Shared Memory Communications over RDMA (SMC-R) Linux Overview

November 2014

Trademarks and copyrights

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

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.
- IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
- 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.
- 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.
- Windows Server and the Windows logo are trademarks of the Microsoft group of countries.
- 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.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
- 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.
- Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Authors / Contributions

SMC-R was produced by multiple IBM teams and labs.

The following individuals contributed to this solution and to the contents of this document:

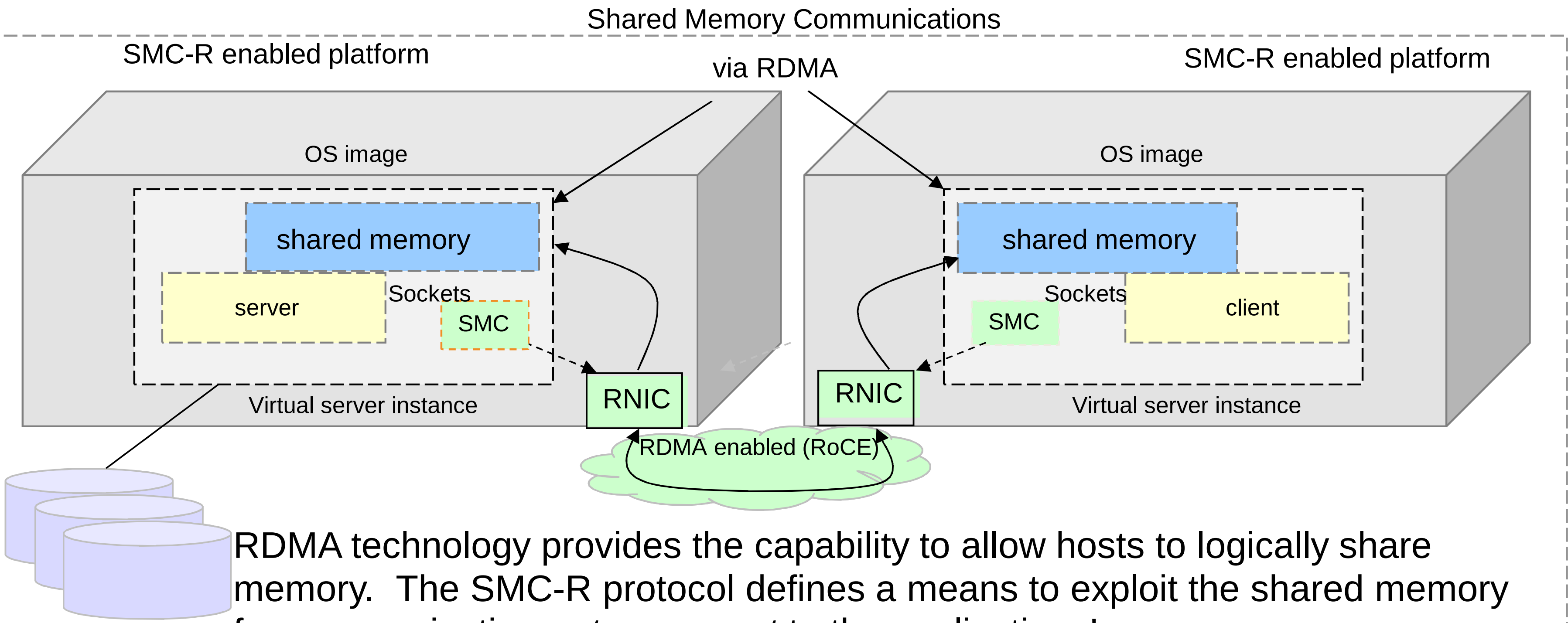
- Architecture and design:
  - Mike Fox (mjfox@us.ibm.com)
  - Gus Kassimis (kassimis@us.ibm.com)
  - Jerry Stevens (sjerry@us.ibm.com)
- Linux implementation
  - Ursula Braun (ursula.braun@de.ibm.com)
  - Joe Fowler (fowlerja@us.ibm.com)
- Performance testing
  - System z: Dan Patel (danpatel@us.ibm.com)
  - x86: Joe Fowler (fowlerja@us.ibm.com)

Agenda Topics

1. SMC-R Overview
2. Linux SMC-R Introduction:
   1. Linux Overview
   2. Installation
   3. Configuring / Enabling
   4. Application Exploitation
   5. Validation / Monitoring
   6. Tuning
3. Performance Summary
4. Backup:

Topic 1 SMC-R Overview

Shared Memory Communications over RDMA Concepts / Overview



RDMA technology provides the capability to allow hosts to logically share memory. The SMC-R protocol defines a means to exploit the shared memory for communications - *transparent* to the applications!

