10GbE UDP using Onload® on Red Hat Enterprise Linux 9.7 with AMD Solarflare™ X4522-PLUS NICs in Supermicro AS -2015HS-TNR Servers using AMD EPYC™ 9475F Processors

SUT ID: N1-20260926

STAC REPORT

Test date: 27 September 2026

Release: v1.0, 05 October 2026
Message size: 66b

These tests followed STAC benchmark specifications proposed or approved by the STAC Benchmark Council (see www.stacresearch.com). Be sure to check the version of any specification used in a report. Different versions may not yield results that can be compared to one another.

This document was prepared by the Strategic Technology Analysis Center (STAC®) at the request of Supermicro. This document is provided for your internal use only and may not be redistributed, retransmitted, or published in any form without the prior written consent of STAC. "STAC" and all STAC names are registered trademarks or trademarks of the Securities Technology Analysis Center LLC. All other trademarks in this document belong to their respective owners.

The test results contained in this report are made available for informational purposes only. Neither STAC nor the vendor(s) supplying the information in this report guarantee similar performance results. All information contained herein is provided on an "AS IS" BASIS WITHOUT WARRANTY OF ANY KIND. STAC explicitly disclaims any liability whatsoever for any errors or otherwise.

Copyright @ 2026 STAC. "STAC" and all STAC names are trademarks or registered trademarks of the Securities Technology Analysis Center, LLC. Other company and product names are trademarks of their respective Owners.

References....3
Summary....3
Vendor Commentary....6
The stack under test....6
Overview....6
Configuration Details....6
Product Commentary....6
Project contributors and their roles....7
Contacts....7
Methodology....8
Specifications....8
STAC Test Harness....8
Message Patterns used in this project....8
STAC-N1 Binding used in this project....8
Limitations....9
Detailed latency analysis....12
Section contents....12
Test Sequence PINGPONG....13
Base rate....14
Highest successful rate....17

Copyright © 2026 STAC

References

[1] Configuration Disclosure: "10GbE UDP using Onload® on Red Hat Enterprise Linux 9.7 with AMD Solarflare™ X4522-PLUS NICs in Supermicro AS -2015HS-TNR Servers using AMD EPYC™ 9475F Processors", www.stacresearch.com/N1-20260926

Summary

STAC recently performed a STAC-N1 Benchmark audit of UDP multicast over 10GbE using AMD Onload kernel bypass with AMD Solarflare X4522-PLUS network adapters in Supermicro AS -2015HS-TNR servers with AMD EPYC™ 9475F processors. These tests were conducted at the request of Supermicro.

STAC-N1 uses the STAC-N toolset to test a network stack using a market data style workload. STAC-N1 is designed to provide insightful network benchmarks that are neutral with respect to vendor, network API, and network transport. STAC-N1 can test many different combinations of network API, I/O mode, network stack, operating system, and system hardware.

The SUT used in these tests consisted of:

STAC-N1 Pack for UDP-TCP binding (udp-tcp-sock-1.0.3) running on two Supermicro AS -2015HS-TNR servers, each with one AMD EPYC™ 9475F 48-core processor at 3.65 GHz (4.8 GHz max boost), 384 GiB of DDR5 memory (12 x 32 GiB at a configured 5200 MT/s), and an AMD Solarflare X4522-PLUS network adapter (sfc driver 6.3.1.1000, firmware bundle v1.3.2.105) in a PCIe Gen5 x8 slot, running Red Hat Enterprise Linux 9.7 (kernel 5.14.0-611.47.1.el9_7.x86_64) with AMD Onload 9.2.0.43 kernel bypass. The two servers were connected directly to one another by a 1 m SFP56 direct-attach copper cable, linked at 10 Gb/s; there was no switch. One server acted as the Producer and hosted the test orchestrator; the other acted as the Consumer.

The key results are tabulated in the STAC Report Card on the following page, and Section 8 provides a detailed analysis of latency and throughput. Of these results, AMD and Supermicro wish to highlight the following:

- *At the base rate, the solution outperforms all publicly disclosed results on mean, median, and P99 (STAC.N1.β1.PINGPONG.LAT1).*
- *The lowest mean, median, and P99 at the highest rate tested, 2.5 million messages per second (STAC.N1.β1.PINGPONG.LAT2), across SUTs with the same or lower highest rate tested.*

The Overview section introduces STAC-N1 and is followed by a high-level overview of the SUT and its configuration. Further details on the configuration are available to STAC Insights subscribers at [1]. This includes a STAC Configuration Disclosure and may also include product-specific diagnostic files (e.g., sosreport).

Note that because STAC-N1 is not tied to a particular network API, it can be used to compare stacks using different APIs (for example, UDP/Ethernet vs RDMA/Infiniband). However, STAC-N1 is often used to compare different stacks using the same API (for example, UDP with one vendor's NIC and driver vs UDP with another vendor's NIC and driver).

When making the latter type of comparison, it is essential that the SUTs you are comparing used the same STAC-N1 binding. It is also essential that they used the same functions in the API. For example, there are several I/O modes for sockets (writev, sendrec, epoll, etc.) as well as for RDMA (read, write, sendrec). If you want to compare network stacks, you should make sure that the SUTs used the same I/O mode. By the same logic, if your goal is to compare different APIs using the same platform, it is essential that you make sure the SUTs you're comparing used the same NIC, OS, server, etc.

Copyright © 2026 STAC

STAC-N1 Report Card

STAC-N1. β1.PINGPONG
SUT: N1-20260926

Results summary

Type	Spec ID	Description	Value	Mean	Med	P99	P99.99	Max	Std Dev
Throughput	STAC-N1.β1.PINGPONG.TPUT1	Highest supply rate tested (msg/sec)	2,500,000						
Latency	STAC-N1.β1.PINGPONG.LAT1	SupplyToReceive latency at base rate (usec)		1.03	1.02	1.18	1.33	4.00	0.06
Latency	STAC-N1.β1.PINGPONG.LAT2	SupplyToReceive latency at max rate (usec)		1.04	0.99	1.55	2.02	4.96	0.15
Latency	STAC-N1.β1.PINGPONG.LAT3	SendToReceive latency at max rate (usec)		0.99	0.95	1.41	1.86	4.48	0.13
CPU	STAC-N1.β1.PINGPONG.CPU1	Consumer CPU utilization at base rate (core equivalents)	Active cores = **2.0**	2.00				2.00	
CPU	STAC-N1.β1.PINGPONG.CPU2	Producer CPU utilization at base rate (core equivalents)	Active cores = **2.0**	2.00				2.00	
Memory	STAC-N1.β1.PINGPONG.CMEM1	Consumer memory usage at base rate (MB, adjusted)	857						
Memory	STAC-N1.β1.PINGPONG.CMEM2	Consumer memory usage at base rate (MB, unadjusted)	6,593						
Memory	STAC-N1.β1.PINGPONG.PMEM1	Producer memory usage at base rate (MB, adjusted)	854						
Memory	STAC-N1.β1.PINGPONG.PMEM2	Producer memory usage at base rate (MB, unadjusted)	16,147						

