

OpenNebula/Reservoir Training, January 27-28

Brussels, Belgium

Session 3 Administration and Basic Usage – Part II

Daniel Molina & Javier Fontán
dmolina/jfontan@opennebula.org

OpenNebula.org



Copyright 2002-2011 © OpenNebula Project Leads (OpenNebula.org). All Rights Reserved.
Creative Commons Attribution Share Alike (CC-BY-SA)

Using the Private Cloud

- Preparing a VM to be used with OpenNebula
 - 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`

Using the Private Cloud: ttylinux machine

- Hands on
 - Copy the ttylinux example from `/automount/share/reservoir/opennebula/images/ttylinux-xen.tar`

```
fe$ tar vzf ttylinux-xen.tar
```

Using the Private Cloud: Images

- An Image in OpenNebula's repository
 - Resource containing an operative system or data, to be used as a virtual machine disk.
 - This data can be saved overwriting the original image, or as a new OpenNebula image.
- Three different types of images
 - **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 are defined in an Image template
- Each Image has a unique name and ID in OpenNebula
- Once registered, Image files are stored in `$ONE_LOCATION/var/images`

Using the Private Cloud: Images

- Hands on... register a ttylinux OS image

```
fe$ cat ttylinux-img.one
NAME           = "ttylinux"
TYPE           = OS
PATH           = /srv/cloud/one/ttylinux-xen/ttylinux.img
PUBLIC         = NO
PERSISTENT    = NO
DESCRIPTION    = "ttylinux OS"

fe$ oneimage register ttylinux-img.one
fe$ oneimage list
  ID      USER      NAME  TYPE          REGTIME  PUB  PER  STAT  #VMS
  0  oneadmin  ttylinux  OS      Dec 10, 2010 14:57  No  No  rdy    0

fe$ oneimage show 0
[ ... ]

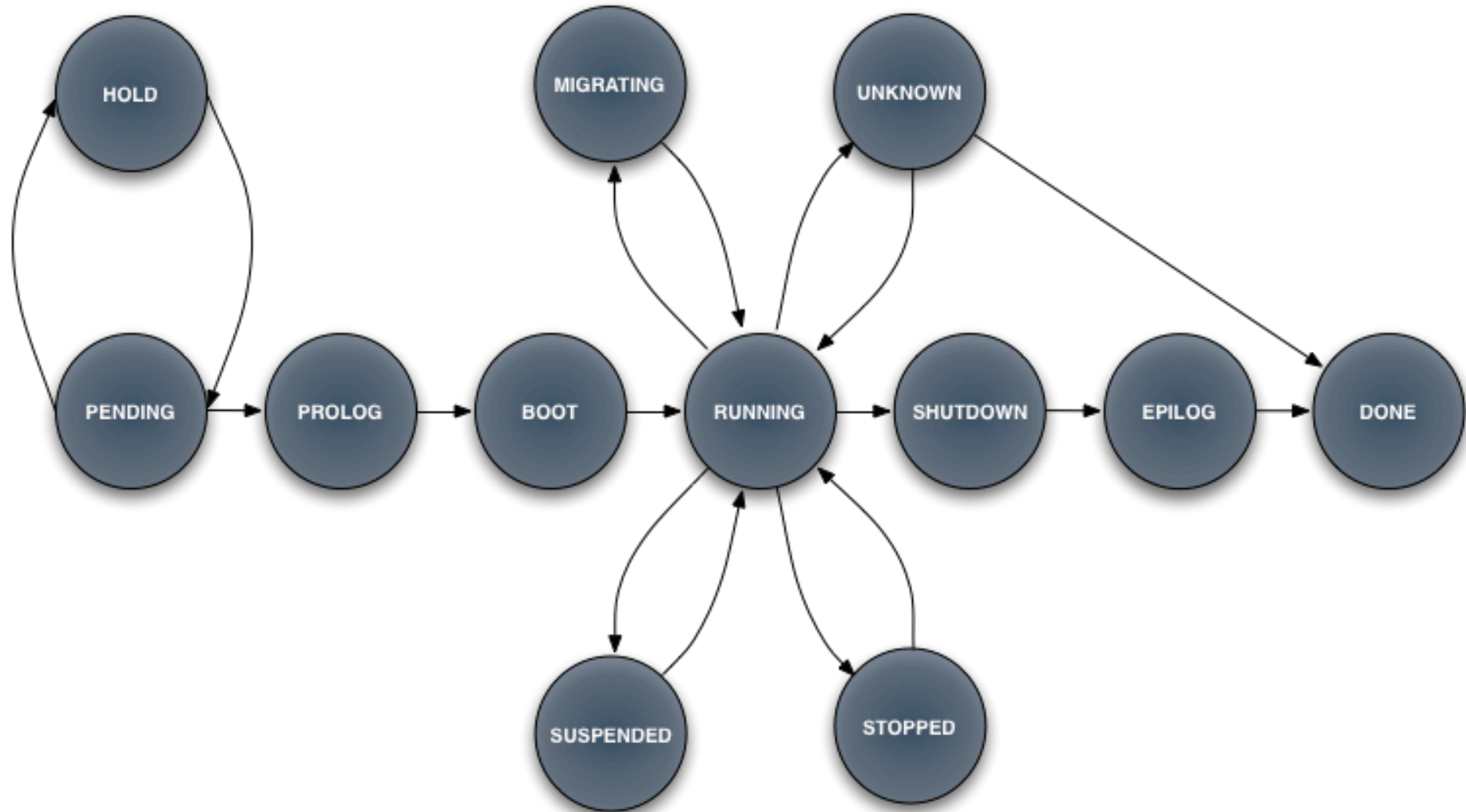
fe$ tree /srv/cloud/one/var/images
/srv/cloud/one/var/images
`-- 8625d68b699fd30e64360471eb2c38fed47
```