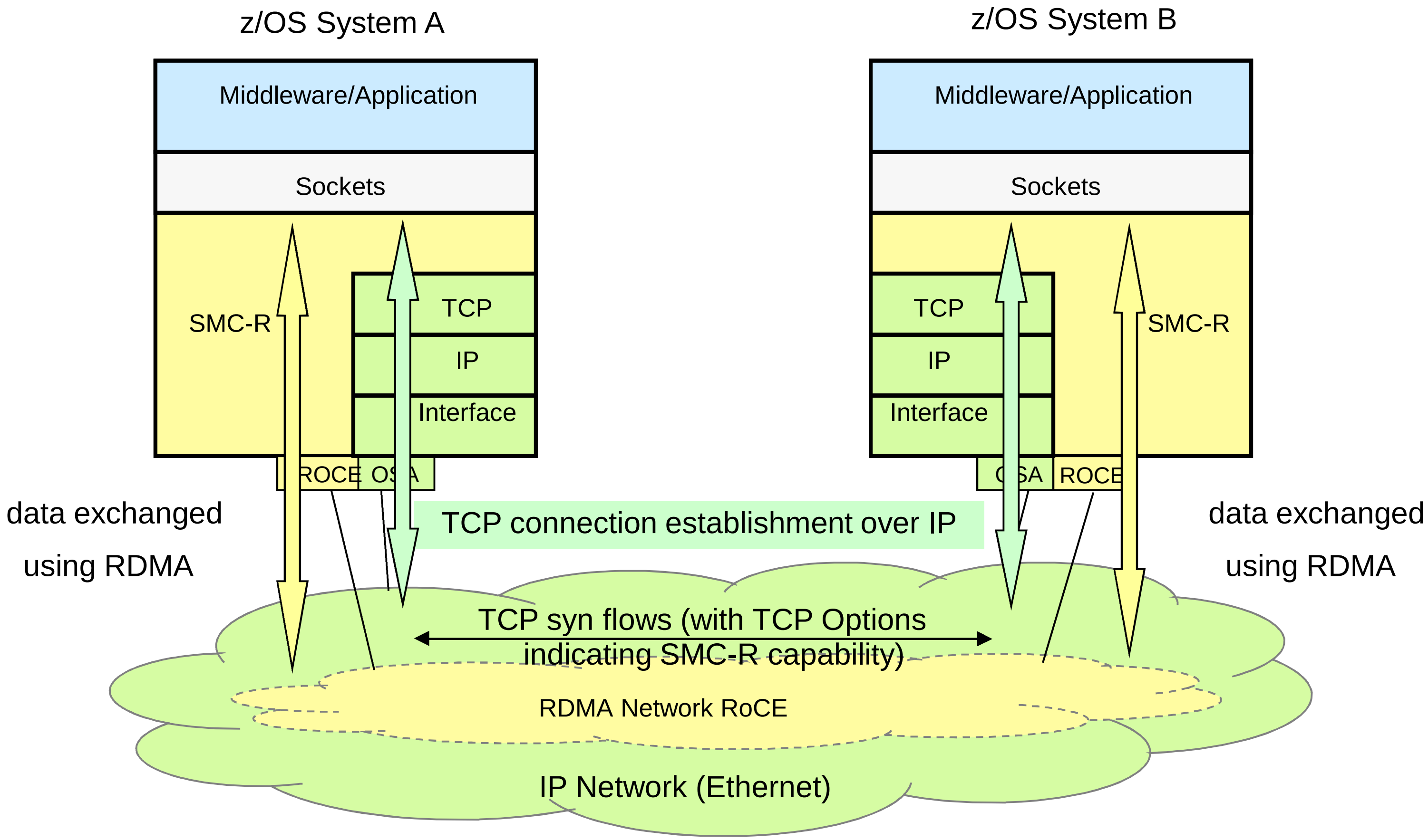
Clustered Systems

SMC-R is an *open* sockets over RDMA protocol that provides transparent exploitation of RDMA (for TCP based applications) while preserving key functions and qualities of service from the TCP/IP ecosystem that enterprise level servers/network depend on!

Draft IETF RFC for SMC-R:

http://tools.ietf.org/html/draft-fox-tcpm-shared-memory-rdma-05

Dynamic Transition from TCP to SMC-R



Dynamic (in-line) negotiation for SMC-R is initiated by presence of TCP Options

TCP connection transitions to SMC-R allowing application data to be exchanged using RDMA

SMC-R Overview

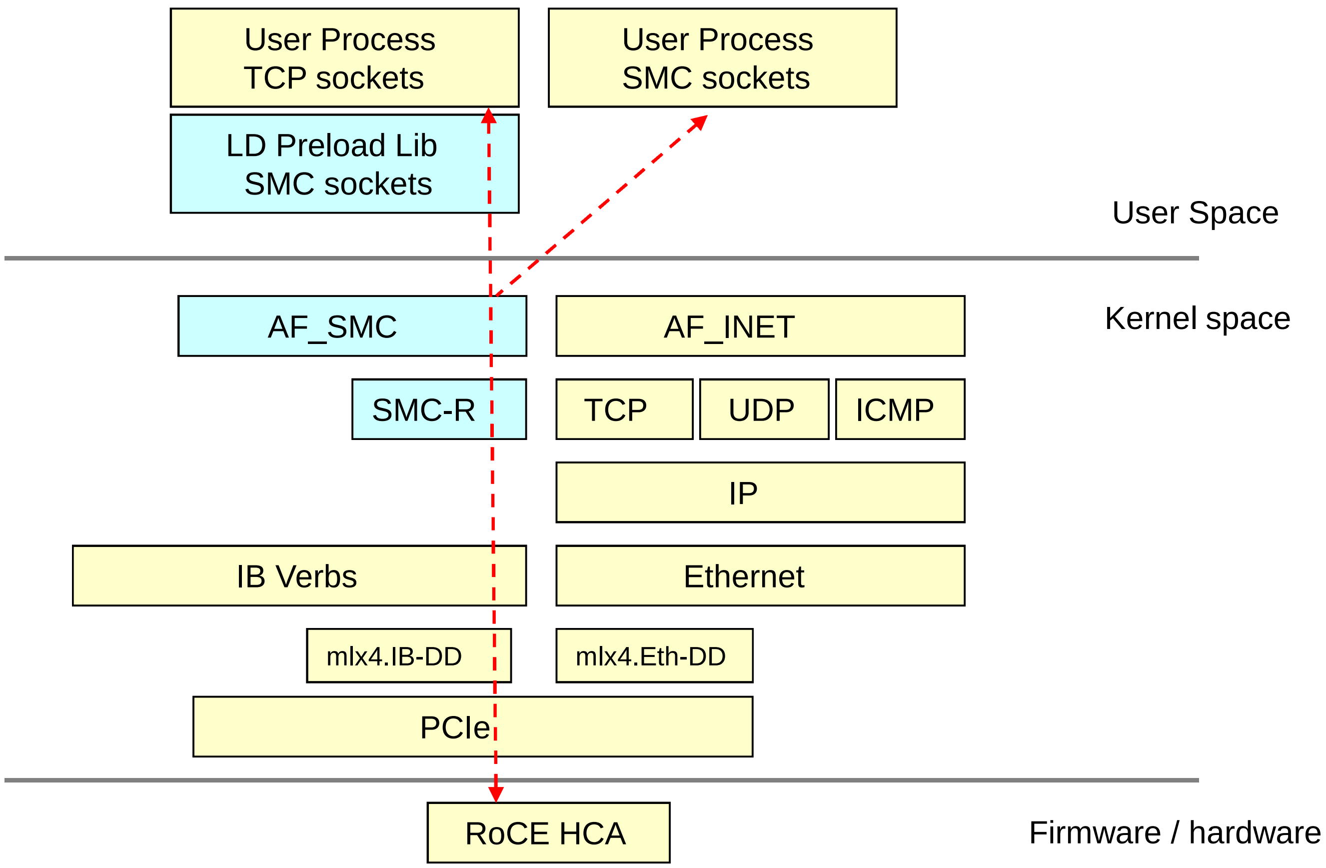
- Shared Memory Communications over RDMA (SMC-R) is a protocol that allows *TCP sockets* applications to transparently exploit RDMA (RoCE)
- SMC-R is a “hybrid” solution that:
  - Uses TCP connection (3-way handshake) to establish SMC-R connection
  - Each TCP end point exchanges TCP options that indicate whether it supports the SMC-R protocol
  - SMC-R “rendezvous” (RDMA attributes) information is then exchanged within the TCP data stream (similar to SSL handshake)
  - Socket application data is exchanged via RDMA (write operations)
  - TCP connection remains active (controls SMC-R connection)
  - This model preserves many critical existing operational and network management features of TCP/IP

