GridKa School 2011
Cloud Computing Workshop

Tobias Kurze, Viktor Mauch
Contents

- Cloud Computing
  - Concept
  - Everything as a Service (XaaS)
  - Private Cloud IaaS Frameworks

What’s behind all this??
Cloud Computing

Characteristics

- On-demand self-service
- Rapid Elasticity
- Broad Network Access
- Resource Pooling
- Measured Service

Resources

- SaaS
- PaaS
- IaaS

Delivery Models

Deployment Models

- Public
- Private
- Hybrid
Toward an Architectural Style for Cloud Computing: Five essential characteristics

1. **Rapid Elasticity**: the ability to scale resources both up and down as needed. To the consumer, the cloud appears to be infinite.

2. **Measured Service**: In a measured service, aspects of the cloud service are controlled and monitored by the cloud provider. This is crucial for billing, access control, resource optimization, capacity planning and other tasks.

3. **On-Demand Self-Service**: The on-demand and self-service aspects of cloud computing mean that a consumer can use cloud services as needed without any human interaction with the cloud provider.

4. **Broad Network Access**: the cloud provider’s capabilities are available over the network and can be accessed through standard mechanisms.

5. **Resource Pooling**: Resource pooling allows a cloud provider to serve its consumers via a multi-tenant model. Physical and virtual resources are assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided.

[NIST]
Concept of Cloud Computing – Organisatorical Types

- **Public Cloud**
  - Providers have commercial interests
  - Users have no costs concerning purchase, operation and maintenance of own hardware
  - Critical situation concerning data privacy and security of sensible information
  - Fear for a Lock-in situation!

- **Private Cloud**
  - Providers and users are from the same organization
  - No security or privacy issues
  - Similar operation costs like a non Cloud-based architecture
  - Lock-in situation cannot happen
  - Compatible with the popular public cloud services (in a perfect world!)

- **Hybrid Cloud**
  - Services of private and public clouds are combined to process load peaks or outsource data copies
Everything as a Service (XaaS)

1. Layer: Infrastructure as a Service (IaaS)
   - Users run virtual server instances with optional operations system configurations (restricted by the providers)
   - Administrative user rights
   - Own firewall rules
   - No direct contact to physical hardware for the user

2. Layer: Platform as a Service (PaaS)
   - Scalable running environment and (sometimes) development environment for 1 or 2 programming languages
   - No administrative effort concerning the operation environment
   - More restriction then in IaaS

3. Layer: Software as a Service (SaaS)
   - Applications a run by a provider
   - No need for a local installation at the users site
   - Users do not need to take care about installation, security updates, ...
   - Users need to trust the provider concerning the process of their data in the cloud (e.g. E-Mail accounts)

4. Layer: Human as a Service (HaaS)
   - Principle of crowd sourcing
   - Human creativity becomes available as a resource in the cloud
   - Interesting for tasks which are difficult to automate by computers (e.g.: translation, image recognition)
Overview of some Private Cloud IaaS Frameworks

- Lots of Private Cloud IaaS solutions available at first sight
  - All of them are Open Source!
- Already used in science projects
  - CERN uses a Cloud Environment with OpenNebula with the goal to manage up to 45,000 Virtual Machine instances (Bittorrent for VM deployment)

<table>
<thead>
<tr>
<th>Cloud.com CloudStack</th>
<th><a href="http://cloud.com">http://cloud.com</a></th>
<th>no storage; XEN; KVM, Vmware; EC2 compatible</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenStack</td>
<td><a href="http://www.openstack.org/">http://www.openstack.org/</a></td>
<td>Storage (Swift); XEN, KVM; EC2, S3 compatible</td>
</tr>
<tr>
<td>OpenNebula</td>
<td><a href="http://www.opennebula.org">http://www.opennebula.org</a></td>
<td>no storage; Xen, KVM, VMware; EC2 compatible</td>
</tr>
<tr>
<td>Nimbus</td>
<td><a href="http://www.nimbusproject.org">http://www.nimbusproject.org</a></td>
<td>Storage (Cumulus); XEN, KVM; EC2, S3 compatible</td>
</tr>
<tr>
<td>Eucalyptus</td>
<td><a href="http://open.eucalyptus.com">http://open.eucalyptus.com</a></td>
<td>Storage (Walrus); VMware, Xen, KVM; EC2, S3 compatible</td>
</tr>
</tbody>
</table>
OpenNebula – Introduction

- OpenNebula is an open-source toolkit to easily build any type of cloud: **private, public** and **hybrid**.
- OpenNebula supports **KVM, Xen and VMware**
- OpenNebula has been designed to be integrated with any networking and storage solution and so to fit into any existing data center.

