37
Reservations about Reservations

Fred Baker
Cisco
fred@cisco.com

Jon Crowcroft
University College London
J.Crowcroft@cs.ucl.ac.uk

Roch Guerin
IBM
guerin@watson.ibm.com

Henning Schulzrinne (editor)
Columbia University
schulzrinne@cs.columbia.edu

Lixia Zhang
UCLA
lixia@parc.xerox.com

Abstract

The panel discussed the current state and likely future development of how to provide differentiated quality of service in an Internet context, primarily through reservations. The panel agreed that it appears unlikely that a single approach will fit all applications, with different trade-offs between complexity, level of guarantee and scaling

Keywords

Quality of Service, reservations, service differentiationa, scaling, RSVP, panel

FRED BAKER

Applications that need QoS include network telephony, teleconferencing, interactive applications, transaction applications, SNA in TCP/IP. QoS may entail controlled latency, dedicated bandwidth or improved loss characteristics. QoS can be improved by a combination of QoS routing, line protocols and queueing.

For QoS routing, getting better service is a matter of getting an appropriate route, using for example, OSPF TOS routing, IS-IS, ATM PNNI or future protocols such as QOSPF. QoS routing has been discussed periodically, is now the subject of a new IETF working group, but has historically not been important outside of specialized networks.

Line protocols such as ATM and multilink PPP are the second approach to improving QoS. ATM allows to multiplex traffic streams at high speed, with different virtual channels (VCs) for different services. Multilink PPP is able to "interrupt" traffic on (typically) low-speed lines, giving preference to some types of traffic. For ATM, we can either have a common or separate virtual channels. For separate VCs, an edge device routes traffic on VC with appropriate set of characteristics, so that each VC contains a set of flows through a given set of neighbors with matching QoS requirements. For the case of a common VC, one VC is used for a given set of neighbors, requiring advanced queueing at the edges and a VC service comparable to the needs of the flow with the most stringent QoS requirements.

Building QoS into Distributed Systems A. Campbell & K. Nahrstedt (Eds.)

PPP multilink fragmentation is used for low-speed circuits, typically T1 and below. Here, the goal is that big messages should not delay small, delay sensitive messages such as voice traffic. PPP multilink defines a message fragmentation mechanism that splits large packets into smaller fragments, with a special encapsulation, which can be used across several or a single link. The scheme suffers from line and segmentation/reassembly overhead.

The third approach to improving QoS is to employ non-FIFO queueing, based on the premise that getting better service is a matter of managing congested queues. The total latency and bandwidth across all flows are constant, but we make some traffic absorb latency and therefore give up bandwidth, shielding other traffic from latency and letting it gain bandwidth.

QoS-aware queueing algorithms can be divided into two classes, congestion management and congestion avoidance algorithms, with priority, class-based (custom) and weighted fair queueing (WFQ) examples of the former and random early detection (RED) an example of the latter. Priority queueing is commonly implemented, e.g., triggered by IP precedence bits or access lists. If not well engineered, it can cause traffic lockout; within each priority class, packets are served FIFO, with unpredictable QoS. Class-based queueing achieves a guaranteed rate or latency for particular classes of traffic by some variation on round-robin queueing, with a limit on the number of bytes removed from the queue at each round. Weighted fair queueing achieves predictable latency and bandwidth for reserved flows. In WFQ, the multiplier on the message length has flows share bandwidth *predictably* unfairly. WFQ requires more sorting than the other algorithms. For TCP, weighted random early detection can discriminate between flows, thus achieving congestion avoidance rather than congestion management. Traffic is dropped in proportion to mean queue depth and time since last discard. RED is still FIFO queueing, with no predictable QoS, which depends on host TCP behavior (throttling) for effectiveness.

Currently, RSVP can be used to set up either guaranteed flows, offering guaranteed delay by guaranteeing bit rates, or controlled load flows, with a simpler design for adaptive applications. Unused capacity is left for best-effort traffic. RSVP services can be offered in broadcast networks through a subnet bandwidth manager where flows ask for permission. For ATM, guaranteed service maps roughly into CBR, controlled load service into VBR and best-effort traffic into ABR or UBR. RSVP does not work well for many small reservations and is likely to be proven in corporate networks before it reaches the Internet backbone.

RSVP is one of the answers for QoS-enhanced traffic; it is not the only one. For voice-over-IP, I recommend using IP precedence – imagine 38,000 RSVP sessions on an OC-12. IP Precedence is also going to be used in the backbone in the near term, as it is simple and lightweight, scales and affords administrative control. However, it suffers from a lack of admission control and a coarse selection of weights. On the other hand, for unengineered edge networks and for large flows, RSVP is a reasonable answer.

JON CROWCROFT

Providing a given QoS can be done in a number of ways, essentially reflecting the time scales over which investment is made on behalf of a user, or set of users. Three points on this spectrum of ways might be:

Over-engineering selected paths: In traditional circuit switched networks, a circuit set up on demand is a version of this, albeit less flexible than a virtual path by multiple circuits, which can be shared. In an Internet, a particular route could be overprovisioned; that route might be shared by multiple networks as well as users, hosts or applications.

