WHITEPAPER

February 2011

Informatica, HP, and Mellanox/Voltaire Benchmark Report

Ultra Messaging TM accelerated across three supported interconnects

1.0 Executive summary

The securities trading market is experiencing rapid growth in volume and complexity with a greater reliance on trading software, which is supported by sophisticated algorithms. As this market grows, so do the trading volumes, bringing existing IT infrastructure systems to their limits.

Trading technology solutions must not only deliver low-latency despite high volumes, but also sustain performance through market data traffic spikes while minimizing latency jitter and unpredictable messaging outliers.

To address this challenge, three market leaders have come together to provide a field-proven infrastructure solution that combines best of breed enterprise messaging middleware, trading transport and server/network infrastructure. The solution slashes end-to-end trading latency by analyzing each layer of the infrastructure— server, storage, server stack and middleware, as well as switching and bridging devices—and optimizing each part for minimum latency and maximum throughput, while keeping an eye on overall cost.

Informatica's Ultra Messaging (formerly known as 29West's Ultra Messaging) has emerged as the standard in next generation messaging software and delivers an unbending commitment to offering the lowest possible latency across the widest possible range of messaging use cases.

HP's reliable ProLiant servers offer the fastest Xeon processors available, with tuning options specifically designed for low jitter messaging. HP incorporates Mellanox's ultra low-latency ConnectX-2 Virtual Protocol Interconnect (VPI) I/O adapters for InfiniBand and/or 10 Gigabit Ethernet in these servers for linkage to external InfiniBand and 10 Gigabit Ethernet switches from Mellanox/Voltaire. Additionally, for transport acceleration, a selection of Mellanox's VMA or VRT is used for optimal OS bypass RDMA or socket acceleration. A key decision facing trading firms and exchanges is the selection of the optimum interconnect fabric and method of accelerating their applications across this interconnect to achieve competitive advantage.

Mellanox's ConnectX-2 VPI adapters provide the highest performance and the most flexible interconnect solution for Enterprise Data Centers. They support 10, 20 and 40Gb/s InfiniBand as well as various Ethernet flavors for Rack, Scale-Out and Blade servers over KR, KX4 or SFP+ connectors at 10Gb/s speed.

Voltaire Message Acceleration (VMA) is a dynamically-linked user-space Linux library for accelerating multicast traffic. Network processing is offloaded from the server's CPU by passing the traffic directly from the user-space application to the network adapter bypassing the kernel and IP stack and thus minimizing context switches, buffer copies and interrupts resulting in extremely low-latency.

Voltaire RDMA Transport (VRT) is an RDMA-based publisher subscriber messaging protocol which enables extremely high volume messaging with low-latency and low jitter networking.

This report aims to assist end-user architecture decisions by quantifying the benefits of this integrated solution in delivering application-to-application messaging at:

- **Accelerated UDP Multicast**
  - 10 Gigabit Ethernet: Average latency as low as 6.1 microseconds
  - InfiniBand: Average latency as low as 5.4 microseconds
- **RDMA**
  - 10 Gigabit Ethernet: Average latency as low as 8.3 microseconds
  - InfiniBand: Average latency as low as 5.3 microseconds
- **All with very low standard deviation, and 99.9th percentile values very close to the average.**

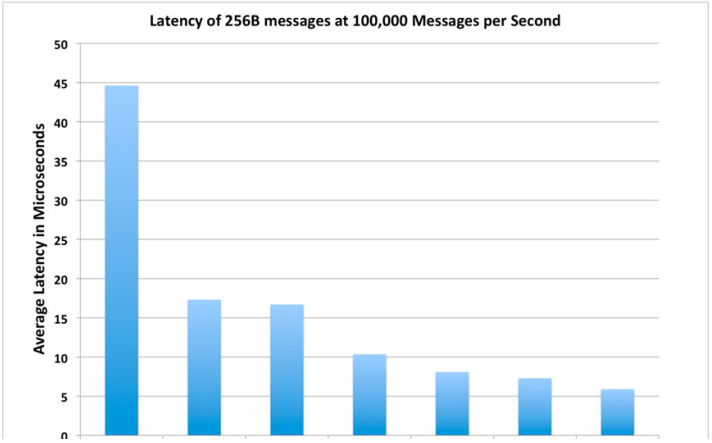
1.1 Benchmark Overview and Baseline

The objective of this benchmark was to use a consistent, highly optimized server platform and the Informatica test suite to measure comparative approaches to application acceleration over three fabrics as shown in this table:

API:	Unaccelerated	Transparent (Preload Library)	RDMA rewrite
Interconnect:	UDP thru kernel	VMA kernel bypass	VRT
1GbE	measured	X	X
10 GbE	measured	measured	measured
InfiniBand	measured	measured	measured

Mean latencies for these seven measurements are shown in the graph below.

