LiveSwitch

White Paper

SFU: Selective Forwarding

by Anton Venema

Chief Technology Officer

Anton Venema
Chief Technology Officer

Anton is a leading expert on Real Time Communications solutions, and the visionary lead architect behind IceLink, WebSync and LiveSwitch.

Selective Forwarding

Scaling Beyond Simple Peer Connections

Introduction

In LiveSwitch, all connections are directed and controlled by the client. It doesn't matter if a peer, forwarded (SFU), or mixed (MCU) connection is desired, it is always initiated by the client and managed by the client.

Peer connections are simple. An offer describing the desired connection is created by a client and signalled through the LiveSwitch gateway to a remote peer. The remote peer then has the opportunity to either reject the offer or accept it by creating an answer and signalling it back.

For some use cases, this is sufficient (think Facetime).

While a gateway/signalling server is still required, peer connections minimize the use of server resources by eliminating the media server, and in doing so, minimize cost as well.

For other use cases, however, much more is needed. As the number of participants in a call grows, so does the need for a more scalable solution that optimizes the bandwidth available for each participant. Applications that call for legal accountability and integrity of recordings are not able to rely on devices to self-report by uploading recorded media files after a peer connection closes. Telephone interoperability, with both VoIP and POTS/PSTN, introduces special requirements that can't readily be satisfied without intermediary services.

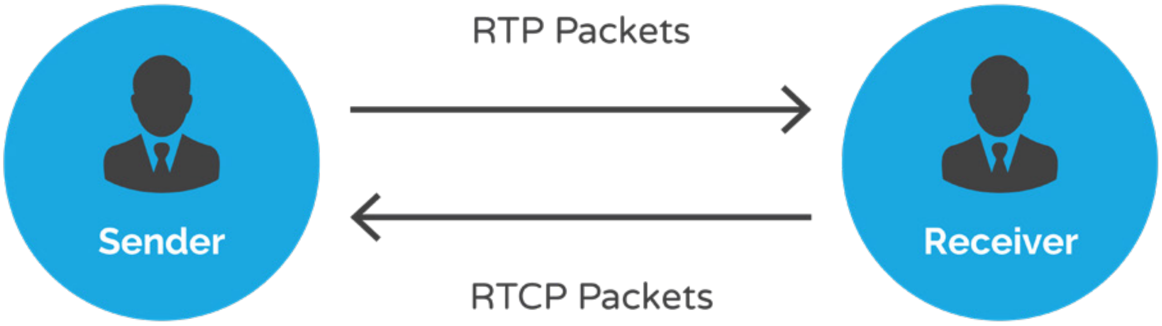
Addressing these use cases is where SFU and MCU connections excel. While technically quite different, they are both designed to allow scaling beyond simple peer connections.

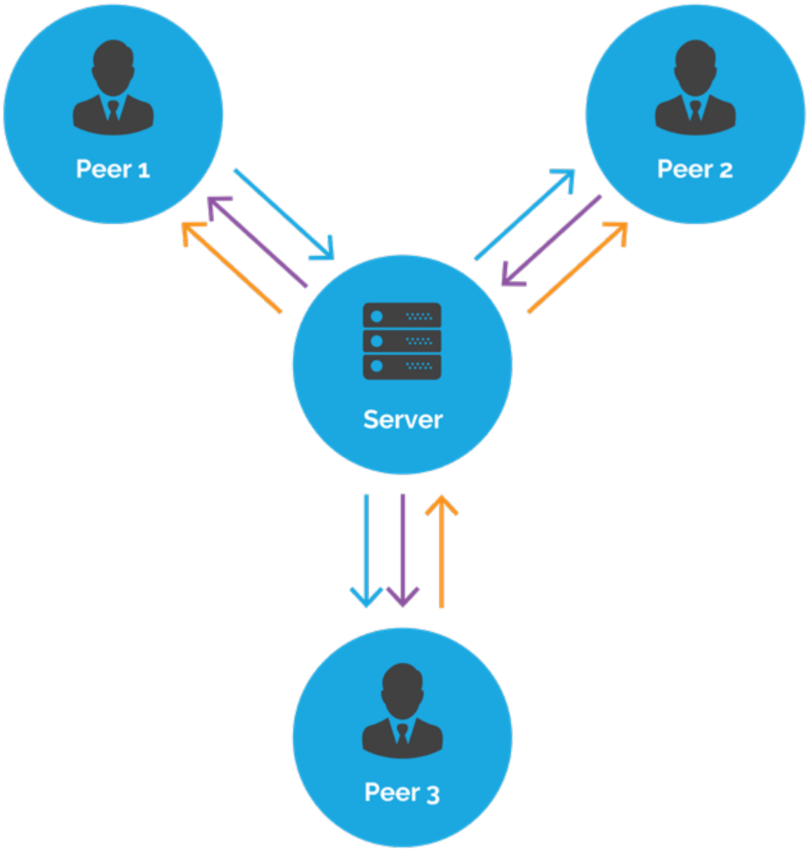
Basic Concepts

Every media stream has two key components:

1. Media packets (RTP packets - the actual audio/video data)
2. Media control packets (RTCP packets - commands, feedback, metadata).

