Experiences with Managed Hosting of Virtual Machines

Dustin Leverman[2], Michael Oberg[2],
Henry M. Tufo[1,2], and Matthew Woitaszek[2]
oberg@ucar.edu

[1] University of Colorado, Boulder, CO
[2] National Center for Atmospheric Research, Boulder, CO

Abstract. Virtual machines (VMs) are emerging as a powerful paradigm to deploy complex software stacks on distributed cyberinfrastructure. As a TeraGrid resource provider and developers of Grid-enabled software, we experience both the client and system administrator perspectives of service hosting. We are using this point of view to integrate and deploy a managed hosting environment based on VM technology that fulfills the requirements of both developers and system administrators. The managed hosting environment provides the typical base benefits of any VM hosting technique, but provides additional software support for developers and system monitoring opportunities for administrators. This paper presents our experiences creating and using a prototype managed hosting environment. The design and configuration of our current implementation is described, and its feature set is evaluated from the context of several current hosted projects.

1 Introduction

From the perspective of application software developers and resource provider system administrators, deploying applications with complex dependencies and software environments is time-consuming and labor-intensive. Virtual machines (VMs) have emerged as a method to deploy these software stacks across distributed resources quickly and easily, as the software developers can produce a golden image of their application and simply ship it – operating system and the entire execution environment – to resources across a variety of administrative domains. Yet hosting a black-box VM requiring external network connectivity is not something that many hosting organizations are prepared to do. Hosting externally provided VMs requires the hosting organization to trust that the author of the VM image has followed common system administration and security best practices, and also implies a committment from the author to ensure the host's security over time.

The appearance of portable applications made possible through VM shipping departs from the highly structured and standards-based community-wide software environment constituting various Grids. Grids generally provide access

to common services through well-defined interfaces, ranging from basic supercomputer job submission to "science as a service" service-oriented architectures. The new paradigm of access to bare-bones distributed resources, now referred to under the Cloud computing umbrella, provides immense flexibility to application developers but requires that they re-assume some of the administrative duties formerly in the purview of system administrators of Grid hosts. Somewhere between Grid computing and Cloud computing is a grid hosting environment design that provides application developers sufficient access to their private project hosts while giving hosting organizations an attractive and secure platform for hosting of community services.

As a TeraGrid resource provider and developers of Grid-enabled software, we have encountered both benefits and challenges while developing and hosting Grid-based software and services. Previously, we dedicated substantial personnel resources to maintaining legacy hosts used to support aging Grid projects on servers with specific software stacks. There has typically been significant local system administrator involvement in the deployment, security hardening, and ongoing support of these hosts and services. Standardizing on VM-based hosting in a managed environment isolates the dependencies while simplifying administrative requirements. Beyond our own projects, we also anticipate being asked to host VMs for other groups at NCAR and our collaborators. We are therefore interested in construcing a hosting environment suitable for user-supplied VMs as well.

However, running a large collection of VMs presents its own challenges. As a pathological case of the sterotypical machine room "pizza box server creep," an organization providing a VM hosting service may find an increase in the administrative burden where a reduced staff time committement was expected, simply due to multiplicative effects in running large pools of hosts. Rather than pure black-box VM hosting, such as provided by commercial cloud computing resource providers, our managed hosting approach moves the administrative burden to the individual projects, but still preserves a minimal amount of local administrator control in order to effectively integrate the VMs with the local security and performance monitoring systems, as well as provide management capability in the case of a time-critical issue.

The remainder of this paper is organized as follows: We begin by describing background and related work in the area of VM hosting on Clouds and their applications on Grids. Section 3 describes the design and implementation of our managed hosting environment, starting with a description of the standard VM hardware and software stack used in our prototype and then the custom software that manages the host and client environments. Section 4 describes several use cases of the managed hosting environment, and the subsequent section discusses strengths and limitations that these hosting scenarios have exposed. The paper concludes with our ideas for adopting existing VM hosting standards and further standardizing managed hosting, both of which constitute our continuing and future work.