SMC-R Key Attributes - Summary

- ✓ Optimized Network Performance (leveraging RDMA technology)
- ✓ Transparent to (TCP socket based) application software
- ✓ Leverages existing Ethernet technology (RoCE)
- ✓ Preserves existing network security model
- ✓ Resiliency (dynamic failover to redundant hardware)
- ✓ Transparent to Load Balancers
- ✓ Preserves existing IP topology and network administrative and operational model

Topic 2 Linux SMC-R Overview

Linux Overview



Linux SMC-R Overview Summary

- To support the SMC-R protocol on Linux, a new address family AF_SMC is created. It keeps the address format of AF_INET sockets and supports streaming socket types only using TCP.
- No special license requirements (GPL)
- 2 usage modes are possible:
  - AF_SMC native usage, defining the socket domain as AF_SMC instead of AF_INET
  - Invoke an AF_INET socket application with an SMC preload library converting AF_INET sockets to AF_SMC sockets. The SMC preload library will be part of an SMC tools package.
- For data traffic RNICs (RDMA Network Interface Cards) are used. For connection setup an auxiliary internal AF_INET TCP socket is maintained which uses a standard Ethernet network interface. This network interface is mapped to one or two available ROCE (RDMA over Converged Ethernet) Adapters.
- How a network interface maps to ROCE Adapters is configured within a table called "pnet table". Any available Ethernet interface can be combined with available RNICs, if they belong to the same Converged Ethernet fabric.

Installing SMC-R on Linux

- AF_SMC native usage: SMC-Code is part of the kernel. No extra install effort needed.
- Preload approach for AF_INET usage: Additional installation of the SMC preload library is required.
- The SMC preload library will be part of an SMC tools package to be provided by IBM.

Configuring and Enabling SMC-R on Linux (part 1)

- Extra loading of module smc is necessary[1]: modprobe smc

- Creation of a pnet table:

  Synopsis: add|del <PNET ID> eth|ib <device name [<port>]

  Sample:

  echo "add pnet0 eth eth4" > /proc/net/smc/pnet_conf

  echo "add pnet0 ib mlx4_0 2" > /proc/net/smc/pnet_conf

  echo "add pnet0 ib mlx4_1 2" > /proc/net/smc/pnet_conf

  Where:

  "pnet0"   is an identifier for a pnet group

  "eth"   defines specification of an Ethernet interface

  "ib"   defines specification of an RNIC

  "eth4"   specifies the Ethernet interface to be coupled

  "mlx4_0" specifies the first RNIC to be coupled

  "mlx4_1" specifies the second RNIC to be coupled

  "2"   specified the port of the RNIC to be coupled

Note 1. This a preliminary requirement.

Configuring and Enabling SMC-R on Linux (part 2)

- Display of a pnet table:

  cat /proc/net/smc/pnet_conf

- ROCEs are hybrids combining an RNIC with an Ethernet interface.
- The Ethernet interfaces of ROCE ports intended for SMC-R usage must be UP.

  ip link set eth1 up

  ip link set eth3 up

  where:

  "eth1" is the corresponding Ethernet interface for port 2 of RNIC "mlx4_0"

  "eth3" is the corresponding Ethernet interface for port 2 of RNIC "mlx4_1"

- The Ethernet interface used for connection setup requires a configured IP-address.

