# RoCE BALBOA: Towards FPGA-enhanced RDMA

Maximilian J. Heer, Benjamin Ramhorst, Jonas Dann, Gustavo Alonso
{maximilian.heer},{benjamin.ramhorst},{jonas.dann},{alonso}@inf.ethz.ch
Systems Group, ETH Zurich
Zurich, Switzerland

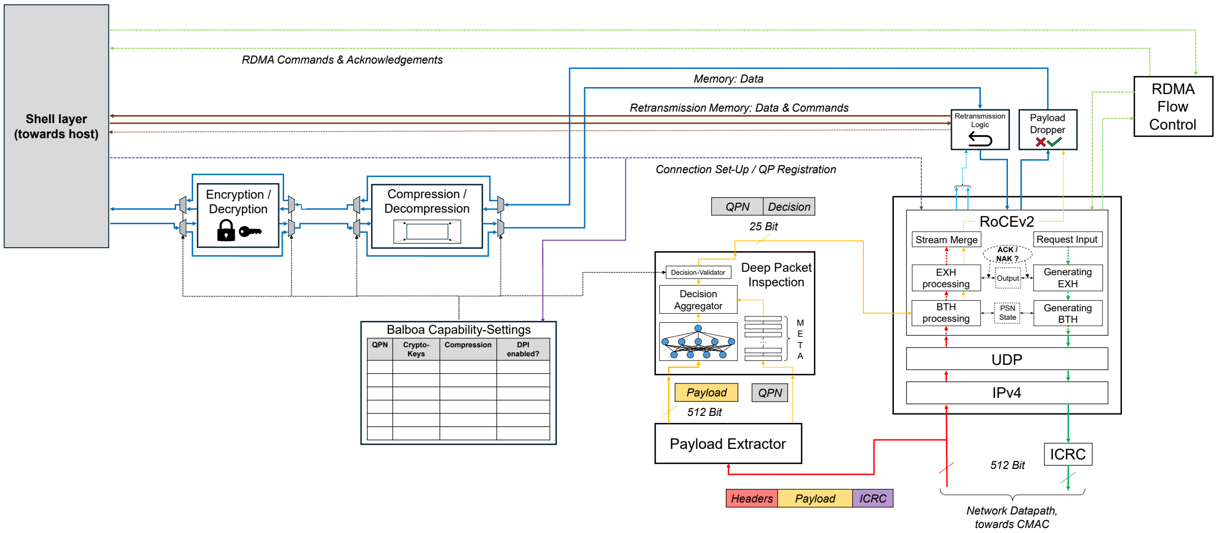

**Figure 1.** FPGA-based RoCE-v2 stack enhanced with Deep Packet Inspection, payload compression, and AES-cryptography.

***Keywords:*** RDMA, SmartNIC, FPGA, high-performance networking

## 1 Introduction

With increasingly data-intensive applications such as Machine Learning training and inference, the network is a bottleneck in data center computing [9]. To fulfill the demands of such applications, RDMA has been adopted from the HPC domain as the dominating high-performance network protocol, which makes up up to 70% of all network volume in a typical setup [1]. Based on the key principles of host bypassing and zero-copy, the network standard allows for low latency, high throughput, and low CPU utilization at the same time [3]. However, these very design features also pose problems when using RDMA in the public cloud and reveal the lack of three key features: RoCE v2 does not specify any form of traffic encryption [5], misses access control as OS-enforced rules are bypassed by design [8], and, finally, does not compress payloads for a reduced bandwidth consumption in already oversubscribed networks. While previous studies demonstrated RDMA-protocols with such enhancements implemented on the host OS [4] or in SmartNICs with embedded CPUs [6], no project has yet shown a truly performance-preserving, 100G-capable RoCE v2 implementation offering expendability for various protocol enhancements through accessible interfaces. In our work, we explore the use of FPGA-based SmartNICs in the context of capability-enhanced RDMA, by combining a self-developed, open-source and fully RoCE v2-compatible networking stack with exemplaric hardware modules for AES-encryption, snappy-compression, and machine learning-based Deep Packet Inspection (DPI) for access control that utilize stream computing to achieve 100G line rate speed without any performance overhead on the host CPU (Figure 1). Furthermore, our design allows to add any other user-defined application directly on the RDMA data streams, leveraging the reconfigurable fabric for offloading network processing from the host CPU to the FPGA-NIC at full line rate.

## 2 Design and performance

### 2.1 RDMA-Stack

Our open-source RDMA-stack [1] offers full RoCE v2-compatibility at 100 Gbps network speed over switched networks, with latency and throughput performance comparable to Mellanox ConnectX-5 commodity NICs. The design implements the key InfiniBand-verbs RDMA READ and WRITE, supports flow control and retransmission, and is highly adaptive for customizations of protocol functions as its main packet processing pipeline is implemented in AMD Vitis HLS. Integrated into the renewed and also open-source Coyote v2 shell [2], this RDMA-stack offers an easy-to-use software API

[1] GitHub: https://github.com/fpgasystems/fpga-network-stack
[2] GitHub: https://github.com/fpgasystems/Coyote

that resembles traditional RoCE-programming. The network stack was designed with offloaded extensions in mind, as its AXI-stream interfaces for data transport allow for seamless integration of any stream computation design.

