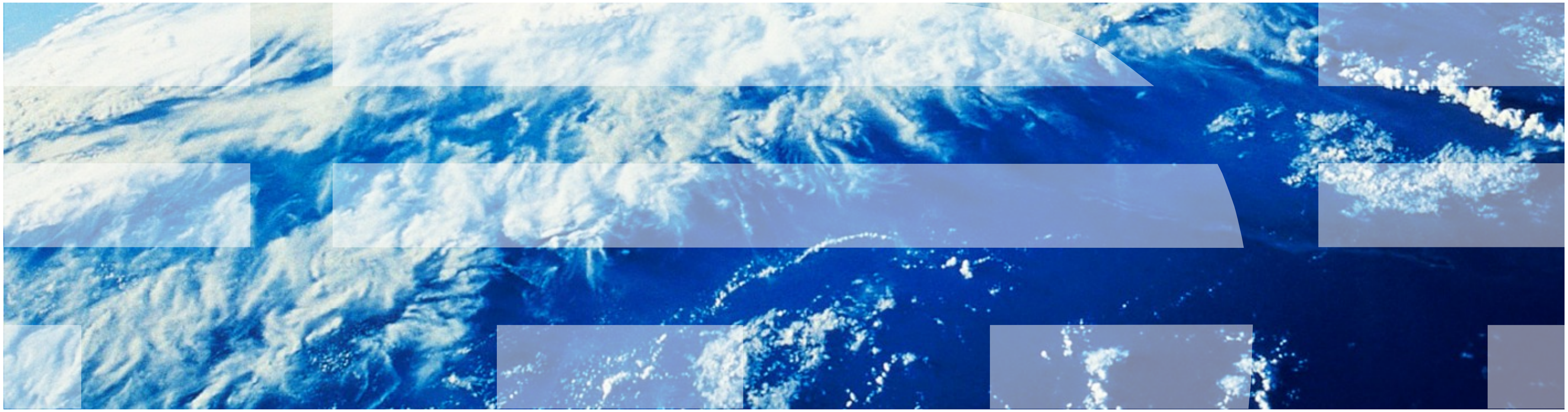
Christian Ehrhardt EHRHARDT@de.ibm.com

7/14/10

CPU time accounting



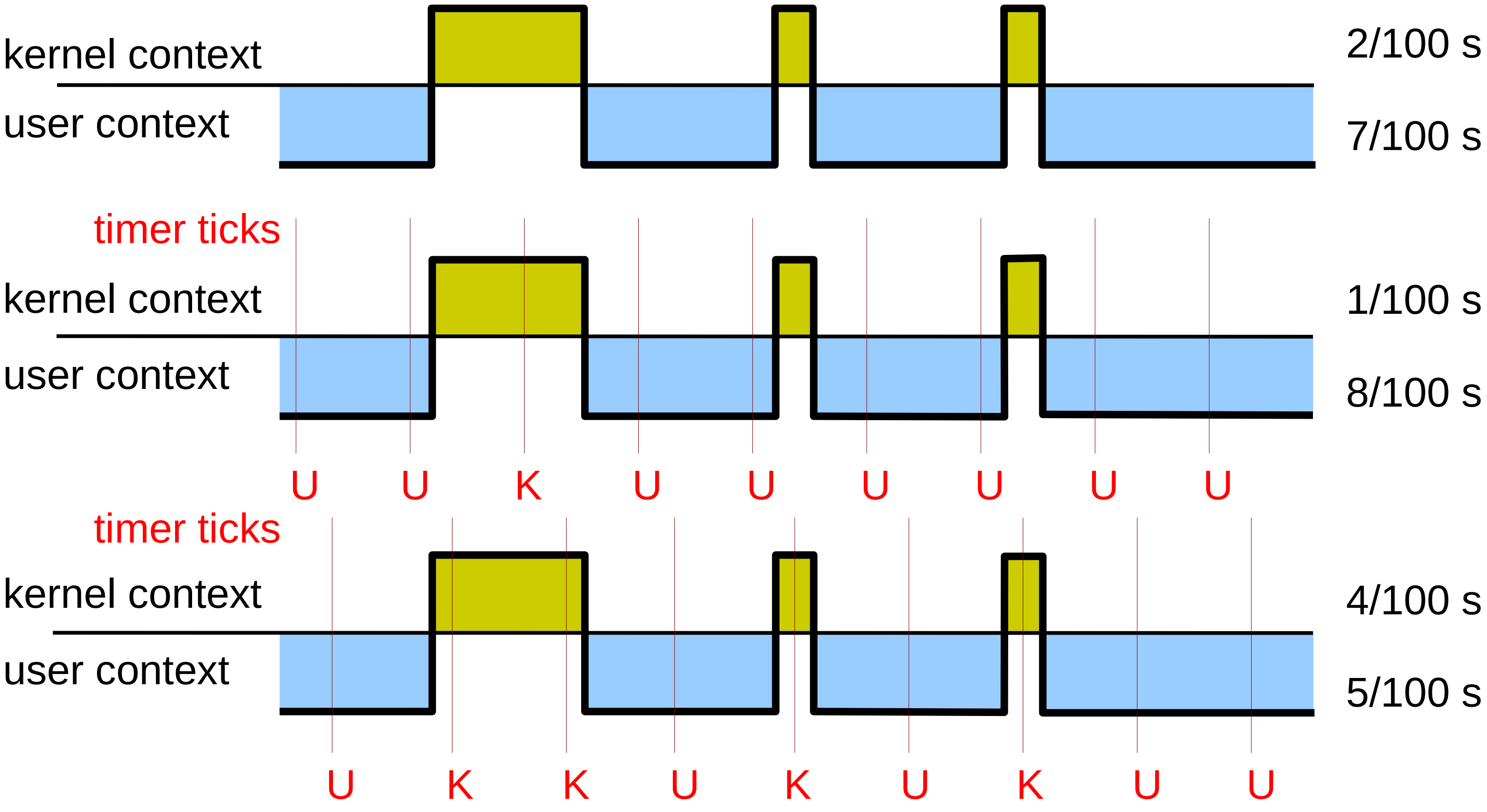
visit us at http://www.ibm.com/developerworks/linux/linux390/perf/index.html

Tick based CPU time accounting

- The Linux tick based CPU time accounting is heuristic
  - When a timer interrupt occurs - which is every 1/100 second on a zSeries machine - Linux checks which context has been interrupted.
  - It accounts the complete time slice to this context.
  - For example, if the timer interrupt occurs in a kernel context, then the complete time slice is accounted as system time.

Tick based CPU time inaccuracy

- This example shows: Depending on when the timer interrupt occurs, the CPU time accounting may look different. For simplification only the user and kernel time is displayed.



