



High Performance I/O with NUMA Systems in Linux

Lance Shelton

How prevalent is NUMA?

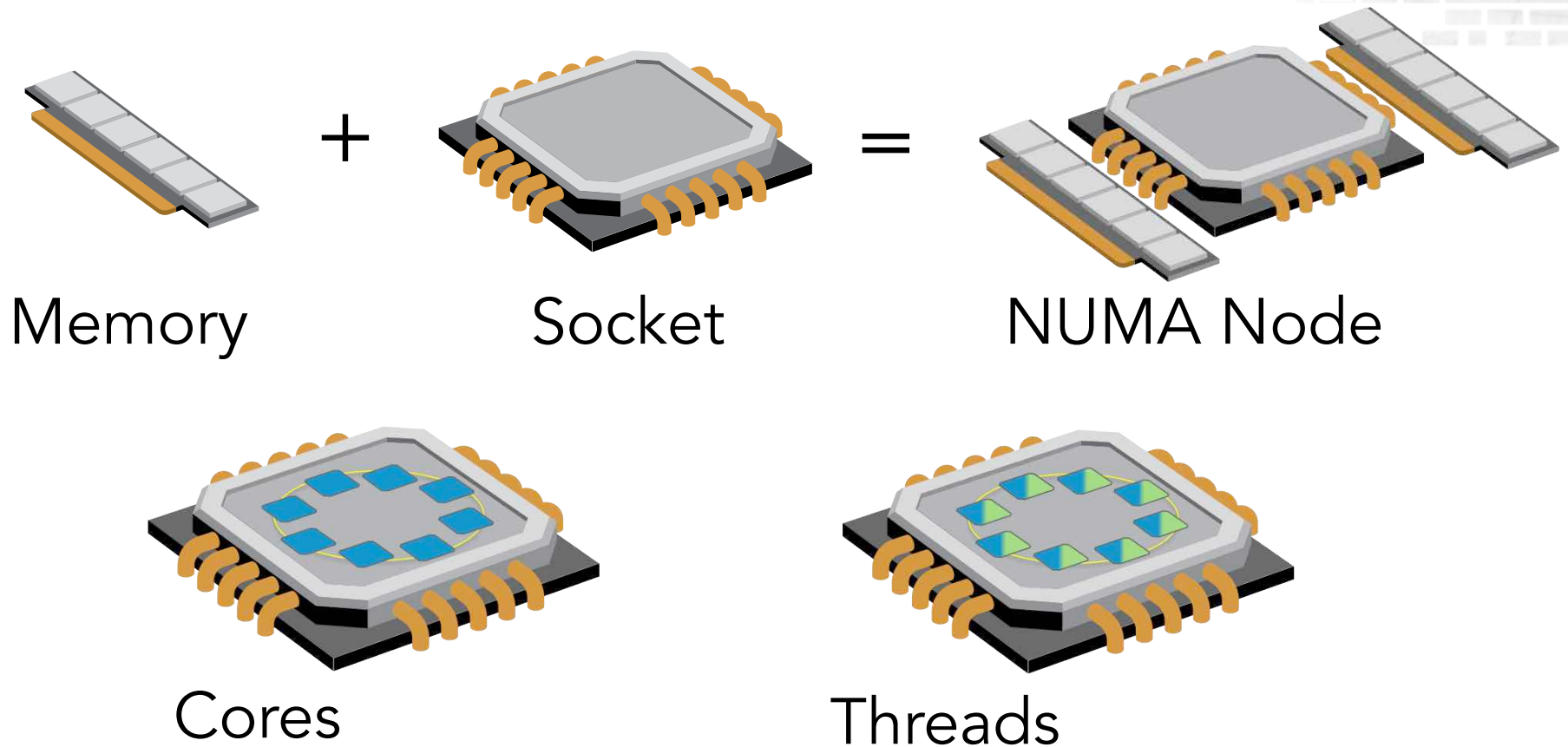
- ▶ All major vendors
 - HP, Dell, IBM, Cisco, SuperMicro, Hitachi, etc.
- ▶ As small as 1U
- ▶ 2, 4, and 8 socket systems
- ▶ 2 to 10 cores per socket
- ▶ Number of cores doubles with HyperThreading
 $8 \times 10 \times 2 = 160$ CPU cores

NUMA is mainstream.

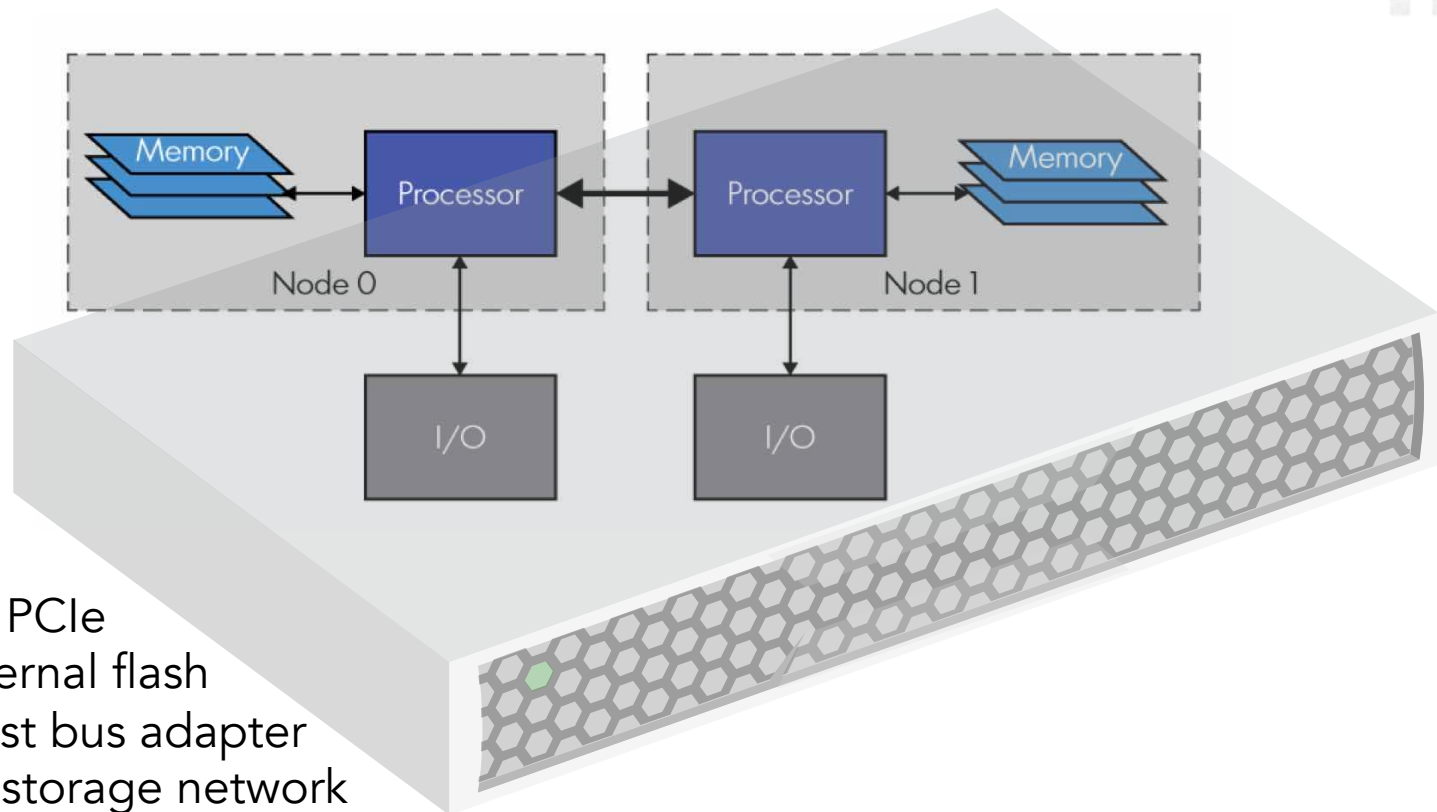
Why use large NUMA servers?

