Plug into the Supercloud

Dan Williams[†‡]
djwillia@us.ibm.com

Hani Jamjoom[†]
jamjoom@us.ibm.com

Hakim Weatherspoon[‡]
hweather@cs.cornell.edu

[†] IBM T. J. Watson Research Center, 1101 Kitchawan Road, Yorktown Heights, N.Y. 10598
[‡] Cornell University, 4130 Upson Hall, Ithaca, NY 14853

Abstract

Cloud computing is often compared to the power utility model as part of a trend towards the commoditization of computing resources. However, today's cloud providers do not simply supply raw computing resources as a commodity, but also act as distributors, dictating cloud services that are not compatible across providers. We propose a new cloud service distribution layer, called a *supercloud*, that is completely decoupled from the cloud provider. Leveraging a nested paravirtualization layer called *the Xen-Blanket*, the supercloud maintains the control necessary to implement hypervisor-level services and management. Using the Xen-Blanket to transform various cloud provider services into a unified offering, we have deployed a supercloud across Amazon's EC2, an enterprise cloud, and Cornell University, and performed live VM migration between the different sites. Furthermore, superclouds create opportunities to exploit resource management techniques that providers do not expose, like resource oversubscription, and ultimately can reduce costs for users.

Categories and Subject Descriptors D.4.0 Operating Systems [*General*]; D.4.7 Operating Systems [*Organization and Design*]: Distributed systems

Keywords Cloud Computing, Virtualization

1. Introduction

As part of the growing trend towards the commoditization of computing resources, cloud computing is often compared to other utility models, like electricity. Akin to power generators, cloud providers offer massive amounts of computing resources. However, unlike today's electricity utility model, in which consumers are generally agnostic to where power is generated, cloud users (consumers) are tightly coupled to the providers' infrastructures and must adhere to the varying specifications (e.g., virtualization stack and management APIs) of the corresponding cloud provider. As it stands, today's cloud delivery model resembles the "War of the Currents" of the late 1880s [8], where direct current (DC) power distribution was tightly coupled to a local generator.

We are interested in building cloud infrastructures that are akin to Westinghouse's "universal system"—the foundation of modern day power generation, distribution, and commoditization. In the "universal system," Westinghouse showed how power—using Tesla's transformer—can be generated and consumed in many different voltages [8]. Consumers, in such a model, only care about the availability of power as it can simply be transformed into the correct voltage. This way, consumers are able be completely agnostic to where and how the electricity is being generated. The "universal system," thus, demonstrated how to decouple power generation from consumption. This decoupling was a corner piece to creating power distribution networks and, ultimately, the commoditization of electricity.

To solve the tight coupling of today's clouds, this paper proposes a new way of building clouds—called *superclouds*. A supercloud is a cloud distribution layer that is not bound to any provider or physical resources. On the surface, users of a supercloud see a collection of computing resources, similar to today's clouds. Beneath the surface, the supercloud "transforms" multiple underlying cloud offerings into a universal cloud abstraction. The supercloud specifies and fully controls the entire cloud stack, independent from the providers' infrastructure. A supercloud, thus, decouples cloud providers and users.

At first glance, the idea of creating a supercloud might require a radical redesign of today's clouds. It does not. This paper demonstrates the creation of superclouds on top of existing clouds by leveraging nested virtualization as a technique to perform similar function to Tesla's transformer in Westinghouse's "universal system." Towards this goal, we present the design of nested *paravirtualization* hypervisor, called *the Xen-Blanket*. The Xen-Blanket can run on different providers, thus enabling superclouds to span clouds. Subsequently, a supercloud can manage guest virtual machines (VMs) across computing resources, irrespective of

