libvirt

November 11, 2013

About Hypervisors

A Hypervisor

Its job is to execute a virtual environment on the HOST (the physical machine), and provide access to virtualized resources

- disk, cd, usb
- network
- memory
- video output

. . . to one or more virtual machines.

About Hypervisors (cont'd)

How are emulated resources in a VM seen, from the point of view of the host the VMs are running on ?

- virtual hard drive <-> disk file on host (for example)
- virtual CD-ROM <-> ISO file on host
- virtual NIC <-> real NIC or tap interface onto bridge on host
- virtual console <-> CLI connection or VNC "graphics card" on host

KVM

KVM stands for Kernel Virtual Machine. It's a popular (the most popular) hypervisor for running on Linux hosts.

http://www.linux-kvm.org

This doesn't mean that it can only run Linux virtual machines! It just means that almost any Linux machine can be used as a HOST that will run virtual machines. The virtual machines (VMs) can run any operating system.

KVM will manage the emulated resources and show them to the virtual machines as devices (disks, memory, network cards, graphics, . . .)

Running KVM

In its simplest form:

```
kvm disk_image_file
```

In reality

```
kvm -lots -of -parameters disk_image_file
```

So we'd have:

```
kvm -m 1024 -drive file=myPC.img -netdev tap -vga cirrus
  -name myPC ...
```

KVM is quite unwieldy and it's hard to remember all the different parameters!

About libvirt

http://libvirt.org

Libvirt is a “wrapper” which controls multiple hypervisors, including KVM

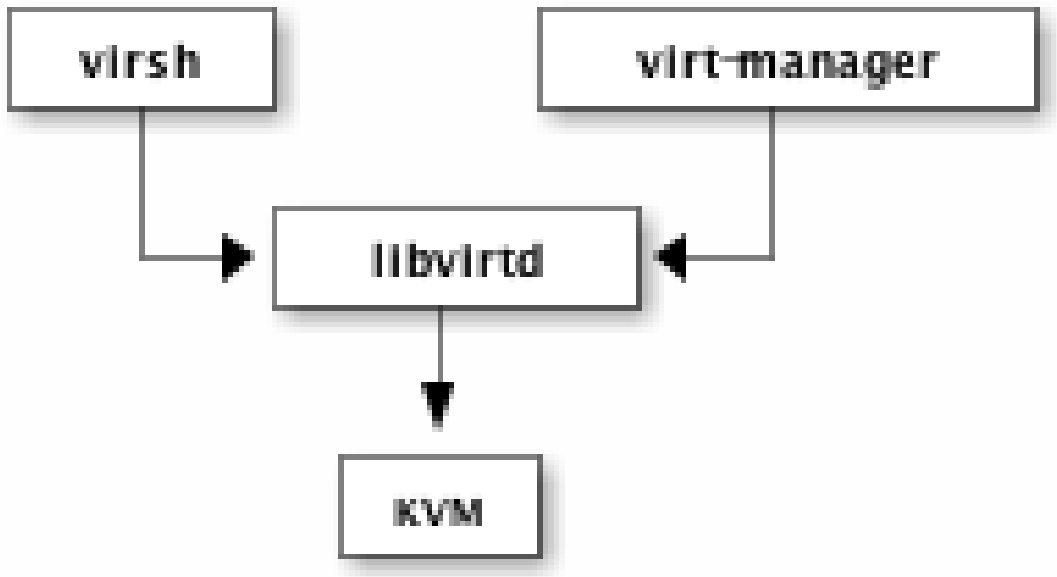


Figure : libvirt

About libvirt (cont'd)

Some legacy terminology:

"guest domain" == "virtual machine" (VM)

Libvirt takes care of all the ugly details.
Remember all the "plumbing" ? (disk handling, network cards, etc. . .)

