Linux on System z
Performance Update - Part 1
z10, CPU and Memory

Mario Held
IBM Research & Development, Germany

August 28, 2009
Session Number 2190
Trademarks

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

- DB2*
- DB2 Connect
- DB2 Universal Database
- e-business logo
- GDPS*
- Geographically Dispersed Parallel Sysplex
- HyperSwap
- IBM*
- IBM eServer
- IBM logo*
- Parallel Sysplex*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

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

• System z10

• GCC compiler

• Java

• CPU hotplug

• Oprofile
IBM z10 EC - the CMOS Mainframe Heritage
IBM System z – system design comparison

**System I/O Bandwidth**

- **CPUs**
  - 1-way: ~900
  - 4-way: 540
  - 8-way: ~300
  - 16-way: ~150
  - 32-way: ~75
  - 64-way: ~50
- **Memory**
  - 512 GB: 24 GB/sec
  - 256 GB: 96 GB/sec
  - 64 GB: 172.8 GB/sec
  - 32 GB: 288 GB/sec

*Servers exploit a subset of its designed I/O capability*
<table>
<thead>
<tr>
<th>Mixed workload, multi-image with HiperDispatch active on z10 EC!</th>
<th>Ratios</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uni-processor</td>
<td>1.62</td>
</tr>
<tr>
<td>16-way z10 EC to 16-way z9 EC</td>
<td>1.49</td>
</tr>
<tr>
<td>32-way z10 EC to 32-way z9 EC</td>
<td>1.49</td>
</tr>
<tr>
<td>56-way z10 EC to 54-way z9 EC</td>
<td>1.54</td>
</tr>
<tr>
<td>64-way z10 EC to 54-way z9 EC</td>
<td>1.70</td>
</tr>
</tbody>
</table>
File server benchmark description

- dbench 3
  - Emulation of Netbench benchmark, rates windows file servers
  - Mainly memory operations
  - Mixed file operations workload for each process: create, write, read, append, delete
  - 8 CPUs and 1, 4, 8, 12, 16, 20, 26, 32, 40 processes
  - 2 GB memory
z10 Performance: dbench 3

- Improvement z10 versus z9:
  - Measured with 8 CPUs: average improvement is 50%
**z10 performance: CPU intensive workloads**

- Overall improvement with z10 versus z9: 1.9x
- gcc-4.3 compiler using -march=z9-109 or -march=z10 option
z10 Performance: Java workload

- System z versus System p, IBM J2RE 1.6.0

Throughput

<table>
<thead>
<tr>
<th></th>
<th>31-bit JVM z9</th>
<th>31-bit JVM z10</th>
<th>32-bit JVM p570</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 CPUs - 1 JVM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 CPUs - 2 JVM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 CPUs - 4 JVM</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CPU clock speeds
- z9: 1.7 GHz
- z10: 4.4 GHz
- p570: 4.7 GHz
z10 with Informix IDS 11 OLTP workload

- Throughput improvements
  - z9 to z10: 65% - 82%
  - A number of z10 CPUs can do the same work as the double number of z9 CPUs

![Graph showing transactions and scaling factor](image-url)
z10 performance summary

• System z evolution continues
• Sharp performance boost from z9 to z10
• Balanced System
• LSPR expectations met
• Excellent on compute intensive and Java workloads
Agenda

• System z10

• GCC compiler

• Java

• CPU hotplug

• Oprofile
GCC compiler evolution

Development makes new GCC features available for System z

GCC-4.3 patches

GCC-4.2 + software DFP

GCC-4.1 patches

GCC-3.4 patches

GCC-3.3 patches

Development exploits new System z hardware features

z10 support
hardware DFP

z9-ec | bc support

z9-109 support

z990 support
## GCC versions supported on System z

<table>
<thead>
<tr>
<th>GCC version</th>
<th>Used in SUSE distribution</th>
<th>Used in Red Hat distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCC-3.3</td>
<td>SLES9</td>
<td></td>
</tr>
<tr>
<td>GCC-3.4</td>
<td></td>
<td>RHEL4</td>
</tr>
<tr>
<td>GCC-4.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GCC-4.1</td>
<td>SLES10</td>
<td>RHEL5</td>
</tr>
<tr>
<td>GCC-4.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GCC-4.3</td>
<td>SLES11</td>
<td></td>
</tr>
<tr>
<td>GCC-4.4</td>
<td></td>
<td>RHEL6 ?</td>
</tr>
<tr>
<td>GCC-4.5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Optimizing C/C++ code

