Deployment Guide

Deployment Guide

v0.1

Publication date 2013-04-04

Abstract

This document is intended for individuals who wish to deploy and use our product or intend to contribute.

Table of Contents

List of Tables

Chapter 1. General Deployment Steps

Configure your environment

Lab Requirements

Table 1.1. Hardware requirements

Criteria	Minimal	Recommended
CPU	4 core @ 2.4 GHz	24 core @ 2.67 GHz
RAM	8 GB	24 GB or more
HDD	2 x 500 GB (7200 rpm)	4 x 500 GB (7200 rpm)
RAID	Software RAID-1 (use mdadm as it will improve read performance almost two times)	Hardware RAID-10

Table 1.2. Software Requirements

List
Ubuntu Server 12.04 LTS

How to test your host performance

We have measured time required to boot 1 to 5 instances of Windows system simultaneously. You can use this data as the baseline to check if your system is fast enough.

You should use sysprepped images for this test, to simulate VM first boot.

Steps to reproduce test:

1. Prepare Windows 2012 Standard (with GUI) image in QCOW2 format. Let's assume that its name is ws-2012-std.qcow2
2. Ensure that there is NO KVM PROCESSES on the host. To do this, run command:

```
ps aux | grep kvm
```

3. Make 5 copies of Windows image file:

```
for i in $(seq 5); do cp ws-2012-std.qcow2 ws-2012-std-$i.qcow2; done
```

4. Create script start-vm.sh in the folder with .qcow2 files:

```
#!/bin/bash
[ -z $1 ] || echo "VM count not provided!"; exit 1
for i in $(seq $1); do
echo "Starting VM $i ..."
```

```
kvm \
-m 1024 \
-drive file=ws-2012-std-$i.qcow2,if=virtio \
-net user -net nic,model=virtio \
-nographic \
-usbdevice tablet \
-vnc :$i &
done
```

5. Start ONE instance with command below (as root) and measure time between VM's launch and the moment when Server Manager window appears. To view VM's desktop, connect with VNC viewer to your host to VNC screen :1 (port 5901):

   ```
   # ./start-vm.sh 1
   ```

6. Turn VM off. You may simply kill all KVM processes by

   ```
   killall kvm
   ```

7. Start FIVE instances with command below (as root) and measure time interval between ALL VM's launch and the moment when LAST Server Manager window appears. To view VM's desktops, connect with VNC viewer to your host to VNC screens :1 thru :5 (ports 5901-5905):

   ```
   ./start-vm.sh 5
   ```

8. Turn VMs off. You may simply kill all KVM processes by

   ```
   killall kvm
   ```

Baseline data

	Boot ONE instance	Boot FIVE instances
Avg. Time	3m:40s	8m
Max. Time	5m	20m

Host optimizations

The following optimizations may improve host performance up to 30%:

- change default scheduler from **CFQ** to **Deadline**
- use **ksm**
- use **vhost-net**

Install software

Currently we use Devstack (http://devstack.org/) to build our lab environment.

Use Devstack's guide to install single VM OpenStack (http://devstack.org/guides/single-vm.html [http://devstack.org/guides/single-vm.html])

localrc example.

```
HOST_IP=
FLAT_INTERFACE=
FLOATING_RANGE=

ADMIN_PASSWORD=swordfish
MYSQL_PASSWORD=swordfish
RABBIT_PASSWORD=swordfish
SERVICE_PASSWORD=swordfish
SERVICE_TOKEN=tokentoken

ENABLED_SERVICES+=,heat,h-api,h-api-cfn,h-api-cw,h-eng

# Image's cache is in $TOP_DIR/files
IMAGE_URLS+=",http://fedorapeople.org/groups/heat/prebuilt-jeos-images/F17-x86

# /etc/nova/nova.conf
EXTRA_OPTS=(force_config_drive=true libvirt_images_type=qcow2 force_raw_images

# Logging
SCREEN_LOGDIR=/opt/stack/log/
LOGFILE=$SCREEN_LOGDIR/stack.sh.log
```

If you need to image builder only, then install only packages required to run **KVM** (see below).

