Virtual Environments for Prototyping Tier-3 Clusters

M Mambelli, R Gardner
University of Chicago, 5735 S Ellis Ave, Chicago, IL 60637, USA
marco@hep.uchicago.edu

Abstract. The deployed hierarchy of Tier-1 and Tier-2 data centers, as organized within the context of the Worldwide LHC Computing Grid (WLCG), have without question been exceedingly successful in meeting the large-scale, group-level production and grid-level data analysis requirements of the experiments in the first full year of LHC operations. However, the plethora of derived datasets and formats thus produced, in particular large volumes of n-tuple-like refined datasets, have underscored the need for additional resource configurations facilitating data access and analysis at the institutional or individual physicist-scale. The ATLAS and CMS collaborations have long ago formalized another level in the hierarchy of their respective computing infrastructures to meet this need, namely the Tier-3 center. Only now are the detailed requirements and optimal deployment configurations for these facilities being understood as the computing models and analysis modalities evolve. Since Tier-3 centers are typically smaller in scale than Tier-2s, and may sometimes have limited staffing resources with requisite computing expertise, reliable and easy to deploy cluster and storage system configurations and recipes targeted for common deployment need to be prototyped and tested in advance. In addition, Tier-3s come in a wide variety of configurations reflecting available resources and institutional goals. This adds complexity to the task of technology providers such as the Open Science Grid (OSG), which aspires to support Tier-3 groups. This paper describes a prototyping environment for creating virtual Tier-3 clusters using virtual machine technology based on VMware ESX deployed on a simple laptop. Using virtual machines with varying network topologies, the components of a Tier-3 cluster have been studied and modularized to simplify and streamline the deployment on real environments. The virtual cluster made it possible to test different solutions and simplify the verification of the software, and more importantly the testing of its installation and configuration instructions for Tier-3 managers. Using virtual machines and machine templates, it was possible to quickly bring up complete prototype clusters to test different systems on a virtual Tier-3 platform, such as the Xrootd distributed storage system, configurations of the Condor resource management system, and data transfer services such as GridFTP and SRM.

1. Introduction
The availability of enormous amounts of data from the LHC [1], the variety of formats, and the many physicists seeking access underscored the need for resources that could be dedicated to local, “privately” managed data analysis. The computing infrastructure of the LHC experiments is comprised of the Tier-0 at CERN, the (largely national-scale) Tier-1 centers, and the many Tier-2 regional computing facilities. Although globally managed production tasks are more-or-less smoothly using this infrastructure, specialized, chaotic, institution-level analyses are sometimes difficult to perform in this environment [2, 3] which has drawn focus to Tier-3 centers. These facilities are
computing resources funded and managed by local research groups, usually at universities but in some
cases co-located in national laboratory settings, have the primary goal of meeting the needs of the
local group. Some experiments, such as ATLAS, have proposed a precise classification and somewhat
strict recommendations [2]. Others, like CMS, have been less prescriptive, leaving the Tier-3 design
and deployment mostly unspecified. Because of the many institutions involved and the diverse
institutional constraints, achieving a common, uniform Tier-3 design is therefore difficult.

The LHC Tier-3 context is similar to that of a small virtual organization (VO), such as we’ve seen
in the Open Science Grid. They too may have limited resources and may be constrained to integrate
those resources with local campus infrastructure. Recognizing this need, OSG began an initiative
targeted for small computing sites and for LHC Tier-3 centers which would provide software
configurations and documentation focusing on their specific requirements. Two important goals of this
initiative were to limit the overall maintenance effort and to help those system administrators less
experienced in computing, grid and cluster technologies. To meet these goals we needed a flexible
platform, and for that we turned to virtual machine technology (VM).

VM technology allows running several virtual machines (guests) on a single computer, the host.
The Operating System (OS) and all the software can be installed on the guests and cannot distinguish
them from a real machine. The technology is now quite mature and is used commercially for resource
consolidation and to simplify management.

