Migratory Push Server Benchmarks

Release Version 3.5
October 5, 2010

Copyright Information

Contents

Chapter 1

Overview

This document explains how to benchmark Migratory Push Server by using the following benchmark tools:

- Migratory PushClient-Benchmark – This tool simulates a huge number of concurrent web clients. It is able to:
  - open a configurable number of client connections to a Migratory PushServer instance
  - subscribe to a configurable number of subjects from a configurable set of subjects
  - receive messages in real-time and continuously on the subscribed subjects, and
  - compute the latency for the received messages.
- Migratory PushAgent-Benchmark – This tool simulates your data. It is able to:
  - generate messages of a configurable size
  - publish messages with a configurable frequency from a configurable set of subjects

It is recommended to read *Migratory Push Server Architecture Guide* before reading this document for a better understanding of the concepts and to have a more comprehensive background.

Also please read the following documentations to learn how to install and configure each component involved into the benchmark tests:

- *Migratory PushServer Installation and Configuration Guide*
- *Migratory PushClient-Benchmark Installation and Configuration Guide*
- *Migratory PushAgent-Benchmark Installation and Configuration Guide*

1.1 Migratory PushClient-Benchmark

Migratory PushClient-Benchmark is a Java utility that behaves like Migratory PushClient-JavaScript library (web browser), being able to:

- Connect to Migratory PushServer
- Subscribe to a list of subjects
- Get in real-time and continuously messages for the subscribed subjects

In addition, the following functionalities are available in Migratory PushClient-Benchmark:

- Migratory PushClient-Benchmark is able to open up to one million of concurrent connections to Migratory PushServer from a single instance
- A time-stamp information is extracted from each message used to compute its latency

Migratory PushClient-Benchmark is able to simulate a huge number of concurrent web clients. It is able to open a configurable number of client connections (parameter `Clients`) to a Migratory PushServer instance (parameter `ServerAddress`), subscribe to a configurable number of subjects (parameter `SubjectsPerClient`) from a configurable set of subjects (parameter `Subjects`), receive messages in real-time and continuously on the subscribed subjects, and compute the latency for the received messages.

1.2 Migratory PushAgent-Benchmark

Migratory PushAgent-Benchmark is used to supply messages for Migratory PushServer to be consumed by the Migratory PushClient-Benchmark that simulates a lot of concurrent web clients.

In addition, you can connect to Migratory PushServer using one or more browsers running any example application that subscribes to the subjects provided by the Migratory PushAgent-Benchmark. This can eventually be used in the initial phase of your tests to see that Migratory PushServer delivers your data at the configured frequency.

Migratory PushAgent-Benchmark is able to connect to a Migratory PushServer instance (parameter `ServerHost`) and provide configurable message size (parameter `Size`) at a configurable frequency (parameter `Frequency`) for a configurable list of subjects (parameter `Subjects`).

Chapter 2

Latency

2.1 How latency is computed?

Latency is defined here as the time for a message to propagate from Migratory PushAgent-Benchmark to Migratory PushClient-Benchmark. End-to-end latency is the delta between the time a message is published by Migratory PushAgent-Benchmark and the time this message is received by Migratory PushClient-Benchmark. See Figure 2.1.

Every time Migratory PushAgent-Benchmark is sending a message, it adds a time-stamp information to that message. Time-stamp is a Unix time expressed in milliseconds, i.e. the number of milliseconds elapsed from January 1, 1970 Coordinated Universal Time (UTC).

Every time Migratory PushClient-Benchmark receives a message it extracts the time-stamp information from the message and computes the message latency as a difference between the current time of message reception and the time information extracted from the message.

Figure 2.1 shows how latency is computed.

2.2 How latency is displayed?

Migratory PushClient-Benchmark displays on standard output and in the log file every 5 seconds the following statistics:

- **Minimum Latency**. This is the smallest latency for all the messages in the system.
- **Max Latency**. This is the highest latency for all the messages in the system.
- **Mean Latency**. Supposing the latency values until a given time are λ_1, λ_2, ..., λ_k (k is the sample size, i. e. the number of received messages or the number of latency values), then their mean is:

