The Integration of Virtualization into the U.S. ATLAS Tier 1 Facility at Brookhaven

Christopher Hollowell <hollowec@bnl.gov>
RHIC/ATLAS Computing Facility (RACF)
Physics Department
Brookhaven National Laboratory

BROOKHAVEN NATIONAL LABORATORY

Computing in High Energy and Nuclear Physics
Prague | Czech Republic | 21 - 27 March 2009

Virtualizaton Overview

- What is Virtualization?
  - A software layer which abstracts a single computer/server into many, allowing for the simultaneous execution of multiple operating system instances
- Implementations for Linux
  - Full Virtualization
    - VirtualBox, VMWare
    - Hardware Assisted Full Virtualization
      - KVM, Xen, VMWare, VirtualBox
  - Paravirtualization
    - Xen
  - Operating System-Level Virtualization
    - OpenVZ

Virtualizaton Overview (Cont.)

- Terminology
  - Host OS
    - Xen Domain0 (Dom0)
    - Privileged control
  - Guest OS
    - Xen DomainU (DomU)
    - Unprivileged
  - Hypervisor
    - Virtualization software itself - virtual machine monitor
    - Type 1
      - Executes in a control layer above the host and guest operating systems
        - Xen, VMware ESX
    - Type 2
      - Runs under an operating system
        - VirtualBox, VMWare, Parallels

Virtualizaton at the RACF

- Running Xen 3.0.3, as shipped with RHEL5/SL5
- Used to split multicore hosts into individual virtual servers where OS segmentation is desirable or necessary
  - Allows for the most efficient use of increasingly prevalent multicore hardware
    - Specific operating system version requirements
    - Testbeds
    - Isolation of low and high security services
    - Reduction of resource contention (i.e. memory, disk space), and the impact of OS crashes

Virtualizaton at the RACF (Cont.)

- U.S. ATLAS Tier1 Processor Farm
  - 12 8-core physical machines paravirtualized into 40 servers: 2-3 guests + 1 Dom0 per host
  - Each physical system contains a single interactive virtual machine, and one or more batch/testbed host components
  - 32-bit SL5 Dom0 (control only), 32-bit SL4 DomUs
  - Physical CPUs pinned to guests
  - Networking via bridging, partitions for virtual disk devices
  - All interactive systems/submit hosts virtualized
    - Many interactive hosts desired for service redundancy
    - Current usage does not require more than 2 CPUs per host
    - Eliminates interactive vs. batch process contention for disk space, memory, and CPU resources

Virtualizaton at the RACF (Cont.)

- U.S. ATLAS Tier1 General Services
  - Consists of WWW servers, database servers, ssh gateways, etc.
  - ~35 systems paravirtualized
  - 64-bit RHEL5 Dom0, 32/64-bit RHEL5/4 DomUs
  - Primarily used to replace retired (out of warranty) hardware with virtual machines
    - Possible for hosts without high bandwith/low latency network and disk access requirements
  - New hosts without extensive hardware requirements virtualized as well
  - Example usage:
    - MonALISA server
    - MyProxy server
    - WWW servers primarily providing static content

Virtualizaton at the RACF (Cont.)

- Potential Future Use
  - Many of the RACF processor farm batch hosts are also dCache or xrootd/rootd storage pools. To increase reliability, it may be desirable to isolate the storage components from batch processing via virtual machines
  - Cloud computing?
  - Both would imply the deployment of Xen on the majority of our 1900-node processor farm systems. Challenges:
    - Number of available public IP addresses at BNL
    - Configuration/management
  - Live migration of VMs to minimize downtime for hardware maintenance/issues

Xen

- Why Use Xen?
  - Open, free
  - Integrated into SL5/RHEL5
  - Performance gains from paravirtualization
  - Running Linux VMs only
- Issues
  - OS image modifications necessary to utilize paravirtualization
  - Red Hat recently announced KVM will become the default hypervisor for RHEL 5.4
    - Xen will continue to be supported in RHEL5, however
    - KVM requires a CPU supporting hardware assisted virtualization (Intel VT-x, AMD-V)

Xen Management

- Needed a mechanism to centrally manage Xen configuration and DomU installation on many hosts
  - Potentially scaling to thousands of systems
  - Desired automated setup/installation of guest domains during the Dom0 system build process
  - Wanted the ability to centrally modify various configuration parameters for guests in batch
  - Preferred a solution which could interact with our existing machine inventory database and host installation infrastructure

Xen Management (Cont.)

