SHARE

Technology · Connections · Results

End to end performance of WebSphere environments with Linux on System z

Session 9291

Martin Kammerer
kammerer@de.ibm.com

Feb 26, 2008 4:30 - 5:30

© Disney

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

- WebSphere Base Environment
- Network (LPAR)
- Network (z/VM)
- Java setup
- Database
- Tuning Results
  - Dynamic Cache
  - Database Setup
- 31-bit versus 64-bit
- Cryptographic hardware support

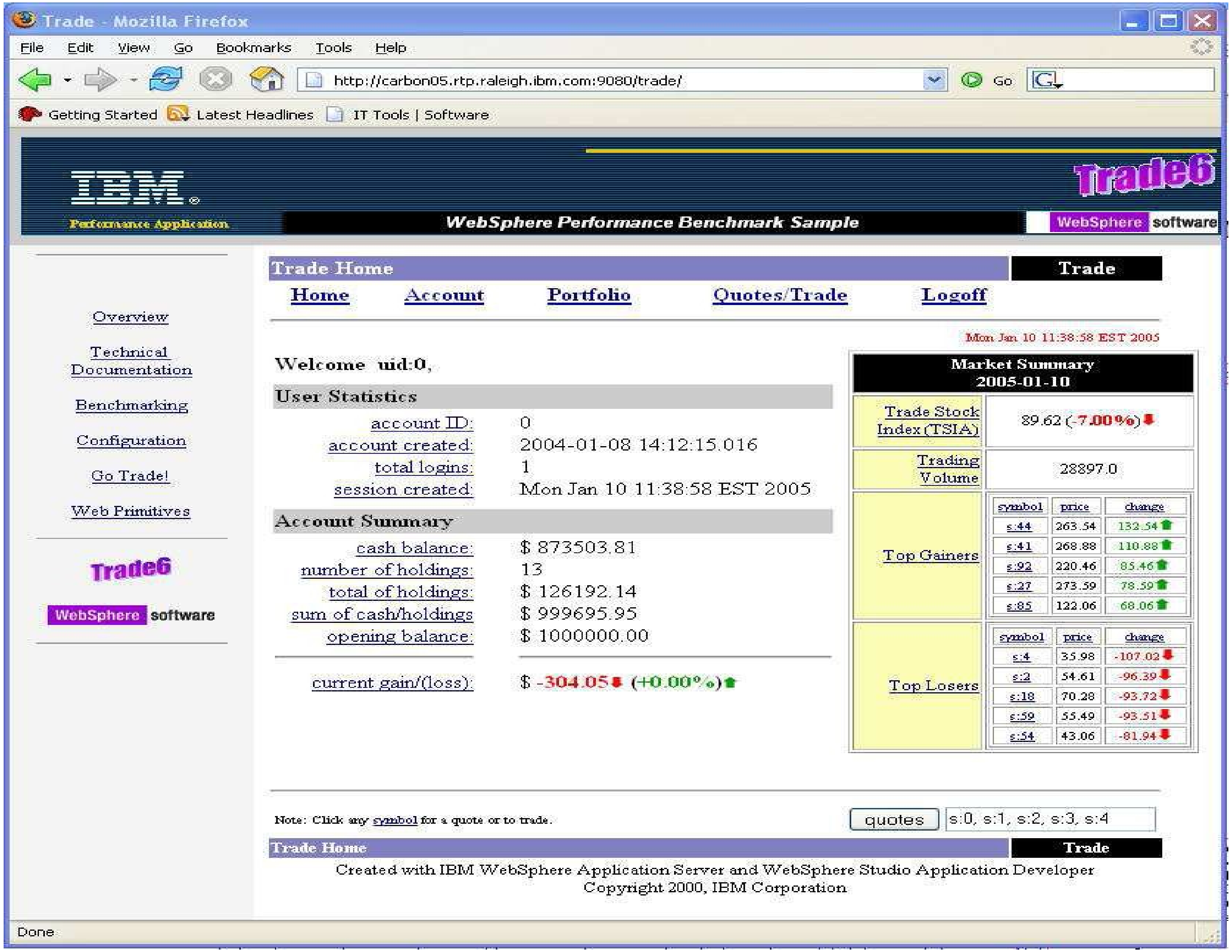
Performance tuning at all layers

- “Optimize your stack from the top to the bottom”
  - Application design
  - Application setup
  - ***Database***
  - ***Application server***
  - ***Operating system***
  - ***Virtualization system***
  - ***Hardware***

Trade workload

- By IBM - is designed to cover the programming model and performance technologies with WebSphere Application Server.
- Supports environments with DB2 and Oracle databases
- Supports newest J2EE programming models (WAS releases)
- Models an electronic stock brokerage providing Web based online securities trading
- Provides a real world business application mix of operations
- Client / server scenario

Trade workload (2)



Mon Jan 10 11:38:58 EST 2005

Welcome uid:0,

User Statistics

account ID:	0
account created:	2004-01-08 14:12:15.016
total logins:	1
session created:	Mon Jan 10 11:38:58 EST 2005

Account Summary

cash balance:	$ 873503.81
number of holdings:	13
total of holdings:	$ 126192.14
sum of cash/holdings	$ 999695.95
opening balance:	$ 1000000.00
current gain/(loss):	$ -304.05 ⬇ (+0.00%) ⬆

Market Summary
2005-01-10

Trade Stock Index (TSIA)	89.62 (-7.00%) ⬇
Trading Volume	28897.0

Top Gainers

symbol	price	change
s:44	263.54	132.54 ⬆
s:41	268.88	110.88 ⬆
s:92	220.46	85.46 ⬆
s:27	273.59	78.59 ⬆
s:85	122.06	68.06 ⬆

Top Losers

symbol	price	change
s:4	35.98	-107.02 ⬇
s:2	54.61	-96.39 ⬇
s:18	70.28	-93.72 ⬇
s:59	55.49	-93.51 ⬇
s:54	43.06	-81.94 ⬇

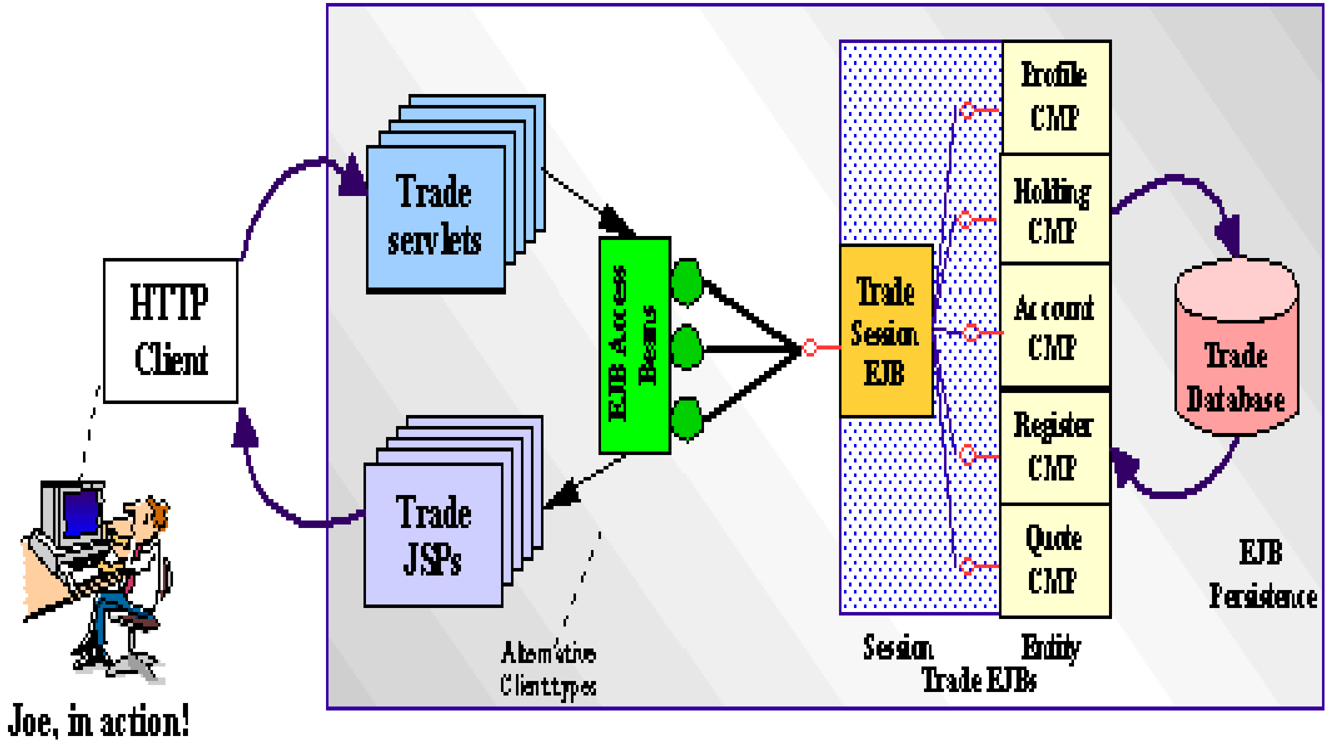
Note: Click any symbol for a quote or to trade.