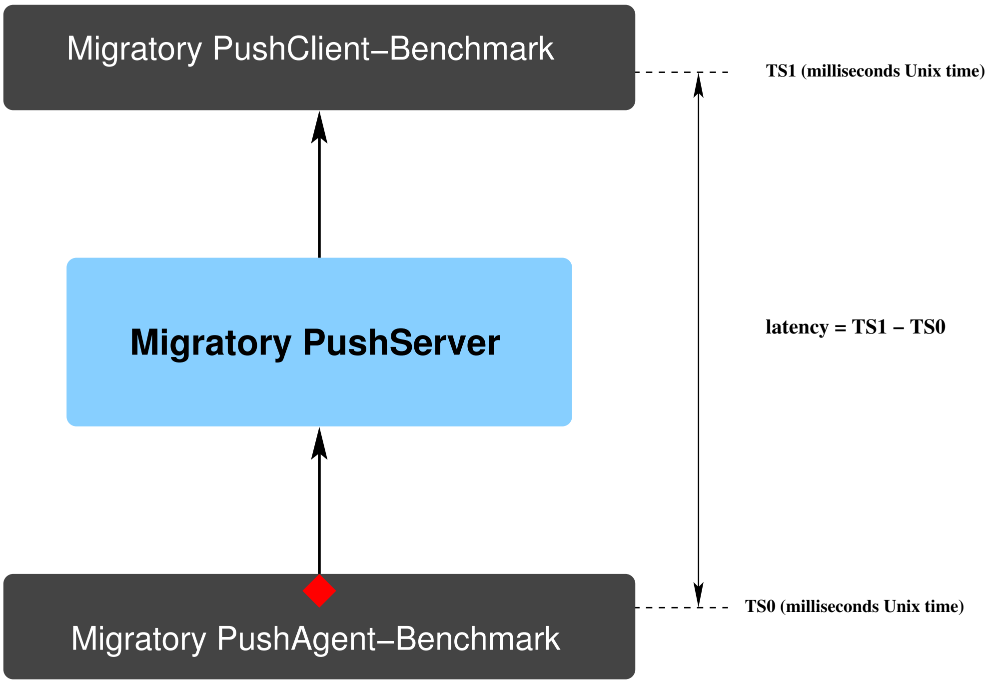


Figure 2.1: Latency computation.

$$\mu_k = \frac{\lambda_1 + \lambda_2 + \ldots + \lambda_k}{k}$$

- **Latency Standard Deviation**. Supposing the latency values until a given time are λ_1, λ_2, . . ., λ_k, and their mean is μ_k, then their standard deviation is calculated with the following formula:

$$\sqrt{\frac{\sum_{i=1}^{k} (\lambda_i - \mu_k)^2}{k-1}}$$

All the statistics above use efficient implementations able to compute their values on the fly for all the messages in the system when a new message is received. In this way, the statistics displayed by the PushClient-Benchmark are the actual statistics of all the messages in the system as if they were computed by the above formulas.

Chapter 3

Benchmarking setup and results

In this chapter we report the performance test results obtained in a particular hardware and software setup.

3.1 Test configuration

3.1.1 Hardware

We used a lot of 8 identical machines as follows:

Vendor Model	Dell PowerEdge SC1435
Number of CPU's	2
Core count per CPU	2
Number of cores	4
CPU type	Dual-Core AMD Opteron™Processor 2212 HE, 2.0 GHz
Cache	2MB Level 2 Cache per CPU
Bus speed	1000 GHz
Memory	16 GB RAM
Eth 0, Eth 1	Broadcom NetXTreme I Server 5721, 1GB
BIOS	1.0.0 Dell (10/09/06)

3.1.2 Operating System

All the machines ran the same version of the operating system as follows:

CentOS 5.5, Linux kernel 2.6.18-194.11.3.el5, 64-bit.

3.2 Tests Procedure

One instance of Migratory PushServer has been deployed on one machine as described in the previous section (machine 1). Another machine was used to run one instance of Migratory PushAgent-Benchmark and one instance of Migratory PushClient-Benchmark (machine 2). See Figure 3.1. The remaining 6 machines were used when necessary to run additional instances of Migratory PushClient-Benchmark (machine 3, . . . , machine 8).

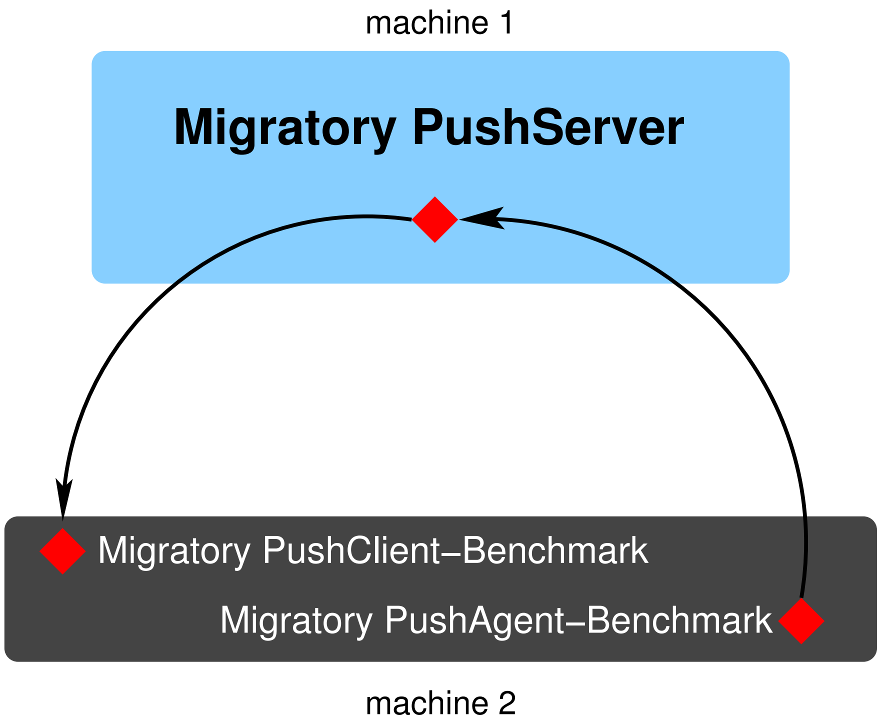


Figure 3.1: Deployment.

The main item measured here is the end-to-end latency against:

- the number of connected clients
- the messages size.