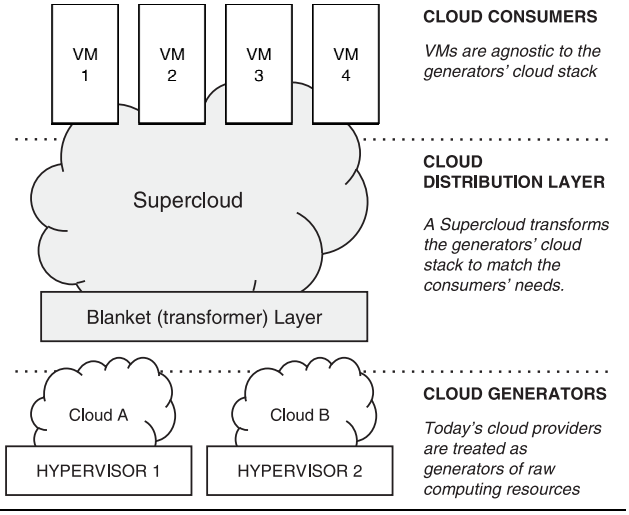

Figure 1. A supercloud creates a distribution layer between the cloud generator and consumer.

the provider's virtualization stack. For example, a supercloud can live-migrate VMs between cloud providers. The supercloud can run any cloud management stack, including OpenStack [1], Eucalyptus [7], or a completely custom stack.

We have built a supercloud that spans several diverse environments by deploying the Xen-Blanket in each. In particular, the Xen-Blanket is running on Xen-based and KVM-based hypervisors, on public and private infrastructures within Amazon Elastic Compute Cloud (EC2), an enterprise cloud, and Cornell University. Within the supercloud, we have migrated VMs to and from Amazon EC2 with no modifications to the VMs; the cloud user does not need to be aware of which provider is supplying the resources supporting the VM. Furthermore, our supercloud exploits a resource oversubscription model not offered by any cloud provider. As a direct result, our supercloud can host 40 CPU-intensive VMs on EC2 for 41% of the price per hour of 40 small instances with matching performance.

2. Supercloud

A supercloud is an abstraction that provides a uniform cloud service that is made up of resources obtained from a number of diverse Infrastructure as a Service (IaaS) [9] cloud resource providers (Figure 1). This section describes the role superclouds play in a utility model and the challenges faced when designing and implementing a supercloud.

2.1 A "Universal System" for the Cloud

Utility models for resources in a "universal system," such as electricity, consist of three distinct roles: generators, distributors, and consumers. At the back-end, *generators* offer resources. A *distributor* consumes the raw resources as a commodity from a variety of generators and builds a service around them. Then, *consumers* interact with the distributor service to utilize the resources.

The common model used for today's IaaS clouds only consists of two entities: a cloud provider and a cloud user. Viewing the cloud as a utility, the cloud provider (e.g., Amazon EC2, Rackspace, and Google Compute Engine, to name a few) acts as a generator by supplying computing resources in the form of virtual machines (VMs). The cloud user acts as a consumer by instantiating VMs to run application workloads.

In today's clouds, the distributor role is either assumed by the cloud provider or a third-party, but is limited in either case. On the one hand, cloud providers may act as both a generator and distributor. Cloud providers fully control the physical resources and can, therefore, implement rich features as services to cloud users. However, these features are intrinsically bound to a single provider, preventing distribution services that span providers, such as tolerance against failure of an entire cloud provider [2]. Some third-party vendors, such as RightScale [5], may act as distributors. By interacting with multiple cloud providers, they offer cloud service and management features. However, third-party vendors suffer from a lack of control over the cloud provider's platform and ultimately cannot add features like live VM migration, CPU bursting, or transparent VM fault tolerance.

Today's cloud model, thus, lacks a robust way of creating cloud distributors. These distributors should be able to consume resources from multiple cloud providers (generators) while maintaining control over the resources. Put another way, what is missing is a *distributor layer* that decouples cloud generators from cloud consumers. We call this layer a supercloud. A supercloud essentially drives the commoditization of compute resources by treating cloud providers as interchangeable generators from which raw resources are consumed. A supercloud can, thus, exploit pricing strategies and spot markets for compute resources between cloud providers, replicate VMs between cloud providers, and simplify management.

