ELSEVIER

Computer Networks 31 (1999) 191–204

COMPUTER NETWORKS

Very large conferences on the Internet: the Internet multimedia conferencing architecture

M. Handley [a,*], J. Crowcroft [b], C. Bormann [c], J. Ott [c]

[a] *Information Sciences Institute, University of Southern California, Marina Del Rey, USA*
[b] *Department of Computer Science, University College London, London, UK*
[c] *Informatics Department, University of Bremen, Bremen, Germany*

Abstract

In this paper we provide an overview of multimedia conferencing on the Internet. The protocols mentioned are all specified elsewhere as Internet-Drafts or RFCs. Each RFC gives details of the protocol itself, how it works and what it does. This document attempts to provide the reader with an overview of how the components fit together and of some of the assumptions made, as well as some statement of direction for those components still in a nascent stage. © 1999 Published by Elsevier Science B.V. All rights reserved.

Keywords: Multimedia; Conferencing; Multicast; Quality of service; Real-time transport

1. Introduction

The Internet is not currently very good at carrying audio and video. This is hardly surprising as it was not designed or engineered with real-time traffic in mind, but there has recently been a great deal of interest in using the Internet for telephony services. Part of this has come from pricing anomalies that make Internet telephony somewhat artificially cheaper than traditional telephone services, but this is not the whole story. The Internet itself is improving to better handle traffic such as audio and video, and in the medium term, the Internet should be able to provide good quality realtime multimedia services, although such quality improvements are likely to incur additional charges.

However, the real interest in using the Internet for audio and video should come from the prospect for a single ubiquitous communications network that not only allows traditional telephony services, but also video, shared collaboration tools, and through IP Multicast, multi-party conferences and multimedia sessions that scale from small group meetings through to television-sized audiences. In principle, this may lead to a ‘‘democratization’’ of telecommunication services, where licenses to broadcast are not required to control physical access to the limited broadcast medium (although they may still be required for political reasons).

It is far from clear what services will eventually emerge using such communications capabilities. We can only say that the technical capability to have large numbers of sessions ranging in audience from hundreds to millions of participants, largely unlimited by geographic boundaries, will lead to services and social structures that do not exist today. However, we can describe the basic technologies that are likely to bring about such changes, and in this paper

* Corresponding author. E-mail: mjh@east.isi.edu.

we attempt to provide such an overview. We leave it to the reader to imagine the uses to which this technology will be put.

2. The technology

In conjunction with computers, the term ‘‘conferencing’’ is often used in two different ways: firstly, to refer to the *asynchronous* exchanges of messages between multiple users, as with bulletin boards or mailing lists; secondly, to refer to *synchronous* or so-called ‘‘real-time’’ conferencing, including audio, video, shared whiteboards and other applications. This paper is about the architecture for this latter application in an Internet environment.

There are other infrastructures for conferencing in the world: POTS (Plain Old Telephone System) networks often provide voice conferencing and phone-bridges, while ISDN provides H.320 [16] for small, strictly organised video-telephony conferencing. The architecture that has evolved in the Internet is far more general as well as being scalable to very large groups, and permits the open introduction of new media and new applications as they are devised.

The key factors of a conferencing architecture are how it supports communication between possibly large numbers of people and the real-time delivery of information. In the Internet, this is supported at a number of levels in the protocol stack. The remainder of this section provides an overview of these features, and the rest of the document describes each aspect in more detail.

In a conference, information is typically distributed to all the conference participants. Early conferencing systems used a fan-out of data streams, e.g., one connection between each pair of participants, which meant that the same information must cross some networks more than once. The Internet architecture uses the more efficient approach of *multicasting* the information to all participants (Section 3).

Multimedia conferences require real-time delivery of at least the audio and video information streams used in the conference. In an ISDN context, fixed rate circuits are allocated for this purpose—whether their bandwidth is required at any particular instance or not. On the other hand, the traditional Internet service model (‘‘best effort’’) cannot make the necessary quality of service available in congested networks. New service models are being defined in the Internet together with protocols to *reserve* capacity or *prioritise* traffic in a more flexible way than that available with circuit switching (Section 4). The nature of the Internet reflects that of the world in that it is very heterogeneous. Techniques exist to exploit this, and to deliver appropriate quality to different participants in the same conference according to their capabilities.

In a datagram network, multimedia information must be transmitted in packets, some of which may be delayed more than others. For audio and video streams to be played out at the recipient with the correct timing, information must be transmitted that allows the recipient to reconstitute this timing. A *transport protocol* with the specific functions needed for this has been defined (Section 5).

The people participating in a conference generally need to have a specific idea of the context in which the conference is happening, which can be formalized as a conference *policy*. Some conferences are essentially crowds gathered around an attraction, while others have very formal guidelines on who may take part (listen in) and who may speak at which point. In any case, initially the participants must find each other, i.e. establish communication relationships (conference *setup*, Section 7). During the conference, some conference *control* information is exchanged to implement a conference policy or at least to inform the crowd of who is present (Section 6).

In addition, *security* measures may be required to actually enforce the conference policy; for example, to control who is listening and to authenticate contributions as actually originating from a specific person. In the Internet, there is little tendency to rely on the traditional ‘‘security’’ of distribution offered by the phone system. Instead, cryptographic methods are used for encryption and authentication, which need to be supported by additional conference setup and control mechanisms (Section 8).

The protocol stacks that support Internet multimedia conferencing are shown in Fig. 1. Most of the protocols are not deeply layered unlike many protocol stacks, but rather are used in a complementary

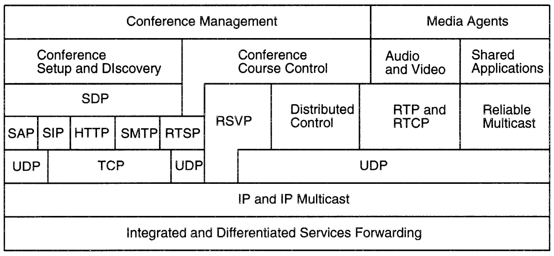


Fig. 1. Internet multimedia conferencing protocol stacks.

fashion for specific tasks to produce a complete conference.

