Linux on System z Performance Update

Thomas Weber (tweber@de.ibm.com)
WAVV Conference 2008
Chattanooga, TN, April 18-22

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
IBM*
IBM eServer
IBM logo*
Informix®
System z
Tivoli*
WebSphere*
z/VM*
zSeries*
z/OS*
ECKD
Enterprise Storage Server®
FICON
FICON Express
HiperSocket
OSA
OSA Express

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

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.

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.

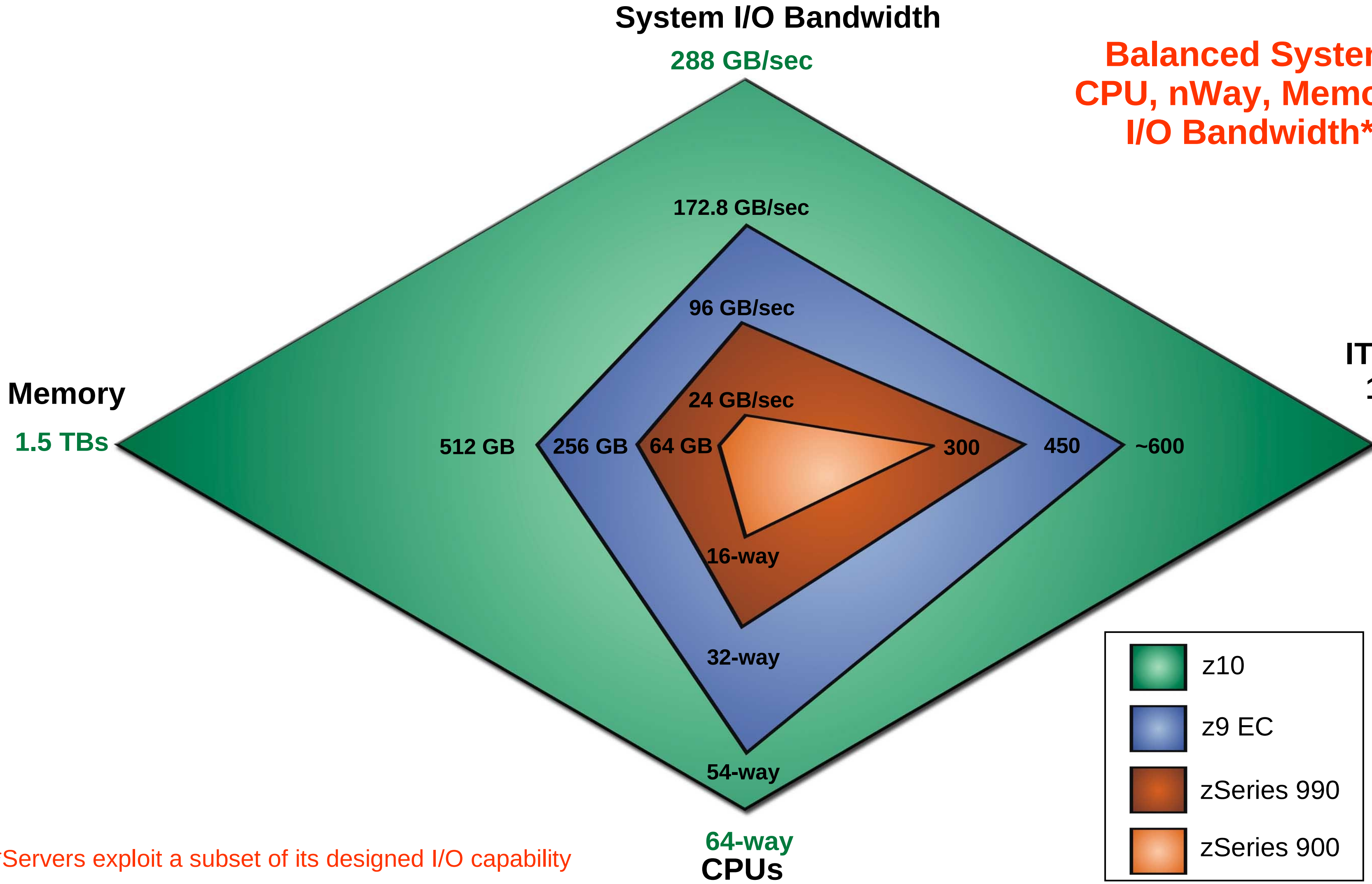
Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products 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 z hardware
- Hardware improvements
  - Processor
  - Networking
  - Disk / Tape
  - Cryptography
- Software improvements
  - Compiler
  - Java
  - WebSEAL
  - Tivoli Storage Manager
- Distribution improvements
  - Red Hat
  - Novell SUSE

IBM System z – system design comparison



Balanced System
CPU, nWay, Memory, I/O Bandwidth*

*Servers exploit a subset of its designed I/O capability

Our hardware for measurements

2084-B16 (z990)
0.83ns (1.2 GHz)
2 Books, 16 CPUs
2 * 32 MB L2 Cache
80 GB
FICON-Express2

2094-S18 (z9-109)
0.58ns (1.7GHz)
2 Books, 18 CPUs
2*40 MB L2 Cache
128 GB
FICON-Express4

HiperSockets
OSA-Express2 (10)GbE

2105-800 (Shark)
32 GB Cache
1 GB NVS
128 * 72 GB disks
15.000 RPM
FCP (2 Gbps)
FICON (2 Gbps)

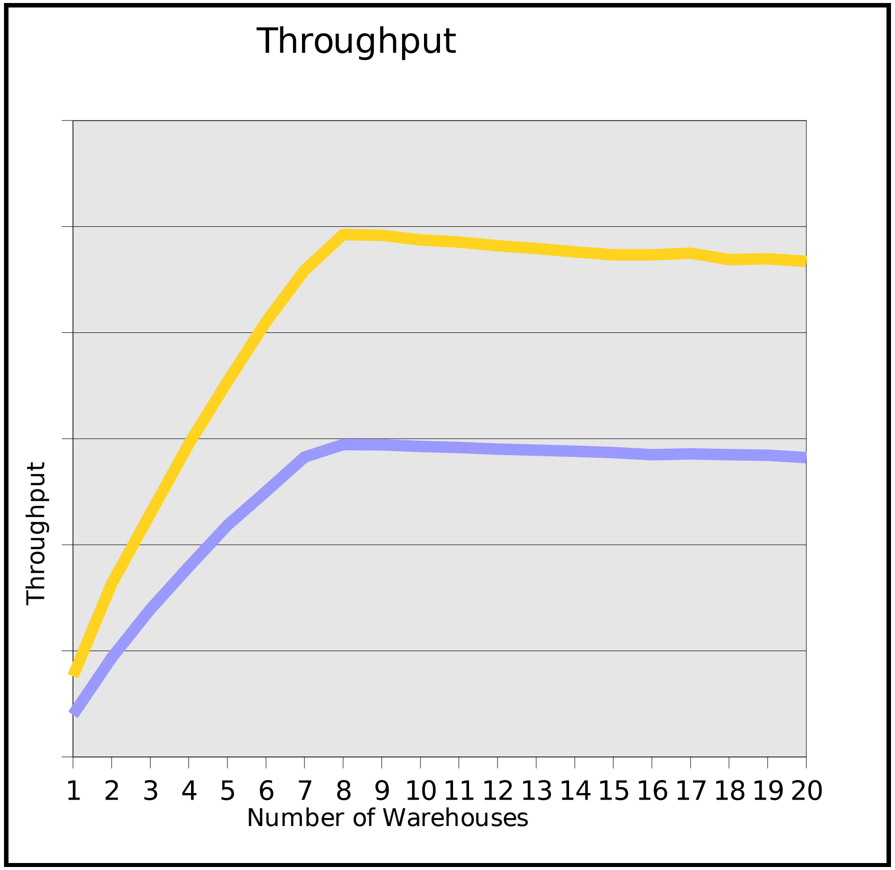
2107-922 (DS8300)
256 GB Cache
8 GB NVS
256 * 72 GB disks
15.000 RPM
FCP (4 Gbps)
FICON (4 Gbps)

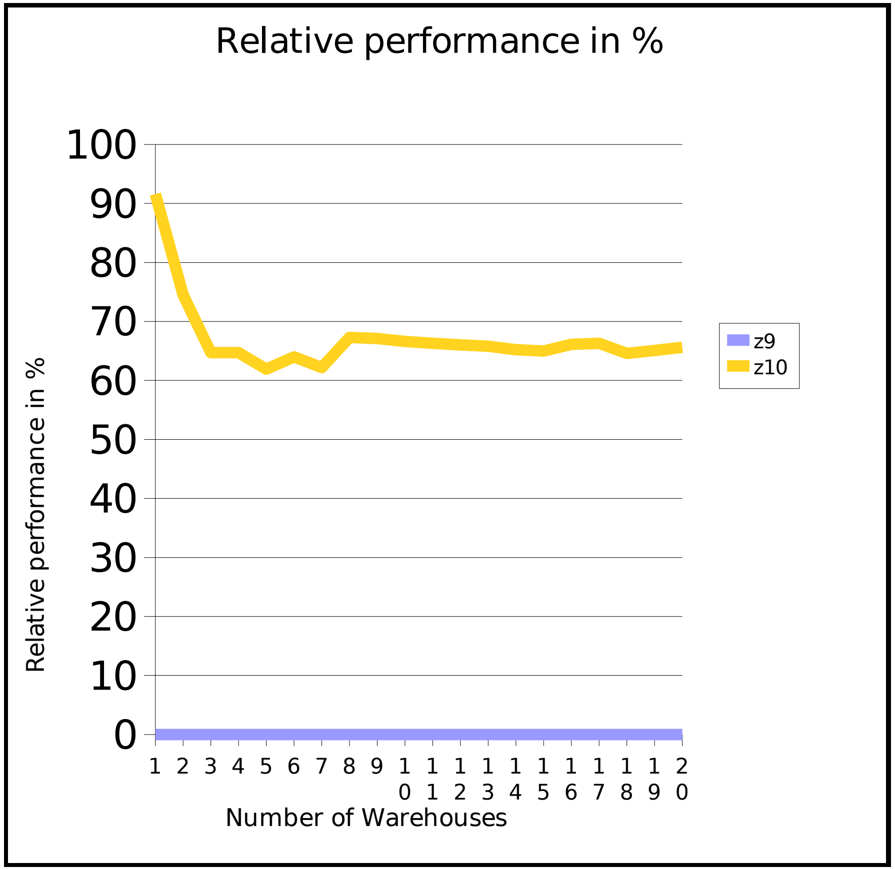
Agenda

- System z hardware
- Hardware improvements
  - Processor
  - Networking
  - Disk / Tape
  - Cryptography
- Software improvements
  - Compiler
  - Java
  - WebSEAL
  - Tivoli Storage Manager
- Distribution improvements
  - Red Hat
  - Novell/SUSE

z10 Performance: Java workload 1

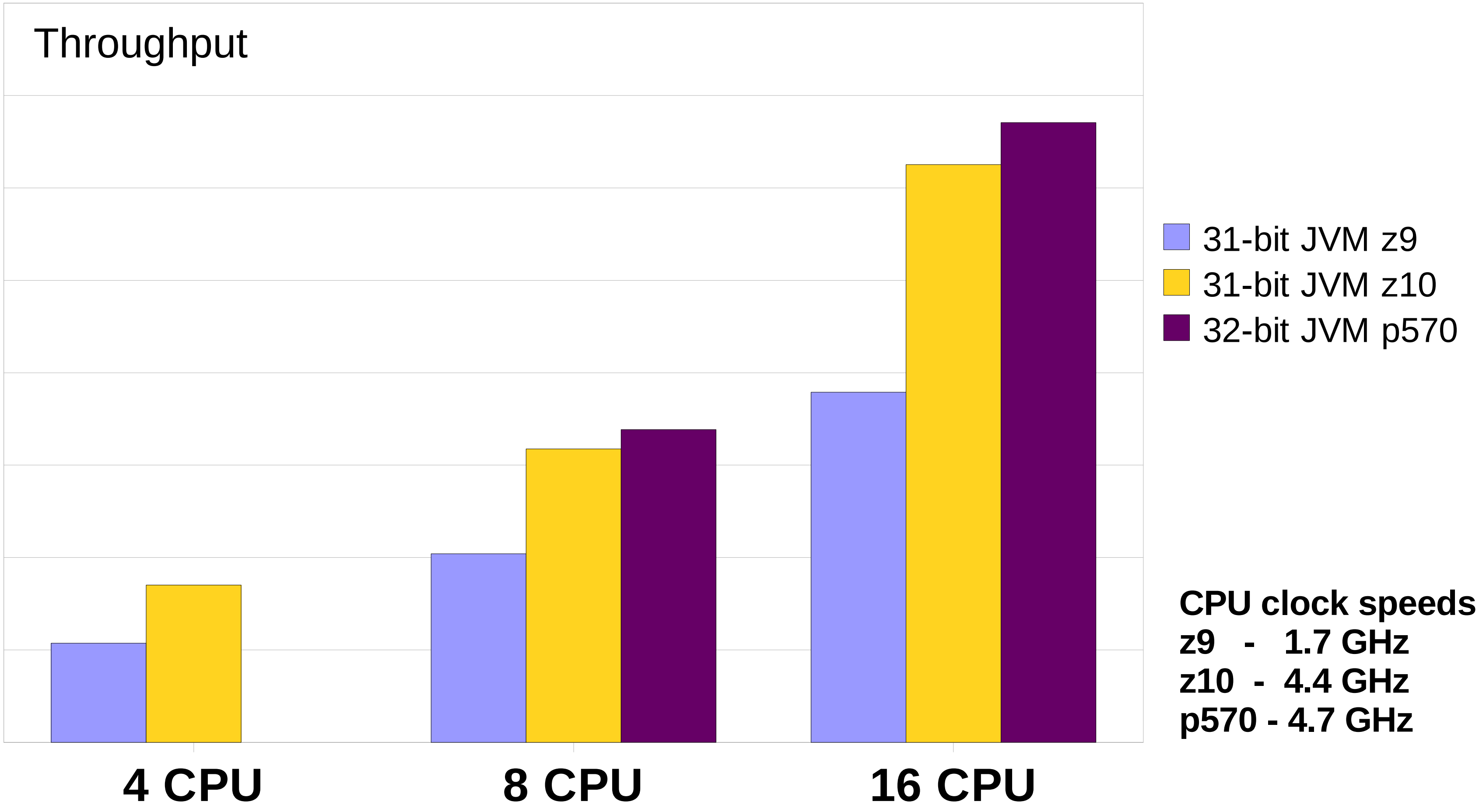
- **Overall improvement with z10 versus z9: 1.65x**