- Only a small part of the EC2 API implemented since OpenNebula 2.0
  - describe images
  - describe, run, reboot and terminate instances
- Trivial architecture
  - Easy to implement additional features
  - Easy to debug because of central log data
- Nodes can be grouped, Important for HPCaaS and network latency (e.g. MPI)
- No storage service included
OpenNebula – Structure Notes

- Installation:
  - Documentation available for Ubuntu, CentOS, Debian, OpenSUSE, MacOS, ...
  - see: http://opennebula.org/documentation:documentation

- Structure:
  - Separation in Front-End and Cluster Nodes
  - Communication based on SSH (password-less login via SSH keys), XML-RPC protocol and Ruby scripts
  - Front-End uses the libvirt library to control the Hypervisor on the Cluster Nodes via SSH
  - To provide one or more physical networks for the VMs, the cluster nodes have to be set up with Ethernet Bridges

- Two operation methods for VM Deployment:
  - via SSH
    - Images are copied via SSH to the Cluster Node partitions
  - on a Shared File System
    - Live Migration is possible
    - FS should be performant enough to manage high I/O -> SAN mount
OpenNebula – Private Cloud Tutorial Instance

- 6x Dell Blades - Dual Intel Xeon Quad Core 2.66 GHz / 16 GB Ram:
  1 Front-End + 5 Cluster Nodes (40 Cores)
- Connection: 1 Gigabit Ethernet
- Image Deployment via SSH
- Based on Ubuntu 11.04/11.10 Server
- Virtualization Technology: KVM Hypervisor
- Version: OpenNebula 3.0 Beta1
- Installation can be found under /srv/cloud/one on the front-end

Deployment via SSH

Shared File System
Exploring the Private Cloud

Hands on... explore the Cloud with some basic OpenNebula commands:

<table>
<thead>
<tr>
<th>Category</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Management</td>
<td>onehost</td>
<td>&lt;list top show create delete enable disable ...&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check out how many cluster nodes are available with <code>onehost list</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Explore the details of one cluster node with <code>onehost show host_id</code>.</td>
</tr>
<tr>
<td>Network Management</td>
<td>onevnet</td>
<td>&lt;list show create delete ...&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check out which virtual networks are available with <code>onevnet list</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Explore the details of one virtual network with <code>onevnet show vnet_id</code>.</td>
</tr>
<tr>
<td>Machine Management</td>
<td>onevm</td>
<td>&lt;create delete migrate supend resume ...&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check out how many virtual machines are running with <code>onevm list</code> or <code>onevm top</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Explore the details of one virtual machine with <code>onevm show vm_id</code>.</td>
</tr>
<tr>
<td>Management</td>
<td>oneimage</td>
<td>&lt;list show ...&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check out how many images are available with <code>oneimage list</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Explore the details of one image with <code>oneimage show image_id</code>.</td>
</tr>
<tr>
<td>User Management</td>
<td>oneuser</td>
<td>&lt;create delete list ...&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Only available for the cloud admin to create and delete cloud users.</td>
</tr>
<tr>
<td>Group Management</td>
<td>onegroup</td>
<td>&lt;create delete list ...&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Only available for the cloud admin to create and delete cloud groups.</td>
</tr>
<tr>
<td>Template Management</td>
<td>onetemplate</td>
<td>&lt;create delete list ...&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>With this command you will define your VM templates for the next exercises.</td>
</tr>
</tbody>
</table>

Steinbuch Centre for Computing
A Virtual Network in OpenNebula
- Defines a MAC/IP address space to be used by VMs
- Each Virtual Network is associated with a physical network through a bridge

