PTP: Better Feedback for Adaptive Distributed Multimedia Applications on the Internet

Michael Welzl
Telecooperation Department
University of Linz
Linz, Austria

Abstract

Many distributed multimedia applications adapt to the network's condition in order to provide an acceptable QoS level at the end systems. On the Internet, practically all adaptation mechanisms are based on packet loss as a sign of congestion which means that congestions actually must be caused before they can be avoided. This situation contrasts with the fact that network administrators can simply ask for information that would drastically facilitate adaptation; therefore, we introduce the Performance Transparency Protocol (PTP), which is trying to bridge that gap by making specific performance parameters available to any Internet application. This approach resembles ATM's Explicit Rate Feedback mechanism.

1 Introduction

The original intention behind the design of the Internet was to merely support robust communication; no care was taken of delay- and bandwidth-sensitive data flows. The Internet Protocol (IP) provides a connectionless service, much to the disadvantage of real-time multimedia applications. TCP turns this into a connection-oriented service through the use of acknowledgements, but it allows no assumptions about the available bandwidth or the expected latency whatsoever. For real-time applications, two basic choices remain:

1. *Resource reservation:* Protocols such as RSVP and ST-II support the reservation of network resources to provide end systems with a certain quality of service (QoS). This involves prioritization and normally brings about charging issues that are not yet agreed upon on a global basis. Therefore, we will not take resource reservation into further regard.

2. *Make the best of what is available:* The traditional model of the Internet consists of a black-box that provides end systems with a so-called "best effort" service. No assumptions were made about its behaviour other than that a reasonable amount of data will probably be transmitted within a reasonable amount of time. A series of "congestion collapses" starting in October 1986 led to the conclusion that this model is inappropriate; the majority of the Internet's traffic consisted and still consists of TCP packets, so in conformance with the "end to end argument" and its notion that no state information should be saved within the network [5], changing the way TCP behaves was the next logical step.

   In 1988, Van Jacobson and Karels [9] summed up the changes that were made to TCP since 1986, including the now well-known "Slow-Start" algorithm. It represents the first major step towards adapting to the network, which can still be regarded as a trend for modern Internet applications. Adaptive applications are network-friendly for their own good. In particular, real-time multimedia applications can benefit from being adaptive; we will explain how such applications can be supported from within the network.

2 Bandwidth adaptation difficulties

While an adaptive application might be interested in a variety of network performance parameters (each corresponding to one or more QoS parameters) such as the probability of error occurences and expected packet delay, determination of a few specific ones remains highly critical. The currently available bandwidth will certainly be most desirable; if we refrain from using TCP — which real-time multimedia applications will usually do to avoid unnecessary delay — the simplest, and probably most widely used way to estimate it is to rely on the amount of data arriving at the receiver as an indicator.

Consider an adaptive videoconferencing tool showing a behaviour as in Figure 1, P1 and P2 being the two peak bandwidths for T0 - T1 and after T1, respectively. Basically, it has two choices:

1. Adjust its encoding mechanism to match the current network throughput, yielding the best possible quality

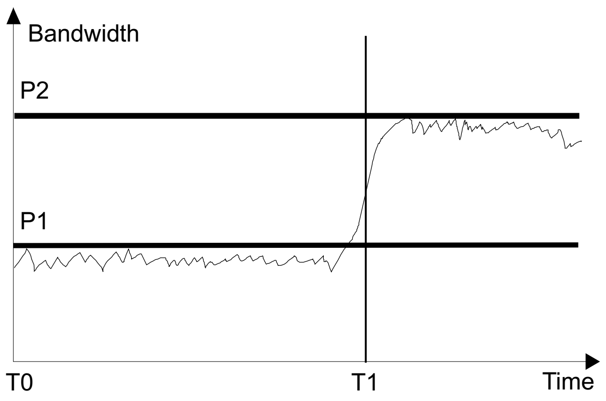

Figure 1: A videoconferencing tool's possible bandwidth behaviour

before T1 but resulting in packet loss after Tl — an effect often seen when users make spontaneous head movements; after that, it can either stick with the current mechanism and assume that after a while, the peak rate will drop back to P1 again or readjust. Currently, packets have to go all the way from the sender to the receiver and back again before the application can adjust its data rate; from the occurence of diminished bandwidth availability until feedback arrives at the sender, this means:

- Bad quality for the receiver
- Too much data being unnecessarily loaded onto the net in times of congestion

2. Presume that P2 will be needed later and switch to a suitable compression method beforehand. This will preferably be done by any application sending loss-critical data such as mathematical models or device controlling commands. Our videoconferencing software would have to send enough packets to fully utilize P2 before starting the normal transmission just to detect the available bandwidth, a process that takes time and increases net load. If the time-span between T0 and T1 is large, this test might even have to be repeated.

If the adaptive application was not dealing with real-time content (e.g. a video-on-demand server), it might be capable of looking ahead and detecting the drastic change at T1 early enough to determine whether P2 would be available in time, so it could accordingly delay the bandwidth test. Such tests are of course unnecessary if we avoid the behaviour shown in figure 1 by means of a compression method that shows a constant bit rate (CBR). H.261, having been designed for usage over ISDN, is such a method. This is still disadvantageous because it wastes bandwidth.

Constant bit rate video encoding has no advantage other than making the data stream easier to handle. Since drastic, rapid changes in a video are generally rare events, most video compression methods use the difference between two frames for encoding (this corresponds with the usage of B-frames and P-frames in MPEG [18]). Therefore, it is perfectly natural for the bandwidth to rise as there are more differences between frames — simply using a CBR compression method to compensate for adaptation difficulties may not be a reasonable solution.

There are more subtle ways to adapt: for instance, a scheme similar to Slow-Start will make the flow ”TCP-friendly”, which is regarded as highly desirable among some Internet researchers [8]. Still, such adaptation methods rely on packet loss as a sign of congestion. Congestions must actually be caused for packet loss to occur. As the basic idea behind adaptation is to relieve the network and avoid congestions, an applications first has to work against the concept in order to fulfill it.

Modern queue management algorithms, such as “Random Early Detection” (RED) [8], actually set up some form of communication between end systems and routers via packet drops. “Explicit Congestion Notification” (ECN) takes that a step further by replacing the packet drops with a notification bit [15]. This clearly demonstrates a need for real communication between end systems and the network — the black-box should rather be transparent with respect to certain performance parameters.

3 Performance Parameters

Since some of the data in question are generally regarded as being confidential, network administrators will have their concerns. They must accept that certain performance-specific information will have to be available to the public in order to make the best utilization of the network. To avoid conflicts, the performance parameters discussed in this paper will be restricted to information that can be detected by anyone through the use of rather bandwidth- or time-consuming tests anyway; actually, it is surprising how much router-specific information can already be determined from end nodes provided that it is a path's bottleneck:

- A timestamp (used for delay measurements) and address can be retrieved via the IP “Internet Timestamp” and “Record Route” options. Similar functionality is provided by traceroute.

- “pathchar” yields estimations of “nominal bandwidths” along a path [7] (we define a link's “nominal bandwidth” as its bandwidth when no packets traverse it).

- “CPROBE” is a simple tool that measures the presence of competing traffic on a path’s bottleneck link and calculates the available bandwidth from that [6].
- [12] explains how a path’s smallest MTU (“Maximum Transfer Unit”) — the so-called “PathMTU” — can be determined.

[6] and [17] make use of the so-called “Packet-Pair” approach described in [1] and [13] to determine the “Bottleneck Bandwidth” (being the minimum nominal bandwidth along a path) in conjunction with “CPROBE” and adapt more precisely. This method is very bandwidth- and time-consuming, but allows to estimate a network performance parameter without having to change the software in the routers involved. These efforts conjointly show that, from a single router’s point of view, at least the following network performance parameters should be made available to end systems with respect to the links the packets traverse:

- The MTU
- The nominal bandwidth
- The currently available bandwidth

Due to the numerous interpretations of what the “currently available bandwidth” (sometimes also called “actual instantaneous bandwidth”) is supposed to be, we exchange this parameter with:

- Some form of traffic measurement in conjunction with the router’s address and a timestamp to help end systems estimate the current bandwidth ([11], a Standard RFC, provides suitable traffic measurement objects called “ifInOctets” and “ifOutOctets”)

This way, calculation is left up to the application.

