Lecture on Cloud Computing

Course 10.
Academic cloud systems
Brief history of Academic & Open Source Cloudwares

- Xen, Xenoserver platform: 2001-2003
- In Vigo Project – till 2005
- OpenNebula, Eucalyptus 2008-
- OpenStack, CloudStack 2010-
- Apache Tashi 2009-
Clouds using Opennebula

Gabor Kecskemeti

kecskemeti.gabor@sztaki.mta.hu
kecskemeti@iit.uni-miskolc.hu

This presentation is heavily based on multiple presentations of the following people: Ignacio M. Llorente, Rubén S. Montero, Jaime Melis, Javier Fontán, Rafael Moreno
Outline

- Virtual infrastructure managers
- OpenNebula as a whole
- Architectural view on OpenNebula
- Constructing a Private cloud
- Virtual Machines in OpenNebula
- Constructing a Hybrid cloud
VIRTUAL INFRASTRUCTURE MANAGERS
Why a Virtual Infrastructure Manager?

- VMs are great!!...but something more is needed
  - Where did/do I put my VM? (scheduling & monitoring)
  - How do I provision a new cluster node? (clone & context)
  - What MAC addresses are available? (networking)
- Provides a uniform view of the resource pool
- Life-cycle management and monitoring of VM
- The VIM integrates Image, Network and Virtualization management

OpenNebula (VIM)
Extending the Benefits of Virtualization to Clusters

- Dynamic deployment and re-placement of virtual machines on a pool of physical resources

- Transform a rigid distributed physical infrastructure into a flexible and agile virtual infrastructure

- Backend of Public Cloud: Internal management of the infrastructure
- Private Cloud: Virtualization of cluster or data-center for internal users
- Cloud Interoperation: On-demand access to public clouds
The three pillars of a Distributed VM Manager

- **Service structure**
  - Service components run in VMs
  - Inter-connection relationship
  - Placement constraints

- **The VM Manager is service agnostic**

---

**Virtual Machine Management Model**

- **Service as Management Entity**
  - **Distributed VM Management System**
  - **Networking**
  - **Image Management**
  - **Virtualization**
WHAT IS OPENNEBULA?
What is OpenNebula?

An Open-source Toolkit to Build your IaaS Cloud

Enabling Technology to Build your Cloud

- Private Cloud to simplify and optimize internal operations
- Hybrid Cloud to supplement the capacity of the Private Cloud
- Public Cloud to expose your Private to external users
What is the OpenNebula Open-Source Project?

Building the Industry Standard Open Source Cloud Computing Tool

Lead Innovation in Enterprise-Class Cloud Computing Management

- Develop, maintain and assure the quality of OpenNebula
- Collaborate with open-source and research projects and communities
- Support the community and the ecosystem

An Active and Engaged Community

- 4,000 downloads/month
- 100 active contributors
- OSS distribution channels

From a Research Project on Scalable Management of VMs:
The Benefits of OpenNebula

• For the Infrastructure Manager
  • Centralized management of VM workload and distributed infrastructures
  • Support for VM placement policies: balance of workload, server consolidation…
  • Dynamic resizing of the infrastructure
  • Dynamic partition and isolation of clusters
  • Dynamic scaling of private infrastructure to meet fluctuating demands
  • Lower infrastructure expenses combining local and remote Cloud resources

• For the Infrastructure User
  • Faster delivery and scalability of services
  • Support for heterogeneous execution environments
  • Full control of the lifecycle of virtualized services management
Interoperability from the Cloud Provider perspective

- Interoperable (platform independent), innovative (feature-rich) and proven (mature to run in production).
• Standards (de facto and de jure) and adapters can be used to leverage existing ecosystems and ensure portability across providers....
The Benefits for System Integrators