2 Background and Related Work

One of the first applications of VM technology to Grid-based science was the Globus Workspace Service, a collection of Grid-enabled utilities that allowed users to create and deploy virtual workspaces – entire virtualized clusters – intended to support the deployment of a single application's complex software stack [5]. Software companies such as rPath started to provide simple methods to package applications for execution on a variety of VM technologies. Meanwhile, Cloud computing attained buzzword status. Whereas prior work with VMs in Grid computing emphasized the creation of virtual clusters (such as Globus Workspace Service) through carefully constructed Grid-compliant methods, Clouds made resources to run VMs available to anyone with a VM image and a credit card. Combined with virtual private networking, current Cloud environments easily support both the dynamic creation of virtual clusters and hosts as exposed network services.

Several resource providers now offer support for VM-based Clouds. For example, Indiana University provides a VM hosting service for TeraGrid projects on a special request basis [2]. In more recent cloud hosting initiatives, Amazon EC2's dominance of the VM-based Cloud computing marketplace has resulted in a de facto standardization around their control interface [3]. Globus Workspaces, now a component of the larger Nimbus project, supports both the Amazon EC2 and older WSRF interfaces [7] and is available for users via the Nimbus Cloud at the University of Chicago and the Stratus Cloud at the University of Florida [8]. Finally, the Eucalyptus project recreated the control interface used by Amazon's EC2 system in an open-source implementation that allows organizations with resources to deploy their own EC2-compatible clouds [9].

As the concept of deploying applications in packaged VM images gained popularity in mid-2006, the issues of VM deployment and continued lifecycle maintenance became of interest to computational resource providers in the academic community. For example, the developers of the Bcfg2 system management software package and the Globus Workspace Service project collaborated to address creating and deploying VMs in a scalable fashion, using existing administration tools to perform site-specific binding and perform site-mandated maintenance automatically [4]. We have found that application developers are very hesitant to accept site resource provider intervention; comments regarding the idea emphasized site administrators accidentally breaking entire projects while trying to helpfully apply patches. Thus, any intervention must be specifically defined in advance and limited to critical operations.

From the perspective of a resource provider, the continued maintenance of a VM is of particular importance. Part of the attractiveness of a VM distribution is that the software environment is indeed solidified in a VM image that is known to work and deployed. "Deploy and Forget" works from the software functionality perspective but is problematic given routine discovery of software bugs and security vulnerabilities. In Amazon's EC2 cloud, security is left as an issue for developers. Developers are encouraged to update their operating system distributions and configure firewalls to protect their VM instances lest they be

charged for data transfers occuring as the result of a compromise [1]. This laissez-faire approach to VM maintenance is problematic for most resource providers, as it requires them to host virtual systems that they do not control. This hosting arrangement lacks incentive for development groups to commit resources to maintenance; moreover, the natural funding lifecycle of many academic projects leads to declining maintenance over time. In our prior work, we examined several complimentary layers of software environment assurance, including both container network monitoring and access-based introspection, that is sufficient for us to securely host a shipped VM on our external network [6]. In this paper, we describe our use of this managed hosting environment in production.

3 Design and Implementation

Our managed hosting environment consists of standard VM hosting hardware infrastructure (e.g., a pool of servers and a storage network), the Xen virtualization software [10], and a collection of common and custom software applications that provide the enhanced managed hosting features. We have augmented the cluster-based design found on many Clouds in academic settings with GPFS-based image storage so we can support a wide range of VM features such as live migration and take advantage of our HPC infrastructure for performance and scalability.

For our prototype, we configured a managed hosting infrastructure with a pool of heterogeneous servers. The project started with the realization that a small number of many-core servers could replace a small stack of hosts and provide enhanced manageability and reliability through the use of virtualization. As the usefulness of the virtual hosting environment became clear, additional servers were converted from single-task purposes to become VM hosts, in many cases hosting the same applications and support software stack but in a more flexible and manageable framework.