4 Administration

For network administrators, parts of the Internet are already pretty transparent: Using the “Simple Network Management Protocol” (SNMP), they can access a router’s “Management Information Base” (MIB). Among exclusively administration-specific data, such as the system’s name, contact person and the time of the last re-initialization, an MIB contains practically all our desired performance parameters.

MRTG, the “Multi Router Traffic Grapher”, is a popular tool to help administrators keep track of the data so they can watch its behaviour over a certain period of time and estimate less evident factors such as the average bandwidth utilization. Figure 2 shows an MRTG output derived from the MIB object ”ifOutOctets” [11], which is merely a counter for the number of bytes passing an interface. This output was taken from *http://www.ee.ethz.ch/stats/mrtg/ezci7-fddi.8.html*, web pages with similar outputs can be reached via the MRTG web site (*http://www.mrtg.org/*).

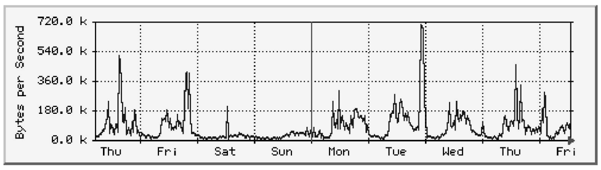

Figure 2: An MRTG trace example

Access is restricted, but it is obvious that making MRTG functionality availabe to any Internet application would be highly desirable. These parts of the MIB-2 definition [11] actually cover all the aforementioned performance parameters:

1. The “Address Translation group”
2. From the “Interfaces group”:
   (a) “IfPhysAddress” (for interface identification via the Address Translation group)
   (b) “ifMTU”
   (c) “ifInOctets” and “ifOutOctets”
   (d) “ifSpeed”

Administrators query specific single routers whereas the everyday Internet user is interested in the overall network performance, which is the worst performance packets encounter along the path in use (several paths for multicast). This means that even if the MIB rights were changed, an adaptive Internet application would first have to use traceroute and then query all the routers given by its output, making SNMP access rather inefficient.

5 The “Performance Transparency Protocol” (PTP)

The IP protocol’s “protocol” field identifies the encapsuled upper layer protocol; this IP-level identification allows routers to give certain packets special treatment without having to look at the contents of every IP packet (additionally, the IP “Router Alert” Option [10] is used). The number 123 uniquely identifies the “Performance Transparency Protocol” (PTP) as it is described in [20]. PTP allows end-nodes to retrieve performance-specific data — identified via so-called “Content Types” — from routers along a path.

The path may change at any time, so the information needs to be retrieved as quickly as possible; the use of acknowledgements as in TCP is not applicable as part of the PTP standard — most applications that use PTP will need to perform some additional communication anyway, e.g., to send computational results back to a sender. It is up to the application to use acknowledgements as part of this communication if a connection-oriented PTP service is desired. PTP supports two distinctive methods (identified through flags):

5.1 Forward Packet-Stamping

The requested information will be added to a packet by all PTP-compliant routers it encounters. The Content Types defined in [20] correspond with the aforementioned performance parameters, leaving extensions open to the "Internet Assigned Numbers Authority" (IANA) for standardization. Content Type 0, for example, identifies the MTU of the outgoing interface where the packet will be forwarded, so a receiver gets a list of MTUs with the minimum value being the PathMTU, provided that the number of datasets equals the number of hops. The IP header's "Time To Live" (TTL) field is reduced by at least one at each hop, therefore TTL_original - TTL donates the maximum number of routers a packet has encountered and can be used for this check.