- Nothing available met all of our needs
  - Virt-manager
    - Did not provide the required level of automation
  - Cobbler/Koan
    - In a development stage at the time we initially investigated the use of Xen
    - Mandated the adoption of a new installation infrastructure
  - Developed a custom solution

Changes to Infrastructure

- Processor Farm's Preexisting Automated OS Deployment Infrastructure
  - SL Kickstart-based
  - PXE
    - Custom PXE management software
    - DHCP/TFTP configuration generated from a server inventory MySQL database
    - Packages obtained via locally maintained HTTP repositories
- Machine Inventory Database Table Structure Modified: Additional Fields Added
  - dom0 – hostname of a DomU system's associated Dom0
  - vdisk – name of the Dom0 disk device allocated to a DomU

Custom Solution

- Comands Executed in Dom0
  - xenconf.py
    - Automatically generates Xen configuration files for all guests associated with a Dom0
      - Fields in inventory database used as configuration parameters
    - Both “running” and “installation” configurations generated
      - /etc/xen/HOSTNAME
      - /etc/xen/HOSTNAME_install
      - Installation configurations point to install kernel/initrd
  - installguest.sh
    - Installs a named guest via its Xen installation configuration file

Custom Solution (Cont.)

- allguests.sh
  - Performs management operations on all guests associated with a Dom0
    - create
    - destroy
    - reboot
    - install
      - All VMs installed in parallel
      - Output/input redirected to/from unused virtual terminals
      - “Running” config automatically started when complete
- Automatic Guest Installation During Dom0 Provisioning
  - Dom0 OS build contains a modified rc.local init script
  - Executes the following on first boot:
    - xenconf.py
    - allguests.sh install

Custom Solution (Cont.)

- Centralized Configuration Changes
  - Modify necessary fields in inventory DB
  - Rerun xenconf.py on Dom0 host(s)
    - Safe to run xenconf.py on systems where configuration hasn't changed
  - Issue appropriate “xm”, “allguests.sh” or “installguest.sh” commands on necessary Dom0 host(s) to make the changes take effect
    - Executing these commands manually for now
      - Could be implemented via cron

Custom Solution (Cont.)

- Example Execution
  - Adding an additional virtual host to a Dom0: acas0002
    - WWW interface used to populate fields in inventory DB prior to execution

```
[root@testdom0 ~]# ls /etc/xen/acas*
/etc/xen/acas0001 /etc/xen/acas0001_install
[root@testdom0 ~]# /guest/xenconf.py
Generating Xen configurations under /etc/xen/:
acas0001
acas0002
[root@testdom0 ~]# ls /etc/xen/acas*
/etc/xen/acas0001 /etc/xen/acas0001_install /etc/xen/acas0002
/etc/xen/acas0002_install
[root@testdom0 ~]# /guest/installguest.sh acas0002
...
[root@testdom0 ~]# xm list
Name                                      ID Mem(MiB) VCPUs State   Time(s)
Domain-0                                   0     1211     8 r----- 966463.4
acas0001                                  26     3267     2 r----- 2489156.1
acas0002                                  27     3267     2 r-----  35082.7
```

Example Configurations

Example Xen DomU “Running” Configuration

```
#################################################################
# /etc/xen/acas0001
#
kernel = "/guest/vmlinuz-ELxenU"
ramdisk = "/guest/initrd-ELxenU.img"
builder='linux'
memory = 3268
name = "acas0001"
cpus = "0-1"
vcpus = 2
vif = [ 'mac=00:16:3e:00:00:01, bridge=xenbr0' ]
disk = [ 'phy:sdb1,xvda,w' ]
root = "LABEL=/ ro"
```

Example Configurations (Cont.)

