TCP/IP for VSE/ESA 1.4 and 1.5

Performance considerations

Ingo Franzki
e-mail: ifranzki@de.ibm.com
VSE/ESA Development

Trademarks

Contents

- TCP/IP basics
- General TCP/IP performance tuning
- TCP/IP for VSE/ESA
  - Multiple TCP/IP partitions
  - Startup, Parameters
- Telnet and FTP Performance results
- News with 1.4
  - OSA-Express and QDIO
- News with 1.5
  - HiperSockets support
  - Hardware Crypro support

Protocol Layers

Application	Message	Telnet FTP HTTP	SNMP NFS ...
Transport	UDP Datagram TCP Segment	TCP connection oriented	UDP connection less
Internetwork	IP Datagram	IP (connectionless)	
Link (Device Drivers)	Frames	Token-Ring, Ethernet, FDDI, ATM	
Hardware		Hardware	

Protocol Layers - continued

- TCP = Transmission Control Protocol
  - Connection oriented
  - Accepts data transmission requests of any length
  - Breaks the transmission data into chunks (TCP segments)
  - Reliably sends them across the network
  - Employs checksums, sequence numbers, timestamps, time-out counters for retransmission
  - Uses and exploits acknowledgments
  - Used for FTP, HTTP, Telnet, ...

Protocol Layers - continued

- UDP = User Datagram Protocol
  - Connectionless
  - UDP Datagram treated as 'single entity'
  - Each UDP Datagram is delivered separately
  - No checking for successful delivery
  - No use of acknowledgments
  - Datagram length is limited
  - Used for SNMP, NFS, ...

Protocol Layers - continued

- IP = Internet Protocol
  - No reliability, flow control or error recovery
  - Can do fragmentation and reassembly of its datagrams
  - No acknowlegments used
  - Just performs the transfer of IP datagrams
- Encapsulation principle for layers
  - Each layer sends its data down the protocol stack
  - Receives its data from the layer below

Protocol Layers - continued

Physical transferred (Frame)

Header	IP Datagram (or fragment)	Trailer

IP datagram

Header	IP Data (TCP segment or UDP datagram)

TCP Segment

Header	TCP data (application message)

UDP Datagram

Header	UDP data (application message)

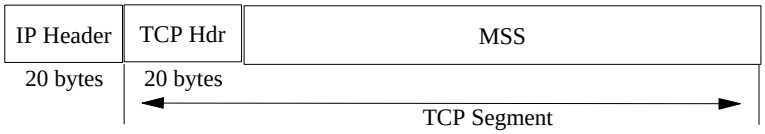
Maximum Transfer Unit

- MTU = Maximum Transfer Unit
  - Maximum amount of data in a frame that can be sent over the physical media
    - Maximum IP datagram size

Adapter	Default	Minimum	Maximum
Ethernet	1500	576	1500
Token-Ring	1500	576	4464 (4 Mbit/s) 17914 (16 Mbit/s) 17914 (100 Mbit/s)
CTC	4096	576	16K (RS/6000 CLAW) 32K (S/390 CTCA)
Fast Ethernet	1500	576	1500
FDDI	1500	576	4K
Gigabit Ethernet	1500	576	9K
HiperSockets	1500	576	64K

Maximum Segment Size

- MSS = Maximum Segment Size
- Biggest amount of data a TCP stack can receive in a single TCP segment
- Sent at connection setup time to communication partner

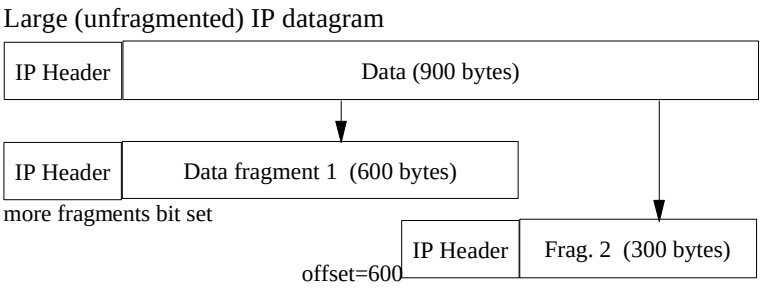


MSS = MTU - 40 bytes (without fragmentation)

Optimal MSS for a TCP Connection: MIN out of
- the MSS value of the other system
- the MTU value of the route minus 40 bytes

IP Fragmentation and Reassembly

- Example: MTU=620



- Large IP datagrams can be fragmented, each getting its own IP header
- Datagram is reassembled at final destination

IP Fragmentation and Reassembly (continued)

- Performance impacts
  - For sender
    - CPU overhead to create and transmit additional packets
    - Retransmit ALL packets in a datagram if a packet is lost
  - For receiver
    - CPU overhead to reassemble the packets
    - Memory overhead for buffers to reassemble the packets
    - Delays if a packet is lost
  - No problem if fragmentation only occurs occasionally

TCP Windowing Technique

- Send as much data as possible/reasonable before waiting for an acknowledgment
- Receiver decides how much data it is willing to accept
- Sender must stay within this limit
- Window is always related to a single session and direction
- At connection setup each partner assigns receive buffer space
- Every ACK sent back by the receiver
  - Contains the highest sequence number received
  - The size of its current receive window left

TCP/IP Performance Tuning

- Operating system tuning
  - Includes to tune local file attributes
- TCP/IP setup tuning
- Communication/network tuning
  - Mainframe end
  - Network
  - Workstation end
- TCP/IP application tuning

TCP/IP Performance Tuning

- TCP/IP Performance is limited by the
  - Speed of the slowest link
  - Window size of receiver, divided by the round trip time
  - Amount of CPU-time available on host
  - Speed of reading/writing data from/to disk
- Many TCP/IP performance problems are
  - Environment specific
  - Implementation specific
  - Not caused by inherent protocol limits

Network Performance

- Long transfer times in a net may be caused by
  - Slow links or small MTUs
  - Too many links involved or routing not efficient
  - Inefficient setup of packet and window sizes
  - Higher share of IP datagrams
    - e.g. 'time to live' expired
  - Higher share of resent TCP segments
    - 'Retransmission rate'
    - ACKs are delayed too long

TCP/IP Acknowledgment Considerations

- TCP ACKs are 'cumulative'
- No packet must be individually and immediately acknowledged
- Packets are only sent as long as the receiver's window can hold the data
- Packets are resent, if after a time-out no ACK was received by the sender
- Performance implications
  - Sender should proceed to send data, as long as receive window is open
  - A too low time-out in the sender may cause unnecessary retransmission of packets
  - A too high time-out my reduce the data rate

Principal Performance Dependencies

Parameter	Host CPU-time	Host Storage	Network Transfer time	DASD time
Host CPU speed	X			
S/390 Op.Syst. & Setup	X	X		X
MTU/MSS used	X	X	X	
Window size		X	X	
# transfer buffers		X	X	
Type of Comm.Adapter			X	
Network/Line speed			X	
Network reliability	X	X	X	
#Appl.-bytes in/out	X	X	X	X
TCP/IP implementation	X	X	X	X
TCP/IP application	X	X	X	X
Other TCP/IP parameters	X	X	X	X
DASD I/O Subsystem				X
DASD I/O Blocking	X			X