Reference Transport Latency



The rest of this whitepaper will examine the specific performance characteristics of the two fastest transports tested: the VMA UDP-Multicast kernel bypass transport and the VRT-based RDMA transport for both 10 Gigabit Ethernet and 40Gb/s InfiniBand.

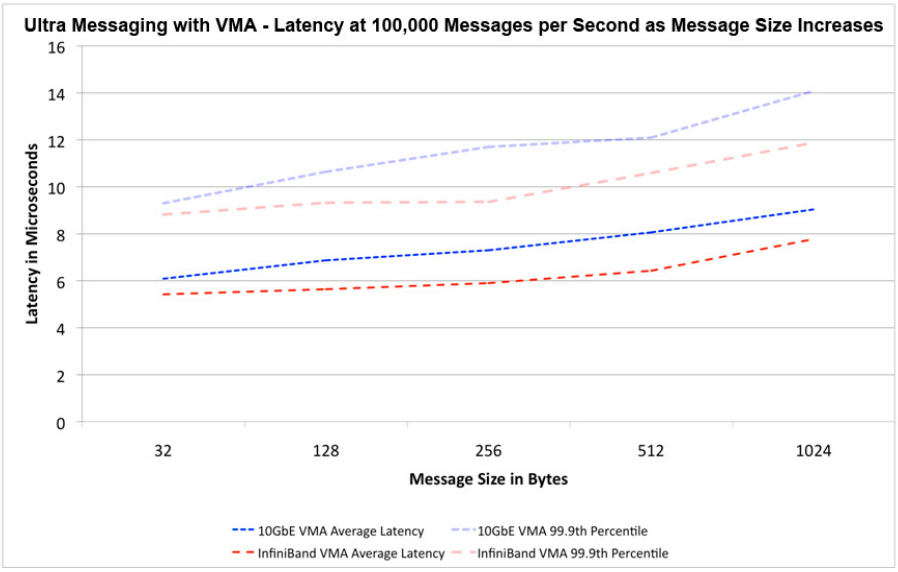
This graph, however, provides a reference to highlight the magnitude of performance improvement that is gained over traditional network and transport choices by moving to a technology stack that supports advanced kernel bypass techniques on fast underlying networks. All data points graphed here are average latencies, taken at a rate of 100,000 messages per second, using 256-Byte individual messages (for a calculated payload throughput of 204.8 Mbps). While 256-Byte messages were chosen because we feel they represent a good real-world average for high-rate financial messages in our customer base, results for other message sizes are also reported.

As you can see, moving from the high serialization latency and low bandwidth of standard Gigabit Ethernet to 10GbE and InfiniBand provides an immediate drop in latency of over 60%. And again, once the operating system kernel is removed from the message path, latencies again drop precipitously.

Now that we've established what the fastest technology stacks are, we will further explore their exact performance characteristics, and document the methodology we used in their testing.

1.2 Latency Versus Message Size

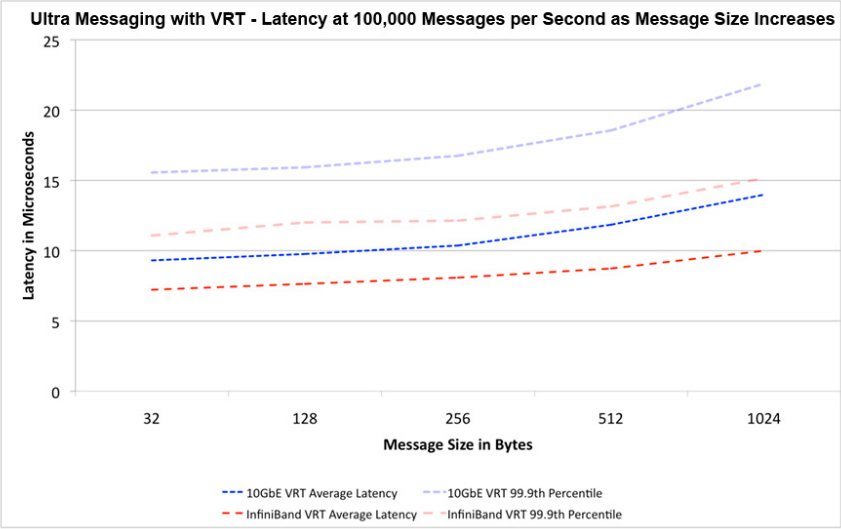
Since it takes a CPU longer to allocate more memory, and in turn it takes networks a long time to serialize all of the data bit-by-bit onto the wire, it is expected for increasing message sizes to have some impact on latency, but not necessarily a large one. The following graph was made with both average and 99.9th percentile data from our test runs with VMA (UD Multicast kernel bypass) on both 10 Gigabit Ethernet and 40Gb/s InfiniBand.