Virtual Network definition
- **Name** of the Network
- **Type**
  - **Fixed**, a set of IP/MAC leases
  - **Ranged**, defines a network range
- **Bridge**, name of the physical bridge in the physical host where the VM should connect its network interface

### # Ranged VNET template file
```
NAME = "Red LAN"
TYPE = RANGED
BRIDGE = eth0
NETWORK_SIZE = C
NETWORK_ADDRESS = 192.168.169.0
```

### # Fixed VNET template file
```
NAME = "Blue LAN"
TYPE = FIXED
BRIDGE = br0
LEASES = [IP=192.168.170.11]
LEASES = [IP=192.168.170.12]
LEASES = [IP=192.168.170.13]
```

**Hands on...** create your own fixed Virtual Network with two IPs.
Virtual Networks II

How to use a Virtual Network with your VMs

- Define NICs attached to a given virtual network. The VM will get a NIC with a free MAC address in the network and attached to the corresponding bridge

```sh
# A VM with two interfaces each one in a different vlan
NIC = [NETWORK_ID = X]
NIC = [NETWORK_ID = Y]

# Ask for a specific IP/MAC
NIC = [NETWORK_ID = X, IP = 192.168.0.11]
```

- Prepare the VM to use the IP. Sample scripts to set the IP based on the MAC are provided for several Linux distributions.

### IP-MAC address correspondence

- IP: 192.168.170.101
- MAC: 02:00:C0:A8:AA:65

oned.conf

IP address
Virtual Machines I

Preparing a VM to be used with OpenNebula
- You can use any VM prepared for the target hypervisor
- Prepare master images: Install once and deploy many;
- Do not put private information (e.g. ssh keys) in the master images, instead use **CONTEXT** (see later)
- Pass arbitrary data to a master image using **CONTEXT**

Virtual Machine Life-cycle:
Virtual Machines II

- Virtual Machines are defined in a VM template file
- Each VM has an unique ID in OpenNebula, the VM_ID
- All log files are stored in /srv/cloud/one/var/<VM_ID> on the head node (after life)
- The images will be copied via a SSH connect to the cluster nodes

A Virtual Machine template in OpenNebula consists of
- a capacity section in terms of name, memory and cpu
- a set of NICs attached to one or more virtual networks
- a set of disk images, to be "transferred" to/from the execution host
- ...
# Capacity Section

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>&quot;name that the VM will get for description purposes&quot;</td>
</tr>
<tr>
<td>CPU</td>
<td>&quot;percentage of CPU divided by 100 required for the Virtual Machine&quot;</td>
</tr>
<tr>
<td>MEMORY</td>
<td>&quot;amount of requested MEM&quot;</td>
</tr>
<tr>
<td>VCPU</td>
<td>&quot;number of virtual cpus&quot;</td>
</tr>
</tbody>
</table>

# OS and boot options

```python
OS = {
    'arch': "CPU architecture to virtualization",
    'kernel': "path to os kernel",
    'initrd': "path to initrd image",
    'kernel_cmd': "kernel command line",
    'root': "device to be mounted as root",
    'bootloader': "path to the boot loader exec",
    'boot': "device to boot from"
}
```

# Features of the hypervisor

```python
FEATURES = {
    'pae': "yes|no",
    'acpi': "yes|no"
}
```
# VM Disks

DISK = [
    image_id = "id of the image managed by ONE",
    type = "image|floppy|disk|cdrom|swap|fs|block",
    source = "path to disk image file|physical dev",
    format = "type for fs disks",
    size = "size_in_GB",
    target = "device to map disk",
    bus = "ide|scsi|virtio|xen",
    readonly = "yes|no",
    clone = "yes|no",
    save = "yes|no" ]

# Network Interface

NIC = [
    network_id = "id of the virtual network managed by ONE",
    target = "device name to map if",
    ip = "ip address",
    bridge = "name of bridge to bind if",
    mac = "HW address",
    script = "path to script to bring up if",
    model = "NIC model" ]
Virtual Machine Definition File 3.0 (VM template) III

