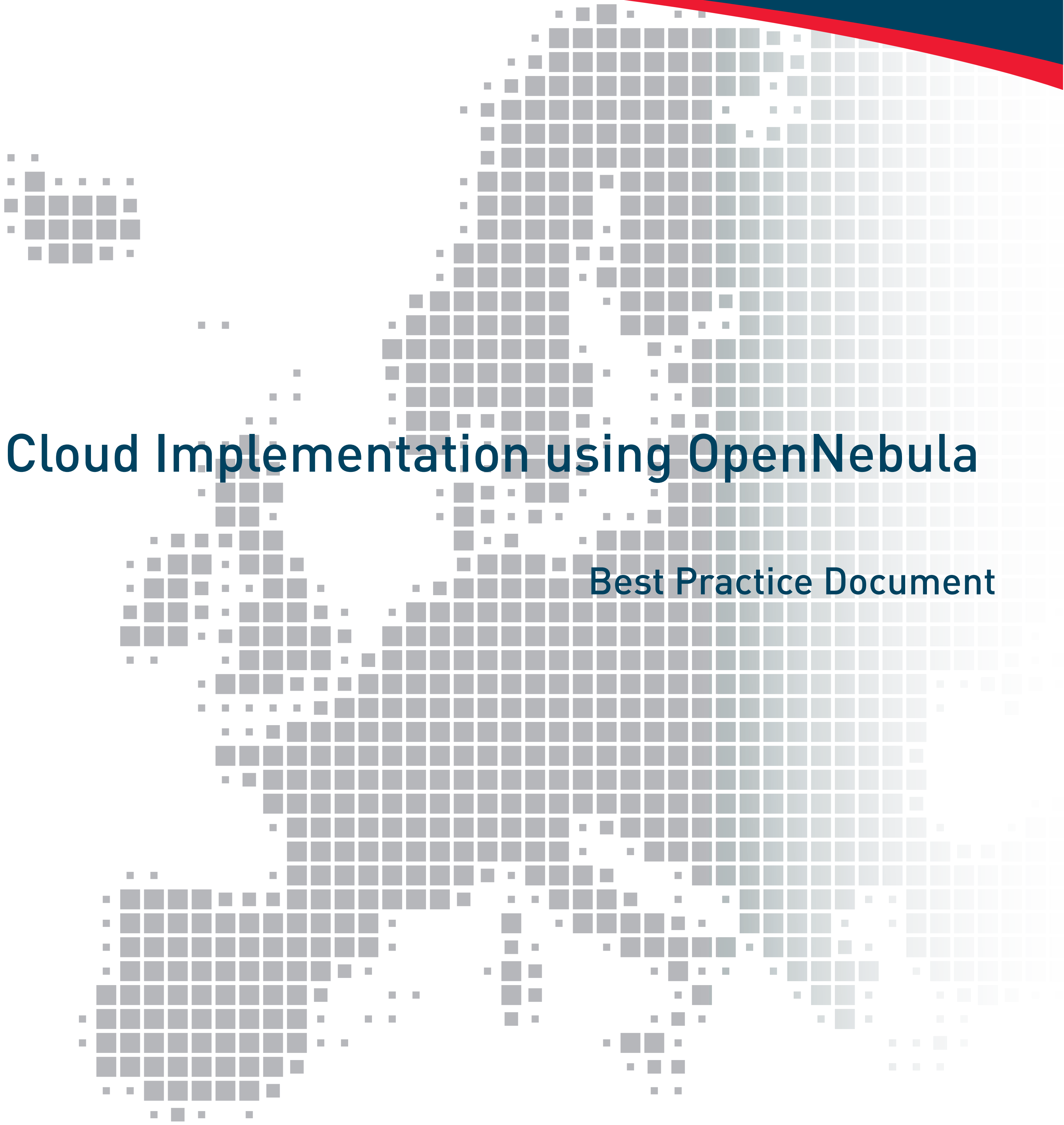

Cloud Implementation using OpenNebula

Best Practice Document

Produced by the MARnet-led working group
on campus networking

Authors: Vasko Sazdovski (FCSE/MARnet), Boro
Jakimovski (FCSE/MARnet)

April 2016

Document No:	GN4-Phase1-MA-BPD-4
Version / date:	V1.0, 30 November 2015
Original language:	Macedonian
Original title:	“Поставување на приватен облак со користење на OpenNebula”
Original version / date:	V0.1, 1 December 2014
Contact:	vasko.sazdovski@finki.ukim.mk

The work has been carried out by a MARnet led working group on campus networking as part of a joint-venture project within the HE sector in Macedonia.

