2009 System z Expo

October 5 – 9, 2009 – Orlando, FL

Session Title:
Performance Monitoring on Linux for IBM System z

Session ID: zLP01

Speaker Name: Dr. Juergen Doelle

Authorized IBM Training

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States, other countries, or both.

Not all common law marks used by IBM are listed on this page. Failure of a mark to appear does not mean that IBM does not use the mark nor does it mean that the product is not actively marketed or is not significant within its relevant market.

Those trademarks followed by ® are registered trademarks of IBM in the United States; all others are trademarks or common law marks of IBM in the United States.

For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

*, AS/400®, e business(logo)®, DBE, ESCO, eServer, FICON, IBM®, IBM (logo)®, iSeries®, MVS, OS/390®, pSeries®, RS/6000®, S/30, VM/ESA®, VSE/ESA, WebSphere®, xSeries®, z/OS®, zSeries®, z/VM®, System i, System i5, System p, System p5, System x, System z, System z9®, BladeCenter®
DB2*, DB2 Connect, DB2 Universal Database, Informix®, Tivoli®, ECKD, Enterprise Storage Server®, FICON Express, HiperSocket, OSA, OSA Express

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:
Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Agenda

- **Objectives**
- **Tools**
  - System
    - vmstat
    - sadc/sar
  - Disk
    - iostat
    - DASD statistics
    - SCSI statistics
  - Network
    - netstat
  - Processes
    - top
    - ps
- **Assuming we have a problem …**
  - .. somewhen
  - .. now
- **Summary**

Objectives

- **There are really many tools available on Linux to monitor the system!**
- **Describe tools available on Linux on System z, which are**
  - available via the Distribution (SUSE, RedHat)
  - specific for Linux on System z (the most are of general use)
  - in use and proven by the Linux Performance team for Linux on System z
- **Help to decide depending on what should be monitored**
  - which is the right tool
  - how should it be used
- **If you are aware of other tools giving more/better information, let me know!**

How and when to use the tools

- **Basic considerations**
  - Monitoring could impact the system
  - Each data gathering averages over a certain period of time
    ==> this flattens peaks
  - Start with defining the problem!
    - which parameter(s) from the application indicates the problem (mostly response or execution times)
    - which range is considered as bad, what is considered as good
    - monitor the good case and save the results for comparison when a problem occurs
  - Use meaningful names for the output files (e.g. tool_test_case_date_and_time)

- **The next slides describe the tools we use**

- **We need a strategy how to use the tools**
  - When we have a problem somewhen
  - To analyze a problem occurring now

- **Disclaimer:**
  - The following advices are no guarantee that each problem can be identified in all cases!

Agenda

- Objectives
- **Tools**
  - **System**
    - **vmstat**
    - **sadc/sar**
  - Disk
    - iostat
    - DASD statistics
    - SCSI statistics
  - Network
    - netstat
  - Processes
    - top
    - ps
- Assuming we have a problem …
  - .. somewhen
  - .. now
- Summary

vmstat

- **Characteristics:** **Easy to use, high-level information**
- **Objective:** **First and fast impression of the current state**
- **Usage:** vmstat [interval in sec]

- **Output sample:**

```
vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- -----cpu------
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 2  2      0 4415152  64068 554100    0    0     4 63144  350   55 29 64  0  3  4
 3  0      0 4417632  64832 551272    0    0     0   988  125   60 32 67  0  0  1
 3  1      0 4415524  68100 550068    0    0     0  5484  212   66 31 64  0  4  1
 3  0      0 4412672  68856 552408    0    0     0    40  109   48 32 68  0  0  0
 3  0      0 4414408  69656 549544    0    0     0     0  103   36 32 68  0  0  0
 3  0      0 4411184  70500 552312    0    0     0     0  104   37 33 67  0  0  0
 3  0      0 4411804  72188 549592    0    0     0  8984  230   42 32 67  0  0  1
 3  0      0 4405232  72896 555592    0    0     0    16  105   52 32 68  0  0  0
```

- **Shows**
  - Data per time interval
  - CPU utilization
  - Disk I/O
  - Memory usage/Swapping

- **Hints**
  - Shared memory usage is listed under 'cache'

sadc/sar

- **Characteristics: Very comprehensive, statistics data on device level**
- **Objective: Suitable for permanent system monitoring and detailed analysis**
- **Usage (recommended):**
  - monitor /usr/lib64/sa/sadc -d [interval in sec] [outfile]
  - view sar -A -f [outfile]