3. Multicast traffic distribution

IP multicast provides efficient many-to-many data distribution in an Internet environment. It is easy to view IP multicast as simply an optimisation for data distribution, and indeed this is the case, but IP multicast can also result in a different way of thinking about application design. To see why this might be the case, examine the IP multicast service model, as described by Van Jacobson [9]:

- Senders just send to *the group*.
- Receivers express an interest in receiving data sent to *the group*.
- Routers conspire to deliver data from senders to receivers.

With IP multicast, the group is indirectly identified by a single IP class-D multicast address.

Several things are important about this service model from an architectural point of view. Receivers do not need to know who or where the senders are to receive traffic from them. Senders never *need* to know who the receivers are. Neither senders or receivers need care about the network topology as the network optimises delivery.

An IP multicast group is scalable because information about group membership and group changes at the IP level are kept local to routers near the relevant members. How this is performed depends on the particular multicast routing scheme in use local to the member, and although it is not a trivial task, several solutions do exist and therefore multicast routing will not be discussed in detail here. For more detailed information on multicast routing, see [1,5,7,8,22]. Typically, as a group with s senders and r receivers increases in size, state in routers scales $O(s)$ or $O(1)$ depending on the routing scheme in use. This state may be in on-tree routers for newer so called sparse-mode algorithms such as PIM, or in off-tree routers for older so-called dense-mode algorithms such as DVMRP. Thus the most scalable current multicast routing algorithms require $O(1)$ state in on-tree routers, and hence the total routing state scales $O(g)$ in a router that is on-tree for g groups. We can also envisage multicast routing schemes which require less than $O(g)$ state [1], but the requirement is not currently urgent, so none of these have yet been implemented.

The level of indirection introduced by the IP class D address denominating the group solves the distributed systems binding problem, by pushing this task down into routing; given a multicast address (and UDP port), a host can send a message to the members of a group without needing to discover who they are. Similarly receivers can ''tune in'' to multicast data sources without needing to bother the data source itself with any form of request.

IP multicast is a natural solution for multi-party conferencing because of the efficiency of the data distribution trees, with data being replicated in the network at appropriate points rather than in end-systems. It also avoids the need to configure special-purpose servers to support the session, which require support, and which cause traffic concentration and can be a bottleneck. For larger broadcast-style sessions, it is essential that data-replication be carried out in a way that only requires per-receiver network-state to be local to each receiver, and that data-replication occurs within the network. Attempting to configure a tree of application-specific replication servers for such broadcasts rapidly becomes a ''multicast routing'' problem, and thus native multicast support is a more appropriate solution.

[1] With IP encapsulation, not all on-tree routers need hold the state for a group whose traffic they are forwarding—traffic for the group can be encapsulated (either unicast of multicast) between on-tree routers nearer the edge of the network, reducing some of the state burden on backbone routers.

3.1. Address allocation

How does an application choose a multicast address to use?

In the absence of any other information, we can bootstrap a multicast application by using *well-known* multicast addresses. Routing (unicast and multicast) and group membership protocols [6] can do just that. However, this is not the best way of managing applications of which there is more than one instance at any one time.

For these, we need a mechanism for allocating group addresses dynamically, and a directory service which can hold these allocations together with some key (session information for example—see later), so that users can look up the address associated with the application. The address allocation and directory functions should be distributed to scale well.

Multicast address allocation is currently an active area of research. For many years multicast address allocation has been performed using multicast session directories (Section 7.1), but as the users and uses of IP multicast increase, it is becoming clear that a more hierarchical approach is required.

An architecture [13] is currently being developed based around a well-defined API that an application can use to request an address. The host then requests an address from a local address allocation server, which in turn chooses and reserves an unallocated address from a range dynamically allocated to the domain. By allocating addresses in a hierarchical and topologically sensitive fashion, the address itself can be used in a hierarchical multicast routing protocol currently being developed (BGMP [29]) that will help multicast routing scale more gracefully than current schemes.

4. Internet service models

Traditionally the Internet has provided so-called *best-effort* delivery of datagram traffic from senders to receivers. No guarantees are made regarding when or if a datagram will be delivered to a receiver, however datagrams are normally only dropped when a router exceeds a queue size limit due to congestion. The best-effort Internet service model does not assume FIFO queuing, although many routers have implemented this.

With best-effort service, if a link is not congested, queues will not build at routers, datagrams will not be discarded in routers, and delays will consist of serialisation delays at each hop plus propagation delays. With sufficiently fast link speeds, serialisation delays are insignificant compared to propagation delays [2].

If a link is congested, with best-effort service, queuing delays will start to influence end-to-end delays, and packets will start to be lost as queue size limits are exceeded. Real-time traffic does not cope terribly well with packet loss levels of more than a few percent, although it is possible to add redundancy [14] to increase the levels at which loss becomes a problem. In the last few years a significant amount of work has also gone into providing non-best-effort services that would provide a better assurance that an acceptable quality conference will be possible.

4.1. Non-best effort service

Real-time Internet traffic is defined as datagrams that are delay sensitive. It could be argued that all datagrams are delay sensitive to some extent, but for these purposes we refer only to datagrams where exceeding an end-to-end delay bound of a few hundred milliseconds renders the datagrams useless for the purpose they were intended. For the purposes of this definition, TCP traffic is normally not considered to be real-time traffic, although there may be exceptions to this rule.

On congested links, best-effort service queuing delays will adversely affect real-time traffic. This does not mean that best-effort service cannot support real-time traffic—merely that congested best-effort links seriously degrade the service provided. For such congested links, a ‘‘better-than-best-effort’’ service is desirable.