The research leading to these results has received funding from the European Union’s Horizon 2020 research and innovation programme under Grant Agreement No. 691567 (GN4-1).

Table of Contents

Table of Figures

Executive Summary

Cloud computing is one of the hot topics that we hear about every day, and is becoming more and more popular. In this best-practice document, we discuss how to create a private cloud with an open solution and technology.

Chapter 1 introduces OpenNebula, a cloud management solution for applications in the HE sector, and lists the main features relevant to the context.

Chapter 2 describes the proposed architecture to use in an HE institution, going into detail on each part of the architecture and the role it plays.

- Chapter 2.1 describes the front node in the cloud, how it is connected to the other parts of the cloud and how to install it.
- Chapter 2.2 describes the server nodes, their role and how they are connected to the other parts of the cloud implementation.

- Chapter 2.3 describes the storage and storage servers and their role in the cloud. A description is given on the specifics of how to create a parallel and high speed shared file system providing the ability to move virtual machines from one server node to another on-line (without the need to take the virtual machine off-line).

Chapter 3 is a conclusion.

1 Introduction

OpenNebula provides a simple and flexible solution for the management of a virtualized data centre. Both *private cloud* and *hybrid cloud* scenarios are supported – from a virtualization tool to manage the local virtual infrastructure, to a combined local infrastructure with a public cloud-based infrastructure. OpenNebula supports public clouds by providing cloud interfaces to expose its functionality. [1]

We recommend using OpenNebula because it is free and simple. It is an alternative to OpenStack, which we consider a more complex solution and harder to setup, and an alternative to VMware which is too expensive and inflexible.

There are a few other reasons why we use OpenNebula:

- It is easy to download, install and update through packages
- OpenNebula is fully open-sourced and distributed under an Apache license
- Enables the easier management of virtualized data centres for both private and hybrid clouds
- It is platform independent and supports different hypervisors, storage and networking resources
- Open and extensible architecture with the possibility to build customized cloud services and allowing a choice of standards and cloud interfaces
- It has a large and involved community and has been tested within the community thru many release cycles, in both simple and massive scalable production environments

OpenNebula is even suitable for deployment in smaller HE computing centres, where the need to optimize the resources and management is crucial. Using OpenNebula they can offer flexible services to departments, research labs/groups, and even individual researchers. They can even be free to supporting teaching staff with their requirements for special services for supporting student project work and hosting their demonstrations and course showcases.

2 Proposed Cloud Architecture

The proposed architecture is intended for a private cloud for an HE institution. A diagram of the architecture is presented in Figure 1. It contains several components that together form the cloud. The main blocks, visible in figure 1, are the servers, storage and network model. We shall describe the architecture starting from the top and working towards the bottom of the diagram.

