Exploring the performance of network adapters for Linux on IBM Z

Nils Hoppmann
Software Performance Analyst

IBM

Notices and disclaimer

U.S. Government Users Restricted Rights – use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed "as is" without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.

IBM products are manufactured from new parts or new and used parts. In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply.

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environment. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

Notices and disclaimer (continued)

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of 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. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

Refer to www.ibm.com/legal for further legal information.

IBM, the IBM logo, and ibm.com are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Contents

Contents (continued)

Introduction

IBM® provides two types of physical network adapters for Linux® standard networking usage on IBM Z®: OSA-Express and RoCE Express.

In this presentation, the performance of the most recent OSA-Express, OSA-Express7S 25 GbE, is compared to its predecessor, OSA-Express6S 10 GbE, on an IBM z15™. To complete the picture, the 25 GbE RoCE Express2.1 is added to the comparison in order to highlight its strengths and weaknesses against OSA-Express.

Before focusing on the performance comparison of the different devices [slides 16-29], the setup [slides 7-9], workload [slides 10-11] and performance metrics [slides 12-15] are explained. The comparison itself is split in two major sections differing in the MTU size (1500 B, 9000 B) used during the measurements.

The general performance comparison is followed by a more detailed view at OSA-Express7S streaming performance [slides 31-34].

Completing this presentation is a note on the benefits and downsides of configuring receive packet steering (RPS) with OSA-Express7S and RoCE Express2.1 [slides 36-46].

Note: Performance data contained herein was generally obtained in a controlled, isolated environment. Actual performance, cost, savings or other results in other operating environments may vary.

General setup

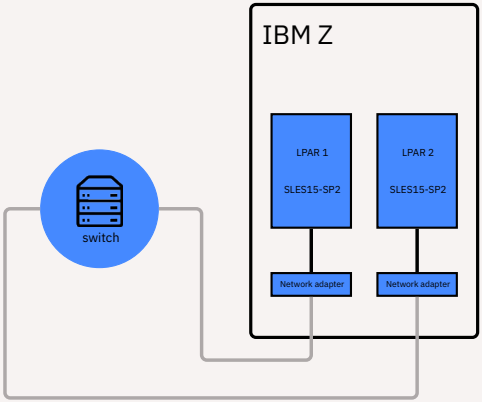
IBM z15

Systems: 2 LPARs (client & server), each with
- cores: 4 with SMT-2, i.e. 8 threads,
- memory: 16GB,
- distribution: SUSE Linux Enterprise Server 15 SP2 (SLES15-SP2),
- kernel version: 5.3.18-22-default.

Network adapters: Three different adapter types
- OSA-Express6S 10 GbE,
- OSA-Express7S 25 GbE,
- 25 GbE RoCE Express2.1.

The adapters are not shared between the LPARs, i.e. both LPARs have different adapters of each type attached.

All adapters are connected to a switch.



Network settings

Maximum transmission unit (MTU): Throughout this document measurements were run with either of the following two different MTU sizes

- 1500 bytes,
- 9000 bytes.

Subsequently, the MTU size used will be highlighted in each section.

Changing the MTU size: To change the MTU size, use

```
ip link set dev <interface name> mtu <mtu size>
```

.

Remark: To make use of an increased MTU size, **every** hop on the path to the target system needs to support the higher MTU size.

In order to identify the path MTU the following command can be issued,

```
tracepath -b <target IP address>
```

The last line summarizes the information, including the path MTU (pmtu).

Network settings

Device specific changes: In the following, only settings are presented that *differ* from default.
Remark: `<ifn> = <interface name>`

OSA-Express: RX- and TX-checksum offload enabled via,

```
ethtool -K <ifn> rx on tx on.
```

TCP segmentation offload (TSO) enabled via,

```
ethtool -K <ifn> tso on.
```

Check offload settings with,

```
ethtool -k <ifn>.
```

Buffer count increased from 64 to 128,

```
# Set device offline
echo 0 > /sys/class/net/<ifn>/device/online
# Increase buffer count to 128
echo 128 >
/sys/class/net/<ifn>/device/buffer_count
# Set device online again
echo 1 > /sys/class/net/<ifn>/device/online.
```

RoCE Express2.1: Multi-Packet Rx Queue (MPRQ aka Striding RQ) enabled via,

```
ethtool --set-priv-flags <ifn>
rx_striding_rq on.
```

Check flags with,

```
ethtool --show-priv-flags <ifn>.
```

Workload

Benchmark: uperf (adapted)

- uperf.org
- github.com/uperf/uperf

Protocol: Transmission Control Protocol (TCPv4)
Remark: Throughout the course of this presentation, the term TCP is used interchangeably with TCPv4.

Workload profiles: During this presentation the focus is on three different traffic patterns, namely

- rr1c-200x1000,
- rr1c-200x30k,
- str-writex30k.

Runtime: The runtime for a single performance measurement – single workload pattern, number of connections stays the same – is 300 seconds.

Connections: Each workload pattern is run with 1, 10, 50 and 250 active parallel connections.

rr1c-200x1000: A request-response workload simulating a highly transactional traffic pattern with medium data sizes,

- client (LPAR 1) is sending a 200 byte request,
- server (LPAR 2) is sending a 1000 byte response.

rr1c-200x30k: A request-response workload simulating a transactional traffic pattern with large data sizes,

- client (LPAR 1) is sending a 200 byte request,
- server (LPAR 2) is sending a 30720 byte response.

str-writex30k: A streaming workload,

- client (LPAR 1) is permanently streaming (writing in 30720 byte chunks),
- server (LPAR 2) is receiving (and acknowledging).

For each workload: The patterns are repeated for each connection until the runtime limit is reached.

Workload (continued)

Transaction duration or throughput?
During subsequent performance comparisons the focus will either be on transaction duration or on throughput – besides processor consumption [see also slide 14].

rr1c-200x1000 represents a latency-critical workload with its small sized request (200 B) and a small sized response (1000 B). In this case the focus is on how fast data can be transferred instead of how much data – although this directly correlates. When considering transaction duration, a lower value means faster transmission and is thus favored. For request-response workloads, a transaction is started by sending a request and is completed by receiving the reply to the request.

rr1c-200x30k and ***str-writex30k*** represent workloads with the intention of transferring significant amounts of data in a decent time frame. Thus, for these two profiles the focus is on throughput (Gbit/s). In this case higher throughput is beneficial.

A little bit of math

uperf throughput calculation: uperf is accounting for (TCP) payload only. This means when calculating throughput, additional protocol data is omitted. For the scenarios presented below, there is the following additional protocol data,

- 14 bytes Ethernet header
  + 4 bytes CRC checksum,
- 20 bytes Internet Protocol header ($\text{Header}_{\text{IP}}$),
- 20 bytes Transmission Control Protocol header [+ 12 bytes TCP options] ($\text{Header}_{\text{TCP}}$).

