KVM in an HPC Cloud
by Conrad Wood
Definition HPC Cloud:

- Can accommodate any “real” network configuration
- Is at least as fast as a common physical networks
- Is highly-available (feature of a cloud)
- “Real” services also available as HPC Services
- “Behaves” like several “real” datacenters

→ KVMs unique Flexibility allows this
The Cornerstones

- The storage: fast and reliable (consistent)
- The network layer: fast, multi-tenant, filterable
- CPUs: mostly to support the above, a “visionary” outlook at the end

→ We discuss network and storage and their relation to CPU cycles
I . The Storage

- Options: ZFS, Netapp, Gluster and others...

- Non-Integration: consistency on snapshot

- Providing a backup without stopping the VM:
  - Caches need to be flushed
  - Blocks written in the right order (journals/write-barriers)

- Mirroring with DM/ZFS to provide live-migration
Storage Consistency

- In a cloud, access to the guest-os is limited to VM-Owner (not cloud-administrators)
- Backups have to be taken on-line
- Most snapshotting solutions require long delays in stopping the machine or slow down with each snapshot

→ a snapshot should be synchronized to a write-barrier
Data travels by elevators on guest, on host, on storage system.

Only the storage system knows how to order blocks.

Other elevators increase latency and have little benefit.

→ Guests need to be modified to 'detect' virtual I/O and disable elevators. A common indicator is needed.
- Side effect of “elevating”: increase block I/O sizes
- Each interrupt is costly
- Block-drivers, e.g. Virtio, may indicate larger block-sizes to the guest
- Worse: switching memory contexts (MMU) (IOMMU has limited effect)
VM-Migration: mesh-like storage access
Mirroring helps with storage-migration
Snapshot required to be part of the block-layer
Replication

VM

1

Array 1

2

Array 2

Migration Scenario with mirrored arrays
II. Networking

• Very recent Hardware includes options for multi-tenant / multi-VM:
  • SR-IOV (exposed pci-cards)
  • VTAG (double-tagging traffic)

• Missing:
  • Live-migration
  • Filterable traffic

• Others: e.g. vlan-in-vlan, IP-tunnelling
Current Network software, e.g.

- OpenVSwitch (userland)
- Linux Bridging
- IP-over-IP

can not perform as well due to extensive checks.

→ The VM traffic needs to bypass as much as possible as early as possible
Vhost modifications

- Reduce amounts of packets by suggesting a 64k MTU to the guest
- Add routing information to the sk_buf and deliver it straight out to the physical interface (bypassing tun/tap, bridging code etc..)
- Disable CRC/Seg. hardware offloading for speed
- Disjunct management of routing information
- Multi-thread vhost = more than 7Gbit/s per TCP Stream → Multi-Threading on multi-cores...
Multi-threading Network Issues

- Multi-thread = better throughput
- Possible starvation of vital number-crunching tasks
  - 'adaptive' pinning of KVM user-processes to CPUs
  - VHost threads follow user-processes (self-regulating effect)
- Sender and receiver must match
  = maintain max. performance over time
Advanced Networking

- Given the speed of a single VM, no IP-stack by itself can deliver faster than that.
- Services dealing with traffic for multiple VMs need to be distributed in the same manner as the VMs.
- For example, traffic inspection for firewalls need to be handled by the same code, in the same core as the VHost processes.
- VHost modified to utilise iptables' connection tracking API to provide this.
Distributed Traffic #1 (this...)

- Firewall 1
  - VM1
  - VM2
  - VM3
Distributed Traffic #2
(...becomes)
Upstream commitment:

- Results in our environment show huge speed improvements (1.2 Gbit/s → 7Gbit/s !)
- Introduction of a “VHost direct routing API”
- VHost direct routing API to allow for the addition of new high-speed networking modes with ease
III. High Availability

- Any changes to devices need to be performed on-line

- PCI-HotPlug: works as expected

- HotPlug: RAM & CPU requires support in guestOS, BIOS, QEMU and KVM
• Any changes to devices need to be performed on-line

• PCI-Hotplug: works as expected

• HotPlug: RAM/Memory/CPU requires support in guestOS, BIOS, QEMU and KVM

• The future: KVM-native-tool?
Given current network latency improvements, it becomes feasible to start scheduling work across high-speed links (e.g. Infiniband).

Using continuous live-migration to keep ram in-sync via RDMA (40GigE/Infiniband).

Overhead of slow RAM vs. more Cores.

Allows for VMs with more Cores than one physical node.
Thank you for listening. - Profitbricks

http://www.profitbricks.com

Questions?