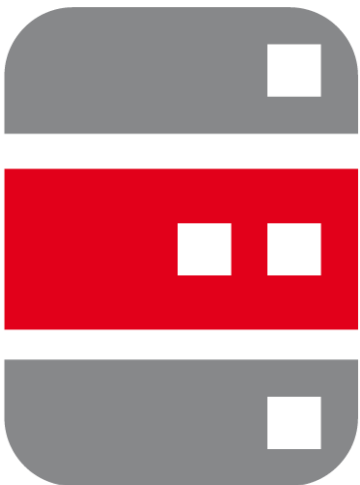


performing databases



Your reliability. Our concern.

The (Un)loved Child of Generation Cloud - Oracle on VMware

Martin Klier 

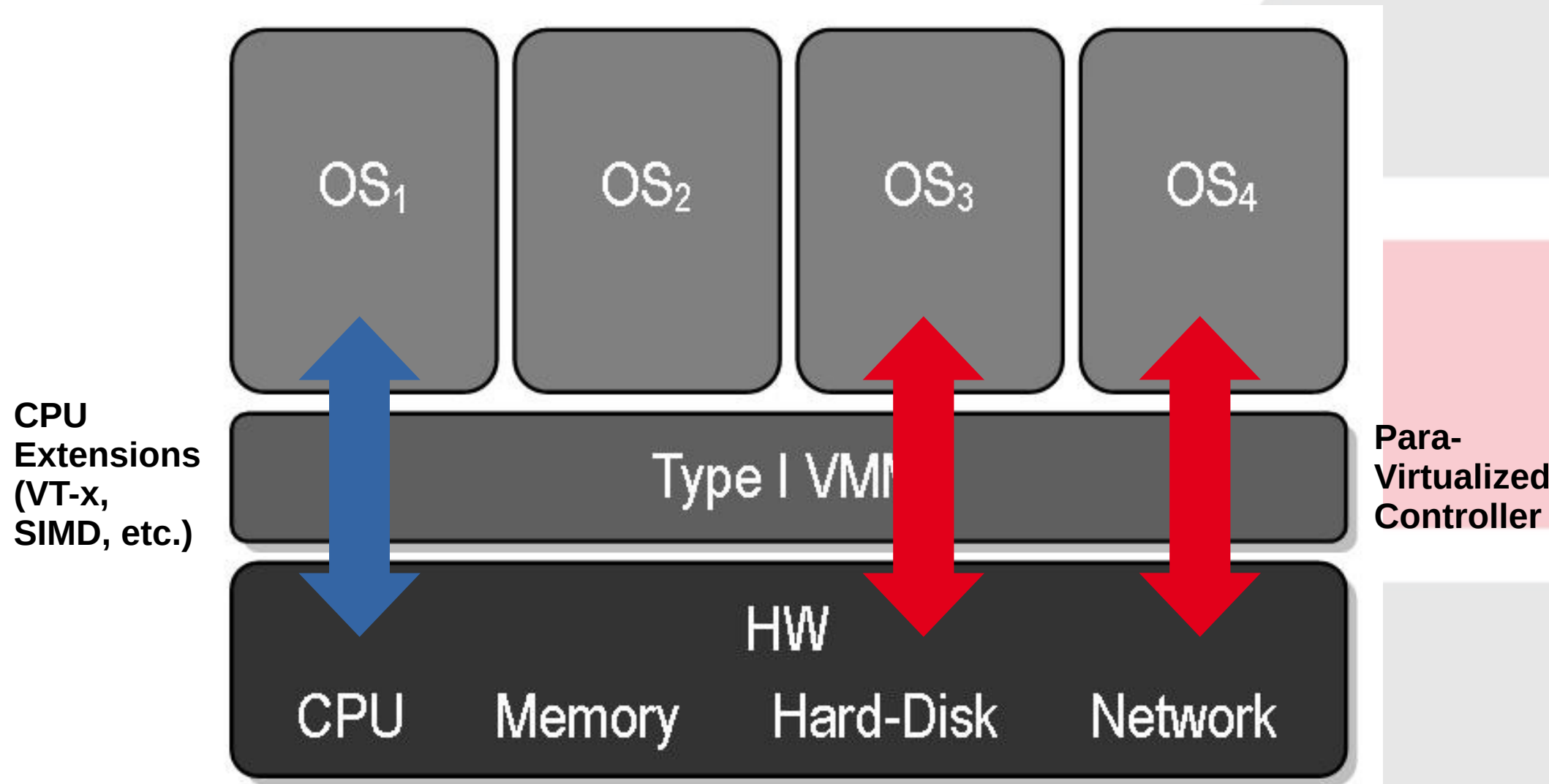
Performing Databases GmbH
Mitterteich / Germany



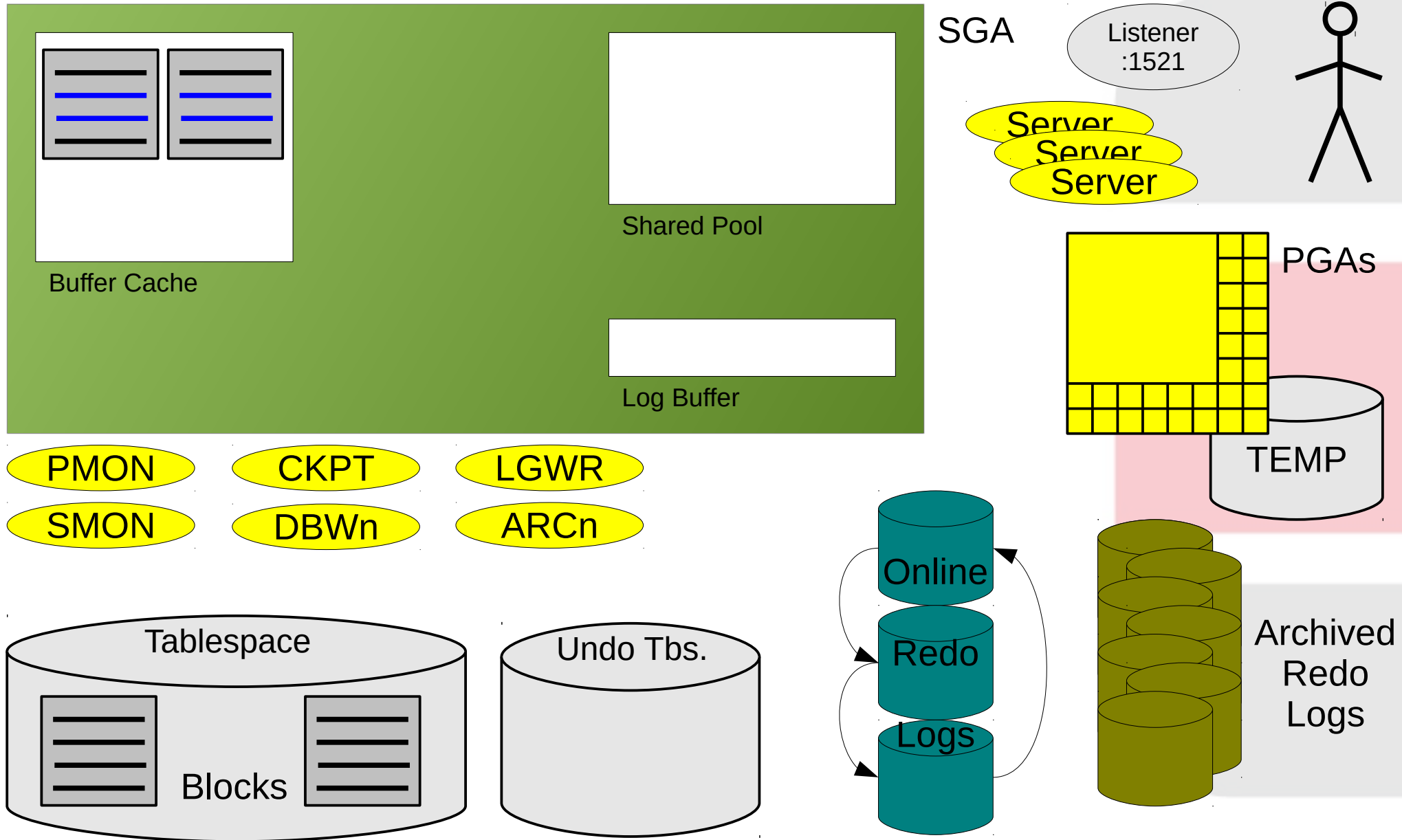
Basics

VMware vSphere

a Type-I-Hypervisor



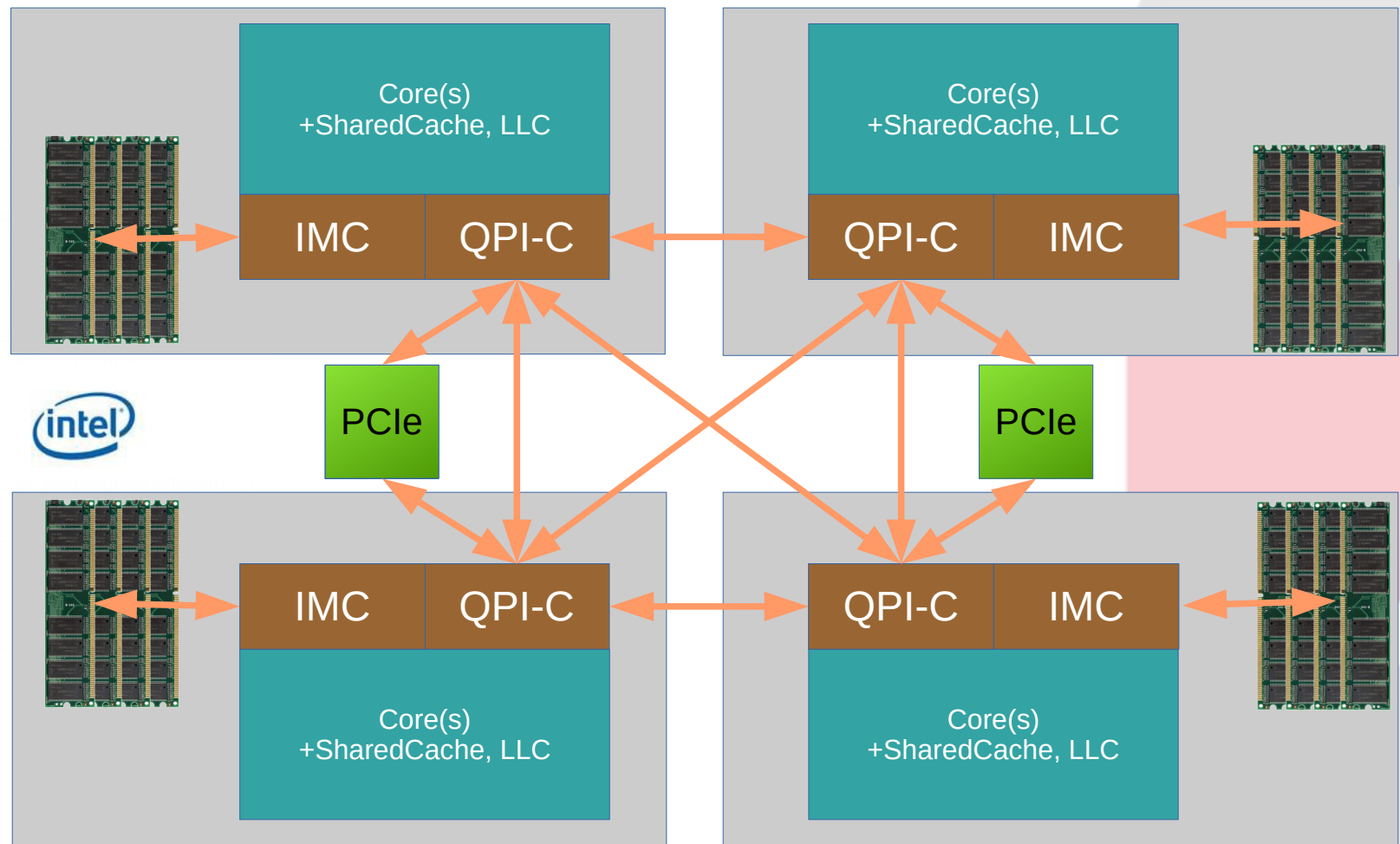
Oracle Architecture (simplified)



