IBM Z Systems | **E**nterprise **N**etworking **S**olutions (**ENS**)

V2R4: Z/OS COMMUNICATIONS SERVER PERFORMANCE SUMMARY REPORT

Created By: Kaji Rashad

Last Updated On: May 22nd, 2020

Table of Contents

Trademarks, Notices, and Disclaimers

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

BigInsights
BlueMix
CICS*
COGNOS*
Db2*
DFSMSdfp
DFSMSdss
DFSMShsm
DFSORT
DS6000*
DS8000*
FICON*
GDPS*
HyperSwap
IBM*
IBM (logo)*
IMS
Language Environment*
MQSeries*
Parallel Sysplex*
PartnerWorld*
RACF*
Rational*
Redbooks*
REXX
SmartCloud*
System z10*
Tivoli*
UrbanCode
WebSphere*
z13
z14
z15
zEnterprise*
z/OS*
zSecure
z Systems
z/VM*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies:
Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce and is registered in the U.S. Patent and Trademark Office.
Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
OpenStack is a trademark of OpenStack LLC. The OpenStack trademark policy is available on the OpenStack website.
Red Hat is a trademark of Red Hat Inc, an IBM Company.
TEALEAF is a registered trademark of Tealeaf, an IBM Company.
Windows Server and the Windows logo are trademarks of the Microsoft group of countries.
Worklight is a trademark or registered trademark of Worklight, an IBM Company.
UNIX is a registered trademark of The Open Group in the United States and other countries.
* Other product and service names might be trademarks of IBM or other companies.

Notes:
Performance results are 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. 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. This information provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g, zIIPs, zAAPs, and IFLs) ("SEs"). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the “Authorized Use Table for IBM Machines” provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html (“AUT”). No other workload processing is authorized for execution on an SE. IBM offers SE at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Acknowledgments

The z/OS Communications Server Performance Team would like to thank the following IBMers for their input on this report:

Dhananjay Patel **|** Senior Software Engineer **|** z/OS Communications Server

David Herr **|** Senior Software Engineer **|** z/OS Communications Server

Christopher Nyamful **|** Software Engineer **|** z/OS Communications Server

Michael Fitzpatrick **|** Senior Technical Staff Member **|** Architecture & Design for ENS

Tips for Reading This Document

1) Clicking on any row in the Table of Contents will take the reader to that specific section or subsection of the document[1]
2) All hyperlinks redirect to an external webpage or internal section/sub-section

---

[1] PDF application must support this feature

Preface

The performance measurements discussed in this document were collected using dedicated system environments. Results obtained in other configurations or operating system environments may vary significantly depending upon environments used. Therefore, no assurance can be given, and there is no guarantee that an individual user will achieve performance or throughput improvements equivalent to the results stated here. Readers of this document should verify the applicable data for their specific environment.

The Central Processor Unit (CPU) numbers listed includes only z/OS host networking related CPU overhead (including dispatching costs) on **general Central Processors** (CPs) from the network device driver layer up through the application socket layer. The socket applications used in the micro-benchmarks for this publication have *no application logic*, so the CPU numbers represent the total application cost which in this case equates to the network related costs. With typical production workloads, network related cost is a *small fraction* of the overall application transaction's cost.

Note: In all benchmarks, the best practices recommended by z/OS Communications Server were utilized *when* applicable:

- ✓ INBPERF DYNAMIC
    - ○ WORKLOADQ (IWQ)
- ✓ SEGMENTATIONOFFLOAD (LSO)
- ✓ AUTODELAYAck
- ✓ QDIOACCELerator
- ✓ MsgWaitAll[2]
- ✓ Jumbo Frames (e.g. HOST MTU 8192)

---

[2] A socket read flag utilized by the application to instruct the TCP layer to delay completion of a Socket Receive or Read call until the full length of the requested data is available in the TCP receive buffer [1].

Hardware Information

z14
Machine Type (Model): 3906 – M04

z15
Machine Type (Model): 8561 – T01

x86
x86 Intel Based Blade Center: 16-way Intel X5570

Workload Naming Convention

Introduction

You decipher the listed workloads in the following way:
[NameOfBenchmark][#OfClients](BytesSentByClient/BytesSentByServer)

For example, [RR][10](1B/100B) is interpreted as Request Response benchmark with 10 clients sending 1 byte and receiving 100 bytes from the server.

Generic Workloads

RRx(y/z): x number of clients doing **R**equest **R**esponse transactions where the client is opening a connection and performing a series of transactions sending y bytes and receiving a response of z bytes
CRRx(y/z): x number of clients doing **C**onnect **R**equest **R**esponse transactions where the client is performing a series of transactions opening a connection, sending y bytes, receiving a response of z bytes, and closing the connection
STRx(y/z): x number of clients doing **Str**eaming transactions where the client is sending y bytes and receiving a response of z bytes

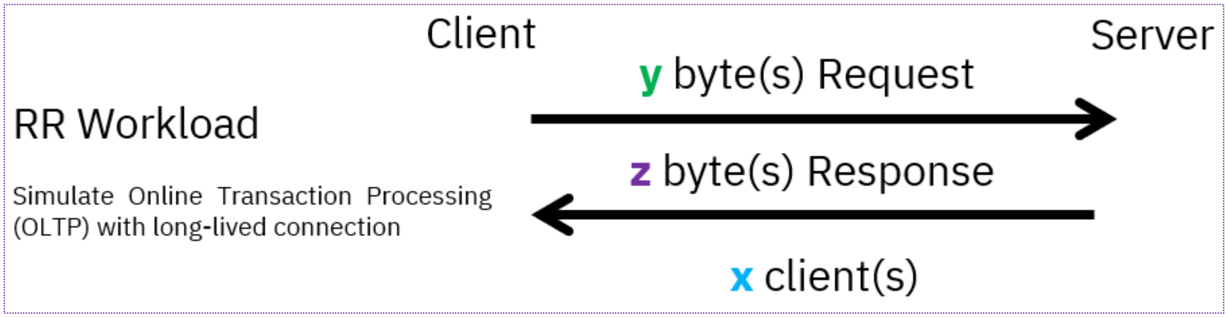

Figure 1: Request Response Workload

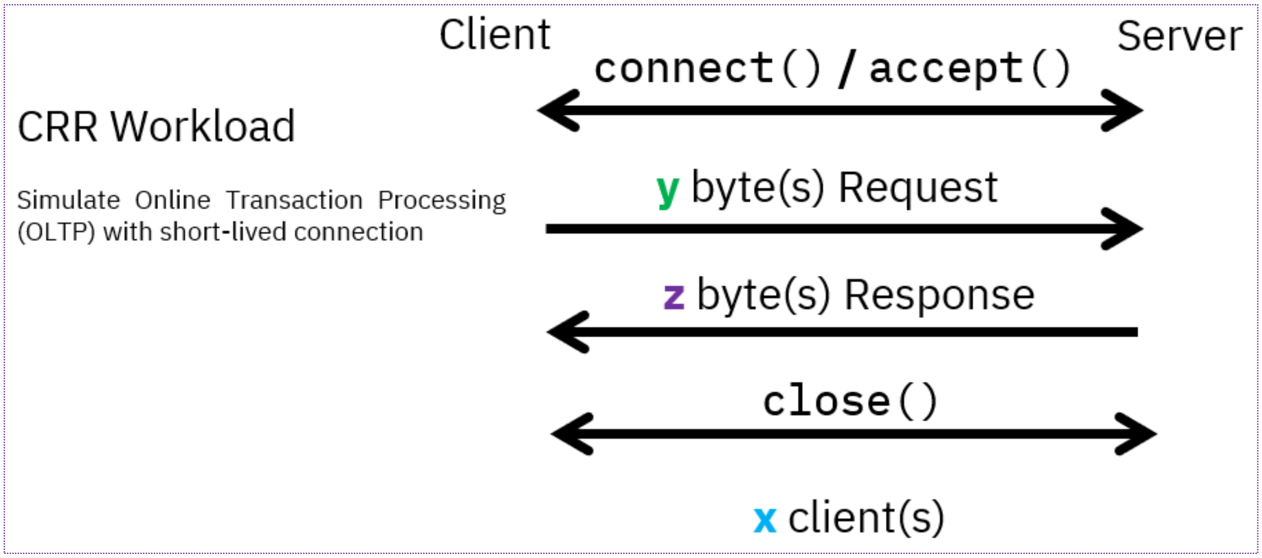

Figure 2: Connect Request Response Workload

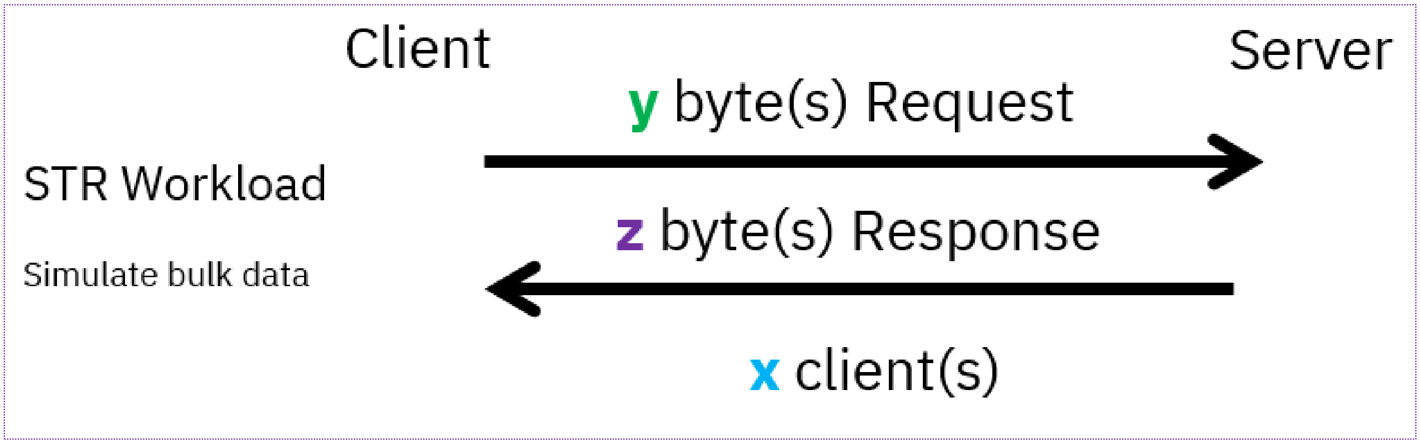

Figure 3: Streaming Workload

Examples

RR40(100B/100B): In an instance of time, there are 40 clients browsing a webpage hosted on a server in which each HTTP GET request of 100B contains a response of 100B.

CRR9(200B/200B): In an instance of time, there are 9 clients sending a HTTP GET request containing 200B and receives a response containing 200B which allows them to log into their bank portal. The core difference between a RR and CRR workload is the duration of the connection. In CRR, the connection is closed after each transaction. A common use case for a bank portal is logging in to audit the balance before logging out.

STR3(1B/20MB): In an instance of time, there are 3 clients sending a 1B request and receiving a 20MB file in response.

Performance Best Practices

INBPERF DYNAMIC

Processing inbound traffic for the OSA-Express interface in Queued Direct Input Output (QDIO) mode dynamically exploits an OSA hardware function called Dynamic LAN Idle. The DYNAMIC setting reacts to changes in traffic patterns and dynamically sets the interrupt-timing values to maximize throughput without incurring additional CPU consumption. Refer to this article for more information.

QDIO Inbound Workload Queueing (IWQ)

The core benefits of IWQ are “finer tuning of read-side interrupt frequency to match the latency demands of the various workloads that are serviced” and “improved multiprocessor scalability as multiple OSA-Express input queues are efficiently serviced in parallel” [2]. Each queue is tailored for its specific need. For instance, the bulk queue is tailored for improved “in-order packet delivery on multiprocessor, which likely results in improvements to CPU consumption and throughput” [2]. Refer to this article for more information.

SEGMENTATIONOFFLOAD (LSO)

Any large amount of data traveling over the network is broken down into smaller segments. This process can be CPU intensive. As an alternative, segmentation offload (i.e. Large Send Offload) is an OSA-Express feature. It reduces host CPU utilization, increases data transfer efficiency, and offloads segmentation processing to OSA [3].

