GloMop QoS Preferences and Chunk Scheduling

GloMop and Daedalus Groups

This document describes how QoS Parameters from applications to GloMop map to packet scheduling decisions in the Network Monitor.

1.0 Overview

1.1 Chunk Schedulers and Network Monitors

The chunk scheduler and the network monitor are the two software components responsible for presenting the abstraction of chunks with priority classes to a client application. There is a chunk scheduler in the GloMop layer and a chunk scheduler in the proxy. The Network Monitor may live anywhere in the logical network, but is called primarily from the proxy.

The **Chunk Scheduler** on the GloMop side manages chunk scheduling and multiplexing of chunk transmissions onto multiple physical NI's. The **Network Monitor** is responsible for the translation between chunks and network packets. It maintains a transport-level connection between the mobile and the proxy for each available network interface. It handles the fragmentation of chunks into network packets when necessary, and handles reliable transmission of chunks from the mobile to the proxy. It also maintains the statistical models that indicate the current performance of each network interface's connection. Note that the actual measurement of the network may be made at the mobile, the base station, the proxy, or some combination of all three. The Network Monitor is responsible for having an accurate summary of the measurement results.

1.2 Network Monitor's Functional Parameters

These are the parameters that the chunk scheduler can ask of the network monitor:

- Max (logical) packet size: for example, the maximum packet size that is supported by the transport-level connection(UDP, TCP) between the mobile and the Proxy.
- Optimal packet size (can be used for chunkification decisions): for example, the frame size of the underlying wireless medium. **Q: Is this too great a violation of layering?**

The maximum and optimal packet sizes are used by GloMop when determining chunk sizes. They are only hints; the network monitor is free to reply with an answer of “unknown”.

- current BW for this mobile-proxy connection: note that this is a measured function of load, not a static property of the network interface.
- current roundtrip latency(per packet)
- Reliability: Whether chunks are being sent reliably. This parameter can be turned on and off at the request of the Chunk Scheduler.
- Encryption: Whether the current link supports link-layer encryption, and if so, whether encryption is enabled. The client still always has the option of doing end-to-end encryption at increased computational cost, but link-layer encryption is free when provided and may suffice for most purposes.
- Cost: The cost(in cents/dollars) for a chunk of *b* bytes, computed using an appropriate cost model for the physical NI in use.

All non-binary parameters will have confidence intervals for magnitude (the measured available bandwidth is *w* with confidence *p*) as well as time (this data is *x* seconds old). The proxy uses this information to determine the amount of distillation necessary for a chunk.

1.3 Client-Specifiable QOS

The client application fills in the following template to specify the constraints it would like met. If a template is not specified for a particular chunk, the *type-default template* for that chunk type is used instead. Type-default templates are specified at the GloMop level (i.e. global across all client applications) but can be overridden by an application.

The template specifies desired bounds on the following QOS components and the relative weights of the importance of achieving the bounds. A weight of zero indicates a parameter whose value is unimportant. A weight of W_{max} means the proxy should go to extreme lengths to meet the bound.

- Maximum end-to-end latency, including distillation and decoding time
- Maximum monetary cost (for metered services)
- Minimum quality (refinement level). As described in the *GloMop Client-Side Architecture*, this can be expressed as vector of refinement-axis parameters or as a scalar on a “gross quality scale”. The former only makes sense when the bound is tight, e.g. “Deliver a representation with exactly the following refinement parameters.”

- Maximum power consumption. Implementation of this is TBD.
- Encryption: none, use link-level if available, or end-to-end. Encryption is done separately for each chunk, and since it's logically orthogonal to other processing we defer discussion of it till later.

For example, specifying an end-to-end latency with a large weight and minimum quality with a small weight has the following semantics: "Make a strong effort to deliver a representation within the specified latency bound, if necessary by sacrificing significant quality."

