

Streaming Live Audio and Video

Streaming live audio/video is similar to the broadcasting of audio and video by radio and TV stations. Instead of broadcasting to the air, the stations broadcast through the Internet. There are several similarities between streaming stored audio/video and streaming live audio/video. They are both sensitive to delay; neither can accept retransmission. However, there is a difference. In the first application, the communication is unicast and on-demand. In the second, the communication is multicast and live.

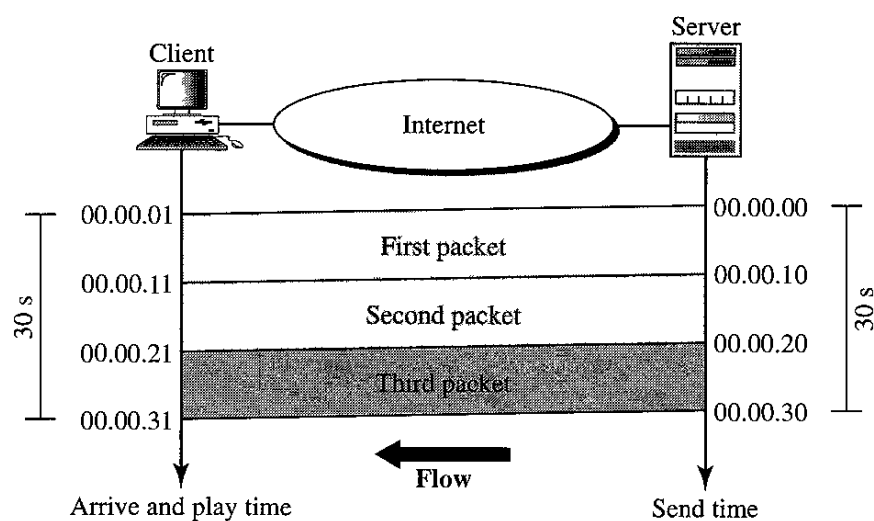
Real time interactive audio and video

In real-time interactive audio/video, people communicate with one another in real time. The Internet phone or voice over IP is an example of this type of application. Video conferencing is another example that allows people to communicate visually and orally.

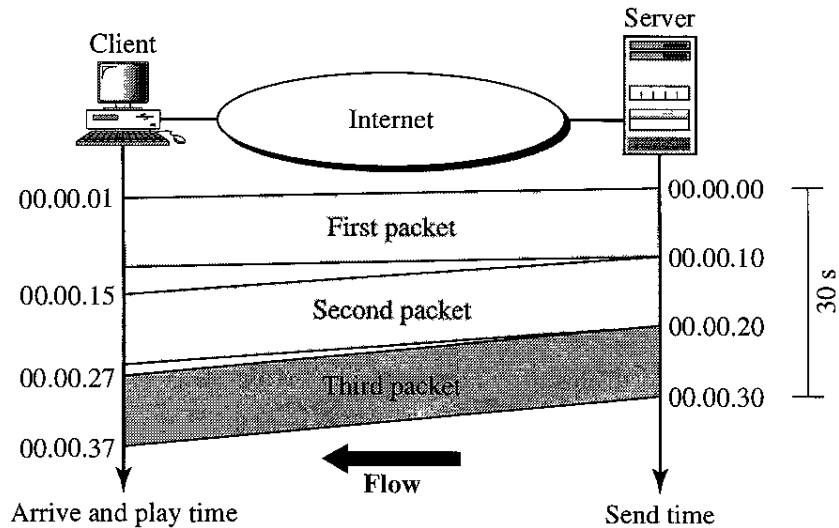
Characteristics

Time Relationship

Real-time data on a packet-switched network require the preservation of the time relationship between packets of a session. For example, let us assume that a real-time video server creates live video images and sends them online. The video is digitized and packetized. There are only three packets, and each packet holds 10s of video information. The first packet starts at 00:00:00, the second packet starts at 00:00:10, and the third packet starts at 00:00:20. Also imagine that it takes 1 s (an exaggeration for simplicity) for each packet to reach the destination (equal delay). The receiver can play back the first packet at 00:00:01, the second packet at 00:00:11, and the third packet at 00:00:21. Although there is a 1s time difference between what the server sends and what the client sees on the computer screen, the action is happening in real time. The time relationship between the packets is preserved. The 1s delay is not important. Figure below shows the idea.



But what happens if the packets arrive with different delays? For example, say the first packet arrives at 00:00:01 (1-s delay), the second arrives at 00:00:15 (5-s delay), and the third arrives at 00:00:27 (7-s delay). If the receiver starts playing the first packet at 00:00:01, it will finish at 00:00:11. However, the next packet has not yet arrived; it arrives 4 s later. There is a gap between the first and second packets and between the second and the third as the video is viewed at the remote site. This phenomenon is called jitter. figure below shows the idea.

