Xen Virtualization and the Art of Virtual Clusters
What We’ll Cover

- Overview of Xen
- Installing a virtual cluster in Rocks
- Extra Xen roll commands
- “Lights Out” VM frontend install
What is Xen

- Xen is “virtual machine monitor” (VMM) used to control VMs
- Xen VMM is called the “hypervisor”
- Xen VMs are called “guests”
What is Xen

- Guests’ traps and exceptions are passed to and handled by hypervisor
Xen in Rocks 5.2
Step 0

- You must install a Rocks 5.2 frontend with the Xen Roll
Step 0.5

- Optionally install at least one cluster node as a "VM Container"

![Image of a graphic showing options for appliance types with "VM Container" selected.]

© 2009 UC Regents
Supported Configurations

- Physical frontend with virtual compute nodes

- Virtual frontend with virtual compute nodes

☞ Note: A virtual frontend with physical compute nodes is doable, but it requires an understanding of VLANs
Physical Frontend and Virtual Computes

- All node names with a white background are physical machines
- All node names with purple backgrounds are virtual
- This was the only configuration that Rocks v5.0 supported
Virtual Frontend and Virtual Computes
Virtual Frontend and Virtual Computes

- Network traffic for VM frontend and VM computes are isolated with a VLAN
- Processes running on the physical machines don’t see the traffic from the virtual cluster
Key VM Functions

- “add cluster”
  - Add a new VM cluster

- “start host vm”
  - Boot a VM

- “set host boot”
  - Set a VM to install or boot its OS
Adding a Cluster

◆ “rocks add cluster” command

```
# rocks add cluster {FQDN of frontend} \
{IP of frontend} {number of computes}
```

◆ What this does:
  ➤ Creates a frontend VM on the physical frontend (frontend-0-0-0)
  ➤ Creates virtual compute nodes on VM containers (in round robin order)
  ➤ Creates a unique VLAN for this cluster and plumbs the VMs to it
    • Adds a new unique VLAN to the physical machines on-the-fly
More on What the Command Does

- Adds an entry to the `vm_nodes` table
  - Keep track of which physical host houses the VM

- Adds an entry to the `vm_disks` tables
  - Allocates disk space for the VM
    - Uses the Xen “file” virtual block device
    - Puts the file on the largest partition of the physical host

- Allocates unique MAC addresses for each VM
  - MAC prefix is based on the frontend’s public IP
MAC Address Prefix

- MAC prefix is based on the frontend’s public IP
  - Take the public IP, toss the first octet, then reverse it
    - Most unique part of IP address is the MAC’s first octet

- Also set the “locally administered” bit and clear the “multicast” bit
Adding a Cluster

◆ Example

```bash
# rocks add cluster vi-1.rocksclusters.org \ 137.110.119.118 2
```

◆ Output:

```
created frontend VM named: frontend-0-0-0
created compute VM named: hosted-vm-0-0-0
created compute VM named: hosted-vm-0-1-0
```
Adding a Cluster

# rocks list cluster
FRONTEND               CLIENT NODES               TYPE
bayou.rocksclusters.org: ---------------- physical
:                       vm-container-0-0 physical
:                       vm-container-0-1 physical
vi-1.rocksclusters.org: ---------------- VM
:                       hosted-vm-0-0-0 VM
:                       hosted-vm-0-1-0 VM
‘rocks add cluster’ Extra Flags

- `[container-hosts=string]`  
  A list of VM container hosts that will be used to hold the VM compute nodes.

- `[cpus-per-compute=string]`  
  The number of CPUs to allocate to each VM compute node. The default is 1.

- `[disk-per-compute=string]`  
  The size of the disk (in gigabytes) to allocate to each VM compute node. The default is 36.

- `[disk-per-frontend=string]`  
  The size of the disk (in gigabytes) to allocate to the VM frontend node. The default is 36.

- `[mem-per-compute=string]`  
  The amount of memory (in megabytes) to allocate to each VM compute node. The default is 1024.