Source: STAC
www.STACresearch.com
Copyright © 2026 STAC

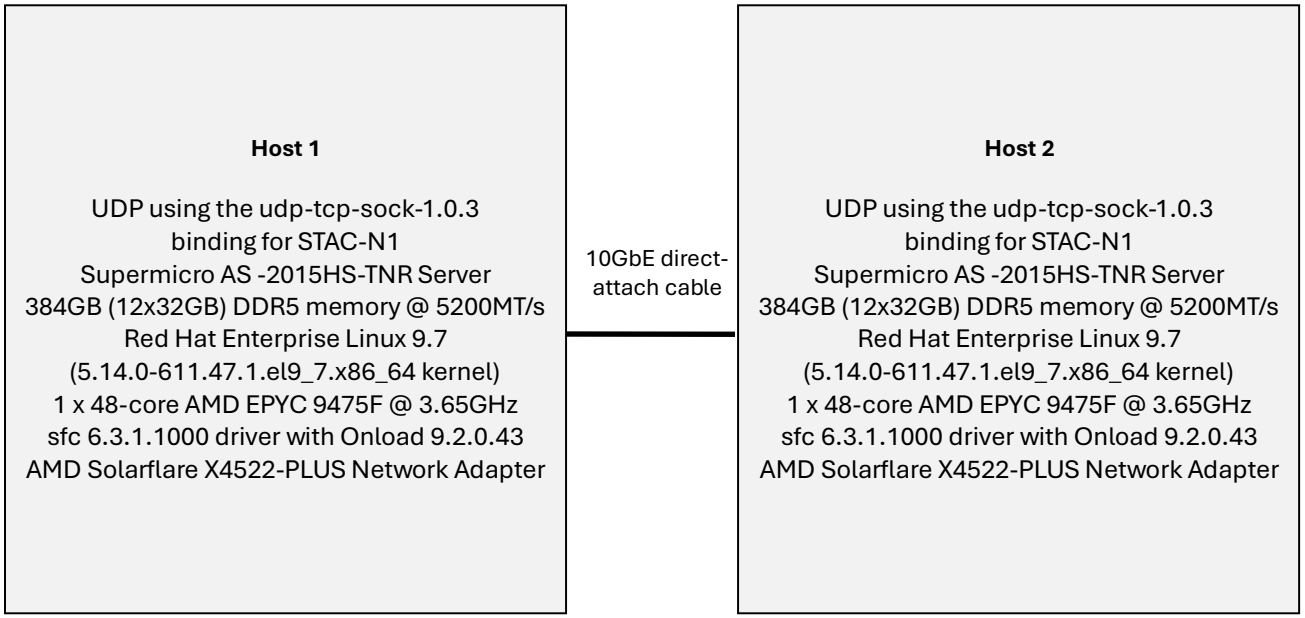


Figure 1 – Test configuration

Copyright © 2026 STAC

Vendor Commentary

AMD and Supermicro provided the following comments:

*The **Supermicro AS -2015HS-TNR** server, powered by **AMD EPYC™ 9475F** processors and **AMD Solarflare™ X4522-PLUS** network adapters running **AMD Onload™** kernel-bypass networking, is a compelling end-to-end solution for latency-critical trading applications. Every layer of the stack was tuned: the BIOS settings were optimized for latency-sensitive workloads, a low-latency tuned profile with core isolation at the OS level and Onload stack tuning to bypass kernel. Together, the Supermicro / EPYC 9475F / X4522-PLUS / Onload combination delivers the highest throughput and lowest mean, median and consistently low tail latency through the 99.99th percentile. For trading firms, where worst-case behavior defines risk, this balance of sustained throughput and bounded tail latency is the key takeaway. AMD and Supermicro continue to work together toward higher MPPS, determinism, and lower latency.*

The stack under test

Overview

The SUT comprised two **Supermicro AS -2015HS-TNR** servers, each equipped with **AMD EPYC™ 9475F** 48-core processors and **AMD Solarflare™ X4522-PLUS** network adapters. The two servers were connected back-to-back by a single direct-attach cable with no intervening switch. Both systems ran **Red Hat Enterprise Linux 9.7**, with **Onload 9.2.0.43** providing kernel-bypass networking. This configuration is described in more detail in Figure 1, as well as the sections that follow.

Configuration Details

Details of the SUT configuration are available in the STAC Vault to qualified members of the STAC Benchmark Council at [1]. This includes a STAC Configuration Disclosure as well as the sosreport.

Product Commentary

AMD provided the following information and claims about its products used in this SUT:

5th Gen AMD EPYC™ processors are designed to deliver exceptional performance, energy efficiency, and advanced security capabilities for demanding data center workloads. Built on the "Zen 5" architecture, the AMD EPYC 9475F processor provides 48 cores and 96 threads, with an all-core boost frequency of 4.4 GHz and a maximum boost frequency of up to 4.8 GHz. Its combination of high frequency, 256 MB of L3 cache, and substantial memory and I/O bandwidth makes it well suited for performance and latency-sensitive workloads, including financial-services applications. With 12 channels of DDR5 memory supporting speeds up to 6400 MT/s and 128 lanes of PCIe® 5.0 connectivity, the AMD EPYC 9475F processor provides a balanced platform for compute-intensive and high-throughput applications. AMD Infinity Guard and AMD Infinity Architecture further support the security, scalability, and reliability requirements of modern data centers.

AMD Solarflare™ X4522-PLUS Ethernet adapters are designed for high-performance electronic trading environments. Built for the speed of modern markets, these adapters enable ultra-low latency and real-time telemetry, empowering high-frequency trading, risk analytics, and financial data flows with precision, scale, and confidence. Providing dual SFP ports of 1/10/25/50 GbE and a PCIe Gen5 x8 host interface, the X4522 achieves lower latency and improved system performance over the previous generation, and presents a complete solution space

Copyright © 2026 STAC

when paired with the Onload suite of acceleration protocols which seamlessly alleviate kernel contention for traditional user workloads.

Supermicro provided the following information and claims about its products used in this SUT:

The Supermicro Hyper AS -2015HS-TNR server is engineered to deliver deterministic, high-throughput compute and low-latency I/O performance for mission-critical financial market infrastructure. Built around a single-socket architecture supporting the AMD EPYC™ 9005/9475F series processor family in an SP5 socket, the AS -2015HS-TNR eliminates NUMA cross-socket interconnect latency (xGMI/UPI hops) by routing 128 PCIe® Gen 5.0 lanes directly from the processor root complex to high-density expansion slots.

To support sustained, deterministic packet processing under high-burst market data conditions, the AS -2015HS-TNR features an enterprise 2U thermal design with optimized fan curves and redundant titanium-level power supplies. This ensures high-TDP, high-boost trading processors and low-latency network accelerators—such as the AMD Solarflare™ X4522-PLUS—operate well within thermal operating margins without frequency throttling or jitter. With support for up to 12 channels of DDR5 memory and flexible PCIe Gen 5 riser configurations, the AS -2015HS-TNR combines the latency determinism of a single-socket system with enterprise-grade serviceability, remote out-of-band management (Redfish/IPMI), and turnkey deployment across global trading co-location facilities.

Project contributors and their roles

The following firms contributed to this project:

- AMD
- Supermicro
- STAC

The Project Participants had the following responsibilities:

- AMD tuned the SUT, provided the network adapters, operating system, lab space, server administration, and sponsored the test.
- Supermicro provided the server chassis and boards.
- STAC supplied the binding, supplied and supported the STAC-N test harness software, inspected the SUT and test harness configuration, conducted the tests, and produced the STAC Report and STAC Configuration Disclosure.

Contacts

- For questions about AMD products, contact Nimisha Raut (nimisha.raut@amd.com), Ashwin Thiagarajan (ashwin.thiagarajan@amd.com), Sylvester Rajasekaran (sylvester.rajasekaran@amd.com)
- For questions about Supermicro products, contact Shahzada Sufyan (shahzadas@supermicro.com), Raj Sugathan (rajs@supermicro.com), Arpitha Srinivasan (arpithas@supermicro.com), Susmitha Kuppireddy (susmithak@supermicro.com)
- For all other questions, contact info@STACresearch.com

Copyright © 2026 STAC

Methodology

Specifications

This project followed STAC-N1 β1 Benchmark specifications. More information is available at www.STACresearch.com/nio.

