Monitoring and Understanding Performance on Linux for zSeries & S/390

SHARE Technical Conference
August 18-23, 2002, San Francisco
Session 9301

Oliver Benke

IBM Böblingen Lab
Schönaicher Str. 220
D-71032 Böblingen
Germany

Email: benke@de.ibm.com

Trademarks

IBM @server zSeries

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

IBM*
the IBM logo*
OS/390*
Parallel Sysplex*
MVS
z/OS
CICS*
DB2*
e-business logo*
IMS
Language Environment*
RACF*
RMF
zSeries

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Lotus, Notes, and Domino are trademarks or registered trademarks of Lotus Development Corporation
LINUX is a registered trademark of Linus Torvalds
Penguin (Tux) compliments of Larry Ewing
Tivoli is a trademark of Tivoli Systems Inc.
Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries
UNIX is a registered trademark of The Open Group in the United States and other countries.
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

* 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. Actu 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 witho 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.

Why Performance Measurement?

- Performance Tuning, Problem drill-down, Online Performance Monitoring & Analysis
- Long-term Performance Monitoring
- Benchmarking, Sizing
  IBM Representatives and IBM Business Partners have access to SIZE390 for sizing
- Program development
  - Tracing and Profiling tools for applications or even the operating system kernel itself
- Workload Management
  - Service Level Agreements

What can be tuned?

- CPU
- I/O
  - DASD
  - Network
  - Channels
- Memory

Forget about IDLE resources !

- A mainframe can drive most resources to their capacity limits without penalties to critical business workloads
- If one virtual server (z/OS, Linux) does not need some resources (Channel bandwith, CPU, ...), the hardware gives it to another image ready to run
- It is like a second level of scheduling - multi-tasking in another dimension

Linux for zSeries: Major Benefits

- Virtual Server; dynamically create and destruct Linux server using z/VM.
- Idle time of one operating system can be used by another operating system, so you are wasting less resources.
- HiperSockets ("Network in a Box", "The Network is in the Computer"): memory speed networking to connect Linux images with other Linux images or z/OS images, leading to a client-server network in a box.

Recent z/Linux enhancements regarding performance

- SCSI using FCP
  - no more translation from block format to ECKD format and back any longer
- PCICA crypto support
- OSA enhancements
  - SNMP support to retrieve management data
  - information like PCI Bus and CHPID Processor Utilization, inbound/outbound transfer rates, error rates
  - integrated in ucd-snmp

LPAR

- A mainframe can be logically partitioned
- Based on LPAR weights and on the number of logical processors, the LPAR Hypervisor allocates CPU resources to the different logical partitions
- If one LPAR has nothing to do, LPAR Hypervisor gives control to another LPAR
- z/OS IRD can influence the LPAR weights

z/OS IRD

- Available with z/OS V1R2
- Linux can be part of a z/OS LPAR cluster (in contrast to OS/390)
- For Linux, only the CPU management is working
  - Adjust number of logical CPUs to reduce LPAR overhead
  - Adjust LPAR weighting factors
  - No Dynamic Channel Management (DCM) or Channel Subsystem Priority Queuing
- Does not work for IFLs

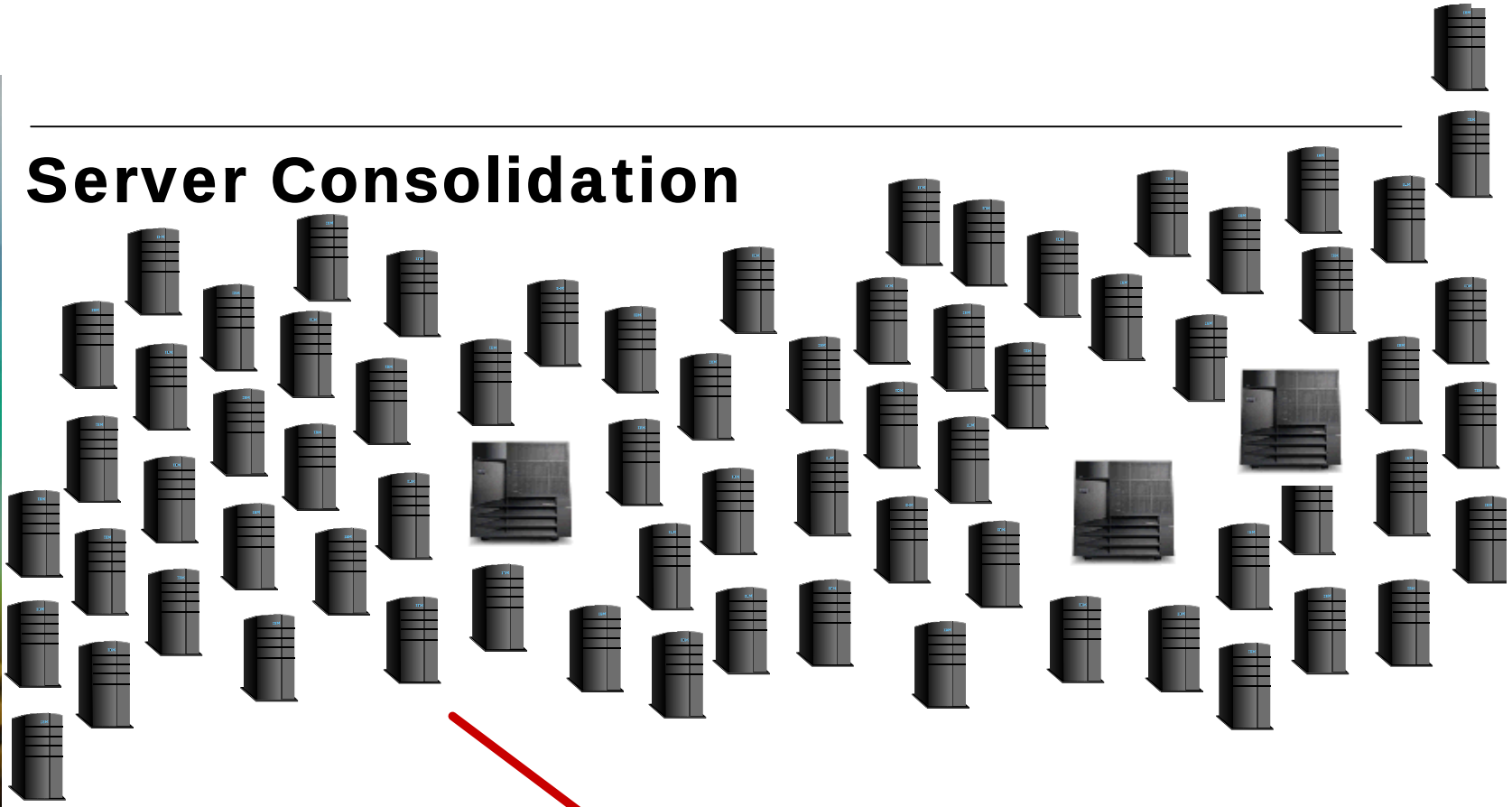
z/VM

- Second level of virtualization (or first level if machine runs in Basic mode)
- Different operating system guests can share memory, CPU and I/O resources if running under z/VM
- Especially for V=R/F guests, the performance can be fairly well
- Very flexible
- Mature systems management tools
- For high end server application, think about how much memory the application needs

z/VM V=F,R guests

- The preferred V=R guest can use hardware facilities to execute faster. V=R guests are faster than V=F guests.
- Up to five V=F and one V=R guests (if not running z/VM under LPAR)
- All V=R,F guests must reside below the 2 GB line (z/VM 4.2)
- For each QDIO device, z/VM allocates a shadow queue below the 2 GB line (z/VM 4.2)
- QDIO is most efficient if running under LPAR

Server Consolidation



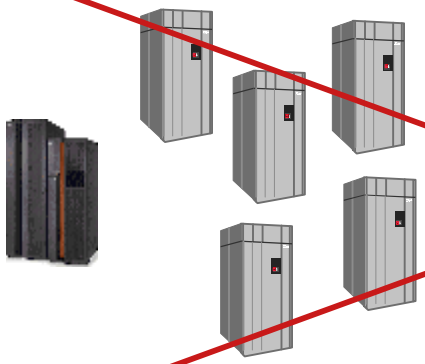
zSeries, z/VM

"Server Farm in a Box"

Horizontal Server Consolidation

- Consolidate lots of under-utilized servers on one box
  - Under-utilized web servers, mail servers, file servers, print servers
  - ISPs, ASPs or universities can give Linux servers with root access to their customers
- For this, you definitely need z/VM
  - Currently, LPAR is limited to 15 logical partitions per box
  - Lots of Linux images can be managed with z/VM systems management facilities

High end client/server



Network

(Cisco, etc.)

