Master's Thesis

A Resilient Transport Layer for Messaging Systems

David Fuchs

September 14, 2007

Supervision:
Dr. Sean Rooney (IBM Research GmbH, Zurich Research Laboratory)
Prof. Dr. Gustavo Alonso (ETH Zurich, Institute for Pervasive Computing)

Outline

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

- About MQTT
  - protocol overview
  - usage
  - QoS levels
- ReTCP Design
  - motivation
  - design overview
  - design challenges & solutions
  - RTO calculation
- Implementation
- Performance Results
- Q&A

MQTT Protocol

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lightweight Publish/Subscribe protocol

- broker/client architecture
- clients subscribe to topics they are interested in
- clients publish messages on a topic
- broker forwards messages to all interested parties

Designed for many-to-many communication with easy configuration

Designed for network edge devices

- typically small devices with limited battery power, processing power
- examples: hand-held devices, temperature sensors, flow meters, power meters, ...

Usage Example

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

"Black Box" in car

- records usage of car (speed, time of day, distance, ...)
- periodically sends data to insurance company
- data transmission uses GPRS-link with high latency, low bandwidth
- premium is based on actual usage rather than age, ethnic group, etc

QoS Levels

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

QoS 0: best effort

- used when loss of messages is tolerable

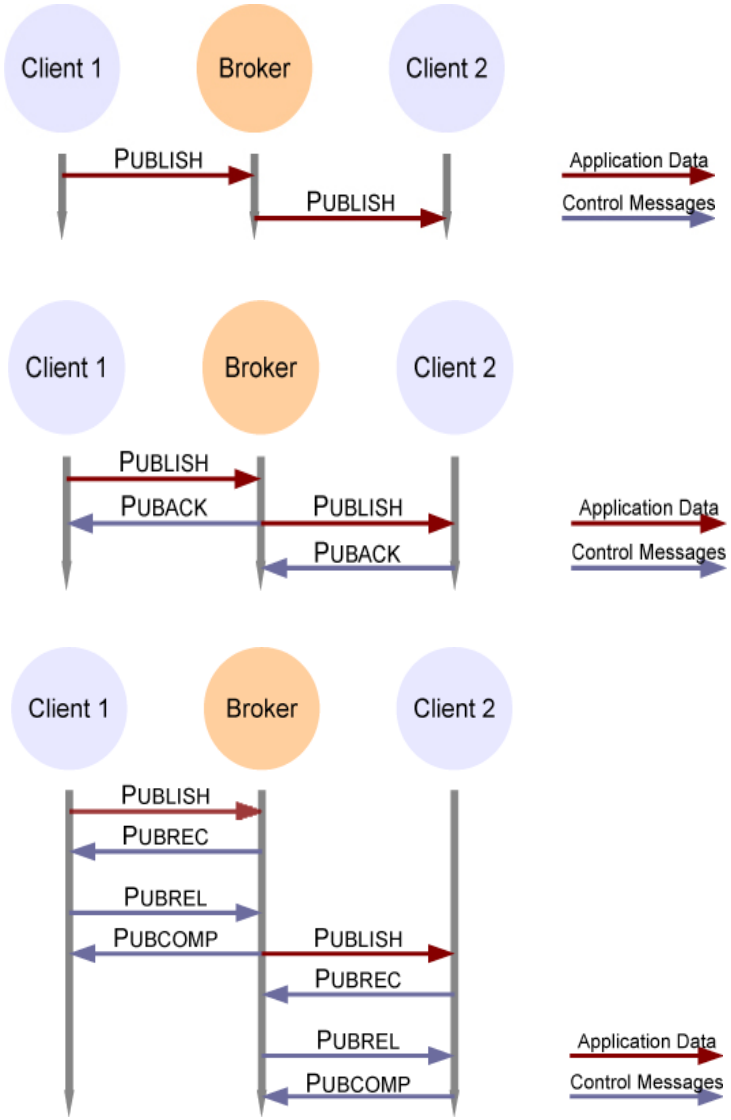
QoS 1: at-least-once

- when loss is not tolerable, but duplicates are
- seldomly used

QoS 2: exactly-once

- when no loss and no duplicates are tolerable

To add resiliency to application failures, msgs may be persisted



Motivation for Resiliency Layer

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Some applications need exactly-once semantic