s:0, s:1, s:2, s:3, s:4

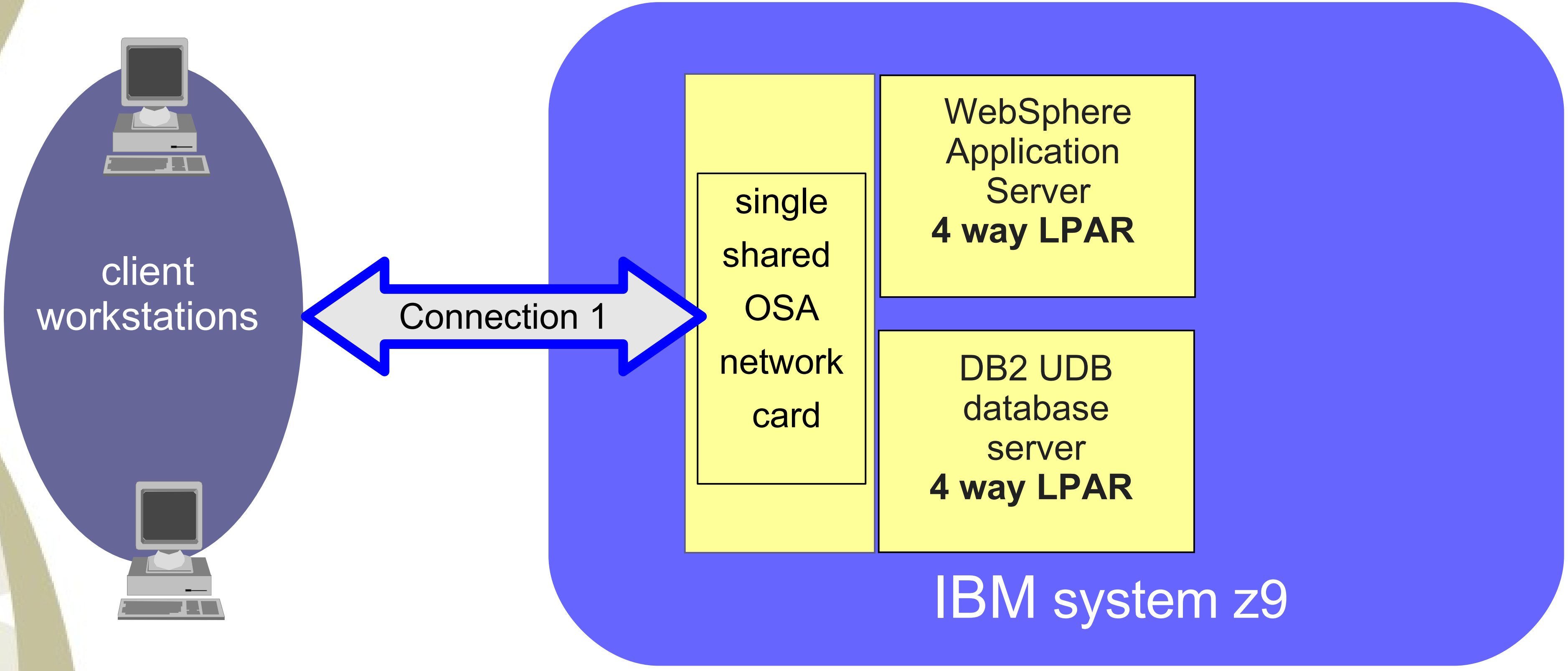
Created with IBM WebSphere Application Server and WebSphere Studio Application Developer
Copyright 2000, IBM Corporation

workload

- The Trade application models an online brokerage firm providing web based services such as login, buy, sell, get quote and more.

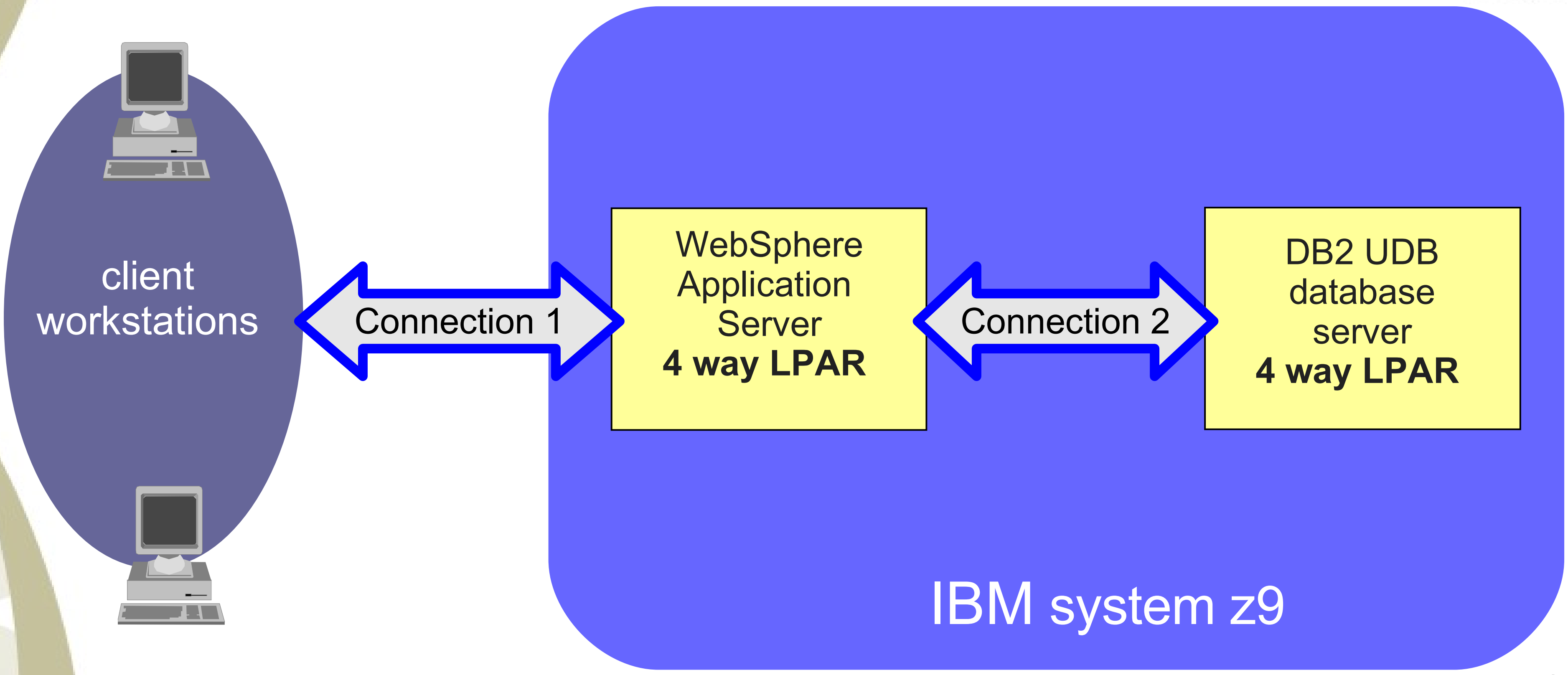


WebSphere base environment (LPAR)



- let's start with a simple setup
- when increasing the load, the first bottleneck was the single shared network connection

Network constraints – base environment



- first tuning step: separate the connection to the database (2nd OSA card)
  improvement +10%
- second step: use Hipersockets for connection 2
  improvement +33%

Network constraints - monitoring

- monitor with `sar -n DEV [interval] [count]`
- Some maximum values observed with benchmark workloads with OSA express2 cards and Hipersockets

	small requests	large requests	Throughput for large packages in one direction		
	pkg/sec recv or send	pkg/sec recv or send	MTU 32K	MTU 1492	MTU 8992
1GEth	35,000	82,000	--	120 Mbyte/sec	120 Mbyte/sec
10GEth	40,000	85,000	--	120 Mbyte/sec	400 Mbyte/sec
Hipersockets	120,000	107,000	1 GByte/sec	--	--

- The scenario described before would exceed 50,000 packages/sec when sharing a single OSA card
  - the traffic from all systems using the card needs to be added!

