RabbitMQ Overview

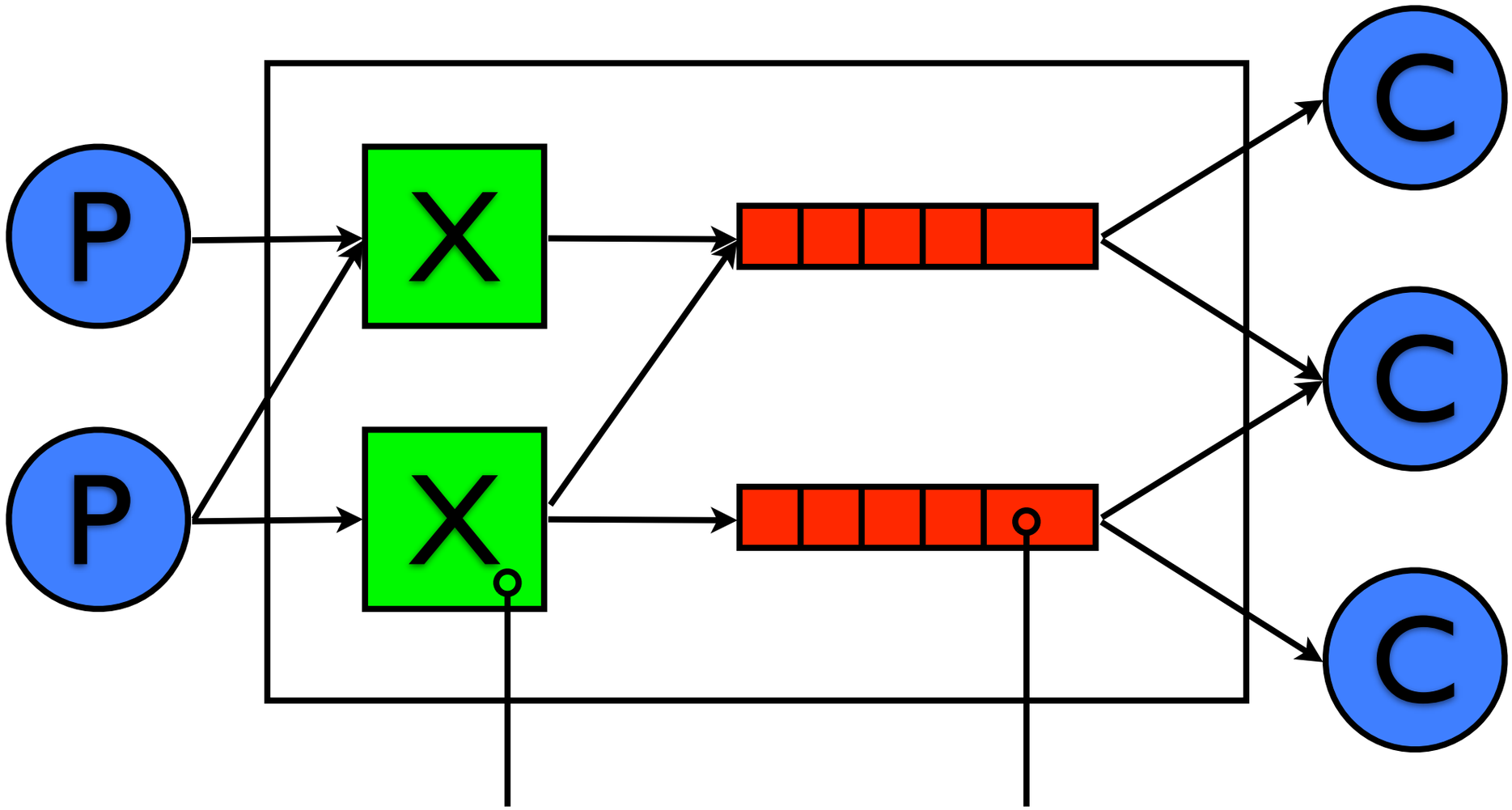
Tony Garnock-Jones <tonyg@lshift.net>

Agenda

- AMQP in 3 minutes
- RabbitMQ architecture
- Availability, Clustering, Federation
- Durability, Persistence, Memory usage
- Security
- Operational Tools
- Ongoing work

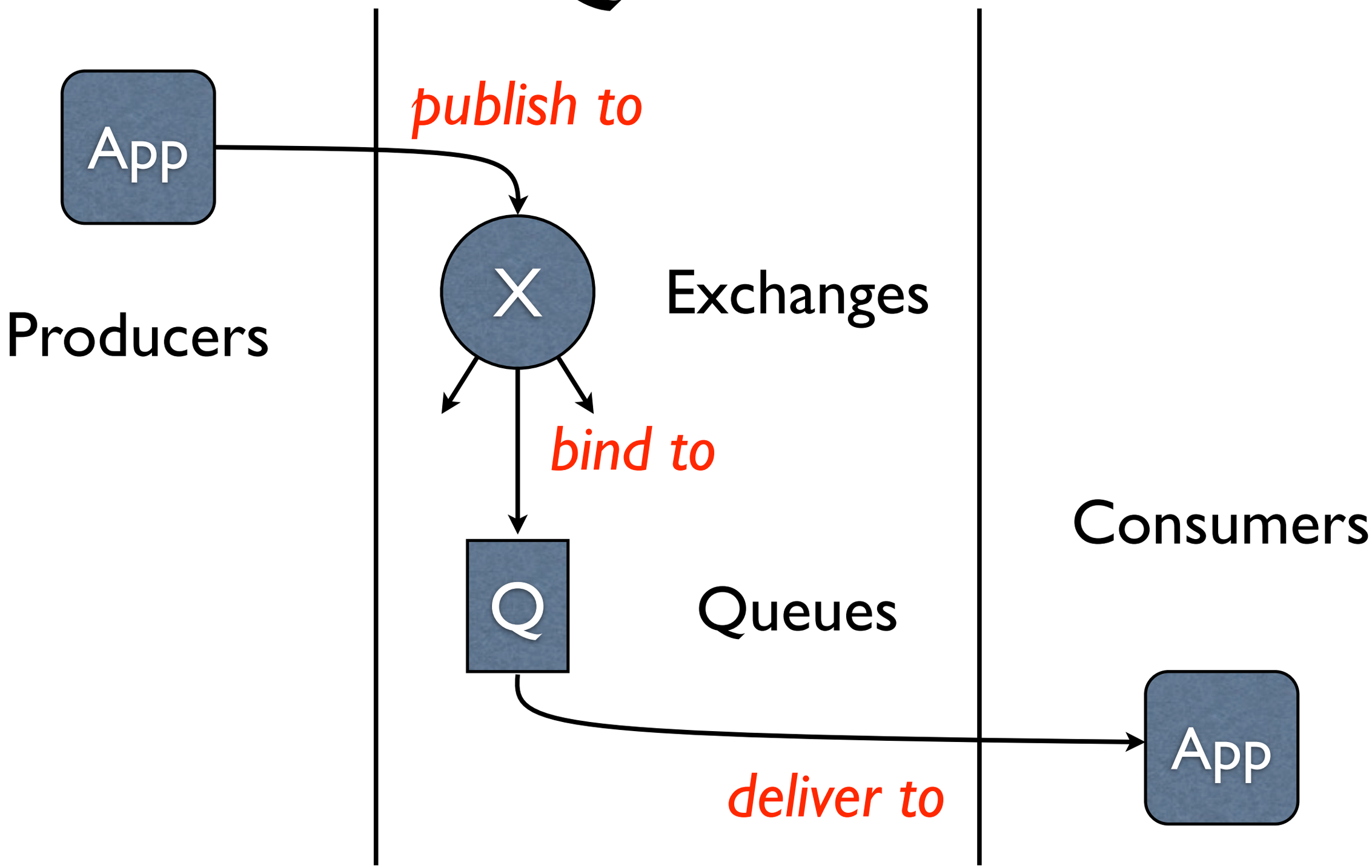
AMQP

AMQP Basics (0-9-1)

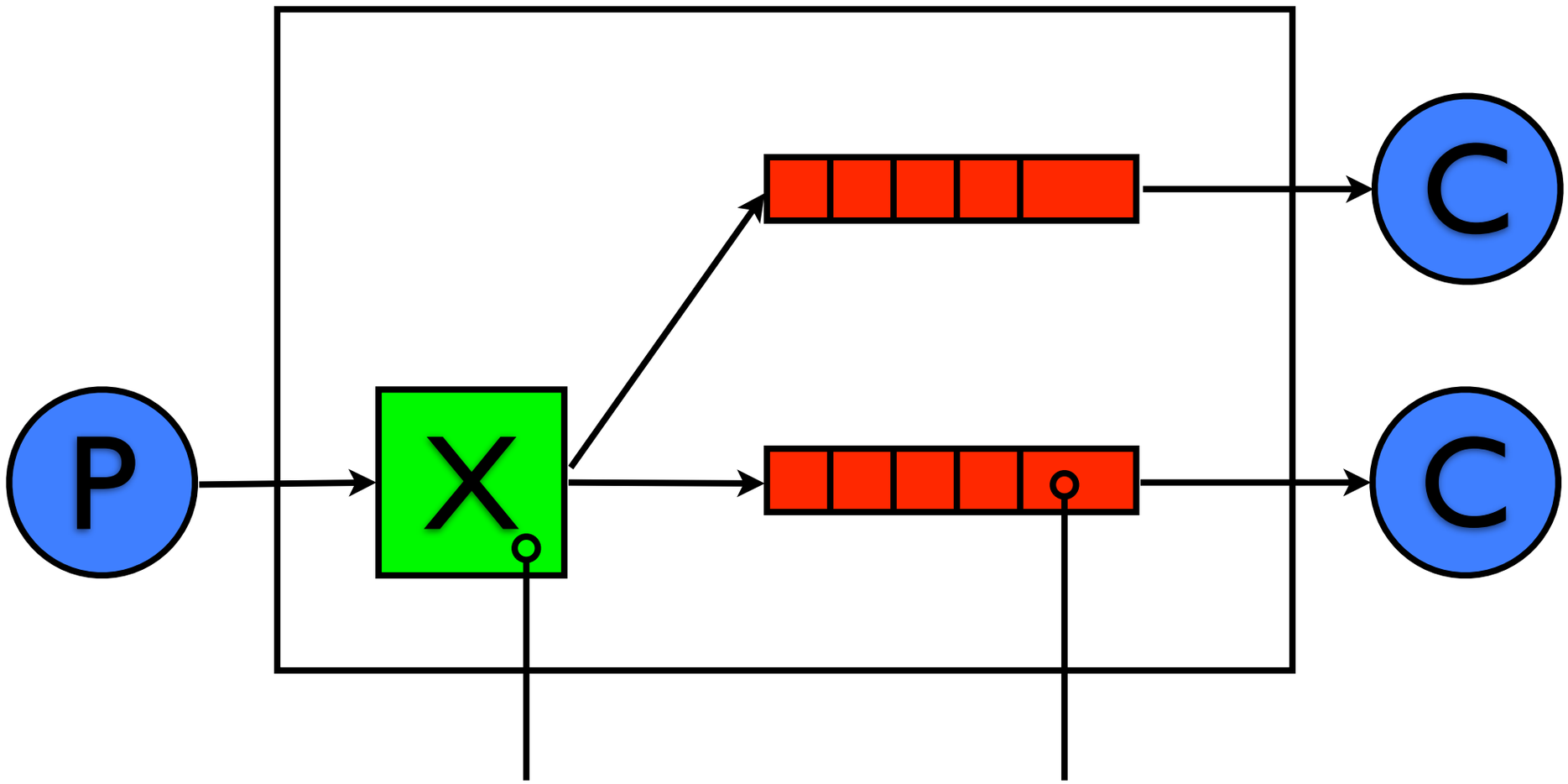


- Exchanges perform relaying, copying, and filtering
- Queues perform buffering and round-robin delivery

AMQP Basics (0-9-1)

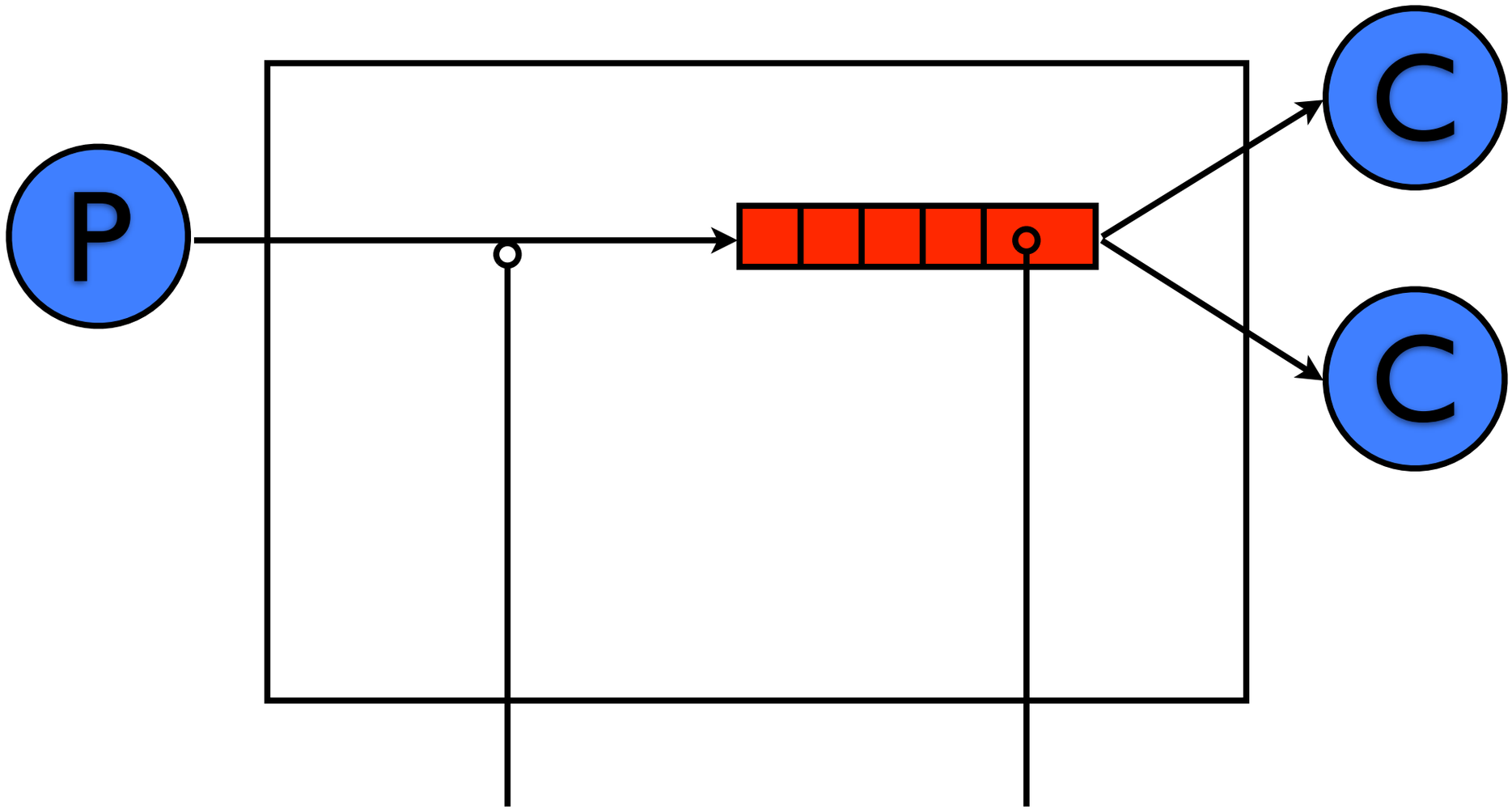


PubSub



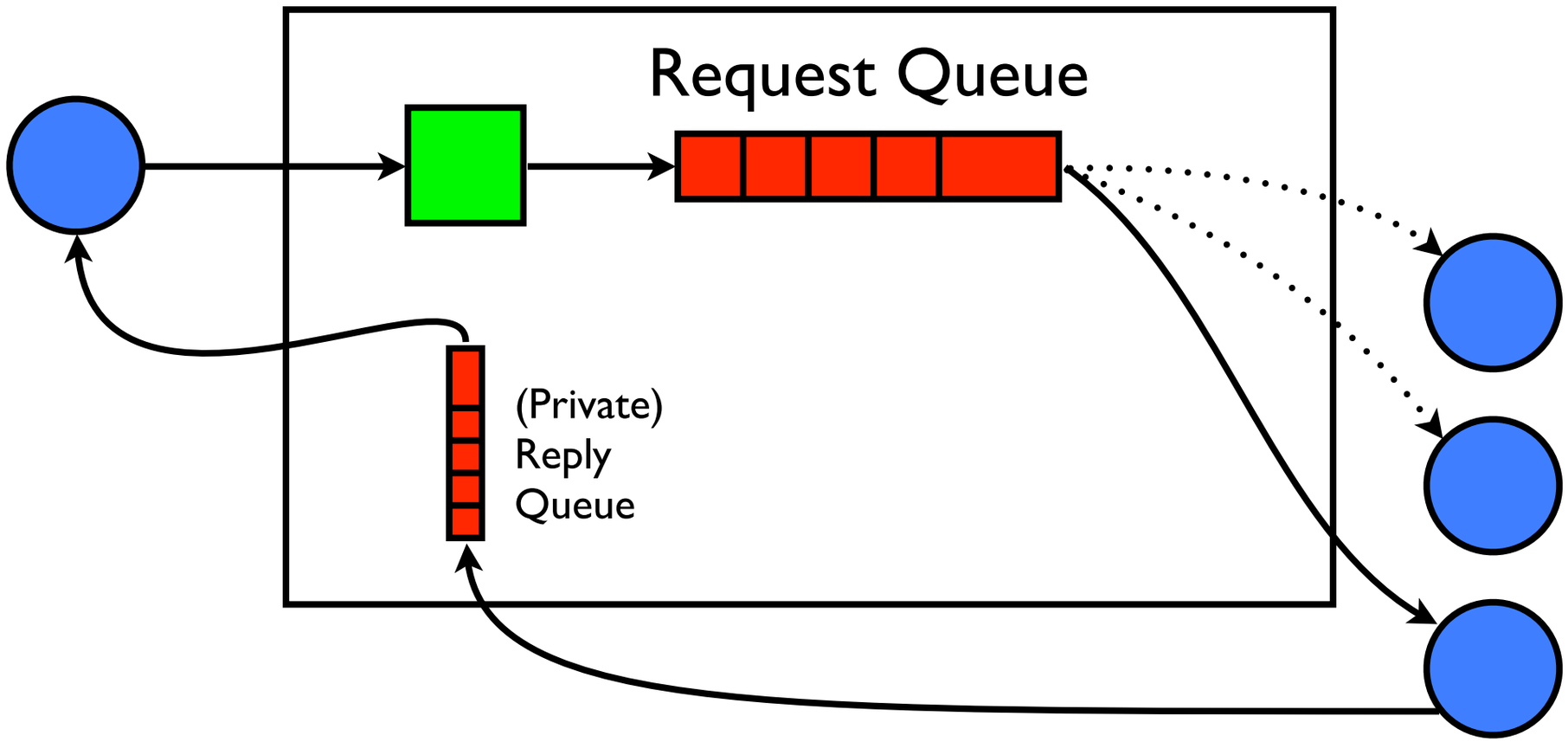
- AMQP exchange used as the destination copies to multiple queues
- Per-consumer private queues receive topical messages

