

Intel® Cloud Builders Guide: Cloud Design and Deployment on Intel® Platforms

Nimbula Cloud Operating System and Nimbula Director



Intel® Xeon® Processor 5500 Series

Intel® Xeon® Processor 5600 Series

nimbula

AUDIENCE AND PURPOSE

The cloud computing paradigm is spreading rapidly as organizations of all sizes realize the benefits of on-demand, self-service compute capacity, improved efficiencies, higher resource utilization and rapid elasticity to scale up and down. By migrating new and legacy applications to use cloud computing services, organizations have become more agile and reduced their time to market while shrinking capital and operating expenses.

The Intel® Cloud Builders program provides reference architectures in the form of basic hardware blueprints with Intel®-based servers running cloud computing software such as Nimbula Director*. This paper describes a basic hardware architecture on which we build a cloud environment using the Nimbula Director software and is not intended to be used "as is." The use cases in this paper can reduce the learning curve while provisioning, administering, and maintaining your first cloud and should be used as a foundation on which more complex, scalable, and resilient cloud computing architectures, adapted to your specific needs, can be built.

The audience for this Intel Cloud Builders guide includes:

Enterprise IT organizations looking to transform their internal data center infrastructure into a cloud environment and make controlled use of external clouds, with the goal of offering better services to their internal customers or users.

Service providers and systems integrators looking to transform the cost structures and agility of their client offerings by providing cloud computing services and solutions that accelerate revenue potential by reducing time to market and stimulating innovation.

Table of Contents

Executive Summary	3
Introduction	3
Nimbula Director Implementation Overview	3
Key Features and Benefits	4
Security	4
Federation	4
Multi-tenancy	4
No Single Point of Failure	4
Advanced Networking	4
Active Directory/LDAP	4
Sophisticated Placement	4
Ease of Use	4
Easy Set-up and Management	4
Compute Control	5
Network Control	5
Customers, Users and Groups	6
Test Bed Blueprint Overview	6
Hardware Description	7
Network Configuration	7
Use Case - Execution Methodology	8
Use Case - Context	13
Use Case - Overview	13
Use Case One: Customer Account and Organizational Hierarchy setup	14
Use Case Two: Provisioning and Administration of business services in the cloud	20
Use Case Three: Scaling Business Services Vertically	24
Use Case Four: Scaling Business Services Horizontally	28
Use Case Five: Cloud Federation to enable capacity bursting	30
Things to Consider	34
Hardware Considerations	34
Fail-over	34
Storage Architecture and Solid-State Drives	34
Networking	34
Security and Trust	34
Glossary	35
References	35
Additional Information	35
Disclaimers	36

Executive Summary

Implementing real-world cloud computing solutions is fraught with challenges and trade-offs. Companies and IT architects have to make decisions concerning capability, performance, security, capacity, deployment models (private, public, hybrid, or community clouds) and service models (Software-, Platform- or Infrastructure-as-a-Service). Concerns around data security, compliance, resource availability, service latency, customization and control further raise the barrier to entry for many organizations considering migrating their business into the public cloud.

Nimbula Director allows the creation of an on-premise cloud (commonly referred to as a private cloud) behind an organization's firewall, utilizing existing infrastructure and providing the benefits of cloud computing while retaining the security, control and trust of the private data center. Nimbula Director also allows customers to utilize public clouds in a controlled fashion, increasing the flexibility and agility of the entire system. Nimbula and Intel have worked together to prototype a private cloud computing test bed running a private cloud on a cluster of 12 Intel® Xeon® processor-based servers. Using the Nimbula Director API, via the command line as well as the Web-based interface, we created accounts based on organizational hierarchies, assigned appropriate permissions to users and groups, monitored and managed resources, demonstrated the elastic nature of resource scaling, and configured the use of cloud federation to enable bursting into external clouds during times of high demand.

Introduction

Based on Nimbula's Cloud Operating System technology, Nimbula Director allows customers to efficiently manage both on- and off-premise resources by transforming under-utilized private data centers into muscular, easily configurable compute capacity and providing controlled access to external clouds. Nimbula Director abstracts the underlying technology to present a coherent view of a completely automated on-premise compute cloud. Providing a one-stop virtual data center management solution, it isolates customers from the operational and hardware complexity associated with deploying and managing compute resources in a private data center.

Nimbula Director Implementation Overview

The Nimbula Cloud Operating System is an automated cloud management system that delivers Amazon* EC2-like services behind the firewall. Nimbula's technology allows customers to easily repurpose their existing infrastructure and build a computing cloud in the trusted environment of their own data center. With access to both on- and off-premise cloud services available via a common API, Nimbula Director, Nimbula's flagship product, combines the benefits of capitalizing on internal resource capacity and controlled access to additional external compute capacity.

Key Features and Benefits

- **Security:** Flexible and powerful group-based Authorization Service enables fine-grained permissions management based on policy and supports advanced access control of multiple users and groups.
- **Federation:** The Federation Service supports request forwarding to external services, such as public clouds, subject to Nimbula's fine-grained permissions management. This authorization filter facilitates powerful control and access management via a uniform API interface to both local and remote private and public clouds while abstracting (and protecting) the credentials used to authenticate against the public cloud.
- **Multi-tenancy:** The Authorization and Identity Service allows multiple customers, groups, and users to co-exist in isolation from each other or share resources on a single site.
- **No single point of failure:** Robust fail over mechanisms, including monitoring of services and nodes, and automated service replacement, ensure system reliability and resilience.
- **Advanced Networking:** Besides supporting flat standard IP allocation, Nimbula Director allows customers to create and declare their own virtual Ethernets. This enables the launching of instances in multiple isolated layer 2 networks where customers may provide their own DHCP server and other layer 2 services, such as multicast, broadcast, and non-IP Ethernet protocol.

In addition, Nimbula Director supports Network Security Lists which provide flexible role-based firewalling. Network Security Lists provide enhanced security and enable scalability by overcoming the limitations imposed by IP-based network security controls.

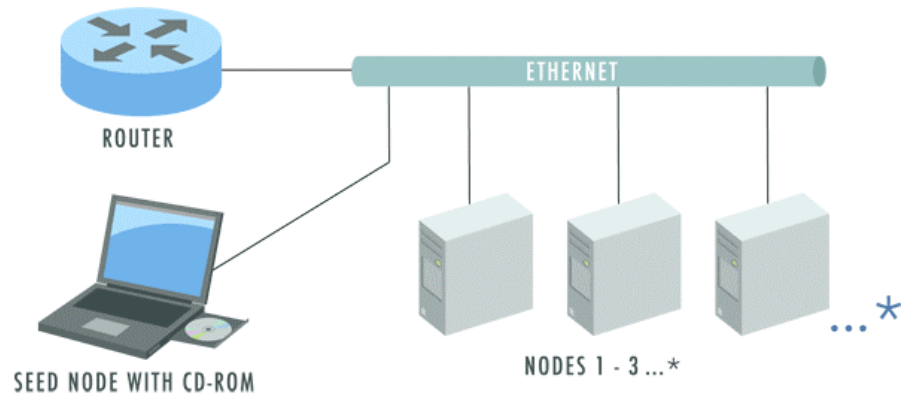


Figure 1: Nimbula Director basic topology

- **Active Directory/LDAP:** Nimbula's Authentication Service supports Active Directory/ LDAP, facilitating hassle free integration and user management and the efficient reuse of existing corporate user databases.
- **Sophisticated placement:** Instance placement can be specified with respect to the type of machine on which the instance should be run and the proximity to other instances, including whether instances should run on the same or a different node or cluster.
- **Ease of Use:** Quickly install up to thousands of servers with a hands-off automated installation on bare metal, with minimum configuration; Reduce demands on system administrators through low-touch automated cloud management; Use the simple and robust API to manage local and external cloud resources.
- **Easy set-up and management:** The Nimbula Director installation process is fully automated through a Live CD (a bootable CD which runs a full system on the CD, and runs independently from the operating system installed on the host machine). The Nimbula Director Live CD contains a machine image of an environment very similar to that which each Nimbula Director site node will run and is used to start the

Nimbula Director network installation process. Once the image on the Live CD has booted, DHCP, TFTP and HTTP services start, which allow other nodes to netboot using PXE, an automated network installer. The machine used to run the Live CD is referred to as the seed node and is typically not a cluster node, but it simply used to "seed" the cluster. This seed node can be any machine connected to the network including, for example, a laptop or by booting from a CD on one of the cluster nodes. Once it has installed the first cluster node, the seed node is used to monitor the installation and installation of other nodes proceeds from the first cluster node (see Figure 1).

New hardware added to the site is plugged in and switched on, and is automatically discovered and installed by the system. No manual intervention or configuration is required.

A RESTful HTTP API ^[1] provides a uniform and comprehensive interface to all aspects of cloud resource control, including access to both on-premise and external private and public clouds. Cloud resources can also be managed via a command line interface (CLI) and Web control panel, built on top of the RESTful HTTP API.

Beneath the virtual data center abstraction sits a physical layer of storage, network and compute hardware managed by multilayer control software. The Nimbula Director software integrates a hypervisor (virtualization layer) with node management software on each node to achieve automated deployment and configuration (see Figure 2).

Compute Control

A cluster is the compute backbone of Nimbula Director and consists of a number of x86 based servers, referred to as nodes, connected to a network. The number of machines in a cluster can be as few as three or as many as thousands. Multiple clusters define a site, which is the Nimbula cloud.

All nodes are controlled by the Infrastructure Controller (IC) which ensures that all services run correctly across the cluster at all times. The IC runs as a distributed service across all nodes, and enables the cluster to be self-healing and self-organizing. Any node in the system can function as the IC. In case the IC goes down, a new node becomes the IC through an automatic election process. This ensures cluster resilience and scalability, with no single points of failure.

Network Control

Nimbula Director facilitates the creation of highly dynamic virtual network topologies, independent of the underlying network topology. Users can dynamically create virtual Ethernets (vEthernets) using existing networking and associate these with instances. Full layer 2 functionality, including broadcast, multicast and non-IP traffic is supported. Network security groups can be used to enforce network policy between instances and externally to provide a distributed firewall.

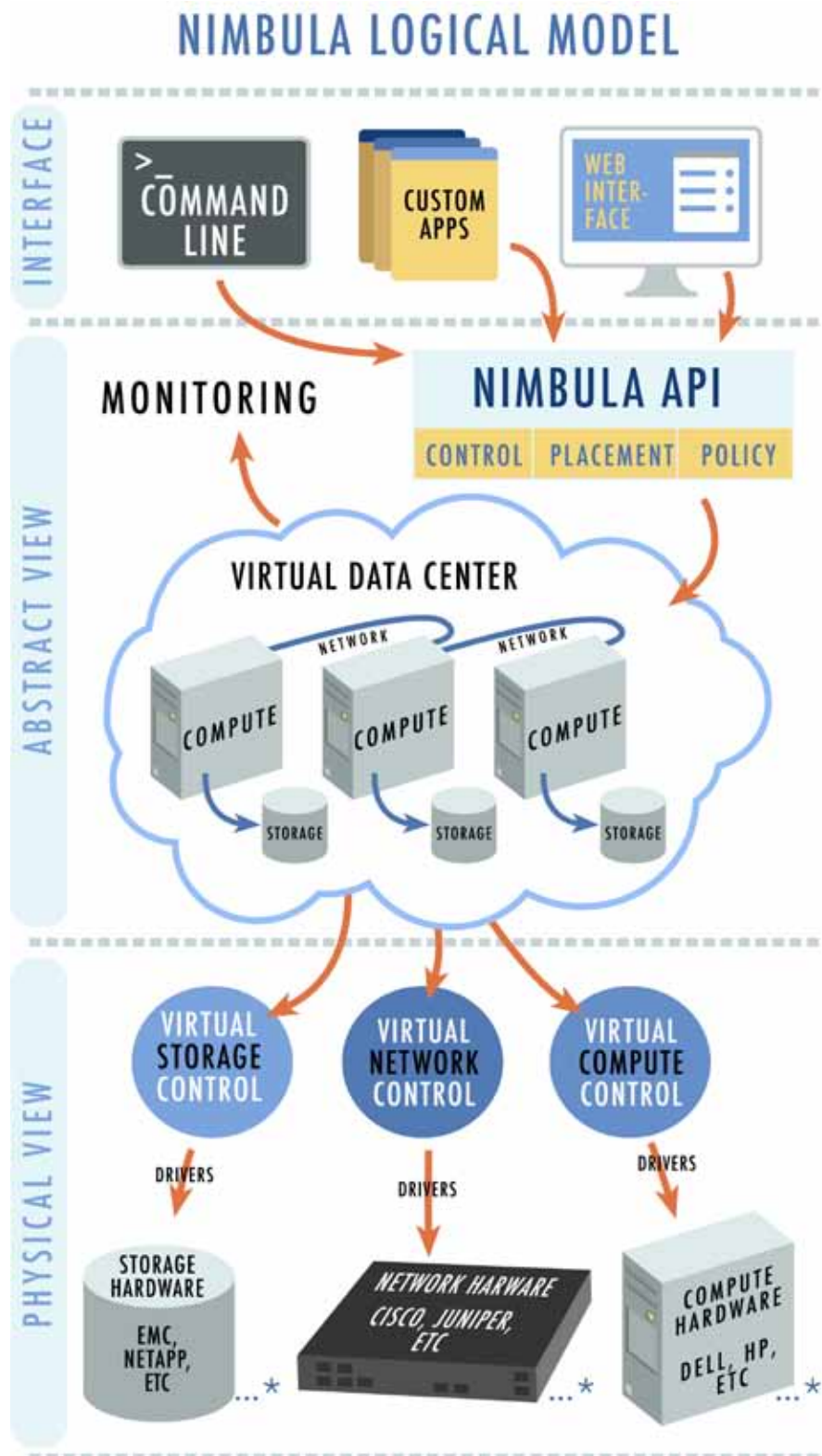


Figure 2: Nimbula Architecture Diagram

Customers, Users and Groups

Individual system users are classified into groups, which belong to customers.