You'll notice that the InfiniBand transport exhibits the lowest latency, both average and 99.9th percentile, while the shape of the curves for both 10GbE and InfiniBand are for the most part identical; there is a smooth, linear progression spanning no more than 3 additional microseconds between the 32-Byte and 1024-Byte tests.

99.9th percentile values (meaning out of every 1,000 samples, only one value was higher than the 99.9th percentile number) shown here are consistently low, measuring only between 3 and 5 microseconds higher than the average.

Similar characteristics are exhibited in the VRT transport data, shown below.

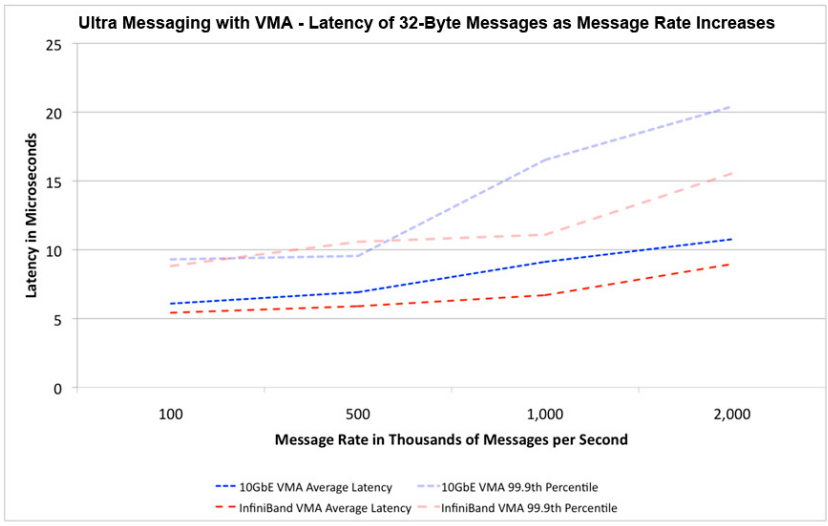


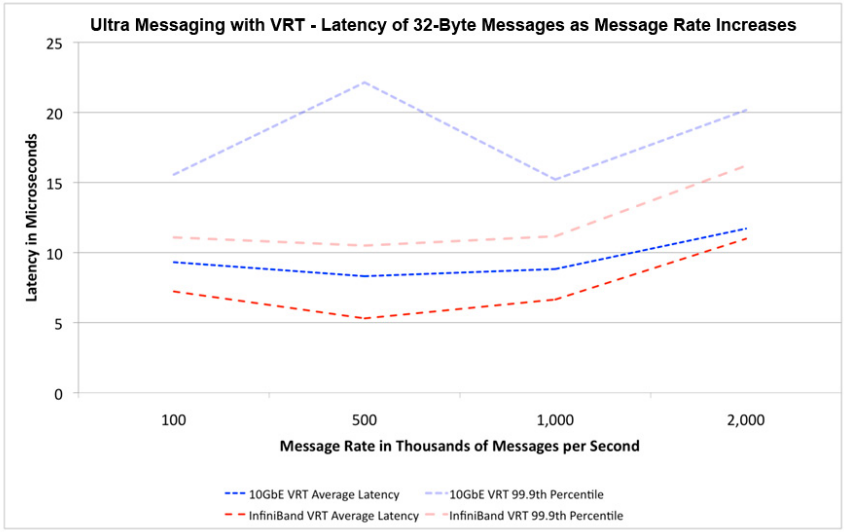
If anything, the progression in the VRT (RDMA transport) data is simply smoother than in the datagram-based data. The latency values themselves are slightly higher than the VMA numbers at 100K msg/sec, but as we'll shortly see in the expanded message-rate testing, this is not always the case.

1.3 Latency Versus Message Rate

We expect message rates to be independent of messaging latency to an extent. If the sending CPU is able to generate messages sufficiently fast, and the receiving CPU is fast enough to process every incoming message at the moment it is notified of its existence, then the measured latency consists, essentially, of only the time it takes to send, propagate and receive an individual message.

If, however, the desired message rate causes either CPU (usually the receiving CPU) to be overburdened, additional latency will be added in order to make higher rates possible. Usually, in some form or another, this means the addition of queuing latency. In this case, we increased the amount of application-level batching, in which multiple logical messages are used to create a single network-level message, to decrease the amount of work both CPUs need to do. This allows the systems to achieve higher message rates, while marginally increasing average latency. Results for 32-Byte messages from 100,000 to 2,000,000 messages per second are shown below. 32-Byte messages were chosen because the widest range of message rates were achieved with this smaller message size.





Both the VMA and VRT results exhibit similar distributions, with a couple of notable dissimilarities. The important thing is the fact that the average latency changes very little while the message rate changes by a factor of 20, from 100,000 messages per second all the way through 2,000,000.

