



Virtual Infrastructure Management in Private and Hybrid Clouds

One of the many definitions of “cloud” is that of an infrastructure-as-a-service (IaaS) system, in which IT infrastructure is deployed in a provider’s data center as virtual machines. With IaaS clouds’ growing popularity, tools and technologies are emerging that can transform an organization’s existing infrastructure into a private or hybrid cloud. OpenNebula is an open source, virtual infrastructure manager that deploys virtualized services on both a local pool of resources and external IaaS clouds. Haizea, a resource lease manager, can act as a scheduling back end for OpenNebula, providing features not found in other cloud software or virtualization-based data center management software.

Borja Sotomayor
University of Chicago

**Rubén S. Montero
and Ignacio M. Llorente**
*Universidad Complutense
de Madrid*

Ian Foster
*Argonne National Laboratory,
University of Chicago*

Cloud computing is, to use a cloud-inspired pun, a nebulously defined term. However, it was arguably first popularized in 2006 by Amazon’s Elastic Compute Cloud (EC2; www.amazon.com/ec2/), which started offering virtual machines (VMs) for US\$0.10 an hour using both a simple Web interface and a programmer-friendly API. Although not the first to propose a utility computing model, Amazon EC2 contributed to popularizing the infrastructure-as-a-service (IaaS) paradigm, which became closely tied to the notion of cloud computing. An *IaaS cloud* enables on-demand provisioning of computational resources in the form of VMs deployed in a cloud provider’s data center (such as Amazon’s), mini-

mizing or even eliminating associated capital costs for cloud consumers and letting those consumers add or remove capacity from their IT infrastructure to meet peak or fluctuating service demands while paying only for the actual capacity used.

Over time, an ecosystem of providers, users, and technologies has coalesced around this IaaS cloud model. More IaaS cloud providers, such as GoGrid, FlexiScale, and ElasticHosts, have emerged, and a growing number of companies base their IT strategy on cloud-based resources, spending little or no capital to manage their own IT infrastructures (see <http://aws.amazon.com/solutions/case-studies/> for several examples). Some providers, such as

Elastra and Rightscale, focus on deploying and managing services on top of IaaS clouds, including Web and database servers that benefit from such clouds' elastic capacity, and let their clients provision services directly instead of having to provision and set up the infrastructure themselves. Other providers offer products that facilitate working with IaaS clouds, such as rPath's rBuilder (www.rpath.org), which enables users to dynamically create software environments to run on a cloud.

Although this ecosystem has evolved around *public clouds* – commercial cloud providers that offer a publicly accessible remote interface for creating and managing VM instances within their proprietary infrastructure – interest is growing in open source cloud computing tools that let organizations build their own IaaS clouds using their internal infrastructures. These *private cloud* deployments' primary aim isn't to sell capacity over the Internet through publicly accessible interfaces but to give local users a flexible and agile private infrastructure to run service workloads within their administrative domains. Private clouds can also support a *hybrid cloud* model by supplementing local infrastructure with computing capacity from an external public cloud. Private and hybrid clouds aren't exclusive with being public clouds; a private/hybrid cloud can allow remote access to its resources over the Internet using remote interfaces, such as the Web services interfaces that Amazon EC2 uses. Here, we look at two open source projects that facilitate the management of such private/hybrid cloud models.

Virtual Infrastructure Management

To provide users with the same features found in commercial public clouds, private/hybrid cloud software must

- provide a uniform and homogeneous view of virtualized resources, regardless of the underlying virtualization platform (such as Xen, Kernel-based Virtual Machine (KVM), or VMware);
- manage a VM's full life cycle, including setting up networks dynamically for groups of VMs and managing their storage requirements, such as VM disk image deployment or on-the-fly software environment creation;
- support configurable resource allocation policies to meet the organization's specific

goals (high availability, server consolidation to minimize power usage, and so on); and

- adapt to an organization's changing resource needs, including peaks in which local resources are insufficient, and changing resources, including addition or failure of physical resources.