X major impact

TCP/IP for VSE/ESA

- VSE native implementation
- Especially developed for VSE (not ported)
- Runs in a separate VSE partition
  - Own multitasking mechanism
  - All daemons/servers run in the TCP/IP partition
    - I/O is done from the TCP/IP partition
  - Each TCP/IP partition has a unique ID in the EXEC card
- Shipped with VSE/ESA 2.3 and up
  - To be key enabled

Communication Hardware

- Communication Hardware
  - 3172/8232 LAN Channel Station Controller
    - Token-Ring, FDDI, Ethernet
  - ES/9221 Integrated Adapter (CETI)
    - Token-Ring, Ethernet
  - OSA-2
    - Token-Ring, Ethernet (10/100), FDDI, ATM-LE
  - 2216 Nways Multiaccess Connector
  - CTCA to any S/390 operating system
  - Channel attached RS/6000 (CLAW)
  - New: OSA-Express
    - Gigabit Ethernet, Fast Ethernet, ATM-LE, TR100
  - New: HiperSockets

TCP/IP Application Types

- TCP/IP Application types (services)
  - TELNET (Client and Server)
  - FTP (Client and Server)
  - GPS (Server)
  - HTTP Server
  - LPR/LPD (Client/Server)
  - NFS (Server only)
- TCP/IP APIs
  - Assembler SOCKET interface
  - C-LE Socket interface
  - EZA Socket Interface(s)

TCP/IP Application Types - continued

- TELNET
  - As server
    - Allow remote access/logon to VTAM applications via TN3270
  - As client
    - Access to other applications from local CICS
- GPS (General Print Server)
  - Allows, in a TN3270 environment, to direct VTAM 328x print to any TCP/IP capable printer
  - Identifies itself to VTAM as a locally attached 3287

TCP/IP Application Types - continued

- FTP
  - Transfer data or files from/to remote systems
  - Supported file types
    - VSAM ESDS and KSDS
    - VSE SD files
    - VSE library members
    - POWER queue entries
    - VSE/ICCF library members (read only)
  - FTP Server = FTP Daemon
  - FTP Client
    - Interactive FTP client
    - Batch FTP
    - FTPBATCH

TCP/IP Application Types - continued

- HTTP Server
  - Allows to retrieve HTML documents via browser
  - CGIs can be used to create the pages dynamically
  - HTML files are stored in VSE libraries
- LPD (= Server)
  - Print data of any TCP/IP system on a VSE printer
  - Printing via POWER
- LPR (=Client)
  - Print VSE data on any TCP/IP network printer
  - AUTOLPR, CICS transaction, batch job

TCP/IP Application Types - continued

- NFS (Network File System)
  - Transparent access from NFS client (PC or UNIX) to files stored in a remote VSE as if they were local
  - Share files across a TCP/IP network
  - NFS assumes
    - A hierarchical file system
    - Each file being a byte stream of a certain length
    - Without a record structure
  - Supported file types
    - VSE library members
    - POWER queue entries
    - VSAM ESDS files

TCP/IP Application Programming Interfaces

- Socket APIs
  - Assembler SOCKET macro
    - SOCKET type,connect,keywords
    - 'Proprietary' interface
  - COBOL, PL/I Socket API
  - C-LE Socket API
    - Standard C socket interface (using VSE/LE)
    - Compatible with OS/390
  - EZA Socket Interface(s)
    - Ported from OS/390
  - REXX Socket API

Performance Aspects

- Socket APIs
  - C-LE Socket interface requires VSE/LE
  - The most efficient API from a performance point of view is the Assembler SOCKET macro interface
- Basic considerations
  - Try to send and receive as much data as possible per socket call
- Number of concurrently active sockets
  - All sockets are chained in a single queue which is searched sequentially

TCP/IP Startup job

- VSE partition size
  - Start with 20-30M partition size
- SETPFIX LIMIT
  - Start with LIMIT=900K
  - OSA Express and HiperSockets needs at least 2100K
- Type of VSE partition
  - Can be static of dynamic
- Parameters to EXEC IPNET
  - SIZE=IPNET
  - IPINIT0x contains all TCP/IP parameters
  - DSPACE=3M max. size of dspace used by VTAM

TCP/IP Dispatch Priority

- Select PRTY sequence (low to high)
  - Batch, DB2, CICS, TCP/IP, VTAM, POWER
- A second TCP/IP partition is recommended
  - If besides Telnet ...
    - Concurrent FTP activity (not FTPBATCH)
    - Concurrent LPR/LPD activities
  - High concurrent FTP (or LPR/LPD) activity may/will impact e.g. Telnet response times
  - Or use FTPBATCH

Multiple TCP/IP Partitions

- Each TCP/IP Partition should have
  - A separate IP address
  - A separate host name
  - Its own set of adapters
  - Its own setup of startup parameters
- Functional reasons
  - Separation of workload
  - Separation of production and test
  - Separation of production workload
  - Separation of network (e.g. security)

Multiple TCP/IP Partitions - continued

- Performance reasons
  - Exploit more than 1 engine for TCP/IP
    - Only one engine per partition
  - Need of more virtual storage below the line
    - e.g. Telnet (VTAM) buffers
  - Individual customization
  - Separation of TELNET and FTP/LPR activities
- IPNET link has no performance benefits
  - Recommendation: let each partition have its own network link

TCP/IP's Access to VSE Data

- VSAM
  - VSAM macros and VSAM code in SVA
- POWER
  - POWER SAS (XPCC)
- LIBR
  - LIBRM macro
- ICCF
  - SLI (Read only) and DTSIPWR in SVA
- SD
  - DTFSD macro (BAM)

Batch FTP From a Separate Partition

- // EXEC FTP
  - Only FTP initialization is done from a batch partition
  - No performance related benefits
- // EXEC FTPBATCH
  - Potential exploitation of >1 engine of an n-way
  - Separate File-I/O routine used per FTP
  - Control of FTP batch CPU dispatch priority
  - More overhead for data transfer between batch and TCP/IP partition
  - Move of data between batch and TCP/IP partition using access registers

Performance Related Parameters

Parameter	Any	outbound only	inbound only	TN3270 in+out	FTP in+out
DEFINE ADAPTER/LINK MTU		X			
TELNETD POOL				X	
SET ALL_BOUND	X				
DISPATCH_TIME	X				
REDISPATCH	X				
ARP_TIME	X				
REUSE_SIZE	X				
FULL_SCAN	X				
GATEWAY	X				
CHECKSUM	X				
SET MAX_SEGMENT			X		
WINDOW_DEPTH			X		
CLOSE_DEPTH			X		
WINDOW_RESTART			X		
SET RETRANSMIT		X			
FIXED_RETRANS		X			
WINDOW		X			
ADDITIONAL_WINDOW		X			

Performance Related Parameters (continued)

Parameter	Any	outbound only	inbound only	TN3270 in+out	FTP in+out
SET SLOW_START		X			
SLOW_RESTART		X			
SLOW_INCREMENT		X			
SET TELNETD_BUFFERS				X	
TRANSFER_BUFFERS					X
MAX_BUFFERS					X