The site ("the cloud") is controlled by a Site Administrator. This Site Administrator user, `/root/root` is created when Nimbula Director is installed. This user has permission to perform all system actions across the site. The password for this user is set in the `site.conf` configuration file before the Nimbula Director software is installed. The Site Administrator then creates customers, and each customer has its own administrator to manage its users and groups.

In Nimbula Director, the customer defines a billable unit. The customer is the context or framework within which system objects such as users, groups and machine images are created.

A **customer** can be an organization, department within an organization or an individual.

A **user** is an entity within the system that is able to make requests. Each user is associated with one customer. User names are unique within a specific customer, and customer names are unique within the system as a whole.

A **group** is a collection of users and subgroups. Organizational customers may choose to organize their users into groups by creating a group hierarchy, which maps to useful groupings in the organization, such as divisions or departments.

The principal reason to establish groups is to be able to easily define authorization policy across collections of users. The group structure provides the ability to set policy governing the allowable actions by users on system objects.

Authorization policy is defined by granting permissions to groups and users. When permissions are granted to a group, these permissions are inherited by all members of the group and its subgroups.

Test Bed Blueprint Overview

Most enterprise clouds use commodity off-the-shelf hardware components to host their cloud-based services. Typical components include dual-processor server platforms, network-based storage platforms to host data, and network components to manage internal and external traffic.

The test bed used to prototype the private cloud described in this white paper was implemented using homogenous commodity hardware and software components across the cloud infrastructure to reduce management and performance complexities. This consistency makes overall cloud maintenance and scalability more reliable—especially important as application workloads scale. The test bed components were deployed in a redundant configuration, providing resilience in case of failure of any infrastructure components.

We used a Cisco Catalyst 3750G Gigabit switch and local storage on all compute nodes.

System	Processor Configuration ^Δ	Configuration Details
Compute Nodes	Intel® Xeon® Processor X5570	Form Factor: 2U Rack Mount Server, Intel® Server Board S5500WBV Processor: Intel® Xeon® Processor 5500 series, 2.93GHz; 2-way x 4 cores = 8 cores Memory: 24GB RAM Storage: 136GB HDD Intel® Xeon® Processor 5000 series product information ... http://www.intel.com/xeon
Compute Nodes	Intel® Xeon® Processor L5630	Form Factor: 2U Rack Mount Server, Intel® Server Board S5520UR Processor: Intel® Xeon® Processor 5600 series, 2.13GHz; 2-way x 6 cores = 12 cores Memory: 24GB RAM Storage: 300GB HDD Intel Xeon Processor 5600 series information ... http://ark.intel.com/ProductCollection.aspx?series=47915 Intel® Xeon® Processor 5000 series product information ... http://www.intel.com/xeon

Figure 3: Hardware Configuration Details

Hardware Description

Our test bed uses Intel® Xeon® processor 5600 series-based servers (see Figure 3). Built on Intel's new 32nm micro-architecture, codenamed Westmere, the Intel Xeon processor 5600 series brings new levels of performance to the cloud with:

- Intelligent performance that automatically varies the processor frequency to meet business and application performance requirements.
- Automated energy efficiency that scales energy usage to the workload to achieve optimal performance/watt and reduce operating costs.
- Intel® Virtualization Technology (Intel® VT) ^[2] ♦ that offers best-in-class performance and manageability in virtualized environments to strengthen the infrastructure and reduce costs. Intel® Virtualization Technology for Connectivity (Intel VT-c) and Intel Virtualization Technology for Directed I/O (Intel VT-d) offer I/O virtualization features that can help end users improve security and reliability of the systems and also improve performance of I/O devices in virtualized environment.
- Intel® Trusted Execution Technology ^[3] that provides enhanced virtualization security through hardware-based resistance to malicious software attacks at launch.

Network Configuration

Figure 4 shows the network topology used in our test bed. Each compute node has two 1Gb network interfaces. The private compute node network is used by the various compute nodes to communicate with each other and to access shared datacenter resources like Storage Area Networks (SANs). The public interface is the link to the public Internet and is used for cloud administration functions like configuration and resource provisioning. These networks rely on vLAN technologies to provide isolation. Nimbula Director allows customers to create and declare their own virtual Ethernet, thus enabling network isolation at the virtual machine level.

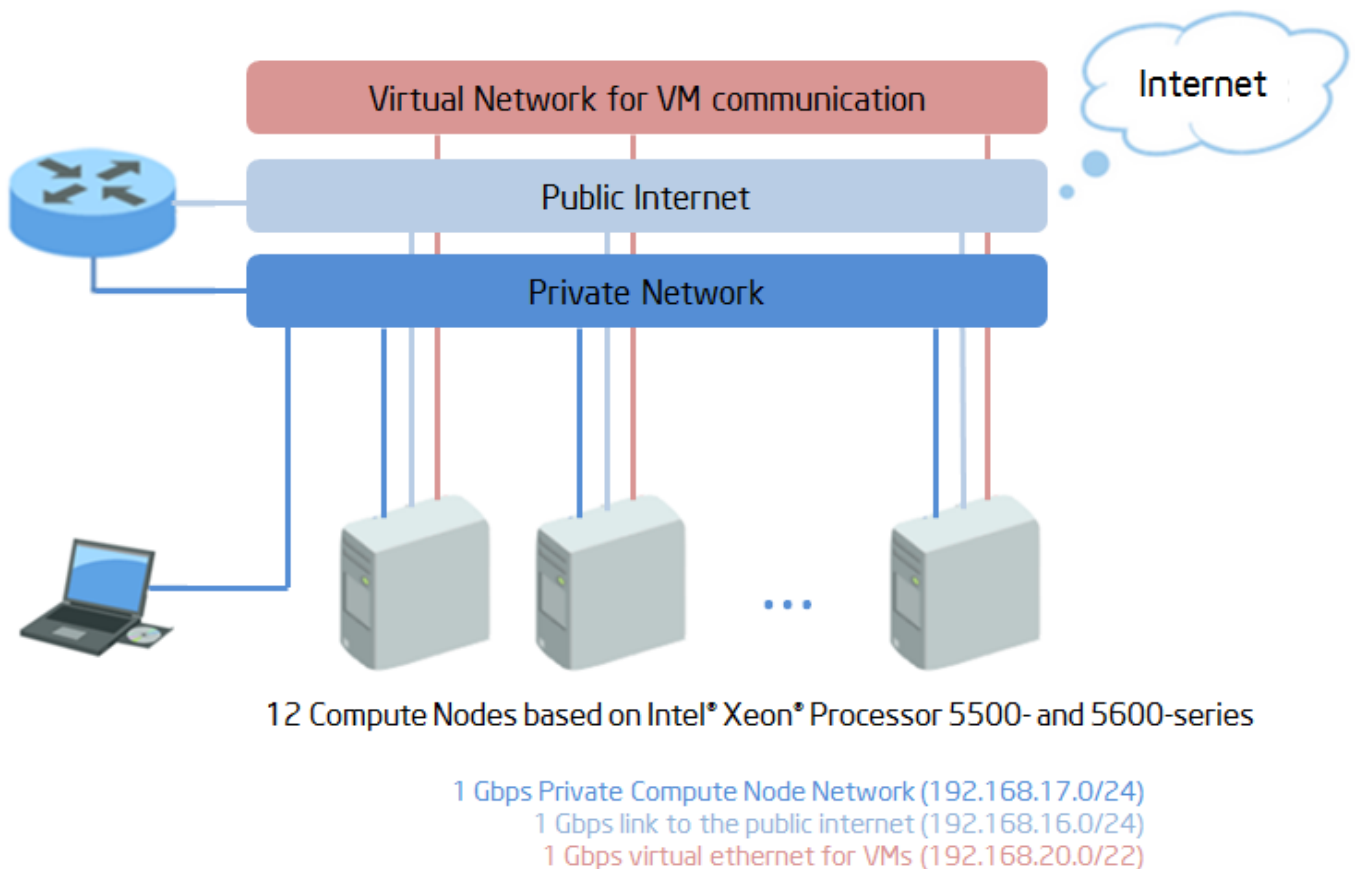


Figure 4: Network Topology Diagram

Use Case – Execution Methodology

To demonstrate the capabilities of our private cloud, we exercised five use cases, all of which are fairly typical for most private IaaS stacks.

An important pre-condition for running through the use cases is a fully configured Nimbula Director installation. The following section walks through the steps of installing Nimbula Director ^[4].

Step 1: Boot the seed node from the Live CD

To install Nimbula Director, we booted the Nimbula Live CD on the seed node running on our private network. Once the image on the Live CD has booted, DHCP, TFTP and HTTP services start which allow other nodes to netboot using PXE, an automated network installer. You will be presented with a Boot menu screen and should select Live for a regular installation (see figure 5).



Figure 5: Nimbula Director installation from a Live CD.

Step 2: Site configuration

Nimbula Director configuration information is specified in a configuration file, `/etc/nimbula/site.conf`, which consists of name-value pairs and nested sections in ConfigObj notation. The Nimbula Live Installer prompts you to configure the site by editing the `site.conf` file. Before cluster nodes can be installed, it is necessary to configure the site and start the Nimbula Infrastructure Controller. The Nimbula Live Installer prompts you to configure the site and instructs you to run the following command as root: `sudo nano /etc/nimbula/site.conf`.

This command will open the configuration file `site.conf` where you can define various site parameters as shown in Figure 6 below. This is the only configuration needed throughout the installation process. The Nimbula Director Quick Start Guide and Nimbula Director Installation and Administration Guide can be consulted for more information on setting configuration information.

```
# site.conf file
# The name for the site
name = acme
# The administrator's name
admin = root
# The administrator's email address
adminemail = root@acme.com
# Root password for API access. You may place a bcrypt password hash,
# generated by invoking nimbula-password, or a plaintext password here.If
# you use a plaintext password, it will be encrypted before the config file
# is installed on any nodes.
root_password = nimbula
# Email address for the root user
root_email = root@acme.com
# Timezone which nodes will be configured to use
timezone = UTC
[network]
# A DNS server which this site can reach
dns = 8.8.8.8
[[clusters]]
# Each subsection here represents a subnet.Use the CIDR format
# network specification for the section name.
[[[ 192.168.128.0/24 ]]]
# One cluster must have primary = True
primary = True
# You may optionally override the dns, ntp_servers and dhcp mode
# from above
[[[[instances]]]]
# You must specify the network on which instances for each
# cluster are started on -- each cluster should have a
# different network
subnet = 192.168.20.0/22
[shapes]
# You may have as many instance shapes as you want defined below.
# The name of the shape
[[small]]
# The amount of RAM (in MB) assigned to each instance of this shape.
ram = 400
# The number of CPUs assigned to each instance of this shape.
cpus = 1
[[medium]]
ram = 800
```

Figure 6

```

    cpus = 2
[services]
  [[api]]
    # An optional IP for the API endpoint and web console to run on. Will
    # always be discoverable with DNS -- this provides an additional
    # mechanism.
    # .5 on the primary subnet is recommended.
    Ip = 192.168.128.5
    # Uncomment and modify if you prefer non-default NTP servers
    # (eg, on-site)
    [[ntp]]
    # Default configuration is to not use any upstream servers
    servers = ,
    # Remove the previous line and uncomment the following line to use
    # NTP servers on the Internet instead.
    # servers = 0.pool.ntp.org,1.pool.ntp.org,2.pool.ntp.org,3.pool.ntp.org
[federation]
  [[ec2proxy]]
    hostname = ec2proxy
    type = ec2
    # Each apiport in the federation should be different.
    # A port number starting at 5500 is recommended.
    apiport = 5500
  [[[regions]]]
    eu-west-1 = ec2.eu-west-1.amazonaws.com
    us-east-1 = ec2.us-east-1.amazonaws.com
    us-west-1 = ec2.us-west-1.amazonaws.com
    ap-southeast-1 = ec2.ap-southeast-1.amazonaws.com
  [[[shapes]]]
    #A mapping of shapes to ec2shapes
    small = m1.small
    medium = m1.large
    large = m1.xlarge
[users]
  # Users here are system users, created at node install time. The
  # nimbulaadmin user is an example and may be removed
  [[nimbulaadmin]]
  # You may use a plaintext or an encrypted password here. If you use
  # a plaintext password, it will be encrypted before installation
  # onto any nodes. An encrypted password should be an MD5 crypted
  # password as used in the shadow file of modern Unix-like systems.
  password = nimbula
  gecost = "Nimbula, Inc."
  # Users should be in the admin group to allow sudo access
  groups = admin,
  uid = 1000
  [[[keys]]]
  # Public parts of SSH keys which should be installed to the user's
  # authorized_keys2 file
  example = ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA49mISR0JH+1iM4+nTJdLms7RK5Wld8eScTIzT
h9pcszHa87TXOyWszg+iJYxHPMeAwqSRJU5xgUaz71yCmu09MMFLvrIMR6xO/9QVl0MIkGmcPTRwUxevolF9YEXw+tU6N-
tEz+UeevZ9L/92MnSAYT5InoO2kounregt4XrCEIACSmtZocEWHaTWX8EGDaMZg8rXFMYZ2grLnedDUuQh3GLCrBNAYMAW
lsaeR2aEw4Q7OQSB5GHQjb5Z/+44RrTqZlIOZnPcPd8UTBe2wybKloJk2ltHSKI6VtlqNHdprO3IW85HQKlQbHTcS8y0iI
Gw+WnuEZCESEx8akbCFUuQSQ== example@example.com

```

Step 3: Start the Nimbula Infrastructure Controller

Once the site configuration is complete, the Nimbula Infrastructure Controller must be started. This is done with the `start-nimbula` command as shown in Figure 7 below. As shown, the installation server displays the installation progress and prompts you to turn on the first three compute nodes in turn.