- **Shows**
  - CPU utilization
  - Disk I/O overview and on device level
  - Network I/O and errors on device level
  - Memory usage/Swapping
  - … and much more
  - Reports statistics data over time and creates average values for each item

- **Hints**
  - Specify -d parameter to sadc to include disk device statistics (increase size of outfile)
  - Shared memory is listed under 'cache'
  - [outfile] is a binary file, which contains all values. It is formatted using sar
    - enables the creation of item specific reports, e.g. network only
    - enables the specification of a start and end time → averages are created for the time of interest

sadc/sar (cont.)

- **Some output samples:**
  - CPU load

```
14:19:29        CPU     %user     %nice   %system   %iowait    %steal     %idle
14:20:29        all      2.61      0.00      0.24      0.26      0.09     96.80
14:20:29          0     13.09      0.00      0.88      0.62      0.45     84.96
14:20:29          1      0.63      0.00      0.12      0.03      0.02     99.20
...
Average:        all     88.13      0.00      1.75      0.05      1.29      8.78
Average:          0     85.61      0.00      5.20      0.03      3.19      5.97
Average:          1     88.98      0.00      1.08      0.04      0.93      8.97
```

  - Network load

```
14:19:29        IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s
...
Average:           lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00
Average:         sit0      0.00      0.00      0.00      0.00      0.00      0.00      0.00
Average:         eth3   3587.47   3604.13   2002.09   5381.80      0.00      0.00      0.00
Average:         eth1      0.04      0.00      0.01      0.00      0.00      0.00      0.04
```

  - Memory usage

```
14:19:29    kbmemfree kbmemused  %memused kbbuffers  kbcached kbswpfree kbswpused  %swpused  kbswpcad
...
Average:        46312   8176012     99.44     13142   1906865    319172    729764     69.57    387029
```

sadc/sar (cont.)

- **Some output samples:**
  - Disk I/O – paging statistics

```
14:19:29     pgpgin/s pgpgout/s   fault/s  majflt/s  pgfree/s pgscank/s pgscand/s pgsteal/s    %vmeff
14:20:29       493.62    117.47   1039.55      0.00    413.26    253.97      0.00    246.28     96.97
14:21:29      1148.91    148.14   3757.45      0.00    734.55    553.94      0.00    536.69     96.89
14:22:29       357.95    183.74   7039.62      0.00    508.28    338.69      0.00    328.42     96.97
14:23:29       286.27    552.20   8352.07      0.00    499.92    327.25      0.00    317.33     96.97
14:24:29       255.05    276.37   9260.43      0.02    486.67    323.29      0.00    313.50     96.97
...
Average:       117.22    503.81   3647.43      0.03    312.71    385.32     67.19    127.55     28.19
```

  - Disk I/O – device level

```
04:46:59        DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
04:47:29    dev94-0      0.83      0.00     80.83     96.96      0.00      1.20      0.80      0.07
04:47:29    dev94-4      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
```

  - Identify the devices via cat /proc/partitions

```
major minor  #blocks  name

  94      0    7212240 dasda
  94      1    7212144 dasda1
  94      4    7212240 dasdb
  94      5         96 dasdb1
  94      6    7212048 dasdb2
```

Agenda

- **Objectives**
- **Tools**
  - System
    - vmstat
    - sadc/sar
  - **Disk**
    - **iostat**
    - **DASD statistics**
    - **SCSI statistics**
  - Network
    - netstat
  - Processes
    - top
    - ps
- **Assuming we have a problem …**
  - .. somewhen
  - .. now
- **Summary**

iostat

- **Characteristics:** **Easy to use, information on disk device level**
- **Objective:** **Detailed input/output disk statistics**
- **Usage:** iostat -xtdk [interval in sec]

- **Shows**
  - Throughput
  - Request merging
  - Device queue information
  - Service times