Performance Related SET Commands

- SET ALL_BOUND - maximum idle time
  - Similar to CICS ICV
  - Default is 9000 (30 sec)
- SET DISPATCH_TIME - maximum time-slice a single TCP/IP pseudo task can get
  - Reduced impact since SERV130K
- SET MAX_SEGMENT - maximum size of a TCP segment
  - 576 .. 32k (64K for HiperSockets), default is 32768
- SET WINDOW - receive window size

Performance Related SET Commands

- SET WINDOW_DEPTH - number of data segments which can be concurrently queued inbound in TCP
- SET CLOSE_DEPTH - number of TCP segments are still accepted, in spite of a fully closed window
- SET RETRANSMIT - time interval before retransmission occurs
- SET TELNETD_BUFFERS - number of 16K buffers in the TELNETD buffer pool (only if POOL=YES)
- SET TRANSFER_BUFFERS - number of 32K transfer buffers allocated to the FTP buffer pool

Remove Unnecessary Actions from TCP/IP

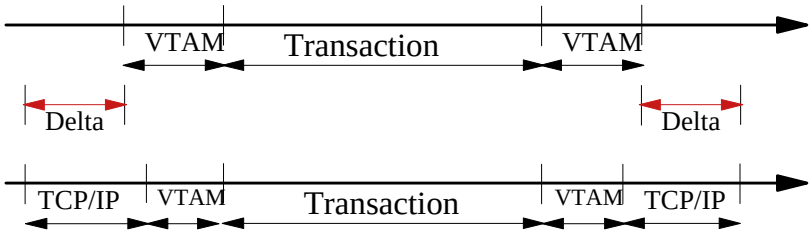
- Symptom
  - TCP/IP partition consumes sporadically CPU-time, 'without doing anything'
- Background Info
  - TCP/IP must inspect EVERY incoming data packet
- Recommendations
  - Filter unnecessary data packets
    - Make sure IP filtering is ON for OSA and 3172
    - Find out the source for frequent ARP updates

TN3270 Measurement Environment

- VSE/ESA 2.3
- TCP/IP 1.3 (E/G/J/K) and also 1.4
- Turbo Dispatcher, single engine
- DSW online workload
- 2 CICS/VSE partitions (F4 and F5)
- TCP/IP for VSE/ESA (F7)
- F4 and F5 balanced with F7
- 125 active terminals per CICS partition
  - driven by TPNS
- POOL=YES and TELNETD_BUFFERS=20

TN3270 Measurement Results

Overhead in terms of CPU-time per transaction



Delta/txn = msec/txn(TCP/IP) - msec/txn(VTAM)

Varies here between 20.0 and 24.9 msec

TN3270 Measurement Results - continued

Expected rel.-CPU-time and ITR-ratio vs SNA

$$\text{ITRR} = \text{ITR ratio} = \frac{\text{msec/txn (VTAM)}}{\text{msec/txn (TCP/IP)}}$$

In the measured cases, average overall (VTAM based) CPU-time of a transaction was about 20 msec (~280KI)
TCP/IP overhead was between 280KI and 350KI

Type/CPU-Heaviness of workload	Rel. CPU-time with TCP/IP	ITRR
DSW, measured (280KI)	2.0	0.5
Medium customer transaction (560KI)	1.5	0.67
Heavier customer transaction (840KI)	1.33	0.75
Heavy customer transaction (1000KI)	1.28	0.78

TN3270 Measurement Results - continued

- TCP/IP for VSE/ESA 1.4 vs. 1.3
  - 7-13 % less TCP/IP CPU-time overhead
- Response time impact is small
- TN3270 overhead
  - VSE/ESA native
    - Online utilization increases from 50% to 62%
  - VM/VSE Guest
    - Online utilization increases from 60% to 74%

TN3270 Measurement Results - continued

- TN3270 Virtual Storage capacity

	125 daemons		per daemon	
	-24	-31	-24	-31
TCP/IP GETVIS	476K	600K	3.8K	4.8K
VTAM GETVIS	0K	52K	0K	0.4K
SVA	20K	524K	0.16K	4.2K

Rough estimate for TN3270 VS-Capacity:

Max. #TN daemons = (remaining GETVIS-24) / 4K

Example:
A remaining GETVIS-24 of about 10M, gives about 2500 Telnet daemons

FTP Measurement Results

- EDR = effective Data Rate (KB/sec)
- It is irrelevant who initiated an FTP transfer
- Transfer of a file from A to B may differ in EDR from transferring the identical file from B to A
  - Speed of physical HDD
  - Read/write caching
  - Blocksizes used (KB/IO)
- The higher the EDR of an FTP transfer, the higher is the required CPU utilization
- EDRs displayed by TCP/IP for VSE/ESA
  - Transfer sends
  - File I/O seconds

FTP Measurement Results - continued

Parameters	FTP speeds Source	Target	Network	CPUT/KB
Network speed and load			X	
TCP/IP parameters	X	X	X	X
FTP parameters	X	X	X	X
DASD speed (READ/WRITE)	X	X		
Local file definition				
- type	X	X		X
- log record length (NFS)	X	X		
- blocksize on disk	X	X		X
- I/O blocking (KB/IO)	X	X		X
- ASCII/EBCDIC/BINARY	X	X		X
size of files	X	X		X
Processor speed	X	X		X
other concurrent activities	X	X	X	
TCP/IP for VSE/ESA PTF level	X	X	X	X

FTP Measurement Results - continued

EDR ranges (KB/sec) observed (1.3)

	FTP to VSE	FTP from VSE	Major impact
LIBR	340	470	DASD, network speed
POWER	115	290	DBLK
VSAM ESDS		460 360 160	to S/390 (CTCA) to RS/6000 CLAW Via CLAW & T/R

CPU resources (KI/KB) required (1.3)

	FTP to VSE	FTP from VSE	Dependencies
LIBR	18.9 - 20.1	11.9 - 13.3	
POWER	85	45	
VSAM ESDS		7.6 - 9.2	Convertion

FTP Measurement Results - continued

- TCP/IP for VSE/ESA 1.4 (ServPack A)
  - EDRs increased by 10% to 30%
  - CPU-time consumption decreased by about 25%
- Virtual Storage Capacity

	10 daemons		per daemon	
	-24	-31	-24	-31
TCP/IP GETVIS	3104K	40K	310K	4K

Max #FTP daemons = (remaining GETVIS-24) / 310K

Example:
A remaining GETVIS-24 of about 10M, gives about 32 FTP daemons

FTP with Batch Measurement Results - continued

Data rate comparison (VSAM)

	Overall EDR Transfer (KB/sec)		File I/O (KB/sec)	
	Real 9345	Virt. Disk	Real 9345	Virt. Disk
Interactive FTP	639	930	682	1462
Batch FTP	639	930	682	1462
FTPBATCH	511		682	

- Same rates as for Interactive FTP
  - Except transfer rate seen by FTPBATCH
- Overall EDRs for (single) FTPBATCH are about 15% lower here than from Batch FTP

FTP with Batch Measurement Results - continued

- FTPBATCH with slightly higher CPU-time and with lower EDR
- FTPBATCH file transfers
  - Can be better workload balanced (controlled)
    - Via PRTY
  - Can run concurrently and thus achieve a higher sum of FTP EDRs
  - Allow to exploit >1 processor engines