Including this information and assuming an MTU of 1500 B as well as further accounting for the Ethernet preamble (7 bytes), start of frame delimiter (SFD, 1 byte) and the interframe gap (IFG, 12 bytes) the following calculation can be conducted to get the maximum throughput in the scenario presented in this presentation.

Overhead (O):

$$\begin{aligned} O &= 14\,\text{B} + 4\,\text{B} + 20\,\text{B} + 20\,\text{B} + 12\,\text{B} + 7\,\text{B} + 1\,\text{B} + 12\,\text{B} \\ &= 90\,\text{B} \end{aligned}$$

TCP payload per frame (D):

$$\begin{aligned} D &= \text{MTU} - (\text{Header}_{\text{IP}} + \text{Header}_{\text{TCP}}) \\ &= 1500\,\text{B} - (20\,\text{B} + (20\,\text{B} + 12\,\text{B})) = 1448\,\text{B} \end{aligned}$$

Ratio TCP payload (R):

$$R = \frac{D}{D+O} = \frac{1448\,\text{B}}{1448\,\text{B} + 90\,\text{B}} = 0.941482444733$$

Max. throughput of TCP payload on 25 GbE line (D_{25GbE}):

$$\begin{aligned} D_{25GbE} &= 25\,\text{Gbit/s} \cdot R \\ &= 23.537\,061\,118\,\text{Gbit/s} \end{aligned}$$

A little bit of math (continued)

Applying previous calculation to 10 GbE lines and jumbo frames with an MTU size of 9000 B as well, gives subsequent (rounded) numbers.

Max. throughput of TCP payload on a 25 GbE line:

- MTU 1500: 23.537 Gbit/s
- MTU 9000: 24.751 Gbit/s

Max. throughput of TCP payload on a 10 GbE line:

- MTU 1500: 9.415 Gbit/s
- MTU 9000: 9.900 Gbit/s

Processor consumption

The processor consumption is measured as processor time (in µs) per transferred data (in Gbit).
With this the amount of data transferred with different scenarios (see slide 10) is considered.
Since there always is a server and a client system, that are sometimes having highly different processor times depending on the workload – especially with asymmetric scenarios –, the processor consumption is calculated on a per system basis.

For the sake of simplicity, in the following the processor consumption calculation of one system (e.g. server) is considered. However, this calculation can be applied to the other system (e.g. client) as well.

Being x_{s_1} the amount of data transferred in scenario 1 with a processing time of y_{s_1}.
Furthermore, being $x_{s_2} = x_{s_1} \cdot z_x$ ($\Rightarrow z_x = x_{s_2}/x_{s_1}$) the amount of data transferred in scenario 2 with a processing time of $y_{s_2} = y_{s_1} \cdot z_y$ ($\Rightarrow z_y = y_{s_2}/y_{s_1}$).

Processor consumption scenario 1 (c_{s_1}):

$$c_{s_1} = \frac{y_{s_1}}{x_{s_1}}$$

Processor consumption scenario 2 (c_{s_2}):

$$c_{s_2} = \frac{y_{s_2}}{x_{s_2}} = \frac{y_{s_1} \cdot z_y}{x_{s_1} \cdot z_x} = c_{s_1} \cdot \frac{z_y}{z_x}$$

Thus:

1. If $z_y > z_x$, then $c_{s_2} > c_{s_1}$.
2. If $z_y = z_x$, then $c_{s_2} = c_{s_1}$.
3. If $z_y < z_x$, then $c_{s_2} < c_{s_1}$.

Processor consumption (continued)

In order to illustrate the calculation from the previous slide, two examples are given.

Example 1: Assuming the processing time of scenario 2 is 10% higher than in scenario 1, i.e. $z_y = 1.1$, but the amount of data transferred is 10% higher as well, ($z_x = 1.1$). Then, we have:

$$c_{s_2} = c_{s_1} \cdot \frac{1.1}{1.1} = c_{s_1}$$

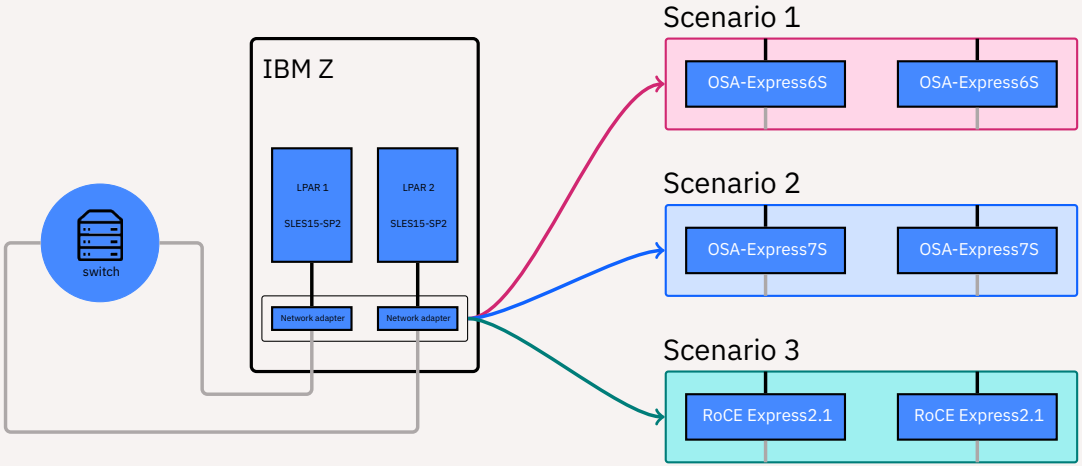
This means the processor consumption for scenario 2 is on the exact same level as for scenario 1.

Example 2: This time assuming the processing time of scenario 2 is 32% higher than in scenario 1, i.e. $z_y = 1.32$, but the amount of data transferred is still 10% higher, ($z_x = 1.1$). Then, we have:

$$c_{s_2} = c_{s_1} \cdot \frac{1.32}{1.1} = 1.2 \cdot c_{s_1} > c_{s_1}$$

This means the processor consumption for scenario 2 is higher (by 20%) than for scenario 1.

Network adapter performance using an MTU size of 1500 B



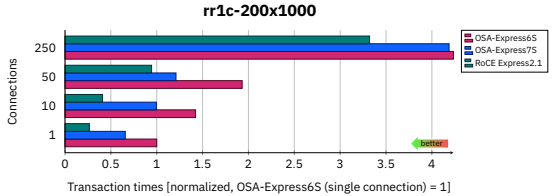
Workload: rr1c-200x1000
Highly transactional with medium data sizes.

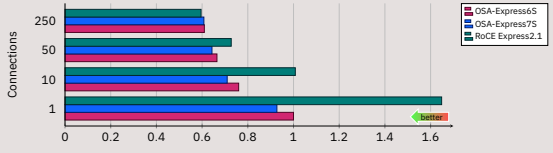
Note: Results normalized to OSA-Express6S single connection.