Another Content Type has routers add their address, a timestamp and a value called "MIB2-ifOctets" that corresponds with the "ifOutOctets" or "ifInOctets" object mentioned above to the packets so that the receiver can build tables from the PTP packets it receives and determine the path's currently available bandwidth like it could be calculated from an MRTG trace. This is sensitive to path changes in that the tables might need to be rebuilt, but it yields a result without having to rely on packet drops. Using this Content Type somewhat regularly, an adaptive application may even detect a negative trend early enough to switch to a different compression method in time.

PTP can "survive" paths where single intermediate routers do not support the protocol by having the next PTP-compliant router add information about incoming traffic concerning the interface where the packet arrived. To detect such intermediate routers, TTL is compared with a special "TTL-Check" field (TTL-C in Figure 3) that must be set to TTL by all PTP-compliant routers on forwarding.

Figure 3 gives an example of how PTP could be used to determine the nominal bandwidth. In this diagram, a small / capital letter represents the address of an incoming / outgoing interface. The PTP packets in figure 3 are shown as they leave the network node above them; "n_BW" stands for the nominal bandwidth, the "?" in *B-?-c* stands for a single non-PTP-compliant router.

The figure also demonstrates that this method implies a certain amount of additional communication between the end systems: before the PTP packet is sent, the value 9 must be transmitted to the receiver, where the maximum amount of "missing" (not compensated for) routers is given by *max(original TTL, original TTL-Check) - TTL - DSCount* ("DSCount" being the number of datasets).

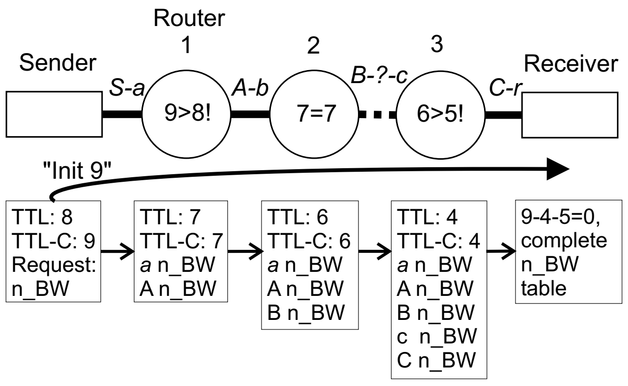

Figure 3: Nominal bandwidth determination with PTP's Forward Packet-Stamping method

If the information is incomplete (the result is not 0), it can be thought of as an upper limit for the path's capabilities. For instance, the PathMTU will certainly not be bigger than the the minimum obtained from the PTP results table. To determine the available bandwidth, the sender would have to send two more PTP packets, this time carrying a request for a traffic counter in conjunction with the interface's address and a timestamp. Subtracting the first packet's results from the second yields a table containing each link's traffic. This information can in turn be subtracted from the nominal bandwidth to calculate the available bandwidth.

A very similar method is used in [19]; this paper explains how to provide guaranteed services without having to manage per flow states.

5.2 Direct Reply

A PTP packet now contains only one dataset with values set by the sender according to its performance requirements. Each router on the way compares these values with its own. The first router that does not meet all the requirements updates the dataset and returns the packet to the sender. Here, the problem of defining "better" and "worse" with respect to performance parameters comes into play: [20] defines them as referring to "values yielding transmission results that are generally conceived in this manner. For instance, much bandwidth is good whereas much delay is bad". Therefore, a packet asking for the path's nominal bandwidth will be updated and redirected if a link shows a smaller nominal bandwidth. As PTP is designed to be a strictly stateless protocol, Direct Reply can of course not be used with traffic counters.

Obviously, this method puts a little more stress on routers through the comparison and packet redirection process, so its implementation is optional. On the other hand, it avoids the round-trip-time that would normally be necessary for an Internet application to adapt. This can be very advantageous (see bandwidth adaptation method 1) — if, for example, the router returning the packet would normally have forwarded it across a satellite link, the benefit is a lot more than half the RTT.

The concept of directly sending notification packets backwards has been discussed a lot in IETF mailing lists. There have been strong arguments against IP's "Source Quench" option and its usage within backwards ECN (the Internet BECN documentation is an expired Internet Draft and is therefore no longer available, but the idea tends to come back every once in a while [14]).