Multi Thread Event Processing

- More than 1 events can be processed at a time
  - Separate TCP/IP internal task is assigned to any event (printout)
- Possible problems
  - Unpredictable order of events from independent jobs
  - Some printers only accepts one connection at a time
- SET SINGLEDEST=ON
  - Only one open connection possible to one destination

SSL and Crypto Overview

- SSL for VSE is part of the TCP/IP base (ServPack C)
- Enabled with the Application Pak
- Integrated into TCP/IP for VSE/ESA
- Supports SSL 3.0 and TLS 1.0
- Key exchange: RSA
- Data Encryption: DES and Triple DES
- Hash algorithm: MD5, SHA
- Supports X.509v3 PKI Certificates
- SSL daemon implementation for HTTPS, Telnet
- SSL API compatible with the OS/390 SSL API

Key Management

- Keys and certificates are stored in a "keyring file"
  - VSE: In a VSE library
- SSL for VSE uses 3 VSE library members:
  - keyname.PRIV - the private key
  - keyname.CERT - the certificate
  - keyname.ROOT - the root certificate
- Stored in library CRYPTO.KEYRING per default
- Utilities available for key management and creation
  - CIALPRVK, CIALCERT, CIALROOT
- $SOCKOPT.PHASE defines the SSL parameters

SSL Daemon (SSLD)

- Define an SSL daemon for each TCP port that you want to secure:
  - DEFINE TLSD,ID=MYSSLD,

PORT=443,	HTTPS port
PASSPORT=443,	
CIPHER=0A096208,	Cipher suites
CERTLIB=CRYPTO,	library name
CERTSUB=KEYRING,	sublibrary name
CERTMEM=MYKEY,	member name
TYPE=1,	server application
MINVERS=0300,	SSL 3.0
DRIVER=SSLD	Driver phase name

Secure Socket Layer API

- Compatible to OS/390 SSL API
- Functions available for
  - Session initiating
  - Sending/receiving data
  - Ending a session
- SSL API is based on Socket API
- SSL API can be called from
  - LE-C programs
  - Assembler programs
  - REXX programs

CryptoVSE API

- Native cryptographic API (not available through LE)
- Provides cryptographic services:
  - Data encryption
    - DES
    - Triple DES
    - RSA PKCS #1
  - Message Digest
    - MD5
    - SHA-1
  - Digital Signatures
    - RSA PKCS #1 with SHA1 or MD5
  - Message Authentication
    - HMAC

Restrictions

- Cipher Suites supported:
  - 01 - RSA512_NULL_MD5
  - 02 - RSA512_NULL_SHA
  - 08 - RSA1024_DES40_CBC_SHA
  - 09 - RSA1024_DES_CBC_SHA
  - 0A - RSA1024_3DES_CBC_SHA
  - 62 - RSA1024_EXPORT_DES_CBC_SHA
- Only one Root certificate
- Certificate revocation lists not supported
- Keyring is not password protected
- Software encryption only for
  - DES, DES CBC, 3DES CBC
  - SHA, MD5

SSL Enabled Applications

- MS Internet Explorer 5.5 or higher (HTTPS)
- Netscape Navigator 4.7 or higher (HTTPS)
- Mozilla
- Several telnet clients
- VSE Connectors (VSE/ESA 2.6 or later)
  - The VSE Connector Server and Client have been SSL enabled
  - SSL client authentication with user ID mapping supported with VSE/ESA 2.7
- CICS Web Support (HTTPS)
  - TCP/IP Service can be SSL enabled

SSL Enabled VSE Connector Server

- The VSE Connector Server can run either in
  - Non SSL mode (as in VSE/ESA 2.5)
  - SSL mode
- Configurable values
  - SSL Version (typically SSL 3.0)
  - Keyring library and key member
  - Cipher Suites
  - Server or Client authentication
- Separate SSL configuration member

SSL Enabled VSE Connector Client

- SSL can be enabled per connection
- New properties in VSEConnectionSpec
  - setSSL(true/false)
  - setSSLProperties(...) / setSSLPropertiesFile(...)
- SSL properties specifies
  - Keyring file (containing the certificates + private key)
  - Keyring password
  - SSL Version
  - Cipher Suites

SSL Enabled VSE Connector Client - continued

- Supports PKCS#12 keyring files as well as JKS
- SSL implementation is based on JSSE (Java Secure Socket Extension)
- JSSE Provider available from
  - Sun (http://java.sun.com/products/jsse/index.html)
  - IBM (based on SSLight)
    - shipped with JDK
- VSE Connectors uses IBM's JSSE implementation
  - Software encryption
- Key management can be done with graphical front-end (IKEYMAN)

Performance Related Parameters

Parameters	Session initiating	Data exchange
Key exchange algorithm		
RSA512	X	-
RSA1024	X	-
Encryption Algorithm		
NULL	-	X
DES40CBC	-	X
EXPORT_DESCBC	-	X
DESCBC	-	X
3DESCBC	-	X
Hash Algorithm		
MD5	X	X
SHA	X	X
Session caching	X	-
Message Length	-	X

-Data exchange overhead is proportional to bytes/msg

-CPU-time overhead caused by SSL is in

-TCP/IP partition for SSL Daemon

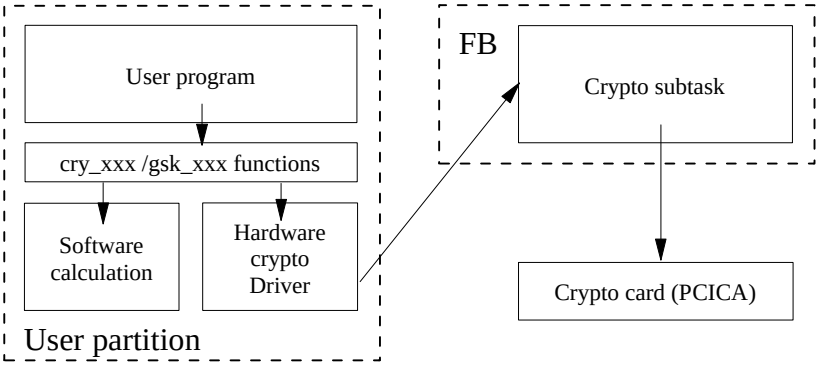
-application partition for API usage

Hardware Crypto Overview

- Requires VSE/ESA 2.7 and TCP/IP for VSE/ESA 1.5
- Supported crypto card
  - PCI Cryptographic Accelerator (PCICA)
    - Feature code 0862
    - Available for zSeries (z800, z900)
- Only RSA (asymmetric) is supported
  - Of benefit for Session initiation (SSL-Handshake)
- Also supported with
  - z/VM 4.2 + APAR VM62905
  - z/VM 4.3

Hardware Crypto Overview - continued

- New crypto subtask in Security Server (SECSERV) running in FB
  - Or as separate job if no SECSERV is running
  - Crypto card is polled by crypto task