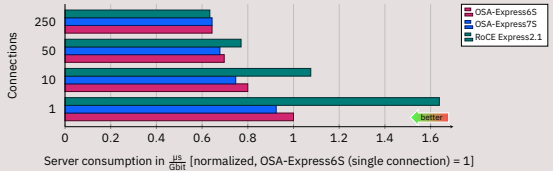
OSA-Express7S: The OSA-Express7S 25 GbE adapter outperforms the OSA-Express6S 10 GbE in speed, i.e. transaction time, as well as in processor consumption.

For 1, 10 and 50 parallel connections the OSA-Express7S is 30% to 37% faster and introduces 3% to 7% in processor consumption savings compared to OSA-Express6S.

Additional information on the 250 connections case – exploiting the benefits of RPS – is presented subsequently [see slide 41].







Workload: rr1c-200x1000
Highly transactional with medium data sizes.

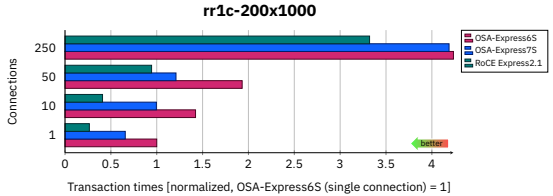
Note: Results normalized to OSA-Express6S single connection.

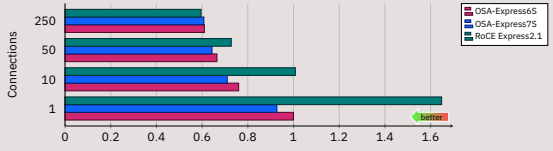
RoCE Express2.1: RoCE Express2.1 highly outperforms OSA-Express adapters in speed for this workload pattern.

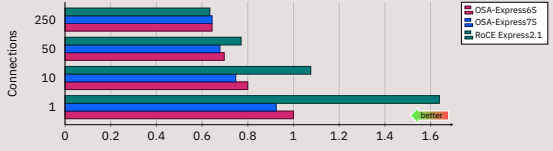
Transaction time for the single connection case as well as with 10 parallel connections is reduced to less than half the time compared to OSA-Express7S. Even for higher numbers of parallel connections RoCE Express2.1 is significantly faster with a 40% reduced transaction time when running with 50 parallel connections. With 250 parallel connections the improvement is more than 15%.

The improved transaction time comes with a certain cost. Processor consumption (µs/Gbit) is around 75% higher for the single connection case and around 40% higher for 10 parallel connections compared to OSA-Express7S. With 50 parallel connections the gap is closing – around 17% – and for 250 parallel connections RoCE Express2.1 is on the same consumption level as the OSA-Express.

Additional information on this workload – making use of RPS – is presented subsequently [see slide 44].







Workload: rr1c-200x30k
Transactional with large data size.

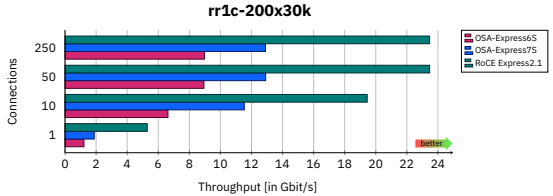
Note: Results for processor consumption normalized to OSA-Express6S single connection.

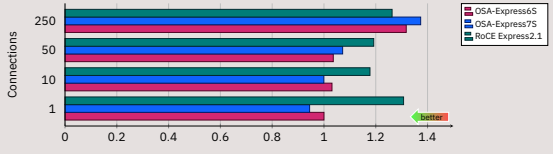
OSA-Express7S: Compared to OSA-Express6S, OSA-Express7S significantly increases throughput for this workload pattern.

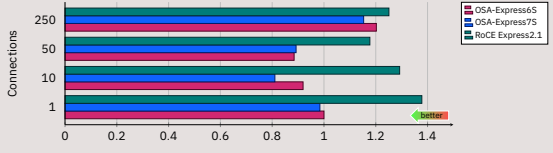
When running in a single connection scenario, the improvement in throughput is more than 50%. With 10 parallel connections the improvement is almost 75%. For 50 and 250 parallel connections both devices hit their respective limits, resulting in a 40% throughput increase for OSA-Express7S.

Be aware that the OSA-Express7S 25 GbE limit is around 13 Gbit/s (TCP payload) here, while the OSA-Express6S 10 GbE is capped by line speed for 50 and 250 parallel connections. The OSA-Express7S throughput limitation will be demonstrated in later charts to be largely a receive-side limit [see slides 31-35].

When it comes to processor consumption (µs/Gbit) both devices are on a similar level.







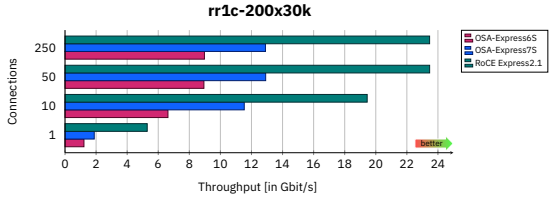
Workload: rr1c-200x30k
Transactional with large data size.

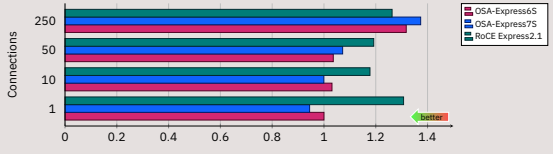
Note: Results for processor consumption normalized to OSA-Express6S single connection.

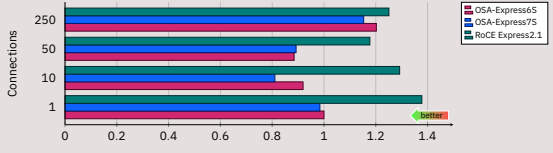
RoCE Express2.1: With regard to throughput the RoCE Express2.1 highly outperforms the OSA-Express.

For the single connection workload RoCE Express2.1 increases throughput by 2.8x compared to OSA-Express7S. Even for multi-connection workloads the improvement is 65% (10 parallel connections) up to 80% (50 and 250 parallel connections).
Furthermore, when running with 50 or 250 parallel connections the throughput is capped by line speed.

Again this improvement comes at a certain cost in processor consumption. For single and 10 parallel connections the ratio µs/Gbit is 40% to 60% higher compared to OSA-Express7S. However, for 250 parallel connections RoCE Express2.1 is roughly on the same level as OSA-Express.







Workload: str-writex30k
Streaming workload.

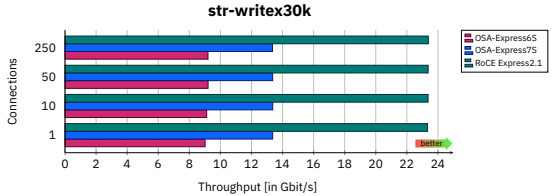
Note: Results for processor consumption normalized to OSA-Express6S single connection.