z10 Performance: Java workload 2

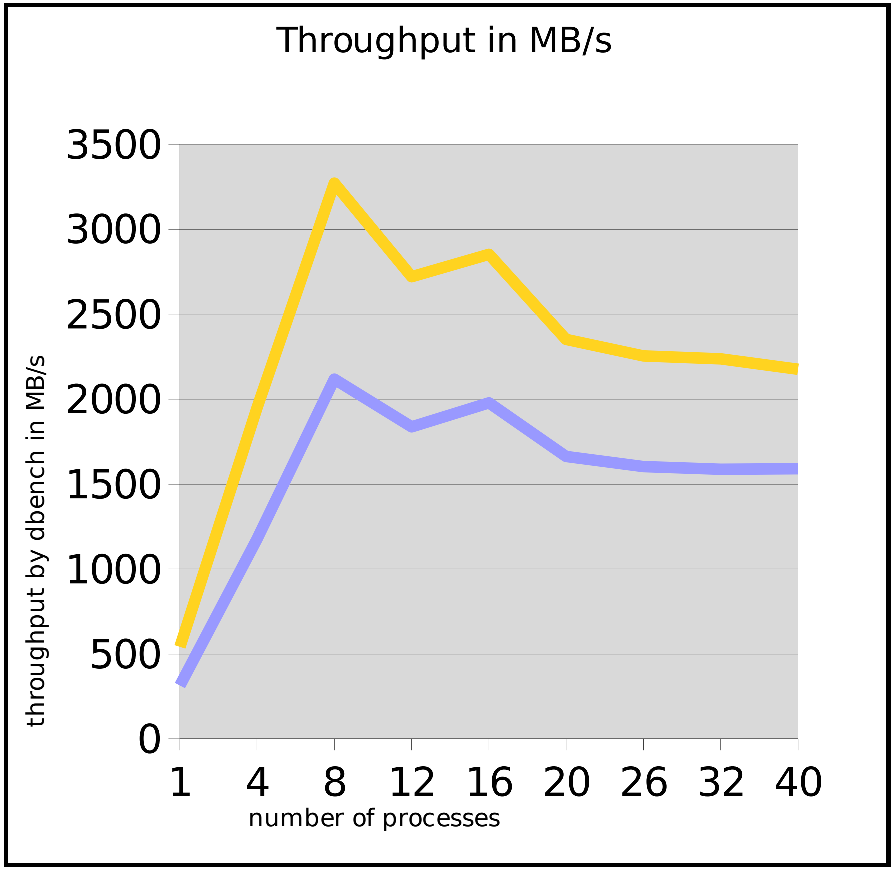
- **System z versus System p**

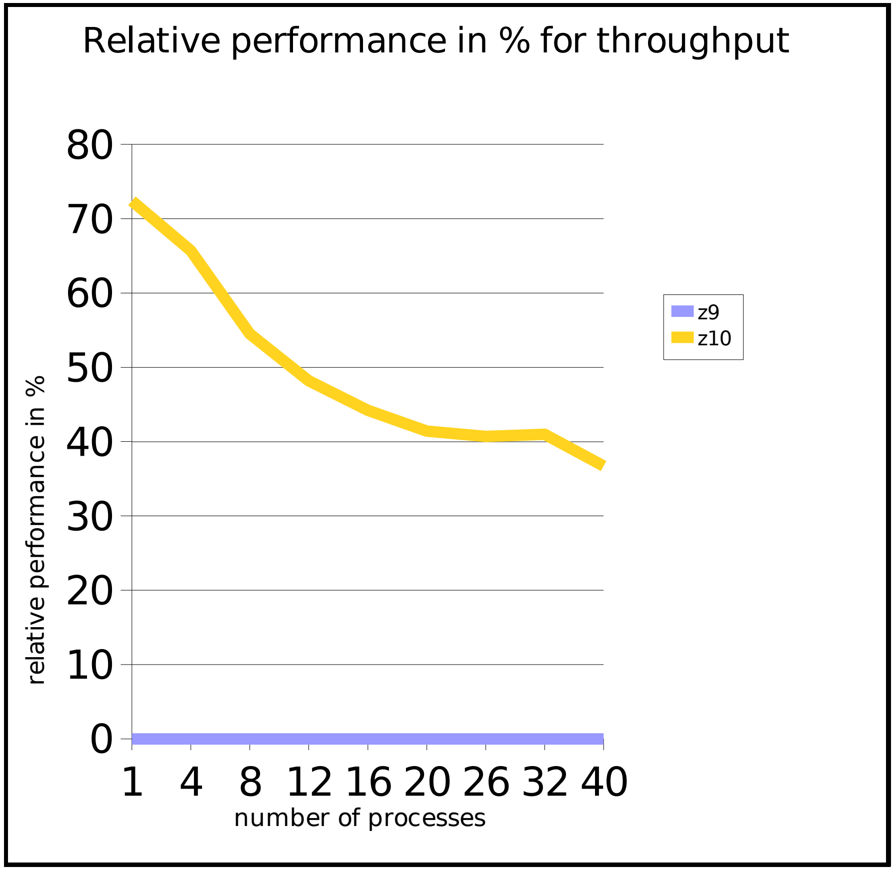


CPU clock speeds
z9 - 1.7 GHz
z10 - 4.4 GHz
p570 - 4.7 GHz

z10 Performance: DBench (file server workload)

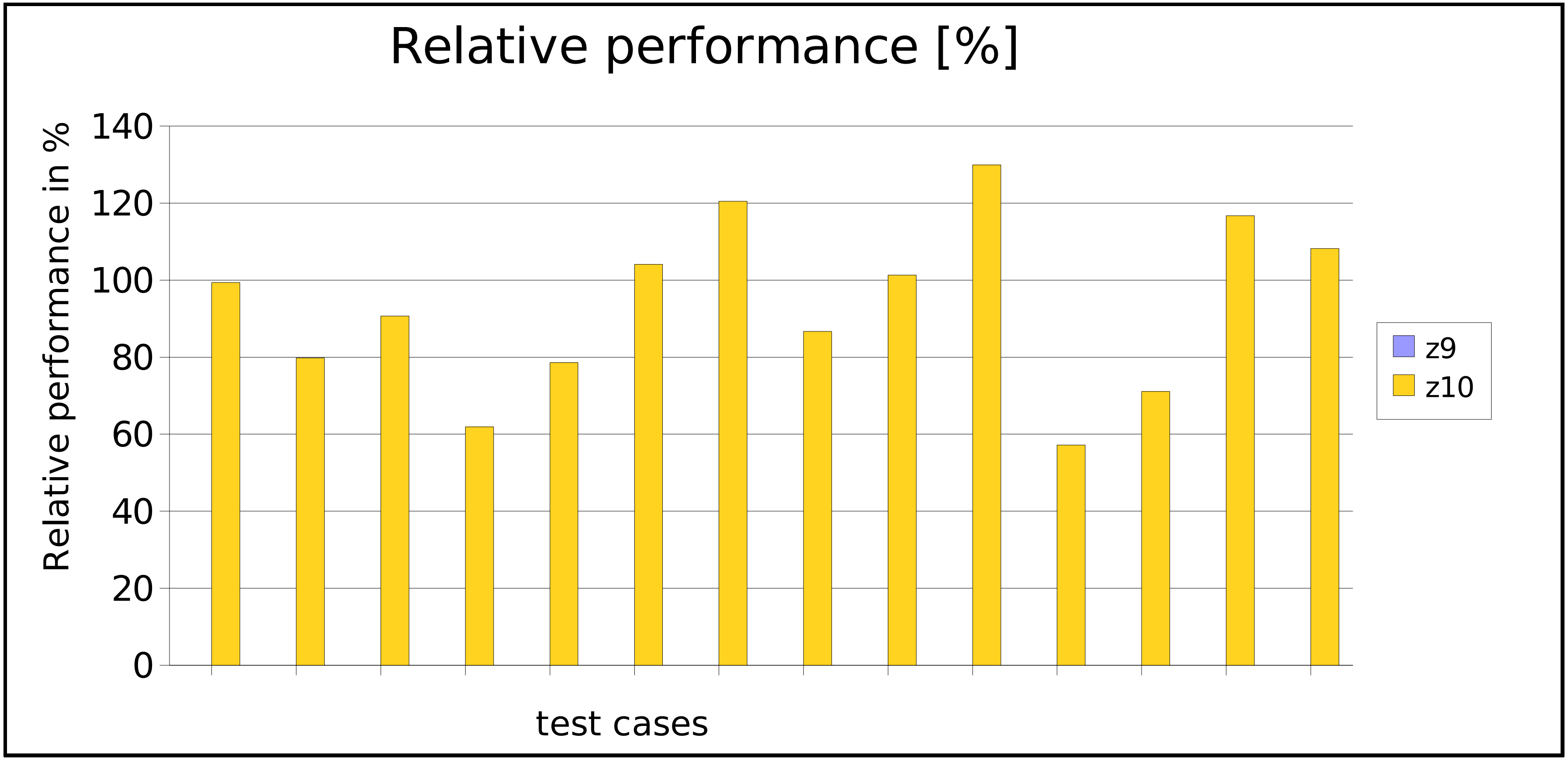
- **Improvement with z10 versus z9:**
  - **For 1 or 2 CPUs = 1.75x, for 8 CPUs = 1.5x (see below)**





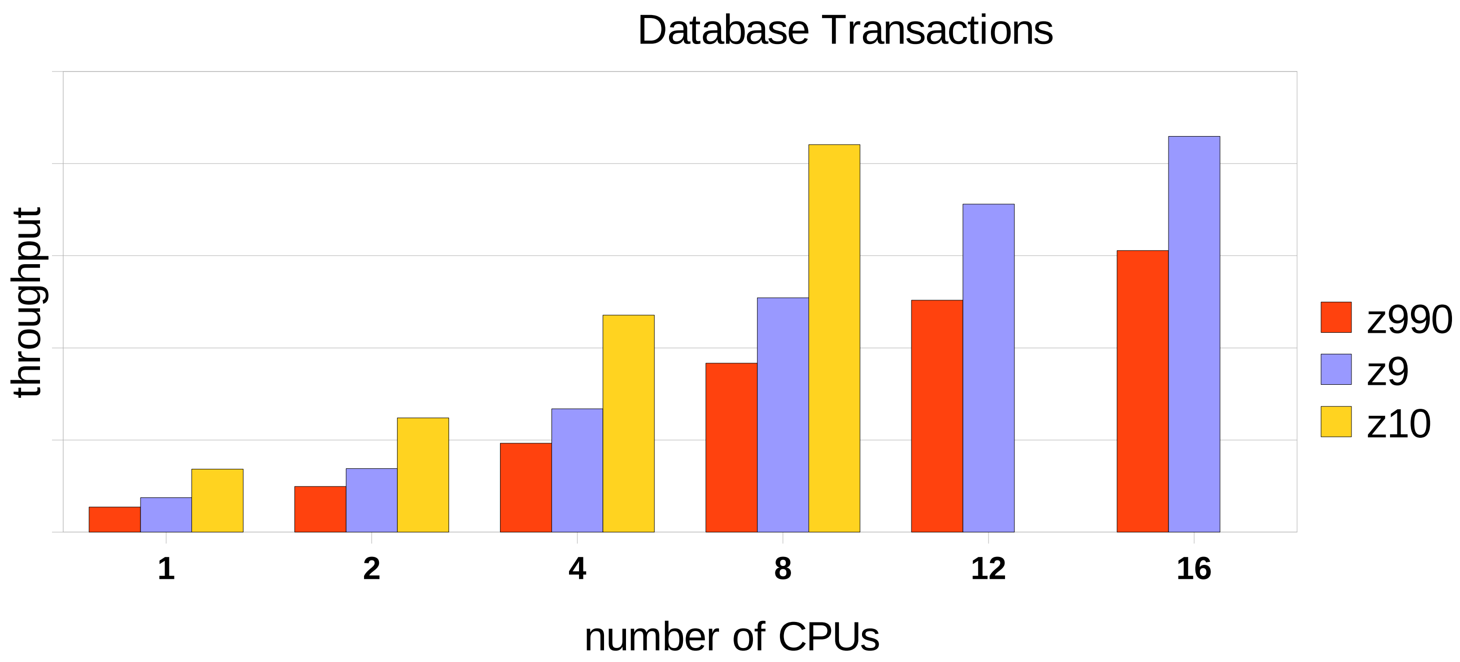
z10 Performance: Compiler (gcc) workloads