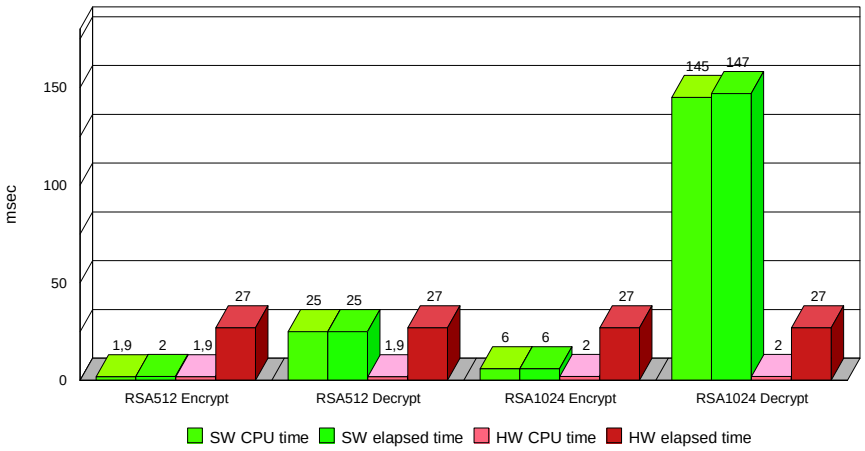
Measurement Environment

- VSE/ESA 2.7 running on a z900 (2064-109)
  - on 1 processor (~2064-101)
  - with a PCI Cryptographic Accelerator
- Testcase programs on VSE
  - Crypto operations measurements
    - calling cry_xxx functions (RSA, DES, SHA, MD5)
    - each crypto operation is performed 10000 times
  - Secured data transfer (SSL)
    - performs SSL handshake
    - performs encrypted data transfer
    - counterpart program running on Windows (SSL-client)
- All RSA operations are measured
  - with Hardware Crypto support
  - with Software Crypto
    - support already available with TCP/IP 1.4/1.5 as shipped in VSE/ESA 2.6

Measurement Environment - continued

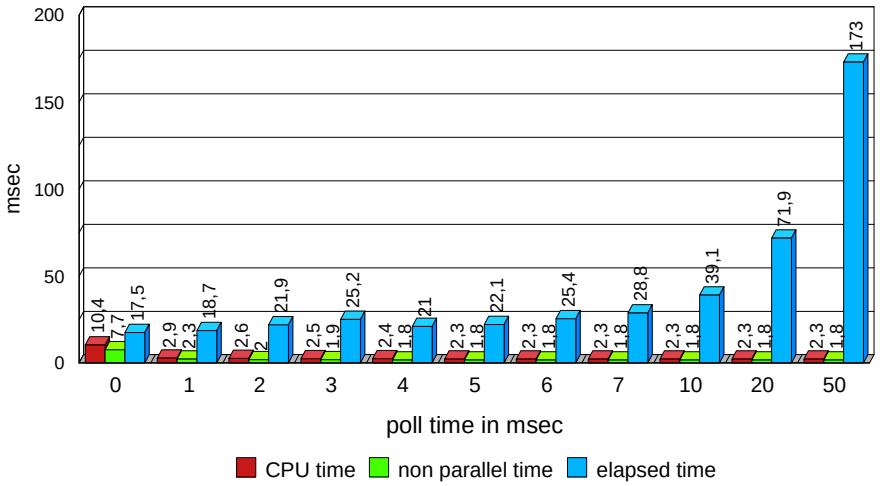
- Variations
  - RSA encrypt/decrypt
    - 512 / 1024 bit key
  - DES, DES CBC, 3DES CBC encrypt/decrypt
    - software crypto only
    - message length (128, 256, 512 bytes)
  - SHA Hash, MD5 Hash, SHA HMAC, MD5 HMAC
    - software crypto only
    - message length (128, 256, 512, 1K, 2K bytes)
  - SSL handshake/data transfer
    - 01 RSA512_NULL_MD5
    - 02 RSA512_NULL_SHA
    - 08 RSA512_DES40CBC_SHA
    - 09 RSA1024_DES_CBC_SHA
    - 0A RSA1024_3DES_EDE_CBC_SHA

Measurements Results - RSA



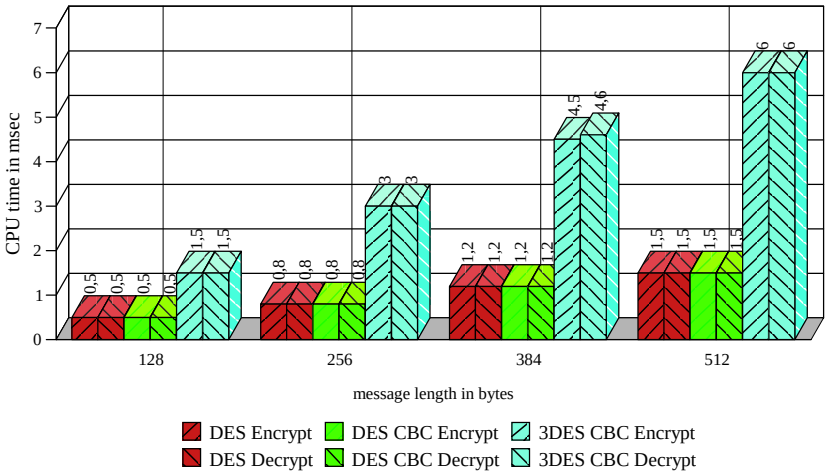
HW Crypto:
- CPU time and elapsed time is independent of operation / key length
- RSA operation takes about 2 msec CPU time and 28 msec elapsed time
- CPU time is always less than software crypto

Measurements Results - RSA polltime



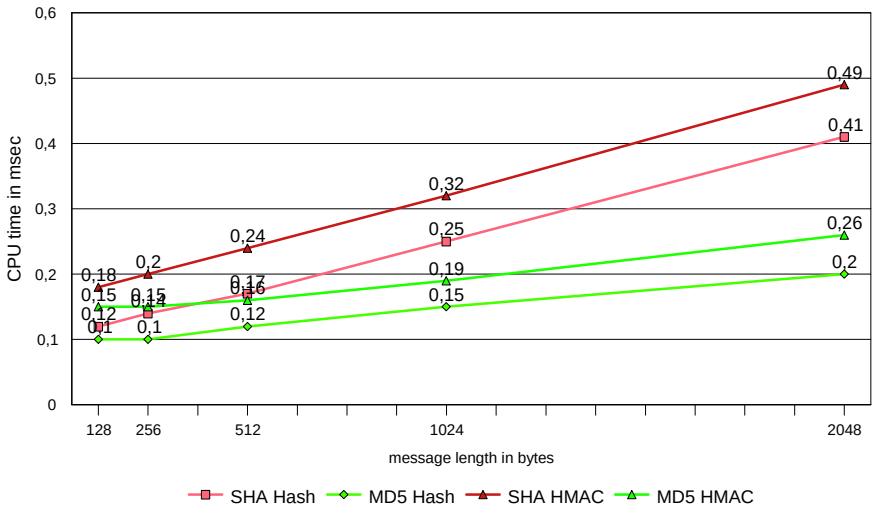
Per default a polltime of 7 msec is used.
Can be changed with: MSG FB,DATA=WAITTIME=nn
Smaller values increase CPU time, higher values increase elapsed time

Measurements Results - DES, DES CBC, 3DES CBC (symmetric)



