RMR_receive_perf

RMR Receiver Performance (on top of NNG)

Using two RMR/NNG based sending applications, and a varied number of receiving applications an attempt to qualify the maximum receive rate of an application. Testing from this point of view is underway; the following are some preliminary observations.

Prelim results

```
sender2_1  | <A2SEND> finished attempted: 1000000  good: 981279   bad: 0  drops: 18721 rate: 20833
sender1_1  | <A2SEND> finished attempted: 1000000  good: 980455   bad: 0  drops: 19545 rate: 21276
app0_1     | =app0= finished  received: 1961734  rate: 38465 msg/sec


sender1_1  | <A2SEND> finished attempted: 1000000  good: 961262   bad: 0  drops: 38738 rate: 15151
sender2_1  | <A2SEND> finished attempted: 1000000  good: 960658   bad: 0  drops: 39342 rate: 14925
sender3_1  | <A2SEND> finished attempted: 1000000  good: 959527   bad: 0  drops: 40473 rate: 14925
app0_1     | =app0= finished  received: 2881447  rate: 39471 msg/sec



sender1_1  | <A2SEND> finished attempted: 1000000  good: 962591   bad: 0  drops: 37409 rate: 14925
sender2_1  | <A2SEND> finished attempted: 1000000  good: 961256   bad: 0  drops: 38744 rate: 14925
sender3_1  | <A2SEND> finished attempted: 1000000  good: 961673   bad: 0  drops: 38327 rate: 14925
app0_1     | =app0= finished  received: 2885520  rate: 39527 msg/sec
```

As expected, the more sending applications are targeting a single receiver, the higher each sender's drop rate will be. We assume that the cause is the in ability to keep up on the receiver side, and TCP buffers are being exhausted.