SAP on high end SMP servers

DB2 on zSeries
(z/OS)

SAP on zSeries
(Linux for zSeries)

DB2 on zSeries
(z/OS)

HiperSockets

Vertical Server Consolidation

- Consolidate some high-end SMP servers on Linux for zSeries
  - WebSphere
  - SAP R/3 Application Server (together with z/OS DB2 Database Server in separate LPAR on same physical box, connected with HiperSockets)
- Probably an LPAR game
  - Faster
  - Only few images needed
  - A Linux partition can be part of a z/OS LPAR cluster, so z/OS IRD can adjust LPAR weights
- Sure, you can combine horizontal and vertical server consolidation, perhaps 4 high-end virtual servers under LPAR and 1 VM LPAR for test systems and low-end server applications

Scalability of the Linux kernel

- On zSeries, Linux kernel 2.4 scales really well; you can efficiently burn all the power of a full-blown z900 with very few Linux and/or z/OS images
- Linux kernel 2.2 does not scale well, even on zSeries hardware
- If you'd like to exploit Linux kernel 2.2, let z/VM do the scalability work for you: define lots of Linux operating systems scheduled and managed by z/VM

Scalability Limits and Scheduling Overhead

PR/SM LPAR Hypervisor

z/VM Hypervisor

Linux Scheduler

Application Middleware

Linux Mail

Linux Apache

Linux Test

z/OS Test

Linux SAP R/3

z/OS DB2

z/VM

LPAR 1

LPAR 2

LPAR 3

IBM S/390 or zSeries Hardware

Some performance related UNIX and Linux concepts

Load average

- Average number of processes in the "run" queue
- A runnable process is one that is ready to consume CPU resources right now; a process waiting for I/O is *not* runnable
- A high load average value (in relation to the number of physical processors) is an indicator for latent demand for CPU

CPU performance data reported by Linux

- You can use it for accounting if running Linux under LPAR (although LPAR CPU data obtained by a hardware interface is more precise)
- If running under z/VM, data reported by Linux can become pretty incorrect. Linux will not notice if z/VM gives all CPU resources to some other guest!

Linux Page Cache

- The page cache contains pages of memory mapped files (page I/O related syscalls like generic_file_read)
- It usually contains unneccessary files which can be freed, and the kernel actually discards those pages if it runs out of free memory
- On Intel Linux or for Linux running in a LPAR, the page cache is always useful as the memory would be wasted otherwise. But running under z/VM, it may cost valuable z/VM memory, leading to z/VM page activity.

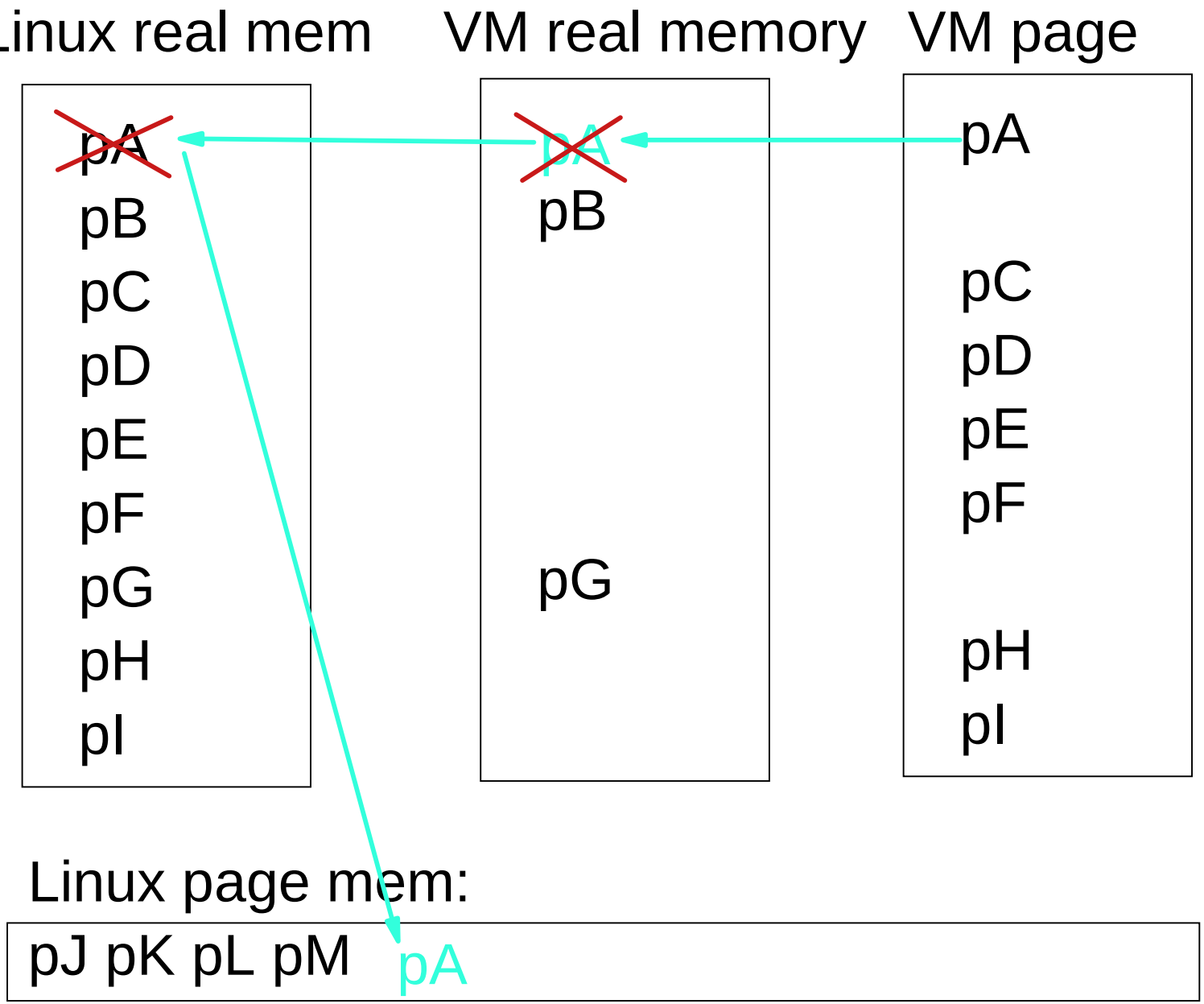
Linux Buffer Cache

- A similar important Linux kernel data structure is the so-called Buffer Cache which contains pages read from or written to physical devices like DASDs (block I/O related syscalls)
- Linux rarely has free space; everything not used is allocated for Page Cache and Buffer Cache, so even if Linux does not really need it all, it uses all available memory up to the last few percent.

Double Paging

- Possible for Linux under z/VM, running V=V mode (not possible for V=R,F)
- Assume page A is marked "swapped in" by Linux but paged out by z/VM; now, if Linux would like to page this page A out, first z/VM needs to page it in in order to enable Linux to page it out
- If Linux wants to page out a whole bunch of pages which were paged out previously by z/VM (not an unrealistic scenario), the system has to do a whole lot of work
- z/VM PAGEX support: Linux can give up a time slice if blocked on I/O due to double paging activity

Double Paging: Illustration



Linux swap to VM virtual disc

- One solution would be to give Linux less memory and allocate a z/VM virtual disk for Linux swap space
- As on other platforms, avoid paging if possible, as it kills performance; virtualization is great, but has its drawbacks especially for memory (so dedicated LPAR memory can actually be an advantage for some high-end applications)
- You can also use XPRAM (z/VM expanded storage) or a z/VM minidisk for Linux paging
- More details on how to efficiently use memory under z/VM are described in the ISP/ASP redbook (SG24-6299)

Linux Process memory: basic terms

- **SIZE: size of the address space seen by the process, virtual size**
- **RSS: Resident Set Size**
  actual amount of memory that the process is using in RAM
- **SHARE:**
  portion of the RSS that is shared with other processes, such as shared libraries