Application Exploitation of SMC-R

- Invocation:
  To port an existing AF_INET TCP socket application to execute native SMC-R, replace the socket creation call:

  tcp_socket = socket(AF_INET, SOCK_STREAM, 0);

  by

  tcp_socket = socket(AF_SMC, SOCK_STREAM, 0);

- To run an existing AF_INET TCP socket application through SMC-R, without changing the application, make use of an SMC preload library, that will be part of the SMC tools package. Both 32 and 64bit preload libraries will be provided.

Validation and Monitoring SMC-R

- In case of SMC negotiation failures or SMC link group problems during connection setup, an automatic fallback to the internal auxiliary AF_INET TCP socket is performed.

- A tool providing information on SMC sockets will be part of the SMC tools package showing (among other information about SMC-connections) whether connected sockets run through RDMA (SMC-R) or TCP/IP.

- Monitoring:
  The tool will show the connected and / or listening SMC sockets.

- Tracing:
  A Wireshark plugin will be made available for formatting SMC-R related RoCE LAN traffic.

Tuning Considerations for SMC-R on Linux

- mtu-size: the highest possible RDMA mtu size is 4096. For ROCE it is derived from the mtu size of the corresponding ROCE port Ethernet interface.
  Sample:

  ```
  ip link set eth1 mtu 4096
  ```

- An AF_SMC socket requires a contiguous send buffer. Its size can be defined through a SETSOCKOPT call of type SO_SNDBUF. Otherwise the sysctl definition in net.ipv4.tcp_wmem determines its size, if it is higher than an SMC-defined default. sndbuf - minimum smc socket send buffer size - default 65532. Sample:

  ```
  echo 131068 > /proc/net/smc/sndbuf
  ```

- An AF_SMC socket requires a contiguous receive buffer. Its size can be defined through a SETSOCKOPT call of type SO_RCVBUF. Otherwise the sysctl definition in net.ipv4.tcp_rmem determines its size, if it is higher than an SMC-defined default. rcvbuf - minimum smc socket receive buffer size - default 65532. Sample:

  ```
  echo 131068 > /proc/net/smc/rcvbuf
  ```

- ctrl_buffer_count - maximum number of transfer units in flight on an IB link. Sample:

  ```
  echo 128 > /proc/net/smc/ctrl_buffer_count
  ```

- max_conn_per_lgr - maximum number of connections sharing the same SMC link group. Sample:

  ```
  echo 32 > /proc/net/smc/max_conn_per_lgr
  ```

- In most cases (depending on your specific environment and performance analysis) the default values will be sufficient.

Topic 3 Linux SMC-R Performance Summary

Performance benchmarks for Linux with SMC-R on x86 and IBM System zEC12

Performance Disclaimer

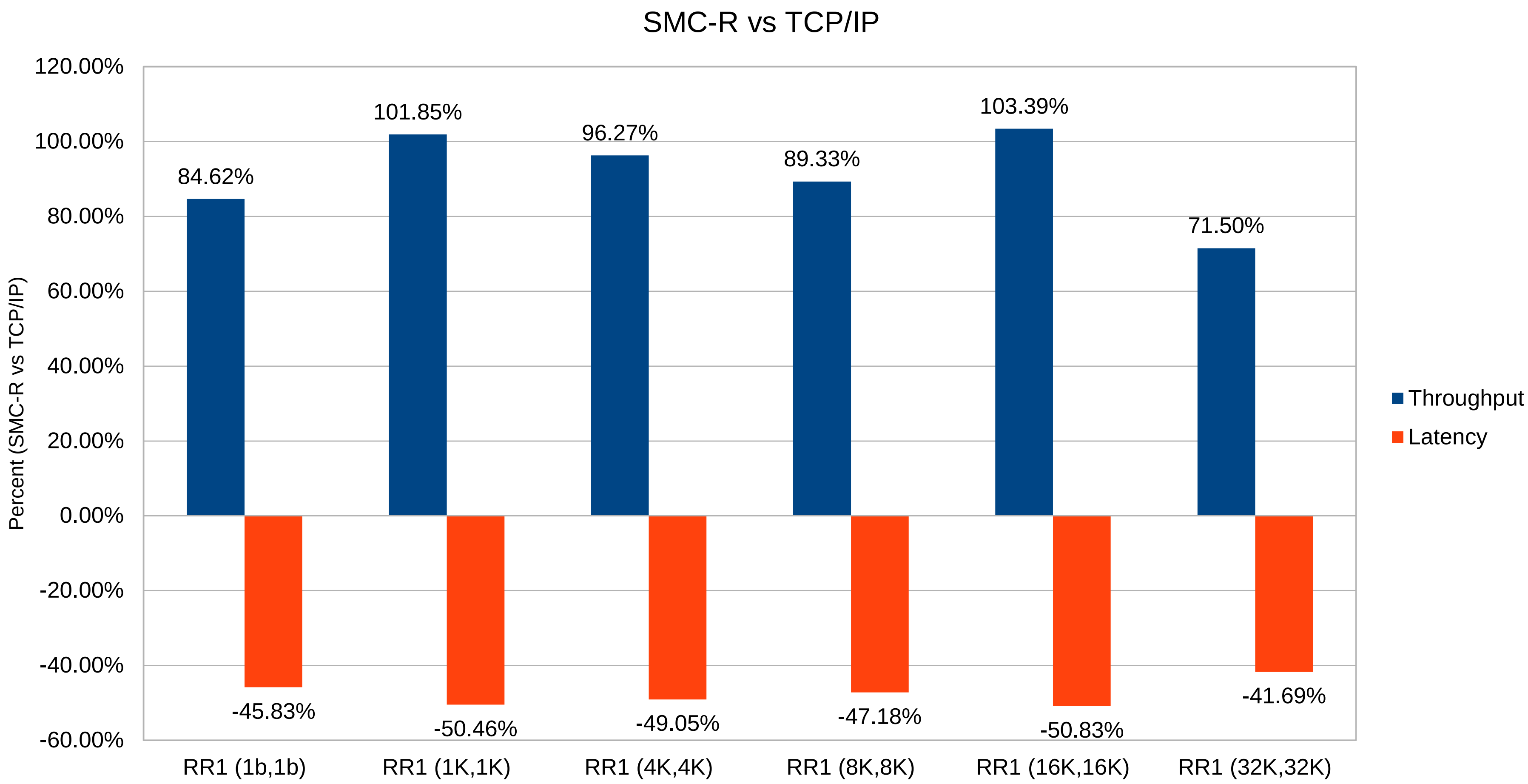
The performance measurements that are discussed in this document were collected by using a dedicated system environment. The results obtained using other configurations or operating system environments could vary significantly depending upon environments used. Therefore, there is no assurance given or guarantee made that an individual user can achieve performance or throughput improvements equivalent to the results stated here. Users of this document need to verify the applicable data for their specific environment.