• Fits into any existing data center, due to its open, flexible and extensible interfaces, architecture and components
• Builds any type of Cloud deployment
• Open source software, Apache license
• Seamless integration with any product and service in the cloud ecosystem and management tool in the data center, such as
  – cloud providers
  – VM managers
  – virtual image managers
  – service managers
  – management tools
  – schedulers
<table>
<thead>
<tr>
<th>Feature</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Internal Interface</strong></td>
<td>• Unix-like CLI for full management of VM life-cycle and physical boxes</td>
</tr>
<tr>
<td></td>
<td>• XML-RPC API and libvirt virtualization API</td>
</tr>
<tr>
<td><strong>Scheduler</strong></td>
<td>• Requirement/rank matchmaker allowing the definition of workload and</td>
</tr>
<tr>
<td></td>
<td>resource-aware allocation policies</td>
</tr>
<tr>
<td></td>
<td>• Support for advance reservation of capacity through Haizea</td>
</tr>
<tr>
<td><strong>Virtualization Management</strong></td>
<td>• Xen, KVM, and VMware</td>
</tr>
<tr>
<td></td>
<td>• Generic libvirt connector</td>
</tr>
<tr>
<td><strong>Image Management</strong></td>
<td>• General mechanisms to transfer and clone VM images</td>
</tr>
<tr>
<td><strong>Network Management</strong></td>
<td>• Definition of isolated virtual networks to interconnect VMs</td>
</tr>
<tr>
<td>**Service Management and</td>
<td>• Support for multi-tier services consisting of groups of inter-connected</td>
</tr>
<tr>
<td>Contextualization**</td>
<td>VMs, and their auto-configuration at boot time</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td>• Management of users by the infrastructure administrator</td>
</tr>
<tr>
<td><strong>Fault Tolerance</strong></td>
<td>• Persistent database backend to store host and VM information</td>
</tr>
<tr>
<td><strong>Scalability</strong></td>
<td>• Tested in the management of medium scale infrastructures with</td>
</tr>
<tr>
<td></td>
<td>hundreds of servers and VMs (no scalability issues has been reported)</td>
</tr>
<tr>
<td><strong>Installation</strong></td>
<td>• Installation on a UNIX cluster front-end without requiring new services</td>
</tr>
<tr>
<td></td>
<td>• Distributed in Ubuntu, Debian and OpenSuse</td>
</tr>
<tr>
<td>**Flexibility and</td>
<td>• Open, flexible and extensible architecture, interfaces and components,</td>
</tr>
<tr>
<td>Extensibility**</td>
<td>allowing its integration with any product or tool</td>
</tr>
</tbody>
</table>
# Comparison with Similar Technologies*

<table>
<thead>
<tr>
<th>Feature</th>
<th>Platform ISF</th>
<th>VMware Vsphere</th>
<th>Eucalyptus</th>
<th>Nimbus</th>
<th>OpenNebula</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Virtualization Management</strong></td>
<td>VMware, Xen</td>
<td>VMware</td>
<td>Xen, KVM</td>
<td>Xen</td>
<td>Xen, KVM, VMware</td>
</tr>
<tr>
<td><strong>Virtual Network Management</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Image Management</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Service Contextualization</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Scheduling</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Administration Interface</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Hybrid Cloud Computing</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Cloud Interfaces</strong></td>
<td>No</td>
<td>vCloud</td>
<td>EC2</td>
<td>WSRF, EC2</td>
<td>EC2 Query, OGF OCCI</td>
</tr>
<tr>
<td><strong>Flexibility and Extensibility</strong></td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Open Source</strong></td>
<td>No</td>
<td>No</td>
<td>GPL</td>
<td>Apache</td>
<td>Apache</td>
</tr>
</tbody>
</table>

* This comparison represents the features of the mentioned technologies in 2009.
INSIDE OPENNEBULA
OpenNebula Architecture

Core:
- SQL Pool
- Request Manager (XML-RPC)
- VM Manager
- Host Manager
- VN Manager

Drivers:
- Transfer Driver
- Virtual Machine Driver
- Information Driver

Tools:
- Scheduler
- Command Line Interface
- Other Tools
The Core

- Request manager: Provides a XML-RPC interface to manage and get information about ONE entities.
- SQL Pool: Database that holds the state of ONE entities.
- VM Manager (virtual machine): Takes care of the VM life cycle.
- Host Manager: Holds handling information about hosts.
- VN Manager (virtual network): This component is in charge of generating MAC and IP addresses.
• **Scheduler:**
  - Searches for physical hosts to deploy newly defined VMs

• **Command Line Interface:**
  - Commands to manage OpenNebula.
    - `onevm`: Virtual Machines
      - create, list, migrate…
    - `onehost`: Hosts
      - create, list, disable…
    - `onevnet`: Virtual Networks
The drivers layer