Network constraints – setup changes

- Choose your MTU size carefully!
  - Avoid fragmentation, lots of small packages can drive up CPU utilization
  - Use the largest MTU size supported in the path, and **verify** it using

```
ping -M do system15.ibm.com  -s 8000 -c3
PING system15.ibm.com 8000(8028) bytes of data.
From dyn-9-152-198-41.ibm.com icmp_seq=0 Frag needed and DF set (mtu =
1500)
```

- For really busy network devices consider to
  - Increase the number of inbound buffers in the qeth driver (default 16)
    - Device has to be offline

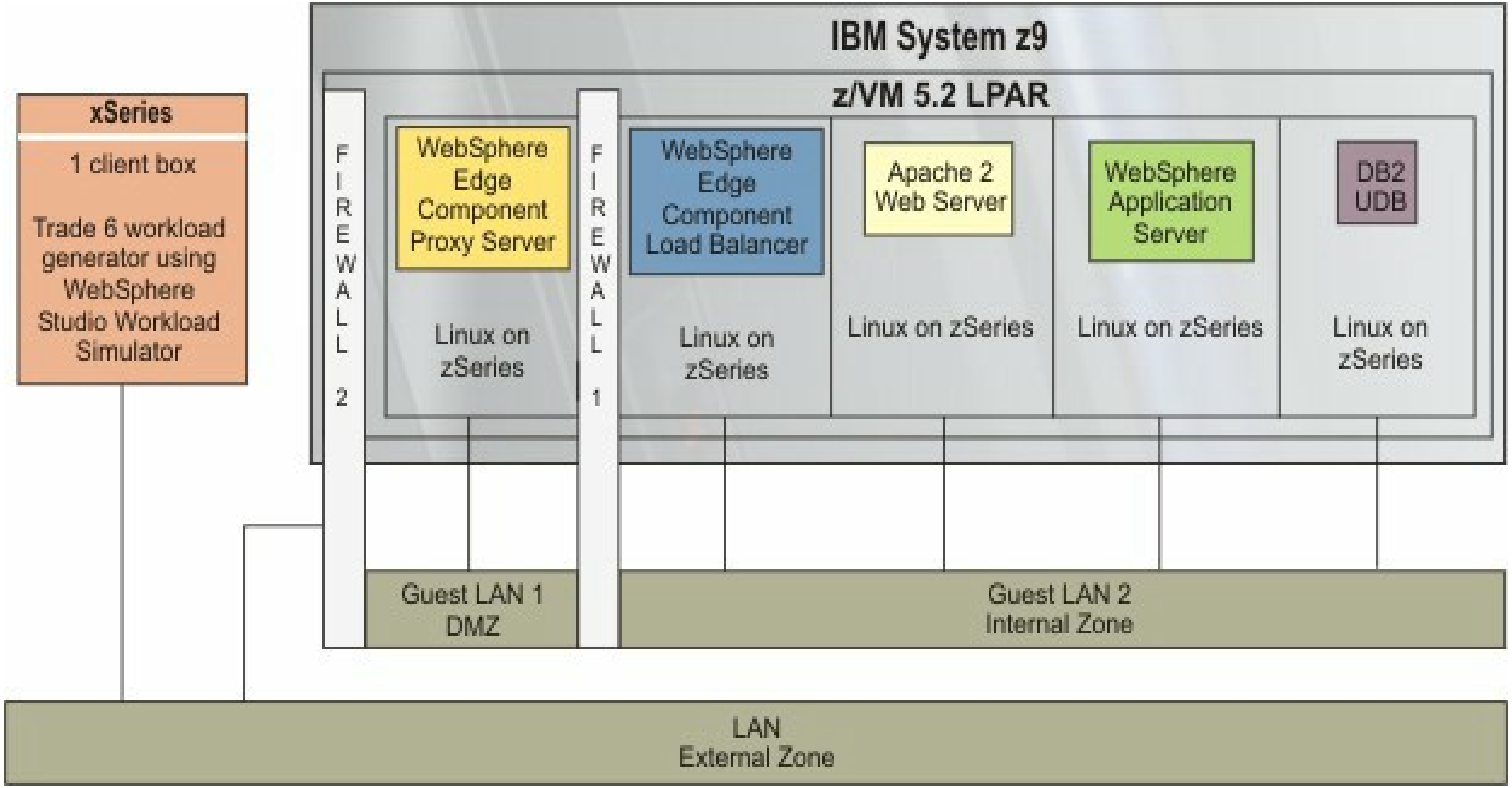
      ```
      echo <number> >
      /sys/bus/ccwgroup/drivers/qeth/<device_bus_id>/buffer_count
      ```

    - or for a SUSE distribution:
      add following line to `/etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.<nnnn>`

      ```
      QETH_OPTIONS="buffer_count=128 checksumming=hw_checksumming"
      ```

    - Consumes memory!
      - *64KB per buffer, maximum 128 buffer = 8 MB per device*
      - *for tuning purpose, start with a large value, monitor the impact and then iterative reduce the number of buffers until throughput drops down*
  - Use channel bonding

WebSphere typical environment



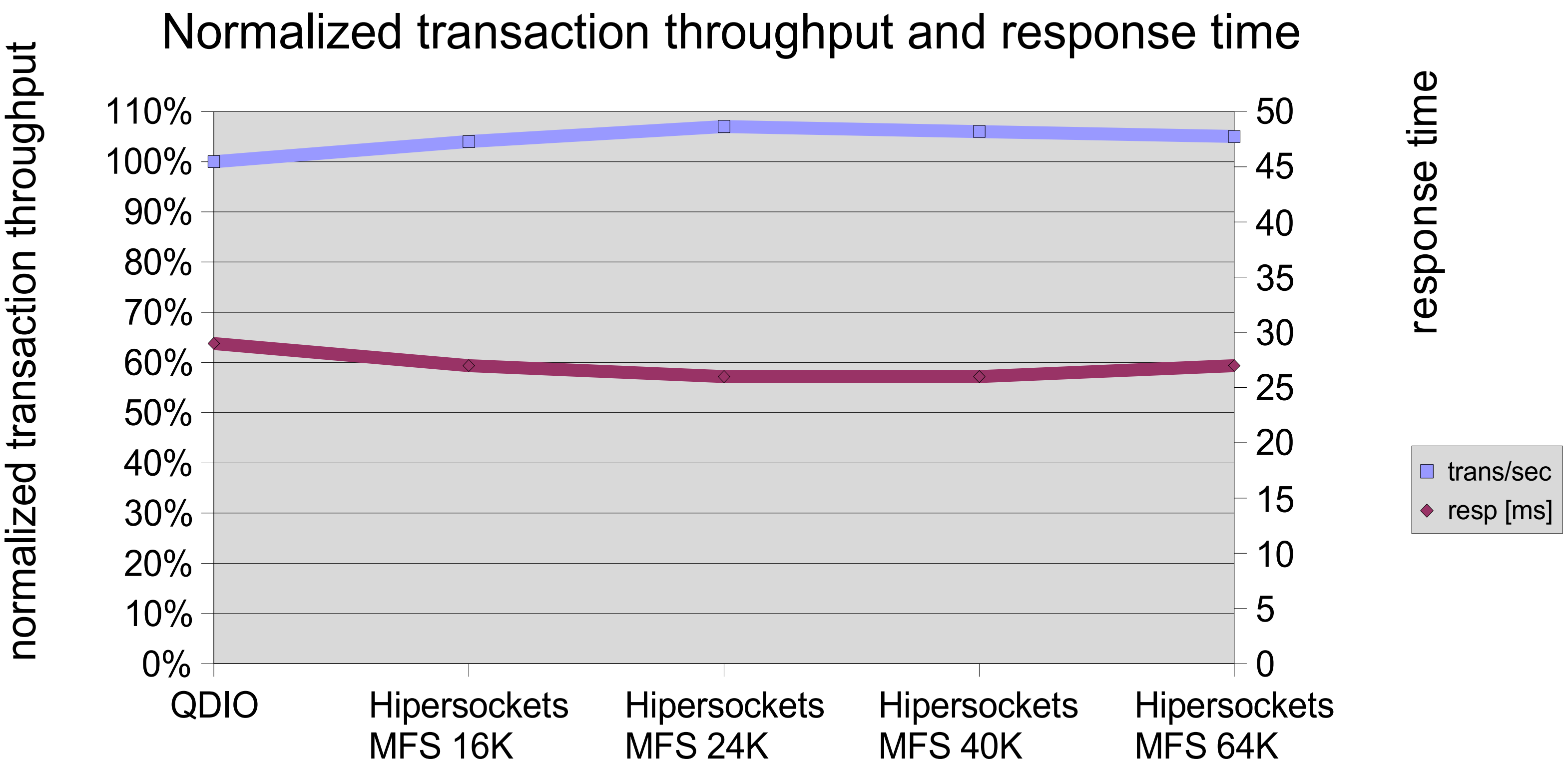
- The application server needs to be protected with a DMZ
- Easy to implement under z/VM using a guest LAN
  - this environment could also be extended to a cluster

Networking – Connection types

- Which connectivity to use:
  - inside z/VM use for guest to guest communication
    - VSWITCH without an OSA card
    - Guest LAN (no layer 2 support)
  - to another LPAR inside the same System z
    - use Hipersockets
      Hipersockets are completely driven by CPU
  - External connectivity:
    - Use new 10 GbE cards with MTU 8992
    - VSWITCH with an OSA card
    - Attach OSA directly to Linux guest image