- **Overall improvement with z10 versus z9: 1.92x**



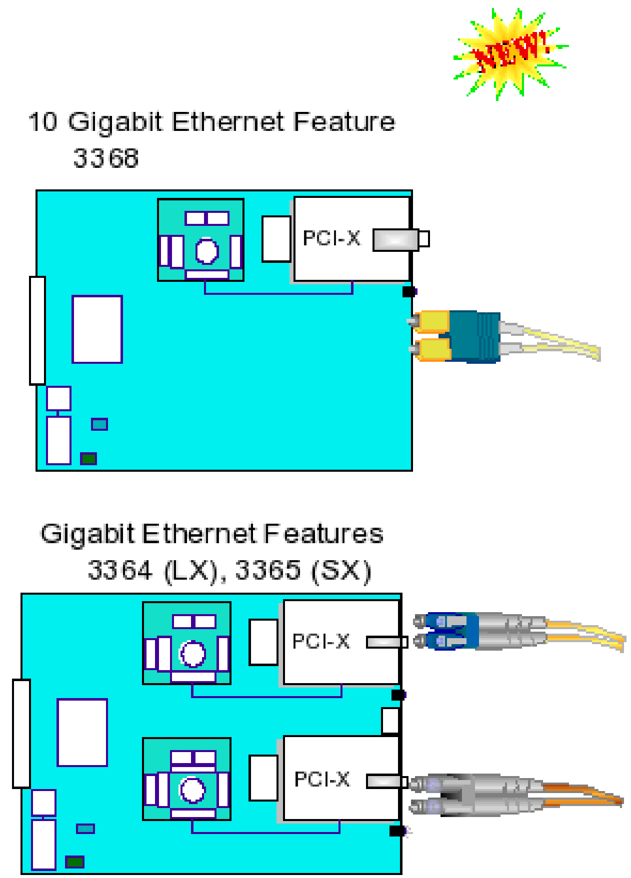
z10 with Informix IDS 11 OLTP workload

- **Throughput improvements**
  - **z9 to z10: 65% to 82% more processed transactions**
  - **x numbers of z10 CPUs can do the same work as 2x z9 CPUs**
  - **bufferpool high hit scenario**



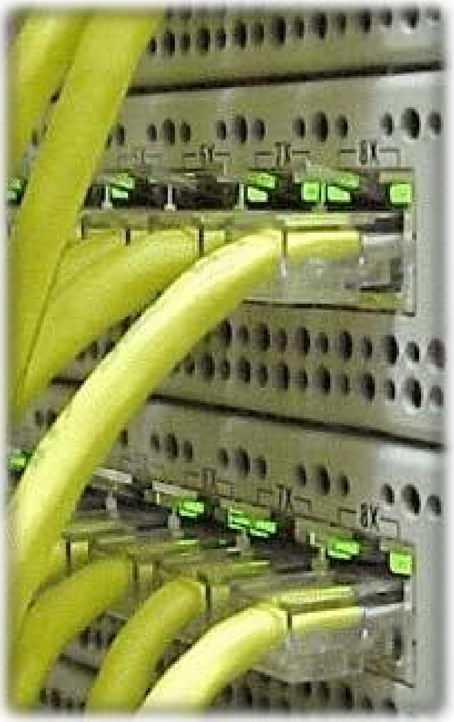
OSA-Express2

- newest member – 10 Gb Ethernet (GbE) LR (long reach)
  - one port per feature
- New – 1 GbE features
  - GbE LX (long wavelength)
  - GbE SX (short wavelength)
- support offered by both 10 GbE and 1 GbE
  - Layer 2 support
  - up to 1920 TCP/IP stacks for improved virtualization
  - large send for CPU efficiency

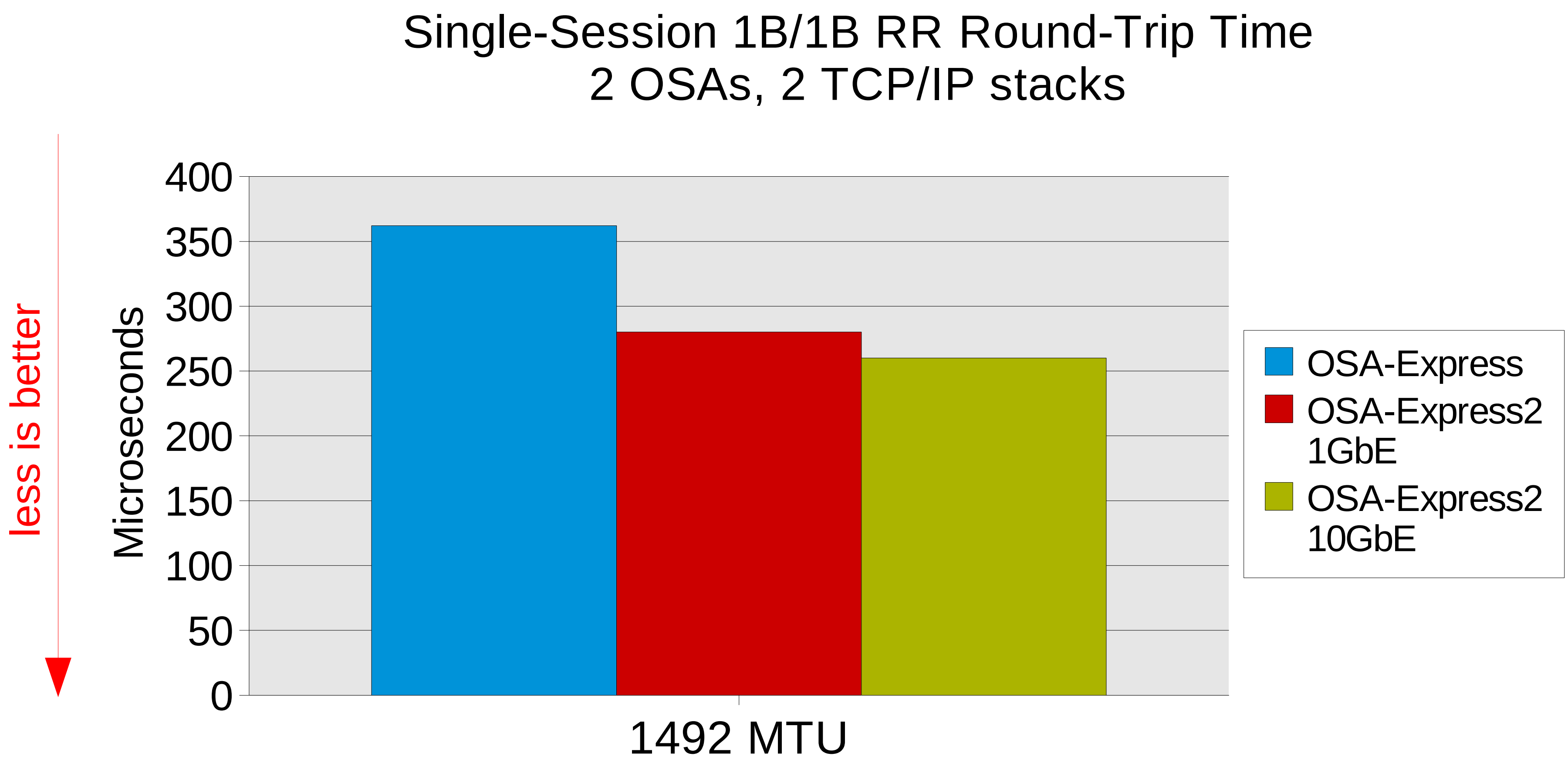


Networking benchmark

- AWM
- several workload models
  - transactional workload
  - streaming workload
  - mixed workload
- measured with GbE (QDIO), Hipersockets, and virtual connections in z/VM
- throughput and CPU cost metrics