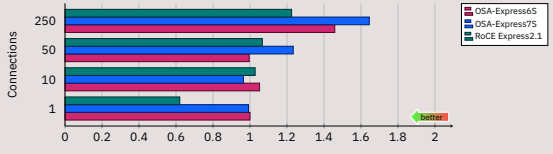
OSA-Express7S: For all tested scenarios, i.e. 1, 10, 50, and 250 parallel connections, the OSA-Express7S 25 GbE hits a limit at roughly 13.4 Gbit/s (TCP payload, i.e. without accounting for protocol data) - shown to be a receive-side limit [slides 31-35]. This is an improvement of more than 45% compared to OSA-Express6S 10 GbE.

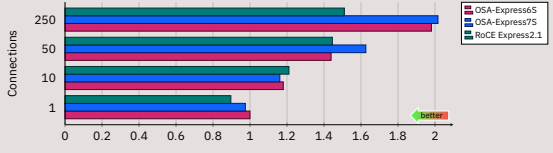
The processor consumption (µs/Gbit) remains on the same level as with OSA-Express6S.

RoCE Express2.1: The RoCE Express2.1 reaches line speed in all tested scenarios running this streaming pattern. Compared to OSA-Express7S this is an improvement of around 75%.

In contrast to the request-response workloads, this improvement does not come with an additional cost in processor consumption. For 250 parallel connections, the RoCE Express2.1 is even showing a reduced µs/Gbit ratio on receiver and sender side compared to OSA-Express7S.







Takeaways

1. The OSA-Express7S 25 GbE outperforms the OSA-Express6S 10 GbE in every tested scenario. When using an MTU size of 1500 B OSA-Express7S hits a throughput limit of around 13.4 Gbit/s. This limitation will be demonstrated to be largely a receive-side limitation [see slides 31-35].

2. The 25 GbE RoCE Express2.1 outperforms the OSA-Express7S 25 GbE in transaction time and throughput. Especially for request-response workloads with a low number of parallel connections, the increased speed comes with a certain cost, i.e. the processor consumption is noticeably higher compared to OSA-Express.

Network adapter performance using an MTU size of 9000 B

Recap [compare slide 8]:

Changing the MTU size to 9000 B,

```
ip link set dev <interface name> mtu 9000
```

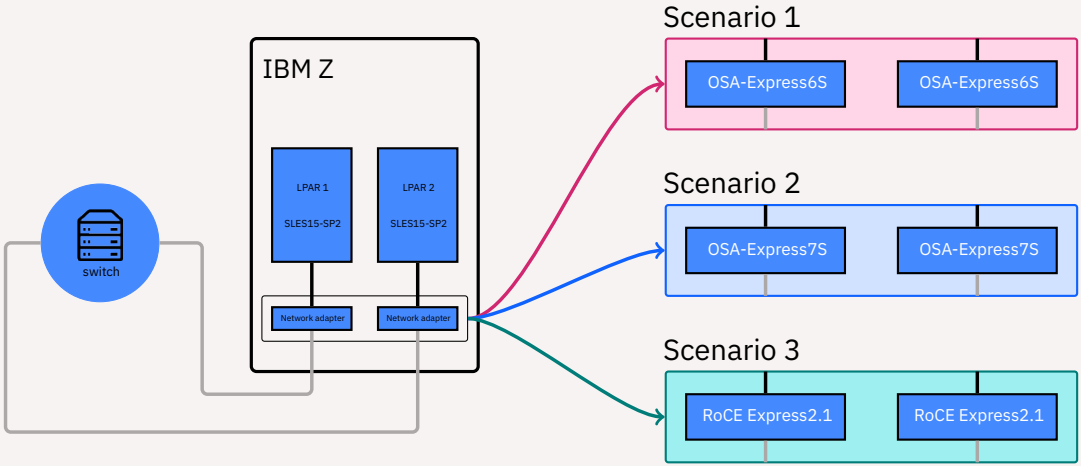
Remark: To make use of an increased MTU size, **every** hop on the path to the target system needs to support the higher MTU size.

In order to identify the path MTU the following command can be issued,

```
tracepath -b <target IP address>
```

The last line summarizes the information, including the path MTU (pmtu).

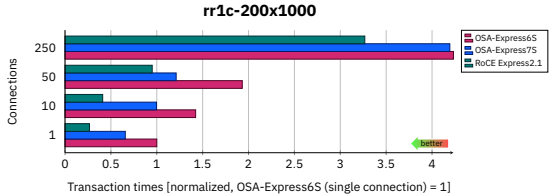
Network adapter performance using an MTU size of 9000 B

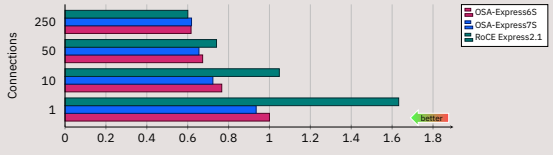


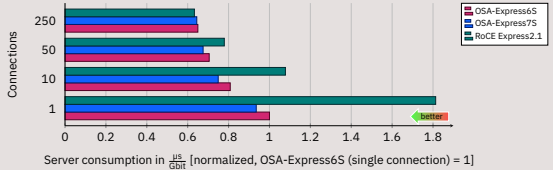
Workload: rr1c-200x1000
Highly transactional with medium data sizes.

Note: Results normalized to OSA-Express6S single connection.

OSA-Express7S & RoCE Express2.1: Increasing the MTU size does not introduce (major) changes for this workload pattern, since request (200 B) and response (1000 B) already fit into a single packet when using an MTU size of 1500 B. Thus, it is referred to the results from before [slides 17-18].





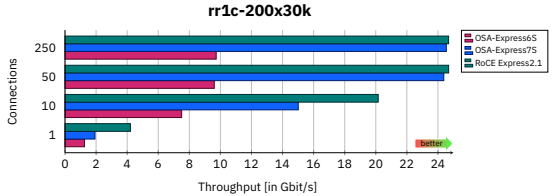


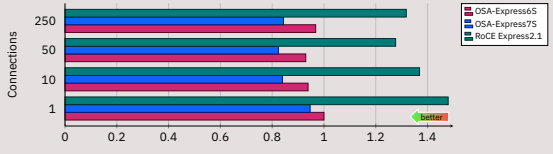
Workload: rr1c-200x30k
Transactional with large data size.

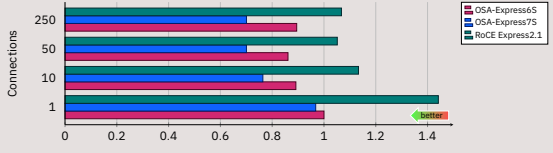
Note: Results for processor consumption normalized to OSA-Express6S single connection.

OSA-Express7S: The OSA-Express7S highly benefits from the increased MTU size.
When compared to OSA-Express6S, the throughput increases by more than 50% for the single connection scenario. Running with 10 parallel connections, the throughput is doubled.
For 50 and 250 parallel connections line speed (25 Gbit/s) is reached, resulting in an improvement of 2.5x over OSA-Express6S, which is capped at its line speed of 10 Gbit/s.

