# Virtual Machines:
# Performance Optimization

## Lab. 3

# Outline

- Impact in performance and use of hardware assist
- Types of virtualizations with Xen

# Task1 (20%)

- Create a minimal HVM installation disk domU with installed
  - Use a “lv” of at max 500MB
  - Configure the domU for “console” mode
- Configure the HVM machine to have access to PV instalation (reuse /)

# To “reuse” a disk image created in a HVM to boot a PV machine

- Copy hvm.cfg file to pv.cfg
  - change “kernel” and “ramdisk” by the existing Dom0 /boot
- Disk devices
  - Replace “xvda” by “hda”  in “phy:”
  - Add root device to boot pv kernel (Assuming hvm installer used 1st partition)
    - root= /dev/xvda1 ro
- Fix console
  - Set extra options for pv kernel
    - extra="console=hvc0 "
  - Edit the /etc/inittab in image file and replace
    - 2:23:respawn:/sbin/getty 38400 tty2
    - By
    - 2:23:respawn:/sbin/getty 38400 **hvc0**
  - **After that, both HVM and PV will work with a single image**
  - **Remember do not boot simultaneously both VM!!**

# The other way around: how to use a PV image on HVM

- Add a partition table to the PV volume
  - Many steps: parted, resizefs, grub, etc… (prone to errors)
  - Not recommended

- Our take: two volumes (one new volumne with PT for /boot and use the PV for /)
  - Create /boot and install grub on it using the installer
  - Instruct  the kernel to use PV volume on /
  - HVM will use the boot and PV is just the same!

# Task 2: CPU Bound Workloads (30%)

- Compare the performance of two SpecCPU benchmarks selected in Lab1 using:
  - Native
  - PV without VT-X
  - PV with VT-x
  - HVM-on-PV

- Where the native performance went? (in the application with the strongest slowdown)
  - Just with PV with VT-x

# Performance overheads analysis

- Unfortunately, PMU seems to be inaccessible to domU with current processors/Xen/kernels versions
  - `dmesg| grep -I pmu`
    - No driver found in kernel (?)
  - Little to no information on the webs…å

- Still the linux kernel has some interesting tracing support (and perf can access to it)
  - `perf list | grep xen`
    - xen:mc  Hypercalls
      - **Denoted as (mc) Multicalls  due to xen “combine” multiple hypercalls in a single “multicall”**
      - **Key is “xen:xen_mc_entry”**
    - xen.mmu Memory management
      - **Includes shadow table, page allocation, page release, …**
      - **Key are “xen:xen_mmu_flush_tlb , xen:xen_mmu_*_ptpage”**
    - xen.cpu Context switches

- Kernel tracing is much more powerful and complex

- Caveat: With “HVM” (even with PV drivers) the kernel don’t expose those tracepoints
  - Requires the use of “xentrace”

# Task 3: QoS for SpecJBB (20%)

- Configure two domU (using best virtualization) for running SpecJBB with latest Oracle JDK
  - Recompile the benchmark

- Configure CPU scheduling to achieve at least a 80% of native performance (SLA) in one of them (when executed concurrently)
  - Use remote ssh commands to synchronize execution
  - Make the configuration persistent

- (*)If answer is “impossible”...
  - How much is the minimum performance that we can guarantee?

# Task 4: SpecWeb (30%)

- Compare the performance of SpecWeb2005
  - Native
  - HVM-on-PV
    - With separate domU for Clients+BeSim/Webserver
  - Add PCI-Pass-through for the network

- Multisystem run (PV-PCI-PT)
  - Configure besim and clients in a separate DomU
  - Run Dom0, and besim, clients in a single CPU (CPU0)
  - Migrate besim, client to another host (PCI-PT)

# Grades

- Guide and presentation (25%)
- 75%
  - Content Lab2 (25%)
  - Content Lab3 (50%)