Once the first node is installed, it takes over the network installation role and the seed machine acts as an installation progress monitor. Nodes can be installed simply by connecting them to the network and powering them on. The nodes receive an IP address from the DHCP server and PXE boot an installer and are installed automatically, with installation time varying based on the speed of the CPU, network connectivity and hard drive(s) size.

Once the first three nodes are installed and running, the service stack will be initialized. Within a few minutes, access to the Nimbula API is available. This process takes a few minutes, as databases will be initialized during the initial startup. After the first three nodes are running, and the Nimbula service stack has started, the rest of the site can be installed. This is done by simply turning on nodes or racks connected to the Nimbula network as configured during site configuration. From this point the entire installation process is completely automated. No additional manual intervention is required. At the end of each node's install process, it will reboot, at which point it will boot from its local hard disk.

```

root@seed:~# start-nimbula
Checking root password configuration: config.
Checking network configuration:
  Checking router of primary subnet (192.168.128.1): OK
  Checking IP ranges: OK
Checking NTP configuration: OK
About to generate keys; this needs entropy from the system. Please
provide extra entropy by pressing random keys if it appears to hang
Creating DNS key:created.
Starting Nimbula Key Generator: config makekeys.
Starting Nimbula Certificate Generator: config makecerts.
Starting Nimbula Infrastructure Controller: config config twisted.
Waiting for infrastructure controller master on seed node... OK
Waiting for network installation server on seed node... OK
Waiting for synchronised time on the seed node... OK
Please power up the first machine (waiting for it to network boot)..OK
Waiting for first node to reboot and start infrastructure controller... OK
Waiting for first node to contact seed node's infrastructure controller... OK
Stopping infrastructure controller on this seed... OK
Waiting for first node's infrastructure controller to become master... OK
Waiting for network installation server on the first node... OK
Waiting for synchronised time on the first node... OK
Please power up the second node (waiting for it to network boot)..OK
Please power up the third node (waiting for it to network boot)..OK
Waiting for infrastructure controller on second node... OK
Waiting for infrastructure controller on third node... OK
Waiting for Nimbula API to become available (this may take some time)... OK

Congratulations, Nimbula Director is successfully installed!

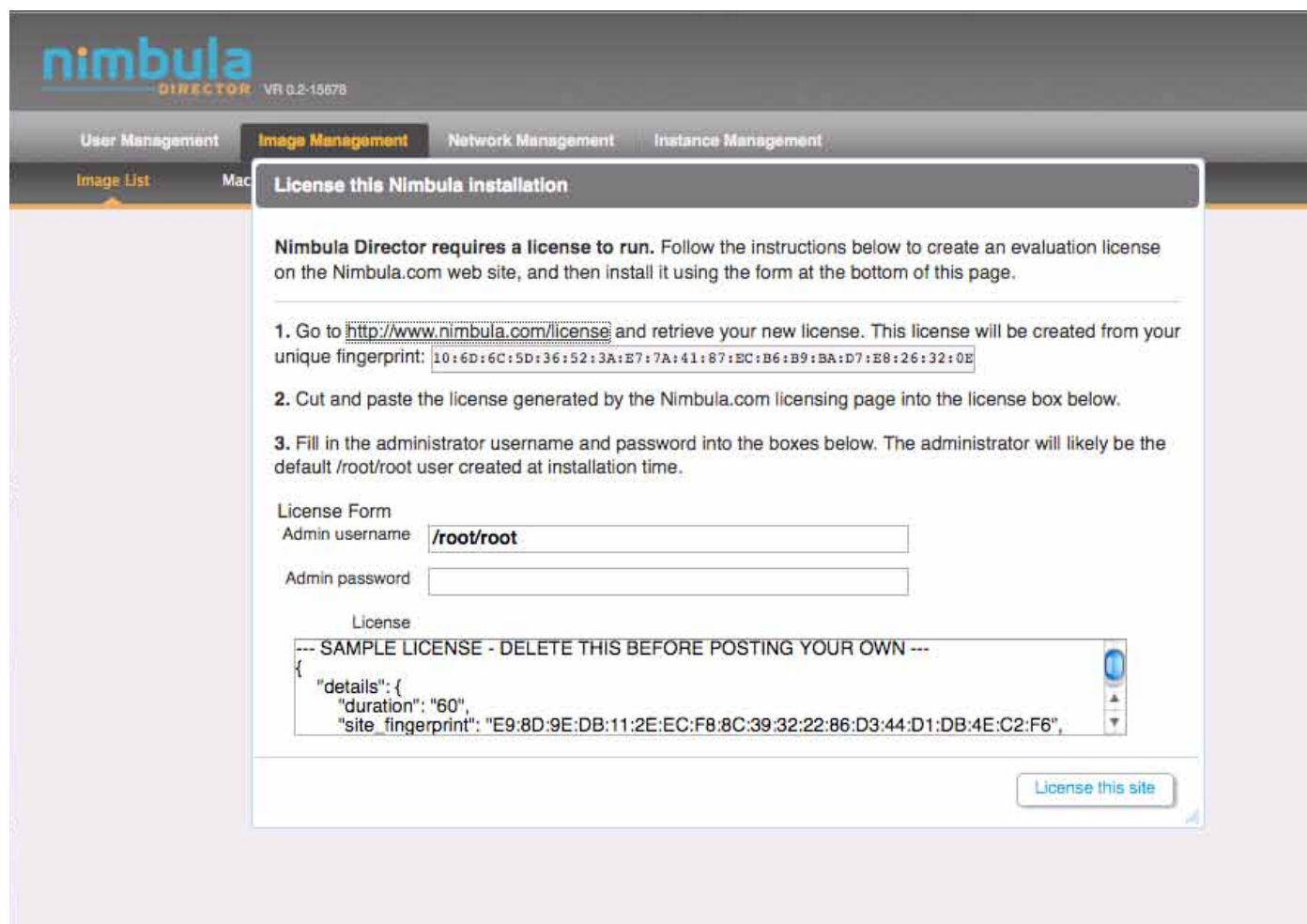
The API and web console are available at https://192.168.128.5/

```

Figure 7: Starting the Nimbula Infrastructure Controller.

Step 4: Activation

Before you can use the cloud, you need to activate your copy of Nimbula Director by following a registration process. When you first access the Nimbula Director Web-based interface, you will be presented with a licensing page (see Figure 8). Follow



The screenshot shows the Nimbula Director web interface. At the top, the logo "nimbula DIRECTOR" is displayed with the version "VR 0.2-15678". Below the logo is a navigation bar with tabs: "User Management", "Image Management" (which is highlighted), "Network Management", and "Instance Management". Under "Image Management", there is a sub-tab "Image List". A modal window titled "License this Nimbula Installation" is open in the center. The modal contains the following text: "Nimbula Director requires a license to run. Follow the instructions below to create an evaluation license on the Nimbula.com web site, and then install it using the form at the bottom of this page." Below this, there are three numbered instructions: 1. Go to <http://www.nimbula.com/license> and retrieve your new license. This license will be created from your unique fingerprint: 10:6D:6C:5D:36:52:3A:E7:7A:41:87:EC:B6:B9:BA:D7:E8:26:32:0E. 2. Cut and paste the license generated by the Nimbula.com licensing page into the license box below. 3. Fill in the administrator username and password into the boxes below. The administrator will likely be the default /root/root user created at installation time. Below the instructions, there is a "License Form" with two input fields: "Admin username" with the value "/root/root" and "Admin password" which is empty. Below the form is a "License" section with a text area containing a sample JSON license: {"details": {"duration": "60", "site_fingerprint": "E9:8D:9E:DB:11:2E:EC:F8:8C:39:32:22:86:D3:44:D1:DB:4E:C2:F6"}. The text area is titled "SAMPLE LICENSE - DELETE THIS BEFORE POSTING YOUR OWN". At the bottom right of the modal is a button labeled "License this site".

Figure 8: Nimbula License Registration.

Use Case – Context

Our use cases were performed on behalf of the fictitious Applied Computer Modeling Enterprises (ACME). At ACME, the ACME Administrator is in charge of the overall administration of the private cloud infrastructure, procurement of servers as well as provisioning, deploying, and managing virtual resources.

Two organizations (and two individuals representing these organizations) need access to the virtual infrastructure:

- The Information Technologies (IT) group is internal to ACME and represents a group of customers that will directly consume resources in the cloud. John Smith is an employee of the IT group and is an administrator of the IT cloud.
- Western Allied Service Provider (WASP) is an external organization that provides consulting services to ACME. Sally Jones is an employee of WASP who is authorized to work in ACME's cloud infrastructure.

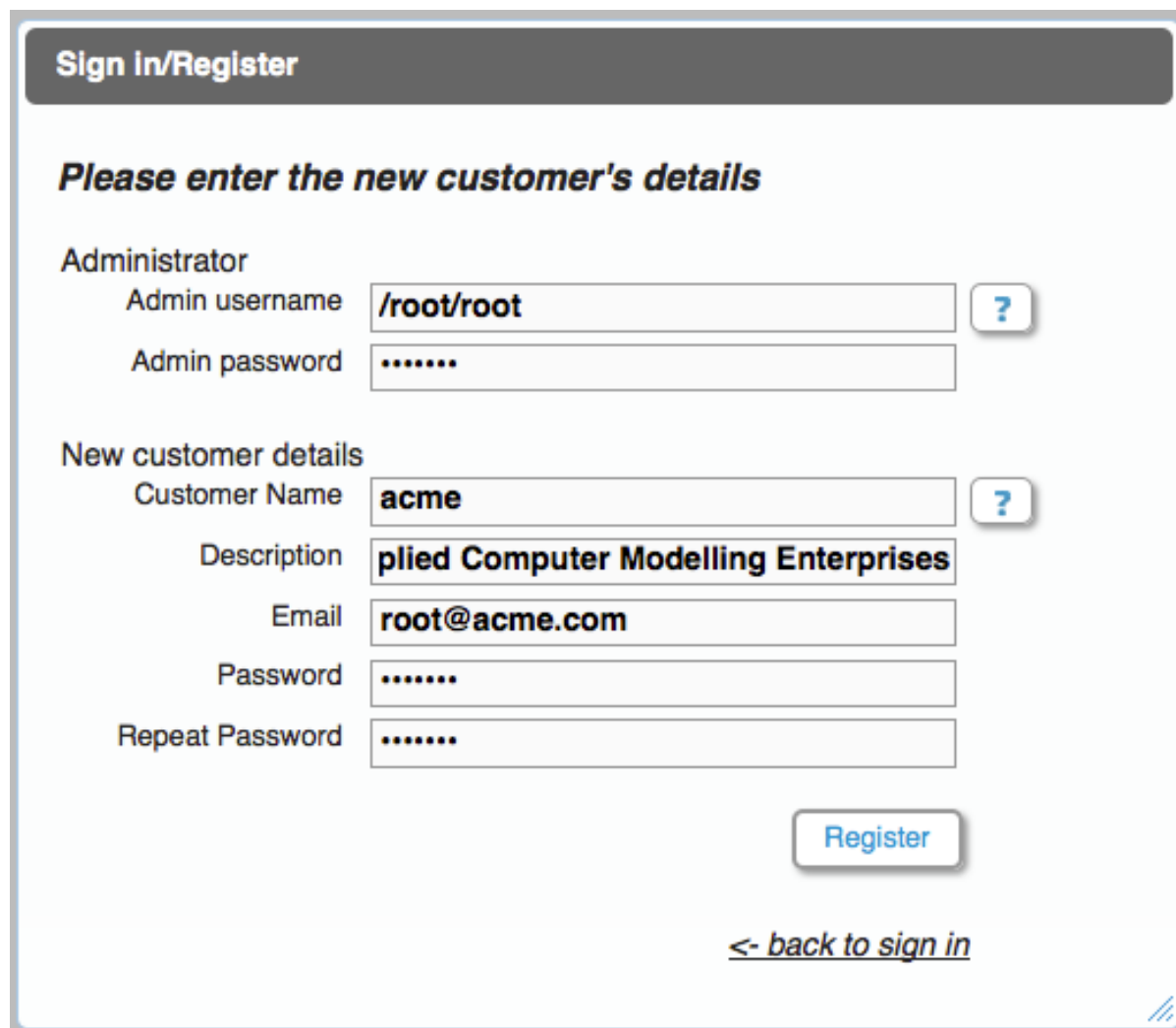
For our use cases, John utilizes the Nimbula Web-based interface and Sally utilizes the command-line interface. This allows us to demonstrate the usability of both interfaces.