Queueing



- Using the default exchange (“”) routes directly to queues
- Both shared and private queues can be addressed like this

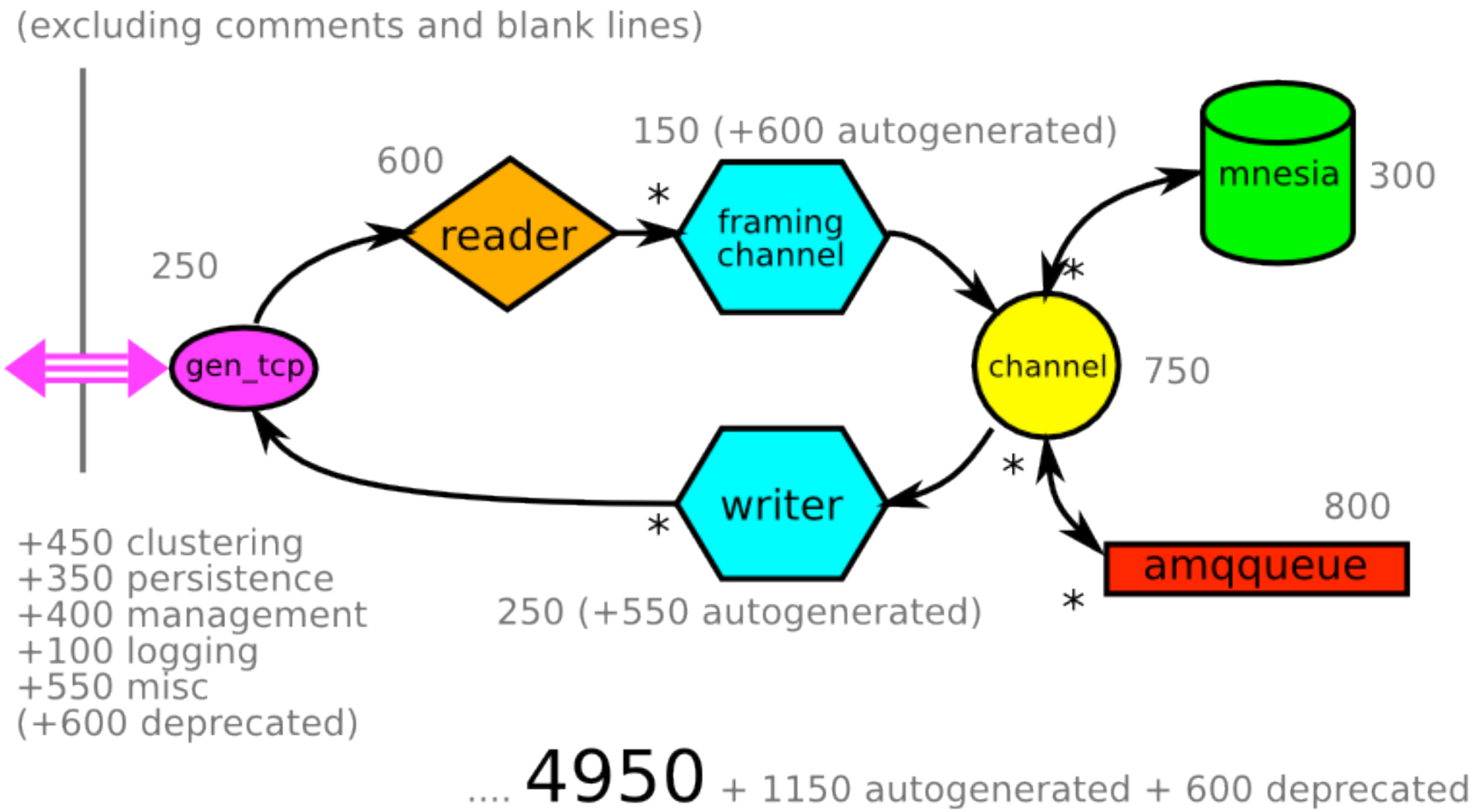
Simple load-balancing



- Shared queue mediates access to service instances
- Load-balancing, live upgrades, fault-tolerance

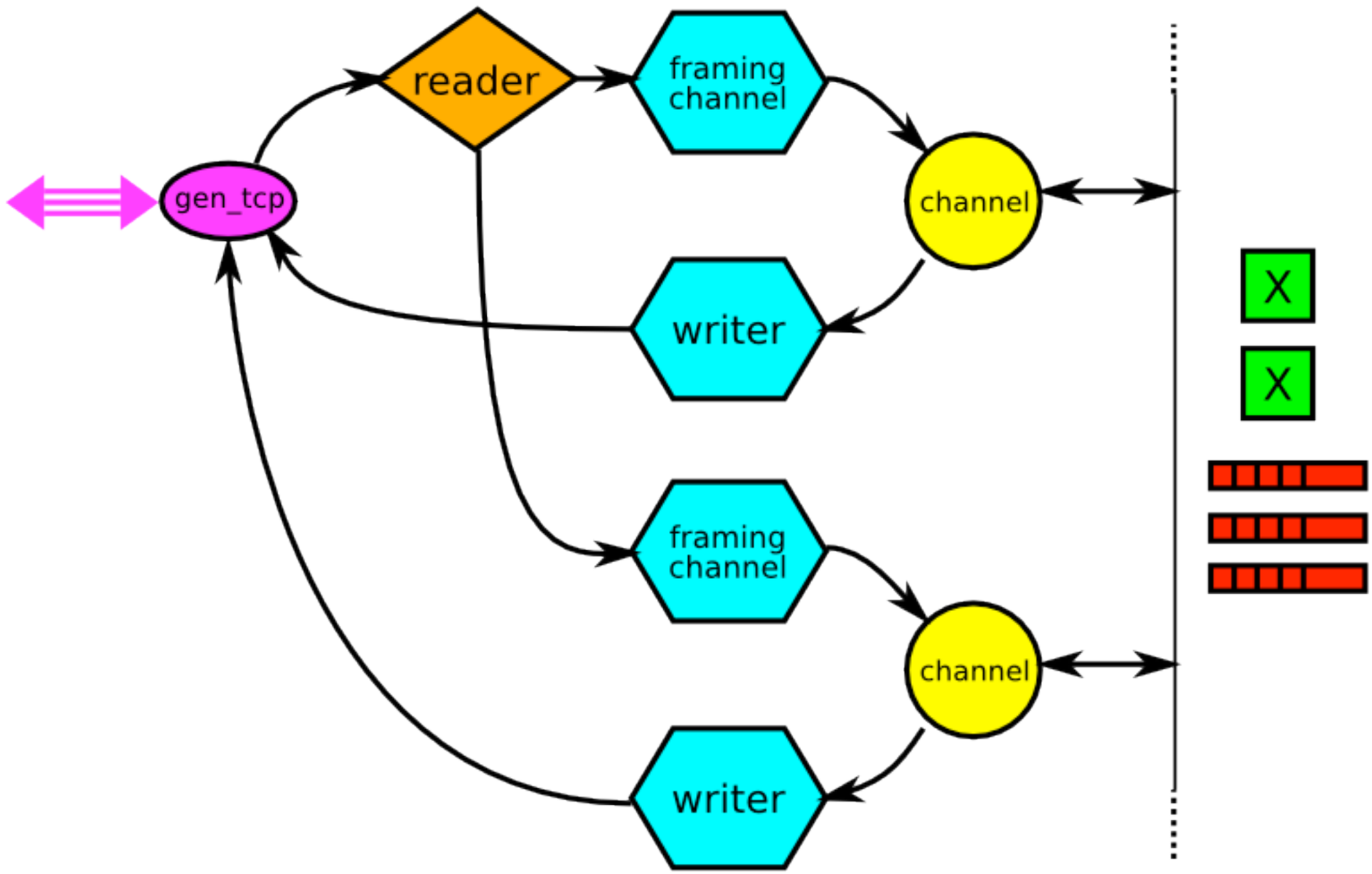
RabbitMQ

Code Overview

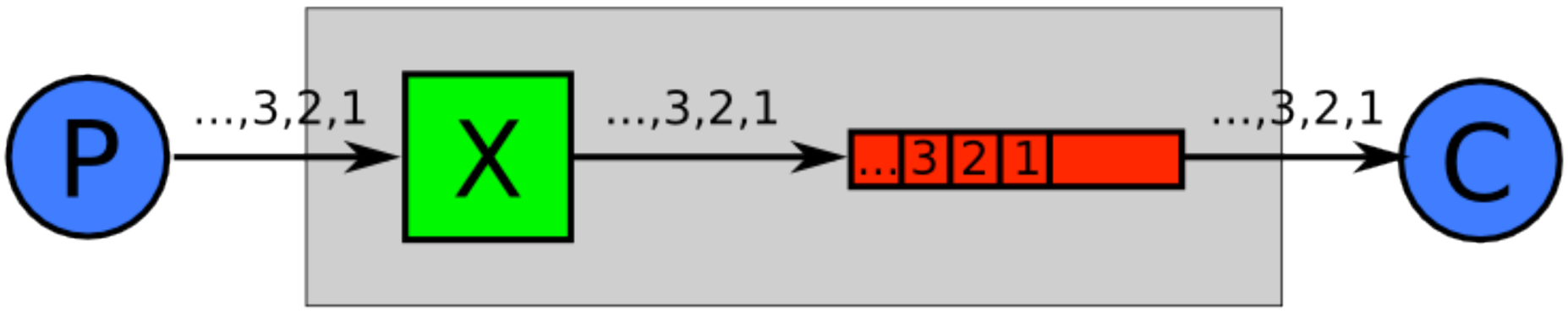


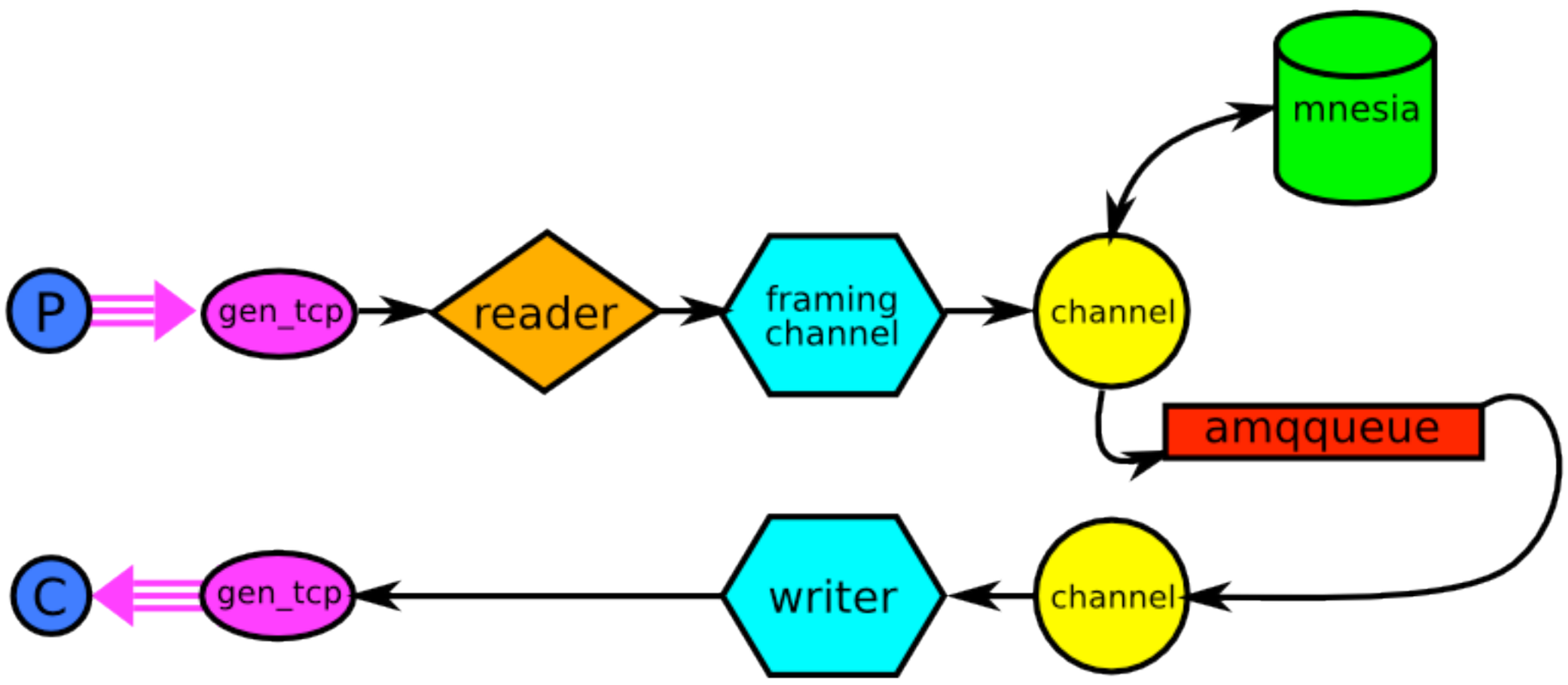
(out of date numbers! it's ~7500 LoC now, mid 2009)