Networking under z/VM: guest LAN

Network type and Maximal Frame Size for Hipersockets



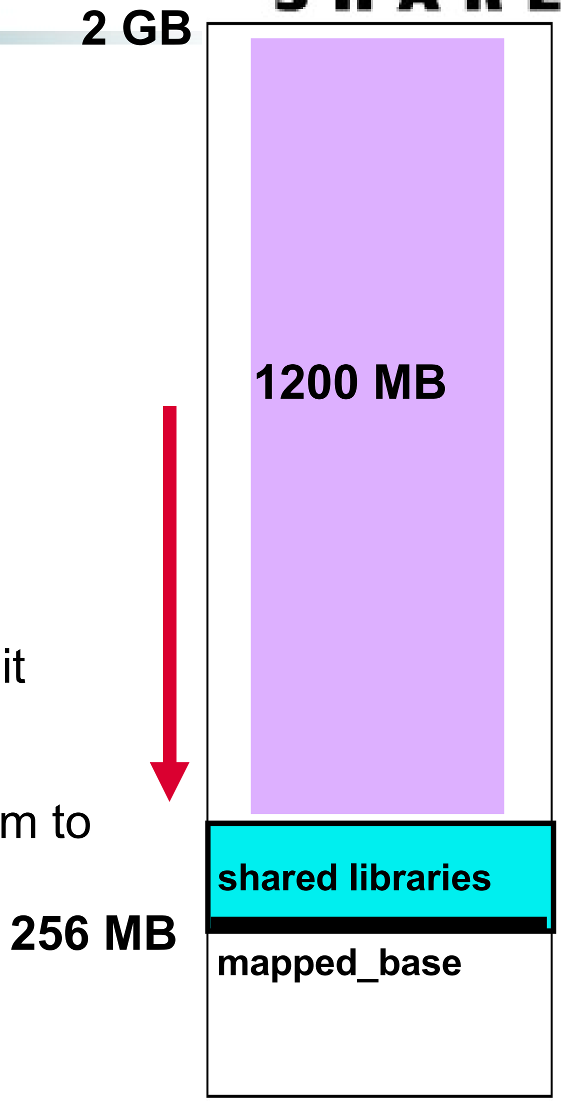
- guest LAN type Hipersockets with a MFS of 24K can be recommended because of higher throughput at lower latencies

Java setup - general

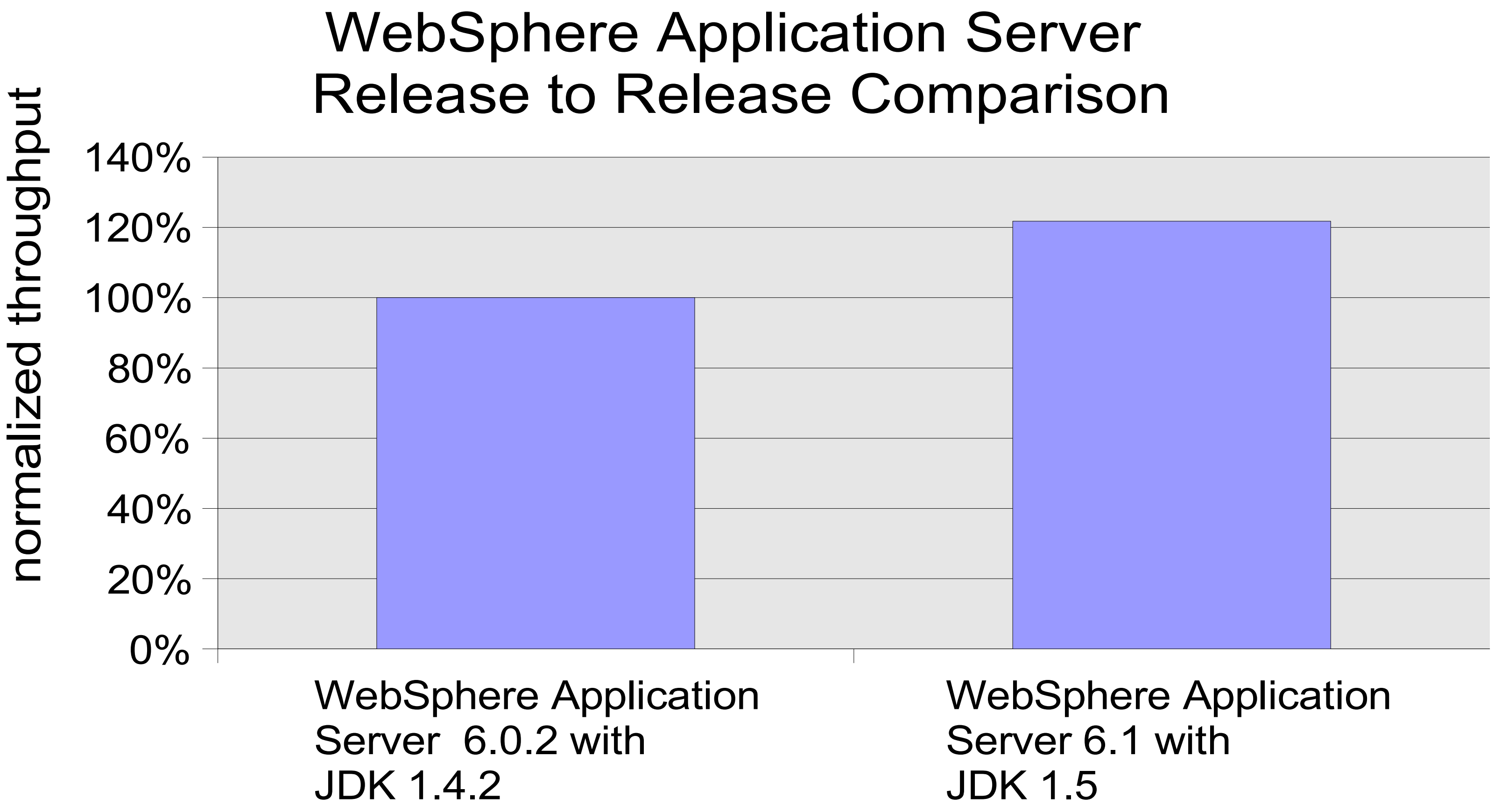
- Assure that the JIT is enabled (java -version)

- increase the heap size
  - Setting heap size: -Xms(minimal), -Xmx(maximal), use min=max, avoids fragmentation
  - Larger heap size implies better performance
  - Avoid swapping!

- Special consideration for 31 bit distributions
  - to define a heap inside the memory up to 1.2 GB in 31bit SLES8, SLES9 use:
    ```
    echo 268435456 >/proc/<pid>/mapped_base
    ```
  - In 31bit RHEL4 environments use flex-mmap mechanism to get a larger heap size, but watch out for prelinked applications!
    - modify /etc/sysconfig/prelink
      ```
      set PRELINKING=no prelink -ua
      ```
    - run /etc/cron.daily/prelink
    - reboot



WebSphere / Java evolution



- WebSphere Application Server 6.1 got a 20% improvement
- **Use the latest WebSphere / Java combination if possible!**

Networking – connection to the database

- Use recent versions of database connectors
  - Type 4 JDBC connectors have a performance advantage of about 10% compared to JDBC type 2 over DB2 Connect
- Monitor the connection pool (number of physical connections to the database). Set the “Maximum pool size” of to a value that there are always some inactive connections
- Keep the latencies in the network communication between the WebSphere server and the database short
  - Use a fast network connection which can handle easily the traffic
  - low number of network hops between the application server and database

Networking – DB2 database on z/OS

- Set the right maximum number of physical connections in the database
  - Set the DSNZPARM parameter **CONDBAT** to the sum of the “maximum pool size” of the all the WebSphere Application servers you use with the database and all other applications
  - Set the DSNZPARM parameter **MDBAT** to the maximum acceptable number of active DBATs (= active connections).
  - Monitor with -dis ddf

    ```
    DSNL080I -DB91 DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
    DSNL081I STATUS=STARTD DSNL082I LOCATION LUNAME GENERICLU
    DSNL083I DB91ZOS USIBMT6.DB91ZOS -NONE DSNL084I TCPPORT=446
    SECPORT=0 RESPORT=447 IPNAME=-NONE DSNL085I
    IPADDR=::9.12.22.95 DSNL086I SQL
    DOMAIN=lndia3.pdl.pok.ibm.com DSNL086I RESYNC
    DOMAIN=lndia3.pdl.pok.ibm.com DSNL090I DT=I CONDBAT= 10000
    MDBAT= 1000 DSNL092I ADBAT= 198 QUEDBAT= 0 INADBAT= 0
    CONQUED= 0 DSNL093I DSCDBAT= 85 INACONN= 320 DSNL099I
    DSNLTDDF DISPLAY DDF REPORT COMPLETE
    ```