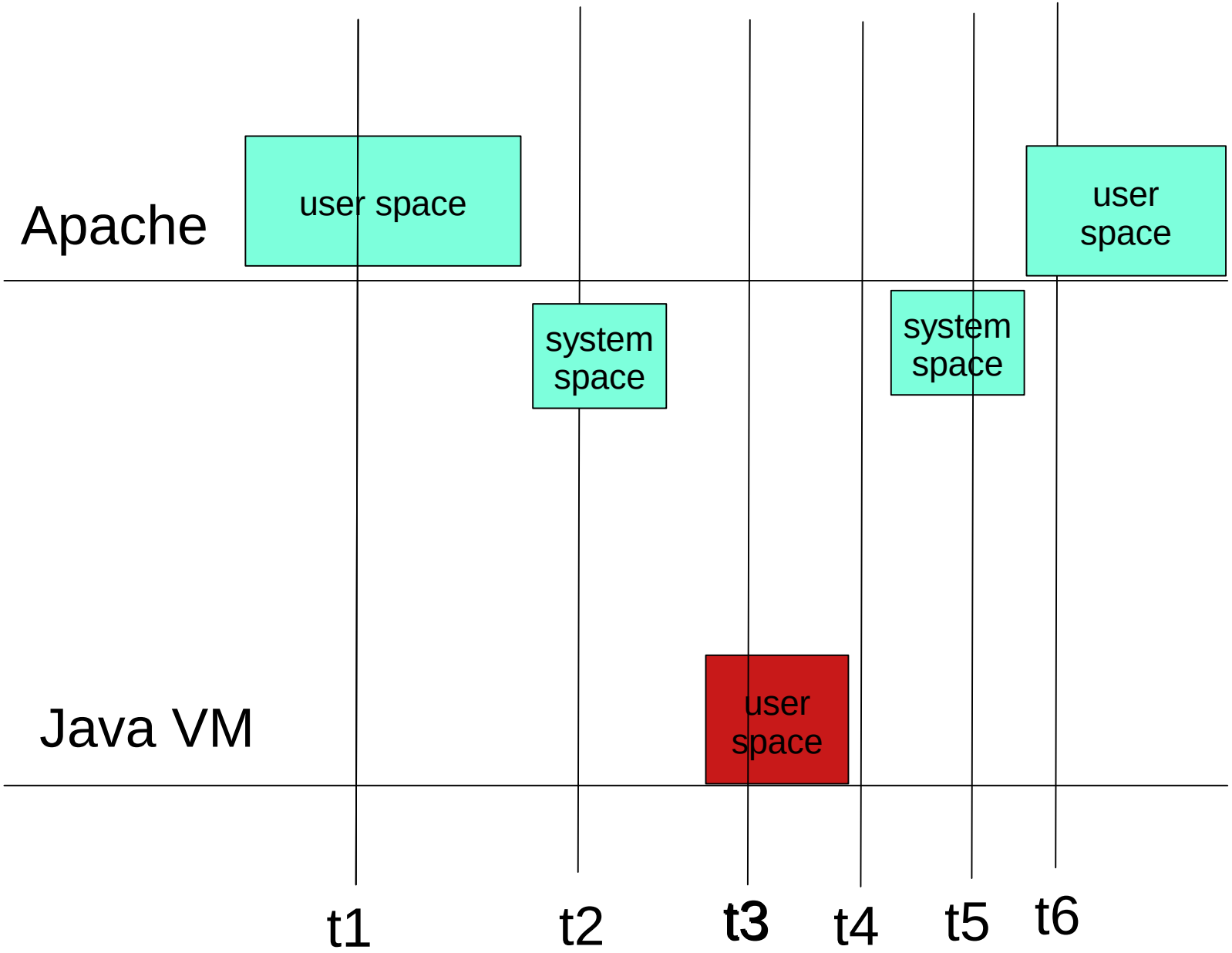
Processes and Threads

- In contrast to some commercial UNIX implementations, in Linux a thread is pretty much the same as a process, it just does not have an own address space
  - For the scheduler, a posix thread is almost like a process
  - In the /proc file system (see below), there is no difference between a process and a thread; so if you are monitoring your system, your threads might appear like processes on first sight
- As an alternative, user-space thread libraries are available today
- Outlook: Next Generation POSIX Threading
  - make Linux strong and competitive even for lots of threads
  - Support integrated in Linux 2.5.17 kernel, high probability it will become standard in future
  - see http://www-124.ibm.com/pthreads/

jiffies

- Derived from PC timer interrupt (100 Hz)
- Every time a timer interrupt occurs (100 times per second), the jiffies variable is incremented by one, that is one tick
- CPU usage is accounted on in units of jiffies
- If a process is running at the time the timer interrupt occurs, its CPU usage counter is incremented
- Accuracy (10 msec) might be enhanced in future Linux versions
- Jiffie-based performance measurement is wrong if running under z/VM

When does the "jiffie event" take place?



On demand timer patch

- For an idle Linux image running under z/VM, CPU resources are used up mainly for generating the jiffi
- If you apply this patch, jiffies are generated on dema
- However, the switch between user and kernel mode slightly slower; therefore, if running under LPAR, the system gets slower if this patch is applied
- see *http://www10.software.ibm.com/developerworks/opensou linux390/current2_4_17-may2002.shtml#timer20020531*

Process priorities

- **Process priority can be changed with *nice/renice* commands**
- **Highest priority is -20, lowest priority is 19**
- **In addition, each process has a *dynamic priority* in Linux; a heavy CPU consumer has a worse dynamic priority than a process mainly doing I/O, giving up the CPU before the end of the time slot**
- **In Linux 2.5, the scheduler will be replaced by Ingo Molnars O(1) scheduler**

System log

- Linux default: /var/log/messages
- Most applications are writing their error messages to /var/log/messages
- You should monitor the system log to find out if something went really wrong.

The /proc filesystem

- Virtual file system
- One of the interfaces between kernel space and user space; if the user gives a command like

  ```
  cat /proc/stat
  ```

  the kernel executes some function to generate the needed "virtual file"
- Parts of the /proc filesystem are human readable
- Most performance measurement tools for Linux are based on /proc file system

/proc/dasd/statistics

- Only available in Linux for zSeries, kernel version 2.4
- Used in rmfpms to calculate the following metrics:
  - dasd io average response time per request (in msec)
  - dasd io average response time per sector (in msec)
  - dasd io requests per second

/proc/dasd/statistics (continued)

```
cat /proc/dasd/statistics
3156192 dasd I/O requests
__<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 6164    0    0    0    0    0    0    0    0    0    0    0    0    0    0
Histogram of I/O times
   0    0    0    0    0    0    0    0  736  628  719  952 1346 1310  448   15
   4    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
Histogram of I/O times per sector
   0    0    0    0    0  736  628  719  952 1346 1310  448   15    4    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
Histogram of I/O time till ssch
 710  218  150   28   22   94   63    8  318  374  457  794 1271 1245  384   14
   4    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    0 3505 2072  414  147   19    2    0    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 per sector
   0    0    0    0    0 3505 2072  414  147   19    2    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 time between irq and end
   3 1199  959 3817  132   12    7    4    3    5    6    6    3    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

/proc/stat

```
$ > cat /proc/stat
cpu  58975 2084 34136 158972653
cpu0 7792 1064 15454 26486998
cpu1 32631 993 15340 26462344
cpu2 17308 27 2320 26491653
cpu3 1240 0 614 26509454
cpu4 4 0 300 26511004
cpu5 0 0 108 26511200
page 188768 6603424
swap 0 0
intr 0
disk_io:
ctxt 1781988
btime 1011713660
processes 9867
```

/proc/slabinfo

- **statistics for frequently used kernel objects**

```
cat /proc/slabinfo
slabinfo - version: 1.1 (SMP)
kmem_cache            68     68    232    4    4    1 :  252  126
nfs_read_data          0      0    384    0    0    1 :    0   62
nfs_write_data         0      0    400    0    0    1 :    0   62
nfs_page               0      0     80    0    0    1 :    0  126
tcp_tw_bucket          1     40     96    1    1    1 :    0  126
tcp_bind_bucket      136    203     16    1    1    1 :    0  126
tcp_open_request      59     59     64    1    1    1 :    0  126
inet_peer_cache        0      0     48    0    0    1 :    0  126
ip_fib_hash            8    203     16    1    1    1 :    0  126
ip_dst_cache          50     72    160    3    3    1 :    0  126
arp_cache              1     70    112    1    2    1 :    0  126
blkdev_requests      768    800     96   20   20    1 :    0  126
dnotify cache          0      0     20    0    0    1 :    0  126
file lock cache      173    240     96    5    6    1 :    0  126
fasync cache           0      0     16    0    0    1 :    0  126
uid_cache              3    113     32    1    1    1 :  252  126
skbuff_head_cache    132    405    144   14   15    1 :  252  126
sock                  85     90    816   17   18    1 :  124   62
inode_cache        28776  30296    464 3787 3787    1 :  124   62
bdev_cache             3     78     48    1    1    1 :  252  126
sigqueue             176    203    132    7    7    1 :  252  126
kiobuf                 0      0    128    0    0    1 :  252  126
ccwcache-4096          0      0   4096    0    0    1 :   60   30
ccwcache-2048          4     10   2048    2    5    1 :   60   30
ccwcache-1024        118    128   1024   30   32    1 :  124   62
```

Trace facilities (Kernel patches)

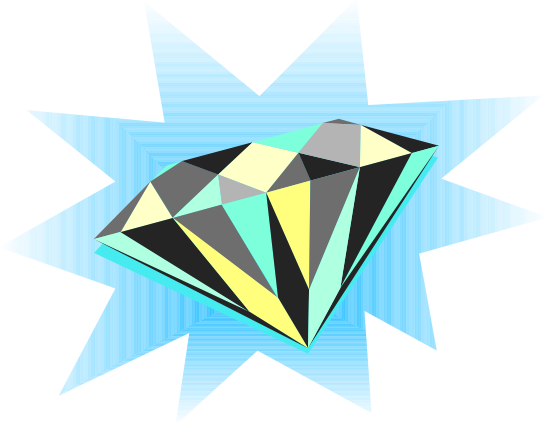
- Take note on what was actually done directly in the kernel; generate trace data for some system activities
- Advantages:
  - High flexibility
  - Possibility to provide very accurate and efficient tools
- Drawbacks:
  - Has to be adopted and enabled by distributors (SuSE, RedHat); otherwise, those installing the patch are losing their service contract
- Example projects:
  - IBM dprobes
    http://www.ibm.com/developerworks/oss/linux/projects/dprobes/
  - LTT (yes, it supports S/390)
    http://www.opersys.com/LTT/

Alternative: Cycle Gatherer

- Cycle Gatherer: "Every 10 msec, make note on which processes are currently running on each of the CPUs."
- Trace Facility: "Every time the scheduler decides to switch to another process, make note on it."

Classical UNIX tools for monitoring

- sysstat package (sar, sadc)
- top
- ps
- vmstat
- free
- strace
- ...



top

- **Nice option: "f - u - enter" to see what the process is waiting for**

gfree18.boeblingen.de.ibm.com - PuTTY

