OSDC 2012

24th April, Nürnberg

Building Clouds with OpenNebula 3.4

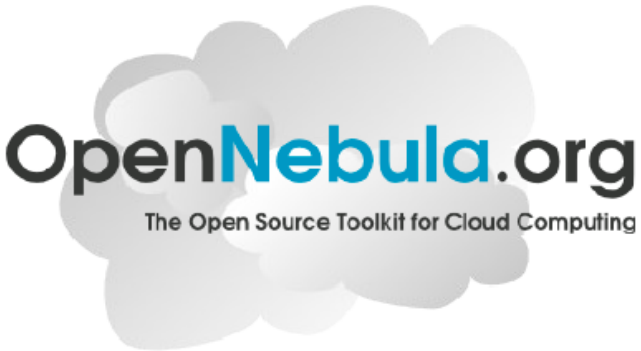
Constantino Vázquez Blanco

dsa-research.org | OpenNebula.org

Distributed Systems Architecture Research Group
Universidad Complutense de Madrid

Building Clouds with OpenNebula 3.4
Basic Usage of the Private Cloud

Constantino Vázquez Blanco
dsa-research.org | OpenNebula.org
Distributed Systems Architecture Research Group
Universidad Complutense de Madrid



- Virtual Networks
- Images
- Virtual Machines
- Templates

Virtual Networks

Overview

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

Virtual Networks

Example, create and manage Virtual Networks

- Define and create two networks

```
$ vi red.net
NAME = "Red LAN"
TYPE = RANGED
BRIDGE = br0
NETWORK_SIZE    = C
NETWORK_ADDRESS = 192.168.XX.0

$ vi blue.net
NAME = "Blue LAN"
TYPE = FIXED
BRIDGE = br0
LEASES = [IP=192.168.YY.5]
LEASES = [IP=192.168.YY.10]
LEASES = [IP=192.168.YY.15]
LEASES = [IP=192.168.YY.20]
LEASES = [IP=192.168.YY.25]
```

Virtual Networks

Example, create and manage Virtual Networks

- Use the onevnet command to list and show networks
- Modify the fixed network to add/remove leases with the *addleases* and *rmleases* option
- Leases can be public or private to the user, check and modify the network status

Virtual Networks

Using Virtual Networks within your VMs

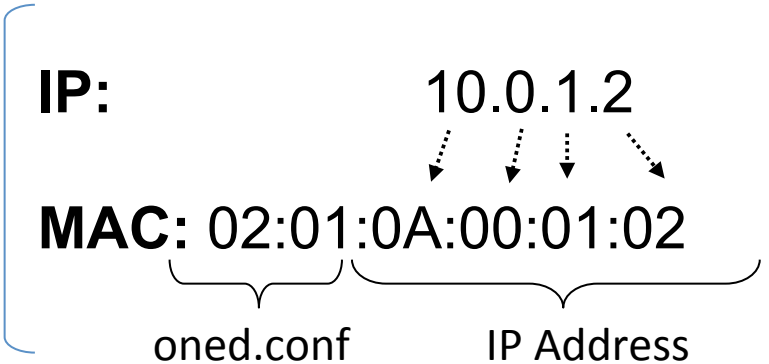
- **Define NICs** attached to a given virtual network. The VM will get a NIC with a free MAC in the network and attached to the bridge

```
#A VM with two interfaces each one in a different vlan
NIC=[NETWORK="Blue LAN"]
NIC=[NETWORK="Red LAN"]

#Ask for a specific IP/MAC of the Red vlan
NIC=[NETWORK="Red LAN", IP=192.168.0.3]
```

- **Prepare the VM** to use the IP. Sample scripts to set the IP based on the MAC are provided.

