

FermiCloud Phase II, III, and IV Project Plan

Steven Timm

Version 3.0
Last modified September 28, 2012

Abstract:

This document provides the program of work to move our pilot FermiCloud Infrastructure-as-a-Service facility to a fully automated and full-featured facility that can host production services as well as development and integration (Phase II) and then to a highly-available service (Phase III), and interoperable with future clouds (Phase IV)

Document Revision History:

Version	Date	Author	Comments
1.0	13-Sep-10	Steven Timm	First draft of Phase 2 project plan
2.0	07-Sep-11	Steven Timm	Phase II and III of project plan incorporating FermiCloud-HA
2.1	11-Oct-11	Steven Timm	Phase II and III of project plan incorporating FermiCloud-HA—continued revisions
2.2	2-Nov-11	Keith Chadwick	Edits on V2.1
2.3	8-Nov-11	Steven Timm	Add the Phase III plan and WBS
3.0	28-Sep-12	Steven Timm	Update Phase II and III, add beginning of Phase IV

Executive Summary:

The second phase of the FermiCloud project has designed and mostly deployed a scalable, reliable, and highly-available facility that is capable of hosting a wide array of scientific services, and that has virtual imaging provisioning and management technologies which can be leveraged across the rest of the Fermilab CD Scientific Facilities such as the GPCF. The goal of the third phase of the project, also known as FermiCloud-HA, is to distribute the cloud across buildings and make it resilient to facility failure. Activities to complete the second and third phase will be performed simultaneously while the second phase is underway.

The scope of the FermiCloud Project as a whole is to build a scientific infrastructure-as-a-service private cloud at Fermilab. A solution that is compatible with existing commercial clouds would offer significant advantages.

The project does not include virtualization of the classic IT and business functions of the lab, which is being implemented by the Virtual Services Group in the Network and Virtual Services Department in the Core Computing Division. This project will include research to support future virtualized “worker nodes” and submission of virtual machines as jobs, but does not include the production virtualization of the worker nodes of FermiGrid or production submission of virtual machines as jobs to FermiGrid

Current Status:

From the fall of 2010 until the present time FermiCloud has been running OpenNebula 2.0. This has proven to be a functional and stable product that has met the needs of a wide variety of user groups. We have made 14 of our 23 production machines available to this production cloud, with the balance being used in fabric and storage testing and in preparation for the next major release of OpenNebula. The table below summarizes the major projects that have run within the last year or are still running on FermiCloud.

Application/Project	Stakeholder	Status
JDEM Demo Data Processing System	LSST	
OSG Software Team build and test	OSG	
GEANT4 validation server	CET	
GlobusOnline test work (CEDPS)	OSG & CEDPS	
CVMFS testing	IF	
Test GUMS and SAZ servers	FermiGrid	
Extenci Lustre-over-WAN	OSG	
Vcluster demo (grid cluster in the cloud)	GCC & KISTI	

Expt-specific gridftp servers (MINOS, NOVA)	IF	
SAMGrid forwarding nodes	D0	
dCache NFS4.1 testing	REX & DMS	
Scientist survey	Directorate	
MCAS	GCC	
OSG-XSEDE interoperability	OSG, XSEDE	

Phase II Introduction:

For each major category of the FermiCloud requirements, we summarize the program of work that is necessary to improve or extend the FermiCloud facility to meet those requirements.

Operating System and Virtualization

We have demonstrated the use of Scientific Linux 5 and 6 both as virtual machine host and as virtual machine client. As anticipated, Scientific Linux 6 brings significant improvement to the I/O throughput of the KVM hypervisor as well as requiring significantly less CPU to perform I/O. We have installed it on half of our VM hosts to date and once the virtual machines are migrated to these hosts, we will upgrade the rest of these hosts. It also has significantly more features available to bind virtual machines to processors and memory banks with functionality similar to the `numa_ctl`.