2.2 Challenges and Requirements

Today's IaaS cloud providers are diverse and heterogeneous, with services tightly coupled to the provider's physical resources. To fill the role of an independent distributor, a supercloud must contain a *transformer* or a mechanism that completely decouples resources from the physical infrastructure, including computation, network, and storage. As a first step, we focus on decoupling computation from the physical infrastructure. In today's IaaS clouds, there are two aspects in which computation (embodied by VMs) is tied to a cloud provider, necessitating a transformer.

First, VM images that run on one cloud provider cannot be easily instantiated on different clouds. For example, EC2 and Rackspace, two popular IaaS clouds, use different image formats: Amazon's AMI format and the Open Virtualization Format (OVF) [6], respectively. Paravirtualized de-

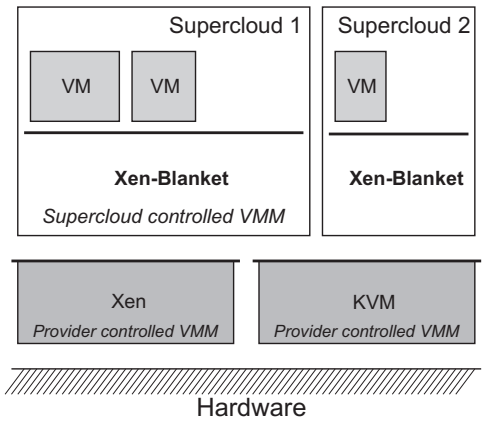

Figure 2. The Xen-Blanket, controlled by the supercloud, provides a distribution layer across heterogeneous clouds without requiring any additional support from the providers.

vice interfaces used by VMs are similarly diverse; Xen and `virtio` are used by VMs on EC2 and Rackspace, respectively. A supercloud must utilize computing resources from various cloud providers as a commodity; therefore, *a supercloud must decouple the VM image format from the provider*.

Second, IaaS clouds are diverse in terms of the services they provide to VMs. For example, Amazon EC2 provides tools such as CloudWatch (integrated monitoring), AutoScaling, and Elastic Load Balancing, whereas Rackspace contains support for VM migration to combat server host degradation and CPU bursting to borrow cycles from other instances. Moreover, a class of resource management opportunities—in particular, tools that operate at the hypervisor level—are not consistently available between providers. For example, there is no unified set of tools with which users can specify VM co-location on physical machines, page sharing between VMs, or resource oversubscription. *A supercloud must enhance cloud provider services with the set of features they may lack such that services can function seamlessly regardless of where the cloud resources are obtained from.*

The remainder of the paper describes an approach towards superclouds that enables a universal system for a cloud utility, by creating a distinct, feature-rich distributor that is completely separate from the generator. At its foundation, we describe a transformer that decouples the VM image format and cloud services from the provider.

3. Enabling a Supercloud

At its core, a supercloud leverages the *Xen-Blanket*, a nested virtualization system that can transform any provider-specific VM instance into a unified, distributor-specified VM instance. As depicted in Figure 2, nested virtualization consists of running a second-layer hypervisor inside a VM instance, which is running on top of a (first-layer) hypervisor. In this model, first-layer hypervisors continue to be owned by the provider (the generator), while the second-layer hypervisors are owned by a supercloud (the distributor). Also, in this model (and depicted in Figure 2), different superclouds (i.e., different distributors) can co-exist.

A *supercloud-user* (the consumer) runs VM instances and the cloud management stack on top of the second-layer Xen-Blanket hypervisor. We refer to the second virtualization layer as the *Blanket layer*. By running the Xen-Blanket on top of many different providers, the supercloud can consume resources from many different cloud generators, offer these aggregated resources to its consumers, and exploit the ability to seamlessly switch providers. The remainder of this section describes the design and implementation of the Xen-Blanket (more details appear in [14]), and a discussion of remaining challenges.