STAC Test Harness

The STAC-N1 binding configuration used in this project is described in the Stack Under Test section. The source code for the binding is available to qualified members of the STAC Benchmark Council.

Message Patterns used in this project

Message Size on socket write call	66 Bytes
Initials message pattern	initials_msg_file_pattern_for_66B.bin
Deltas message pattern	deltas_msg_file_pattern_for_66B.bin

STAC-N1 Binding used in this project

Producer binary name	udp_tcp_sock_producer
Producer binary version	Built from stac-n-test-harness commit 2509453 (30 July 2026). Byte-identical to an independent rebuild of that commit; local changes are confined to orchestration and configuration scripts, which are not compiled. Built 2026-09-19, release build, GCC 11.5.0
Producer binary configuration	Mode: UDP. IOMUX: libeventepoll. Send function: sendmsg. Receive function: recvmsg. Blocking: enabled
Consumer binary name	udp_tcp_sock_consumer
Consumer binary version	Built from stac-n-test-harness commit 2509453 (30 July 2026). Byte-identical to an independent rebuild of that commit; local changes are confined to orchestration and configuration scripts, which are not compiled. Built 2026-09-19, release build, GCC 11.5.0
Consumer binary configuration	Mode: UDP. IOMUX: libeventepoll. Send function: sendmsg. Receive function: recvmsg. Blocking: enabled
Compiler version used to build STAC-N Library and Producer/Consumer binaries	GCC 11.5.0 (gcc (GCC) 11.5.0 20240719, Red Hat 11.5.0-11); glibc 2.34

Key to binding config parameters:

- Mode: UDP sockets IOMUX: Network notification via libevent library in epoll mode Send function: Socket library sendmsg function used to send messages Receive function: Socket library recvmsg function used to read messages Blocking: Send and receive functions are set to blocking

Copyright © 2026 STAC

Limitations

- The STAC-N1 Test Harness relies on software instrumentation for measurement, which has an impact on performance. These limitations are consistent for all systems tested with this harness but are worth noting. In particular: 1) Because it is a reflection methodology, the latency measurements include some latency for the STAC-N Library to reflect messages (which is divided by 2 along with the rest of the round-trip latency); 2) reflection in the Consumer also takes CPU time, which increases the observed CPU utilization of the Consumer and may decrease throughput; 3) the Producer is required to call the STAC-N Library to record a "Send" timestamp, which consumes some CPU and may reduce the max sustainable rate.
- When deployed in a Consumer, the threading model of the STAC Library used for this version of the specifications emulates an application that requires business logic to execute in a different thread from message-consumption threads and requires serialized input to the business-logic thread from those other threads. This imposes a single-threaded bottleneck on performance of a single Consumer. Another common application pattern in the industry is for an application to scale across multiple cores by partitioning business logic by symbol and running multiple partition threads, each performing its own in-line message consumption. The latter pattern would potentially allow a single client to achieve higher throughput, but this use case is not currently defined for STAC-N1. Similarly, each Producer has a single Supplier thread, which limits the total possible message supply rate for a given Producer. Specifications that allow an arbitrary number of Suppliers on a given machine may be added in the future but are not part of the current STAC-N1 definition.
- The message sizes of STAC-N1 are unlikely to push a high-bandwidth link to saturation. However, the maximum message rates of the harness observed in the lab exceed most of the relevant message rates for a single network interface in today's financial trading. This can be measured using an inline binding that directly connects the Supplier and Collector queues on a single machine without using a network stack. On a 2010-generation Westmere processor, such a "loopback" harness sustained over 3.3 million messages per second. Nevertheless, enhancements that allow higher rates across a single interface are under consideration for the future.
- The STAC-N test harness enforces an in-order requirement on messages. For market data, such ordering must typically be performed somewhere before reaching the application business logic, so the harness captures the performance impact, if any, of this requirement.
- The philosophy behind STAC-N1 bindings is to design each binding optimally for each API while making the same assumptions about use cases. However, these bindings may have different authors, thus they may have subtle differences in implied use cases. Readers comparing the performance of different bindings should keep this in mind.
- This project used STAC-N1 sockets binding version udp-tcp-sock-1.0.3. Since the last September 2024 STAC-N1 results, the STAC-N1 Working Group has made the test harness more efficient at handling benchmark data during execution. The benchmark specification, methodology, rules, metrics, and reporting requirements are unchanged — results remain directly comparable to all prior STAC-N1 publications. The lower overhead in the updated harness may contribute to some throughput and latency gains alongside genuine system-level improvements from vendors. STAC has not quantified the harness's specific contribution.
- As described in the Overview section, the Max Message Rate is only accurate to the granularity of the step size.

Copyright © 2026 STAC

Overview of the STAC-N1 Benchmark Specifications

Organizations in the financial markets remain focused on reducing the latency and jitter of network communication while maintaining sufficient headroom for anticipated traffic peaks. The ability to respond to events quickly and predictably--especially during bursts of activity--continues to be crucial to profitability. At the same time, infrastructure vendors continue to deliver new and optimized solutions to these challenges. In 2012, trading organizations and vendors in the STAC Benchmark Council created a new methodology and toolset called STAC-N to test network solutions under financial market workloads.

The objective was to establish a vendor-independent benchmark standard that would accommodate the current and future variety of APIs and implementations. While network performance had always been fundamental to certain STAC Benchmarks (e.g., STAC-M1 for market data feed handler solutions, STAC-M2 for market-data distribution middleware solutions), STAC-N represented the considerable group of trading applications that integrate directly with network APIs, without intervening third-party software. At first, the STAC-N tools were used only to produce research that stayed within the STAC Benchmark Council; but in 2013, STAC began making some of these benchmark results public.

The key requirements informing the development of STAC-N were to:

- provide a vendor-neutral, level playing field across numerous network APIs that nevertheless allows for configurations that exploit each API to its fullest
- provide a detailed latency analysis
- provide statistics on throughput, CPU, and memory
- automate tests and analysis as much as possible

STAC-N is capable of representing many different kinds of message patterns and application usage patterns. Council members using the STAC-N tools are free to change these patterns as it suits them. The first set of patterns to be proposed as a benchmark standard is STAC-N1, the methodology used in this project. Further patterns and STAC-N*x* Benchmarks may be defined in the future.

The core of the test harness is the STAC-N Library, a close derivative of the STAC-M2 Library. STAC-N uses a "reflection" methodology for round-trip time-stamping, illustrated in Figure 2. A Producer transmits a "primary" message; a Consumer consumes the message and republishes it as a "reflected" message; and the Producer consumes the reflected message. Unlike STAC-M2, the reflection ratio is 100%; that is, every message from the Producer to Consumer results in a message from Consumer to Producer. (The STAC-M2 Advanced Test Harness also supports one-way measurement, but that feature has not been incorporated into STAC-N at this point.)

The STAC-N library takes a pre-defined message pattern as input. The message pattern for STAC-N1 is modeled on observed output of US equities order-book feed handlers deployed in the field, with a 232-byte payload. This message pattern is configured for a steady-state rate. The STAC-N Library supplies messages to the Producer Adapter as pointers to C structs containing a 24- to 32-byte header and the message payload.

STAC-N1 currently uses a classic "ping pong" distribution pattern. This sequence involves a single Producer and Consumer. Starting with the configured Base Message Rate (100K mps), the harness executes a test at the given rate, then moves to the configured Max Message Rate and continues down to the configured minimum rate at regular steps. There are two separate runs of each rate set. Once all rates have been tested twice, the harness automatically generates a detailed analysis report and CSV files to allow further analysis in Excel or other tools.

