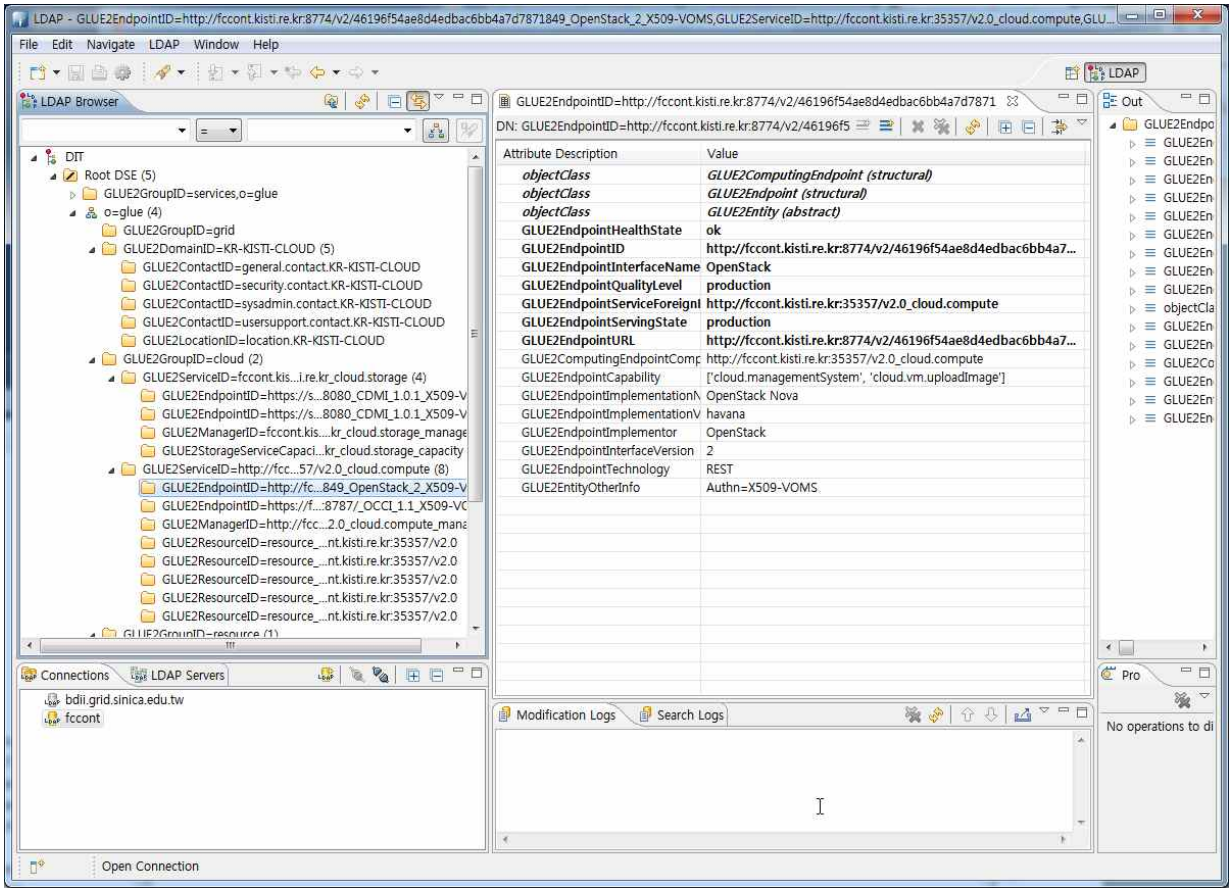


그림 11 Apache Directory Studio를 이용한 LDAP 브라우징

(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

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 체크함