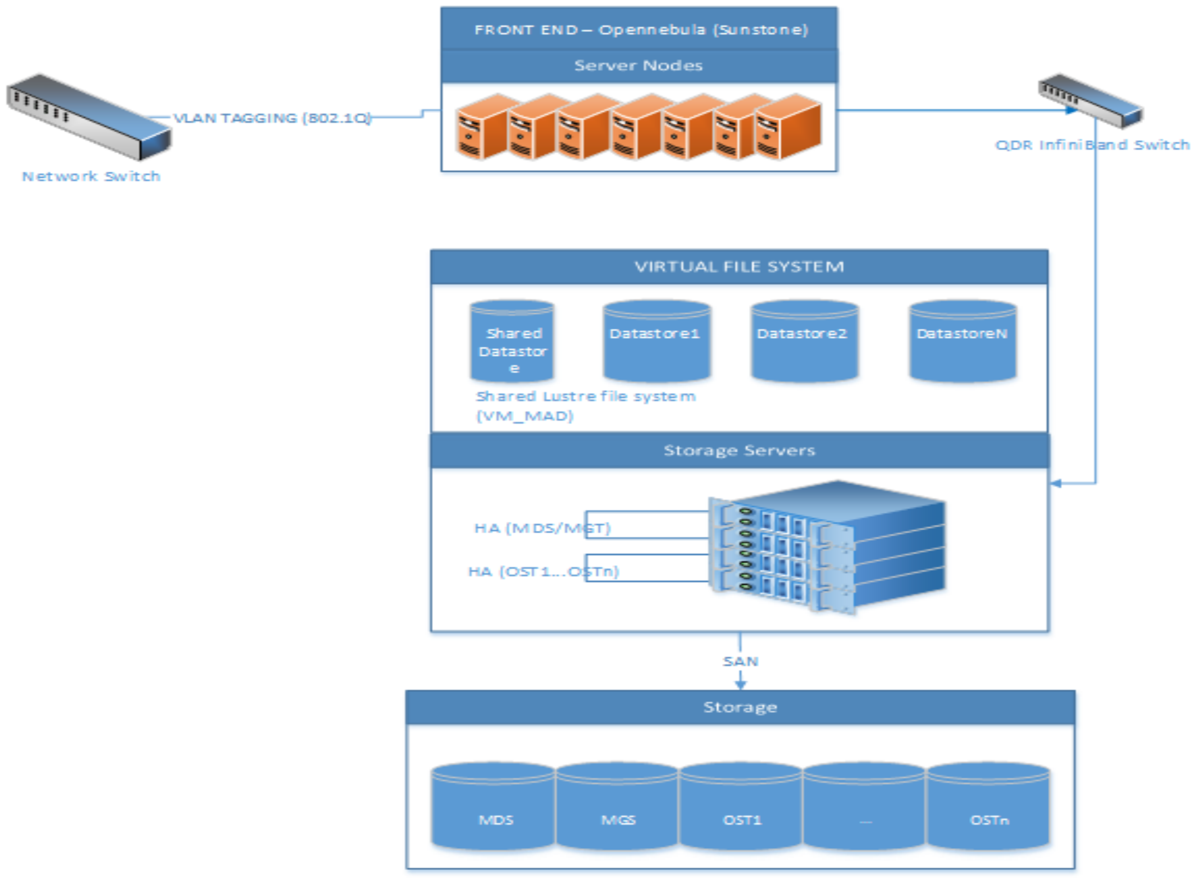

Figure 2.1: OpenNebula deployment architecture at FCSE

The server nodes are distributed in several locations, starting from the Front end, where the user interface is installed and server nodes (working nodes) are set-up hosting the virtual machines. Having a separate storage solution or a SAN in place with a high-speed, highly-available and redundant network connection is recommended, so one can provide fast and secure storage for the server nodes.

As an example, at the time of writing, the Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University in Skopje (FCSE), uses *blade servers* as server nodes and 802.1q is used in order to distribute the traffic from many VLANs to the respective server nodes. In terms of storage, Infiniband interconnect is used so that all the server nodes have a high-speed connection to their assigned storage.

2.1 Front End

The front end or the cloud operation centre is where all the users create their VMs and all the administration of the cloud is performed. For this purpose, Sunstone nebula is used. This interface makes it easy to add or delete new server nodes. One can create users, images and templates.

Within this operation centre, one also creates the networks for the cloud and it is the place from where the cloud is put into production operation. The operation centre interface is installed on one server – called a *front node*. It can be installed on almost all Linux distributions, but we recommend using CentOS (6.5 or more recent versions). The operation centre can be part of the cloud or can be hosted somewhere else. In order to be able to control the cloud nodes (server nodes), passwordless SSH access has to be set-up on all of the nodes. This will then be used by the front node to operate the server nodes.

The management of the cloud can be also done via the command-line as the user *oneadmin*. The accessible options via command-line are the same as when using the GUI.

To install the front node in a CentOS system, one can either set-up and use the official OpenNebula repository with the yum package manager, or one can manually download and install the RPM packages. It is recommended to use the automated updates from the repository in order to ease the maintenance.

The overall set-up involves several steps, described in detail.

Step 1. Activate the EPEL Repository

In CentOS, this can be done with the following command:

```
# yum install epel-release
```

Step 2. Import the OpenNebula Repository

Create a new *OpenNebula.repo* file within the *yum.repos.d* directory.

```
cat << EOT > /etc/yum.repos.d/opennebula.repo
[opennebula]
name=opennebula
baseurl=http://downloads.opennebula.org/repo/4.10/CentOS/6/
x86_64
enabled=1
gpgcheck=0
EOT
```

Then execute the following commands to install the required packages.

```
# yum install opennebula-server \
              opennebula-sunstone \
              opennebula-ruby
```

As mentioned above, the user *oneadmin* has to be configured, so that it will be used to control all the necessary functions. In order to be able to do this, passwordless SSH access to all the nodes has to be configured.