20

SMC-R on Linux x86 benchmark environment

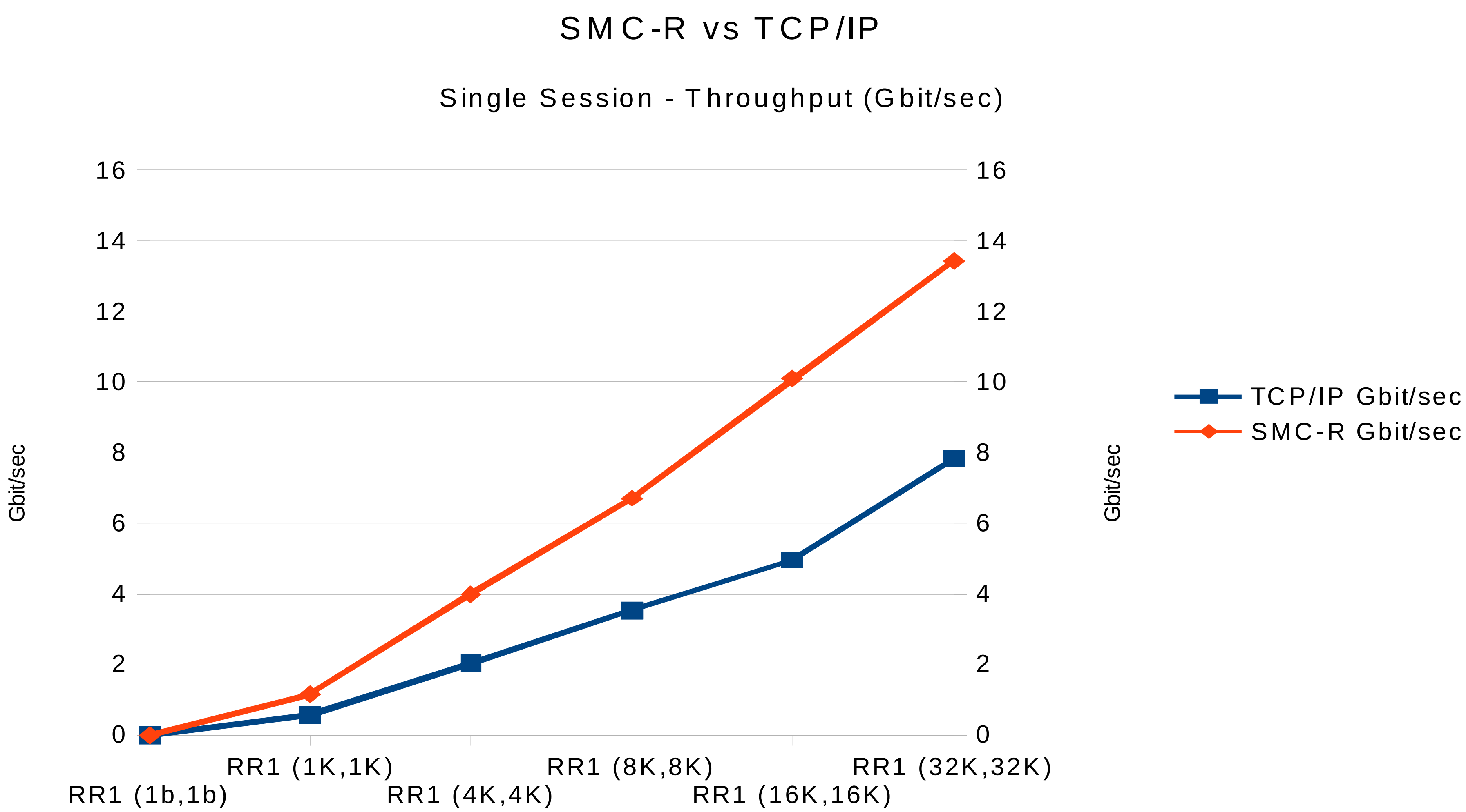
- Environment
  - Software: Linux Red Hat Enterprise Linux 6.4, stock 3.16 kernel with SMC patches
  - Client / server platform: IBM x3650 M4, 128GB RAM
  - Network Adapters: Mellanox ConnectX-3 FDR 40GbE (model CX354A), 2 interfaces per card
  - Network configuration: The two hosts were connected directly, adapter to adapter with no intervening switch (port 1 and port 2 of system A connected to port 1 and port 2 on System B)
- TCP/IP vs SMC-R tests performed over the same Mellanox cards
  - IP Configuration: All offloads enabled
    - rx-checksumming: on, tx-checksumming: on, tcp-segmentation-offload: on, generic-segmentation-offload: on, generic-receive-offload: on
  - MTU Size: IP (9000) RoCE (4K)
- Netperf used for all benchmarks
  - Request/Response (RR) patterns with various number of concurrent sessions and payload sizes
  - Streaming data patterns (single session)