Some scientific developers who support products on Cygwin as well as Linux have asked for Windows virtual machines. We have also had requests for other Linux distributions, in particular Debian and Ubuntu. There is also significant interest in Scientific Linux systems of previous versions with frozen patch levels. The Windows test has not yet been done due to fairly low relative priority to other activities but we expect to do it within the course of this project year. We do not anticipate any technical impediments to doing it since Windows support is an advertised feature of OpenNebula. The security discussion group has made a preliminary report on these alternative operating systems which is pending comments and actions by the CSBoard.

We are in the process of documenting the method to move virtual machines from desktop-based systems such as VMWare, Parallels, and VirtualBox, to the cloud and back.

Provisioning and Contextualization

We will provide the following:

- A mechanism such that a virtual machine can boot a Scientific Linux installation image and install a clean virtual machine via kickstart, either a supplied kickstart file or an auto-generated kickstart from Rocks or Cobbler. This is complete.

- A mechanism such that the configuration of the machine can be managed by common configuration management systems currently in use at Fermilab, such as Puppet. Any node-specific configuration files can be supplied either by one of these configuration systems or by the contextualization mechanisms of OpenNebula which allow the user to combine extra files into a CD ISO image and mount it at boot time. This work is still in progress.

In addition, the existing mechanisms to securely store and retrieve secrets such as Kerberos keytabs and X.509 host/service credentials during VM “booting” (contextualization) have been enhanced.

Object Store

The current OpenNebula image repository is implemented on the local disk of the head node. Operating system images are copied to the virtual machine host via scp at the launch of the virtual machine, and if saving the image is requested, copied back to the head node via scp. The persistent data blocks are also copied that way. For the currently observed application pattern of a large number of users each running one or two virtual machines, this is a reasonable way to operate. However, users do complain that virtual machines, particularly those with persistent data blocks, take a long time to launch. We continue using this method with the initial deployment of OpenNebula 3.2 which is now in progress.

Once the SAN is completed, we will shift to the shared file system model of OpenNebula, in which the virtual machine OS images (at least the 24x7 persistent ones) live on the SAN the whole time. Our baseline design is for a shared file system with GFS2 on top of Clustered LVM, provided by Red Hat Clustering. Clustered LVM provides a capacity to extend a future mirrored file system between our two buildings.

By contrast, the LVM launching method is built to launch a large number of the same virtual machine in a very short amount of time. This is code that was developed at CERN and contributed back to OpenNebula, and they used it to launch 4000 virtual machines in a matter of minutes. Using the copy-on-write feature of LVM snapshots, only those blocks that change from the initial snapshot need to be copied to the new area. There have been several packages such as SCPWave or LanTorrent that are used to keep copies of the OS image synced to all the various VM hosts. We have a number of tests internal to the Grid and Cloud Computing department in which the bulk launch of 30-40 simultaneous virtual machines is necessary. This launch mode will be revisited once OpenNebula 3.6 is deployed and we are doing large interoperability tests that require fast launch of many worker node VM's.

Interoperability

FermiCloud staff has been participating in the HEPiX virtualization taskforce which has been outlining procedures for exchanging virtual machines between sites. They have draft procedures for signing images, endorsing images, and making image catalogues. OpenNebula has a feature to “cloudburst” to Amazon EC2. We need to test that.

OpenNebula 3.2 has an X.509 authentication mechanism that was developed by Fermilab staff and contributed back to OpenNebula. This X.509 authentication mechanism is pluggable and has been applied to the OpenNebula command line, the SunStone Web GUI by which users can launch and monitor virtualized machines from the web, the EC2 Query API which is a limited emulation of the Amazon EC2 Query API, and the Open Cloud Computing Interface. Todd Miller of the Condor team has modified the Condor-G Amazon EC2 client to be able to submit to the EC2 interface. OpenNebula

also has a SOAP API as part of their long-term road map but it is not so critical to have this now that the full support for X.509 authentication is available in the Query API.