- The earlier example was quite simple. . . There can be MANY options to run a VM
- If you were running kvm manually, you'd have to remember all those options or write them down somewhere.
- libvirt manages this plumbing. It stores the configuration in an XML file for each VM, so you don't have to remember them each time.
- libvirt also provides an "abstraction later", so you can actually use libvirt to manage other hypervisors (Note: we'll only use KVM with libvirt in our workshop)

About libvirt (cont'd)

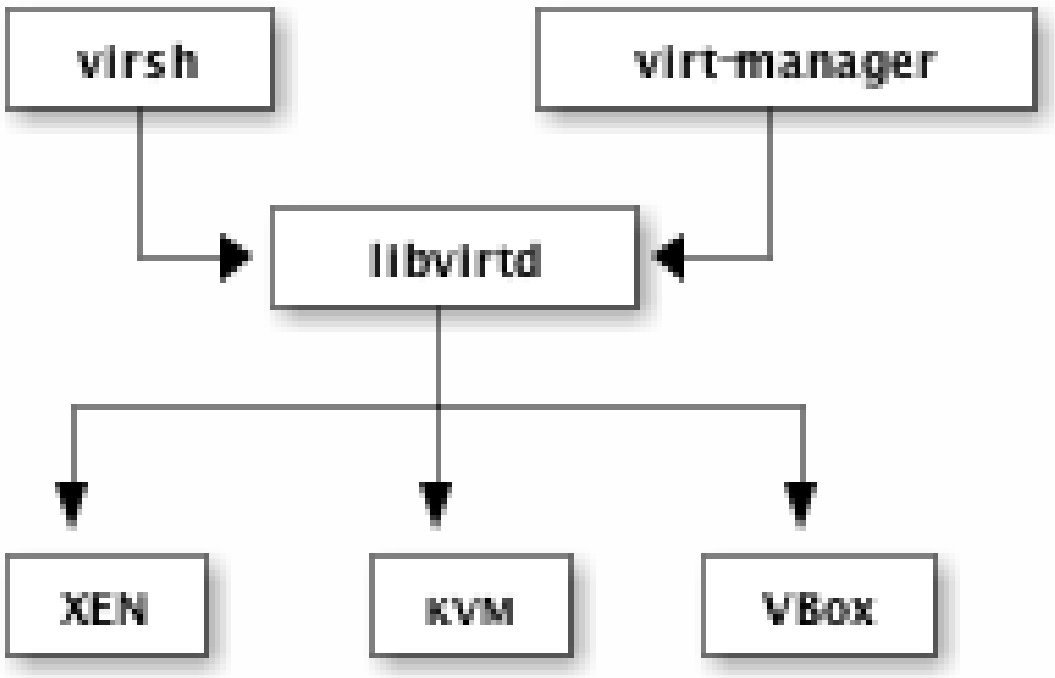


Figure : libvirt

virsh and virt-manager

Libvirt provides two useful interfaces to manage virtual machines.

virsh and **virt-manager**

- `virsh` is command line (CLI) driven
- `virt-manager` is a graphical (GUI) application

We'll start by looking at `virsh`, as it's the command we will be using the most, both interatively and from within automation tools (ansible, etc.)

Using `virsh`

Here are some base commands for `virsh`

Command	Description
`virsh list [--all]`	list running (or all) VMs
`virsh start` *VM*	start the VM named *VM*
`virsh shutdown` *VM*	shutdown VM (properly)
`virsh destroy` *VM*	kill a VM (power off)
`virsh console` *VM*	connect to the console of a VM
`virsh define` *XMLFILE*	create VM definition from this file
`virsh undefine` *VM*	erase the machine definition (danger!)

We'll review some of these on the screen together.

Using virsh

Another useful command is `virsh dumpxml` *VM*

This will print out the configuration for a virtual machine. For instance:

```
$ virsh dumpxml noc.ws.nsrc.org

<domain type='kvm'>
  <name>noc.ws.nsrc.org</name>
  <uuid>4641a945-abab-1c0b-0fb0-2db681c28130</uuid>
  <memory>1048576</memory>
  <currentMemory>1048576</currentMemory>
  <vcpu>1</vcpu>
  <os>
    <type arch='x86_64' machine='pc-1.0'>hvm</type>
    <boot dev='hd'/>
  </os>
  ...
```