AUTODELAYAck

Reduction in network traffic and CPU utilization can be achieved by delaying the TCP acknowledgement (ACK) *depending* on the traffic pattern. AUTODELAYAck enables the TCP stack to “automatically enable or disable a delayed ACK in a TCP connection based on the characteristic of the traffic” [4].

QDIOACCELerator

QDIO Accelerator specifies that inbound packets that are to be forwarded by a TCP/IP stack are eligible to be routed directly between any of the following combinations of interface types: a HiperSockets interface and an OSA-Express QDIO interface, two OSA-Express QDIO interfaces, and two HiperSockets interfaces. These packets do not need to be sent to this TCP/IP stack for forwarding. Therefore, valuable TCP/IP resources (storage and CPU) are not expended for purposes of routing and forwarding packets. This option also applies to packets that would be forwarded by the Sysplex Distributor. Refer to this article more information.

MsgWaitAll

MstWaitAll is beneficial in streaming workloads. The flag bit decreases the frequency of interrupts occurring for the application receiving data as less interrupts can result in improvements to CPU consumption and throughput. The receiving application is interrupted only when all requested data can be returned. To avoid blocking the application indefinitely, the flag bit should only be set in scenarios where the application expects to receive enough data to fill its buffer or the connection will terminate.

Jumbo Frames

When a client and host communicate with each other over a network, it is possible to utilize a higher Maximum Transmission Unit (MTU) size if the *entire network path* supports it. A higher MTU size can reduce the amount of segmentation for larger payloads, which may result in a higher throughput and reduced CPU cycles [5]. If Jumbo Frames is configured, then enable path MTU discovery.

V2R4: New Function

zCX

Background

IBM z/OS Container Extensions (zCX) enables end users to deploy and manage any open source or Linux on Z application as a Docker image without any modification in the z/OS ecosystem. Refer here to learn more about this new advancement. All the following measurements were collected with APARs OA58296, OA58300 and PH16581 applied.

z/OS to zCX Co-Located Versus z/OS to x86 Docker

The following configuration creates an environment where a client application, running within a Docker container in a zCX instance on z/OS, communicates with a server application running on the same z/OS system.

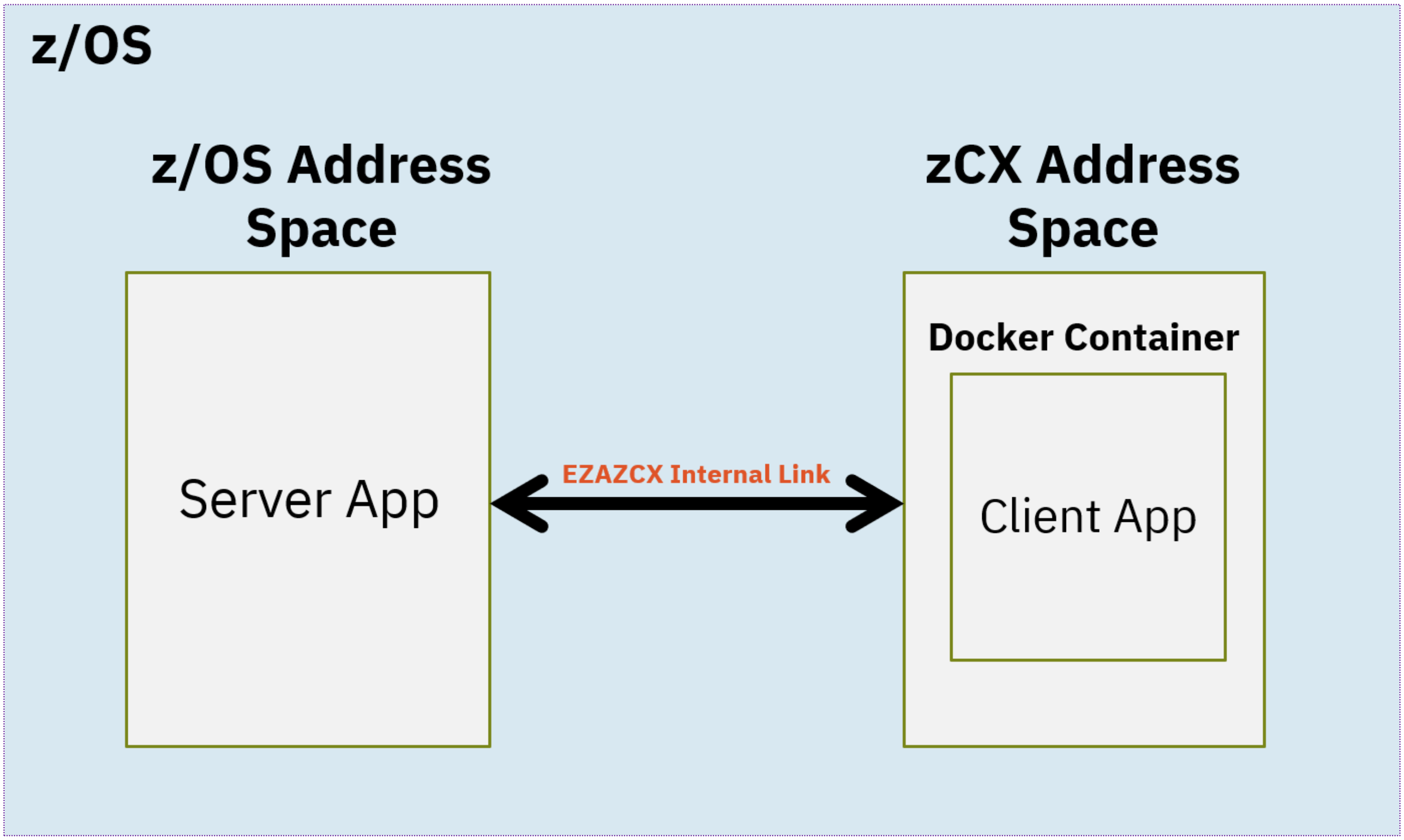

Figure 4: zCX client to z/OS server traversing a virtual network

This is compared against a configuration where a client application, running within a Docker container on an x86 server, communicates over a network with a server application running on z/OS. This comparison highlights the advantage of running your client and server application on the same system versus communicating over a network connection.

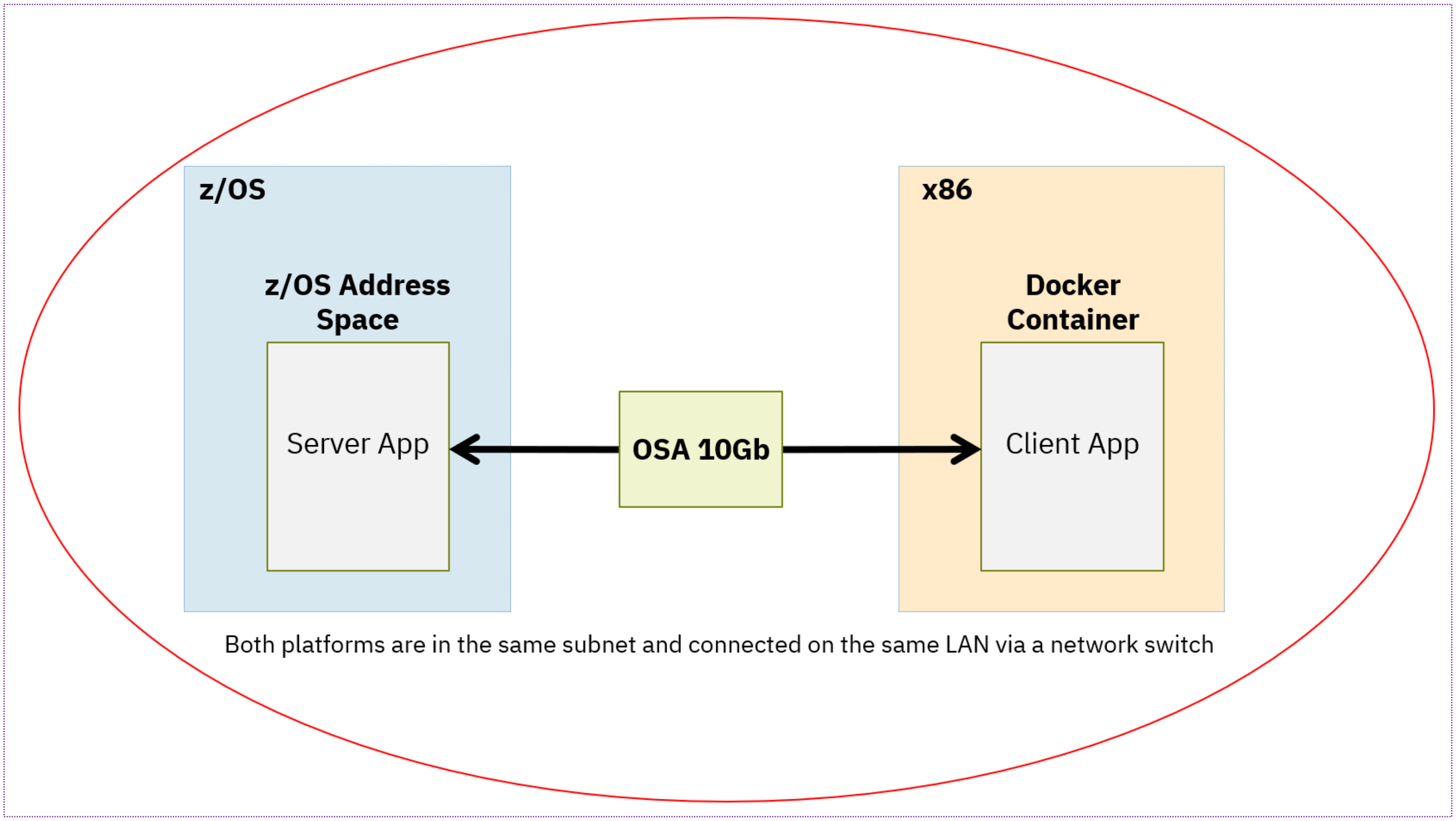

Figure 5: Distributed Docker client to z/OS server traversing a physical network

z/OS Environment Configuration

Below is the environment configuration in which the data was collected:

- Central Processor Complex (CPC): z14
- Release: V2R4
- Number of CPUs: 4 (Dedicated)
- Physical Memory: 64GB
- Interface: OSA-Express 6S 10Gb
    - MTU: 1492 Bytes
- Workloads
    - RR10(4KB/4KB)
    - STR1(1B/20MB)
    - STR1(20MB/1B)

Additional zCX Environment Configuration

The zCX environment configuration consists of the above configuration plus the following:

- z Integrated Information Processors (zIIPs): 4 (SMT-1)
- Virtual CPs assigned to container: 3
- Physical Memory assigned to container: 4GB
- Interface: EZAZCX
    - MTU: 1492 Bytes

x86 Environment Configuration

- IBM BladeCenter X5570
    - 16 CPs
    - Physical Memory: 48GB
    - Single-Tenant Physical Server
- Linux OS: RedHat Enterprise Linux Server 7.7
- Docker Image: Ubuntu 18.04
- Interface: NetXtreme 10GbE
    - MTU: 1492 Bytes

If an x86 virtual server had been utilized, then the following lab measurements would have shown favorable results for zCX because zCX is virtualized by design. Also, the x86 server was in proximity to the z/OS system. In real world scenarios, the increased latency between the x86 server and z/OS would have amplified the co-location benefits between zCX and z/OS.

Co-Located Versus x86 Docker Observations

One can assume that a co-located environment will offer *much less* transaction latency because there is no physical network traversed for communication. This assumption is exactly what was observed in the lab. For RR and STR workloads, transaction rate and throughput improved while transaction latency reduced significantly. In lab measurements, the following observations were made:

- RR workloads in a co-located environment provided
    - Up to 81% improvement in transactions per second
    - Up to 45% reduction in transaction latency
- STR workloads in a co-located environment provided
    - Up to 205% improvement in throughput
    - Up to 67% reduction in transaction latency

From the observations, *if you currently host your application as a Docker image on distributed systems while accessing databases or services on z/OS, you may achieve a significant reduction in transaction latency by migrating the Docker image to a zCX instance on the same z/OS system where the database or service resides*.