# I/O Interfaces
INPUT = [
    type   = "mouse|tablet",
    bus    = "usb|ps2|xen"
]
GRAPHICS = [
    type   = "vnc|sdl",
    listen = "IP to listen on",
    port   = "port for VNC server",
    passwd = "password for VNC server",
    keymap = "keyboard configuration locale to use in the VNC display"
]

# RAW Hypervisor attributes
RAW = [
    type   = "xen|kvm",
    data   = "raw domain configuration"
]

# CONTEXT Section used for Customization of VMs
CONTEXT = [ ... ]  # see later

# Placement Section
REQUIREMENTS = "Boolean expression that rules out provisioning hosts form list"
RANK = "Attribute which will be used to sort the suitable hosts for VM"

Complete reference and examples for all sections:
http://www.opennebula.org/documentation:rel3.0:template
Hands on... define a minimal VM template and create your first VM:

```bash
# VM template for the ttylinux image
NAME = my_test_vm # define a name for your VM
MEMORY = 128
DISK = [ IMAGE_ID = X, BUS = ide ] # enter the image ID of ttylinux
NIC = [ NETWORK_ID = Y ] # enter the id of your created vnet
OS = [ ARCH = x86_64, BOOT = hd ]
```

- Submit your VM template: `onetermin create vm_template_file`
- Instantiate one VM instance: `oneterminstantiate vm_template_id`
- Monitor the status for your VM: `onevm list`
- Get detailed information, (e.g. IP): `onevm show VM_ID`
- Try to perform some VM operation: `onevm <migrate|suspend|resume|delete|...>`
- Try to login (User: “root”, PW: “password”): `ssh root@VM_IP`
- Take a look to the script file `/etc/rc.d/init.d/vmcontexttty` within the VM, which is part of the boot procedure and try to understand how the network will be configured
- Optional: Modify the template: create - on the fly – another empty DISK, e.g.: `TYPE=fs, FORMAT=ext2, SIZE=100, TARGET=hdb` and try to mount it within the VM
Customization of VMs

ONE provides a method to modify created VMs. The master image `ubuntu` is already preconfigured to support the CONTEXT Block:

- The ISO Image will be mounted under `/mnt/context`
- The `init.sh` script will be executed with root privileges
- Afterwards the ISO Image will be un-mounted

Hands on... define a VM template for the `Ubuntu` Image and try to use the CONTEXT Block (see Handout).
Performing some Rendering Jobs

- high quality pictures can be rendered using ray tracing
- the rendering can be done in parallel
  - regions of a picture
  - single frames of an animation
- POV-Ray is open source

Hands on... define a new CONTEXT section for the Ubuntu Image to perform a rendering job. Divide the complete rendering procedure of the pictures in 2 parts:
- First VM: 0..49
- Second VM: 50..99

See handout!!
How to copy the results from the headnode

- **Linux:**
  - `scp -P 24 USER@141.52.174.199:~/render/results/movie.avi`

- **Windows:**
  - Download WinSCP from [http://winscp.net](http://winscp.net) and install it
  - The Remote address is 141.52.174.199
  - Port number is 24
  - Protocol is SFTP with SCP fallback or SCP directly
  - Navigate to `~/render/results`
  - Copy the file `movie.avi` to your PC
OpenNebula Sunstone

- Browser GUI available
- Open your browser an go to:
  http://141.52.174.199:2222

- Provides the full functionality of ONE
- Still some bugs concerning the consistency with the CLI commands
- Future releases will provide VNC connections to the VMs
Further Feature

- **Hybrid Cloud:**
  - Provides the possibility to control AWS / ElasticHosts resources with the same basic ONE commands
  - Creates a simple abstraction layer over the EC2-API-Tools
  - However there is no simple way to deploy own images to AWS / ElasticHosts

- **Public Cloud:**
  - Extension of a Private Cloud to expose RESTful Cloud interfaces
  - Can be added to your Private or Hybrid Cloud if you want to provide partners or external users with access to your infrastructure

- **EC2 Compatible Management:**
  - Since ONE 2.0 there is the possibility to control ONE resources via EC2 compatible GUI tools, like
    - HybridFox / ElasticFox (Firefox Plug-Ins)
    - KOALA (PaaS Browser Service-
Thank You!

Links:

- OpenNebula Website: http://opennebula.org/