Although throughput is increased significantly, running an OSA-Express7S achieves a 5% (single connection) up to 13% (multi-connection) reduced µs/Gbit ratio compared to OSA-Express6S.





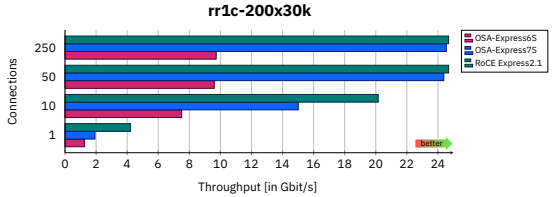


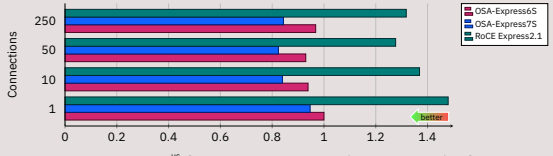
Workload: rr1c-200x30k
Transactional with large data size.

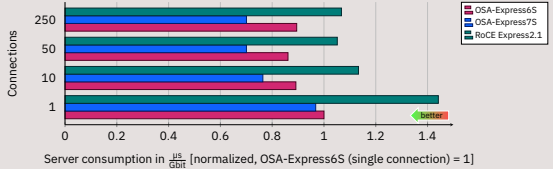
Note: Results for processor consumption normalized to OSA-Express6S single connection.

RoCE Express2.1: Again, the RoCE Express2.1 achieves the best throughput numbers.
In the single connection scenario RoCE Express2.1 reaches twice the throughput of the OSA-Express7S. For 10 parallel connections the improvement is still 34%. Since RoCE-Express2.1 and OSA-Express7S both hit line speed for 50 and 250 parallel connections, there is no difference in throughput for these cases anymore.

As seen before for request-response workloads the improved response time of the RoCE Express2.1 comes at a certain cost. This results in a 50% to 60% higher processor consumption for all of the scenarios – including the 50 and 250 parallel connection cases.





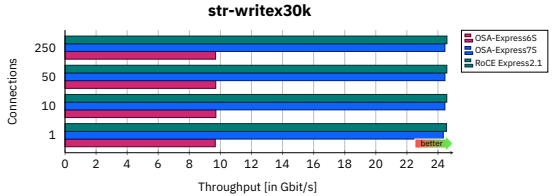


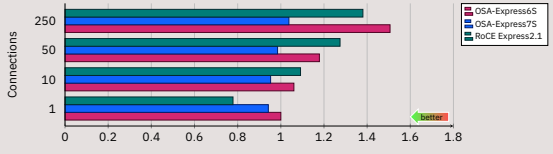
Workload: str-writex30k
Streaming workload.

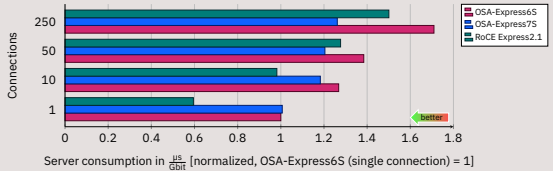
Note: Results for processor consumption normalized to OSA-Express6S single connection.

OSA-Express7S: In all scenarios the OSA-Express7S 25 GbE reaches line speed. Since this holds true for the OSA-Express6S 10 GbE as well, the throughput improvement using an OSA-Express7S 25 GbE is 2.5x.

In addition to the improvement in throughput, the processor consumption is reduced compared to OSA-Express6S. Especially for multi-connection scenarios the OSA-Express7S is able to achieve a noticeably lower µs/Gbit ratio. The savings are 12% (client) and 16% (server) for 50 parallel connections and 26% (client) and 31% (server) for 250 parallel connections.





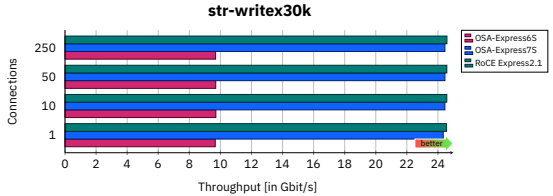


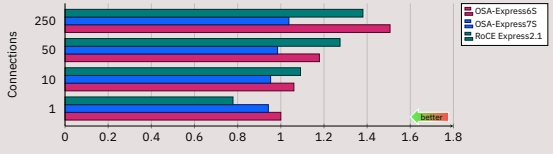
Workload: str-writex30k
Streaming workload.

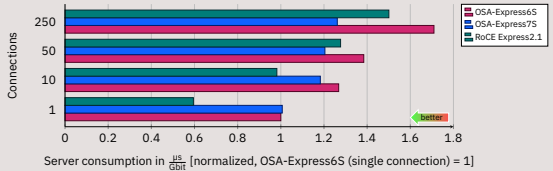
Note: Results for processor consumption normalized to OSA-Express6S single connection.

RoCE Express2.1: Like the OSA-Express7S, the RoCE Express2.1 reaches line speed in all presented scenarios. Thus, both adapters – OSA-Express7S and RoCE Express2.1 – are on the same throughput level here.

With regard to processor consumption, the RoCE Express2.1 is less demanding in the single connection scenario saving 17% (client) and 40% (server). This behavior is reversed for multi-connection scenarios. Running 250 parallel connections the µs/Gbit ratio is 33% (client) and 19% (server) higher.







Takeaways

1. Again, the OSA-Express7S 25 GbE outperforms the OSA-Express6S 10 GbE in every tested scenario. Using an MTU size of 9000 B the OSA-Express7S 25 GbE reaches line speed when running streaming workloads - independent of the number of parallel connections.

2. Again, the 25 GbE RoCE Express2.1 outperforms the OSA-Express7S 25 GbE in transaction time and throughput (as long as OSA-Express does not reach line speed). However, since OSA-Express7S 25 GbE is also reaching line speed for workloads with high amounts of data being transferred, there is no difference in throughput for these cases. As before, for request-response workloads with a lower number of parallel connections the increased speed comes with a certain cost, i.e. the processor consumption of the 25 GbE RoCE Express2.1 is noticeably higher compared to OSA-Express.

