Messaging Performance Testing: tips, tricks and tools

Otavio Rodolfo Piske <opiske@redhat.com>

About the talk

- Basics
- Tips, tricks and anti-patterns
- Tools
- Tweaks

About me

- Software Engineer in Test at Red Hat Messaging QE team
- Working with messaging for ~10 years
  - About 7 of those with IBM WebSphere MQ
  - Mostly with C/C++ and Java
- Social media:
  - Twitter: @otavio021
  - Github: orpiske
  - Site: http://www.orpiske.net

What this talk is not about

- Tuning code or brokers for performance
- Tuning OS for messaging
- Demo

Why performance is important

- Cost
- User experience
- IoT and embedded devices
- Business:
  - Business opportunities (ie.: charge by transaction)
  - Enforced by contracts
  - Law and other regulations

Definition of performance

- Latency
  - One way
  - Round-trip
- Throughput
  - Sustained throughput
- Resource usage
  - CPU
  - Memory
  - I/O

Messaging QE testing

- Past
- Current
- Future

Messaging QE testing: past

- Spec JMS
- No history
- No standards

Messaging QE testing: current

- JBoss A-MQ 6 (Apache ActiveMQ)
  - JBoss A-MQ 7 (Apache Artemis) in progress
- Protocols: AMQP 1.0
- 2 message sizes
- 2 broker configurations
- Performance DB

Messaging QE testing: future

- Multi-protocol iteration
- CI integration
    - Performance gating
- Explore performance on containers
- Upstream contribution

Messaging performance testing

- Anti-patterns
- Tips
- Tricks

Anti-patterns

- Lack of objective or performance goal
- Short testing duration
  - Broker warm up
  - JVM: GC hit
- Lack of baseline
- Lack of (evolutionary) history

Tips

- Define a goal
- Define a reasonable duration
  - At least 3h for non-critical systems
  - Much for for mission critical
- Measure your system
  - Network performance
  - Database performance (if applicable)
- Execute more than once
  - Establish baselines: broker, network, system

Tricks

- Be gentle: don't flood the broker at once
- Understand real-world usage scenarios
  - Development/testing lab vs. real-world
    - Network congestion
    - Different resource capacity
- Messaging protocol differences
- Botlenecks
  - Application
  - Database
  - Network
  - JVM
  - OS or other environment

Messaging performance testing tools

- MPT: msg-perf-tool
- MPT UI: msg-perf-ui
- BMIC
- Others
  - PBench
  - Quiver

MPT: msg-perf-tool

- Multiprotocol:
  - AMQP 1.0
  - STOMP 1.2
  - MQTT 3.1 and 3.1.1
- Performance testing
- Tune "guessing"
- ElasticSearch DB
- Apache 2.0
  - Source and RPMs: http://orpiske.github.io/msg-perf-tool/