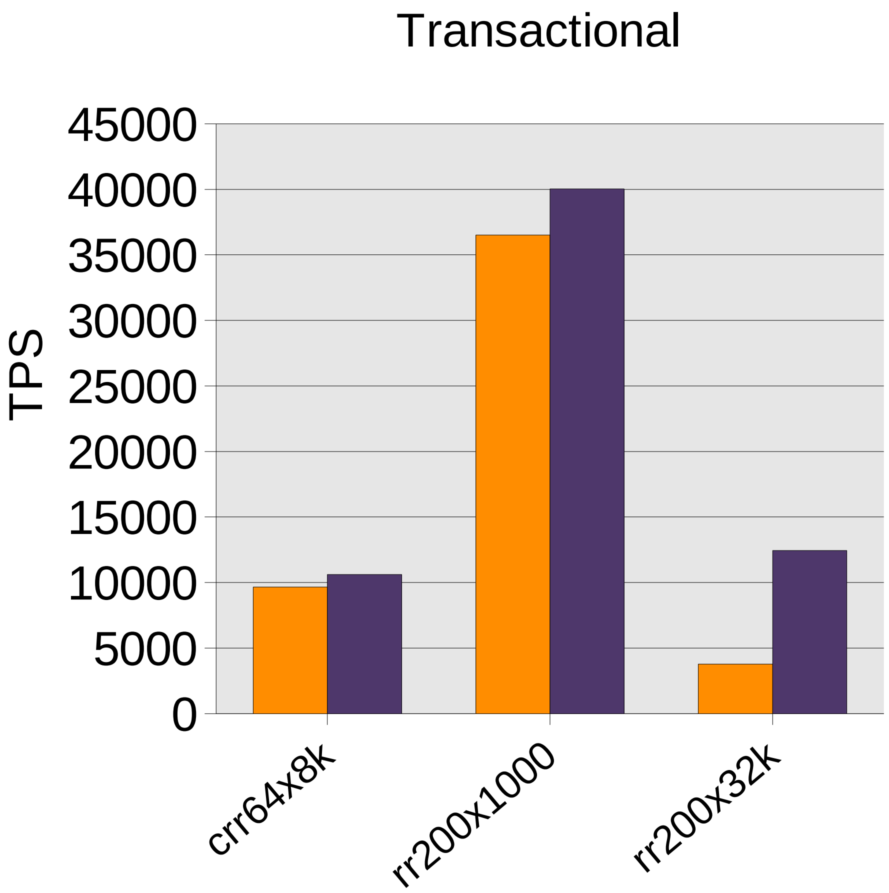


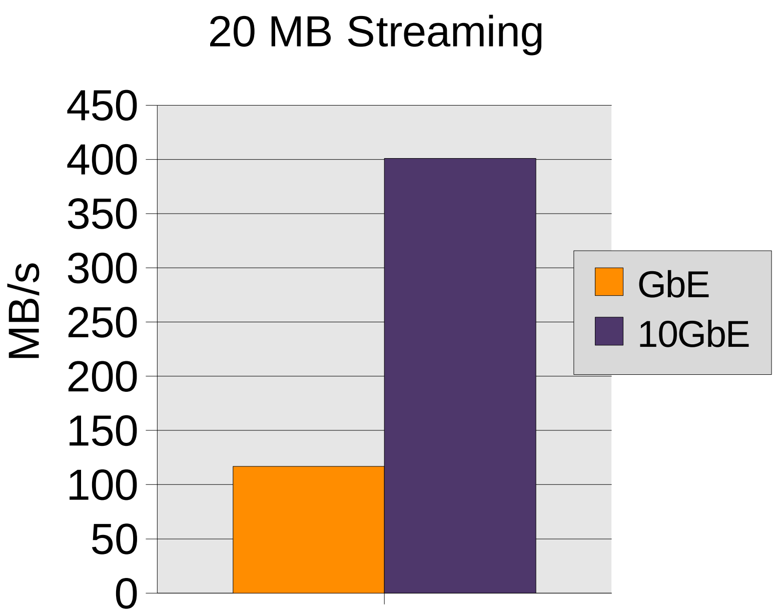
Response times



- **More than 20% improvement with OSA-Express2**

OSA-Express2 - 1GbE / 10GbE, MTU 8992





crr64x8k – website request

rr200x1000 – online transaction

rr200x32k – database query

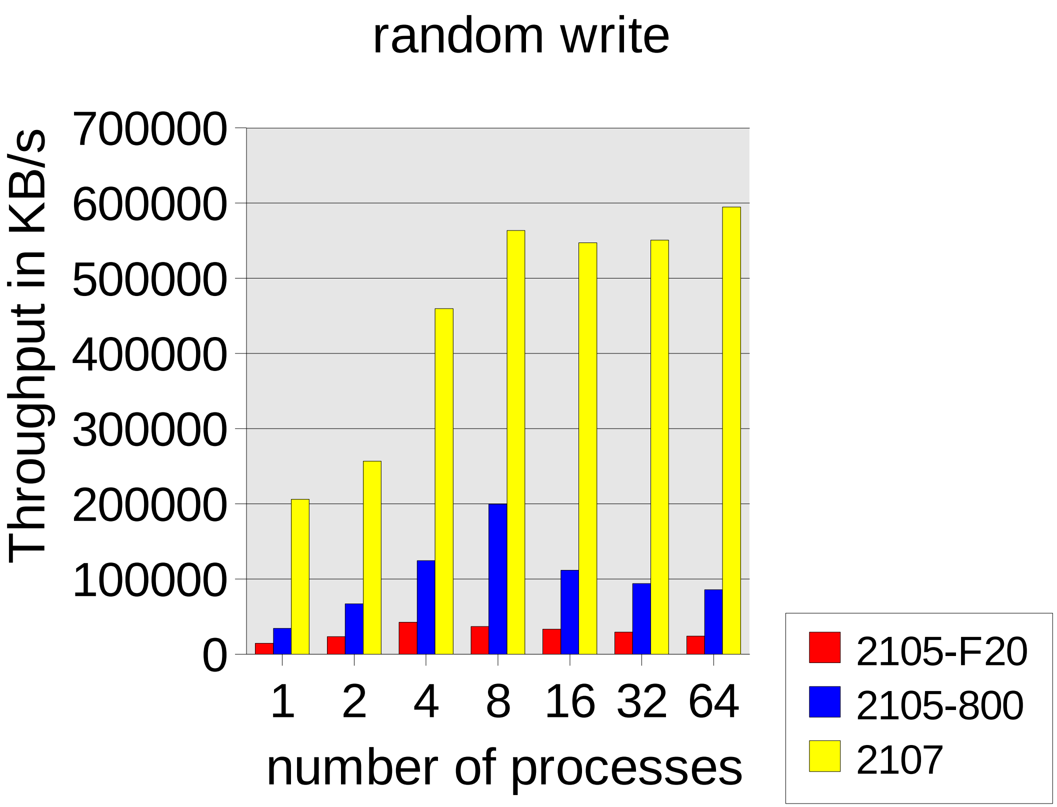
- **Advantage for 10 GbE over 1 GbE is increasing with data size**
- **Improvements up to 3.4x with streaming workload**

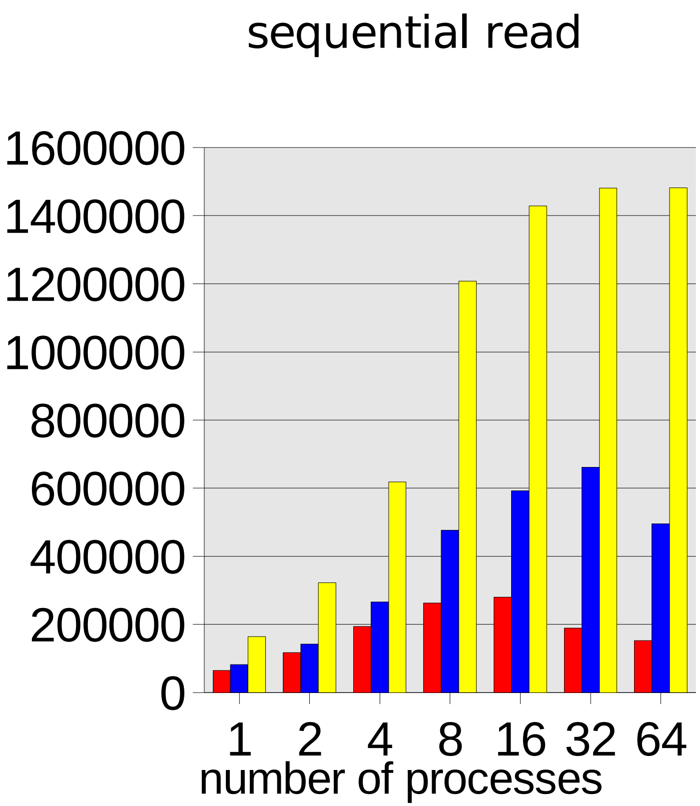
Disk I/O benchmark

- Iozone benchmark
- threaded file system benchmark used to measure synchronous I/O
- sequential/random write, rewrite, read of a large enough file (e.g. 700MB = almost 3x of memory size)
- main memory was restricted to 256MB
- 1, 2, 4, 8, 16, 32, 64 threads, each operating on its own disk or a Logical Volume with striping
- tests with ECKD and FCP/SCSI disks

Disk I/O performance on different storage servers

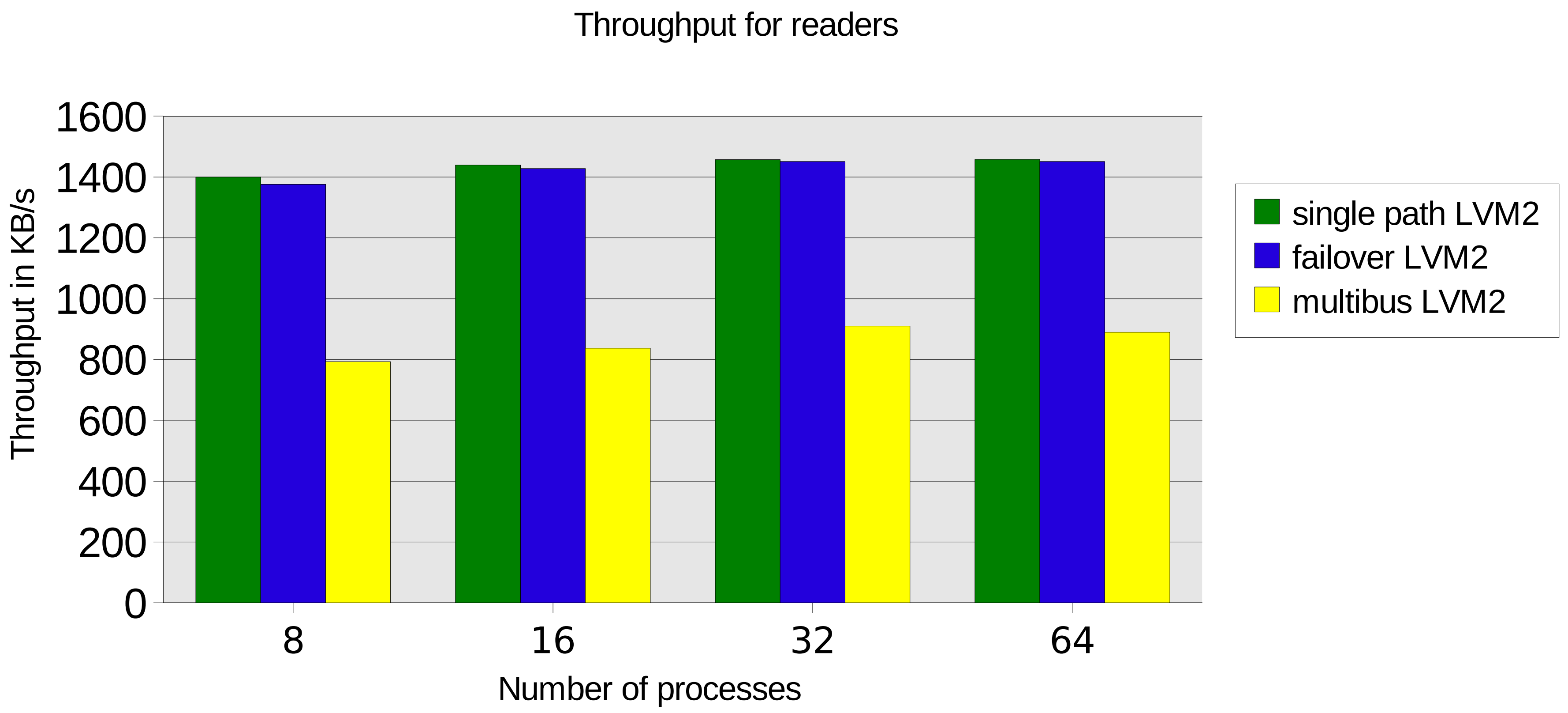
- **DS8300 is much faster than ESS800 and ESSF20**
- **examples for FCP/SCSI disks**





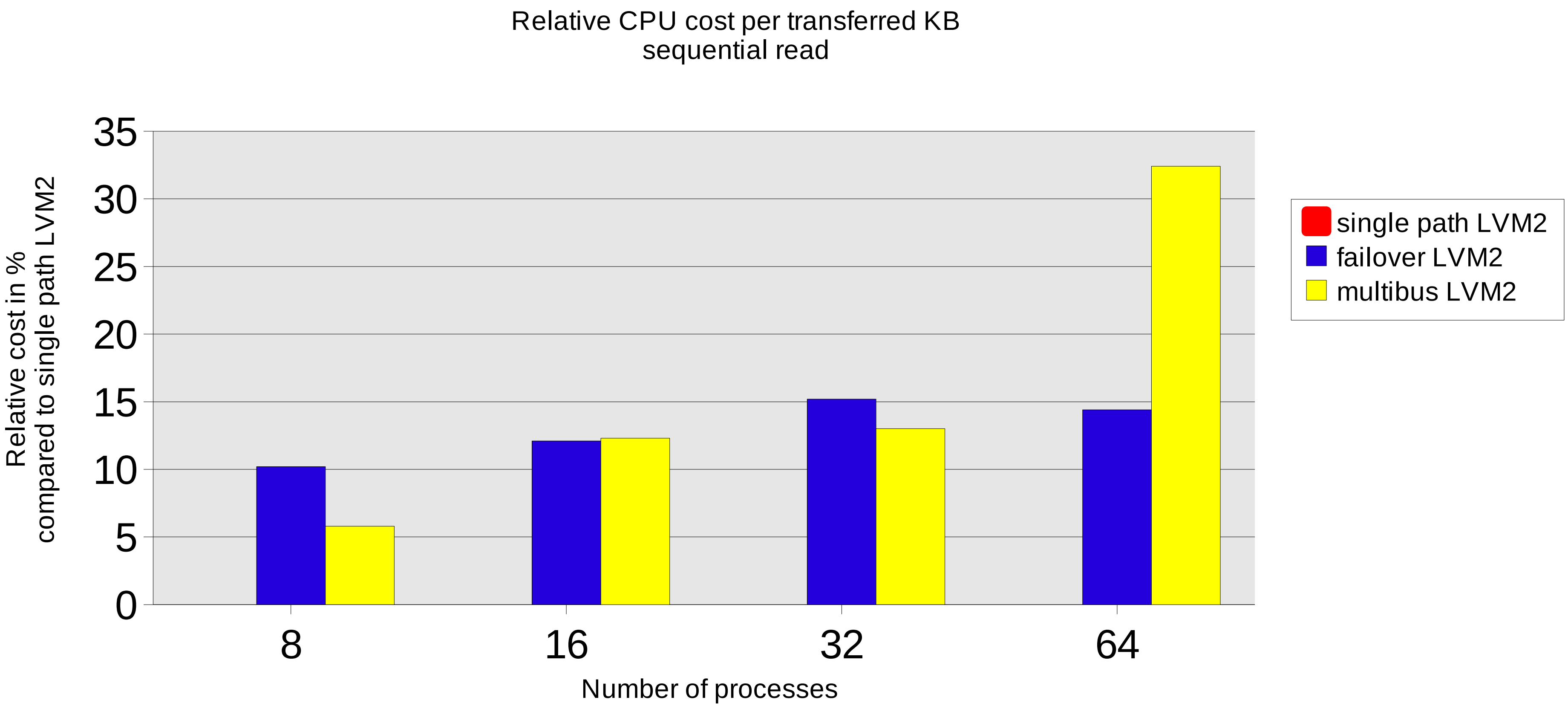
FCP/SCSI single path versus multipath (1)