- `[vlan=string]`  
  The VLAN ID to assign to this cluster. All network communication between the nodes of the virtual cluster will be encapsulated within this VLAN. The default is the next free VLAN ID.
Install the Frontend VM

- “rocks start host vm” command

  # rocks start host vm frontend-0-0-0

- This starts a standard Rocks installation on the VM
Install the Frontend VM

- To input data at user screens, need to bring up the console for the VM frontend

  # virt-manager
Virt-manager

- Double click on ‘localhost’
Double click on ‘frontend-0-0-0’ to bring up the console
Virt-manager

Welcome to Rocks

Selected Rolls

No rolls have been selected.

If you have CD/DVD-based rolls (that is, ISO images that have been burned onto CDs or a DVD), then click the CD/DVD-based Roll button. The media tray will eject. Then, place your first roll disk in the tray and click Continue. Repeat this process for each roll disk.

If you are performing a network-based installation (also known as a central installation), then input the name of your roll server into the Hostname of Roll Server field and then click the Download button. This will query the roll server and all the rolls that the roll server has available will be displayed. Click the selected checkbox for each roll you will to install from the roll server.

When you have completed your roll selections, click the Next button to proceed to cluster input screens (e.g., IP address selection, root password setup, etc.).
Install the Frontend VM

◆ Input the data
  ➲ Hint: use the FQDN of the physical frontend for the “Hostname of the Roll Server”

◆ The frontend will install, then reboot
◆ X will start and you’ll see the graphical login screen
  ➲ Just like a physical frontend!
Install VM Compute Nodes

- Login to the VM frontend
  - Run ‘insert-ethers’
- On the physical frontend, execute:
  
  ```bash
  # rocks start host vm hosted-vm-0-0-0
  ```

- The software configuration comes from the VM frontend
- The “power control” comes from the physical frontend
Virtual Frontend and Virtual Computes
Get Status

- Use ‘rocks list cluster status=y’
- On the physical frontend:

```
# rocks list cluster status=y
FRONTEND            CLIENT NODES           TYPE          STATUS
bayou.rocksclusters.org: -------------- physical ------
:                        vm-container-0-0  physical ------
:                        vm-container-0-1  physical ------
vi-1.rocksclusters.org: -------------- VM active
:                        hosted-vm-0-0-0  VM active
:                        hosted-vm-0-1-0  VM nostate
```
Other Rocks Xen Commands
# rocks list host vm status=y

<table>
<thead>
<tr>
<th>VM-HOST</th>
<th>SLICE</th>
<th>MEM</th>
<th>CPUS</th>
<th>MAC</th>
<th>HOST</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>frontend-0-0-0</td>
<td>0</td>
<td>1024</td>
<td>1</td>
<td>72:77:6e:80:00:00</td>
<td>bayou</td>
<td>active</td>
</tr>
<tr>
<td>frontend-0-0-0</td>
<td>-----</td>
<td>-----</td>
<td>----</td>
<td>72:77:6e:80:00:01</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>hosted-vm-0-0-0</td>
<td>0</td>
<td>1024</td>
<td>1</td>
<td>72:77:6e:80:00:02</td>
<td>vm-container-0-0</td>
<td>active</td>
</tr>
<tr>
<td>hosted-vm-0-1-0</td>
<td>0</td>
<td>1024</td>
<td>1</td>
<td>72:77:6e:80:00:03</td>
<td>vm-container-0-1</td>
<td>nostate</td>
</tr>
</tbody>
</table>
set

◆ Change VM parameters

# rocks set host vm {host} [disk=string] [disksize=string] \ [mem=string] [physnode=string] [slice=string] \ [virt-type=string]

◆ Example, allocate 4 GB of memory to a VM:

    # rocks set host vm hosted-vm-0-0-0 mem=4096
pause/resume

- Execute the “pause” and “resume” Xen commands on a VM
  
  ```bash
  # rocks pause host vm hosted-vm-0-0-0
  # rocks resume host vm hosted-vm-0-0-0
  ```

- “pause” holds the VM in memory, but the hypervisor doesn’t schedule the VM
  - Like hitting a breakpoint
save/restore