NUMA Architecture

NUMA

Non Unified Memory Access



Oracle & NUMA

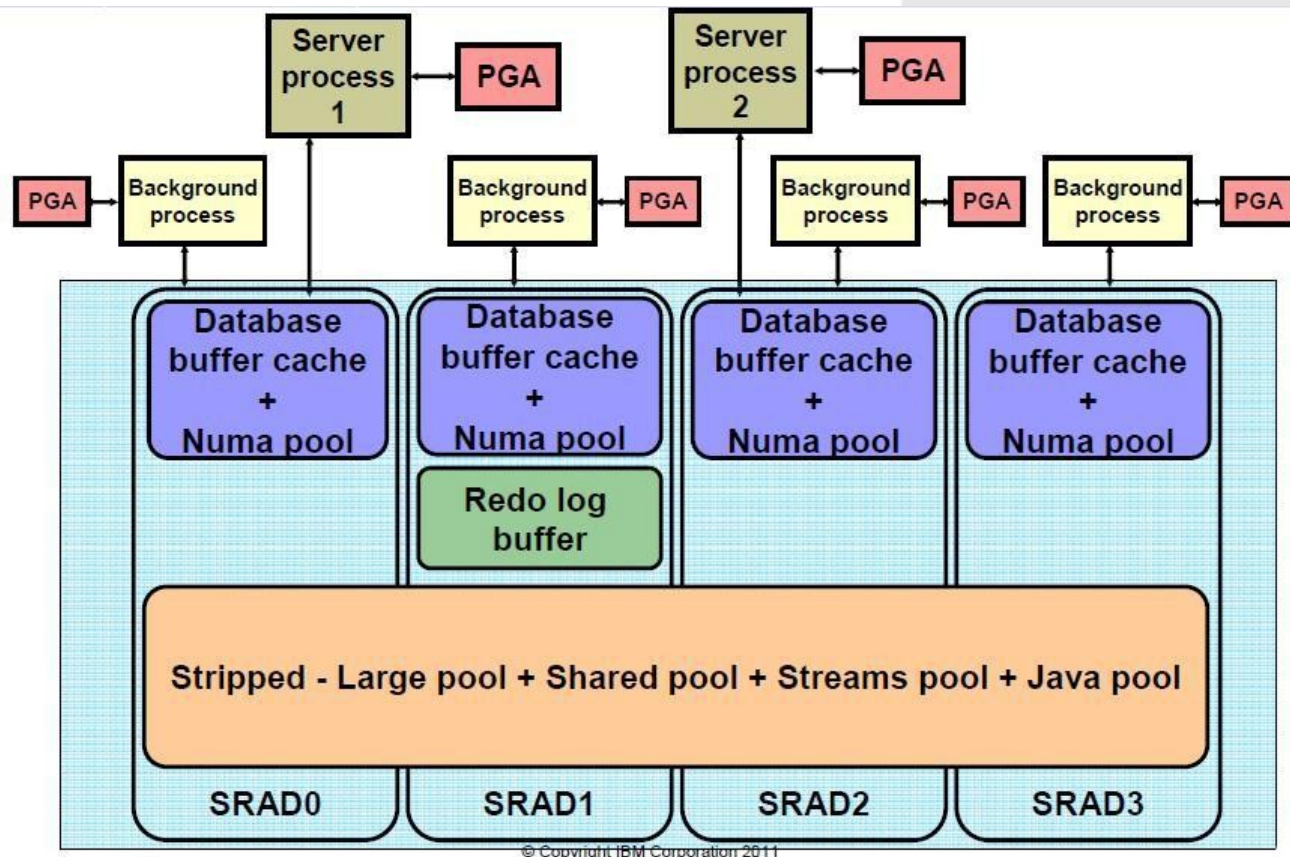
Oracle & NUMA

`_enable_NUMA_support = TRUE`
MOS Doc ID 864633.1

- Multiple Buffer Caches
- Striped pools

=> cross context :((

=> pool access :(



Oracle & NUMA

```
[root@ora05 ~]# ipcs -ma
```

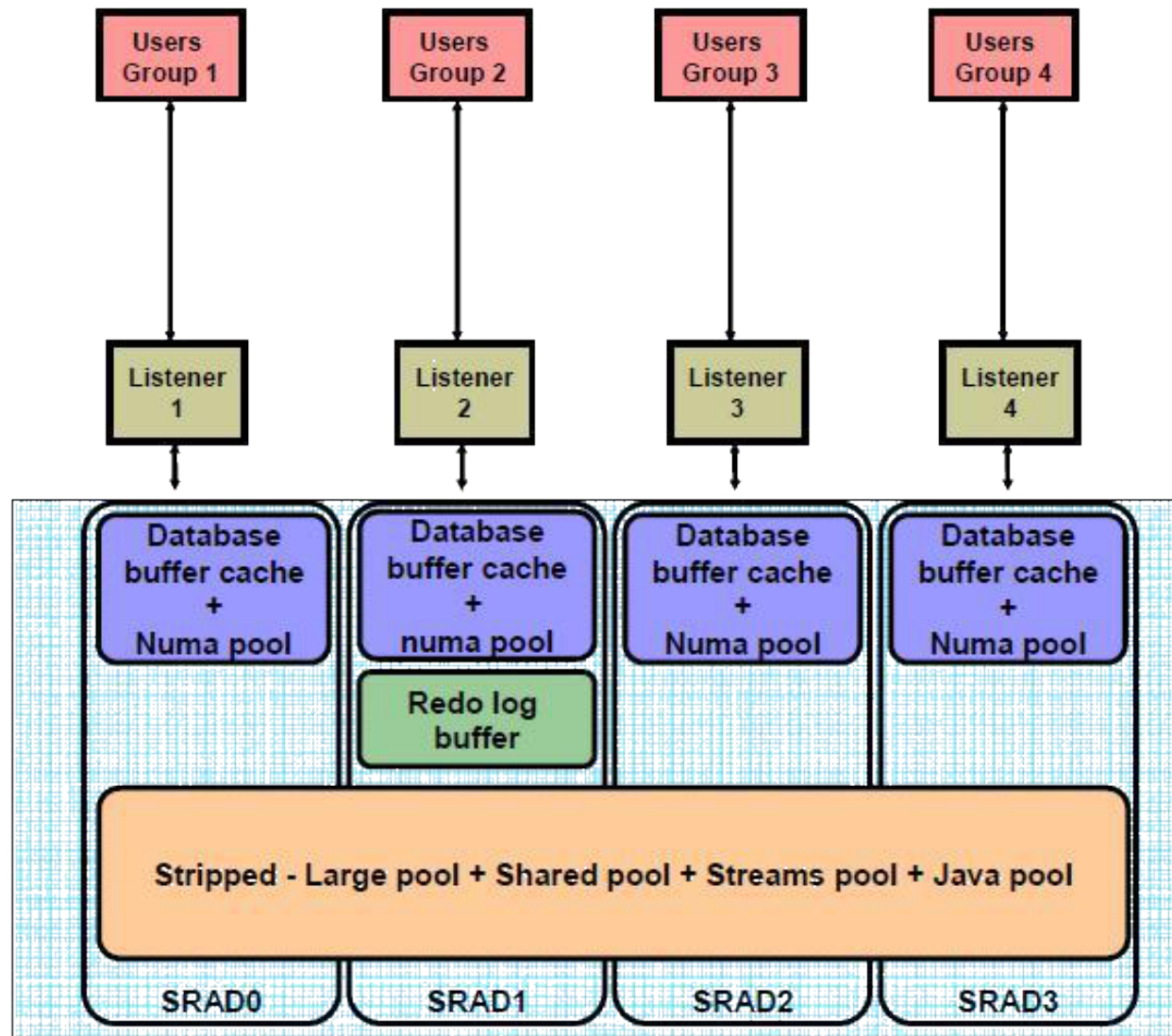
```
----- Shared Memory Segments -----
key          shmid      owner      perms      bytes      nattch     status
0x740301e9   2457600    root       600         4           0
0x00000000   2752513    root       644         80          2
0x00000000   2785282    root       644        16384        2
0x00000000   2818051    root       644         280         2
0x00000000   2883588    oracle     640         4096         0
0x00000000   2916357    oracle     640         4096         0
0xed304ac0   2949126    oracle     640         4096         0
0x00000000   3735559    oracle     640      138412032    464
0x00000000   3768328    oracle     640       8388608      464
0x00000000   3801097    oracle     640      13555990528  464
0x00000000   3833866    oracle     640      13623099392  464
0x00000000   3866635    oracle     640      2684354560   464
0xc93391ac   3899404    oracle     640       2097152      464
```