```
  8:33pm  up  3:58,  3 users,  load average: 1.74, 0.66, 0.24
50 processes: 47 sleeping, 3 running, 0 zombie, 0 stopped
CPU0 states: 58.0% user,  3.1% system,  0.0% nice, 38.2% idle
CPU1 states: 99.3% user,  0.2% system,  0.0% nice,  0.0% idle
CPU2 states:  1.2% user,  0.4% system,  0.0% nice, 97.3% idle
CPU3 states: 37.3% user,  0.2% system,  0.0% nice, 61.4% idle
CPU4 states:  0.0% user,  0.0% system,  0.0% nice, 100.0% idle
CPU5 states:  0.0% user,  0.0% system,  0.0% nice, 100.0% idle
Mem:   122772K av,  112752K used,   10020K free,       0K shrd,   40684K buff
Swap:       0K av,       0K used,       0K free                   23996K cached

  PID USER     PRI  NI  SIZE  RSS SHARE WCHAN     STAT %CPU %MEM   TIME COMMAN
 1170 root      19   0   400  400   332           R    99.9  0.3   2:14 load
 1339 root      15   0 14844  14M  2236           R    48.1 12.0   0:02 cc1plus
 1203 root      13   0  1560 1560  1280           R     1.3  1.2   0:01 top
    7 root       9   0     0    0     0 kupdate   SW    0.5  0.0   0:01 kupdate
 1337 root       9   0   612  612   508 wait4      S     0.1  0.4   0:00 g++
    1 root       9   0   660  660   572 do_select  S     0.0  0.5   0:03 init
    2 root       9   0     0    0     0 down_inte SW    0.0  0.0   0:00 kmcheck
    3 root       9   0     0    0     0 context_t SW    0.0  0.0   0:00 keventd
    4 root       9   0     0    0     0 kswapd    SW    0.0  0.0   0:00 kswapd
    5 root       9   0     0    0     0 kreclaimd SW    0.0  0.0   0:00 kreclai
    6 root       9   0     0    0     0 bdflush   SW    0.0  0.0   0:00 bdflush
   63 root      -1 -20     0    0     0 end       SW<   0.0  0.0   0:00 mdrecov
```

ps - report process status

- **common set of parameters:**
  `ps aux`
- **single out a user:**
  `ps u --User apache`

```
bash-2.05# ps aux|more
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1   1536   160 ?        S    Jan22   0:12 init
root         2  0.0  0.0      0     0 ?        SW   Jan22   0:00 [kmcheck]
root         3  0.0  0.0      0     0 ?        SW   Jan22   0:00 [keventd]
root         4  0.0  0.0      0     0 ?        SW   Jan22   0:22 [kswapd]
root         5  0.0  0.0      0     0 ?        SW   Jan22   0:00 [kreclaimd]
root         6  0.0  0.0      0     0 ?        SW   Jan22   0:00 [bdflush]
root         7  0.0  0.0      0     0 ?        SW   Jan22   1:05 [kupdated]
root        63  0.0  0.0      0     0 ?        SW<  Jan22   0:00 [mdrecoveryd]
root       248  0.0  0.0      0     0 ?        SW   Jan22   0:00 [keventd]
root       310  0.0  0.2   1732   292 ?        S    Jan22   0:12 syslogd -m 0
root       315  0.0  0.6   2088   768 ?        S    Jan22   0:00 klogd -2
rpc        325  0.0  0.0   1732   120 ?        S    Jan22   0:00 portmap
rpcuser    338  0.0  0.1   1844   140 ?        S    Jan22   0:00 rpc.statd
root       385  0.0  0.6   3180   800 ?        S    Jan22   0:00 /usr/sbin/sshd
root       401  0.0  0.4   2876   512 ?        S    Jan22   0:00 xinetd
```

The Process forest

- **See process together with their parents or children with the `pstree` command**

```
[root@lnxbenk1 /root]# pstree
init-+-apachegat
     |-bdflush
     |-clustergat
     |-crond
     |-dasdgat
     |-filegat
     |-find
     |-gengat
     |-gpmddsrv---gpmddsrv---2*[gpmddsrv]
     |-httpd---5*[httpd]
     |-inetd---in.telnetd---login---bash---xterm---bash---pstree
     |-keventd
     |-klogd
     |-kmcheck
```

time

- **Find out how many CPU resources a command is taking**
- **Example:**

```
$ > time make dep
...
72.52user 8.87system 2:03.72elapsed 65%CPU
(0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (131158major+106391minor)
pagefaults 0swaps
$ >
```

elapsed: **real time elapse**
user: **time this command (and its children) have spent in user space**
sys: **time spent in kernel space**

"netstat -s" for detailed network statistiscs

```
$ > netstat -s
Ip:
    3608 total packets receivedgs
    0 forwardedonnection openings
    0 incoming packets discarded
    3587 incoming packets delivered
    4080 requests sent outhed
Icmp:493 segments received
    4 ICMP messages received
    0 input ICMP message failed.
    ICMP input histogram:ed.
        echo requests: 4
    4 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:ort received.
        echo replies: 4rors
Tcp:112 packets sent
    7 active connections openings
    0 passive connection openings
    0 failed connection attempts
    0 connection resets received
    3 connections established
    3493 segments received
    3964 segments send out
    10 segments retransmited
    0 bad segments received.
    13 resets sent
Udp:
    111 packets received
    0 packets to unknown port received.
    0 packet receive errors
    112 packets sent
TcpExt:
```

```
TcpExt:
    ArpFilter: 0
    TW: 6
    TWRecycled: 0
    TWKilled: 0
    PAWSPassive: 0
    PAWSActive: 0
    PAWSEstab: 0
    DelayedACKs: 71
    DelayedACKLocked: 0
    DelayedACKLost: 0
    ListenOverflows: 0
    ListenDrops: 0
    TCPPrequeued: 114
    TCPDirectCopyFromBacklog: 0
    TCPDirectCopyFromPrequeue: 3585
    TCPPrequeueDropped: 0
    TCPHPHits: 312
    TCPHPHitsToUser: 41
    TCPPureAcks: 1668
    TCPHPAcks: 283
    TCPRenoRecovery: 0
    TCPSackRecovery: 0
    TCPSACKReneging: 0
    TCPFACKReorder: 0
    TCPSACKReorder: 0
    TCPRenoReorder: 0
    TCPTSReorder: 0
    TCPFullUndo: 0
    TCPPartialUndo: 0
    TCPDSACKUndo: 0
    TCPLossUndo: 3
    TCPLoss: 0
```

free

- **Give free memory; important is the second line, as buffer/cache memory is not really needed by Linux**

```
[root@lnxbenk1 /root]# free
             total       used       free     shared    buffers     cached
Mem:        118092     116872       1220          0       4148      66124
-/+ buffers/cache:      46600      71492
Swap:            0          0          0
```

vmstat

- **Gives information about memory, swap usage, I/O activity and CPU usage**

```
bash-2.05# vmstat 1 10
   procs                  memory    swap          io     system          cpu
 r  b  w   swpd   free   buff  cache  si  so    bi    bo   in    cs  us  sy  id
 1  1  0      0  18608   4424  51516   0   0     0     4    0     1   0   0   4
 0  1  0      0  17884   4912  51516   0   0   488     0    0   711   0   6  93
 0  1  0      0  17224   5388  51516   0   0   476     0    0   512   0   9  90
 0  1  0      0  16480   5800  51516   0   0   412  1196    0   447   1   7  93
 0  1  0      0  14672   7016  51516   0   0  1220     0    0  1268   1  12  87
 0  0  0      0  13832   7504  51516   0   0   484     0    0   571   1   3  97
 0  1  0      0  12848   8080  51516   0   0   576     0    0   628   1   7  92
 0  1  0      0  12228   8456  51544   0   0   376     0    0   480   2  14  84
 0  1  0      0  11508   8932  51544   0   0   476  1260    0   530   0   6  94
 0  1  0      0  10540   9568  51544   0   0   636     0    0   674   1   6  93
```

strace

- **Example:**