Thus, a key component in private/hybrid clouds will be *virtual infrastructure (VI) management*, the dynamic orchestration of VMs that meets the requirements we've just outlined. Here, we discuss VI management's relevance not just for creating private/hybrid clouds but also within the emerging cloud ecosystem. Our two open source projects, OpenNebula (www.opennebula.org/) and Haizea¹ (<http://haizea.cs.uchicago.edu/>), are complementary and can be used to manage VIs in private/hybrid clouds. OpenNebula is a VI manager that organizations can use to deploy and manage VMs, either individually or in groups that must be coscheduled on local resources or external public clouds. It automates VM setup (preparing disk images, setting up networking, and so on) regardless of the underlying virtualization layer (Xen, KVM, or VMware are currently supported) or external cloud (EC2 or ElasticHosts are currently supported). Haizea is a resource lease manager that can act as a scheduling back end for OpenNebula, providing leasing capabilities not found in other cloud systems, such as advance reservations (ARs) and resource preemption, which are particularly relevant for private clouds.

The Cloud Ecosystem

VI management tools for data centers have been around since before cloud computing became the industry's new buzzword. Several of these, such as the Platform VM Orchestrator (www.platform.com/Products/platform-vm-orchestrator), VMware vSphere (www.vmware.com/products/vsphere/), and Ovirt (<http://ovirt.org>), meet many of the VI management requirements we outlined earlier, providing features such as dynamic placement and VM management on a pool of physical resources, automatic load balancing, server consolidation, and dynamic infrastructure resizing and partitioning. Although creating what we now call a private cloud was already possible with existing tools, these tools lack other features that are relevant for building IaaS clouds, such as pub-

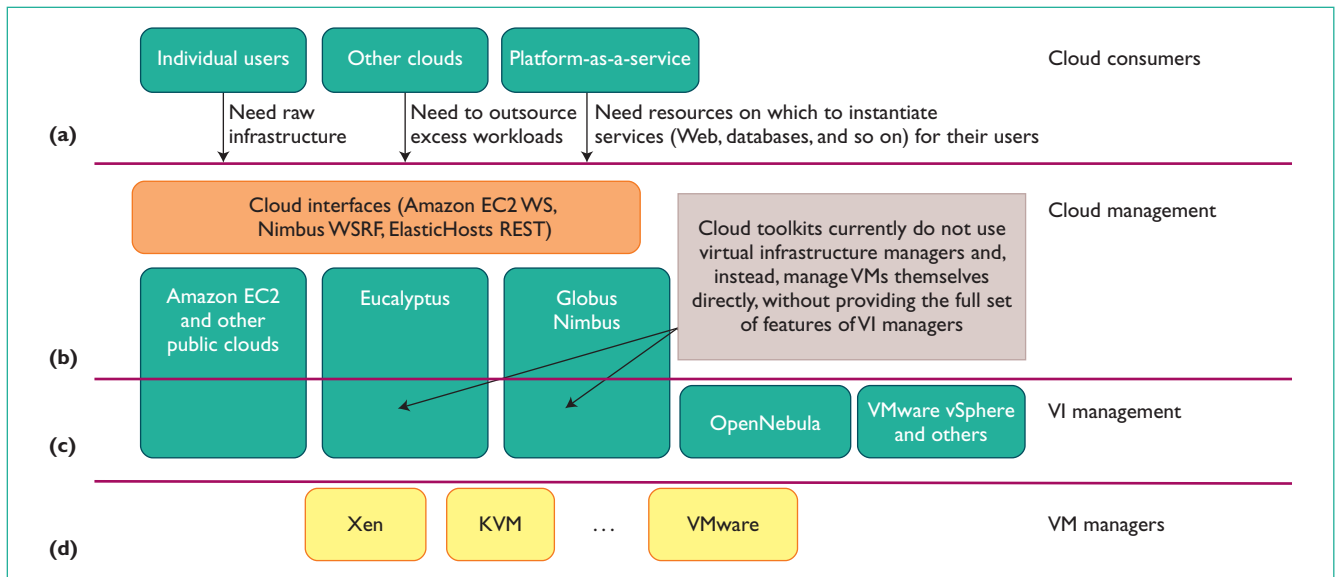


Figure 1. The cloud ecosystem for building private clouds. (a) Cloud consumers need flexible infrastructure on demand. (b) Cloud management provides remote and secure interfaces for creating, controlling, and monitoring virtualized resources on an infrastructure-as-a-service cloud. (c) Virtual infrastructure (VI) management provides primitives to schedule and manage VMs across multiple physical hosts. (d) VM managers provide simple primitives (start, stop, suspend) to manage VMs on a single host.

lic cloud-like interfaces, mechanisms for adding such interfaces easily, and the ability to deploy VMs on external clouds.