- **Transfer Driver**: Takes care of the images.
  - cloning, deleting, creating swap image...
- **Virtual Machine Driver**: Manager of the lifecycle of a virtual machine
  - deploy, shutdown, poll, migrate...
- **Information Driver**: Executes scripts in physical hosts to gather information about them
  - total memory, free memory, total cpus, cpu consumed...
Scheduler is a separated process, just like command line interface. Drivers are also separated processes using a simple text messaging protocol to communicate with OpenNebula Core Daemon (oned).
CONSTRUCTING A PRIVATE CLOUD
System Overview

- Executes the OpenNebula Services
- Usually acts as a classical cluster front-end
- Repository of VM images
- Multiple backends (LVM, iSCSI..)
- Modular components to interact with the cluster services
- Types: storage, monitoring, virtualization and network
- Provides physical resources to VMs
- Must have a hypervisor installed

The same host can be a front-end and a node
## Complex Storage behind OpenNebula

### Datastores (SAN/NAS)
- System datastore

### Service Network
- OpenNebula front-end

<table>
<thead>
<tr>
<th>Datastore</th>
<th>Transfer Manager Drivers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>shared</td>
</tr>
<tr>
<td>System</td>
<td>OK</td>
</tr>
<tr>
<td>File-System</td>
<td>OK</td>
</tr>
<tr>
<td>iSCSI</td>
<td></td>
</tr>
<tr>
<td>VMware</td>
<td>OK</td>
</tr>
</tbody>
</table>

Virtual machines and their images are represented as files.

Virtual machines and their images are represented as block devices (just like a disk).
System Datastore with Shared Transfer Manager Driver

Storage

MTA SZTAKI LPDS

ONED
$ONE_LOCATION/var

Shared FS

VM_DIR
CLUSTER NODE

System Datastore
Image Repository

Shared FS
Preparing the storage for a simple private cloud

- **Image Repository (system datastore):** Any storage medium for the VM images (usually a high performing SAN)
  - OpenNebula supports multiple transfer manager drivers
  - The front-end must have access to the repository
- **VM Directory:** The home of the VM in the cluster node
  - Stores checkpoints, description files and VM disks
  - Actual operations over the VM directory depends on the storage medium
  - Should be shared for live-migrations
  - You can go on without a shared FS and use the SSH transfer manager driver
  - Defaults to $ONE_LOCATION/var/$VM_ID

⚠️ **Dimensioning the Storage...** Example: A 64 core cluster will typically run around 80 VMs, each VM will require an average of 10GB of disk space. So you will need ~800GB for /srv/cloud/one, you will also want to store 10-15 master images so ~200GB for /srv/cloud/images. A 1TB /srv/cloud will be enough for this example setup.
Networking for private clouds

- OpenNebula management operations use ssh connections
- *Image traffic*, may require the movement of heavy files (VM images, checkpoints). Dedicated storage links may be a good idea
- *VM demands*, consider the typical requirements of your VMs. Several NICs to support the VM traffic may be a good idea
- OpenNebula relies on bridge networking for the VMs
Example network setup in a private cloud

Virtual LAN – Red (Ranged)

Virtual LAN – Blue (Ranged)

Virtual LAN – Public (Fixed)

Network

Host A

Bridge

02:01:0A:00:01:01
10.0.1.1/24

VM

02:01:93:60:51:f1
147.96.81.241/24

Bridge

Host B

Bridge

02:01:0A:00:02:01
10.0.2.1/24

VM

02:01:0A:00:02:02
10.0.2.2/24

VM

02:01:0A:00:01:03
10.0.1.3/24

VM

Switch

Internet
Virtual Networks

- A Virtual Network in OpenNebula
  - Defines a separated MAC/IP address space to be used by VMs
  - Each virtual network is associated with a physical network through a bridge
  - Virtual Networks can be isolated (at layer 2 level) with `ebtables` and `hooks`
- Virtual Networks are managed with the `onevnet` utility
Users

• A User in OpenNebula
  – Is a pair of username:password

• Only oneadmin can add/delete users

• Users are managed with the oneuser utility
User Management

• Native user support since v1.4
  – oneadmin: privileged account

• Usage, management, administrative rights for:
  – Templates, VMs, Images, Virtual Networks

