White Paper

Network-Optimized 5th Gen Intel® Xeon® Scalable Processors
Load Balancing Solution

Accelerating HAProxy with Intel® QuickAssist Technology

Intel® QuickAssist Technology (Intel® QAT) and Intel Crypto Acceleration enhance the performance of HAProxy software load balancer.

Authors:
Mrittika Ganguli
Leo Miao
Divya Pendyala
Xiaobing Qian
Sunny Wang

HAProxy is an open source, fast and reliable reverse proxy that offers high availability, load balancing and proxying for TCP and HTTP-based applications. It is continuously optimized to run on Intel® Xeon® processors including 5th Gen Intel Xeon Scalable processors with built-in Intel QAT accelerators. This processor provides a highly optimized and cost-effective solution that can be used to design load balancing systems to ensure outstanding efficiency and advanced security in the allocation of network resources and traffic management.

This white paper describes the system architecture, components and testing parameters used for HAProxy benchmarking. It introduces the performance advantages of HAProxy on 5th Gen Intel Xeon Scalable processors and showcases the acceleration and performance benefits of HAProxy with Intel QAT. According to the benchmarking results for key performance metrics (handshake and throughput), we observed substantial improvements to HAProxy's highly regarded performance in TLS connections and HTTPS encryption/decryption when accelerated by Intel QAT compared with using the native OpenSSL library. This is of significant reference value for using Intel QAT to accelerate HAProxy in real-world business scenarios.

Table of Contents

HAProxy and Intel QAT

HAProxy

HAProxy is widely used in web hosting, networking and application delivery. It is known for its high performance, flexibility and reliability, making it a popular choice for managing network traffic and ensuring the availability and scalability of web applications.

- **Load balancing:** HAProxy is primarily used for distributing incoming network traffic across multiple backend servers. It balances the load by directing requests to the most appropriate server based on various algorithms such as round robin, least connections and source IP hash.
- **Proxy server:** HAProxy can operate as a reverse proxy, sitting between clients and backend servers. It accepts client requests, forwards them to the appropriate backend server and returns the responses to clients. This allows for a level of abstraction and security, as clients do not interact directly with backend servers.
- **High performance:** HAProxy is known for its speed and efficiency. It's designed to handle many concurrent connections and can perform SSL termination, which offloads the SSL/TLS encryption and decryption process from backend servers to improve performance.

- **Health checking:** HAProxy can continuously monitor the health of backend servers to ensure they are responsive and available. If a server becomes unhealthy, HAProxy can automatically remove it from the load-balancing pool, preventing client requests from being sent to a failing server.
- **SSL/TLS termination:** HAProxy can handle SSL/TLS encryption and decryption, which is crucial for securing web traffic. This allows for SSL termination, improving backend server performance and simplifying certificate management.

HAProxy is commonly used in scenarios where high availability, scalability and performance are critical, such as web applications, e-commerce sites and content delivery networks (CDNs). It can be deployed in various architectures, including as a standalone load balancer, in front of web servers or within containerized environments.

Intel QuickAssist Technology (Intel QAT)

Intel QAT is a hardware acceleration technology built into Intel Xeon Scalable processors to accelerate cryptographic and data compression workloads, enabling performance and efficiency in compute-intensive processes. Starting with 4th Gen Intel Xeon Scalable processors introduce integrated accelerators to achieve greater performance in various workloads, including AI, analytics, high-speed networking and application and content delivery. Intel QAT is one of the integrated accelerators.

Intel QAT offloads compute-intensive cryptographic operations and can significantly boost CPU efficiency and application throughput, while reducing data footprint and power utilization, enabling organizations to strengthen encryption without sacrificing performance.

Intel Crypto Acceleration using Intel AVX-512

Intel Advanced Vector Extensions 512 (Intel AVX-512) is an advanced set of CPU instructions introduced by Intel to enhance the performance of compute-intensive and data-centric workloads. AVX-512 builds upon the foundation of earlier SSE and AVX instruction sets, providing even more powerful vector processing capabilities for modern processors. These vectorized instructions help provide the computational power for HAProxy and are used in software acceleration libraries that include Intel Integrated Performance Primitives (Intel IPP) cryptography and Intel Multi-Buffer Crypto for IPsec with Intel QAT Engine for OpenSSL.