On the other hand, projects such as Globus Nimbus² (<http://workspace.globus.org>) and Eucalyptus³ (www.eucalyptus.com), which we term *cloud toolkits*, can help transform existing infrastructure into an IaaS cloud with cloud-like interfaces. Eucalyptus is compatible with the Amazon EC2 interface and is designed to support additional client-side interfaces. Globus Nimbus exposes EC2 and Web Services Resource Framework (WSRF) interfaces and offers self-configuring virtual cluster support. However, although these tools are fully functional with respect to providing cloud-like interfaces and higher-level functionality for security, contextualization, and VM disk image management, their VI management capabilities are limited and lack the features of solutions that specialize in VI management.

Thus, an ecosystem of cloud tools is starting to form (see Figure 1) in which cloud toolkits attempt to span both cloud management and VI management but, by focusing on the former, don't deliver the same functionality as software written specifically for VI management. Although integrating cloud management solutions with existing VI managers would seem like the obvious solution, this is compli-

cated by the lack of open and standard interfaces between the two layers, and the lack of certain key features in existing VI managers. Our aim, therefore, is to produce a VI management solution with a flexible and open architecture that organizations can employ to build private/hybrid clouds.

With this goal in mind, we started developing OpenNebula and continue to enhance it as part of the EU's Reservoir project (www.reservoir-fp7.eu), which aims to develop open source technologies to enable the deployment and management of complex IT services across different administrative domains. OpenNebula provides similar functionality to that found in existing VI managers but also aims to overcome those solutions' shortcomings – namely,

- the inability to scale to external clouds;
- monolithic and closed architectures that are hard to extend or interface with other software, not allowing seamless integration with existing storage and network management solutions deployed in data centers;
- a limited choice of preconfigured placement policies (first fit, round robin, and so on); and
- a lack of support for scheduling, deploying, and configuring groups of VMs (for example, a group of VMs representing a cluster, where

Table 1. Comparison of tools providing virtual infrastructure management capabilities.

Tool	Provisioning model	Default placement policies	Configurable placement policies	Support for hybrid cloud	Remote interfaces
Amazon EC2	Best-effort	Proprietary	Proprietary	No	EC2 Web services API
VMware vSphere	Immediate	Initial placement on CPU load and dynamic placement to balance average CPU or memory load and consolidate servers	No	Only when both the local and external cloud use vSphere	vCloud API
Platform Orchestrator	Immediate	Initial placement on CPU load and migration policies based on policy thresholds on CPU utilization level	No	No	No
Nimbus	Immediate	Static-greedy and round-robin resource selection	No	Includes an “EC2 back end” that can forward requests to EC2, but local and remote resources must be managed separately	EC2 Web services API and Nimbus Web Services Resource Framework
Eucalyptus	Immediate	Static-greedy and round-robin resource selection	No	No	EC2 Web services API
oVirt	Immediate	Manual mode	No	No	No
OpenNebula 1.2	Best-effort	Initial placement based on requirement/rank policies to prioritize those resources more suitable for the virtual machine (VM) using dynamic information and dynamic placement to consolidate servers	Support for any static/dynamic placement policy	Driver-based architecture allows interfacing with multiple external clouds; supports EC2-compatible clouds and ElasticHosts	No
OpenNebula 1.2/ Haizea	Immediate, best-effort, and advance reservation (AR)	Dynamic placement to implement AR leases	VM placement strategies supporting queues and priorities	Driver-based architecture allows interfacing with multiple external clouds; supports EC2-compatible clouds and ElasticHosts	No
OpenNebula 1.2/ Reservoir	Immediate and best-effort	Load balancing and power-saving policies	Support for policy-driven probabilistic admission control and dynamic placement optimization to satisfy site-level management policies	Driver-based architecture allows interfacing with multiple external clouds; supports EC2-compatible clouds and ElasticHosts	Reservoir VEE (virtual execution environment) manager interface

all the nodes either deploy entirely or don't deploy at all, and where some VMs' configuration depends on others', such as the head-worker relationship in compute clusters).