### 2.2 AES-encryption

AES-encryption in ECB-mode is added to both in- and outgoing datapaths as open-source pipelines with 100 Gbps throughput to avoid any performance bottleneck. The initial key exchange is executed as part of the required QP-connection via a side-channel TCP-connection, the resulting keys are cached in BRAM-buffers, accessible by QP-number for incoming or outgoing RDMA-operations.

### 2.3 Snappy-compression

To achieve full 100G performance, seven open-source Snappy compression and decompression cores each are used in parallel, operating on separate chunks of the incoming AXI-stream [7]. To guarantee the correct mapping to compression windows during inflation, multiple subheaders for detection are inserted into the compressed payload before transmission.

### 2.4 ML-based DPI

Given that RDMA circumvents the OS and its access control rules in, for example, firewalls entirely and writes directly to application memory, it is especially susceptible for attacks where existing flows are hijacked and malicious executables are inserted in normal payloads. To counter this threat, our enhanced RDMA stack utilizes a ML-model parallel to the packet processing pipeline to differentiate between acceptable payloads and malicious executables, impacting the final decision whether to drop or accept an incoming packet. Generated with the ML-compiler hls4ml [2], this model combines a latency of 44ns with a detection accuracy of roughly 90% and a minimal hardware footprint of around 1% of the available resources, posing no performance overhead to RDMA-transaction at 100G line rate (Figure 2). This form of RDMA-DPI is a key example for the advantages of functional offloads on a reconfigurable network fabric:

- Early detection and blocking of malicious traffic is only possible on a SmartNIC before reaching the target host. This enhancement simply cannot be implemented in software on the host itself.
- The presented line rate DPI benefits largely from high-throughput inference on deeply pipelined hardware.

## 3 Future Work

Future work will include a full performance comparison of our FPGA-based protocol enhancements with the same features implemented on a conventional SmartNIC with an embedded CPU to highlight the differences between off- and on-datapath offloaded network computation. Another step will be the integration of partial reconfiguration to allow for runtime customization of the network stack.

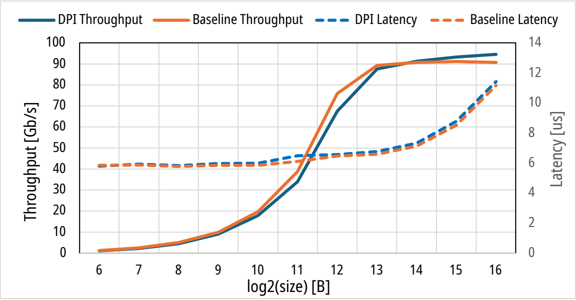

**Figure 2.** RDMA network performance (throughput and latency) with and without the added DPI-module.

## Acknowledgments

We would like to thank AMD for the donation of the Heterogeneous Accelerated Compute Cluster (HACC) which was used for the development and testing of this project.

## References

[1] Wei Bai et al. 2023. Empowering azure storage with RDMA. In *20th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2023, Boston, MA, April 17-19, 2023*. Mahesh Balakrishnan et al., (Eds.) USENIX Association, 49–67. https://www.usenix.org/conference/nsdi23/presentation/bai.

[2] Javier M. Duarte et al. 2018. Fast inference of deep neural networks in fpgas for particle physics. *CoRR*, abs/1804.06913. http://arxiv.org/abs/1804.06913 arXiv: 1804.06913.

[3] Chuanxiong Guo et al. 2016. RDMA over commodity ethernet at scale. In *Proceedings of the ACM SIGCOMM 2016 Conference, Florianopolis, Brazil, August 22-26, 2016*. Marinho P. Barcellos et al., (Eds.) ACM, 202–215. doi:10.1145/2934872.2934908.

[4] Maksym Planeta et al. 2023. Cord: converged RDMA dataplane for high-performance clouds. *CoRR*, abs/2309.00898. arXiv: 2309.00898. doi:10.48550/ARXIV.2309.00898.

[5] Benjamin Rothenberger et al. 2021. Redmark: bypassing RDMA security mechanisms. In *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*. Michael D. Bailey et al., (Eds.) USENIX Association, 4277–4292. https://www.usenix.org/conference/usenixsecurity21/presentation/rothenberger.

[6] Konstantin Taranov et al. 2020. Srdma - efficient nic-based authentication and encryption for remote direct memory access. In *Proceedings of the 2020 USENIX Annual Technical Conference, USENIX ATC 2020, July 15-17, 2020*. Ada Gavrilovska et al., (Eds.) USENIX Association, 691–704. https://www.usenix.org/conference/atc20/presentation/taranov.

[7] Xilinx. 2020. Snappy compression library. Accessed: 2025-02-04. (2020). https://xilinx.github.io/Vitis_Libraries/data_compression/2020.1/source/L2/snappy.html.

[8] Jiarong Xing et al. 2022. Bedrock: programmable network support for secure RDMA systems. In *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*. Kevin R. B. Butler et al., (Eds.) USENIX Association, 2585–2600. https://www.usenix.org/conference/usenixsecurity22/presentation/xing.

[9] Zhen Zhang et al. 2020. Is network the bottleneck of distributed training? In *Proceedings of the 2020 Workshop on Network Meets AI & ML, NetAI@SIGCOMM, Virtual Event, USA, August 14, 2020*. Behnaz Arzani et al., (Eds.) ACM, 8–13. doi:10.1145/3405671.3405810.