To achieve this, the service model of the routers can be modified. FIFO queuing can be replaced by packet forwarding strategies that discriminate differ-

[2] For slow links, a set of mechanisms has been defined that helps minimize serialisation and link access delays [2,18].

ent ‘‘flows’’ of traffic. The idea of a flow is very general. A flow might consist of ‘‘all marketing site web traffic’’, or ‘‘all file server traffic to and from teller machines’’. On the other hand, a flow might consist of a particular sequence of packets from an application in a particular machine to a peer application in another particular machine set up on request, or it might consist of all packets marked with a particular Type-of-Service bit.

There is really a spectrum of possibilities for non-best-effort service something like that shown in Fig. 2. This spectrum is intended to illustrate that between best-effort, and hard per-flow guarantees lie many possibilities for non-best-effort service, including having hard guarantees based on an aggregate reservation, assurances that traffic marked with a particular type-of-service bit will not be dropped so long as it remains in profile, and simpler priorisation-based services.

Towards the right-hand side of the spectrum, flows are typically identifiable in the Internet by the tuple: {source machine, destination machine, source port, destination port, protocol} any of which could be ‘‘ANY’’ (wildcarded).

In the multicast case, the destination is the group, and can be used to provide efficient aggregation.

Flows can be grouped in classes, where each class has an associated service model applied. This can default to best effort.

Through network management, we can imagine establishing classes of long lived flows—enterprise networks (‘‘Intranets’’) often enforce traffic policies that distinguish priorities which can be used to discriminate in favor of more important traffic in the event of overload (though in an underloaded network, the effect of such policies will be invisible, and may incur no load/work in routers).

The router service model to provide such classes with different treatment can be as simple as a priority queuing system, or it can be more elaborate.

Although best-effort services can support real-time traffic, classifying real-time traffic separately from non-real-time traffic and giving real-time traffic priority treatment ensures that this traffic sees minimum delays. Non-real-time TCP traffic tends to be elastic in its bandwidth requirements, and will then tend to fill any remaining bandwidth.

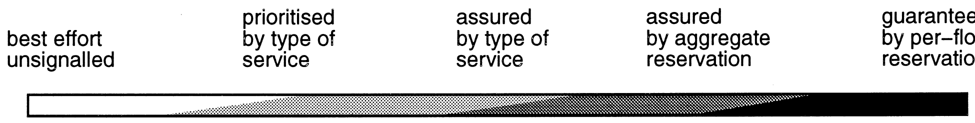
We could imagine a future Internet with sufficient capacity to carry all of the world’s telephony traffic. Since this is a relatively modest capacity requirement, it might be simpler to establish ‘‘POTS’’ as a static class which is given some fraction of the capacity overall, and then within the backbone of the network no individual call need be given an allocation (i.e. we would no longer need the call setup/tear down that was needed in the legacy POTS which was only present due to under-provisioning of trunks, and to allow the trunk exchanges the option of call blocking). The vision is of a network that is engineered with capacity for all of the non-best-effort average load sources to send without needing individual reservations.

4.2. Reservations

For flows that may take a significant fraction of the network (i.e. are ‘‘special’’ and can’t just be lumped under a static class), we need a more dynamic way of establishing these classifications. In the short term, this applies to many multimedia calls since the Internet is largely under-provisioned at the time of writing.

RSVP has been standardised for just this purpose. It provides flow identification and classification. Hosts and applications are modified to speak RSVP client language, and routers speak RSVP.

Since most traffic requiring reservations is delivered to groups (e.g. TV), it is natural for the receiver to make the request for a reservation for a flow. This has the added advantage that different receivers can make heterogeneous requests for capacity from the


best effort unsignalled
prioritised by type of service
assured by type of service
assured by aggregate reservation
guaranteed by per-flow reservation
Fig. 2. Spectrum of internet service types.

same source. Thus with appropriate codecs, RSVP can accommodate monochrome, color and HDTV receivers from a single source (also see Section 5).

Again the routers conspire to deliver the right flows to the right locations.

To support the wildcarding described earlier, RSVP can specify several different filter types. At one extreme, a filter may specify a flow of packets of a specified protocol from a specified port on a sending host to a specified port on the receiving host. Alternatively, shared reservations can be set up, where several specified senders use a single reservation, and at the other extreme, even an unspecified set of senders can share a reservation.

If a network is provisioned such that it has excess capacity for all the real-time flows using it, a simple priority classification ensures that real-time traffic is minimally delayed. However, if a network is insufficiently provisioned for the traffic in a real-time traffic class, then real-time traffic will be queued, and delays and packet loss will result. Thus in an under-provisioned network, either all real-time flows will suffer, or some of them must be given priority.

RSVP provides a mechanism by which an admission control request can be made, and if sufficient capacity remains in the requested traffic class, then a reservation for that capacity can be put in place.

If insufficient capacity remains, the admission request will be refused, but the traffic will still be forwarded with the default service for that traffic's traffic class. In many cases even an admission request that failed at one or more routers can still supply acceptable quality as it may have succeeded in installing a reservation in all the routers that were suffering congestion. This is because other reservations may not be fully utilising their reserved capacity in those routers where the reservation failed.

4.2.1. Billing

If a reservation involves setting aside resources for a flow, this will tie up resources so that other reservations may not succeed, and depending on whether the flow fills the reservation, other traffic is prevented from using the network. Clearly some negative feedback is required in order to prevent pointless reservations from denying service to other users. This feedback is typically in the form of billing.

Billing requires that the user making the reservation is properly authenticated so that the correct user can be charged. Billing for reservations introduces a level of complexity to the Internet that has not typically been experienced with non-reserved traffic, and requires network providers to have reciprocal usage-based billing arrangements for traffic carried between them. It also suggests the use of mechanisms whereby some fraction of the bill for a link reservation can be charged to each of the downstream multicast receivers.