RR & STR Performance: Co-Located Versus x86 Docker

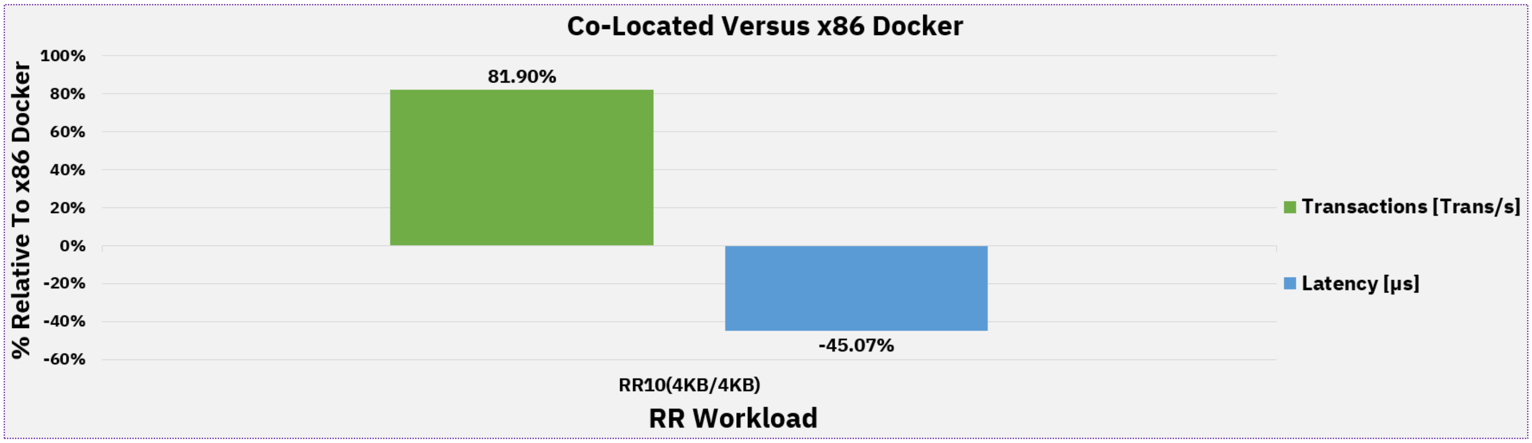

Figure 6: A co-located zCX instance improves transaction rate and reduces transaction latency for RR workloads

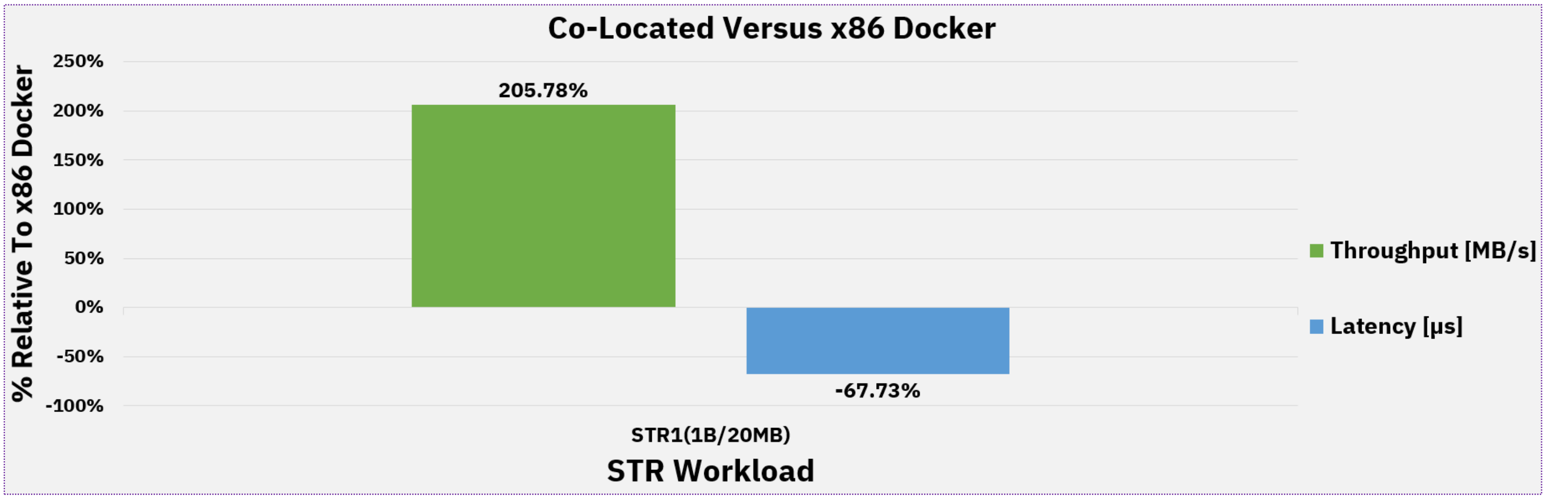

Figure 7: A co-located zCX instance improves throughput and reduces transaction latency for STR workloads

Higher MTU Benefits for Co-Located Workloads

Since there is no physical network for communication between co-located client and server applications, a larger MTU size should provide some benefit. For STR workloads, increasing the MTU size should result in a higher throughput as less time is spent on segmenting the data being sent. The maximum zCX interface MTU size is 65535 bytes. In lab measurements, the following observations were made:

- Increasing the MTU size from 1492 bytes to 65535 bytes provided
  - Up to 79% improvement in throughput
  - Up to 44% reduction in transaction latency
  - Up to 34% reduction in network related Server CPU (z/OS utilization on general CPs) cost per MB
  - Up to 60% reduction in network related Client CPU (zCX utilization on zIIPs) cost per MB

From the observations, *it is recommended to configure a higher MTU size for the zCX interface when the client and server are co-located on the same z/OS system*.

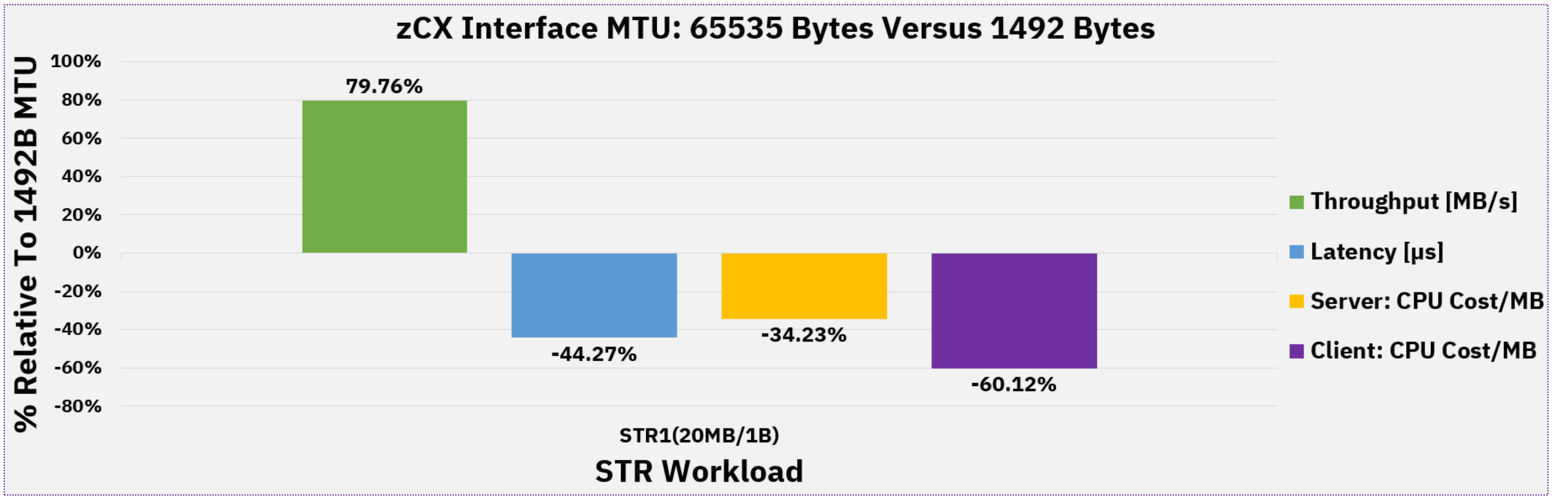

Figure 8: A larger zCX interface MTU size of 65535 bytes for STR workloads improves throughput and reduces Client & Server CPU cost per MB

IWQ Versus Non-IWQ

The recommended best practice of configuring IWQ to provide better workload traffic separation also applies to inbound traffic to zCX. By configuring IWQ, network traffic destined to zCX will be placed on a separate inbound queue, enabling z/OS Communications Server to maintain in-order processing of the inbound traffic on the zCX queue in parallel with inbound traffic for the other queues on this OSA-Express interface. Additionally, configuring IWQ allows for inbound traffic to be processed directly on zIIPs, helping to reduce the cost of running zCX on z/OS.

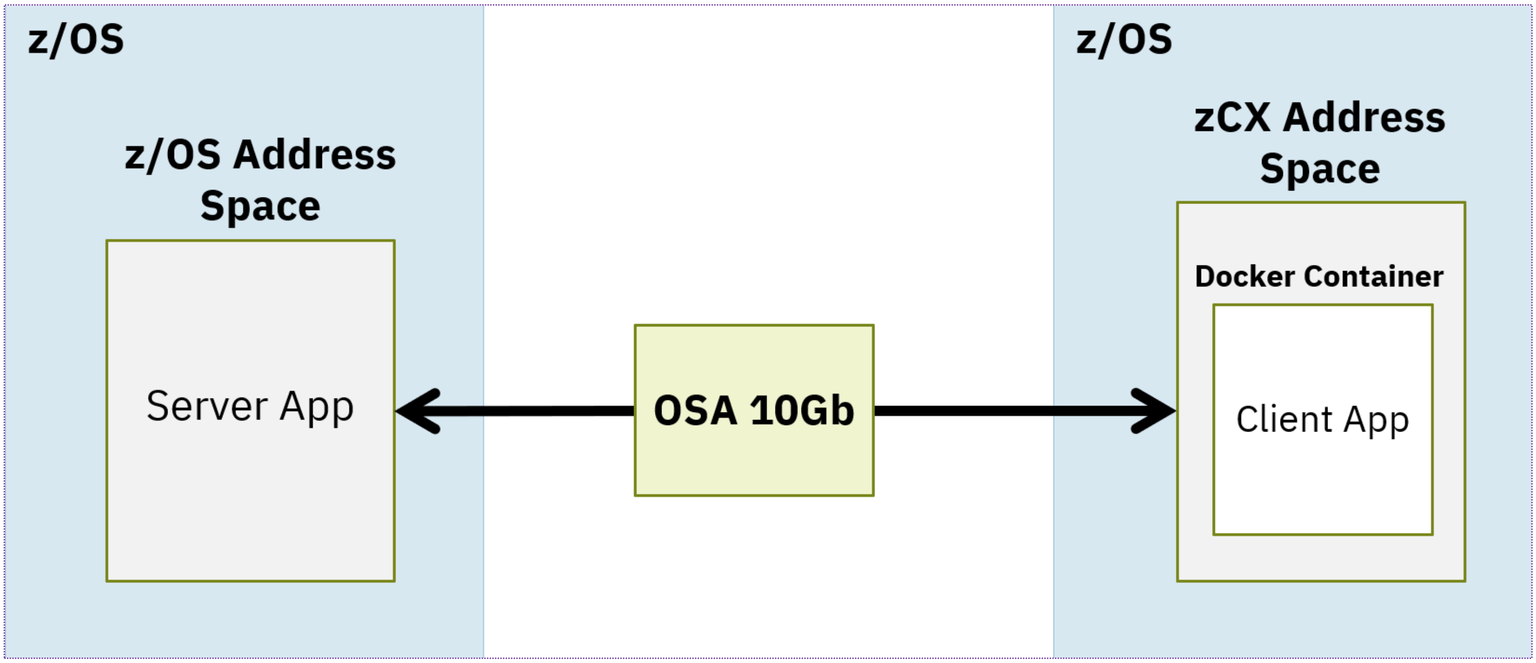

Figure 9: zCX client to z/OS server traversing a physical network

Environment Configuration

The same z/OS Environment Configuration and zCX Environment Configuration listed above (e.g. z/OS Environment Configuration and Additional zCX Environment Configuration) was used.