Concurrency

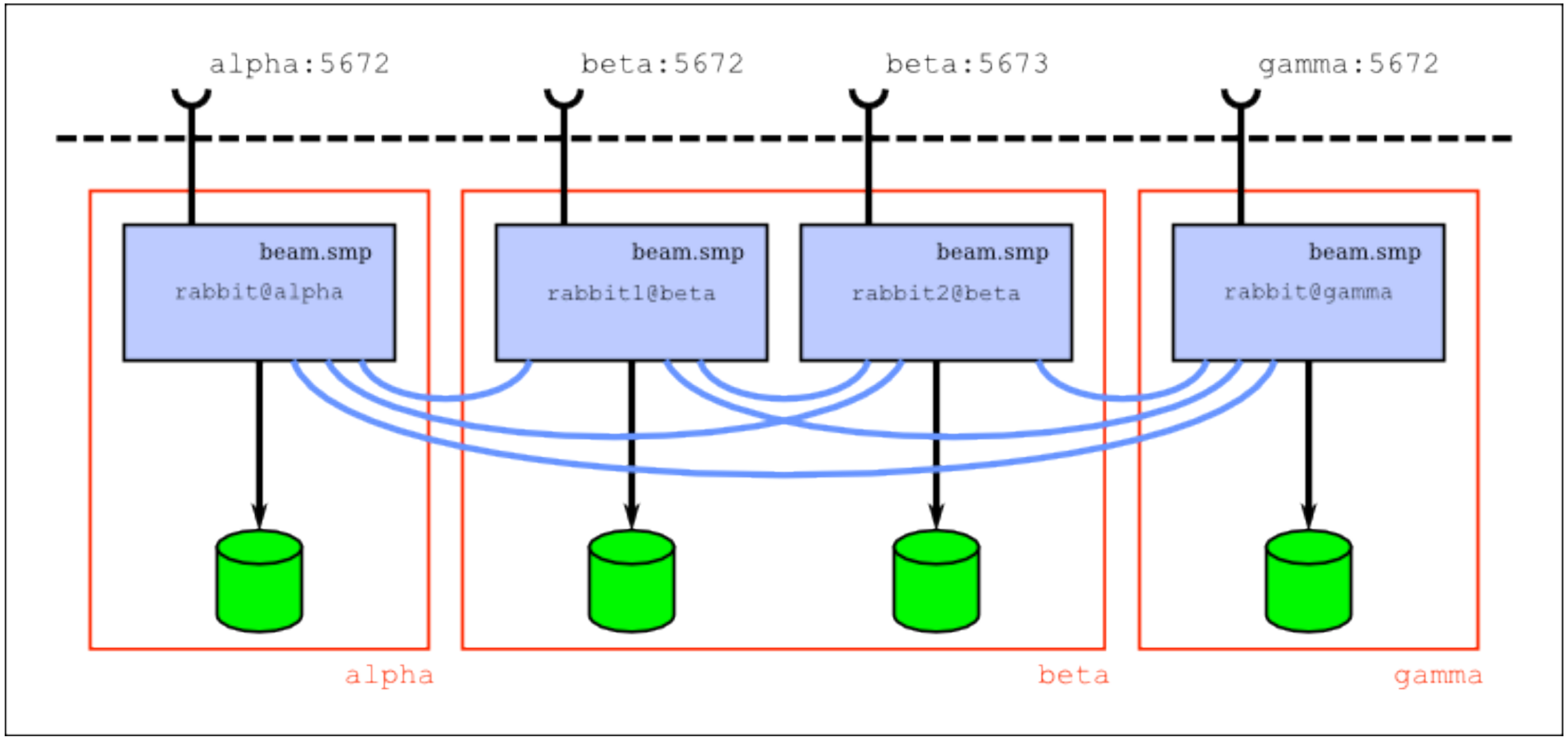


Order Preservation

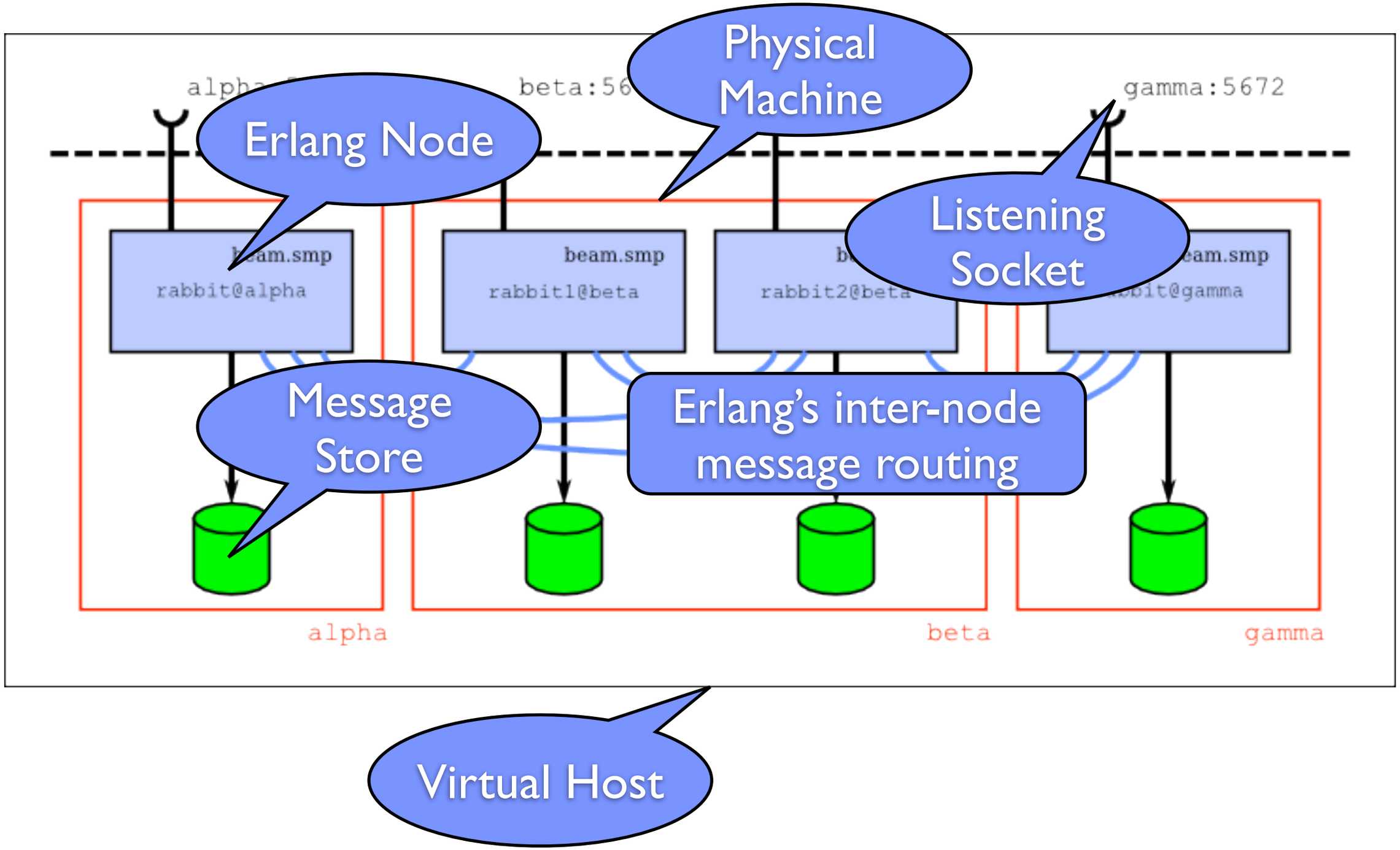




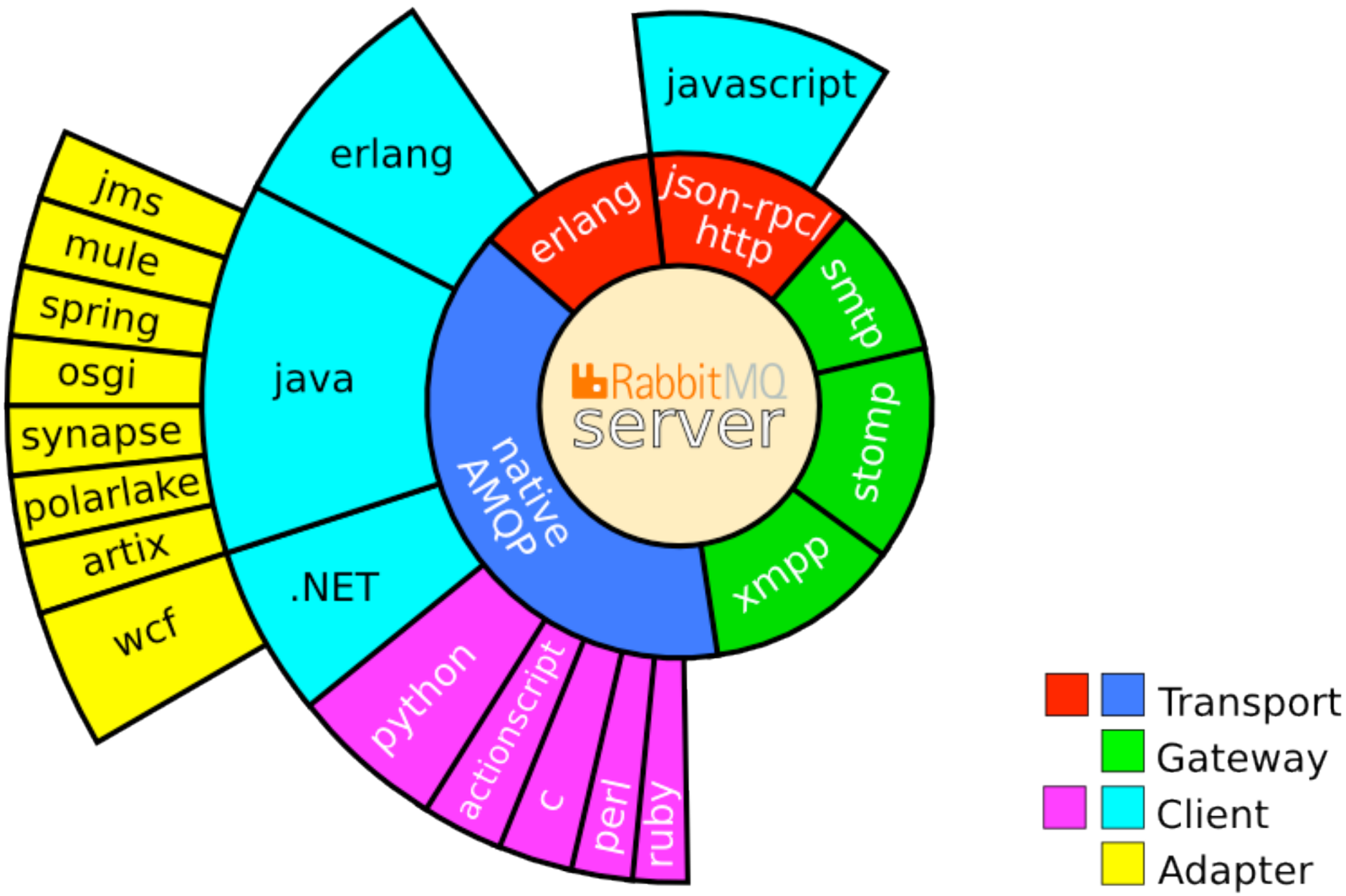
Clustering


alpha:5672
beta:5672
beta:5673
gamma:5672
beam.smp
rabbit@alpha
beam.smp
rabbit1@beta
beam.smp
rabbit2@beta
beam.smp
rabbit@gamma
alpha
beta
gamma

Clustering


Physical Machine
Erlang Node
Listening Socket
Message Store
Erlang's inter-node message routing
Virtual Host
beta:56
gamma:5672
beam.smp
rabbit@alpha
beam.smp
rabbit1@beta
rabbit2@beta
rabbit@gamma
alpha
beta
gamma

Connectivity



Availability, Clustering, Federation

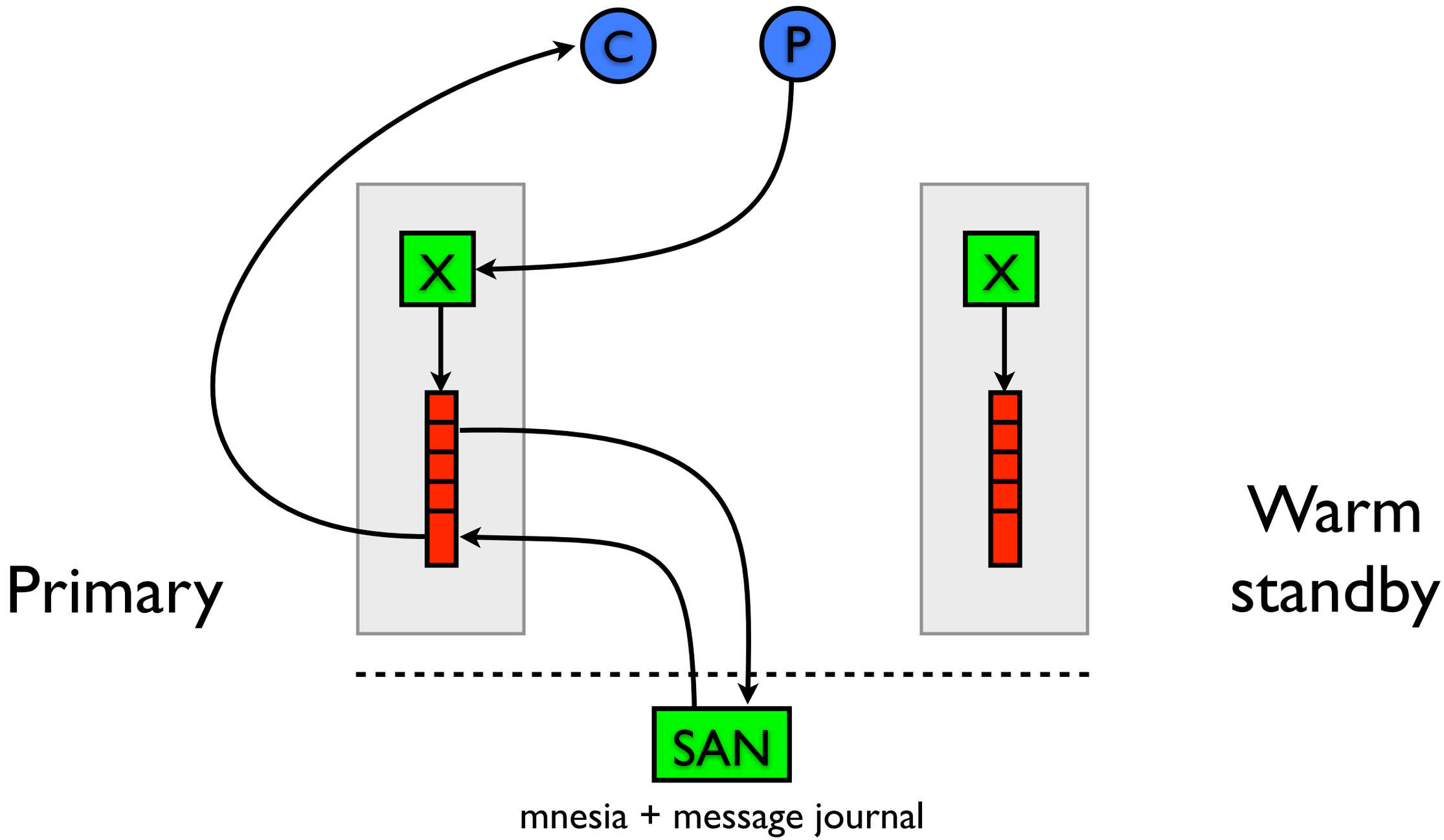
Failover

- Split "availability" into
  - Service availability: a broker's ready when you need one
  - Data availability: your persisted messages survive failures
- Short outage during failover; non-ack'd messages will need to be retransmitted
- Need better? Use redundant data paths

Failover

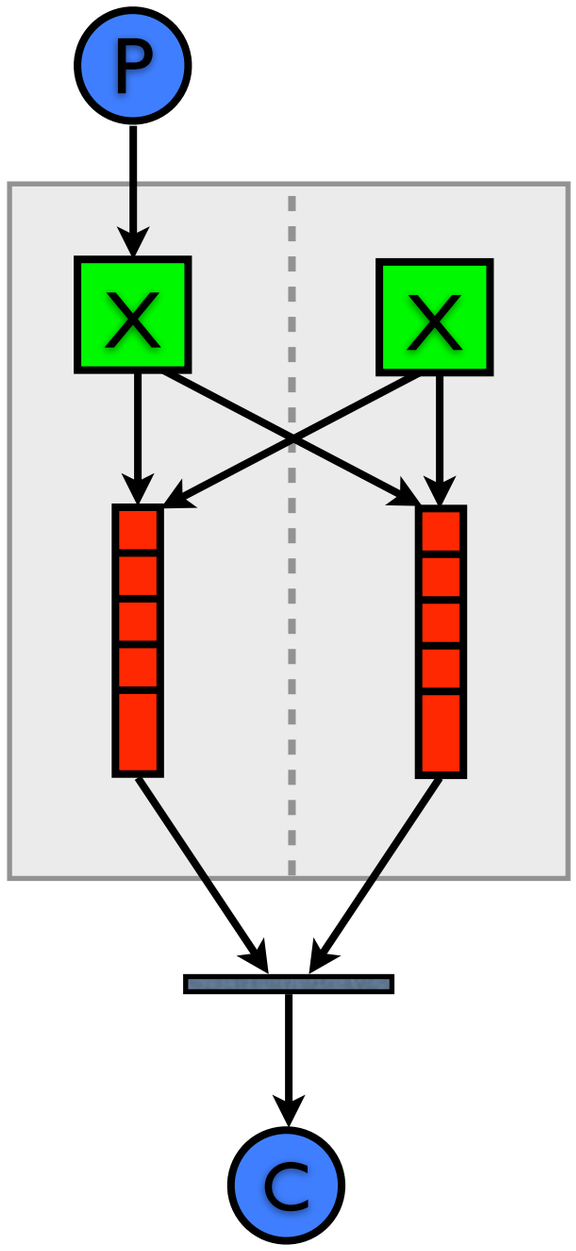
- Off-the-shelf components:
  - Networked fsync()able filesystem
  - Failure monitor: Linux-HA, ping + virtual ethernet, ...

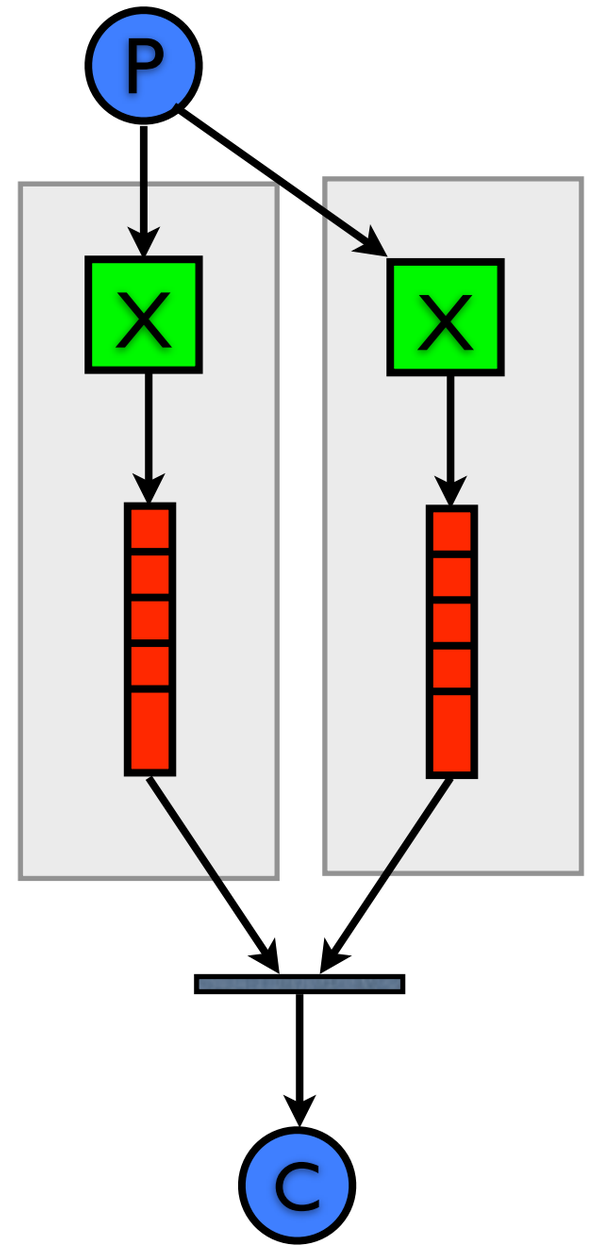
Failover



Redundancy for HA

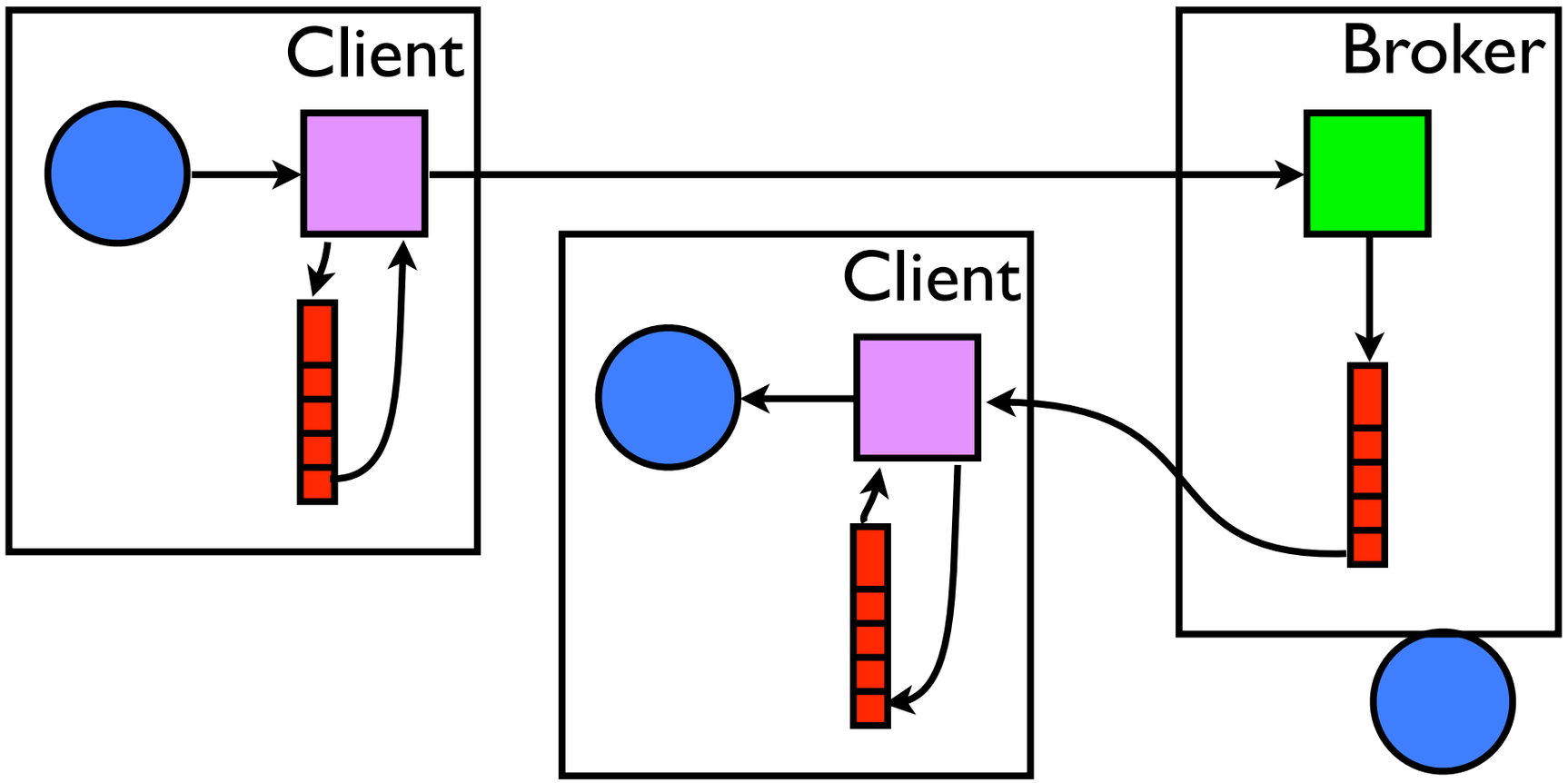
One broker, two nodes each





Two brokers, one node each

Exactly-Once Delivery



- "guaranteed delivery", even with intermittent links
- "auto-deduplication"