For each performed test we get:

- **mean latency**, **minimum latency**, **maximum latency**, and **standard deviation** computed for a given **sample size** (i.e. the total number of messages transmitted by Migratory PushServer during a test);

- **CPU load average** as reported by Linux – see `http://en.wikipedia.org/wiki/Load_(computing)` for further details;
- **transmit bandwidth**: the amount of data – expressed in Megabyte per second – transmitted by Migratory PushServer and received by the Migratory PushClient instance(s).

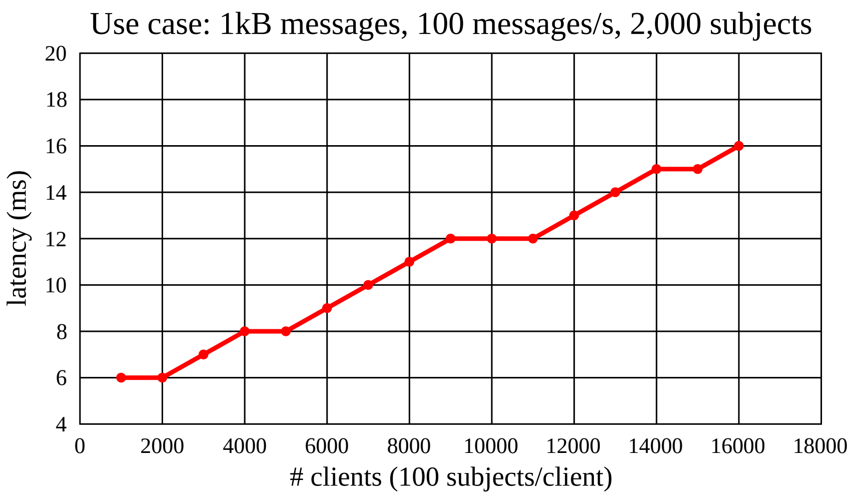
3.3 Case 1: Large data messages

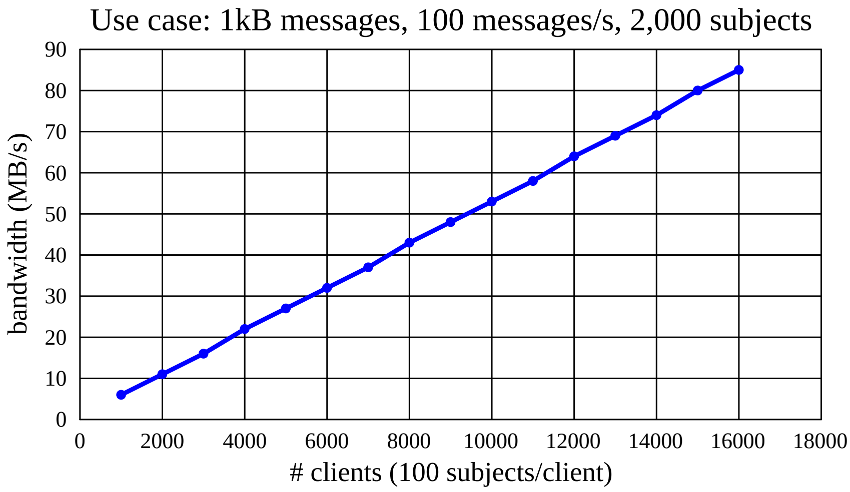
Case 1.1 – 1 Kilobyte messages

- Total number of subjects: 2,000
- Message rate: 100 messages/second (each subject is updated every 20 seconds)
- Subjects per client: 100 (5 messages/second for each client)

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
1,000	6	4	48	<1	6	1,523,050	<0.1
2,000	6	4	75	<1	11	3,013,188	0.3
3,000	7	5	209	<1	16	4,504,708	0.8
4,000	8	5	224	<1	22	6,066,941	0.5
5,000	8	5	224	<1	27	7,630,800	0.6
6,000	9	5	221	<1	32	9,049,884	1.2
7,000	10	6	263	<1	37	10,564,194	1.5
8,000	11	7	275	<1	43	11,809,484	1.7
9,000	12	8	264	<1	48	13,579,769	1.9
10,000	12	9	265	<1	53	15,095,323	2.2
11,000	12	7	242	<1	58	16,700,210	2.5
12,000	13	7	258	<1	64	18,280,765	2.8
13,000	14	9	432	<1	69	19,686,426	3
14,000	15	9	432	<1	74	21,780,724	3.2
15,000	15	7	263	<1	80	22,858,607	4
16,000	16	10	489	<1	85	24,392,030	4

Table 3.1: Case 1.1 data.