Using the Private Cloud: Virtual Machines

- 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.
- Virtual Machines are defined in a VM template
- Each VM has a unique ID in OpenNebula → the VM_ID
- All the files (logs, images, state files...) are stored in `$ONE_LOCATION/var/<VM_ID>`

Using the Private Cloud: Virtual Machines

- Virtual Machine Life-cycle



Using the Private Cloud: Virtual Machines

- Virtual Machine Definition File (VM *templates*)

```
#-----  
# Name of the VM  
#-----  
NAME = "vm-example" # Optional, Default: one-$VMID  
  
#-----  
# Capacity  
#-----  
CPU      = "amount_of_requested_CPU"  
MEMORY   = "amount_of_requested_MEM"  
VCPUs    = "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" ]
```


Using the Private Cloud: Virtual Machines

- Virtual Machine Definition File (VM *templates*)

```
#-----  
#           Features of the hypervisor  
#-----  
  
FEATURES = [  
    pae = "yes|no",      # Optional, KVM  
    acpi = "yes|no" ]  # Optional, KVM  
  
#-----  
#           VM Disks, using Images  
#-----  
  
DISK = [  
    IMAGE      = "Name of the Image to use",  
    IMAGE_ID   = ID,                # Optional, can be used instead of IMAGE  
    BUS        = "ide, scsi, etc.",  # Optional  
    TARGET     = "device_to_map_disk", # Optional  
    DRIVER     = "raw|qcow2|tap .. etc." ] # Optional
```

Using the Private Cloud: Virtual Machines

- Virtual Machine Definition File (VM *templates*)

```
#-----  
#           VM Disks, advanced usage  
#-----  
  
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" ]
```

Using the Private Cloud: Virtual Machines

- Virtual Machine Definition File (VM *templates*)

```
#-----  
#           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" ]
```

Using the Private Cloud: Virtual Machines

- Virtual Machine Definition File (VM *templates*)

```
#-----  
# 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:rel2.0:template>

Using the Private Cloud: Virtual Machines

- Hands on... define a new Virtual Machine:
 - Using the ttylinux Image
 - Connected to the Public and One-TD VirtualNetworks

```
fe$ cat ttylinux.one
NAME      = ttylinux-public
CPU       = 0.1
MEMORY   = 64

DISK=[
  IMAGE=ttylinux,
  READONLY=no,
  TARGET=hda ]

NIC       = [ NETWORK=Public ]
NIC       = [ NETWORK=One-TD ]

FEATURES = [ ACPI=no ]

OS=[
  INITRD=/srv/cloud/one/ttylinux-xen/initrd.gz,
  KERNEL=/srv/cloud/one/ttylinux-xen/vmlinuz-xen,
  ROOT=hda1 ]
```

Using the Private Cloud: Virtual Machines

- Virtual Machines are managed with the onevm utility
 - Operations: create, deploy shutdown, livemigrate, stop, cancel, resume, suspend, delete, restart
 - Information: list, show, top, history

```
fe$ onevm create ttylinux.one
```

```
fe$ onevm list
```

ID	USER	NAME	STAT	CPU	MEM	HOSTNAME	TIME
0	oneadmin	ttylinux	pend	0	0K		00 00:00:09

```
fe$ onevm show 0
```

```
[ ... ]
```

```
fe$ onevnet list
```

ID	USER	NAME	TYPE	BRIDGE	P	#LEASES
0	oneadmin	One-TD	Ranged	xenbr0	N	1
1	oneadmin	One-TD-Invisibl	Fixed	xenbr0	N	0

```
fe$ oneimage list
```

ID	USER	NAME	TYPE	REGTIME	PUB	PER	STAT	#VMS
0	oneadmin	ttylinux	OS	Dec 10, 2010 14:57	No	No	used	1

```
fe$ onevm top
```

Using the Private Cloud: Virtual Machines

- Hands on...
 - Create a basic VM
 - Create a couple of network enabled VMs
 - Check virtual network usage (onevnet)
 - Try control operations with the VMs
 - stop, shutdown, resume...
 - migrate – check xm list
 - Register a new persistent Datablock Image

```
NAME           = "storage"  
TYPE           = DATABLOCK  
PERSISTENT     = YES  
SIZE           = 10  
FSTYPE         = ext3
```

- Modify the template
 - Add one more NIC for the One-Td-Invisible network
 - Add another `DISK` for the persistent datablock image