Paul Baran's Networks

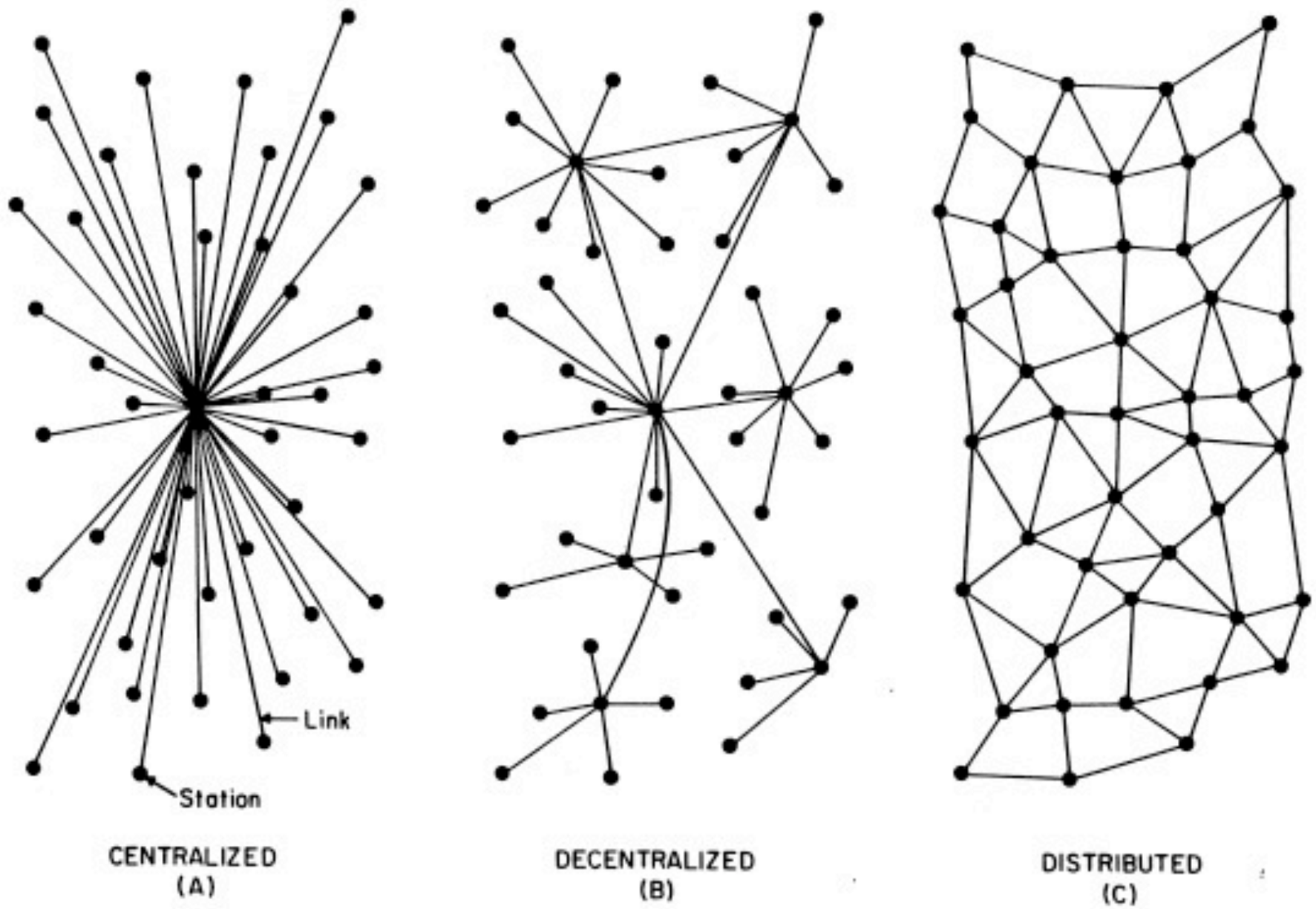
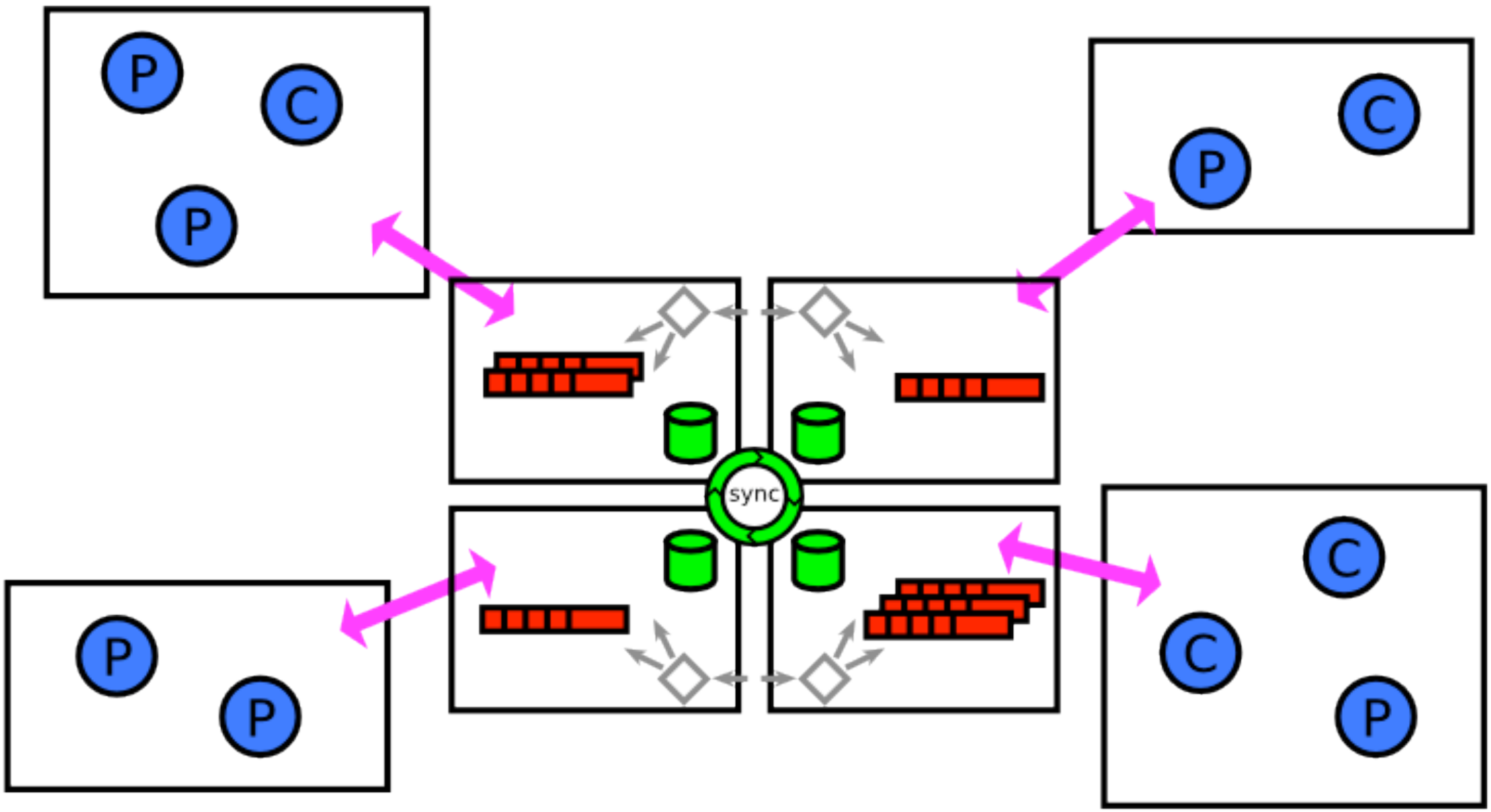


FIG. 1 — Centralized, Decentralized and Distributed Networks

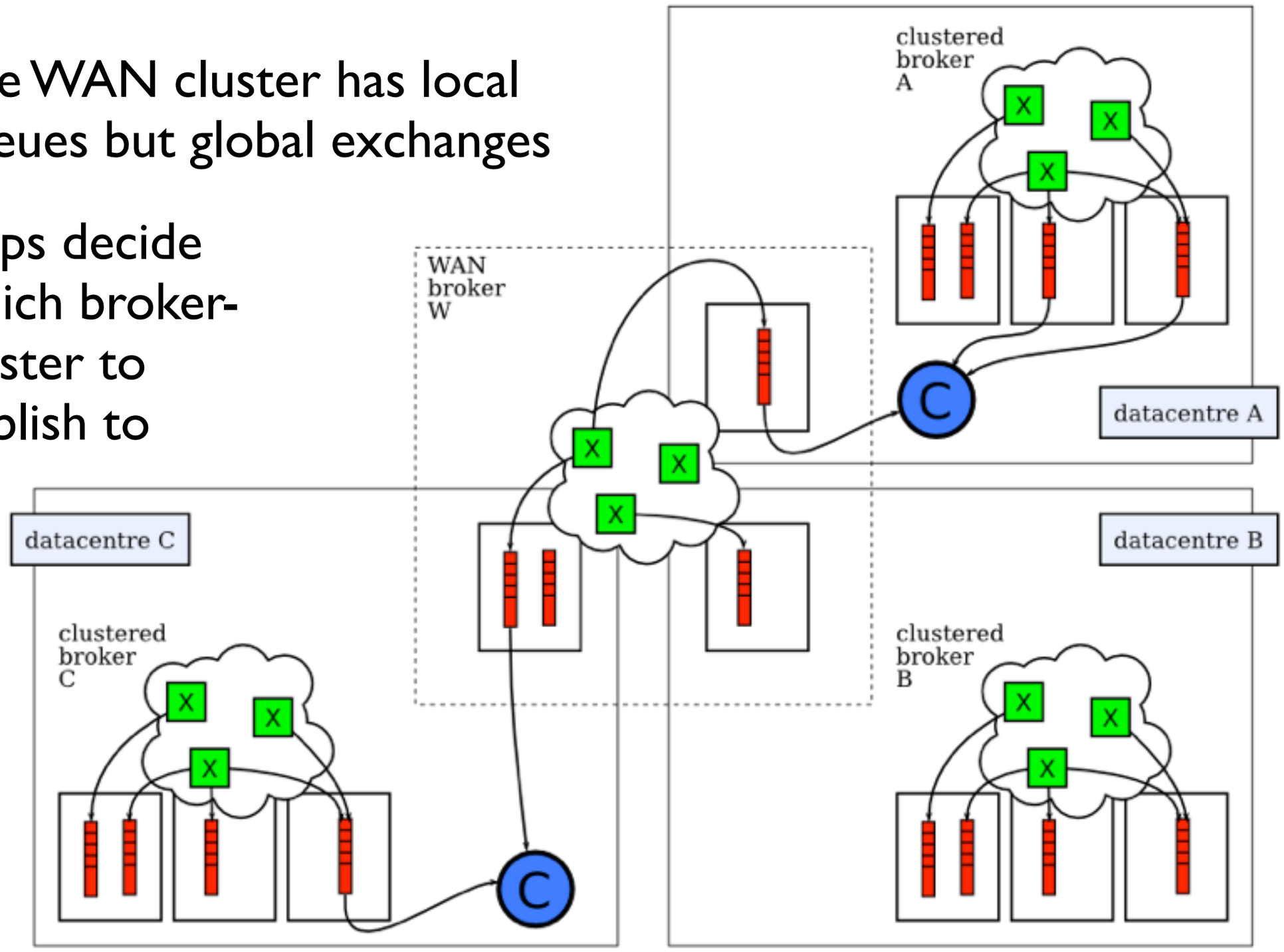
Clustering



WAN Cluster

The WAN cluster has local queues but global exchanges

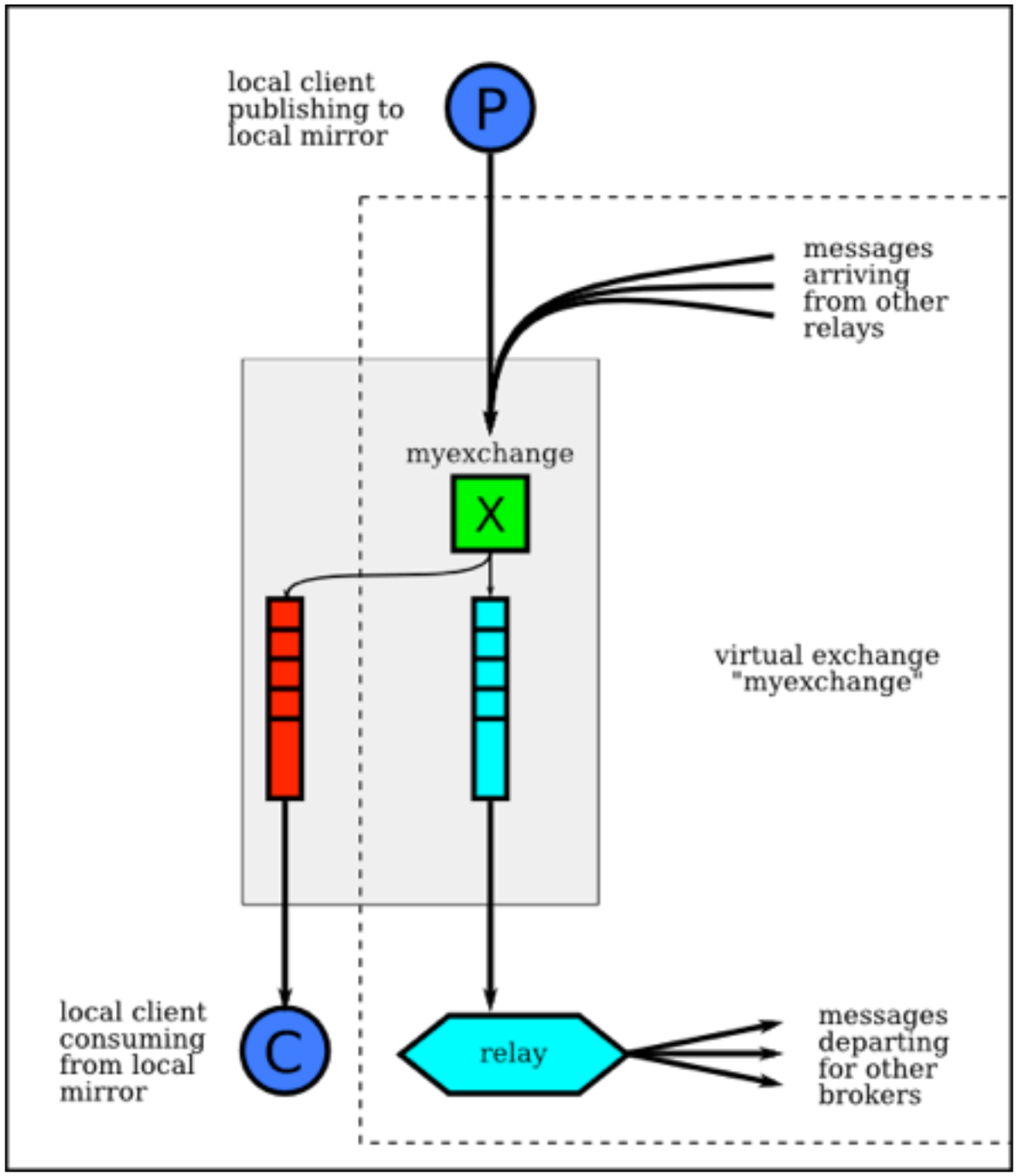
Apps decide which broker-cluster to publish to