- e.g., scanner in a warehouse scanning outgoing goods

But QoS 2 is “expensive”:

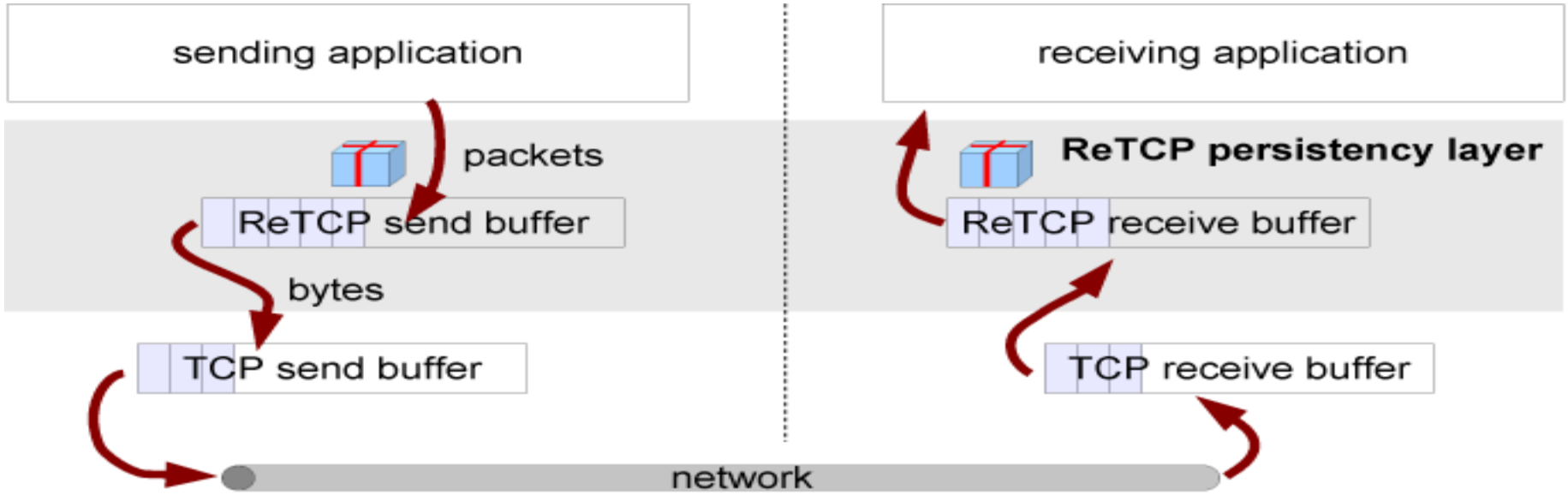
- requires 1.5 round trips until message gets delivered
- adds to bandwidth consumption, which is especially bad for wireless connections (power consumption)
- inefficient use of bandwidth, because delivery protocol is executed sequentially for each single message

➔ Goals of ReTCP:

- exactly-once delivery
- resilient against network failures & application failures
- less byte overhead than QoS 2
- smaller delivery latency than QoS 2
- assumptions: network connections fail; applications hang, crash,loose packets

ReTCP Overview

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich



Runs on top of TCP

- TCP connections may break, but when connected, assure in-order delivery, flow control, congestion control, error detection

Packet-oriented

- rather than byte-stream like TCP

Buffers are stored persistently on disk

ReTCP Design

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

ReTCP “session”:

- can encompass several TCP connections
- client is responsible for reconnecting after crash or network failure
- identified by a connection id; client sends client id and connection id in connection request

ReTCP packets are sent with sequence numbers

- to assure in-order delivery when sessions are aborted and resumed...
- ...and to detect packets that the receiving application lost
- cumulative Ack scheme is used
- RTO timers are used, as in TCP

"TCP-over-TCP" Problems

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

"TCP Meltdown" effect

- happens when upper layer starts retransmitting packets faster than lower layer
- solution: when RTO timer expires, only resend packet when $RTO_{ReTCP} > RTO_{TCP}$. Otherwise, backoff the timer w/o retransmission.