3.1 The Xen-Blanket Transformer

The Xen-Blanket consists of two concepts. First, the *top half* of the Xen-Blanket exposes a single—supercloud controlled—VM interface and service suite to supercloud users such that a guest VM image can run on any provider infrastructure without modifications. Second, the *bottom half* of the Xen-Blanket communicates with the underlying hypervisor interface, which could vary depending on the provider. No modifications are expected or required to the underlying hypervisor.

Transformer Top Half. The top half of the Xen-Blanket exposes a consistent VM interface to supercloud users. A supercloud can therefore place VMs on any provider that can run the Blanket layer without requiring any modifications to the VMs. In order to maximize the number of clouds that the Blanket layer can run on, the Blanket layer does not depend on the provider exposing state of the art nested virtualization interfaces (e.g., the Turtles Project [3]). The Blanket layer instead relies on other x86 virtualization techniques, such as paravirtualization or binary translation. For our prototype implementation, we chose to adopt the popular open-source Xen hypervisor, which uses paravirtualization techniques when virtualization hardware is not available. The Xen-Blanket subsequently inherits the limitations of paravirtualization, most notably the inability to run unmodified operating systems, such as Microsoft Windows. However, this limitation is not fundamental. A Blanket layer can be constructed using binary translation (e.g., a VMWare [12]-Blanket), upon which unmodified operating systems would be able to run. Blanket layers can also be created with other interfaces or even customized hypervisors developed from scratch.

Transformer Bottom Half. The bottom half of the Xen-Blanket ensures that a supercloud can span a number of different clouds without requiring changes to the underlying cloud system or hypervisor. We assume that hardware-assisted full virtualization for x86 (called HVM in Xen terminology) is available from cloud providers. However, we

do not assume that device I/O is emulated, so the Blanket hypervisor must be aware of the underlying hypervisor's paravirtualized I/O interfaces. The Xen-Blanket interfaces with a variety of underlying cloud paravirtualized device I/O implementations. Paravirtualized device I/O has proved essential for performance and is required by some clouds, such as Amazon EC2. However, there is currently no standard paravirtualized device I/O interface. For example, older Xen-based clouds, including Amazon EC2, require device drivers to communicate with Xen-specific subsystems, such as the XenBus and XenStore, whereas KVM-based systems expect device drivers to interact with the hypervisor through `virtio` interfaces. The Xen-Blanket supports such non-standard interfaces by modifying the bottom half to contain cloud-specific *Blanket drivers*.

3.2 Discussion

The Xen-Blanket transforms provider-specific VM image formats and hypervisor environments into a common distributer-defined format. However, for a supercloud to completely decouple cloud generators from cloud consumers, the supporting provider infrastructure must also be decoupled. In particular, a supercloud must provide a uniform mechanism to locate VMs on the network and interact with storage.

Cloud providers are beginning to offer virtual network abstractions which decouple network addresses from cloud infrastructure. For example, Amazon's Virtual Private Cloud (VPC) enables cloud users to extend their private network into the cloud. After eliminating technical addressing challenges, a supercloud may need to implement a policy to ensure that, even as VMs migrate between providers, heavily communicating VMs remain co-located on the same cloud provider [11].

By solving network addressing issues, storage can be accessed across the network from anywhere within the supercloud. However, accessing storage that resides on another cloud provider will result in poor performance. A supercloud may employ caching and replication techniques and balance performance with the cost of storing and transferring data between providers. A supercloud may even use RAID-like techniques across providers [2].

4. Evaluation and Experience

We have built a supercloud using the Xen-Blanket across three clouds, performing an extensive evaluation that appears in [14]. In short, the Xen-Blanket does introduce some overhead due to the second layer of virtualization. Microbenchmarks show the Xen-Blanket introduces 3% overhead for simple operations and up to 12.5% for context switch microbenchmarks. In the worst case, overheads can rise to 68%, due to APIC emulation for guest VMs with many VCPUs. However, Blanket drivers ensure that I/O performance is good: network drivers can receive packets at line speed on a 1 Gbps link, while disk I/O throughput is within 12% of single level paravirtualized disk performance. Furthermore, the performance of the Xen-Blanket is sufficient for common applications, like Web servers (demonstrated with the SPECweb2009 benchmark). Given the performance of the Xen-Blanket, the remainder of this section focuses on two supercloud features: resource oversubscription and cross-provider live VM migration.