- ▶ NUMA is key to the current evolution of performance
 - Storage rates are drastically increasing
 - Processor speeds are stagnant
 - Multi-core cannot scale infinitely without NUMA
- ▶ NUMA meets the needs of the application
 - No partitioning is required
 - Faster than clustering multiple servers
 - Works well in combination with scale out

Nodes, Sockets, Cores, Threads



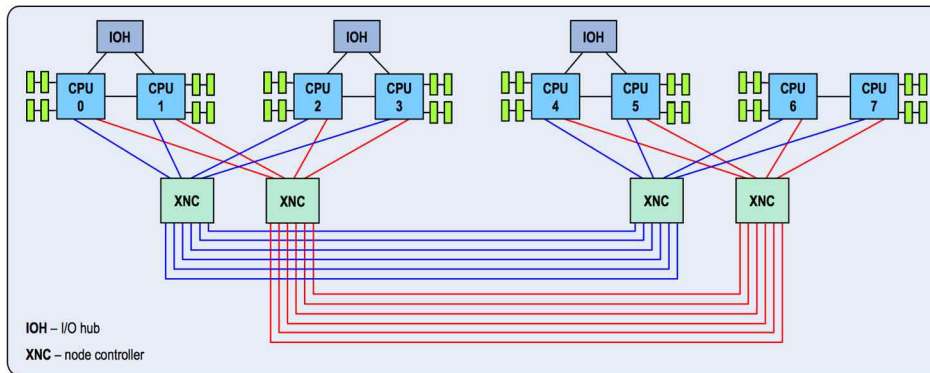
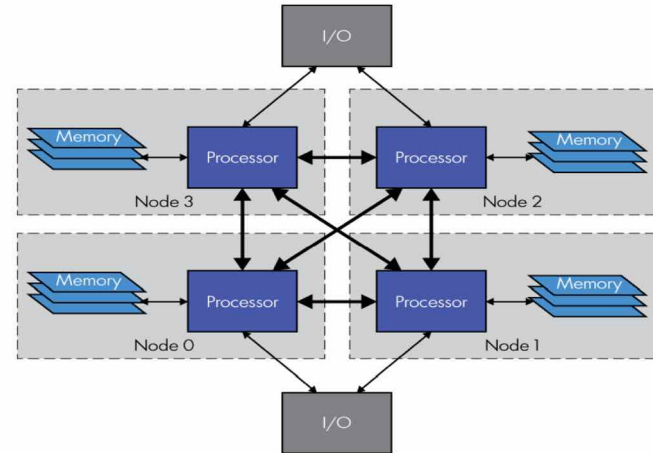
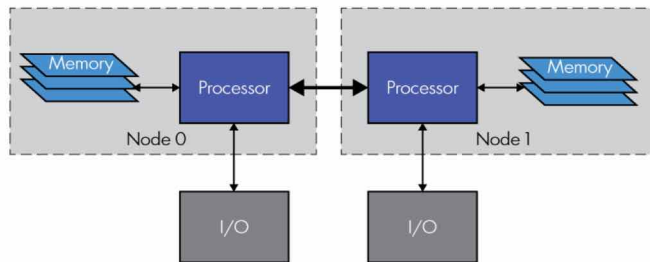
Non-Uniform I/O Access