kernel context

user context

2/100 s

7/100 s

timer ticks

kernel context

user context

1/100 s

8/100 s

U U K U U U U U U

timer ticks

kernel context

user context

4/100 s

5/100 s

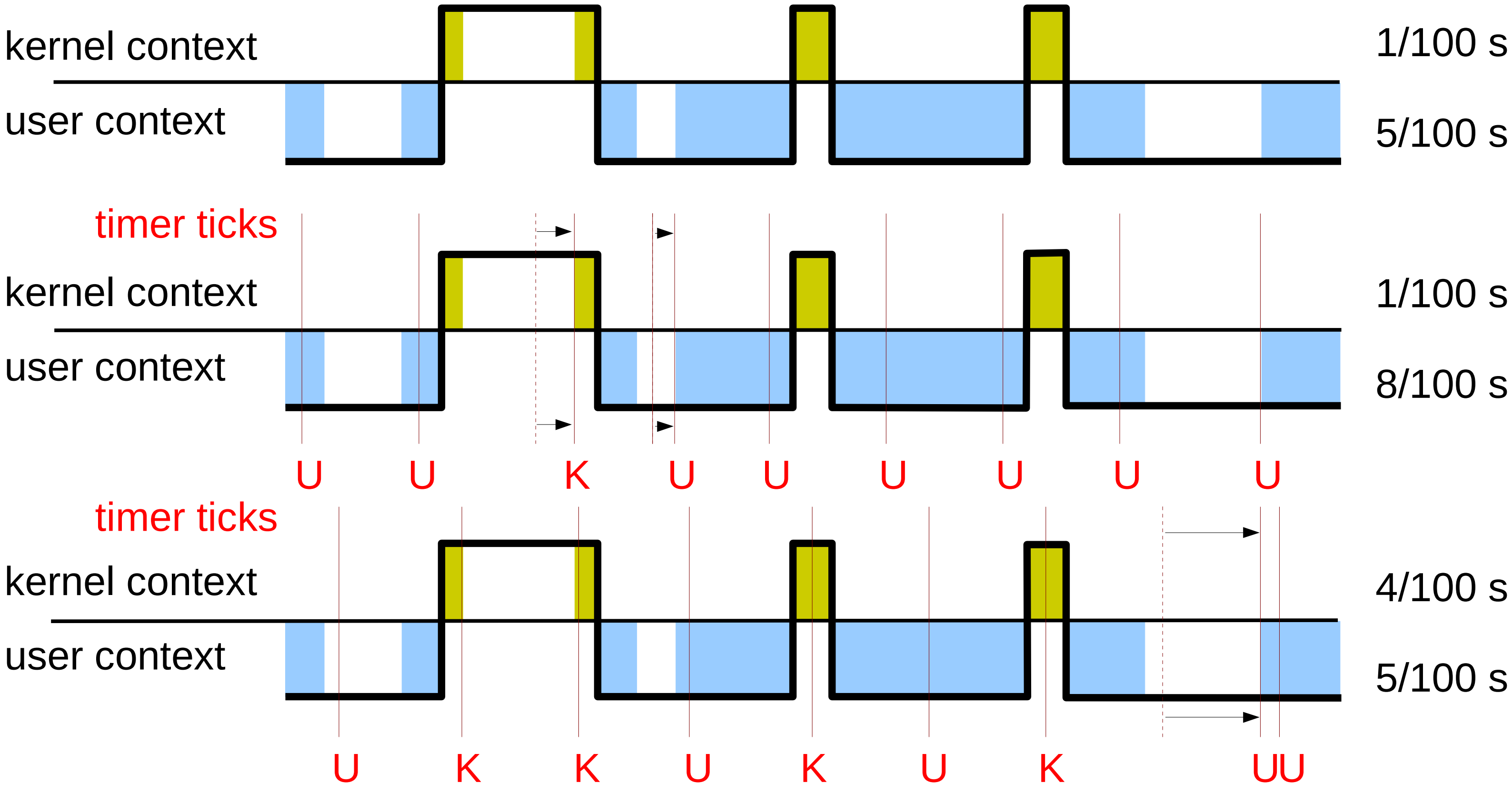
U K K U K U K U U

Tick based CPU accounting on virtual systems (1)

- On systems with virtual CPUs (like z/VM, or Xen) the tick based CPU time accounting is not precise enough, when using more virtual CPUs than real CPUs on the virtual platform.
- In this case, the real CPUs might spend part of their time servicing another virtual processor, while the time slice might be accounted to a process which actually could not utilize it.
- Therefore the Linux reported CPU times can highly deviate from the load numbers reported by the hypervisor of a virtual platform.

Tick based CPU accounting on virtual systems (2)