- **use failover instead of multibus**



FCP/SCSI single path versus multipath (2)

- **costs for multipathing are about 10-15%**

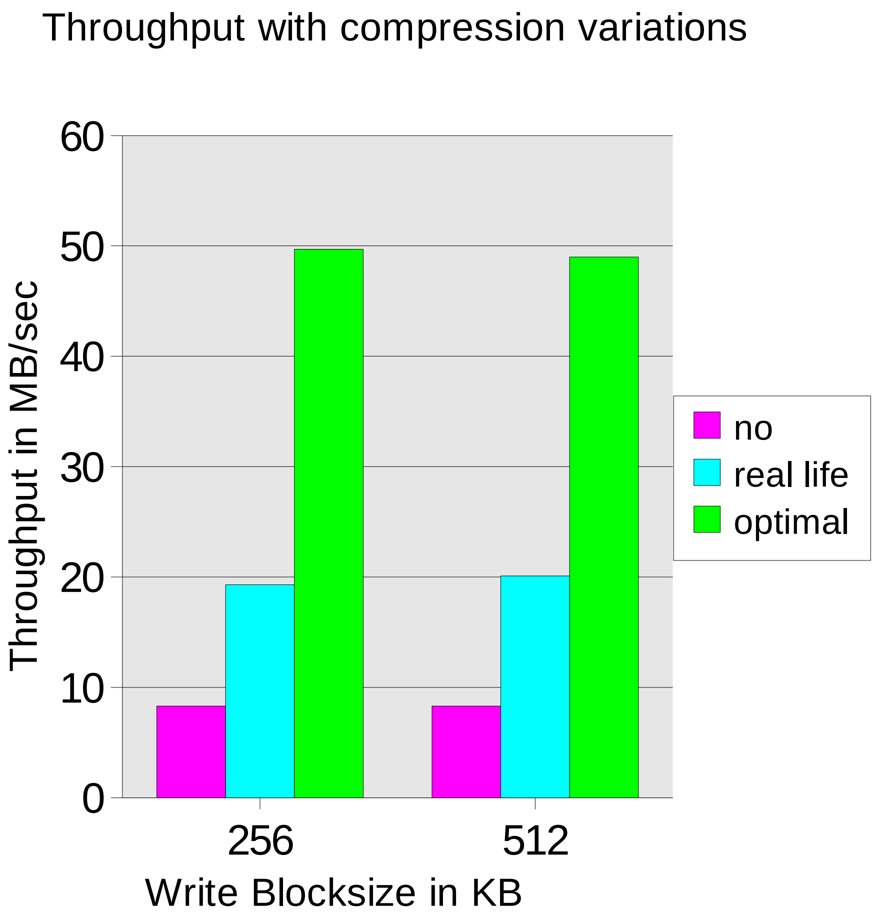


Disk I/O considerations

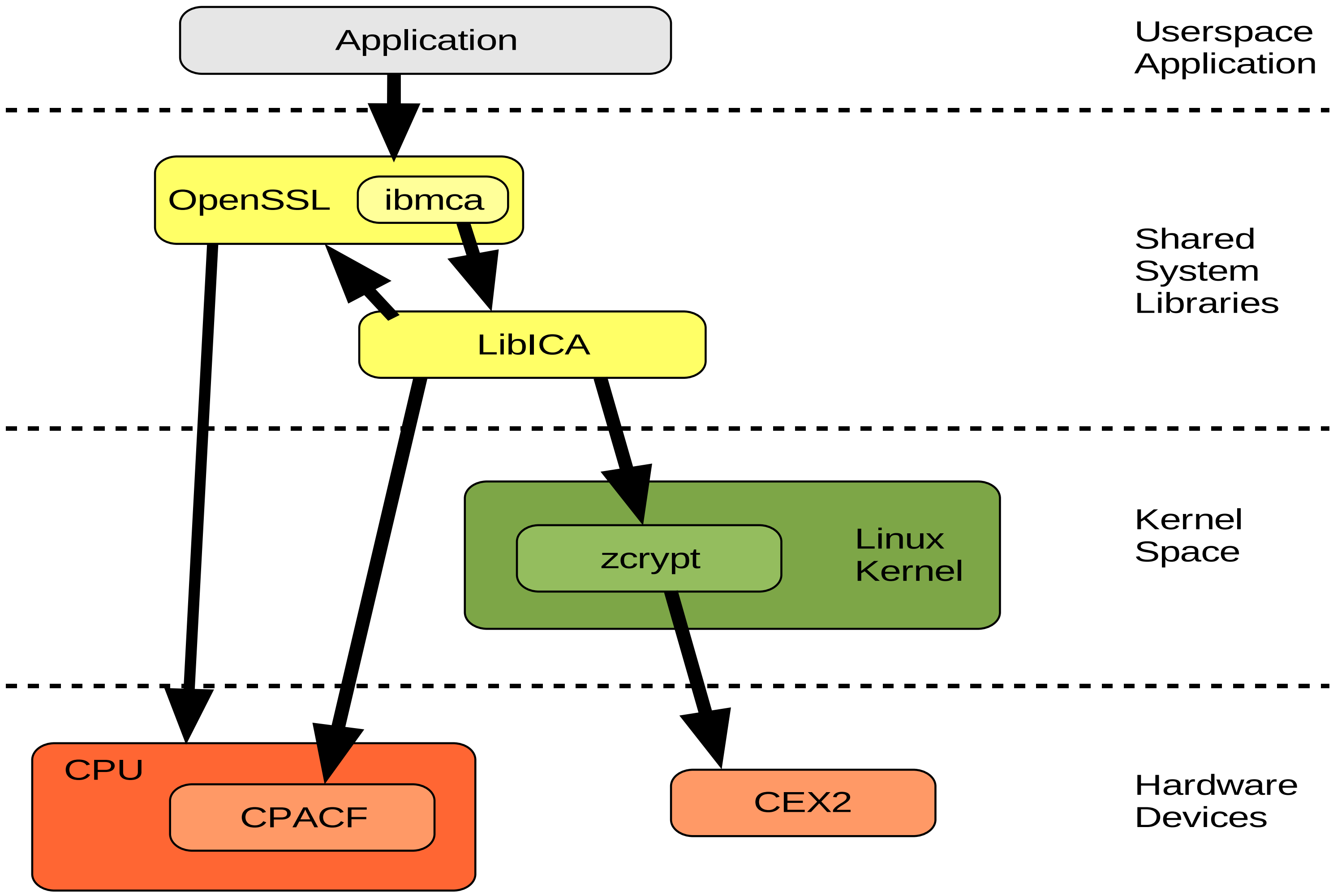
- **higher throughput rates with the new storage server generation require also higher CPU utilization**
- **this applies also to FCP/SCSI I/O when there is a throughput win versus FICON/ECKD I/O**
- **take care that any specific path assignments for FCP/SCSI disks are still valid after re-IPL – see HOWTO at**
  - **`www.ibm.com/developerworks/linux/linux390/perf/tuning_how_dasd_multipath.html`**

SCSI tape performance

- **measurements on IBM 3590 with optimal compression, compression of real life data (Linux source code), without compression**
- **tests were done with Linux dd command, 1 FCP channel to the tape unit**
- **select a large blocksize for the tape, e.g. 256 KB**

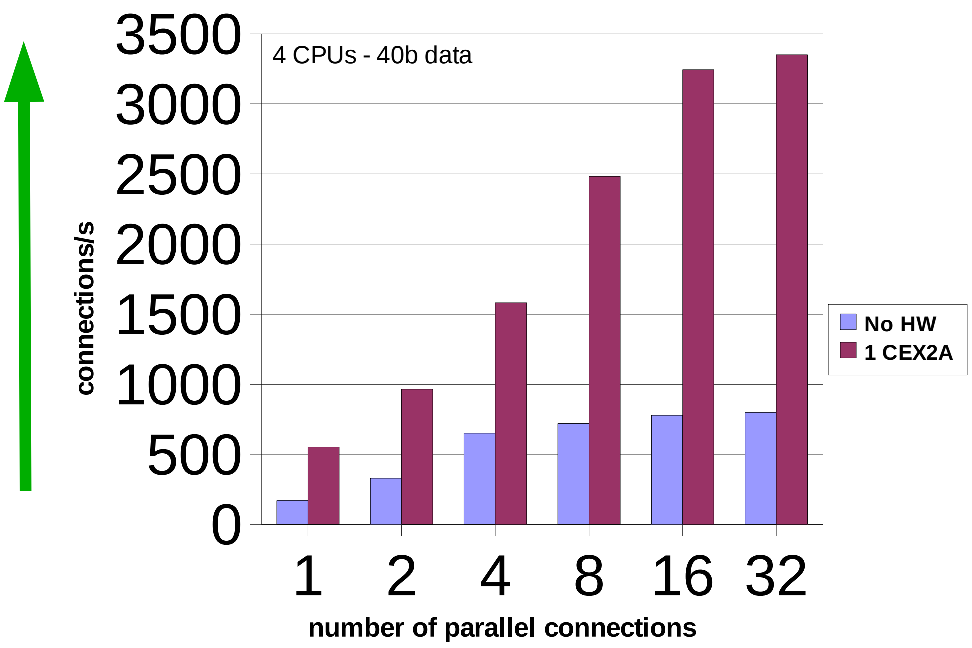


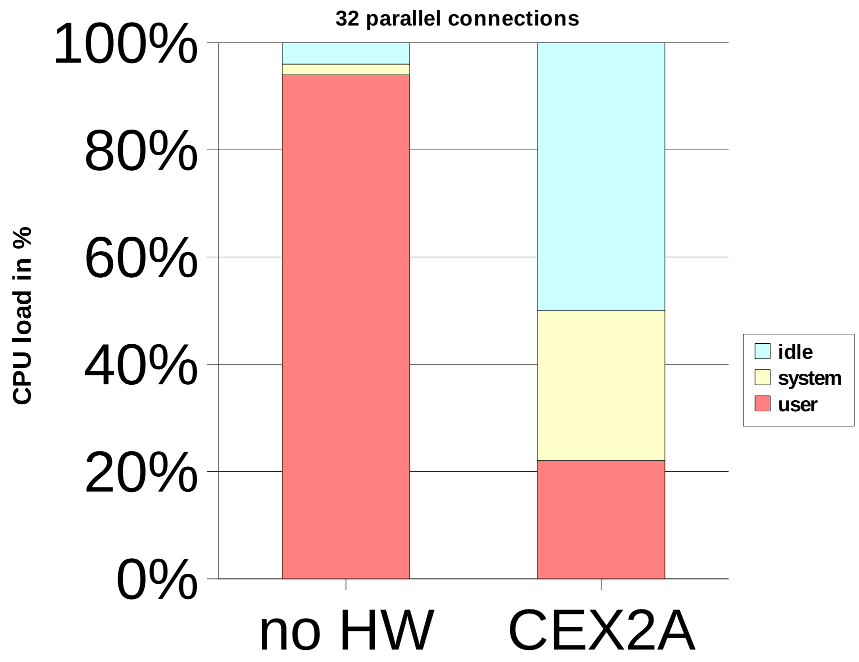
Cryptographic hardware support – SSL Stack



Crypto Express2 Accelerator (CEX2A) - SSL handshakes

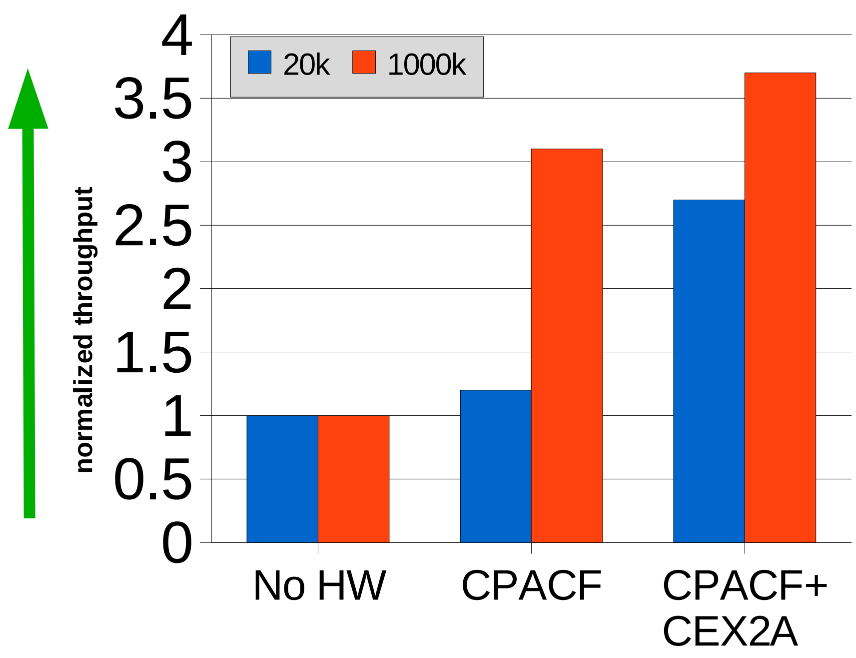
- **the number of SSL handshakes is up to 4x higher with HW support**
- **in the 32 connections case we save about 50% of the CPU resources**