I/O = PCIe

- Internal flash
- Host bus adapter to storage network

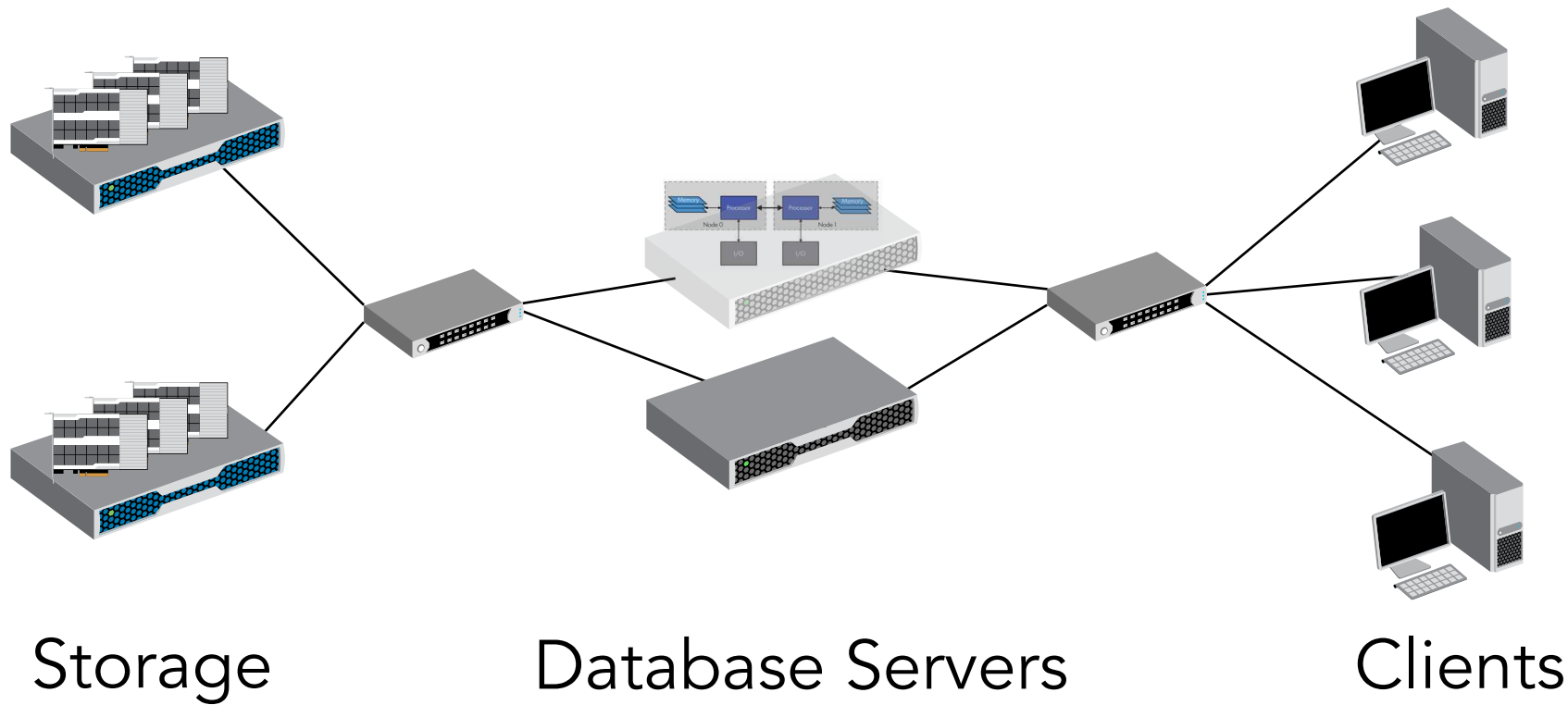
2, 4, and 8 Socket Servers



numactl --hardware

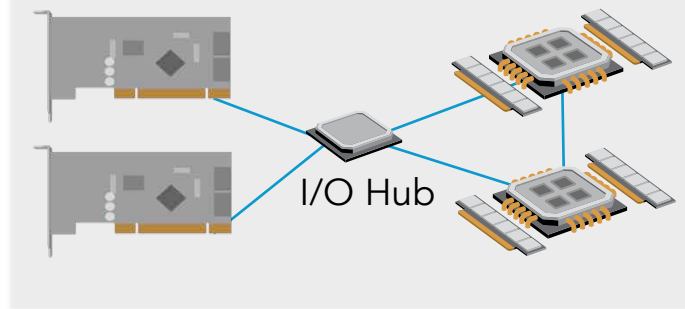
| node | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|----|----|----|----|----|----|----|----|
| 0: | 10 | 12 | 17 | 17 | 19 | 19 | 19 | 19 |
| 1: | 12 | 10 | 17 | 17 | 19 | 19 | 19 | 19 |
| 2: | 17 | 17 | 10 | 12 | 19 | 19 | 19 | 19 |
| 3: | 17 | 17 | 12 | 10 | 19 | 19 | 19 | 19 |
| 4: | 19 | 19 | 19 | 19 | 10 | 12 | 17 | 17 |
| 5: | 19 | 19 | 19 | 19 | 12 | 10 | 17 | 17 |
| 6: | 19 | 19 | 19 | 19 | 17 | 17 | 10 | 12 |
| 7: | 19 | 19 | 19 | 19 | 17 | 17 | 12 | 10 |

Enterprise Storage

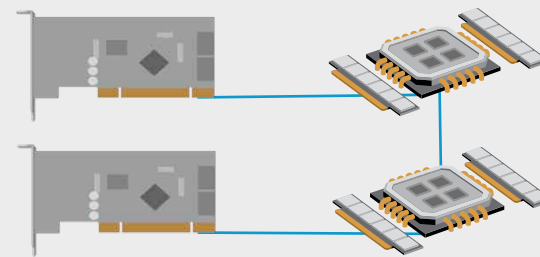


PCIe Local Nodes

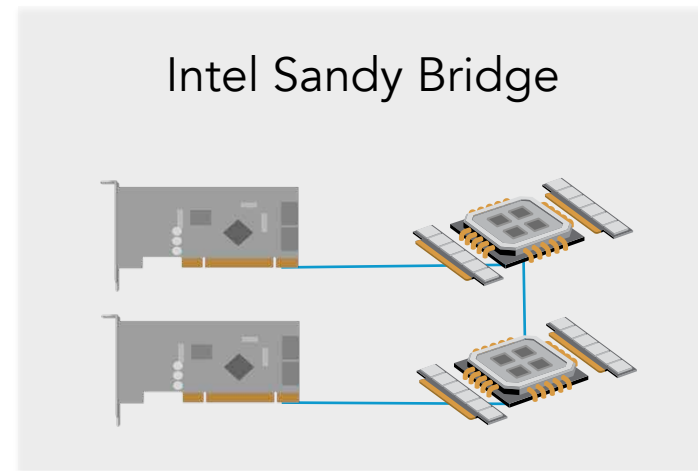
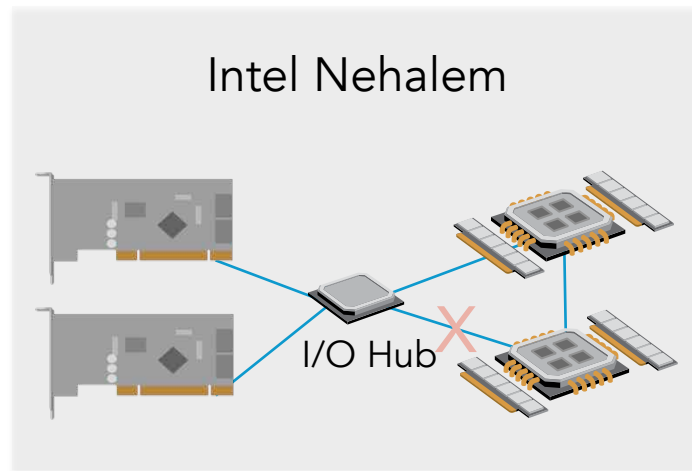
Intel Nehalem