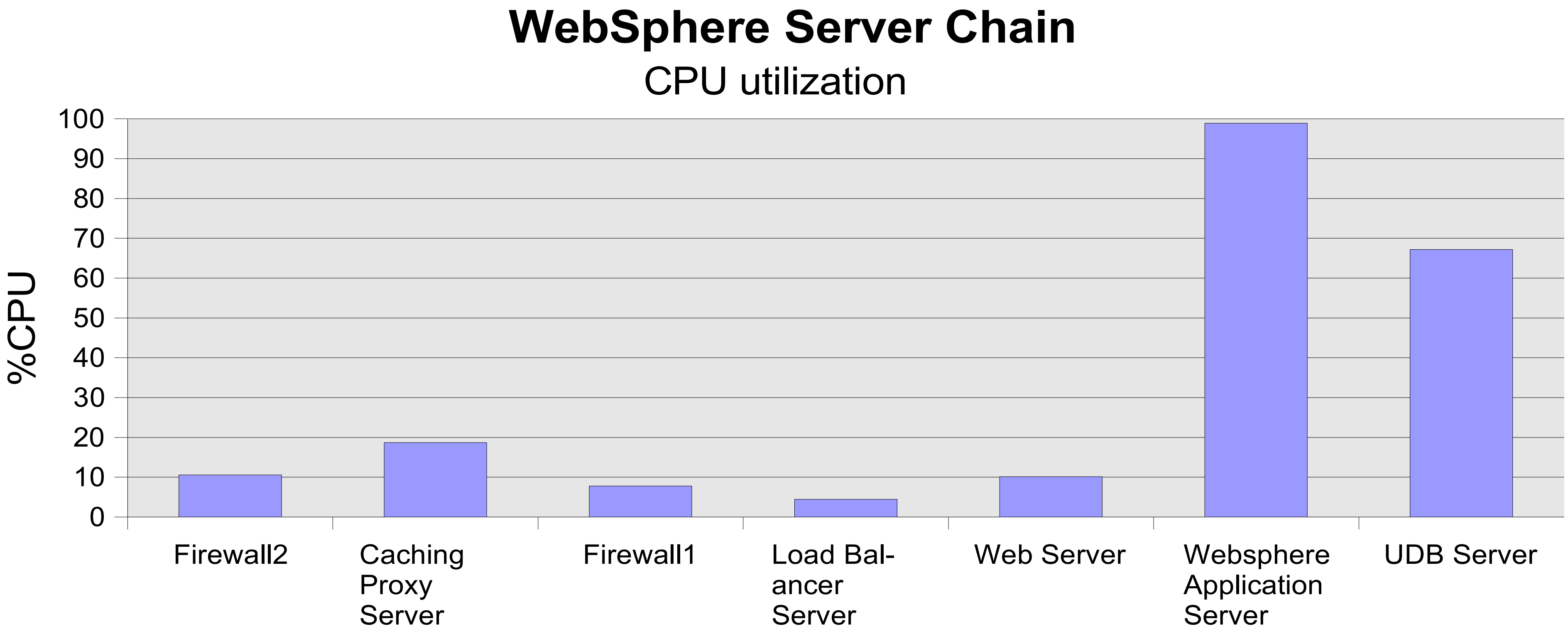
- when ADBAT exceeds MDBAT then new or inactive connections must be queued

Database on z/OS

On z/OS define proper WLM policies

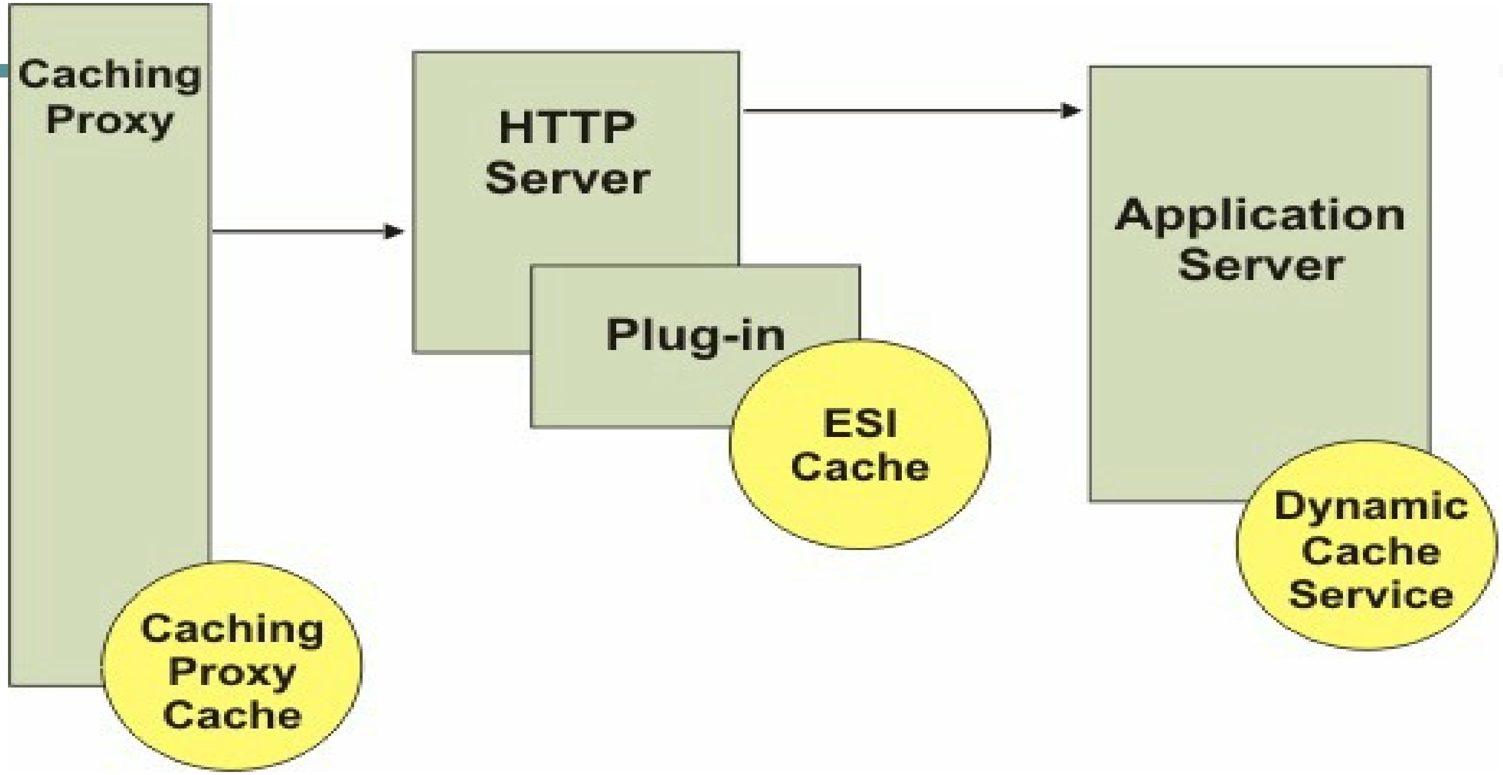
- SYSSTC Built in service class.
  Used for DB91IRLM. High priority service class. Only ’SYSTEM’ service class is higher.
- DB2ADDRS Service class for DB91MSTR, DB91DBM1, and DB91DIST.
  - Uses importance=1, velocity=80.
    Slightly lower than the IRLM address space.
- DDFWORK Service class for DDF.
  - Uses importance=2, velocity=80.
    Slightly lower priority than the DB2 address spaces.