SMC-R vs TCP/IP – Linux on x86 – Request/Response (RR1)

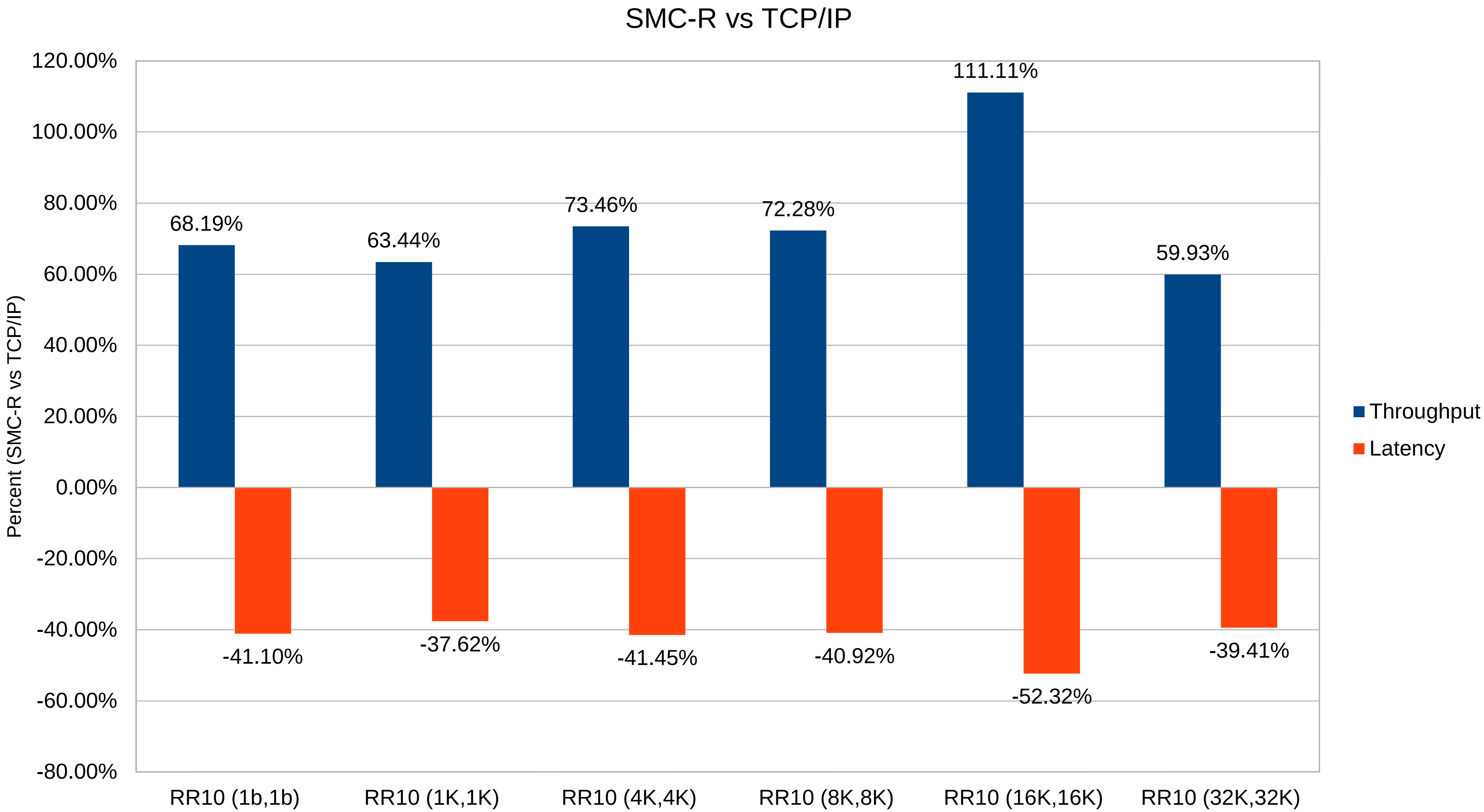


- RR1: Request/Response workload, persistent single TCP connection, various message sizes
  - Significant latency and throughput benefits across all message sizes