- **Hints**
  - Most critical parameter is *await*
    - average time (in milliseconds) for I/O requests issued to the device to be served.
    - includes the time spent by the requests in queue and the time spent servicing them.
  - Also suitable for network file systems

iostat

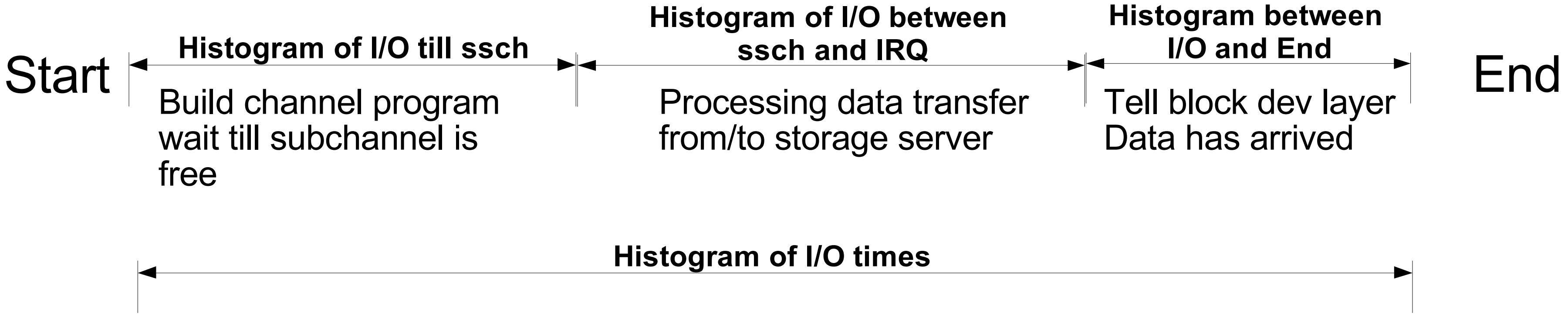
- **Output sample:**

```
Time: 10:56:35 AM
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await  svctm  %util
dasda             0.19     1.45    1.23    0.74    64.43     9.29    74.88     0.01    2.65   0.80   0.16
dasdb             0.02   232.93    0.03    9.83     0.18   975.17   197.84     0.98   99.80   1.34   1.33

Time: 10:56:36 AM
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await  svctm  %util
dasda             0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
dasdb             0.00  1981.55    0.00  339.81     0.00  9495.15    55.89     0.91    2.69   1.14  38.83

Time: 10:56:37 AM
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await  svctm  %util
dasda             0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
dasdb             0.00  2055.00    0.00  344.00     0.00  9628.00    55.98     1.01    2.88   1.19  41.00
```

DASD statistics

- **Characteristics:** **Easy to use, very detailed**
- **Objective:** **Collects statistics of I/O operations on DASD devices**
- **Usage:**
  - enable: echo on > /proc/dasd/statistics
  - show:
    - overall cat /proc/dasd/statistics
    - for individual DASDs tunedasd -P /dev/dasda

- **Shows:**
  - various processing times:


Start
Histogram of I/O till ssch
Build channel program wait till subchannel is free
Histogram of I/O between ssch and IRQ
Processing data transfer from/to storage server
Histogram between I/O and End
Tell block dev layer Data has arrived
End
Histogram of I/O times

DASD statistics - report

- **Sample:**

4KB <= request size < 8 KB

1ms <= response time < 2 ms

```
29432 dasd I/O requests
with 6227424 sectors(512B each)
   __<4    ___8    __16    __32    __64    _128    _256    _512    __1k    __2k    __4k    __8k    _16k    _32k    _64k    128k
   _256    _512    __1M    __2M    __4M    __8M    _16M    _32M    _64M    128M    256M    512M    __1G    __2G    __4G    _>4G

Histogram of sizes (512B secs)
      0       0    9925    3605    1866    4050    4102     933    2700    2251       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0       0       0       0       0       0       0

Histogram of I/O times (microseconds)
      0       0       0       0       0       0       0    1283    1249    6351    7496    3658    8583     805       7       0
      0       0       0       0       0       0       0       0       0       0       0       0       0       0       0       0

Histogram of I/O time till ssch
   2314     283      98      34      13       5      16     275     497    8917    5567    4232    7117      60       4       0
      0       0       0       0       0       0       0       0       0       0       0       0       0       0       0       0
Histogram of I/O time between ssch and irq
      0       0       0       0       0       0       0   14018    7189    2402    1031    4758      27       4       3       0
      0       0       0       0       0       0       0       0       0       0       0       0       0       0       0       0

Histogram of I/O time between irq and end
   2733       6    5702    9376    5781     940    1113    3781       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0       0       0       0       0       0       0

# of req in chanq at enqueuing (1..32)
      0    2740     628    1711    1328   23024       0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0       0       0       0       0       0       0
```