4.3. Differentiated services

Whereas RSVP asks routers to classify packets into classes to achieve a requested quality of services, it is also possible to explicitly mark packets to indicate the type of service required. Of course, there has to be an incentive and mechanisms to ensure that ''high-priority'' is not set by everyone in all packets, and this incentive is provided by edge-based policing and by buying profiles of higher priority service. In this context, a profile could have many forms, but a typical profile might be a token-bucket filter specifying a mean rate and a bucket size with certain time-of-day restrictions.

This is still an active research area, but the general idea is for a customer to buy from their provider a profile for higher quality service, and the provider polices marked traffic from the site to ensure that the profile is not exceeded. Within a provider's network, routers give preferential services to packets marked with the relevant type-of-service bit. Where providers peer, they arrange for an aggregate higher-quality profile to be provided, and police each other's aggregate if it exceeds the profile. In this way, policing only needs to be performed at the edges to a provider's network on the assumption that within the network there is sufficient capacity to cope with the amount of higher-quality traffic that has been sold. The remainder of the capacity can be filled with regular best-effort traffic.

One big advantage of differentiated services over reservations is that routers do not need to keep per-flow state, or look at source and destination addresses to classify the traffic, and this means that routers can be considerably simpler. Another big advantage is that the billing arrangements for differ-

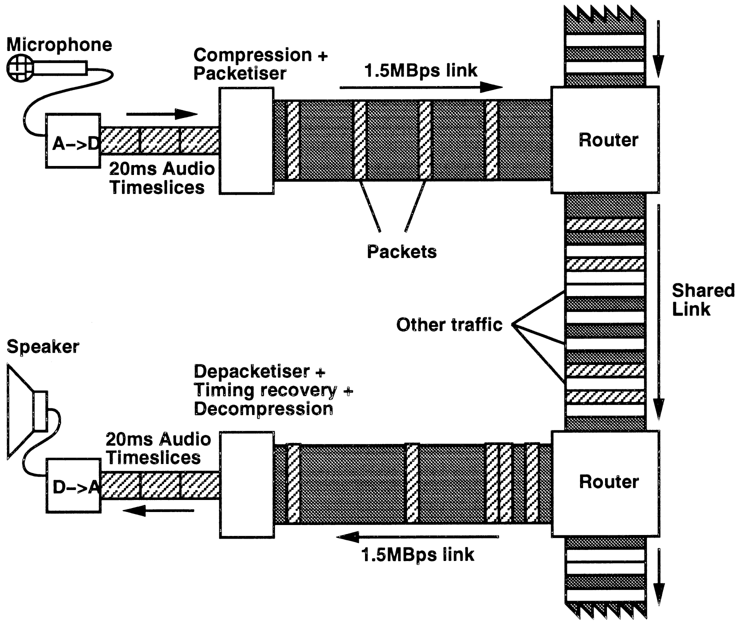


Fig. 3. Network jitter and packet audio.

entiated services are pairwise between providers at boundaries—at no time does a customer need to negotiate a billing arrangement with each provider in the path [3].

5. Transport protocols

So-called real-time delivery of traffic requires little in the way of transport protocol. In particular, real-time traffic that is sent over more than trivial distances is not retransmittable.

With packet multimedia data there is no need for the different media comprising a conference to be carried in the same packets. In fact it simplifies receivers if different media streams are carried in separate flows (i.e., separate transport ports and/or separate multicast groups). This also allows the different media to be given different quality of service. For example, under congestion, a router might preferentially drop video packets over audio packets. In addition, some sites may not wish to receive all the media flows. For example, a site with a slow access link may be able to participate in a conference using only audio and a whiteboard whereas other sites in the same conference with more capacity may also send and receive video. This can be done because the video can be sent to a different multicast group than the audio and whiteboard. This is first step towards coping with heterogeneity by allowing the receivers to decide how much traffic to receive, and hence allowing a conference to scale more gracefully.

5.1. Receiver adaptation and synchronisation

Best-effort traffic is delayed by queues in routers between the sender and the receivers. Even reserved priority traffic may see small transient queues in routers, and so packets comprising a flow will be delayed for different times. Such delay variance is known as jitter, and is illustrated in Fig. 3.

Real-time applications such as audio and video need to be able to buffer real-time data at the receiver for sufficient time to remove the jitter added by the network and recover the original timing relationships between the media data. In order to know

[3] With reservations there may be ways to avoid this too, but they're somewhat more difficult given the more specific nature of a reservation.

how long to buffer for, each packet must carry a timestamp which gives the time at the sender when the data was captured. Note that for audio and video data timing recovery, it is not necessary to know the absolute time that the data was captured at the sender, only the time relative to the other data packets.

As audio and video flows will receive differing jitter and possibly differing quality of service, audio and video that were grabbed at the same time at the sender may not arrive at the receiver at the same time. At the receiver, each flow will need a playout buffer to remove network jitter. Inter-flow synchronisation can be performed by adapting these playout buffers so that samples/frames that originated at the same time are played out at the same time (Fig. 4). This requires that the time base of different flows from the same sender can be related at the receivers, e.g. by making available the times at which each of them was captured relative to a common clock.

5.2. RTP

The transport protocol for real-time flows is RTP [27]. This provides a standard format packet header which gives media specific timestamp data, as well as payload format information and sequence numbering amongst other things. RTP is normally carried using UDP. It does not provide or require any connection setup, nor does it provide any enhanced reliability over UDP. For RTP to provide a useful media flow, there must be sufficient capacity in the relevant traffic class to accommodate the traffic. How this capacity is ensured is independent of RTP.

Every original RTP source is identified by a source identifier, and this source id is carried in every packet. RTP allows flows from several sources to be mixed in gateways to provide a single resulting flow. When this happens, each mixed packet contains the source ids of all the contributing sources.

RTP media timestamp units are flow specific—they are in units that are appropriate to the media flow. For example, 8 kHz sampled PCM encoded audio has a timestamp clock rate of 8 kHz. This means that inter-flow synchronisation is not possible from the RTP timestamps alone.

