STAC-T0 Overview

14 October 2017

STAC-T0 evaluates the tick-to-trade network-I/O latency of any trading platform extremely accurately, no matter what sort of hardware and software make up the "stack under test" (SUT).[1] For example, a SUT might consist of a thin layer of software that implements the rudimentary business logic and uses a kernel bypass library with a high-performance NIC. Or a SUT might be an FPGA card with particular set of IP cores for networking functions (PHY, TOE, etc.).

STAC-T0 treats the "stack under test" (SUT) as a black box, interacting with it solely via network packets, which it timestamps in hardware. The benchmarks measure the time between transmission of simulated UDP market data to the SUT and receipt of simulated TCP orders from it, without the SUT performing any trading logic or market-specific protocol handling. This mirrors the most common network protocols to and from exchanges today and measures the fundamental I/O latency of any trading platform (i.e., latency that cannot be squeezed out by optimizing business logic).

How does STAC-T0 differ from other STAC Benchmarks?

The STAC Benchmark Council has developed several benchmark suites that focus on latency. STAC-T0 is closely related to two: STAC-N1 and STAC-T1. These are still very important benchmarks, but we believe that STAC-T0 fills a void.

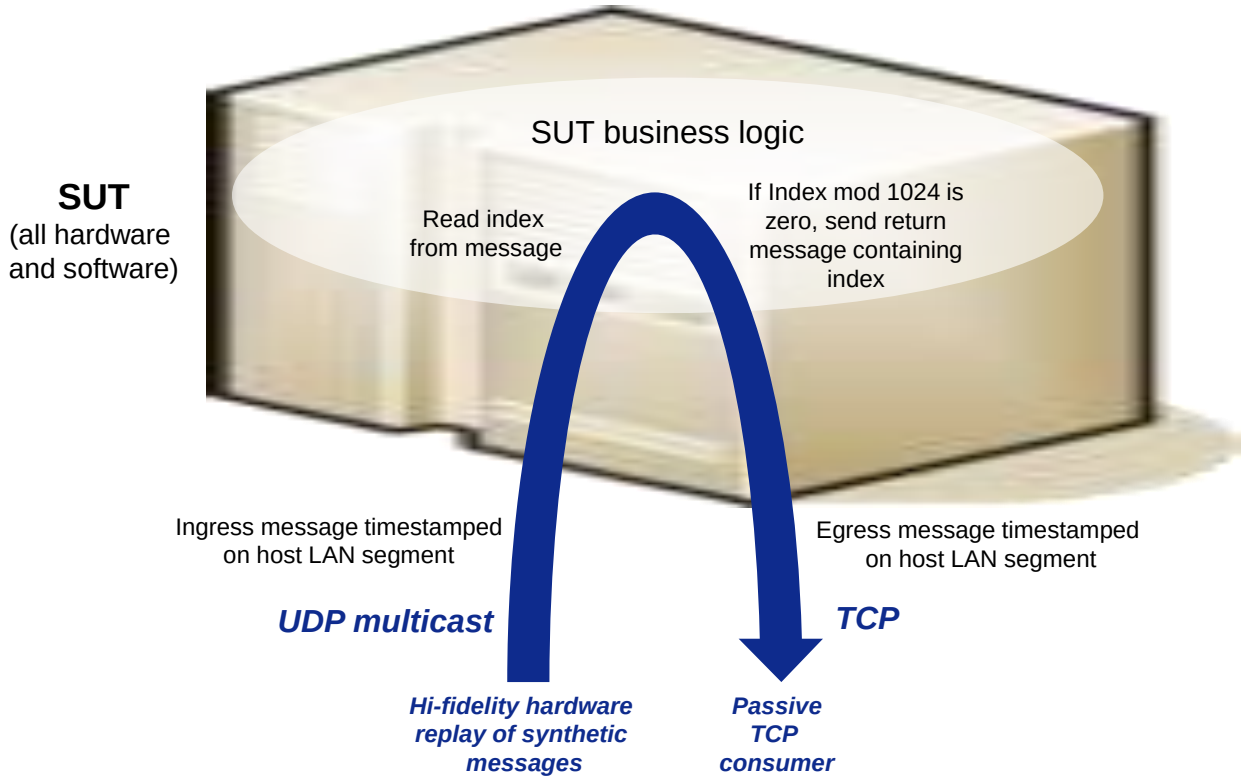
STAC-N1 uses software timestamps for round-trip measurement of network-message latency. STAC-N1 is a convenient toolset because it does not require any equipment for wire capture or hardware-based replay. It is the best option when measuring latencies in microseconds. However, software timestamps are subject to jitter that can represent considerable error when measuring latencies in tens or hundreds of nanoseconds. STAC-N1 also requires the SUT to run STAC software, which means that STAC-N1 can't be used to measure FPGA-based solutions (or even unsupported software environments). By contrast, STAC-T0 uses highly accurate wire timestamps and does not require STAC software to be deployed on the SUT.