Use Case – Overview

1. The Site Administrator (`/root/root`, aka "root") creates customer accounts for ACME and WASP, and the necessary business groups and users with the appropriate permissions.
2. John and Sally provision and administer business services in the private cloud.
3. In response to moderate demand, ACME Administrator scales the service vertically by adding more hardware resources to the service.
4. In response to continued heavy demand, John then scales the service horizontally.
5. ACME Administrator configures Federation to enable burst capability for workloads that can be run in external clouds.

Use Case One: The Site Administrator creates customer accounts for ACME and WASP, and business groups and users with the appropriate permissions

1. The Site Administrator (`root`) creates customer accounts for ACME and WASP using the Nimbula API Web interface. The Sign-in/Register page (see Figure 9) appears when you use the Nimbula Web-based interface after registration. To create a customer, click on the **Create a new customer** link. You will need to authenticate as the site administrator (`root`) with the password used in the `site.conf` configuration file.



The screenshot shows a web interface titled "Sign in/Register". Below the title is a heading "Please enter the new customer's details". The form is divided into two main sections: "Administrator" and "New customer details".

Administrator section:

- Admin username: (with a help icon)
- Admin password:

New customer details section:

- Customer Name: (with a help icon)
- Description:
- Email:
- Password:
- Repeat Password:

At the bottom right of the form is a "Register" button. Below the button is a link: [<- back to sign in](#).

Figure 9: The Site Administrator creates the ACME customer.

2. The ACME administrator creates the user John Smith under the ACME customer.

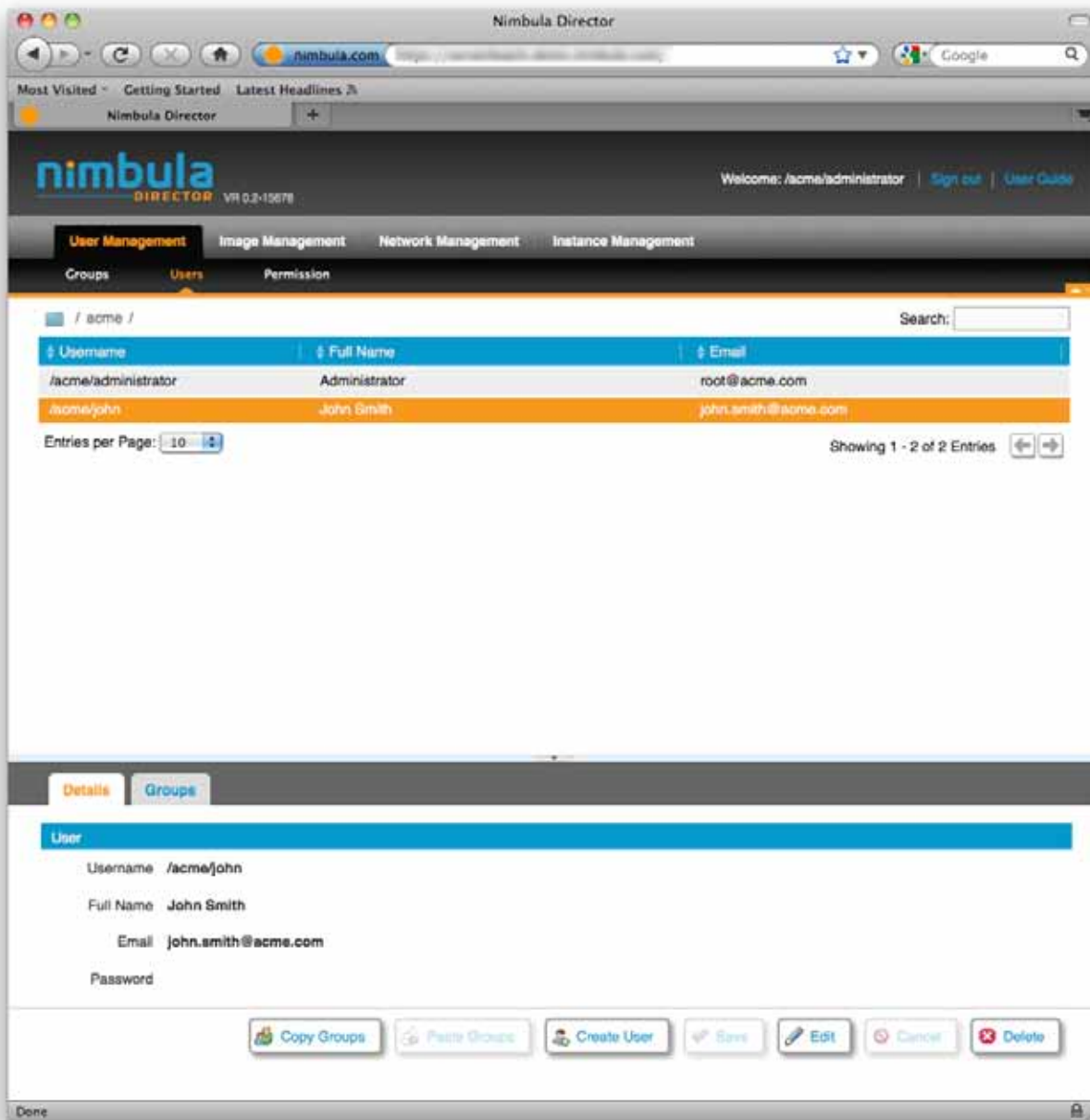


Figure 10: New user John is added to the system.

3. ACME Administrator creates the ACME IT Department using the Nimbula API Web interface. The Web interface makes it easy to create organizational hierarchies and assign appropriate permissions.

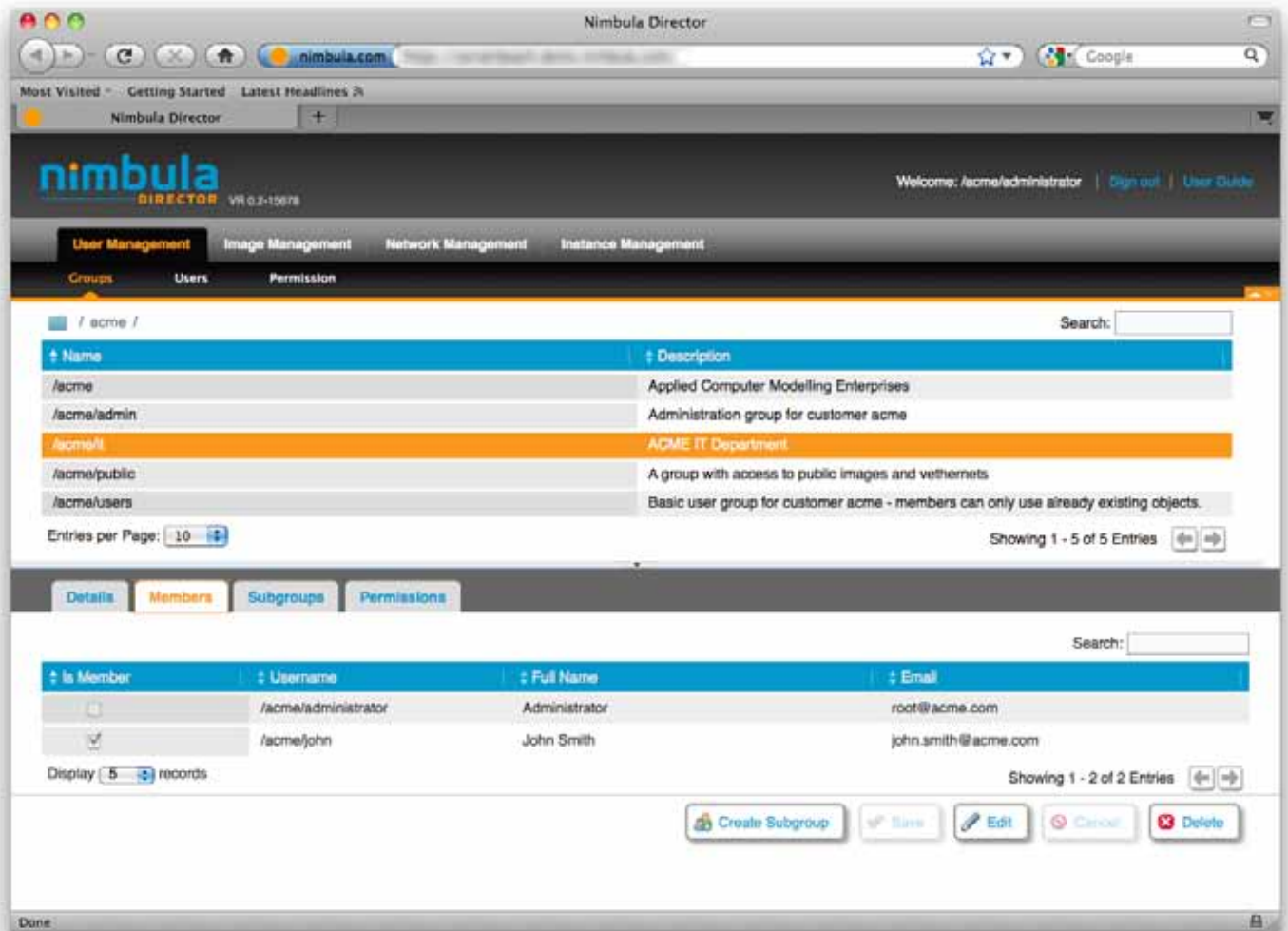


Figure 11: The ACME Administrator views the status of the newly created IT group.

4. ACME Administrator then adds John Smith as a member of the ACME IT group by going to the Members Tab and checking the “Is Member” box next to the user’s entry.

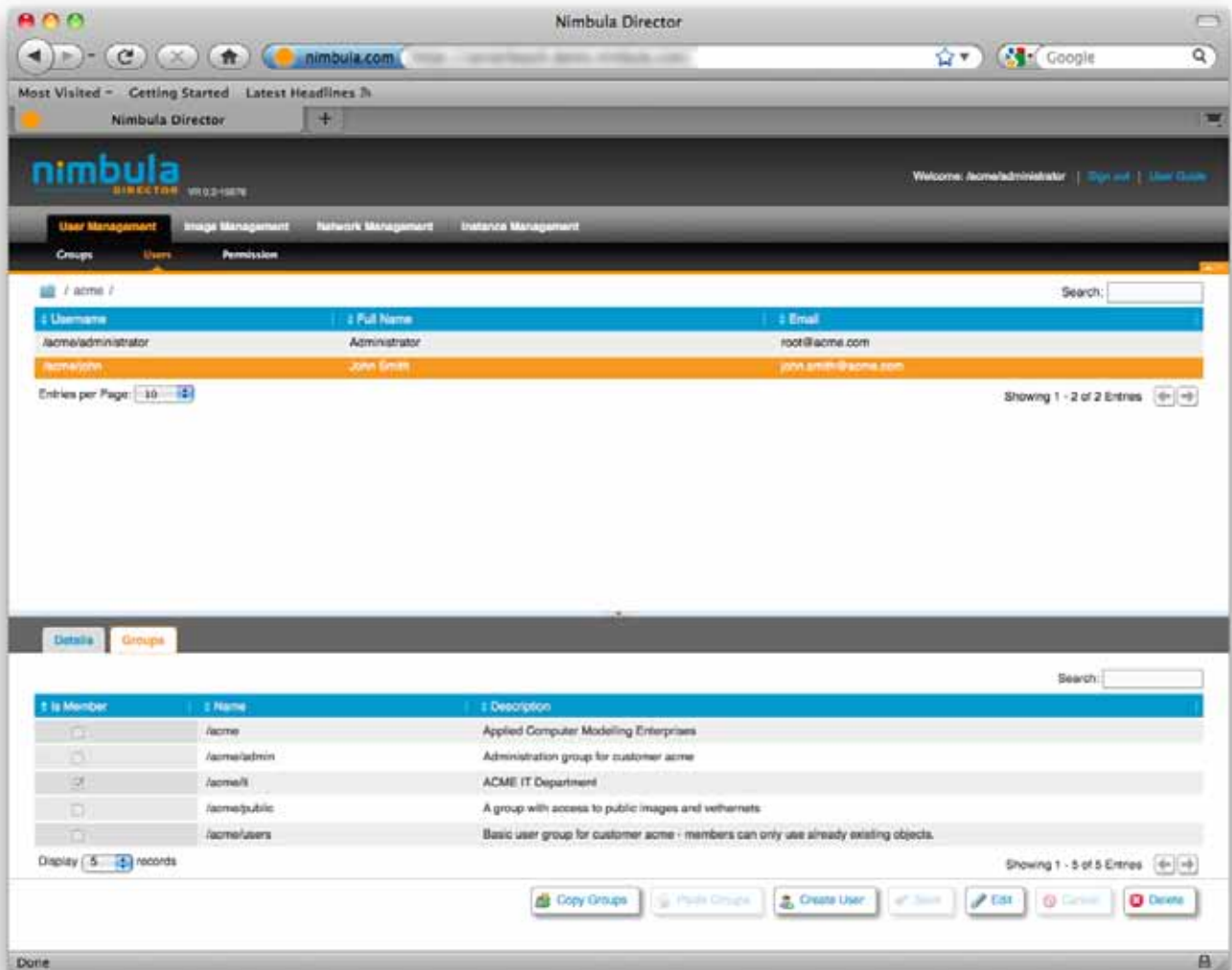



Figure 12: The ACME Administrator adds John as a member of the IT group.

