VerneMQ with "Cluster HA improvements" compared to 1.11.0

1 Rationale

"High Availability" testing on VerneMQ clusters based on VerneMQ-1.11.0 revealed frequent message loss in QoS1,

See:

- Vernemq cluster HA : reporting results obtained for VerneMQ cluster deployments in k8s
- HA VerneMQ in Dialog : reporting similar message loss behavior in Dialog deployment

After detailed testing and analysis it was decided to fix the VerneMQ source code to improve HA behavior.

The open-source code fixes are given back to the VernmeMQ community via a pull request named "**Cluster HA improvements**".
At the time of writing this pull request is under evaluation for inclusion in a next release of the upstream source code.

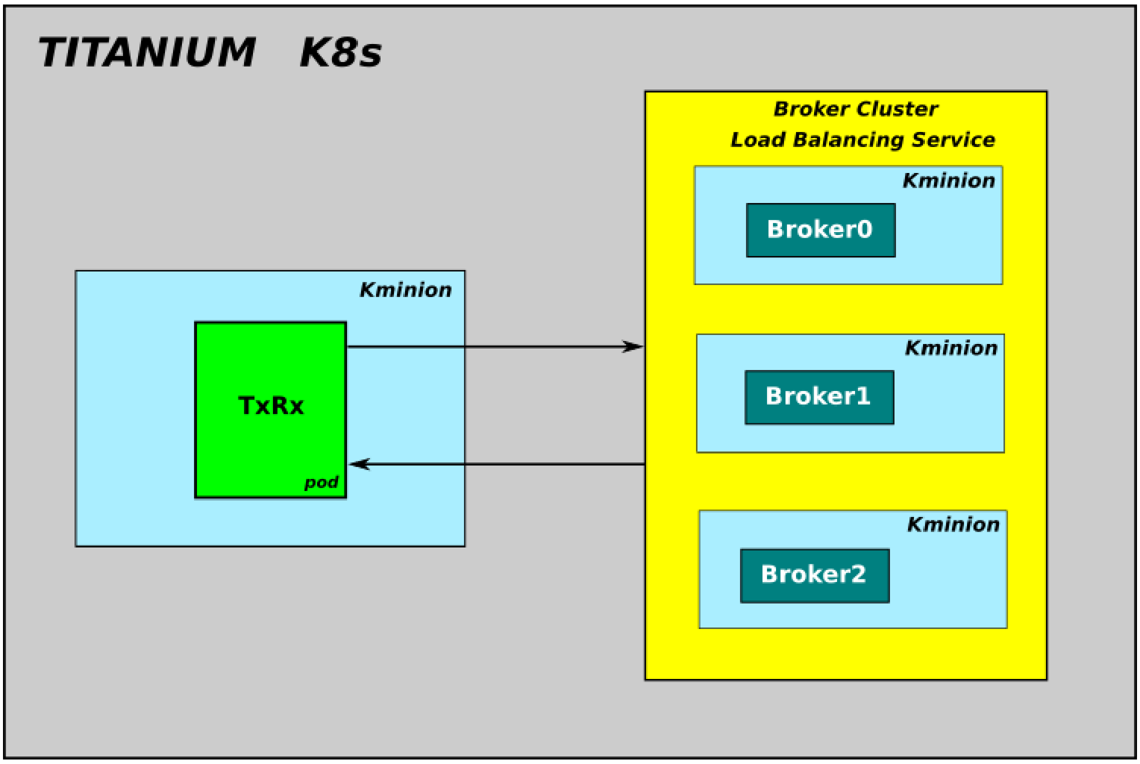
HA fixes inevitably come at the cost of some performance loss.
Therefore the test-setup that was used earlier for MQTT Broker Benchmarking was re-used to assess this performance loss.

This document reports the result of those tests.

2 Test Setup

The original test setup was based on VM's running on WindRiver Titanium.
See MQTT Broker Benchmarking for more details.

However, since currently all POC's are running in k8s, the test setup was modified to run in kubernetes pods.



Description :

- Test Environment :
    - HW platform : ARCH-AIO-2 (titan2)
    - WindRiver Titanium 18.01(R5)
    - Kubernetes 1.17.6 , installed via "kubespray", consisting of 5 VM nodes :
        - kmaster
        - kminion1, running the test transmitter and receiver (TxRx) microservice
        - kminion2, kminion3, kminion4 running the broker cluster nodes
- TxRx microservice :
    - single publisher(Tx) and subscriber(Rx)
    - **Tx part publishes messages to a single topic** to the broker cluster
    - publish rate is programmable
    - nr of messages is programmable
    - payload size is programmable
    - QoS is programmable
    - **Rx part subscribes** to the **topic** and buffers all received messages
    - At the end of the testrun the received messages are analysed:
        - compute tx and rx rates
        - computes "latency" between tx and rx times
- Broker cluster : n-node cluster deployed using helm chart or k8s operator
- TxRx runs in kminion1 node while the broker nodes are deployed in the 3 remaining kminions
- TxRx interfaces with the broker cluster via a k8s service
  (this automatically loadbalances connections accross the cluster nodes)

3 Test Results

3.1 Single Node

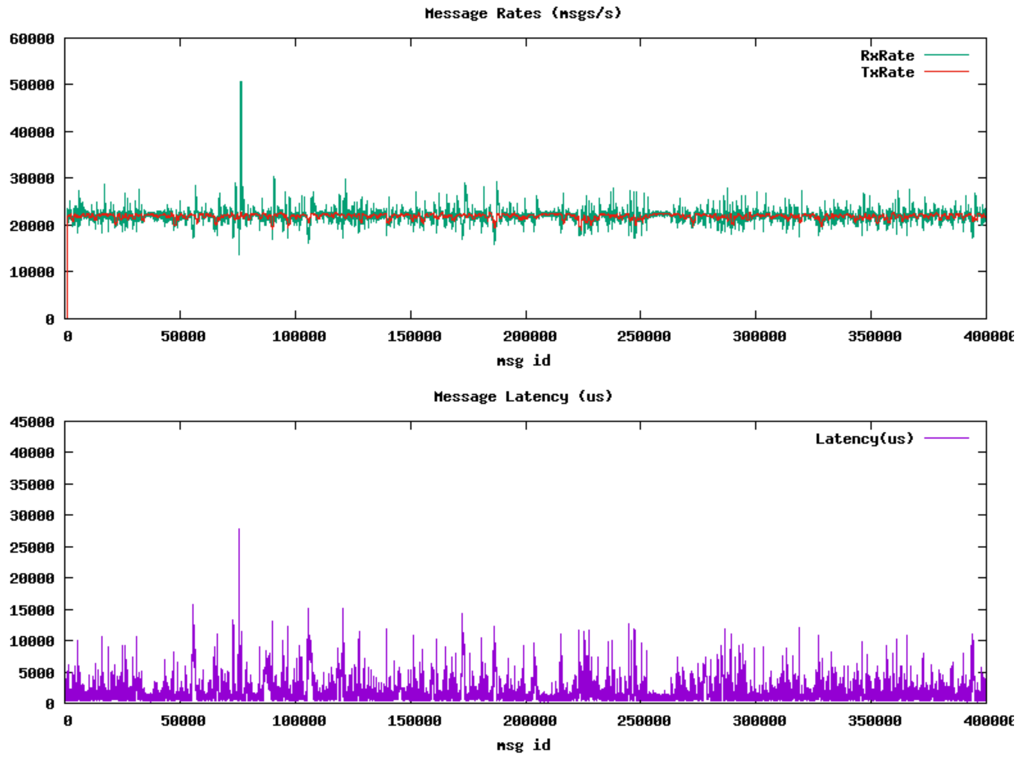
Performance tests in the original VM-based test setup on Vernemq v1.4.0-1 gave following results: (only Q0S 0 was tested then)

- QoS0 100 Byte : 30k msgs/s
- QoS0 64K : 3k1 msgs/s

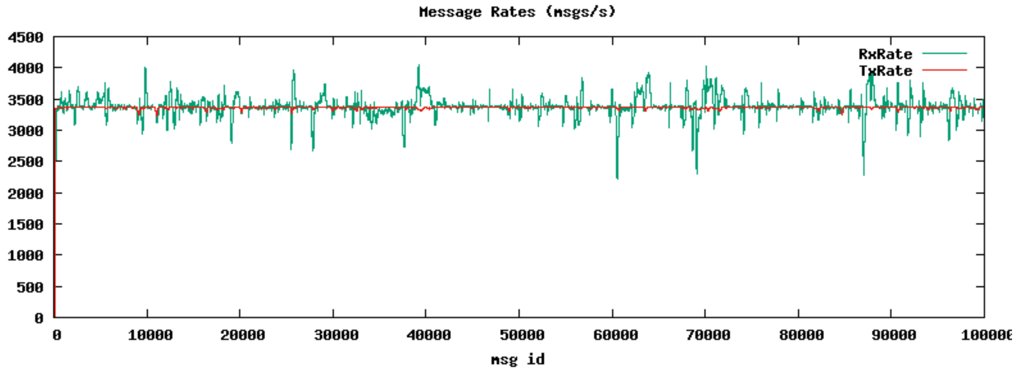
3.1.1 VerneMQ 1.11.0

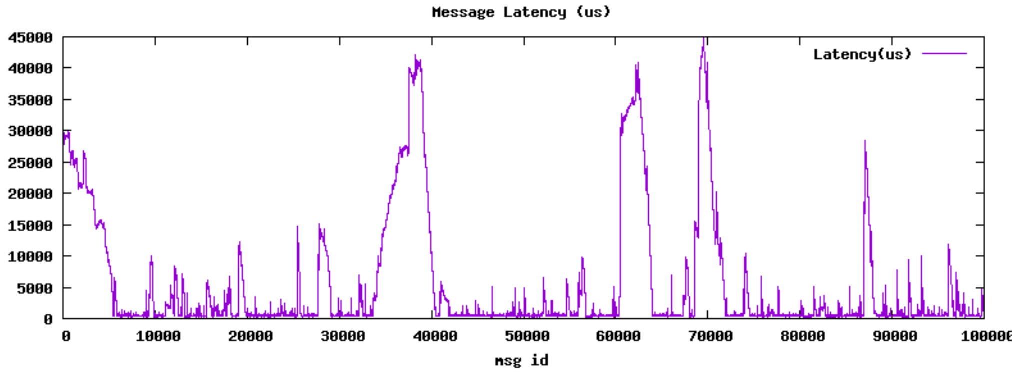
1.11.0	**100 Byte**		**64 KByte**	
QoS	msgs/s	Mbit/s	msgs/s	Mbit/s
0	21800	17.5	1955	1000
1	3400	2.7	991	507
2	2074	1.7	663	340