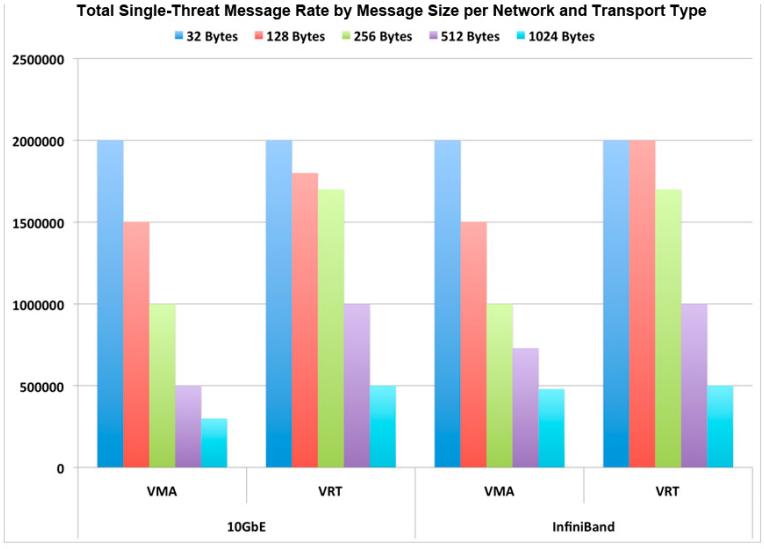
The relatively higher 99.9th percentile values are to be expected with the introduction of the higher levels of batching that were necessary to achieve the higher rates. On average, the latencies increase only slightly, but because of the way that messages are processed in chunks, a relatively higher percentage of latency measurements will be above the average. This is reflected in the standard deviation values as message rates increase as well, and we'll explore those values later in this paper.

1.4 Throughput Results

Per-Transport Single-Threaded Message Rates

In addition to simple sequential ping-pong style latency testing – which gives an idea of latency performance under only a single level of load – our test applications are able to run at higher rates. This enables us to show the effects of batching and buffering on system latency and stability.

In our testing, an interesting trend showed itself when comparing the achievable single-threaded message rates of both the datagram (VMA) and RDMA (VRT) tests. Note that the 2,000,000 messages per second threshold was simply the highest rate we tested, in some cases the 32 Byte tests were able to exceed this value, but data for those rates were not gathered.



In all cases, the VRT-based transport was able to process significantly higher message rates than the VMA-based transport. This was especially evident in the mid-range message sizes, where upwards of 50,000 message-per-second improvements were seen by moving to VRT. This is interesting, because in most cases the latencies themselves are very similar to, if not faster than, VMA.

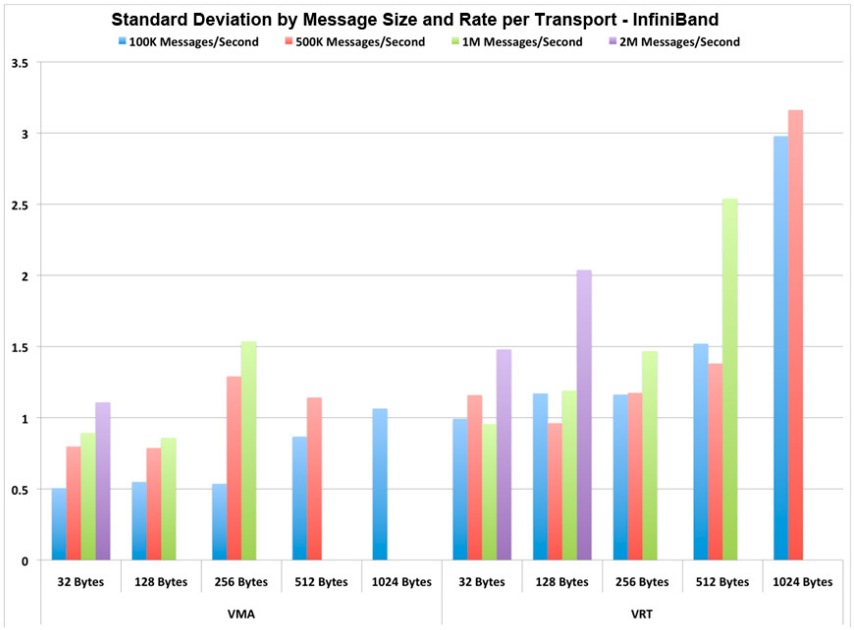
Based on these data, RDMA (VRT) seems to be the best choice for sending high volumes of data point-to-point.

It should be noted that VMA supports UDP-multicast, a distribution mode which allows many multiples of receivers who are interested in the same data to each receive copies of the data with no additional substantive overhead on the sender. In the case of multicast, the aggregate throughput can become greater than a point-to-point solution, depending on the number of participating receivers. However, VRT remains faster on a per-stream basis.

1.5 Standard Deviation (Jitter)

Regardless of the low-latency characteristics of a transport, the industry continues to rightly demand that the predictability of any messaging system should be very high. In terms of statistical analysis, this means that the standard deviation of the collected data must be very low.