The next phase of the Interoperability plan is making X.509-based authorization. We have a proof of principle to call the standard C bindings to the XACML-based interoperability profile that can talk to SAZ and GUMS. The extensions to the Authorization Interoperability profile that would be needed for cloud authorization have already been proposed to the working groups. We expect to deploy these callouts in production early in FY2013.

Functionality

OpenNebula 3.2 meets most of the functionality requirements out of the box. A significant effort in phase II is to do idle virtual machine detection and backfill with worker node virtual machines. Work from an INFN summer student has given us a comprehensive set of metrics that can be used to measure the idleness of the VM. Our next work is to make a collector to collect this information and decide in a central way which machines should be suspended in favor of worker node virtual machines. We have a full plan to make the OpenNebula head nodes highly available, this includes heartbeat and a shared multi-master mysql backend. One of our KISTI visitors developed “vCluster” which can look at a condor queue and launch virtual machines based on the type of job that is submitted. The functionality requirements also specify fair-share scheduling in the case of a resource-limited cloud. This is available through an alternative pluggable scheduler for OpenNebula.

Accounting and Billing

The Gratia project has written accounting probes for OpenNebula 2.0, 3.0, and 3.2, and they are deployed on our clouds and we are collecting accounting information. The Usage Record is similar to a job usage record for the grid, and we continue to work with the various standards bodies to compare to what others are doing with cloud accounting. A summer student, Siyuan Ma, modified the gratiaweb package of graphing so that it could show some of our cloud-specific graphs, and this software has been deployed in production.

Network Topology

The basic network topology configurations (public fixed IP, public variable IP, public-private cluster, and private-only) are already in production in OpenNebula. In phase II we will investigate IPv6 operations. We will continue work begun by KISTI visitors on inter-machine communication via Infiniband. Long-promised KVM virtualization drivers for Infiniband became available within the last year and we were able to show computing performance that was equal to 96% of a bare metal machine.

We have begun investigations on subjecting new virtual machines to the site virus scanner and node registration mechanisms and granting full network access only after that procedure is complete. New OpenNebula features which restrict the way in which images can be uploaded, and by whom, also help with this.

Clustered File System

Extensive performance testing of Lustre, Hadoop, dCache, OrangeFS, BestMan and BlueArc filesystems were performed using FermiCloud resources as part of the storage evaluation project. The next phase of clustered file systems concentrated on shared file systems for the internal use of

FermiCloud itself, file systems that can be used to serve a shared repository of OS Images and persistent datablocks via the SAN to all the worker nodes. GFS2 appears to be the most straightforward to deploy and to have satisfactory performance.

Security Policy

At the request of the Computer Security Board, we convened a group of security policy experts, similar to what was done to form the Open Science Enclave at the beginning of FermiGrid. Internal discussions among the FermiCloud team to date have identified unique security issues which do not neatly fit into the General Computing or Open Science environments. We made a report on these issues together with proposed solutions and have presented it to the CS Board.

In addition to making the policy, we have significant effort required to leverage the site “network jail” and to develop the system that wakes up dormant virtual machines for their patches, as well as whatever new requirements the security policy group may establish.

Additional FermiCloud Phase II Activities this year beyond 2010 Requirements statement:

Management: Establishing a Service Level Agreement, and interface with user groups and potential stakeholders.

User Support and Outreach: Writing and giving training courses, getting request forms up in ServiceNow.

Technology Investigation: Evaluate OpenStack. (Forthcoming incorporation into Red Hat Linux is a potential game-changer).

Monitoring: Leverage existing open-source graphics packages such as Ganglia and Nagios to plot as many relevant monitoring and metrics quantities as possible.