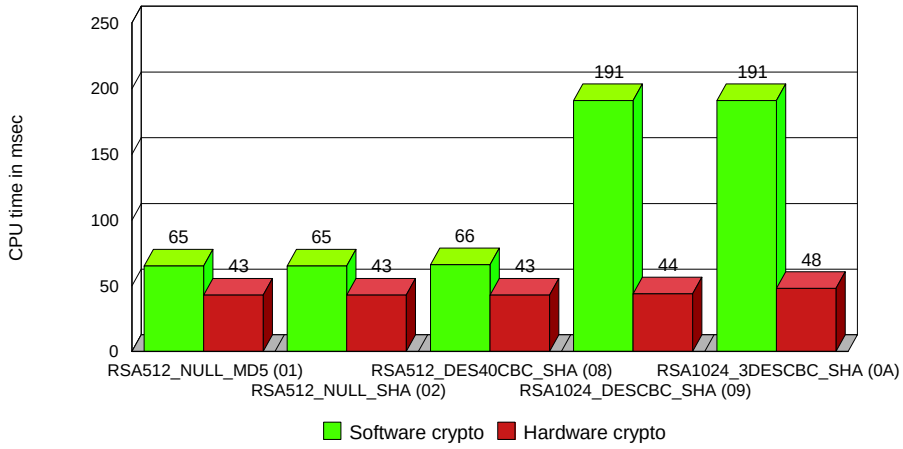
Software Crypto only!
DES and DES CBC takes similar CPU times, 3DES CBC about 3.8 times

Measurements Results - SHA, MD5



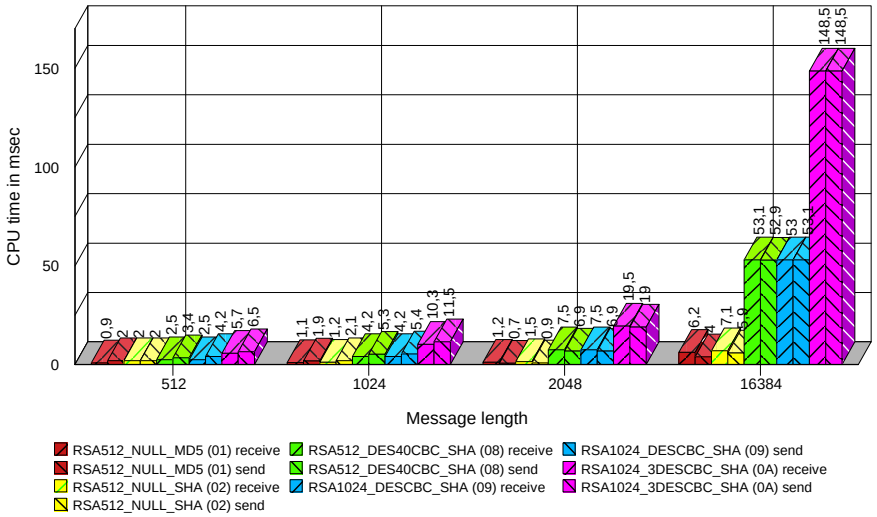
SHA takes about 1.8 times more CPU time compared to MD5
Software Crypto only!

Measurements Results - SSL Handshake



HW Crypto:
- CPU time and elapsed time is independent of cipher suite used
- SSL handshake takes about 43-48 msec CPU time (connection establishment)

Measurements Results - SSL data transfer



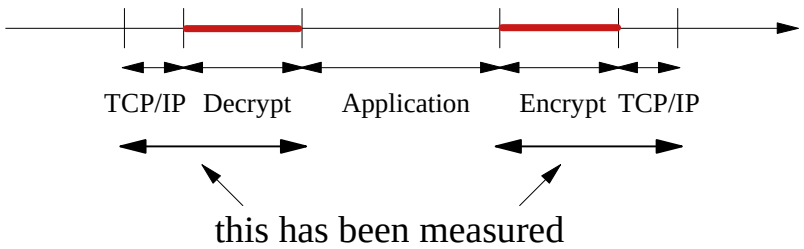
CPU time depends on used hashing (SHA/MD5) and encryption algorithm (DES/3DES)

Software Crypto only!

SSL data transfer overhead

Non SSL

SSL



Measurements Results - conclusion

- HW Crypto
  - Supports RSA operations only (e.g used by SSL handshake)
  - CPU time/elapsed time is independent of operation and key length
  - Software RSA encryption is faster in terms of elapsed time (on large processors)
    - but hardware crypto saves CPU time
- SW Crypto
  - CPUtime /elapsed time is very dependent on CPU speed and utilization

SSL Performance Recommendations

- Use SSL only if there is a need for
  - If at least one of the following is required
    - Keeping secrets
    - Proving identity
    - Verifying information
- Cipher Suites 01 and 02 has less CPU-time consumption, but NO data encryption
  - RSA512_NULL_MD5, RSA512_NULL_SHA
- If data encryption is required
  - Use cipher suites 08, 09 or 0A
  - 08 uses 512 bit keys, others 1024
  - 1024 bit RSA keylength is recommended (from a security point of view)

OSA-Express

- Requires VSE/ESA 2.6 or later
- Available for G5 and above
- Exploits Queued Direct I/O

	Gigabit Ethernet	Fast Ethernet 100 Mbps	ATM-LE 155 Mbps	Tokenring 4/16/100 Mbps
CHIPID TYPE=OSE (non-QDIO)	no	yes	yes	yes
CHPID TYPE=OSD (QDIO)	yes	yes	yes	yes

OSA-Express for IBM eServer zSeries and S/390, G221-9110-01, 11/2001

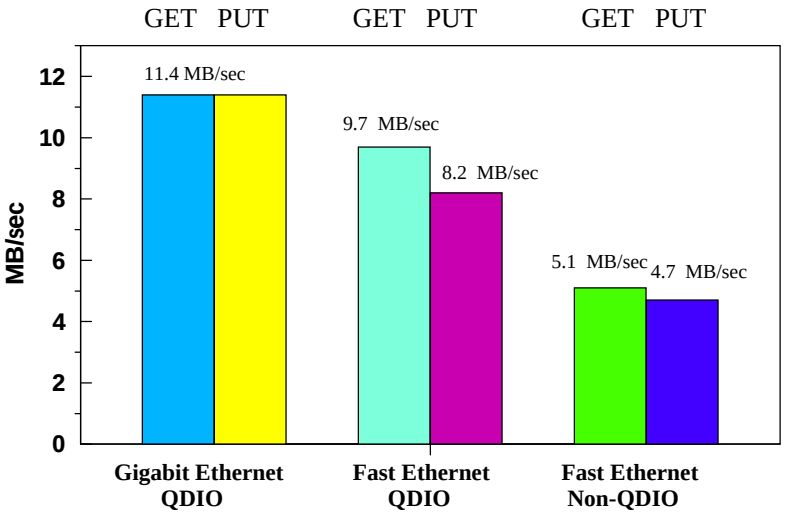
OSA-Express - continued

- Queued Direct I/O
  - Designed for very efficient exchange of data
  - Uses the QDIO Hardware Facility, without traditional S/390 I/O instructions
  - Without interrupts (in general)
  - Use of internal queues
  - With pre-defined buffers in memory for asynchronous use