IP to MAC correspondence



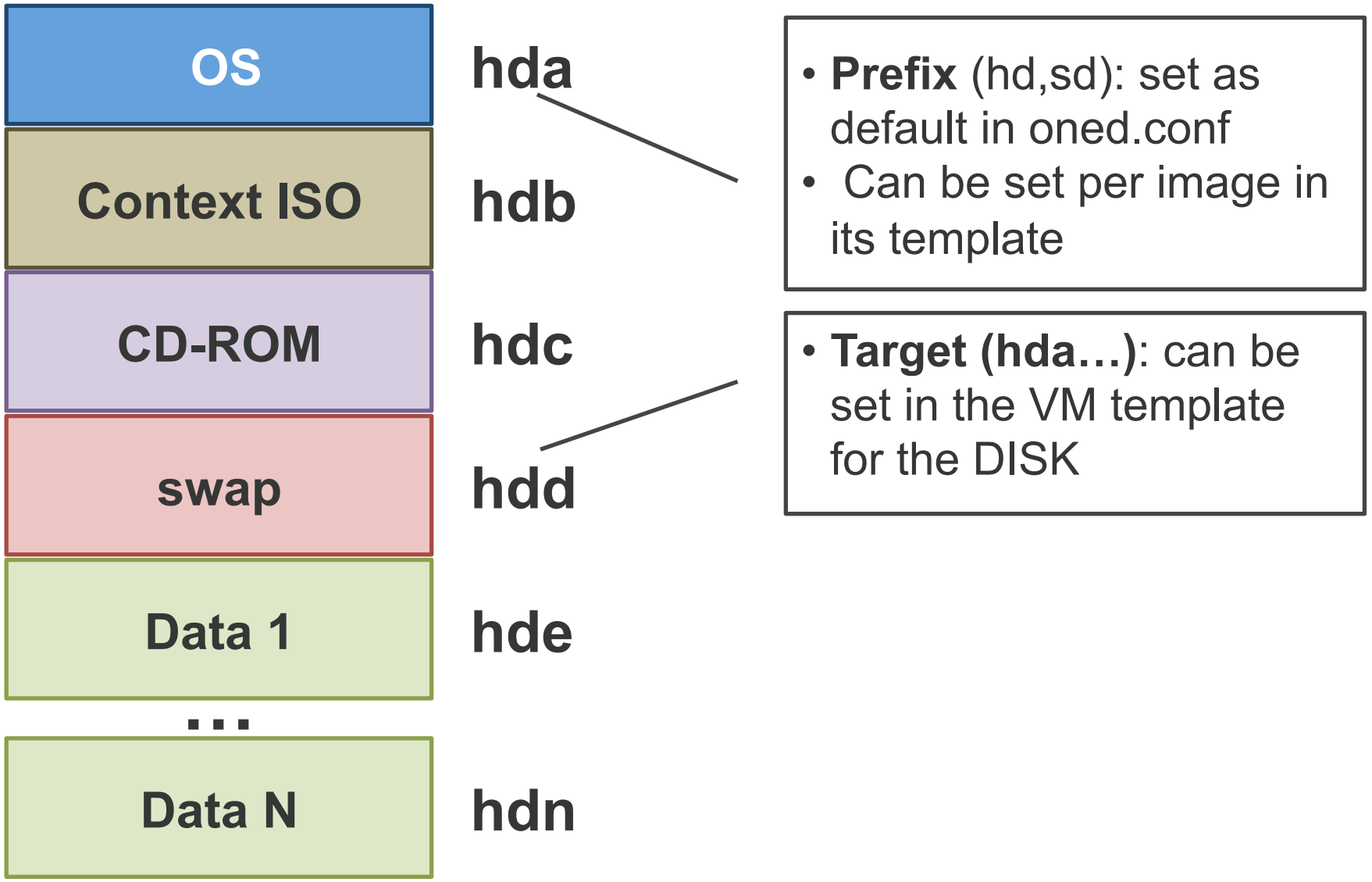
Images

Overview

- An **Image** in OpenNebula's repository
  - A virtual machine disk to be used as OS or DATA device.
  - Defined **within** a datastore
  - Images can be **presistent** and/or **public**
  - Images modifications can be saved as another image
- **Image Types**:
  - **OS**: contains a working operative system
  - **CDROM**: readonly data
  - **DATABLOCK**: A storage for data. Can be created either from previous existing data, or as an empty drive.