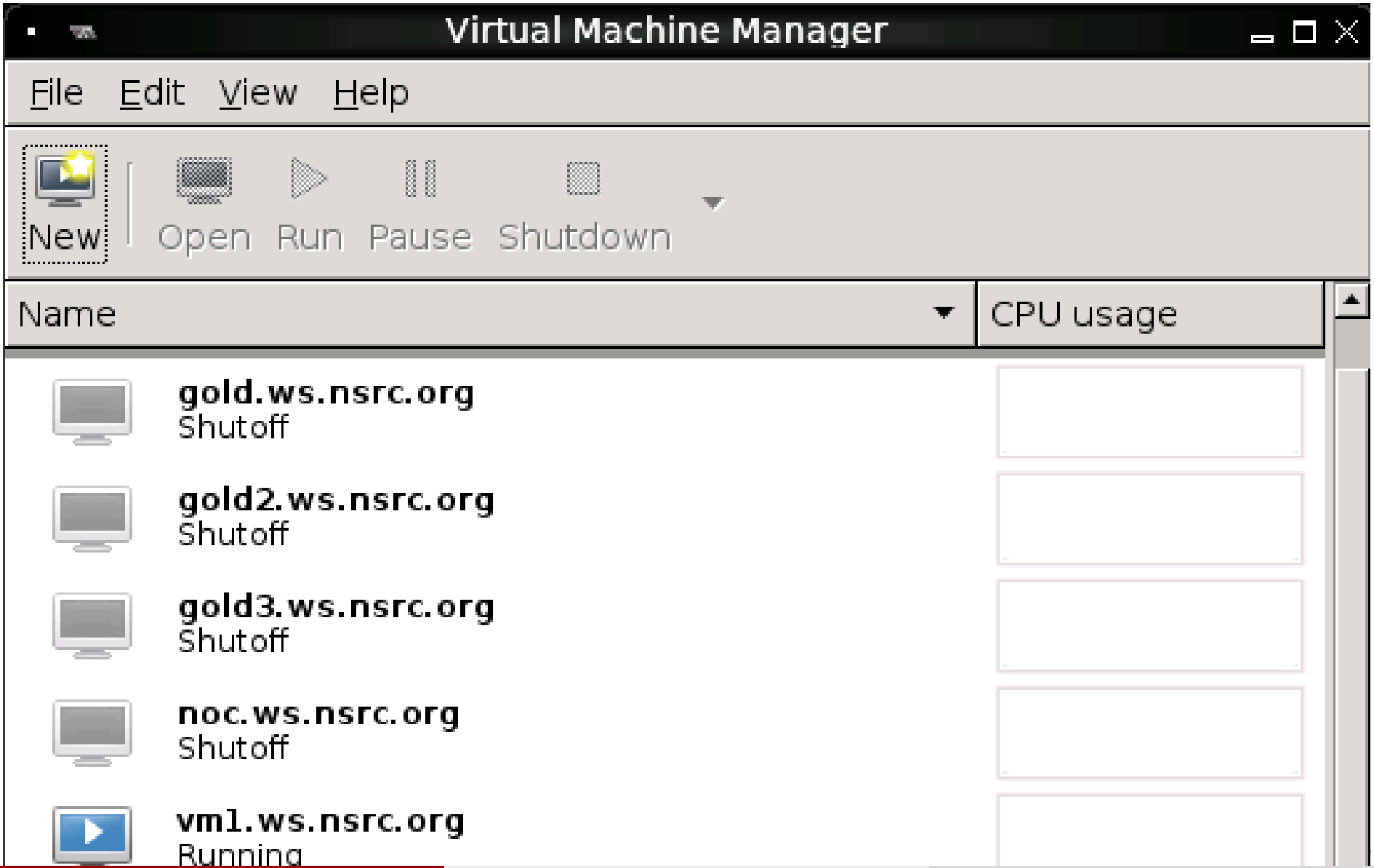
Using `virt-manager`

GUI: virt-manager. A live demo!