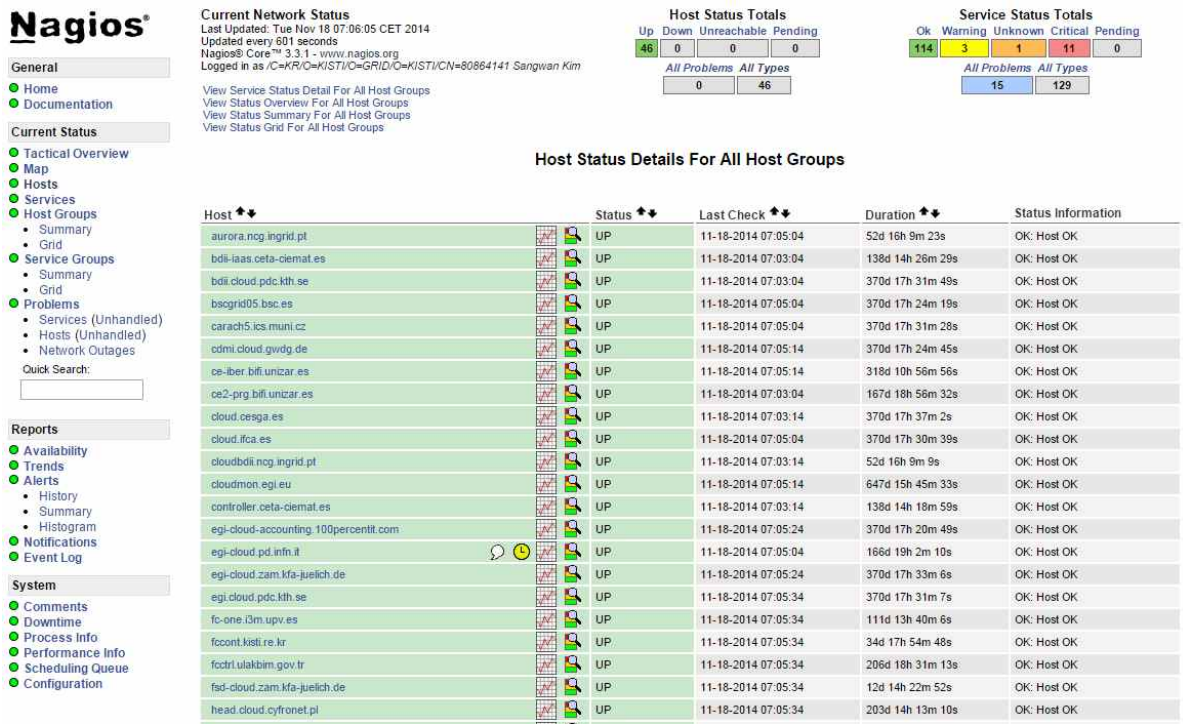


그림 12 FedCloud Nagios SAM 모니터링

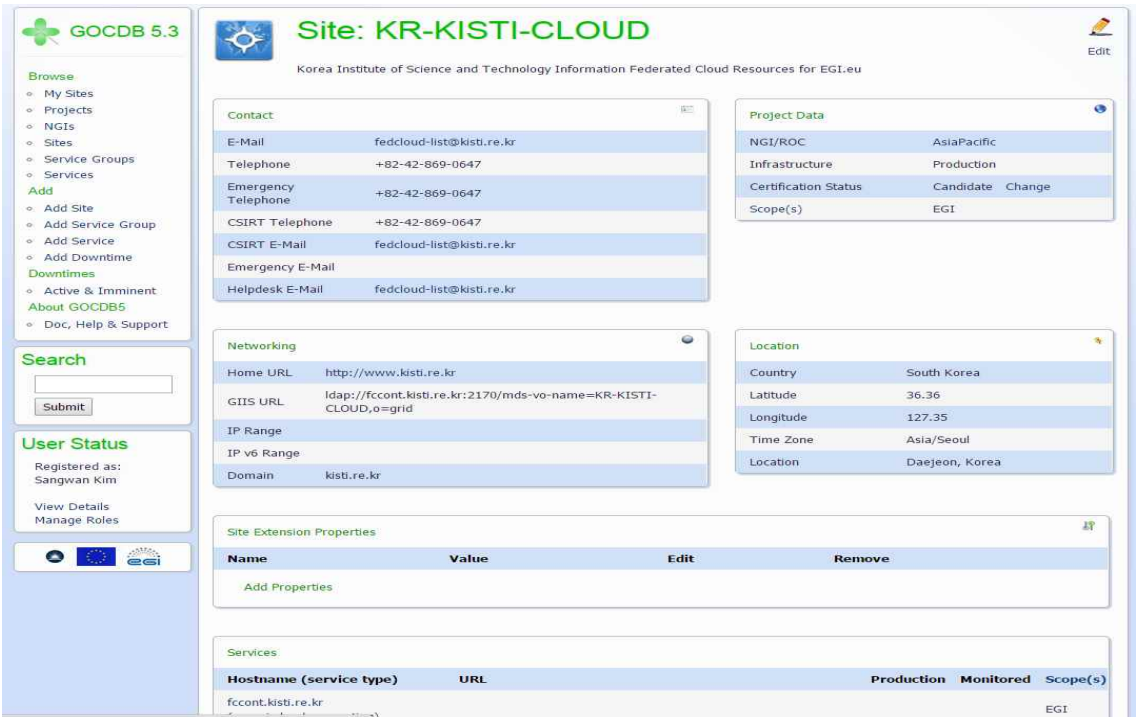


그림 13 GOC DB Site Infomation : KR-KISTI-CLOUD