The availability of a test environment is valuable to compare different Tier-3 cluster setups, to test
the proposed alternatives, and to check the correctness systems user documentation by trial-building
of sites. We found the VM machine environment ideal for this kind of testing: it does not subtract
resources from production and it is very flexible, e.g. allowing tests of network configurations, or
adding and/or removing new hosts. Since the goal was not to test scalability or performance, but
rather configuration and cluster building, a very small system could be used.

Virtual machines are used commercially, especially in the Web hosting industry [4] and there have
been tests demonstrating that grid servers can be run on a virtual machine [5]. Here the goal was to
virtualize an entire grid site, with its various headnode services, worker nodes, and associated
management nodes.

The remainder of this paper is as follows: Section 2 presents the hardware setup, the software and
the procedures used to host and manage the virtual machines. Section 3 describes the components of
the test cluster. Section 4 introduces some of the tests performed on the cluster to compare different
setups or validate the OSG documents, and we conclude with a summary and outlook in Section 5.

2. Hardware and the Virtual Cluster

There are several virtual machine frameworks available on the market including Xen, KVM,
VMware and VirtualBox among others. VMware was chosen for its reliability and performance with
Linux guests, for the wide available support, the low cost of the chosen setup, and because of existing
local expertise. VMware, like many other virtualization products, is available in two main flavors: as
an application that runs on the local operating system or as hypervisor, i.e. software that replaces the
operating system and runs directly on the bare hardware. VMware Server for Linux and the VMware
ESXi hypervisor are two virtualization products distributed for free. Two identical laptops were
prepared: one to install Linux SL 5.4 and VMware Server 2.0, and the other to install VMware ESXi
version 4 (ESXi4). The first one was more flexible, easier to adapt different network configurations,
and was compatible with the VMware client software. The second was the more efficient platform for
running many virtual machines. Specific virtual machine images were compatible with both of these
setups and could be transferred from one to the other.

2.1. Hardware
The goal for the hardware choice was to have a flexible and portable machine capable of running
VMware ESXi4. ESXi is a thin hypervisor and has limited hardware support when compared to a
regular OS. A laptop computer would not be a typical platform to run hypervisors but we wanted
portability if possible, for training purposes. The chosen model was HP EliteBook 6930p [6], selected for both technology and cost. The processor is a Mobile Intel Core 2 Duo T9600/2.8 GHz, a fast dual core 64 bit processor that supports Intel VT-x technology [7, 8] and, at the time of purchase, was more affordable than high-end quad-core processors. ESXi4 requires hardware virtualization support, either Intel's VT-x or AMD's AMD-V. The memory was upgraded to 8 GB, the maximum supported for this model. It has a fast 320 GB SATA-300 7200 rpm hard drive and a Gigabit Ethernet port; there was no support for wireless internet in ESXi. The total cost including the memory upgrade was below two thousand dollars.

As ESXi4 is a thin hypervisor, it needs a wired network connection. Several administration functions, such as the creation of new virtual machines, required client access (the vSphere client), typically from a second host.

The first laptop with VMware Server installed has the full host operating system, Scientific Linux 5.4. This was used in deploying virtual clusters in several ways:

- One can run the clients necessary to administer the virtual machines both on VMware server and ESXi
- Additional servers can be run from the host, such as a Squid Web proxy, which sped up repeated downloads of software packages
- One can provide network connectivity for both the machines, e.g. bridging the wireless and wired Ethernet network

However in practice there were drawbacks with this setup: the full VMware client requires Windows; there were problems with the wireless configuration; frequent changes in configuration, software updates, etc., frequently adversely affected VMware, requiring re-installation. Therefore in this paper we focus on the laptop setup with VMware ESXi, its configurations and the tests performed using the virtual machines running on it.