Table 1 provides a more detailed comparison between OpenNebula and several well-known VI managers, including cloud toolkits that perform VI management.

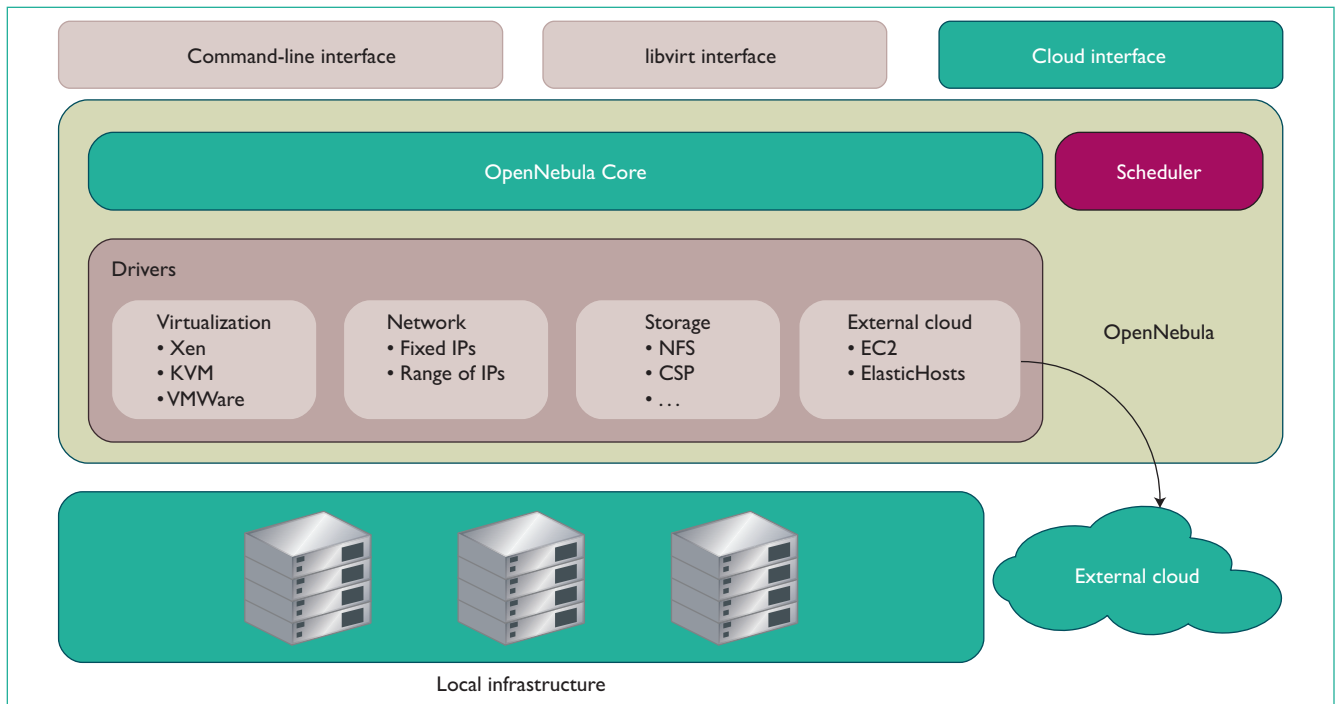


Figure 2. OpenNebula virtual infrastructure engine components. By using a driver-based architecture, OpenNebula can be integrated with multiple virtual machine managers, transfer managers, and external cloud providers.

A key feature of OpenNebula’s architecture, which we describe more in the next section, is its highly modular design, which facilitates integration with any virtualization platform and third-party component in the cloud ecosystem, such as cloud toolkits, virtual image managers, service managers, and VM schedulers. For example, it specifies all actions pertaining to setting up a VM disk image (transferring the image, installing software on it, and so on) in terms of well-defined hooks. Although OpenNebula includes a default “transfer manager” that uses these hooks, it can also leverage existing transfer managers or VM image contextualizers just by writing code that interfaces between the hooks and the third-party software.