x86 Linux – RR1 – Throughput comparison

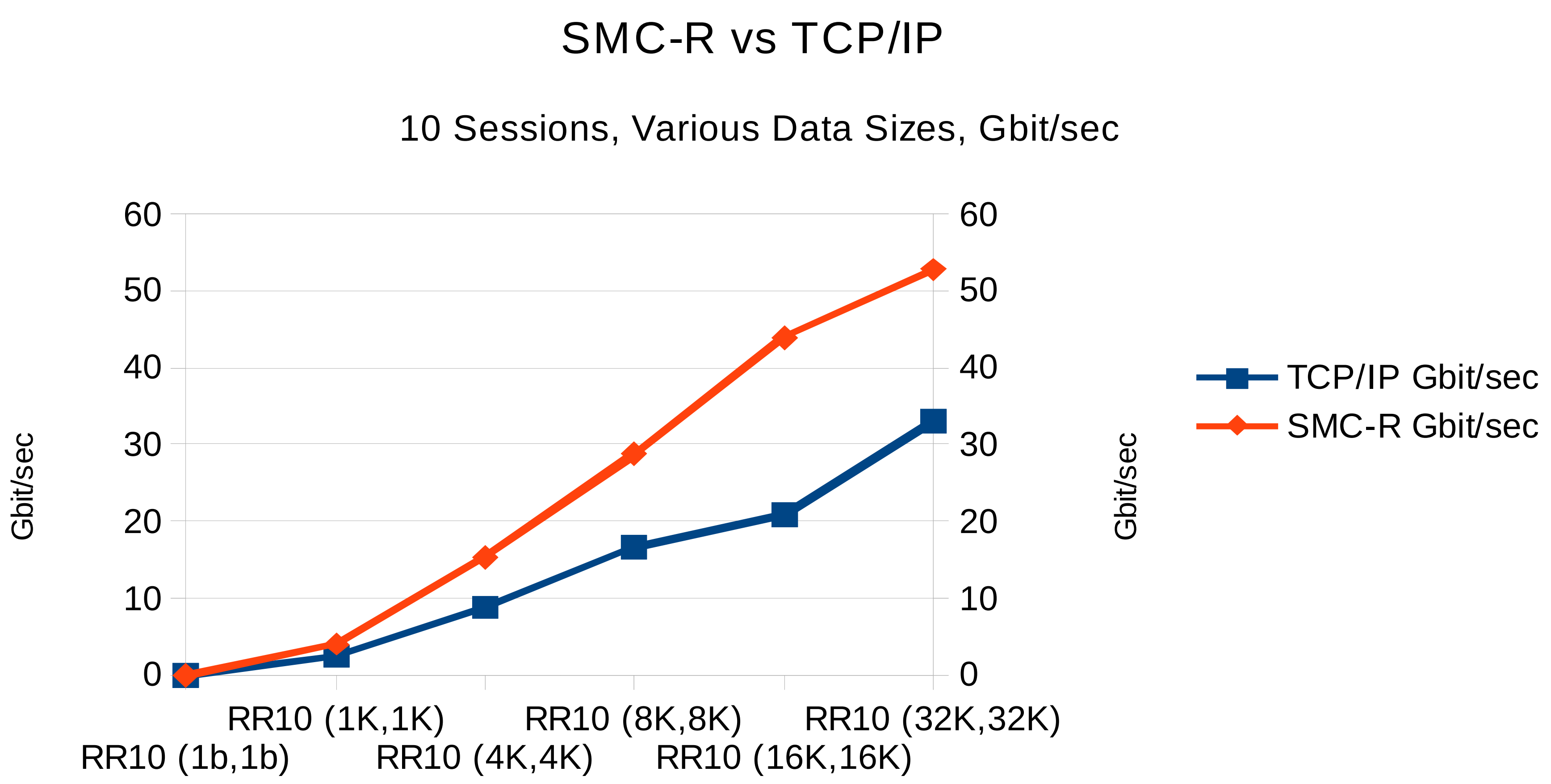


SMC-R vs TCP/IP – Linux on x86 – Request/Response (RR10)