Identify bottlenecks in the environment



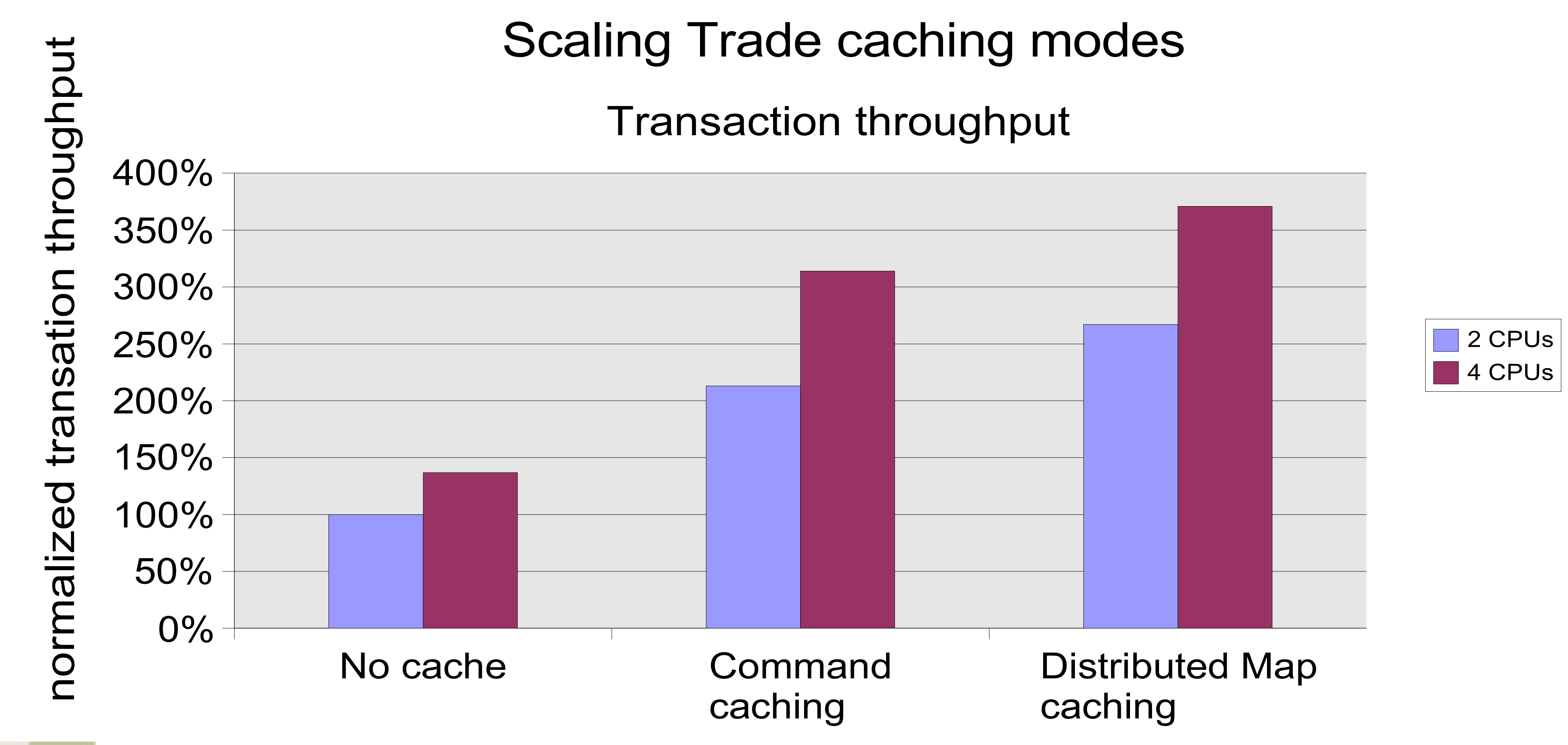
- Tuning should start with the bottleneck
  - WebSphere application server and UDB server in the example environment
  - don't run the WebSphere Application Server permanently over 90% CPU utilization

The caches of WebSphere servers



- Caching Proxy Cache for static content
  - ServerConnPool value ON allows reuse of existing sockets
  - ServerConnTimeout is used to limit the network idle time
- Dynamic Cache services of the application server and ESI cache can be used for dynamic content

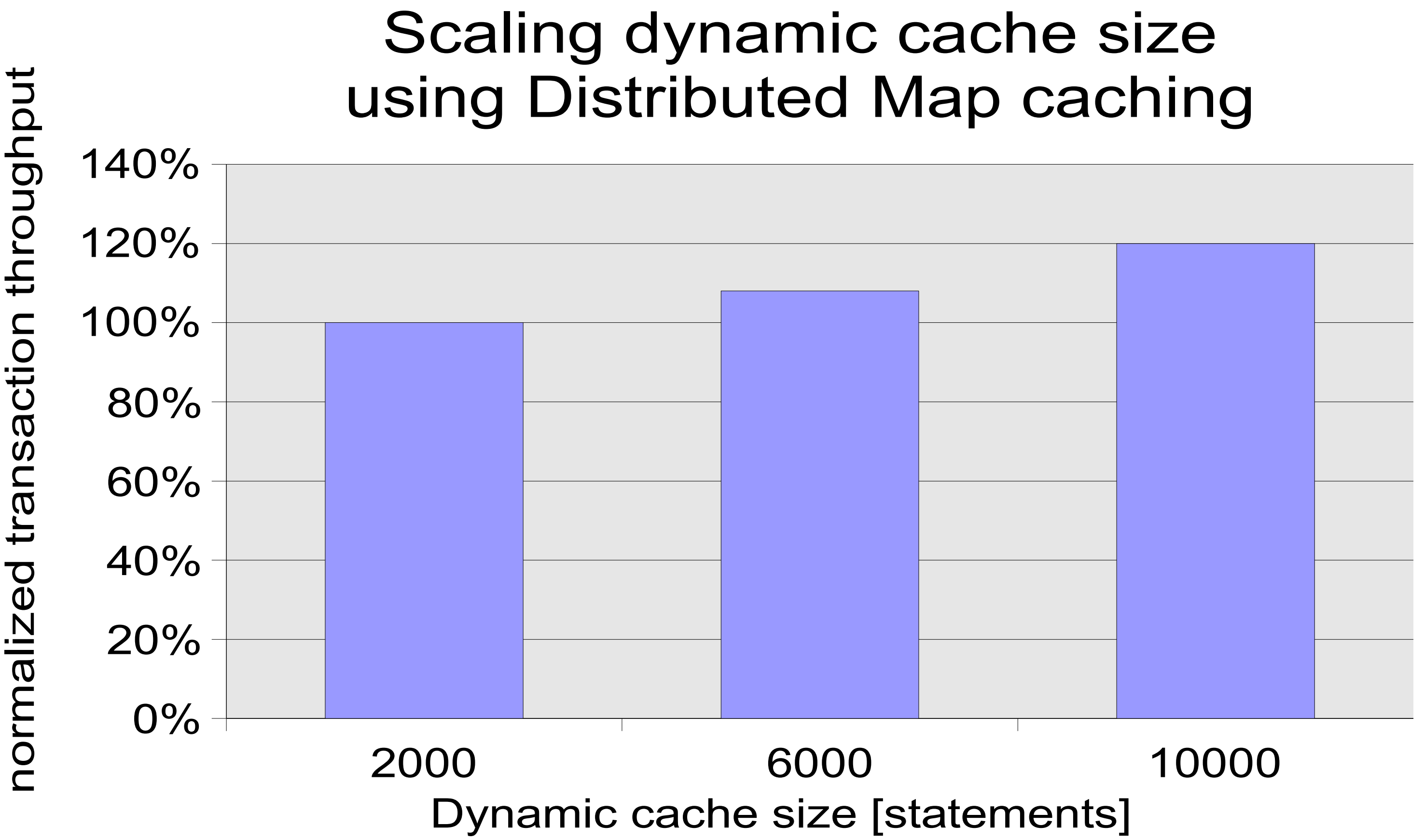
Caching modes (Trade benchmark)



- Significant performance gains are achieved when caching technology can be used
- Application support required (cache usage, data consistency !)

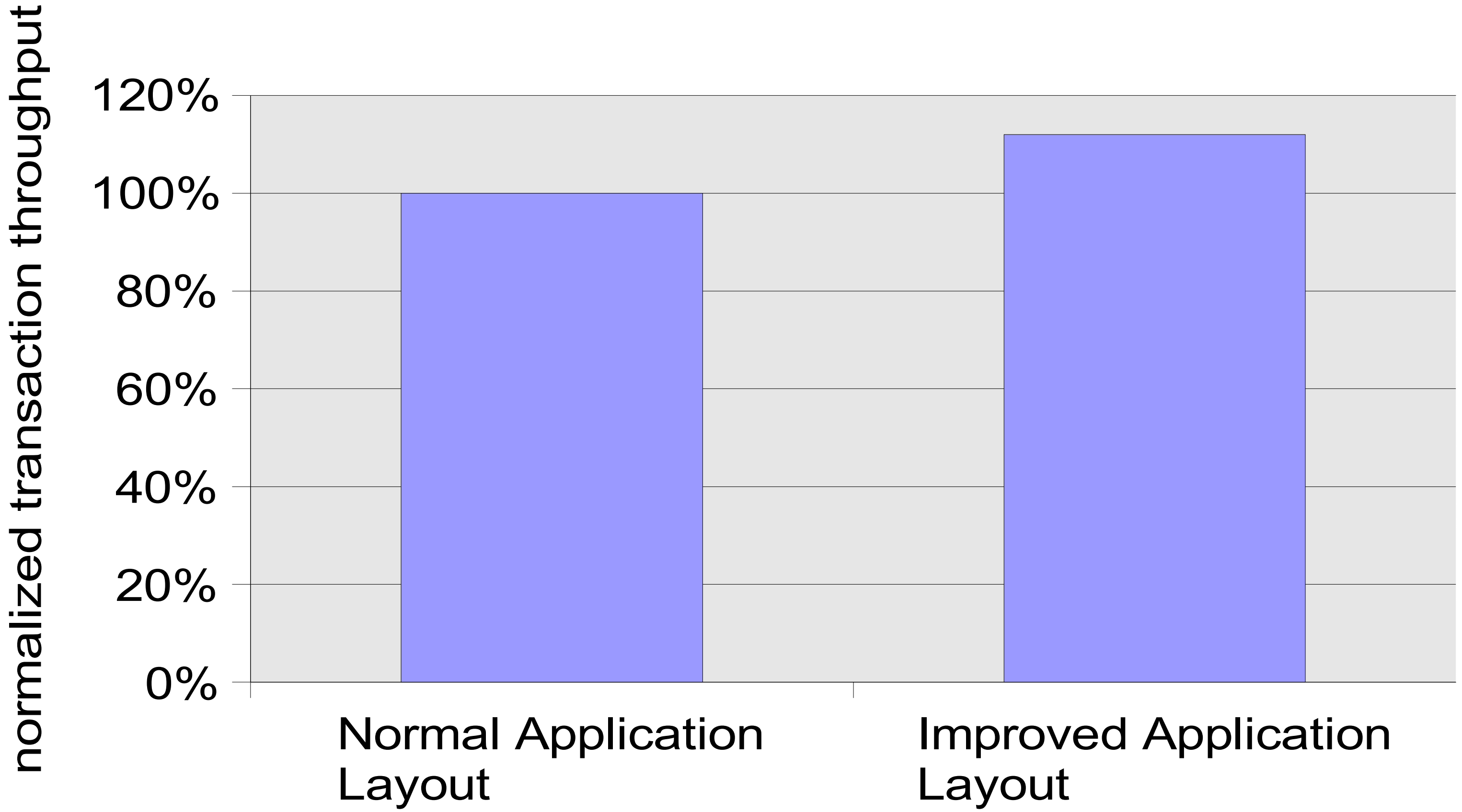
Varying dynamic cache size

- Best results seen in our experiment with 10.000 cached statements
- Default cache size is only 2000 statements