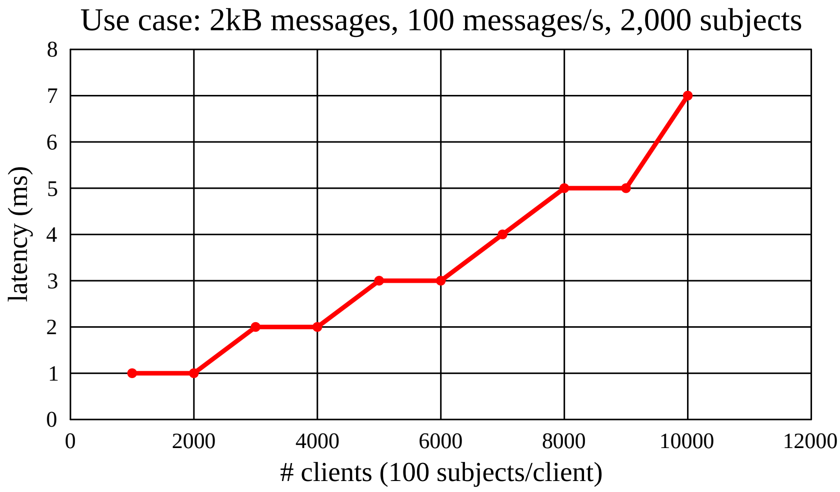


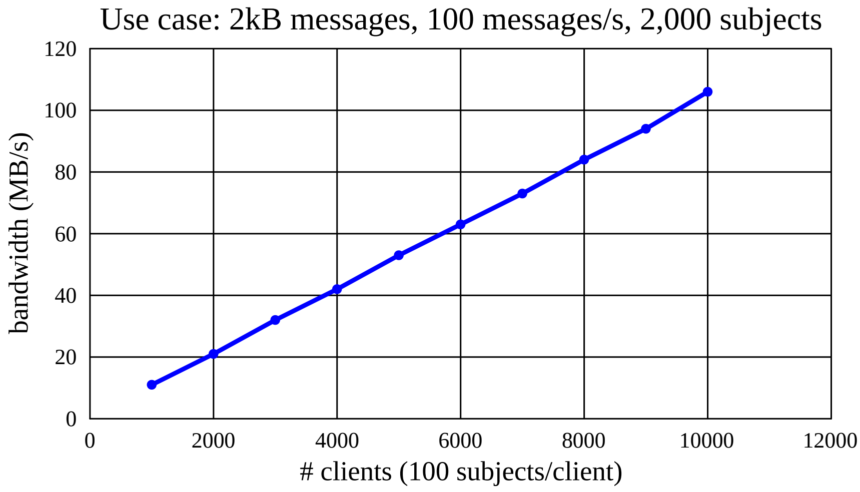
Case 1.2 – 2 Kilobytes messages

- Total number of subjects: 2,000
- Message rate: 100 messages/second (each subject is updated every 20 seconds)
- Subjects per client: 100 (5 messages/second for each client)

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
1,000	1	1	50	<1	11	1,506,433	0.2
2,000	1	1	71	<1	21	3,036,233	0.3
3,000	2	2	396	<1	32	4,505,509	0.4
4,000	2	2	316	<1	42	6,090,810	1
5,000	3	2	405	<1	53	7,512,875	1.1
6,000	3	3	412	<1	63	9,018,321	1.4
7,000	4	5	419	<1	73	10,623,341	1.8
8,000	5	6	423	<1	84	12,063,589	2.4
9,000	5	5	421	<1	94	13,656,095	1.7
10,000	7	8	444	<1	106	15,180,040	2.4

Table 3.2: Case 1.2 data.



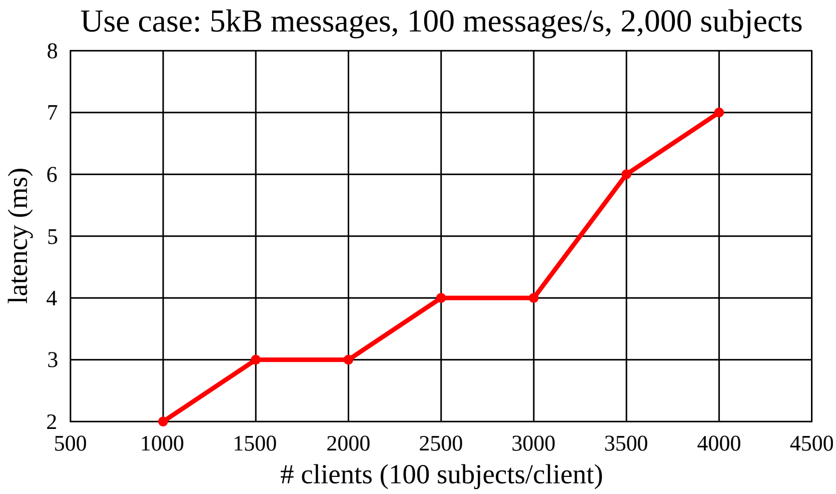


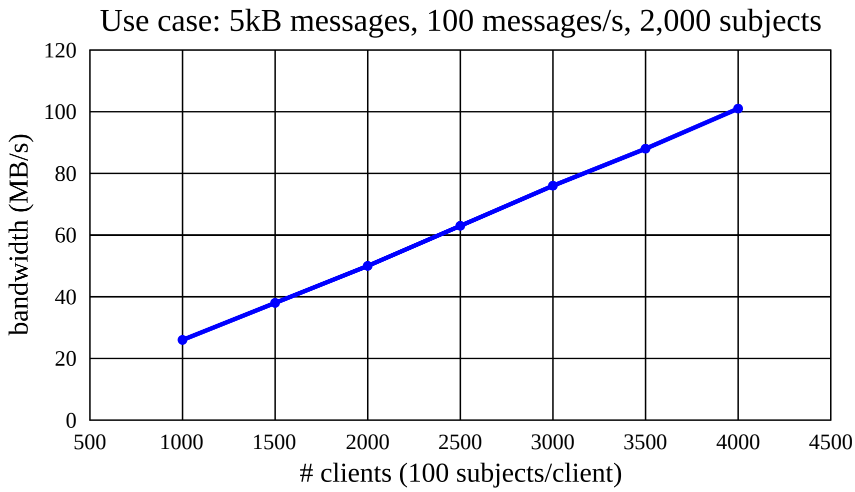
Case 1.3 – 5 Kilobytes messages

- Total number of subjects: 2,000
- Message rate: 100 messages/second (each subject is updated every 20 seconds)
- Subjects per client: 100 (5 messages/second for each client)