In a live stream, media packets flow from sender to receiver, while media control packet flow from receiver to sender, providing crucial feedback as to the quality of the stream and changing network conditions.





An SFU works by forwarding media packets between a sender and one or more receivers, while analyzing media control packets coming back from the receiver(s) and using that information to intelligently manage the flow of media packets.

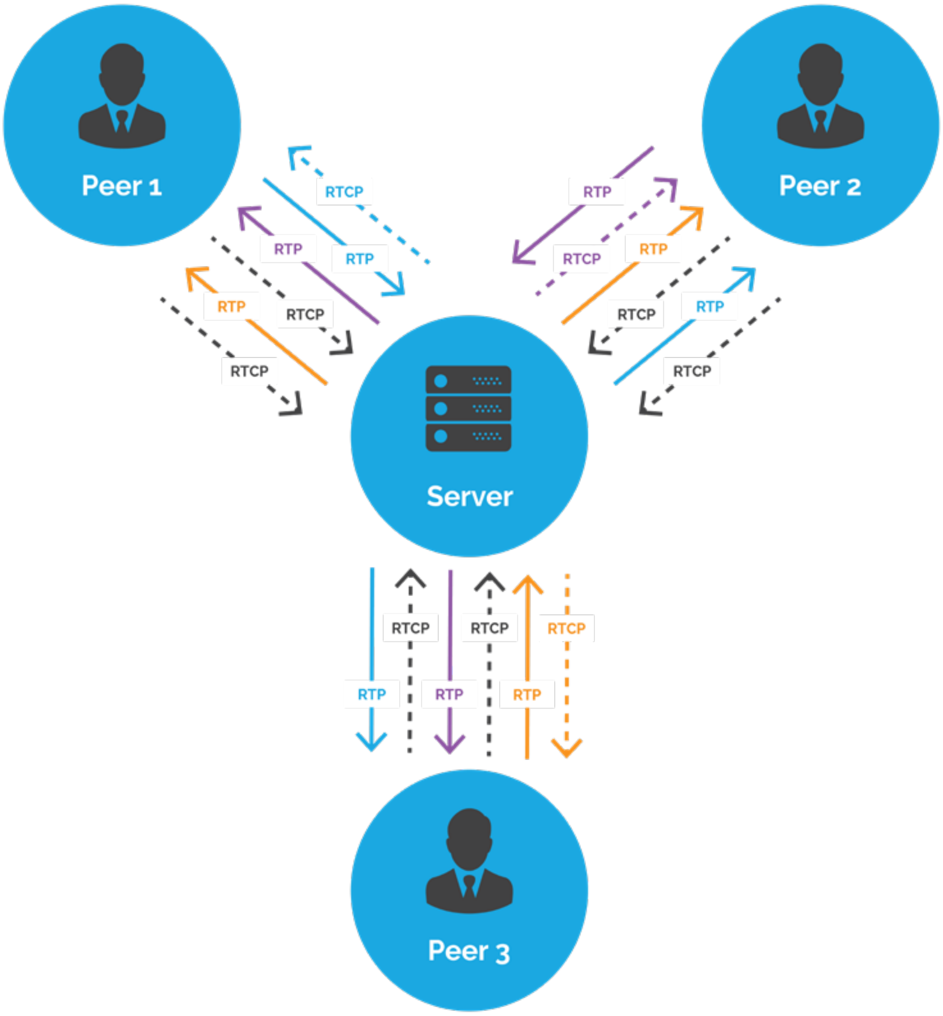
Consider a simple three-person call.

Assuming each client opens forwarded connections (LiveSwitch allows all connection types to coexist in a session), each client should end up with one SFU upstream connection and two SFU downstream connections. Media packets flow from the client to the server on each of the three upstream connections, and distributed out over the six related downstream connections, with each upstream connection feeding into exactly two related downstream connections.