- **Hints**
  - Also shows data per sector

FCP statistics

- **Characteristics:** **Detailed latency information (SLES9 and SLES10)**
- **Objective:** **Collects statistics of I/O operations on FCP devices on request base, separate for read/write**

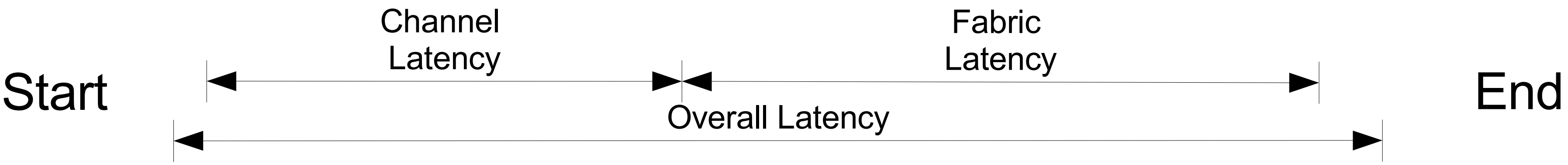
- **Usage:**
  - enable
    - CONFIG_STATISTICS=y must be set in the kernel config file
    - debugfs is mounted at /sys/kernel/debug/
    - For a certain LUN in directory `/sys/kernel/debug/statistics/zfcp-<device-bus-id>-<WWPN>-<LUN>`
      - issue `echo on=1 > definition` (turn off with on=0, reset with data=reset)
  - view
    - `cat /sys/kernel/debug/statistics/zfcp-<device-bus-id>-<WWPN>-<LUN>/data`

- **Hint**
  - FCP and DASD statistics are not directly comparable, because in the FCP case many I/O requests can be sent to the same LUN before the first response is given. There is a queue at FCP driver entry and in the storage server

FCP statistics

- **Shows:**
  - Request sizes in bytes (hexadecimal)
  - Channel latency Time spent in the FCP channel in nanoseconds
  - Fabric latency processing data transfer from/to storage server incl. SAN in nanoseconds
  - (Overall) latencies whole time spent in the FCP layer in milliseconds

  - Calculate the pass through time for the FCP layer as
    `pass through time = overall latency – (channel latency + fabric latency)`
    → Time spent between the Linux device driver and FCP channel adapter inclusive in Hypervisor



FCP statistics example

```
cat /sys/kernel/debug/statistics/zfcp-0.0.1700-0x5005076303010482-0x4014400500000000/data
...
request_sizes_scsi_read 0x1000 1163
request_sizes_scsi_read 0x80000 805
request_sizes_scsi_read 0x54000 47
request_sizes_scsi_read 0x2d000 44
request_sizes_scsi_read 0x2a000 26
request_sizes_scsi_read 0x57000 25
request_sizes_scsi_read 0x1e000 25
request_sizes_scsi_read 0x63000 24
request_sizes_scsi_read 0x6f000 19
request_sizes_scsi_read 0x12000 19
...
latencies_scsi_read <=1 1076
latencies_scsi_read <=2 205
latencies_scsi_read <=4 575
latencies_scsi_read <=8 368
latencies_scsi_read <=16 0
...
channel_latency_read <=16000 0
channel_latency_read <=32000 983
channel_latency_read <=64000 99
channel_latency_read <=128000 115
channel_latency_read <=256000 753
channel_latency_read <=512000 106
channel_latency_read <=1024000 141
channel_latency_read <=2048000 27
channel_latency_read <=4096000 0
...
fabric_latency_read <=1000000 1238
fabric_latency_read <=2000000 328
fabric_latency_read <=4000000 522
fabric_latency_read <=8000000 136
fabric_latency_read <=16000000 0
...
```

request size 4KB, 1163 occurrences

response time <= 1ms

response time <= 32µs

response time <= 1ms

Agenda

- Objectives
- **Tools**
  - System
    - vmstat
    - sadc/sar
  - Disk
    - iostat
    - DASD statistics
    - SCSI statistics
  - **Network**
    - **netstat**
  - Processes
    - top
    - ps
- Assuming we have a problem …
  - .. somewhen
  - .. now
- Summary

netstat -s

- **Characteristics:** **Easy to use, very detailed information**
- **Objective:** **Display summary statistics for each protocol**
- **Usage:** netstat -s

- **Shows**
  - Information to each protocol
  - Amount of incoming and outgoing packages
  - Various error states, for example TCP segments retransmitted!