Each RTP flow is supplemented by Real-Time Control Protocol (RTCP) packets. There are a number of different RTCP packet types. RTCP packets provide the relationship between the realtime clock at a sender and the RTP media timestamps so that inter-flow synchronisation can be performed, and they provide textual information to identify a sender in a conference from the source id.

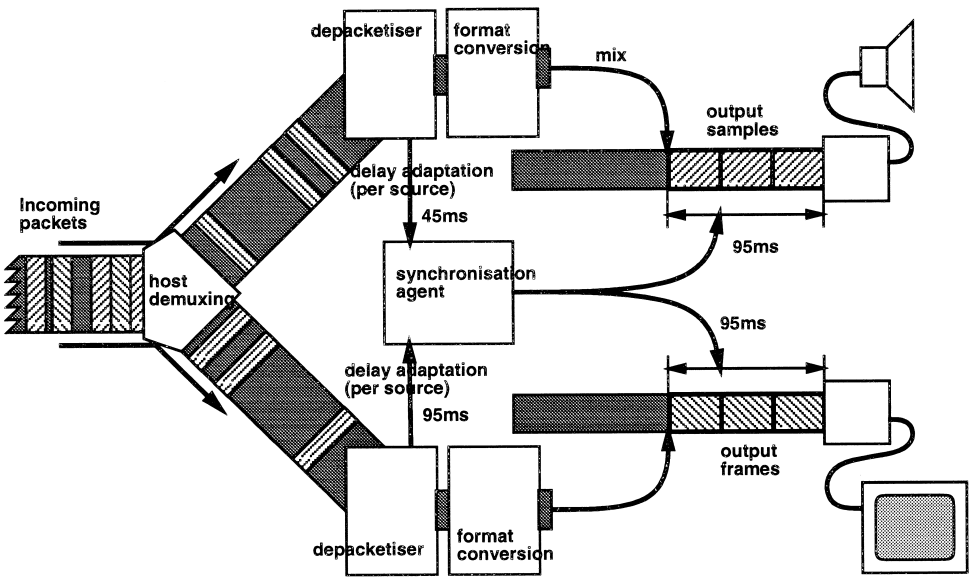


Fig. 4. Inter-media synchronisation.

5.3. Conference membership and reception feedback

IP multicast allows sources to send to a multicast group without being a receiver of that group. However, for many conferencing purposes it is useful to know who is listening to the conference, and whether the media flows are reaching receivers properly. Accurately performing both these tasks restricts the scaling of the conference. IP multicast means that no-one knows the precise membership of a multicast group at a specific time, and this information cannot be discovered, as to try to do so would cause an implosion of messages, many of which would be lost [4]. Instead, RTCP provides approximate membership information through periodic multicast of session messages which, in addition to information about the recipient, also give information about the reception quality at that receiver. RTCP session messages are restricted in rate, so that as a conference grows, the rate of session messages remains constant, and each receiver reports less often. A member of the conference can never know exactly who is present at a particular time from RTCP reports, but does have a good approximation to the conference membership. This is analogous to what happens in a real-world meeting hall; the meeting organisers may have an attendance list, but if people are coming and going all the time, they probably do not know exactly who is in the room at any one moment.

Reception quality information is primarily intended for debugging purposes, as debugging of IP multicast problems is a difficult task. However, it is possible to use reception quality information for rate adaptive senders, although it is not clear whether this information is sufficiently timely to be able to adapt fast enough to transient congestion.

5.4. Scaling issues and heterogeneity

The Internet is very heterogeneous, with link speeds ranging from 14.4 kbit/s up to 1.2 Gbit/s, and very varied levels of congestion. How then can a single multicast source satisfy a large and heterogeneous set of receivers?

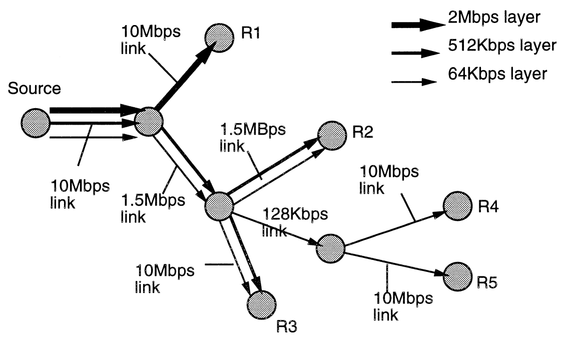


Fig. 5. Receiver adaptation using multiple layers and multiple multicast groups.

In addition to each receiver performing its own adaptation to jitter, if the sender layers [21] its video (or audio) stream, different receivers can choose to receive different amounts of traffic and hence different qualities. To do this the sender must code their video as a base layer (the lowest quality that might be acceptable) and a number of enhancement layers, each of which adds more quality at the expense of more bandwidth. With video, these additional layers might increase the framerate or increase the spatial resolution of the images or both. Each layer is sent to a different multicast group, and receivers can decide individually how many layers to subscribe to. This is illustrated in Fig. 5. Of course, if receivers in a conference are going to respond to congestion in this way, then we also need to arrange that the receivers behind a common bottleneck tend to respond in a synchronised manner. If this is not done, de-synchronised experiments by different receivers will have the net effect that too many layers are always being drawn through a common bottleneck. RLM [20] is one way that this might be achieved, although there is continuing research in this area.

6. Conference control

Conferences come in many shapes and sizes, but there are only really two models for conference control: light-weight sessions and tightly coupled conferencing. For both models, rendezvous mechanisms are needed. Note that the conference control

[4] Note that a conference policy that restricts conference membership can be implemented using encryption and restricted distribution of encryption keys, of which more later.

model is orthogonal to issues of quality of service and network resource reservation, and it is also orthogonal to the mechanism for discovering the conference [25].

Light-weight sessions are multicast based multimedia conferences that lack explicit conference membership control and explicit conference control mechanisms. Typically a lightweight session consists of a number of many-to-many media streams supported using RTP and RTCP using IP multicast. [5] Typically, the only conference control information needed during the course of a light-weight session is that distributed in the RTCP session information, i.e. an approximate membership list with some attributes per member.