QoS0_100byte

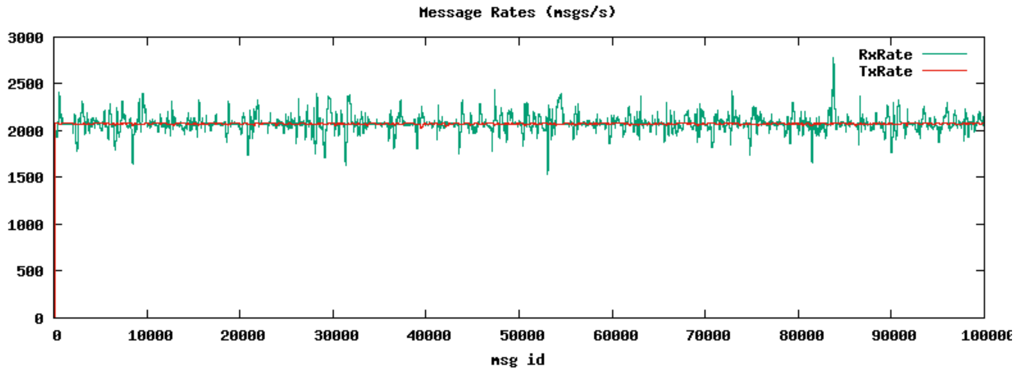


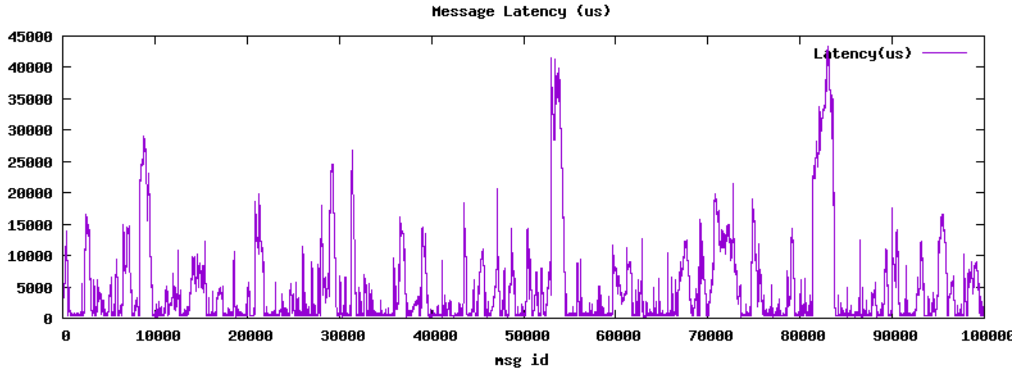
QoS1_100byte



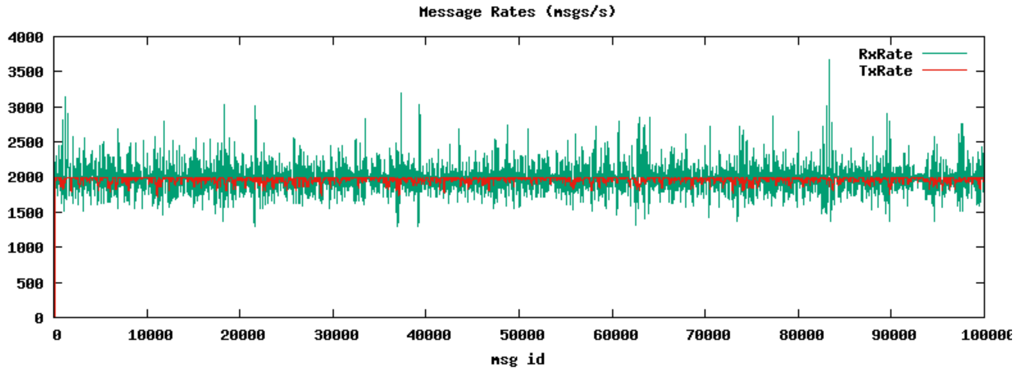


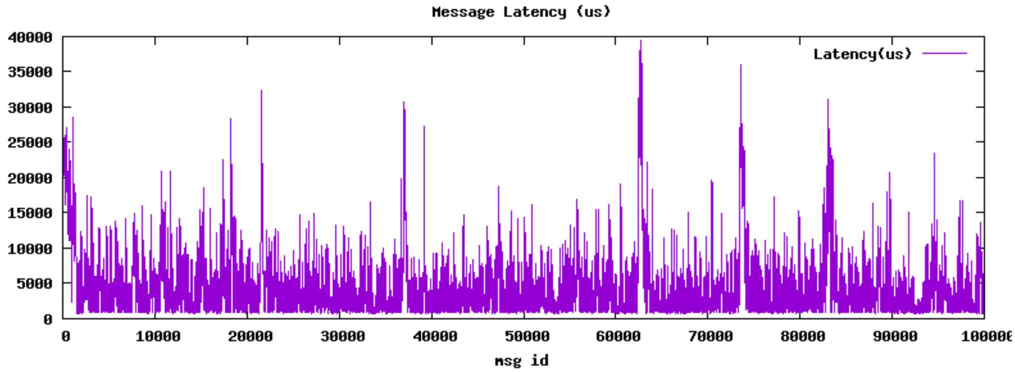
QoS2_100byte



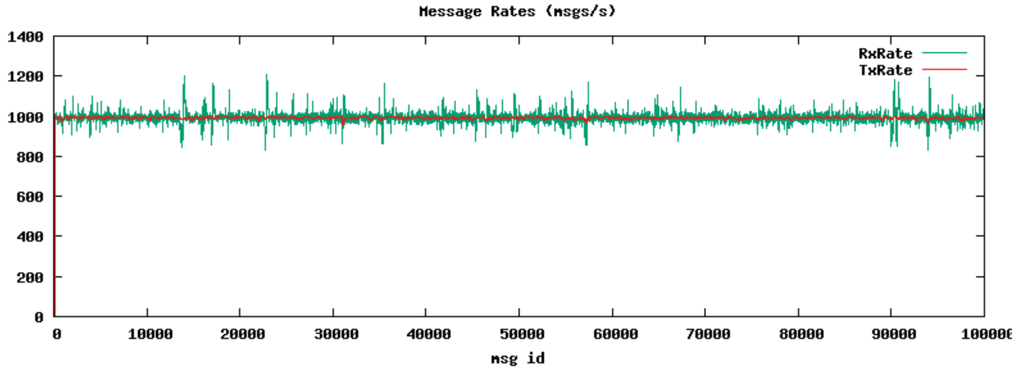


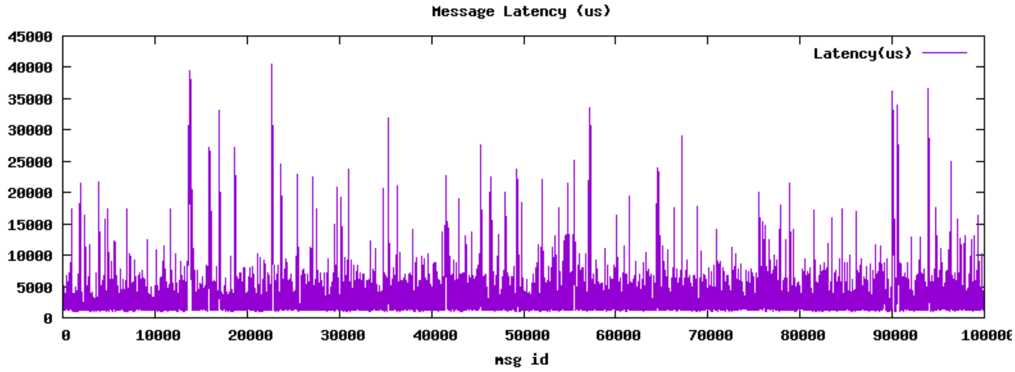
QoS0_64k



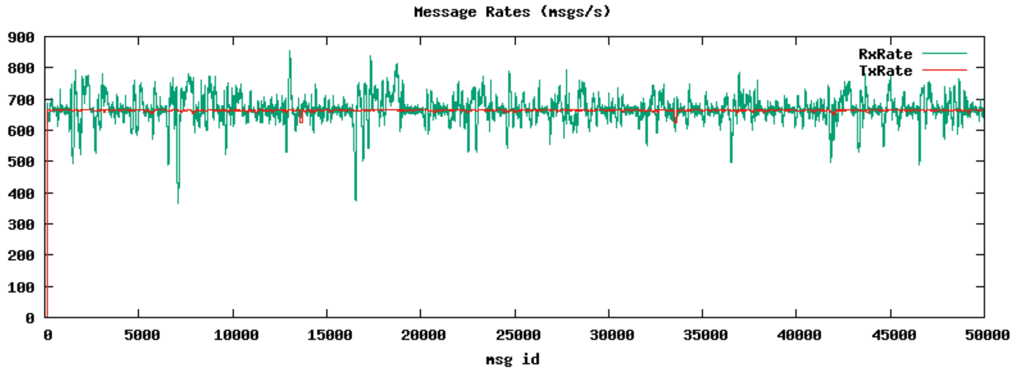


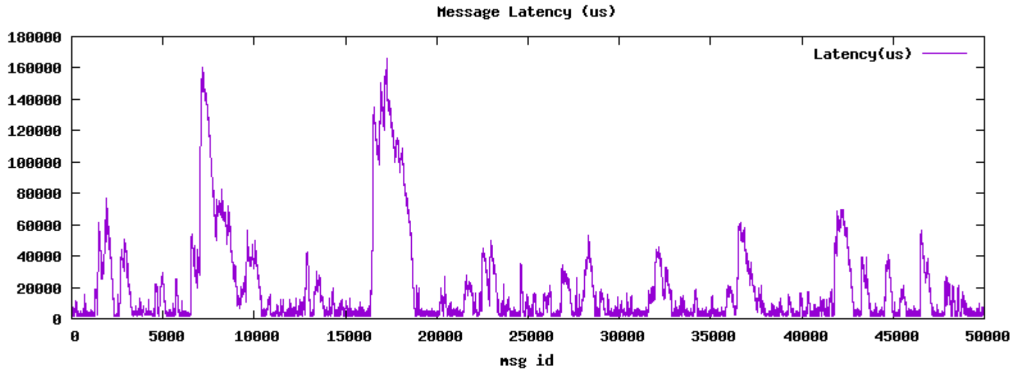
QoS1_64k





QoS2_64k

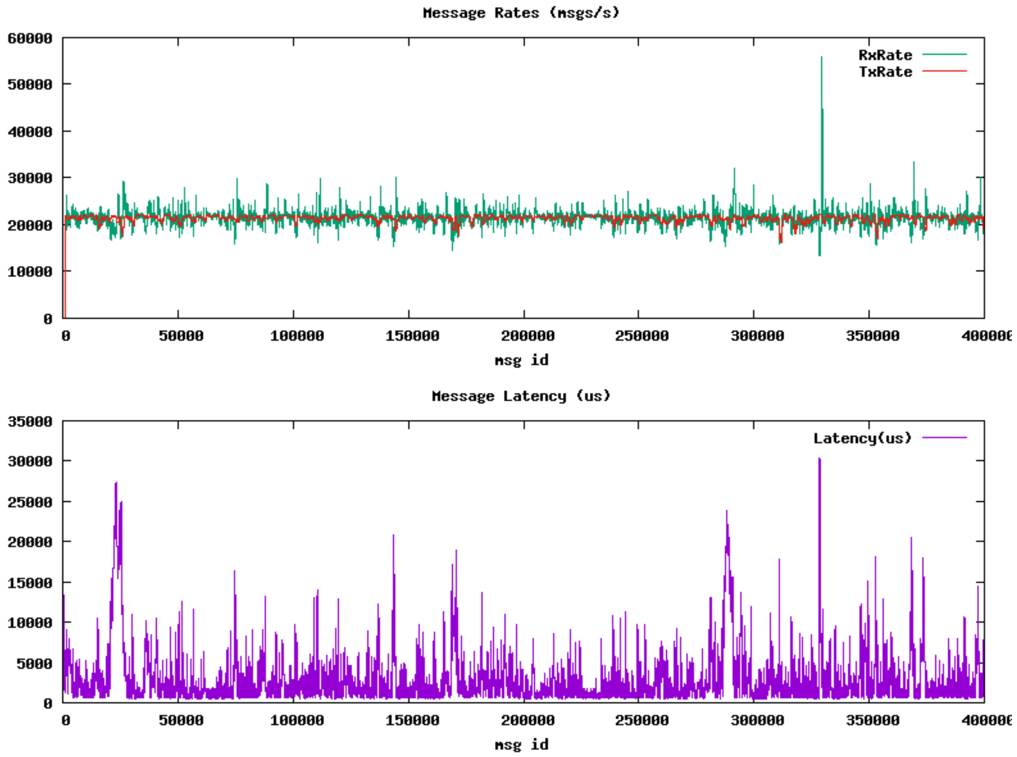

Message Rates (msgs/s)
RxRate
TxRate