Images

Automatic Disk Layout for Images



Images

Defining a Virtual Machine Disk Image

```
#-------------------------------------
# Name of the Image
#-------------------------------------
NAME = "vm-example" # Mandatory

#-------------------------------------
# Image Meta-Data
#-------------------------------------
TYPE        = OS | CDROM | DATABLOCK
DESCRIPTION = "of the contents of the Image"
PUBLIC      = YES | NO
PERSISTENT  = YES | NO

#-------------------------------------
# VM Attach attributes
#-------------------------------------
DEV_PREFIX = "to generate disk targets"
BUS        = "type of device to emulate (ide,scsi,virtio)"
```

Images

Defining a Virtual Machine Disk Image

```
#-----------------------------------
# Source of the Image (use just one)
#-----------------------------------
PATH   = "URL to copy the image to the repo"
SOURCE = "raw disk source (no copy)"

#-----------------------------------
# DATABLOCK generation (no path given)
#-----------------------------------
SIZE   = "for the data disk in MB"
FSTYPE = "to format the image"
```

Images

Example, Register Images

- Define and create two images

```
$ vi ttylinux.img
NAME = “ttylinux"
TYPE = OS
PUBLIC      = yes
DESCRIPTION = “ttylinux with context “
PATH        = <put_the_path_here>
PERSISTENT  = no

$ vi data.img
NAME = “data"
TYPE = DATABLOCK
DESCRIPTION = “user data”
PUBLIC     = no
PERSISTENT = yes
SIZE   = 100
FSTYPE = ext2
```

Images

Example, Register Images

- **Hands on!**
  - Check images with oneimage list and show
  - Change public and persistent attributes
  - Check the contents of the datatores

    /srv/cloud/one/var/datastores/1

Images

Using Images with your Virtual Machines

- **Define DISKs** attached to the virtual machine.
  - Select the image by name or id (IMAGE_ID **preferred**)
  - Overwrite attributes if needed (TARGET, BUS)
- **Prepare the VM** to use the disk layout to ease usage

```
# OS image, mapped to sda.
DISK = [ IMAGE = "Debian 5.0" ]

# First DATABLOCK image, mapped to sde
DISK = [ IMAGE_ID = 4 ]

# swap, sdd
DISK = [ TYPE = swap, SIZE = 1024, READONLY = "no" ]
```

Virtual Machines

Overview

- A **Virtual Machine** in OpenNebula
  - A **capacity** in terms 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.
  - A **state file** (optional) or recovery file, with the memory image of a running VM plus some hypervisor specific information.
- Virutal Machines are defined in a **VM template**
- Each VM has an unique ID in OpenNebula the VMID
- All the files (logs, images, state files...) are stored in **$ONE_LOCATION/var/<VMID>**

Virtual Machines

Virtual Machine Template

```
# Name of the VM
NAME = "vm-example" # Optional, Default: one-$VMID

# Capacity
CPU    = "amount_of_requested_CPU"
MEMORY = "amount_of_requested_MEM"
VCPU   = "number of virtual cpus"

# OS and boot options
OS = [
  kernel     = "path_to_os_kernel",     # para-virtualization
  initrd     = "path_to_initrd_image",  # para-virtualization
  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
FEATURES = [
  pae  = "yes|no",    # Optional, KVM
  acpi = "yes|no" ]   # Optional, KVM
```

Virtual Machines

Virtual Machine Template

```
# VM Disks
# Using the Image Repository
DISK = [
  image      = "name of the image (deprecated)",
  image_id   = "id of the image",
  bus        = "override image attribute”,
  target     = "override default layout",
  driver     = “override image attribute" ]

# Using a source URL
DISK = [
  type       = "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" ]
```

Virtual Machines

Virtual Machine Template

```
# Network Interfaces
NIC = [
  network = "name_of_the_virtual_network",
  ip      = "ip_address",
  bridge  = "name_of_bridge_to_bind_if",
  target  = "device_name_to_map_if",
  mac     = "HW_address",
  script  = "path_to_script_to_bring_up_if",
  Model   = "NIC model"]

#  I/O Interfaces
INPUT = [
  type = "mouse|tablet",
  bus  = "usb|ps2|xen" ]
```