Intel Sandy Bridge



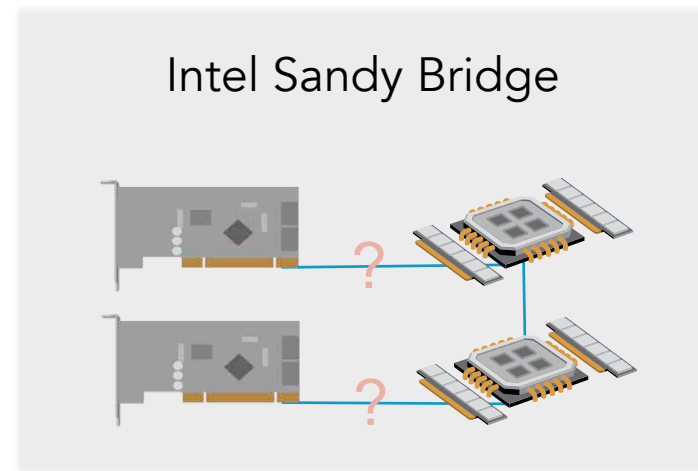
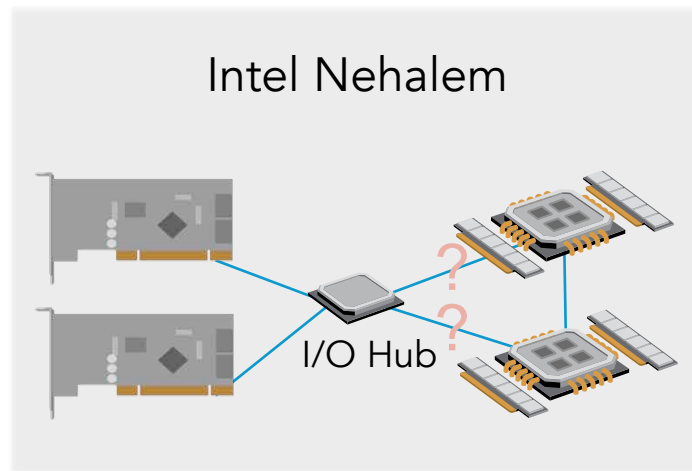
PCIe Local Nodes~~x~~



```
# cat /sys/devices/pci0000:50/0000:50:09.0/numa_node  
0
```

Only one local node is presented to the OS

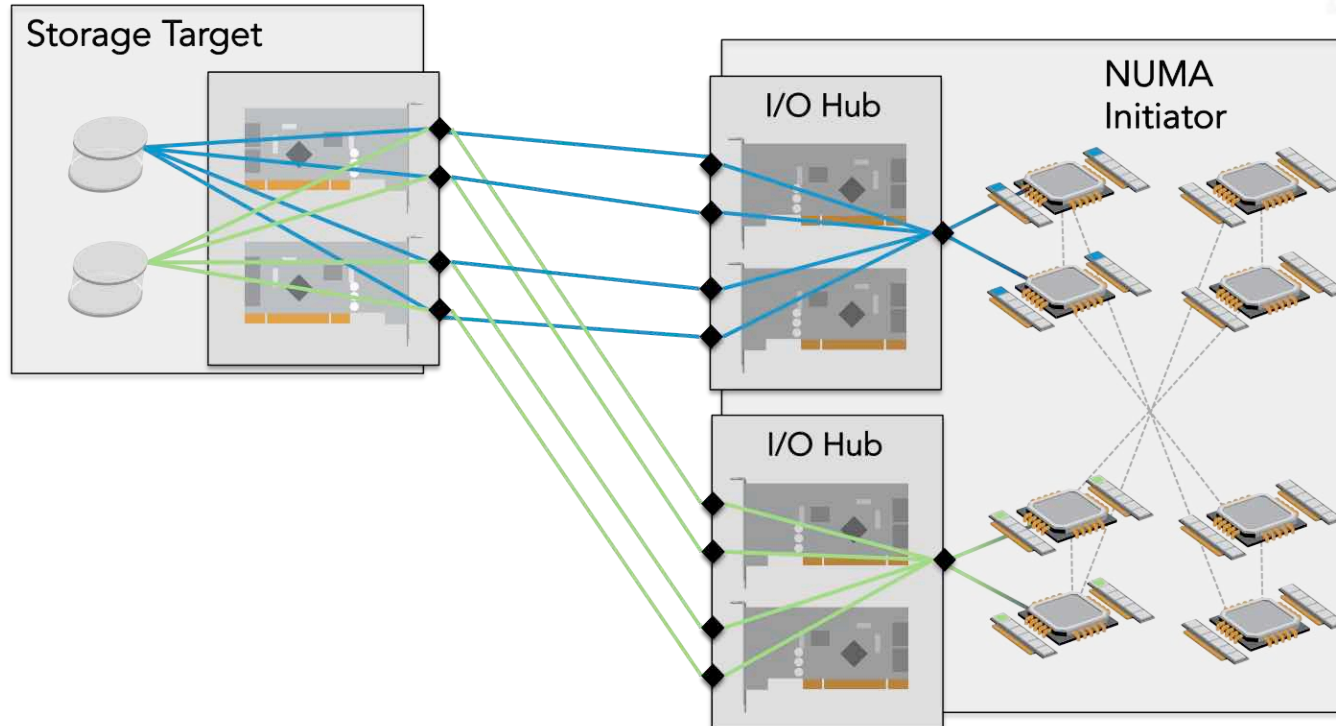
PCIe Local Node?



```
# cat /sys/devices/pci0000:50/0000:50:09.0/numa_node  
-1
```