A. InfiniBand Results

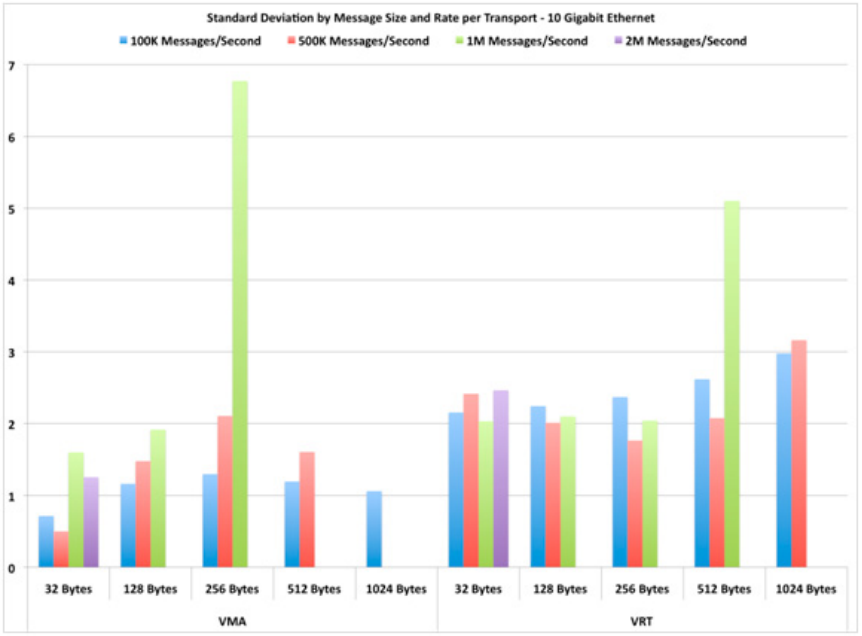


Here we see the standard deviation numbers for all of the tests we did over InfiniBand. The VMA numbers are on the left and the VRT numbers on the right. The first thing to notice is the scale. Regardless of the relative height of these bars, there wasn't a single standard deviation over 3.16 microseconds; these values are incredibly low, resulting in extremely predictable latency.

Broadly, the trends that can be seen are what we expected – very low numbers, with values increasing along with increasing message rates, and to smaller degree, along with increasing message size. Highest deviations are seen when the transport is running as fast as it can. In addition, VMA tends to show lower standard deviation than VRT.

It should be noted that the graph includes blank spots for message size and rate pairs that could not be achieved during testing.

B. 10 Gigabit Ethernet results.



The scale here is slightly higher than it was with the InfiniBand results. Still, there were no deviations above 6.77 microseconds, and that one outlier is significantly higher than the vast majority of values.

It should be noted that the tests in which higher-than-normal standard deviations are seen are tests which are running at the very top edge of the performance curve. It is unremarkable that these tests have higher deviation, since the system resources are being taxed much more heavily than in the other runs.

Aside from that difference, the trends are essentially identical to the InfiniBand results. VMA has slightly lower standard deviation, with values generally increasing along with increasing message sizes and rates.

1.6 Concusion

The joint solution offered by Informatica, HP and Mellanox/Voltaire demonstrates incredibly low latencies and jitter, while operating at very high rates – under 9 microseconds all the way up to 2 million messages per second for 32 Byte messages with both InfiniBand and with 10 Gigabit Ethernet. Not only does this show excellent speed, it also shows the ability of the system to perform at those excellent speeds while under very high loads, and to behave in a predictable fashion throughout the spectrum.

Lower latency across the entire trading infrastructure means more money to financial services firms. Informatica believes that this solution is the fastest, most robust, most efficient offering on the market.

All of this taken together makes this joint solution a demonstrated competitive advantage for our customers throughout the financial industry.