The total size of the message buffer as passed to the UDP/TCP socket write call was always 66 bytes (a 24-byte binding header plus a 42-byte STAC-N payload). Also, the Minimum Message Rate was set to 1000 mps to simulate a "request/response" messaging paradigm.

Copyright © 2026 STAC

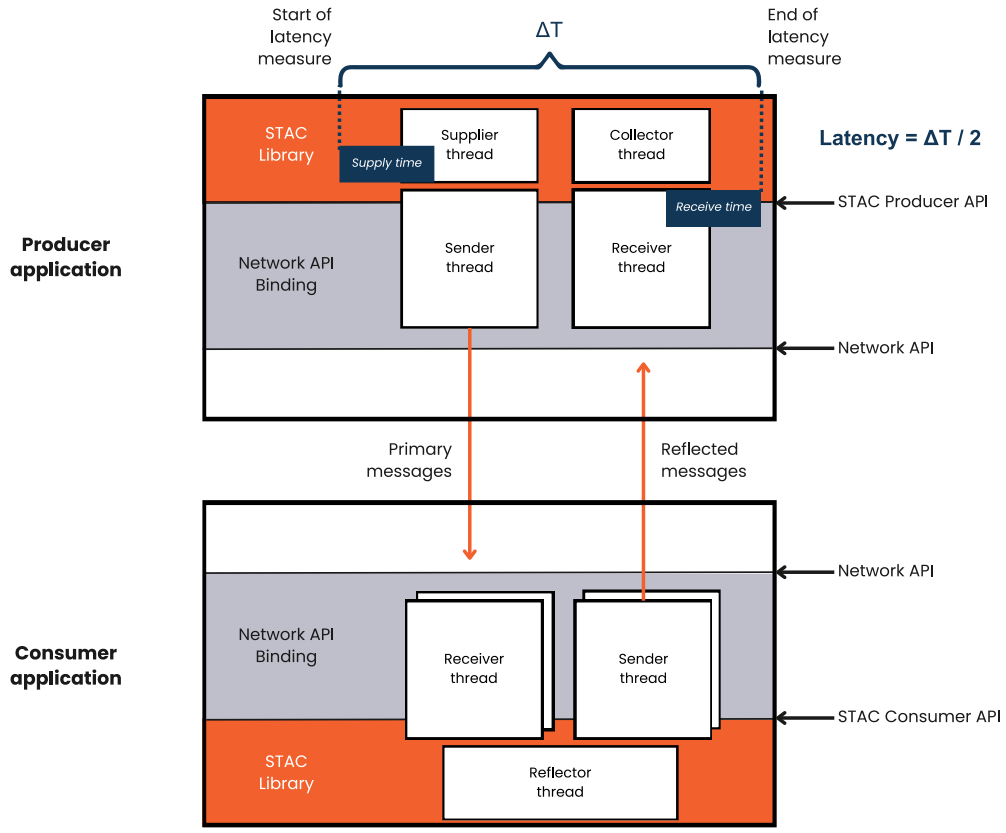


Figure 2 – Construction of STAC-N Test Clients

The harness computes several metrics from each test sequence, including the following:

- SupplyToReceive Latency (Hybrid) - (LAT1). Latency measured from the earliest moment a primary message is available for sending in the Producer to the moment the reflected message payload is available for consumption in the Producer (ReceiveTime minus SupplyTime in Figure 2). This result is divided by two to provide a "Hybrid Latency" approximation of one-way latency. (LAT1 is defined as the SupplyToReceive Latency while running at the Base Rate, and LAT2 is while running at the Max Message Rate).
- SendToReceive Latency (Hybrid) - (LAT3). Latency measured from the moment the Producer Adapter notifies the STAC Library that it has removed the message pointer from the queue, to the moment the reflected message payload is available for consumption in the Producer. This result is also divided by two to provide a "Hybrid Latency" approximation of one-way latency. SendToReceive latency is measured only at the Max Message Rate. The difference between SendToReceive and SupplyToReceive Latency indicates how much of SupplyToReceive Latency is due to queuing.
- Max Message Rate (TPUT1). The highest supply rate that does not result in a Failure Event. A Failure Event is defined as the occurrence of a Primary Message that is supplied by the STAC Library and is not observed at a Consumer within 1 second, or the occurrence of a Reflected Message that does not arrive at the Producer within 2 seconds of the supply of its corresponding Primary Message. (In other words, there is a 1 second timeout in each direction.) Max Message Rate is only accurate to the step size configured for a test. In these tests, the step size was 100K mps, starting from a base rate of 100K mps. This means, for example, that the max rate for a SUT whose true max rate is 390K mps would be reported as 300K mps, because the test at 400K mps would fail.
- Consumer Memory 1 (CMEM1) and Producer Memory 1 (PMEM1). Maximum memory used by the Consumer or Producer application process, respectively, minus the memory allocated by the STAC Library. Memory measurements were obtained by sampling the "VmRSS" kernel statistic from the application-specific " /proc/<pid>/status" system file and finding the max value during the run. STAC Library memory is

Copyright © 2026 STAC

determined separately from the test runs, in a measurement taken of a no-op Consumer or Producer, as appropriate, on the platform. If the Consumer uses very little memory, small variations and rounding can cause this result to be zero or slightly negative.

- Consumer Memory 2 (CMEM2) and Producer Memory 2 (PMEM2). Maximum memory used by the Consumer or Producer application process, respectively. Memory measurements were obtained by sampling the "VmRSS" kernel statistic from the application-specific " /proc/<pid>/status" system file and finding the max value during the run. This includes memory used by the STAC Library.

Testing a given network API requires a binding between that API and the STAC-N Library. The binding used in this project is described in Section 6. The source code to the binding is available to qualified members of the STAC Benchmark Council. To request this, see www.STACresearch.com/nio.

This project used a binding to the Berkeley Sockets API. The author of the initial version was Solarflare. Chelsio updated the binding to provide multiple read/write, iomux, and blocking modes as well as an option to use TCP. See section 6.3, "STAC-N1 Binding used in this project", for the specific read/write/iomux/blocking modes used in this benchmark.

Detailed latency analysis

Section contents

The sections that follow represent latency statistics in a number of ways to facilitate analysis. These tables and charts were created automatically by STAC report-generation tools from the test data.

First is a chart of latency vs throughput at a range of throughputs (with just a few latency statistics per throughput). The range was determined by the max message rate of the system (TPUT1). See Section 1 for details about this rate.

Next are tables and charts that provide an in-depth look at latency when running two specific message rates:

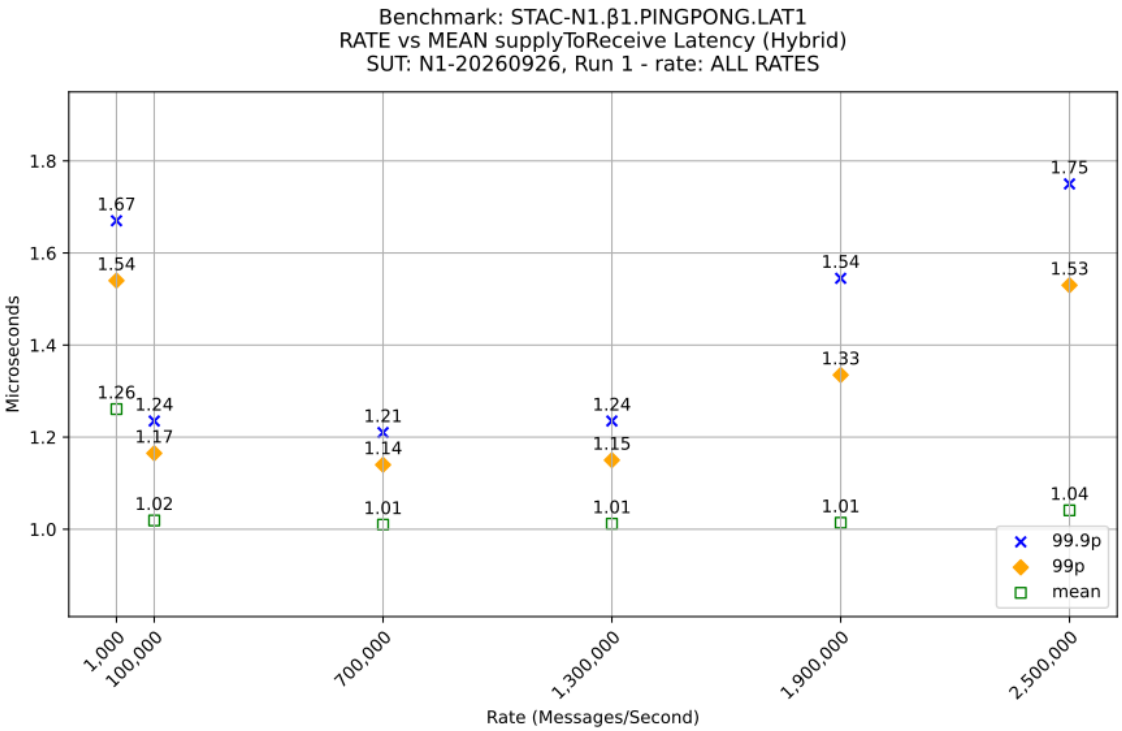
- Base rate. This rate is used for all systems tested, to provide a common basis for comparing latency.
- Max rate (TPUT1). This rate will, in general, vary from system to system. The latencies at this rate are very important for understanding the performance profile of this system, but they are not directly comparable to those of a system with a different TPUT1.

Each analysis for a given message-supply rate contains:

- A tabular summary of latency statistics across all test runs.
- A graphical summary of the same.
- A histogram plotting frequency of occurrence against latency values from 1-10,000 microseconds using a logarithmic scale for latency that ranges from 1-microsecond to 1,000-microsecond buckets.
- Time plots of MEAN and MAX latency over the course of one test run, using 100-millisecond intervals. NOTE: Time plots are taken from just one run (the first run). For this reason, they may not correspond exactly to the summary statistics, which are taken across both runs. For example, if the max latency occurs in the second run, it will not appear in the time plot.

Copyright © 2026 STAC

Test Sequence PINGPONG



Source: STAC®
www.STACresearch.com
Copyright © 2026 STAC

Base rate

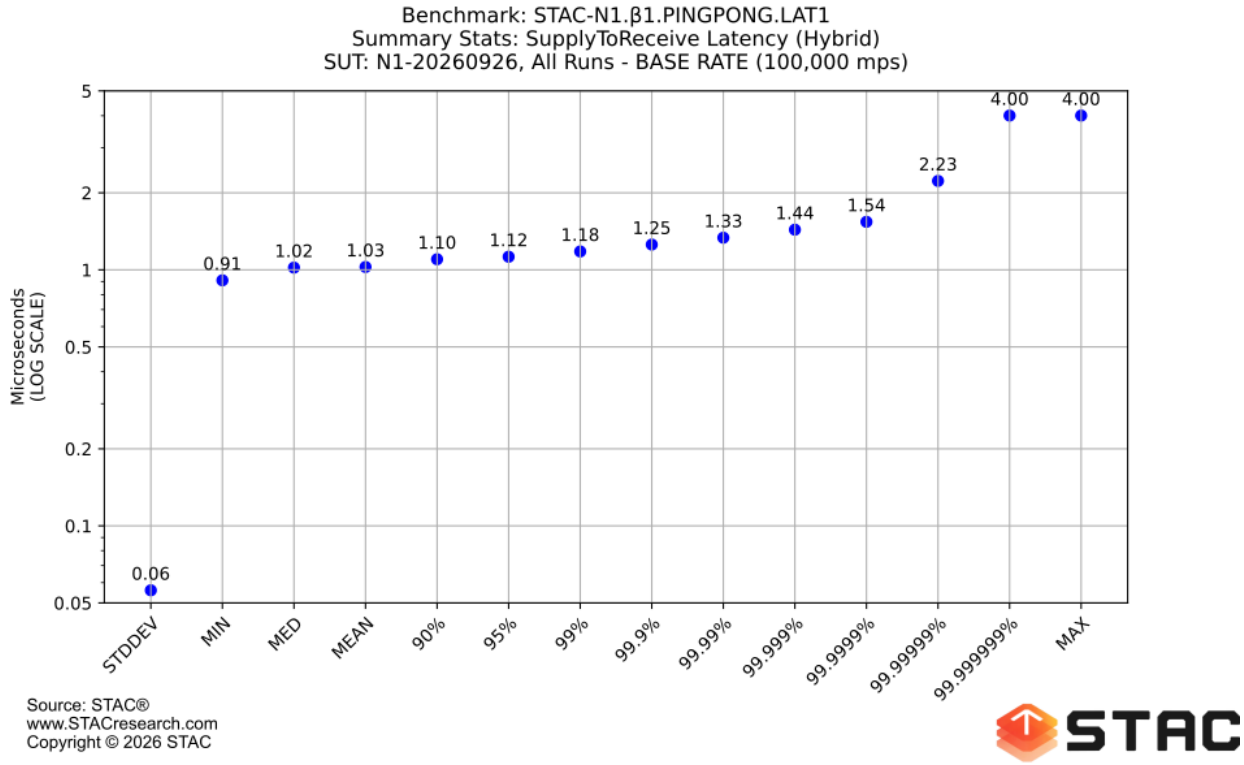
Summary stats for BASE_RATE runs in test sequence PINGPONG