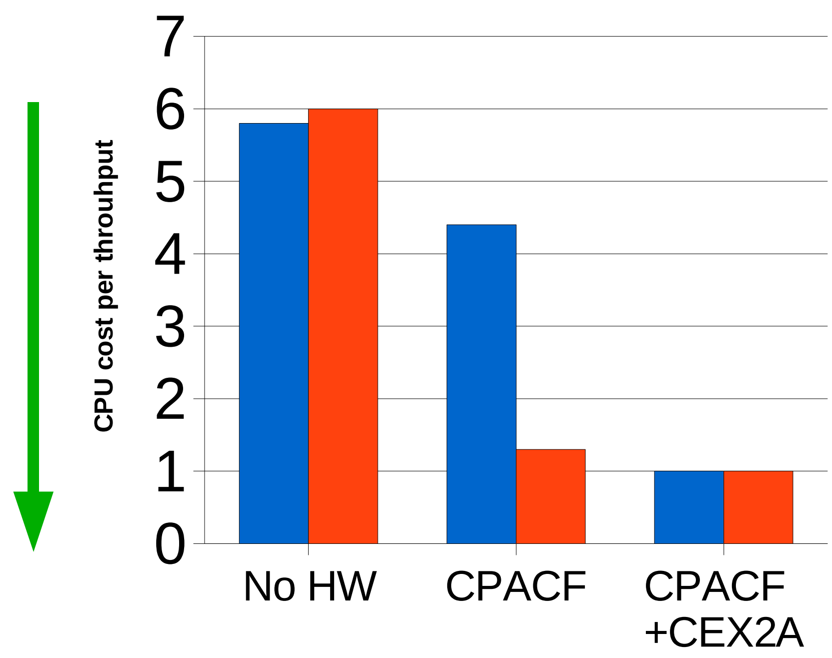




Crypto Express2 Accelerator (CEX2A) and CPACF

- **the use of both hardware features leads to 3.5x more throughput**
- **using software encryption costs about 6x more CPU**





Agenda

- System z hardware
- Hardware improvements
  - Processor
  - Networking
  - Disk / Tape
  - Cryptography
- **Software improvements**
  - **Compiler**
  - **Java**
  - **WebSEAL**
  - **Tivoli Storage Manager**
- **Distribution improvements**
  - **Red Hat**
  - **Novell SUSE**

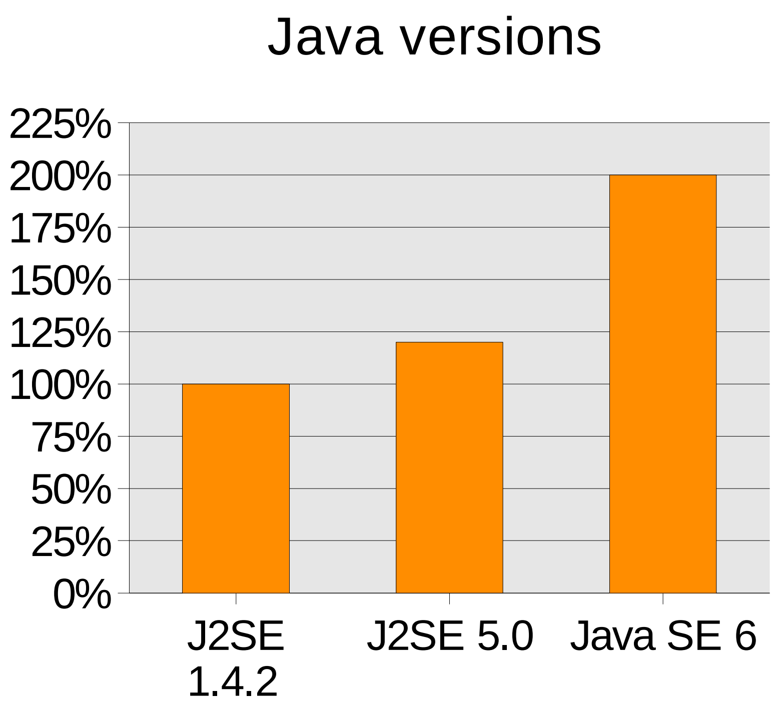
gcc 64bit compiler – SLES9 (gcc-3.3.3) vs. SLES10 (gcc-4.1.0)

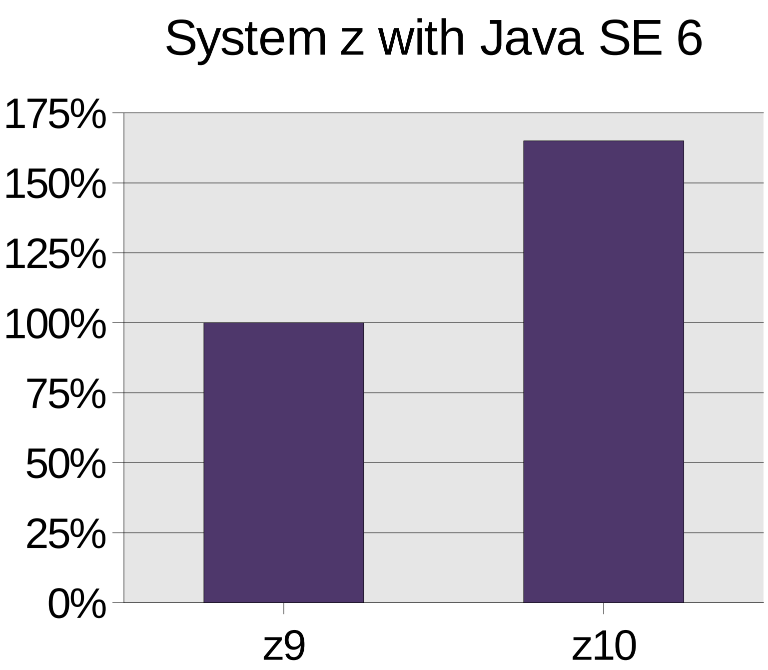
- **gcc 4.1 supports -mtune=z9-109 and -march=z9-109**

Compiler - why isn't 64-bit for free?

- **Hardware effects**
  - **primarily D-cache "pressure"**
    - **z/Architecture supports both 31-bit and 64-bit addressability**
      - **data cache is fixed size for machine**
      - **64-bit pointers "twice" as large as 31-bit pointers**
  - **also impacts I-cache performance**
    - **64-bit instructions tend to be 6-byte instead of 2 or 4**
- **Software effects**
  - **some 31-bit instructions have no 64-bit equivalent**
    - **must be implemented with series of 64-bit opcodes**
    - **= additional path length for same function**
  - **increased cost for entry/exit linkage**
    - **registers are twice as wide**

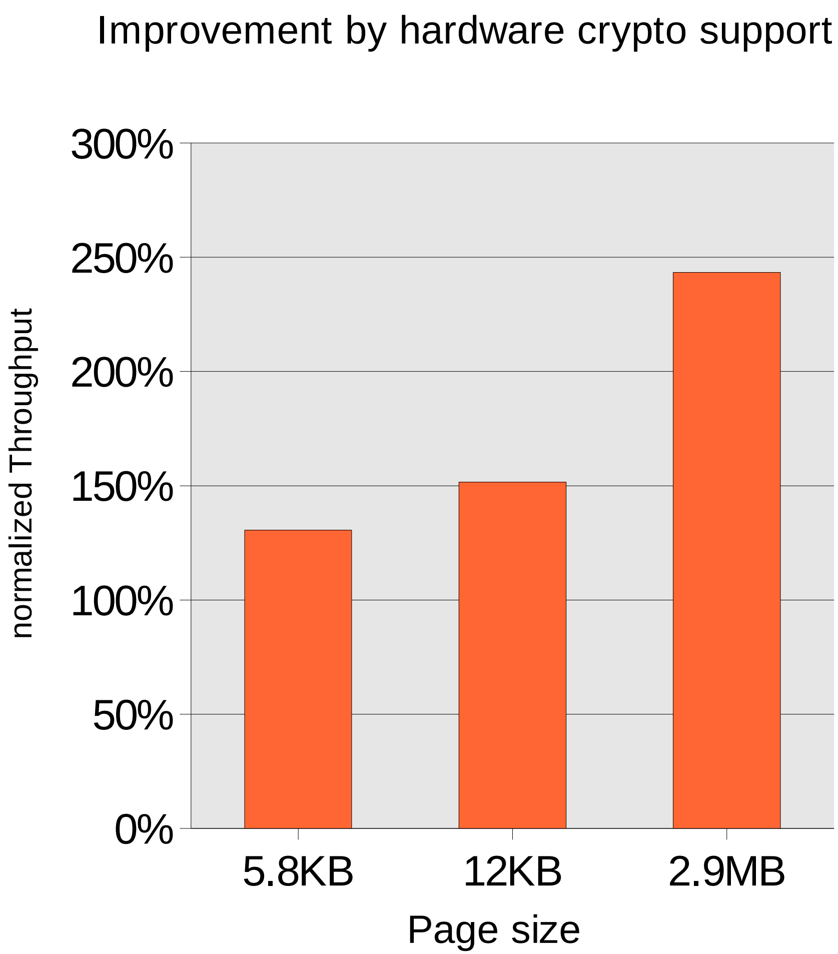
Java Results 64-bit





- **Java improvements through newer JVM and JIT**
- **improvements through new hardware**
- **64-bit Java is production ready**

Crypto performance – WebSEAL SSL access



- **Websphere AppServer on z/OS**
- **WebSEAL running on Linux System z using SSL with AES-128**
- **scaling the size of the requested page**
- **uses mostly CPACF**
- **improvement up to factor 2.4 for hardware encryption versus software encryption**

Agenda

- System z hardware
- Hardware improvements
  - Processor
  - Networking
  - Disk / Tape
  - Cryptography
- **Software improvements**
  - **Compiler**
  - **Java**
  - **WebSEAL**
  - **Tivoli Storage Manager**
- **Distribution improvements**
  - **Red Hat**
  - **Novell SUSE**

Comparison SLES10 / RHEL5

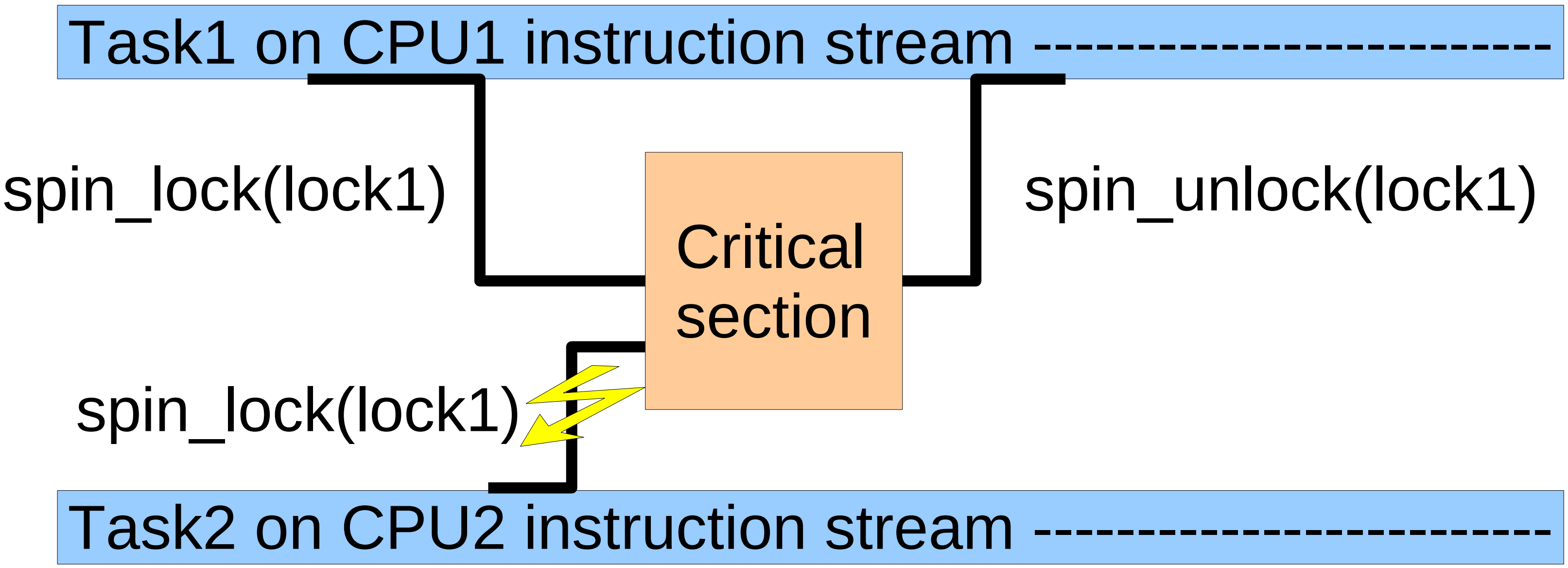
measurement portfolio SLES10 SP1 vs. RHEL5 U1	LPAR 64	LPAR 31 (emu)	z/VM 64	z/VM 31 (emu)
Scaling	equal	n/a	n/a	n/a
File serving ECKD	equal	n/a	equal	n/a
File serving SCSI	equal	n/a	equal	n/a
Kernel	SLES advantages	n/a	SLES advantages	SLES advantages
Compiler INT	equal	n/a	n/a	n/a
Compiler FP	equal	n/a	n/a	n/a
Seq. I/O ECKD	RHEL advantages	n/a	RHEL advantages	n/a
Seq. I/O SCSI	RHEL advantages	n/a	RHEL advantages	n/a
Rnd I/O ECKD	RHEL advantages	n/a	equal	n/a
Rnd I/O SCSI	RHEL advantages	n/a	RHEL advantages	n/a
Network 1000Base-T QDIO	equal	n/a	equal	n/a
Network 1GbE QDIO	equal	n/a	equal	n/a
Network 10GbE QDIO	equal	n/a	equal	n/a
Network HiperSockets	equal	n/a	equal	n/a
Java	equal	equal	n/a	n/a