OSA-Express Measurements

- Environment
  - VSE/ESA 2.6 on G6 native (LPAR)
    - TCP/IP 1.4 ServPack C
  - Linux on Netfinity
- Network attachment
  - Gigabit Ethernet QDIO (MTU=1500)
  - Fast Ethernet QDIO (MTU=1500)
  - Fast Ethernet Non-QDIO (MTU=1500)
- Workload
  - GET = VSE to Linux, 100MB $NULL file
  - PUT = Linux to VSE, 100MB $NULL file

OSA-Express Measurements - continued

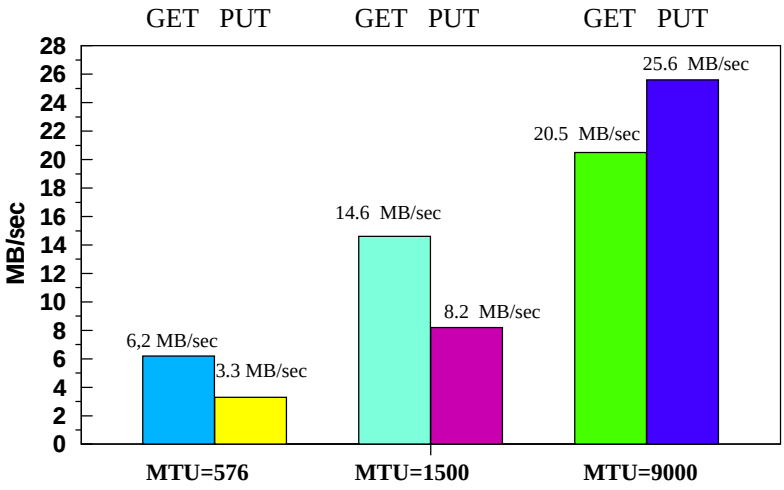


MTU = 1500

Gigabit Ethernet Measurements

- Environment
  - VSE/ESA 2.6 on G6 native (LPAR)
    - TCP/IP 1.4 ServPack C
  - Linux on G6 native (LPAR)
- Network attachment
  - Gigabit Ethernet QDIO
  - MTU = 576...9000
- Workload
  - GET = VSE to Linux, 100MB $NULL file
  - PUT = Linux to VSE, 100MB $NULL file

Gigabit Ethernet Measurements - continued



HiperSockets hardware elements ('Network in a box')

- Synchronous data movement between LPARs and virtual servers within a zSeries server
  - Provides up to 4 "internal LANs" HiperSockets accessible by all LPARs and virtual servers
  - Up to 1024 devices across all 4 HiperSockets
  - Up to 4000 IP addresses
  - Similar to cross-address-space memory move using memory bus
- Extends OSA-Express QDIO support
  - LAN media and IP layer functionality (internal QDIO = iQDIO)
  - Enhanced Signal Adapter (SIGA) instruction
    - No use of System Assist Processor (SAP)

HiperSockets hardware elements ('Network in a box') - continued

- HiperSockets hardware I/O configuration with new CHPID type = IQD
  - Controlled like regular CHPID
  - Each CHPID has configurable Maximum Frame Size
- Works with both standard and IFL CPs
- No physical media constraint, no physical cabling, no priority queuing
- Secure connections
- Requires TCP/IP 1.5

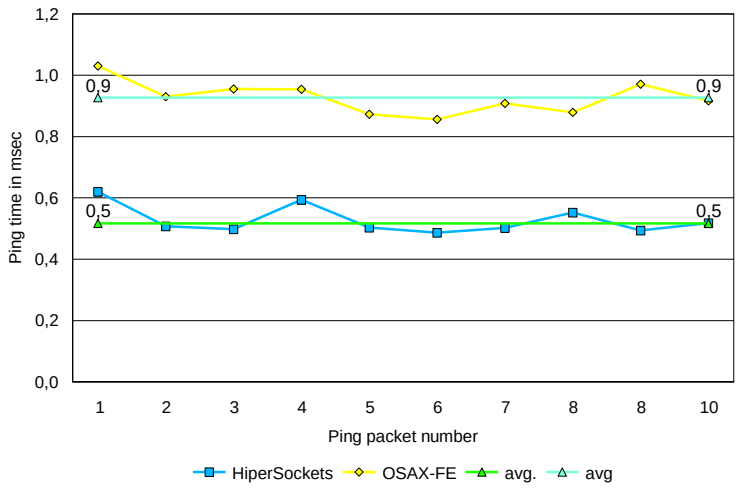
Measurement Environment

- z800 (2066-004)
  - 4 processors
- VSE/ESA 2.7 GA Driver in an LPAR (native)
  - 1 CPU active (~2066-001)
  - TCPIP00 (F7): OSA Express Fast Ethernet
  - TCPIP01 (F8): HiperSockets
- Linux for zSeries in an LPAR (native)
  - 3 CPUs active (shared)
  - eth0: OSA Express Fast Ethernet
  - hsi10: HiperSockets

Latency (Round trip time) - results

- Measurements has been done with PING command
  - Issued at Linux side
  - 10 Pings
  - PING sends a datagram to VSE
  - VSE sends an answer back to Linux
  - Time until answer arrives is measured
    - Round trip time

Latency (Round trip time) - results

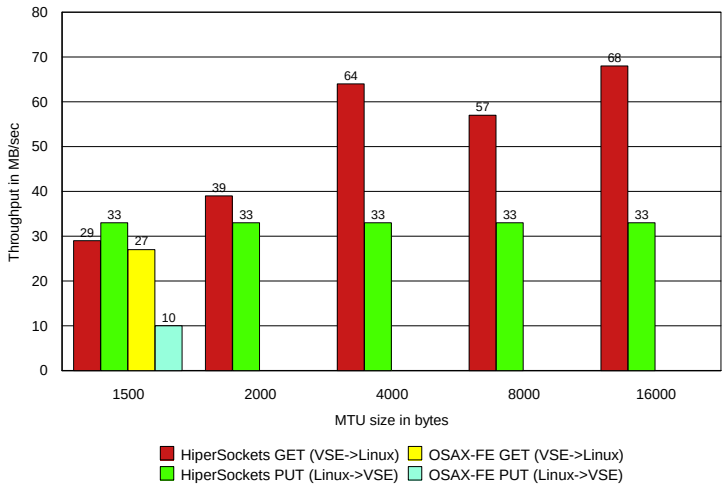


HiperSockets is about 1.8 times faster in terms of latency

Throughput (MB/sec)

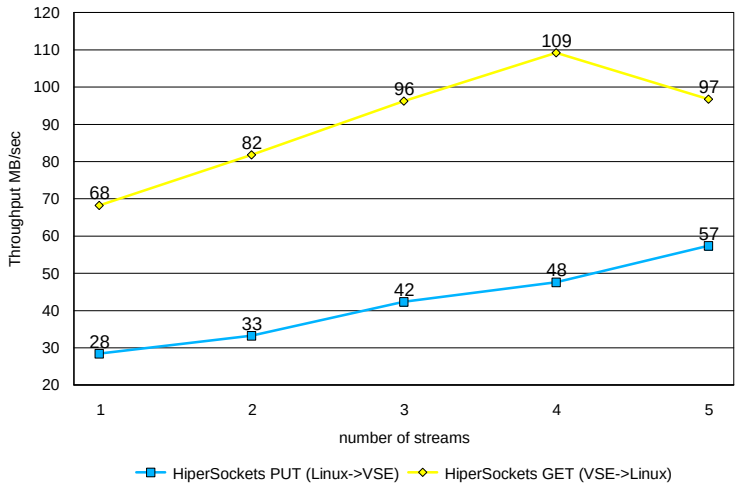
- Measurements has been done with FTP
  - Initiated at the Linux side
  - Transferring 1GB (1000MB)
    - without translation (binary)
    - 1 to 5 parallel streams
  - PUT: send data to VSE
    - VSE inbound
    - sending a 1GB file to $NULL file (in memory file)
    - No file I/O is done by VSE/Linux
  - GET: receive data from VSE
    - VSE outbound
    - receiving $NULL file (in memory file) into /dev/null
    - No file I/O is done by VSE/Linux