Tightly coupled conferences may also be multicast based and use RTP and RTCP, but in addition they have an explicit conference membership mechanism and may have an explicit conference control mechanism that provides facilities such as floor control.

The most widely used tightly coupled conference control protocols suitable for Internet use are those belonging to the ITU's H.323 family [17]. However it should be noted that this is inappropriate for large sessions where scaling problems will be introduced by the conference control mechanisms.

In order to try and address this, the ITU is currently standardising H.332, which is essentially a small tightly coupled H.323 conference with a larger lightweight-sessions-style conference listening in as passive participants. It is not yet clear whether H.332 will see large scale acceptance, as its benefits over a simple lightweight session are not terribly obvious. It seems likely that lightweight sessions combined with stream authentication (Section 8.3) might be a more appropriate solution for many potential users.

6.1. Controlling multimedia servers

The Real-Time Stream-control Protocol (RTSP) [28] provides a standard way to remote control a multimedia server. While primarily aimed at web-based media-on-demand services, RTSP is also well suited to provide VCR-like controls for audio and video streams, and to provide playback and record functionality of RTP data streams. A client can specify that an RTSP server plays a recorded multimedia session into an existing multicast-based conference, or can specify that the server should join the conference and record it.

6.2. Protocols for non-A/V applications

Applications other than audio and video have evolved in Internet conferencing, e.g. Imm [4], Wb [9], NTE [10]. Such applications can be used to substitute for meeting aids in physical conferences (whiteboards, projectors) or replace visual and auditory cues that are lost in teleconferences (e.g., a speaker list application); they also can enable new styles of joint work.

Most non-A/V applications have in common that the application protocol is about establishing and updating a shared state. Loss of information is often not acceptable, so some form of multicast reliability is required. The applications' requirements differ: Some applications make per-participant additions to the shared state that are orthogonal to each other (e.g., whiteboards), some evolve a more closely interrelated common state (e.g., additions to a speaker list must be properly sequenced). Some applications can make use of added bandwidth/react to congestion in an elastic way, others transport data that, although not strictly real-time, is time-critical.

In the IRTF research group on Reliable Multicast [15], work is in progress on common protocol elements that can be used in such applications. At the time of writing, some aspects of reliable multicast are not well-understood, such as the proper way to provide congestion control in a multicast environment. As congestion control is considered an essential element, standards track protocols are not expected before this can be solved.

7. Conference setup

There are two basic forms of conference discovery mechanism. These are session advertisement and

[5] There is some confusion on the term session, which is sometimes used for a conference and sometimes for a single media stream transported by RTP. In this paper, we prefer to use the less ambiguous term conference except where existing protocols use the term session.

session invitation. Session advertisements are provided using a session directory, and inviting a user to join a session is provided using a session invitation protocol such as SIP [12,26] or H.323 [17].

7.1. Session directories

The rendezvous mechanism for many light-weight sessions is a multicast based session directory. This ‘‘broadcasts’’ session descriptions [11] to all the potential session participants. These session descriptions provide an advertisement that the session will exist, and also provide sufficient information including multicast addresses, ports, media formats and session times so that a receiver of the session description can join the session. The session description protocol (SDP) describes the content and format of a multimedia session, and the session announcement protocol (SAP) is used to distribute it to all potential session recipients.

This mechanism can also be applied to advertised tightly coupled sessions, and only requires that additional information about the mechanism to use to join the session is given. However, as the number of sessions in the session directory grows, we expect that only larger-scale public sessions will be announced in this manner, and smaller, more private sessions will tend to use direct invitation rather than advertisement.

7.2. Session invitation

Not all sessions are advertised, and even those that are advertised may require a mechanism to explicitly invite a user to join a session. Such a mechanism is required regardless of whether the session is a lightweight session or a more tightly coupled session, although the invitation system must specify the mechanism to be used to join the session.

The Session Initiation Protocol (SIP) [12,26] provides a mechanism whereby a user can be invited to participate in a conference. SIP does not care whether the session is already ongoing, or is just being created, and it doesn’t care whether the conference is a small tightly coupled session or a huge broadcast —it merely conveys an invitation to a user in a timely manner, inviting them to participate, and provides enough information for them to be able to know what sort of session to expect. Thus although SIP can be used to make telephone-style calls, it is by no means restricted to that style of conference.

As users are mobile, it is important that such an invitation mechanism is capable of locating and inviting a user in a location independent manner. Thus user addresses need to be used as a level of indirection rather than routing a call to a specific terminal. The invitation mechanism should also provide for alternative responses, such as leaving a message or being referred to another user, should the invited user be unavailable.

Although SIP is also somewhat lighter weight than H.323 call signalling, the key feature of SIP that distinguishes it from other invitation mechanisms is this support for user mobility. It is explicitly designed to be relayed via proxies, which are typically closer to the caller and so have more up-to-date information than the caller. Several proxies may sometimes relay the call before it reaches the caller’s current location. Without careful design, this could potentially be a security problem, allowing the tracking of a user’s location. SIP is designed from the outset to allow the careful restriction of how location information is released, whilst still allowing the user to be callable.

8. Security

There is a temptation to believe that multicast is inherently less private than unicast communication since the traffic visits so many more places in the network. In fact, this is not the case except with broadcast and prune type multicast routing protocols [5]. However, IP multicast does make it simple for a host to anonymously join a multicast group and receive traffic destined to that group without the other senders’ and receivers’ knowledge. If the application requirement (conference policy) is to communicate between some defined set of users, then strict privacy can only be enforced in any case through adequate end-to-end encryption.

RTP specifies a standard way to encrypt RTP and RTCP packets using private key encryption schemes such as DES [23]. It also specifies a standard mechanism to manipulate plain text keys using MD5 [24] so that the resulting bit string can be used as a DES key. This allows simple out-of-band mechanisms such as privacy-enhanced mail to be used for encryption key exchange [19].