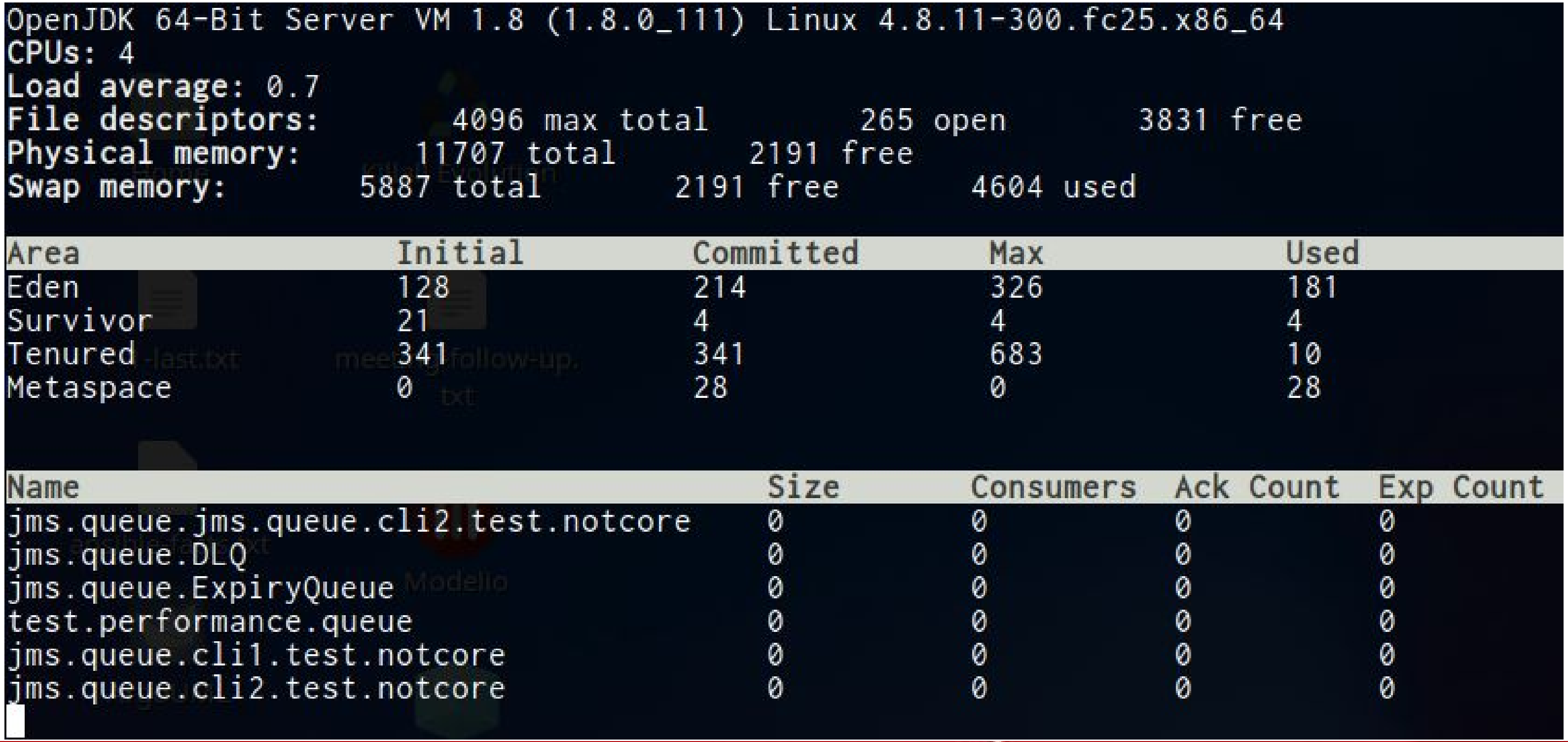
MPT UI: msg-perf-ui

- Responsive Web UI
- AngularJS
- ElasticSearch front-end
- Metrics:
  - One-way latency
  - Latency percentiles
  - Sender/receiver throughput
- MIT license
  - Source: https://github.com/orpiske/msg-perf-ui

BMIC: broker management interface client

- A client for REST management interface
  - JBoss A-MQ 6
  - JBoss A-MQ 7
  - Apache ActiveMQ
  - Apache Artemis
- Components
  - Shared library
  - A CLI management tool
    - Broker top
- Apache 2.0
  - Source: https://github.com/orpiske/bmic

BMIC: broker management interface client



```
OpenJDK 64-Bit Server VM 1.8 (1.8.0_111) Linux 4.8.11-300.fc25.x86_64
CPUs: 4
Load average: 0.7
File descriptors:       4096 max total        265 open       3831 free
Physical memory:      11707 total         2191 free
Swap memory:       5887 total       2191 free       4604 used

Area                 Initial          Committed        Max              Used
Eden                 128              214              326              181
Survivor             21               4                4                4
Tenured              341              341              683              10
Metaspace            0                28               0                28


Name                                     Size       Consumers  Ack Count  Exp Count
jms.queue.jms.queue.cli2.test.notcore    0          0          0          0
jms.queue.DLQ                            0          0          0          0
jms.queue.ExpiryQueue                    0          0          0          0
test.performance.queue                   0          0          0          0
jms.queue.cli1.test.notcore              0          0          0          0
jms.queue.cli2.test.notcore              0          0          0          0
```