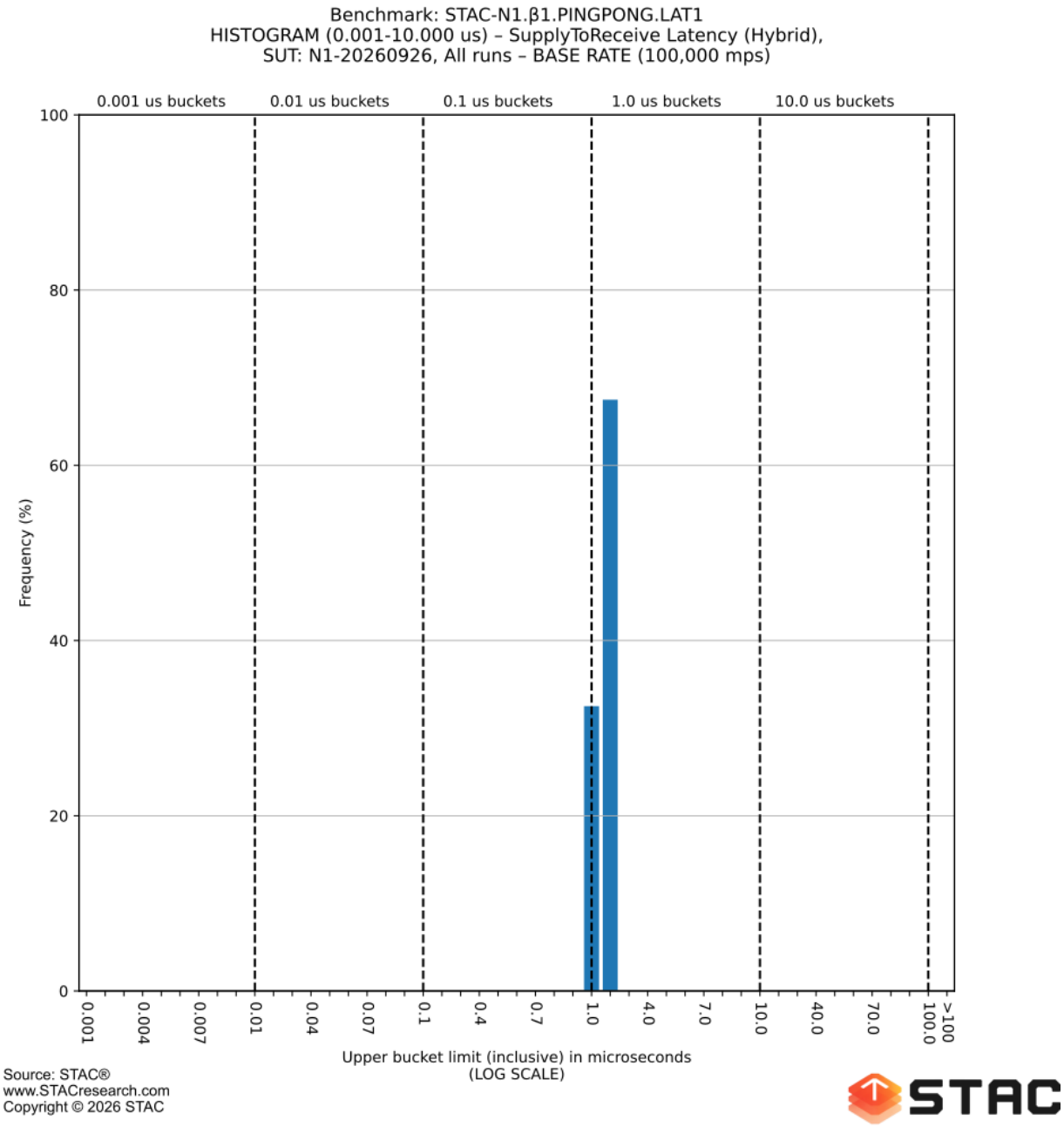
STAC-N1. β1.PINGPONG.LAT1
SUT: N1-20260926

Latency statistics (us)

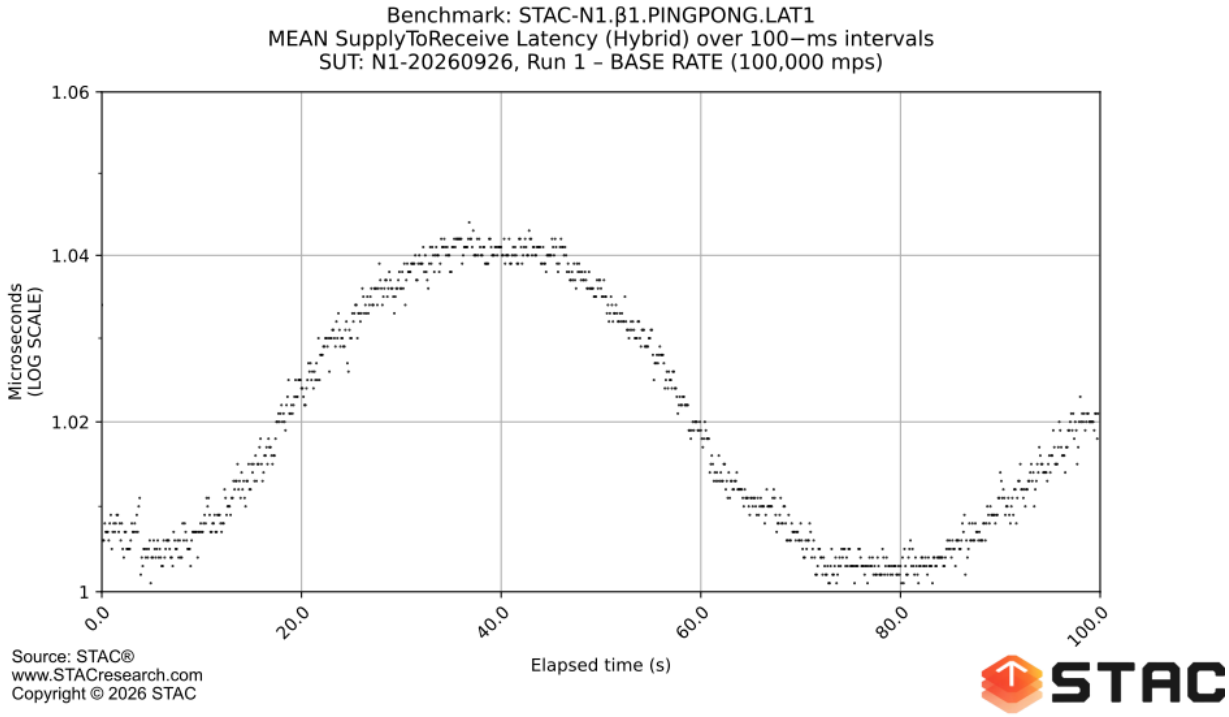
Rate (msgs/sec)	Mean	Med	Std Dev	Min	Max	P99
100,000	1.03	1.02	0.06	0.91	4.00	1.18

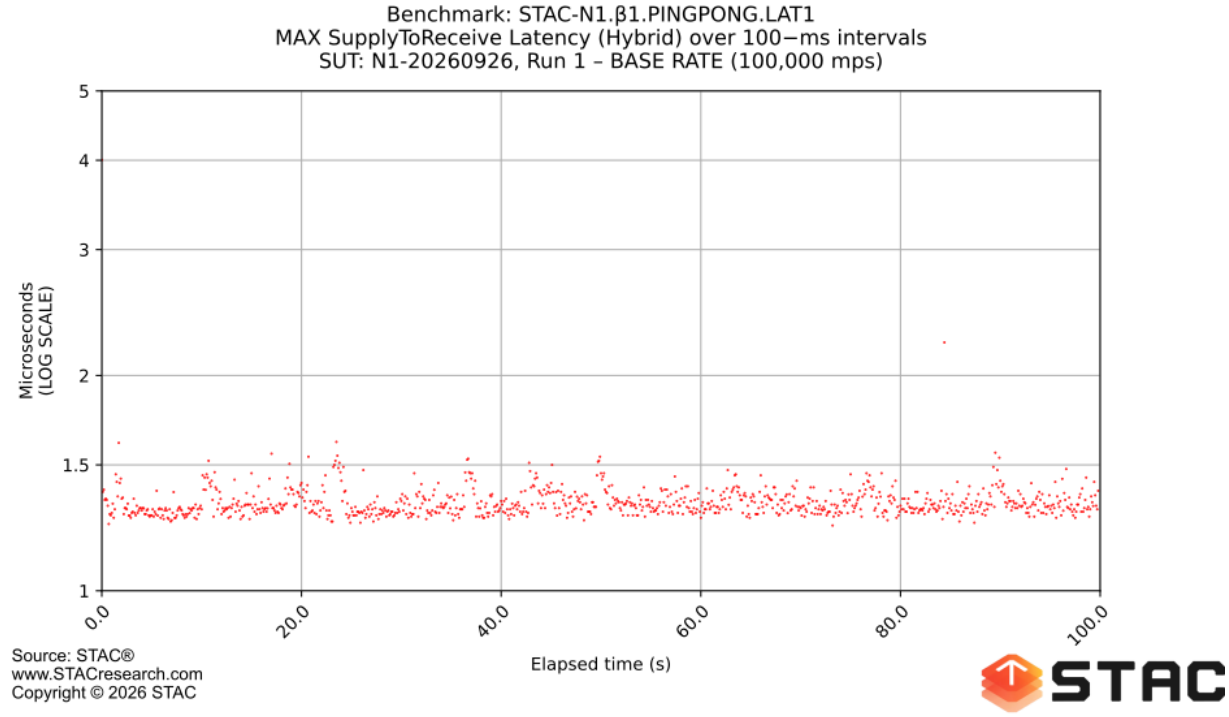
Source: STAC
www.STACresearch.com
Copyright © 2026 STAC