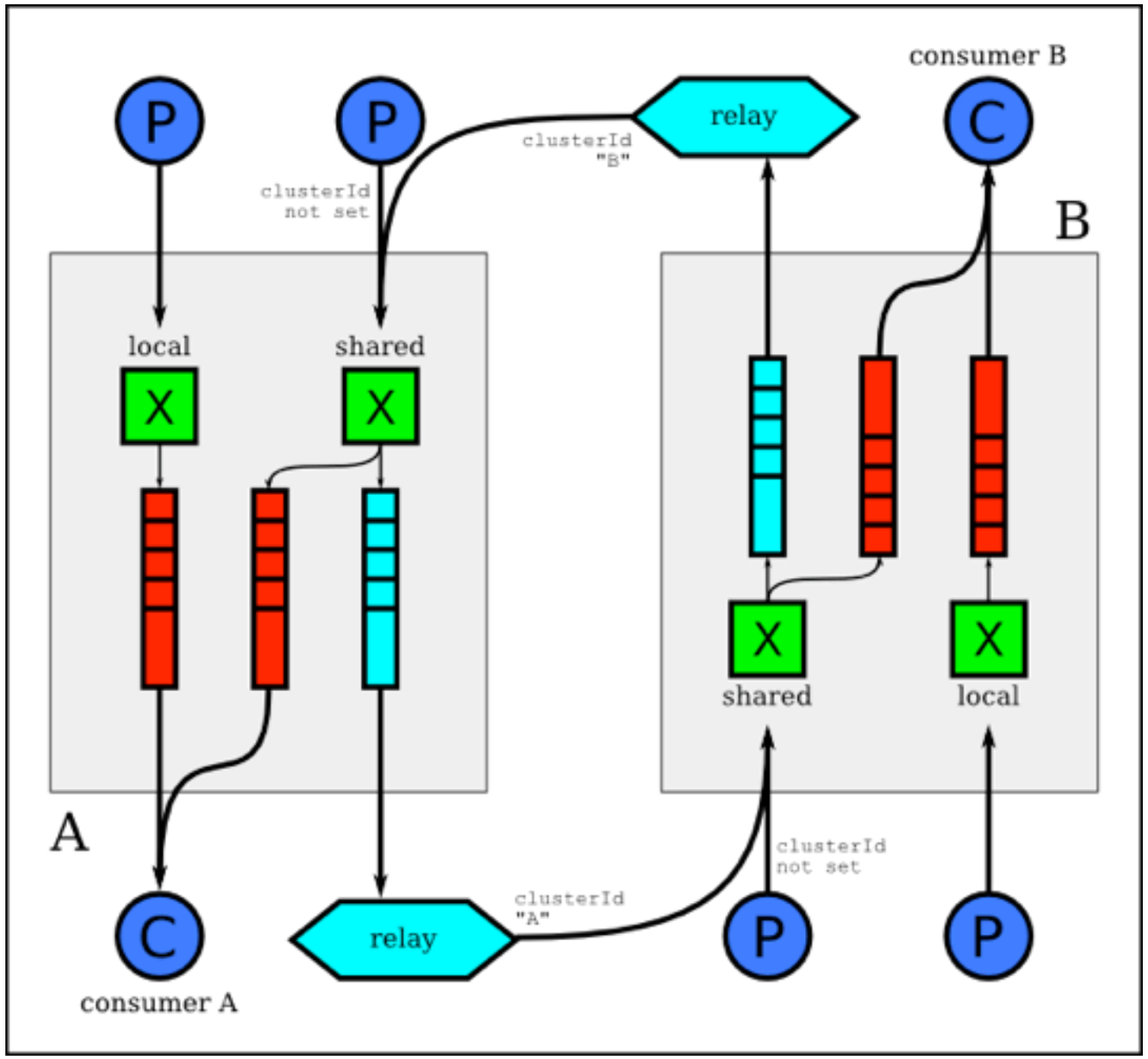
Synchronisation

When clustering might not be right:

- huge networks
- intermittent connectivity
- ruling bandwidth with an iron fist
- different administrative domains

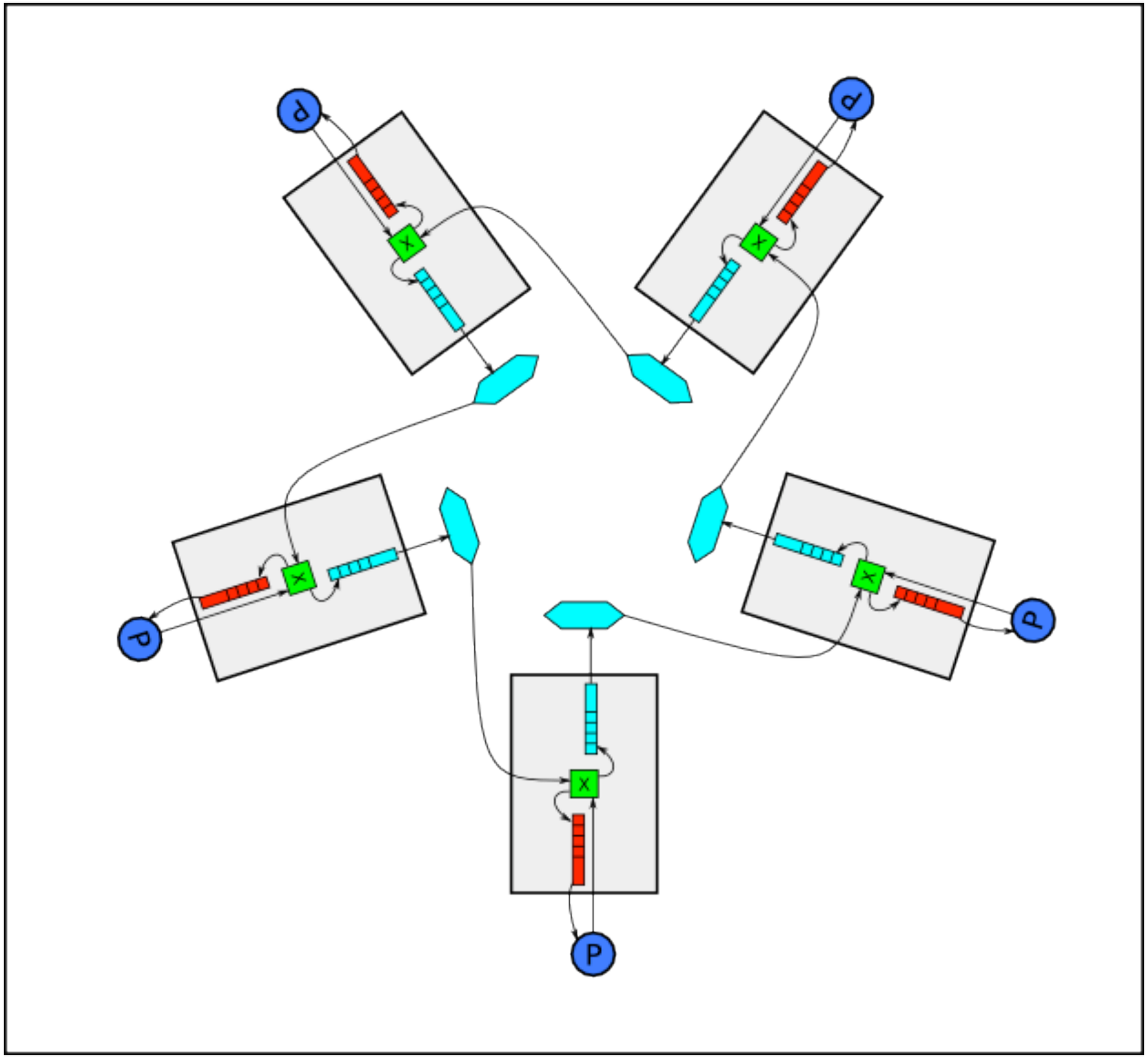


Synchronisation



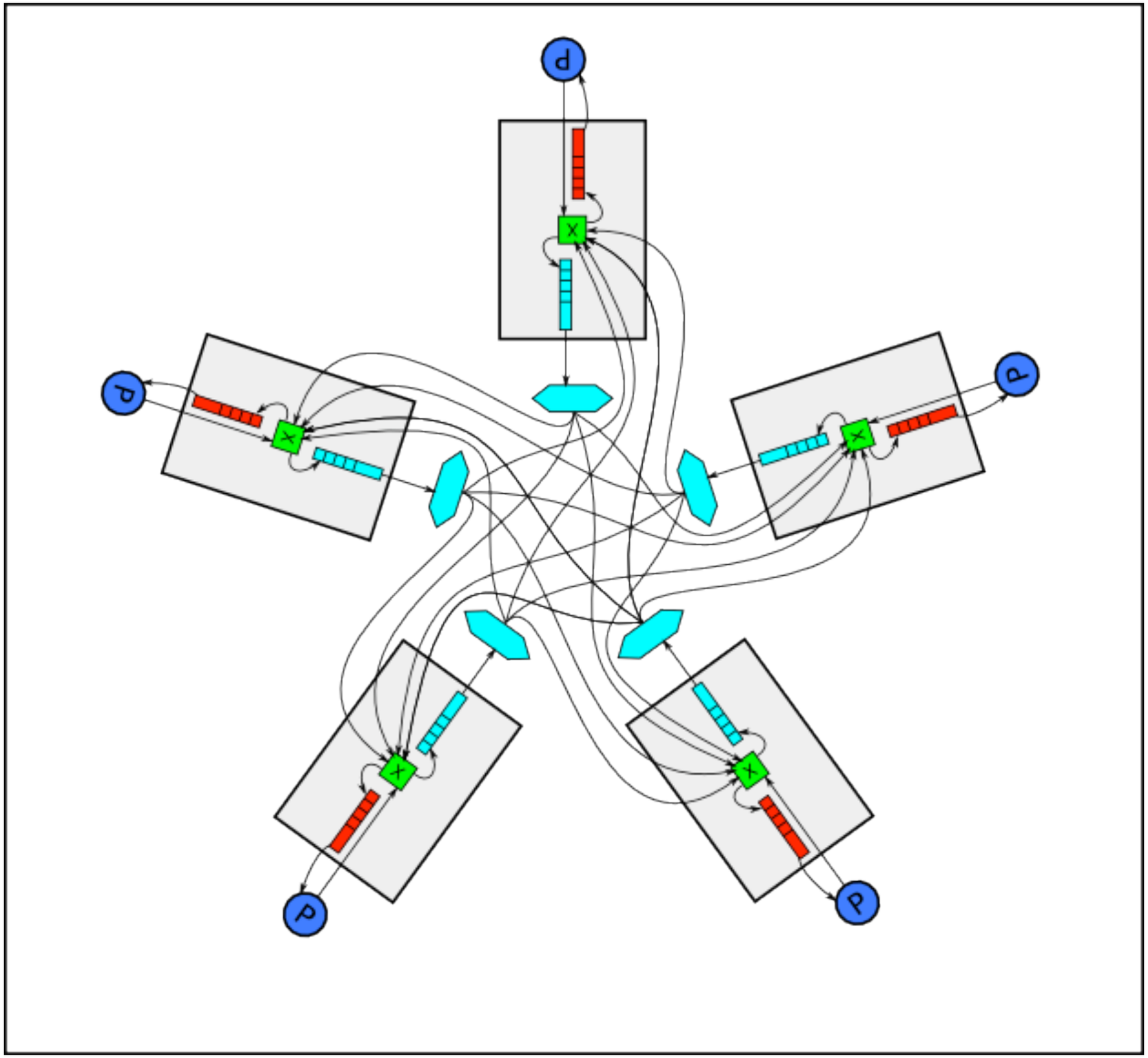
Ring

Pass it on to your neighbour if your neighbour's name isn't in the list yet



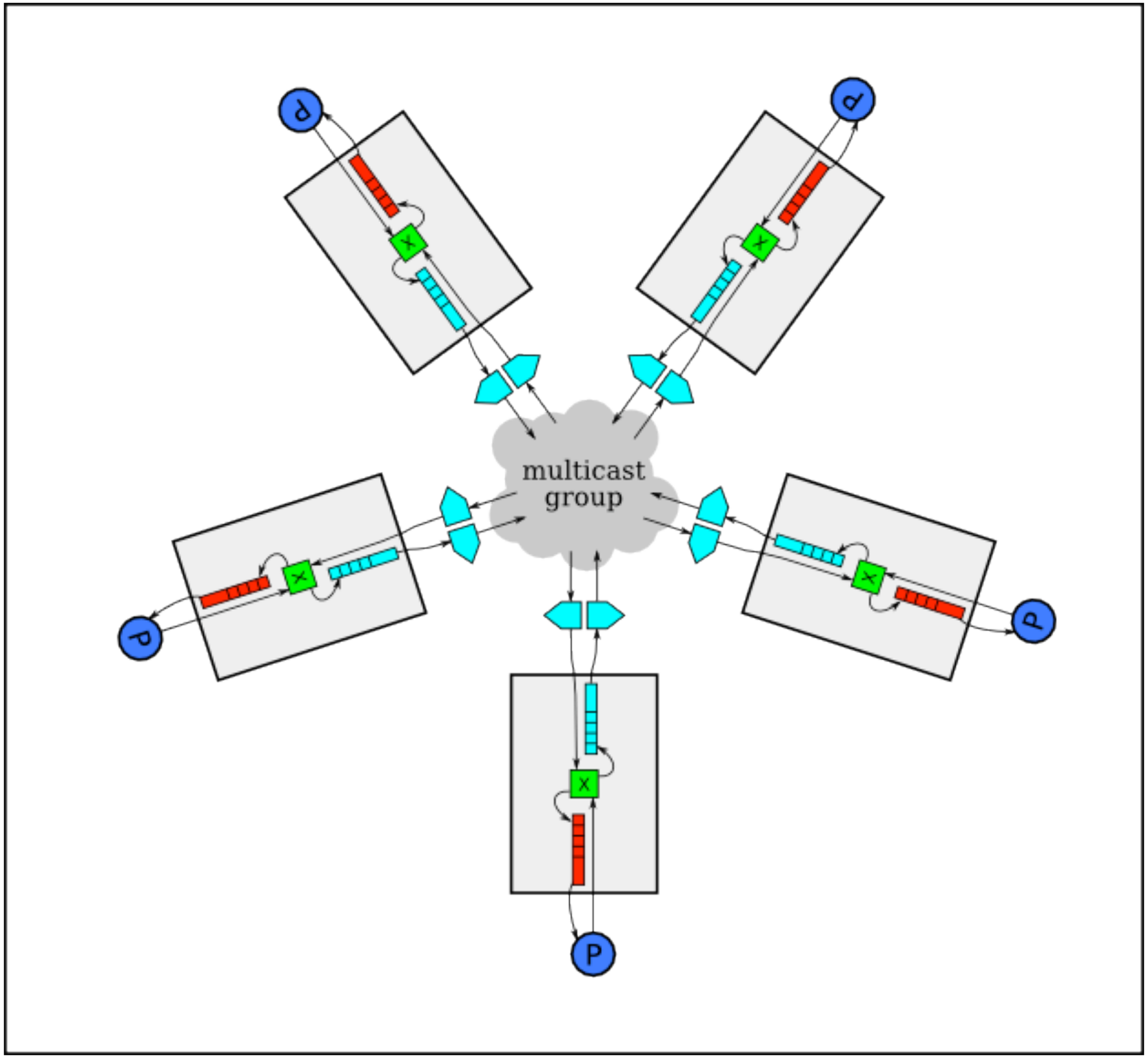
Complete Graph

Pass it on to your neighbour if it hasn't been labelled at all yet



Multicast

Pass it on to your neighbour if it hasn't been labelled at all yet



Durability, Persistence & Memory Usage

Terminology

- Durable: resource (exchange, queue, binding) that survives broker restart
- Persistent: message that survives broker restart
- Durability **and** persistence required for robust storage at the broker

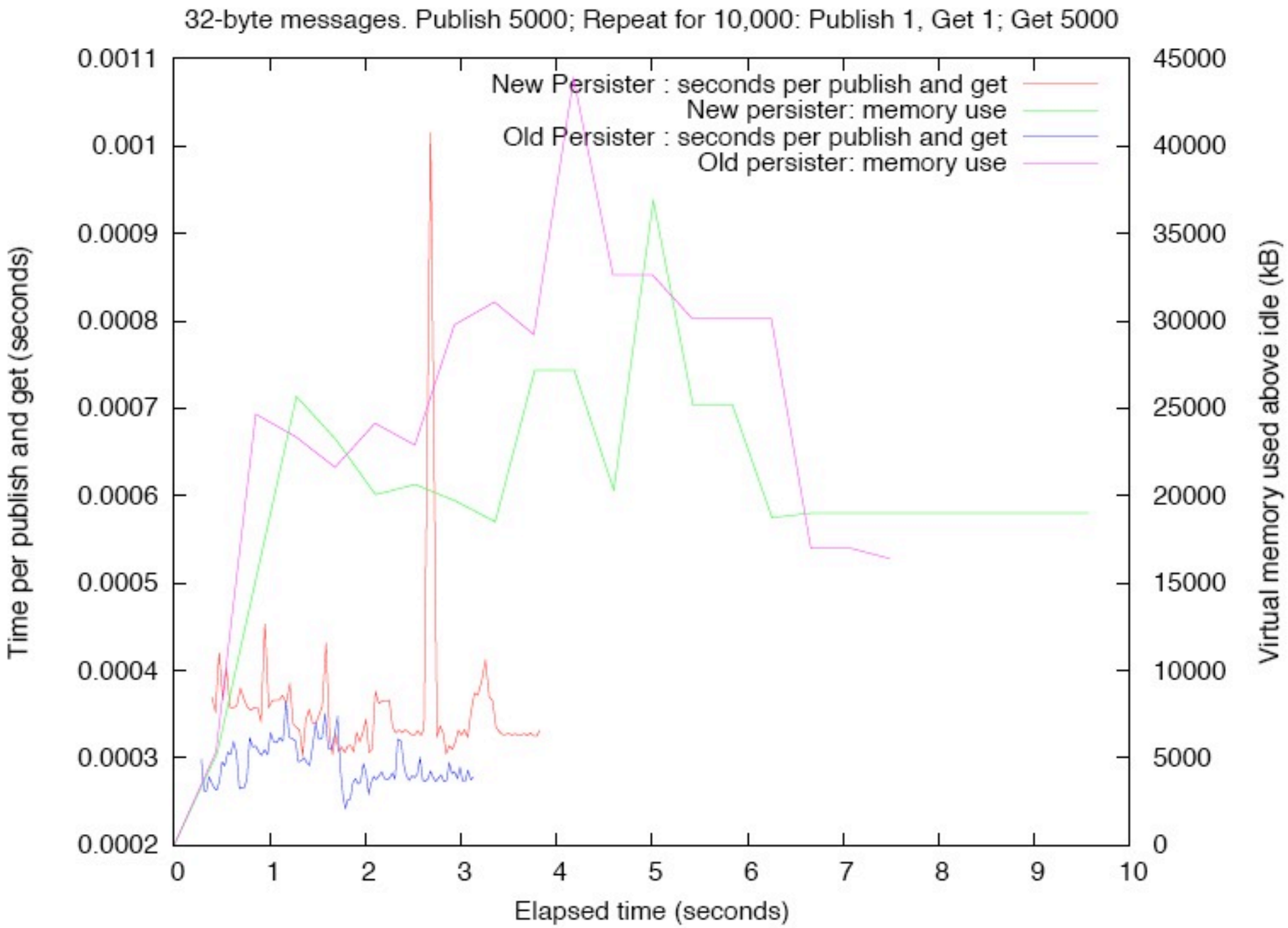
Persister: Old

- Old persister slows down dramatically once a backlog builds up
- Old persister uses disk as a backup only: still limited by RAM + swap
- Once swap runs out, it's all over

