Aula 11 - Gerenciamento de CPU e tempo

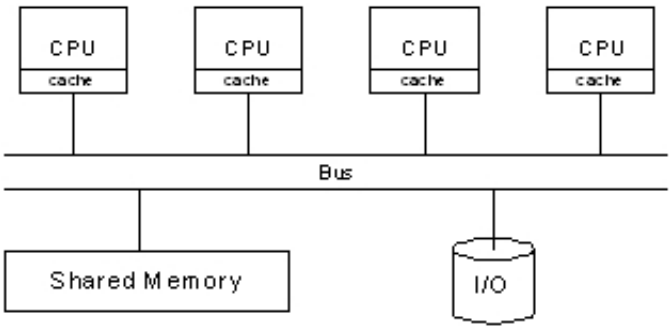
Sobre

- Objetivos:
  - Entender o funcionamento de processadores multi-core e as implicações para máquinas virtuais.
  - Ajustar prioridades de execução para diferentes máquinas virtuais.
  - Entender como manter e ajustar relógios virtuais.
  - Ajustar o host para balanceamento de IRQs e melhorar performance.

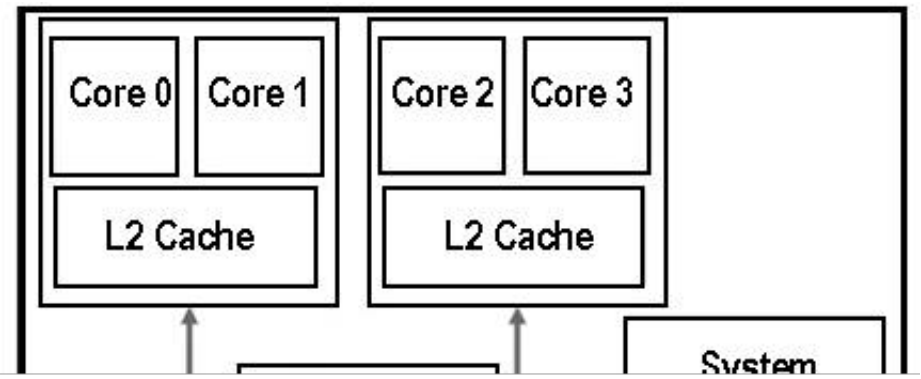
Overcommit de CPU

- O KVM suporta overcommitting de CPUs virtuais.
- CPUs virtualizadas podem ser sobrecarregadas na medida em que os guests tenham suas demandas atendidas.
- CPUs virtualizados estão utilizam melhor overcommit quando cada guest virtualizado tem apenas uma única CPU.
- O escalonador Linux é muito eficiente.
- Você não pode overcommit guests SMP em mais do que o número de núcleos físicos. Por exemplo, um guest com quatro vCPUs não deve ser executado em uma máquina com um processador de dois núcleos.
- Overcommitting do número de CPUs físicas em relação ao número de CPUs virtuais causa degradação significativa no desempenho.

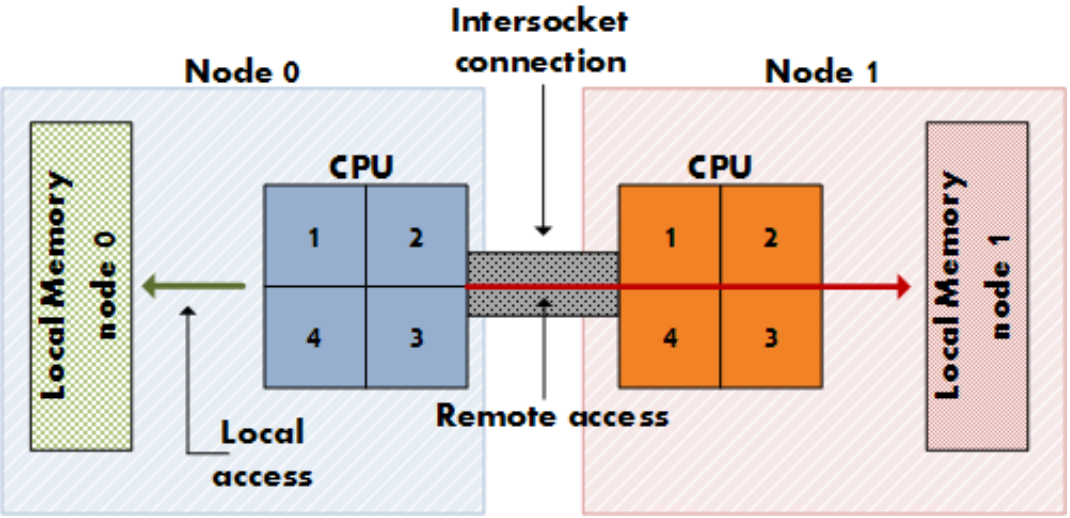
Arquitetura SMP



Arquitetura Multicore



Arquitetura NUMA



Topologia

```
# cat /proc/cpuinfo
processor       : 0 <logical cpu #>
physical id     : 0  <socket #>
siblings        : 16 <logical cpus per socket>
core id         : 0  <core # in socket>
cpu cores       : 8  <physical cores per socket>

# cat /sys/devices/system/node/node*/cpulist
node0: 0-3
node1: 4-7
```

