Designing Transport-Level Encryption for Datacenter Networks

Tianyi Gao, Xinshu Ma, Suhas Narreddy, Eugenio Luo, Steven W. D. Chien, Michio Honda
University of Edinburgh

ABSTRACT

Datacenter transport protocols, pioneered by DCTCP [1], have evolved over the last decade to achieve high throughput for bulk data transfer while maintaining low latency for small messages. The latest ones, including NDP [2] and Homa [5], are not extensions to TCP—they are message-based and often receiver-driven, where the receiver schedules the senders for packet transmissions to enable fine-grained network utilization. While most of these protocols have been implemented in user space or simulators, the availability of the Linux kernel implementation of Homa since 2021 [7], along with its demonstrated use in industry [4], makes widespread adoption of alternative transport protocols within reach.

However, what if the applications want data encryption to isolate themselves from other tenants and protect themselves from network infrastructure? Many cloud operators host multiple tenants. Furthermore, even hyperscalers do not build every datacenter component on their own; they source many, such as switches, cables, and interface modules, from external vendors, which could be later found vulnerable or malicious. It is thus common for datacenter applications or tenants to seek encryption for their network data, just as they would over the Internet.

Adding encryption to datacenter transport is challenging because it may sacrifice important transport properties. We present Secure Datacenter Transport protocol (SDT), a design for datacenter transport encryption that preserves three key properties important to protocol designers and datacenter operators.

The *first* property is transport-level support for in-network compute, as defined in MTP [9]. MTP argues the need for the network to identify boundaries of application-level messages, which can be encrypted, within the flow for load balancing, congestion control and fairness guarantee.

The *second* is compatibility with existing hardware offload, particularly for cryptographic operations. Although hardware-based transports with custom NICs have been introduced by hyperscalers, smaller operators would perfer a protocol that is open and can be accelerated by commodity NICs.

The *last*, but not least is natural introduction of a new transport protocol. Although deploying a new transport protocol without UDP encapsulation in the Internet is almost hopeless due to unmodifiable middleboxes [6, 3], this is not the case in datacenters. New protocols, alongside TCP and UDP, would ease network management, in-network compute and load balancing within the host stack.

SDT supports unordered, arbitrary-length encrypted messages over an authenticated session, but with plaintext message identifiers and offsets in packets. This enables the network or the host stack to perform message-granularity operations, such as load balancing. SDT uses TLS offload and segmentation offload available in commodity NICs [8]. Unlike TLS/TCP, SDT enables message-level parallelism in the host stack by serializing the messages into the TLS record sequence only at the bottom of the stack in a rate-controlled manner, preserving the in-host load balancing property required by message-based transports.

This work makes two main contributions:

(1) Identifying a design point for an encrypted message-based datacenter transport protocol that is protocol-number-agnostic, general for other datacenter transports like NDP, and compatible with existing TLS offload.
(2) A proof-of-concept implementation of SDT that exhibits at most 41% lower overhead than TLS/TCP. We also report the implications of this contribution: the porting effort of the application, Redis, and in-kernel user, NVMe-oF, to use SDT, and enabling fast, 0-RTT key exchange for SDT.

REFERENCES

[1] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. "Data Center TCP (DCTCP)". *ACM SIGCOMM*. 2010.
[2] M. Handley, C. Raiciu, A. Agache, A. Voinescu, A. W. Moore, G. Antichi, and M. Wójcik. "Re-Architecting Datacenter Networks and Stacks for Low Latency and High Performance". *ACM SIGCOMM*. 2017.
[3] J. Iyengar and M. Thomson. *QUIC: A UDP-Based Multiplexed and Secure Transport*. RFC 9000. May 2021. URL: https://www.rfc-editor.org/info/rfc9000.
[4] X. Lu and zijian Zhang. *Leveraging Homa: Enhancing Datacenter RPC Transport Protocols*. The Technical Conference on Linux Networking (Netdev 0x17), https://netdevconf.info/0x17/docs/netdev-0x17-paper36-talk-paper.pdf. 2023.

[5] B. Montazeri, Y. Li, M. Alizadeh, and J. Ousterhout. "Homa: A Receiver-Driven Low-Latency Transport Protocol Using Network Priorities". *ACM SIGCOMM*. 2018.

[6] M. F. Nowlan, N. Tiwari, J. Iyengar, S. O. Amin, and B. Ford. "Fitting Square Pegs Through Round Pipes: Unordered Delivery {Wire-Compatible} with {TCP} and {TLS}". *USENIX NSDI*. 2012.

[7] J. Ousterhout. "A Linux Kernel Implementation of the Homa Transport Protocol". *USENIX ATC*. Jul. 2021.

[8] B. Pismenny, H. Eran, A. Yehezkel, L. Liss, A. Morrison, and D. Tsafrir. "Autonomous NIC offloads". *ACM ASPLOS*. 2021.

[9] B. E. Stephens, D. Grassi, H. Almasi, T. Ji, B. Vamanan, and A. Akella. "TCP is Harmful to In-Network Computing: Designing a Message Transport Protocol (MTP)". *ACM HotNets*. 2021.