Step 3. Starting OpenNebula

The procedure to start OpenNebula starts with a log in as the *oneadmin* user.

If OpenNebula was installed from packages, the *one/one_auth* file will be created and a randomly-generated password will be placed. Otherwise, the *oneadmin*'s OpenNebula credentials (username and password) should be set by adding the following to *~/.one/one_auth* (change password to the desired password):

```
$ mkdir ~/.one

$ echo "oneadmin:password" > ~/.one/one_auth

$ chmod 600 ~/.one/one_auth

$ su - oneadmin

$ one start
```

These commands will start the OpenNebula front node.

There may be some additional configurations that should be done to fit the requirements of the specific infrastructure.

NOTICE: It is not recommended to start virtual machines within the front end node itself, because this node is only intended to be used for management, not as a node where one can host VMs.

2.2 Server Nodes

Server nodes or working nodes are servers where the virtual machines are deployed and run. The practice at FCSE is that HP Blade Servers are used for server nodes and they are all connected on switches with 802.1Q support. In this way all the available VLANs are offered between all server nodes and live migration is enabled. When migrating a VM from one server node to another, it can be done without delay and downtime. This is also useful when a certain VM is present in some VLAN, but cannot be on one of the server nodes.

If one needs live migration of a VM from one server node to another, having proper VLAN support in the infrastructure is necessary, but a mounted shared file-system is also required so that the source and target node have access to the files.

Due to the above-mentioned requirements, the referent set-up at FCSE employs the Lustre shared file-system, which is mounted on all the server nodes. Then, all the servers are connected to Infiniband QDR switches to enable high-speed access to the storage.

Step 4. Installation of Software for all Server Nodes

The installation process for the server nodes is almost identical to the one for the front node, the only difference being that the front-end interface is not needed (and thus not installed) on the server nodes. Again, one has to import all the necessary repositories and then install the packages *opennebula-node-kvm* and *opennebula-common.*

```
# yum install opennebula-node-kvm

# yum install opennebula-common
```

NOTICE: The *oneadmin* user has to be manually created at the front node so that the front node will be able to execute remote commands on the server nodes.

For this, the user id (uid) and group id (gid) of *oneadmin* are needed. These ids will be used later, when creating users within the hosts with the same ids. In the front-end, execute the following command as *oneadmin*:

```
$ id oneadmin

uid=1001(oneadmin) gid=1001(oneadmin) groups=1001(oneadmin)
```

In this case, the *uid* is 1001 and *gid* is (also) 1001.

The following steps are then performed while logged in as a root user.

Step 5: Create the *oneadmin Group*

Make sure the gid is the same as the one at the front-end. In this example it was 1001:

```
# groupadd --gid 1001 oneadmin
```

Create the *oneadmin* account, which will use the OpenNebula */var* directory as the home directory for this user.

```
# useradd --uid 1001 -g oneadmin -d /var/lib/one oneadmin
```

One will also have to manually configure SSH access. To do this, ssh keys need to be created for the *oneadmin* user and the host machines will need to be configured to enable passwordless access via SSH, using the created keys.

Step 6: Follow These Steps in the Front-End

Generate *oneadmin* ssh key:

```
$ ssh-keygen
```

When prompted for password, one can press enter to prevent having the private key encrypted.

The generated public key should be appended to *~/.ssh/authorized_keys* to let the *oneadmin* user log in without the need to type a password.

```
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

Many distributions (RHEL/CentOS for example) have additional permission requirements for the storage of public keys, in order for the authentication part to work properly:

```
$ chmod 700 ~/.ssh/

$ chmod 600 ~/.ssh/id_dsa.pub

$ chmod 600 ~/.ssh/id_dsa

$ chmod 600 ~/.ssh/authorized_keys
```

The ssh client can be configured to not ask for confirmation before adding hosts to the *known_hosts* file. Also it is a good idea to reduce the connection time-out, in case of network problems. This is all configured in the *~/.ssh/config* file. Please check *man ssh_config* for a complete reference.