8.1. Authentication and key distribution

Key distribution is closely tied to authentication. Conference or session directory keys can be securely distributed using public-key cryptography on a one-to-one basis (by e-mail, a directory service, or by an explicit conference setup mechanism), but this is only as good as the certification mechanism used to certify that a key given by a user is the correct public key for that user. Such certification mechanisms [3] are however not specific to conferencing, and it looks likely that certificates such as those provided by PGP will be most widely used in the near term.

Session keys can be distributed using encrypted Session Descriptions carried in SIP session invitations, or in encrypted session announcements as described below. Neither of these mechanisms provide for changing keys during a session as might be required in some tightly coupled sessions, but they are probably sufficient for many used in the context of lightweight sessions.

Even without privacy requirements in the conference policy, strong authentication of a user is required if making a network reservation results in usage based billing.

8.2. Encrypted session announcements

Session Directories can make encrypted session announcements using private key encryption, and carry the encryption keys to be used for each of the conference media streams in the session. Whilst this does not solve the key distribution problem, it does allow a single conference to be announced more than once to more than one key-group, where each group holds a different session directory key, so that the two groups can be brought together into a single conference without having to know each other's keys.

8.3. Secured ‘‘broadcasts’’

While private-key encryption is sufficient to exclude non-members from sending or receiving multicast conference traffic, it does mean that all members of a session are equal. This is normally acceptable for multi-way conferences, but will not be acceptable for many broadcasters who require the ability to ensure that only they can send, perhaps in addition to ensuring that only their paid customers can receive. This is nicely illustrated by the multicast of the Rolling Stones concert in 1994 which was billed as being the first live concert on the Mbone. In fact, this honour goes to a little known band called Severe Tire Damage who had multicast an impromptu concert a year previously. To make their point, just before the Stones were due to go on stage, Severe Tire Damage suddenly started broadcasting one of their songs live to the same multicast group. Clearly commercial broadcasters want to avoid occurrences like this one.

Such secured broadcasts can be performed by encrypting a hash (digitally signing) of each packet with the senders private key of a public-private key pair. The public key is then given to the receivers, and they discard (and prune if possible) any packets that are unsigned. The problem with this is that even encrypting a 128 bit hash with a public key algorithm can be relatively expensive to perform at high packet rates sometimes seen with video. The use of public-key cryptography for this purpose has not yet been standardised, but some such mechanism will clearly be needed before the Mbone becomes an acceptable environment for commercial broadcasters.

9. Summary

This document is an attempt to gather together in one place the set of assumptions behind the design of the Internet Multimedia Conferencing architecture, and the services that are provided to support it.

Fig. 6 shows an example time sequence involved in setting up a light-weight session between two sites. In this case, site A creates a session advertisement, and some time later starts sending a media stream even though there may be no receiver at that time. Some time later, site B joins the session (the multicast routing protocol here is PIM), and starts to receive the traffic. At the earliest opportunity site B also makes an RSVP reservation to ensure the flow quality is satisfactory. This example should be taken as illustrative only—there are different ways to join

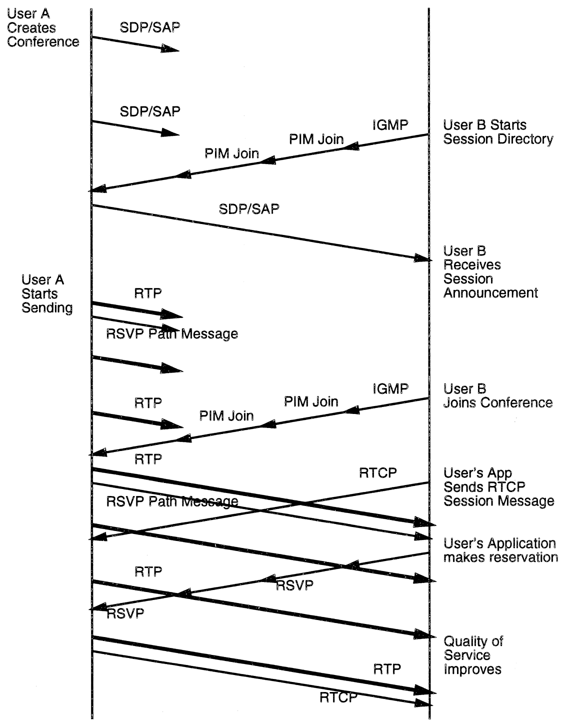

User A Creates Conference
SDP/SAP
SDP/SAP
IGMP
User B Starts Session Directory
PIM Join
PIM Join
SDP/SAP
User B Receives Session Announcement
User A Starts Sending
RTP
RSVP Path Message
IGMP
User B Joins Conference
RTP
PIM Join
PIM Join
RTP
RTCP
User's App Sends RTCP Session Message
RSVP Path Message
User's Application makes reservation
RTP
RSVP
RSVP
Quality of Service Improves
RTP
RTCP

Fig. 6. Joining a light-weight multimedia session.

sessions, and different ways to get improved quality of service.

The lightweight sessions model for Internet multimedia conferencing may not be appropriate for all conferences, but for those sessions that do not *require* tightly-coupled conference control, it provides an elegant style of conferencing that scales from two participants to millions of participants. It achieves this scaling by virtue of the way that multicast routing is receiver driven, keeping essential information about receivers local to those receivers. Each new participant only adds state close to them in the network. It also scales by not requiring explicit conference join mechanisms; if everyone were to need to know exactly who is in the session at any time, the scaling would be severely adversely affected. RTCP provides membership information that is accurate when the group is small, and increasingly only a statistical representation of the membership as the group grows. Security is handled through the use of encryption rather than through the control of data distribution.

For those that require tightly coupled conferences, solutions such as H.323 are emerging there too.