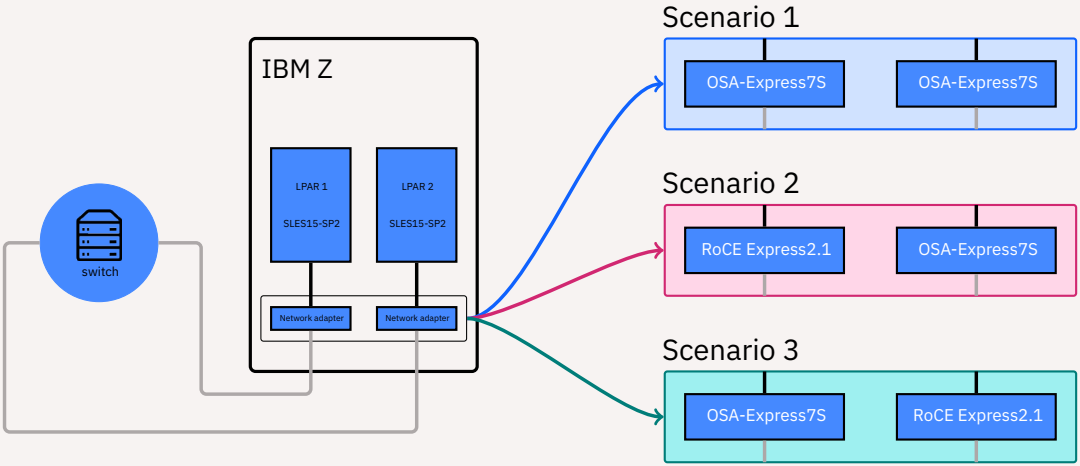
Additional view at OSA-Express7S streaming performance

As shown previously [slides 19 & 21], the OSA-Express7S network adapter used in conjunction with an MTU size of 1500 bytes was suffering reduced throughput when transferring large-sized data. In the former comparison an OSA-Express7S was used at sender and receiver side, i.e. at both LPARs, client and server.

In order to identify independent limits for transmitting and receiving streaming traffic, measurements between OSA-Express7S and RoCE Express2.1 – RoCE Express2.1 reaching line speed for transmitting and receiving – are presented.

The results from before – where the OSA-Express7S was used at client and server side – will serve as baseline. Subsequent new scenarios will have one of the OSA-Express7S from the baseline scenario replaced by a RoCE Express2.1 at either client or server side, see also next slide [slide 32].

Maximum transmission unit (MTU): The measurements were run with an MTU size of **1500 bytes**.

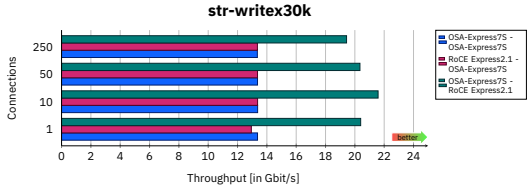


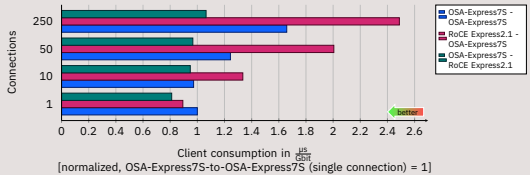
Workload: str-writex30k
Streaming workload.

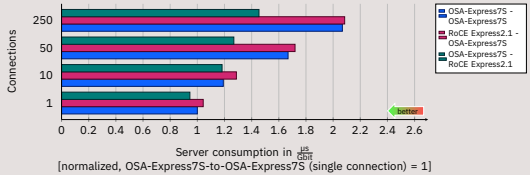
Note: Results for processor consumption normalized to OSA-Express7S-to-OSA-Express7S single connection.

RoCE Express2.1-to-OSA-Express7S:
Recap: RoCE Express2.1 client (here: sender), OSA-Express7S server (here: receiver)
This scenario hits the same throughput limit as OSA-Express7S-to-OSA-Express7S. Thus, the OSA-Express7S is capped by its reading capabilities at roughly 13.4 Gbit/s.

The processor consumption on server side is absolutely comparable to the OSA-Express7S-to-OSA-Express7S scenario, as expected. The client consumption is much higher due to the use of RoCE Express2.1, which is more cost-intensive. This effect is amplified by the reduced throughput – increasing the µs/Gbit ratio.





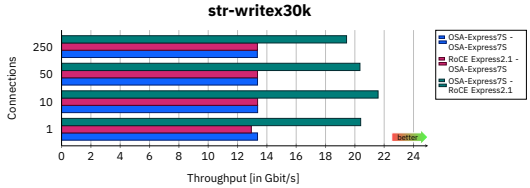


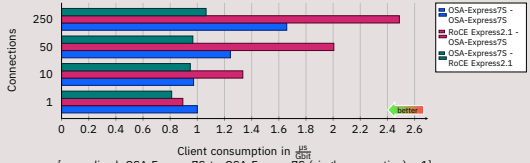
Workload: str-writex30k
Streaming workload.

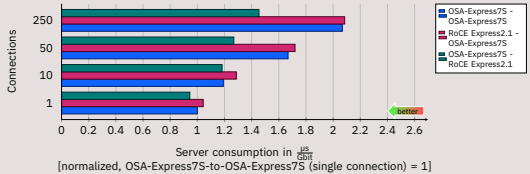
Note: Results for processor consumption normalized to OSA-Express7S-to-OSA-Express7S single connection.

OSA-Express7S-to-RoCE Express2.1:
Recap: OSA-Express7S client (here: sender), RoCE Express2.1 server (here: receiver)
With this scenario the former OSA-Express7S limit is exceeded. When transmitting data, the OSA-Express7S is able to reach up to 20.5 Gbit/s – instead of 13.4 Gbit/s for receiving data.

The throughput improvement also lowers the µs/Gbit ratio – compare client consumption (teal) against OSA-Express7S-to-OSA-Express7S (blue). Especially for higher numbers of parallel connections the ratio is decreased.







Takeaways

1. When using an MTU size of 1500 B the maximum inbound performance (receiving data) of the OSA-Express7S 25 GbE is 13.4 Gbit/s.

2. When using an MTU size of 1500 B the maximum outbound performance (transmitting data) of the OSA-Express7S 25 GbE is 20.5 Gbit/s.

A note on receive packet steering (RPS)

With receive packet steering (RPS) capability enabled within the Linux operating system, packet processing can be directed to predefined processors. This helps avoiding bottlenecks at the adapters receive queues.
RPS can be configured independent of the underlying network adapter.

Subsequently, the focus is on the request-response workload rr1c-200x1000 (see also slide 10). As the following slides will show, there can be clear benefits in configuring RPS for multi-connection scenarios with this workload pattern.

Nevertheless, the use of RPS needs to be considered carefully. Depending on the traffic pattern RPS may also introduce overhead. This especially holds true for single connection workloads or workloads with a low number of connections. Furthermore, for streaming workloads if line speed is reached without RPS already, using RPS may introduce additional overhead resulting in a noticeable increase in processor consumption.

On the other hand, if in a multi-connection scenario a high number of softirqs (SIs) is monitored on a single processor while others seem to have idle capacity, RPS might help by distributing packet processing across multiple processors and thus resolve a possible bottleneck.

All in all, before enabling RPS – in any way – the traffic pattern and possible bottlenecks should be identified and understood.

Configuring RPS

RPS has to be enabled on a per receive queue basis. If there are multiple receive queues, RPS can be configured on each of them independently.

In the following, it is presented how to configure RPS on a single queue. This procedure can be applied to each of the receive queues available.