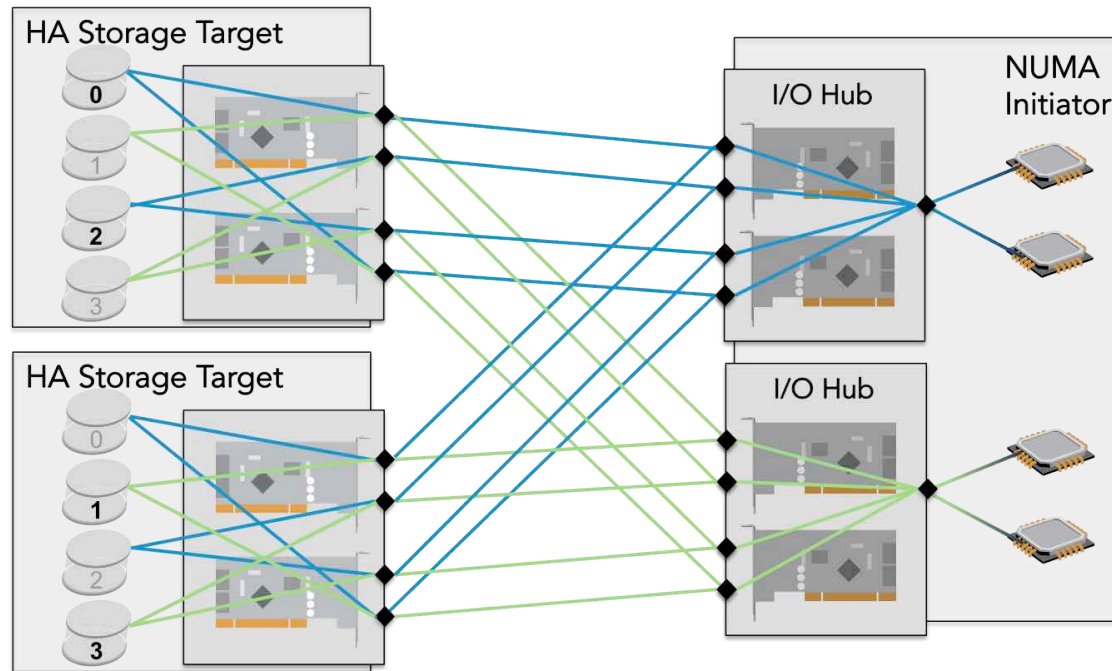
Not detected? Update the BIOS.

Localizing I/O from Storage



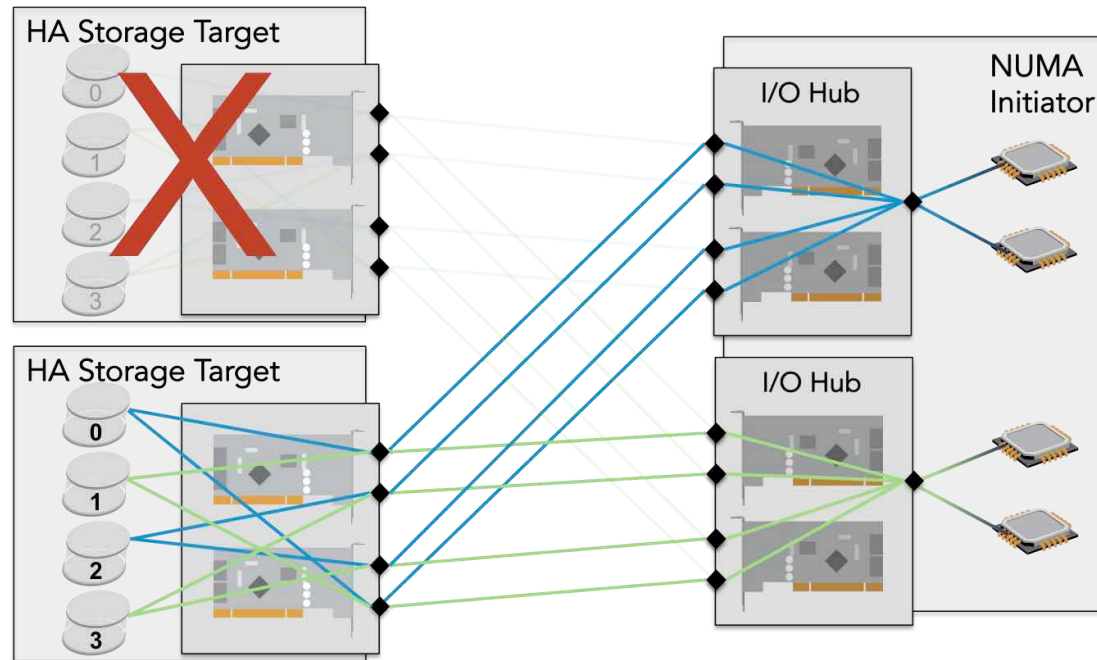
- ▶ 50-100% performance improvement vs. non-NUMA aware volume placement

Localizing I/O – High Availability



- ▶ Node locality of volumes must still be maintained

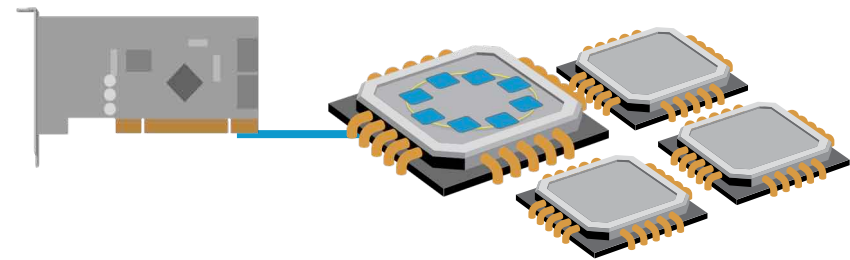
Localizing I/O – HA Failover



- ▶ Node locality maintained
- ▶ All available ports are still used

Localizing I/O – Components

- ▶ HBA placement
- ▶ Interrupt affinity
- ▶ Kernel thread affinity
- ▶ Application affinity



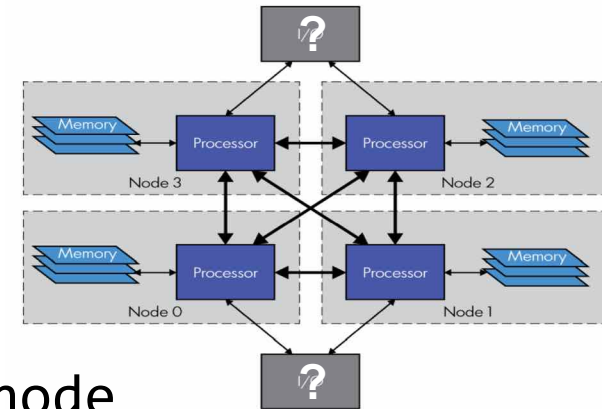
Analyze everything in the data path.