One buffer cache
for each node



13GB+13GB=26 GB

Oracle & NUMA



© Copyright IBM Corporation 2011

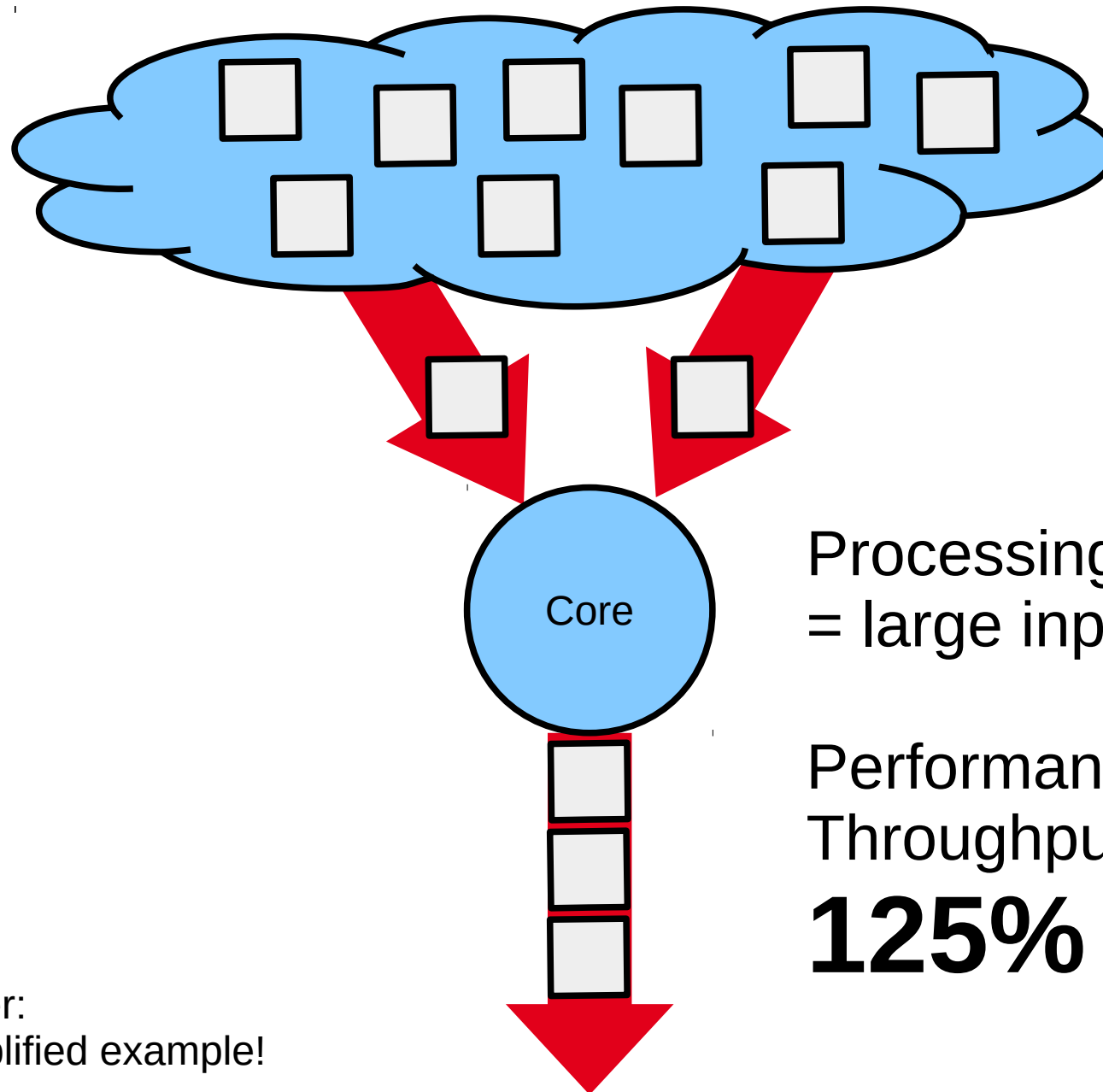
Oracle NUMA aware

Suggestions

- Useful in big environments only (think: DB consolidation)
- Test thoroughly and quantify use vs. effort (think: bugs)
- Use only if necessary to achieve your goals!

Hyperthreading

HyperThreading (HTT)

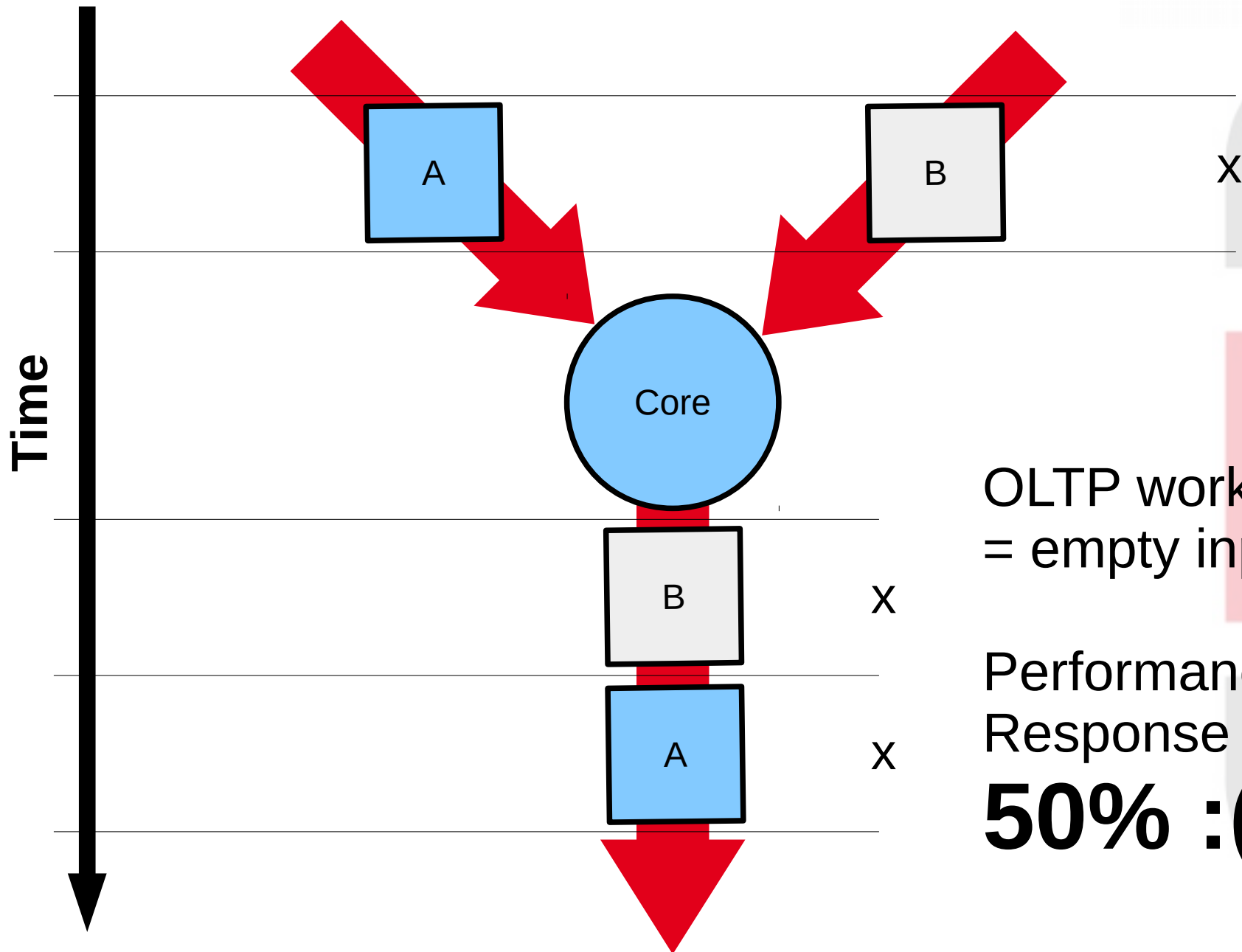


Processing big workloads
= large input queue

Performance =
Throughput
125% :)

Disclaimer:
Very simplified example!

OLTP & HTT

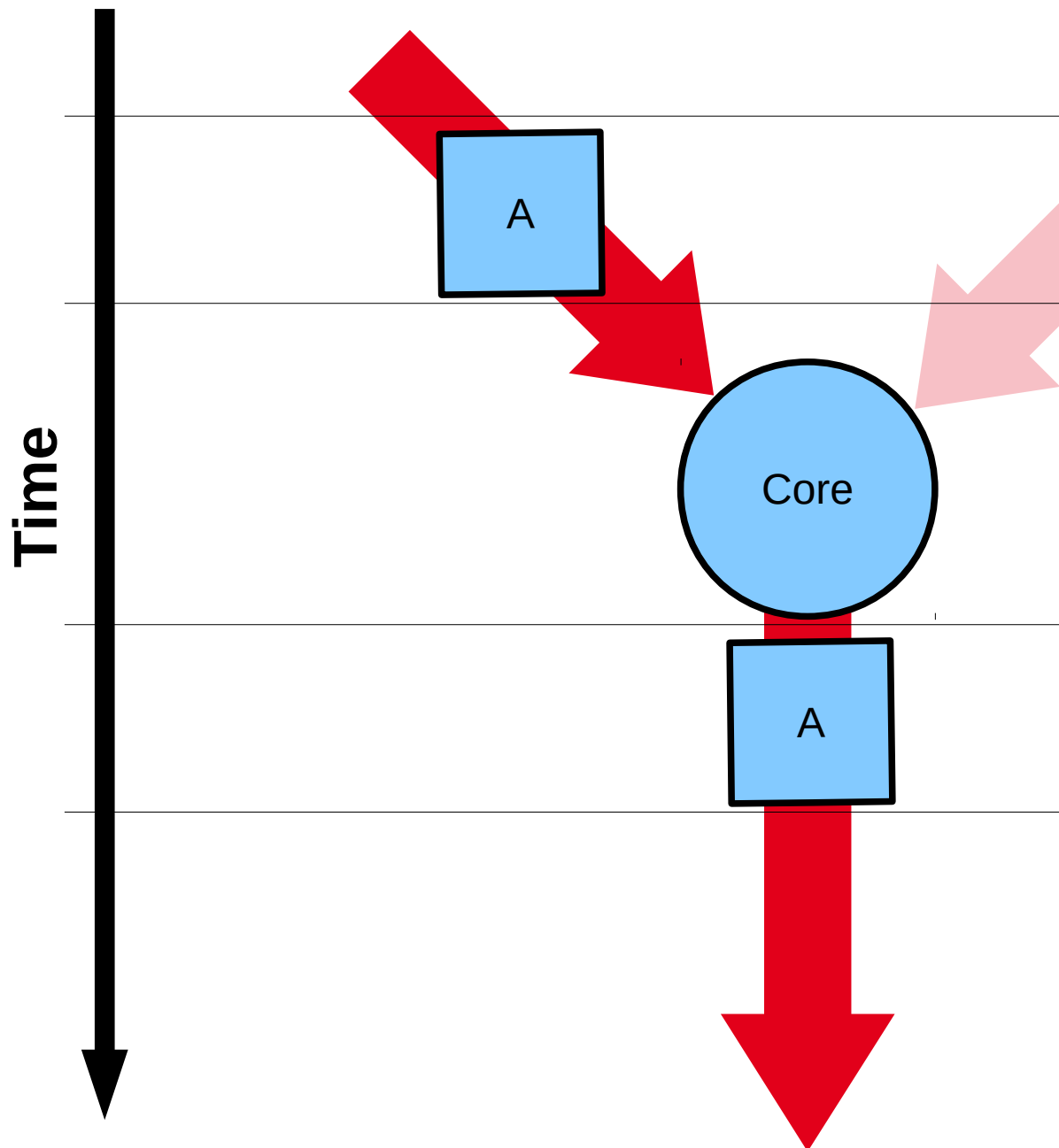


OLTP workloads
= empty input queue

Performance =
Response Time

50% :(

OLTP & HTT



**Disable
HyperThreading
for OLTP systems!**

- in hardware BIOS
- on VMware level
(„Latency Sensitivity“ high)

