On Testing Network Quality Of Service

Working with MPLS test network

V. Savkin

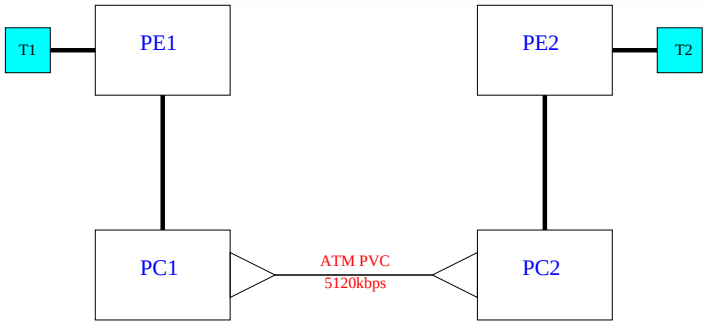
vsavkin@msu.ru

Institute for Information Security, Moscow State University

The Purpose Of The Test Network

- Testing of modern QoS mechanisms
- How they fit into MPLS infrastructure
- Whether math. models adequately describe them

The Test Network Structure



PC1–PC2 is the bottlneck link: VBR-RT circuit to model physical link

The Test Network Structure

Description of components:

- PE1, PC1 — Moscow State University
- PE2, PC2 — Russian Institute for Public Networks
- PC1–PC2 — virtual ATM circuit via MSK-IX
- T1, T2 — computers running Linux

2 networks:

- Test MPLS network
- Control network (common Internet)

Software Used

Software created for the project:

- UDP traffic generator, collector and packet logger.
- Statistical analyzer of collected logs.

Other software used:

- Iperf — for generating background TCP traffic. (`http://dast.nlanr.net/Projects/Iperf/`)

UDP traffic generator

Components:

- `cbr_test` — generator
- `cbr_sink` — collector
- `unilog.pl` — packet log producer

Order of an experiment:

1. Run `cbr_sink` on T2.
2. Run `cbr_test` on T1 and wait for it to finish.
3. Move data logs onto one PC.
4. Use `unilog.pl` to make packet log.
5. Use a statistical analyzer.

UDP traffic generator

Features:

- Constant rate, constant packet size (for now).
- Collector may answer received packets to measure RTT precisely.
- Precise timer for sending packets.
- Control on IP addresses and UDP ports used.
- Every sent or received packet logged.
- Command line interface.

Written in C and perl to be used on Linux.

StatQoS — Statistical analyzer

- Takes packet log from `unilog.pl`.
- Makes a report, calculates QoS parameters observed.

Features:

- Processes large logs effectively, as an whole or by parts.
- Calculates packet losses in each direction (T1 $\rightarrow$ T2, and T2 $\rightarrow$ T1).
- Analyzes RTT and jitter: minimum, maximum, average, dispertion, histogram.
- Dialog interface.

Tests Performed

- Calibration: checking a stability of the virtual ATM circuit, how well it can represent a dedicated physical link.
- Priority queueing.
- Fair queueing.

Calibration

- Continuous testing during 2 weeks.
- 400 packets/sec with 1200 bytes of payload each.
- Collector replied on every received packet.

Analysis of the results made possible to separate effects of QoS mechanisms, tested later, from «background noise».

Calibration results

- Average RTT was 11–12 milliseconds.
- Packet loss was less than 10^{-8}.
- There was easily detectable and measurable (with our software tools) influence by uncontrolled external factors.

Conclusion: virtual ATM circuit had measurable differences from real dedicated link, but was good enough for our tests.

Testing Priority Queueing

CBWFQ (Class-Based Fair Queueing) was used on Cisco routers to test priority queueing and fair queueing.
Test scenario:

- Low-rate UDP flow for priority class.
- Several TCP flows, generated by `Iperf`, for best-effort class. 6 and 25 flows in our tests.

Fair Queueing testing: UDP flow was put in same best-effort class.

Test Results

Observed were good QoS parameters, perfectly suitable for e.g. voice traffic.

- Average RTT was 12 milliseconds.
- No packet loss during 2 days and more than 3 million packets.
- Less than $3 \cdot 10^{-6}$ of all packets had RTT greater than 50 milliseconds.

Test Results

Practical utility of our software tools was observed: non-optimal configuration and errors of configuration were detected on test network during experiments.

Example: no good QoS for priority traffic was obtained before changing `tx-ring-limit` value for the ATM PVC.

On-going experiments and future plans

- Class-Based Queueing: static and dynamic case.
- Modeling other traffic sources: bursty, interactive, etc.
- Developing and testing full QoS-enabled network infrastructure: dynamically changing, applying and enforcing policies.
- Developing and modeling new QoS mechanisms.