• Through ACLs further operations/rights are available:
  – Rights for users, groups, datastores and clusters
  – Creation operation

• SHA1 passwords (+AA module)
  – Stored in FS
  – Alternatively in environment
Configuration

- $ONE_LOCATION/etc/oned.conf
  - General configuration
  - Defines the drivers used in the private cloud

- Match-making scheduler (default)
  - Placement policies configured per VM

- $ONE_LOCATION/etc/im_/im_.conf
  - Defines monitoring probes

- $ONE_LOCATION/etc/vmm_/vmm_.conf
  - Defaults values for the hypervisor

- $ONE_LOCATION/etc/tm_/tm_.conf
  - Defines action for generic storage operations
VIRTUAL MACHINES
Preparing VMs for OpenNebula

- Virtual Machines are managed with the oneuser utility
- You can use any VM prepared for the target hypervisor
- **Hint I:** Place the vmcontext.sh script in the boot process to make better use of vlans
- **Hint II:** Do not pack useless information in the VM images:
  - swap. OpenNebula can create swap partitions on-the-fly in the target host
  - Scratch or volatile storage. OpenNebula can create plain FS on-the-fly in the target host
- **Hint III:** Install once and deploy many; prepare master images
- **Hint IV:** Do not put private information (e.g. ssh keys) in the master images, use the CONTEXT
- **Hint V:** Pass arbitrary data to a master image using CONTEXT
### VM Description

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>Name that the VM will get for description purposes.</td>
</tr>
<tr>
<td>CPU</td>
<td>Percentage of CPU divided by 100 required for the Virtual Machine.</td>
</tr>
<tr>
<td>OS (KERNEL, INITRD)</td>
<td>Path of the kernel and initrd files to boot from.</td>
</tr>
<tr>
<td>DISK (SOURCE, TARGET, CLONE, TYPE)</td>
<td>Description of a disk image to attach to the VM.</td>
</tr>
<tr>
<td>NIC (NETWORK)</td>
<td>Definition of a virtual network the VM will be attached to.</td>
</tr>
</tbody>
</table>

- Multiple disk and network interfaces can be specified just adding more disk/nic statements.
- To create swap images you can specify TYPE=swap, SIZE=<size in MB>.
- By default disk images are cloned, if you do not want that to happen CLONE=no can be specified and the VM will attach the original image.
VM States overview

- PENDING
- PROLOG
- BOOT
- RUNNING
- SHUTDOWN
- EPILOG
- DONE
- SUSPENDED
- STOPPED
- HOLD
- MIGRATE
Pending state

• After submitting a VM description to ONE it is added to the database and its state is set to PENDING.
• In this state IP and MAC addresses are also chosen if they are not explicitly defined.
• The scheduler awakes every 30 seconds and looks for VM descriptions in PENDING state and searches for a physical node that meets its requirements. Then a deploy XML-RPC message is sent to oned to make it run in the selected node.
• Deployment can be also made manually using the Command Line Interface:

⇒ onevm deploy <vmid> <hostid>
In PROLOG state the Transfer Driver prepares the images to be used by the VM.

Transfer actions:

- **CLONE**: Makes a copy of a disk image file to be used by the VM. If Clone option for that file is set to false and the Transfer Driver is configured for NFS then a symbolic link is created.
- **MKSWAP**: Creates a swap disk image on the fly to be used by the VM if it is specified in the VM description.
Boot state

- In this state a deployment file specific for the virtualization technology configured for the physical host is generated using the information provided in the VM description file. Then Virtual Machine Driver sends deploy command to the virtual host to start the VM.
- The VM will be in this state until deployment finishes or fails.
The ISO image has the contextualization for that VM:

- **context.sh**: contains configuration variables
- **init.sh**: script called by VM at start to configure specific services
- **certificates**: directory that contains certificates for some service
- **service.conf**: service configuration

Requirements against the VM:

- Should be prepared to use the contextualization ISO image.
- Should mount the ISO image at boot time.
- After boot it should use the scripts on the ISO image to make use of the information provided.