The Haizea project – which we developed independently from OpenNebula – was the first to leverage such an architecture in a way that was beneficial to both projects. Haizea originally could simulate VM scheduling only for research purposes, but we modified it to act as a drop-in replacement for OpenNebula’s default scheduler, with few changes required in the Haizea code and none in the OpenNebula code. By working together, OpenNebula could offer resource leases as a fundamental provisioning abstraction, and Haizea could operate with real hardware through OpenNebula.

In fact, integrating OpenNebula and Haizea provides the only VI management solution offering advance reservation of capacity. As Table 1 shows, other VI managers use *immediate provisioning* or *best-effort provisioning*, which we discuss in more detail later. However, private clouds – specifically those with limited resources in which not all requests are satisfiable immediately owing to lack of resources – stand to benefit from more sophisticated VM placement strategies supporting queues, priorities, and ARs. Additionally, service provisioning clouds, such as the one being developed in the Reservoir project, have requirements that are insupportable with only an immediate provisioning model – for example, they need capacity reservations at specific times to meet service-level agreements (SLAs) or peak capacity requirements.

The OpenNebula Architecture

The OpenNebula architecture (see Figure 2) encompasses several components specialized in different aspects of VI management.

To control a VM’s life cycle, the OpenNebula *core* orchestrates three different management areas: image and storage technologies (that is, virtual appliance tools or distributed file systems) for preparing disk images for

VMs, the network fabric (such as Dynamic Host Configuration Protocol [DHCP] servers, firewalls, or switches) for providing VMs with a virtual network environment, and the underlying hypervisors for creating and controlling VMs. The core performs specific storage, network, or virtualization operations through pluggable *drivers*. Thus, OpenNebula isn't tied to any specific environment, providing a uniform management layer regardless of the underlying infrastructure.

Besides managing individual VMs' life cycle, we also designed the core to support *services* deployment; such services typically include a set of interrelated components (for example, a Web server and database back end) requiring several VMs. Thus, we can treat a group of related VMs as a first-class entity in OpenNebula. Besides managing the VMs as a unit, the core also handles the delivery of context information (such as the Web server's IP address, digital certificates, and software licenses) to the VMs.

A separate *scheduler* component makes VM placement decisions. More specifically, the scheduler has access to information on all requests OpenNebula receives and, based on these requests, keeps track of current and future allocations, creating and updating a resource schedule and sending the appropriate deployment commands to the OpenNebula core. The OpenNebula default scheduler provides a rank scheduling policy that places VMs on physical resources according to a ranking algorithm that the administrator can configure. It relies on real-time data from both the running VMs and available physical resources.

OpenNebula offers *management interfaces* to integrate the core's functionality within other data center management tools, such as accounting or monitoring frameworks. To this end, OpenNebula implements the libvirt API (<http://libvirt.org>), an open interface for VM management, as well as a command line interface (CLI). Additionally, a subset of this functionality is exposed to external users through a *cloud interface*.

Finally, OpenNebula can support a hybrid cloud model by using *cloud drivers* to interface with external clouds. This lets organizations supplement the local infrastructure with computing capacity from a public cloud to meet peak demands, better serve their access requests (for example, by moving the service

closer to the user), or implement high availability strategies. OpenNebula currently includes an EC2 driver, which can submit requests to Amazon EC2 and Eucalyptus, as well as an ElasticHosts driver.

The Haizea Lease Manager

Haizea is an open source resource lease manager and can act as a VM scheduler for OpenNebula or be used on its own as a simulator to evaluate different scheduling strategies' performance over time. The fundamental resource provisioning abstraction in Haizea is the *lease*. Intuitively, a lease is a form of contract in which one party agrees to provide a set of resources (an apartment, a car, and so on) to another. When a user wants to request computational resources from Haizea, it does so in the form of a lease; the leases are then implemented as VMs managed by OpenNebula. The lease terms Haizea supports include hardware resources, software environments, and the period during which the hardware and software resources must be available. Currently, Haizea supports *AR leases* in



CISCO

Cisco Systems, Inc. is accepting resumes
for the following position in:

Boxborough, MA

**Customer Support Engineer
(Ref#: BOX7)**

Responsible for providing technical support regarding the company's proprietary systems and software to field engineers, technicians, product support and company customers who are diagnosing, troubleshooting, repairing and debugging complex electro/mechanical equipment, computer systems and/or complex software.