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
1,000	2	1	211	<1	26	1,521,388	0.3
1,500	3	2	215	<1	38	2,279,720	0.5
2,000	3	2	268	<1	50	3,045,472	0.5
2,500	4	3	268	<1	63	3,784,124	1.3
3,000	4	3	301	<1	76	4,512,182	1.2
3,500	6	4	235	<1	88	5,330,264	1.5
4,000	7	5	259	<1	101	6,056,389	1.6

Table 3.3: Case 1.3 data.





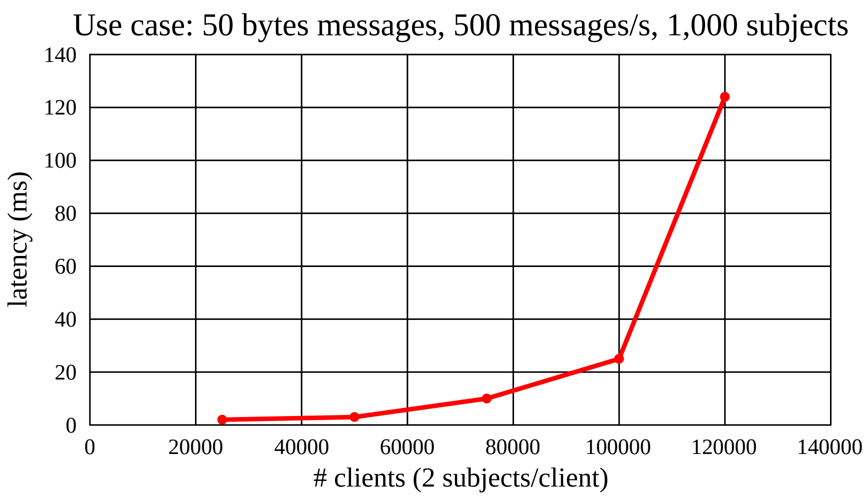
3.4 Case 2: Large number of clients

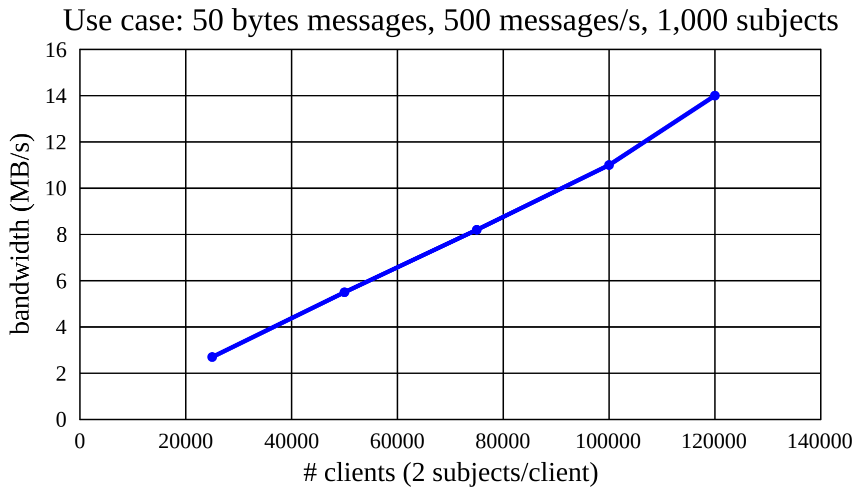
Case 2.1 – low update rates

- Total number of subjects: 1,000
- Message rate: 500 messages/second (each subject is updated every 2 seconds)
- Subjects per client: 2 (1 message/second for each client)
- Message size: 50 bytes

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
25,000	2	5	270	<1	2.7	8,829,941	0.8
50,000	3	13	552	<1	5.5	15,133,654	3.3
75,000	10	26	1,478	<1	8.2	22,684,272	4
100,000	25	49	1,561	<1	11	34,899,324	4
120,000	124	137	3,584	<1	14	36,579,764	6.5

Table 3.4: Case 2.1 data.