Effects of packet size

- packet sizes span many orders of magnitude (MQTT: 1B...265MB)
- Ack might not be received simply because it is queued behind a large packet that takes long to send
- solution: never resend a packet while data is being read from the incoming byte stream

How to calculate the RTO based on packet size?

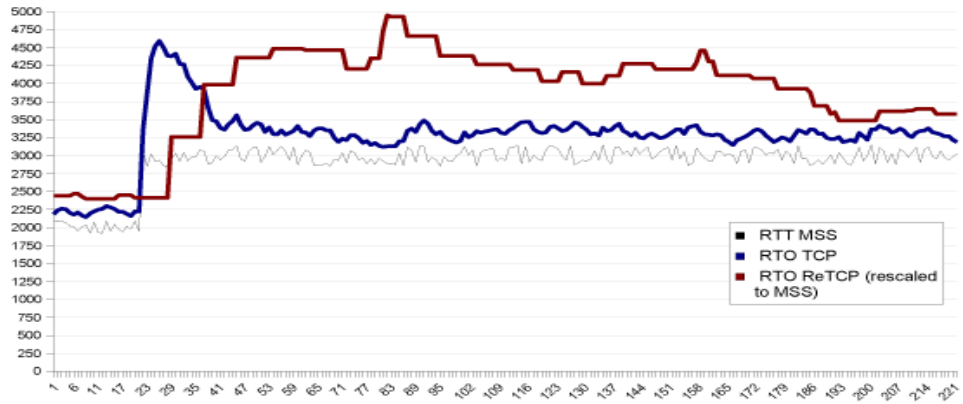
RTO Calculation

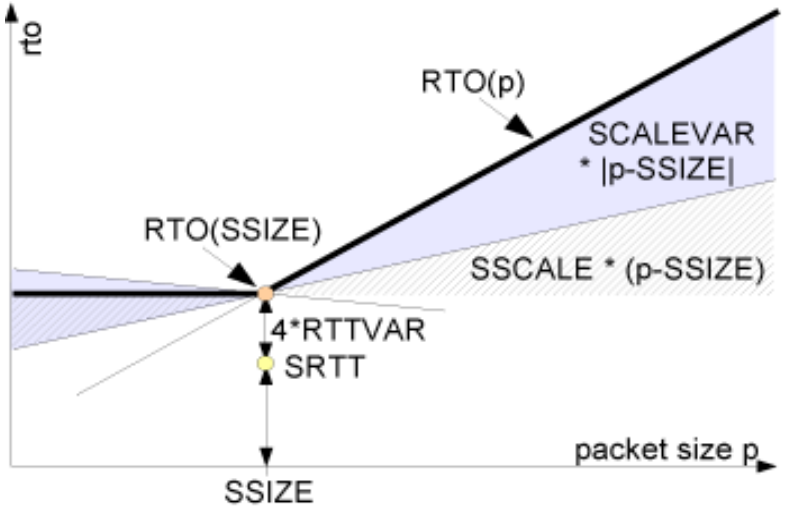
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Assumption: RTT rises linearly with packet size, with an initial fixed latency.

- TCP bases its RTO an averaged past RTTs and their variance
- instead of calculating a scalar value as TCP does, the ReTCP RTO algorithm works in 2 dimensions (RTO time, packet size)
- position and slope of the averaged RTT line are updated when new (packet size, RTT time) measurements are available
- calculated RTO is based on averaged RTT line, variance of position, and variance of slope

Simulation shows it works





Implementation

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Implementation is provided for Java-based MQTT protocol stack

- to handle persistent storing of objects, IBM's `ObjectManager` Java library is used
- the `ObjectManager` offers transactions with the typical ACID properties of a database, but is designed to be light-weight and much faster than most DBMS