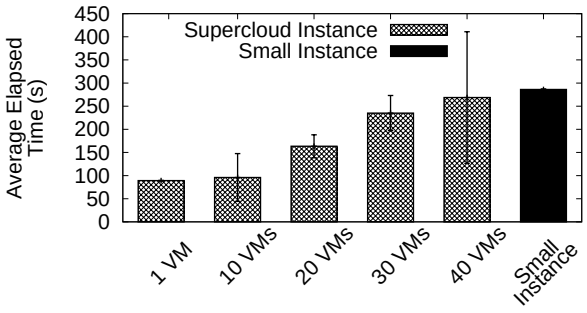

Figure 3. The Xen-Blanket gives the flexibility for a supercloud to oversubscribe cloud resources such that each of 40 VMs on a single 4XL instance can simultaneously complete compilation tasks in the same amount of time as a small instance.

4.1 Oversubscription in a Supercloud

We evaluate oversubscription in a Xen-Blanket based supercloud on Amazon EC2 and show that the supercloud can host CPU-intensive VMs at 41% the cost of native EC2. The supercloud exploits the pricing per hour on Amazon EC2 to rent a small instance versus a quadruple extra large cluster compute instance (cluster 4XL). In particular, as of July 2012, while the cluster 4XL instance is almost a factor of 16.5 times more expensive than a small instance ($0.08 vs. $1.30 per hour), some resources are greater than 16.5 times more abundant (e.g., 33.5 times more for CPU) while other resources are less than 16.5 times more abundant (e.g., 10.5 times more for disk). This suggests that if a cloud user has a number CPU intensive VMs normally serviced as small instances, it may be more cost efficient to rent a cluster 4XL instance and oversubscribe the memory and disk. This is not an option provided by Amazon. However, a supercloud (using the Xen-Blanket) can implement such a configuration.

To illustrate this point, we ran a CPU-intensive macrobenchmark, `kernbench`, simultaneously in a various numbers of VMs running on a supercloud made up of a single cluster 4XL instance running the Xen-Blanket. We also ran the benchmark inside a small EC2 instance for a comparison point. Figure 3 shows the elapsed time to run the benchmark in each of these scenarios. Each number of VMs on the supercloud corresponds to a different monetary cost. For example, to run a single VM, the cost is $1.30 per hour, while running 40 VMs reduces the cost per VM to $0.0325 per hour. Running a single VM, the benchmark completes

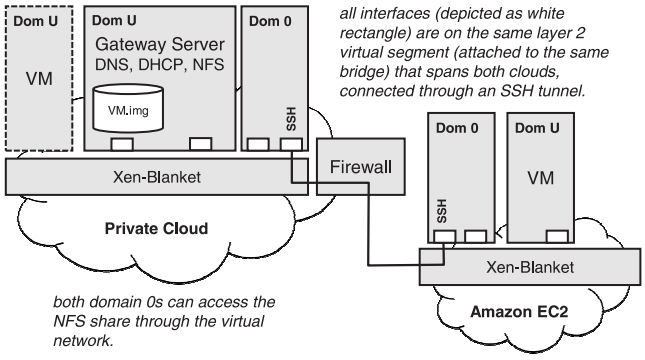

Figure 4. Xen-Blanket instances are connected with a layer-2 tunnel, while a gateway server VM provides DNS, DHCP and NFS to the virtual network, eliminating the communication and storage barriers to multi-cloud live migration.

in 89 seconds on the supercloud, compared to 286 seconds for a small instance. This is expected, because the cluster 4XL instance is significantly more powerful than a small instance. Furthermore, the average benchmark completion time for even 40 VMs remains 33 seconds faster than for a small instance; although the variance of the benchmark performance significantly increases for large numbers of VMs on the same instance. Since a small instance costs $.08 per VM per hour, this translates to about 41% of the price per VM per hour.