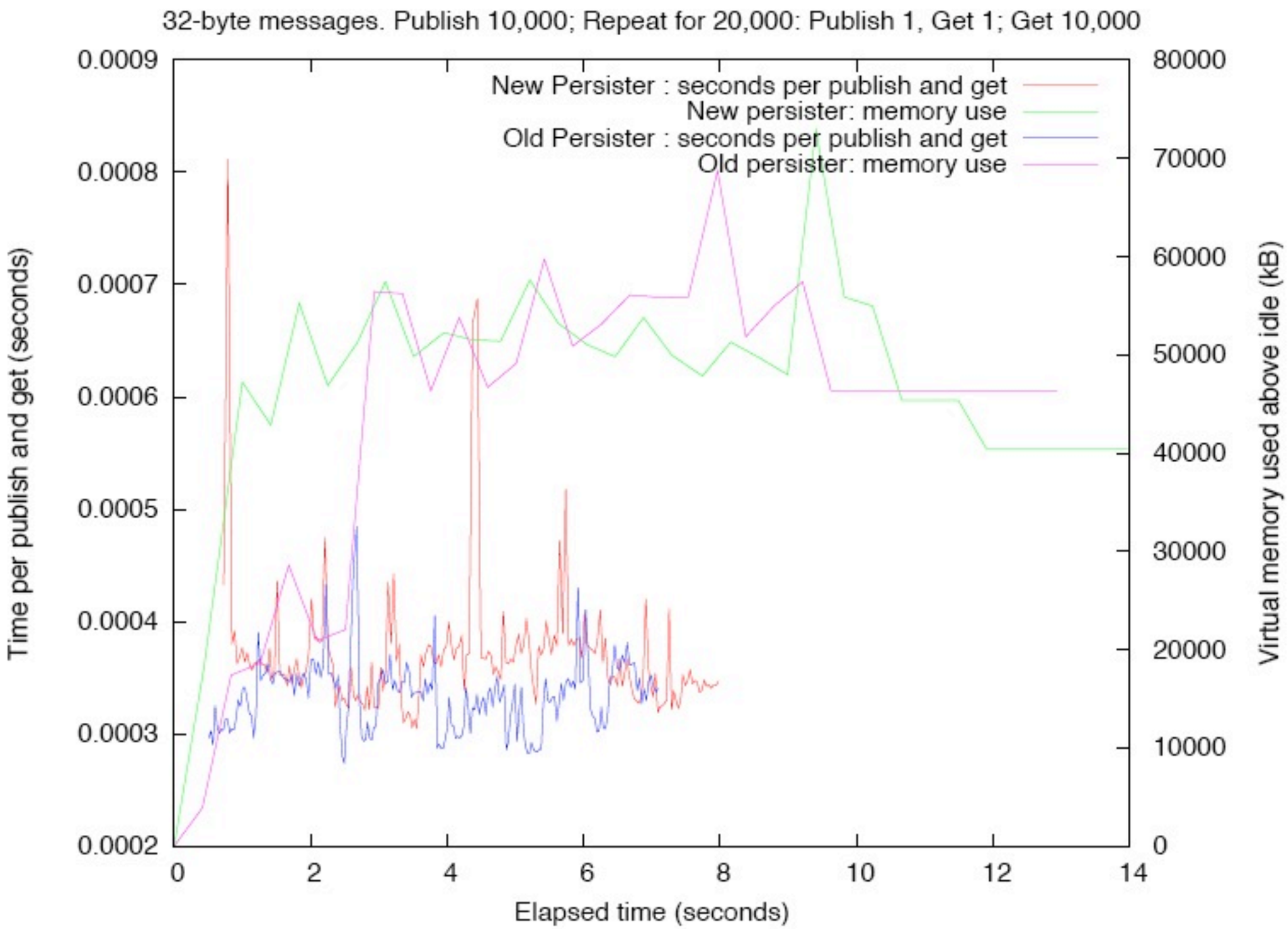
Persister: New

- New persister doesn't slow down with backlog
- New persister doesn't overflow RAM
  (but does make good use of the RAM you have installed!)
- Automatic or manual decision when to switch to disk and then disk-only mode
- Carefully tuned to give good default behaviour

