Scheduling in The Age of Virtualization

Dario Faggioli
dario.faggioli@citrix.com

Bruxelles – 30th of January, 2016

Welcome

- Hello, my name is Dario
- I'm with Citrix since 2011 (in the Xen Platform Team)

Outline

CPU Scheduling in the Virtualization World:

- hypervisor and guest scheduler: same or different?
- hypervisor scheduler: what are the key features?
- hypervisor and guest scheduler: independent or interactive?

Scheduling in The Virtualization World

Virtualization means 2 schedulers always running:

- hypervisor level: schedules virtual CPUs over physical CPUs
- guest OS level: schedules processes over virtual CPUs

Implemented by:

- two instances of the same scheduler (Linux/KVM)
- two different schedulers (Xen, VMWare, Hyper-V)

Same or different: What's better?

Opinions...

Same scheduler approach (Linux/KVM):

- benefit from feature and tuning done by others for other reasons **pro**
- (virtualization) specific tweaks may not always be welcome **contra**

Different schedulers approach (Xen):

- developing a good scheduler is entirely on you **contra**
- virtualization specific tricks could be added at leisure **pro**

My opinion: I like the Xen way better

Same or different: What's better?

Opinions...

Same scheduler approach (Linux/KVM):

- benefit from feature and tuning done by others for other reasons **pro**
- (virtualization) specific tweaks may not always be welcome **contra**

Different schedulers approach (Xen):

- developing a good scheduler is entirely on you **contra**
- virtualization specific tricks could be added at leisure **pro**

My opinion: I like the Xen way better
would have you ever guessed? :-)

Same or different: What's better? (cont.)

There's a story that could be an interesting example. It talks about co-scheduling, but not right now...

What Makes a Good Hypervisor Scheduler?

One thing is **key**:

- *fairness*: if the VMs are equal, they should get equal service in term physical CPU time. If they are not equal, *weighted fairness*.

A couple of other **wish list** things:

- *limit*: this VM should not run more than XX% of physical CPU time.
- *reservation*: whatever the load is, this VM should never get less than YY% physical CPU time.

Where do Linux/KVM and Xen Stand?

	Linux/KVM	**Xen**
Wght Fairness	CFS (Linux 2.6.23)	Forever
Limit	CFS BW Control (Linux 3.2)	Credit (2006)
Reservation	No	Planned for Credit2

Scheduler Example

Wakeup latency test: measure difference between desired and actual wakeup time (min, avg, max).

	Min	**Avg**	**Max**
no other load			
KVM	**25.5**	**30.3**	**41.8**
XEN	68.3	117.3	174.3
load on host/dom0			
KVM	23.6	345.5	17785.3
Xen	**28.3**	**81.3**	**1145.5**
load on other VM			
KVM	36.5	336.8	7423.5
Xen	**28.5**	**90.5**	**1131.5**

Should Hypervisor and Guest OS "Talk to Each Other"?

There is a word: **Paravirtualization**

- let's not go that far (today!)
- maybe just some "enlightenment"

Example 1: Topology Based Scheduler Load Balan

Linux scheduler (in a guest) takes topology into account when load balancing.

- vCPUs wander around among pCPUs: the hypervisor scheduler moves them!
- at time t_1 vCPU 1 and vCPU 3 run on pCPUs that are SMT-siblings
- at time $t_2! = t_1$ ... Not anymore!

"Hey, you're virtualized, please do not make assumptions on topology!"

Example 1: Topology Based Scheduler Load Balancing (cont.)

We're down at doing at, and it looks promising...

	Iperf (VMs to host) % incr. thput.
Sequential host load (1 VM)	+3.976608%
Small host load	+3.903162%
Medium host load	+7.753479%
Large host load	+2.152059%
Full host load	+6.830207%
Overloaded host	+5.257887%
Overwhelmed host	+3.502063%

Example 2: Generic Load Balancing Behaviour

When does Linux's scheduler's load balancer triggers?

- configurable (scheduling domains' `flags`)
- each architecture benchmarks and tune behaviour for best perf.
- virtualized guests (Xen/KVM)? Just what x86 does...