```
strace -p 6148
```

  to trace all system calls by process with ID 6148

- **Usage:**
  - **As you can see what the process is doing, you may be able to tune it**
  - **If you suspect a process to loop, you may check using strace; if the process consumes CPU but does not initiate any system call, it may be looping**

Example: "strace ping <hostname>"

```
bash-2.05# strace ping lnxbenk1
execve("/bin/ping", ["ping", "lnxbenk1"], [/* 23 vars */]) = 0
uname({sys="Linux", node="gfree18", ...}) = 0
brk(0)                                  = 0x80017bd8
open("/etc/ld.so.preload", O_RDONLY)    = -1 ENOENT (No such file or
directory)
open("/etc/ld.so.cache", O_RDONLY)      = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=31761, ...}) = 0
mmap(NULL, 31761, PROT_READ, MAP_PRIVATE, 3, 0) = 0x2000001c000
close(3)                                = 0
open("/lib/libresolv.so.2", O_RDONLY)   = 3
read(3, "\177ELF\2\2\1\0\0\0\0\0\0\0\0\0\3\0\26\0\0\0\1\0\0\0"..., 1024) =
1024
fstat(3, {st_mode=S_IFREG|0755, st_size=95105, ...}) = 0
mmap(NULL, 92712, PROT_READ|PROT_EXEC, MAP_PRIVATE, 3, 0) =
0x20000024000
mprotect(0x20000037000, 14888, PROT_NONE) = 0
mmap(0x20000037000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED, 3, 0x12000) = 0x20000037000
mmap(0x20000039000, 6696, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x20000039000
close(3)                                = 0
```

file system usage

- **df, du**

```
bash-2.05# df
Filesystem           1k-blocks      Used Available Use%
Mounted on
/dev/dasd/6148/part1   2366164   1040288   1205680  47% /
bash-2.05# du | more
28      ./lost+found
6332    ./bin
32448   ./boot
0       ./dev/pty
0       ./dev/pts
0       ./dev/3270
0       ./dev/rd
0       ./dev/dasd/6148
0       ./dev/dasd/6149
0       ./dev/dasd
0       ./dev/discs
0       ./dev/loop
0       ./dev/md
0       ./dev
20      ./etc/X11/applnk/Utilities
```

inode utilization

- In UNIX, an inode is a structure containing meta data about files and directories.
- The number of inodes is limited, can be changed at filesystem creation time.
- If you are running out of inodes, you can not store anything more on this filesystem.
- Check with "*df -i*" command:

```
benke@tux390:/projects/home/benke > df -i
Filesystem            Inodes   IUsed   IFree IUse% Mounted on
/dev/dasdb1           601312   59034  542278   10% /
/dev/dasdc1           300960   63886  237074   21% /projects
```

BSD Accounting

- Writes one accounting record per terminated process or thread (as threads are something like processes in Linux...)
- Currently, SuSE decided to disable this feature for performance reasons
- Information provided:
  - user ID, group ID, process name
  - CPU resource consumption
  - average memory usage, page faults, swap activity
- An alternative to accounting Linux "from the inside" is accounting it "from the outside", with the aid of z/VM or z/OS performance tools

sysstat package

- Contains sar and sadc, long term data collector
- Normally, it collects data about overall system activity like CPU usage, swapping; no data about processes
- start with