For configuration the following file needs to be modified:

```
/sys/class/net/<ifn>/queues/<rx-queue>/rps_cpus
```

This file describes a comma-delimited processor bitmap. As long as RPS is not configured, the file contains all 0, like in the following example:

```
cat /sys/class/net/oe7s/queues/rx-0/rps_cpus
00000000,00000000,00000000,00000000,00000000,
00000000,00000000,00000000,00000000,00000000,
00000000,00000000
```

In order to direct packets to certain processors, these processors have to be selected corresponding to the bitmap. The rightmost entry represents processor 0, followed by an entry for processor 1, and so on.

Only entries representing available processors can be modified. This means, if 8 processors are available at the system, only the corresponding 8 bits can be modified. Trying to alter entries out of range results in keeping these at 0.
To redirect packets from the rx-queue to a certain processor, its entry has to be set to 1 in the configuration file. When writing to the "rps_cpus" file a hex translation of the binary value representing the desired configuration is used.

Enabling RPS – Examples

Example 1: Assuming there are 8 cores (0 to 7) available to the Linux system. Furthermore, assuming that the packets arriving on rx-0 (of device *oe7s*) shall be redirected to processors 0, 1, 3 and 6.

CPU	7	6	5	4	3	2	1	0
Bin	0	1	0	0	1	0	1	1
Hex	(	4	)	(	B	)		

The binary representation is "01001101" translating to "4B" (hex). Thus, activating RPS with this configuration,

```
echo 4b >
/sys/class/net/oe7s/queues/rx-0/rps_cpus.
```

Finally, checking the configuration file,

```
cat /sys/class/net/oe7s/queues/rx-0/rps_cpus
00000000,00000000,00000000,00000000,00000000,
00000000,00000000,00000000,00000000,00000000,
00000000,0000004b
```

Example 2: Assuming there are 8 cores (0 to 7) available to the Linux system. Furthermore, assuming that the packets arriving on rx-0 (of device *oe7s*) shall be redirected to all processors available, i.e. 0 to 7.

CPU	7	6	5	4	3	2	1	0
Bin	1	1	1	1	1	1	1	1
Hex	(	F	)	(	F	)		

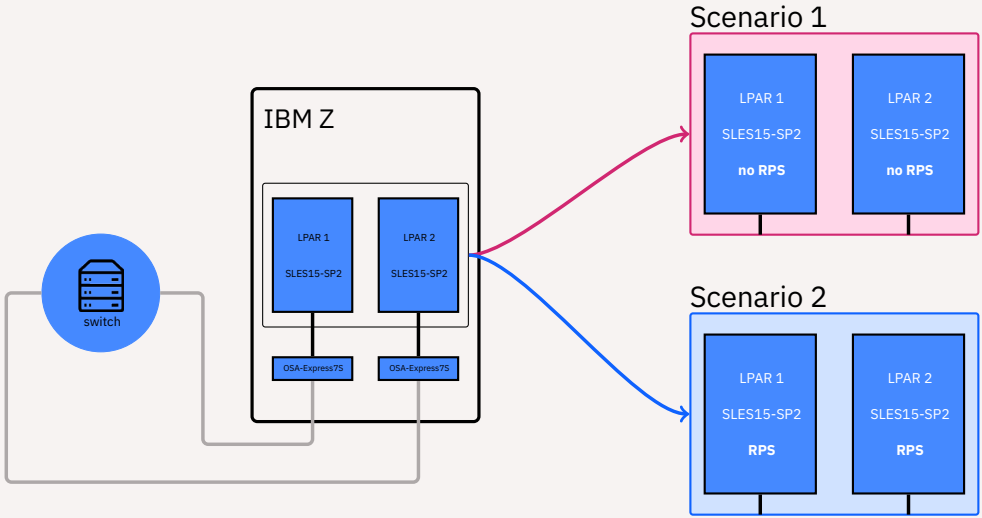
The binary representation is "11111111" translating to "FF" (hex). Thus, activating RPS with this configuration,

```
echo ff >
/sys/class/net/oe7s/queues/rx-0/rps_cpus.
```

Finally, checking the configuration file,

```
cat /sys/class/net/oe7s/queues/rx-0/rps_cpus
00000000,00000000,00000000,00000000,00000000,
00000000,00000000,00000000,00000000,00000000,
00000000,000000ff
```

RPS setup – OSA-Express7S



RPS setup – OSA-Express7S (continued)

Running Linux, OSA-Express7S offers a single receive queue rx-0. Thus, modification is performed on rx-0 only. In the following, the configuration for subsequently presented results for OSA-Express7S is presented, i.e. the meaning of *no RPS* and *RPS* in this context is explained.

no RPS: The *rps_cpus* configuration file for receive queue rx-0 is set to all 0, i.e.

```
echo 0 >
/sys/class/net/<ifn>/queues/rx-0/rps_cpus.
```

This results in RPS not being used.

RPS: The *rps_cpus* configuration file for receive queue rx-0 is set to "ff", i.e.

```
echo ff >
/sys/class/net/<ifn>/queues/rx-0/rps_cpus.
```

This results in packets from rx-0 being directed to all eight available processing threads – 4 cores with SMT-2.

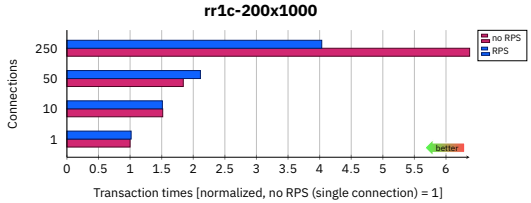
Workload: rr1c-200x1000
Highly transactional with medium data sizes.

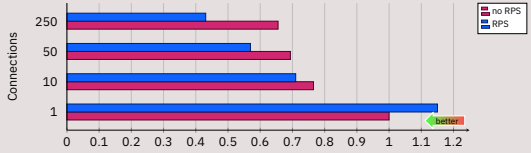
Note: Results for processor consumption normalized to "no RPS" single connection case.

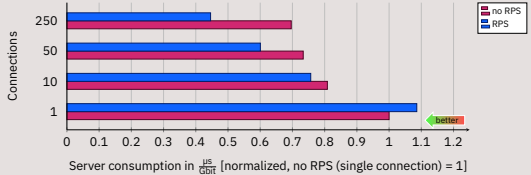
no RPS vs RPS (OSA-Express7S): As expected, there is no benefit in using RPS in the single connection case. Transaction time and consumption are increased. Thus, using RPS – in the way we do here – in a single connection scenario is not recommended.

For 10 parallel connections there is no decrease in transaction time, but a slight decrease in processor consumption. When running 50 parallel connections, transaction time is around 13% higher compared to not using RPS. On the other hand, processor consumption is reduced by around 18% at client and server. So there is no clear benefit for OSA-Express7S and these types of workload.