- Execute the “save” and “restore” Xen commands on a VM
  ```
  # rocks save host vm hosted-vm-0-0-0
  # rocks restore host vm hosted-vm-0-0-0
  ```

- What’s the difference between “pause” and “save”?
  - “pause” keeps the VM in memory
  - “save” writes VM state to a file and releases memory and CPU
stop

◆ Destroy a VM

    # rocks stop host vm hosted-vm-0-0-0

◆ This is equivalent to pulling the power cord on a physical machine
move

- Move a VM from one physical node to another

  # rocks move host vm hosted-vm-0-0-0 vm-container-1-0

- This operation will take some time
  - It “saves” the current VM
  - Copies the VMs disk file to the new VM container
    - If your diskfile is 36 GB, it will move 36 GB across the network
  - Then “restores” the VM
Other “Internal” Commands

- “dump”
  - Used on the restore roll to capture VM configuration

- “report”
  - Called by “rocks start host vm” to create Xen VM configuration files

- “remove”
  - Called by “rocks remove host” to remove the VM specific info for a host
Lights Out Virtual Frontend Install
What is “Lights Out”? 

- Install a frontend without filling out any user screens
- Accomplish this with “attributes” and by assigning rolls to a VM frontend
Attributes

- Assign values to variables in the graph
- An evolution of the `<var>` tags and the `app Globals` table
- Previous syntax:
  ```
  ServerName <var name="Kickstart_PublicHostname"/>
  ```
- New syntax:
  ```
  ServerName &Kickstart_PublicHostname;
  ```
Attributes

- Attributes can be set at 4 levels:
  - Globally
    - ‘rocks set attr’
  - By appliance type
    - ‘rocks set appliance attr’
  - By OS (linux or sunos)
    - ‘rocks set os attr’
  - By host
    - ‘rocks set host attr’
### Attributes

```
# rocks list host attr tile-0-0

<table>
<thead>
<tr>
<th>HOST</th>
<th>ATTR</th>
<th>VALUE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>tile-0-0</td>
<td>Info_CertificateCountry</td>
<td>US</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Info_CertificateLocality</td>
<td>San Diego</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Info_CertificateOrganization</td>
<td>CalIT2</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_DistroDir</td>
<td>/export/rocks</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PrivateAddress</td>
<td>10.1.1.1</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PrivateBroadcast</td>
<td>10.1.255.255</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PrivateDNSServers</td>
<td>10.1.1.1</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PrivateGateway</td>
<td>10.1.1.1</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicDNSServers</td>
<td>132.239.0.252</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicGateway</td>
<td>137.110.119.1</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicHostname</td>
<td>vizagra.rocksclusters.org</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicKickstartHost</td>
<td>central.rocksclusters.org</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicNTPHost</td>
<td>pool.ntp.org</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicNetmask</td>
<td>255.255.255.0</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicNetmaskCIDR</td>
<td>24</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_PublicNetwork</td>
<td>137.110.119.0</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Kickstart_Timezone</td>
<td>America/Los_Angeles</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>Server_Partitioning</td>
<td>force-default-root-disk-only</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>arch</td>
<td>x86_64</td>
<td>H</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>hostname</td>
<td>tile-0-0</td>
<td>I</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>rack</td>
<td>0</td>
<td>I</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>rank</td>
<td>0</td>
<td>I</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>rocks_version</td>
<td>5.2</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>HideBezels</td>
<td>false</td>
<td>G</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>HttpConf</td>
<td>/etc/httpd/conf</td>
<td>O</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>HttpConfigDirExt</td>
<td>/etc/httpd/conf.d</td>
<td>O</td>
</tr>
<tr>
<td>tile-0-0</td>
<td>HttpRoot</td>
<td>/var/www/html</td>
<td>O</td>
</tr>
</tbody>
</table>
```
Setting Host Attributes

