Outline

• KVM: what made it successful?
• What came out of it?
• How do you use it?
• What will the future bring?
What is KVM?

From: Avi Kivity <avi@qumranet.com>
To: linux-kernel <linux-kernel@vger.kernel.org>
Subject: [PATCH 0/7] KVM: Kernel-based Virtual Machine (v2)
Date: Mon, 23 Oct 2006 15:28:48 +0200

The following patchset adds a driver for Intel's hardware virtualization extensions to the x86 architecture. The driver adds a character device (/dev/kvm) that exposes the virtualization capabilities to userspace. Using this driver, a process can run a virtual machine (a "guest") in a fully virtualized PC containing its own virtual hard disks, network adapters, and display.

Using this driver, one can start multiple virtual machines on a host. Each virtual machine is a process on the host; a virtual cpu is a thread in that process. kill(1), nice(1), top(1) work as expected.

...

--
error compiling committee.c: too many arguments to function
KVM's distinctive features

- Design for future hardware
- Reuse Linux kernel infrastructure
- A witty maintainer
A KVM maintainer's life is a tough one
To integrate or not to integrate?

- Linux community is bigger and can be a bit unfriendly
- But KVM cannot fork the kernel
  - It is not Android!
- Start with quick and painless integration
- Features can get in later
- This was quite successful!
The golden rule of contributing to Linux

Slip your stuff in, in small increments, and with good reasons for why you aren't crazy

Every new crazy feature should be hidden in a nice solid "Trojan Horse" gift: something that looks obviously good at first sight.

-- Linus Torvalds
To integrate or not to integrate?

- Integration with QEMU was much less important
  - x86 KVM usually run with the qemu-kvm fork
  - Several features still implemented only in qemu-kvm

- Three rules for sanity:
  - Work as much as possible with upstream
  - Try to get your stuff upstream
  - Merge regularly

- < 7000 lines of code still to be merged
Crazy features contributed by KVM

- Scheduler notifiers
  - lightweight guest → kernel → guest context switching
- MMU notifiers
  - swapping and overcommitting guest memory
- Samepage merging
- Transparent huge pages

... and for QEMU:

- Virtual machine migration
- SMP guests
- Stable guest hardware
- Virtual machine monitor RPC
- KVM support
KVM's killer feature

- The hypervisor will only run the VM for you
- No policy decision is the hypervisor's business:
  - Security checks
  - Memory management & scheduling
  - NUMA
  - ...
- Not just an “economic” decision, it gives enormous flexibility!
A free software success story

- Built around free components itself
  - Linux, QEMU
  - Successfully contributed back to those projects!
- Stimulated the growth of a large and open community
- Enabled the creation of a large software ecosystem

- Technical and social factors reinforce each other!
So, how do you use KVM?

$ qemu-kvm -S -M rhel6.2.0 -cpu Conroe

-enable-kvm -m 2048 -smp 1sockets=1,cores=1,threads=1 -name z-win7x86-1 -uuid e3e19b36-f6b7-4ab9-b604-1f8b5c471bda -smbios type=1,manufacturer=Red Hat,product=RH,version=6Server-6.1.0.2,serial=50C1C6F0-B18B-11DE-ADG1-00215EC7FC0C_00-1A:64:E7:0E:EO,uuid=e3e19b36-f6b7-4ab9-b604-1f8b5c471bda -nodefconfig -ndefaults -chardev socket,id=charmonitor,path=/var/lib/libvirt/qemu/z-win7x86-1.monitor,server,nowait -mon chardev=charmonitor,id=monitor,mode=control -rtc

Voila! We have a virtual machine

Don't forget the fine print
So, how do you use KVM?
Enter libvirt!

The goal of libvirt: to provide a common and stable layer sufficient to securely manage domains on a node, possibly remote.
Enter libvirt!

- An API that can be used both locally and remotely
- A daemon to:
  - Handle remote communication
  - Configure a server to run a virtual machine (networking, SELinux, ...)
  - Handle the virtual machine lifecycle for KVM
- Support for KVM and Xen around mid 2007

---

virsh, virt-manager

libvirt

QEMU

Xen

Linux+KVM

Xen tools
Enter libvirt!