Configure OpenStack installation

If you are using Devstack, then it’s have been done.

Otherwise, configure your OpenStack installation.

Install Murano components

Chapter 2. Environment

Hypervisor platform

We use KVM and Devstack.

Install Packages

Table 2.1. Software installed

Package name	Package version
ipxe-qemu	1.0.0+git-4.d6b0b76-0ubuntu2
kvm-ipxe	1.0.0+git-4.d6b0b76-0ubuntu2
qemu-kvm	1.4.0+dfsg-1expubuntu4
munin-libvirt-plugins	0.0.6-1
python-libvirt	1.0.2-0ubuntu11
libvirt-bin	1.0.2-0ubuntu11
libvirt0	1.0.2-0ubuntu11
munin-libvirt-plugins	0.0.6-1
python-libvirt	1.0.2-0ubuntu11
virt-goodies	0.4-2
virt-manager	0.9.4-2ubuntu3
virt-top	1.0.7-1
virt-what	1.12-1
virtinst	0.600.3-3ubuntu1
python	2.7.4-0ubuntu1

Additional Software

Windows ADK

http://www.microsoft.com/en-us/download/details.aspx?id=30652 [http://www.microsoft.com/en-us/download/details.aspx?id=30652] Windows Assessment and Deployment Kit (ADK) for Windows® 8

X11 server for Windows

http://sourceforge.net/projects/xming/ [http://sourceforge.net/projects/xming/]

Could be used any X11 software. During work process I used XMing free X11 server

config.xlaunch

```
<?xml version="1.0"?>
```

```
<XLaunch xmlns="http://www.straightrunning.com/XmingNotes"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.straightrunning.com/XmingNotes
ClientMode="NoClient" Display="0" Clipboard="true" NoAccessCon
```

Putty

We use Putty v.0.62.

You can download it from http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html [http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html]

Windows Server 2012 ISO image

We use the following Windows installation images:

Windows Version	Image Name
Windows Server 2008 R2	7601.17514.101119-1850_x64fre_server_eval_en-us-GRMSXEVAL_EN_DVD.iso
Windows Server 2012	9200.16384.WIN8_RTM.120725-1247_X64FRE_SERVER_EVAL_E US-HRM_SSS_X64FREE_EN-US_DV5.iso

You may download them using one of the following links:

Windows Version	Download Link
Windows Server 2008 R2	http://www.microsoft.com/en-us/download/details.aspx?id=11093 [http://www.microsoft.com/en-us/download/details.aspx?id=11093]
Windows Server 2012	http://technet.microsoft.com/en-US/evalcenter/hh670538.aspx?ocid=&wt.mc_id=TEC_108_1_33 [http://technet.microsoft.com/en-US/evalcenter/hh670538.aspx?ocid=&wt.mc_id=TEC_108_1_33]

VirtIO Red Hat drivers ISO image

Download drivers from http://alt.fedoraproject.org/pub/alt/virtio-win/stable/ [http://alt.fedoraproject.org/pub/alt/virtio-win/stable/]

Please, choose stable version instead of latest, We've got errors with unstable drivers during guest unattended install.

Create floppy drive image

Run commands as root

1. Create emtpy floppy image in your home folder

```
dd bs=512 count=2880 if=/dev/zero of=~/floppy.img
                            mkfs.msdos ~/floppy.img
```

2. Mount the image to **/media/floppy**

```
mkdir /media/floppy
                                mount -o loop ~/floppy.img /media/floppy
```

3. Download **autounattend.xml** file from https://raw.github.com/stackforge/murano-deployment/master/image-builder/share/files/ws-2012-std/autounattend.xml [https://raw.github.com/stackforge/murano-deployment/master/image-builder/share/files/ws-2012-std/autounattend.xml]

```
cd ~
                                wget
                                https://raw.github.com/stackforge/murano-deployment/
```

4. Copy our **autounattend.xml** to **/media/floppy**

```
cp ~/autounattend.xml /media/floppy
```

5. Unmount the image

```
umount /media/floppy
```

Chapter 3. Build System Preparation

Prepare Host System

Note

Please check that hardware virtualization supported and enabled in BIOS.