On the other hand, such mechanisms have been successfully deployed in:

- *RSVP:* Once a router cannot meet the desired QoS objectives, a "ResvErr" message indicating the problem is sent to the initiator [3].
- *Frame Relay:* Here, a variant of BECN which is based on the instantaneous queue size (as opposed to the average queue size) is used.
- *ATM:* The "Available Bit Rate" (ABR) service's "Explicit Rate Feedback" mechanism encompasses the option to generate "Resource Management" (RM) cells on the backward path [2]. Explicit Rate Feedback closely resembles PTP.

6 PTP and RTP

The "Real Time Protocol" (RTP) aids real-time applications mainly through synchronisation, payload type identification and quality feedback. While RTP directly supports adaptive multimedia applications through RTCP "Receiver Reports" [4], the way in which the network's condition is detected relies on packet loss, which in turn relies on congestions to occur. Considering that Receiver Reports are sent back to the sender in times of congestion, giving fast and reliable feedback is very important.

[17] explains how RTP was extended with the "packet pair approach" as described in [1] to determine the bottleneck bandwidth at the receiver. Then, based on the method explained in [6], the average bandwidth is calculated and sent back to the sender with the next receiver report ([17] also suggests enhancing the RTP feedback information to accordingly communicate this result).

Instead of probing the network with the enhanced RTP version, a single PTP packet carrying "ifSpeed" could be sent to the receiver. Provided that enough routers support PTP, the receiver would get a list where the minimum value represents the path's bottleneck bandwidth. In fact, the whole average bandwidth estimation could be replaced by PTP. Since it leaves end-to-end operation pretty much up to the upper layer, PTP could be invoked by RTP without having to change its API.

XTP, another protocol that supports real-time applications, is not related to RTP; its basic functionality does not contain any direct support for adaptive applications [4]. Thus, it would not make sense to combine it with PTP.

7 QoS implications

"Quality of Service" (QoS) is normally regarded as a set of requirements that must be met by the network. The traditional model — which is used in ATM and RSVP — consists of an application that enquires whether a certain service will be granted and the network that either grants or denies this service. If the service is granted, the network is expected to strictly follow the application's requirements. The concept of querying routers for performance parameters brings about new requirements featured in a different model:

As the PTP method of calculating the average bandwidth will take as least as long as the interval used, it would be desirable to have routers directly add their average bandwidth. For the time being, this is impossible because the interpretation of "average bandwidth" might vary; if you look at figure 2, you will notice that Sunday's mean will be different than the week's mean. What if the request arrives *on* Sunday? Should the last five minutes' average be chosen, or the last five hours'? Obviously, in the early Sunday afternoon, the day's average would not be representative for the rest of the day either. This is, of course, an oversimplification: Among others, [16] demonstrates that traffic prediction based on self-similar modelling can yield much better results than a moving average process.

The prediction will have to depend on the application's requirements. Some applications will prefer a value that is representative of a longer period but underlies less fluctuations, some would rather have it the other way round, depending on how quickly an application can adapt and how loss-sensitive the transmitted data is. Routers should keep track of the traffic to provide applications with an "average bandwidth" that corresponds with a specific QoS type. [21] defines two such QoS types:

1. *Short Term Expected Average:* Expected to change soon but show few fluctuations
2. *Long Term Expected Average:* Expected to last longer but show less fluctuations

For the time being, these QoS types are merely a framework for network traffic research in the context of performance feedback; depending on the mathematical model in use, separation of short- from long-term trends may or may not be feasible.

8 Conclusion

In this paper, we proposed a lightweight rapid method of feeding back a bottleneck IP router's relevant performance statistics to a sending application, so that the application can more quickly adapt to the network's condition. The proposed solution, a protocol that does signalling similar to RSVP or the Explicit Rate Feedback mechanism that we find in ATM's ABR Service, can be seen as a step towards the decoupling of packet loss and network performance notification.