5. `root` uses the CLI to create Customer Wasp, and the Wasp Administrator adds the Wasp Support Department group and adds Sally Jones to this group.



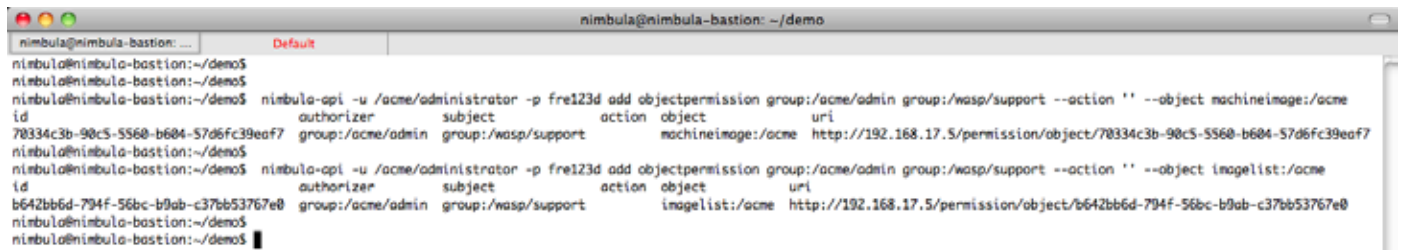
```

nimble@nimble-bastion: ~/demo
nimble@nimble-bastion:~/demo$ nimble-api signup -u /root/root -p nimble wasp "Western Allied Service Provider" root@wasp.com sti456ng True
uri http://192.168.17.5/signup/wasp
name wasp
description Western Allied Service Provider
multiuser True
email root@wasp.com
password sti456ng
nimble@nimble-bastion:~/demo$ nimble-api add group -u /wasp/administrator -p sti456ng /wasp/support "Support Department" --members '[]'
uri http://192.168.17.5/group/wasp/support
name /wasp/support
description Support Department
customer wasp
members []
subgroups []
nimble@nimble-bastion:~/demo$ nimble-api -u /wasp/administrator -p sti456ng add user /wasp/sally "Sally Jones" sally@wasp.com sallypassword --groups '["/wasp/support"]'
uri http://192.168.17.5/user/wasp/sally
username /wasp/sally
fullname Sally Jones
email sally@wasp.com
password sallypassword
customer wasp
groups ["/wasp/support"]
nimble@nimble-bastion:~/demo$

```

Figure 13: `root` uses the Nimble CLI to create Customer Wasp and the Wasp Administrator adds the Wasp Support Department group and adds Sally Jones to this new group.

6. ACME Administrator grants all users in the Wasp Support Department group permission to perform any action on ACME machine images and image lists. Note that users of Wasp are not allowed the ability to manipulate launch plans of launch virtual machines.



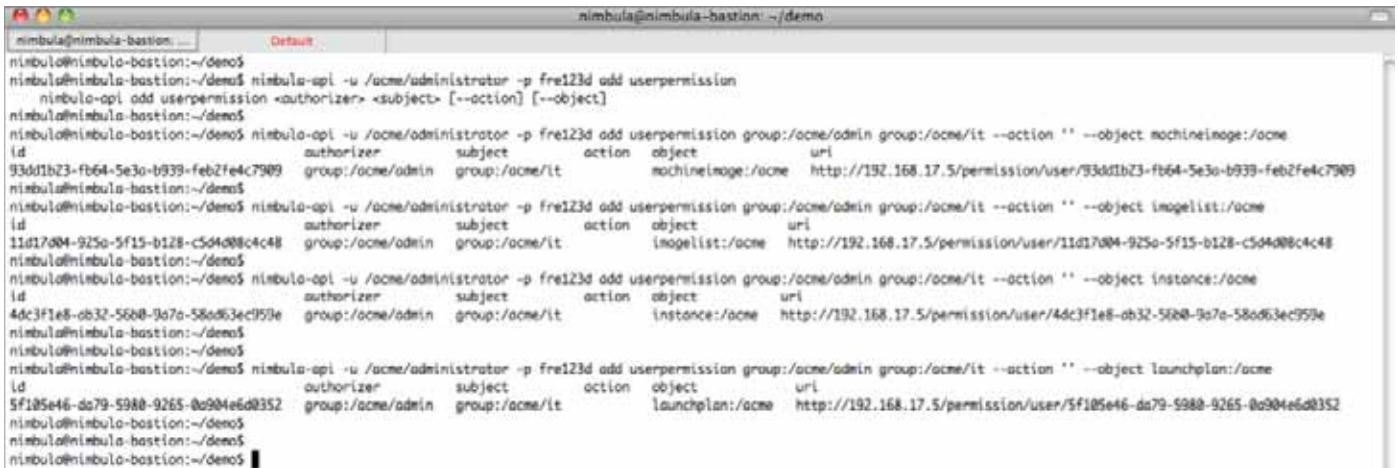
```

nimble@nimble-bastion: ~/demo
nimble@nimble-bastion:~/demo$ nimble-api -u /acme/administrator -p fre123d add objectpermission group:/acme/admin group:/wasp/support --action '*' --object machineimage:/acme
id 70334c3b-90c5-5560-b604-57d6fc39eaf7
authorizer group:/acme/admin
subject group:/wasp/support
action machineimage:/acme
object http://192.168.17.5/permission/object/70334c3b-90c5-5560-b604-57d6fc39eaf7
nimble@nimble-bastion:~/demo$ nimble-api -u /acme/administrator -p fre123d add objectpermission group:/acme/admin group:/wasp/support --action '*' --object imagelist:/acme
id b642bb6d-794f-56bc-b9ab-c37bb53767e0
authorizer group:/acme/admin
subject group:/wasp/support
action imagelist:/acme
object http://192.168.17.5/permission/object/b642bb6d-794f-56bc-b9ab-c37bb53767e0
nimble@nimble-bastion:~/demo$

```

Figure 14: ACME Administrator grants appropriate permissions to Wasp group users.

7. ACME Administrator grants users of the IT group permission to perform any action on ACME machine images and image lists, including the ability to manipulate instances and launch virtual machines.



```
nimbula@nimbula-bastion: ~/demo
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /acme/administrator -p fre123d add userpermission
nimbula-api add userpermission <authorizer> <subject> [--action] [--object]
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /acme/administrator -p fre123d add userpermission group:/acme/admin group:/acme/it --action "" --object machineimage:/acme
id          authorizer  subject    action    object    url
93dd1b23-fb64-5e3a-b939-feb2fe4c7909 group:/acme/admin group:/acme/it machineimage:/acme http://192.168.17.5/permission/user/93dd1b23-fb64-5e3a-b939-feb2fe4c7909
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /acme/administrator -p fre123d add userpermission group:/acme/admin group:/acme/it --action "" --object imagelist:/acme
id          authorizer  subject    action    object    url
11d17d04-925a-5f15-b128-c5d4d08c4c48 group:/acme/admin group:/acme/it imagelist:/acme http://192.168.17.5/permission/user/11d17d04-925a-5f15-b128-c5d4d08c4c48
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /acme/administrator -p fre123d add userpermission group:/acme/admin group:/acme/it --action "" --object instance:/acme
id          authorizer  subject    action    object    url
4dc3f1e8-ab32-56b8-9a7a-58d63ec959e group:/acme/admin group:/acme/it instance:/acme http://192.168.17.5/permission/user/4dc3f1e8-ab32-56b8-9a7a-58d63ec959e
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /acme/administrator -p fre123d add userpermission group:/acme/admin group:/acme/it --action "" --object launchplan:/acme
id          authorizer  subject    action    object    url
5f105e46-da79-5980-9265-0a904e6d8352 group:/acme/admin group:/acme/it launchplan:/acme http://192.168.17.5/permission/user/5f105e46-da79-5980-9265-0a904e6d8352
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
```

Figure 15: ACME Administrator grants appropriate permissions to users of the IT group.

Use Case Two: Users of the service can now provision and administer business services in the cloud

Nimbula Director uses the concept of image lists, which is a list of machine images, to manage multiple versions of images. Machine images are added to an image list containing a selection of machine images of various operating system releases. At launch time, the administrator can then specify which entry in the list is to be used to launch instances.

1. Sally uses the CLI to create an image list called /acme/it/demolist in the ACME IT repository.

```
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /wasp/sally -p sallypassword add imagelist
  nimbula-api add imagelist <name> <description> <default>
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /wasp/sally -p sallypassword add imagelist /acme/it/demolist "Acme IT List" 1
uri                                     name                                description                         default      entries
http://192.168.17.5/imagelist/acme/it/demolist  /acme/it/demolist                Acme IT List                        1           1
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
```

Figure 16: Sally adds an image list.

2. Sally then uploads a machine image and names it /acme/it/image1. A machine image is a virtual hard drive to be used when starting the VM.

```
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /wasp/sally -p sallypassword add machineimage
  nimbula-api add machineimage <name> <file> [--attributes] [--account]
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /wasp/sally -p sallypassword add machineimage /acme/it/image1 ~/acme.tar.gz
uri                                     name                                file                                attributes                                     account
http://192.168.17.5/machineimage/acme/it/image1  /acme/it/image1                    None                                {"nimbula_compressed_size": 61774678, "nimbula_decompressed_size": 536983168}  None
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
```

Figure 17: Sally uploads a machine image.

3. The next step is for Sally to add this new machine image to the demolist image list. The last argument to the nimbula-api call denotes the version of the machine image added. For one entry in an image list, Nimbula Director allows multiple images for launching into different endpoints. The version number is used to specify which machine image to use by default at launch time.

```
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /wasp/sally -p sallypassword add imagelistentry
nimbula-api add imagelistentry <imagelist name> <machineimages> <version> [--attributes]
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$ nimbula-api -u /wasp/sally -p sallypassword add imagelistentry /acme/it/demolist /acme/it/imagel 1
uri                                imagelist                          machineimages                      version                          attributes
http://192.168.17.5/imagelist/acme/it/demolist/entry/1  imagelist/acme/it/demolist        /acme/it/imagel                  1                                {}
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
nimbula@nimbula-bastion:~/demo$
```

Figure 18: Sally adds her newly-created machine image to the demolist list and makes it the default image to use at launch time.

- John uses the Web interface to upload his machine image to the /acme/john repository and calls it /acme/it/JohnsList.

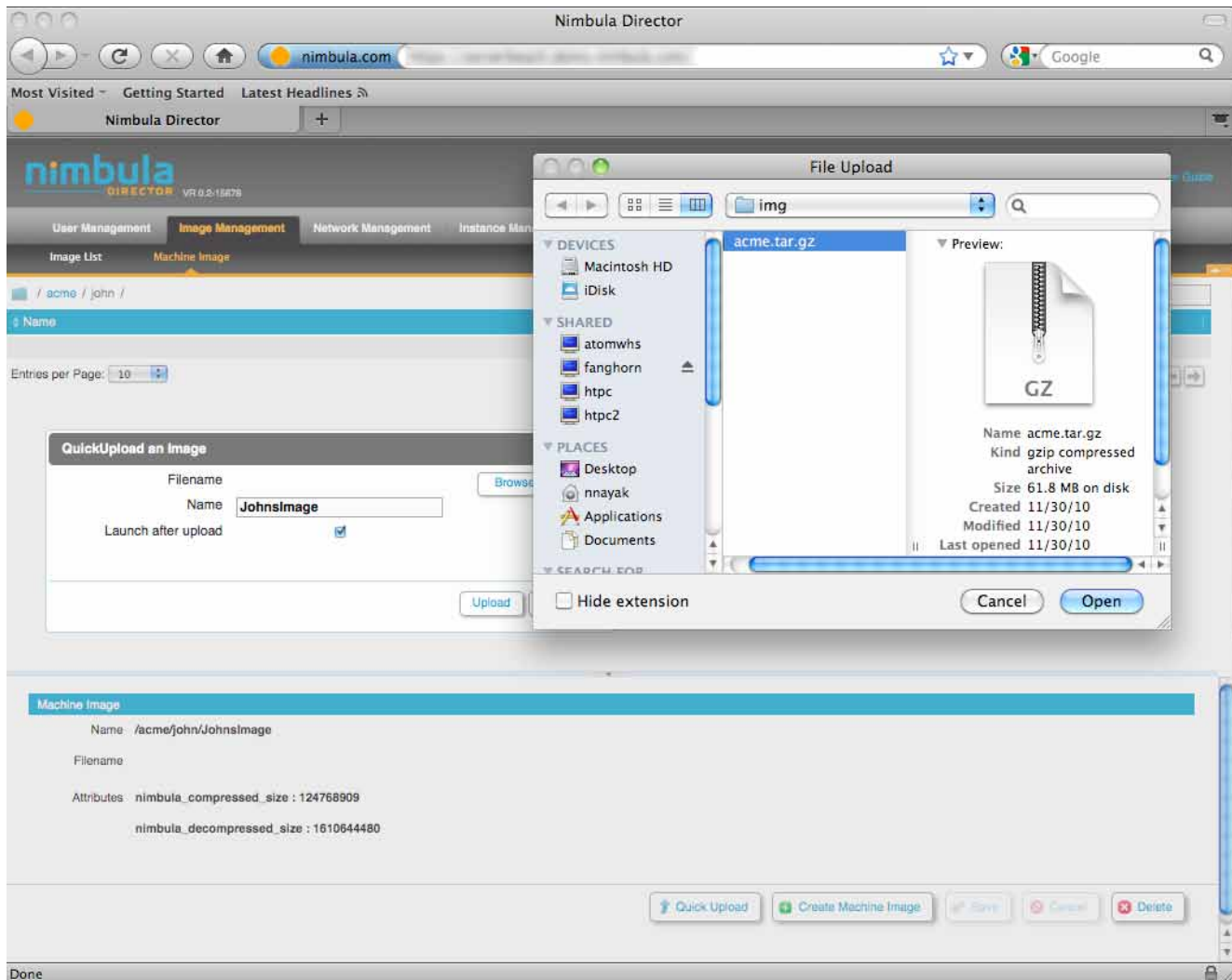


Figure 19: John performs a Quick Upload to upload his machine image.