Install Hypervisor and tools

- Install packages

```
apt-get install ipxe-qemu kvm-ipxe qemu-kvm virt-goodies virtinst virt-manager lib
                munin-libvirt-plugins python python-libvirt python-libxml2 python-
                python-requests python-six samba samba-common openssh-server virt-
```

- Create shared resource on the system

Configure samba based share.

```
mkdir -p /opt/samba/share
                        chown -R nobody:nogroup /opt/samba/share
```

- Edit the /etc/samba/smb.conf

1. /etc/samba/smb.conf

```
...
[global]
server string = %h server (Samba, Ubuntu)
map to guest = Bad User
obey pam restrictions = Yes
pam password change = Yes
passwd program = /usr/bin/passwd %u
passwd chat = *Enter\snew\s*\spassword:* %n\n *Retype\snew\s*\spassword:* %n\n
*password\supdated\ssuccessfully* .
unix password sync = Yes
syslog = 0
log file = /var/log/samba/log.%m
max log size = 1000
dns proxy = No
usershare allow guests = Yes
panic action = /usr/share/samba/panic-action %d
idmap config * : backend = tdb
...
[share]
comment = Deployment Share
path = /opt/samba/share
read only = No
```

```
create mask = 0755
guest ok = Yes
...
```

Upload installation images to shared resource

Add DISPLAY env variable on HOST system

If you do not use/have UI on HOST, add to **/etc/profile** for whole system or to **~/.bashrc** or **~/.profile** for your own session:

```
export DISPLAY=IP_OF_YOUR_PC_WITH_X11_SERVER:0
```

What you should get after relogin.

```
root@ubuntu: env | grep DISPLAY
                        DISPLAY=IP_OF_YOUR_PC_WITH_X11_SERVER:0
```

Chapter 4. Manual Build

Warning

Please note that the preferred way to build images is to use **Automated Build** described later in this book.

Get Windows Post Install script

All post-install actions are performed by script named**wpi.ps1**. You may download it using the link https://raw.github.com/stackforge/murano-deployment/master/image-builder/share/scripts/ws-2012-std/wpi.ps1 [https://raw.github.com/stackforge/murano-deployment/master/image-builder/share/scripts/ws-2012-std/wpi.ps1]

To finish image preparation **Start-Sysprep.ps1** script is used. You may download it using the link https://raw.github.com/stackforge/murano-deployment/master/image-builder/share/scripts/ws-2012-std/Start-Sysprep.ps1 [https://raw.github.com/stackforge/murano-deployment/master/image-builder/share/scripts/ws-2012-std/Start-Sysprep.ps1]

Note

There are a few scripts named**wpi.ps1**, each supports only one version of Windows image. The script above is intended to be used to create Windows Server 2012 Standard. To build other version of Windows please use appropriate script from **scripts** folder.

Copy scripts to the shared folder

All scripts should be copied to the shared resource folder, subfolder **Scripts**.

Create guest VM

Way 1 - from console

Run all commands as root.