- An API that can be used both locally and remotely
- A daemon to:
  - Handle remote communication
  - Configure a server to run virtual machine (networking, SELinux, ...)
  - Handle the virtual machine lifecycle for KVM
- Support for KVM and Xen around mid-2007
  - Now: Hyper-V, ESX, LXC, UML, ...
Enter libvirt!

virsh, virt-manager, Boxes, ...

libvirt

LXC

QEMU

Linux+KVM

Xen

Hyper-V

ESX
libvirt features

- Comprehensive host management
  - Virtual machines
  - Virtual networks (bridging, NAT, VEPA, ...)
  - Storage (local disks, SAN, NFS)
- Isolation between virtual machines and host
  - Compromised virtual machine cannot access other VMs or hosts
  - Containment in case of hypervisor breaches
  - Based on SELinux and cgroups, shines on KVM!
libvirt users: virt-manager
Libvirt users: Boxes
libvirt users: other APIs

- libvirt-snmp
  - Access domain information via SNMP
- libvirt-qmf
  - Manage hosts, domains, pools via AMQP
  - An agent running within Matahari
- libvirt-cim
  - Open standard to describe guests and resource pools
  - CIM objects abstract the XML schemas
libvirt users: other APIs

- **libvirt-glib / libvirt-gobject**
  - Access libvirt event loop from GLib
  - Provide a GObject-based API for libvirt objects
  - Usable in many languages via GObject introspection

- **libvirt-gconfig**
  - an API that abstracts the libvirt XML schemas
Other virtualization tools

- **virt-install**
  
  libvirt-based, command-line virtual machine installation

- **Oz**
  
  Part of Aeolus project, next-gen virtual machine installer

- **libguestfs/guestfish**
  
  library and utilities to inspect virtual machine disks
KVM virtualization tools summary

Local & remote!

SNMP, CIM, QMF
virsh, virt-install
virt-manager
guestfish, FUSE

libvirt
libguestfs

QEMU

Linux+KVM
But is this how you really use KVM?
The challenge

• Manage tens of thousands of virtual machines
• Each virtual machine potentially accessible from hundreds (thousands?) of nodes
Large-scale virtualization architecture

**Management engine**
- Provisioning
- Load balancing
- High availability
- Power saving
- Storage
- Node lifecycle
- Monitoring & reporting

**Host agent**
- VM lifecycle
- Security
- VM storage

**Guest agent**
- Monitoring
- Integration
- Management actions

**Host agent**
Large-scale virtualization solutions

- OpenNebula, OpenStack Compute (Nova)
  - Cloud deployment
- Ganeti
  - Distributed cluster, replicated storage
- oVirt
  - Datacenter virtualization, VDI
# Large-scale virtualization solutions

<table>
<thead>
<tr>
<th>Feature</th>
<th>ONE</th>
<th>Nova</th>
<th>Ganeti</th>
<th>oVirt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management API</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Web control panel</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Libvirt-based</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Live migration</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Distributed storage</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Replicated storage</td>
<td>DRBD</td>
<td>Ceph</td>
<td>DRBD</td>
<td>No</td>
</tr>
<tr>
<td>Specialized distro</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Guest agent</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Non-KVM hypervisors</th>
<th>Xen, ESX</th>
<th>Many</th>
<th>Xen</th>
<th>None</th>
</tr>
</thead>
</table>
oVirt

- An open platform for datacenter virtualization
- Comprises multiple integrated projects
- Development also started at Qumranet
- Forms the basis of Red Hat's RHEV 3.0 product
- Backed by Red Hat, IBM, NetApp, Cisco, SuSE, Intel
oVirt

- Single sign-on
- Clipboard management
- Linux + Windows
- Libvirt-based, KVM-only
- Written in Python
- Can run in <100 MB
- REST
- Web portals (admin, user)
- SDK/command line
VDSM

- High-level virtualization management API
- Register with oVirt engine
- VM lifecycle via libvirt
- Storage management (iSCSI, LVM, thin provisioning)
- Monitoring host and VMs
- Interaction with guest agent
Roadmap: oVirt engine

- Complete new web user interface
  - Based on GWT, removes last Windows dependency
  - Integrate reporting into the web UI
- Non-admin API
- Gluster support
Roadmap: oVirt node

- VDSM
  - Live snapshots
  - Live storage migration
  - Shared disks
  - Backup API
  - Service level agreements
- Support additional distributions
Roadmap: libvirt

- Security
  - Fine-grained access control per (user, object, action)
  - Confined LXC containers
- Networking
  - Open vSwitch integration
  - SR-IOV NIC pools
- Interaction with guest agents
Roadmap: QEMU

- Storage
  - virtio-scsi
  - Thin provisioning
  - Live storage migration
- Unified object model
  - Hot-plug improvements
Roadmap: KVM

- Virtualized performance counters
- Networking performance improvements
- Open vSwitch integration
Links and other cool projects

- OpenNebula: http://opennebula.org
- OpenStack: http://openstack.org
- oVirt: http://ovirt.org
- Ganeti: http://code.google.com/p/ganeti/
- libvirt: http://libvirt.org
- Aeolus: http://aeolusproject.org/

- Questions?