5. John then adds his new machine image to the /acme/it/demolist image list created by Sally.
 - **Note:** He is allowed to change the Entry of the images in in demolist. At this time, John uses the list entry number 2 for his image, possibly to test out his image before changing it to be the default entry

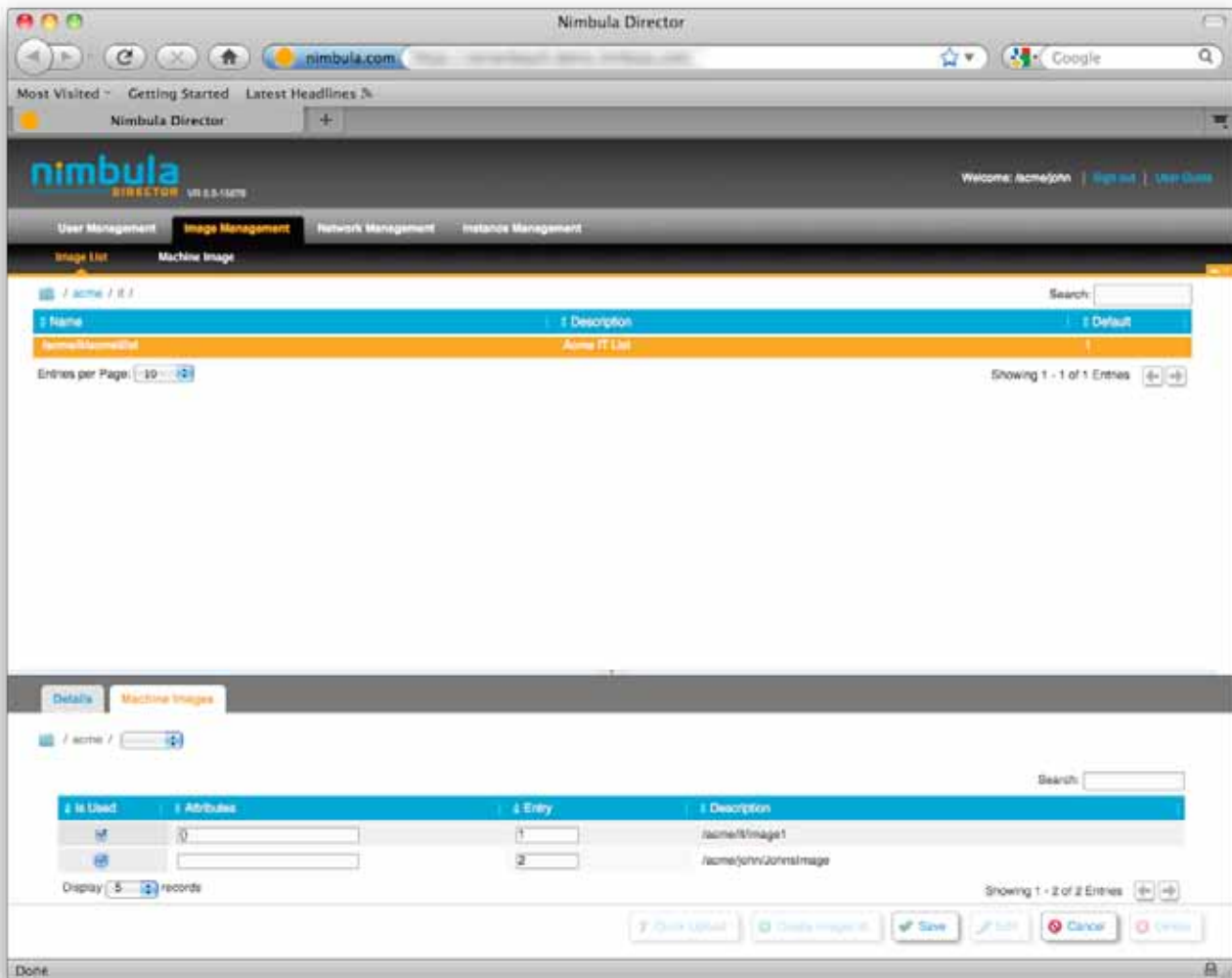
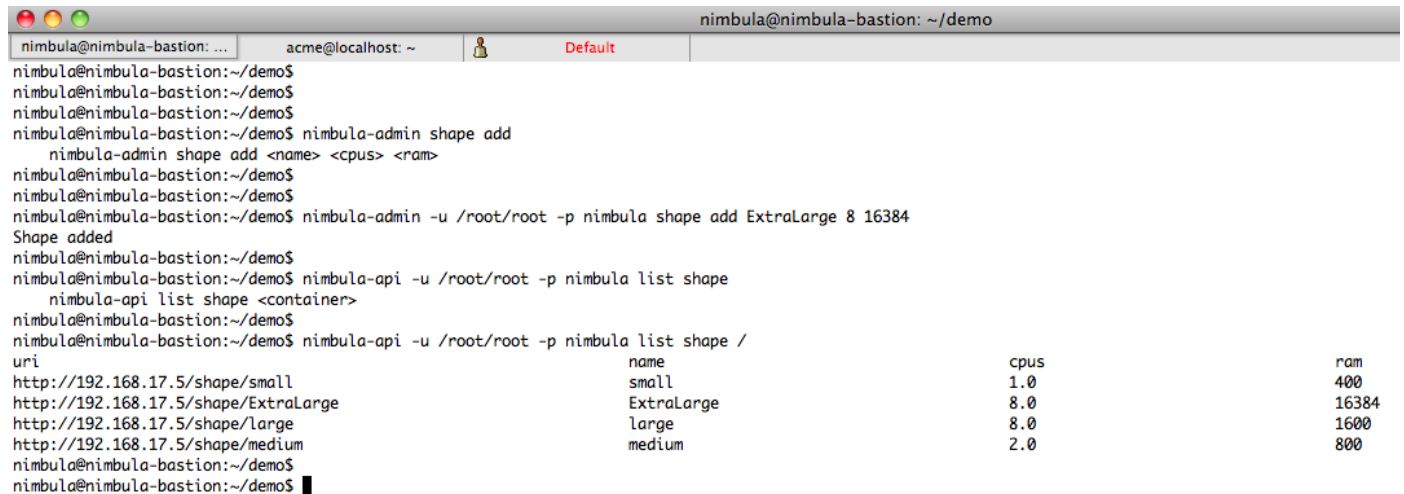


Figure 20: John adds a new machine image to the /acme/it/demolist image list.

Use Case Three: The Site Administrator scales service vertically

As ACME and its internal customers grow, so do the needs of its internal private cloud. The Site Administrator must either scale services vertically (up) or horizontally (out) as the workload demands. The steps to scale the service vertically are as follows:

1. `root` adds a new shape called `large` with eight CPUs and 16 GB RAM to replace the default small shape that was being used by IT. A shape refers to the combination of the number of CPUs and amount of RAM assigned to the VM.



```
nimbula@nimbula-bastion: ~/demo
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-admin shape add
nimbula-admin shape add <name> <cpus> <ram>
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-admin -u /root/root -p nimbula shape add ExtraLarge 8 16384
Shape added
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-api -u /root/root -p nimbula list shape
nimbula-api list shape <container>
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-api -u /root/root -p nimbula list shape /
uri                                name                                cpus                                ram
http://192.168.17.5/shape/small    small                               1.0                                400
http://192.168.17.5/shape/ExtraLarge ExtraLarge                           8.0                                16384
http://192.168.17.5/shape/large    large                               8.0                                1600
http://192.168.17.5/shape/medium    medium                              2.0                                800
nimbula@nimbula-bastion: ~/demo$
```

Figure 21: `root` (the Site Administrator) adds a new VM shape with eight cores.

2. Once the new large shape has been created, John can launch new instances from the image list with this shape.
- **Note:** New shapes can only be added when they can be mapped onto existing cluster hardware. For example, you cannot declare an eight core shape, if you do not have a machine or machines in your clusters which will be able to satisfy a placement request for this shape.

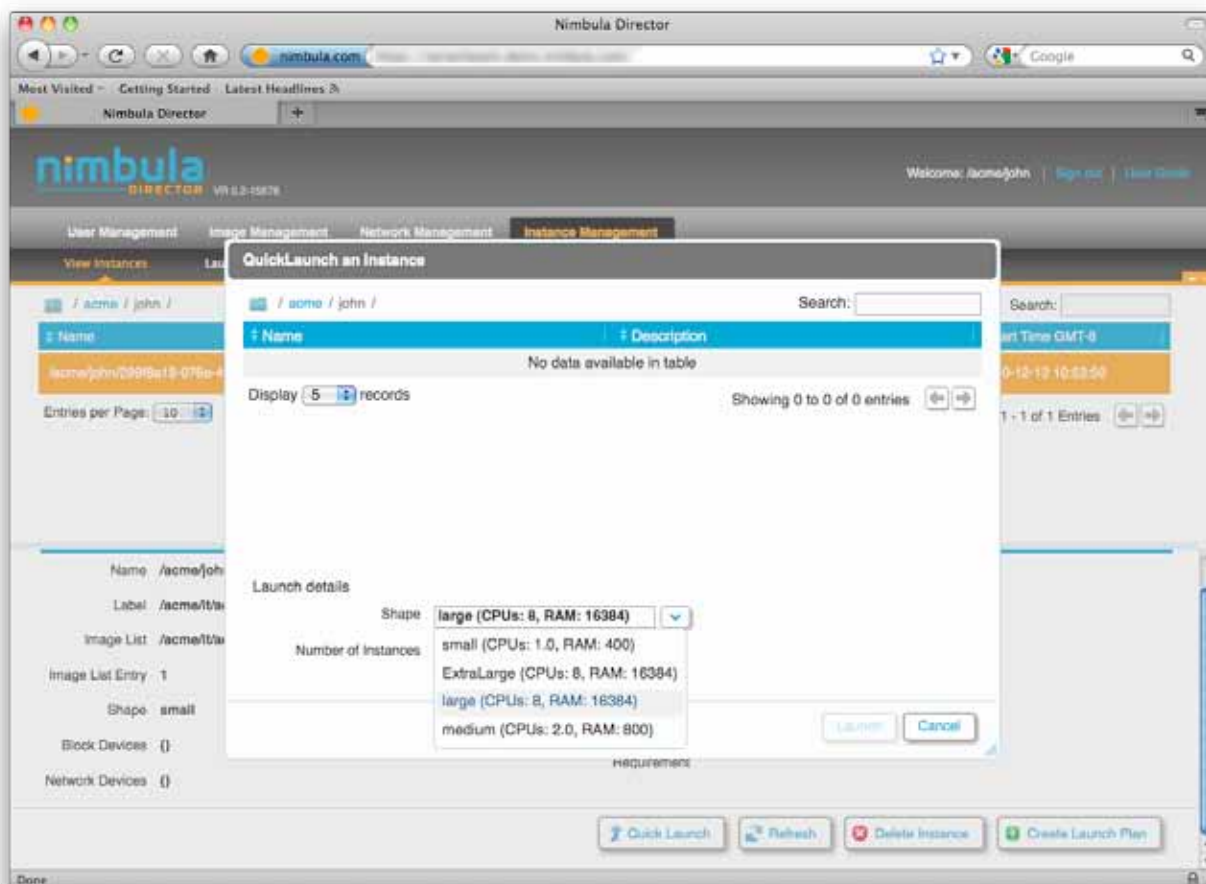


Figure 22: John uses Quick Launch with the newly added shape to launch a new VM.

3. John then verifies that his new VM has been launched and is running. The Web interface displays details like the image list, state, placement requirements, etc. for the new VM. It also provides him with an IP address to use to connect to the VM (192.168.20.10 in this case).