Subscription for selected terminals or addresses: A virtual private Internet offering improved QoS can be created by assigning resources based on address prefixes or IP network numbers. Flows from and to such sites receive preferential treatment via precedence or fair queueing.

On demand: Here, a virtual circuit or flow is set up using a signaling protocol, call admission control (CAC), traffic policing, traffic accounting, and service discrimination through scheduling.

As we progress down the spectrum, and through time in the evolution of multiservice networks, we find increasing costs and difficulties in deploying the necessary technology. We find contradictions between the requirements brought on by new technology, and our solutions to prior parts of the networking problem. Prime examples of these are

- receiver driven multicast, and traditional sender based signaling;
- resource reservation and QoS routing;
- open signaling for multi-metric QoS, and open network provisioning (e.g., how do a set of providers on a set of paths get to share out a delay budget);
- on the fly re-negotiation and IP Security;
- mobility and route pinning.

It is my contention that the costs of solving these problems outweighs the benefit, and that the balance should be redressed towards the simpler end of the QoS provisioning spectrum.

ROCH GUERIN

The main issue is: what is driving the need for reservations? It is not (mostly) the requirements of various multimedia applications. In the same fashion that videoconferencing is not the killer application, there is no single application that is so critical and has so unique requirements that the only way it can be supported is by making reservations. Or the other way around, there are so many applications with so many unique requirements, that it is impossible to define a reservation framework that will make them all happy. So, forget about the feasibility, suitability, applicability, etc. of reservation models.

The main driver is that as the Internet moves more and more towards a commercial network, people want to know exactly what they are paying for. In particular, business and commercial customers who rely on the Internet for critical parts of their business want guarantees on the level of service they get. As a result, the need for reservations is driven largely by the need for enforceable and observable service contracts. So, any reservation model where I can easily check and specify what I'm paying for will do.

Given that enforceable and observable contracts are a main driver, the next issue is what kind of (reservation/service) contracts are likely to emerge.

I contend that deterministic or pseudo-deterministic contracts are much easier to deal with. Hence, a service such Guaranteed Service should actually be easier to offer than a "fuzzier" one such as Controlled Load. This is certainly not to say that it is easier to implement, but if an important requirement is verification of compliance with the contract terms, deterministic services will prevail.

Similarly, a hard-line usage parameter control such as the ATM UPC represents a much simpler and cleaner mechanism on which to base a contract. Allowing *excess* traffic above and beyond what is specified in a contract, not only adds quite a bit of complexity in the infrastructure, but also makes for much fuzzier contracts.

Again, this is not to say that there is no need for other types of service offerings and contracts, but their evolution will largely depend on future economic trade-offs. For example, if link and switch/router bandwidth become a cheap commodity, then only some trivial traffic contract will be needed (CBR-like). This is very much like what's happening with applications in terms of memory and CPU requirements, now that these resources are less and less constrained. So, the evolution of traffic contract will also be driven by the interactions between customer requirements and the cost of network resources. This is, however, likely to be a gradual process.

To summarize, economic and contractual factors are what is driving the need for reservations. This argues, at least initially, for simple and deterministic services and contracts. Economic factors such as cost of resources and their exploitation will be the important drivers, but are many conflicting forces, e.g., cheap resources lead to simple signaling, where value is added at end-systems and not by service providers or equipment vendors. If resources are expensive, a network needs complex signaling, thus leading to providers and equipment vendors adding most of the value.

So, where are the incentives and who will push for what?

Maybe with good caching in the servers, the only reservation mechanism I need is a circuit-switched network à la ISDN...

HENNING SCHULZRINNE

Service Models

In any communications network, an expectation of a "standard service" will emerge, similar to 64 kb/s voice channel for telephony and the 16 to 24 kb/s or so effective modem throughput. Standard service cannot require reservation, in the sense that there is any likelihood of being denied that service. For customers with access bandwidth, standard service will be equal to the access bandwidth or a substantial fraction thereof, just like the telephone network. Internet telephony and RealAudio-style low-bandwidth continuous media are likely to be considered standard services.

The existing consumer ISP model is based roughly on a multiplexing model where 200–300 concurrent users share a T1, and 10–15 customers share an ISP line. With Internet telephony, radio-like services and content "pushing", the $20/month model is no longer sustainable. However, this not imply that reservations are appropriate, rather that volume-based charging may be instituted. (Already, the $20/month is typically silently limited to less than 8 hours/day.) In some cases, crude distance differentiation (domestic vs. transoceanic) may be appropriate to encourage use of local multimedia caches.

Traffic Scheduling

As mentioned elsewhere in this note, three possibilities of scheduling traffic are commonly discussed. It is fairly straightforward and instructive to compute their end-to-end delay bounds D for a network with $h = 10$ hops, connected at a rate $r = 622$ Mb/s (OC-12). Assume that the maximum packet length l_* is 1500 bytes, while the flow of interest is packet voice with a packet length of 80 bytes and a rate of $\rho_1 = 16$ kb/s.