Appendix

Testing Methodology

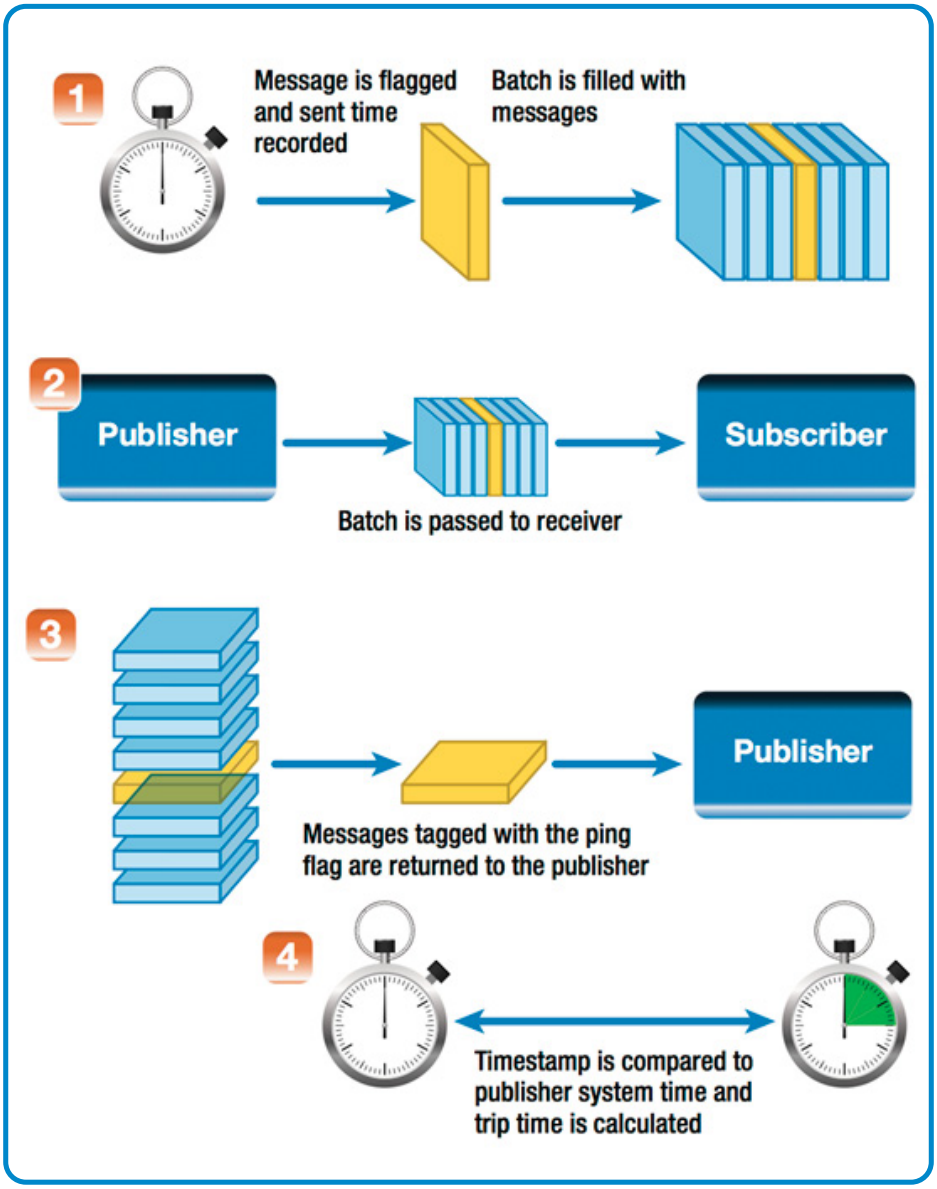
Measurements were performed using the Informatica lbmpubload and lbmsubload applications. The lbmpubload app publishes a stream of application messages at a requested rate and message size. Most of those messages are "load" messages, but periodically, a message is designated a "ping" message, which means that it will be used to measure latency.

As each ping message is sent, a high-resolution clock is sampled. The lbmsubload app subscribes to the message stream. It discards load messages, but for each ping message it "reflects" (re-publishes) the ping back to lbmpubload. The lbmpubload app, in addition to publishing the message stream, also subscribes to the reflected ping messages. When a reflected ping is received, the high-resolution clock is sampled again.

By subtracting the original send time from the reflection receive time, the total round-trip time for the ping message is calculated. This value is divided by two in order to calculate the one-way latency. (This assumes the latency is symmetrical, which is approximate. More accurate one-way measurements could be made with GPS-synchronized clocks.) The algorithm used by lbmpubload to generate messages at a requested rate is:

1. "Outer" loop wakes up from sleep.
2. High-resolution time is sampled and compared to the sample taken at program initialization. Using the requested message rate, the number of messages "desired" by this time is calculated.
3. The actual number of messages sent so far is subtracted from the "desired" number.
4. This represents the number of messages that must be sent to get caught up, and represents the application-level batch. At very low message rates, this will always be 1. At very high rates, this can be many hundreds.
5. The "inner" loop executes, sending these messages. At the end of the loop, the batch is flushed (this is the "intelligent" part of the batching).
6. A number of milliseconds to sleep is calculated based on what time the next message should be sent. At low message rates, this can be tens of milliseconds. At very high rates, this will often be 0. If the value is greater than 1, the application sleeps, thus completing the "outer" loop.

Lower-rate tests were done without any batching (the combining of multiple logical messages into a single datagram to increase total achievable throughput). Higher rates were done with appropriate levels of batching to ensure low latencies while running at high message rates. See the image below for a visualization of this process. When batching is selected care is taken to ensure that the ping message is uniformly distributed across both the application-level batches and the middleware-level batches. This eliminates the averaging errors that would result from aliasing between the batching and the ping periods.