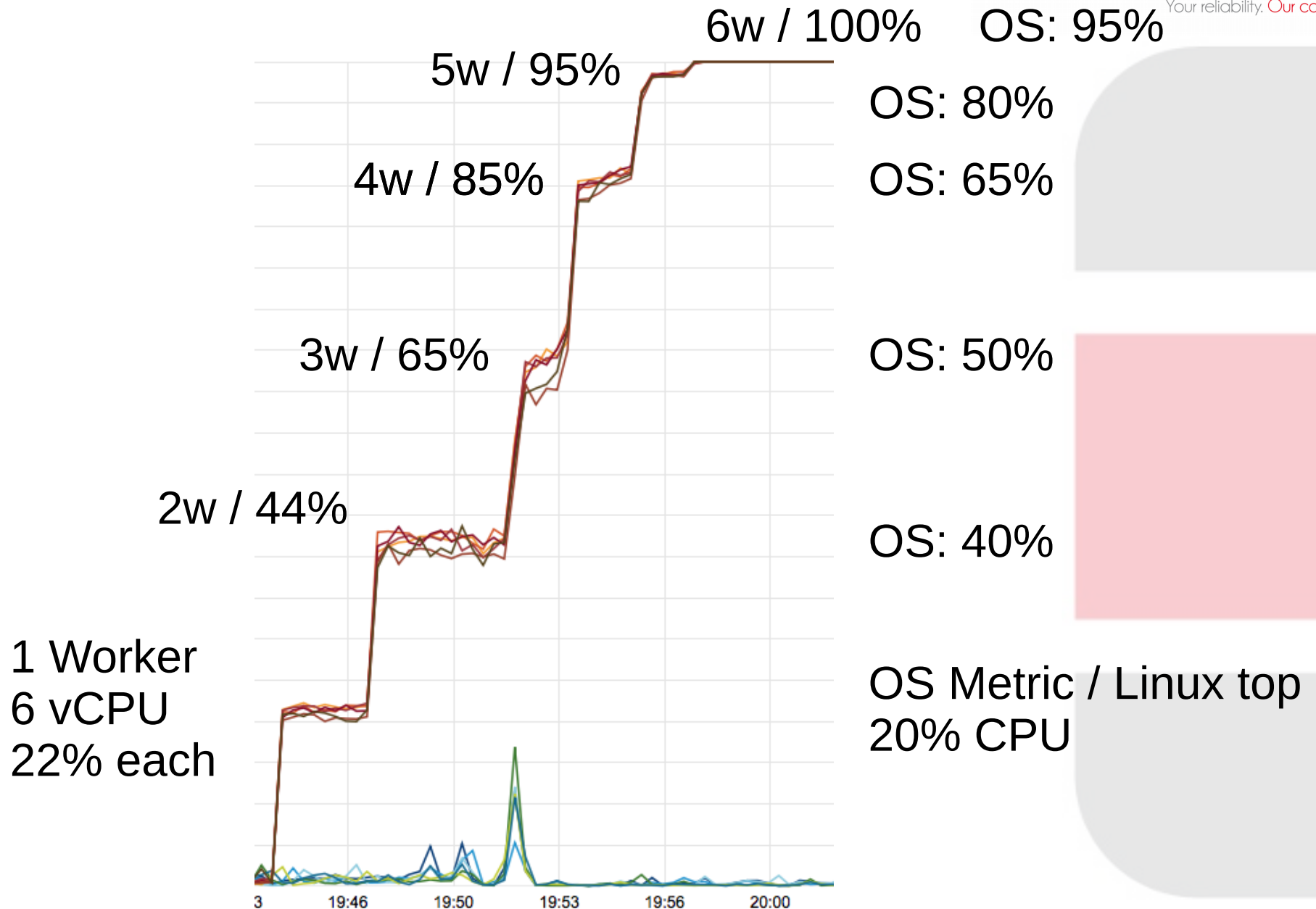
VMware vSphere (ESXi) CPU Usage

VMware CPU Usage

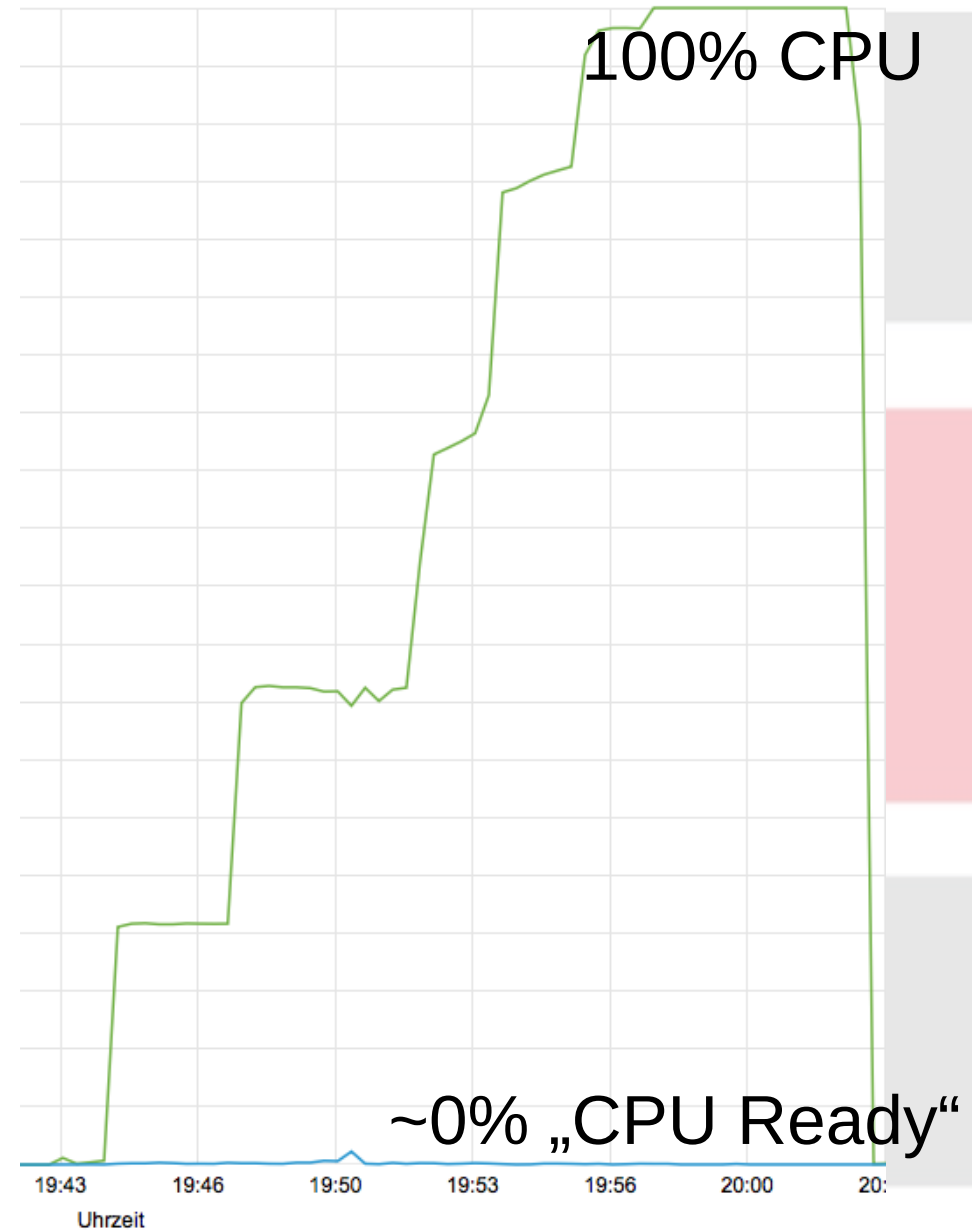
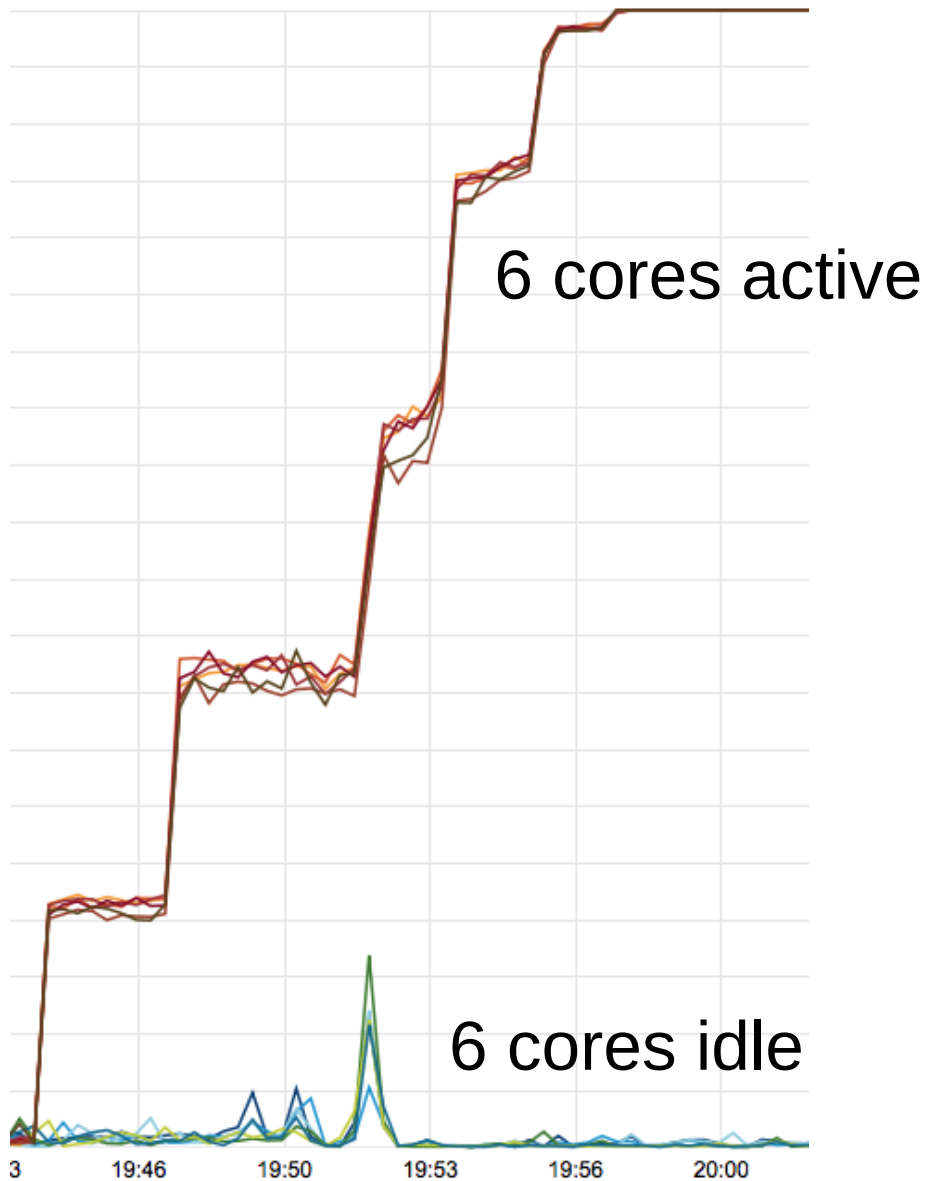
My Test Setup