Priority:

WFQ for individual flow: We assume that voice flows have a leaky-bucket depth of zero (i.e., are CBR).

$$D = \frac{(h-1)l}{\rho_1} + h\frac{l_*}{r} = 360\,\text{ms}$$

WFQ for all voice flows: Assume that voice traffic makes up roughly half of all Internet traffic, for $\rho_a = 300$ Mb/s. We need to allocate a bucket depth β to accommodate the case that all voice sources inject their packet at the same time.

Note that this model does not preclude individual admission control for each voice stream, although this seems unnecessary.

$$D = \frac{\beta}{\rho_a} + (h-1)\frac{(h-1)l}{\rho_a} + h\frac{l_*}{r} = 4.0\,\text{ms}$$

Priority for voice: Due to cross-traffic, a voice packet may have to wait for a burst of other voice packets at every hop.

$$D = h\left(\frac{\beta}{r} + \frac{l_*}{r}\right) = 193\,\text{ms}$$

Unfortunately, these delay bounds are fairly meaningless for delay-adaptive applications, where measures such as a 99% percentile of delay are more predictive of perceived performance. In particular, the upper bound for priority is extremely unlikely.

Resource Reservation Protocols

Thus, for reasons of service expectation and scaling, as Fred Baker pointed out above, the use of RSVP for each Internet phone call is unlikely, but the ability to use RSVP to set aside a given fraction of the bandwidth for IP packet marked with type-of-service (TOS) bits may well turn out to be useful, as it alleviates some of the starvation problems of priority queueing. It should also be added that, due to its predictable per-flow bandwidth and anticipated large volume, Internet telephony is a rather good candidate for priority queueing.

Initially, RSVP was perceived as a light-weight reservation protocol, as compared to, say, ATM signaling. However, a number of factors have made the final product far from simple. Among these are

flow merging: It is not yet clear that multiple classes, with fine-grained parameters are truly needed, as most applications can probably be fit into a relatively small set of bandwidth classes, say, increasing at factor-2 or factor-$\sqrt{2}$ increments. Flow merging also necessitates introduction of a "blockade" state to prevent killer reservations.

receiver orientation: In some cases, receivers have no direct knowledge as to the bandwidth to be delivered by the sender and, due to large-scale multicast, senders have to guess in any event what their intended audience can support. The separation of reservation and path-finding messages for receiver-oriented reservation mechanisms imposes additional processing and protocol complexity.

receiver diversity: At least for bandwidth diversity, reservations are an inappropriate means to distinguish classes of receivers as random packet dropping to thin a stream is usually not useful. In addition, other mechanisms such as layered multicast were found to be superior. In networks with sufficient bandwidth where each flow consumes only a small part of the total capacity, queueing delays should be only a

application modification: Applications need to be modified to take advantage of resource reservation, or an external agent needs to be added, both incurring complexity.

For many applications, particularly multicast delivery of streaming content, a sender-based approach may offer greater simplicity. Receiver diversity in bandwidth or latency requirements for each flow are not relevant here. (This also avoids the problem of receiver-based charging for multicast reservations, which encourages smart receivers to wait until a neighbor has paid for the bottleneck and then drop the reservation.)

One particular sender-based approach that we are studying is to use RTCP and router-alert IP options to do either burst or flow reservations, with no additional protocol overhead and automatic bandwidth-scaling. (We blithely assume that protocol layering violation is no longer a federal offense punishable by having to read a binder's worth of ITU documents.)

It appears likely that we will see a range of QoS mechanisms, both for traffic scheduling and setting up state, with different trade-offs between QoS predictability, granularity, scaling and . RSVP may end up being primarily used to "nail down" virtual private networks, with somewhat better set-up latency and fault recovery capabilities than manually doing this via SNMP.

LIXIA ZHANG

The Internet needs to provide services differentiated in quality now and in the future, just like other industries. That is not so much because some applications need a better QoS than others (just like living: you take what you can afford) but more fundamentally it is needed due to economic reasons.

Christian Huitema has said *WHERE*: "Giving everyone the very same service, as we do in the Internet today, has a definite economical consequence, a price war, that leads towards 'dirt for cheap'. If we want to pour money into the network so we can actually build it, we must make sure that whoever pays more gets more."

Differentiated services mean different packets get treated differently, and that in turn means some state inside the network. Given the necessity for state, the question becomes how to set up and maintain that state. Indeed, one could always do everything manually (say, using SNMP to set up class-based queueing (CBQ) state at *each* node), but using an automated protocol seems a much better way.

Within the above context, the amount of state and the frequency of state changes are engineering decisions, depending on whether it is cheaper to have less (coarse) state and more provisioning, or one needs a tighter control. Examples for the former include the unengineered edge links mentioned by Fred Baker, while intrinsically low bandwidth links such as wireless or modem links fall into the latter category.