PBench

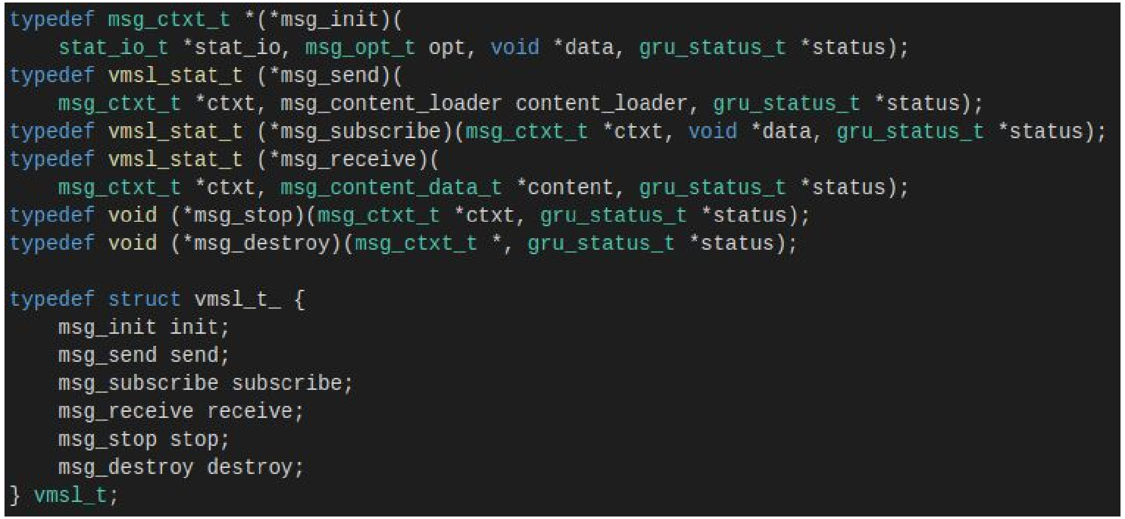
- A performance test framework
- Test orchestration
- Test post-processing
- GPL 3.0
  - Source and packages: http://distributed-system-analysis.github.io/pbench/

Quiver

- A set of tools for testing messaging clients and brokers
- Developed by Qpid Proton Developers
- Focused on AMQP 1.0 clients/brokers
    - Some JMS implementations available
- Apache 2.0
    - Source: https://github.com/ssorj/quiver/

Tools: tweak

- msg-perf-tool
    - VMSL
    - Content Loader



```
typedef msg_ctxt_t *(*msg_init)(
    stat_io_t *stat_io, msg_opt_t opt, void *data, gru_status_t *status);
typedef vmsl_stat_t (*msg_send)(
    msg_ctxt_t *ctxt, msg_content_loader content_loader, gru_status_t *status);
typedef vmsl_stat_t (*msg_subscribe)(msg_ctxt_t *ctxt, void *data, gru_status_t *status);
typedef vmsl_stat_t (*msg_receive)(
    msg_ctxt_t *ctxt, msg_content_data_t *content, gru_status_t *status);
typedef void (*msg_stop)(msg_ctxt_t *ctxt, gru_status_t *status);
typedef void (*msg_destroy)(msg_ctxt_t *, gru_status_t *status);

typedef struct vmsl_t_ {
    msg_init init;
    msg_send send;
    msg_subscribe subscribe;
    msg_receive receive;
    msg_stop stop;
    msg_destroy destroy;
} vmsl_t;
```

Tools: tweak

- BMIC
  - Transports
  - Products

Tools: future

- Other products and protocols
  - AMQP 0-9-1 (RabbitMQ)
  - WebSphere MQ
  - Openwire
  - Others
- Improve extensibility
  - Probes
  - Improved resource usage metrics
- Improve coverage:
  - JMS client support
- AMQP management

Tools: future

- Leverage existing metrics collection/management tools
  - Hawkular, grafana, prometheus, etc
- Leverage other tools

Closing comments

- Lots of room for improvements
- Industry tools for messaging performance are retired
- Opportunity for the community to work together