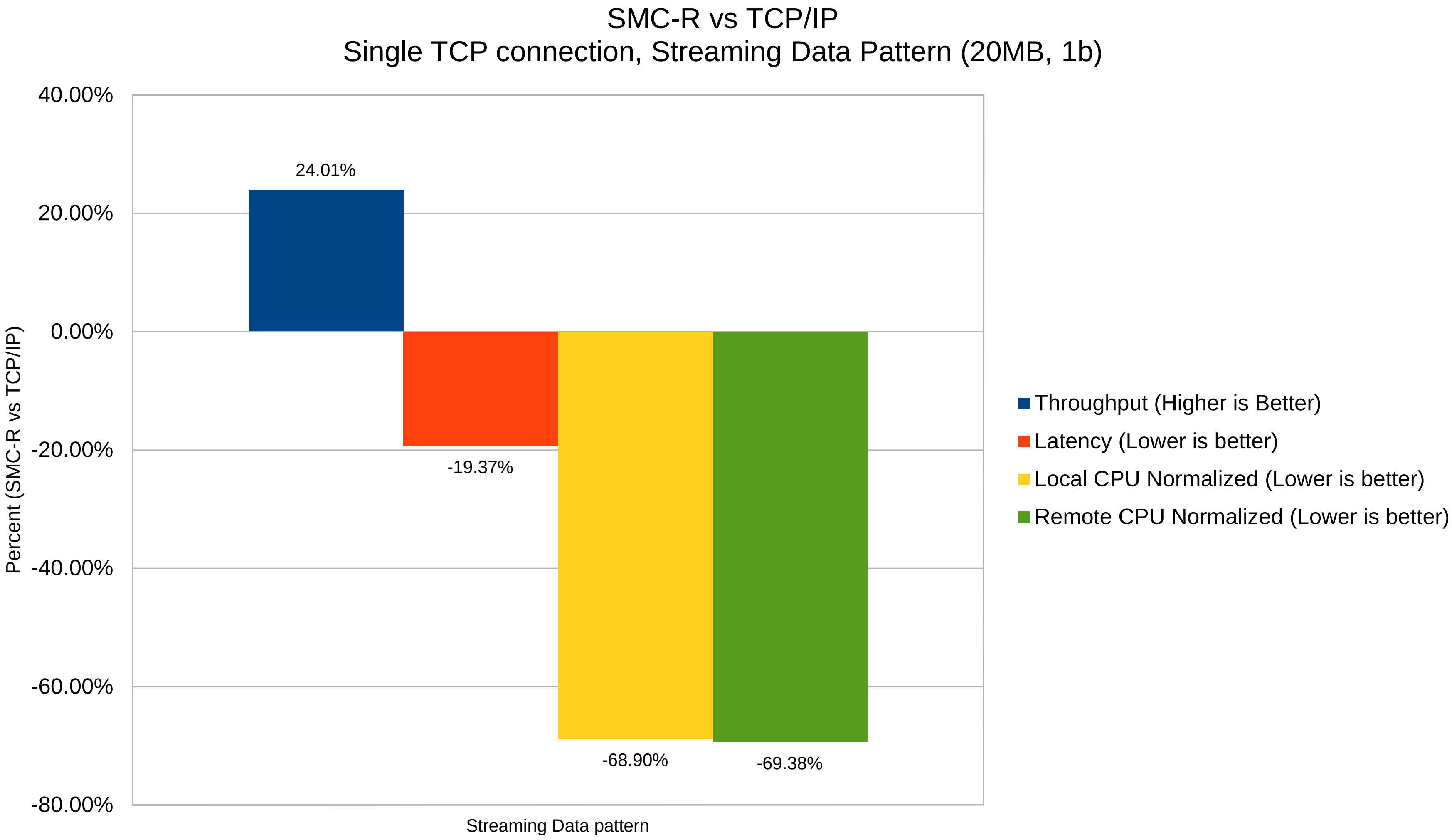


- RR10: Request/Response workload, 10 concurrent persistent TCP connections, various message sizes
  - Significant latency and throughput benefits across all message sizes

x86 Linux – RR10 – Throughput comparison

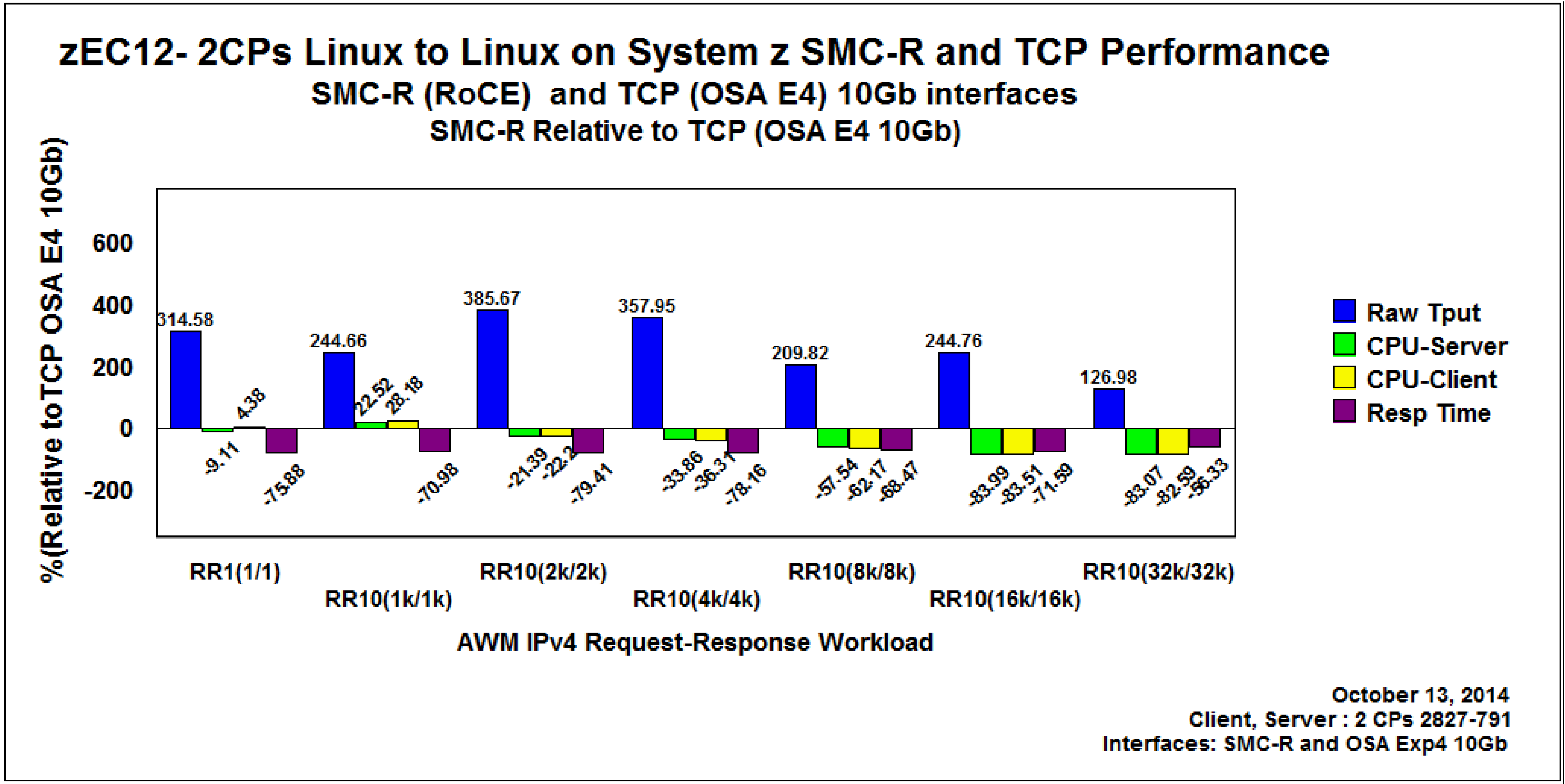


x86 Linux Streaming – Single Session



- RR1: Streaming Data Pattern, Single TCP connections, 20MB in one direction, 1 byte response
  - Substantial latency and throughput benefits
  - Significant CPU reduction benefits (Normalized CPU cost comparison – CPU per byte moved)

IBM System z Linux to Linux SMC-R Performance



- RR10: Request/Response workload, 10 concurrent persistent TCP connections, various message sizes
  - Significant latency and throughput benefits across all message sizes
  - Significant CPU savings for larger message sizes

Backup:

- Additional SMC-R Reference materials:

  http://www-01.ibm.com/software/network/commserver/SMCR/

THANK YOU