IWQ Versus Non-IWQ Observations

For RR and STR workloads, IWQ improves the micro-benchmark's transaction rate and throughput respectively, by better preserving the order of packets being delivered to zCX and utilizing zIIPs for much of its communication with zCX. This results in reduced network related Client CPU cost and transaction latency. In lab measurements, the following observations were made:

- Enabling IWQ for RR workloads provided
    - Up to 35% improvement in transactions per second
    - Up to 25% reduction in transaction latency
    - Up to 38% reduction in network related Client CPU (z/OS utilization on general CPs) cost per transaction
- Enabling IWQ for STR workloads provided
    - Up to 5% improvement in throughput
    - Up to 5% reduction in transaction latency
    - Up to 12% reduction in network related Client CPU (z/OS utilization on general CPs) cost per MB

From the observations, *IWQ is the recommended configuration for zCX*. *Also, if you have Jumbo Frames configured for your OSA interfaces, then you should consider specifying the zCX MTU to at least 8992 bytes to improve network communication performance to/from zCX and external peers* ***provided*** *the entire network path supports the larger MTU size.*

RR & STR Performance: IWQ Versus Non-IWQ

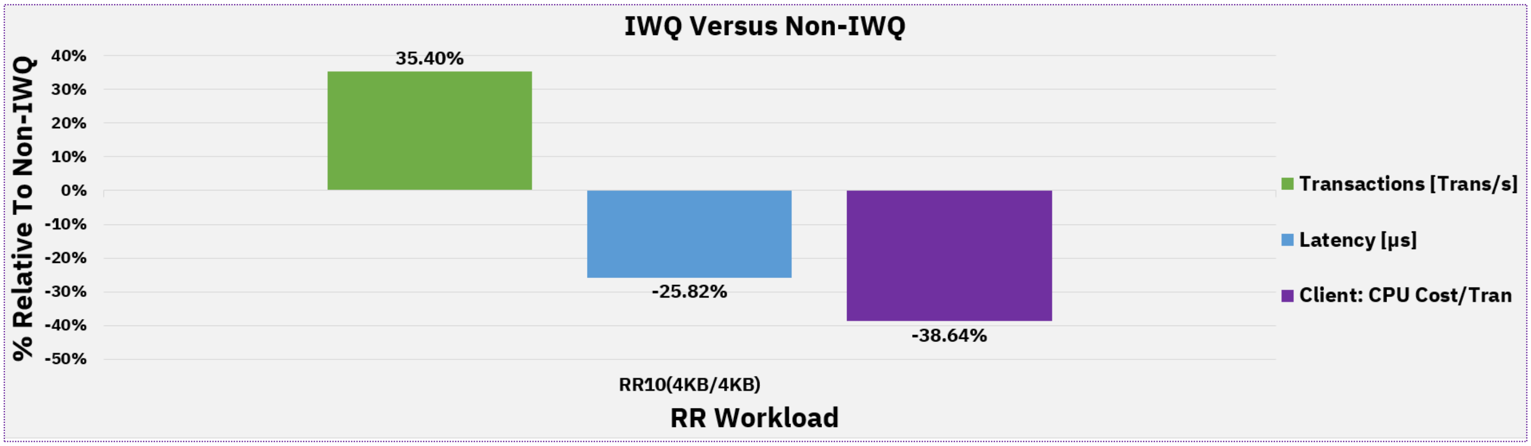

Figure 10: Enabling IWQ for <u>RR</u> workloads improves transaction rate, reduces transaction latency and network related Client CPU cost per transaction

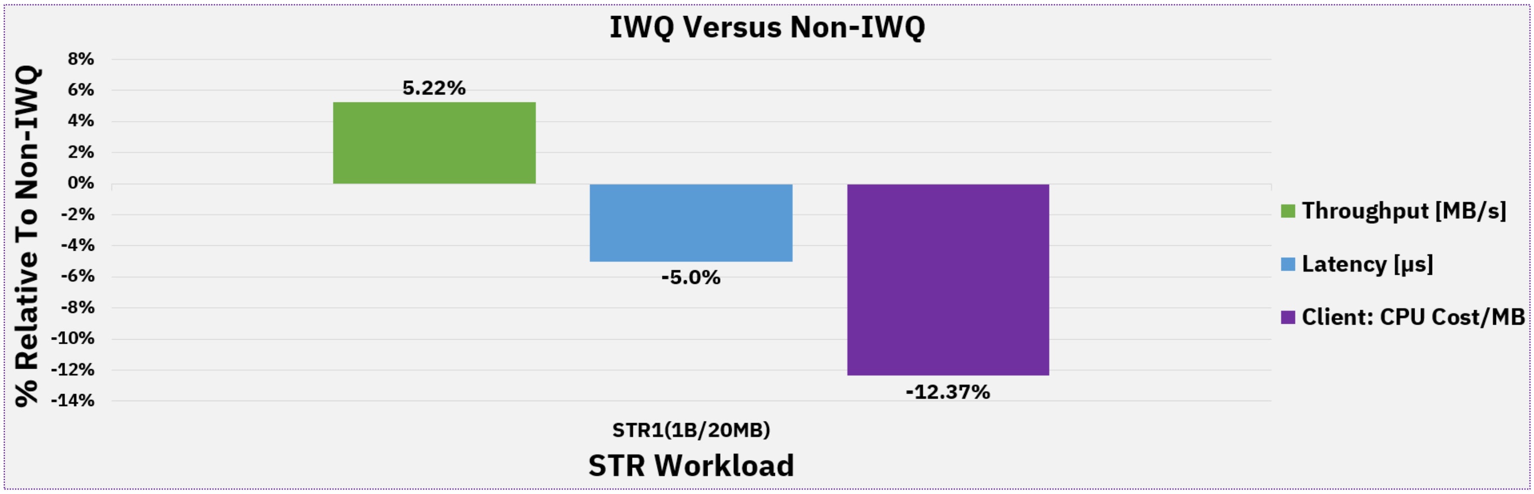

Figure 11: Enabling IWQ for <u>STR</u> workloads improves throughput and reduces network related Client CPU cost per MB

V2R4: OSA-Express Performance

OSA-Express 7S 25Gb

Background

OSA-Express continues to release new models with additional features and hardware updates. In the following measurement, the focus was on benchmarking V2R4 with OSA-Express 7S 25Gb on the latest hardware: z15. Refer here for more background information on Open Systems Adapter (OSA).

z/OS Environment Configuration

Below is the environment configuration in which the data was collected:

- CPC: z15
- Release: V2R4
- Number of CPUs: 4 (Dedicated)
- Interface: OSA-Express 7S 25Gb
- Workloads
    - RR40(100B/100B)
    - CRR10(64B/8KB)
    - STR3(20MB/1B)

Remark

The graphs within this section contains raw data. In other words, it does not make relative comparisons. Instead, the observations sub-section for each type of workload makes the relative comparisons.

STR Observations

For STR workloads, OSA-Express 7S is able to reach *near* line speed with a higher MTU size. In lab measurements, the following observations were made:

- Increasing the MTU size from 1500 bytes to 8992 bytes (*i.e.* using Jumbo Frames) provided
    - Up to 87% improvement in throughput
- In comparison to OSA-Express 7S 10Gb with an MTU size of 8992 bytes, OSA-Express 7S 25Gb with an MTU size of 8992 bytes provided
    - Up to 147% improvement in throughput

From the observations, *it is recommended to utilize a higher MTU size such as Jumbo Frames for STR workloads* ***provided*** *the entire network path supports the larger MTU size.*

STR Throughput: OSA-Express 7S 25Gb vs. OSA-Express 7S 10Gb

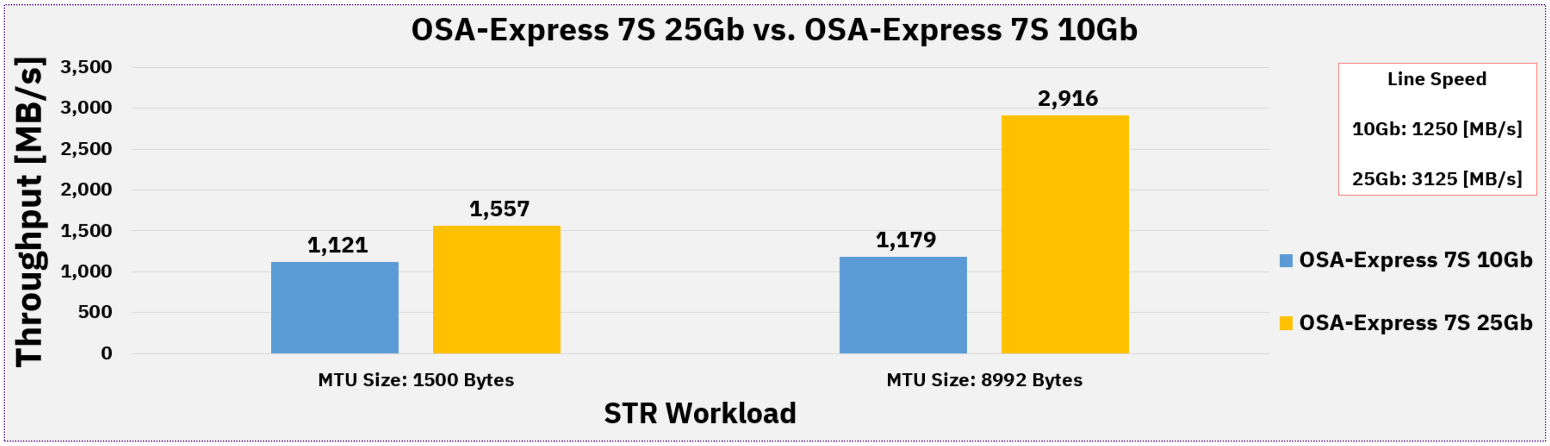

Figure 12: Comparing OSA-Express 7S 25Gb against OSA-Express 7S 10Gb for STR workloads

RR Observations

As with STR workloads, RR workloads with higher payloads may benefit from a higher MTU size as the data traffic requires less segmentation. Increasing the MTU size improves the transaction rate. In lab measurements, the following observations were made:

- Increasing the MTU size from 1500 bytes to 8992 bytes provided
  - Up to 3% improvement in transactions per second
- In comparison to OSA-Express 7S 10Gb with an MTU size of 8992 bytes, OSA-Express 7S 25Gb with an MTU size of 8992 bytes provided
  - Up to 82% improvement in transactions per second

From the observations, *it is recommended to utilize a higher MTU size for RR workloads* ***provided*** *the entire network path supports the larger MTU size.*

CRR Observation

As with RR workloads, increasing the MTU size improves the transaction rate. In lab measurements, the following observation was made:

- Increasing the MTU size from 1500 bytes to 8992 bytes provided
  - Up to 41% improvement in transactions per second

CRR benefits are lower than RR benefits as CRR has several packet flows per transaction to establish and tear down the TCP connection.

RR & CRR Performance: OSA-Express 7S 25Gb vs. OSA-Express 7S 10Gb

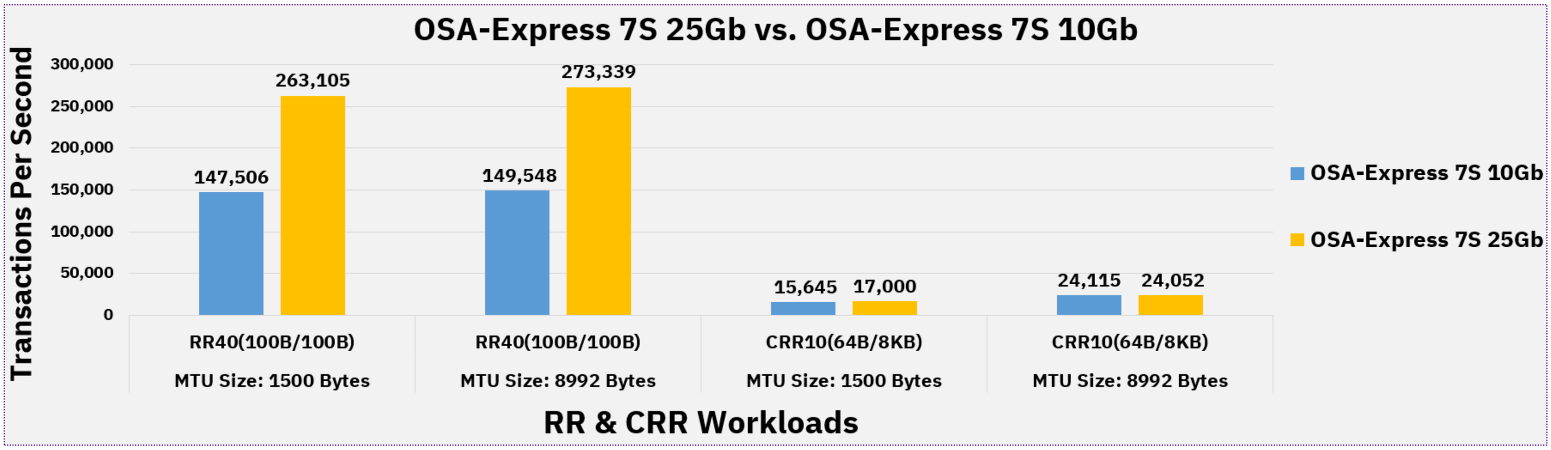

Figure 13: Comparing OSA-Express 7S 25Gb against OSA-Express 7S 10Gb for RR & CRR workloads

OSA-Express 7S 10Gb vs. OSA-Express 6S 10Gb

Introduction

The Background and z/OS Environment Configuration stated in OSA-Express 7S 25Gb is used here. The only differentiating factor is the comparison. In this section, the focus is on OSA-Express 6S 10Gb and OSA-Express 7S 10Gb on z15.