STAC-T1 is a tick-to-trade benchmark that sends recorded exchange data to the SUT and accepts order messages via a real exchange protocol. Like STAC-T0, STAC-T1 uses wire timestamping and treats the SUT as a black box. However, STAC-T1 includes market-specific logic (market data decode, order generation) and product functionality (e.g., handling ACKs and fills) that obscures the performance of underlying networking. Requiring market logic makes STAC-T1 a more complete workload (basically everything but the trading algorithm) but also makes it market specific. It also presents a logistical obstacle to benchmarking interesting technology platforms that do not already have business logic for the market in question.

[1] STAC-T0 is a proposed set of benchmark specifications developed in consultation with trading firms in the STAC Benchmark Council. Feedback from members and non-members is welcome.

How it works

The diagram below illustrates the basics of STAC-T1. Recorded packet streams are replayed via hardware to the SUT. Business logic in the SUT reads an index from each packet and performs a simple comparison. If the index meets the criterion, the SUT sends a TCP message containing the index to a passive TCP listener. Ingress and egress messages are timestamped on the wire and captured for later analysis.



Each packet contains an “index”, which is a uint64 that stands in for actionable information in real market data, such as price, side, and size. In a given message structure, the location of the index is fixed. Index values are constructed in a way that makes them impossible for the SUT to predict and that requires the SUT to read the entire index value. The SUT logic extracts the index from each message, and if the index is a multiple of 1024 (decimal), the SUT sends out a 68-byte TCP frame containing the index, to simulate an order message (or other transactional message, such as a cancel). The 68 bytes of the frame includes all headers and the FCS.

STAC-T0 has two UDP data streams, each containing packets of a constant size. Message structure A results in 507-byte frames (including headers and checksums), while message structure B results in 68-byte frames. We play these streams to the SUT one at a time, and we play each stream at three rates:

- LOWRATE, a very low bandwidth that results in one packet per millisecond for each frame size. This is designed to see how systems behave when they are mostly idle.
- MEDRATE, a moderate bandwidth of 1Gbps. This rate should be sustainable by all SUTs, enabling apples-to-apples comparisons in the future.
- HIGHRATE, a higher rate typically at the edge of the SUT’s capability. This is to see how systems behave when they are very busy. Typically the vendor chooses HIGHRATE based on the point beyond which latency may no longer be acceptable to users or messages could be lost.

Measurements

The STAC-T0 test harness software records several timestamps:

- t_{FI} (first in) is time of the first bit of the ingress frame
- t_{IND} (index) is time of the last bit of the index
- t_{LI} (last in) is time of the last bit of the ingress frame
- t_{FO} (first out) is the time of the first bit of the egress packet
- t_{LO} (last out) is the time of the last bit of the egress packet

Two of these timestamps are observations reported by the capture equipment: a single timestamp on the ingress frame and a single timestamp on the egress frame. Different capture equipment timestamp different bits within the frame (some the first bit, some the last bit, some a bit in between).

The rest of the timestamps are inferred from one of these two timestamps using the theoretical serialization delay. For example, if we have the timestamp T on the first bit of a given 507-byte packet on a 10Gbps network (operating at 10.3125 billion bits per second), then the timestamp of the last bit of that packet will be T + (507* 8 – 1) bits / 10.3125 billion bits per second = T + 390 nanoseconds.[2]

An advantage of STAC-T0 replay files over recorded market data is that it is trivial to determine the exact moment nanosecond that all of the information necessary to initiate a “trade” was delivered to the SUT. That time is t_{IND}.

This allows us to compute a latency metric that is unconventional but ultimately pivotal:

$$t_{FO} - t_{IND}$$

This is how long it takes the SUT to start transmitting an order after it has received all the necessary information to evaluate whether to send an order. We call this “actionable latency”. In other words, this is the latency that is under the control of the SUT, if we do not consider cables to be part of the SUT.

STAC-T0 also reports more traditional latency metrics, such as FILO (first in to last out, the end-to-end latency reported in STAC-T1), LIFO (last in to first out, a metric of particular interest when the business logic is unable to begin processing until it has received the entire packet), and FIFO (first in to first out). Each of these latency metrics includes some serialization delay, which we consider to be outside the control of the SUT. However, given their common use, one or more may be meaningful comparison point for customers.

To summarize then, the STAC-T0 latency metrics are:

- STAC-T0.ACTIONABLE.LAT: $t_{FO} - t_{IND}$
- STAC-T0.FILO.LAT: $t_{LO} - t_{FI}$
- STAC-T0.LIFO.LAT: $t_{FO} - t_{LI}$
- STAC-T0.FIFO.LAT: $t_{FO} - t_{FI}$

[2] While the Ethernet specification allows equipment to vary its frequency by +/- 100 parts per million, that variance has no nanosecond-level impact on serialization delay of a few hundred bits.