Intel QAT Engine for OpenSSL

Intel QuickAssist Technology Engine for OpenSSL (Intel QAT_engine) supports acceleration for cryptographic operations integrating with OpenSSL framework. It supports acceleration via both hardware path (Intel QAT accelerator) and software (AVX512,Intel crypto libraries) path.

Load balancing applications like HAProxy, that interface with OpenSSL seamlessly integrate with Intel QAT_engine and provide asynchronous cryptographic acceleration. OpenSSL is a versatile toolkit for TLS/SSL protocols that features a modular system designed to accommodate device specific engines. By making the most of this technology, HAProxy optimizes CPU core utilization, allowing for faster encryption tasks with fewer cores. This efficiency improvement enables each core to effectively serve more clients, making it especially suitable for scenarios where network security and performance are paramount.

Workload architecture and configurations

Performance indicators

TLS handshakes per second (TPS): Clients send HTTPS requests with a data payload of zero size. This will utilize key exchange and certificate authentication, exercising only the TLS 1.3 handshake with no bulk data transfer.

Throughput: Throughput is used to measure the maximum network throughput that HAProxy can support under certain test conditions when acting as a proxy server. Real files of a certain size are transmitted to evaluate the network processing power of HAProxy in the HTTPS scenario.

Latency: Latency is the time taken by a request to reach the server and receive a response. We use mean latency as a performance indicator, which is calculated by adding up all the latencies and dividing by the total number of requests. It provides an average value of the time taken by each request to complete.

Workload architecture

HAProxy server is set up as the proxy server in front of an originating Nginx web server, which provides the web content. We use the benchmark tool wrk to send a certain number of HTTPS requests to the HAProxy server, receive responses and finally collect performance data such as handshake, throughput and latency.

This workload consists of three main scenarios, using the workload architecture shown in Figure 1.[1]

- **Native:** In this scenario, the native OpenSSL library is used for TLS connections and HTTPS data encryption and decryption without using any acceleration technology.
- **QATSW:** In this scenario, CPU vAES (Vectorized Advanced Encryption Standard) instructions and crypto libraries are used for TLS connections and HTTPS data encryption and decryption.
- **QATHW:** In this scenario, Intel QAT hardware is used for TLS connections and HTTPS data encryption and decryption.