- HP ProLiant DL380 G7
- 2x 64GB RAM
- 2 CPUs, **2x 6 cores = 2 NUMA nodes**
Intel(R) Xeon(R) CPU X5670 @ 2.93GHz boost 3.33GHz
„Westmere EP“ 32nm technology
- HyperThreading disabled (BIOS)
- VMware vSphere ESXi 6.0 U2
- 2 VMs / Oracle Linux 7.4 w/ UEK Kernel

VMware CPU Usage

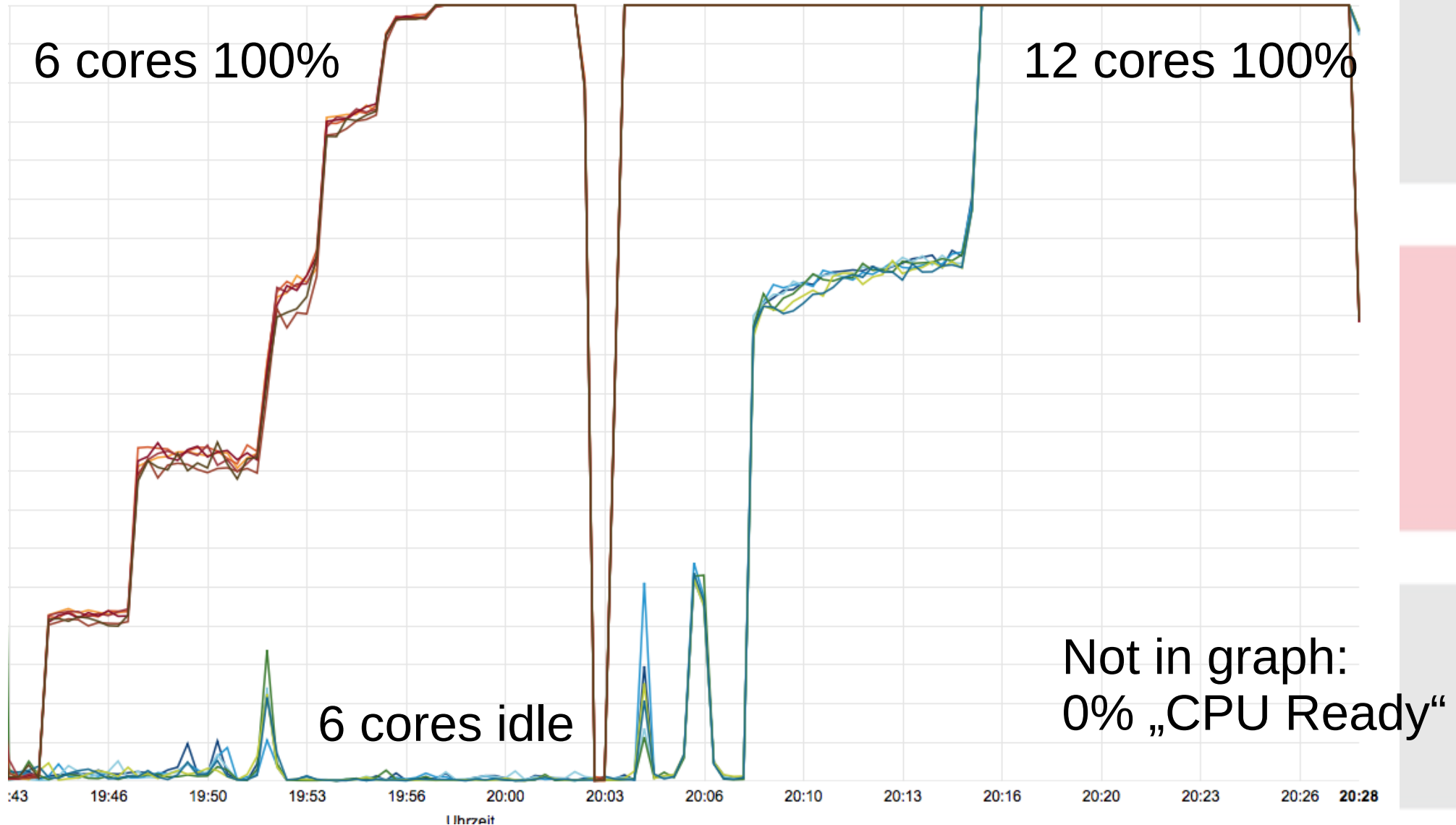


VMware CPU Usage



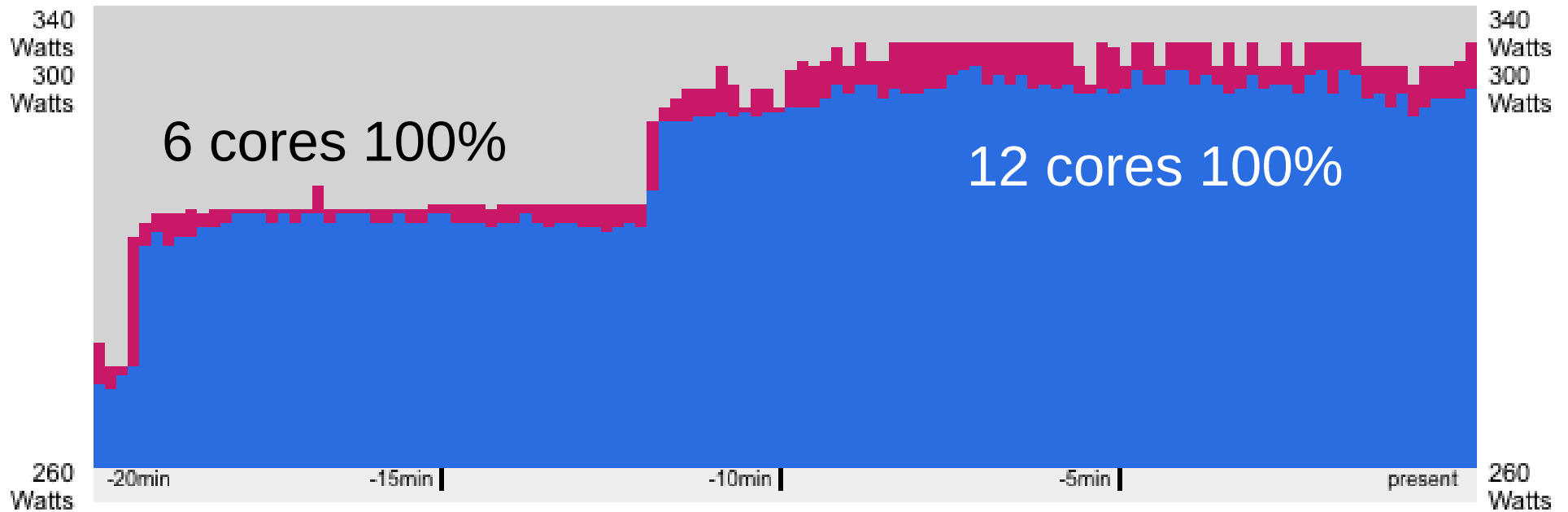
CPU Ready
=
Wait for free CPU time

VMware CPU Usage



Green IT :)

Power consumption over the past 20 minutes at ten-second intervals.