900
800
700
600
500
400
300
200
100
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id


Message Latency (us)
Latency(us)
180000
160000
140000
120000
100000
80000
60000
40000
20000
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id

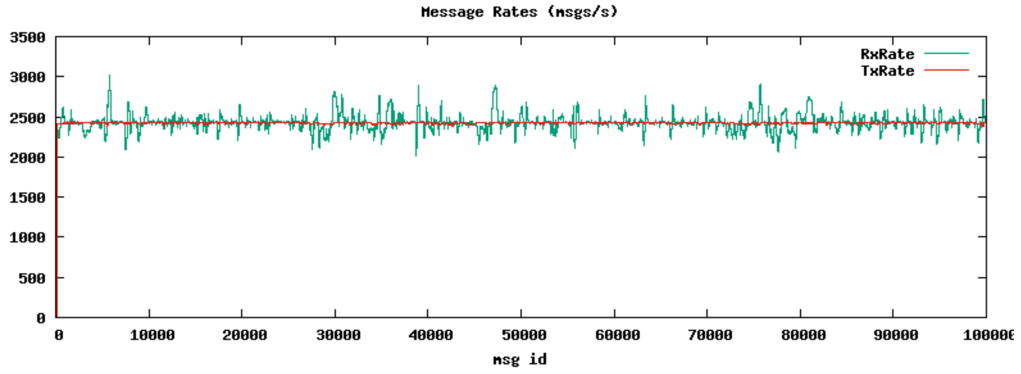
3.1.2 VerneMQ with "Cluster HA improvements"

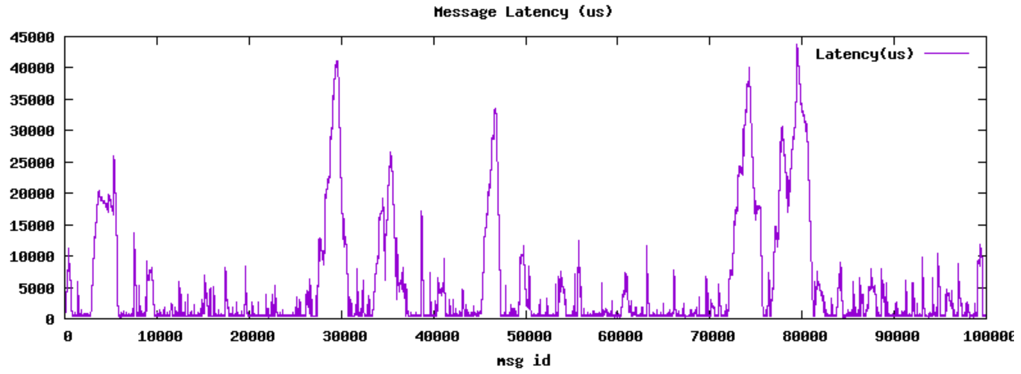
HA fix	**100 Byte**		**64 KByte**	
QoS	msgs/s	Mbit/s	msgs/s	Mbit/s
0	21300	17.0	1880	962
1	2424	1.9	829	425
2	1689	1.4	586	300