Please mail resumes with reference number to Cisco Systems, Inc., Attn: JSIW, 170 W. Tasman Drive, Mail Stop: SJC 5/1/4, San Jose, CA 95134. No phone calls please. Must be legally authorized to work in the U.S. without sponsorship. EOE.

www.cisco.com

which resources must be available at a specific time; *best-effort leases*, in which resources are provisioned as soon as possible, and requests are placed in a queue, if necessary; and *immediate leases*, in which resources are provisioned when requested or not at all.

To satisfy use cases where resources must be guaranteed to be available at certain times, various researchers have studied advance reservation of computational resources in the context of parallel computing.⁴⁻⁶ In the absence of suspension/resumption capabilities, this approach produces resource underutilization due to the need to vacate resources before an AR starts. By using VMs to implement leases, an organization can support ARs more efficiently^{1,7} through resource preemption, suspending the VMs of lower-priority leases before a reservation starts, resuming them after the reservation ends, and potentially migrating them to other available nodes or even other clouds. We can do this without having to make the applications inside the VM aware that they're going to be suspended, resumed, or even migrated. However, using VMs introduces *runtime overhead*, which poses additional scheduling challenges, as does the *preparation overhead* of deploying the VM disk images that the lease needs. Such overheads can noticeably affect performance if they aren't adequately managed.⁸ Haizea's approach is to separately schedule preparation overhead rather than assuming it should just be deducted from a user's allocation. However, this is complicated when Haizea must support multiple lease types with conflicting requirements that it has to reconcile – for example, the transfers for a lease starting at 2 p.m. could require delaying transfers for best-effort leases, resulting in longer wait times. Haizea uses several optimizations, such as reusing disk images across leases, to minimize preparation overhead's impact. Similarly, Haizea also schedules the runtime overhead of suspending, resuming, and migrating VMs.

Haizea bases its scheduling on a resource slot table that represents all the physical nodes it manages over time. It schedules best-effort leases using a first-come-first-serve queue with backfilling (a common optimization in queue-based systems), whereas AR leases use a greedy algorithm to select physical resources that minimize the number of preemptions. Although the resource selection algorithm is cur-

rently hardcoded, future versions will include a policy decision module to let developers specify their own resource selection policies (for instance, policies to prioritize leases based on user, group, project, and so on). This policy decision module will also specify the conditions under which Haizea should accept or reject a lease.

Experiences with OpenNebula and Haizea

Although OpenNebula and Haizea both originated in research projects, one of our goals is to produce production-quality releases that meet other communities' needs. In fact, we feel strongly about using a development model that, first and foremost, produces stable software suitable for production environments, which we can also use for our own research, incorporating the results into the next stable version. This lets us support VI users' requirements while incorporating novel techniques and solutions into our releases. OpenNebula has already seen several stable releases and has a growing user base through its inclusion in the popular Ubuntu GNU/Linux distribution (www.ubuntu.com), starting with Ubuntu 9.04 ("Jaunty Jackalope"). Our first-hand experiences with OpenNebula have mostly occurred in the EU Reservoir project, where we are enhancing OpenNebula to meet the requirements of several business use cases.⁹ In recent work, we've shown OpenNebula to be effective in managing clustered services, using it to deploy and manage the back-end nodes of a Sun Grid Engine compute cluster and an nginx Web server¹⁰ on both local resources and an external cloud.

Other integration efforts with OpenNebula are currently under way (see <http://opennebula.org/doku.php?id=ecosystem>), including an implementation of the libvirt interface and a VM consolidation scheduler designed to minimize energy consumption. The Reservoir project is also developing other tools around OpenNebula for service elasticity management, VM placement to meet SLA commitments, support for public cloud interfaces, and a VM scheduler (termed *policy engine* within the project) that adds support for policy-driven probabilistic admission control and dynamic placement optimization to satisfy site-level management policies. We've also experimented with integrating OpenNebula with Globus Nimbus.