VMware CPU Usage

8 vCores

13 workers to fill up to 100% in top

```
top - 15:29:01 up 24 min, 2 users, load average: 1.24, 0.54, 0.27
Tasks: 160 total, 4 running, 156 sleeping, 0 stopped, 0 zombie
%Cpu0  : 30.0/70.0 100[
%Cpu1  : 30.3/69.7 100[
%Cpu2  : 34.7/64.4 99[
%Cpu3  : 31.0/69.0 100[
%Cpu4  : 32.0/68.0 100[
%Cpu5  : 30.7/69.3 100[
%Cpu6  : 30.4/68.6 99[
%Cpu7  : 30.0/70.0 100[
KiB Mem : 3781400 total, 3458524 free, 137844 used, 185032 buff/cache
KiB Swap: 1952764 total, 1952764 free, 0 used. 3564828 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
8659	root	20	0	113524	2508	1872	S	11.9	0.1	0:03.09	/bin/bash ./loader.sh
8853	root	20	0	113524	2508	1872	S	10.9	0.1	0:03.06	/bin/bash ./loader.sh
9148	root	20	0	113524	2500	1872	S	10.9	0.1	0:02.97	/bin/bash ./loader.sh
10140	root	20	0	113524	2496	1872	S	10.9	0.1	0:02.93	/bin/bash ./loader.sh
10896	root	20	0	113524	2492	1872	S	10.9	0.1	0:02.92	/bin/bash ./loader.sh
13837	root	20	0	113524	2488	1872	S	10.9	0.1	0:02.83	/bin/bash ./loader.sh
18933	root	20	0	113524	2476	1872	S	10.9	0.1	0:02.72	/bin/bash ./loader.sh
9579	root	20	0	113524	2500	1872	S	9.9	0.1	0:02.97	/bin/bash ./loader.sh
11951	root	20	0	113524	2492	1872	S	9.9	0.1	0:02.86	/bin/bash ./loader.sh
12789	root	20	0	113524	2488	1872	S	9.9	0.1	0:02.82	/bin/bash ./loader.sh
14834	root	20	0	113524	2484	1872	R	9.9	0.1	0:02.79	/bin/bash ./loader.sh
15731	root	20	0	113524	2484	1872	S	9.9	0.1	0:02.76	/bin/bash ./loader.sh
16929	root	20	0	113524	2480	1872	S	9.9	0.1	0:02.77	/bin/bash ./loader.sh
14	root	rt	0	0	0	0	S	1.0	0.0	0:05.13	[migration/1]
18	root	20	0	0	0	0	S	1.0	0.0	0:04.86	[rcuos/1]

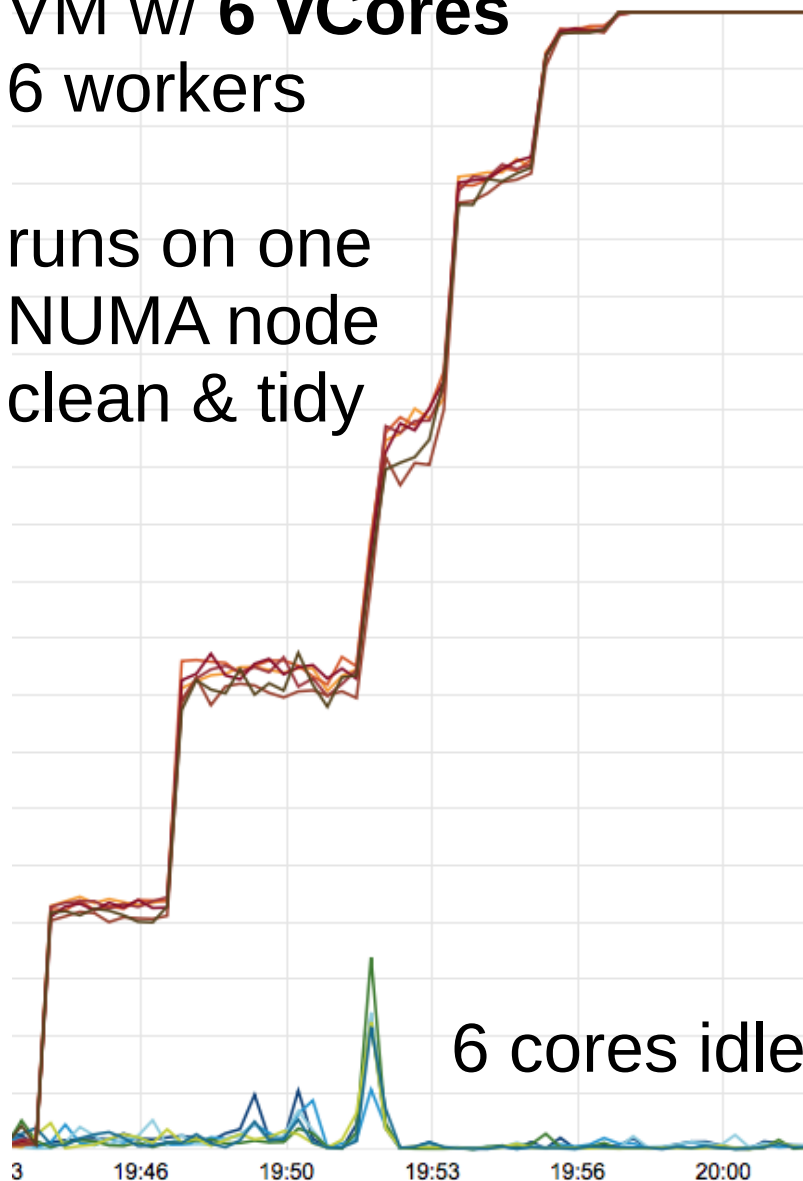
Each worker:

while true
(bc < 5.5*7.7 etc.
)

VMware CPU Usage

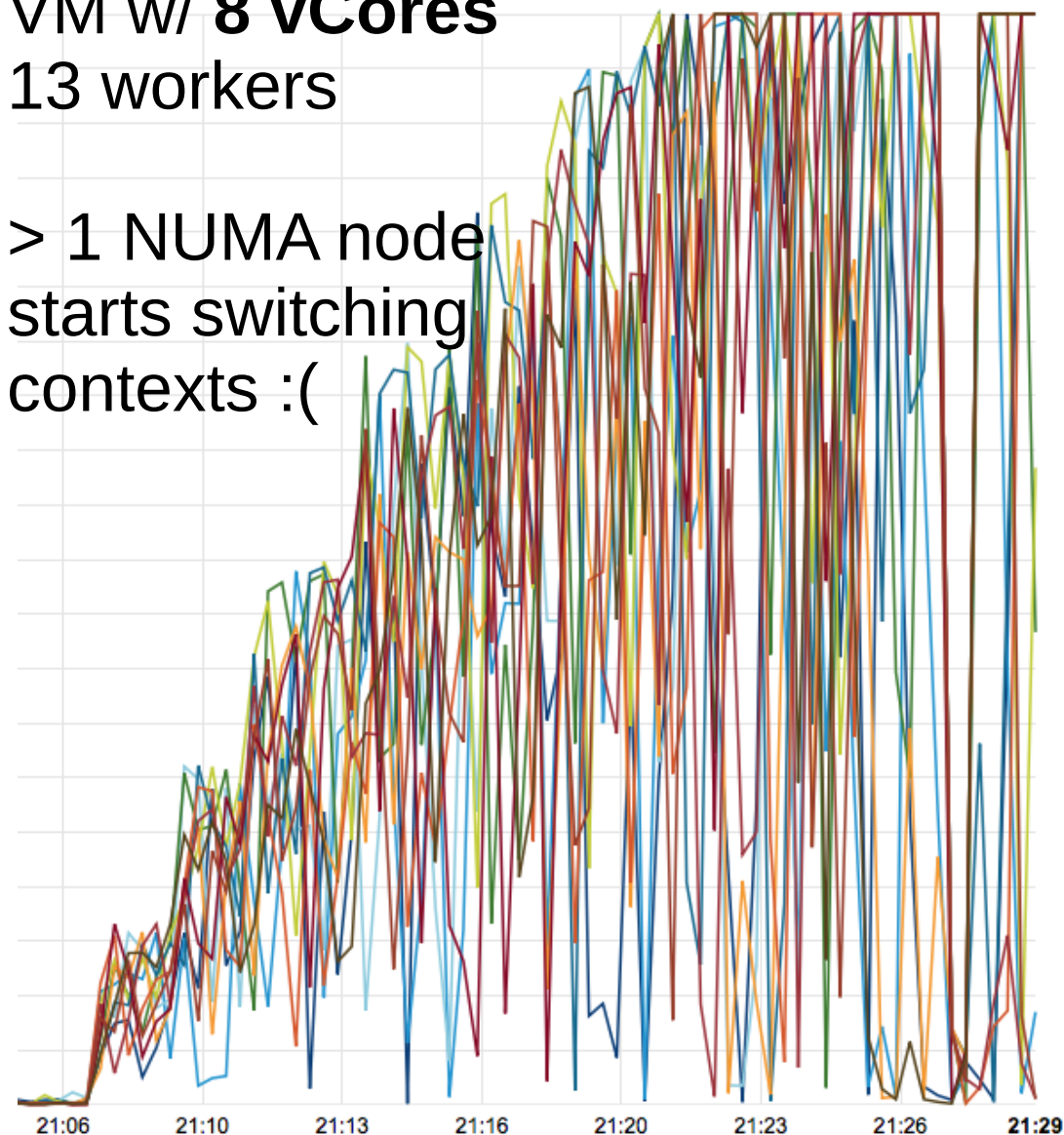
VM w/ 6 vCores
6 workers

runs on one
NUMA node
clean & tidy



VM w/ 8 vCores
13 workers

> 1 NUMA node
starts switching
contexts :(



Oracle Workload on ESXi

Oracle Stress in VM

```
top - 16:41:25 up 1:14, 5 users, load average: 2.45, 1.21, 1.39
Tasks: 244 total, 9 running, 235 sleeping, 0 stopped, 0 zombie
%Cpu0 : 94.1/5.9 100[
%Cpu1 : 95.0/5.0 100[
%Cpu2 : 94.1/5.9 100[
%Cpu3 : 93.1/6.9 100[
%Cpu4 : 93.1/6.9 100[
%Cpu5 : 93.1/6.9 100[
KiB Mem : 64943912 total, 27610432 free, 20261048 used, 17072432 buff/cache
KiB Swap: 1953788 total, 1953788 free, 0 used. 44306400 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
14076	oracle	20	0	19.233g	68996	65624	R	100.0	0.1	1:11.69	oraclePDB12201 (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=beq)))
14074	oracle	20	0	19.233g	68532	65168	R	100.0	0.1	1:11.87	oraclePDB12201 (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=beq)))
14075	oracle	20	0	19.233g	69960	66604	R	100.0	0.1	1:11.77	oraclePDB12201 (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=beq)))
14077	oracle	20	0	19.233g	69948	66448	R	100.0	0.1	1:11.99	oraclePDB12201 (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=beq)))
14078	oracle	20	0	19.233g	69924	66428	R	100.0	0.1	1:11.50	oraclePDB12201 (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=beq)))
14079	oracle	20	0	19.233g	69544	66184	R	100.0	0.1	1:11.66	oraclePDB12201 (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=beq)))
2598	oracle	-2	0	19.230g	56432	53760	S	2.0	0.1	0:27.34	ora_vktm_PDB12201

loop

(select 1 row;
update this row;
commit)

6 workers

each on a different set of 10k rows

runs ~20min until Free Buffer Waits

Anti-Load in VM

```
top - 15:36:29 up 6:39, 2 users, load average: 0.24, 0.08, 0.06
Tasks: 198 total, 6 running, 191 sleeping, 0 stopped, 1 zombie
```

```
%Cpu0 : 12.2/36.6 49[
%Cpu1 : 13.8/36.2 50[
%Cpu2 : 12.2/44.6 57[
%Cpu3 : 13.3/34.4 48[
%Cpu4 : 11.1/37.0 48[
%Cpu5 : 10.5/43.4 54[
%Cpu6 : 14.5/47.0 61[
%Cpu7 : 13.3/53.3 67[
%Cpu8 : 17.3/52.0 69[
%Cpu9 : 16.0/51.9 68[
%Cpu10 : 13.3/55.4 69[
%Cpu11 : 15.5/48.8 64[
```

Not nearly full power
(CPU load 55%)