- **Hints**
  - Shows accumulated values since system start, therefore mostly the differences between two snapshots are needed
  - There is always a low amount of packets in error or resets
  - Retransmits occurring only when the system is sending data
    When the system is not able to receive, then the sender shows retransmits
  - Use sadc/sar to identify the device

netstat -s

- **Output sample:**

```
Tcp:
  15813 active connections openings
  35547 passive connection openings
  305 failed connection attempts
  0 connection resets received
  6117 connections established
  81606342 segments received
  127803327 segments send out
  288729 segments retransmitted
  0 bad segments received.
  6 resets sent
```

Agenda

- Objectives
- **Tools**
  - System
    - vmstat
    - sadc/sar
  - Disk
    - iostat
    - DASD statistics
    - SCSI statistics
  - Network
    - netstat
  - **Processes**
    - **top**
    - **ps**
- Assuming we have a problem …
  - .. somewhen
  - .. now
- Summary

top

- **Characteristics:** **Easy to use**
- **Objective:** **Shows resource usage on process level**
- **Usage:** top -b -d [interval in sec] > [outfile]

- **Shows**
  - CPU utilization
  - Detailed memory usage

- **Hints**
  - Parameter -b enables to write the output for each interval into a file
  - Use -p [pid1, pid2,...] to reduce the output to the processes of interest
  - Configure displayed columns using 'f' key on the running top program
  - Use the 'W' key to write current configuration to ~/.toprc
    → becomes the default

top (cont.)

- **See ~/.toprc file in backup**

- **Output sample:**

```
top - 11:12:52 up  1:11,  3 users,  load average: 1.21, 1.61, 2.03
Tasks:  53 total,   5 running,  48 sleeping,   0 stopped,   0 zombie
Cpu(s):  3.0%us,  5.9%sy,  0.0%ni, 79.2%id,  9.9%wa,  0.0%hi,  1.0%si,  1.0%st
Mem:   5138052k total,   801100k used,  4336952k free,   447868k buffers
Swap:       88k total,        0k used,       88k free,   271436k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  P SWAP DATA WCHAN     COMMAND
 3224 root      18   0  1820  604  444 R  2.0  0.0   0:00.56 0 1216  252 -         dbench
 3226 root      18   0  1820  604  444 R  2.0  0.0   0:00.56 0 1216  252 -         dbench
 2737 root      16   0  9512 3228 2540 R  1.0  0.1   0:00.46 0 6284  868 -         sshd
 3225 root      18   0  1820  604  444 R  1.0  0.0   0:00.56 0 1216  252 -         dbench
 3230 root      16   0  2652 1264  980 R  1.0  0.0   0:00.01 0 1388  344 -         top
    1 root      16   0   848  304  256 S  0.0  0.0   0:00.54 0  544  232 select    init
    2 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 0    0    0 migration migration/0
    3 root      34  19     0    0    0 S  0.0  0.0   0:00.00 0    0    0 ksoftirqd ksoftirqd/0
    4 root      10  -5     0    0    0 S  0.0  0.0   0:00.13 0    0    0 worker_th events/0
    5 root      20  -5     0    0    0 S  0.0  0.0   0:00.00 0    0    0 worker_th khelper
```

- **Hints**

▶ virtual memory:	VIRT = SWAP + RES	unit KB
▶ physical memory used:	RES = CODE + DATA	unit KB
▶ shared memory	SHR	unit KB