Preallocate disk image.

```
qemu-img create -f qcow2 -o preallocation=metadata
                           /var/lib/libvirt/images/winserv_vio.qcow2 20G
```

Start guest install.

```
virt-install --connect qemu:///system --hvm --name WinServ --ram 2048 --vcpus
--cdrom
/opt/samba/share/9200.16384.WIN8_RTM.120725-1247_X64FRE_SERVER_EVAL_EN-US-HRM_
\
--disk path=/opt/samba/share/virtio-win-0.1-52.iso,device=cdrom \
--disk path=/opt/samba/share/flop.img,device=floppy \
```

```
    --disk path=/var/lib/libvirt/images/winserv_vio.qcow2,format=qcow2,bus=virtio,
    --network network=default,model=virtio \
    --memballoon model=virtio --vnc --os-type=windows --os-variant=win2k8 --noauto
--accelerate --noapic --keymap=en-us --video=cirrus --force
```

Way 2 - from virt-manager UI

- launch virt-manager from shell as root
- set Guest VM name and Local install media
- add 1 cdrom - Windows Server ISO image
- set OS type: Windows and version: Windows Server 2008
- set CPU and RAM amount
- deselect Enable storage for this virtual machine
- select Customize configuration before install
- add 2 cdrom - virtio ISO image
- add floppy - our floppy image
- add/or create HDD image with Disk bus: VirtIO and storage format: QCOW2
- set network device model: VirtIO
- If everything OK - start installation process, guest vm screen accessible through Console button

Finish process

After install process finished for reference image compression run as root

```
qemu-img convert -O qcow2 /var/lib/libvirt/images/winserv_vio.qcow2
                /var/lib/libvirt/images/winserv_vio_ref.qcow2
```

Chapter 5. Automated Build

1. Clone **murano-deployment** repository

   ```
   git clone git://github.com/stackforge/murano-deployment.git
   ```

2. Change directory to **murano-deployment/image-builder** folder.

3. Create folder structure for image builder

   ```
   make build-root
   ```

4. Create shared resource

   Add to /etc/samba/smb.conf.

   ```
   [image-builder-share]
   comment = Image Builder Share
   browsable = yes
   path = /opt/image-builder/share
   guest ok = yes
   guest user = nobody
   read only = no
   create mask = 0755
   ```

   Restart samba services.

   ```
   restart smbd && restart nmbd
   ```

5. Test that all required files are in place

   ```
   make test-build-files
   ```

6. Get list of available images

   ```
   make
   ```

7. Run image build process

   ```
   make ws-2012-std
   ```

8. Wait until process finishes

9. The image file **ws-2012-std.qcow2** should be stored under **/opt/image-builder/share/images** folder.

Chapter 6. Image registration

Murano is one of the Openstack services which communicates with other Openstack components. Therefore just created image should be registered in Openstack Glance - image operation service.

1. Use the glance image-create command to import your disk image to Glance:

```
$ glance image-create --name "NAME" --is-public IS_PUBLIC --disk-format DISK
--container-format CONTAINER_FORMAT --file IMAGE --property IMAGE_METADATA
```

   Replace the command line arguments to glance image-create with the appropriate values for your environment and disk image:

   - Replace NAME with the name that users will refer to the disk image by.
   - Replace IS_PUBLIC with true or false. Setting this value to true means that all users will be able to view and use the image.
   - Replace DISK_FORMAT with the format of the virtual machine disk image. Valid values include raw, vhd, vmdk, vdi, iso, qcow2, aki, and ami.
   - Replace CONTAINER_FORMAT with the container format of the image. The container format is bare unless the image is packaged in a file format such as ovf or ami that includes additional metadata related to the image.
   - Replace IMAGE with the local path to the image file to upload.
   - Replace IMAGE_METADATA with the following property: *murano_image_info="{'title': 'Windows 2012 Core edition', 'id': NAME }"*, where *title* - full image description for user and *id* is an image name.

     ### Warning

     Setting *murano_image_info* property is required to pick up image from Murano Dashboard.

2. To update image metadata perform this command:

```
$ glance image-update IMAGE-ID --property IMAGE_MATADATA
```

   - Replace IMAGE-ID with image id from the previous command output.
   - Replace IMAGE_METADATA with murano_image_info property, e.g. *murano_image_info="{'title': 'Windows 2012 Core edition','id': 'win2k12core'}"*,

After these steps desired image can be chosen in Murano dashboard and used for services platform.