Owner will feel
no guilt

```
KiB Mem : 3765012 total, 3154872 free, 163420 used, 446720 buff/cache
KiB Swap: 1952764 total, 1952764 free, 0 used. 3505280 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
6333	root	20	0	113524	2688	2108	S	22.8	0.1	0:05.37	/bin/bash ./loader.sh
6890	root	20	0	113524	2684	2108	S	21.8	0.1	0:05.03	/bin/bash ./loader.sh
23999	root	20	0	113396	2620	2108	S	20.8	0.1	0:04.92	/bin/bash ./loader.sh
27403	root	20	0	113396	2612	2108	S	19.8	0.1	0:04.99	/bin/bash ./loader.sh
5624	root	20	0	113524	2696	2108	S	18.8	0.1	0:05.14	/bin/bash ./loader.sh
8387	root	20	0	113524	2676	2108	S	18.8	0.1	0:04.93	/bin/bash ./loader.sh
34049	root	20	0	113396	2596	2108	R	17.8	0.1	0:04.44	/bin/bash ./loader.sh
15880	root	20	0	113396	2648	2108	S	16.8	0.1	0:05.02	/bin/bash ./loader.sh
7	root	20	0	0	0	0	R	15.8	0.0	0:05.75	[rcu_sched]
21966	root	20	0	113396	2624	2108	R	15.8	0.1	0:04.77	/bin/bash ./loader.sh
5923	root	20	0	113524	2692	2108	S	14.9	0.1	0:04.81	/bin/bash ./loader.sh
7542	root	20	0	113524	2680	2108	S	13.9	0.1	0:04.56	/bin/bash ./loader.sh
30760	root	20	0	113396	2604	2108	S	13.9	0.1	0:04.49	/bin/bash ./loader.sh
18607	root	20	0	113396	2636	2108	S	12.9	0.1	0:05.30	/bin/bash ./loader.sh
56	root	rt	0	0	0	0	S	9.9	0.0	0:01.53	[migration/7]
5448	root	20	0	113524	2696	2108	S	9.9	0.1	0:05.13	/bin/bash ./loader.sh

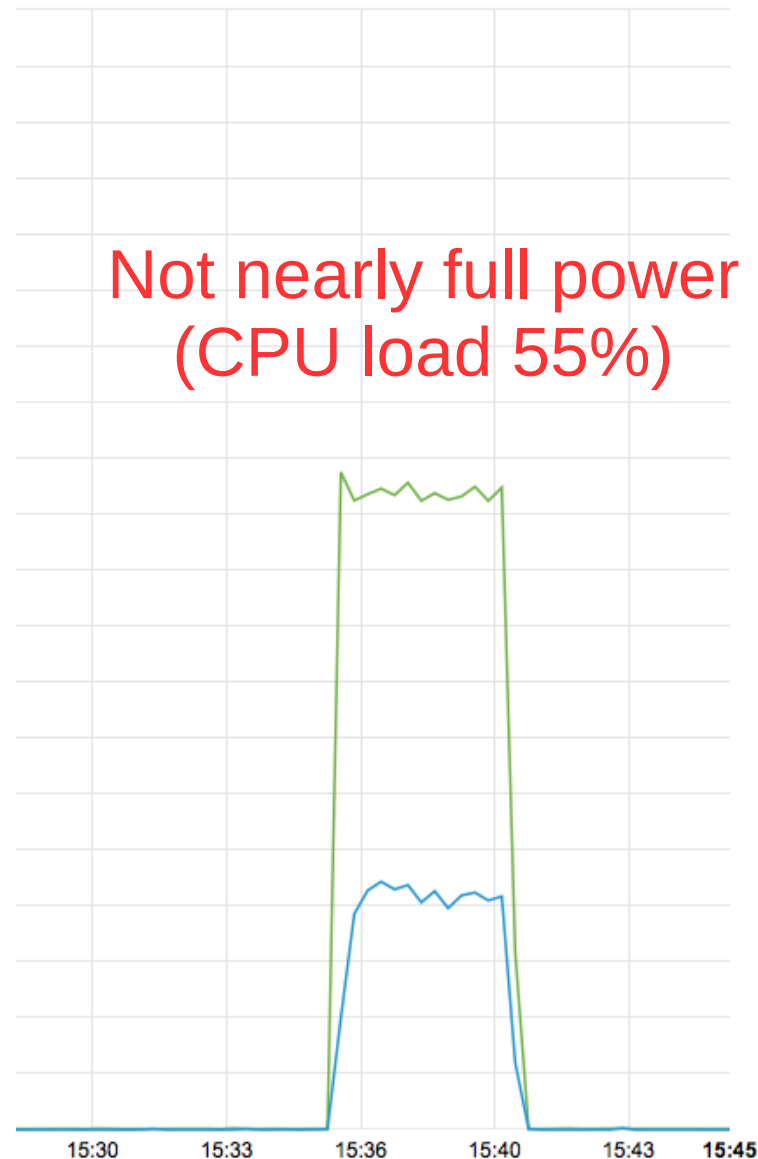
14 workers (bc)
55% avg. load
over 12 vCores

we know:

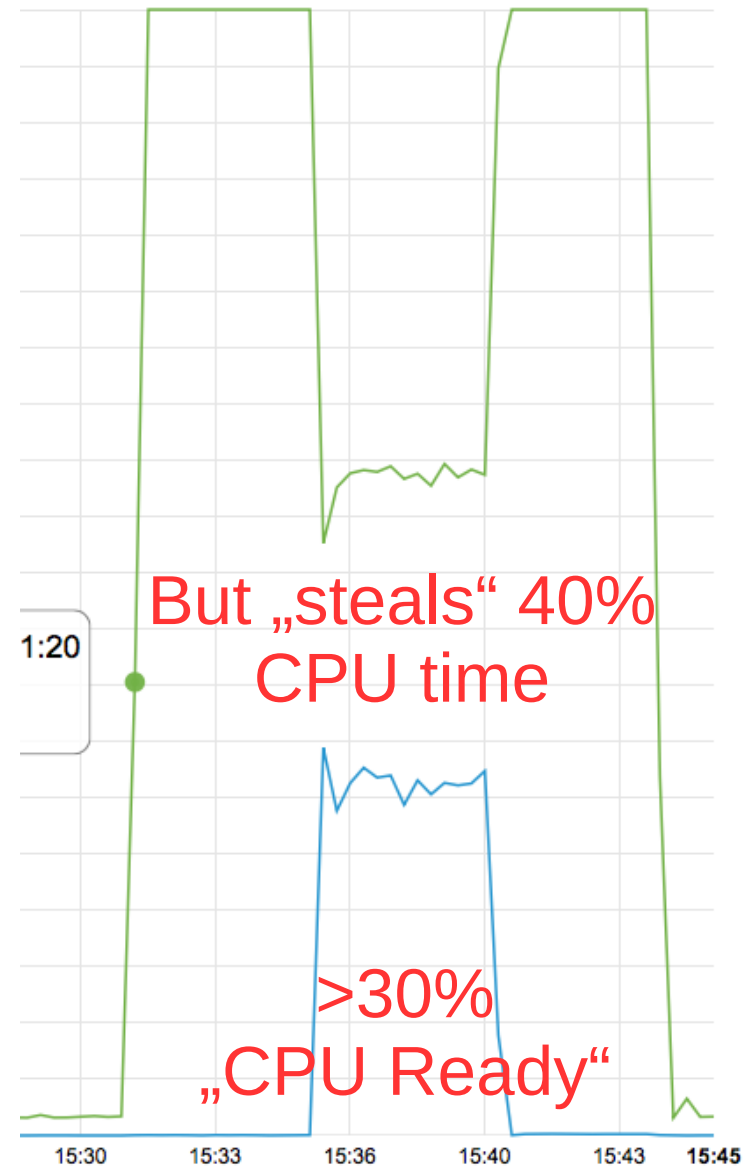
**Will heavily switch
CPU context!**

Oracle vs. Anti-Load

Anti-Load VM CPU



Oracle VM CPU



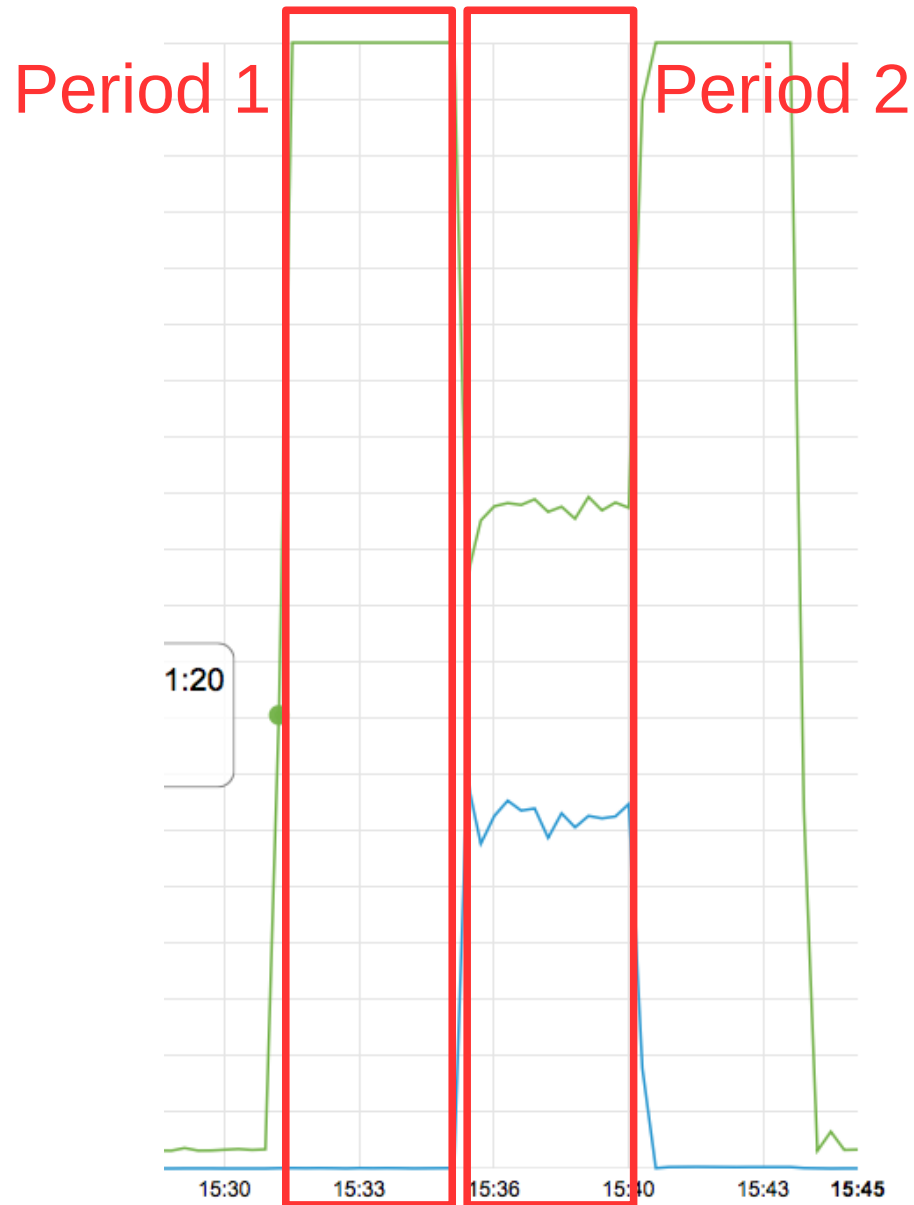
So far, much was
measured w/ help of the
VMware Admin
:(

SMS
=
Surrogate
Measure
Sucks

Let's do it ourselves
Let's do what we do best

Use Oracle

AWR Diff Report



AWR Diff Report

Workload Comparison

	1st Per Sec	2nd Per Sec	%Diff
DB time:	5.9	5.8	-1.2
CPU time:	5.9	5.5	-6.5
Background CPU time:	0.0	0.1	450.0
Redo size (bytes):	17,639,257.8	10,080,982.1	-42.8
Logical read (blocks):	409,888.3	234,796.8	-42.7
Block changes:	149,455.3	85,418.5	-42.8

39% less buffer gets
with SAME CPU TIME
and WORKLOAD

(Linux sees 100% all times)

$409.888,3 \text{ BG} / 5,9 \text{ s} = 69.473 \text{ BG/s}$
 $234.796.8 \text{ BG} / 5,5 \text{ s} = 42.690 \text{ BG/s} = 61\%$

and use your brain!

Simple Benchmark