Haizea is still in a “technology preview” stage, although we plan a first stable release later in 2009. In previous joint work with Kate Keahey (Argonne National Laboratory),¹ we used Haizea to simulate 72 30-day workloads in six different configurations, or 36 years of lease scheduling, producing experimental results showing that, when using workloads that combine best-effort and AR requests, a VM-based approach with suspend/resume can overcome the utilization problems typically associated with AR use. More specifically, when measuring the total time required to process all the requests in the workload, we found that a VM-based approach performed consistently better (up to 32.97 percent), despite the overhead required to use VMs. Our results also showed that we can minimize VMs’ preparation overhead – in the form of transferring VM disk images from a repository – using image transfer scheduling and caching strategies. In more recent work,^{11,12} we’ve used OpenNebula and Haizea together to perform experiments on a physical testbed and develop a resource model for predicting the runtime overhead of suspending/resuming VMs under various configurations. We found that, similar to scheduling preparation overhead, explicitly and separately scheduling suspensions and resumptions is necessary to avoid unnecessary delays in leases (for example, if a lease must be suspended to make way for a higher-priority one, such as an AR). Furthermore, we found that accurately estimating the time to suspend and resume leases depends on a variety of factors that we must take into account when scheduling them.

As interest in private and hybrid IaaS clouds grows, so will the need for a diverse ecosystem of tools and technologies that can be used as building blocks to create and manage them. Although some solutions have emerged across three broad categories – cloud, VI, and VM management – the challenge ahead will be integrating multiple components to create complete IaaS cloud-building solutions.


Private and hybrid clouds will also face the challenge of efficiently managing finite resources. However, existing VI managers rely on an immediate resource provisioning that implicitly assumes that capacity is practically

infinite. Although this is a fair assumption for large cloud providers, such as Amazon EC2, it’s not applicable to smaller providers where the likelihood of being overloaded is greater. To satisfy SLAs, requests for resources will inevitably have to be prioritized, queued, pre-reserved, deployed on external clouds, or even rejected, and VI management solutions with these capabilities will be required.

OpenNebula and Haizea address these two challenges. By relying on a flexible, open, and loosely coupled architecture, OpenNebula is designed from the outset to be easy to integrate with other components, such as the Haizea lease manager. When used together, OpenNebula and Haizea are the only VI management solution that provides leasing capabilities beyond immediate provisioning, including best-effort leases and advance reservation of capacity. □

Acknowledgments

We gratefully acknowledge the hard work of the OpenNebula developers, Javier Fontán and Tino Vázquez. We also thank our anonymous reviewers for their insightful



CISCO

Cisco Systems, Inc. is accepting resumes
for the following position in:

Herndon, VA

**Network Consulting
Engineer
(Ref#: HER4)**

Provide consultative, proactive and/or
reactive support to Company accounts.

Please mail resumes with reference number to Cisco
Systems, Inc., Attn: JSIW, 170 W. Tasman Drive, Mail
Stop: SJC 5/1/4, San Jose, CA 95134. No phone calls
please. Must be legally authorized to work in the
U.S. without sponsorship. EOE.

www.cisco.com

and detailed comments. Development of OpenNebula is supported by Consejería de Educación de la Comunidad de Madrid, Fondo Europeo de Desarrollo Regional (FEDER), and Fondo Social Europeo (FSE), through BIOGRIDNET Research Program S-0505/TIC/000101; the Ministerio de Educación y Ciencia through research grant TIN2006-02806; and the EU through Reservoir research grant number 215605. Haizea development is supported by Reservoir, the University of Chicago, and the US Department of Energy under contract DE-AC02-06CH11357. Early work on Haizea was done in collaboration with Kate Keahey (Argonne National Laboratory) and funded by US National Science Foundation grant 509408, "Virtual Playgrounds."