Localize all components to the HBA local nodes.

Reduces latency and improves caching

Discovering Device Locality

- ▶ Devices associated with each volume
 - `multipath -ll`
 - `ls -d /sys/bus/pci/devices/*/host*`
- ▶ Device location
 - `dmidecode -t slot`
 - `lspci -tv`
- ▶ NUMA node of devices
 - `/sys/devices/pci*/*/numa_node`



Pinning Interrupts

- ▶ Pin IRQs to the local node
- ▶ Distribute IRQs between cores in the local node

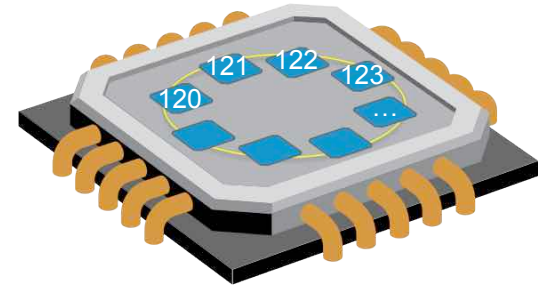
```
# grep [driver] /proc/interrupts
```

```
[num]: ...
```

```
# cat /proc/irq/[num]/node
```

```
[node]
```

```
# echo [CPU mask] > /proc/irq/[num]/smp_affinity
```



~100% improvement

Reduces latency & CPU contention, improves caching

Persistently Pinning Interrupts

- ▶ irqbalance: run-time load balancing
 - Mixed results
 - Better with RHEL 6.4 + kernel update + Sandy Bridge
- ▶ Customized init service to set affinity on boot
 - Best results for a known application

Driver Kernel Thread Affinity

- ▶ Sometimes handled by Linux drivers
- ▶ Verify and adjust

```
# taskset -p -c [cpumask] [pid]
```

```
pid [pid]'s current affinity list: 0-159
```

```
pid [pid]'s new affinity list: 20-29
```

Block Device Tuning

- # echo noop > /sys/block/[device]/queue/scheduler
 - 10% improvement
- # echo 0 > /sys/block/[device]/queue/add_random
 - 10% improvement
- # echo 2 > /sys/block/[device]/queue/rq_affinity
 - 5% improvement
- # echo [0 or 1] > /sys/block/[device]/queue/rotational
 - Mixed results

Reduces latency and CPU utilization

Persistent Block Device Tuning

► udev rules (/etc/udev/rules.d/)

- I/O devices

```
ACTION=="add|change", SUBSYSTEM=="block", ATTR{device/vendor}=="FUSIONIO", ATTR{queue/scheduler}=="noop", ATTR{queue/rq_affinity}="2", ATTR{queue/add_random}=""
```

- devices with I/O slave devices (DM multipath)

```
ACTION=="add|change", KERNEL=="dm-*", PROGRAM="/bin/bash -c 'cat /sys/block/$name/slaves/*/device/vendor | grep FUSIONIO'", ATTR{queue/scheduler}="noop", ATTR{queue/rq_affinity}="2", ATTR{queue/add_random}=""
```

Note: udev rules may only rely on sysfs parameters that are available at the time the device is created.

Power/Performance Tuning

- ▶ Disable c-states in the BIOS
- ▶ Disable c-states in the boot loader (grub)
 - `intel_idle.max_cstate=0`
 - `processor.max_cstate=0`
- ▶ 10% improvement