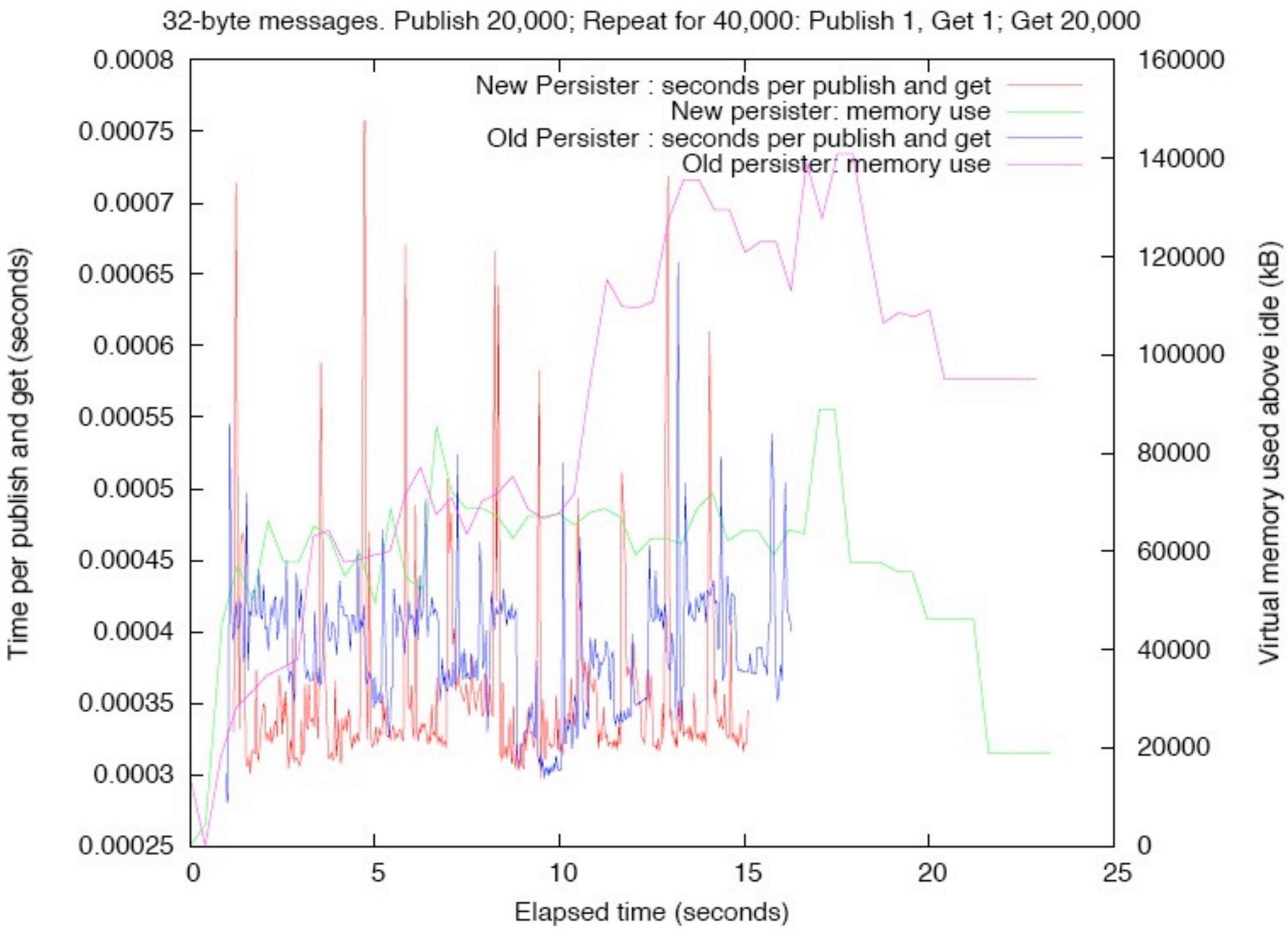
32B, Backlog 5,000



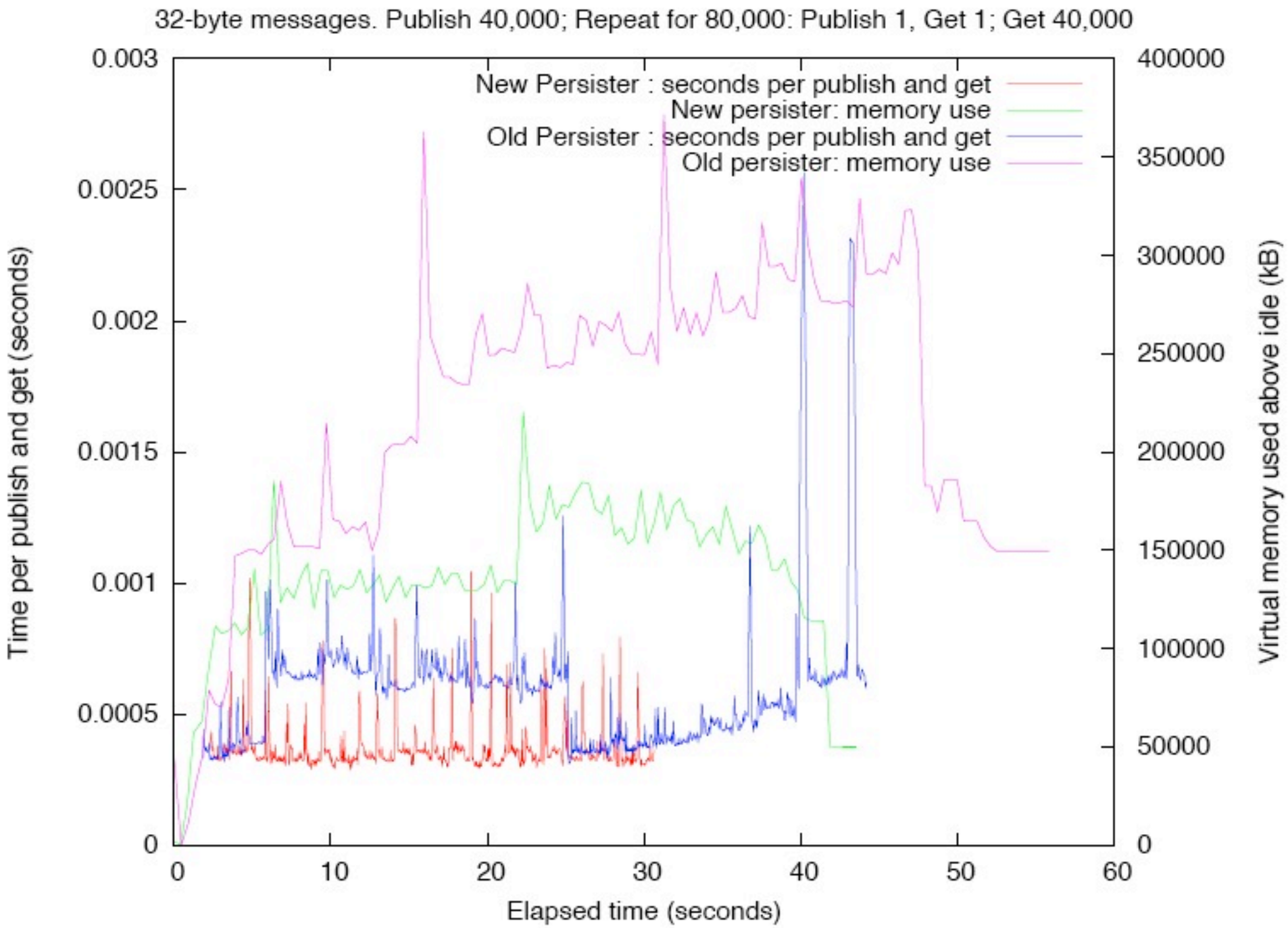
32B, Backlog 10,000



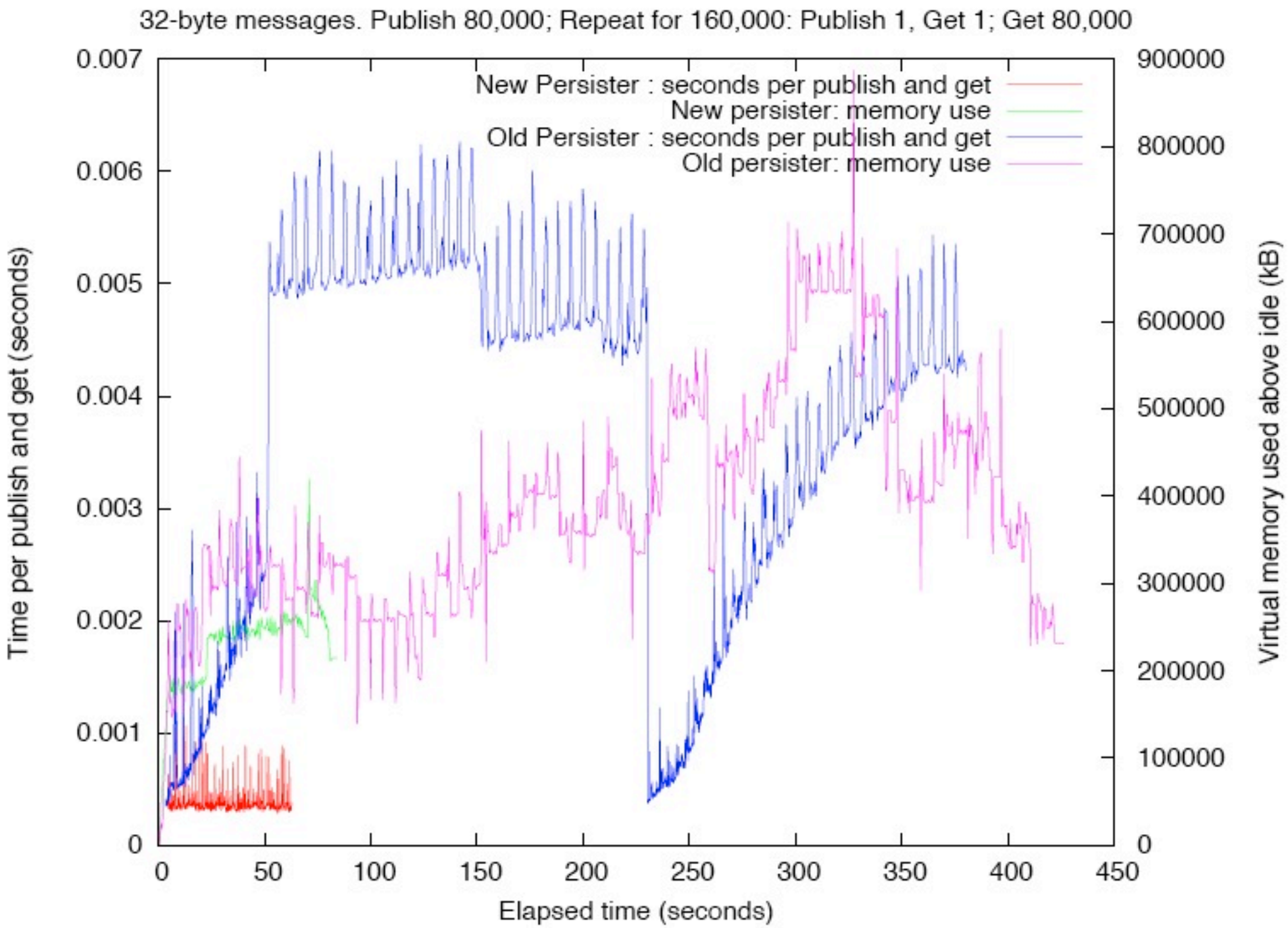
32B, Backlog 20,000



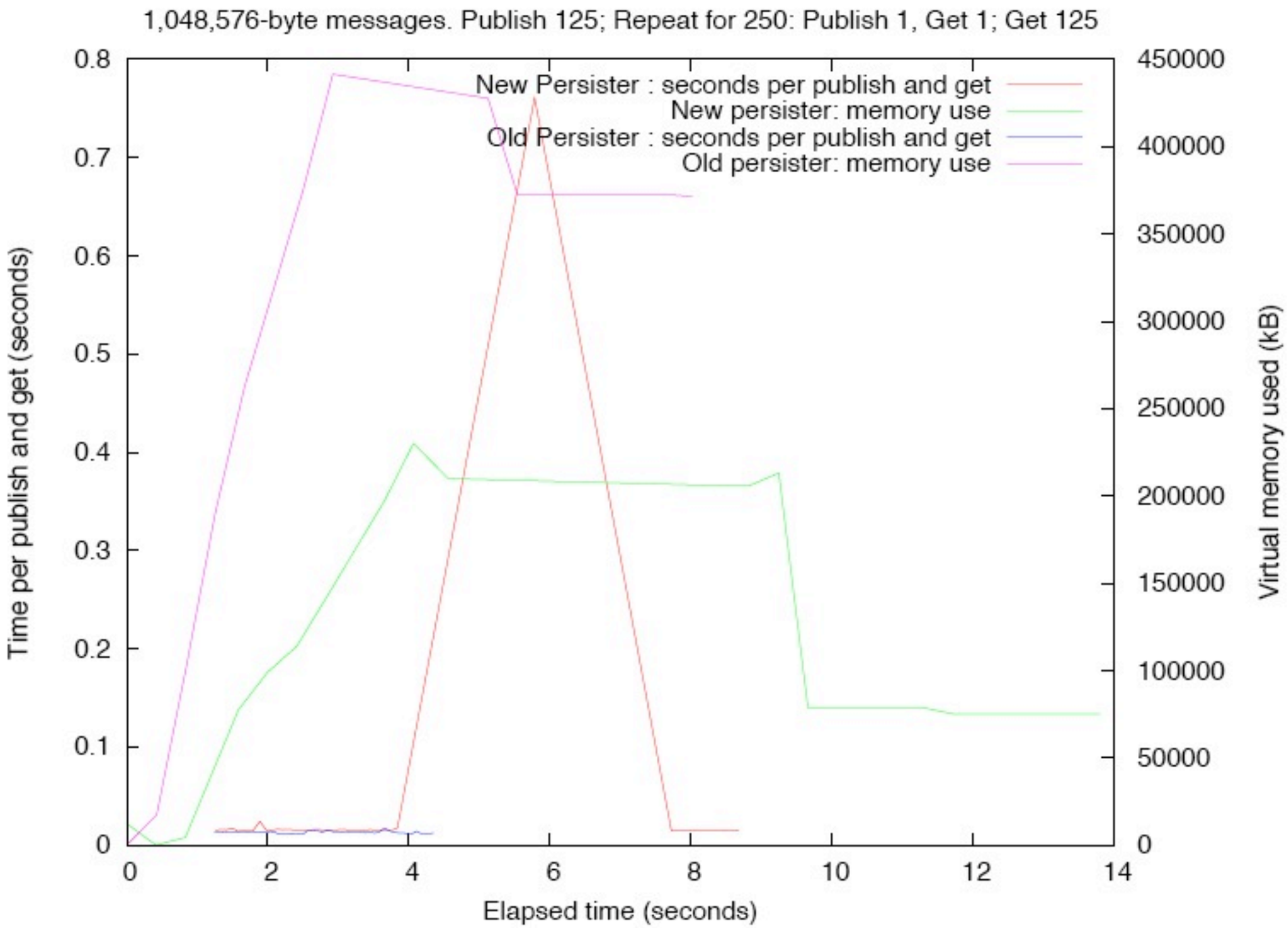
32B, Backlog 40,000



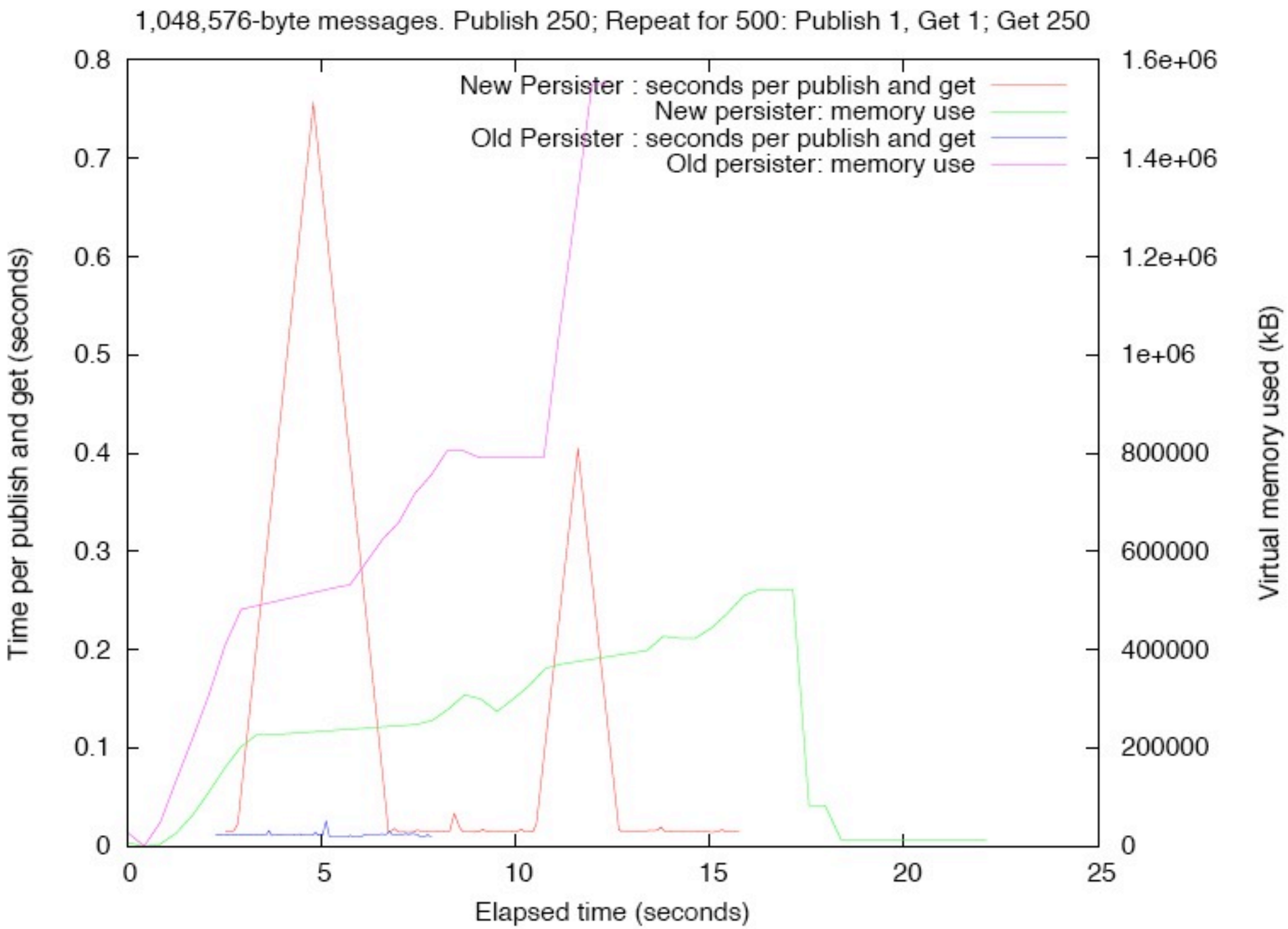
32B, Backlog 80,000



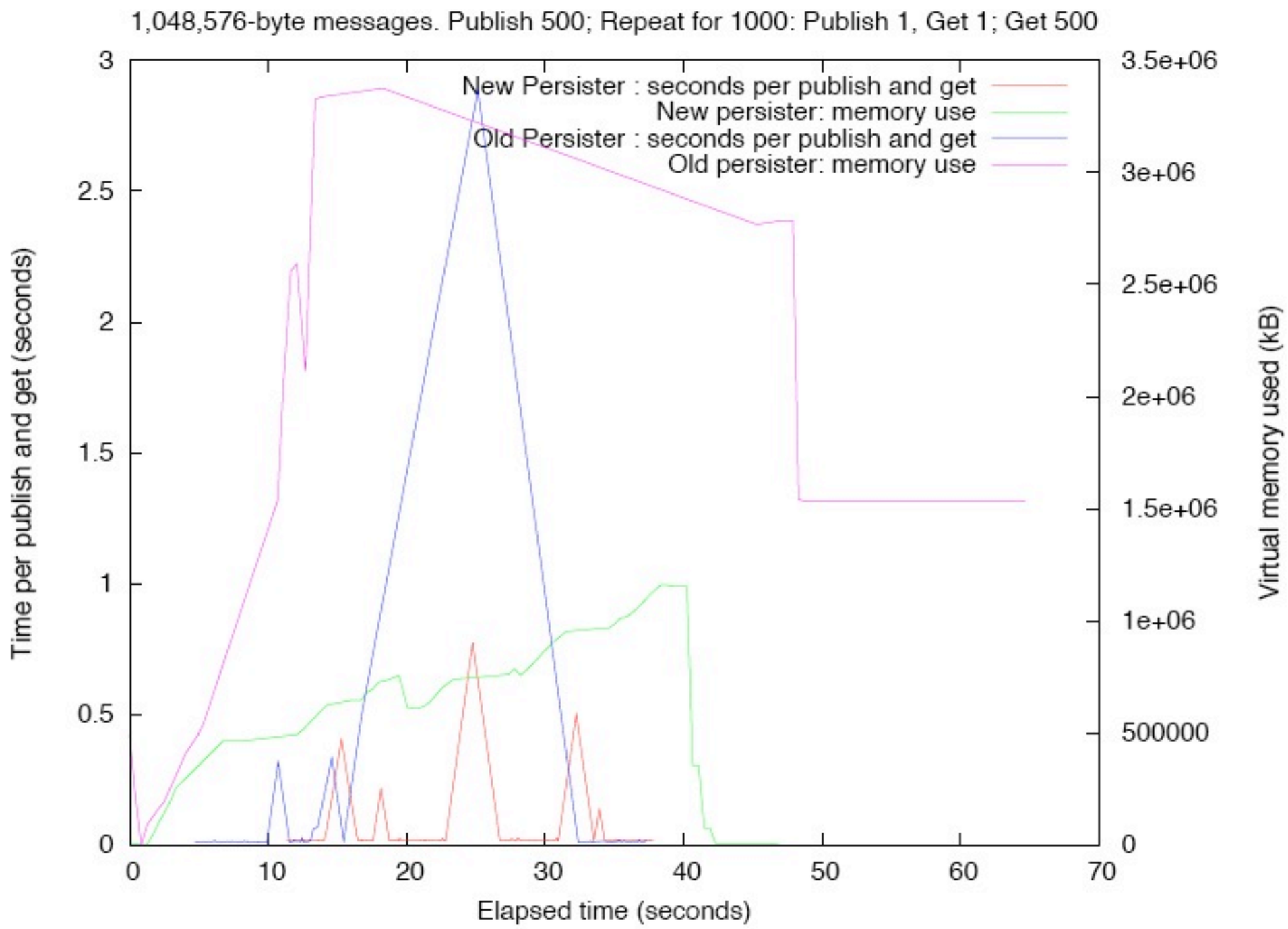
1MB, Backlog 125



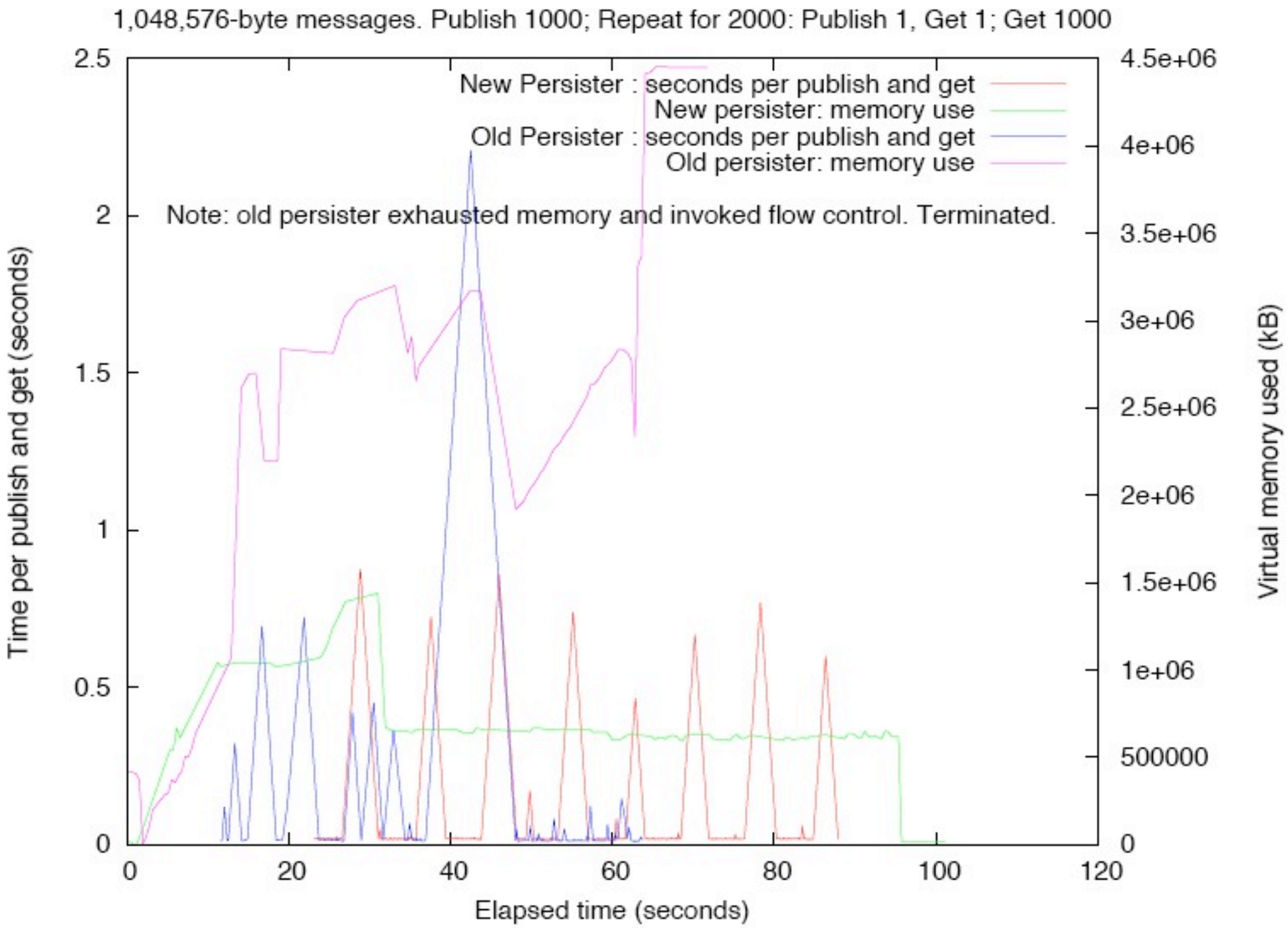
1MB, Backlog 250



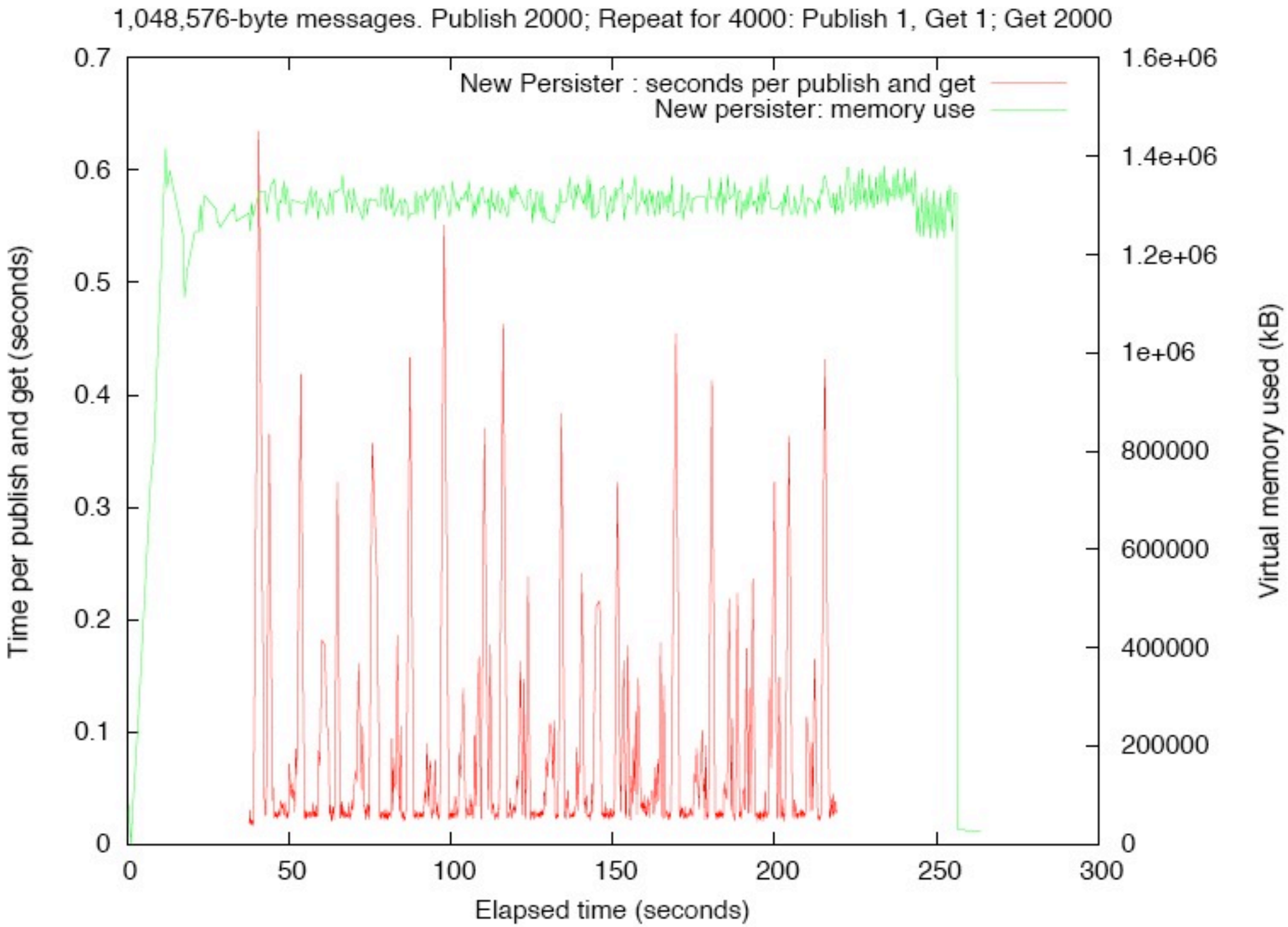
1MB, Backlog 500



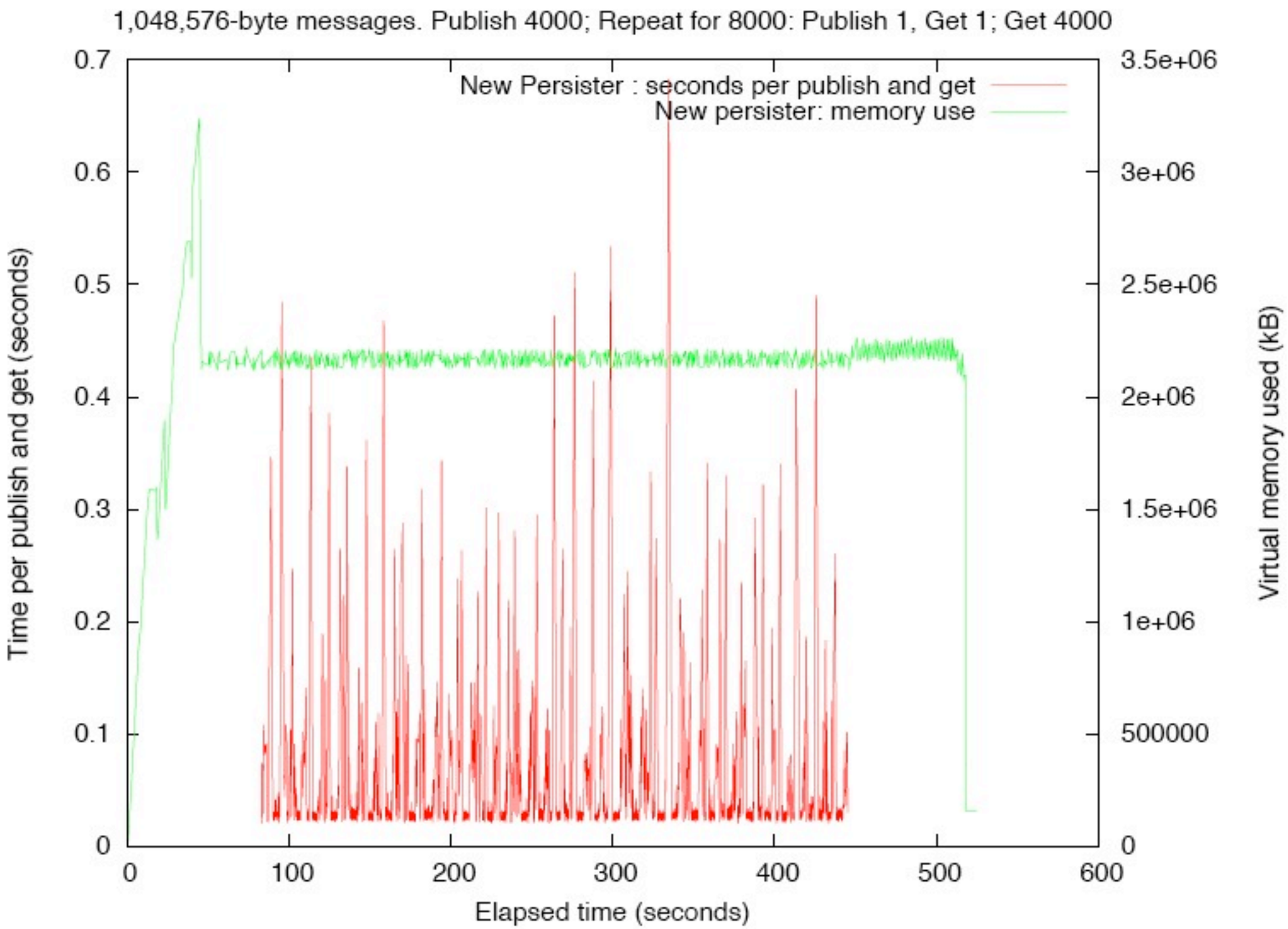
1MB, Backlog 1,000



1MB, Backlog 2,000



1MB, Backlog 4,000



Persister: Summary

- Fast (slower than old persister, but still fast)
- Scales to fill your disks without filling RAM
- In QA at the moment; will land for 1.7