Our external TeraGrid-based systems are generally under intense scrutiny from NCAR's computer security department due to their presence outside of centralized firewalls. As a result, we configure our systems in trust hierarchies in an attempt to mitigate the effects of a security incident through isolation and separation of privilige. For virtual machine hosting, our underlying threat model considers the possibility of an attacker breaking out of a compromised VM and obtaining root on the underlying VM container. In this situation, the attacker could easily obtain root access on all other VMs on that container as well as improperly configured mounted network file systems. Thus, we chose to segregate our virtual hosting environment into security domains based on VM origination and network access requirements, and we expect to create new security domains for collections of projects requiring such isolation. At the present time, we host one security domain for internal management servers, one for external hosts running limited non-public support services, and one for "more trusted" (locally administered) publicly accessible NCAR Grid projects. We create additional security domains for external projets as required, such as our cluster construction

educational activities. Virtual machines and containers are assigned to these logical security domains. This limits deployment flexibility but provides basic domain isolation so that we can run projects from a wide variety of users without artificially increasing risk.

File storage for VM image hosting is provided by a GPFS-based storage cluster used in our high-performance computing environment. Our existing storage cluster was designed to provide the primary working space for a Blue Gene/L rack, and to support this work we allocated a small portion of this system to store VM images. To isolate security domains, we create individual GPFS file systems from the storage pool and control which VM containers may mount those file systems. The use of GPFS provides all of the benefits commonly associated with parallel file systems to our image storage space, including fault-tolerance, large capacity, flexible management, and high-throughput access. By using shared storage, we can also take advantage of Xen's live migration capabilities. This provides administrators the ability to transparently perform maintenance on container hardware and statically balance host load based on project computational requirements.

In our prototype virtual hosting environment, we have been using a collection of custom scripts along with our standard administrative tools to manage our virtual machines. Our software management suite performs image creation and cloning, basic host localization and both configures and monitors our ability to enter the host administratively. Moreover, each VM container performs networking monitoring on all traffic to and from the individual hosts [6]. For the administrative and NCAR-based Grid projects running in this environment, the hosts have been integrated with our local filesystem integrity scanning and logging infrastructure.

To support a broad range of internal and external connectivity requirements, each container provides Xen Ethernet bridges to multiple networks as appropriate for that security domain. Each Xen image is configured to use one or more of these interfaces. While each container uses a host-based firewall to protect itself, our custom Xen network startup scripts add individual firewall rules on the appropriate Xen bridge interfaces to provide forwarding of all traffic to and from every guest. This protects the container while allowing individual projects to manage their own host-based firewall configuration.

Our basic VM software utility collection, implemented as a series of bash and python scripts, assists in the areas of VM cloning, deployment, and monitoring. While external groups will use the VM creation technology of their choice, we use a simple cloning tool to create new VMs from a golden image similar to our process for imaging physical nodes in our previous Linux cluster. Our deployment software configures new VMs for activation on our network based on centralized IP allocation. VMs under our direct control are integrated with our existing monitoring infrastructure; those not under our control are monitored only using network intrusion detection.

The containers themselves are managed similar to a small cluster, where common configuration files, system packages, and routine management tasks

are performed by a central management server. These hosts are integrated with the datacenter monitoring and security infrastructure, such as the Samhain file system integrity scanner and centralized logging, further simplifying the management of the containers and providing a high level of monitoring to ensure availability and early detection of security issues.

4 Managed Hosting in Action

We are currently using our managed hosting environment for five persistent projects and a collection of short-term activities. These projects are described below. The hosting environment has been quite successful; we have been able to manage our own systems as a cluster, create separate domains for educational projects with additional priviliges for certain users, and use VM live migration to perform container maintenance.

4.1 GridBGC - A Monolithic Grid Science Gateway

GridBGC is a Grid-based terrestrial carbon cycle modeling system. Users interact with GridBGC via a science gateway website, and the remote Grid job and file transfer operations are performed using a custom service-oriented architecture. As the website requires Apache Tomcat, a custom database, the Spring and Struts web development frameworks, the Hibernate, Globus, and BouncyCastle libraries, and custom code to perform geographic domain decomposition tasks, deploying the GridBGC science gateway is quite complex. To avoid the need to reinstall the software again, we have placed the system in a single virtual machine. The GridBGC gateway requires substantial hardware resources – it was unable to run on a VM with less than 4GB RAM and requires over a terabyte of storage to facilitate data staging and processing.