Legend	*n/a*	*SLES advantages*	*equal*	*RHEL advantages*

SLES / RHEL improved resource usage

- Linux kernel uses spinlocks to wait for exclusive use of kernel resources
- on System z it is not desirable to use processors for active waiting
- The older solution issues a DIAG 44 instruction to the LPAR or z/VM hypervisor to give the CPU back instead of looping for the lock. This allowed more useful work to be done.
- 2 new features:
  - spin_retry counter in Linux to avoid excessive use of diagnose instructions
  - use of DIAG 9C instruction to pass information along with the instruction, who should get the processor

Avoiding spin locks on System z



Task2: Spin for the lock(non z platforms)

DIAG 44: return control to hipervisor

Spin(count) + DIAG 44

Spin(count) + DIAG 9C CPU1

SLES10 / RHEL5 virtual CPU time accounting

- the standard Linux implementation is based on a heuristic model with a 10 ms timer interrupt
  - the complete time slice is accounted to the interrupted context
- on systems with virtual CPUs this approach is too inaccurate
- the new implementation is based on the System z virtual timer
  - CPU times get now accounted whenever the execution context changes
  - a new category of Linux wait state is showing, how often the Linux system is waiting for CPU (current sysstat version required)
  - the feature is enabled by default in SLES10 and RHEL5

Linux command 'top' – the snapshot tool

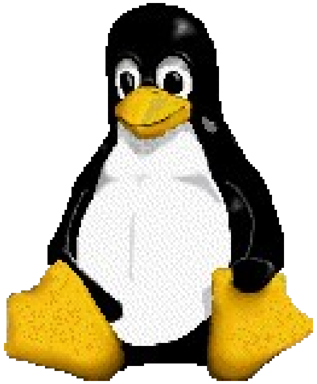
- **adds new field "CPU steal time"**
  - **is the time Linux wanted to run on a processor, but the hipervisor was not able to schedule CPU**
  - **is included in SLES10 and RHEL5**