System Configuration

VRT version: libvrt-1.3.10-2437

VMA version: libvma-4.5.4-0

Two HP ProLiant® DL380 G7 servers, each with two Intel ® Xeon ® X5680 processors – 6 cores @ 3.33Ghz (each) and 48 GB of 1333MHz memory

BIOS Release Date 09/13/2010

NIC Interfaces: Mellanox ConnectX ®-2 InfiniBand/10GbE PCIe Adapters:

HP IB 4X QDR CX-2 PCI-e G2 Dual-port HCA part #592520-B21

Mellanox Technologies MT26428 firmware 2.7.0

HP 10GbE PCI-e G2 Dual Port NIC part #516937-B21

Mellanox Technologies MT26448 firmware 2.7.700

External Switches

Voltaire 4036E InfiniBand switch (HP part #632220-B21)

Volaire 6024 10GbE switch (HP 3PO)

Interrupt Coalescing

1GbE:

Adaptive-rx: off

Adaptive-tx: off

Rx-usecs: 1

All other values: 0

10GbE:

Adaptive-rx: off

Adaptive-tx: off

Rx-usecs: 1

Rx-frames: 1

Rx-usecs-high: 1

All other values: 0

BIOS settings:

HP_Power_Regulator = HP_Static_High_Performance_Mode

HP_Power_Profile = Maximum_Performance

Intel_Minimum_Processor_Idle_Power_State = No_C-States

Memory_Interleaving = Full_Interleaving

Dynamic_Power_Savings_Mode_Response = Fast

Memory_Speed_with_2_DIMMs_per_Channel = 1333MHz_Maximum

PowerMonitoring = Disabled

DisableMemoryPrefailureNotification = Yes

Node_Interleaving = Disabled

Intel_Hyperthreading = Disabled

Hyperthreading = Disabled

Intel_Processor_Turbo_Mode = Disabled

HW_Prefetch = Enabled

Adjacent_Sector_Prefetch = Enabled

Ultra-Messaging Configuration

Transport Settings

source transport: lbtrm or lbtrdma (depending on test)

context ud_acceleration: 1 (for VMA transport)

context fd_management_type: poll (set for all tests)

context resolver_multicast_interface: (this option was used to differentiate NICs or HCAs based on desired network)

source implicit_batching_minimum_length: 1 through 1472 (depending on test)

context transport_lbtrm_datagram_max_size: 1472

context transport_lbtrm_receiver_socket_buffer: 8388608

context transport_lbtrm_data_rate_limit: 10000000000

Topic Resolution Settings

source resolver_advertisement_minimum_initial_interval: 0

source resolver_advertisement_sustain_interval: 0

receiver resolver_query_minimum_initial_interval 0

receiver resolution_number_of_sources_query_threshold 1

wildcard_receiver resolver_query_minimum_interval 0

Full Results

Shown here are the full results of all test runs. All values are latency in microseconds.

InfiniBand with VMA						
Message Size/Rate	Min	Avg	Max	Median	99.9th %	Std. Dev
32B/100K	4.82	5.42	13.62	5.21	8.82	0.51
32B/500K	4.84	5.90	25.06	5.67	10.58	0.80
32B/1M	5.24	6.70	21.13	6.53	11.09	0.89
32B/2M	6.69	8.96	16.17	8.73	15.56	1.11
	Min	Avg	Max	Median	99.9th %	Std. Dev
128B/100K	4.92	5.64	15.60	5.43	9.32	0.55
128B/500K	5.11	6.24	20.55	6.02	10.07	0.79
128B/1M	5.64	7.40	19.42	7.20	12.31	0.86
128B/1.5M	7.37	9.83	19.49	8.24	18.15	2.92
	CPU-bound after 1.5M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
256B/100K	5.25	5.90	12.92	5.69	9.36	0.54
256B/500K	5.37	6.99	26.68	6.64	18.62	1.29
256B/1M	6.39	7.97	16.11	7.36	15.67	1.54
	CPU-bound after 1M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
512B/100K	5.68	6.43	36.94	6.09	10.59	0.87
512B/500K	5.85	7.87	24.14	7.64	17.37	1.14
512B/730K	6.59	7.90	22.13	7.65	16.57	1.04
	CPU-bound after 730K msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
1024B/100K	6.58	7.78	15.05	7.46	11.88	1.06
1024B/480K	7.05	8.78	19.62	8.61	13.76	1.04
	CPU-bound after 480K msgs/sec					