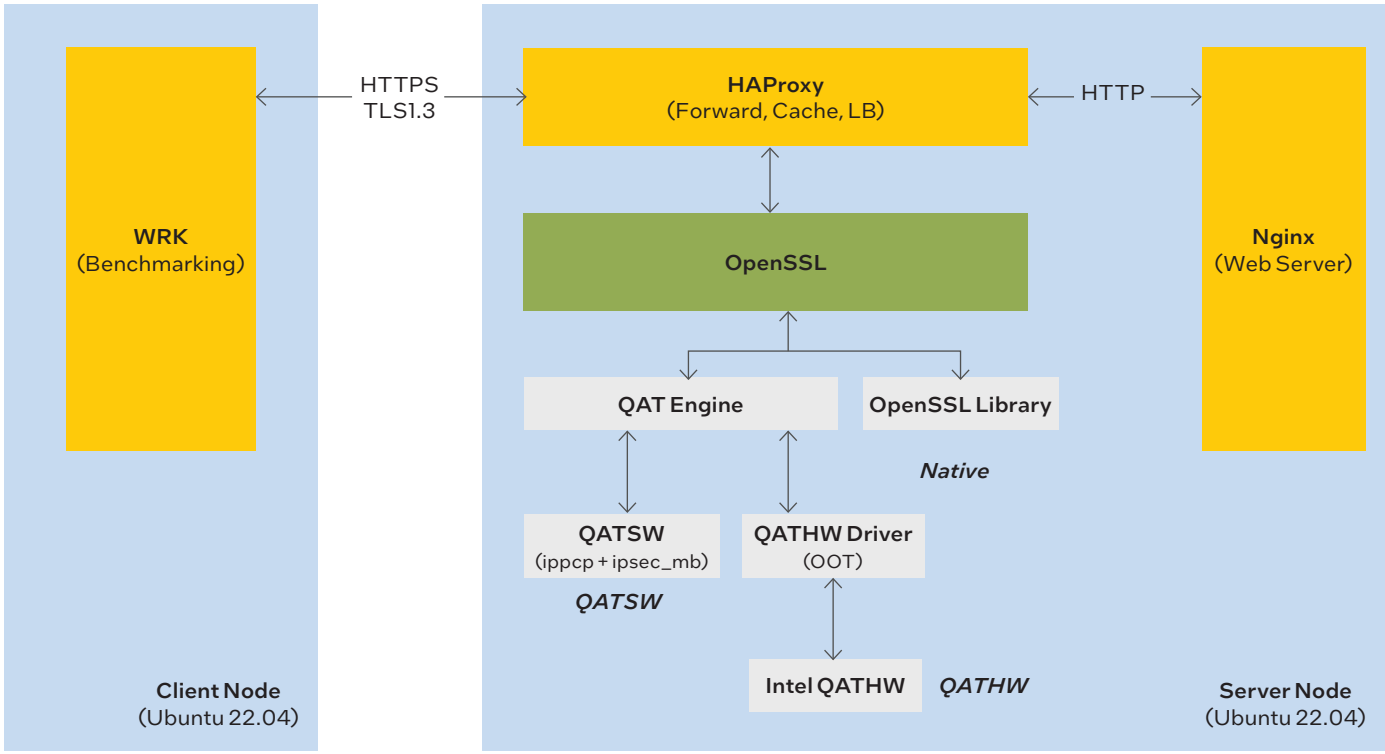

Figure 1. Workload architecture.

Performance testing and analysis

Handshake case

We tested HAProxy handshake and related average latency in Native, QATSW and QATHW scenarios with different HAProxy cores and compared the performance improvement of QATSW and QATHW relative to Native.[2] The HTTPS transferred file size used for handshake case is 0 KB.

Handshake results are shown in Figure 2. HAProxy handshake performance with QATSW is up to 3.25x better than native OpenSSL and up to 4.40x better with QATHW than native OpenSSL. QATHW continues benefiting performance and saving cores for high core count (8C/16T) environments when the accelerator queue is fully utilized.

Latency for the handshake case is shown in Figure 3. HAProxy handshake average latency compared to native OpenSSL is up to 11.98x less with QATSW and up to 42.40x less with QATHW.

The latency tests were performed at maximum handshake performance for each test. The QATHW accelerator queue is fully utilized on the 8C/16T system, so the CPU is no longer saturated and new connections can be handled immediately, which causes the handshake times to be much lower.