Database tuning effects (application layout)

- DB optimization is a key!
- Database optimization steps improve throughput by 12% in this example of optimization on database layout



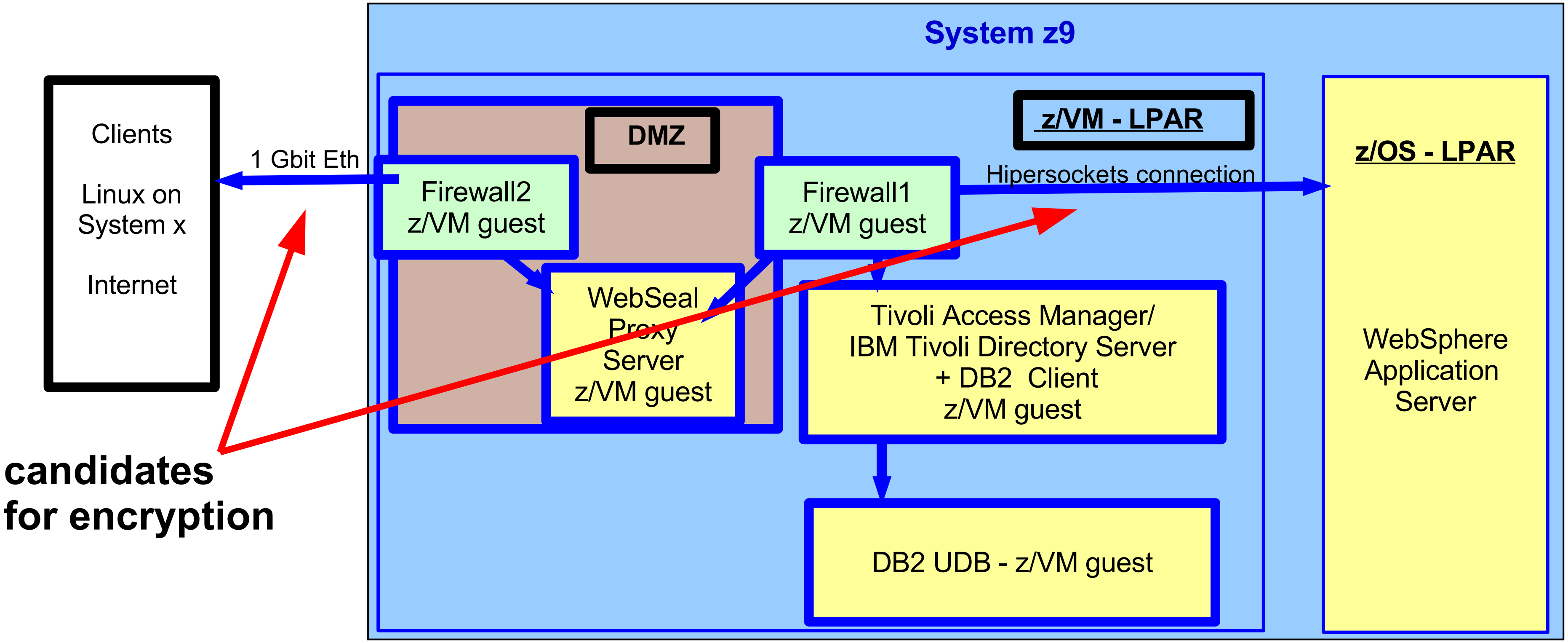
31-bit versus 64-bit

- 64-bit WebSphere and Java is production ready today!
- The 64-bit WAS environment needs additional CPU cycles and memory resources
- If running 64-bit define 20% to 30% more JVM heap to get the equivalent Java garbage collection behavior as seen with 31-bit
- **If the application does not need the additional memory size and heap then the use of 31-bit is recommended**
  - You can run 31-bit WebSphere in the 31-bit emulation layer of 64-bit distributions (RHEL5, SLES10)
  - There may be constraints like supported configuration, local 64-bit database connection

Crypto hardware support - basics

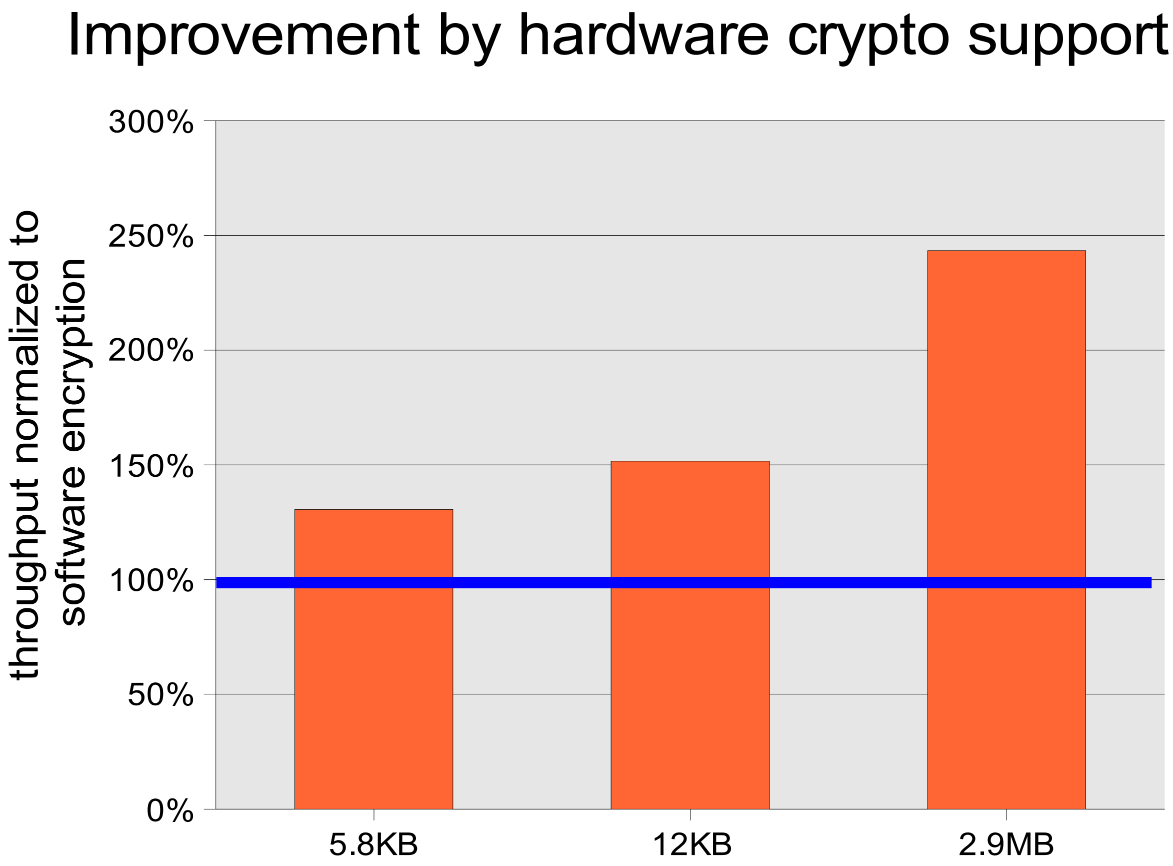
- There are two types of crypto hardware support on system z:
  - Crypto cards used for encryption related with authentication (userid +password/certificates)
    - Asymmetric or 'public key' crypto used for SSL handshake to establish SSL session & create session key
    - System z PCI crypto cards (PCICC, PCICA, PCIXCC, CEX2) can accelerate asymmetric crypto operations for Linux on System z
  - CPACF (system z processor feature) used for data encryption
    - Symmetric or 'private key' crypto used to encrypt/decrypt data - uses session key
    - The CP assist for Cryptographic Functions (CPACF) offers a set of symmetric cryptographic functions that enhance the encryption/decryption performance of clear key operations

Cryptographic hardware support another WebSphere environment – using WebSEAL



- WebSEAL provides an authentication and authorization mechanism
  - based on Tivoli Access Manager
  - enables an end-to-end Single Sign On (SSO) solution for secure transactions for WebSphere application servers residing on z/OS).