Remark

The graphs within this section contains raw data. In other words, it does not make relative comparisons. Instead, the observations sub-section for each type of workload makes the relative comparisons.

STR Observation

For STR workloads, the following observation was made when comparing OSA-Express 7S 10Gb to OSA-Express 6S 10Gb:

- With a constant MTU size, the throughput is comparable when contrasting OSA-Express 7S 10Gb to OSA-Express 6S 10Gb

STR Throughput: OSA-Express 7S 10Gb vs. OSA-Express 6S 10Gb

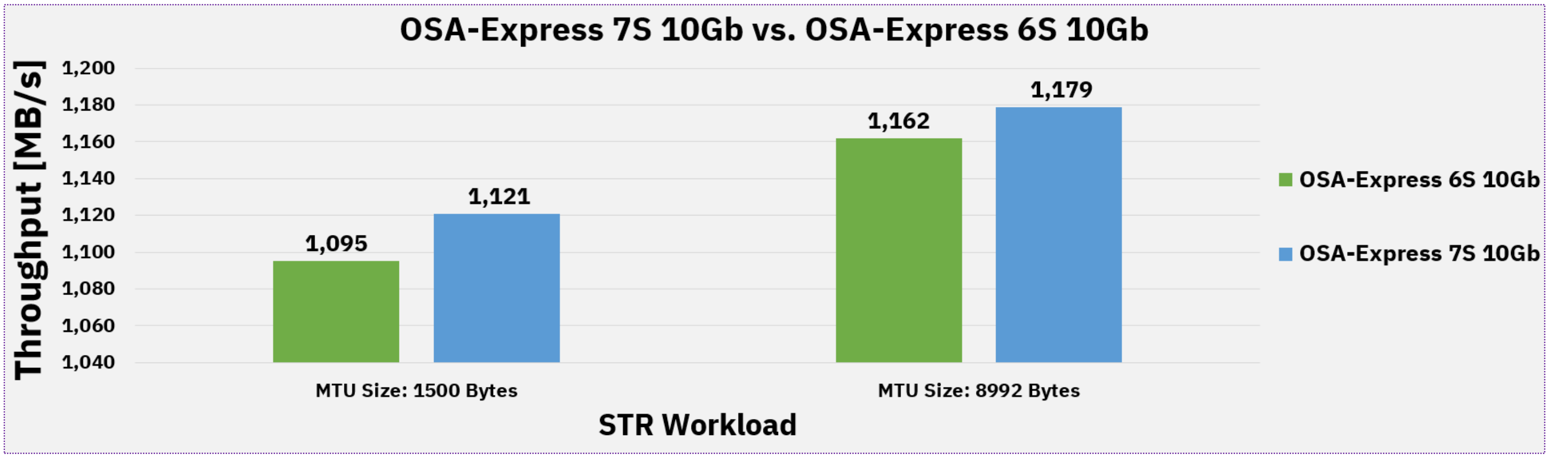

Figure 14: Comparing OSA-Express 7S 10Gb against OSA-Express 6S 10Gb for STR workload

RR Observations

OSA-Express 7S 10Gb offers a moderate improvement in transaction rate as evident by the following observations:

- In comparison to OSA-Express 6S 10Gb with an MTU size of 1500 bytes, OSA-Express 7S 10Gb with an MTU size of 1500 bytes provided
    - Up to 16% improvement in transactions per second
- In comparison to OSA-Express 6S 10Gb with an MTU size of 8992 bytes, OSA-Express 7S 10Gb with an MTU size of 8992 bytes provided
    - Up to 21% improvement in transactions per second

From the observations, *it is recommended to utilize OSA-Express 7S 10Gb for RR workloads as it improves the transaction rate regardless of MTU size.*

CRR Observation

OSA-Express 7S 10Gb offers a minor improvement in transaction rate as evident by the following observation:

- In comparison to OSA-Express 6S 10Gb with an MTU size of 1500 bytes, OSA-Express 7S 10Gb with an MTU size of 1500 bytes provided
    - Up to 15% improvement in transactions per second

RR & CRR Performance: OSA-Express 7S 10Gb vs. OSA-Express 6S 10Gb

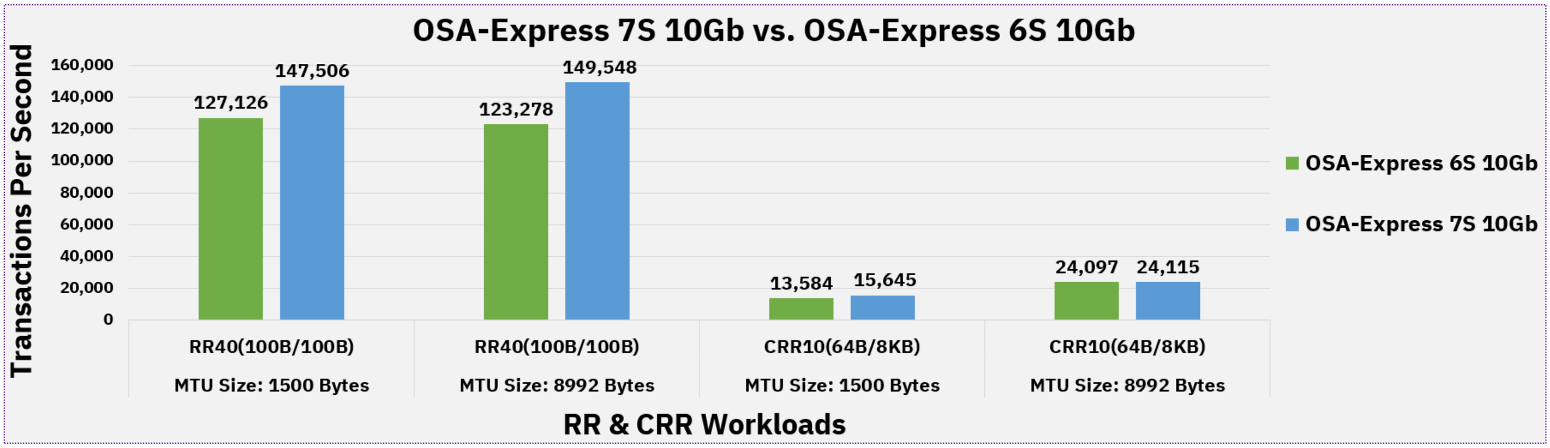

Figure 15: Comparing OSA-Express 7S 10Gb against OSA-Express 6S 10Gb for RR & CRR workloads

V2R4: Hardware Performance

z15 vs. z14: SMC-D & HiperSockets

Background

Shared Memory Communications – Direct Memory Access (SMC-D) extends the benefit of Shared Memory Communications Over RDMA (SMC-R) to applications on the same CPC using Internal Shared Memory (ISM), but without additional hardware. The communicating peers (e.g. TCP/IP stacks) can detect SMC-D capability during TCP/IP connection establishment flows. Refer here for more background information.

The following measurements demonstrate the hardware improvements with z15 for workloads utilizing SMC-D or HiperSockets for network communication.

z/OS Environment Configuration

Below is the environment configuration in which the data was collected:

- CPC: z15 & z14
- Release: V2R4
- Number of CPUs: 4 (Dedicated)
- Interfaces: SMC-D & HiperSockets
  - HiperSockets Maximum Frame Size (MFS): 64KB
    - MTU: 56 KB
- Workloads
  - RR10(1KB/1KB)
  - RR10(4KB/4KB)
  - RR10(8KB/8KB)
  - RR10(16KB/16KB)
  - RR10(32KB/32KB)
  - STR1(1B/20MB)
  - STR3(1B/20MB)
  - STR1(20MB/1B)
  - STR3(20MB/1B)

z15 vs. z14: RR & STR Workload Observations for SMC-D

Workloads utilizing SMC-D can benefit from running on z15 as evident by the following observations:

- Running RR workloads over a SMC-D interface on z15 versus z14 provided
    - Up to 29% improvement in transactions per second
    - Up to 22% reduction in transaction latency
- Running STR workloads over a SMC-D interface on z15 versus z14 provided
    - Up to 52% improvement in throughput

z15 vs. z14 RR & STR Performance: SMC-D

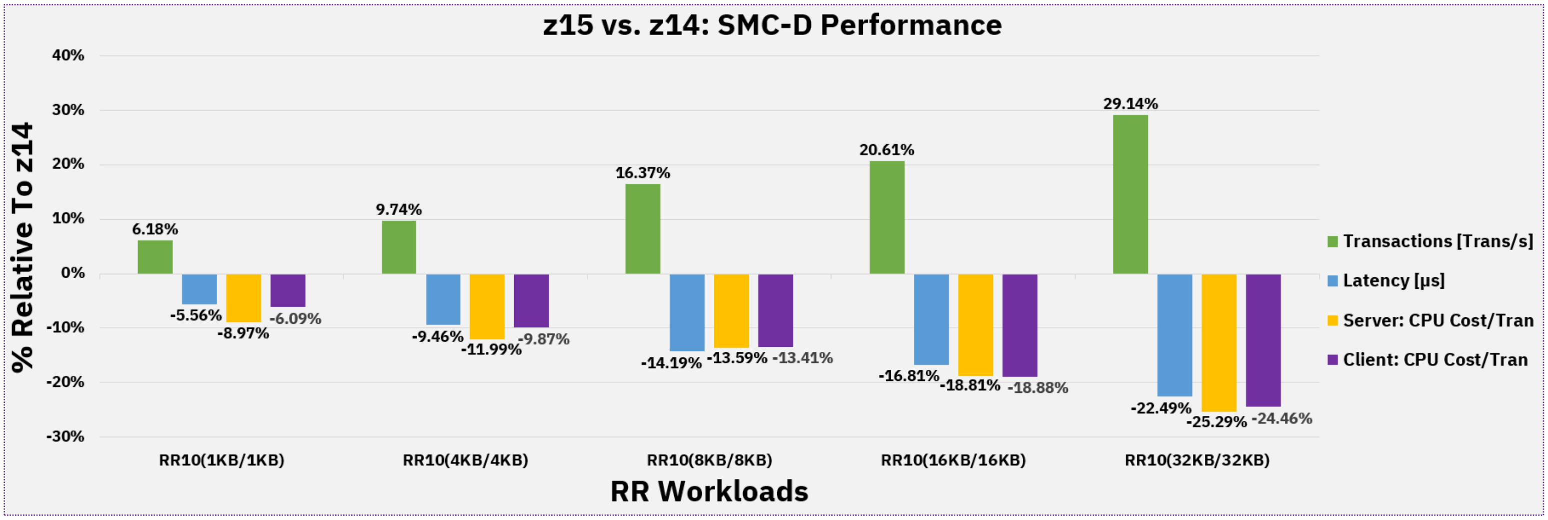

Figure 16: Provides z15 vs. z14 transactions, transaction latency, and Client & Server CPU cost per transaction comparison for SMC-D interface with RR workloads