User provided, OpenNebula provided contextualization info
Running and Shutdown states

- While the VM is in RUNNING state it will be periodically polled to get its consumption and state.
- In SHUTDOWN state Virtual Machine Driver will send the shutdown command to the underlying virtual infrastructure.
• In EPILOG state the Transfer Manager Driver is called again to perform this actions:
  – Copy back the images that have `SAVE=yes` option.
  – Delete images that were cloned or generated by `MKSWAP`.
HYBRID CLOUD
Overview

- VMs can be local or remote
- VM connectivity has to be configured, usually VPNs

- External Clouds are like any other host
- Placement constraints
Making an Amazon EC2 hybrid

- Amazon EC2 cloud is managed by OpenNebula as any other cluster node
  - You can use several accounts by adding a driver for each account (use the arguments attribute, -k and -c options). Then create a host that uses the driver
  - You can use multiple EC2 zones, add a driver for each zone (use the arguments attribute, -u option), and a host that uses that driver
  - You can limit the use of EC2 instances by modifying the IM file
Using an EC2 hybrid cloud

- Virtual Machines can be instantiated locally or in EC2
- The VM template must provide a description for both instantiation methods.
- The EC2 counterpart of your VM (AMI_ID) must be available for the driver account
- The EC2 VM template attribute should describe not only the VM’s properties but the contact details of the external cloud provider
Local private network

PHYSICAL NODE

Bridge

Worker Node

Worker Node

VPN Tunnels

Internet Connection

PHYSICAL NODE

Bridge

Front-End

PHYSICAL NODE

Bridge

Worker Node

Worker Node

PHYSICAL NODE

Bridge

Worker Node

Worker Node

Service perspective
Hybrid cloud Use Cases

On-demand Scaling of Computing Clusters
- Elastic execution of a SGE computing cluster
- Dynamic growth of the number of worker nodes to meet demands using EC2
- Private network with NIS and NFS
- EC2 worker nodes connect via VPN

On-demand Scaling of Web Servers
- Elastic execution of the NGinx web server
- The capacity of the elastic web application can be dynamically increased or decreased by adding or removing NGinx instances
Eucalyptus
Eucalpytus

CLC: cloud controller
CC: cluster controller
NC: node controller
Walrus: storage controller similar to S3

Keke Chen
Introduction

• Eucalyptus is an open source software mimicking AWS
  – How VM instances are managed
  – How to provide virtual network (like elastic IP)
  – How to provide data storage (like S3)
  – A very brief description, but we can get something

• Use the same user interface AWS provided (EC2, S3, mainly)
  – Boto library can be used, too
Components: Node Controller

• Make queries to discover physical resources
  – # of cores
  – Size of memory
  – Available disk space
  – State of VM instances

• Propagate the information to Cluster Controller
  – DescribeResource
  – DescribeInstances

• Run/terminate instances
  – CLC→CC→ NC → hypervisor (Xen)
Node controller

- Start an instance
  - Copy instance image from walrus or local cache
  - Create endpoint in the virtual network overlay
  - Instruct hypervisor to boot the instance

- Stop an instance
  - Instruct hypervisor to terminate the VM
  - Tear down the virtual network endpoint
  - Clean up the files associated with the instance
Cluster Controller

• Gather/report information of NCs
  – Through the interface provided by NCs
  – Report the summary to CLC
• Schedule incoming instance “run” requests to specific NCs
• Control the virtual network overlay
Virtual network overlay

- VM instance interconnectivity (between different nodes/networks)
  - Not very well mentioned in Xen
  - Connectivity, isolation and performance

- At least one of a set of VMs be exposed externally
  - Map the public IP to that instance
  - The owner can login as the root

- Restricted communication
  - VMs in the same set can talk to each other
  - VMs from different sets should be isolated

- Performance

Keke Chen
Virtual network overlay

- Each VM has a private IP; one VM in the set also has a public IP
- VLAN tag defines the subnet – to isolate sets of VMs
- Cluster Controller serves as the router between VM subnets
  - CC uses Linux iptable control traffics
  - Use iptable Network Address Translation (NAT) to define the map from Public IP to private IP
Virtual network overlay

- Example of using iptables packet filtering
Virtual network overlay

- VMs on different networks (same cluster)
  - CC as IP router – extra hop
- VMs on different clusters
  - Tunnel (VTUN)
Storage Controller (Walrus)