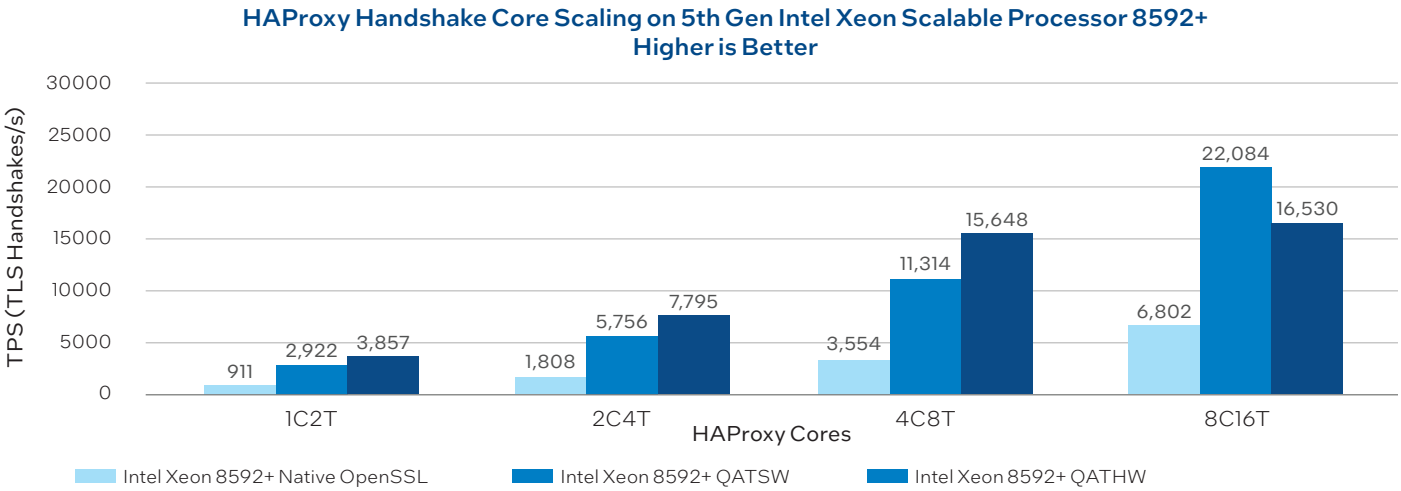

Figure 2. HAProxy core scaling for handshake case.

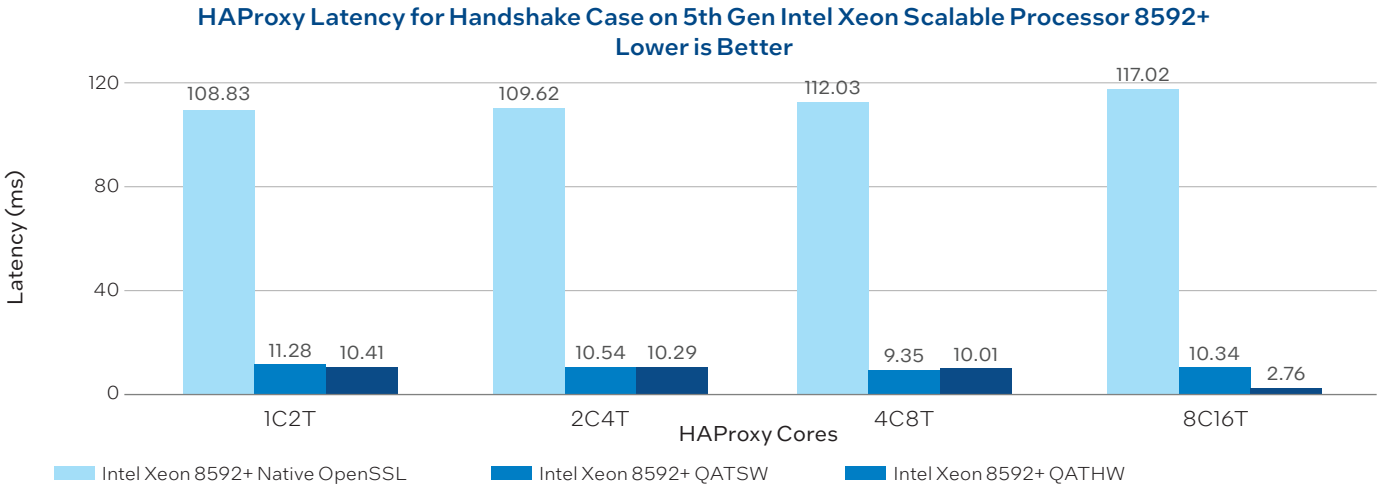

Figure 3. HAProxy latency for handshake case.

Throughput case

We tested the HAProxy throughput and related average latency in Native, QATSW and QATHW scenarios with different HAProxy cores and compared the performance improvement of QATSW and QATHW relative to Native.[3] The HTTPS transferred file size used for the throughput case is 1024 KB.