Distribua carga de IRQ

- IRQ affinity: aliviar tratamento de interrupções e distribuir carga entre as CPUs.

```
# cat /proc/interrupts
# echo 2 > /proc/irq/217/smp_affinity
```

- http://www.alexonlinux.com/smp-affinity-and-proper-interrupt-handling-in-linux
- IRQ Balance: http://www.irqbalance.org

```
# irqbalance --debug
IRQ delta is 0
IRQ delta is 0, switching to power mode
Package 0:  cpu mask is 0000000f (workload 10901)
        Cache domain 0: cpu mask is 00000003  (workload 179)
                CPU number 0  (workload 120)
                CPU number 1  (workload 176)
                  Interrupt 21 (ethernet/55)
          Interrupt 22 (storage/0)
          Interrupt 18 (storage/0)
          Interrupt 30 (legacy/0)
        Cache domain 2: cpu mask is 0000000c  (workload 280)
                CPU number 2  (workload 120)
                CPU number 3  (workload 120)
          Interrupt 28 (storage/158)
          Interrupt 16 (legacy/0)
  Interrupt 24 (other/4097)
  Interrupt 25 (other/3155)
  Interrupt 27 (other/1654)
  Interrupt 26 (other/1651)
  Interrupt 4 (other/0)
```

Configurando uma CPU específica no KVM

```
# kvm -cpu ?
x86           [n270]
x86         [athlon]
x86       [pentium3]
x86       [pentium2]
x86        [pentium]
x86            [486]
x86        [coreduo]
x86          [kvm32]
x86         [qemu32]
x86          [kvm64]
```

```
x86       [core2duo]
x86         [phenom]
x86         [qemu64]

# kvm -cpu host

# kvm -smp 2
```

Ajustando o uso de CPU de VMs

- Prioridade:
  - nice
  - renice
  - chrt (específico do Linux)
- Posicionamento de vCPU em CPU real:
  - taskset (multicore e SMP): Running your VM on specific CPUs: http://www.linux-kvm.com/content/tip-running-your-vm-specific-cpus
  - numactl

Ligando e desligando CPUs

- Desligando uma CPU:

```
echo 0 > /sys/devices/system/cpu/cpu1/online
```

- Ligando:

```
echo 1 > /sys/devices/system/cpu/cpu1/online
```

Relógio virtual

- Fontes de tempo:

```
# cat /sys/devices/system/clocksource/clocksource0/current_clocksource
kvm-clock|tsc|hpet|acpi_pm

# dmesg | egrep -i "(time|clock|tsc)"
```

- NTP
  - Se usar kvm-clock, o relógio do guest acompanha. Instalar daemon NTP somente no host.
  - Se o guest não suporta kvm-clock, colocar NTP no guest pois o clock varia e

muito.

- Desabilite o hwclock (Don't run hwclock on guests running kvmclock)

```
# cd /sbin/
# ls -l hwclock*
# mv hwclock hwclock.dist
# touch hwclock
# chmod +x hwclock
```

TSC (timestamp counter)

- Flags na cpu: tsc, nonstop_tsc, constant_tsc
- Se CPU não tem nonstop_tsc:
  - Mensagem do kernel: Marking TSC unstable due to TSC halts in idle
- O kernel tentatá usar HPET ou ACPI se o TSC não for confiável

HPET (High Precision Event Timer)

- Guests que usam HPET consomem **MUITA** CPU inutilmente, devido ao alto número de interrupções.
- Alguns guests podem suportar TSC e HPET, mas não usam TSC preferencialmente. (CentOS 5, usar **kvm -no-hpet**)
  - http://home.coming.dk/index.php/don-t-emulate-hpet-with-kvm
- Mais referências:
  - Time and KVM - best practices: http://kerneltrap.org/mailarchive/linux-kvm/2010/3/21/6259882/thread
  - Timekeeping Virtualization for X86-Based Architectures

Referências

- Otimização de desktops com kernel Linux no /proc e /sys
- http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Virtualization/chap-Virtualization-KVM_guest_timing_management.html
- Conselhos preciosos sobre NUMA: http://permalink.gmane.org/gmane.comp.emulators.kvm.devel/56323
- Controlling guest CPU & NUMA affinity in libvirt with QEMU, KVM & Xen
- Why My Two vCPU VM is Slow: http://lonesysadmin.net/2008/04/22/why-my-two-vcpu-vm-is-slow/

- blog com vários benchmarks: http://vmstudy.blogspot.com/
- Best practices for KVM: http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaat/liaatbestpractices_pdf.pdf
- http://kerneltrap.org/mailarchive/linux-kvm/2010/3/14/6259558/thread

Extra

- virtio-serial: http://www.linux-kvm.org/page/VMchannel_Requirements
- Watchdog: http://rwmj.wordpress.com/2010/03/03/what-is-a-watchdog/
- kvm: the Linux Virtual Machine Monitor: http://www.kernel.org/doc/ols/2007/ols2007v1-pages-225-230.pdf