  ```
  $ > sadc 60 /var/log/sa/sa25 &
  ```

  to let it generate one report every 60 seconds and write it in binary format to */var/log/sa/sa25*
- http://freshmeat.net/projects/sysstat/

Mainframe-related Tools

- Some zSeries performance data is currently only available in z/VM or z/OS performance monitors
  - Coupling facility activity
  - LPAR partition data, VM CPU activity
  - Channel utilization (including OSA cards, HiperSockets)
- Tools like z/OS RMF PM and z/VM FCON can display Linux performance data together with z/OS or z/VM performance data

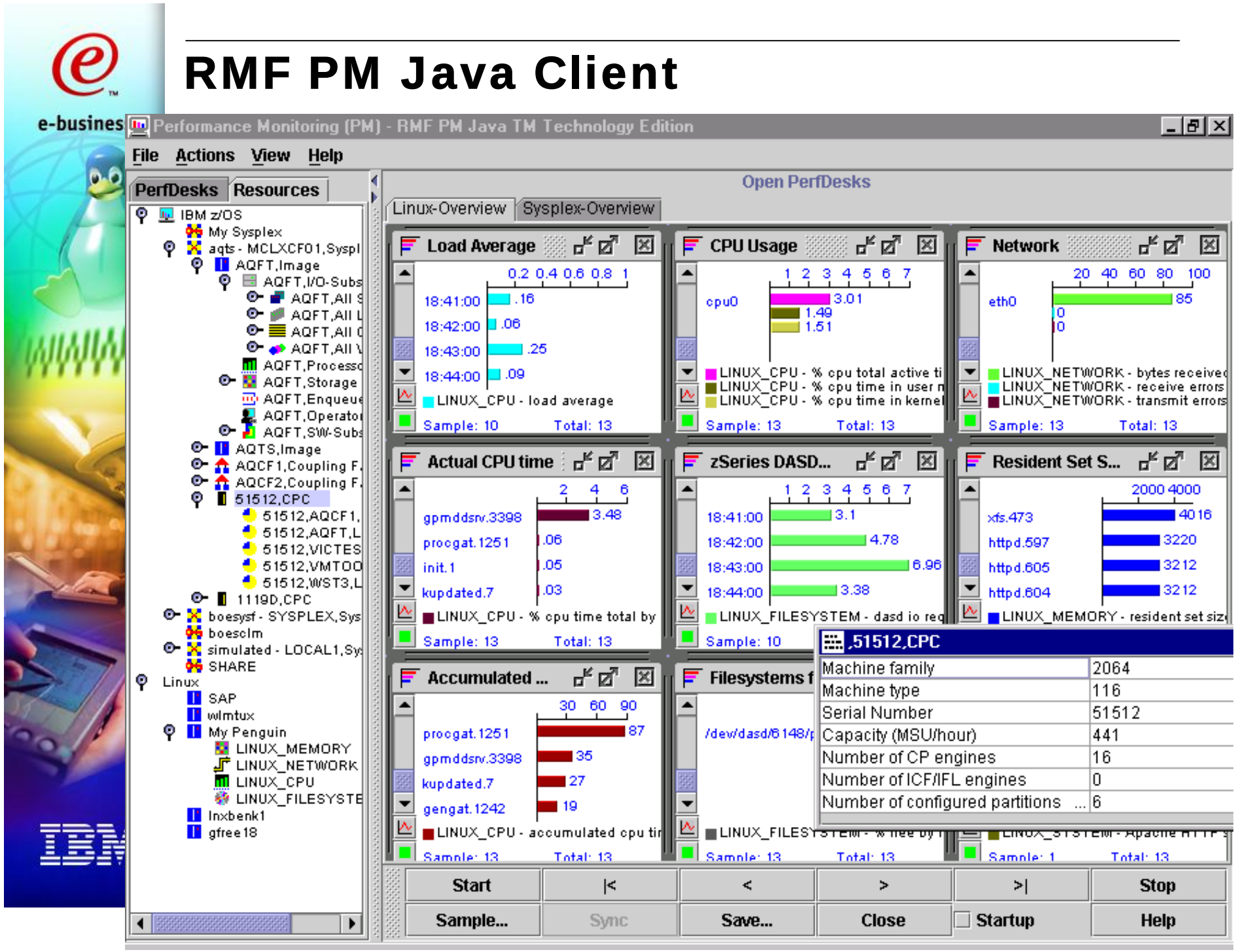
rmfpms

- **Long term data gathering**
- **XML over HTTP interface**
- **independant from z/OS; with z/OS, you can also have an LDAP interface to Linux performance data**
- **Modular architecture**
- **see** *http://www.s390.ibm.com/rmf/rmfhtmls/pmweb/pmlin.htm*

rmfpms (continued)

- **Integrated with z/OS RMF PM and z/VM FCON**
  - **If you have a mixed environment with z/OS and Linux or z/VM and FCON, you can have all relevant performance metrics in one application**
  - **Data reported by host tools like RMF (LPAR CPU performance data, iQDIO channel utilization, etc.) is very relevant for Linux; unfortunately, we cannot make all this data available for Linux currently**
  - **If you have a mixed environment with z/OS, z/VM and Linux, you currently might need third-party systems management software like Tivoli DM**
- **FCON is IBM's strategic tool for z/VM performance monitoring**

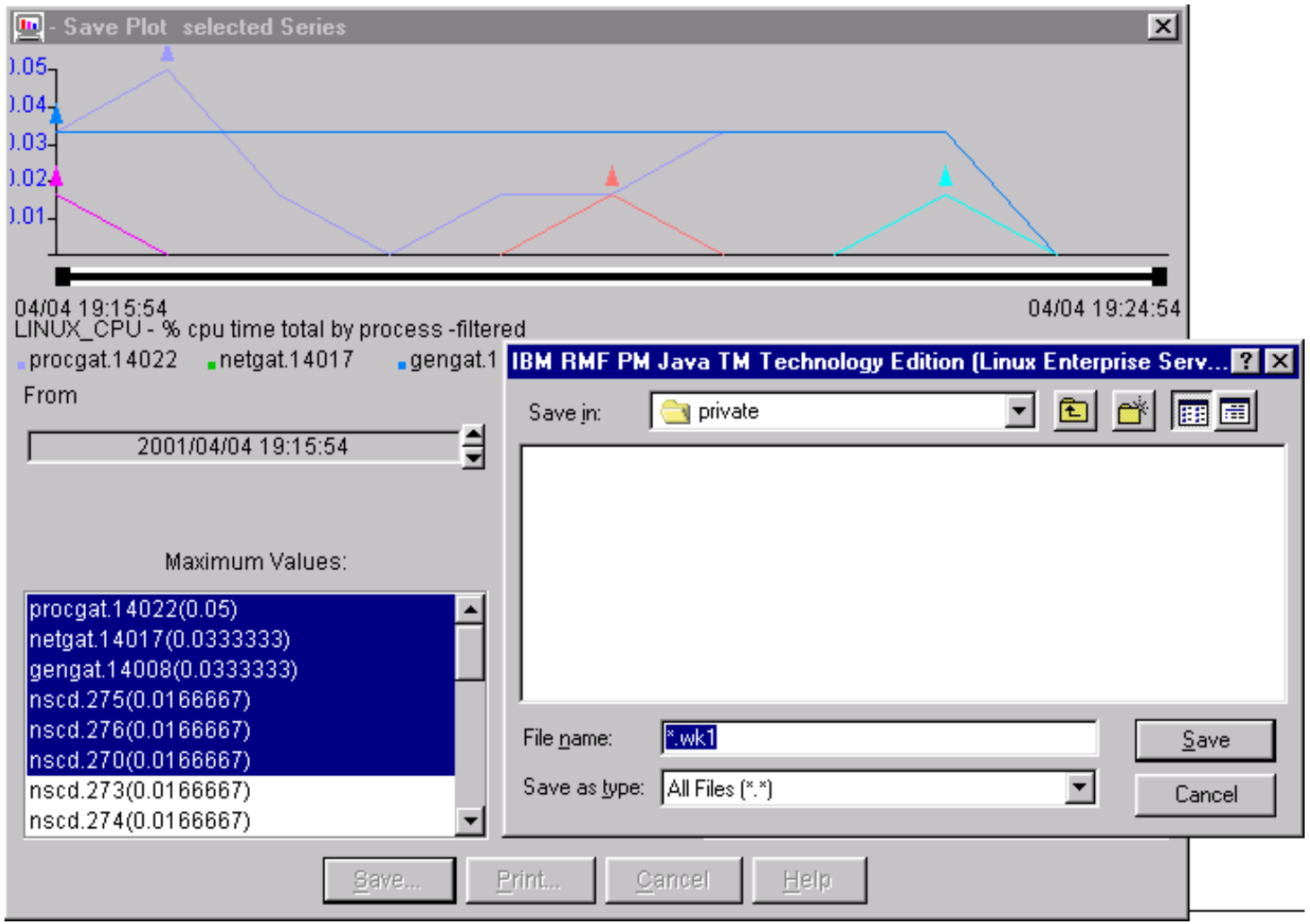
RMF PM Java Client



RMF PM Java Client (continued)

- Developed for OS/390 and z/OS
- positioned for online performance analysis and problem drill-down
- Can monitor multiple Linux server and multiple z/OS or OS/390 Sysplexes at the same time, in one application
- The performance analysis scenario can be saved

RMF PM: Save data in WK1 format



RMF PM: Spreadsheet Converter

SpreadCon

Images Options Advanced Help

Choose the fixed component:

image is fixed

image is fixed
resource/metric is fixed
time is fixed

Metrics

lnxbenke/6-resource(s)/

Select some metrics:

% used by file system
available (in MB) by file system
dasd io average response time p
dasd io average response time p
sd io requests per second
e (in MB) by file system
al size of all file systems (in MI

back ok

Time

lnxbenke/6-resource(s)/1- metric(s)

Insert the period of time:

start

date: << < 15 / 08 / 2002 > >>

time: << < 08 : 00 : 00 > >>

end

date: << < 15 / 08 / 2002 > >>