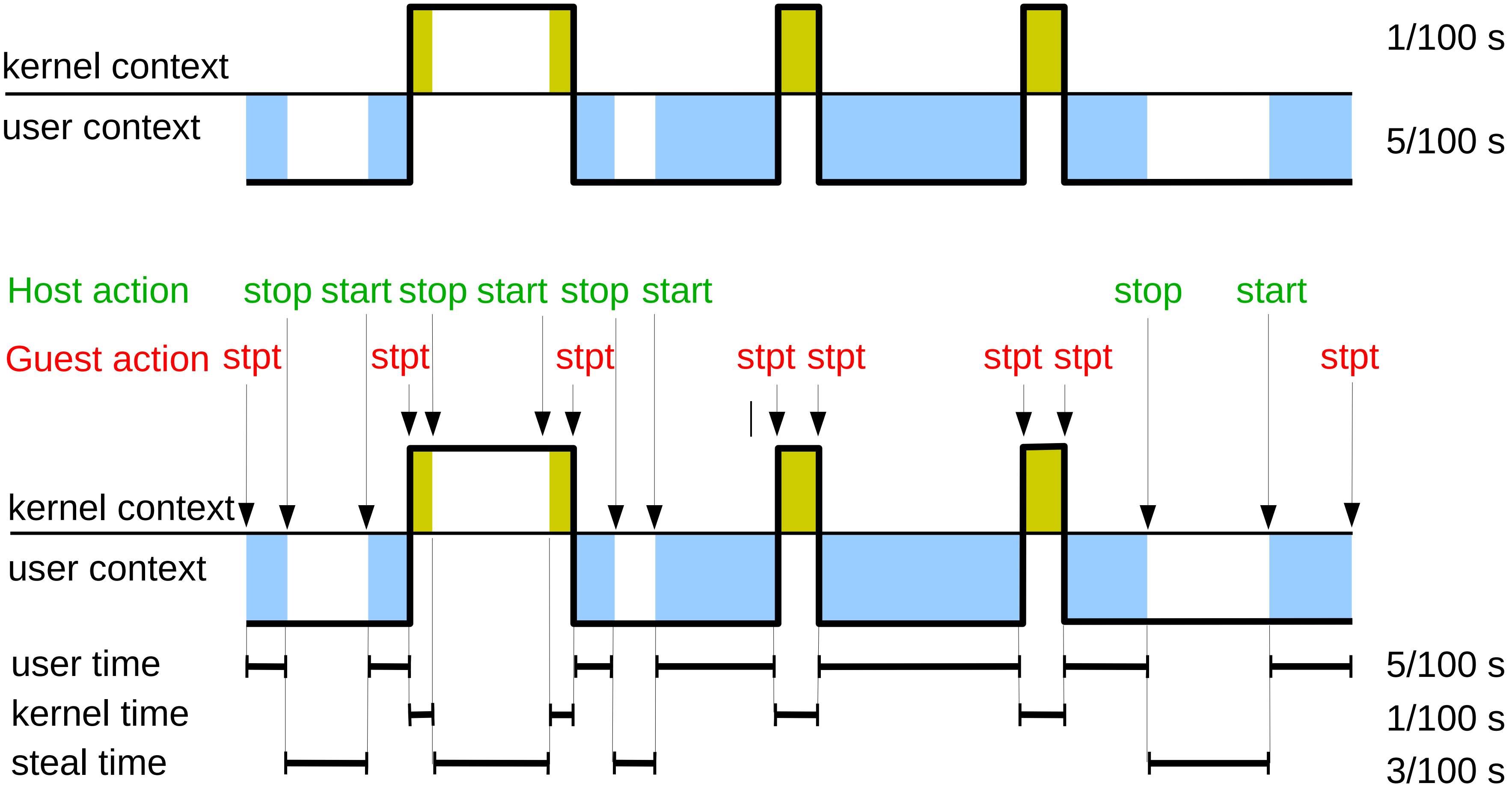
- This example shows involuntary wait states (called "steal time", in white) for Linux images running on virtual systems.



Virtual CPU time accounting

- A precise CPU time accounting for virtual systems must provide the following features
  - Distinction between real and virtual CPU time
  - Provision of a concept for showing wait states caused by involuntary wait (steal time)
- For Linux on System z a new CPU time accounting, called "virtual CPU time accounting" has been implemented from Linux kernel 2.6.11 on, which is based on the virtual System z virtual CPU timer instead of using wall-time as in the tick based CPU accounting
  - Each CPU has its own CPU timer.
  - The stepping rate of a CPU timer is synchronized with the system TOD (Time-of-Day) clock, but is only incremented when a virtual CPU is backed by a physical CPU.
  - The stpt (Store CPU Timer) instruction is added to the Linux system call path.
  - By storing the CPU timer, the really used CPU time can be calculated for a Linux image.
- Virtual CPU time accounting accounts CPU times, whenever the execution context changes
  - This is much more precise than the tick based accounting scheme.
  - Those two features - distinction between real and virtual CPU time and explicitly exposing steal times - guarantee correct CPU time accounts for Linux images running on virtualized or on non-virtualized platforms.