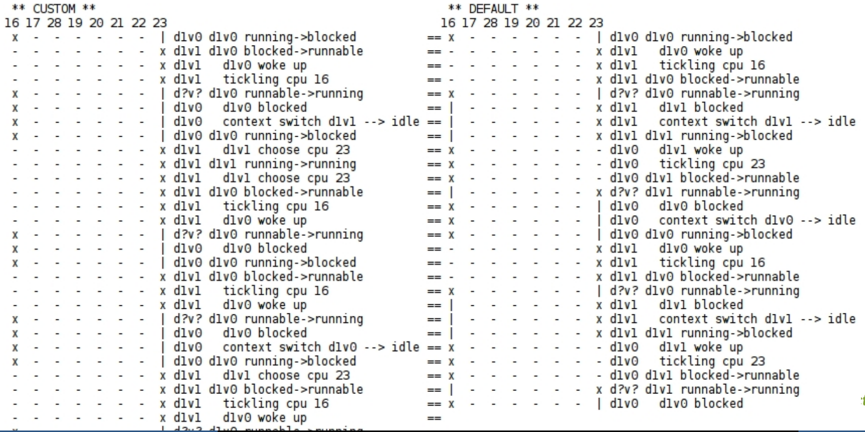
`execl` benchmark from UnixBench. Default vs customised set of flags (removed `SD_BALANCE_EXEC`):

Table: My caption

	DEFAULT	CUSTOM
KVM	675.3	1051.6
XEN	779.9	1009.8

Example 2: Generic Load Balancing Behaviour (cont.)

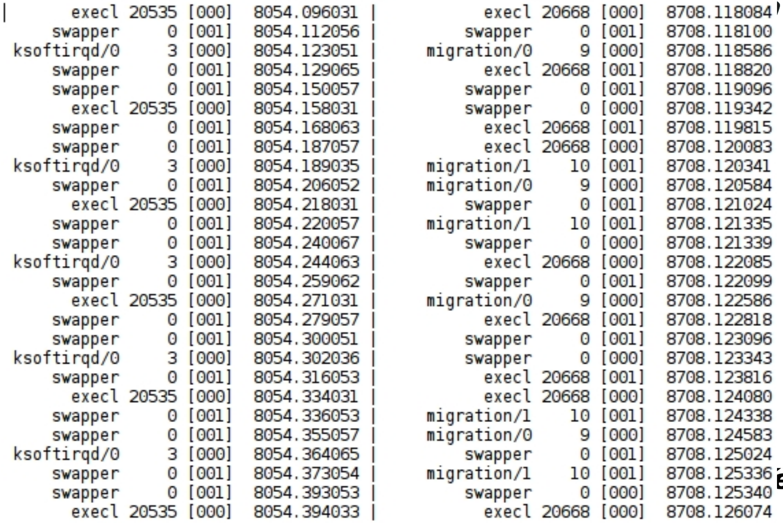
Why? Traces (Xen):
'-' CPU is idle, '|' CPU is doing something, 'x' event happening on CPU



```
  ** CUSTOM **                                                    ** DEFAULT **
16 17 28 19 20 21 22 23                                        16 17 28 19 20 21 22 23
 x  -  -  -  -  -  -  | d1v0 d1v0 running->blocked          == x  -  -  -  -  -  -  | d1v0 d1v0 running->blocked
 -  -  -  -  -  -  -  x d1v1 d1v0 blocked->runnable         == -  -  -  -  -  -  -  x d1v1   d1v0 woke up
 -  -  -  -  -  -  -  x d1v1   d1v0 woke up                 == -  -  -  -  -  -  -  x d1v1   tickling cpu 16
 -  -  -  -  -  -  -  x d1v1   tickling cpu 16              == -  -  -  -  -  -  -  x d1v1 d1v0 blocked->runnable
 x  -  -  -  -  -  -  | d?v? d1v0 runnable->running         == x  -  -  -  -  -  -  | d?v? d1v0 runnable->running
 x  -  -  -  -  -  -  | d1v0   d1v0 blocked                 == |  -  -  -  -  -  -  x d1v1   d1v1 blocked
 x  -  -  -  -  -  -  | d1v0   context switch d1v1 --> idle == |  -  -  -  -  -  -  x d1v1   context switch d1v1 --> idle
 x  -  -  -  -  -  -  | d1v0 d1v0 running->blocked          == |  -  -  -  -  -  -  x d1v1 d1v1 running->blocked
 -  -  -  -  -  -  -  x d1v1   d1v1 choose cpu 23           == x  -  -  -  -  -  -  - d1v0   d1v1 woke up
 -  -  -  -  -  -  -  x d1v1 d1v1 running->running          == x  -  -  -  -  -  -  - d1v0   tickling cpu 23
 -  -  -  -  -  -  -  x d1v1   d1v1 choose cpu 23           == x  -  -  -  -  -  -  - d1v0 d1v1 blocked->runnable
 -  -  -  -  -  -  -  x d1v1 d1v0 blocked->runnable         == |  -  -  -  -  -  -  x d?v? d1v1 runnable->running
 -  -  -  -  -  -  -  x d1v1   tickling cpu 16              == x  -  -  -  -  -  -  | d1v0   d1v0 blocked
 -  -  -  -  -  -  -  x d1v1   d1v0 woke up                 == x  -  -  -  -  -  -  | d1v0   context switch d1v0 --> idle
 x  -  -  -  -  -  -  | d?v? d1v0 runnable->running         == x  -  -  -  -  -  -  | d1v0 d1v0 running->blocked
 x  -  -  -  -  -  -  | d1v0   d1v0 blocked                 == -  -  -  -  -  -  -  x d1v1   d1v0 woke up
 x  -  -  -  -  -  -  | d1v0 d1v0 running->blocked          == -  -  -  -  -  -  -  x d1v1   tickling cpu 16
 -  -  -  -  -  -  -  x d1v1 d1v0 blocked->runnable         == -  -  -  -  -  -  -  x d1v1 d1v0 blocked->runnable
 -  -  -  -  -  -  -  x d1v1   tickling cpu 16              == x  -  -  -  -  -  -  | d?v? d1v0 runnable->running
 -  -  -  -  -  -  -  x d1v1   d1v0 woke up                 == |  -  -  -  -  -  -  x d1v1   d1v1 blocked
 x  -  -  -  -  -  -  | d?v? d1v0 runnable->running         == |  -  -  -  -  -  -  x d1v1   context switch d1v1 --> idle
 x  -  -  -  -  -  -  | d1v0   d1v0 blocked                 == |  -  -  -  -  -  -  x d1v1 d1v1 running->blocked
 x  -  -  -  -  -  -  | d1v0   context switch d1v0 --> idle == x  -  -  -  -  -  -  - d1v0   d1v1 woke up
 x  -  -  -  -  -  -  | d1v0 d1v0 running->blocked          == x  -  -  -  -  -  -  - d1v0   tickling cpu 23
 -  -  -  -  -  -  -  x d1v1   d1v1 choose cpu 23           == x  -  -  -  -  -  -  - d1v0 d1v1 blocked->runnable
 -  -  -  -  -  -  -  x d1v1 d1v0 blocked->runnable         == |  -  -  -  -  -  -  x d?v? d1v1 runnable->running
 -  -  -  -  -  -  -  x d1v1   tickling cpu 16              == x  -  -  -  -  -  -  | d1v0   d1v0 blocked
 -  -  -  -  -  -  -  x d1v1   d1v0 woke up                 ==
 x  -  -  -  -  -  -  | d?v? d1v0 runnable->running
```