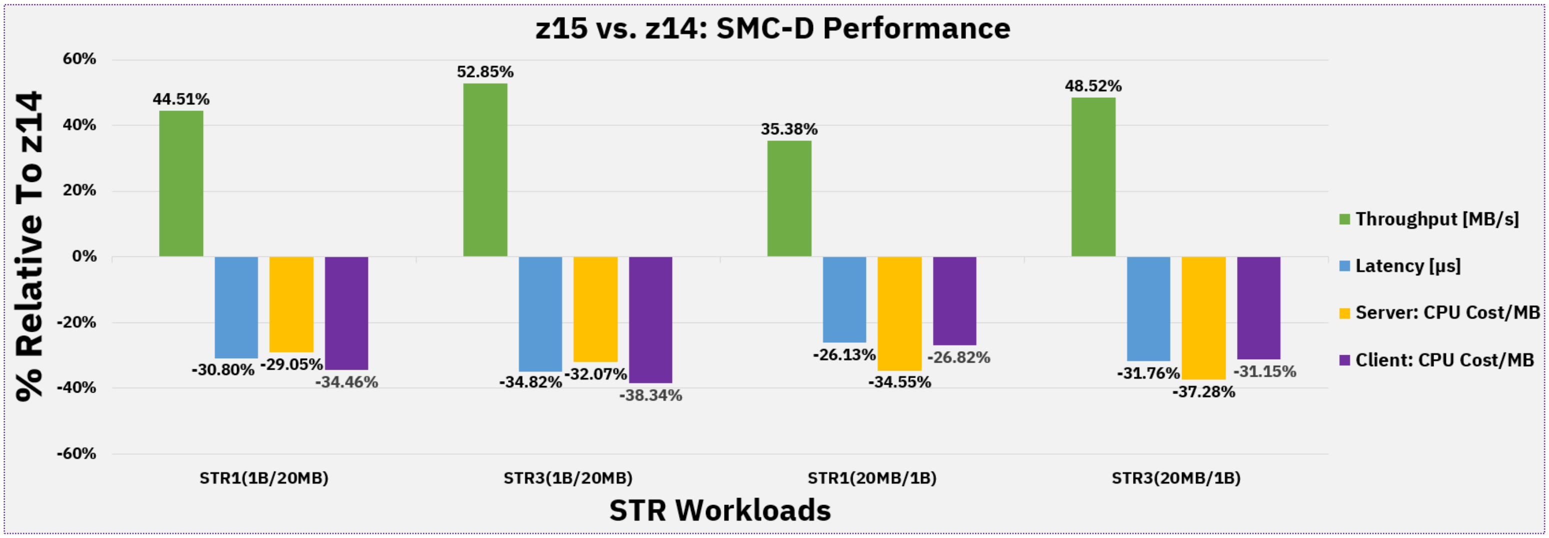

Figure 17: Provides z15 vs. z14 throughput and Client & Server CPU cost per MB comparison for SMC-D interface with STR workloads

z15 vs. z14: RR & STR Workload Observations for HiperSockets

Workloads utilizing HiperSockets can benefit from running on z15 as evident by the following observations:

- Running RR workloads over a HiperSockets interface on z15 versus z14 provided
    - Up to 10% improvement in transactions per second
    - Up to 9% reduction in transaction latency
- Running STR workloads over a HiperSockets interface on z15 versus z14 provided
    - Up to 22% improvement in throughput

z15 vs. z14 RR & STR Performance: HiperSockets

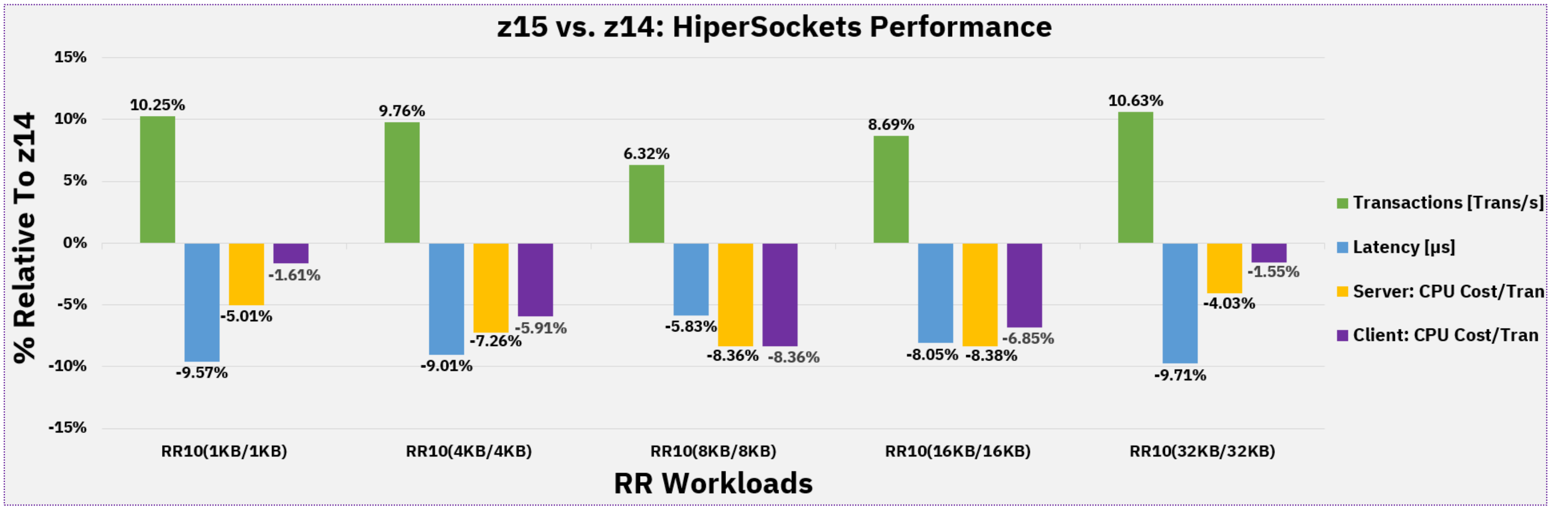

Figure 18: Provides z15 vs. z14 transactions, transaction latency, and Client & Server CPU cost per transaction comparison for HiperSockets interface with RR workloads

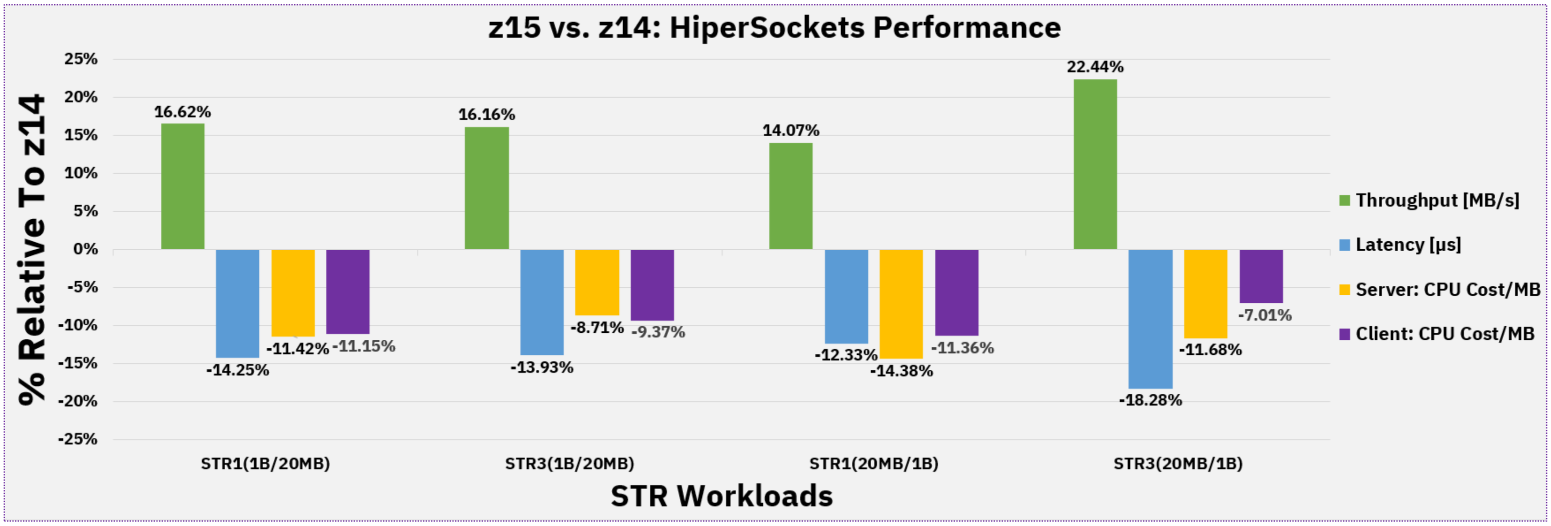

Figure 19: Provides z15 vs. z14 throughput and Client & Server CPU cost per MB comparison for HiperSockets interface with STR workloads

z15: RR & STR Workload Observations for SMC-D vs. HiperSockets

It is highly recommended to utilize the SMC-D interface when possible as it offers significant benefits as evident by the following observations:

- Running RR workloads over a SMC-D interface versus a HiperSockets interface on z15 provided
    - Up to 203% improvement in transactions per second
    - Up to 66% reduction in transaction latency
    - Up to 68% reduction in network related Client and Server CPU cost per transaction
- Running STR workloads over a SMC-D interface versus a HiperSockets interface on z15 provided
    - Up to 471% improvement in throughput
    - Up to 79% reduction in network related Server CPU cost per MB
    - Up to 80% reduction in network related Client CPU Cost per MB

z15 RR & STR Performance: SMC-D vs. HiperSockets

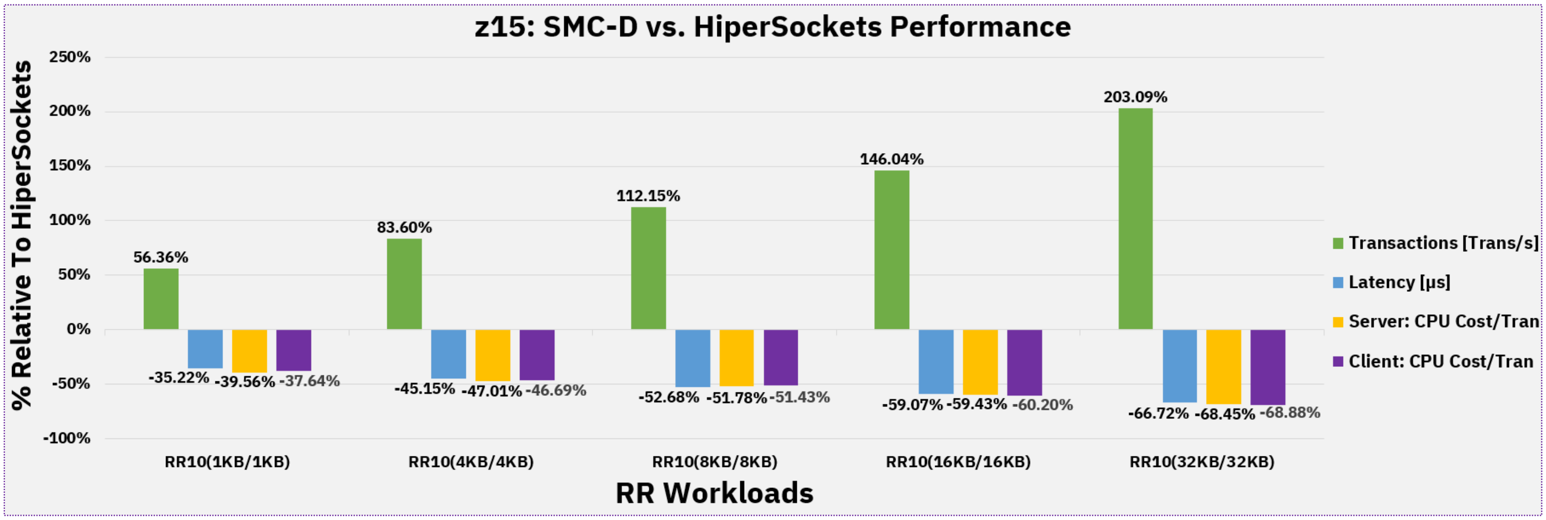

Figure 20: Provides z15 transactions, transaction latency, and Client & Server CPU cost per transaction comparison for SMC-D & HiperSockets interfaces with RR workloads