Keeping processors in active states reduces latency

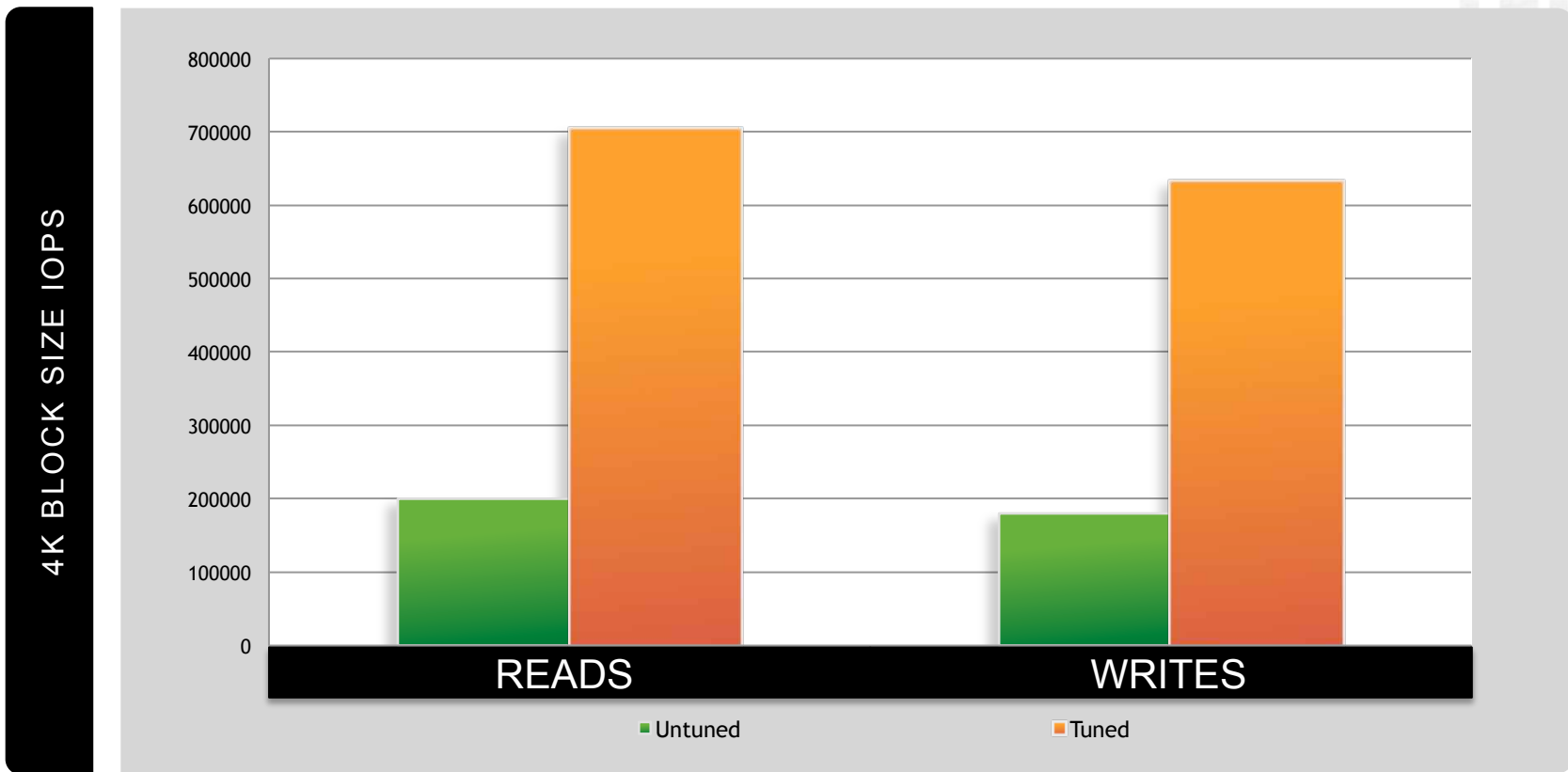
Application Tuning

- ▶ OS-provided tools
 - taskset
 - cgroups
 - numad
- ▶ Application-specific settings
 - Oracle
 - ▶ `_enable_NUMA_support`

Benchmarking Performance

- ▶ Test thread affinity
 - FIO
 - ▶ cpus_allowed
 - ▶ numa_cpu_nodes
 - ▶ numa_mem_policy
 - Oracle Orion and others
 - ▶ taskset -c [cpulist] [test command]
- ▶ Measuring performance
 - iostat
 - Application-specific

8 Socket Total Performance Gains



Tools for NUMA Tuning

| | |
|------------|-----------|
| numactl | cgroups |
| taskset | lstopo |
| dmidecode | sysfs |
| irqbalance | numad |
| top | numatop |
| htop | tuna |
| irqstat | tuned-adm |

top: terminal is not big enough

root@RHEL980:~

root@RHEL980:~

ssh ...

top - 13:31:23 up 1 day, 5:17, 2 users, load average: 3.66, 0.96, 0.42

Tasks: 3897 total, 1 running, 3896 sleeping, 0 stopped, 0 zombie

Cpu(s): 0.2%us, 1.4%sy, 0.0%ni, 95.7%id, 0.5%wa, 0.0%hi, 2.2%si, 0.0%st

Mem: 264521876k total, 118689448k used, 145832428k free, 243928k buffers

Swap: 20971504k total, 0k used, 20971504k free, 2139856k cached

Sorry, terminal is not big enough

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|--------|--------|----|----|-------|------|------|---|-------|------|----------|---------------|
| 124807 | root | 20 | 0 | 824m | 59m | 48m | S | 583.5 | 0.0 | 1:47.06 | fio |
| 89 | root | 20 | 0 | 0 | 0 | 0 | S | 5.1 | 0.0 | 0:46.11 | ksoftirqd/21 |
| 85 | root | 20 | 0 | 0 | 0 | 0 | S | 4.5 | 0.0 | 0:36.00 | ksoftirqd/20 |
| 405 | root | 20 | 0 | 0 | 0 | 0 | S | 4.5 | 0.0 | 0:37.30 | ksoftirqd/100 |
| 421 | root | 20 | 0 | 0 | 0 | 0 | S | 4.2 | 0.0 | 0:12.97 | ksoftirqd/104 |
| 124743 | root | 20 | 0 | 17964 | 4232 | 956 | R | 3.9 | 0.0 | 0:02.39 | top |
| 121 | root | 20 | 0 | 0 | 0 | 0 | S | 2.6 | 0.0 | 0:10.04 | ksoftirqd/29 |
| 441 | root | 20 | 0 | 0 | 0 | 0 | S | 2.6 | 0.0 | 0:10.26 | ksoftirqd/109 |
| 101 | root | 20 | 0 | 0 | 0 | 0 | S | 2.2 | 0.0 | 0:13.09 | ksoftirqd/24 |
| 113 | root | 20 | 0 | 0 | 0 | 0 | S | 1.3 | 0.0 | 0:16.79 | ksoftirqd/27 |
| 10953 | oracle | 20 | 0 | 1388m | 135m | 114m | S | 1.3 | 0.1 | 24:00.31 | oracle |
| 409 | root | 20 | 0 | 0 | 0 | 0 | S | 1.0 | 0.0 | 0:12.80 | ksoftirqd/101 |
| 413 | root | 20 | 0 | 0 | 0 | 0 | S | 1.0 | 0.0 | 0:24.10 | ksoftirqd/102 |
| 425 | root | 20 | 0 | 0 | 0 | 0 | S | 1.0 | 0.0 | 0:05.92 | ksoftirqd/105 |
| 437 | root | 20 | 0 | 0 | 0 | 0 | S | 1.0 | 0.0 | 0:08.67 | ksoftirqd/108 |
| 10945 | oracle | -2 | 0 | 1365m | 14m | 12m | S | 1.0 | 0.0 | 6:36.87 | oracle |
| 97 | root | 20 | 0 | 0 | 0 | 0 | S | 0.6 | 0.0 | 0:31.75 | ksoftirqd/23 |
| 117 | root | 20 | 0 | 0 | 0 | 0 | S | 0.6 | 0.0 | 0:08.93 | ksoftirad/28 |

