

Source: BT
Title: Changing Network Bandwidth Usage Within an H.323 Conference
Purpose: Proposal

Introduction

Gatekeeper based bandwidth allocation/reservation is an important issue in H.323. The current H.323 description allows a terminal to allocate bandwidth at call setup time, but the currently defined methods of changing bandwidth (allocating bandwidth based on the aggregate of the maximum that any stream will use, or opening and closing H.245 logical channels) do not conveniently allow a terminal's bandwidth allocation to be changed part way through a conference. This contribution discusses why the current proposals for changing bandwidth should be avoided, and suggests an alternative proposal.

Discussion

Below is a discussion on why it is felt that the currently proposed methods for changing bandwidth (allocating bandwidth based on the aggregate of the maximum that any stream will use, or opening and closing H.245 logical channels) within H.323 are inadequate.

- 1) The bandwidth requested from the gatekeepers should be as accurate as possible. At some point the aggregate bandwidth from all the terminals will be compared with a fixed threshold. If terminals reserve more than they need, then, either, fewer simultaneous calls can be supported, or the gatekeeper has to have a higher threshold set than it would otherwise. The first case is very undesirable, and the second case tends to work against the purpose of the gatekeeper. Therefore, reserving the sum of the maximum bandwidths should be avoided.
- 2) Having an audio packet of a different coding algorithm arriving at a decoder is potentially not a problem. You would buffer them as they come in for out of order delivery correction (if necessary). You would then pass them to a decoder block. This would start with a switch statement (or language equivalent) to select the desired decoding algorithm and pass the data block to the algorithm. There are cases where the relevant decoder block is not in memory, but this is an implementation issue. For a PC implementation on-demand loading of decode algorithms would probably be an unnecessary exercise (although the appropriate code segment could be swapped to disk). For DSPs, G.711 is a trivial algorithm, and could easily be co-resident with any other algorithm. For algorithms like G.722 and G.723, running on modern DSPs, you would probably have time to download the code and decode a block all in one period. For a smooth change with G.728 you might have to have the code in local DSP memory. SRAM is quite cheap now-a-days, and if a vendor wishes to support a timely change of decoding algorithm and is willing to pay (/charge the customer) a few extra pounds (dollars/Marks/Francis), then the standard should not prevent this. Therefore, opening and closing of logical channels does not help with cleanly changing audio modes.
- 3) While audio mode changes do happen fairly rarely, they do tend to happen at the start of a call. Customers find it very distracting if audio is in some way distorted at the beginning of a call. Therefore, audio mode changes should be made as quickly as possible, and not rely on opening and closing logical channels.
- 4) We are re-interpreting the definition of the maxBitRate in the data and video capability parts of the OpenLogicalChannel messages to mean 'this is the bandwidth required', rather than 'this is the decoder capability required'. This could cause confusion in the future. Therefore, we should not rely on the maxBitRate field in the H.245 OpenLogicalChannel messages for allocating terminal bandwidth.

- 5) The H.245 flow control message provides the sort of control that is sought, but specifies the bit rate to be used in the direction recipient to transmitter of the message. This is the opposite direction to what is required. Also, it does not support acknowledgement and rejection depending on whether the change in bandwidth can be supported. Therefore, the H.245 flow control mechanism can not be used, but it does provide a template for the new message that is needed.

Proposal

A better way to solve this problem is to add a new message to H.245, which for the purposes of this discussion shall be called 'MaxTxBitRate', with associated acknowledgements and rejects. This would have as a parameter the new maximum that the sending terminal would like to transmit when operating in a unicast mode. Extra fields are needed to support transmission of multicast streams, as remote terminals may or may not be receiving multicast streams. The ASN.1 of the message would be:

```

MaxTxBitRate                               ::=SEQUENCE
{
    sequenceNumber                          SequenceNumber,
    unicastBitRate                          CHOICE
    {
        maximumBitRate                    INTEGER (0..16777215),  -- units 100
                                                                    --bits per
                                                                    --second
        noRestriction                     NULL
    } OPTIONAL,
    multicastBitRate                       SEQUENCE OF
    {
        maximumBitRate                    INTEGER (0..16777215),  -- units 100
                                                                    --bits per
                                                                    --second
        multicastAddress                  LANAddress
    } OPTIONAL,
    ...
}

```

And the associated acknowledgement and rejection:

```

MaxTxBitRateAck                             ::=SEQUENCE
{
    sequenceNumber                          SequenceNumber,
    ...
}

MaxTxBitRateReject                         ::=SEQUENCE
{
    sequenceNumber                          SequenceNumber,
    cause                                  CHOICE
    {
        toBeDecided                       NULL,
        ...
    },
    maxBitRate                             INTEGER (0..16777215),  -- units 100
                                                                    --bits per
                                                                    --second
    ...
}

```

This would be used in the following manner for the unicast case:

- 1) Terminal 1 does BRQ to gatekeeper 1 asking for bandwidth.

- 2) Gatekeeper 1 agrees and sends BCF to terminal 1.
- 3) Terminal 1 sends the new H.245 message to its H.245 end-point (and eventually to terminal 2) specifying the new bandwidth.
- 4) Terminal 2 sends a BRQ to its gatekeeper and gets permission to go ahead.
- 5) Terminal 1 sends acknowledgement to the H.245 MaxTxBitRate message.

As long as the sending terminal's aggregate bandwidth remains within this bandwidth the terminal would not have to submit further bandwidth requests.

For the multicast case, the following set of operations would take place:

- 1) Terminal does a BRQ with gatekeeper 1 asking for bandwidth.
- 2) Gatekeeper 1 agrees and sends BCF to terminal 1.
- 3) Terminal 1 sends the maxTxRate message to the MC, filling in the bandwidth it is using in all of its multicast feeds.
- 4) The MC forwards the maxTxRate message to all of the terminals that are multicast capable. It should update the sequence number prior to doing this.
- 5) Each terminal that receives the message checks with its gatekeeper that it is allowed to receive such bandwidth using the BRQ message.
- 6) Each terminal responds to the H.245 message using the appropriate acknowledgement or rejection.
- 7) When all terminals have responded to the MC, the MC informs the originating terminal of the combined result.
- 8) If the increase in bandwidth is rejected, then the originating terminal should send a MaxTxBitRate message with the values set for the previously acknowledged bit rate allowing any unused allocated bandwidth to be recovered.

A combination of unicast and multicast allocation can take place in the same message. However, in this instance, the MC must remove the unicast part of the message before forwarding it to the other terminals in the conference.

When a call is initiated, the following sequence could occur. A terminal reserves 64kbit/s prior to doing Q.931 setup, does Q.931 setup and then H.245 negotiation. When this is done, it should know the maximum likely bandwidth for the entire call. It can then do the sequence described above. For the unicast case, as long as the bandwidth stays within the limits set, the terminal can send any mix of audio, video and data using signalling included in the RTP header (for video and audio that is) without further messaging. Unfortunately, in a multicast conference, as different streams from a terminal may go to different destinations it is likely that multiple reservations will be required over time.

Conclusion

A simple and efficient mechanism for changing terminal bandwidth usage within an H.323 conference has been proposed that has a number of advantages over the methods currently suggested in H.323.