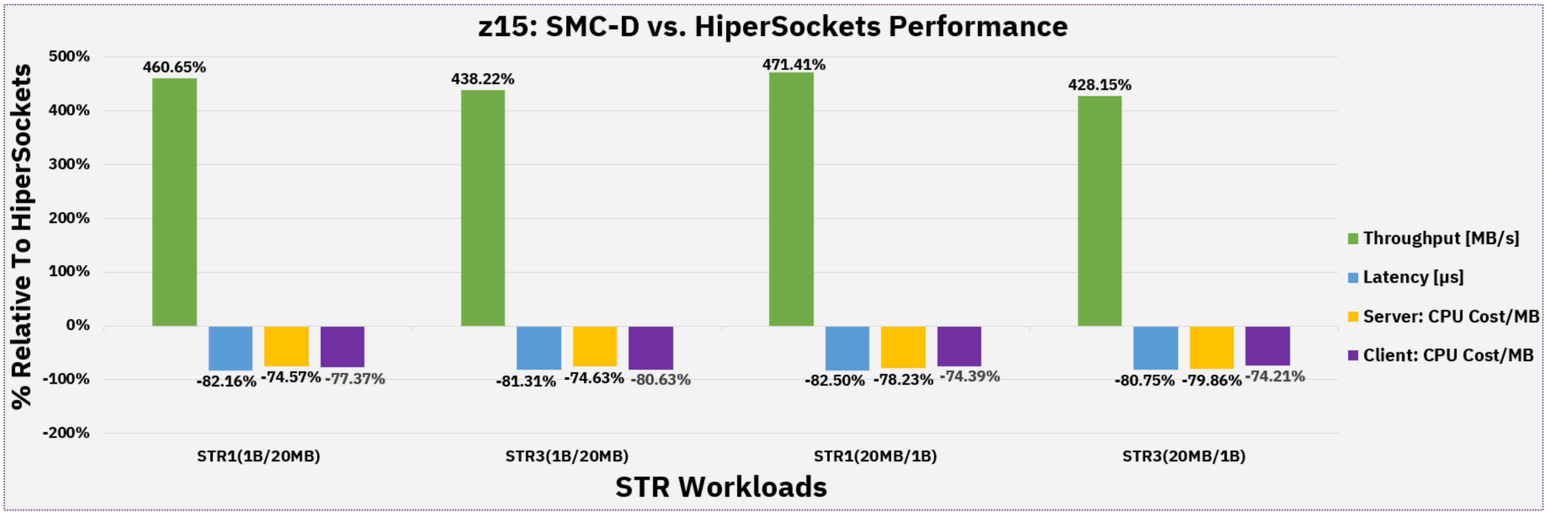

Figure 21: Provides z15 throughput and Client & Server CPU cost per MB comparison for SMC-D & HiperSockets interfaces with STR workloads

z14: RR & STR Workload Observations for SMC-D vs. HiperSockets

With the z14 measurements, the aforementioned recommendation is reiterated as evident by the following observations:

- Running RR workloads over a SMC-D interface instead of a HiperSockets interface on z14 provided
    - Up to 159% improvement in transactions per second
    - Up to 61% reduction in transaction latency
    - Up to 59% reduction in network related Client and Server CPU cost per transaction
- Running STR workloads over a SMC-D interface instead of a HiperSockets interface on z14 provided
    - Up to 381% improvement in throughput
    - Up to 71% reduction in network related Client and Server CPU cost per MB

z14 RR & STR Performance: SMC-D vs. HiperSockets

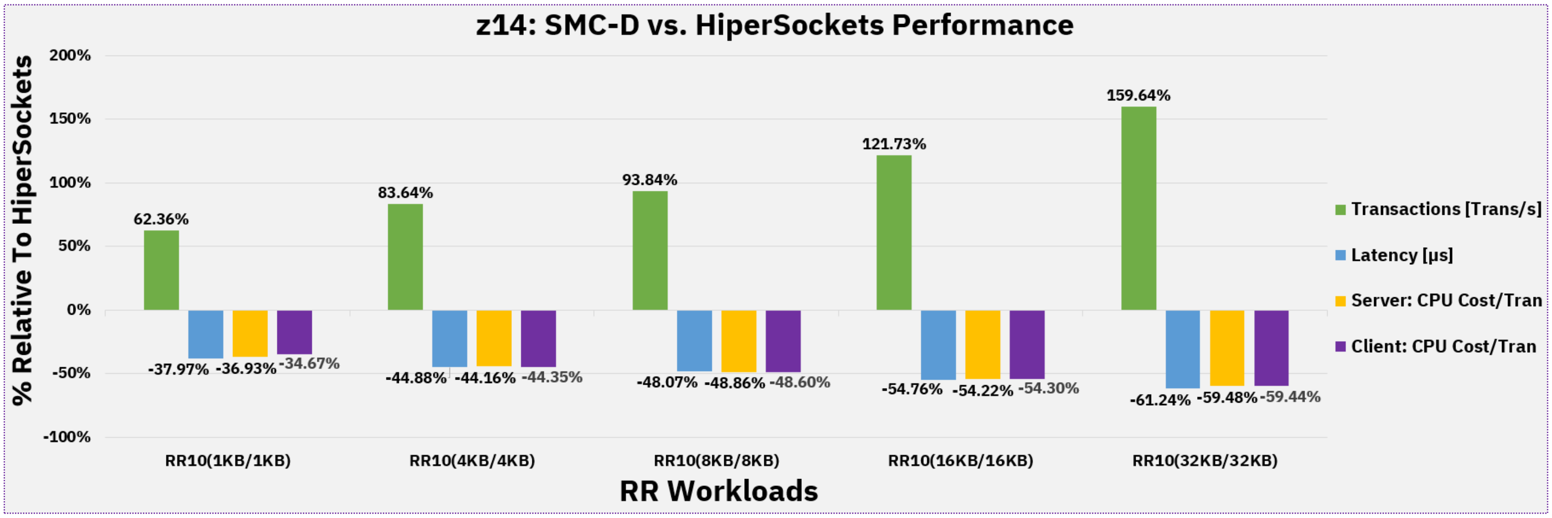

Figure 22: Provides z14 transactions, transaction latency, and Client & Server CPU cost per transaction comparison for SMC-D & HiperSockets interfaces with RR workloads

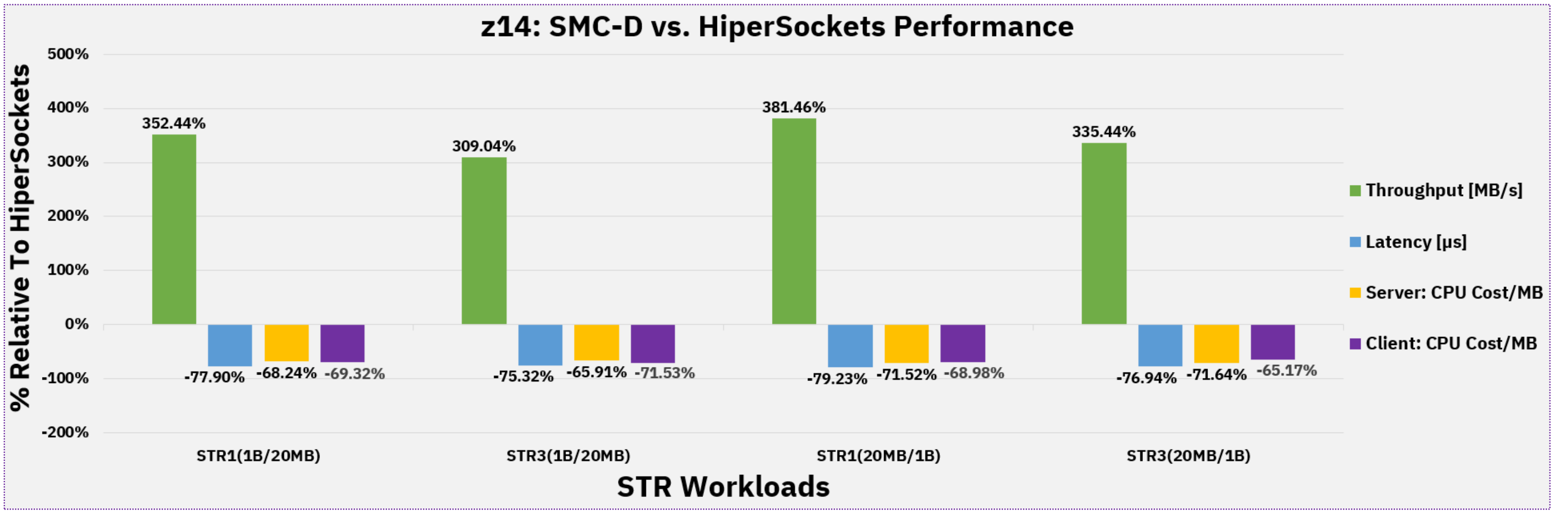

Figure 23: Provides z14 throughput and Client & Server CPU cost per MB comparison for SMC-D & HiperSockets interfaces with STR workloads

SMC Applicability Tool

Many customers express interest in Shared Memory Communications (SMC). However, they are not quite sure of SMC's full potential in their environment. With expertise and significant time commitment, one can determine their environment traffic patterns that can take advantage of SMC.

SMC Applicability Tool (SMCAT) alleviates a customer's significant time commitment by monitoring and evaluating their TCP/IP network traffic. A system administrator can utilize the tool's evaluation to determine the applicability of SMC in their ecosystem. To enable SMCAT, refer here.

z15 vs. z14: SMC-R

Background

SMC-R is a protocol solution that is based on sockets over Remote Direct Memory Access (RDMA). SMC-R enables TCP sockets applications to transparently use RDMA, which enables direct, high-speed, low-latency, memory-to-memory (peer-to-peer) communication. TCP/IP stacks dynamically learn about the shared memory capability between client and server applications by using traditional TCP/IP connection establishment flows, enabling the TCP/IP stacks to switch from TCP network flows to more optimized direct memory access flows that use RDMA. Refer here for more background information.

The following measurements demonstrate the hardware improvements with z15 for workloads utilizing SMC-R for network communication.

z/OS Environment Configuration

Below is the environment configuration in which the data was collected:

- CPC: z15 & z14
- Release: V2R4
- Number of CPUs: 4 (Dedicated)
- Interface: RoCE Exp2 25GbE[3]
    - MTU: 1KB
- Workloads
    - RR10(4KB/4KB)
    - STR3(1B/20MB)
    - STR3(20MB/1B)

---

[3] z15 utilized an upgraded RoCE Exp2 25GbE card

z15 vs. z14: RR & STR Workload Observations for SMC-R

Utilizing SMC-R provides a CPU cost reduction on z15 as evident by the following observations:

- Running RR workloads over a SMC-R interface on z15 versus z14 provided
  - Up to 12% improvement in transactions per second
  - Up to 11% reduction in transaction latency
  - Up to 19% reduction in network related Server CPU cost per transaction
- Running STR workloads over a SMC-R interface on z15 versus z14 provided
  - Up to 39% reduction in network related Server CPU cost per MB

z15 vs. z14 RR & STR Performance: SMC-R

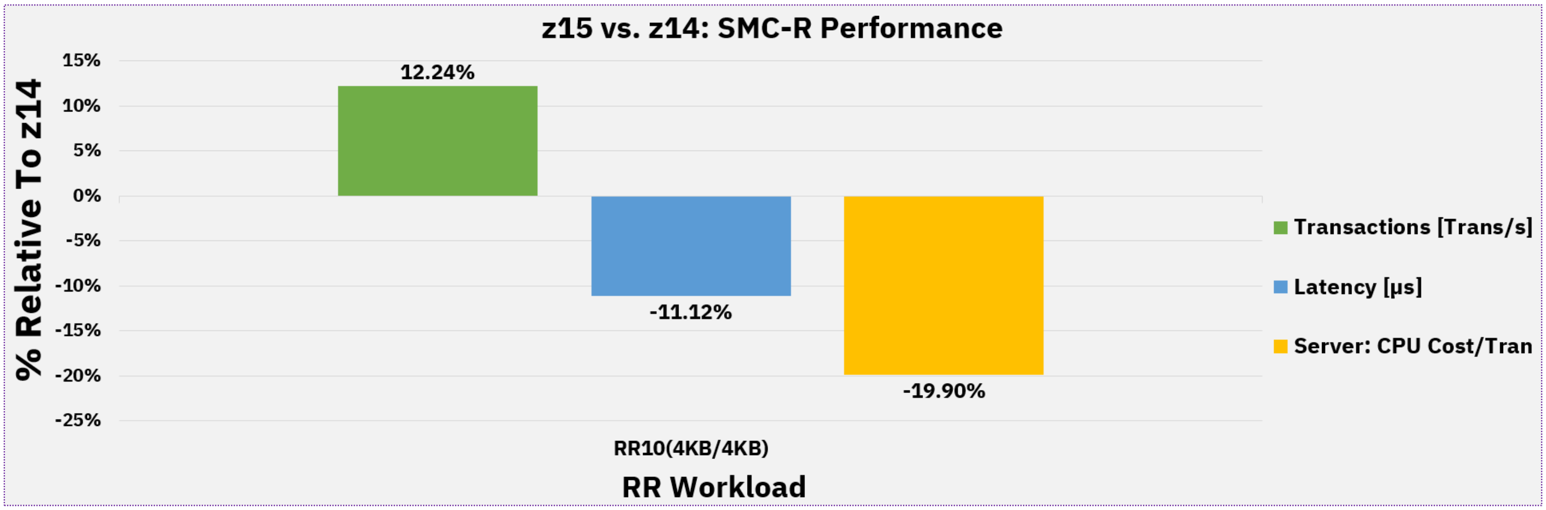

Figure 24: Provides z15 vs. z14 transactions, transaction latency, and Server CPU cost per transaction comparison for SMC-R interface with <u>RR</u> workloads