4.2 Cross-Provider Live Migration in a Supercloud

While it is possible to migrate VMs between multiple clouds today, the process is cloud-specific and fundamentally limited. To the best of our knowledge, our supercloud, using the Xen-Blanket, is the first implementation that can perform live VM migration between arbitrary cloud providers (including Amazon EC2). As such, we do not have a comparison point to present.

Live migration typically relies on memory tracing: a hypervisor-level technique that the Xen-Blanket inherits from Xen. Additionally, the supercloud must provide a uniform network and storage environment (as discussed in Section 3.2) to perform live multi-cloud migration.

Thus, within the Xen-Blanket, we implement a proof-of-concept virtual network and shared storage abstraction. Each Xen-Blanket instance within the supercloud runs a virtual switch in Domain 0 to which the virtual network interfaces belonging to Blanket guest VMs are attached. A layer-2 tunnel connects the virtual switches across the Internet. The result is that VMs on either of the two Xen-Blanket instances appear to be sharing a private LAN. A few basic network services are useful to introduce onto the virtual network. A gateway server VM can be run with two virtual network interfaces: one attached to the virtual switch and the virtual network, and the other attached to the externally visible interface of the Xen-Blanket instance. The gateway server VM, shown in Figure 4, runs `dnsmasq` as a lightweight DHCP and DNS server. It also runs an NFS server. The NFS server exports files onto the virtual network and is mounted by the Domain 0 of each Xen-Blanket instance. Both Xen-Blanket instances mount the NFS share at the same location. Therefore, during VM migration, the VM root filesystem image can always be located at the same filesystem location, regardless of the physical machine.

With VMs anywhere in the supercloud able to communicate, maintain their network addresses, and access storage within either cloud, live VM migration proceeds by following the typical procedure in the Blanket hypervisor. However, while we have successfully live-migrated a VM from an enterprise cloud to Amazon EC2 and back, it is clearly inefficient to rely on an NFS disk image potentially residing on another cloud instead of a local disk. Moreover, the layer-2 tunnel only connects two machines. As future work, the supercloud can be augmented with more sophisticated network virtualization, storage, and wide-area live migration techniques.

5. Related Work

There are several techniques that exist today to deploy applications on multiple clouds. While these techniques are positive steps towards creating Superclouds, none afford the user the flexibility or level of decoupling of the Xen-Blanket on today's public clouds.

Using tools from RightScale [5], a user can create ServerTemplates, which can be deployed on a variety of clouds and utilize unique features of clouds without sacrificing portability. However, RightScale cannot perform hypervisor-level services across providers, such as live VM migration. The RESERVOIR project [10] is a multi-cloud agenda to federate two or more independent cloud providers. However, standardization is necessary before federation can extend beyond the testbed. Like a Supercloud, *fos* [13] aims to expose a coherent environment that spans cloud resources and is deployed on EC2 today. However, fos exposes a single system image, forgoing the familiar VM interface and legacy applications contained within.

Through the Xen-Blanket, Superclouds leverage nested virtualization. The Turtles Project [3] enables nested virtualization with one or more levels of full virtualization on Intel hardware. Berghmans [4] describes the performance of several nested virtualization environments. VMworld Hands-On Labs (HOLs) [15] utilize VMware ESX in nested mode to produce a private lab environment for each participant. The Xen-Blanket sacrifices full nested virtualization for immediate deployment of a Supercloud on a variety of existing clouds.

6. Conclusion

We have presented supercloud, a new IaaS cloud distribution layer that is decoupled from the cloud provider. superclouds leverage the Xen-Blanket—a nested virtualization

approach—to act as a "Tesla transformer" for the cloud. Similarly to how the Tesla transformer enabled electricity to become commodity in the late 19th century, superclouds may enable cloud resources to become commodity. Through the Xen-Blanket, superclouds maintain the hypervisor-level control necessary to implement rich cloud services. Moreover, the Xen-Blanket requires no modifications to existing cloud provider infrastructures, enabling superclouds to be deployed today.