4.2 GridAMP - A Decoupled Grid Science Gateway

The Asteroseismic Modeling Portal (AMP) is a science gateway that allows astronomers to execute a modeling pipeline that identifies the characteristics of stars matching observed oscillations. Rather than the traditional monolithic science gateway design, the system has been split into two lightweight components – an interactive database-backed website and a workflow processing engine referred to as GridAMP. These services are run on separate VMs in separate security domains and only interact through the database, so that a website compromise on one VM (or indeed an entire VM container) cannot provide access to a TeraGrid community account proxy certificate.

4.3 Virtual Clusters for Education

One of the most successful impromptu uses of the managed hosting environment was the creation of a systems platform for a laboratory exercise as part

of the University of Colorado's High Performance Scientific Computing (HPSC) course. Students in the HPSC course typically develop software and examine performance on various TeraGrid systems. This year, to give students practical experience on smaller-scale systems similar to departmental clusters, a single week cluster construction lab exercise was introduced emphasizing the basic system administration skills required to turn a collection of hosts into a MPI-capable cluster. (We have found that after this type of exercise, many students install Torque on their departmental multi-core servers even if only to schedule single-processor jobs submitted by their colleagues.)

Using the managed hosting environment, we were able to prepare five virtual clusters with less than a business day of advance notice. Each virtual cluster consisted of a head node accessible through the Internet and four compute notes on a private network shared with the head node. The compute nodes were placed on dedicated container hosts without oversubscribing hardware so that each group could execute software benchmarks on their virtual cluster and obtain results free from interference from other VMs.

To minimize the instructional overhead, all of the HPSC virtual machines were placed on dedicated containers and the course teaching assistants were given superuser priviliges on those containers. The managed hosting component proved invaluable during the laboratory session. While many groups required a bit of assistance, one group managed to overwrite the /tt shadow file with an incompletely specified /tt pdcp command; a course instructor was able to quickly remount and correct the problem. A hardware-oriented cluster construction exercise would have required attaching a console and performing basic system recovery.

4.4 BitTorrent Hosting

We are in the process of deploying an additional virtual machine to serve as a NCAR community BitTorrent host. All of NCAR's other networks, both internal and external, are protected by firewalls that degrade the performance of BitTorrent file transfers. We intend to offer this service not only for NCAR's information technology needs (for trivial tasks such as downloading ISOs of new operating system distributions), but also to host certain popular data sets using the BitTorrent protocol. Our initial deployment, still under construction, will use Grid technologies to maintain replicas of data sets on 10Gbps-connected systems at NCAR and the University of Colorado. The virtual hosting platform, in particular its direct access to our high-capacity GPFS parallel file system, provides a convenient location for this service.

4.5 Server Consolidation

Finally, the managed virtual hosting environment has been useful for satisfying the typical production supercomputing group's relentless demand for additional ancillary servers for various specific purposes. There are, of course, no externally observable benefits to running a collection of web servers providing mailman,

Ganglia, and similar services in our managed hosting environment instead of a traditional virtual machine deployment. It is the ease of creating and deploying additional virtual machines – preconfigured to integrate with our server security and network monitoring infrastructure – that makes use of the managed hosting environment attractive. While configuring additional virtual machines is fairly straightforward, the custom client-side configuration support built as part of this prototype saves administrator time and ensures the consistency of the deployment.

5 Discussion and Future Work

We have found that our unified VM hosting deployment has satisfied the diverse set of requirements in our environment, providing support for administrative and support systems, NCAR based Grid projects, and projects from other research groups. This hosting environment provides a resilient platform for production servers while maintaining sufficient flexibility to allow for rapid deployment of temporary and test systems. This infrastructure allows us to rapidly pursue interesting research topics without the need for supplemental hardware.