1.4 Summary of Invariants

- GloMop client deals with nothing smaller than a chunk. The proxy will attempt to create chunks that fit into network packets, but the NM will transparently fragment and reassemble chunks that are too large.
- The NM does not have to provide in-order delivery of chunks. Any ordering is managed by the Chunk Scheduler.

1.5 Reservations

Because there is only a single network monitor and potentially many GloMop clients talking to it, the Network Monitor is responsible for handling bandwidth reservation and admission control. Chunk Schedulers may request some fixed bandwidth, and the Network Monitor can reserve the bandwidth or deny the request. The Network Monitor removes any reserved bandwidth before reporting available bandwidth.

2.0 Packet Scheduling

2.1 Priority Classes

The Chunk Scheduler is responsible for assigning a **priority class** to a chunk based on the following deadlines given by the application:

The Network Monitor only schedules the transmission of chunks according to priority classes, and does not take part in deadline management.

Lottery scheduling is used to schedule packets belonging to a fixed number of priority classes. Comparisons between priority classes are valid, but absolute performance within a priority class depends on the network characteristics. Sample classes are:

1. Out-of-band urgent: send this notification *now* (e.g. "I have burst into flames")
2. Priority: in-band but high priority (e.g. distilled inlines)
3. First class
4. Second class (like the Post Office)
5. Junk: send when idle (e.g. email)

This or some similar set of default classes will be specified at connection startup time. The Chunk Scheduler may specify new classes by indicating the new class's priority relative to some previously defined class. Although the Chunk Scheduler can specify new priority classes, the number of classes should be relatively static(i.e. don't define a new class for every chunk).

3.0 Using Multiple Physical NI's

When there are multiple NI's, someone must decide which chunks get scheduled on which NI's. There are two options as to who decides scheduling of chunks on NI's, and both have problems:

1. If the Chunk Scheduler decides, there is a violation of layering; the Chunk Scheduler should only present chunks to the network and expect that they get sent to the destination.
2. If the Network Monitor decides, then the statistics that it presents in Section 1.2 must somehow combine characteristics from several (possibly quite different) sources.

Chunks of the same type should behave consistently in terms of end-to-end latency, etc.

We came up with the following alternatives for the multiple-NI problem:

- Aggregate: report the "average" metrics over all NI's when requested by proxy. NM decides which NI to schedule a packet on, based on expected latency (queue sizes). Problem: aggregate performance numbers may be useless if NI's are vastly different, but Venkat points out that if this is the case, the added benefit from using the slower one in parallel with the faster one is doubtful. On the other hand, this ignores monetary cost issues.
- Spillover: use some NI (the fastest?) by default; when its expected performance drops below some threshold due to saturation, spill excess packets to the remaining NI's. Measured performance appears to change slowly. This also ignores cost issues.
- By priority class: a particular priority class always uses same NI. Problem: prone to saturation, hard to do fine-grained load balancing. Perhaps a hybrid approach will solve this, which degenerates to spillover when saturation occurs.
- By abstract per-chunk type, which is distinct from the chunk's MIME type/subtype, although the mapping may be static, and is orthogonal to priority class. Type is used to pick an NI, priority is used to order within an NI.

4.0 Other Questions

- How reliably can the Network Monitor make network measurements (i.e. how tight can the confidence intervals be)? Loose bounds could have disasterous effects; if the actual bandwidth is one half of what the NM reports, the Proxy will not distill outgoing information enough to meet application deadlines. **Q: What does TCP slow start do?**

- What happens with bandwidth reservation schemes when there is not a 1-1 mapping between traffic source and low-bandwidth link: for example, when there are many unrelated proxies competing for the same bandwidth, or (worse) when there is a non-proxy source competing with the proxy for bandwidth?
- Time scales: it is important to avoid large first derivatives in the network parameters; for example, spikes in available bandwidth. Who should be responsible for handling these spikes: the Chunk Scheduler, which must adapt slowly to changes in network characterics, or the Network Monitor, which must present statistics that are averaged across time?