2.2. Installation and configuration of the hypervisor and router

Once VT-x is enabled in the BIOS of the laptop, ESXi installs easily using a CD with and image distributed by VMware. Both a minimal local console and the vSphere client for Windows are necessary to configure ESX and to define virtual machines. ESXi comes with no management interface, differing in this from other hypervisors like VMware ESX. Regardless, there are forums online that explain how to access a hidden console and enable an officially unsupported SSH access [9]. The shell is very limited, a custom version of BusyBox [10], but still sufficient for some troubleshooting and basic file operations.

ESXi provides a very basic network configuration: it can provide the equivalent of networks and layer-2 switches but no routing functions. Therefore a virtual router, or soft router, running in a virtual machine, was added to allow complex network configurations and Network Address Translation (NAT) for the hosts in the private network. Vyatta Community Edition [11], version 5, was chosen because it is a robust and feature-rich router and is free. Some alternatives considered but discarded were a generic Linux box with iptables; m0n0wall [12], a free BSD based router; IPCop [13], a Linux based, small office type router. The router was configured to provide additional useful features: a VPN configuration using PPTP [14, 15], used mainly to manage the cluster; a DNS server that helps in the deployment of new nodes; and port forwarding to allow the connection to servers in the private network.

The guest operating system installed on all the virtual servers is Scientific Linux [16], version 5.4. This is the OS of choice for the LHC experiments and the most common OS in Open Science Grid; therefore it is fully supported by the Virtual Data Toolkit (VDT), the middleware produced by OSG.

2.3. Guest deployment and management

vSphere [17, 18], the VMware infrastructure to deploy and manage virtual machines, provides a wide set of features including resource partitioning, the migration of running VMs between servers, and the cloning of templates to simplify the deployment of clusters. It also provides a control center,
graphical clients, a scripting language and an API to interact in the more convenient way depending on the scale of the resources. Using the free products many of these features are not available but the access to the console of the ESXi server allowed one to simplify the deployment and to prototype using some rudimentary cloning and templating mechanisms based on files copy.

To avoid the installation from scratch of each new virtual machine, sometimes the disk of an existing host was copied. This could only be done when the VMs were turned off. A couple of disks were copied and modified to serve as a template for a host with minimal software, a worker node, a more complex host, and for a dual-homed host, i.e. a node with multiple network interfaces.

The typical worker node consisted of a virtual machine with 256MB of RAM and a minimal installation of SL 5.4 on a 10GB virtual drive. Server nodes had 512MB RAM and additional virtual drives or network interfaces if needed.

The vSphere client is demanding: it is graphic intensive and it requires a Windows machine or VM. Therefore it is used only to define new VMs, from scratch or from a copied disk, to turn the host on and off and whenever console access is needed. Once the guest nodes have been installed and configured to access the network they are managed via an SSH connection. The guests on the extranet, connected to the host network interface, are accessible directly. Thanks to the Vyatta router it is also easy to access the guests on the private network, using a static route or a VPN.

3. Cluster Setups

The Vyatta router, labeled as NAT because it hosts the Network Address Translation server, separates a private network (intranet) from the external one (extranet) as depicted in Figure 1. The external network can be a public network or departmental LAN. This was not important because the virtual environment was not affected by it.

Depending on the needs and the desired setup, hosts are created and connected to the appropriate networks. Most of the servers were connected to the private network, like most of the nodes in a typical Tier-3. Some servers were on both the private and the external networks, like the head nodes of a Tier-3; others were placed in different configurations for specific tests.

![Figure 1. Network topology of a virtual Tier-3 cluster](image)

In a virtual Tier-3 cluster, just like in an actual Tier-3 cluster, there are typically three classes of nodes: worker nodes (WN in Figure 1) that execute jobs submitted via a batch queue, shared interactive nodes (UI) where users can login directly and run applications or submit jobs, and nodes that serve various roles such as batch queues (HN for headnode), file systems (NFS), or grid middleware services (e.g. CE, SE in Figure 1). The first two classes are composed by multiple instances of the same node, while the third class is the most important for the virtual cluster since it includes services that are distributed by OSG: these are the nodes and the software that require extensive testing and documentation checking. In some cases one node can host multiple roles while in other cases, for a variety of reasons including security or performance, nodes should be set aside for single purpose use.