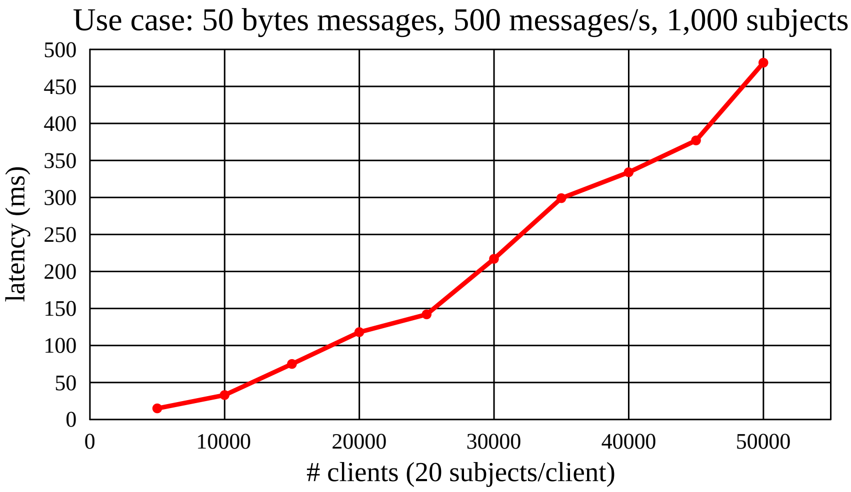


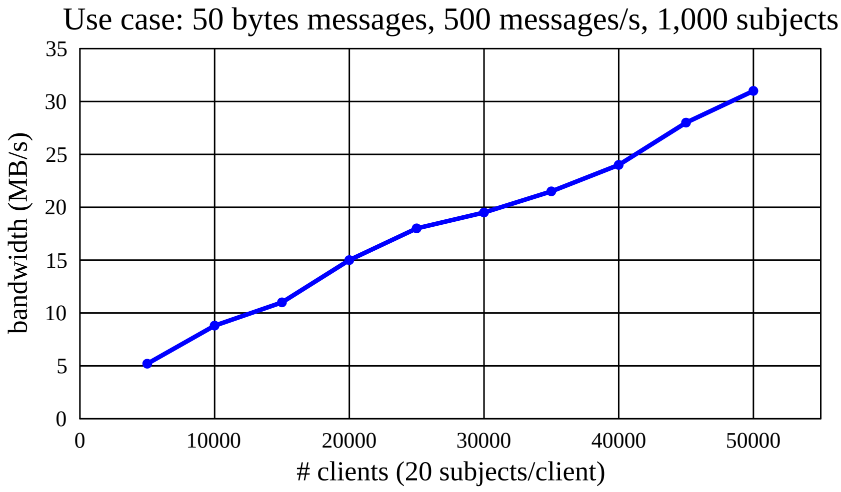
Case 2.2 – medium update rates

- Total number of subjects: 1,000
- Message rate: 500 messages/second (each subject is updated every 2 seconds)
- Subjects per client: 20 (10 messages/second for each client)
- Message size: 50 bytes

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
5,000	15	28	675	<1	5.3	3,976,649	2
10,000	33	41	1,547	<1	8.8	7,387,188	4.5
15,000	75	40	657	<1	11	12,597,928	5.4
20,000	118	60	1,777	<1	15	16,432,743	5.4
25,000	142	113	765	<1	18	26,787,419	5.6
30,000	217	149	821	<1	19.5	30,354,426	5.8
35,000	299	196	2,051	<1	21.5	30,101,934	6.0
40,000	334	205	1,185	<1	24	34,254,588	5.8
45,000	377	214	1,344	<1	28	37,964,889	6.6
50,000	482	255	2,603	<1	31	61,334,482	7.6

Table 3.5: Case 2.2 data.



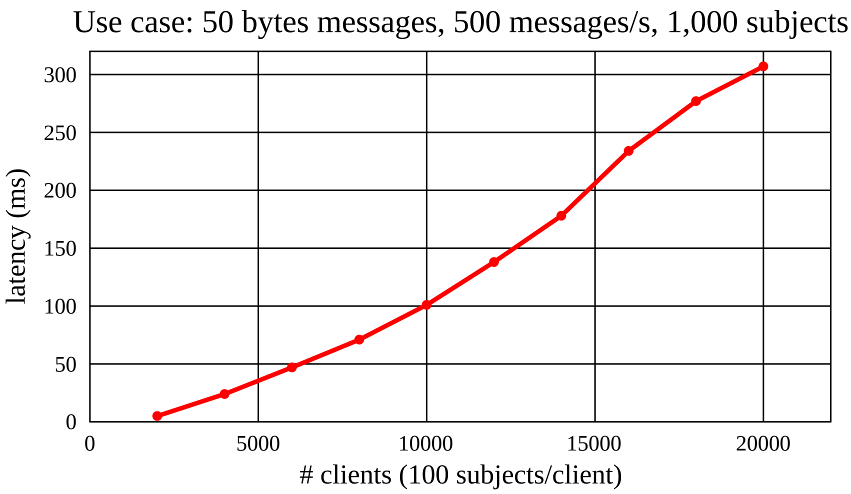


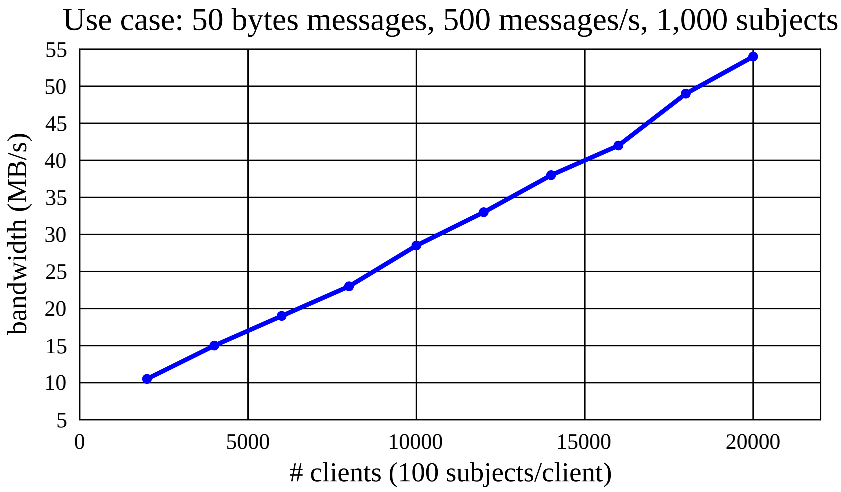
Case 2.3 – high update rates

- Total number of subjects: 1,000
- Message rate: 500 messages/second (each subject is updated every 2 seconds)
- Subjects per client: 100
- Message size: 50 bytes (50 messages/second for each client)