(top in RHEL 6.4)

mpstat with 160 cores

```

root@RHEL980:~ root@RHEL980:~ ssh ...
[root@RHEL980 ~]# mpstat -I ALL
Linux 2.6.32-358.el6.x86_64 (RHEL980.int.fusionio.com) 04/13/2013 _x86_64_ (16

01:57:00 PM CPU intr/s
01:57:00 PM all 14284.04

01:57:00 PM CPU 0/s 1/s 3/s 4/s 8/s 9/s 12
103/s 104/s 105/s 106/s 107/s 108/s 109/s 110/s 12
120/s 121/s 122/s 123/s 124/s 125/s 126/s 127/s 12
s 138/s 139/s 140/s 141/s NMI/s LOC/s SPU/s PMI/s
MIS/s
01:57:00 PM 0 0.01 0.00 0.00 0.00 0.00 0.00 0.
0.01 0.08 0.00 0.00 0.01 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0 0.00 0.00 0.00 0.00 0.01 19.33 0.00 0.01
0.00

01:57:00 PM 1 0.00 0.00 0.00 0.00 0.00 0.00 0.
0.00 0.00 0.00 0.00 0.00 131.68 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0 0.00 0.00 0.00 0.00 0.02 15.82 0.00 0.02
0.00

01:57:00 PM 2 0.00 0.00 0.00 0.00 0.00 0.00 0.
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0 0.00 0.00 0.00 0.00 0.01 4.17 0.00 0.01

```

ores

top: added NUMA support

```

root@RHEL980:~                               root@RHEL980:~  ssh ...
lt-top - 13:32:31 up 1 day, 5:18, 2 users, load average: 10.55, 3.58, 1.36
Tasks: 3898 total, 1 running, 3897 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 1.3 sy, 0.0 ni, 95.7 id, 0.6 wa, 0.0 hi, 2.2 si, 0.0 st
%Node0 : 0.6 us, 4.4 sy, 0.0 ni, 82.7 id, 3.6 wa, 0.0 hi, 8.7 si, 0.0 st
%Node1 : 0.1 us, 0.1 sy, 0.0 ni, 99.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Node2 : 0.8 us, 5.9 sy, 0.0 ni, 82.2 id, 1.2 wa, 0.0 hi, 10.0 si, 0.0 st
%Node3 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Node4 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Node5 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Node6 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Node7 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 26452187+total, 11869251+used, 14582937+free, 243976 buffers
KiB Swap: 20971504 total, 0 used, 20971504 free, 2140132 cached

```

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|---------------|-------------|-----------|----------|---------------|-------------|-------------|----------|--------------|--------------|----------------|---------------|
| 124807 | root | 20 | 0 | 843984 | 60460 | 50076 | S | 578.6 | 0.023 | 8:37.67 | fio |
| 89 | root | 20 | 0 | 0 | 0 | 0 | S | 6.097 | 0.000 | 0:48.85 | ksoftirqd/21 |
| 93 | root | 20 | 0 | 0 | 0 | 0 | S | 5.776 | 0.000 | 0:25.55 | ksoftirqd/22 |
| 413 | root | 20 | 0 | 0 | 0 | 0 | S | 4.172 | 0.000 | 0:25.72 | ksoftirqd/102 |
| 97 | root | 20 | 0 | 0 | 0 | 0 | S | 3.851 | 0.000 | 0:33.76 | ksoftirqd/23 |
| 124881 | root | 20 | 0 | 117652 | 4820 | 1136 | R | 3.851 | 0.002 | 0:01.37 | lt-top |
| 105 | root | 20 | 0 | 0 | 0 | 0 | S | 2.888 | 0.000 | 0:18.82 | ksoftirqd/25 |
| 405 | root | 20 | 0 | 0 | 0 | 0 | S | 2.567 | 0.000 | 0:39.12 | ksoftirqd/100 |
| 425 | root | 20 | 0 | 0 | 0 | 0 | S | 2.567 | 0.000 | 0:06.28 | ksoftirqd/105 |
| 85 | root | 20 | 0 | 0 | 0 | 0 | S | 2.246 | 0.000 | 0:37.75 | ksoftirqd/20 |

<https://gitorious.org/procps/procps>

irqstat: IRQ viewer for NUMA

| root@RHEL980:~/numatools | | | | root@RHEL980:~ | | | | ssh ... | | |
|--------------------------|--------|-------|-------|----------------|-------|-------|-------|---------|-------|----------------------|
| IRQs / 5 second(s) | | | | | | | | | | |
| IRQ# | TOTAL | NODE0 | NODE1 | NODE2 | NODE3 | NODE4 | NODE5 | NODE6 | NODE7 | NAME |
| 125 | 361145 | 0 | 0 | 361145 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge qla2xxx |
| 123 | 356357 | 0 | 0 | 356357 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge qla2xxx |
| 109 | 46118 | 46118 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge lpfc:fp |
| 111 | 46085 | 46085 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge lpfc:fp |
| 115 | 44369 | 0 | 0 | 44369 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge lpfc:fp |
| 117 | 43951 | 0 | 0 | 43951 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge lpfc:fp |
| 107 | 37149 | 37149 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge lpfc:fp |
| 113 | 36305 | 36305 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge lpfc:fp |
| 126 | 11 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge eth0[0] |
| 129 | 5 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCI-MSI-edge eth0[3] |

<https://github.com/lanceshelton/irqstat>

Must end users be NUMA-aware?

- ▶ Unfortunately, yes.
 - Users must be aware of PCIe device slot placement
 - Optimal NUMA tuning is not yet performed by the OS
 - Persistent tuning is a non-trivial task
 - Performance challenges are changing faster than tools

How can this be improved?

- NUMA architectures must be detected properly and tuned by default
 - Phase out Nehalem or add SLIT support for multiple local nodes
- Linux distributions need to provide optimal tuning across applications and devices at the OS level
- Improve existing tools

What's next?

- ▶ More cores per socket
 - 15 core CPUs by the end of next year?
- ▶ Removal of existing bottlenecks
 - Multi-queue block layer: <http://kernel.dk/blk-mq.pdf>
- ▶ Improved tools
 - numatop: <https://01.org/numatop>
 - top: <https://gitorious.org/procps/procps>
 - irqstat: <https://github.com/lanceshelton/irqstat>

References

- ▶ HP DL980 Architecture

- ▶ <http://h20195.www2.hp.com/V2/GetPDF.aspx/4AA3-0643ENW.pdf>

- ▶ HP NUMA Support

- ▶ <http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c03261871/c03261871.pdf>

- ▶ Performance profiling methods

- ▶ <http://dtrace.org/blogs/brendan/2012/03/07/the-use-method-linux-performance-checklist/>

THANK YOU



LShelton@fusionio.com