Benchmark: STAC-N1.β1.PINGPONG.LAT1
HISTOGRAM (0.001-10.000 us) – SupplyToReceive Latency (Hybrid),
SUT: N1-20260926, All runs – BASE RATE (100,000 mps)
0.001 us buckets
0.01 us buckets
0.1 us buckets
1.0 us buckets
10.0 us buckets
Frequency (%)
0
20
40
60
80
100
0.001
0.004
0.007
0.01
0.04
0.07
0.1
0.4
0.7
1.0
4.0
7.0
10.0
40.0
70.0
>100
100.0
Upper bucket limit (inclusive) in microseconds
(LOG SCALE)
Source: STAC®
www.STACresearch.com
Copyright © 2026 STAC

Example stats for BASE RATE runs in test sequence PINGPONG





Highest successful rate

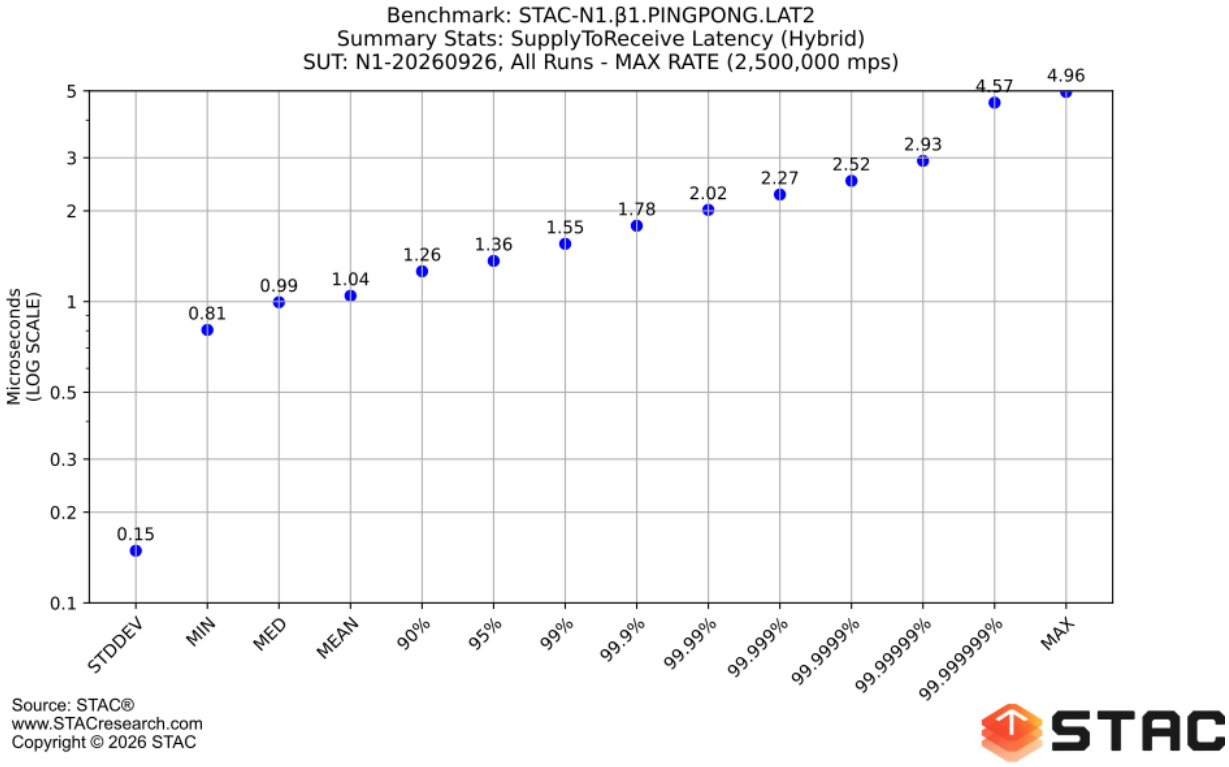
Summary stats for HIGHEST SUCCESSFUL RATE runs in test sequence PINGPONG

STAC-N1. β1.PINGPONG.LAT2
SUT: N1-20260926

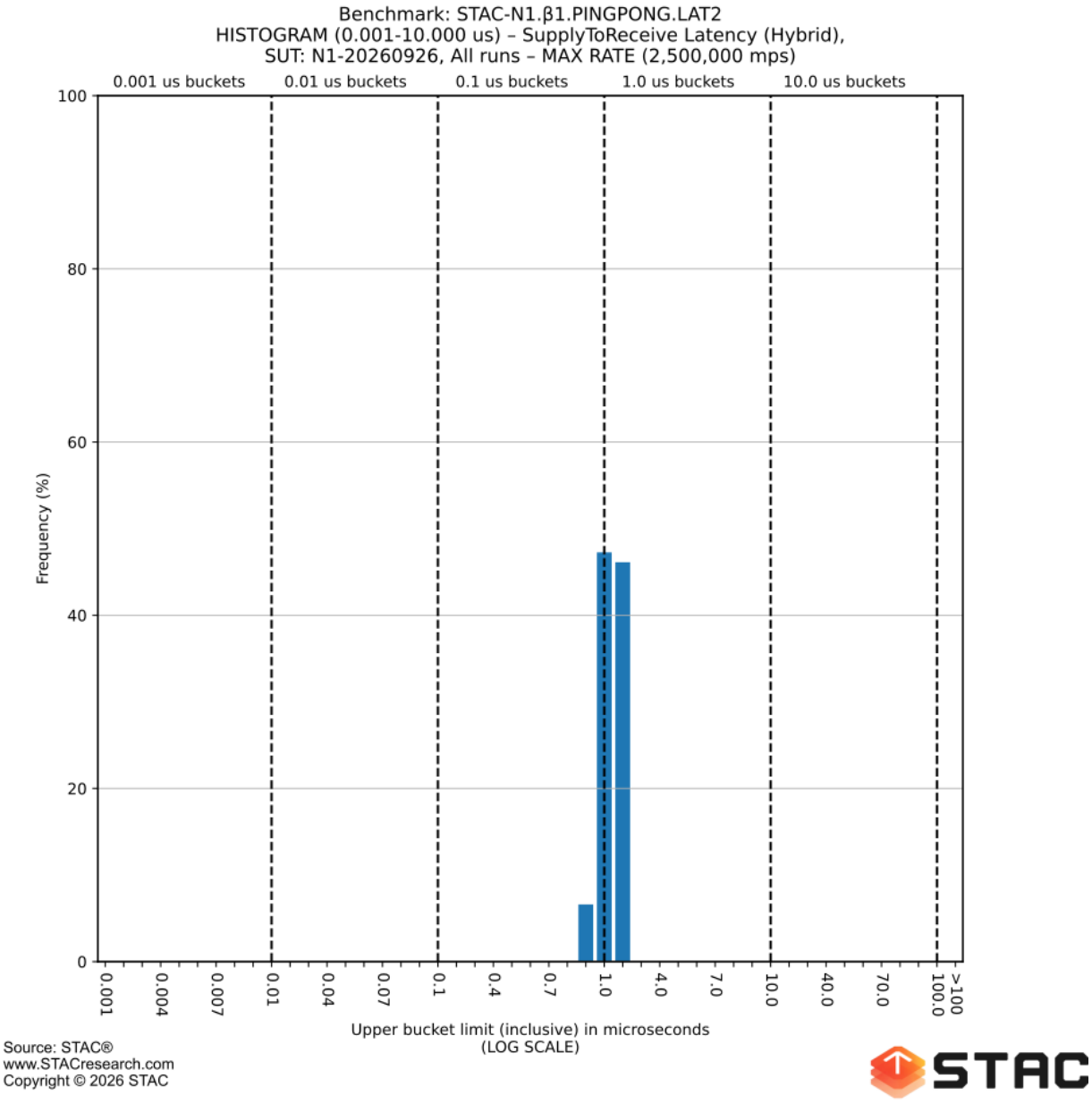
Latency statistics (us)