Media control packets, on the other hand, flow in the opposite direction. They arrive at the server from each of the six downstream connections. Unlike the media packets, they do not get distributed out to the related upstream connections. Instead, LiveSwitch's SFU analyzes the information and uses it to manipulate the upstream and downstream connections. Ultimately, it's goal is to make the end user experience as good as possible working within the environmental conditions, which include client resources, server resources, and network quality/bandwidth on each leg.

Simple Forwarding

In a simple SFU, media packets are forwarded as-is with no manipulation, and media control packets are used only as needed to sustain the video stream, which generally can't recover from missing media packets as easily as the audio stream.



At minimum, the SFU needs to forward packet retransmission requests (e.g. generic “NACK”, negative acknowledgement, packets) from the receivers as well as keyframe requests for reconstructing decoder state when missing packets can’t be recovered in a timely fashion.

A more intelligent SFU would only forward packet retransmission requests for packets lost on the upstream connection and address lost packets on the downstream connection directly without adding any extra overhead on the upstream side. It would also maintain sufficient state for each connection so as to avoid requesting retransmission more frequently than is necessary on the upstream side and to avoid distributing retransmitted media packets to downstream connections that don’t require it.

Video Scalability

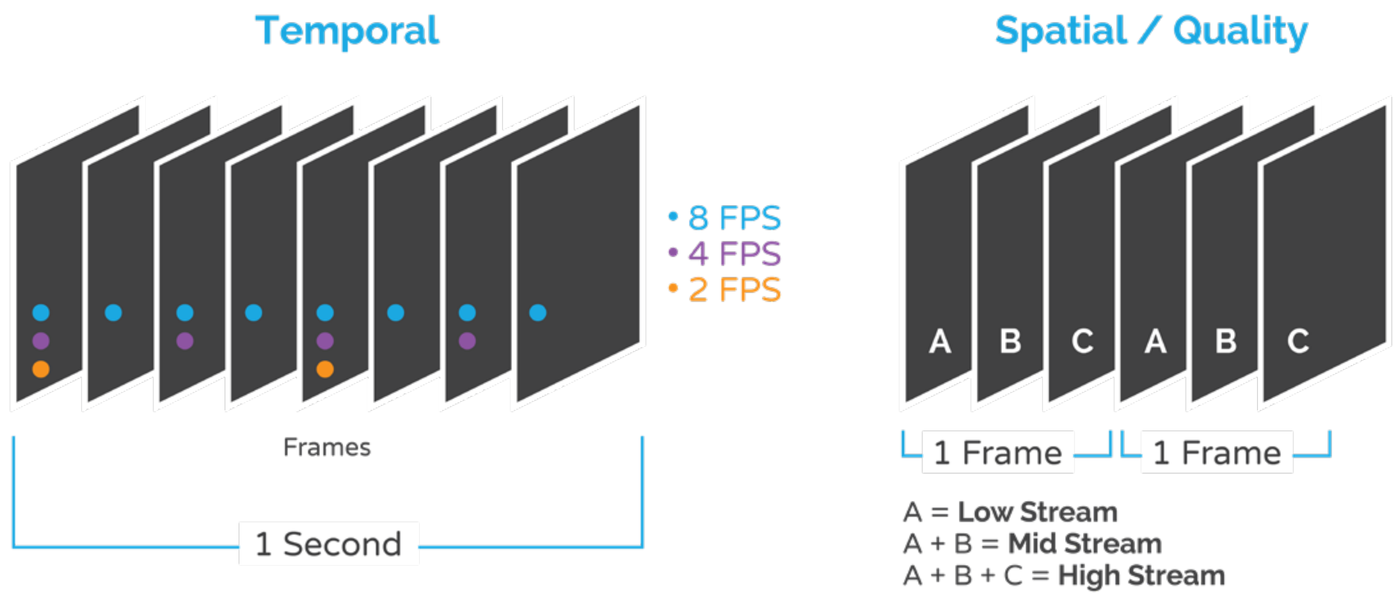
Basic forwarding works really well for audio, which generally functions well even in low-bandwidth conditions and doesn’t rely on negative acknowledgement or expensive keyframes. Video, however, is another story entirely. Even with intelligent handling of retransmission and keyframe requests from receivers, video is especially demanding and can fail spectacularly in congested networks.

In a peer connection, the resolution for this is (relatively) simple. If congestion is detected and exceeds an allowable threshold on the receiving side, the sender reduces the demands its placing on the network by taking steps to reduce bandwidth consumption. There are several ways to do this (reducing resolution, dropping frames, increasing encoder quantization) but they all have the same effect - decreasing the amount of information that hits the wire in exchange for a lower-quality video stream.