Throughput results are shown in Figure 4. HAProxy throughput compared to native OpenSSL is up to 1.35x better with QATSW, and QATHW is consistent with Native. QATSW accelerates symmetric cryptography, and QATHW provides the best results for asymmetric/PKE operations.

Latency for the throughput case is shown in Figure 5. HAProxy throughput average latency compared to native OpenSSL is up to 1.37x less with QATSW, and QATHW is consistent with Native.

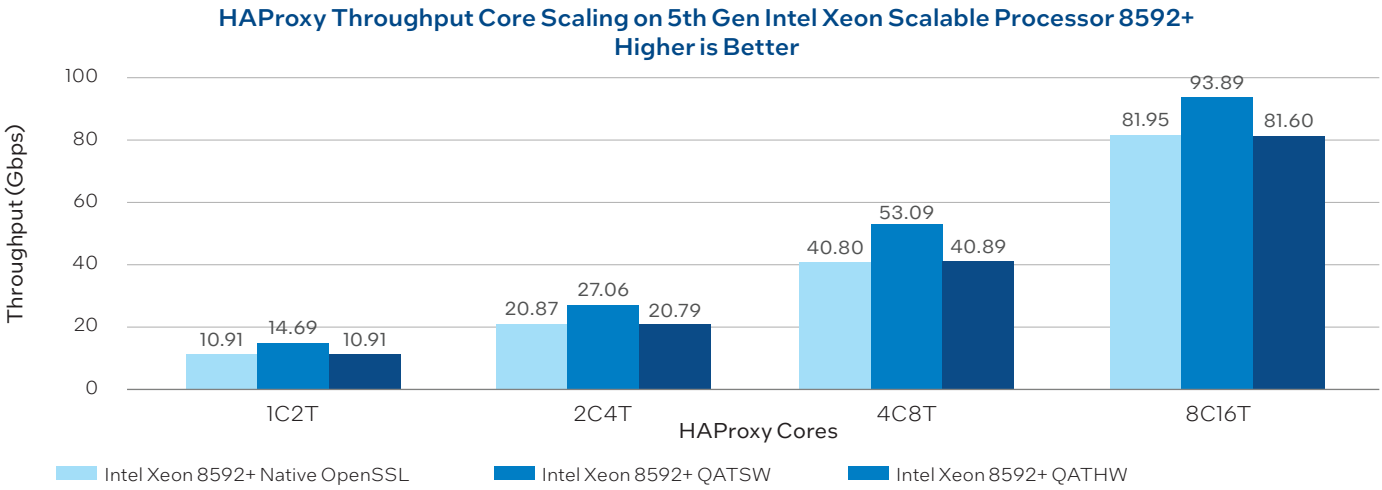

Figure 4. HAProxy core scaling for throughput case.