Linux ps command

- **Characteristics:** **very comprehensive, statistics data on process level**
- **Objective:** **reports a snapshot of the current processes**
- **Usage (recommended):**
  ps -eo pid,tid,nlwp,policy,user,tname,ni,pri,psr,sgi_p,stat,wchan:12,start_time,time, pcpu,pmem,vsize,size, rss,share,command

```
 PID   TID NLWP POL USER     TTY       NI PRI PSR P STAT WCHAN        START     TIME %CPU %MEM   VSZ    SZ   RSS - COMMAND
...
 871   871    1 TS  root     ?         -5  29   0 * S<   kauditd_thre 10:01 00:00:00  0.0  0.0     0     0     0 - [kauditd]
2319  2319    1 TS  root     ?          0  23   0 * Ss   poll         10:01 00:00:00  0.0  0.0  2332   264   756 - /sbin/syslog-ng
2322  2322    1 TS  root     ?          0  23   0 * Ss   syslog       10:01 00:00:00  0.0  0.0  1940   376   588 - /sbin/klogd -c 7 -x -x
2324  2324    1 TS  daemon   ?          0  23   0 * Ss   poll         10:01 00:00:00  0.0  0.0  4524   288  1168 - /usr/sbin/slpd
2350  2350    2 TS  root     ?         -3  26   0 * S<sl select       10:01 00:00:00  0.0  0.0 10188  8452   696 - /sbin/auditd -n
2352  2352    1 TS  nobody   ?          0  23   0 * Ss   poll         10:01 00:00:00  0.0  0.0  1856   244   572 - /sbin/portmap
2675  2675    1 TS  root     ?          0  23   0 * Ss   select       10:02 00:00:00  0.0  0.0  5772   520  1532 - /usr/sbin/sshd -o PidFile=/var/
run/sshd.init.pid
2680  2680    1 TS  root     ttyS0      0  21   0 * Ss+  read_chan    10:02 00:00:00  0.0  0.0  2008   244   656 - /sbin/mingetty --noclear
/dev/ttyS0 dumb
2737  2737    1 TS  root     ?          0  24   0 * Ss   select       10:30 00:00:00  0.0  0.0  9512   868  3228 - sshd: root@pts/0
2739  2739    1 TS  root     pts/0      0  24   0 * Ss   wait4        10:30 00:00:00  0.0  0.0  5140   824  2668 - -bash
2766  2766    1 TS  root     ?          0  23   0 * Ss   select       10:35 00:00:00  0.0  0.0  9364   720  3136 - sshd: root@pts/1
2768  2768    1 TS  root     pts/1      0  23   0 * Ss   wait4        10:35 00:00:00  0.0  0.0  5140   824  2680 - -bash
2833  2833    1 TS  root     ?          0  23   0 * Ss   select       10:38 00:00:00  0.0  0.0  9512   868  3152 - sshd: root@pts/2
2835  2835    1 TS  root     pts/2      0  23   0 * Ss+  read_chan    10:38 00:00:00  0.0  0.0  5140   824  2644 - -bash
3437  3437    1 TS  root     pts/1      0  23   0 * S+   wait4        11:39 00:00:00  0.0  0.0  1816   248   644 - dbench 3
3438  3438    1 TS  root     pts/1      0  20   0 0 R+   -            11:39 00:00:24 33.1  0.0  1820   252   604 - dbench 3
3439  3439    1 TS  root     pts/1      0  20   0 0 R+   -            11:39 00:00:23 32.8  0.0  1820   252   604 - dbench 3
3440  3440    1 TS  root     pts/1      0  20   0 0 R+   -            11:39 00:00:23 31.8  0.0  1820   252   604 - dbench 3
3461  3461    1 TS  root     pts/0      0  22   0 0 R+   -            11:40 00:00:00  0.0  0.0  2688   588   976 - ps -eo
pid,tid,nlwp,policy,user,tname,ni,pri,psr,sgi_p,stat,wchan:12,start_time,time,pcpu,pmem,vsize,size,rss,shar
```

- **Hints**
  - Do not specify blanks inside the -o format string
  - Many more options available

Agenda

- **Objectives**
- **Tools**
  - System
    - vmstat
    - sadc/sar
  - Disk
    - iostat
    - DASD statistics
    - SCSI statistics
  - Network
    - netstat
  - Processes
    - top
    - ps
- **Assuming we have a problem …**
  - **.. somewhen**
  - **.. now**
- **Summary**

We have the problem somewhen - Strategy

- **Requires permanent monitoring**
  - Use sadc with a suitable interval, e.g. 1 – 5 minutes
  - Because of the flattening effects of averaging, limits for critical ranges are much lower!
  - Try to identify time patterns
→ use the 'at' command and
→ the 'counts' parameter from the monitoring tool (sadc, iostat) limits the amount of samples gathered, reduces the impact and the amount of data
  - Allows to gather data with a much higher granularity during a certain period
  - Follow the advices on page 'We have the problem now'