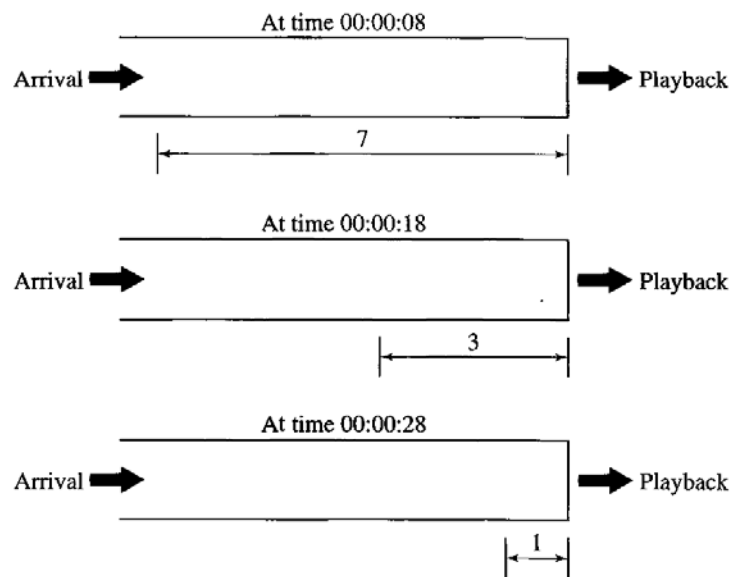


Timestamp

One solution to jitter is the use of a timestamp. If each packet has a timestamp that shows the time it was produced relative to the first (or previous) packet, then the receiver can add this time to the time at which it starts the playback. In other words, the receiver knows when each packet is to be played. Imagine the first packet in the previous example has a timestamp of 0, the second has a timestamp of 10, and the third has a timestamp of 20. If the receiver starts playing back the first packet at 00:00:08, the second will be played at 00:00:18 and the third at 00:00:28. There are no gaps between the packets.

Playback Buffer

To be able to separate the arrival time from the playback time, we need a buffer to store the data until they are played back. The buffer is referred to as a playback buffer. When a session begins (the first bit of the first packet arrives), the receiver delays playing the data until a threshold is reached. In the previous example, the first bit of the first packet arrives at 00:00:01; the threshold is 7 s, and the playback time is 00:00:08. The threshold is measured in time units of data. The replay does not start until the time units of data are equal to the threshold value.



Ordering

In addition to time relationship information and timestamps for real-time traffic, one more feature is needed. We need a *sequence number* for each packet. The timestamp alone cannot inform the receiver if a packet is lost. For example, suppose the timestamps are 0, 10, and 20. If the second packet is lost, the receiver receives just two packets with timestamps 0 and 20. The receiver assumes that the packet with timestamp 20 is the second packet, produced 20 s after the first. The receiver has no way of knowing that the second packet has actually been lost. A sequence number to order the packets is needed to handle this situation.

Multicasting

Multimedia plays a primary role in audio and video conferencing. The traffic can be heavy, and the data are distributed by using multicasting methods. Conferencing requires two-way communication between receivers and senders.

Translation

Sometimes real-time traffic needs translation. A translator is a computer that can change the format of a high-bandwidth video signal to a lower-quality narrow-bandwidth signal. This is needed, for example, for a source creating a high-quality video signal at 5 Mbps and sending to a recipient having a bandwidth of less than 1 Mbps. To receive the signal, a translator is needed to decode the signal and encode it again at a lower quality that needs less bandwidth.

Mixing

If there is more than one source that can send data at the same time (as in a video or audio conference), the traffic is made of multiple streams. To converge the traffic to one stream, data from different sources can be mixed. A mixer mathematically adds signals coming from different sources to create one single signal.

Support from Transport Layer Protocol

The procedures mentioned in the previous sections can be implemented in the application layer. However, they are so common in real-time applications that implementation in the transport layer protocol is preferable. Let's see which of the existing transport layers is suitable for this type of traffic. TCP is not suitable for interactive traffic. It has no provision for time-stamping, and it does not support multicasting. However, it does provide ordering (sequence numbers). One feature of TCP that makes it particularly unsuitable for interactive traffic is its error control mechanism. In interactive traffic, we cannot allow the retransmission of a lost or corrupted packet. If a packet is lost or corrupted in interactive traffic, it must be ignored.

Retransmission upsets the whole idea of time-stamping and playback. Today there is so much redundancy in audio and video signals (even with compression) that we can simply ignore a lost packet. The listener or viewer at the remote site may not even notice it.

UDP is more suitable for interactive multimedia traffic. UDP supports multicasting and has no retransmission strategy. However, UDP has no provision for time-stamping, sequencing, or mixing. A new transport protocol, Real-time Transport Protocol (RTP), provides these missing features.