Security

Resource ACLs

- Users are granted access to each vhost
- Each user has three regular expressions:
  - for Administrative actions (create, delete)
  - for Reading (bind from exch, consume from queue)
  - for Writing (publish to exch, bind to queue)

Encryption, PKI

- Securing the Transport
  - stunnel4
  - native SSL support in final QA
- Securing the Messages
  - key directory + per-message encryption and signing

Operational Tools

rabbitmqctl

WALK XTERM

```
Jul 20 14:39:08 tonyg@nodachi2
~/dev/rabbitmq-umbrella/rabbitmq-server/scripts$ ./rabbitmqctl list_queues name
durable auto_delete messages_ready consumers memory
Listing queues ...
ArchivesQ2      true    false   0       0       1976
ArchivesQ       true    false   0       0       1976
...done.

Jul 20 14:40:50 tonyg@nodachi2
~/dev/rabbitmq-umbrella/rabbitmq-server/scripts$ ./rabbitmqctl list_bindings
Listing bindings ...
        ArchivesQ       ArchivesQ       []
        ArchivesQ2      ArchivesQ2      []
ArchivesEx      ArchivesQ               []
ArchivesEx2     ArchivesQ2      Hello%20World   []
...done.

Jul 20 14:46:12 tonyg@nodachi2
~/dev/rabbitmq-umbrella/rabbitmq-server/scripts$
```

BDL

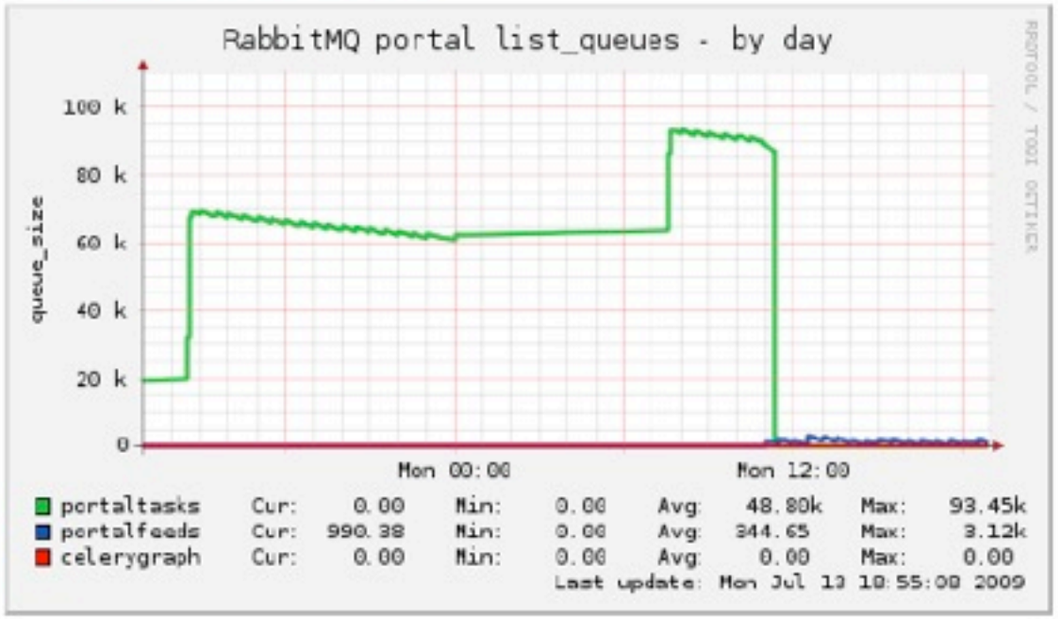
WALK XTERM

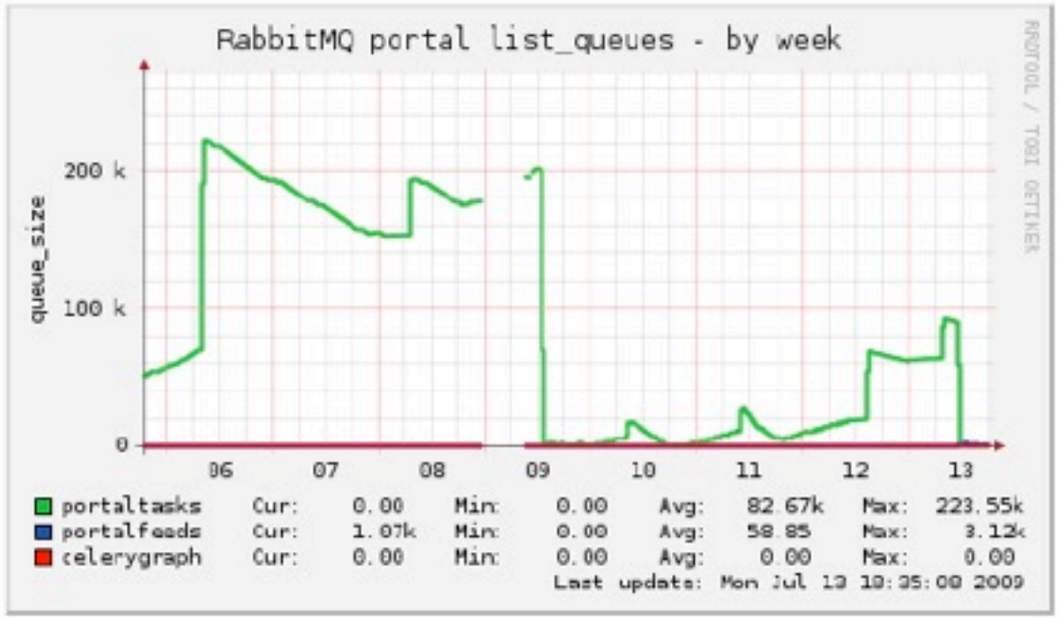
```
BDL> create durable exchange 'ArchivesEx2';
ok
BDL> create durable queue 'ArchivesQ2';
ok
BDL> create route from 'ArchivesEx2' to 'ArchivesQ2' when routing_key is 'Hello
World';
ok
BDL> select name, 'durable', memory from queues where memory > 1000;
--------------------------------
| name       | durable | memory |
--------------------------------
| ArchivesQ2 | true    | 1976   |
| ArchivesQ  | true    | 1976   |

BDL>
```

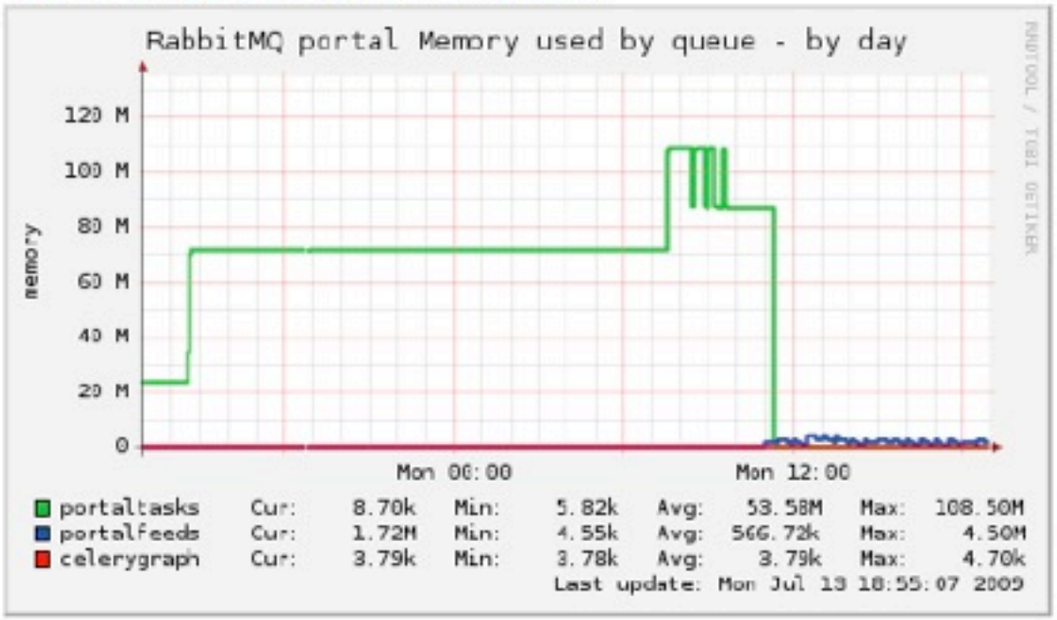
Munin plugins

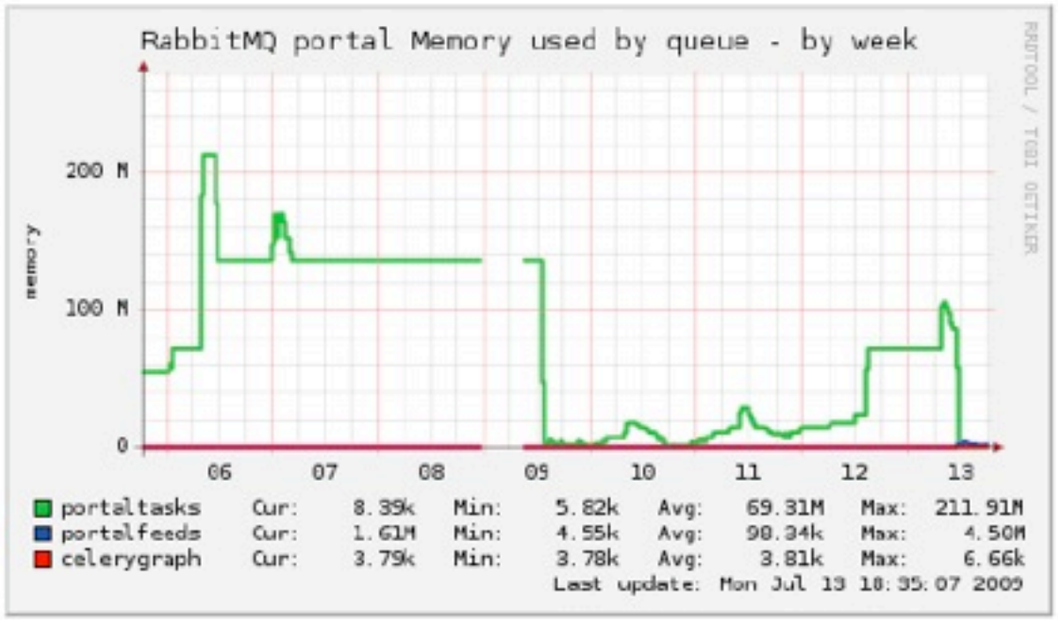
:: RabbitMQ portal list_queues





:: RabbitMQ portal Memory used by queue





RESTful APIs: Trixx

```
GET    http://localhost:8080/exchanges/:vhost
POST   http://localhost:8080/exchanges
GET    http://localhost:8080/queues/:vhost
POST   http://localhost:8080/queues
GET    http://localhost:8080/bindings/:vhost
GET    http://localhost:8080/vhosts
POST   http://localhost:8080/vhosts
GET    http://localhost:8080/connections
GET    http://localhost:8080/users/:user/permissions

GET    http://localhost:8080/users
PUT    http://localhost:8080/users

POST   http://localhost:8080/users

DELETE http://localhost:8080/users/:user

GET    http://localhost:8080/rabbit/status
PUT    http://localhost:8080/rabbit/start
PUT    http://localhost:8080/rabbit/stop
PUT    http://localhost:8080/rabbit/reset

POST   http://localhost:8080/sessions/authenticate
```

RESTful APIs: Alice

```
/conn - Current connection information
/exchanges - Current exchanges information
/queues - Current queues
/users - Current users
/bindings - Current bindings
/control - Access to the RabbitMQ control
/permissions - Current permissions
/vhosts - Current vhosts
```

- -

```
auser $ curl -i -XPOST \
        -d'{"username":"ari", "password":"weak password"}' \
        http://localhost:9999/users

HTTP/1.1 200 OK
Server: MochiWeb/1.0 (Any of you quaids got a smint?)
Date: Thu, 16 Jul 2009 00:10:35 GMT
Content-Type: text/json
Content-Length: 25

{"users":["ari","guest"]}
```

Web UIs: Trixx

Trixx Web Console

http://localhost:3000/

Most Visited · To Read · GST Widget Layout · [[comix]] · Open AMQ Bugs · Indexing · McSweeney · worldclock.swf (appl...

Start · Stop · Reset · Log Out

Trixx Dashboard

Running Services: rabbit, mnesia, os_mon, sasl, stdlib, kernel

NODES

running rabbit@nodachi2

VHOST: /

Select vhost

Queues:

QUEUE: ARCHIVESQ2 DURABLE

Status

0 consumers · 0 transactions · 1.9 KB

Messages

0 total · 0 ready · 0 unacknowle

Done

Trixx Web Console

http://localhost:3000/

Most Visited · To Read · GST Widget Layout · [[comix]] · Open AMQ Bugs · Indexing · McSweeney · worldclock.swf (appl...

AMQ.FANOUT FANOUT

Bindings:

DIRECT

Bindings:

- Queue: ArchivesQ (Routing Key: ArchivesQ)
- Queue: ArchivesQ2 (Routing Key: ArchivesQ2)

Users: Add User

Name	Config	Write	Read
guest	.*	.*	.*

Connections:

CONNECTION: 127.0.0.1:5672 FROM 127.0.0.1 BY GUEST RUNNING

1 channels · 131,072 frame-max · 0 timeout

225 recv oct · 11 recv cnt

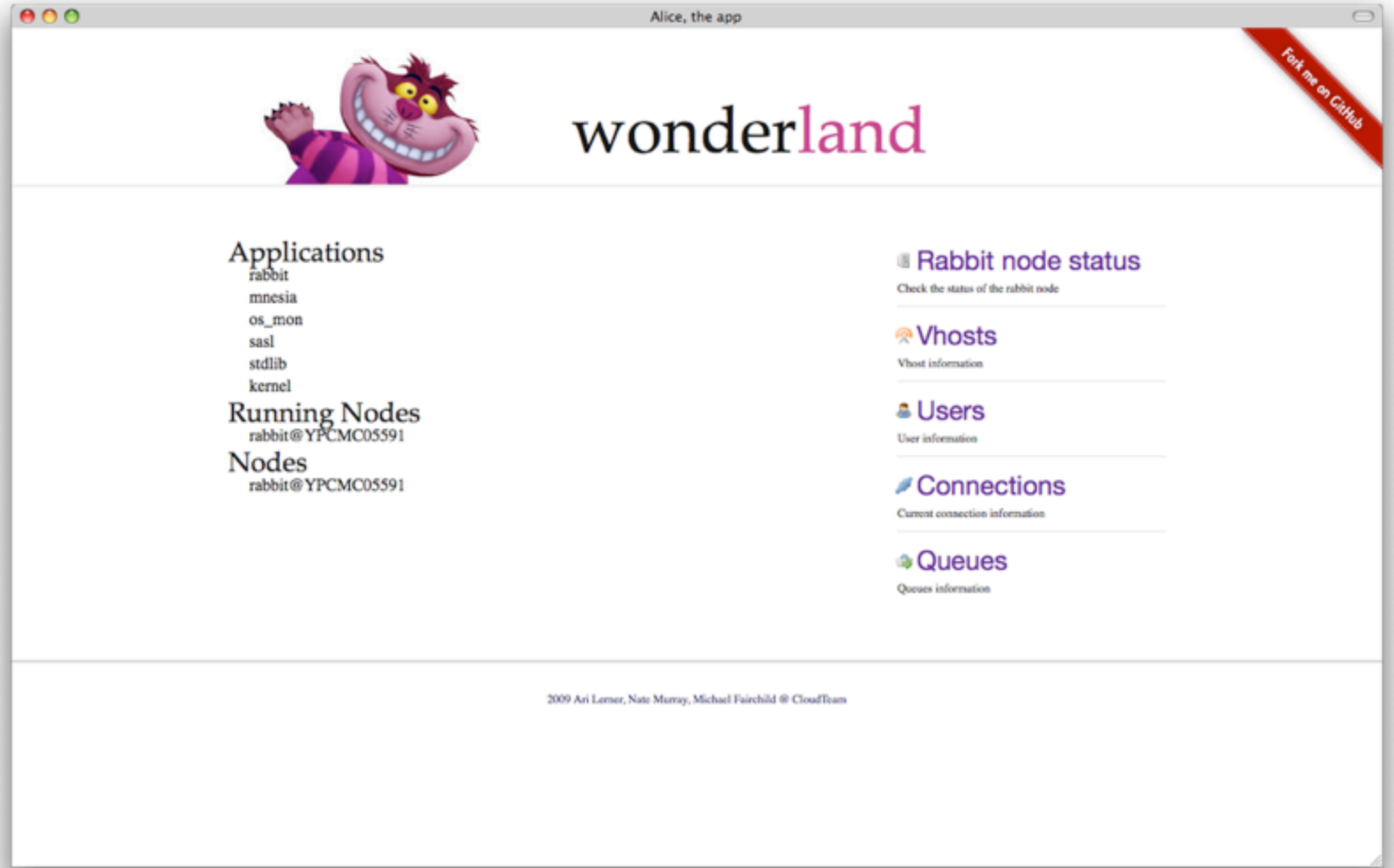
0 send pend · 0 sent oct · 5 sent cnt

CONNECTION: 127.0.0.1:5672 FROM 127.0.0.1 BY GUEST RUNNING

1 channels · 131,072 frame-max

Done

Web UIs: Wonderland



Other ongoing projects

- Management
  - SNMP
  - JMX, HermesJMS
  - Presence, meta-events
  - AMQP Protocol extensions for management
- Gateways, federation
  - RabbitHub (HTTP)
  - Multicast
  - XMPP, JMS bridges
  - AMQP Protocol extensions for generalised gatewaying