We have the problem now - Analysis

- **Check CPU utilization**
  - Start with vmstat 1 for a fast view or sadc/sar for a comprehensive data gathering (especially if the problem is only temporary)
  - user + system + nice >95%?
    - Is the system doing the right things? Continue with top
      → Do the right processes use the CPU?
    - Is kswapd using significant amount of CPU? Continue with top or ps
      → memory is constrained
    - Does the system use Hipersockets? Continue with netstat -s on sender and receiver (or sadc/sar)
      → Hipersockets are CPU-driven
    - Add CPUs! Or analyze the application used.
  - High steal times?
    - Under z/VM:
      - monitor z/VM using the z/VM performance tool kit
      - high steal times on one CPU might be caused by VM activity for virtual interfaces from the guest, e.g. VSWITCH
    - In LPAR: monitor LPAR activity with the SE

We have the problem now - Analysis

- **Check CPU utilization (cont)**
  - I/O wait times
    - counts as idle!
    - indicates disk I/O contention. Continue with 'check disk utilization'
  - Significant idle times?
    - system is waiting. Continue with 'system is waiting'

- **Check disk utilization**
  - sadc/sar or iostat → identify concerned disks
  - Contention on DASD devices → continue with DASD statistics
  - Contention on FCP devices → continue with FCP statistics

- **System is waiting**
  - If no contention on disk devices,
  - Check for errors/retransmits using netstat -s or sadc/sar
  - Analyze application for locking scenarios

Agenda

- Objectives
- Tools
  - System
    - vmstat
    - sadc/sar
  - Disk
    - iostat
    - DASD statistics
    - SCSI statistics
  - Network
    - netstat
  - Processes
    - top
    - ps
- Assuming we have a problem …
  - .. somewhen
  - .. now
- **Summary**

Summary

- **I showed various tools providing performance data for**
  - The overall system
  - Disk I/O
  - Network I/O
  - Processes
- **The recommended tool for general purpose is sadc/sar!**
  - Suitable for permanent system monitoring.
  - The other tools are for a detailed analysis of specific problems
- **Stay current with updates on your preferred monitoring tool.**
  - Might provide more accurate values or more information.