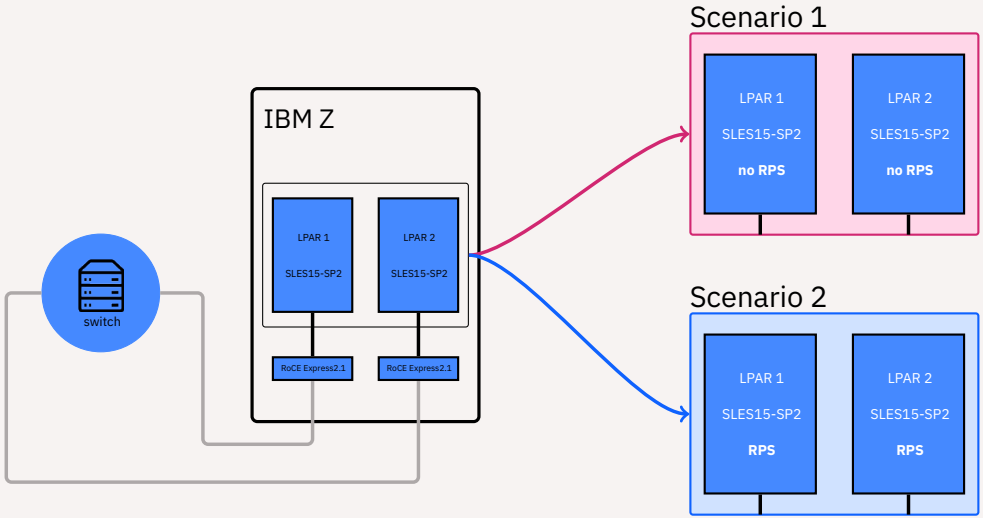
Things look totally different for the 250 connections case. Transaction time is lowered by around 37% and processor consumption is reduced by around 35% at client and server. Thus, using RPS in this case brings a huge benefit.







RPS setup – RoCE Express2.1



RPS setup – RoCE Express2.1 (continued)

RoCE Express2.1 offers multiple receive queues rx-x, $x \in \{y \mid 0 \leq y \leq 15\}$. As stated before, configuration of each queue could be different, but this is not the case in the following scenario, i.e. the same modification is performed on all rx-queues. Again, the meaning of *no RPS* and *RPS* for subsequent results is presented.

no RPS: The *rps_cpus* configuration file for receive queue rx-x, $x \in \{y \mid 0 \leq y \leq 15\}$, is set to all 0, i.e.

```
echo 0 >
/sys/class/net/<ifn>/queues/rx-x/rps_cpus.
```

This results in RPS not being used.

RPS: The *rps_cpus* configuration file for receive queue rx-x, $x \in \{y \mid 0 \leq y \leq 15\}$, is set to "ff", i.e.

```
echo ff >
/sys/class/net/<ifn>/queues/rx-x/rps_cpus.
```

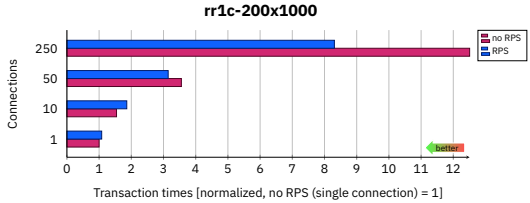
This results in packets from rx-x being directed to all eight available processing threads – 4 cores with SMT-2.

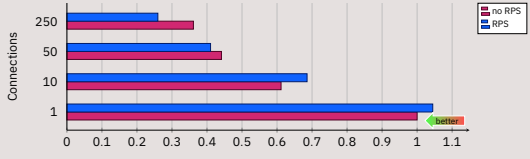
Workload: rr1c-200x1000
Highly transactional with medium data sizes.

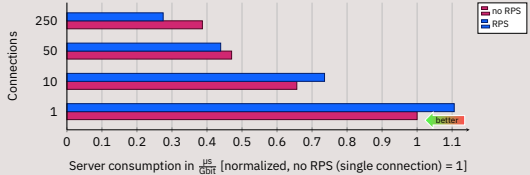
Note: Results normalized to "no RPS" single connection case.

no RPS vs RPS (RoCE Express2.1): In the single connection scenario as well as the scenario with 10 parallel connections the transaction times (7% and 17%) and processor consumption (client: 4% and 12%, server: 11% and 12%) are noticeably higher when using RPS. Thus, configuring RPS – the way as stated before – is not beneficial for these scenarios.

A noticeable decrease in transaction time and processor consumption, i.e. µs/Gbit, is observed with 50 parallel connections already. Transaction time is decreased by around 12% and processor consumption by around 7% at client and server. For 250 parallel connections the improvement by using RPS is huge. Transaction time is lowered by around 34% and processor consumption (µs/Gbit) is lowered by around 28% at client and server. Hence, there is a significant benefit in configuring RPS for this type of request-response workload – medium data sizes – when having a high number of active parallel connections.







Important remark

As stated on slide 36 already, configuring RPS can also introduce performance degradation.

On the one hand, it can induce overhead in single connection workloads resulting in overall performance regressions (throughput, transaction times, processor consumption).

On the other hand, it may also introduce overhead in processor consumption to (multi-connection) scenarios where RPS does not give a speed-up, like in scenarios where line-speed is reached without using RPS already.

Takeaways (RPS)

1. RPS can noticeably improve multi-connection scenarios especially for request-response workloads with small to medium data sizes. It may generally boost multi-connection scenarios as long as certain limits (e.g. line-speed, network adapter performance limit) are not reached.

2. Configuring RPS needs to be well considered. There might occur performance degradation when configuring RPS without previously analyzing bottlenecks. A good indication when to use RPS is a high softirq percentage on a single processor while others have idle capacity.

Takeaways (Overall)

1. OSA-Express7S 25 GbE outperforms OSA-Express6S 10 GbE in every aspect.
   When using an MTU size of 1500 B the OSA-Express7S 25 GbE is capped at 13.4 Gbit/s inbound and 20.5 Gbit/s outbound performance.

2. 25 GbE RoCE Express2.1 outperforms OSA-Express7S 25 GbE in speed, i.e. throughput and transaction times, and reaches line speed in throughput-centric workloads.
   This comes with a certain cost such that in most cases the 25 GbE RoCE Express2.1 increases processor consumption compared to OSA-Express7S 25 GbE.

3. Receive packet steering (RPS) can help to noticeably improve network performance of certain multi-connection workloads, especially workloads with a request-response character and small to medium data sizes.
   Nevertheless, the use and configuration of RPS needs to be considered carefully as it can also lower network speed, e.g. single connection cases, or increase processor consumption in cases where there is no speed-up induced by RPS.

Remarks:

1. For OSA-Express7S performance numbers on z/OS, see IBM z14 OSA-Express7S 25 GbE Performance Report version 2019-04-19.
2. While this presentation concentrates on performance aspects of different network devices under Linux on IBM Z, there might be other aspects to take into consideration as well when deciding for a particular device.