- Provide SOAP/REST interfaces
  - Compatible with S3 – you can use S3 tools
- Use Walrus to stream data in/out of the cloud
- Store VM images (same as AMI)
  - Root file system, kernel image, ramdisk image
- No locking for object writes
  - Conflict writes – late write overwrites the earlier
• Provides the same tool Amazon uses
  – Generate AMI
• Maintains a cache of images
• Authentication is applied when NC accesses images
Cloud Controller

• A collection of web services
  – Resource services
  – Data services
  – Interface services
Cloud Controller: resource services

• Receive user requests
• Interact with CCs to allocate/deallocate
• System Resource State (SRS) is maintained by querying CCs
  – CCs will collect information from NCs
• Follows a “transactional” operation
  – Reservation, VM creation → commit
  – Or errors → rollback
• Realizing SLAs

Keke Chen
Cloud Controller: data services

• Handles the creation, modification, interrogation, and storage of stateful system and user data
  – There is a system database...

• Users can query the services
  – Discover resource info (images, clusters)
  – Manipulate abstract parameters (keypairs, security groups, network definitions)
  – Recall some of AWS interfaces...
Cloud Controller: interface services

- User-visible interfaces
  - Programmatic interfaces (SOAP/REST)
  - Web interface

- Handling authentication
- Provide system management tools
Basic euca2ools

- euca-describe-images
- euca-describe-instances
- euca-run-instances
- euca-create-volume
- euca-attach-volume
- euca-terminate-instances
- euca-describe-availability-zones
“Workspaces”

• Dynamically provisioned environments
  – Environment control
  – Resource control
• Hardware implementations vs virtualization
A Brief History of Nimbus

- Xen released
- EC2 goes online
- Nimbus Cloud comes online
- STAR production runs on EC2

2003
- Research on agreement-based services

2006
- First Workspace Service release
- EC2 gateway available

2009
- Support for EC2 interfaces

The Nimbus Toolkit: http://workspace.globus.org
Kate Keahey (keahey@mcs.anl.gov)
Nimbus Overview

• Goal: open source, extensible, IaaS implementation and tools
  – Specifically targeting scientific community
  – A platform for experimentation with features for scientific needs
  – Set up private clouds (privacy, expense considerations)

• Tools
  – IaaS layer (Workspace Service)
  – Orchestration layer (Context Broker, gateway)

• http://workspace.globus.org/
The Workspace Service

The Workspace Service (VWS) is a service that provides a workspace for users to collaborate and work on projects. It is part of the Nimbus Toolkit, which can be found at http://workspace.globus.org. Kate Keahey, who is associated with the service, can be reached at keahey@mcs.anl.gov.
The workspace service publishes information on each workspace as standard WSRF Resource Properties. Users can query those properties to find out information about their workspace (e.g., what IP the workspace was bound to). Users can interact directly with their workspaces the same way they would with a physical machine.
Workspace Service Interfaces and Clients

- Web Services based
- Web Service Resource Framework (WSRF)
  - GT-based
- Elastic Computing Cloud (EC2)
  - Supported: ec2-describe-images, ec2-run-instances, ec2-describe-instances, ec2-terminate-instances, ec2-reboot-instances, ec2-add-keypair, ec2-delete-keypair
  - Unsupported: availability zones, security groups, elastic IP assignment, REST
- Used alongside WSRF interfaces
  - E.g., the University of Chicago cloud allows you to connect via the cloud client or via the EC2 client
Security

• GSI authentication and authorization
  – PKI credential required
  – Works with Grid proxies
  – VOMS, Shibboleth (via GridShib), custom PDPs

• Secure access to VMs
  – EC2 key generation or accessed from .ssh

• Validating images and image data
  – Collaboration with Vienna University of Technology
Networking

• Network configuration
  – External: public IPs or private IPs (via VPN)
  – Internal: private network via a local cluster network
• Each VM can specify multiple NICs mixing private and public networks (WSRF only)
  – E.g., cluster worker nodes on a private network, headnode on both public and private network
Workspace **WSRF front-end** that allows clients to deploy and manage virtual workspaces.

**Workspace back-end:**

**Resource manager** for a pool of physical nodes. Deploys and manages Workspaces on the nodes.

Each node must have a VMM (Xen) installed, as well as the **workspace control** program that manages individual nodes.