- Produce optimized code
  - Options -O3 or -O2 (often found in delivered Makefiles) are a good starting points
  - Optimize GCC instruction scheduling with the performance critical target machine in mind
    - -mtune=values <z9-109 from gcc-4.1 and up> <z10 with SLES11 gcc-4.3>
  - If you know the target machine exploit improved machine instruction set
    - -march=values <z9-109 from gcc-4.1 and up> <z10 with SLES11 gcc-4.3>
    - -march is only upward compatible

- Fine Tuning: additional general options on a file by file basis
  - Use of inline assembler for performance critical functions may have advantages
  - -funroll-loops --param max-unrolled-insns=100 has advantages
  - -ffast-math speeds up calculations (if not exact implementation of IEEE or ISO rules/specifications for math functions is needed)
  - Don't use debugging options in the final executable
GCC performance evolution on System z

- Run time of industry standard benchmark applications with newer GCC versions is much shorter
DFP – Decimal Floating Point

Limitations of binary numbers in economy

- Mercantile goods and amounts of money cannot be calculated or represented exactly by binary floating point numbers
  - Many numbers cannot be represented properly (1/5, 1/10)
  - People who are used to decimal numbers expect results and calculations to be available with full precision
  - The traditional binary representation is not suitable for usual calculations
    
    \[
    0.70 \times 1.05 = 0.73499999999999998667732370449812151491641998291015625
    \]
    
    Rounding to two digital places gives $0.73$
    
    Expected is $0.70 \times 1.05 = 0.735 \Rightarrow$ rounded to $0.74$

- If you rely on correctly calculated results without DFP you have to add many lines of code to your program
  - Example: troublesome binary floating point rounding mechanisms
    - Approx. more than 50 times the number of lines than in an DFP implementation
  - Depending on the amount of calculations a performance degradation is to expect
  - TCO is higher due to service, maintenance, run time
**DFP - support added in GCC**

- **Front end support (C, C++, Fortran, Java):**
  - Support for the 3 new data types: `_Decimal{32|64|128}`
  - Support for DFP constants written with DF suffix
- **Middle end support:**
  - Complete DFP arithmetic layer for constant folding
  - Support for integer or IEEE floating point conversion routines
- **GCC - versions**
  - Software DFP support in GCC-4.2 added
  - Hardware DFP support in GCC-4.3 added (usable with z9-ec, z10)
- **For the first time GCC-4.3.2 as available in SLES11 offers DFP in a supported environment on Linux on System z**
  - The usage of DFP arithmetics in applications requires the explicit use of DFP data types
  - If GCC is used with `-march=z9-ec` or `-march=z10` the HW DFP support is used by default
DFP - decimal floating point performance

- Telco testcase models a telephone company's billing system
  - Billing of one million telephone calls including tax using DFP arithmetics
- Big advantage if DFP hardware support is exploited
  - z9-ec DFP hardware support in millicode
  - z10 DFP hardware support by real hardware -> much faster

![Telco testcase: calculates 1 million telephone bills](chart)

<table>
<thead>
<tr>
<th></th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>z9-ec</td>
<td>Software: 8 µsec, Hardware: 2 µsec</td>
</tr>
<tr>
<td>z10</td>
<td>Software: 4 µsec, Hardware: 1 µsec</td>
</tr>
</tbody>
</table>

Lower is better.
Agenda

- System z10
- GCC compiler
- Java
- CPU hotplug
- Oprofile
Java on servers: Workload

• Evaluates server side Java
  • Emulates 3-tier system
    • Random input from user
    • Middle tier business logic implemented in Java
    • No explicit database --> emulated by Java objects

• Stressed components
  • Java
    • Virtual Machine (VM)
    • Just-In-Time compiler (JIT)
    • Garbage Collection (GC)
  • Linux operating system
    • Threads
    • CPUs
    • Caches and Memory
Java on servers: Performance Improvements

- Better virtual machines (VMs) and just-in-time (JIT) compilers
- Better garbage collection (GC) technologies
- Improvements through new hardware

History of Java versions

System z with Java SE 6 GA
Java: Large page support (z10 feature)

- use of -Xlp improves throughput
- large page size was 2 MB (default for SLES10, RHEL5)
Java on servers: 31-bit vs. 64-bit

- Use of -Xcompressedrefs provides relief for 64-bit (new with Java SE 6 SR2)
Java on servers: Heap size