There are still many parts of this architecture that are incomplete, and are still the subject of active research. In particular, *differentiated services* for better-than-best-effort service show great promise to provide a more scalable alternative to individual reservations. *Multicast routing* scales well to large groups, but scales less well to large numbers of groups; we expect this will become the subject of significant research over the next few years. *Multicast congestion control* mechanisms are still a research topic, although in the last year several schemes have emerged that show promise. *Layered codecs* show great promise to allow conferences to scale in the face of heterogeneity, but the join and leave mechanisms that allow them to perform receiver-based congestion control are still being examined. We have several working examples of *reliable-multicast-based shared applications*; the next few years should see the start of standardisation work in this area as appropriate multicast congestion control mechanisms emerge. Finally a complete *security architecture* for conferencing would be very desirable; currently we have many parts of the solution, but are still waiting for an appropriate key-distribution architecture to emerge from the security research community.

The Internet Multimedia Conferencing architecture and the Mbone have come a long way from their early beginnings on the DARTnet testbed in 1992. The picture is not yet finished, but it has now taken shape sufficiently that we can see the form it will take. Whether or not the Internet does evolve into the single communications network that is used for most telephone, television, and other person-to-person communication, only time will tell. However, we believe that it is becoming clear that if the industry decides that this should be the case, the Internet should be up to the task.

Acknowledgements

Acknowledgments are due to the End-to-End Research Group, the Int-serv, RSVP, MMUSIC and AVT working groups of the IETF, and discussion with colleagues at UCL. The earliest clear exposition of some of the ideas here was presented at ACM SIGCOMM 1994 in London by Van Jacobson.

References

[1] A. Ballardie, P. Francis, J. Crowcroft, An architecture for scalable inter-domain multicast Routing, ACM SIGCOMM 1993, pp. 85–95.

[2] C. Bormann, Providing Integrated Services over Low-bitrate Links, Internet-Draft draftietf-issll-isslow-03.txt, Work in Progress, March 1998.

[3] CCITT (Consultative Committee on International Telegraphy and Telephony), Recommendation X.509: The Directory – Authentication Framework, 1988.

[4] W. Dang, Reliable file transfer in the multicast domain, University of Hawaii Technical Report, August 1993.

[5] S. Deering, C. Partridge, D. Waitzman, Distance Vector Multicast Routing Protocol, IETF RFC 1075, November 1988.

[6] S. Deering, Multicast routing in inter-networks and extended LANs, ACM SIGCOMM 88, August 1988, pp. 55–64 and Host Extensions for IP Multicasting, IETF RFC 1112.

[7] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, L. Wei, An architecture for wide area multicast routing, ACM SIGCOMM 1994, London, October 1994, Computer Communications Review 24 (4) (1994) 126–135.

[8] Estrin, Farinacci, Helmy, Thaler, Deering, Handley, Jacobson, Liu, Sharma, Wei, Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification, IETF RFC 2117.

[9] S. Floyd, V. Jacobson, S. McCanne, C.-G. Liu, L. Zhang, A reliable multicast framework for light-weight sessions and application level framing, ACM SIGCOMM 1995, pp. 342–356.

[10] M. Handley, J. Crowcroft, Network Text Editor (NTE): A scalable shared text editor for the MBone, Proc. ACM SIGCOMM 97, Cannes, France, 1997.

[11] M. Handley, V. Jacobson, SDP: Session Description Protocol, IETF RFC 2327, April 1998.

[12] M. Handley, H. Schulzrinne, E. Schooler, SIP: Session Initiation Protocol, IETF Internet-Draft, Work in Progress.

[13] M. Handley, D. Thaler, D. Estrin, The Internet Multicast Address Allocation Architecture, IETF Internet-Draft, Work in Progress.

[14] V. Hardman, A. Sasse, M. Handley, A. Watson, Reliable audio for use over the Internet, Proc INET '95, Hawaii, Internet Society, Reston, VA, 1995.

[15] IRTF Research Group on Reliable Multicast, http://www.east.isi.edu/RMRG/.

[16] ITU, Recommendation H.320: Narrow-band Visual Telephone Systems and Terminal Equipment, ITU, Geneva, July 1997.

[17] ITU, Recommendation H.323: Visual Telephone Systems and Equipment for Local Area Networks which Provide a Non-guaranteed Quality of Service, ITU, Geneva, November 1996.

[18] V. Jacobson, Congestion avoidance and control, ACM SIGCOMM 88, August 1988.

[19] J. Linn, Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures, IETF RFC 1421, February 1993.

[20] S. McCanne, V. Jacobson, M. Vetterli, Receiver-driven layered multicast, ACM SIGCOMM, Stanford, CA, August 1996, pp. 117–130.

[21] S. McCanne, M. Vetterli, Joint source/channel coding for multicast packet video, Proc. IEEE Int. Conf. on Image Processing, Washington, DC, October 1995.

[22] J. Moy, Multicast Extensions to OSPF, IETF RFC 1584, March 1994.

[23] National Institute of Standards and Technology (NIST), FIPS Publication 46-1: Data Encryption Standard, January 22, 1988.

[24] R. Rivest, The MD5 Message-Digest Algorithm, RFC 1321, MIT Laboratory for Computer Science and RSA Data Security, April 1992.

[25] E. Schooler, A distributed architecture for multimedia conference control, ISI Research Report ISI/RR-91-289, November 1991. ftp://ftp.isi.edu/pub/hpcc-papers/mmc/mmcc.ps.

[26] H. Schulzrinne, Personal mobility for multimedia services in the Internet, IMDS'96, March 4–6, 1996.

[27] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson RTP: A Transport Protocol for Real-Time Applications, IETF RFC 1889.

[28] H. Schulzrinne, A. Rao, R. Lanphier, Real Time Streaming Protocol (RTSP), IETF RFC 2326, April 1998.

[29] K. Kumar, P. Radoslavov, D. Thaler, C. Alaettinoglu, D. Estrin, M. Handley, The MASC/BGMP architecture for inter-domain multicast routing, Proc. ACM SIGCOMM 98, Vancouver, Canada, September 1998.