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
2,000	5	12	300	<1	10.5	43,690,736	4.4
4,000	24	21	696	<1	15	18,070,474	3
6,000	47	36	224	<1	19	23,758,245	2.2
8,000	71	49	299	<1	23	33,148,327	5
10,000	101	64	593	<1	28.5	46,701,688	4.3
12,000	138	86	500	<1	33	49,136,370	4.3
14,000	178	113	617	<1	38	59,312,296	5.5
16,000	234	142	1,140	<1	42	189,854,372	4.4
18,000	277	193	1,442	<1	49	339,817,603	6.3
20,000	307	165	1,684	<1	54	824,259,984	4.2

Table 3.6: Case 2.3 data.



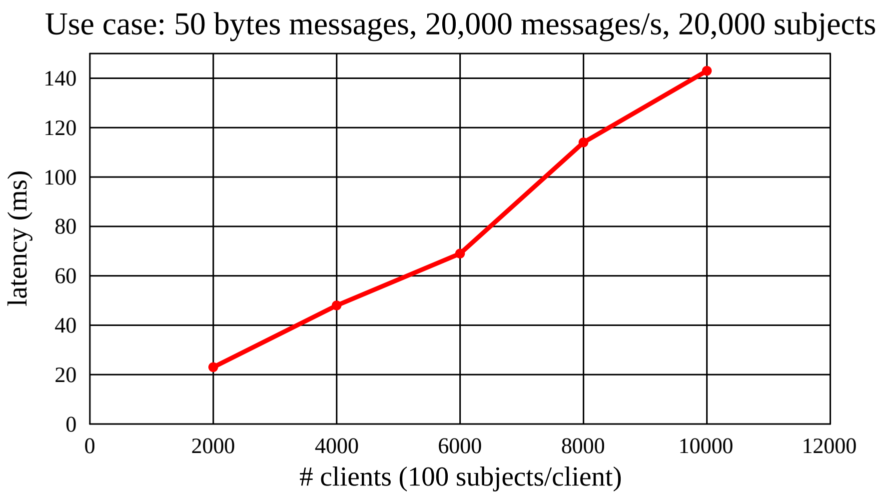

Use case: 50 bytes messages, 500 messages/s, 1,000 subjects
bandwidth (MB/s)
5
10
15
20
25
30
35
40
45
50
55
0
5000
10000
15000
20000
clients (100 subjects/client)

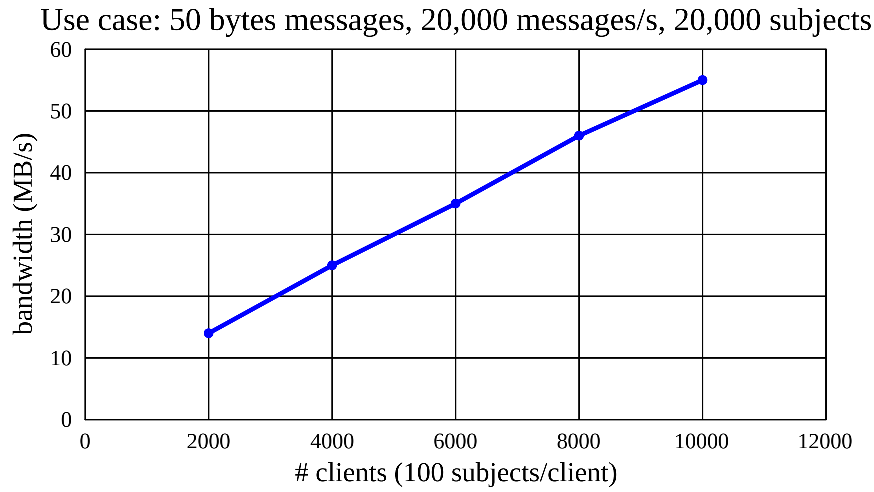
3.5 Case 3: Very high update rates

- Total number of subjects: 20,000
- Message rate: 20,000 messages/second (each subject is updated every 1 second)
- Subjects per client: 100 (100 messages/second for each client)
- Message size: 50 bytes

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
2,000	23	11	358	<1	14	18,692,650	3.7
4,000	48	40	657	<1	25	38,742,523	4.1
6,000	69	48	769	<1	35	51,072,742	3.9
8,000	114	63	816	<1	46	71,960,390	7.2
10,000	143	79	1,223	<1	55	120,223,404	5.8

Table 3.7: Case 3 data.