Using the Xen-Blanket, we have experimented developing supercloud services that oversubscribe resources to provide low-cost VMs for CPU intensive jobs and migrate VMs between cloud providers without requiring downtime or modifications to VMs. However, we have just scratched the surface in terms of the applications and functionality that can be implemented in a supercloud. In the future, superclouds will offer complete cloud management stacks, innovative or experimental cloud services, or custom features for a specific class of application, all spanning multiple providers.

The Xen-Blanket project website is located at `http://xcloud.cs.cornell.edu/`, and the code for the Xen-Blanket is publicly available at `http://code.google.com/p/xen-blanket/`.

References

[1] OpenStack. `http://www.openstack.org/`, Oct. 2010.

[2] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon. RACS: a case for cloud storage diversity. In *Proc. of ACM SoCC*, Indianapolis, IN, June 2010.

[3] M. Ben-Yehuda, M. D. Day, Z. Dubitzky, M. Factor, N. Har'El, A. Gordon, A. Liguori, O. Wasserman, and B.-A. Yassour. The turtles project: Design and implementation of nested virtualization. In *Proc. of USENIX OSDI*, Vancouver, BC, Canada, Oct. 2010.

[4] O. Berghmans. Nesting virtual machines in virtualization test frameworks. *Masters thesis, University of Antwerp*, May 2010.

[5] T. Clark. RightScale. http://www.rightscale.com, 2010.

[6] Distributed Management Task Force, Inc. (DMTF). Open virtualization format white paper version 1.00. `http://http://www.dmtf.org/sites/default/files/standards/documents/DSP2017_1.0.0.pdf`, Feb. 2009.

[7] Eucalyptus Systems, Inc. Eucalyptus open-source cloud computing infrastructure - an overview. `http://www.eucalyptus.com/pdf/whitepapers/Eucalyptus_Overview.pdf`, Aug. 2009.

[8] J. Jones. *Empires of Light: Edison, Tesla, Westinghouse, and the Race to Electrify the World*. Random House, 2004.

[9] P. Mell and T. Grance. The NIST definition of cloud computing. In *National Institute of Standards and Technology Special Publication 800-145*, Sept. 2011.

[10] B. Rochwerger, D. Breitgand, A. Epstein, D. Hadas, I. Loy, K. Nagin, J. Tordsson, C. Ragusa, M. Villari, S. Clayman, E. Levy, A. Maraschini, P. Massonet, H. Muñoz, and G. Tofetti. Reservoir - when one cloud is not enough. *IEEE Computer*, 44(3):44–51, 2011.

[11] V. Shrivastava, P. Zerfos, K. won Lee, H. Jamjoom, Y.-H. Liu, and S. Banerjee. Application-aware virtual machine migration in data centers. In *Proc. of IEEE INFOCOM Mini-conference*, Shanghai, China, Apr. 2011.

[12] J. Sugerman, G. Venkitachalam, and B.-H. Lim. Virtualizing I/O devices on VMware workstation's hosted virtual machine monitor. In *Proc. of USENIX Annual Technical Conf.*, Boston, MA, June 2001.

[13] D. Wentzlaff, C. Gruenwald, III, N. Beckmann, K. Modzelewski, A. Belay, L. Youseff, J. Miller, and A. Agarwal. An operating system for multicore and clouds: mechanisms and implementation. In *Proc. of ACM SoCC*, Indianapolis, IN, June 2010.

[14] D. Williams, H. Jamjoom, and H. Weatherspoon. The Xen-Blanket: Virtualize Once, Run Everywhere. In *Proc. of ACM EuroSys*, Bern, Switzerland, Apr. 2012.

[15] A. Zimman, C. Roberts, and M. V. D. Walt. VMworld 2011 Hands-On Labs: Implementation and Workflow. *VMware Technical Journal*, 1(1):70–76, Apr. 2012.