Full Results Continued

InfiniBand with VMA						
Message Size/Rate	Min	Avg	Max	Median	99.9th %	Std. Dev
32B/100K	4.82	5.42	13.62	5.21	8.82	0.51
32B/500K	4.84	5.90	25.06	5.67	10.58	0.80
32B/1M	5.24	6.70	21.13	6.53	11.09	0.89
32B/2M	6.69	8.96	16.17	8.73	15.56	1.11
	Min	Avg	Max	Median	99.9th %	Std. Dev
128B/100K	4.92	5.64	15.60	5.43	9.32	0.55
128B/500K	5.11	6.24	20.55	6.02	10.07	0.79
128B/1M	5.64	7.40	19.42	7.20	12.31	0.86
128B/1.5M	7.37	9.83	19.49	8.24	18.15	2.92
	CPU-bound after 1.5M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
256B/100K	5.25	5.90	12.92	5.69	9.36	0.54
256B/500K	5.37	6.99	26.68	6.64	18.62	1.29
256B/1M	6.39	7.97	16.11	7.36	15.67	1.54
	CPU-bound after 1M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
512B/100K	5.68	6.43	36.94	6.09	10.59	0.87
512B/500K	5.85	7.87	24.14	7.64	17.37	1.14
512B/730K	6.59	7.90	22.13	7.65	16.57	1.04
	CPU-bound after 730K msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
1024B/100K	6.58	7.78	15.05	7.46	11.88	1.06
1024B/480K	7.05	8.78	19.62	8.61	13.76	1.04
	CPU-bound after 480K msgs/sec					

Full Results Continued

InfiniBand with VMA						
Message Size/Rate	Min	Avg	Max	Median	99.9th %	Std. Dev
32B/100K	4.82	5.42	13.62	5.21	8.82	0.51
32B/500K	4.84	5.90	25.06	5.67	10.58	0.80
32B/1M	5.24	6.70	21.13	6.53	11.09	0.89
32B/2M	6.69	8.96	16.17	8.73	15.56	1.11
	Min	Avg	Max	Median	99.9th %	Std. Dev
128B/100K	4.92	5.64	15.60	5.43	9.32	0.55
128B/500K	5.11	6.24	20.55	6.02	10.07	0.79
128B/1M	5.64	7.40	19.42	7.20	12.31	0.86
128B/1.5M	7.37	9.83	19.49	8.24	18.15	2.92
	CPU-bound after 1.5M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
256B/100K	5.25	5.90	12.92	5.69	9.36	0.54
256B/500K	5.37	6.99	26.68	6.64	18.62	1.29
256B/1M	6.39	7.97	16.11	7.36	15.67	1.54
	CPU-bound after 1M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
512B/100K	5.68	6.43	36.94	6.09	10.59	0.87
512B/500K	5.85	7.87	24.14	7.64	17.37	1.14
512B/730K	6.59	7.90	22.13	7.65	16.57	1.04
	CPU-bound after 730K msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
1024B/100K	6.58	7.78	15.05	7.46	11.88	1.06
1024B/480K	7.05	8.78	19.62	8.61	13.76	1.04
	CPU-bound after 480K msgs/sec					

Full Results Continued

InfiniBand with VMA						
Message Size/Rate	Min	Avg	Max	Median	99.9th %	Std. Dev
32B/100K	4.82	5.42	13.62	5.21	8.82	0.51
32B/500K	4.84	5.90	25.06	5.67	10.58	0.80
32B/1M	5.24	6.70	21.13	6.53	11.09	0.89
32B/2M	6.69	8.96	16.17	8.73	15.56	1.11
	Min	Avg	Max	Median	99.9th %	Std. Dev
128B/100K	4.92	5.64	15.60	5.43	9.32	0.55
128B/500K	5.11	6.24	20.55	6.02	10.07	0.79
128B/1M	5.64	7.40	19.42	7.20	12.31	0.86
128B/1.5M	7.37	9.83	19.49	8.24	18.15	2.92
	CPU-bound after 1.5M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
256B/100K	5.25	5.90	12.92	5.69	9.36	0.54
256B/500K	5.37	6.99	26.68	6.64	18.62	1.29
256B/1M	6.39	7.97	16.11	7.36	15.67	1.54
	CPU-bound after 1M msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
512B/100K	5.68	6.43	36.94	6.09	10.59	0.87
512B/500K	5.85	7.87	24.14	7.64	17.37	1.14
512B/730K	6.59	7.90	22.13	7.65	16.57	1.04
	CPU-bound after 730K msgs/sec					
	Min	Avg	Max	Median	99.9th %	Std. Dev
1024B/100K	6.58	7.78	15.05	7.46	11.88	1.06
1024B/480K	7.05	8.78	19.62	8.61	13.76	1.04
	CPU-bound after 480K msgs/sec					

Acknowledgements

The team for this benchmark included:

Greg Lorence, Informatica

Lee Fisher, HP

Chuck Newman, HP

Mike Nikolaiev, HP

Sagi Schlanger, Mellanox/Voltaire

Tzahi Oved, Mellanox/Voltaire

Motti Beck, Mellanox/Voltaire

Aviv Cohen, Mellanox/Voltaire

350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
Tel: 408-970-3400 • Fax: 408-970-3403
www.mellanox.com