Example 2: Generic Load Balancing Behaviour (co

Why? Traces (Linux):



```
|      execl 20535 [000]  8054.096031 |          execl 20668 [000]  8708.118084
     swapper     0 [001]  8054.112056 |        swapper     0 [001]  8708.118100
 ksoftirqd/0     3 [000]  8054.123051 |    migration/0     9 [000]  8708.118586
     swapper     0 [001]  8054.129065 |          execl 20668 [001]  8708.118820
     swapper     0 [001]  8054.150057 |        swapper     0 [001]  8708.119096
       execl 20535 [000]  8054.158031 |        swapper     0 [000]  8708.119342
     swapper     0 [001]  8054.168063 |          execl 20668 [001]  8708.119815
     swapper     0 [001]  8054.187057 |          execl 20668 [000]  8708.120083
 ksoftirqd/0     3 [000]  8054.189035 |    migration/1    10 [001]  8708.120341
     swapper     0 [001]  8054.206052 |    migration/0     9 [000]  8708.120584
       execl 20535 [000]  8054.218031 |        swapper     0 [001]  8708.121024
     swapper     0 [001]  8054.220057 |    migration/1    10 [001]  8708.121335
     swapper     0 [001]  8054.240067 |        swapper     0 [000]  8708.121339
 ksoftirqd/0     3 [000]  8054.244063 |          execl 20668 [000]  8708.122085
     swapper     0 [001]  8054.259062 |        swapper     0 [001]  8708.122099
       execl 20535 [000]  8054.271031 |    migration/0     9 [000]  8708.122586
     swapper     0 [001]  8054.279057 |          execl 20668 [001]  8708.122818
     swapper     0 [001]  8054.300051 |        swapper     0 [001]  8708.123096
 ksoftirqd/0     3 [000]  8054.302036 |        swapper     0 [000]  8708.123343
     swapper     0 [001]  8054.316053 |          execl 20668 [001]  8708.123816
       execl 20535 [000]  8054.334031 |          execl 20668 [000]  8708.124080
     swapper     0 [001]  8054.336053 |    migration/1    10 [001]  8708.124338
     swapper     0 [001]  8054.355057 |    migration/0     9 [000]  8708.124583
 ksoftirqd/0     3 [000]  8054.364065 |        swapper     0 [001]  8708.125024
     swapper     0 [001]  8054.373054 |    migration/1    10 [001]  8708.125336
     swapper     0 [001]  8054.393053 |        swapper     0 [000]  8708.125340
       execl 20535 [000]  8054.394033 |          execl 20668 [000]  8708.126074
```

Q&A

Thanks again,

~~Paravirtualization!~~
Questions?