- Heap size needs to be sized adequately
  - Maximum heap size ≤ available memory
    - avoids paging in Linux and z/VM
  - Heap too small: frequent garbage collection and OutOfMemoryErrors
  - Heap too big: infrequent garbage collection; Linux starts swapping
  - 31-bit Java kits: larger heap sizes up to 1.6 GB (modify memory layout)
    - also true for 31-bit Java kits in a 64-bit Linux environment

- Useful Java interpreter parameters for fine tuning – workload dependent
  - Setting a fixed heap size: -Xms (initial), -Xmx (maximum), when initial==maximum
  - Monitor garbage collection (GC): -verbose:gc
  - -Xlp tries to allocate large pages for the heap
    - prereq: Linux kernel needs to be setup for large pages (vm.nr_hugepages)
  - Control GC behavior: -Xgcpolicy:[optthrput, optavgpause, gencon]
  - 64-bit: smaller size of heap objects: -Xcompressedrefs
Java: larger heaps for 31-bit Java kits (1)

- Modify Linux memory layout
  - Reorder mapped base for shared libraries
- 31-bit emulation mode for Novell SLES9, SLES10

**HOWTO:**
- PID is the process ID of the process you want to change the layout (usually the bash shell)
  - `$$` gives the current shell PID, `cat /proc/self/maps` works as well
- Display memory map of any PID by
  - `cat /proc/<PID>/maps`
- Check the mapped base value by
  - `cat /proc/<PID>/mapped_base`
- Lower the value to e.g. 256 MB by
  - `echo 268435456 >/proc/<PID>/mapped_base`

`==> Now retry to allocate a larger heap size`
Java: larger heaps for 31-bit Java kits (2)

- Modify Linux memory layout
  - RHEL includes flex-mmap patch; turn off Linux prelinking
  - Applies to RHEL4, RHEL5 distributions (31-bit emulation mode)
- HOWTO:
  - Show state of flex-mmap patch
    - cat /proc/sys/vm/legacy_va_layout
    - 0 means flex-mmap is enabled; 1 means old memory layout
  - Enable flex-mmap if disabled
    - echo 0 > /proc/sys/vm/legacy_va_layout
  - Disable Linux prelinking
    - in /etc/sysconfig/prelink set PRELINKING=no
    - Apply setting by running the daily cron prelink job immediately
      - # /etc/cron.daily/prelink <ENTER>

=> Now retry to allocate a larger heap size
Java: Summary & Hints

- Try to use the **latest Java version**
  - Up to 60% release to release improvements
  - Up to 15% with newer service releases (SR) for a release
  - Middleware applications often bring their own Java Kit

- Make sure that you've got **JIT enabled**
  - Command 'java -version' says “JIT enabled/disabled”

- Lots of java interpreter `-X... parameters` for fine tuning
  - To get an idea type 'java -X'

- Provide an **optimal heap size** to your application

- Don't use the java interpreter in batch mode
  - Don't call x-times 'java Myprog'
  - Instead try to put the loop logic into your Java application
Agenda

- System z10
- GCC compiler
- Java
- CPU hotplug
- Oprofile
CPU hotplug function

- Changes the number of used processors on the fly, depending on the current overall utilization and load
- Is available with SLES10 SP2 and SLES11
- Expectation:
  - Increases the performance of single threaded applications within a z/VM or LPAR environment with multiple CPUs
- Enables or disables CPUs based on a set of rules
- Is enabled in the kernel configuration by setting

```plaintext
Base setup --->
--- Processor type and features ---
64 bit kernel (CONFIG_64BIT)
Symmetric multi-processing support (CONFIG_SMP)
└─ Support for hot-pluggable CPUs (CONFIG_HOTPLUG_CPU)
```
CPU hotplug parameters

- The control information is stored at /etc/sysconfig/cpuplugd
- Minimum number of CPUs is set with `cpu_min="<number>"`
- Maximum number of CPUs is set with `cpu_max="<number>"`
- The update interval is set with `update="<value in seconds>"`
- Consider the effect of kernel “cpu” parameters:
  - `maxcpus=<n>` sets the number of processors which will be active after system boot
  - `possible_cpus=<n>` is the upper limit for hotpluggable CPUs
  - If possible_cpus is not specified but maxcpus is, then maxcpus is the upper limit for hot-pluggable CPUs
CPU hotplug rules

- The default rule for increasing the number of CPUs is
  \[
  \text{HOTPLUG} = "\left( \text{loadavg} > \text{onumcpus} + 0.75 \right) \land \left( \text{idle} < 10.0 \right)"
  \]
  \[
  \text{HOTPLUG} = "(loadavg > onumcpus + 0.75) \& (idle < 10.0)"
  \]
  - An additional CPU is enabled, if the loadaverage is greater than the number of active (online) CPUs plus 0.75 and the current idle percentage is less than 10 percent.

