STAC-N1 Overview

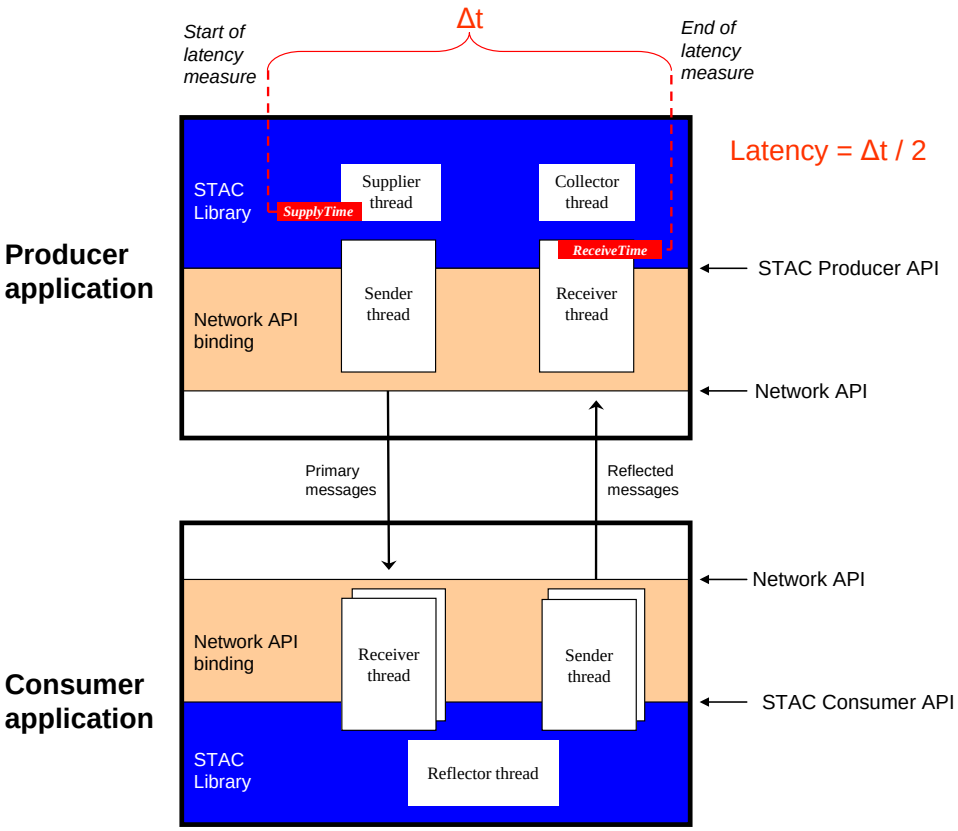
Updated 14 October 2017

STAC-N1 enables direct tests of network stacks using workloads common in financial trading without requiring intervening application software such as middleware or exchange feed handlers. The key requirements informing its development 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 detailed latency analysis
- provide statistics on throughput, CPU, and memory
- automate tests and analysis as much as possible.

For details and the latest STAC-N1 results, see www.STACresearch.com/nio.

How it works



The core of the test harness is the STAC-N Library, a derivative of the STAC-M2 Library (which is used for tests of messaging middleware). Like STAC-M2, this harness uses a "reflection" methodology for round-trip time-stamping, illustrated in the diagram above. 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. For the single test sequence defined so far, the reflection ratio is100%; that is, every message from the Producer to Consumer results in a message from Consumer to Producer. The STAC Library supplies messages to the Producer Adapter as pointers to C structs. Each struct contains a pointer to a variable length key, a pointer to a variable length payload, and header information such as a sequence number and messages type. (Note that the pointers to variable length data means the message is in non-contiguous memory.) Key plus header totals 24 to 32 bytes. The default is 232-byte payloads modeled on observed output of US equities order-book feed handlers deployed in the field.

Many test cases are potentially in scope for STAC-N1. The first sequence defined is simple: classic “ping pong” tests. This sequence involves a single Producer and Consumer. The STAC-N Library supplies messages at a steady-state rate. Starting with the configured Base Message Rate (default: 100K mps), the harness executes a pair of test runs at the given rate, then moves the rate one Step Size higher (default: 50K mps) and repeats the process untl there is a failure (see Max Message Rate, below). Once all rates have been tested, the harness automatically generates a detailed analysis report and CSV files to allow further analysis in Excel or other tools.

Metrics

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

- **SupplyToReceive Latency (Hybrid) – (LAT1, LAT2).** Latency measured from the 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 1). 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).
- **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 50K 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 350K 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 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.

- **Consumer CPU (CPU1) and Producer CPU (CPU2).** Utilization statistics gathered once per second for each core by sar. MEAN is computed by summing the utilizations across all Active Cores at each second, then computing the mean of those sums over the test run. MAX is computed by summing the utilizations across all Active Cores at each second, then computing the max of those sums over the test run. “Active Cores” are defined as cores with mean utilization greater than 5% over the test run, excluding cores dedicated to STAC Library threads. A core that was shielded for use by a SUT process should be treated as 100% utilized. Affinity settings do not affect how utilization is calculated.

Network API bindings

Testing a given network API requires a binding to the STAC-N Library. STAC currently provides four bindings:

- Sockets (UDP and TCP)
- RMDA
- RDS
- Lightfleet

The design goal is for all bindings to use the same set of use-case assumptions but be written optimally for its respective API. Each binding was developed by or in close consultation with vendors with a vested interest in making sure the binding was optimized for the API in question. The bindings and their source code are available to qualified members of the STAC Benchmark Council. Consult STAC for the status of each binding.