time:

RMF PM Web Browser Interface

RMF DDS Browser-Interface - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss

Address http://tux390:8803/ Go Links

RMF DDS Browser Interface

Overview | Explore | RMF | Home

tux390,',LINUX_CPU
% cpu time total by process

Local Time: 01/24/2002 19:08:00

in.telnetd.7401	0.05
bash.7403	0.05
nscd.287	0.0166667

tux390,',LINUX_CPU
% cpu total active time by processor

Local Time: 01/24/2002 19:08:00

cpu0	4.13918
cpu1	2.35818

tux390,',LINUX_FILESYSTEM
% used by file system

Local Time: 01/24/2002 19:08:00

/dev/dasdc1	67.7007
/dev/dasdb1	48.3766

tux390,',LINUX_FILESYSTEM
size (in MB) by file system

Local Time: 01/24/2002 19:08:00

/dev/dasdc1	2310
/dev/dasdb1	2274

tux390,',LINUX_MEMORY
major page fault rate including children by process

Local Time: 01/24/2002 19:08:00

init.1	114
bash.7403	64
atd.211	0

tux390,',LINUX_MEMORY
resident set size in KB by process

Local Time: 01/24/2002 19:08:00

httpd.5970	1896
httpd.300	1828
bash.7403	1488

Done Local intranet

... same technology for z/OS

RMF DDS Browser-Interface - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss

Address http://aqts.pok.ibm.com:8803/ Go Links

RMF DDS Browser Interface

Overview | Explore | RMF | Home

,MCLXCF01,SYSPLEX
% processor utilization by MVS image

Local Time is: 01/24/2002 13:15:00

Name	Value	Graph
AQFT	34	
AQTS	26	

,MCLXCF01,SYSPLEX
performance index by important WLM service class period

Local Time is: 01/24/2002 13:15:00

Name	Value	Graph
OMVSTASK.1	3.2	
HOTTSO.2	1.3	
TSOPRIME.3	0.5	
TSOPRIME.2	0.5	

,MCLXCF01,SYSPLEX
i/o intensity by volume

Local Time is: 01/24/2002 13:15:00

Name	Value	Graph
AQTS.OEDEVP	2354	
AQTS.C90LNH	1992	
AQTS.C90LN3	1990	
AQTS.C90BG2	1983	

,MCLXCF01,SYSPLEX
% CSA utilization by MVS image

Local Time is: 01/24/2002 13:15:00

Name	Value	Graph
AQFT	41	
AQTS	36	

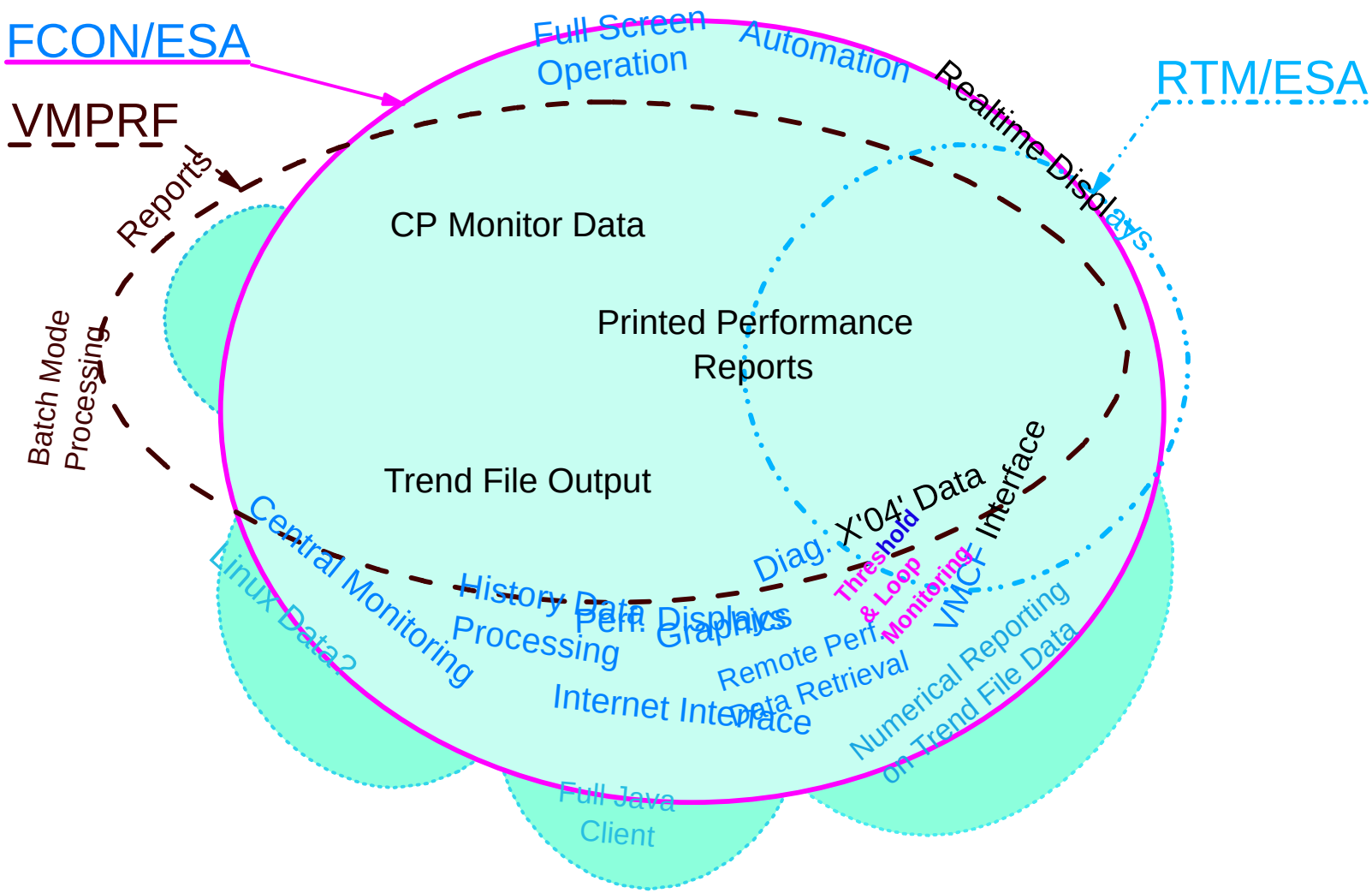
Done Internet

IBM FCON/ESA V.3.2.03

VM/ESA Full Screen Operator Console and Graphical Realtime Performance Monitor (5788-LGA) is IBM's strategic z/VM performance monitor. As it can display performance data collected by rmfpms in Linux, you can see VM and Linux performance data in one application.

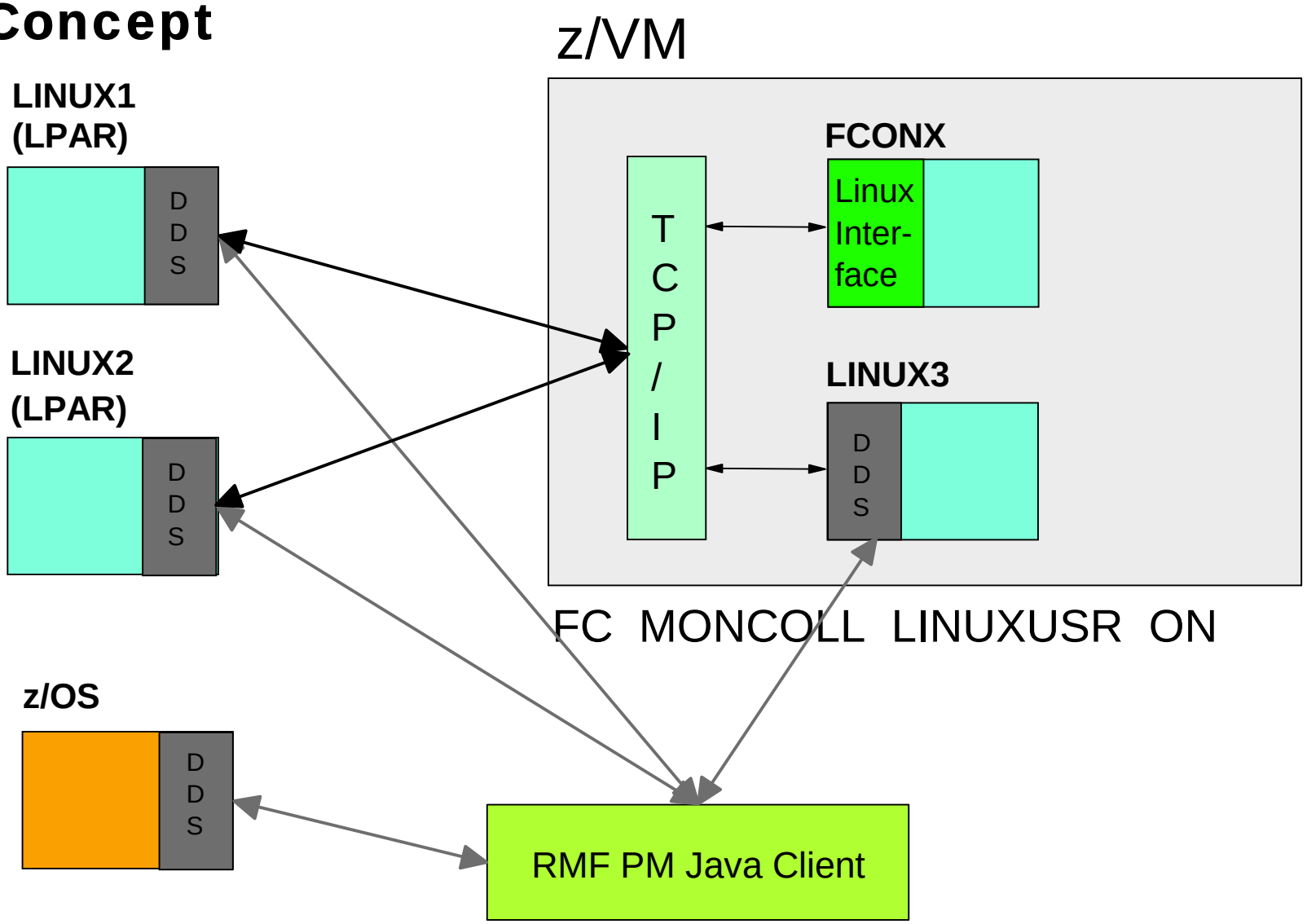
The developer is Eginhard Jaeger (ja@ch.ibm.com), IBM Switzerland.

FCON: The Wishlist



Accessing Linux Performance Data

Concept



Accessing Linux Perf. Data ...

System Definition

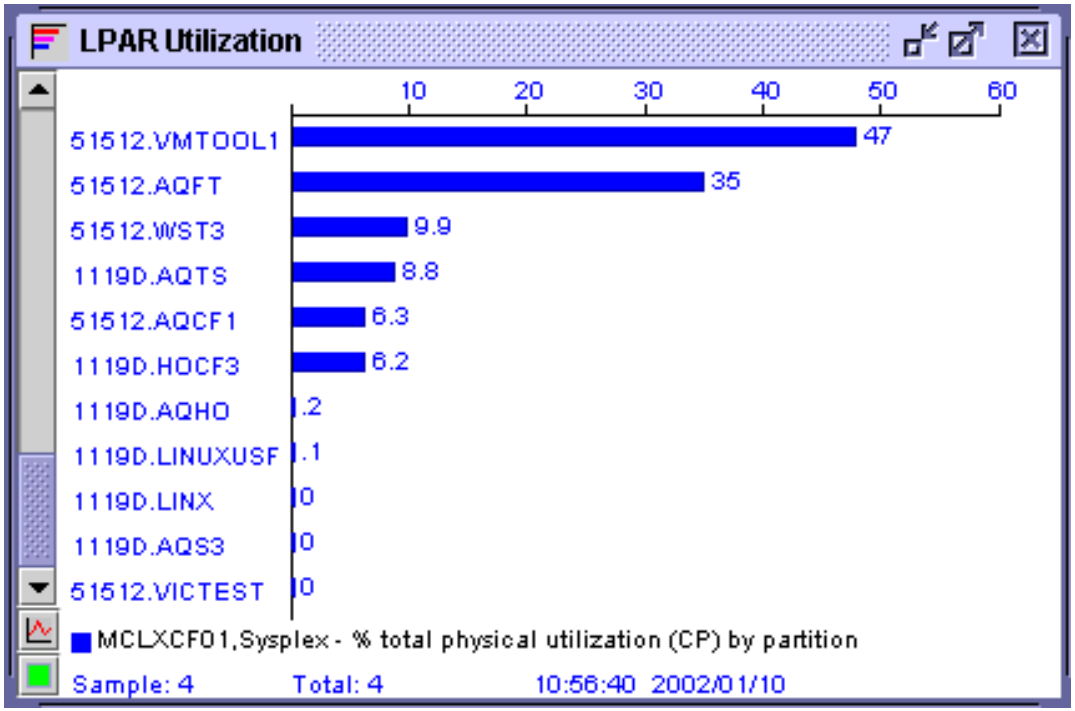
File **FCONX LINUXUSR**