- The default rule for decreasing the number of CPUs is
  \[
  \text{HOTUNPLUG} = "(\text{loadavg} < \text{onumcpus} - 0.25) \lor (\text{idle} > 50)"
  \]
  \[
  \text{HOTUNPLUG} = "(loadavg < onumcpus - 0.25) \lor (idle > 50)"
  \]
  - A CPU is disabled, either if the current load is below the number of active CPUs minus 0.25 or if the idle percentage is greater than 50%.

- The formulas for these rules can be modified. See “Device Drivers, Features and Commands” for valid expressions.

- Note:
  - \textit{loadavg} is a value that changes slowly
  - \textit{idle} changes fast
  - Increments and decrements of active CPUs are done in steps of 1 every time when the rules are checked.
CPU hotplug test workload

• dbench 3
  • Emulation of Netbench benchmark, rates windows file servers
  • Mainly memory operations
  • Mixed file operations workload for each process: create, write, read, append, delete
  • Scaling with 1,2,4,8,16 CPUs and 1,4,8,12,16,20,26,32 and 40 clients
  • 2 GB memory

• Modification to the standard code:
  • Purpose: Need more interaction between clients
    • Create two processes per client and communicate with POSIX message queues
  • First process:
    • *Read the I/O commands from the control file*
    • *Pass this information to the second process*
  • Second process:
    • *Performs the execution of this command*
    • *Reports the end of the operation back to the first process*
CPU hotplug performance results

- Improvements in case where the default (high) number of CPUs is not needed
- Up to 40% more throughput, up to 40% CPU cost savings

Throughput by dbench [MB/s]

Relative CPU consumption savings based on the test run without cpu hotplug [%]
CPU hotplug summary

- This feature improves the performance by
  - sizing the correct amount of processors for a Linux system depending on its current load
  - avoiding the Linux scheduler queue balancing in partial load situations
- Set the minimum and maximum number of CPUs to values which apply to the real workload:
  - Setting `cpu_min` to 2 may be too high
  - `cpu_max` should be set so that it really covers the peaks
- Linux guests under z/VM: use z/VM 5.4
  - Guarantees that stopped processors are no longer included in virtual processor prioritization calculations
  - Ensures share redistribution
Agenda

• System z10
• GCC compiler
• Java
• CPU hotplug
• Oprofile
Oprofile – the Open Source sampling tool

- Oprofile offers profiling of all running code on Linux systems, providing a variety of statistics
  - By default, kernel mode and user mode information is gathered for configurable events
- System z hardware currently does not have support for hardware performance counters, instead timer interrupt is used
  - Enable the hz_timer(!)
- The timer is set to whatever the jiffy rate is and is not user-settable
- Novell / SUSE: OProfile is on the SDK CDs
- Also available with RHEL4 and RHEL5
- More info at:
  http://oprofile.sourceforge.net/docs/
Oprofile – short HowTo

```bash
sysctl -w kernel.hz_timer=1
gunzip /boot/vmlinux-2.6.16.46-0.4-default.gz
specify the kernel level of `uname -r`

opcontrol --vmlinux=/boot/vmlinux-2.6.16.46-0.4-default

opcontrol --start

<DO THE TEST>

opcontrol --shutdown
opreport

any next test to run? If yes

opcontrol --reset
```
> opreport
CPU: CPU with timer interrupt, speed 0 MHz (estimated)
Profiling through timer interrupt

<table>
<thead>
<tr>
<th>TIMER:0</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>samples</td>
<td>%</td>
</tr>
</tbody>
</table>

140642 94.0617 vmlinux-2.6.16.46-0.4-default
3071 2.0539 libc-2.4.so
1925 1.2874 dbench
1922 1.2854 ext3
1442 0.9644 jbd
349 0.2334 dasd_mod
152 0.1017 apparmor
  6 0.0040 oprofiled
  5 0.0033 bash
  5 0.0033 ld-2.4.so
  1 6.7e-04 dasd_eckd_mod
  1 6.7e-04 oprofile

Kernel
glibc
application
file system
journaling
dasd driver
security
...