Virtual Machines

Virtual Machine Template

```
#  I/O Interfaces
GRAPHICS = [
  type    = "vnc|sdl",
  listen  = "IP-to-listen-on",
  port    = "port_for_VNC_server",
  passwd  = "password_for_VNC_server" ]

# Raw Hypervisor attributes
RAW = [
  type = "xen|kvm",
  data = "raw_domain_configutarion"]
```

Not all the parameters are supported for each hypervisor. Complete reference and examples for all sections in

http://opennebula.org/documentation:template

Virtual Machines

Example, define a simple VM

create a simple VM

- Use the ttylinux image
- Use the Red network
- Enable VNC access to monitor the boot process

```
NAME   = ttylinux
CPU    = 0.1
MEMORY = 64

DISK   = [ IMAGE_ID   = 0 ]
NIC    = [ NETWORK_ID = 0 ]

FEATURES = [ acpi="no" ]

GRAPHICS = [ type="vnc", listen="0.0.0.0", keymap="es" ]
```

Virtual Machines

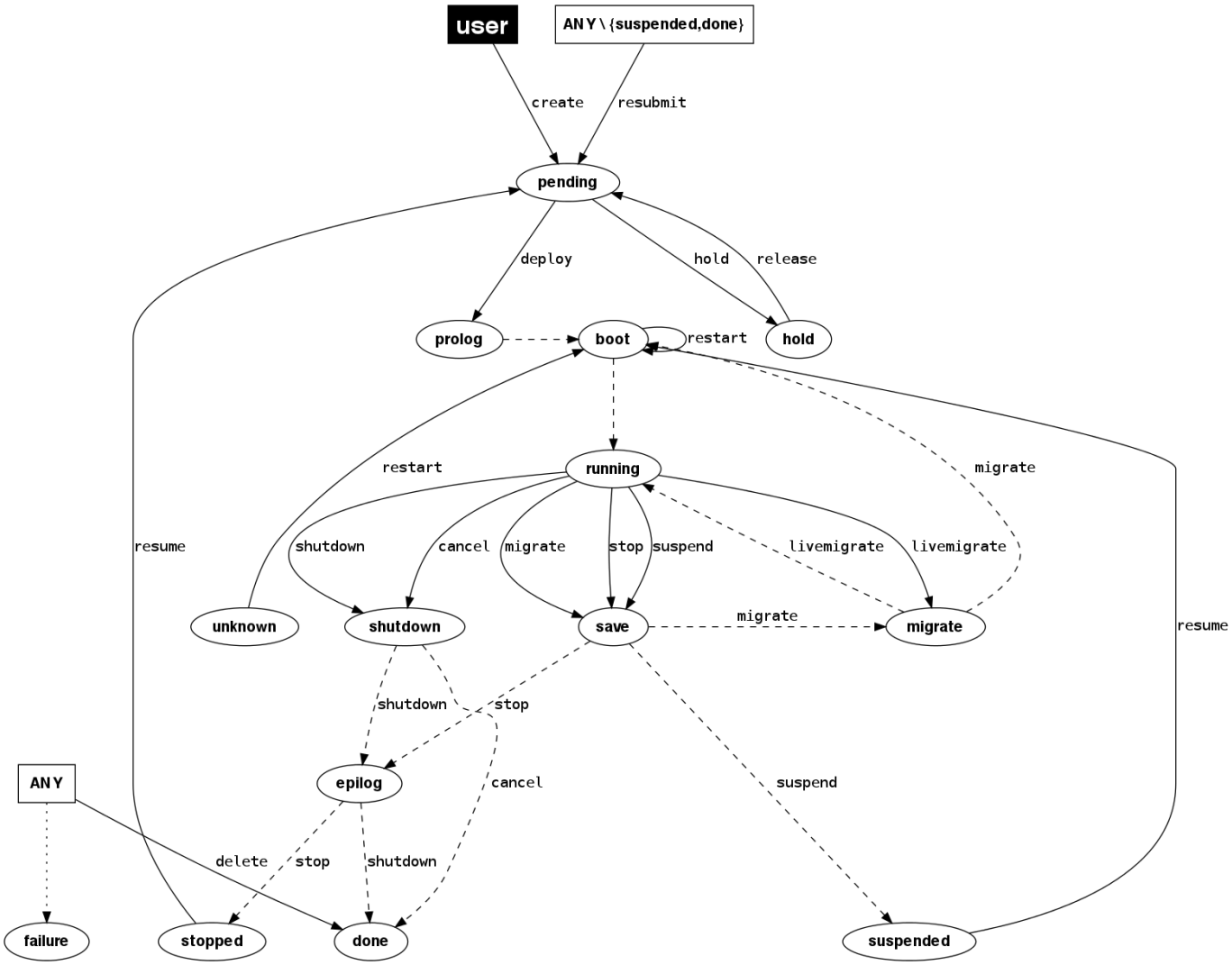
Example, define a simple VM

- Check the progress of the VM with onevm top
- Check the log with $ONE_LOCATION/var/0/vm.log

- **onevm command** options:
  - **Operations**: create, deploy shutdown, livemigrate, stop, cancel, resume, suspend, delete, restart
  - **Information**: list, show, top, history

Virtual Machines

Life-cycle of a VM (simplified)



Virtual Machines

Example, manage a simple VM

- Check status of the vnets and images in use by the VM
- Stop/Resume the Virtual Machine, check VM directory
- Migrate the Virtual Machine (cold migration)
- Live Migrate the VM
  - Update the QEMU protocol to “qemu+ssh” in

    `$ONE_LOCATION/var/remotes/kvm/kvmrc`
  - onehost sync (wait to monitor) – check /var/tmp/one