```
$ cat ~/.ssh/config

ConnectTimeout 5
Host *
StrictHostKeyChecking no
```

Step 7. Deploy the Configuration to all Server Nodes

A check should be made that the *sshd* daemon is running in the server nodes hosts. Also, one can remove any Banner option from the *sshd_config* file in the hosts.

Finally, the front-end directory */var/lib/one/.ssh* should be copied to each of the server nodes hosts, within in the same path in the file-system.

To test that the configuration is successful, one just has to verify that *oneadmin* can log in to the server nodes hosts without being prompted for a password.

Step 8. Networking Configuration

A network connection is needed by the OpenNebula front-end daemons to access the server nodes hosts, in order to allow the management and monitoring of the hypervisors and the migration of image files. It is highly recommended to set-up a dedicated network for such management purposes.

There are various network models that can be used, but they all have something in common. They rely on network bridges with the same name as all of the hosts, to be able to connect VMs to the physical network interfaces. The simplest network model corresponds to the dummy drivers, where only the network bridges are needed.

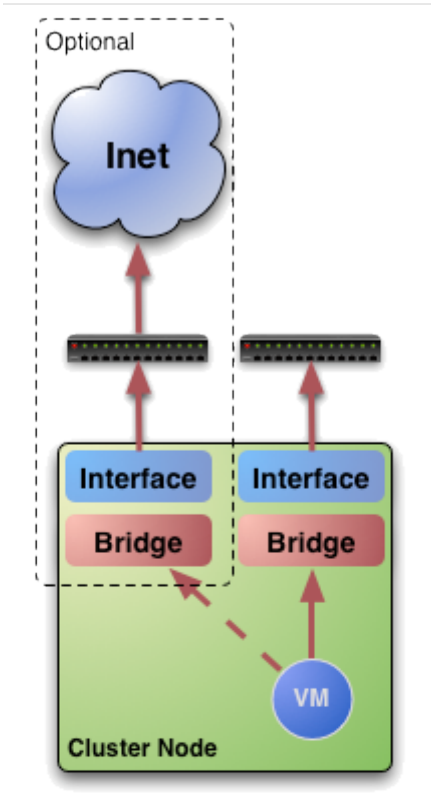

Figure 2.2: Network diagram

As an example, a typical host with two physical networks, one for public IP addresses (attached to eth0 NIC) and the other for private VLANs (NIC eth1) should have two bridges:

```
$ brctl show

bridge name bridge id          STP enabled interfaces
br0         8000.001e682f02ac  no          eth0
br1         8000.001e682f02ad  no          eth1
```

2.3 The Storage and the Storage Servers

As storage is used everywhere in the computer world, it is also used in the cloud. Here we propose SAN storage with the SAS interface connected to the storage servers. In order to provide redundancy and protection of the data, it is recommended to use RAID6. From our experience, we recommend connecting the storage servers directly to the SAN storage with SAS, so one can access the storage with high bandwidth.

As an example, at the time of writing, FCSE uses two servers for management and four servers as storage servers. These four server form a high availability cluster. This cluster is connected with SAS directly to the SAN. All the storage servers, management servers and server nodes are connected with Infiniband, so high speed access is available to the storage.

As we mentioned in the previous chapter, we use the Lustre shared file-system. To use this type of shared file system, one has to put a layer over the storage. What this means is that the Lustre file system is a kind of a virtual file system. On the two management servers, the Lustre file system is installed creating a layer over the directly connected storage. These management servers create shares that are mounted on the server nodes. What this provides is a parallel high speed file system. There are also other ways to achieve a similar setup.

3 Conclusion

Building a cloud with OpenNebula can be of benefit to gaining more experience and knowledge on what a cloud is and how it works. After the cloud is built, many experiments can be performed within in the cloud. One can do calculations in the cloud and give users the new experience of deploying virtual machines quickly and efficiently.

Using a cloud solution can save money just by starting and shutting down virtual machines appropriately. After certain services are set-up in the cloud, one can easily achieve elasticity that fits user needs.

For example: let's assume that an application is served by 4 application servers and one database. Such an architecture is predicted to serve 1000 simulations users. But what happens when there are 100 new ones? If this was a classic scenario with a load-balancer and application servers, one would have to add the additional servers manually. When using OpenNebula, it is possible to define flows and services that can monitor the application and by measuring the CPU and/or memory usage, the cloud orchestrator can automatically start or shut-down the application server or database server. In this way, one would not have to care whether the service or application could fail.

References

[1] http://opennebula.org/

Glossary

802.1Q	**S**hort for IEEE 802.1Q – the IEEE networking standard that supports multiple VLANs on an Ethernet network
HE	Higher Education
Lustre	A parallel distributed file-system, generally used for large-scale cluster computing
SAN	Storage Area Network
SAS	Serial Attached SCSI
SCSI	Small Computer System Interface
RAID6	Redundant Array of Inexpensive Disks
VLAN	Virtual Local Area Network
VM	Virtual Machine