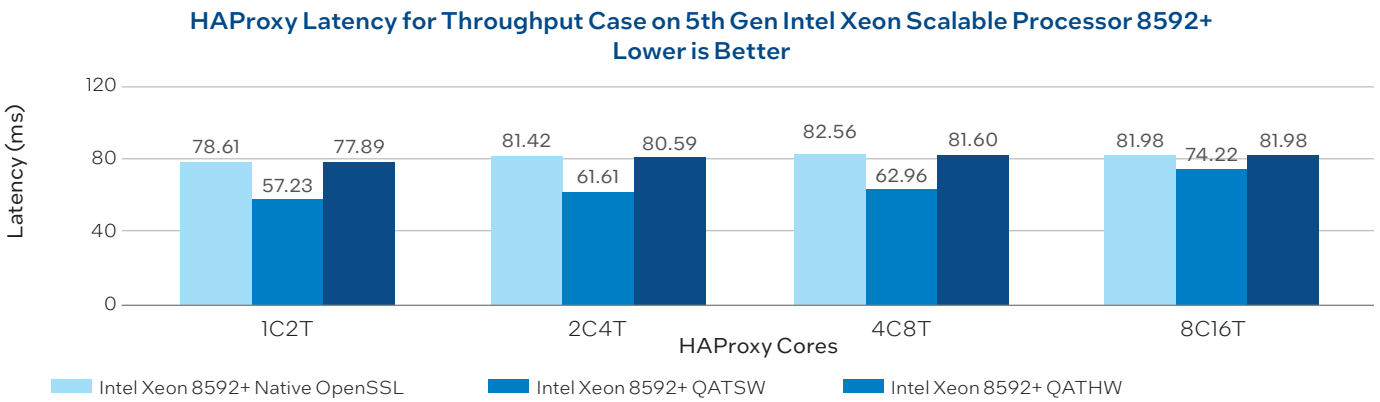

Figure 5. HAProxy latency for throughput case.

Call stack analysis

Table 1 represents the CPU usage distribution across the whole software stack in the 4C/8T scenario as measured by the standard **perf** utility. The table below shows that by using QATSW and QATHW, less CPU cycles are consumed for TLS stack, resulting in higher performance and core savings.

Handshake:

- **For Native mode**, libcrypto.so.1.1 takes about 87.49% CPU usage for handshake and data encryption/decryption.
- **For QATSW**, handshake operations are offloaded to libcrypto_mb.so.11.6 which takes 34.11%, and encryption/decryption (request/response headers) operations are offloaded to libIPSec_MB.so.1.3.0 which takes 0.60%.
- **For QATHW**, only handshake operations can be offloaded to QATHW; it still needs CPU for encryption/decryption, so libcrypto.so.1.1 takes 41.88%, which is higher than QATSW 23.74%.

Throughput:

- **For Native mode**, libcrypto.so.1.1 takes about 35.80% CPU usage for data encryption/decryption, and kernel.kallsyms takes 49.27% to handle packet transmission, which is higher than in the handshake scenario.
- **For QATSW**, libcrypto.so.1.1 takes only 1.48%; this is because payload encryption/decryption is transferred to libIPSec_MB.so.1.3.0, which takes 14.08%.
- **For QATHW**, the CPU usage is similar to Native.

Table 1. Call stack analysis.

CPU Usage	Handshake Case			Throughput Case		
	Native	QATSW	QATHW	Native	QATSW	QATHW
libcrypto.so.1.1	87.49%	23.74%	41.88%	35.80%	1.48%	35.82%
kernel.kallsyms	5.76%	20.16%	27.74%	49.27%	62.72%	49.42%
libc.so.6	2.25%	8.50%	11.28%	7.52%	10.80%	7.47%
libssl.so.1.1	1.87%	5.86%	8.46%	2.75%	3.62%	2.55%
haproxy	2.59%	6.49%	8.40%	4.65%	6.28%	4.73%
libcrypto_mb.so.11.6	N/A	34.11%	N/A	N/A	N/A	N/A
libIPSec_MB.so.1.3.0	N/A	0.60%	N/A	N/A	14.08%	N/A
qatengine.so	N/A	0.68%	0.92%	N/A	0.98%	N/A
libqat_s.so	N/A	N/A	1.02%	N/A	N/A	N/A

Conclusion

This white paper demonstrates the performance of HAProxy load balancer on 5th Gen Intel Xeon Scalable processors, including benchmarking test architecture, performance data and call stack analysis. It also demonstrates how Intel QAT technology can enable significant HAProxy performance improvements for handshake, throughput and corresponding latency.

For 5th Gen Intel Xeon Platinum 8592+ processors with a core base frequency of 1.9 GHz, the performance of HAProxy handshake with QATSW is up to 3.25x of Native and up to 4.40x of Native with QATHW; HAProxy throughput with QATSW is up to 1.35x of Native. Through the call stack analysis, it is further proved that these performance improvements are indeed due to integrating Intel QAT and Intel Crypto technologies. These performance results and systematic analysis methods are of great reference value when building HAProxy architecture based on 5th Gen Intel Xeon Scalable processors.