- Create another VM and check connectivity
- Add another disk with the datablock

Virtual Machines

Guidelines to Prepare a Virtual Machine

- 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**: Use the Image Repository and default layout
- **Hint V:** Do not put private information (e.g. ssh keys) in the master images, use the CONTEXT
- **Hint VI:** Pass arbitrary data to a master image using CONTEXT

Virtual Machine Templates

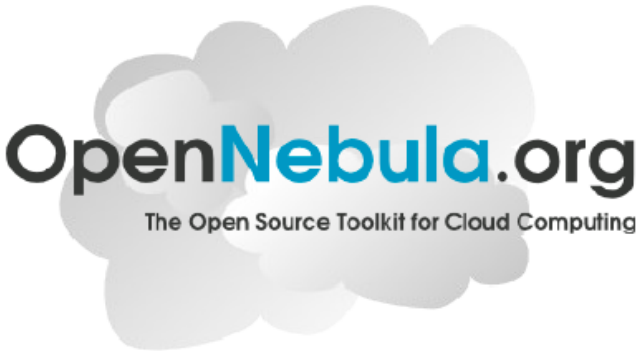
Overview

- A **Virtual Machine Template** in OpenNebula
  - VM definition
  - Can be instantiated multiple times
  - VM template repository (subject to permissions)

```
$ onetemplate list a
  ID USER     GROUP    NAME                          REGTIME
   0 oneadmin oneadmin template-0             09/27 09:37:00
   1 oneuser  users    template-1             09/27 09:37:19
   2 oneadmin oneadmin Ubuntu_server          09/27 09:37:42
```

Building Clouds with OpenNebula 3.4
Basic Usage of the Private Cloud

Constantino Vázquez Blanco
dsa-research.org | OpenNebula.org
Distributed Systems Architecture Research Group
Universidad Complutense de Madrid



- Virtual Networks
- Images
- Virtual Machines
- Templates

Virtual Machines

Example, manage a Simple VM

- Enable network access by adding a NIC to Red and Blue networks (no needed with VNC…)

```
Add a tap interface to the physical host and put it on "Red LAN"
# apt-get install openvpn

# openvpn --mktun --dev tap0

# ifconfig tap0 192.168.XX.50/24 up
# brctl addif br0 tap0

# route del -net 192.168.XX.0/24 tap0
# route add -net 192.168.XX.0/24 br0
```

- Test ssh, ping and VM connectivity