```
38 ##### 1.547463
39 ##### 1.54470933
39 ##### 1.54470933
40 ##### 1.541621
41 ##### 1.5444705
42 ##### 1.54240183
43 ##### 1.54291683
44 ##### 1.54445533
44 ##### 1.54445533
45 ##### 1.5499345
46 ##### 1.547596
47 ##### 1.547272
48 ##### 1.54804967
48 ##### 1.54804967
49 ##### 1.7625875
49 ##### 1.7625875
49 ##### 1.7625875
50 ##### 3.44203833
50 ##### 3.44203833
51 ##### 3.26774017
51 ##### 3.26774017
51 ##### 3.26774017
52 ##### 3.62154467
52 ##### 3.62154467
53 ##### 3.70635533
53 ##### 3.70635533
54 ##### 3.60270667
54 ##### 3.60270667
55 ##### 3.21821567
56 ##### 2.05004033
56 ##### 2.05004033
57 ##### 1.4670215
58 ##### 1.42301867
59 ##### 1.41232367
60 ##### 1.403707
61 ##### 1.40650833
62 ##### 1.40582217
63 ##### 1.403707
```

First value:

bar:

Decimal value:

ID of test iteration

Visualization

Seconds for

10.000 x SELECT+UPDATE

of different single rows

(Avg. per iteration over all workers,
all workers on different rows)

**Guess,
what happened
here? :(**

**Simulated User Experience of a
DB on VMware (if built poorly)**

Knowledge is important Proof is better

Prepare to Prove

What you need in your VM (Linux)

- VMware tools
(VMware recommends package from your distributor)
- Python environment (usually present)
- git (or copy yourself)
- build environment like make (Oracle needs that anyway)
- vmguestlib with vmguest-stats (made by Dag Wieers)
uses an API provided by vSphere (via VMW tools)

```
$ git clone git://github.com/dagwieers/vmguestlib
$ cd vmguestlib
$ python setup.py install
```

vmguest-stats

```
[root@oracle-vm1 vmguestlib]# ./vmguest-stats -c  
Wed Dec 13 12:12:26 2017
```

VM Processor

Processor Time: 639.07 %

CPU stolen time: 1.09 %

Effective VM Speed: 18737 MHz

Host processor speed: 2932 MHz

Limit: unlimited

Reservation: 0 MHz

Shares: 6000

```
[root@oracle-vm1 vmguestlib]# ./vmguest-stats -c | egrep 'Proc'
```

Processor Time: 627.54 %

CPU stolen time: 0.80 %

Processor Time: 595.20 %

CPU stolen time: 0.50 %

Processor Time: 684.35 %

CPU stolen time: 0.50 %

Processor Time: 653.78 %

CPU stolen time: 0.60 %

Processor Time: 586.93 %

CPU stolen time: 0.50 %

Processor Time: 681.80 %

CPU stolen time: 0.70 %

Processor Time: 607.68 %

vmguest-stats

```
[root@oracle-vm1 vmguestlib]# ./vmguest-stats -c | egrep 'Processor Time|CPU stolen time'
```

```
Processor Time: 658.49 %  
CPU stolen time: 0.83 %  
Processor Time: 653.23 %  
CPU stolen time: 0.20 %  
Processor Time: 672.85 %  
CPU stolen time: 0.60 %  
Processor Time: 606.09 %  
CPU stolen time: 0.30 %  
Processor Time: 729.38 %  
CPU stolen time: 0.50 %  
Processor Time: 613.57 %  
CPU stolen time: 2.99 %  
Processor Time: 620.96 %  
CPU stolen time: 0.70 %  
Processor Time: 677.84 %  
CPU stolen time: 0.20 %  
Processor Time: 651.60 %  
CPU stolen time: 0.80 %  
Processor Time: 673.25 %  
CPU stolen time: 26.35 %  
Processor Time: 498.80 %  
CPU stolen time: 134.50 %  
Processor Time: 477.25 %  
CPU stolen time: 150.70 %  
Processor Time: 476.75 %  
CPU stolen time: 147.90 %  
Processor Time: 463.11 %  
CPU stolen time: 167.40 %  
Processor Time: 487.55 %  
CPU stolen time: 158.37 %  
Processor Time: 452.34 %  
CPU stolen time: 158.92 %  
Processor Time: 468.36 %  
CPU stolen time: 173.65 %
```

Processor Time: 651.60 %

CPU stolen time: 0.80 %

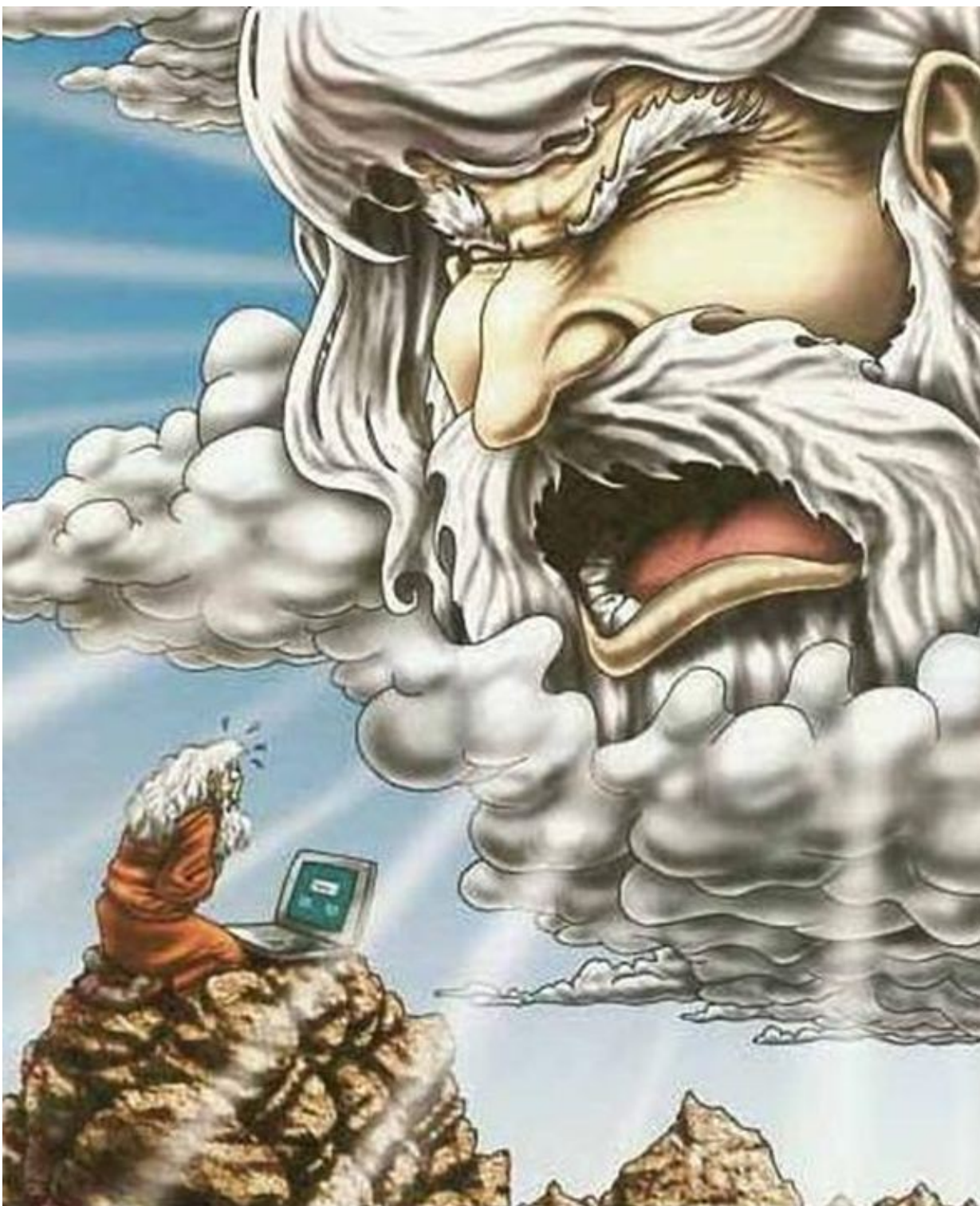
Processor Time: 673.25 %

CPU stolen time: 26.35 %

Processor Time: 498.80 %

CPU stolen time: 134.50 %

Countermeasures



MOSES!!!!

It crashed AGAIN???

I will postpone the
invention of the computer
by 3,000 years!

You can **chisel**
my Ten Commandments
INTO STONE!!

Ten Commandments for OLTP on VMware

And the Lord said:

- 1) Thou shalt not overcommit your machine!
- 2) Thou shalt not use Hyperthreading, because HTT is CPU overcommitment
- 3) Thou shalt make 100% CPU + RAM reservations
- 4) Thou shalt disable CPU Hot Add (enable breaks vNUMA)
- 5) Thou shalt disable RAM Hot Add (enable breaks vNUMA)
- 6) Thou shalt not use more vCores in your VM than a NUMA node of the host has
- 7) Thou shalt not use more RAM in your VM than a NUMA node of the host has
- 8) Thou shalt configure VMs to Latency Sensitivity „HIGH“
- 9) Thou shalt disable all BIOS based Energy Saving Options
- 10) Thou shalt use Paravirtual Drivers at any cost

And peace be on earth.

(More) Ressource Consumption

Makes you (more) vulnerable

Speaker

- Martin Klier
- Solution Architect and Database Expert
- My focus
 - Performance Optimization
 - High Availability
 - Architecture DBMS
- Linux since 1997
- Oracle Database since 2003



ORACLE
ACE Director



Speaker

- Meet & Greet



- Contact: martin.klier@performing-db.com
- Weblog: <http://www.usn-it.de> (English)

Performing Databases

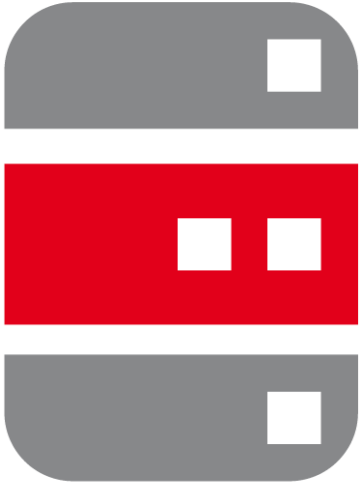
- Experts for Database Technology
 - Concept
 - Planning & Sizing
 - Licensing
 - Implementation and Troubleshooting
- Get in touch
 - Performing Databases GmbH
Wiesauer Straße 27
95666 Mitterteich, GERMANY
 - Web: <http://www.performing-databases.com>
 - Twitter: @PerformingDB





Download my Presentations and Whitepapers
<http://www.performing-databases.com>

performing databases



Your reliability. Our concern.