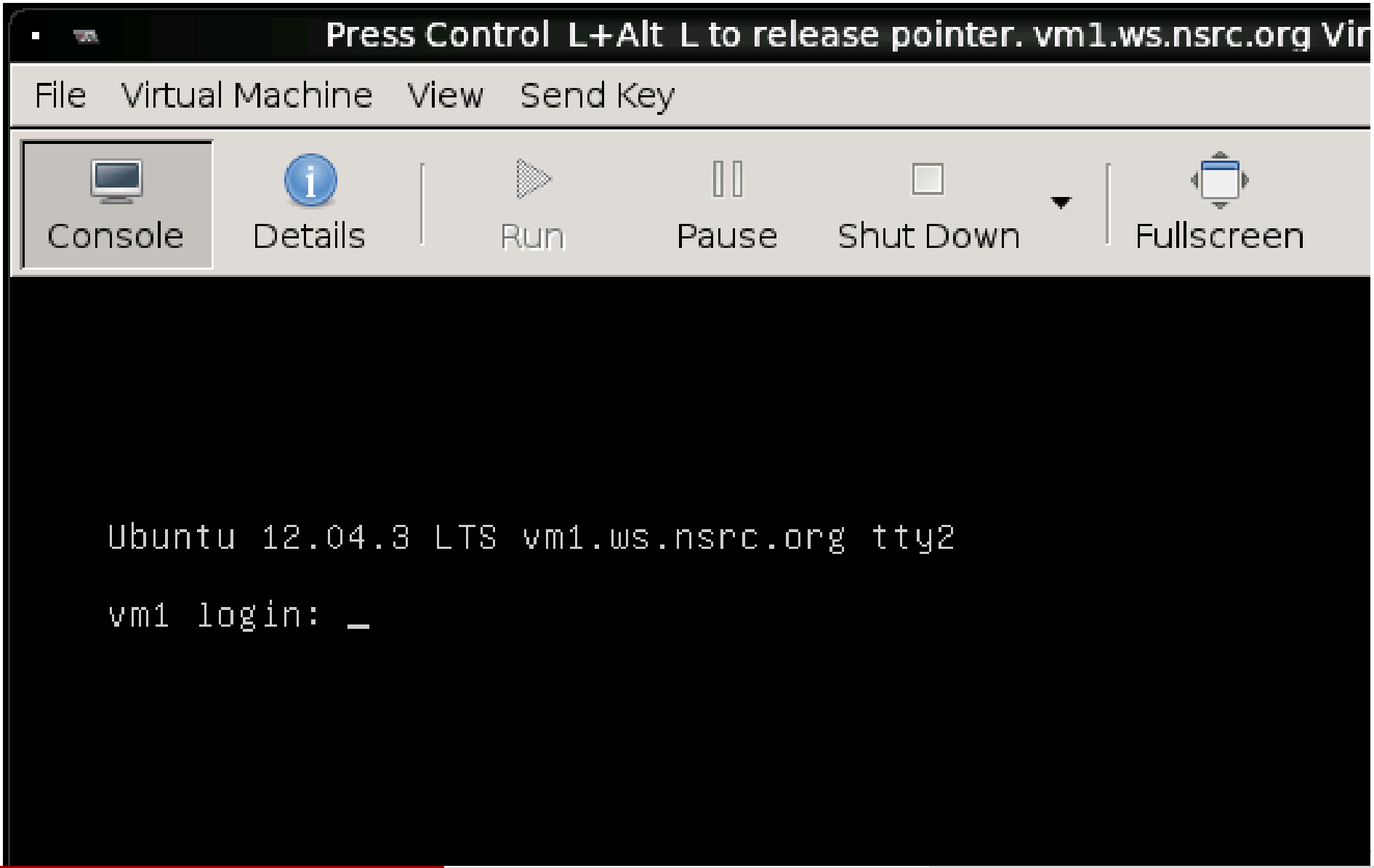
What we'll do:

- use VNC to connect to a desktop session
- start virt-manager
- show existing VMs
- create a new VM in the GUI
- attach an ISO image to the VM
- boot the VM and start installing the OS interactively
- show the disk backing file using dumpxml

Using `virt-manager` - main view



Using virt-manager - console view



Using `virt-manager` - VM details/settings