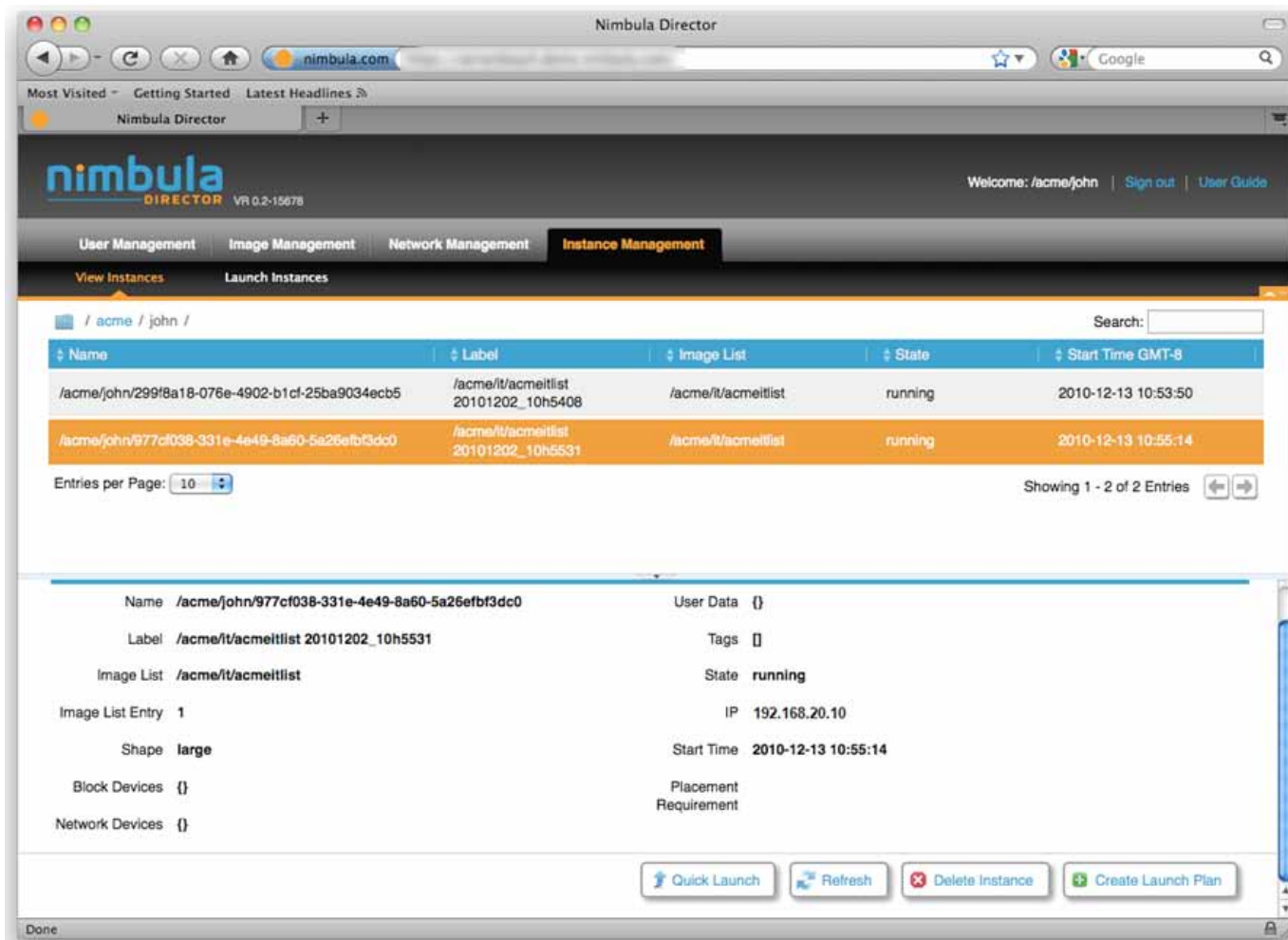


Figure 23: Once the VM is launched, the Nimbula API Web interface displays its status as Running and an IP address to connect to the VM.

- Once logged in to the VM, John verifies that he has 8 CPUs and 16 GB RAM. At this point, he can modify his DNS entries to point to the new VM, thus allowing him to scale his workload vertically.

```
nimbula@nimbula-bastion:~$
nimbula@nimbula-bastion:~$
nimbula@nimbula-bastion:~$ ssh acme@192.168.20.10
acme@192.168.20.10's password:
Linux RaaS Nimble 2.6.35-22-server #35-Ubuntu SMP Sat Oct 16 22:02:33 UTC 2010 x86_64 GNU/Linux
Ubuntu 10.10

Welcome to the Ubuntu Server!
 * Documentation:  http://www.ubuntu.com/server/doc

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

acme@RaaS Nimble:~$ uptime
12:21:33 up 2 min,  1 user,  load average: 0.12, 0.07, 0.02
acme@RaaS Nimble:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/vda1        5.5G  1.4G  3.9G  27% /
none            7.9G  148K  7.9G   1% /dev
none            7.9G   0  7.9G   0% /dev/shm
none            7.9G   36K  7.9G   1% /var/run
none            7.9G   0  7.9G   0% /var/lock
acme@RaaS Nimble:~$ grep -i processor /proc/cpuinfo
processor       : 0
processor       : 1
processor       : 2
processor       : 3
processor       : 4
processor       : 5
processor       : 6
processor       : 7
acme@RaaS Nimble:~$ free
              total        used        free      shared    buffers     cached
Mem:      16467316     377960    16089356           0       8168       29456
-/+ buffers/cache:      340336    16126980
Swap:      317436           0       317436
acme@RaaS Nimble:~$
```

Figure 24: John logs in to his new Large VM and verifies that he now has more CPU and memory resources.

Use Case Four: John scales service horizontally

As demand and throughput needs increase, more resources beyond those available from vertical scaling are often needed. Traditionally, more physical servers would be wheeled in to the datacenter and configured for use. With cloud computing and the benefits of virtualization, new VMs can be started up in minutes assuming the physical resources are available.

1. John starts up seven more instances of large VMs to add to the existing one.

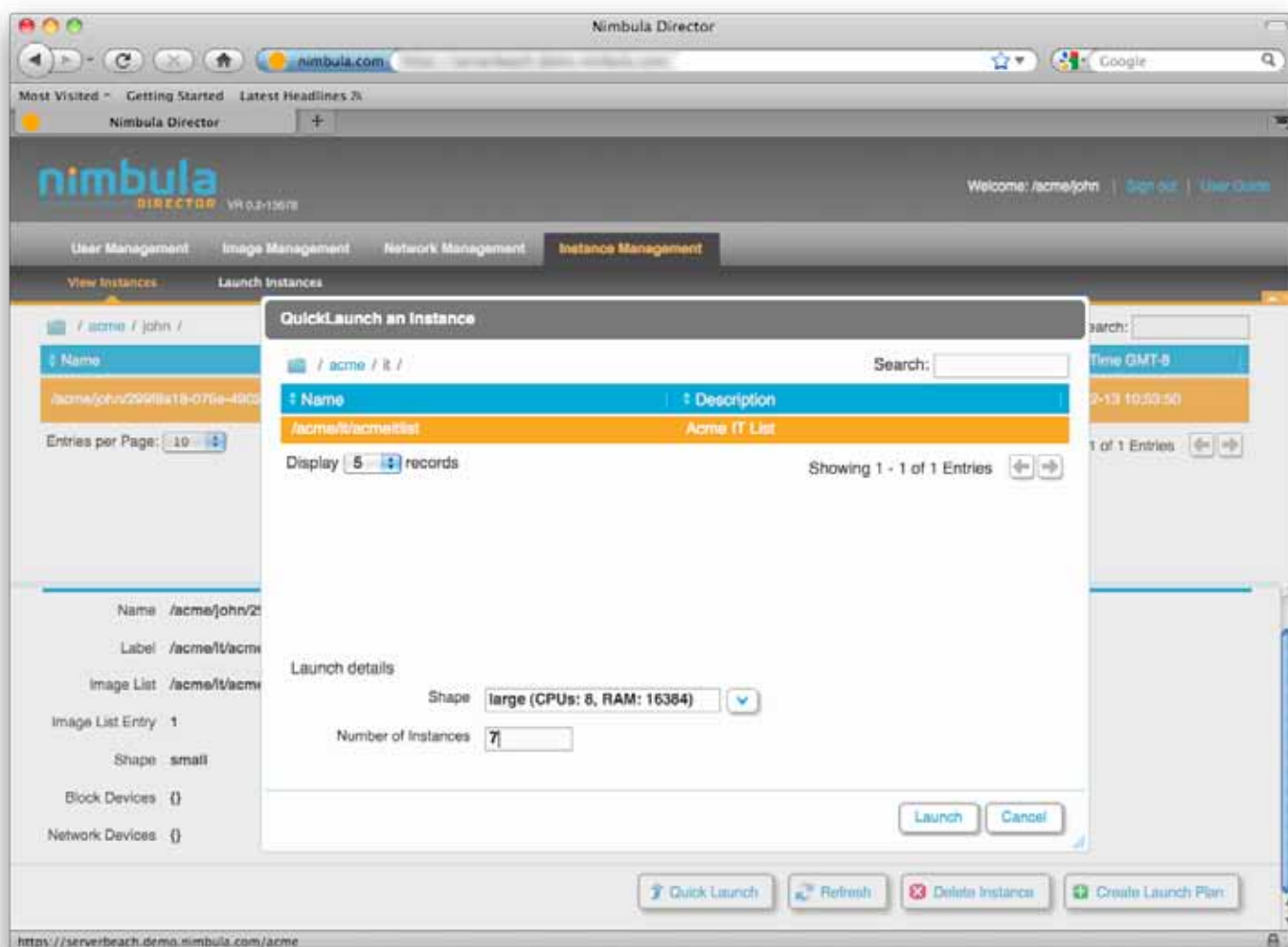


Figure 25: John launches seven Large VMs in anticipation of demand.

- The Web interface verifies that all the eight VMs are now running. At this point, John can configure a loadbalancer to spread requests across all eight VMs, thus increasing the throughput of the service eight-fold on the fly.

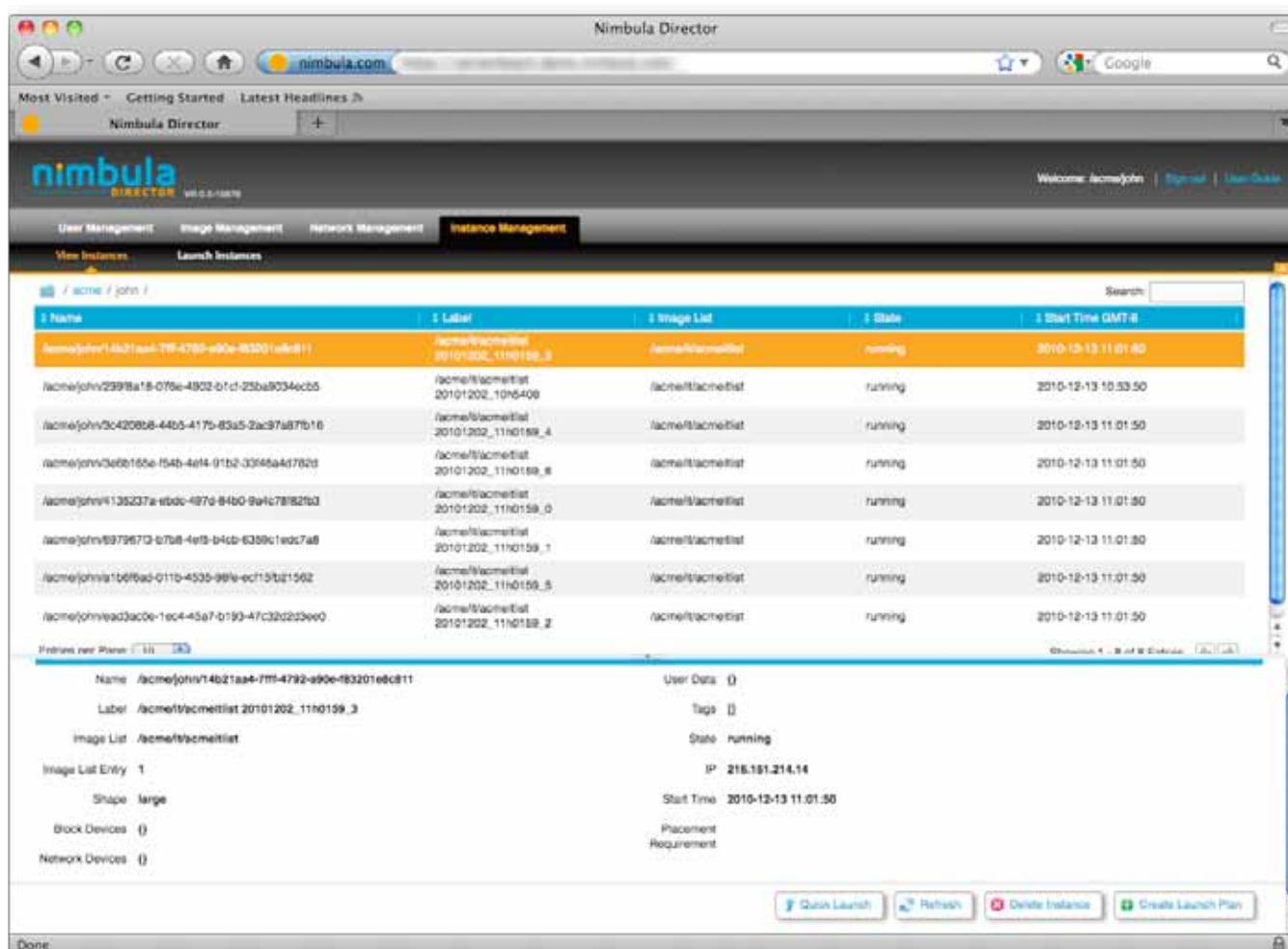


Figure 26: The Nimbula API web interface displays all eight VMs as running.