Phase II WBS:

The following is a summary of the major tasks in Phase II:

1.	OS and Virtualization
1.1	Test launching Windows virtual machines
1.2	Virtual machine migration from desktop to cloud and back
2.	Provisioning and Contextualization
2.1	Virtual machine boot Sci. Linux install image and install via kickstart *completed
2.2	Generate and use auto-kickstart file *completed
2.3	Manage node-specific configuration via Puppet *in progress
2.4	Enhance secure secret store, make accessible from private net *in progress
3.	Object stores
3.1	Test shared file system launches of virtual machines *completed
3.2	Test LVM copy-on-write launches of virtual machines
3.3	Test shared file systems on SAN, especially CEPH FS *completed

3.4	Test live migration of virtual machines
4.	Interoperability
4.1	Test HEPiX VM store, image catalog, image exchange
4.2	Test cloudburst function to Amazon EC2
4.3	Test Condor-G EC2 universe with X.509 OpenNebula EC2 query API
4.4	X.509 Authorization / Ruby XACML callout *working prototype
5.	Functionality
5.1	Idle machine detection plus backfill with worker node virtual machines *in progress
5.2	Virtual machine launch based on user requests in queue
5.3	Fair-share scheduling in case of resource-limited cloud.
6.	Accounting and Billing
6.1	Final Requirements definition for cloud accounting. *completed
6.2	Definition of cloud usage record. *completed
6.3	Population of cloud usage record based on available OpenNebula info. *completed
6.4	Modification of OpenNebula to generate other needed information. *not necessary
6.5	Integration of data with Gratia accounting system. *completed
7.	Network and fabric investigation
7.1	IPv6 investigations (in collaboration with Network Research).
7.2	Virtualization drivers for Infiniband cards *completed
7.3	New virtual machines subjected to virus scanner / node registration *in progress
8.	Clustered file systems
8.1	CEPH FS test, OCFS2, GFS *completed
8.2	NFS v4.1 *in progress
9.	Security
9.1	Cloud Security discussion group *prelim report done
9.2	Get cloud vm's on restricted 10. Subnet to start *in progress
9.3	System to wake up dormant virtual machines for patches.
10.	Management
10.1	Approved Service Level Agreement
11.	User Support and Outreach
11.1	Writing and giving training courses

11.2	Get request forms up in ServiceNow *completed
11.3	Get and keep documentation up to date
11.4	Communications to Fermilab and outside the lab.
12.	Technology Investigation
12.1	Evaluate OpenStack
13.	Monitoring

Phase III Project Plan—FermiCloud High Availability:

A large number of power and cooling outages in the Grid Computing Center over the summer of 2011, and of network and power outages in the Feynman Computing Center in 2010, underscored the necessity to have a facility which is resilient to the failure of one of its buildings. The FermiCloud-HA plan was formed as a result. There are four major components to this plan:

Network:

We need a subnet or subnets that is available in multiple facilities and that has access to a router and the outside network even if one of the facilities suffers a power failure. Details are found in the document FermiGrid and FermiCloud Network Requirements, CD-Docdb 4426. FermiCloud is expected to be one of the first subnets which will use an implementation which does not require dedicated fiber between buildings. Network Services has implemented Cisco FabricPath for layer 2 connectivity and Gateway Load Balancing Protocol for layer 3. We have a budget request for an Infiniband cross-building interconnect as well this year.

SAN and Shared File System:

To be able to live-migrate virtual machines between buildings, we need a Storage Area Network that extends between buildings. Our reference design is a pair of two Nexsan Satabeast disk servers, one in each building, which have a file system which is mirrored between them. We also need a shared file system on that SAN which can be mounted read/write by a number of machines simultaneously. Both SATABeast and switching hardware are deployed and connected to each other across buildings. We continue work on the mirrored file system.

High Availability OpenNebula Daemons:

High availability of OpenNebula daemons can be addressed by a number of the same techniques that have been used in FermiGrid HA. We have deployed an active/passive setup with a single active head node using heartbeat for the service IP as is done for MyProxy, a shared software base on NFS, and a common replicated MySQL database and a common service IP address by which they are accessed.