...
Opreport -- long-filenames | -l

```plaintext
>opreport -l
warning: /apparmor could not be found.
warning: /dasd_eckd_mod could not be found.
warning: /dasd_mod could not be found.
warning: /ext3 could not be found.
warning: /jbd could not be found.
warning: /oprofile could not be found.
CPU: CPU with timer interrupt, speed 0 MHz (estimated)
Profiling through timer interrupt

<table>
<thead>
<tr>
<th>samples</th>
<th>%</th>
<th>app name</th>
<th>symbol name</th>
</tr>
</thead>
<tbody>
<tr>
<td>130852</td>
<td>87.5141</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>cpu_idle</td>
</tr>
<tr>
<td>1922</td>
<td>1.2854</td>
<td>ext3</td>
<td>(no symbols)</td>
</tr>
<tr>
<td>1442</td>
<td>0.9644</td>
<td>jbd</td>
<td>(no symbols)</td>
</tr>
<tr>
<td>734</td>
<td>0.4909</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>memcpy</td>
</tr>
<tr>
<td>662</td>
<td>0.4427</td>
<td>libc-2.4.so</td>
<td>strchr</td>
</tr>
<tr>
<td>619</td>
<td>0.4140</td>
<td>dbench</td>
<td>next_token</td>
</tr>
<tr>
<td>567</td>
<td>0.3792</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>do_gettimeofday</td>
</tr>
<tr>
<td>536</td>
<td>0.3585</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__link_path_walk</td>
</tr>
<tr>
<td>525</td>
<td>0.3511</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>copy_to_user_std</td>
</tr>
<tr>
<td>435</td>
<td>0.2909</td>
<td>libc-2.4.so</td>
<td>strstr</td>
</tr>
<tr>
<td>413</td>
<td>0.2762</td>
<td>dbench</td>
<td>child_run</td>
</tr>
<tr>
<td>349</td>
<td>0.2334</td>
<td>dasd_mod</td>
<td>(no symbols)</td>
</tr>
<tr>
<td>347</td>
<td>0.2321</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>_spin_lock</td>
</tr>
<tr>
<td>328</td>
<td>0.2194</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>sysc_do_svc</td>
</tr>
<tr>
<td>285</td>
<td>0.1906</td>
<td>dbench</td>
<td>all_string_sub</td>
</tr>
<tr>
<td>283</td>
<td>0.1893</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__d_lookup</td>
</tr>
<tr>
<td>251</td>
<td>0.1679</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__find_get_block</td>
</tr>
<tr>
<td>231</td>
<td>0.1545</td>
<td>libc-2.4.so</td>
<td>__strtol_l__internal</td>
</tr>
<tr>
<td>216</td>
<td>0.1445</td>
<td>dbench</td>
<td>vsnprintf</td>
</tr>
<tr>
<td>209</td>
<td>0.1398</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>filldir64</td>
</tr>
<tr>
<td>205</td>
<td>0.1371</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>memset</td>
</tr>
<tr>
<td>196</td>
<td>0.1311</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>_atomic_dec_and_lock</td>
</tr>
<tr>
<td>166</td>
<td>0.1110</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>strtrc</td>
</tr>
<tr>
<td>155</td>
<td>0.1037</td>
<td>libc-2.4.so</td>
<td>memmove</td>
</tr>
<tr>
<td>152</td>
<td>0.1017</td>
<td>apparmor</td>
<td>(no symbols)</td>
</tr>
<tr>
<td>148</td>
<td>0.0990</td>
<td>libc-2.4.so</td>
<td>readdir</td>
</tr>
<tr>
<td>147</td>
<td>0.0983</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__brelse</td>
</tr>
<tr>
<td>146</td>
<td>0.0976</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>generic_file_buffered_write</td>
</tr>
</tbody>
</table>
```

Almost idle
Unresolved symbols
opreport -l --image-path -p [paths]

```bash
>opreport -l --image-path=/lib/modules/2.6.16.46-0.4-default/kernel/fs/ext3/,/lib/modules/2.6.16.46-0.4-default/kernel/fs/jbd/,/lib/modules/2.6.16.46-0.4-default/kernel/drivers/s390/block/,/lib/modules/2.6.16.46-0.4-default/kernel/security/apparmor/,/lib/modules/2.6.16.46-0.4-default/kernel/arch/s390/oprofile
```

CPU: CPU with timer interrupt, speed 0 MHz (estimated)