Use Case Five: The ACME Administrator configures cloud federation to enable burst capability

Cloud Federation allows the merging of private and public clouds to create a hybrid cloud. Nimbula Director supports request forwarding to external clouds, subject to the fine-grained permissions management available via the Nimbula API, which acts as a common interface to administer both types of clouds. The steps to configure and use Cloud Federation are as follows:

1. ACME Administrator creates a public cloud account (Amazon EC2 in this case) using the credentials provided by the cloud operator. This account information is passed through to all calls to EC2.

```
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-api -u /acme/administrator -p fre123d add account /acme/public_cloud \
> --credentials '{ "access_key": "AKIAI44QH8DHBEXAMPLE", "secret_key": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0u1v2w3x4y5z6" }'
uri      name      credentials
http://192.168.4.5/account/acme/pub /acme/public_cloud {"access_key": "AKIAI44QH8DHBEXAMPLE", "secret_key": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0u1v2w3x4y5z6"}
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$
```

Figure 27: ACME Administrator adds an account for Amazon EC2 access with the appropriate access and secret keys.

2. ACME Administrator grants users in the IT group permission to perform all actions on the remote site.

```
nimbula@nimbula-bastion: ~/demo$ nimbula-api -u /acme/administrator -p fre123d add userpermission POLICY group:/acme/it --action '' --
object site:/ec2proxy/us-east-1
id      authorizer subject      acti object      uri
11566185-7cc8-50d9-8db4-ecc4dcda0470 POLICY  group:/acme/i site:/ec2proxy/us-e http://192.168.4.5/permission/user/11566185-7cc
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-api -u /acme/administrator -p fre123d add objectpermission POLICY group:/acme/it --action ''
--object site:/ec2proxy/us-east-1
id      authorizer subject      acti object      uri
f464eca6-0850-56e4-8e21-c8bbeec02eb7 POLICY  group:/acme/i site:/ec2proxy/us-e http://192.168.4.5/permission/object/f464eca6-0
nimbula@nimbula-bastion: ~/demo$ █
```

Figure 28: ACME Administrator adds user and object permissions to access the appropriate EC2 site.

- John then adds an image list in the EC2 cloud called public and adds an Amazon Machine Image (AMI, denoted by ami-3c47a355 in this example) to the public list. He also configures this AMI to be the default image during launch time.

```
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-api -u /acme/john -p johnpassword add imagelist /acme/it/public "Public Cloud List" 1
uri                                     name          description      default      entries
http://192.168.4.5/imagelist/acme/it/public /acme/it/public Public Cloud List 1
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-api -u /acme/john -p johnpassword add imagelistentry /acme/it/public '["ec2://ec2.us-east-1.a
mazonaws.com/image/ami-3c47a355"]' 1 --attributes '{}'
uri                                     imagelist      machineimages      version a
http://192.168.4.5/imagelist/acme/it/public/en imagelist/acme/it/public ec2://ec2.us-east-1.amazonaws.com/image/ami-3c47a355 1 {
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ █
```

Figure 29: John can now add an image list and add his Amazon Machine Image to this image list.

4. To launch an instance in EC2, John needs to specify the site on, which the instances should be launched. He can do this with the help of a launch plan expressed via a JSON file shown below.

```
{
  "relationships": [
  ],
  "instances": [
    {
      "shape": "small",
      "version": 1,
      "tags": [
        "ACME Datacenter"
      ],
      "network_devices": {
      },
      "attributes": [
        ""
      ],
      "imagelist": "/acme/it/public", "block_devices": {
      },
      "user_data": {},
      "label": "ACME IT workloads"
    }
  ]
}

{
  "relationships": [
  ],
  "instances": [
    {
      "shape": "small",
      "version": 1,
      "tags": [
        "ACME Datacenter"
      ],
      "network_devices": {
      },
      "attributes": [
        ""
      ],
      "imagelist": "/acme/it/public", "block_devices": {
      },
      "user_data": {},
      "label": "ACME IT workloads"
    }
  ]
}
```

Figure 30: JSON file describing the site and instance type to use in the public cloud.

5. John then deploys the VM using the launch command available via the CLI.

```
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$ nimbula-api -b /acme/public_cloud -s us-east-1 launch publiclaunch.json \
> -a http://ec2proxy.beach.nimbula:5500/ec2/
uri          name          label  imagelis  ent  sha  block  network_dev  veth  placement_requi  use  tags  stat  ip  start_tim  account  site
ec2://ec2.us-east-1 i-61511a0c ACME IT /acme/it Non sma {} {"eth0": "d {} r-7c99fe16, us- {} ACME Da pend No 2010-11-3 /acme/pub us-e
nimbula@nimbula-bastion: ~/demo$
nimbula@nimbula-bastion: ~/demo$
```

Figure 31: Launching the VM in Amazon EC2 using the Nimbula API CLI.

6. ACME Administrator can log in to the Amazon EC2 account to check on resource consumption and for auditing purposes. The only VM running in this case is the one John started up in the previous step.

The screenshot shows the Amazon EC2 console interface. The top navigation bar includes links for AWS, Products, Developers, Community, Support, and Account. The main header shows 'Welcome, [User Name]' and links for Settings and Sign Out. The left sidebar contains a navigation menu with categories like INSTANCES, IMAGES, ELASTIC BLOCK STORE, and NETWORKING & SECURITY. The main content area is titled 'My Instances' and shows a table with one instance: 'i-61511a0c'. Below the table, the details for the selected instance are displayed, including AMI ID, Security Groups, Status, VPC ID, Virtualization, Reservation, Platform, Kernel ID, AMI Launch Index, Root Device, Block Devices, Lifecycle, Public DNS, Private DNS, Private IP Address, Launch Time, and State Transition Reason.

Name	Instance	AMI ID	Root Device	Type	Status	Security Groups	Key Pair Name	Monitoring	Virtualization	Placement Group
i-61511a0c	ami-3c47a355	instance-store	m1.small	running	default	-	-	disabled	paravirtual	-

1 EC2 Instance selected

EC2 Instance: i-61511a0c

Description		Monitoring		Tags	
AMI ID:	ami-3c47a355	Zone:	us-east-1a	Type:	m1.small
Security Groups:	default	Owner:	527415664720	Subnet ID:	-
Status:	running	Placement Group:	-	RAM Disk ID:	ami-a51cf9cc
VPC ID:	-	Key Pair Name:	-	Monitoring:	disabled
Virtualization:	paravirtual	Elastic IP:	-	Root Device Type:	instance-store
Reservation:	r-7c99fe16				
Platform:	-				
Kernel ID:	aki-a71cf9ce				
AMI Launch Index:	0				
Root Device:	-				
Block Devices:	N/A - Instance Store				
Lifecycle:	normal				
Public DNS:	ec2-184-73-81-22.compute-1.amazonaws.com				
Private DNS:	domU-12-31-38-04-70-91.compute-1.internal				
Private IP Address:	10.220.115.95				
Launch Time:	2010-11-30 14:30 PST				
State Transition Reason:					

Figure 32: Amazon EC2 account page displaying the VM launched from ACME's private cloud.

Things to Consider

The scalability of a production-deployed private cloud is determined by a plethora of factors that we have not considered in this basic reference architecture. Here are a few things you may want to consider:

Hardware Considerations

While a detailed discussion of processor and server performance and power considerations is beyond the scope of this paper, it is important to note that the performance and density of virtual machines running on a server can be heavily influenced by the performance and headroom provided by the processor architecture. Specific feature sets available in the processor such as Intel VT-d, Intel® Turbo Boost Technology, and Intel® Hyper-Threading Technology+ can greatly assist in better performance, energy efficiency and ultimately lower Total Cost of Ownership.

Fail-over

Hardware and service failures are a way of life in the data center industry. Nimbula Director ensures integrity and resilience by eliminating single points of failure with sophisticated failover mechanisms. When any particular server fails (or is taken down for maintenance), the system will automatically transfer all cloud services (Nimbula Control Plane) running on that server to another node. When the server is ready for use, switching it on and connecting it to the network is all that is required for discovery and inclusion into the system.

Storage Architecture and Solid-State Drives (SSDs)

Nimbula Director supports several storage architectures. For this test bed, we selected the simplest solution, local-attached storage. In typical data center deployments, more advanced Storage Attached Network architectures are likely to be used. The performance of storage nodes as well as overall data center power

consumption may be favorably impacted by the use of SSDs. Since our use cases did not focus on either of these two characteristics, SSDs were not specifically tested.

Networking

While network capabilities was not part of this test bed, Nimbula Director provides a flexible and scalable capabilities that include vEthernets and Security Lists.

Nimbula Director allows customers to create and declare their own highly dynamic virtual network topologies. Users can create virtual Ethernets, which enable the launching of instances in multiple isolated layer 2 networks where customers may provide their own DHCP server and other layer 2 services, such as multicast broadcast and non-IP Ethernet protocols.

Nimbula Security lists allow role-based firewall management. Instead of defining firewall rules on a per machine (IP) basis, security profiles are defined and access between VMs can be defined in terms of these profiles, called security lists. In a cloud environment, where dynamic resource provisioning can see instances launched or terminated frequently, assigning an instance to one or more security lists at launch time removes the complexity of constantly updating machine based firewall rules.

Instances launched inside Nimbula Director can be tagged as belonging to one or more security lists. In addition, security IP lists can be configured to contain one or more IP subnets to represent IP addresses/ computers outside the Nimbula Director cloud.

Security lists provide a flexible distributed firewall that doesn't grow in complexity with scale. They provide instance isolation while still permitting instances to interact with machines outside the Nimbula Director cloud.

Security and Trust

Security and trust are two of the most important considerations in the cloud computing market. A thorough discussion of best practices for cloud security is beyond the scope of this paper. However, the following points should be considered:

- Established best practices for host security in a conventional physical host context (e.g. password management patch management, server hardening, anti-malware, etc.) should be applied equally to virtual hosts operating in a private cloud.
- Private cloud management systems such as Nimbula Director provide full isolation between virtual servers by employing full hardware-assisted virtualization. This provides each virtual server with its own virtual hardware, its own private operating system instance, etc. This is in contrast with cloud platforms based on "domains" or "containers" in which some virtual hardware and some operating system components are shared between virtual hosts, which may create additional avenues of attack.

Glossary

- **Compute node:** A single physical server that can host multiple virtual machines.
- **Seed node:** The machine used to run the Live CD during initial installation and to “seed” the cluster.
- **Infrastructure-as-a-Service (IaaS):** The delivery mechanism of the use of computing resources (such as network, storage, and CPU cycles) as a service, typically through virtualization.
- **Platform-as-a-Service (PaaS):** The delivery of a computing platform stack as a service.
- **NIC:** A network interface card.
- **VM:** Virtual machine.
- **Launch Plan:** The mechanism used for launching a suite of machine images. A launch plan facilitates the configuration of parameters describing how each instance will run.
- **Permission:** A permission is a delegation of privileges and/or a delegation of authority by an entity with granting authority within the customer cloud account.
- **Instance:** A virtual machine run by Nimbula Director. Instances have attributes such as allocated RAM, number of CPUs available, virtual block devices and network interfaces attached. Instances are created using a launch plan that specifies the desired set of machines, which image lists they are to be launched from and placement relationships that exist between them.
- **Machine image:** A machine image is a virtual disk image from which an instance is launched.
- **Hypervisor:** Allows multiple operating systems to run concurrently on a host computer effectively providing hardware virtualization.

References

1. A RESTful Web API is a simple Web service implemented using HTTP and the principles of Representational State Transfer (REST), a style of software architecture for distributed hypermedia systems. A RESTful Web API consists of a collection of resources, with three defined aspects: the base URI for the Web service, such as <http://example.com/resources/>; the Internet media type of the data supported by the Web service (often JSON, XML or YAML but can be any other valid Internet media type); and the set of operations supported by the Web service using HTTP methods (e.g., POST, GET, PUT or DELETE). (source: http://en.wikipedia.org/wiki/Representational_State_Transfer).
2. Intel® Virtualization Technology (Intel® VT): <http://www.intel.com/technology/virtualization/>
3. Intel® Trusted Execution Technology (Intel® TXT): <http://www.intel.com/technology/security/downloads/TrustedExecOverview.pdf>
4. The Nimbula Director Installation and Administration Guide can be downloaded on the Nimbula website at <http://nimbula.com/>.

Additional Information

- For more information on the Intel Cloud Builders Program, visit www.intel.com/cloudbuilders
- For more information on Intel Xeon processors, visit www.intel.com/xeon
- For more information on Nimbula Director, visit <http://www.nimbula.com/products/overview>
- Amazon Elastic Compute Cloud* (EC2): <http://aws.amazon.com/ec2/>
- Cloud Builders Reference Architecture Library: <http://software.intel.com/en-us/articles intel-cloud-builders-reference-architecture-library/>

Disclaimers

Δ Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See www.intel.com/products/processor_number for details.

+ Requires an Intel® Hyper-Threading Technology (Intel® HT Technology)-enabled system, check with your PC manufacturer. Performance will vary depending on the specific hardware and software used. Not available on the Intel® Core™ i5-750. For more information, including details on which processors support HT technology, visit www.intel.com/technology/platform-technology/hyper-threading/index.htm.

◊ Intel® Virtualization Technology requires a computer system with an enabled Intel® processor, BIOS, virtual machine monitor (VMM) and, for some uses, certain platform software enabled for it. Functionality, performance or other benefits will vary depending on hardware and software configurations and may require a BIOS update. Software applications may not be compatible with all operating systems. Please check with your application vendor.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request. Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order. Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting Intel's Web site at www.intel.com.

Copyright © 2011 Nimbula. Copyright © 2011 Intel Corporation. All rights reserved. Intel, the Intel logo, Xeon, Intel Cloud Builders, Intel Virtualization Technology and Intel Trusted Execution Technology are trademarks of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.