We have also deployed a number of virtual machines to manage services such as Nagios, Ganglia, the secrets repository, puppet, LVS, mysql, and syslog. These are either redundant or will be live-migratable. For the moment, the OpenNebula head node itself needs to run on bare metal to be part of the GVS2/CLVM clustered file system cluster. Our goal is to present a unified cloud service in which any cloud middleware, hardware, or operating system can be upgraded while still keeping the service available, and which is resilient against a building failure in any one building.

Migration and Relaunching of Virtual Machines:

The capacity to migrate the OpenNebula master daemons from one building to another, or have them live in both buildings simultaneously, is not much good if the virtual machines they managed are trapped in the building that is going down. In the case of a preplanned building outage, it would be possible to migrate the virtual machines that were in that facility ahead of time. In the case of an unplanned building outage, we would detect the high availability virtual machines that had been running in that facility, and relaunch them in the other facility from the OS image on the mirrored SAN. In both cases we would need software to be able to push opportunistic-grade virtual machines out of the way to make room for the high-priority virtual machines.

Phase III WBS

1.	Networking
1.1	Test of multi-building, redundantly routed FermiCloud subnet *complete
1.2	Production deployment of multi-building, redundantly routed FermiCloud subnet *complete
2.	SAN and shared file system
2.1	Installation of FCC3 SAN *complete
2.2	Ordering and installation of GCCB SAN *complete
2.3	Identification and installation of shared file system for single-building SAN *in progress
2.4	Investigation of methods to deploy cross-building mirroring *in progress
2.5	Deployment of cross-building mirrored SAN
3.	High Availability of OpenNebula Daemons
3.1	Active-passive opennebula daemon failover with service ip + heartbeat + DRBD *complete
3.2	Discussions with OpenNebula developers on location of state in OpenNebula processes *complete
3.3	Deployment of active-active OpenNebula daemons with shared replicated mysql databases *in progress
4.	High Availability of Virtual Machines
4.1	Automated migration of virtual machines between buildings before outage
4.2	Detection and relaunching of virtual machines taken out by outage of building or hardware.

Phase IV: Interoperability

Interoperability has always been part of our project plan and is included in some of the deliverables for phase II, but in the new phase we are targeting it specifically. In particular we need to study the issue of VM image format so that we can make the process of running on a number of different cloud providers as seamless as possible for the users. These explicitly include Nimbus-based clouds at FutureGrid and Purdue, the CERN Agile Infrastructure, Amazon EC2, and OpenStack-based clouds. We also need to study interoperability between cloud API's of these different providers, and of the various X.509 authentication and authorization schemes that are emerging. Finally we need to implement in production a look-ahead cloud manager that can launch appropriate styles of grid worker

node virtual machines based on the job load, and to implement the procedure to accept cloud VM's as batch jobs.

Original FermiCloud Deliverables as given in March 2010 cloud plan:

1. Test Cloud Cluster up for investigations: Completed June 2010 (Phase I)
2. Design plan for production FermiCloud cluster, including finalized hardware and software technical requirements and Statement of Work: Completed Sep. 2010 (Phase I)
3. Available Technology Investigation report complete: Completed Dec. 2010 (Phase I)
4. Requirements collection from potential VM users complete: Completed Sep 2010 (Phase I)
5. Production Cloud VM provisioning, OS Provisioning, and security patching solution in production: (Phase II)
6. Generalized VM library for users of pre-built virtual machines: Completed Feb 2011 (Phase II).
7. Policy draft of cloud computing at FNAL (Phase II).
8. Transfer of all DOCS Group development and integration work and DMS storage development work to FermiCloud (Phase II)
9. Integration of FermiCloud technology into other facilities such as General Physics Computing Facility (to be determined)
10. Documentation and training on how to use the cloud (Phase II).

Additional Phase II Deliverables

Additional Phase III Deliverables