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

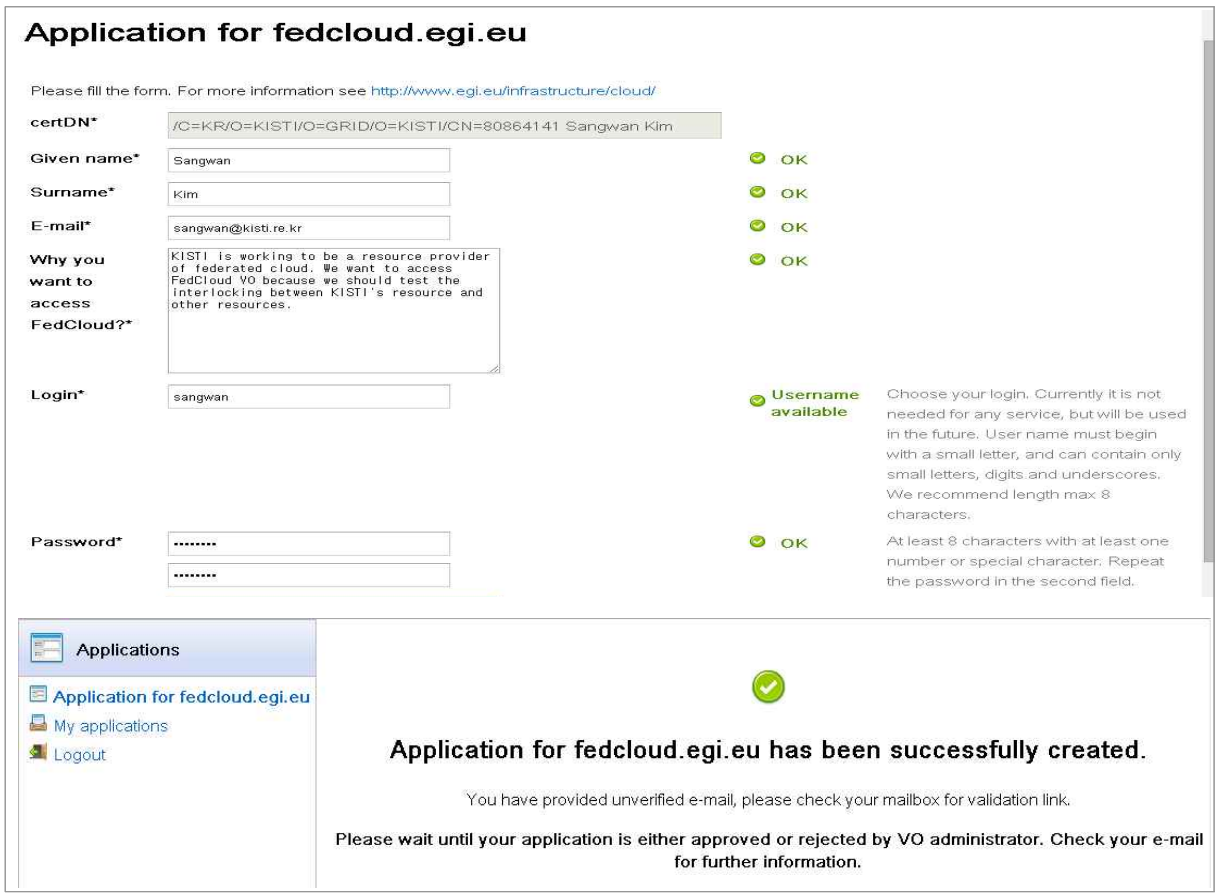


그림 14 FedCloud VO 가입화면

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.