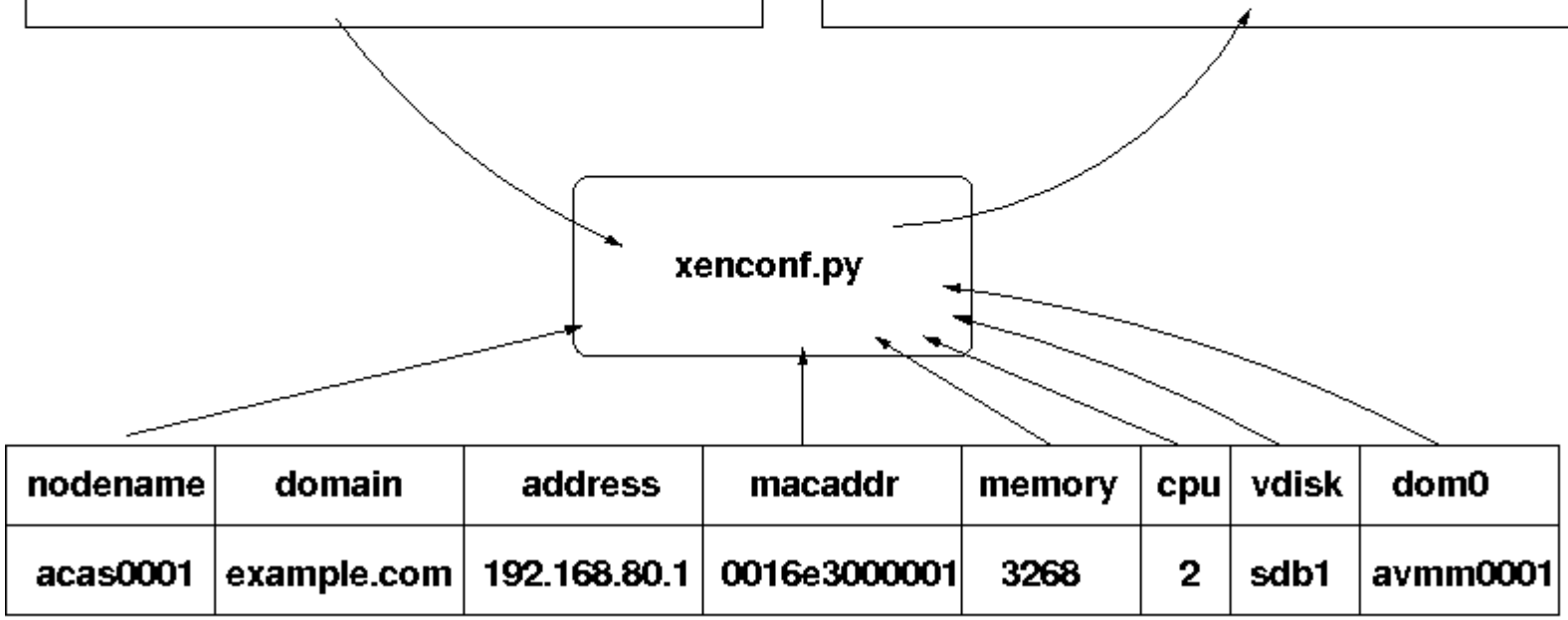
Example Xen DomU “Installation” Configuration

```
#################################################################
# /etc/xen/acas0001_install
#
kernel = "/guest/vmlinuz.xen_install"
ramdisk = "/guest/initrd.img.xen_install"
builder='linux'
memory = 3268
name = "acas0001_install"
vcpus = 1
vif = [ 'mac=00:16:3e:00:00:01, bridge=xenbr0' ]
disk = [ 'phy:sdb1,xvda,w' ]
extra = "ks=http://repo.example.com/SL4_Kickstart/atlas-ks-XEN.cfg"
```

Configuration Generation

```
# Template Xen Configuration
kernel = "/guest/vmlinuz-ELxenU"
ramdisk = "/guest/initrd-ELxenU.img"
builder='linux'
memory = %XEN_MEMORY%
name = "%XEN_NODENAME%"
cpus = "%XEN_PINCPUS%"
vcpus = %XEN_VCPUS%
vif = [ 'mac=%XEN_MAC%, bridge=xenbr0' ]
disk = [ 'phy:%XEN_DISK%,xvda,w' ]
root = "LABEL=/ ro"
```

```
# acas0001 Xen Configuration
kernel = "/guest/vmlinuz-ELxenU"
ramdisk = "/guest/initrd-ELxenU.img"
builder='linux'
memory = 3268
name = "acas0001"
cpus = "0-1"
vcpus = 2
vif = [ 'mac=00:16:3e:00:00:01, bridge=xenbr0' ]
disk = [ 'phy:sdb1,xvda,w' ]
root = "LABEL=/ ro"
```



Inventory Database