# rocks set host attr vi-1 Kickstart_PrivateHostname vi-1

# rocks set host attr vi-1 Kickstart_PublicAddress \ 137.110.119.118

# rocks set host attr vi-1 Kickstart_PublicHostname \ vi-1.rocksclusters.org

# rocks set host attr vi-1 Kickstart_PublicKickstartHost \ bayou.rocksclusters.org
Assign Rolls to the VM Frontend

```
# rocks set host roll vi-1 os 5.2 x86_64
# rocks set host roll vi-1 base 5.2 x86_64
# rocks set host roll vi-1 kernel 5.2 x86_64

# rocks list host roll vi-1
HOST            NAME   VERSION ARCH   OS
frontend-0-0-0: os     5.2     x86_64 linux
frontend-0-0-0: base   5.2     x86_64 linux
frontend-0-0-0: kernel 5.2     x86_64 linux
```
Assign Rolls to VM Frontend

```xml
# rocks report host roll vi-1
<rolls>
  <roll
      name="os"
      version="5.2"
      arch="x86_64"
      url="http://bayou.rocksclusters.org/install/rolls/"
      diskid=""
  />
  <roll
      name="base"
      version="5.2"
      arch="x86_64"
      url="http://bayou.rocksclusters.org/install/rolls/"
      diskid=""
  />
  <roll
      name="kernel"
      version="5.2"
      arch="x86_64"
      url="http://bayou.rocksclusters.org/install/rolls/"
      diskid=""
  />
</rolls>
```
Start the VM

```bash
# rocks start host vm vi-1
```

- This will automatically install the VM frontend
  - Internals:
    - This creates the files `/tmp/siteattrs` and `/tmp/rolls.xml` in the installing node
    - If these two files exist, then the user input screens will be skipped

- When the installation completes, the VM frontend will reboot
  - If you connect to the VM console with `virt-manager`, you’ll see that X is up at the login screen
Error Conditions

◆ When the VM frontend is trying to get its kickstart file: “Could not get file”
  ➤ Try opening up firewall
  ➤ By default, http/https is open on public network to hosts in the same subnet
    • E.g., if host is 137.110.119.118/24, then any host on the subnet 137.110.119.0 can access the web server over the public interface
Xen in Rocks Futures
Futures

- Support fully-virtualized VMs
  - Can run any OS in a VM container
Supporting Different Architectures for VMs
Support both i386 and x86_64 VMs

- On a x86_64 physical cluster, can support both 32-bit and 64-bit VMs

- Need to:
  - Get i386 Rocks ISOs
  - Create new distro
  - Add i386 version of ‘rocks-boot’ to the frontend
  - Add bootactions to the database
Adding rocks-boot

# cd /export/rocks/install
# rpm -i --force rocks-dist/i386/RedHat/RPMS/rocks-boot-xen-5.2-1.i386.rpm

◆ This adds the files:
  ✉ /boot/kickstart/xen/vmlinuz-5.2-i386
  ✉ /boot/kickstart/xen/initrd-xen.iso.gz-5.2-i386
Add bootactions

```bash
# rocks add bootaction action="install vm i386" \
  kernel="file:///boot/kickstart/xen/vmlinuz-5.2-i386" \
  ramdisk="file:///boot/kickstart/xen/initrd-xen.iso.gz-5.2-i386" \
  args="ks ramdisk_size=1500000 lang= devfs=nomount kssendmac \n       selinux=0 noipv6"

# rocks add bootaction action="install vm frontend i386" \
  kernel="file:///boot/kickstart/xen/vmlinuz-5.2-i386" \
  ramdisk="file:///boot/kickstart/xen/initrd-xen.iso.gz-5.2-i386" \
  args="ramdisk_size=150000 lang= devfs=nomount pxe kssendmac \n       selinux=0 noipv6 \n       ks=http://bayou.rocksclusters.org/install/sbin/kickstart.cgi \n       ksdevice=eth1 build"

◆ Change ‘bayou.rocksclusters.org’ to your physical frontend
```
Assign bootactions to i386 VMs

# rocks set host installaction frontend-0-0-0 \ action=“install vm frontend i386”

# rocks set host installaction hosted-vm-0-0-0 \ action=“install vm i386”