References

1. B. Sotomayor, K. Keahey, and I. Foster, "Combining Batch Execution and Leasing using Virtual Machines," *Proc. 17th Int'l Symp. High Performance Distributed Computing* (HPDC 08), ACM Press, 2008, pp. 87–96.
2. K. Keahey et al., "Virtual Workspaces: Achieving Quality of Service and Quality of Life on the Grid," *Scientific Programming*, vol. 13, no. 4, 2005, pp. 265–276.
3. D. Nurmi et al., "The Eucalyptus Open-Source Cloud-Computing System," *Cloud Computing and Applications 2008* (CCA 08), 2008; www.cca08.org/papers.php.
4. I. Foster et al., "A Distributed Resource Management Architecture that Supports Advance Reservations and Co-Allocation," *Proc. Int'l Workshop on Quality of Service*, IEEE Press, 1999, pp. 27–36.
5. W. Smith, I. Foster, and V. Taylor, "Scheduling with Advanced Reservations," *Proc. 14th Int'l Symp. Parallel and Distributed Processing* (IPDPS 00), IEEE CS Press, 2000, p. 127.
6. Q. Snell et al., "The Performance Impact of Advance Reservation Meta-Scheduling," *Proc. Workshop Job Scheduling Strategies for Parallel Processing* (IPDPS 00/JSSPP 00), Springer-Verlag, 2000, pp. 137–153.
7. B. Sotomayor et al., "Enabling Cost-Effective Resource Leases with Virtual Machines," *Hot Topics Session in ACM/IEEE Int'l Symp. High-Performance Distributed Computing 2007* (HPDC 07), 2007; http://workspace.globus.org/papers/HPDC2007_hottopic_SotomayorKeaheyFosterFreeman.pdf.
8. B. Sotomayor, K. Keahey, and I. Foster, "Overhead Matters: A Model for Virtual Resource Management," *Proc. 1st Int'l Workshop on Virtualization Technology in Distributed Computing* (VTDC 06), IEEE CS Press, 2006, p. 5.
9. B. Rochwerger et al., "The Reservoir Model and Architecture for Open Federated Cloud Computing," *IBM Systems J.*, Oct. 2008.
10. R. Moreno-Vozmediano, R. Montero, and I. Llorente, "Elastic Management of Cluster-Based Services in the Cloud," *Proc. 1st Workshop on Automated Control for Datacenters and Clouds* (ACDC 09), ACM Press, 2009, pp. 19–24.
11. B. Sotomayor et al., "Capacity Leasing in Cloud Systems using the OpenNebula Engine," *Proc. Cloud Computing and Applications 2008* (CCA 08), 2008; www.cca08.org/papers.php.
12. B. Sotomayor et al., "Resource Leasing and the Art of Suspending Virtual Machines," *Proc. 11th IEEE Int'l Conf. High-Performance Computing and Communications* (HPCC 09), IEEE Press, 2009, pp. 59–68.

Borja Sotomayor is a PhD candidate in the Department of Computer Science at the University of Chicago. His research interests include resource provisioning and scheduling, distributed systems, and virtualization. Sotomayor has an MSc in computer science from the University of Chicago, and a computer engineering degree from the University of Deusto, Bilbao, Spain. Contact him at borja@cs.uchicago.edu.

Rubén S. Montero is an associate professor in the Department of Computer Architecture at the Complutense University of Madrid. His research interests lie mainly in resource-provisioning models for distributed systems, in particular, grid resource management and scheduling, distributed management of virtual machines, and cloud computing. Montero has a PhD in physics from Complutense University. Contact him at rubensm@dacya.ucm.es.

Ignacio M. Llorente is a full professor and the head of the Distributed Systems Architecture Research group at the Complutense University of Madrid. His research interests include advanced distributed computing and virtualization technologies, architecture of large-scale distributed infrastructures, and resource-provisioning platforms. Llorente has a PhD in computer science from the Complutense University of Madrid and an Executive Master in business administration from the Instituto de Empresa Business School. Contact him at llorente@dacya.ucm.es.

Ian Foster is director of the Computation Institute at the University of Chicago and Argonne National Laboratory and the Arthur Holly Compton Distinguished Service Professor of computer science at the University of Chicago. His research interests include distributed computing, parallel computing, and computational science. Foster has a PhD in computer science from Imperial College, London. Contact him at foster@anl.gov.