**Trusted Computing Base (TCB)**
Workspace Components

- EC2
- WSRF
- workspace service
- workspace client
- workspace resource manager
- workspace pilot
- workspace control

Kate Keahey (keahey@mcs.anl.gov)
Workspace Control

• VM image propagation
• Image management and reconstruction
  – Creating blank partitions, sharing partitions
• VM control
  – Starting, stopping, pausing, etc.
• Integrating a VM into the network
  – Assigning MAC addresses and IP addresses
  – DHCP delivery tool
  – Building up a trusted (non-spoofable) networking layer
• Contextualization information management
• Talks to the workspace service via ssh
• Standalone component
• Some functionality overlap with libvirt
• Implementations in Xen and KVM (queued up for release)
The Workspace Resource Manager

• Basic slot fitting
• Implements “immediate leases”
• Extensible vehicle to experiment with different leases
• Open source resource manager for multiple different VMMs
• Datacenter technology equivalent
  – Can be replaced by OpenNebula or other datacenter technologies
• Deployment
  – University of Chicago, University of Florida, Purdue, Masaryk University and all the other Science Cloud sites
The Workspace Pilot

• Challenge: how can I provide a virtualization solution without disrupting the current operation of my cluster?

• Flying Low: the Workspace Pilot
  – Integrates with popular LRMIs (such as PBS, SGE)
  – Implements “best effort” leases
  – Glidein approach: submits a “pilot” program that claims a resource slot
  – Includes administrator tools

• Deployment
  – Testing @ U of Victoria (Atlas), Ian Gable and collaborators
  – Adapting for the use of the Atlas experiment @ CERN, Omer Khalid
Cloud Closure

- Storage service
- Workspace service
- WSRF
- EC2
- Workspace resource manager
- Workspace pilot
- Workspace control
- Cloud client
- Workspace client

The Nimbus Toolkit: http://workspace.globus.org
Kate Keahey (keahey@mcs.anl.gov)
IaaS Gateway

- Goals
  - Access to different IaaS infrastructures
  - Account management
  - Facilitate movement between academic and commercial clouds and creation of meta-clouds
  - Combine higher-level tools and IaaS
- Released as service, not as code
- First online in June 2007, currently in a rewrite
- Used to move e.g., HEP STAR experiments between Science Clouds and EC2
The IaaS Gateway

- EC2 gateway
- potentially other providers
- workspace control
- workspace pilot
- workspace resource manager
- storage service
- workspace service
- IaaS gateway
- EC2
- cloud client
- workspace client

The Nimbus Toolkit: http://workspace.globus.org

Kate Keahey (keahey@mcs.anl.gov)
One-click Virtual Clusters

- Parameterizable appliance
- Tightly-coupled clusters

Reciprocal exchange of information: networking and security
Goals for Context Broker

• Can work with every appliance
  – Appliance schema, can be implemented in terms of many configuration systems

• Can work with every cloud provider
  – Simple and minimal conditions on generic context delivery

• Can work across multiple cloud providers, in a distributed environment
Status for Context Broker

• Release history:
  – In alpha testing since August ‘07
  – First released summer July ‘08 (v 1.3.3)
  – Latest update January ‘09 (v 2.2)
• Used to contextualize 100s of nodes for EC2 STAR runs
• Contextualized images on workspace marketplace
• Working with rPath to make contextualization easier for the user
End of Nimbus Tour

The Nimbus Toolkit: http://workspace.globus.org

Kate Keahey (keahey@mcs.anl.gov)
Science Clouds

- Make it easy for scientific projects to experiment with cloud computing
  - Can cloud computing be used for science?
- Evolve software in response to the needs of scientific projects
  - Start with EC2-like functionality and evolve to serve scientific projects: virtual clusters, diverse resource leases
  - Federating clouds: moving between cloud resources in academic and commercial space
Science Cloud Resources

- University of Chicago (Nimbus):
  - first cloud, online since March 4th 2008
  - 16 nodes of UC TeraPort cluster, public IPs
- University of Florida
  - Online since 05/08
  - 16-32 nodes, access via VPN
- Other Science Clouds
  - Masaryk University, Brno, Czech Republic (08/08), Purdue (09/08)
  - Installations in progress: IU, Grid5K, others
- Using EC2 for overflow
- Minimal governance model
- http://workspace.globus.org/clouds

The Nimbus Toolkit: http://workspace.globus.org
Kate Keahey (keahey@mcs.anl.gov)
Additional reading