Information pertinent to TCP's connection state is read from the `/proc` pseudo file system

- information about buffer states, RTO and much more can be found in `/proc/net/tcp` and `/proc/net/tcp6`
- works for Linux-based OSes only

Performance Results

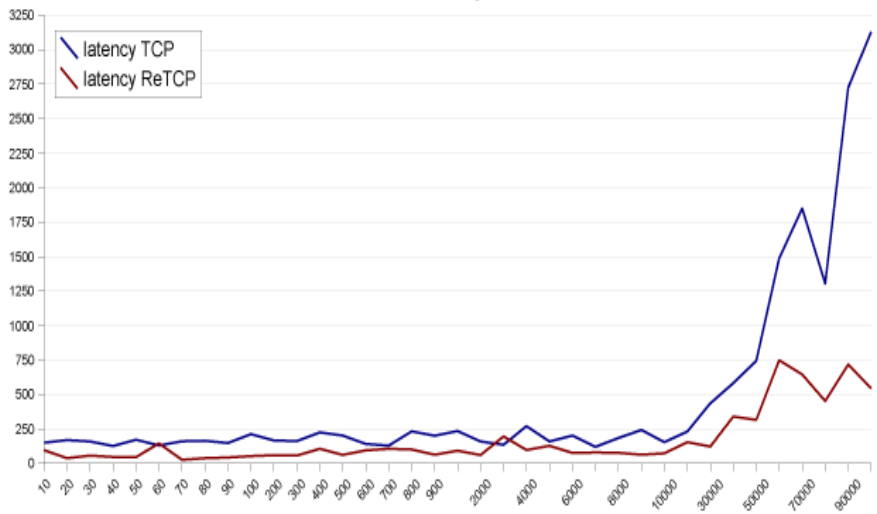
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

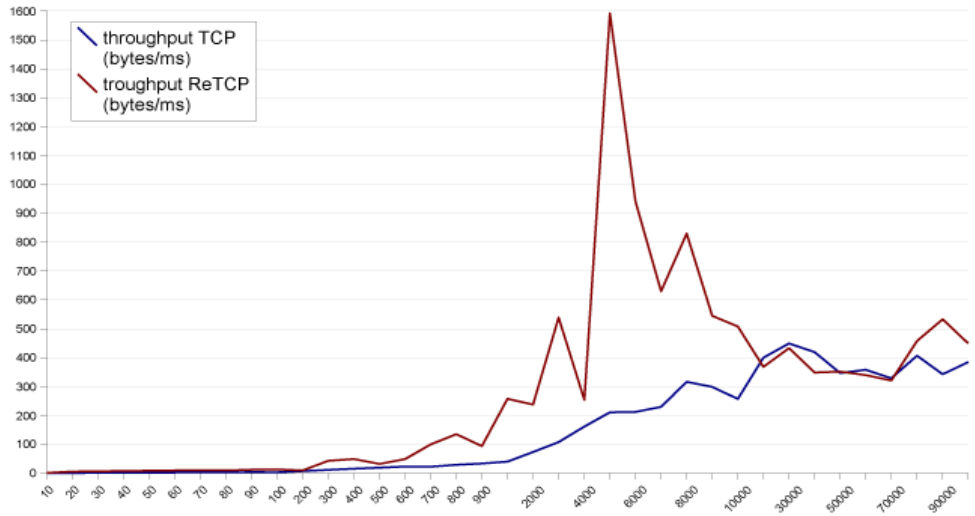
Performance is compared to existing QoS 2 impl.

- various network conditions emulated with NIST Net

Results indicate better throughput and delivery latency

- graphs show latency, throughput for network with link delay of 1.5 ms, and highly correlated loss probability of 10%





Thanx for your Attention!

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Useful resources:

- MQTT protocol information and specification:
  http://mqtt.org/
- NIST Net network emulator:
  http://www-x.antd.nist.gov/nistnet/
- this presentation, master's thesis report:
  http://n.ethz.ch/~fuchsd/ReTCP/