Throughput (MB/sec) - results



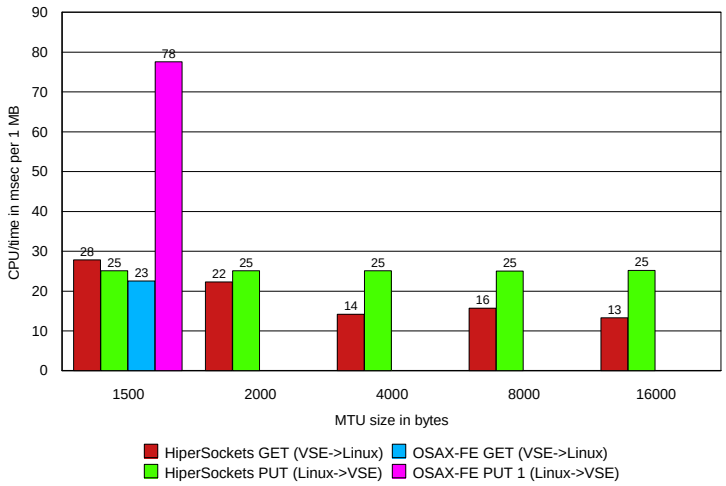
HiperSockets throughput is between 30-80 MB/sec

Throughput (MB/sec) - results (2)



Max. HiperSockets throughput of 109 MB/sec at 4 concurrent connections

CPU time per MB - results

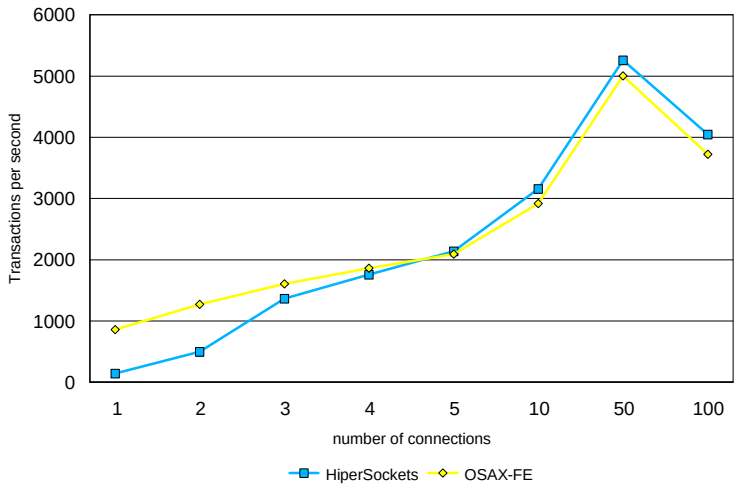


About 15-30 msec CPU time per MB for HiperSockets
(on a z800 2066-001)

Transaction per second

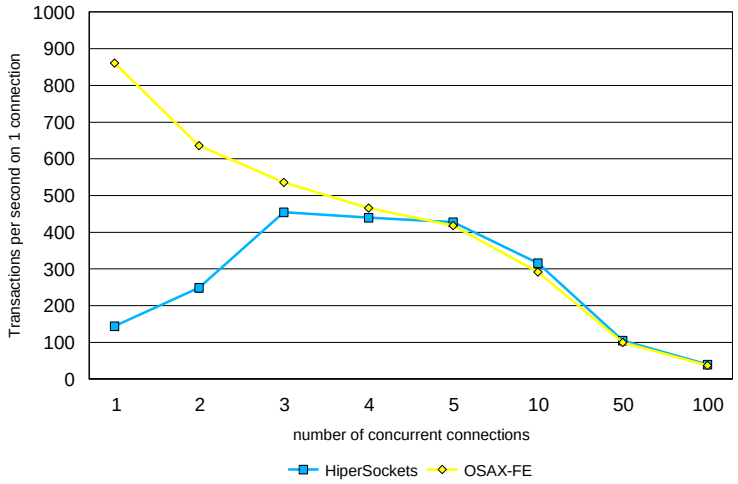
- Measurements has been done with an ECHO server
  - Client on Linux sends 100 bytes to server
  - Server on VSE echoes 100 bytes
  - Per TCP connection 10000 transactions are driven
  - Variations: Number of TCP connections
    - 1,2,3,4,5
    - 10,50,100
  - Measurements
    - Transactions per second
    - CPU time per transaction

Transactions per second - results



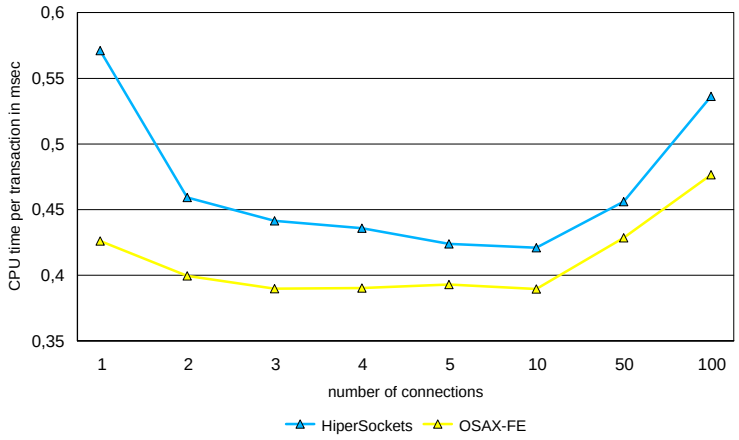
Maximum of 5200 transactions per second at 50 concurrent connections

Transactions per second on 1 connection - results



HiperSockets: Maximum of about 450 transactions per second on 1 connection (= about 2 msec response time)

CPU time per transaction



HiperSockets: About 0.45 msec CPU time per transaction
for 2-50 connections

Measurement Results - conclusion

- HiperSockets
  - Throughput
    - Between 30-80 MB/sec
    - Maximum throughput of 109 MB at 4 concurrent connections
    - About 15-30 msec CPU time per MB
  - Transactions per second
    - Maximum of 5200 Transactions per second at 50 concurrent connections
    - About 0.45 msec CPU time per transaction
      - includes ECHO Server code

Further Information

- VSE Homepage:
  http://www.ibm.com/servers/eserver/zseries/os/vse/
- VSE Performance Homepage:
  http://www.ibm.com/servers/eserver/zseries/os/vse/library/vseperf.htm
- Performance Documents from W. Kraemer
  - available on the Performance Homepage
- VSE/ESA e-business Connectors User's Guide
  http://www.ibm.com/servers/eserver/zseries/os/vse/pdf/ieswue20.pdf