Learn More

HAProxy

Intel Xeon Scalable Processors

[1] IPERFORMANCE BENCHMARK TEST SYSTEM CONFIGURATIONS: Manufacturer: Intel Corporation; Product name: Intel Reference Platform; BIOS version: EGSDCRB1.86B.0105.D74.2308261927; OS: Ubuntu 22.04.3 LTS; Kernel: 5.15.0-73-generic; Microcode: 0xa1000161; CPU model: Intel® Xeon® Platinum 8592+; Base frequency: 1.9 GHz; Maximum frequency: 3.9 GHz; All-core maximum frequency: 2.9 GHz; CPUs: 256; Threads per core: 2; Cores per socket: 64; Sockets: 2; NUMA nodes: 2; Prefetchers: L2 HW, L2 Adj., DCU HW, DCU IP; Turbo: enabled; Power & perf policy: performance; TDP: 350 watts; Frequency driver: intel_pstate; Frequency governor: performance; Max C-State: 9; Installed memory: 512 GB (16x32GB DDR5 5600 MT/s [5600 MT/s]); Huge pages size: 2048 KB; Transparent huge pages: madvise; Automatic NUMA balancing: enabled; Network adapter summary: 1x Ethernet Controller I225-LM, 4x Ethernet Controller E810-C for QSFP (1 port for HAProxy server); Drive summary: 1x 465.8G Hitachi HTS72505.

PERFORMANCE BENCHMARK TEST SOFTWARE CONFIGURATIONS: HAProxy version: v2.8.0; wrk version: 4.2.0; Nginx version: 1.25.1; OpenSSL version: 1.1.1u; TLS version: 1.3; QAT engine version: v1.2.0; ipp-crypto version: ippcp_2021.7.1; ipsec-mb version: v1.3; QAT hardware driver version: QAT20.L.1.0.50-00003; QAT device number: 1; Compiler: gcc version 11.4.0 (Ubuntu 11.4.0-1ubuntu1~22.04); OS version: Ubuntu 22.04.3; Kernel version: 5.15.0-73-generic; HAProxy cores: 65-80, 193-208 (On socket1, depends on core scaling); Nginx cores: 96-103, 224-231 (On socket1); Nginx workers: 16; NIC model: Ethernet Controller E810-C for QSFP; NIC firmware: 4.30 0x8001af29 1.3429.0; NIC driver: ice; Cipher suite: TLS_AES_256_GCM_SHA384; Certificate: RSA-2048; Curve: X25519.

[2] TEST CONFIGURATIONS FOR HANDSHAKE CASE: Tested by Intel as of 10/18/2023. CPU operating frequency: 1.9 GHz; Uncore frequency: 1.9 GHz; File size: 0KB; Keep-alive: off; IRQ: bind to HAProxy worker cores; wrk cores: 2/4/8; wrk threads: 2/4/8; wrk concurrency: 100 * wrk thread; wrk duration: 30s; QAT device: 1 QATHW device on single socket; Nginx core: 8C16T; Nginx workers: 16; Cipher: TLS_AES_256_GCM_SHA384; Certificate: RSA-2048; Curve: X25519.

[3] TEST CONFIGURATIONS FOR THROUGHPUT CASE: Tested by Intel as of 10/18/2023. CPU operating frequency: 1.9 GHz; Uncore frequency: 1.9 GHz; File size: 1024KB; Keep-alive: on; IRQ: bind to HAProxy worker cores; wrk cores: 2/4/8; wrk threads: 2/4/8; wrk concurrency: 50 * wrk thread; wrk duration: 30s; QAT device: 1 QATHW device on single socket; Nginx core: 8C16T; Nginx workers: 16; Cipher: TLS_AES_256_GCM_SHA384; Certificate: RSA-2048; Curve: X25519.

Availability of accelerators varies depending on SKU. Visit the Intel Product Specifications page for additional product details. Performance varies by use, configuration and other factors.

Learn more at https://www.intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See configuration disclosure for configuration details.

No product or component can be absolutely secure.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy. Your costs and results may vary.

Intel technologies may require enabled hardware, software or service activation.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a nonexclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

0124/DL/MESH/PDF 356873-001US