Profiling through timer interrupt

<table>
<thead>
<tr>
<th>samples</th>
<th>%</th>
<th>image name</th>
<th>app name</th>
<th>symbol name</th>
</tr>
</thead>
<tbody>
<tr>
<td>130852</td>
<td>87.5141</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>cpu_idle</td>
</tr>
<tr>
<td>734</td>
<td>0.4909</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>memcpy</td>
</tr>
<tr>
<td>662</td>
<td>0.4427</td>
<td>libc-2.4.so</td>
<td>libc-2.4.so</td>
<td>strchr</td>
</tr>
<tr>
<td>619</td>
<td>0.4140</td>
<td>dbench</td>
<td>dbench</td>
<td>next_token</td>
</tr>
<tr>
<td>567</td>
<td>0.3792</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>do_gettimeofday</td>
</tr>
<tr>
<td>536</td>
<td>0.3585</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__link_path_walk</td>
</tr>
<tr>
<td>525</td>
<td>0.3511</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>copy_to_user_std</td>
</tr>
<tr>
<td>435</td>
<td>0.2909</td>
<td>libc-2.4.so</td>
<td>libc-2.4.so</td>
<td>strstr</td>
</tr>
<tr>
<td>413</td>
<td>0.2762</td>
<td>dbench</td>
<td>dbench</td>
<td>child_run</td>
</tr>
<tr>
<td>361</td>
<td>0.2414</td>
<td>ext3.ko</td>
<td>ext3</td>
<td>ext3_get_block_handle</td>
</tr>
<tr>
<td>347</td>
<td>0.2321</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>_spin_lock</td>
</tr>
<tr>
<td>328</td>
<td>0.2194</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>sysc_do_svc</td>
</tr>
<tr>
<td>285</td>
<td>0.1906</td>
<td>dbench</td>
<td>dbench</td>
<td>all_string_sub</td>
</tr>
<tr>
<td>283</td>
<td>0.1893</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__d_lookup</td>
</tr>
<tr>
<td>251</td>
<td>0.1679</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__find_get_block</td>
</tr>
<tr>
<td>231</td>
<td>0.1545</td>
<td>libc-2.4.so</td>
<td>libc-2.4.so</td>
<td>__strtol_l_internal</td>
</tr>
<tr>
<td>226</td>
<td>0.1511</td>
<td>ext3.ko</td>
<td>ext3</td>
<td>ext3_try_to_allocate</td>
</tr>
<tr>
<td>223</td>
<td>0.1491</td>
<td>dasd_mod.ko</td>
<td>dasd_mod</td>
<td>dasd_malloc_request</td>
</tr>
<tr>
<td>216</td>
<td>0.1445</td>
<td>dbench</td>
<td>dbench</td>
<td>vsnprintf</td>
</tr>
<tr>
<td>209</td>
<td>0.1398</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>filldir64</td>
</tr>
<tr>
<td>205</td>
<td>0.1371</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>memset</td>
</tr>
<tr>
<td>196</td>
<td>0.1311</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>_atomic_dec_and_lock</td>
</tr>
<tr>
<td>188</td>
<td>0.1257</td>
<td>ext3.ko</td>
<td>ext3</td>
<td>ext3_new_inode</td>
</tr>
<tr>
<td>166</td>
<td>0.1110</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>strchrb</td>
</tr>
<tr>
<td>157</td>
<td>0.1050</td>
<td>jbd.ko</td>
<td>jbd</td>
<td>journal_init_dev</td>
</tr>
<tr>
<td>155</td>
<td>0.1037</td>
<td>libc-2.4.so</td>
<td>libc-2.4.so</td>
<td>memmove</td>
</tr>
<tr>
<td>148</td>
<td>0.0990</td>
<td>libc-2.4.so</td>
<td>libc-2.4.so</td>
<td>readdir</td>
</tr>
<tr>
<td>147</td>
<td>0.0983</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__brelse</td>
</tr>
<tr>
<td>146</td>
<td>0.0976</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>generic_file_buffered_write</td>
</tr>
<tr>
<td>144</td>
<td>0.0963</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>generic_permission</td>
</tr>
<tr>
<td>140</td>
<td>0.0936</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>__getblk</td>
</tr>
<tr>
<td>140</td>
<td>0.0936</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>vmlinux-2.6.16.46-0.4-default</td>
<td>kmem_cache_free</td>
</tr>
</tbody>
</table>
```
Visit us!

- Linux on System z: Tuning Hints & Tips

- Linux-VM Performance Website:

- IBM Redbooks
  http://www.redbooks.ibm.com/

- IBM Techdocs
Questions