The protocol described within this paper was designed to support performance feedback in an efficient (it can make up for single intermediate non-supportive routers) and flexible (end-to-end operation is left up to the application so that it could be implemented as part of RTP, for example) way. With its QoS implications, we also expect it to serve as a playground for further research.

9 Future Work

The fact that PTP leaves its usage pretty much up to the application makes it hard if not impossible to simulate or test enough cases to really see how big its benefits are. PTP not only needs real-life traffic but also real-life administrators — depending on the router vendor, some of the would-be-mandatory MIB objects that are necessary for PTP to work need to (and might never) be manually set. In the author's opinion, PTP needs thorough and widespread tests.

Participation in experiments is very welcome. By the time of writing, an implementation is in the works; along with other related information, it will be made available at *http://www.tk.uni-linz.ac.at/∼michael/ptp/*.

The *Active Networks* "NodeOS" architecture currently does not support access to a channel's properties, but this is being worked at. If all the relevant performance parameters were available to an Active Application, *Active Networks* would probably be the perfect testbed for PTP.

References

[1] Bolot, Jean-Chrysostome, "End-to-End Packet Delay and Loss Behavior in the Internet", *Proceedings of SIGCOMM 1993*, pp. 289-298. also in *Computer Communication Review* 23 (4), Oct. 1992.

[2] Bonomi, F., Fendick, K., and Giroux, N., "The Available Bit Rate Service", *The ATM Forum*, http://www.atmforum.org/

[3] Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S., "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification", *RFC 2205*, September 1997.

[4] Braun, T., and Zitterbart, M., *Hochleistungskommunikation. Band 2: Transportdienste und -protokolle*, R. Oldenbourg Verlag Mnchen Wien, 1996.

[5] Carpenter, B., "Architectural Principles of the Internet", *RFC 1958*, June 1996.

[6] Carter, R. L., and Crovella, M. E., *Measuring Bottleneck Link Speed in Packet-Switched Networks*, Technical Report BU-CS-96-006, Boston University, 1996.

[7] Downey, A. B., "Using pathchar to estimate Internet link characteristics", *Proceedings of SIGCOMM 1999*.

[8] Floyd, S., and Fall, K., "Promoting the Use of End-to-End Congestion Control in the Internet", *IEEE/ACM Transactions on Networking*, August 1999.

[9] Jacobson, V., and Karels, M. J., "Congestion Avoidance and Control", *Proceedings of SIGCOMM 1988*, pp. 314-329.

[10] Katz, D., "IP Router Alert Option", *RFC 2113*, February 1997.

[11] McCloghrie, K., and Rose, M.T., "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", STD 17, RFC 1213, March 1991.

[12] Mogul, J.C., and Deering, S.E., "Path MTU discovery", *RFC 1191*, November 1990.

[13] Paxson, V., "End-to-End Internet Packet Dynamics", *IEEE/ACM Transactions on Networking* 7(4), pp. 1-16.

[14] Peng, F., Ma, and Ma, J., "A proposal to Cooperate ECN into Wireless and Mobile Networks", *Internet Draft*, draft-fpeng-ecn-00.txt

[15] Ramakrishnan, K., and Floyd, S., "A Proposal to add Explicit Congestion Notification (ECN) to IP", *RFC 2481*, January 1999.

[16] Shu, Y., Jin, Z., Zhang, L., and Wang, L., "Traffic Prediction Using FARIMA Models", *Proceedings of ICC 1999*.

[17] Sisalem, D., and Schulzrinne, H., "The Loss-Delay Based Adjustment Algorithm: A TCP-friendly Adaptation Scheme", *Proceedings of NOSSDAV 1998*.

[18] Steinmetz, R., *Multimedia-Technologie*, Springer, 1999.

[19] Stoica, I., and Zhang, H., "Providing Guaranteed Services Without Per Flow Management", *Proceedings of SIGCOMM 1999*.

[20] Welzl, M., "The Performance Transparency Protocol (PTP)", *Internet Draft*, draft-welzl-ptp-02.txt

[21] Welzl, M., “QoS-Types for Effective Calculation and Distribution of Bandwidth Information”, *Proceedings of QoS Summit 1999*.