```
*******************************************************
**  Initialization file with IP address definitions  **
**  for Linux systems that may have to be monitored. **
*******************************************************
*
LINUX1   1.111.111.111:8803
LINUX2   2.222.222.222:8803
LINUX3   3.333.333.333:8803
...
...
```

→ Defines IP addresses of Linux systems from which performance data may have to be retrieved.

You can only monitor systems defined in this file!

LPAR Partition Data (from z/OS RMF)



HiperSockets display in VM FCON

```
FCX231      CPU 2064  SER 51524  Interval 06:55:22 - 06:56:22    Perf. Monitor

_______           .         .         .         .         .         .         .
                 <----------------- Hipersocket Activity/Sec. ----------------->
Channel          <--- Total for System --->  <--------- Own Partition --------->
Path             <-Transferred-->    Failed  <-Transferred-->  <--- Failed ---->
ID       Shrd    T_Msgs  T_DUnits  T_NoBuff  L_Msgs  L_DUnits  L_NoBuff  L_Other
FB       No           0         0         0       0         0         0        0
FC       No           0         0         0       0         0         0        0
FD       No           0         0         0       0         0         0        0
FE       No           0         0         0       0         0         0        0
```

HiperSockets Display in z/OS RMF

```
                                C H A N N E L   P A T H   A C T I V I T Y                                  PA

          z/OS V1R2              SYSTEM ID CB88            DATE 07/22/2001          INTERVAL 22.54.336
                                 RPT VERSION V1R2 RMF      TIME 15.37.05            CYCLE 1.000 SECONDS

IODF = 01   CR-DATE: 05/10/2000   CR-TIME: 21.00.01   ACT: POR          MODE: LPAR        CPMF: EXTENDED MODE

------------------------------------------------------------------------------------------------------------------------
                                          OVERVIEW FOR DCM-MANAGED CHANNELS
------------------------------------------------------------------------------------------------------------------------

    CHANNEL          UTILIZATION(%)     READ(MB/SEC) WRITE(MB/SEC)
   GROUP  G NO     PART  TOTAL    BUS    PART  TOTAL   PART  TOTAL

   FC_SM  1  8    15.36  55.86   6.00  15.36  60.00  15.36  60.36
   FCV_M     12   30.00  45.00   5.00  45.00  50.00  45.00  50.00
   CNC_M      1   17.23  34.45

------------------------------------------------------------------------------------------------------------------------
                                          DETAILS FOR ALL CHANNELS
------------------------------------------------------------------------------------------------------------------------

  CHANNEL PATH     UTILIZATION(%)    READ(MB/SEC) WRITE(MB/SEC)     CHANNEL PATH     UTILIZATION(%)    READ(MB/SEC) WRITE(
ID TYPE   G SHR  PART  TOTAL    BUS   PART  TOTAL   PART  TOTAL   ID TYPE   G SHR  PART  TOTAL    BUS   PART  TOTAL   PART

78 CVC_P        OFFLINE                                          80 CTC_S        OFFLINE
79 CNC_S        OFFLINE                                          81 CNC_S         0.04   0.04
7A FC     1  Y  20.00  30.00   5.00  20.00  30.00  20.00  50.00  82 FC         Y  20.00  30.00   6.00  20.00  30.00  20.00
7B FC_SM     Y  15.36  55.86   6.00  15.36  60.00  15.36  60.36  83 FC     1   Y  15.36  55.66   7.00  15.36  60.00  15.36
7C FCV       Y  10.00  30.00   5.00  10.00  50.00  10.00  50.00  84 FCV        Y  10.00  30.00   5.00  10.00  50.00  50.00
7D FCV_M     Y  30.00  45.00   5.00  45.00  50.00  45.00  50.00  85 FCV        Y  30.00  45.00   6.00  45.00  50.00  45.00
7E CNC_M        17.23  34.45                                     86 CNC_S         0.00   0.00
7F CNC_S        OFFLINE                                          8C CNC_S         0.00   0.00



  CHANNEL PATH     WRITE(B/SEC)     MESSAGE RATE     MESSAGE SIZE   SEND FAIL    RECEIVE FAIL
ID TYPE   G SHR     PART   TOTAL     PART   TOTAL     PART   TOTAL     PART       PART   TOTAL

AB IQD       Y    645.12M 2500.2G   850.23K 4.2K    760.12   779.56    12         85     120
```

Interface between Linux kernel and z/VM CP

- **CP device driver, developed by Neale Ferguson; interface between Linux and z/VM**
- **http://penguinvm.princeton.edu/programs (cpint.tar.gz)**
- **"*#cp ind user*" in Linux console:**

```
CP IND
AVGPROC-069% 07
XSTORE-000037/SEC MIGRATE-0000/SEC
MDC READS-000001/SEC WRITES-000000/SEC HIT RATIO-094%
STORAGE-024% PAGING-0000/SEC STEAL-000%
Q0-00071 Q1-00000        Q2-00000 EXPAN-001 Q3-00000
EXPAN-001
```

Example Scenario

- **The following Linux image may be completely idle:**

  $ > top 12:30pm
  up 4 min, 2 users, load average: 0.02, 0.07, 0.03
  24 processes: 23 sleeping, 1 running, 0 zombie, 0 stopped
  CPU0 states: 0.1% user, 19.1% system, 0.0% nice, 80.8% idle
  CPU1 states: 0.0% user, 23.2% system, 0.0% nice, 76.8% idle
  ...

- **... as z/VM is heavily loaded and does not give Linux many resources, so even for simple tasks, Linux needs about 20% of its CPU resources just to do almost nothing:**

  $ > #CP IND
  AVGPROC-099% 07
  ...

Conclusion

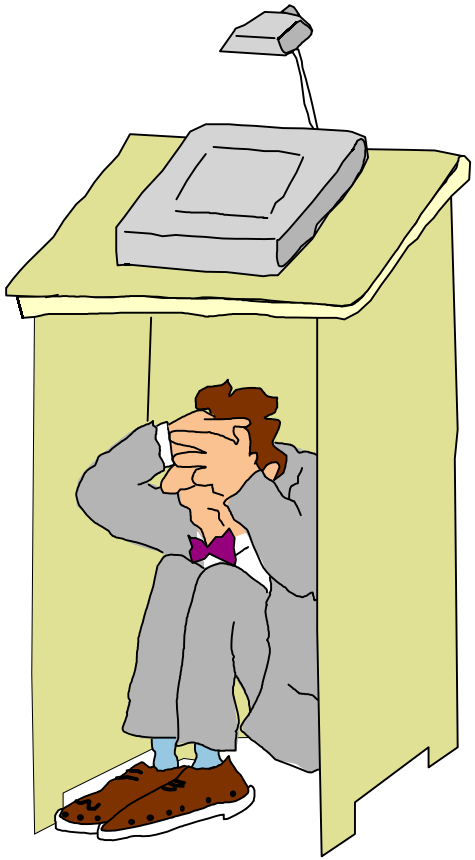
- zSeries virtualization technologies are far away from any competitive platform
- HiperSockets allow you to combine strength of Linux and z/OS; network elimination has lots of advantages
- Understand what can happen if you over-commit your memory under z/VM
- For tuning in an environment where every resource can be shared between heterogeneous instances, you need information from all layers (like LPAR Hypervisor, z/VM, Linux operating system, DB2 and SAP)
- Think about LPAR for high-end applications

Further Reading

- **Linux on zSeries and S/390: Systems Management Redbook, SG24-6820**
- **Linux for IBM eServer zSeries and S/390: "*ISP/ASP Solutions*" Redbook, SG24-6299**
- **Jason R Fink & Matthew D Sherer: "*Linux Performance Tuning and Capacity Planning*", SAMS 2001, ISBN 0-672-32081-9**

Related Sessions

- **2590: *Linux for zSeries Performance Update* by Klaus Bergmann/ Ulrich Weigand**
- ***~~2591: Details of Linux for zSeries DASD-Performance by Klaus Bergmann~~* (cancelled)**
- **9322: *Measuring and Tuning Linux on VM and Other Platforms* by Barton Robinson**
- **... and many many more sessions in the Linux and VM tracks**



Questions

?

Email: benke@de.ibm.com