Related Topics at 2009 System z Expo

- **Problem Determination with Linux on System z**

zLG05	Thursday	1:00 PM

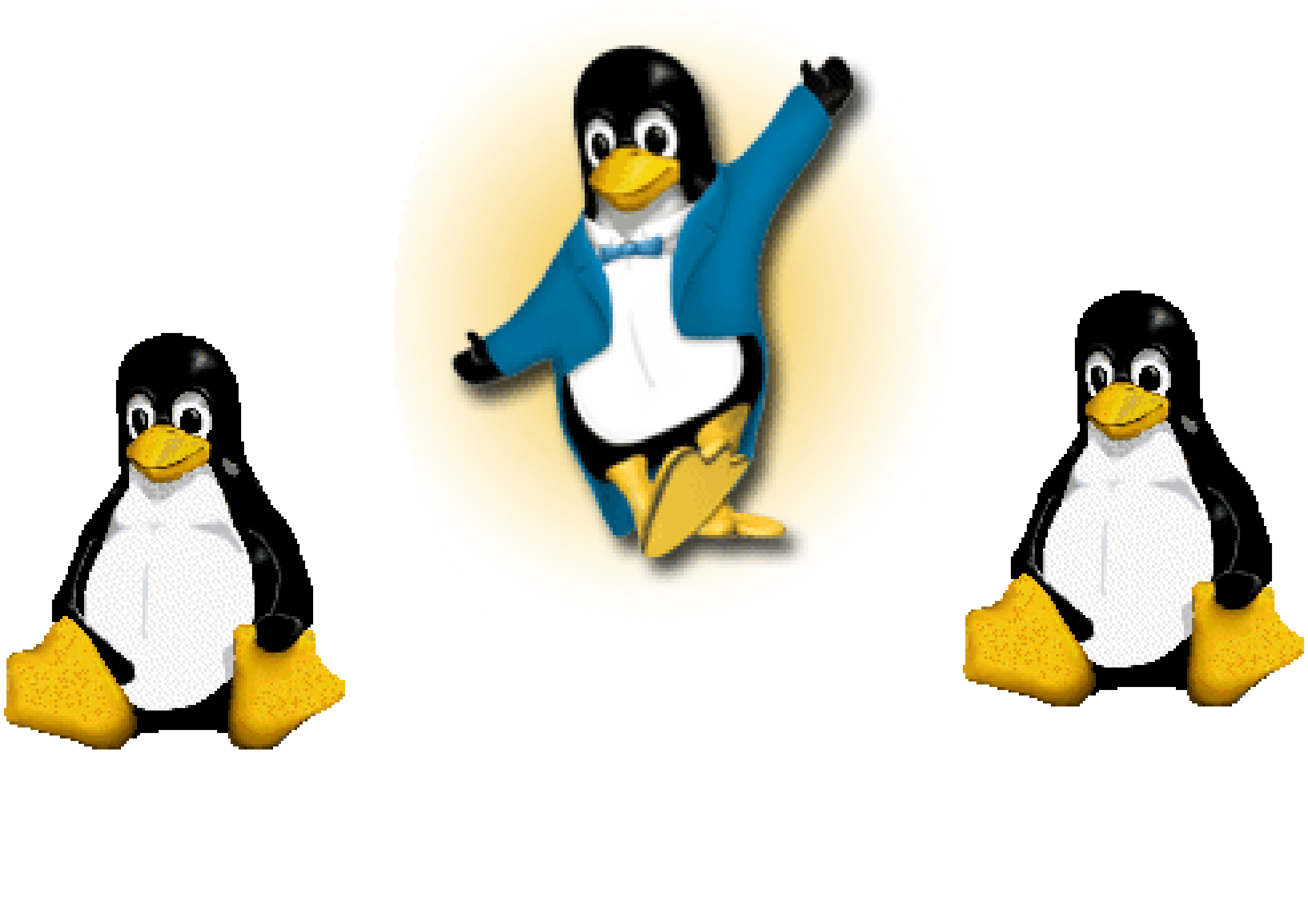
- **Performance Tuning and Monitoring: DB2 for Linux, Unix and Windows (LUW) for Linux**

zLA08	Monday	4:10 PM
zLA08	Wednesday	4:10 PM

Visit us !

- **Linux on System z: Tuning Hints & Tips**
  - http://www.ibm.com/developerworks/linux/linux390/perf/
- **Linux-VM Performance Website:**
  - http://www.vm.ibm.com/perf/tips/linuxper.html
- **IBM Redbooks**
  - http://www.redbooks.ibm.com/
- **IBM Techdocs**
  - http://www.ibm.com/support/techdocs/atsmastr.nsf/Web/Techdocs

Questions



Backup

top (config file)

- **Sample .toprc**
  - used for the output on the next slide

```
RCfile for "top with windows"           # shameless braggin'
Id:a, Mode_altscr=0, Mode_irixps=1, Delay_time=1.000, Curwin=0
Def     fieldscur=AEHIOQTWKNMbcdfgJPlrSuvYzX
        winflags=64825, sortindx=10, maxtasks=0
        summclr=1, msgsclr=1, headclr=3, taskclr=1
Job     fieldscur=ABcefgjlrstuvyzMKNHIWOPQDX
        winflags=62777, sortindx=0, maxtasks=0
        summclr=6, msgsclr=6, headclr=7, taskclr=6
Mem     fieldscur=ANOPQRSTUVbcdefgjlmyzWHIKX
        winflags=62777, sortindx=13, maxtasks=0
        summclr=5, msgsclr=5, headclr=4, taskclr=5
Usr     fieldscur=ABDECGfhijlopqrstuvyzMKNWX
        winflags=62777, sortindx=4, maxtasks=0
        summclr=3, msgsclr=3, headclr=2, taskclr=3
```

sysstat tools

- **provides a bunch of tools:**
  - /usr/bin/iostat
  - /usr/bin/mpstat
  - /usr/bin/pidstat
  - /usr/bin/sadf
  - /usr/bin/sar
  - /usr/lib64/sa
  - /usr/lib64/sa/sa1
  - /usr/lib64/sa/sa2
  - /usr/lib64/sa/sadc
  - /usr/sbin/rcsysstat