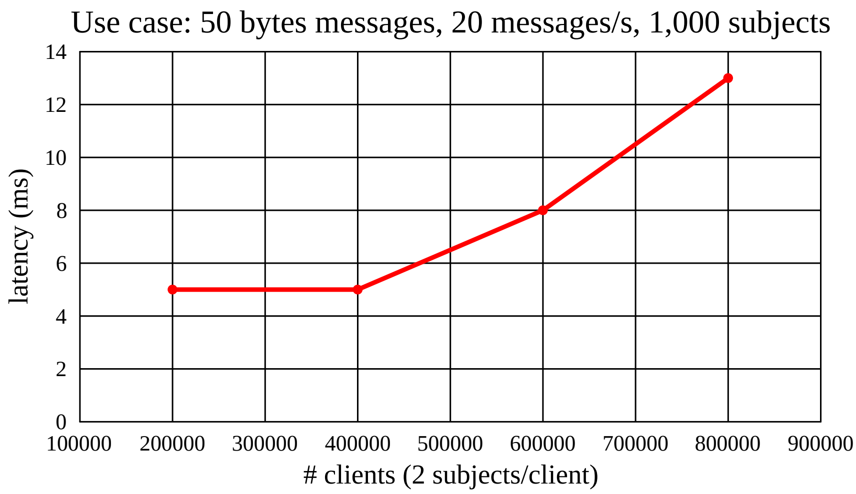

Use case: 50 bytes messages, 20,000 messages/s, 20,000 subjects
bandwidth (MB/s)
60
50
40
30
20
10
0
0
2000
4000
6000
8000
10000
12000
clients (100 subjects/client)

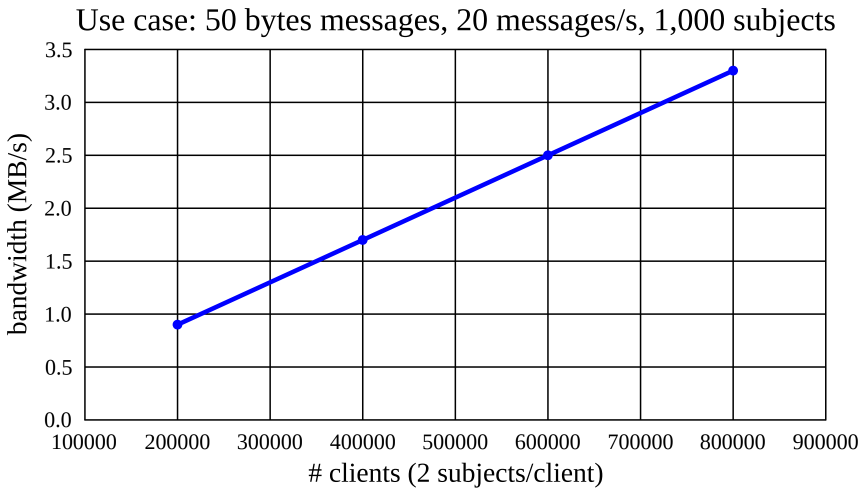
3.6 Case 4: Very large number of clients

- Total number of subjects: 1,000
- Message rate: 20 messages/second (each subject is updated every 50 seconds)
- Subjects per client: 2 (each client receives 1 message every 25 seconds)
- Message size: 50 bytes

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
200,000	5	29	828	<1	0.9	3,523,096	0.5
400,000	5	19	396	<1	1.7	2,408,087	0.6
600,000	8	41	846	<1	2.5	3,619,141	0.7
800,000	13	65	1,310	<1	3.3	4,862,843	0.8

Table 3.8: Case 4 data.




Use case: 50 bytes messages, 20 messages/s, 1,000 subjects
bandwidth (MB/s)
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
100000
200000
300000
400000
500000
600000
700000
800000
900000
clients (2 subjects/client)

3.7 Case 5: Extreme number of clients

Case 5.1

- Total number of subjects: 1,000
- Update rate: 10 messages/second (each subject is updated every 100 seconds)
- Subjects per client: 2 (each client receives 1 message every 50 seconds)
- Message size: 50 bytes

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
1,000,000	15	75	1,443	<1	2.1	2,391,180	1.1

Table 3.9: Case 5.1 data.

Case 5.2

- Total number of subjects: 10,000
- Update rate: 20 messages/second (each subject is updated every 500 seconds)
- Subjects per client: 2 (each client receives 1 message every 250 seconds)
- Message size: 50 bytes

Number of clients (ms)	Mean Latency (ms)	Standard Deviation (ms)	Maximum Latency (ms)	Minimum Latency (ms)	Transmit Bandwidth (MB/s)	Sample size	CPU average load
1,000,000	7	81	2,068	<1	0.5	822,105	0.2

Table 3.10: Case 5.2 data.

3.8 Document change log

- 10.03.10: added 120k clients test to Case 2.1 (low update rates);
- 10.03.10: removed decimals from latency values;
- 10.05.10: corrected message frequency from 10 to 20 in Case 4;
- 10.05.10: added Case 5.1;

3.9 Acknowledgements

Linux®is a registered trademark of Linus Torvalds.

AMD and Opteron are trademarks of Advanced Micro Devices, Inc.

Dell and PowerEdge are trademarks of Dell, Inc.

NetXTreme™is a trademark of Broadcom, Corp.

3.10 Contact

Please contact us at `support@migratory.ro` for any question.