WebSEAL – page size with SSL access

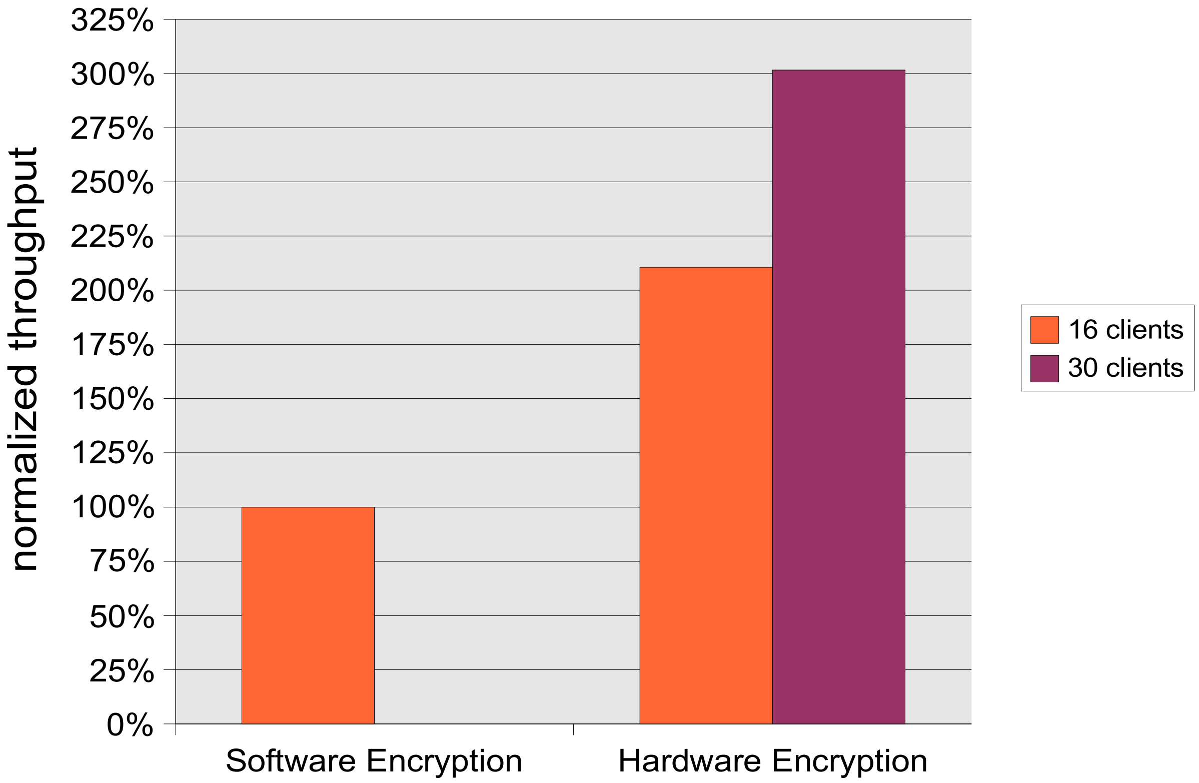


- the connection from client to WebSEAL server runs encrypted using SSL (AES-128)
- increase the size of the requested page
- uses mostly the CPACF feature from the processor

➢ Improvement up to factor 2.4x!

WebSEAL – authentication workload

Improvement by hardware crypto support



- access to very small pages (100 bytes) but authentication required
- the connection from client to WebSEAL server runs encrypted using SSL (AES-128)
- WebSEAL server with software encryption runs CPU constrained
- both crypto facilities can be used
  - CPACF from processor
  - CEX2C crypto card
  - ➢ increases the throughput and
  - ➢ releases the CPU
- total improvement up to factor 3x

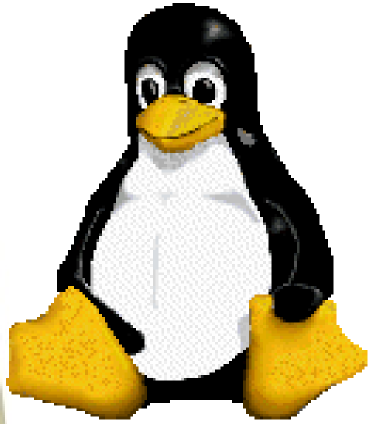
Summary

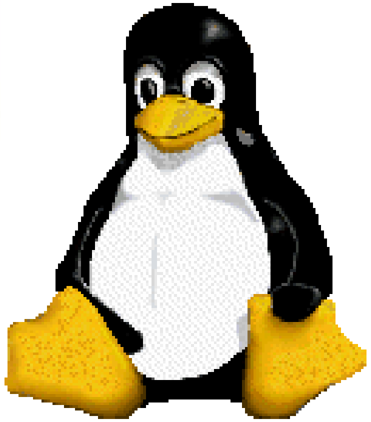
- Setup of a WebSphere environment requires optimization on all levels
- first step is monitoring
- identify the resources which are utilized at its limit
  - do not run a WebSphere application server above 90% CPU utilization
  - one critical point is the network connection between WebSphere and the database
  - check the utilization of the whole network
  - Java heap size
  - always an item is the layout in the database (indexes, table structures)
  - consider using the crypto features available on System z for encrypting data
- Tuning activities are often not independent from each other
- Monitor performance critical environments at least periodically

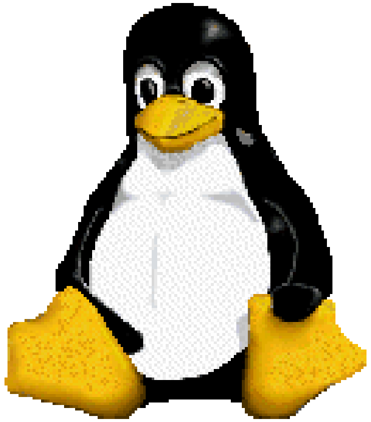
Visit us !

- Linux on zSeries Tuning Hints and Tips
  http://www.ibm.com/developerworks/linux/linux390/perf/index.html
  - White Paper WebSphere Application Server
    http://www.ibm.com/developerworks/linux/linux390/perf/tuning_pap_websphere.html
  - White Paper WebSEAL
    http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101100
- Linux-z/VM Performance Website
  http://www.vm.ibm.com/perf/tips/linuxper.html

Questions







z/VM setup

- Size the CPUs appropriate (use benchmarks, prototyping, size390 / techline)
- Shares set for the Linux guests prioritize CPU resources
  - Use relative shares with a soft limit
  - Give production guests higher shares
  - “Infrastructure Servers” (e.g. DBMS) should be given even higher shares
- Define xstore because z/VM has evolved around to have a memory hierarchy (25 % xstore as a rule of thumb up to 4GB)
- Make sure there is sufficient central storage plus paging space in z/VM to back the virtual memory request of all your Linux guests
- Provide twice as much DASD paging space than the sum of the Linux guests' virtual storage sizes (fast entire volumes)
- Enable QUICKDSP only for production guests and guests which perform critical system functions (VM TCP/IP, routers)

Linux on System z setup on z/VM

- Use as few number of processors as possible
  - Start with a reasonable number of processors (from sizing or prototyping)
  - Then reduce the number for each guest regarding the consumption (use your favorite monitoring tool)
  - **Do not define more virtual processors for the guest than are physical available to the z/VM LPAR**
- You should always define a swap file. This could be a VDISK (15% -20% of the Linux guests virtual memory) or if memory constraint in z/VM use a full minidisk (MDC turned off)
- Size your Linux guest to have enough virtual memory to run without swapping excessively except for a short peak time
- “Surplus” virtual memory larger than the working set size is used by Linux for caches and buffers but will cause z/VM paging if over-committed

Data Access - Disk

- Hardware choices
  - Use SCSI instead of ECKD
  - Use FICON instead of ESCON
    - 4Gb FICON > 2Gb FICON > 1Gb FICON
- Utilize your hardware
  - Use “striped” logical volumes from different ranks
  - Consider using PAV
  - Carefully set up your storage system
    - ESS Caching modes (normal, inhibit or record)
  - http://www.ibm.com/developerworks/linux/linux390/perf/tuning_rec_dasd_optimizedisk.shtml

WebSphere tuning

- JVM settings
  - Choose the proper maximum heap size for WebSphere and JVM
    - Leave a cushion of about 35% above normal high water mark
    - don't disable the JIT compiler
- Set the “Maximum pool size” of the connection pool (maximum number of physical connections to the database) accordingly to the sum of all data sources in this application server
- Static pages are best served via an HTTP server
- Check for bottlenecks in your server chain
  - Provide more resources to constraint servers
  - Various optimization actions are probably not independent
- Monitor the WebSphere Application Server dynamic cache size utilization
  - Use therefore the cache monitor application on the application server
- make sure to have no disk I/O constraints on the database