Rate (msgs/sec)	Mean	Med	Std Dev	Min	Max	P99
2,500,000	1.04	0.99	0.15	0.81	4.96	1.55

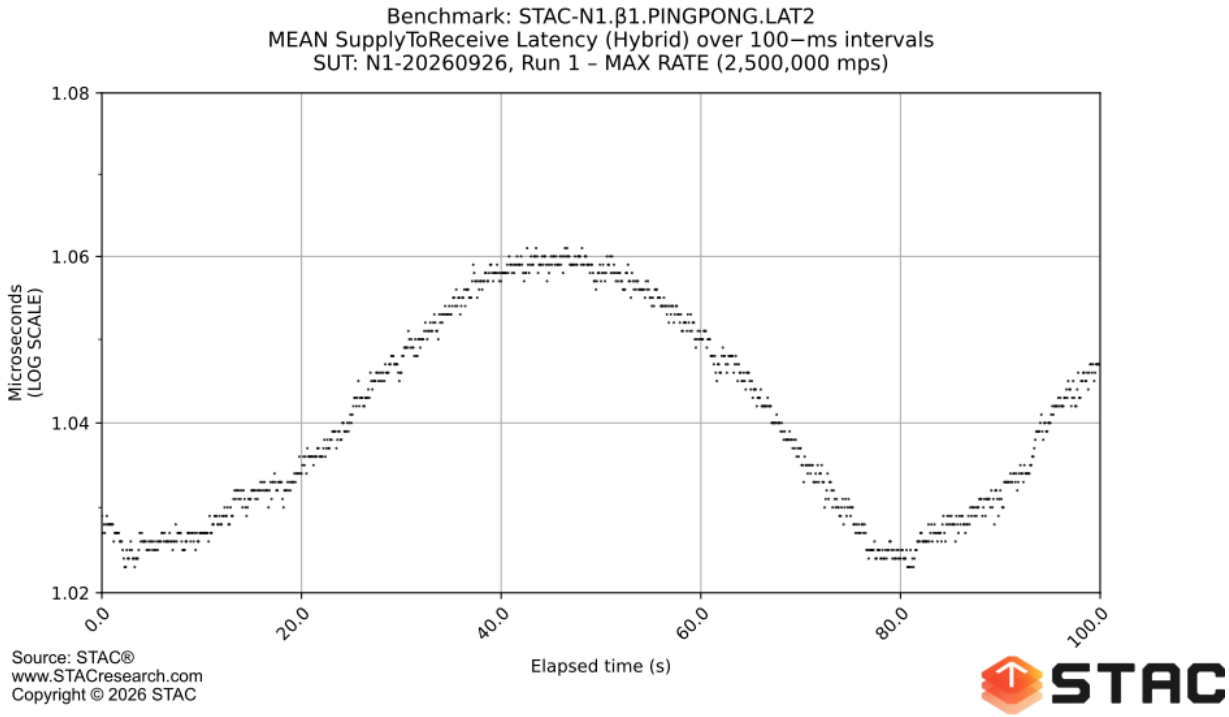
Source: STAC
www.STACresearch.com
Copyright © 2026 STAC

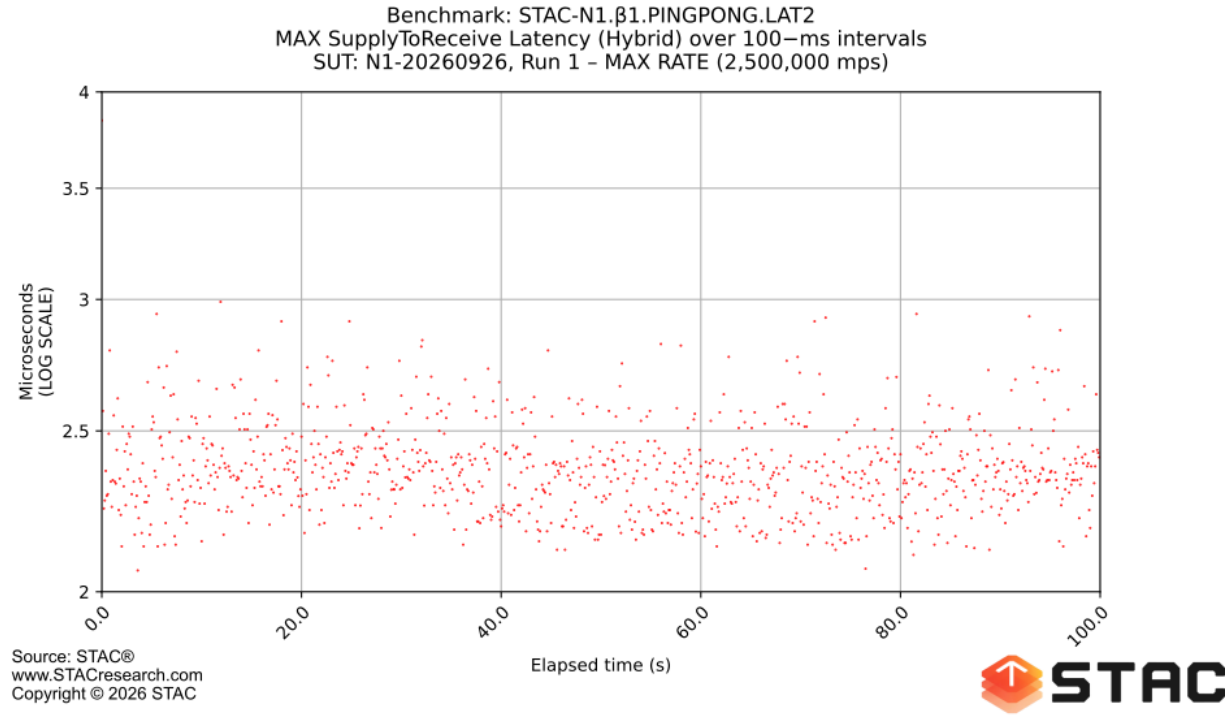


Source: STAC®
www.STACresearch.com
Copyright © 2026 STAC



Example stats for HIGHEST SUCCESFUL RATE runs in test sequence PINGPONG


Benchmark: STAC-N1.β1.PINGPONG.LAT2
MEAN SupplyToReceive Latency (Hybrid) over 100−ms intervals
SUT: N1-20260926, Run 1 – MAX RATE (2,500,000 mps)
Microseconds (LOG SCALE)
1.08
1.06
1.04
1.02
0.0
20.0
40.0
60.0
80.0
100.0
Elapsed time (s)
Source: STAC®
www.STACresearch.com
Copyright © 2026 STAC
STAC


Benchmark: STAC-N1.β1.PINGPONG.LAT2
MAX SupplyToReceive Latency (Hybrid) over 100−ms intervals
SUT: N1-20260926, Run 1 – MAX RATE (2,500,000 mps)
Microseconds (LOG SCALE)
4
3.5
3
2.5
2
0.0
20.0
40.0
60.0
80.0
100.0
Elapsed time (s)
Source: STAC®
www.STACresearch.com
Copyright © 2026 STAC
STAC