QoS0_100byte

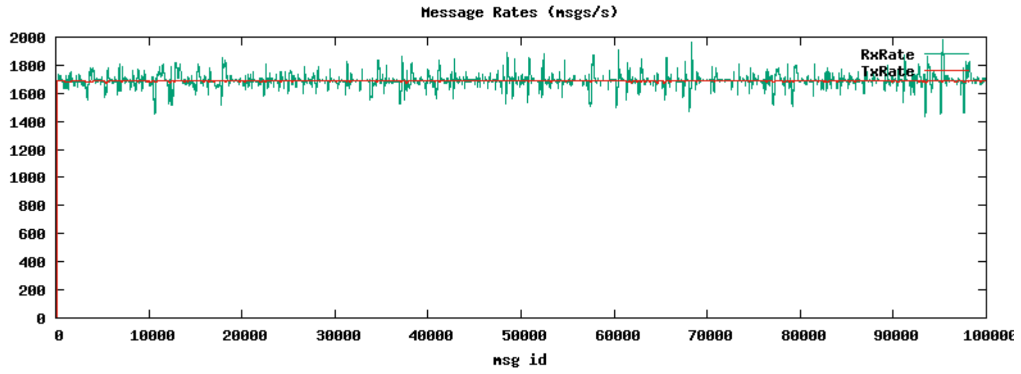


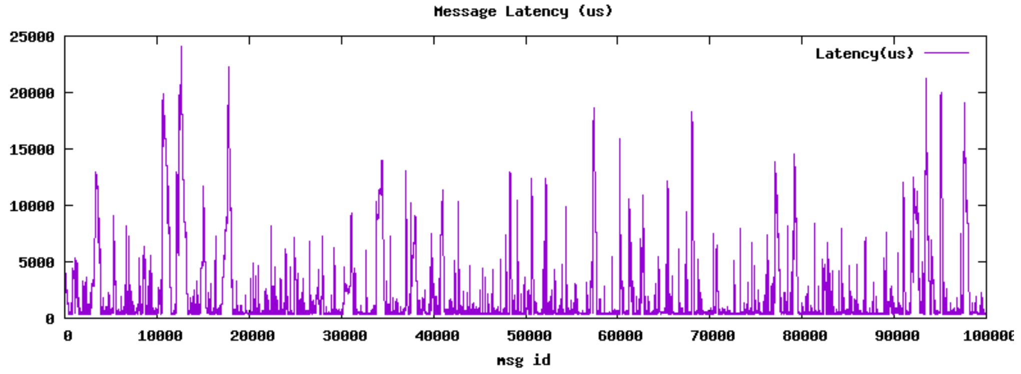
QoS1_100byte



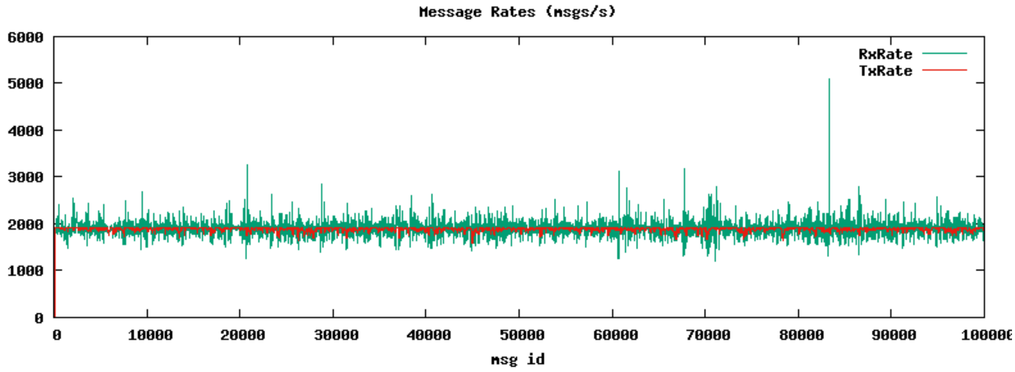


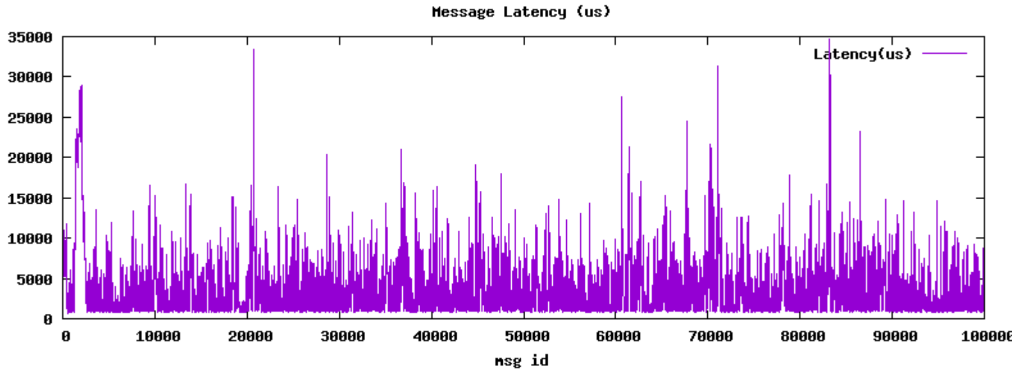
QoS2_100byte



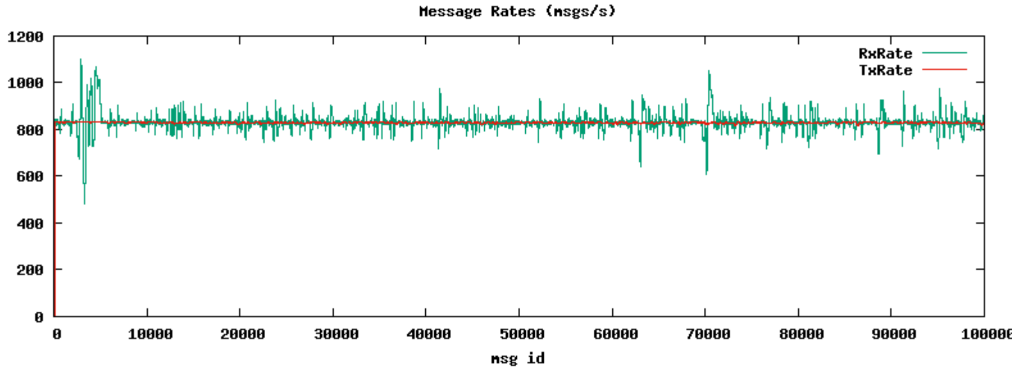


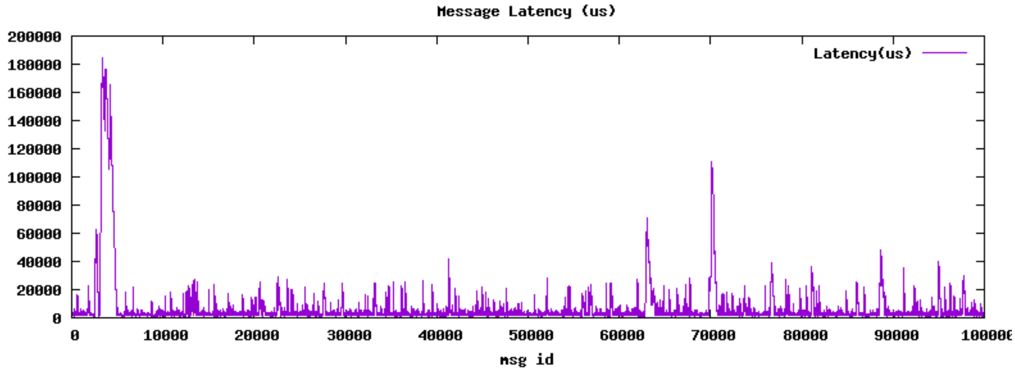
QoS0_64k



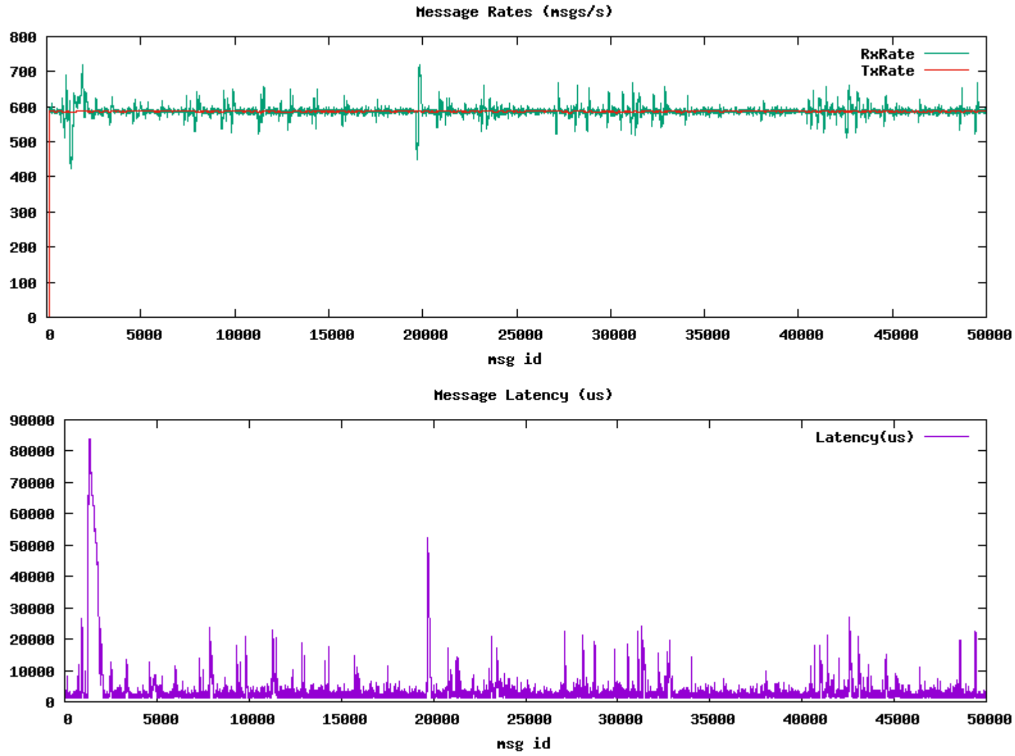


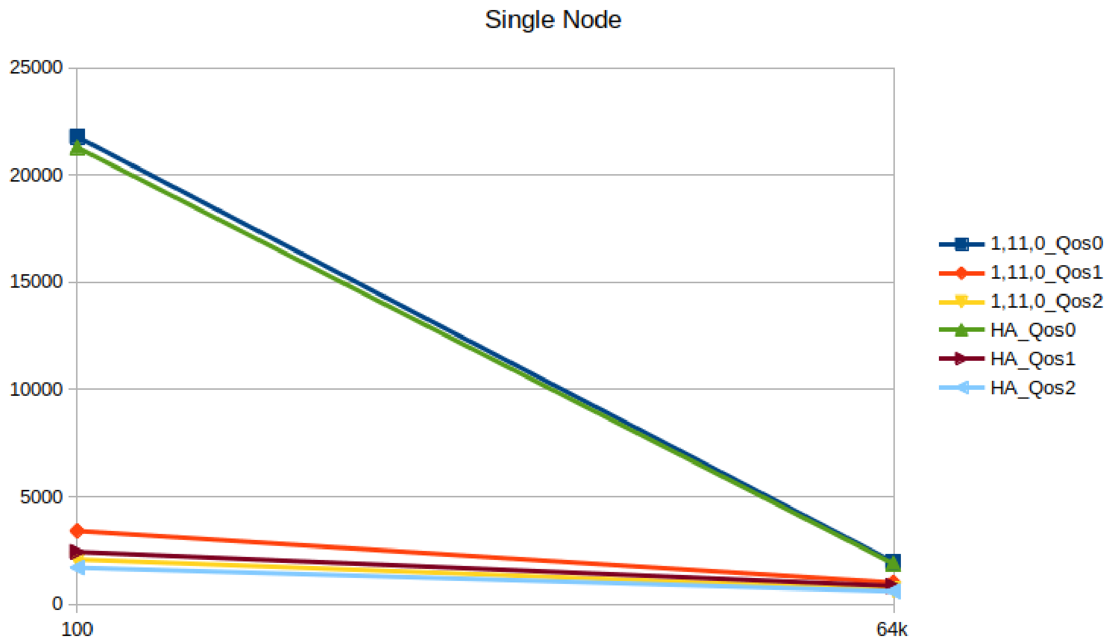
QoS1_64k





QoS2_64k


Message Rates (msgs/s)
RxRate
TxRate
800
700
600
500
400
300
200
100
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id
Message Latency (us)
Latency(us)
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id

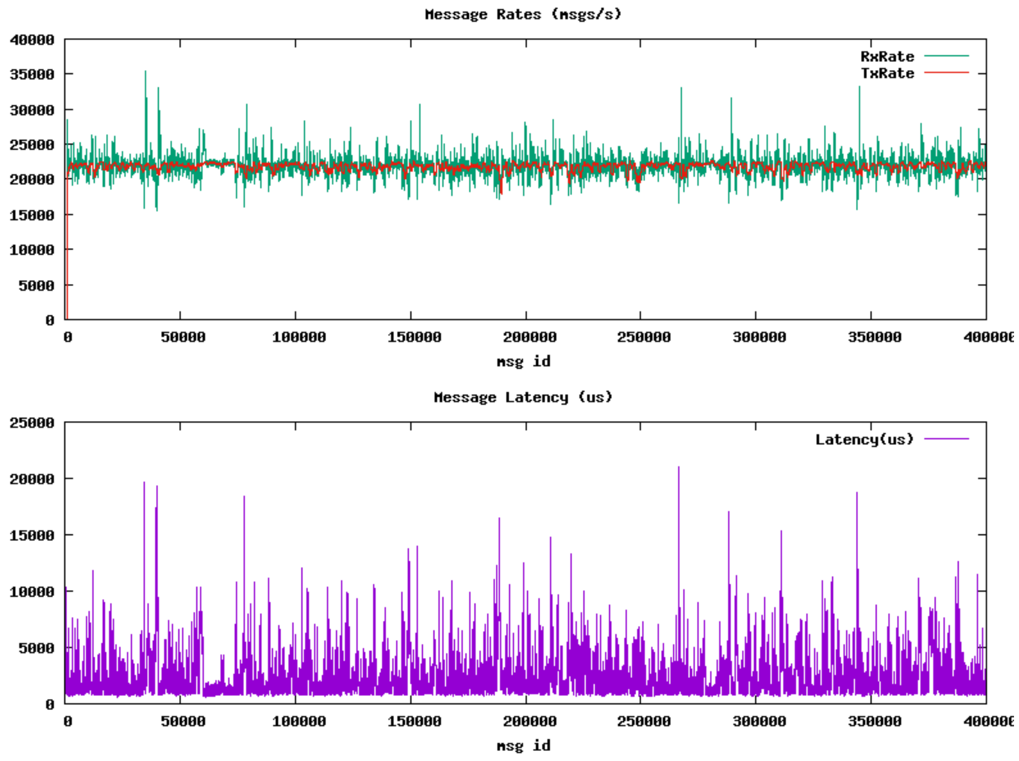

Single Node
25000
20000
15000
10000
5000
0
100
64k
1,11,0_Qos0
1,11,0_Qos1
1,11,0_Qos2
HA_Qos0
HA_Qos1
HA_Qos2

3.2 2-node cluster

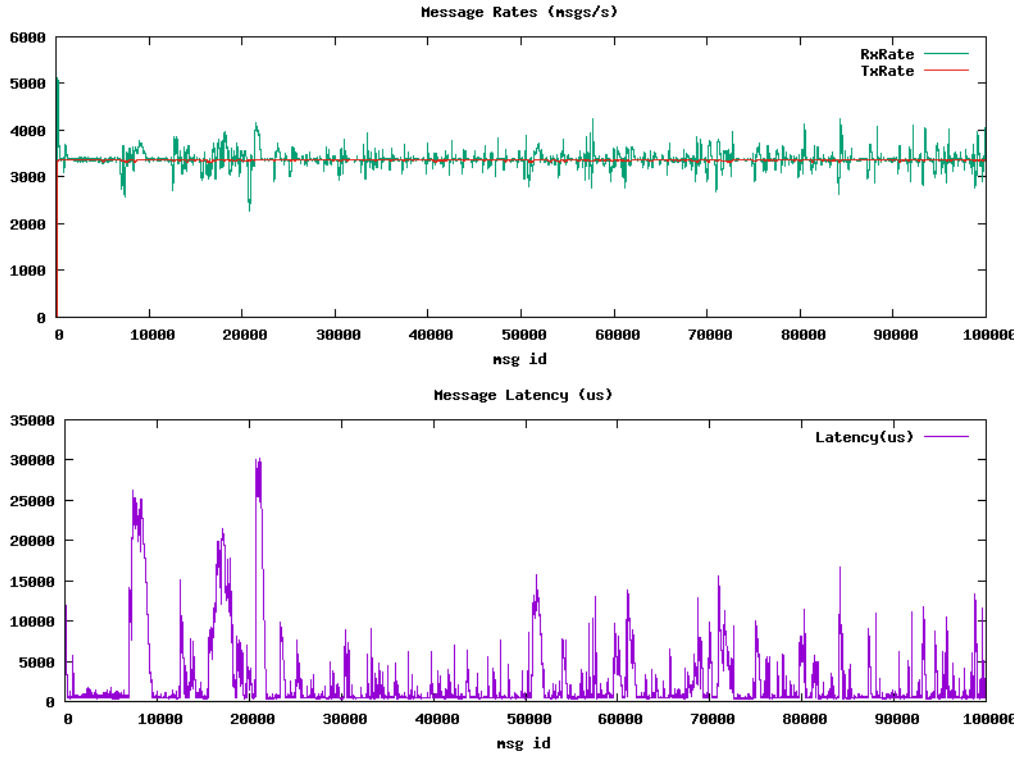
3.2.1 VerneMQ 1.11.0

1.11.0	**100 Byte**		**64 KByte**	
QoS	msgs/s	Mbit/s	msgs/s	Mbit/s
0	21740	17.4	1514	775
1	3366	2.7	991	507
2	2072	1.7	711	364

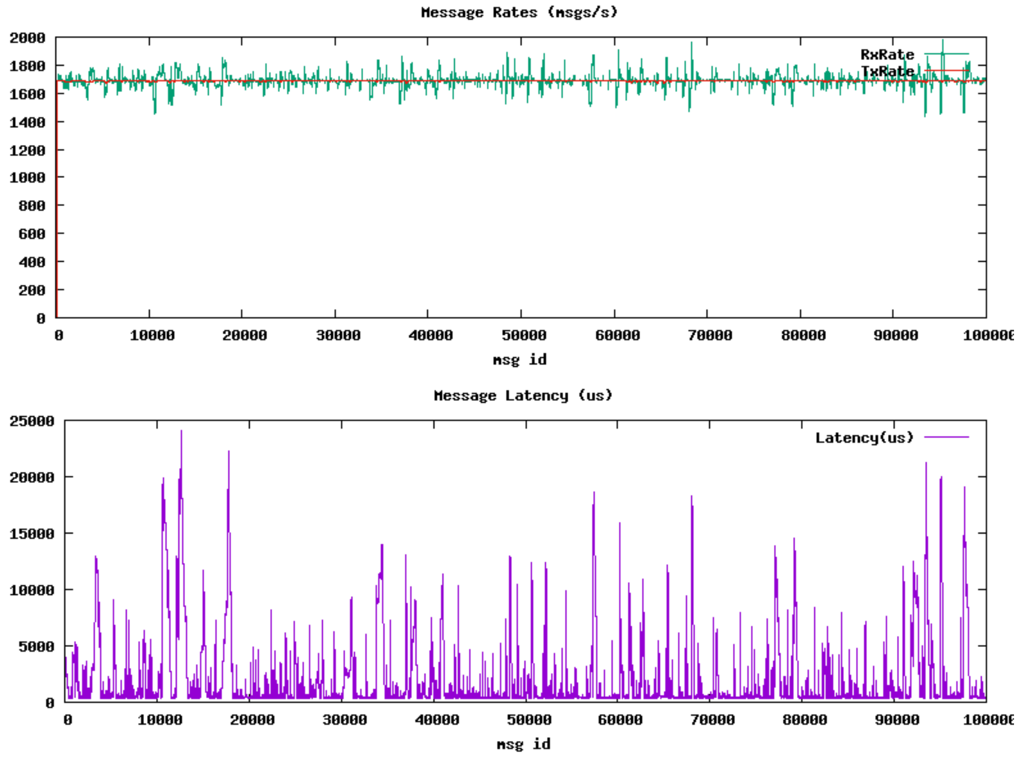
QoS0_100byte



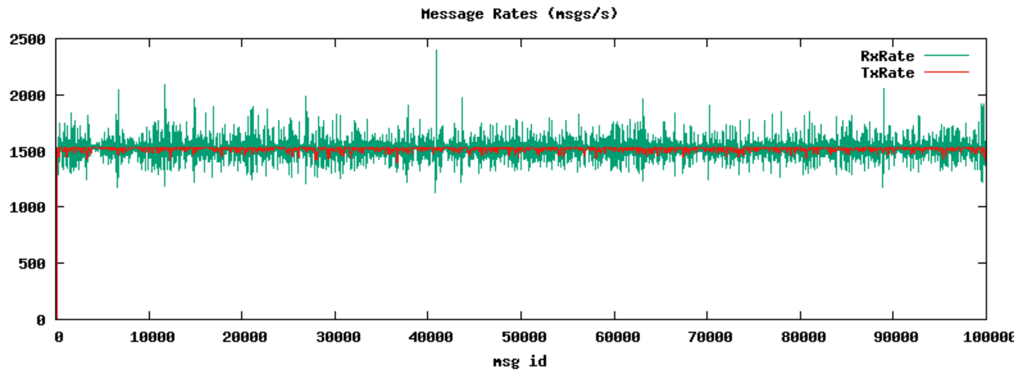
QoS1_100byte

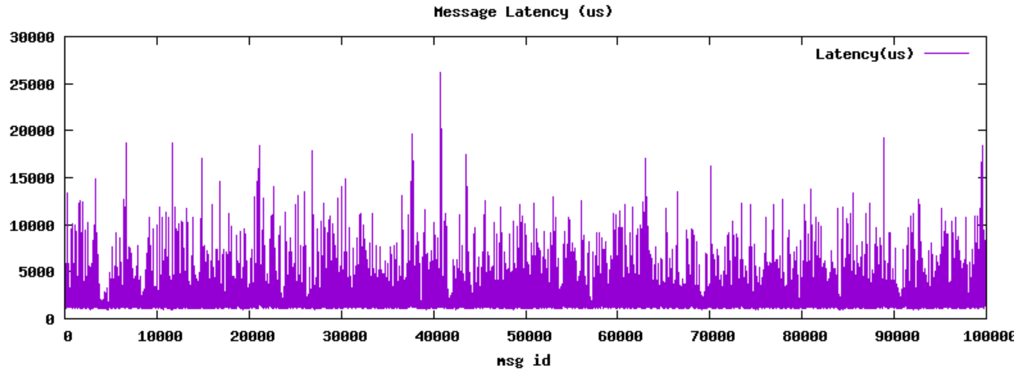


QoS2_100byte

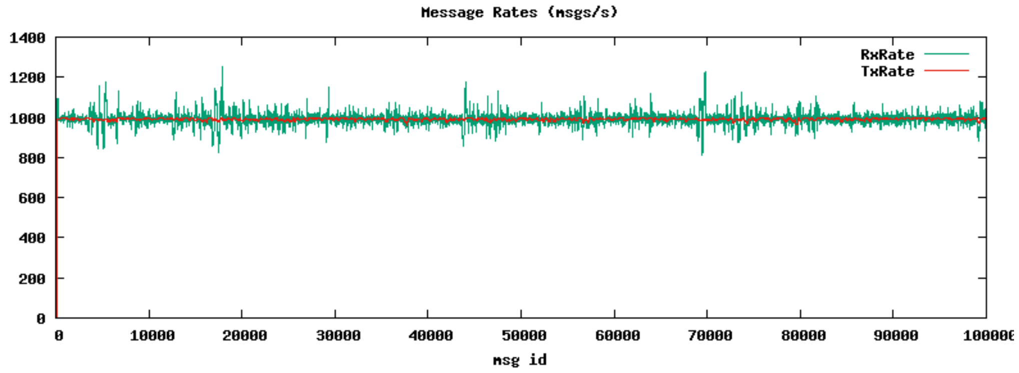


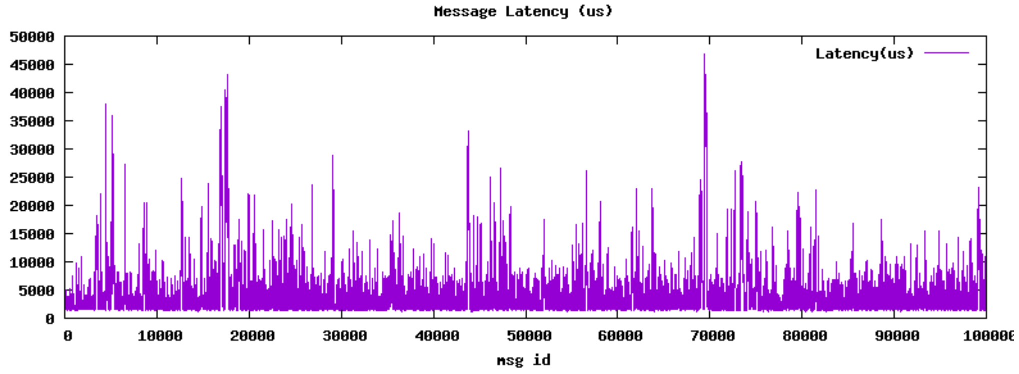
QoS0_64k



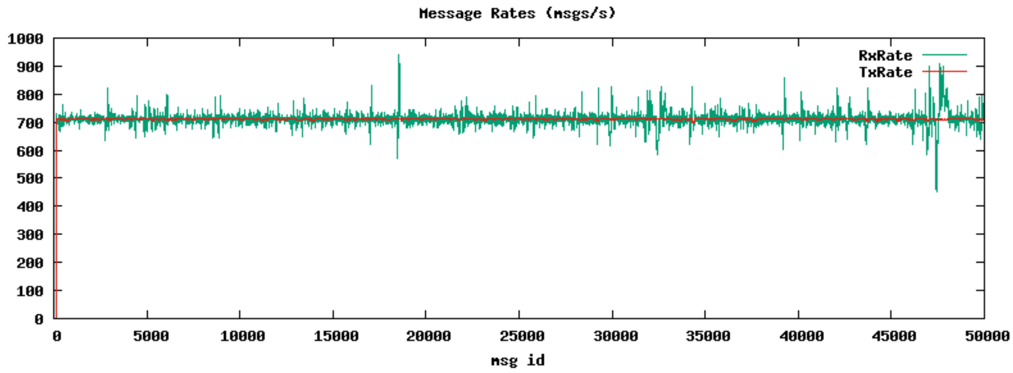


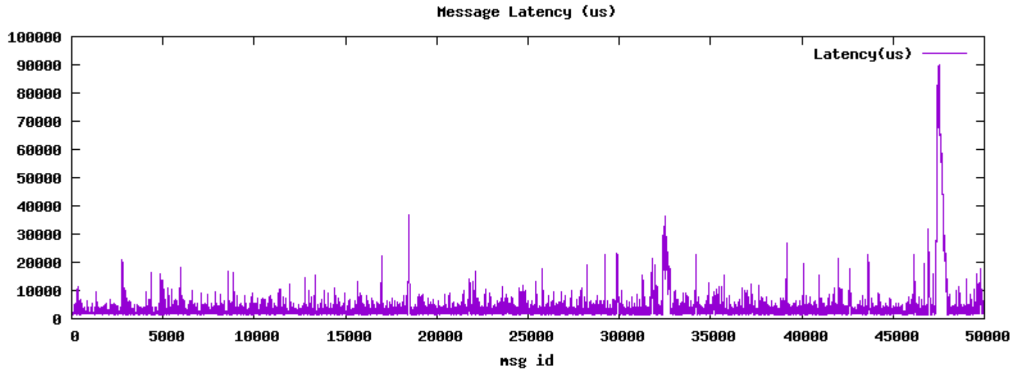
QoS1_64k





QoS2_64k

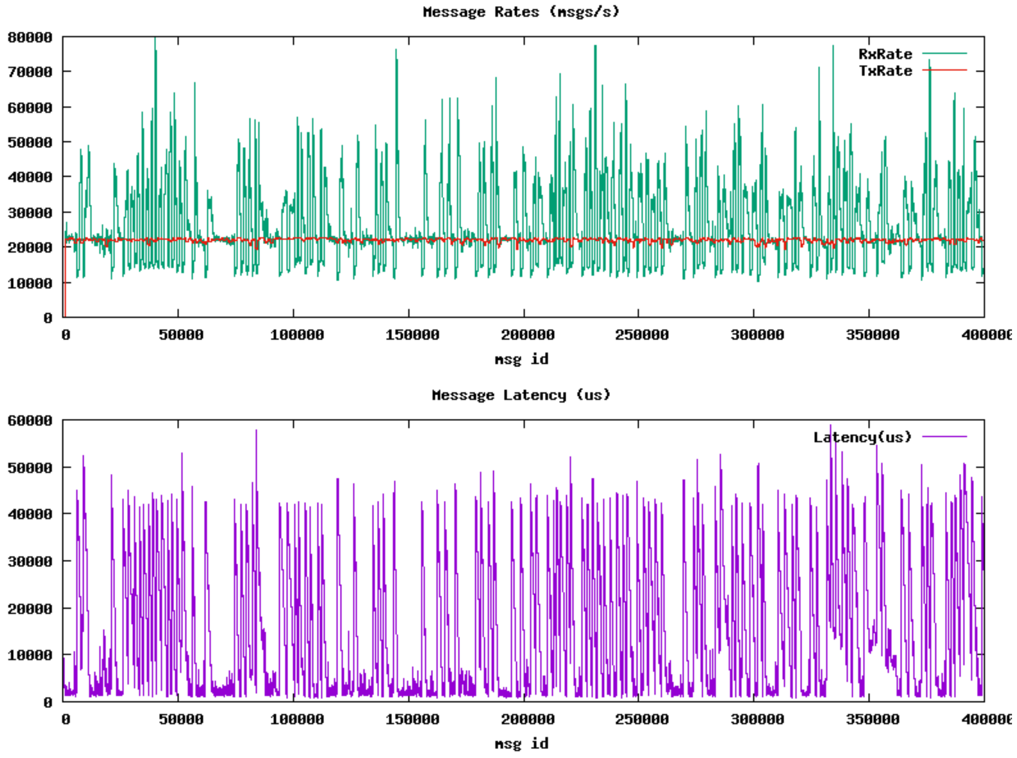

Message Rates (msgs/s)
RxRate
TxRate
1000
900
800
700
600
500
400
300
200
100
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id


Message Latency (us)
Latency(us)
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id

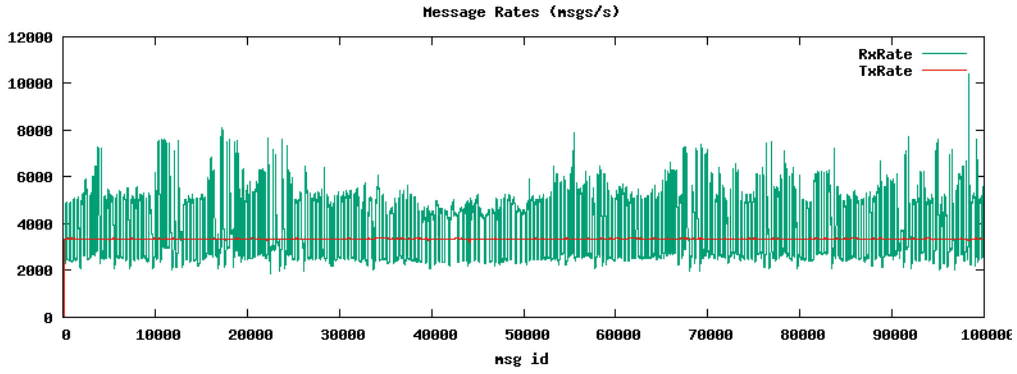
3.2.2 VerneMQ with "Cluster HA improvements"

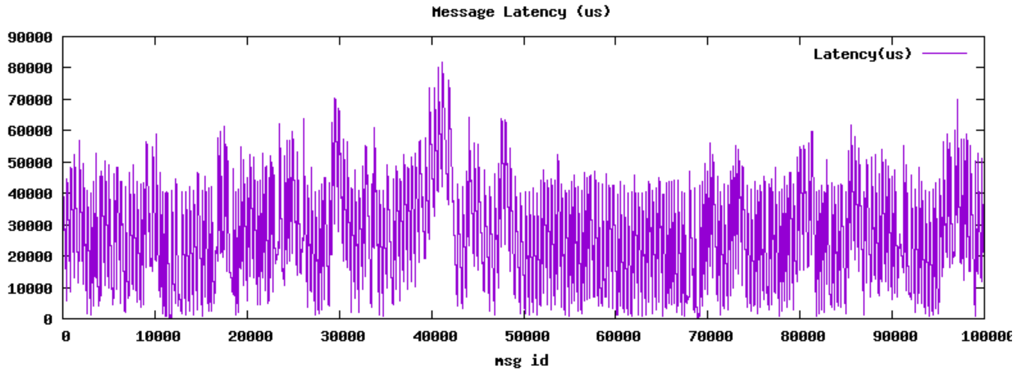
HA fix	**100 Byte**		**64 KByte**	
QoS	msgs/s	Mbit/s	msgs/s	Mbit/s
0	21907	17.5	1492	764
1	3356	2.7	828	424
2	2072	1.7	664	340

QoS0_100byte

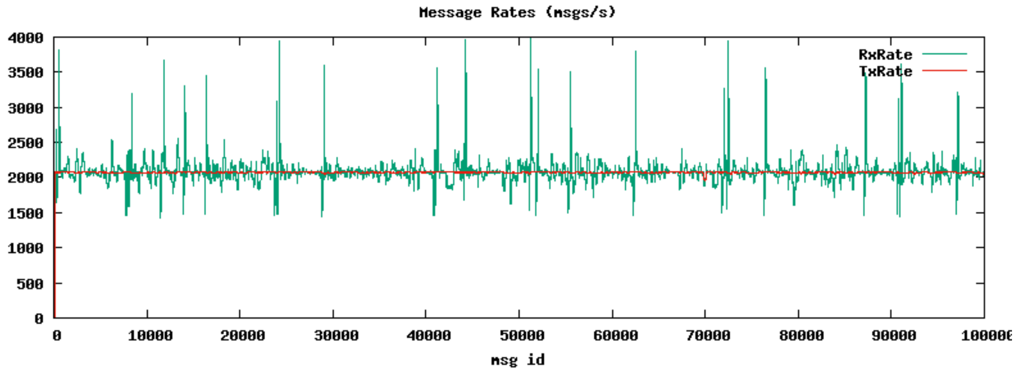


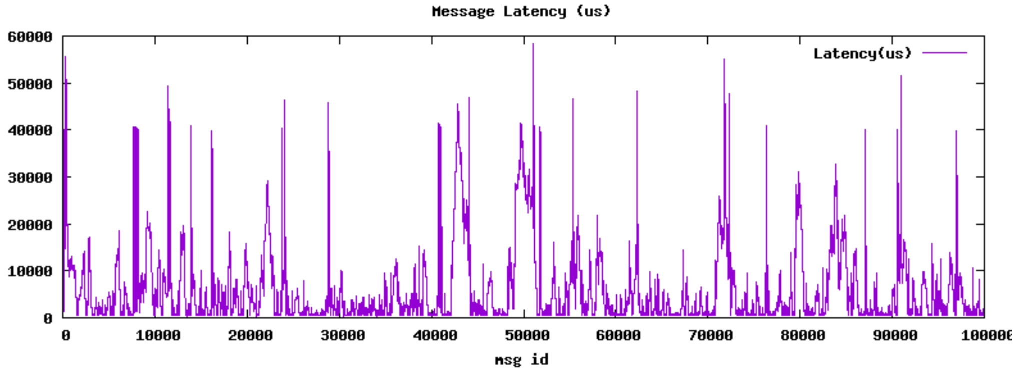
QoS1_100byte



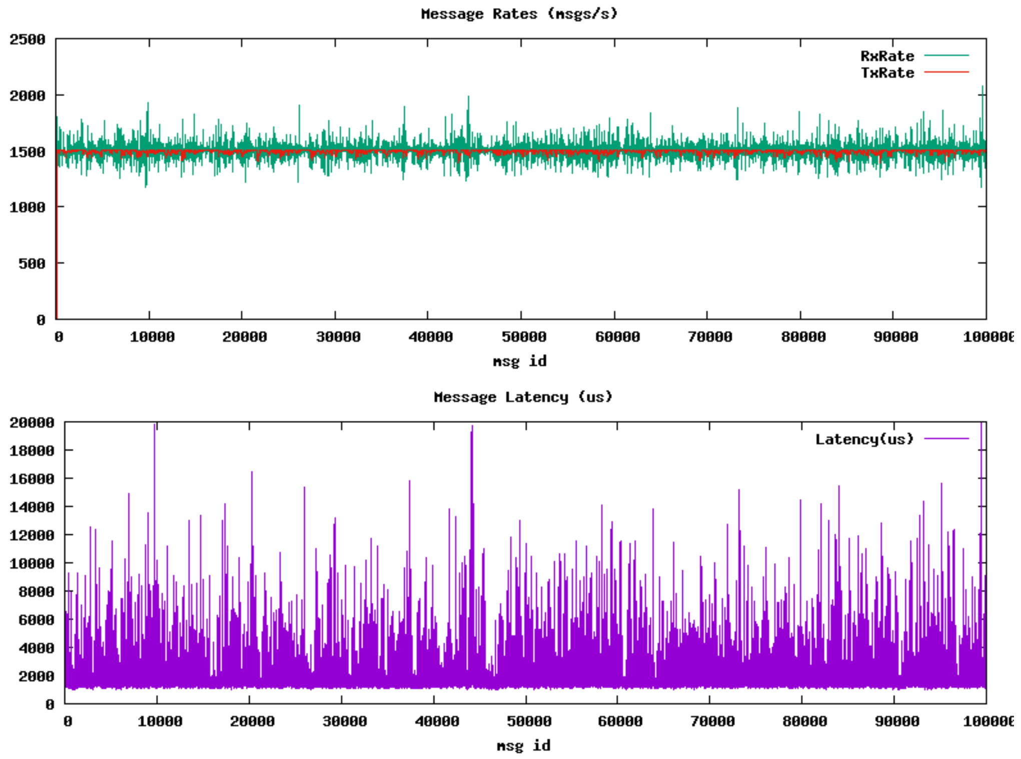


QoS2_100byte

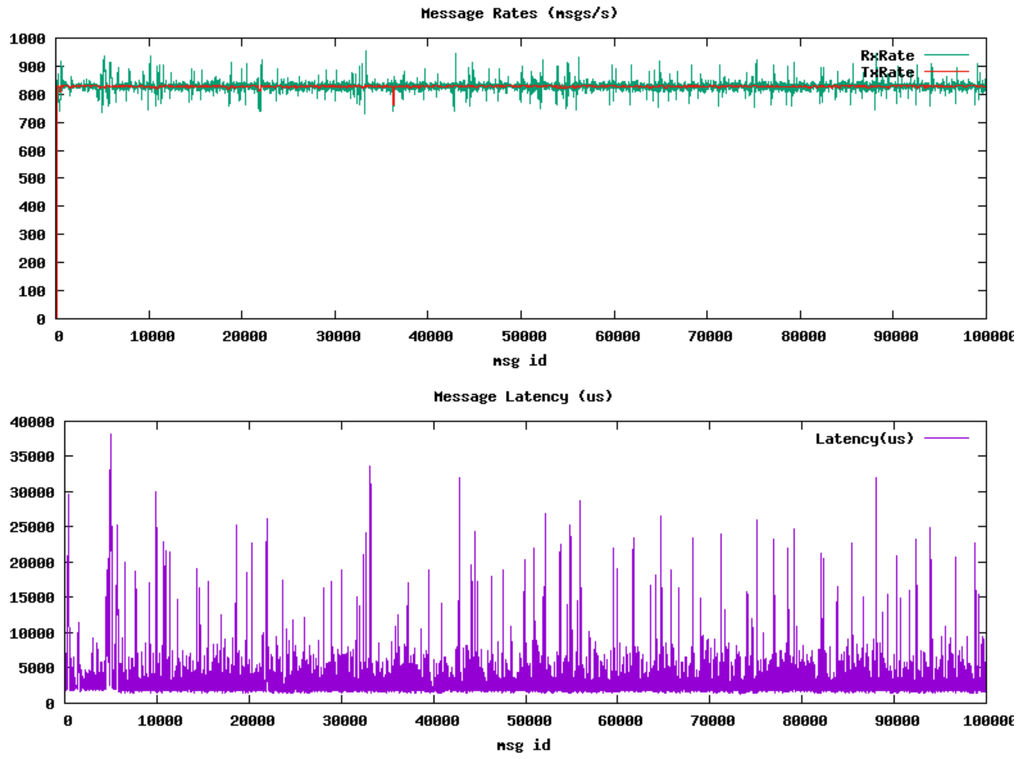




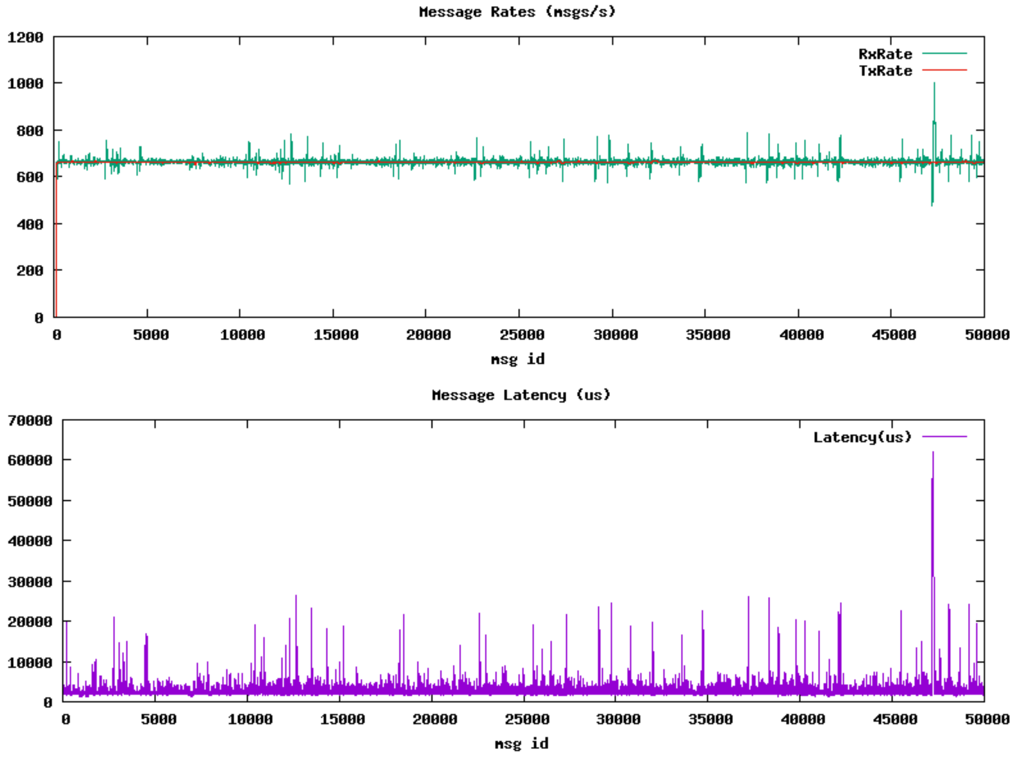
QoS0_64k

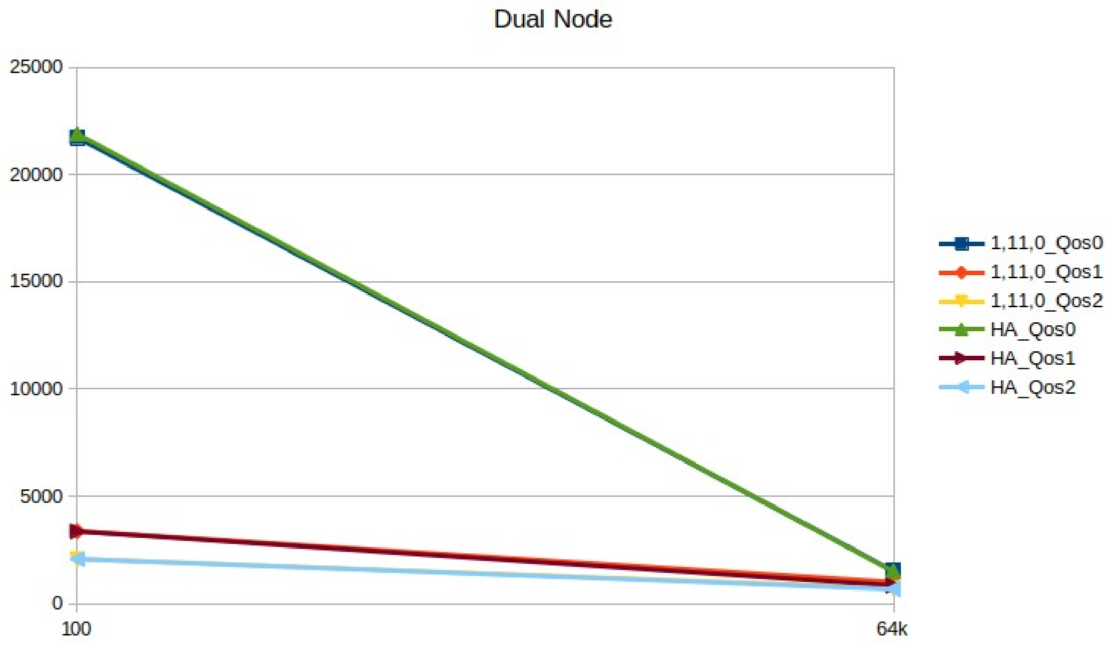


QoS1_64k



QoS2_64k


Message Rates (msgs/s)
RxRate
TxRate
1200
1000
800
600
400
200
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id
Message Latency (us)
Latency(us)
70000
60000
50000
40000
30000
20000
10000
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
msg id


Dual Node
25000
20000
15000
10000
5000
0
100
64k
1,11,0_Qos0
1,11,0_Qos1
1,11,0_Qos2
HA_Qos0
HA_Qos1
HA_Qos2

4 Conclusions

We can only compare relative performance between configurations in this setup because of too many differences with the original setup in MQTT Broker Benchmarking.

This is directly obvious when comparing the absolute performance (QoS0, single node) :
=> In the k8s-based deployment the measured max rates are about 30% lower than in the original setup.

Note also that from the original tests we know that with small message sizes the max rate was actually limited by the TCP connection.

We calculate the relative performance loss introduced by the HA fixes, compared to VerneMQ 1.11.0 release :

Relative performance loss when connected to single node :

Single	**100 Byte**	**64 KByte**
QoS	perf. loss	perf. loss
0	2.3 %	3.8 %
1	28.7 %	16.4 %
2	18.6 %	11.6 %

Relative performance loss when connected to different nodes of a 2-node cluster :

Dual	**100 Byte**	**64 KByte**
QoS	perf. loss	perf. loss
0	-0.8 %	1.5 %
1	0.3 %	16.5 %
2	0.0 %	6.6 %

- The HA fixes indeed have an impact on max throughput performance
- In QoS0 only a slight performance decrease can be seen
- Performance decrease in QoS1 and QoS2 is significantly higher
- The performance decrease is higher when publisher and subscriber are connected to the same broker
  With small message size there is almost no performance loss when connected to different brokers.