All nodes had outbound network connectivity, either directly or through the router. To simplify the configuration all the nodes in the private network had IP addresses assigned in a hosts file replicated
across the cluster. Also other main configuration files, like user, group and password files, were replicated across the cluster. One node served as an NFSv4 shared file system from a 30GB virtual drive. This was used for home directories and shared space for grid and application software. Other nodes were created and defined as needed for specific tests, as described in the next section.

4. Testing and Validation
An OSG Tier-3, in general, is a small to medium cluster/grid resource targeted at supporting a small group of physicists local to one institution. Tier-3 systems typically provide one or more of the following capabilities: access to local computational resources using a batch queue, interactive access to local computational resources, storage of large amounts of data using a distributed storage system, access to external computing resources on the Grid, and the ability to transfer large datasets to and from the Grid. Tier-3s can also offer computing resources and data to fellow grid users.

The Local Resource Manager (LRM), or queue manager, is a critical component of a batch cluster but normally OSG leaves the selection, configuration and support of the LRM to the local system administrator who usually has an existing preference. However OSG decided to recommend Condor [19] to Tier-3s without an existing or preferred LRM for its ease of use, flexibility, and wide support within the OSG community. Four nodes of the virtual cluster were used to test different setups of Condor: one node for the services matching the available resources with the jobs (the Condor collector and negotiator); two to run the jobs and verify the scheduling policies; and one to submit the jobs. Both an installation on a shared file system and the local RPM [20] installation of Condor documented in the OSG Twiki [21] were tested. This setup was used to test and provide useful feedback also to the Condor team to define the current Condor RPM distribution.

The Storage Element (SE) is the server used to provide access to the local storage system from the grid. Nodes in the cluster were used to test new versions of the software or new configurations for different storage elements: GridFTP servers and multiple BeStMan setups were tested. Some tests involved machines connected to a single network, some involved machines connected to both to a private and a public network. This last configuration, uncommon in larger resources but increasingly common in Tier 3s, allowed us to discover and fix several shortcomings of the software and of the recommended configurations.

Not all the tests required bringing up new virtual machines. The SE node and the Condor worker nodes were reused to test the Xrootd distributed file system [22]. Xrootd requires one redirector, the front-end of the system installed on the SE node, and one or more pools serving the actual disk installed on other nodes. Besides testing the Xrootd instructions [23], this setup was used to iterate with OSG’s VDT team and ATLAS to debug Xrootd and to define the recommended configuration for ATLAS Tier-3s.

The portability of the host allowed for live demonstrations at grid schools and in venues with limited bandwidth to OSG software caches. Furthermore the fact that a simple laptop could run a complete grid cluster was encouraging for new system administrators.

5. Summary and Outlook
The laptop virtual environment presented in this paper was used for successful deployments of complete and flexible grid sites, allowing useful prototyping of typical Tier-3 setups. It has been used to test several configurations useful for LHC Tier-3s and small OSG sites. It also has contributed to the testing of integration release candidates of OSG software.

We expect that new Tier-3 configurations will be tested on the cluster, including future versions of the OSG and storage system software; we expect to discover ways to simplify the deployment and management of a cluster and the possibility to grid enable external resources. A parallel effort will improve the portability of the cluster, e.g. exploring the use of dynamic DNS, to simplify the remote deployment and to allow more complete demonstrations.

6. Acknowledgements
The research was supported in part by the National Science Foundation, Grant 0621704, and by the Department of Energy, DE-FC02-06ER41436.

References
[12] The m0n0wall project, http://m0n0.ch/wall/
[16] Scientific Linux: https://www.scientificlinux.org/