New Virtual CPU time accounting example



stpt = Store CPU Timer

Virtual CPU time accounting (2)

- The Linux distributions SUSE SLES10 and Red Hat RHEL5 use the virtual CPU time accounting by default. However this feature is kernel configurable and allows the alternative enabling of tick based CPU time accounting.

- The patches for virtual CPU time accounting are included from kernel 2.6.11 on.

Precise CPU times

- For Linux on System z running in an LPAR or as a z/VM guest, the Linux standard commands which display process time information (top, vmstat, ps, time) show precise CPU times when virtual CPU time accounting is enabled.
- The semantics of the reported CPU time numbers has changed with the introduction of virtual CPU time accounting
  - The numbers based on tick based CPU time accounting report CPU times spent in a virtual CPU context.
  - The numbers based on virtual CPU time accounting report CPU times spent in a real CPU context.
  - The internal precision of the CPU time numbers is at least 1 microsecond on a System z machine.
- The Linux distributions SUSE SLES10 and Red Hat RHEL5 use the virtual CPU time accounting by default. However this feature is kernel configurable and allows the alternative enabling of tick based CPU time accounting.
- The patches for virtual CPU time accounting are included from kernel 2.6.11 on.

- Further a new field "steal time" is added to the /proc/stat file interface.

Steal time

- Steal time is the percentage of time a virtual CPU waits for a real CPU while the hypervisor is not scheduling this virtual CPU.

- Tools displaying steal time
  - Newer versions of the Linux standard commands displaying CPU time information now show this number. Older versions include steal time in CPU idle time.
  - The popular sysstat utilities package, which is often used for detailed performance analysis purposes, has been updated and now shows the CPU steal time.
    - sysstat 6.0.2: mpstat, iostat display CPU steal time
    - sysstat 7.0.0: mpstat, iostat, and sar -u display the CPU steal time

Sample TOP output

- adds field “CPU steal time”
  - – the time Linux wanted to run on a CPU, but the hipervisor was not able to schedule CPU
  - – true for LPAR and z/VM hipervisor
  - – included in SLES10 and RHEL5

```
top - 09:50:20 up 11 min,  3 users,  load average: 8.94, 7.17, 3.82
Tasks:  78 total,   8 running,  70 sleeping,   0 stopped,   0 zombie
 Cpu0 : 38.7%us,  4.2%sy,  0.0%ni,  0.0%id,  2.4%wa,  1.8%hi,  0.0%si, 53.0%st
 Cpu1 : 38.5%us,  0.6%sy,  0.0%ni,  5.1%id,  1.3%wa,  1.9%hi,  0.0%si, 52.6%st
 Cpu2 : 54.0%us,  0.6%sy,  0.0%ni,  0.6%id,  4.9%wa,  1.2%hi,  0.0%si, 38.7%st
 Cpu3 : 49.1%us,  0.6%sy,  0.0%ni,  1.2%id,  0.0%wa,  0.0%hi,  0.0%si, 49.1%st
 Cpu4 : 35.9%us,  1.2%sy,  0.0%ni, 15.0%id,  0.6%wa,  1.8%hi,  0.0%si, 45.5%st
 Cpu5 : 43.0%us,  2.1%sy,  0.7%ni,  0.0%id,  4.2%wa,  1.4%hi,  0.0%si, 48.6%st
Mem:    251832k total,   155448k used,    96384k free,     1212k buffers
Swap:   524248k total,    17716k used,   506532k free,    18096k cached
```

Typical activities accounted as steal time

- If steal time shows high numbers, the first question is about the cause of this undesired situation. Possible answers are
  - processors shared by too many running systems
  - time spent in the z/VM control program
  - "share per virtual processor" for this Linux guest too low
  - high number of diagnose 44 or 9C issued by the Linux guest
  - Linux guest network I/O through VSWITCH

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies.