RocketBufs: A Framework for Building Efficient, In-Memory, Message-Oriented Middleware

Huy Hoang, Benjamin Cassell, Tim Brecht, Samer Al-Kiswany
Message-Oriented Middleware (MOM)
Message-Oriented Middleware

- Apache Kafka
- RabbitMQ
- Redis
- Amazon Kinesis
- Google Cloud Pub/Sub
Message-Oriented Middleware

• Lots of MOM systems
• Yet, new systems still being built: 6 new projects in last 5 years
• Not using advanced data center networking technologies
  • RDMA, DPDK, TCPDirect
    • APIs and abstractions different from TCP, hard to program,
    • Difficult to support multiple networking technologies
Goals of RocketBufs

- Natural abstractions and easy-to-use APIs for MOM systems
- Support for different networking technologies / transport protocols
  - Add new technologies to RocketBufs
  - Applications access them with no changes to code
- Support for flexible MOM topologies
- Resulting applications are efficient and scalable
RocketBufs

Application

RocketBufs

RocketNet

Networking
RocketBufs

Application

RocketBufs

RocketNet

Networking

Kernel Sockets

TCP

ibverbs

RDMA
RocketBufs

Application

RocketBufs

RocketNet

Networking

Kernel Sockets
TCP

ibverbs
RDMA

F-stack
DPDK

Others
RocketBufs

Application

RocketBufs

RocketNet

Networking

Kernel Sockets
TCP

ibverbs
RDMA

F-stack
DPDK

Others
Message-Oriented Middleware (MOM)

Messages copied from output buffers on one host to input buffers on another
RocketBufs

Application

RocketBufs

rIn buffer

rOut buffer

RocketNet

Networking

Kernel Sockets

TCP

ibverbs

RDMA

F-stack

DPDK

Others
Use Case Pub/Sub System: RBMQ

- Topics mapped to buffer
- Publishers send messages using rOut
- Brokers and subscribers receive messages using rIn
- Broker uses buffer splicing to forward messages (w/o copying)
RBMQ Evaluation

2.0 GHz Intel Xeon, 8 cores, 1x 40 Gbps NIC

2.6 GHz Intel Xeon, 10 cores, 4x 40 Gbps NICs

2.0 GHz Intel Xeon, 8 cores, 1x 40 Gbps NIC
RBMQ Configurations

Publisher - rOut

Data Center

Broker

rIn

splice

splice

rOut

rIn

Subscriber

rIn

Subscriber

rIn

Subscriber

TCP

TCP

RBMQ-tcp-tcp

RBMQ-tcp-no-fc
RBMQ Configurations

TCP

Publisher

rOut

Publisher

rOut

RDMA

rIn

Broker

splice

rOut

rIn

splice

rIn

Subscriber

rIn

Subscriber

rIn

Subscriber

Data Center

RBMQ-tcp-rdma
RBMQ Configurations

![Diagram showing RBMQ configurations with RDMA connections]

Data Center

RBMQ-rdma-rdma
RBMQ Comparison

Redis (pub/sub mode)

- Used by Twitter, Github, StackOverflow
- Brokers, subscribers, publishers implemented in C

RabbitMQ

- Used by Reddit, VMWare, Mozilla
- Brokers implemented in Erlang (RabbitMQ is in Erlang)
- Publishers, subscribers implemented in C
RBMQ Comparison

Redis (pub/sub mode)
- Used by Twitter, Github, StackOverflow
- Brokers, subscribers, publishers implemented in C

RabbitMQ
- Used by Reddit, VMWare, Mozilla
- Brokers implemented in Erlang (RabbitMQ is in Erlang)
- Publishers, subscribers implemented in C

Broker disseminates all messages to all subscribers

Measure: Throughput, CPU utilization, latency (see paper)
System Throughput (4 subscribers)

![Graph showing system throughput with message size on the x-axis and throughput (msgs/s x 10^3) on the y-axis. The graph compares RabbitMQ, RBMQ-tcp-no-fc, RBMQ-tcp-rdma, RBMQ-rdma-rdma, and Redis.]
System Throughput (4 subscribers)

![Graph showing system throughput vs message size for different message sizes and byte counts.](image)

- **Tput** (msgs/s x 10^3)
- **Goodput** (Gbps)

- **RabbitMQ**
- **RBMQ-tcp-no-fc**
- **RBMQ-rdma-rdma**
- **Redis**
- **RBMQ-tcp-rdma**
System Throughput (4 subscribers)

![Graph showing system throughput and goodput vs. message size (bytes) for不同 systems with various message sizes and throughput metrics in messages per second (msgs/s) and Gbps. The graph illustrates the performance of RabbitMQ, RBMQ-tcp-no-fc, RBMQ-rcma-rcma, Redis, and RBMQ-tcp-rcma.]
System Throughput (4 subscribers)

The graphs depict the system throughput and goodput for different message sizes and time parameters. The x-axis represents the message size in bytes, ranging from 8 to 512K, and the y-axis represents throughput (Tput) and goodput (Gbps) in messages per second ($10^3$) and Giga bits per second (Gbps), respectively.

**RabbitMQ** and **Redis** are compared, with RabbitMQ showing generally higher throughput and goodput across all message sizes compared to Redis. Specific markers and line styles are used to differentiate between the various tests and configurations, such as RabbitMQ-tcp-no-fc, RabbitMQ-tcp-rdma, and RabbitMQ-rdma-rdma.

The graphs illustrate the performance characteristics under different test conditions, highlighting the impact of message size on system performance.

*July 17, 2020*
System Throughput (4 subscribers)

![Graph showing System Throughput for different message sizes and protocols.](image-url)
Conclusions

RocketBufs: framework for high-performance MOM systems

• Memory-based buffer abstractions
• APIs for producing, disseminating, and subscribing to data
• Supports buffer splicing and flexible MOM topologies
• Supports multiple transport protocols (no application code changes)

Performance:

• Good TCP performance compared to production systems
• RDMA produces significant performance gains
Discussion Points (stuff not in the talk)

See the paper and Huy Hoang’s M.Math Thesis for:

- API details / code snippets
- Latency results
- Live streaming video application and performance results

Possible Future Work:

- Ports to other transports (e.g., DPDK, TLS, etc.)
- Modify an existing MOM to use RocketBufs (e.g., Redis)
- Fault tolerance and persistence