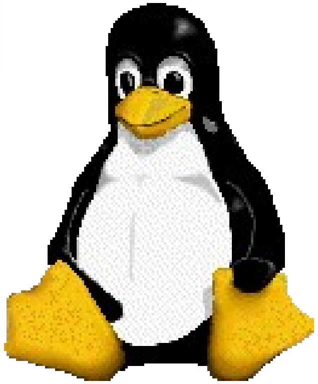
```
top - 09:50:20 up 11 min,  3 users,  load average: 8.94, 7.17, 3.82
Tasks:  78 total,   8 running,  70 sleeping,   0 stopped,   0 zombie
 Cpu0 : 38.7%us,  4.2%sy,  0.0%ni,  0.0%id,  2.4%wa,  1.8%hi,  0.0%si, 53.0%st
 Cpu1 : 38.5%us,  0.6%sy,  0.0%ni,  5.1%id,  1.3%wa,  1.9%hi,  0.0%si, 52.6%st
 Cpu2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

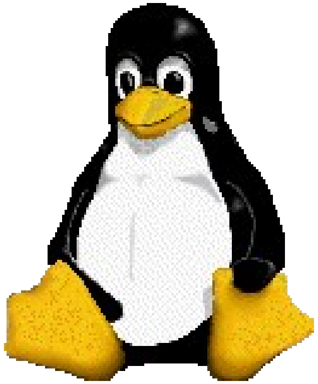
Visit us !

- **Linux on zSeries Tuning Hints and Tips**
  `http://www.ibm.com/developerworks/linux/linux390/perf/index.html`
- **Linux-z/VM Performance Website**
  `http://www.vm.ibm.com/perf/tips/linuxper.html`

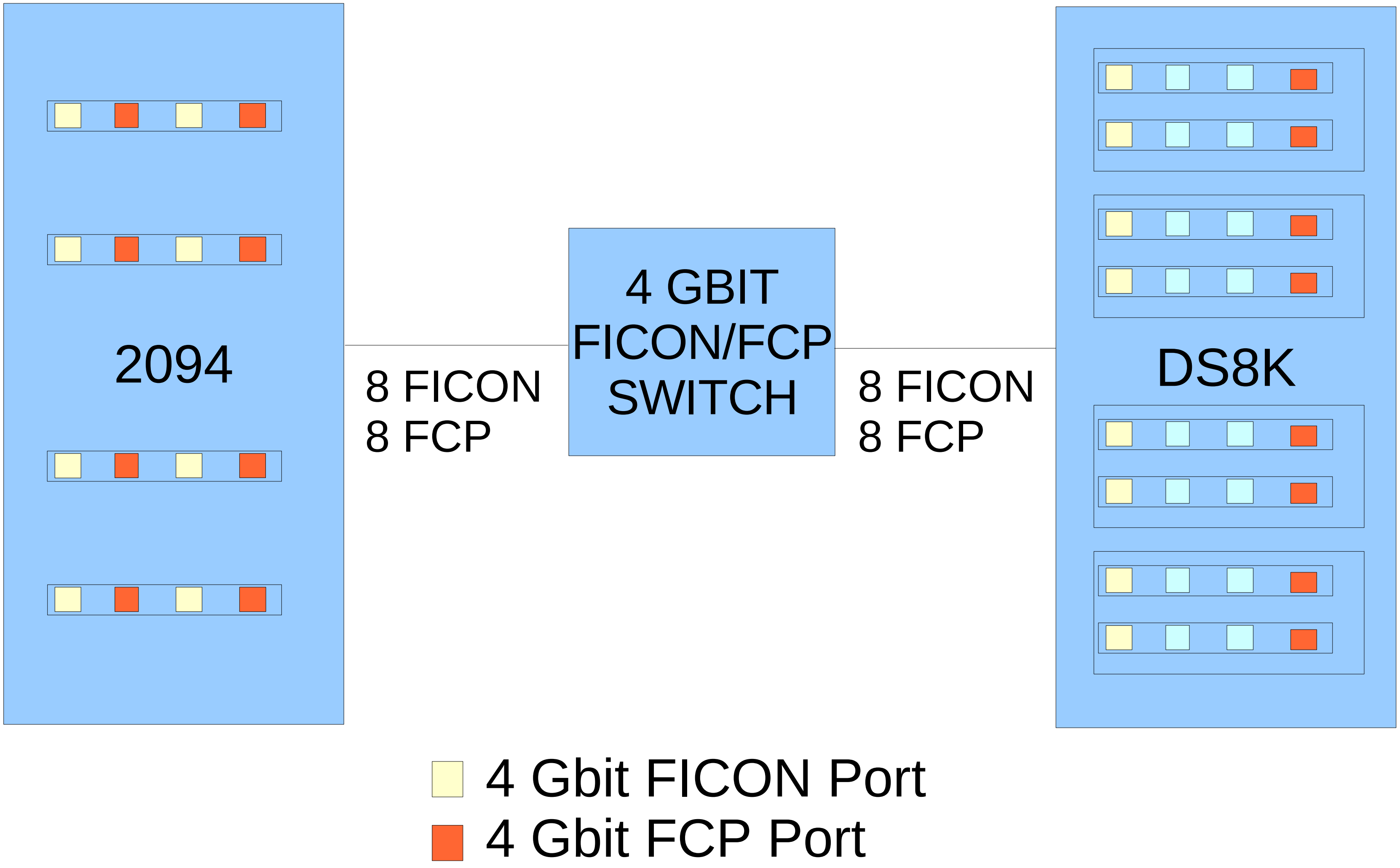
Questions





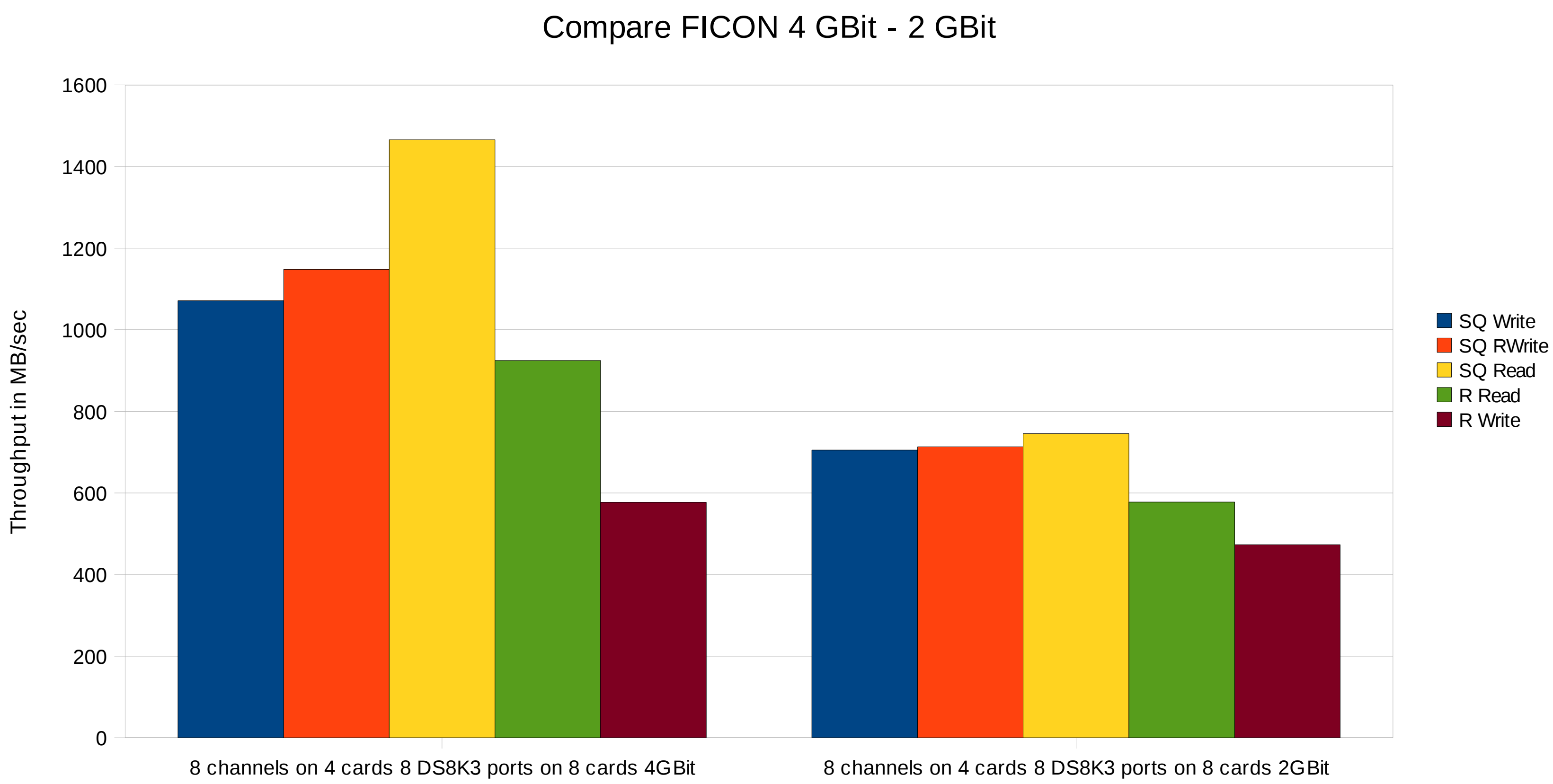


Configuration for 4Gbps disk I/O measurements



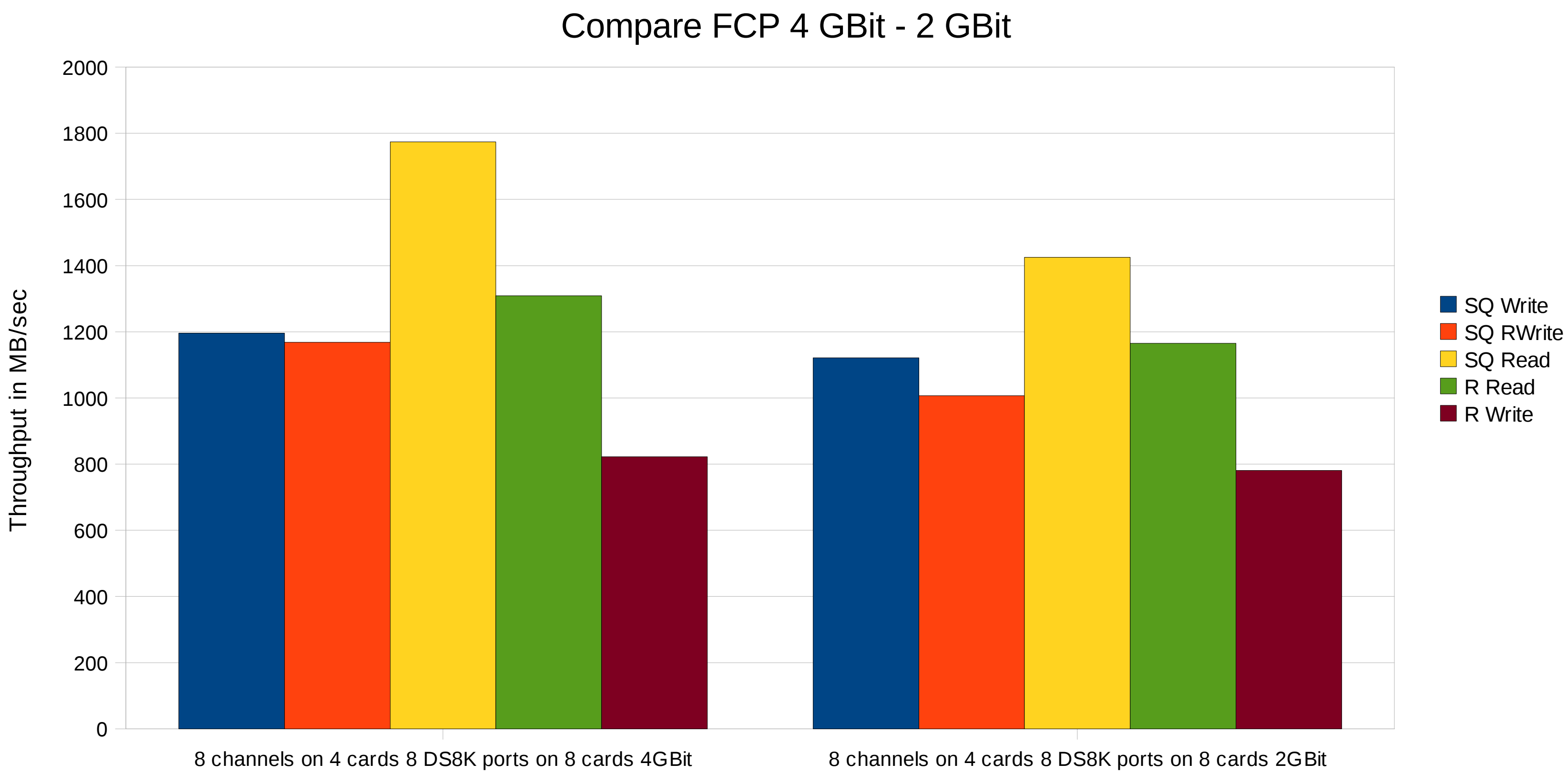
Disk I/O performance with 4Gbps links - FICON

- **strong throughput increase (average 1.6x)**
- **the best increase is with sequential read at 2x**



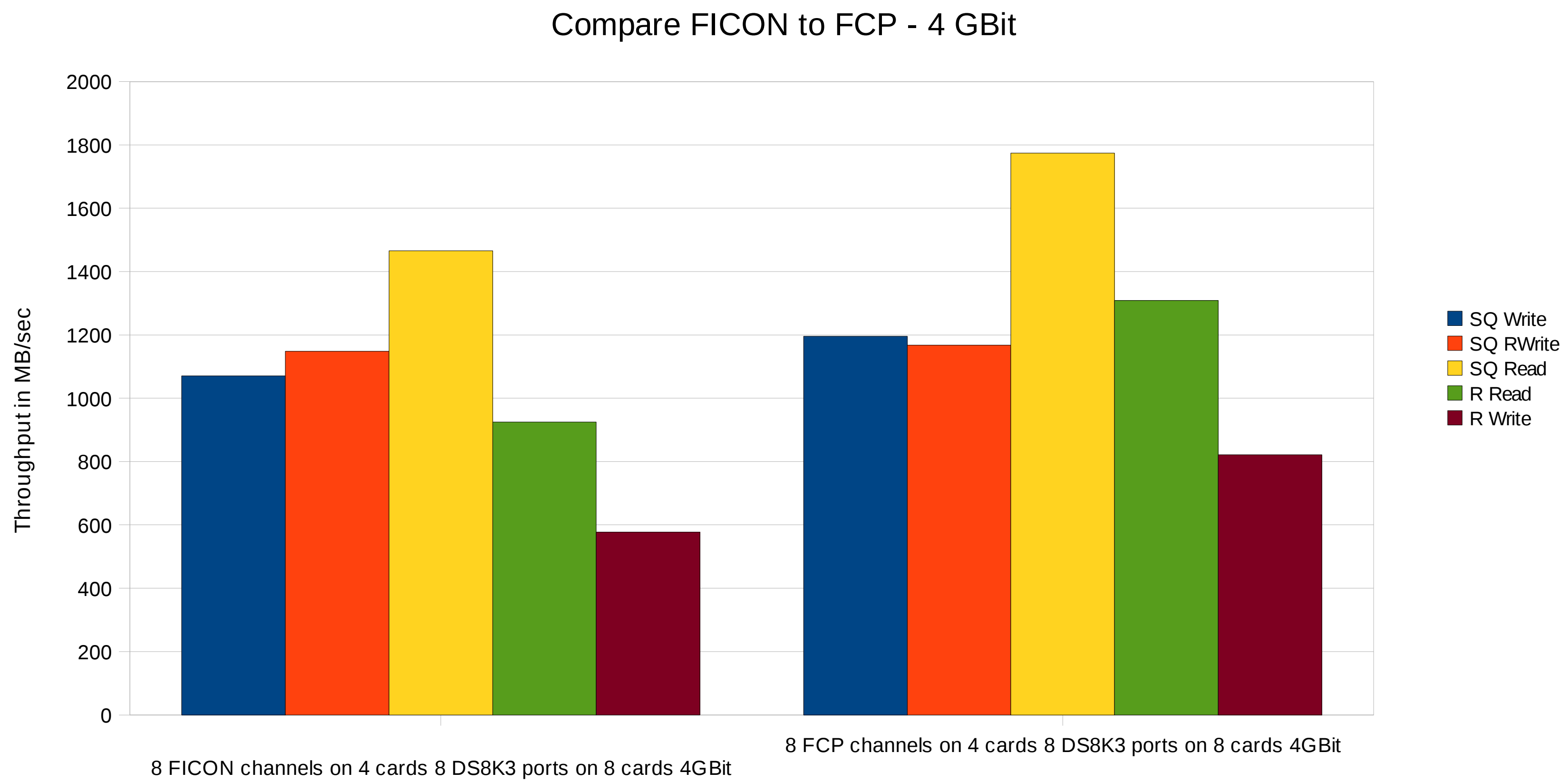
Disk I/O performance with 4Gbps links - FCP

- **moderate throughput increase**
- **best improvement with sequential read at 1.25x**



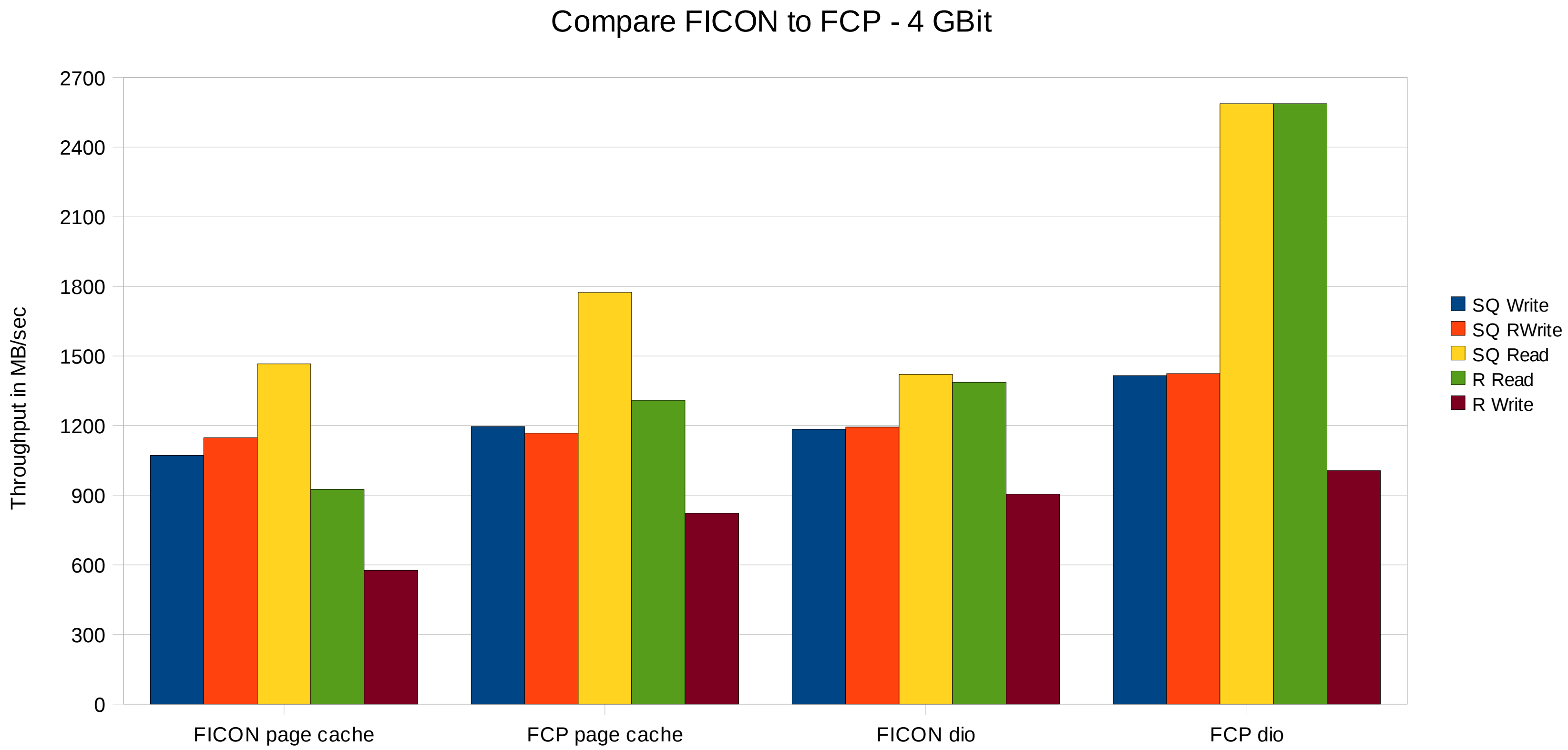
Disk I/O performance with 4Gbps links – FICON versus FCP

- **throughput for sequential write is similar**
- **FCP throughput for random I/O is 40% higher**



Disk I/O performance with 4Gbps links – FICON versus FCP / direct I/O

- **bypassing the Linux page cache improves throughput for FCP up to 2x, for FICON up to 1.6x.**
- **read operations are much faster on FCP**



Special study with Tivoli Storage Manager

- **ECKD versus SCSI**
- **Configured and measured on our system together with TSM performance specialist**
- **Entry statement from TSM, based on their tests for backing up 70 GB data:**
  - ***"execution time with SCSI is 25% shorter than with ECKD"***
  - ***"average CPU consumption with SCSI is 67% more than with ECKD"***
- **Common exit statement after the tests:**
  - ***"execution time with SCSI is 50% shorter than with ECKD"***
  - ***"costs were almost equal (more CPU resources need to be provided for SCSI)"***