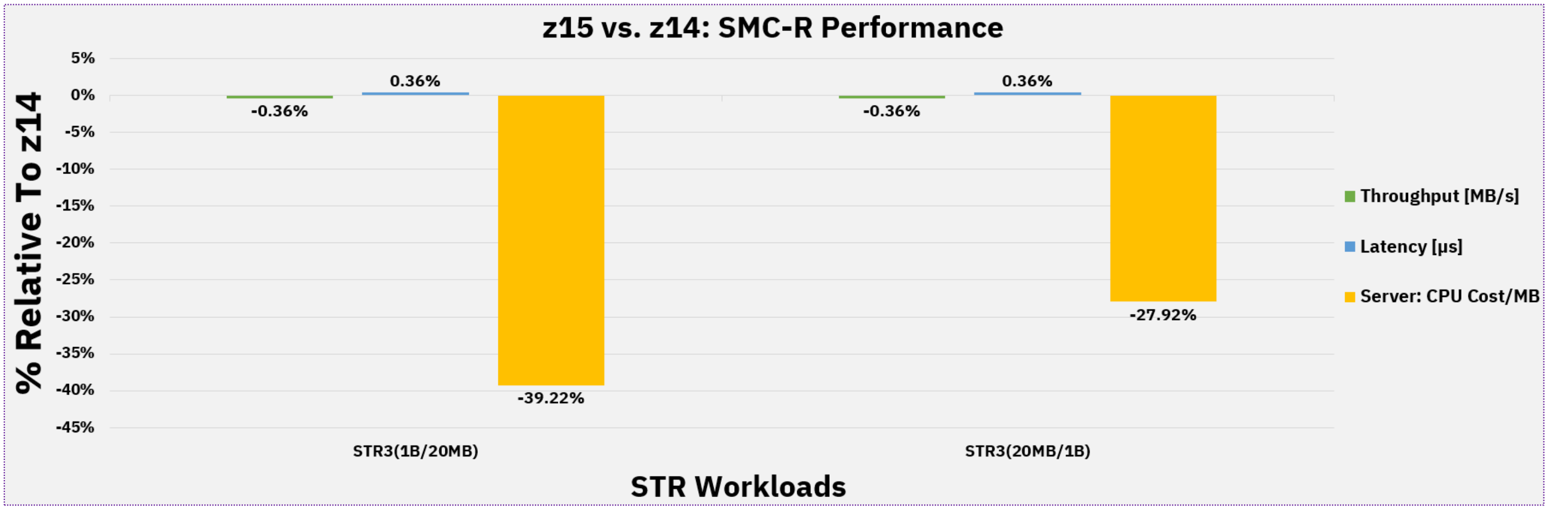

Figure 25: Provides z15 vs. z14 throughput and Server CPU cost per MB comparison for SMC-R interface with <u>STR</u> workloads

z15 vs. z14: Security Improvements

Background

IBM Z systems contain two hardware components enabling significant advantages for encrypting/decrypting network traffic. IBM Z processors have Central Processor Assist for Cryptographic Function (CPACF) for high-speed symmetric crypto operation while Crypto Express 6S and Crypto Express 7S are used for offloading costly asymmetric crypto functions.

z/OS Environment Configuration

Below is the environment configuration in which the data was collected:

- CPC: z15 & z14
- Release: V2R4
- Number of CPUs: 4 (Dedicated)
- Cryptographic Card (Co-Processors) Adapters
  - z14
    - Crypto Express 6S: 1 (Dedicated)
  - z15
    - Crypto Express 6S: 1 (Dedicated)
    - Crypto Express 7S: 1 (Dedicated)
- TLSv1.2 Cipher: TLS_RSA_WITH_AES_128_GCM_SHA256
  - Signing Algorithm: SHA256RSA
  - Key Length: 2048
- Workloads
  - RR40(100B/100B)
  - RR10(1KB/1KB)
  - CRR10(2KB/2KB)
  - CRR20(64B/8KB)

RR & CRR Workload Observations for z15 TLSv1.2 Crypto Express 7S vs. z14 TLSv1.2 Crypto Express 6S

On z15, it was observed that Crypto Express 7S improved transactions and reduced Client and Server CPU cost per transaction as evident by the following:

- Encrypting and decrypting RR workloads on z15 with Crypto Express 7S versus z14 with Crypto Express 6S provided
  - Up to 18% improvement in transactions per second
  - Up to 33% reduction in network related Server CPU cost per transaction
  - Up to 29% reduction in network related Client CPU cost per transaction
- Encrypting and decrypting CRR workloads on z15 with Crypto Express 7S versus z14 with Crypto Express 6S provided
  - Up to 11% improvement in transactions per second
  - Up to 15% reduction in network related Server CPU cost per transaction
  - Up to 14% reduction in network related Client CPU cost per transaction

RR & CRR Performance: z15 TLSv1.2 Crypto Express 7S vs. z14 TLSv1.2 Crypto Express 6S

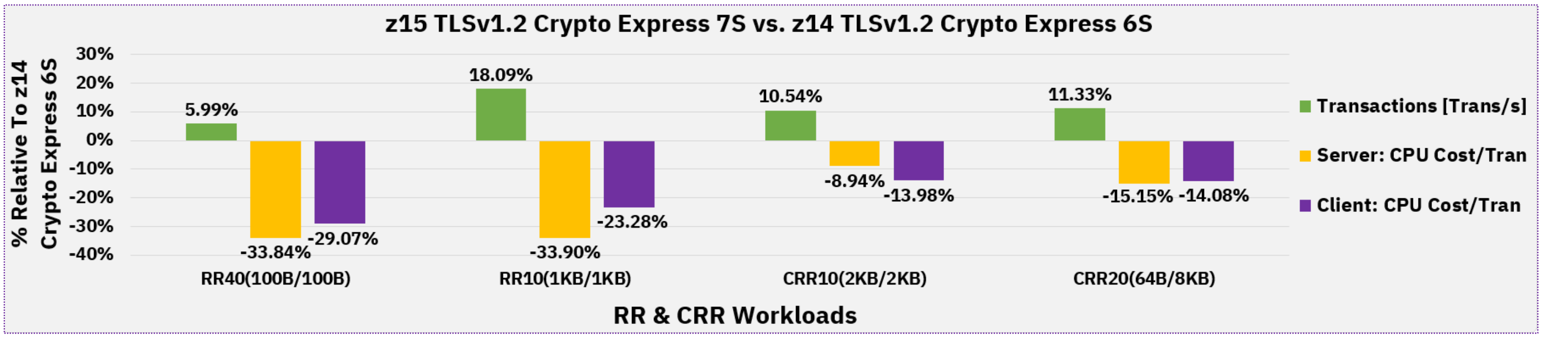

Figure 26: Provides z15 TLSv1.2 Crypto Express 7S vs. z14 TLSv1.2 Crypto Express 6S transactions and Client & Server CPU cost per transaction comparison for RR & CRR workloads

On the z15 environment, the differences between running a Crypto Express 6S versus a Crypto Express 7S are negligible. The above improvements were the result of faster hardware processors.

V2R4 vs. V2R3: Release to Release Performance Comparison

V2R4 vs. V2R3

Introduction

In this sub-section, the pure focus was benchmarking the latest release, V2R4, against the previous release, V2R3.

z/OS Environment Configuration

Below is the environment configuration in which the data was collected:

- CPC: z14
- Release: V2R4 & V2R3
- Number of CPUs: 4 (Dedicated)
- Interface: OSA-Express 6S 10Gb
- Workloads
    - RR40(100B/100B)
    - CRR10(64B/8KB)
    - STR3(1B/20MB)
    - STR3(20MB/1B)

Synopsis

Performance of V2R4, which consists of new functions and improved existing functions, is just as good as V2R3.

FTP

Background

File Transfer Protocol (FTP) allows a user to transfer data sets and files from one host to another. It enables batch transfer jobs. For more information, refer here.

Synopsis

In comparison to V2R3, the performance was equivalent.

TN3270E

Background

TN3270 Enhanced (TN3270E) is a Telnet server enabling users to remotely access their host application. It provides access to z/OS VTAM SNA applications on the MVS host. For more information, refer here.

Synopsis

In comparison V2R3, the performance was equivalent.

IPv6 vs. IPv4

Background

z/OS Communications Server supports IPv6 and IPv4 internet protocol addresses. In comparison to IPv4, IPv6 provides no practical limit on global addressability.

Synopsis

In z/OS V2R4, IPv6 performance was equivalent to IPv4.

References

z/OS Communications Server Performance Index

The following index contains all z/OS Communications Server Performance related publications. The posted materials are updated as necessary.

URL: https://www.ibm.com/support/pages/node/317829

Share 2020 Winter Technical Conference

G. Kassimis, S. Reynolds, *The 2020 Edition: What's New in z/OS Communications Server? – Part 1 of 2*. RTP: IBM, 2020, [Online].
Available: share.confex.com/data/handout/share/134/Session_26075_handout_13326_0.pdf.

B. Rau, J. Stevens, *IBM z/OS Communications Server Shared Memory Communications (SMC)*. RTP: IBM, 2020, [Online].
Available: share.confex.com/data/handout/share/134/Session_26068_handout_13353_0.pdf.

B. Rau, J. Stevens, *Getting the most out of OSA and HiperSockets with z/OS Communications Server*. RTP: IBM, 2020, [Online].
Available: share.confex.com/data/handout/share/134/Session_26060_handout_13352_0.pdf.

M. Fitzpatrick, *Optimizing your network encryption with z14 & z15*. RTP: IBM, 2020, [Online]. Available: share.confex.com/data/handout/share/134/Session_26074_handout_13626_0.pdf.

M. Fitzpatrick, *Understanding z/OS Communications Server Storage Usage*. RTP: IBM, 2020, [Online].
Available: share.confex.com/data/handout/share/134/Session_26072_handout_13645_0.pdf.

G. Kassimis, S. Ogonsula, *z/OS V2R4: Introduction to z/OS Container Extensions: Running Linux on z Docker Containers Inside z/OS*. RTP: IBM, 2020, [Online].
Available: share.confex.com/data/handout/share/134/Session_26204_handout_13518_0.pdf.

Additional References

[1] C. Rufus, *IBM z/OS V2R1 Communications Server TCP/IP Implementation Volume 3: High Availability, Scalability, and Performance,* 1st ed. USA: IBM, 2013, pp. 292 - 293

[2] "*QDIO inbound workload queueing",* IBM. Accessed On: Mar. 31, 2020. [Online]. Available: Here

[3] D. Herr, *Getting the most out of your OSA (Open Systems Adapter) with z/OS Comm Server*. RTP: IBM, 2013, [Online]. Available: share.confex.com/share/121/webprogram/Handout/Session13222/SHARE%20OSA_Boston.pdf.

[4] B. White, O. Ferreira, T. Missawa, T. Sudewo, *IBM z/OS V2R2 Communications Server TCP/IP Implementation: Volume 3 High Availability, Scalability, and Performance.* USA: IBM, 2016, pp: 305.

[5] "*Fragmentation consideration*", IBM. Accessed On: Mar. 31, 2020. [Online]. Available: Here