The minimal level of custom software development was surprisingly useful in reducing administrative overhead. The host customization scripts simplified deployment of new VMs and provided image consistency for our own projects. Managing the containers as a small cluster (complete with a parallel file system) provided additional management and scalability benefits. This eased the integration with the datacenter managment and monitoring infrastructure, and gives administrative staff access to common tools, reports and procedures.

The primary weakness of our approach is that it does not support the newly emerging Cloud computing administrative standards. At the time this work began, the Amazon EC2 interfaces was not a de facto standard, and the Globus Workspace Service emphasized the creation of entire virtualized clusters instead of persistent network services. We are evaluating the recently developed VM management tools that support standard Cloud computing protocols, such as Nimbus and Eucalyptus, to appeal to the broader Cloud computing community. This will require substantial re-integration work so that the deployment software stack recognizes our hosting cluster's storage and networking services.

We would also like to expand the set of software that we offer to VMs hosted in our environment. Notably, TeraGrid related projects require both account synchronization and support of the TeraGrid single sign-on system. These tasks have already been integrated with our current environment, and extending this support to hosted VMs would reduce the burden on those groups and projects.

6 Conclusion

Overall, our experiences with managed service hosting through VM technology have proven to be worth the investment in establishing a new cluster and related support infrastructure. The decision to utilize a parallel file system in the design

to store the VM images provided a reliable, scalable and high-performance solution while allowing easy separation into security domains. We are confident that this design will support our VM hosting requirements for the forseeable future.

While we started with very specific goals for using VMs to host our own projects, the use of this technology has been popularized by Cloud computing. To make the hosting environment useful to research groups experimenting with this technology, we intend to evaluate and deploy a standards-based Cloud control mechanism such as Nimbus or Eucalyptus. This standardization will support the establishment of common Grid hosting environments across a large number of organizations, which is critical to promoting investment in large-scale science on distributed infrastructure.

Acknowledgments

We would like to thank Bobby House and Paul Marshall for the initial work with virtual machines at NCAR. Computer time and support were provided by NSF MRI Grant #CNS-0421498, NSF MRI Grant #CNS-0420873, NSF MRI Grant #CNS-0420985, NSF sponsorship of the National Center for Atmospheric Research, the University of Colorado, and a grant from the IBM Shared University Research (SUR) program.

References

1. Tips for securing your EC2 instance. http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1233&categoryID=100.
2. Indiana University announces new TeraGrid services. http://rtinfo.uits.indiana.edu/news/index.shtml?/archive/2007-09s.single, September 2007.
3. Amazon.com, Inc. *Amazon Web Services*. http://www.amazon.com/aws/.
4. R. Bradshaw, N. Desai, T. Freeman, and K. Keahey. A scalable approach to deploying and managing appliances. In *Proceedings of TeraGrid 2007*, Madison, WI, June 2007.
5. I. Foster, T. Freeman, K. Keahey, D. Scheftner, B. Sotomayer, and X. Zhang. Virtual clusters for grid communities. In *Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid*, May 2006.
6. B. House, P. Marshall, M. O. abd H. M. Tufo, and M. Woitaszek. Grid service hosting on virtual clusters. In *Proceedings of the 9th IEEE/ACM International Conference on Grid Computing (Grid 2008)*, Tsukuba, Japan, September 2008.
7. K. Keahey, R. Figueiredo, J. Fortes, T. Freeman, and M. Tsugawa. Science clouds: Early experiences in cloud computing for scientific applications. In *Proceedings of Cloud Computing and its Applications*, Chicago, IL, August 2008.
8. K. Keahey and M. Pierce. Virtualization, cloud computing, and TeraGrid. Presentation at TeraGrid 2008, Las Vegas, NV, June 2008. http://workspace.globus.org/papers/TeraGrid2008.ppt, June 2008.
9. D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The Eucalyptus open-source cloud-computing system. In *Proceedings of Cloud Computing and Its Applications*, Chicago, IL, October 2008.
10. XenSource Inc. *Xen Community*. http://xen.xensource.com/.