This is much more difficult for an SFU. Without direct access to the video source and encoder, a simple SFU can’t directly reduce the bandwidth on a congested downstream connection without making a change on the upstream side (by instructing it to reduce resolution, drop frames, increase encoder quantization). Doing this does solve the immediate problem for the congested connection, but it also degrades the video quality in the server-side recording and all the other downstream connections, which could be more than capable of handling the higher bandwidth stream.

A more intelligent SFU makes use of video scalability, a feature of some video codecs which allows a lower-quality “sub-stream” to be extracted from a sequence of media packets without having to actually decode the video stream. There are three types of scalability that a video codec can support:

1. Spatial scalability, which reduces the resolution of the forwarded stream.
2. Temporal scalability, which drops frames from the forwarded stream.
3. Quality scalability, which increases the quantization of the forwarded stream.



If the SFU understands the video scalability features of the media packets it is forwarding, then it has a very powerful tool in its belt. By selectively discarding specific packets from the forwarded stream, it can reduce the bandwidth requirements of a particular downstream connection without affecting any others.

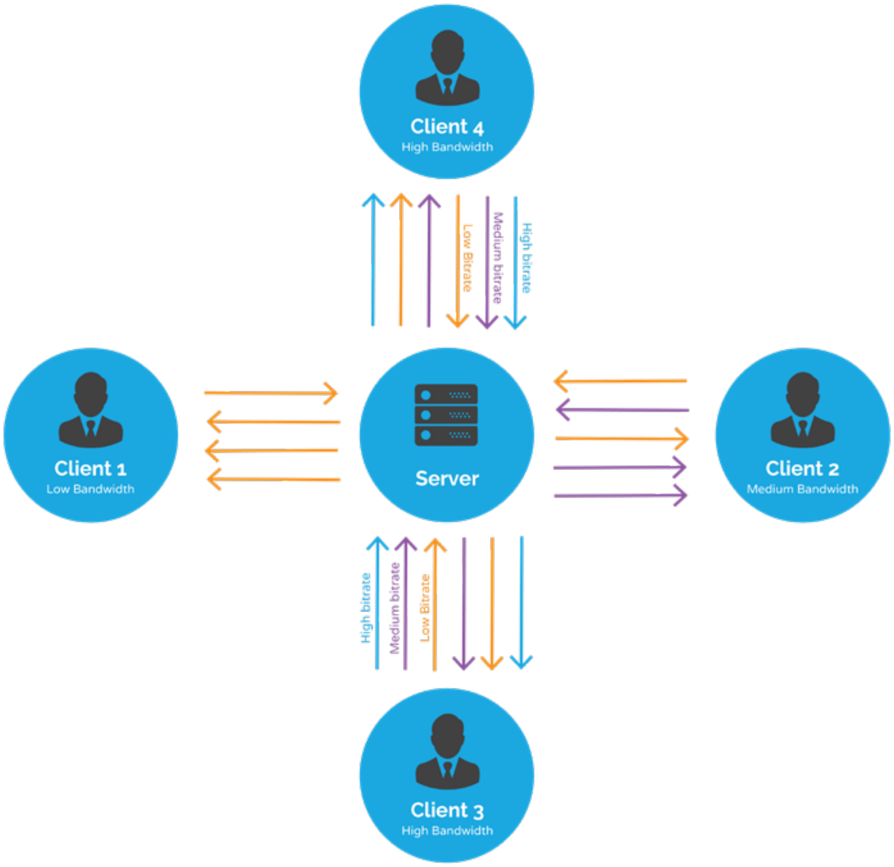
Both VP8 and H.264/AVC, the two video codecs mandated by WebRTC, support temporal scalability, but not spatial/quality. H.264/SVC supports all three types, but is patent-encumbered, not included in the WebRTC specification, and not currently a part of the “free” OpenH264 binary distributed by Cisco. VP9 supports all three types, but is not included in the WebRTC specification, and being as new as it is, limited in its adoption by all the major browser vendors.

Video Simulcast

While video scalability is a robust solution, it depends on support being built into the codec. Video simulcast is a much simpler option that works for every video codec, regardless of its featureset. Simply open multiple upstream connections to the media server at different bitrates. Downstream connections can then choose between the quality levels and even switch streams as network conditions change.

Many content delivery networks work this way for pre-recorded content. Both HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming over HTTP (DASH, or MPEG-DASH) use this technique, with the media recorded at various quality levels and split up into small chunks. Clients can start downloading chunks from a lower-quality stream if network conditions degrade, or switch over to higher-quality chunks if things improve.

The obvious downside to this technique is the added burden placed on the media producer. In the case of pre-recorded content, this is easy to justify, as it all takes place in advance or on the server where hardware, bandwidth, and time is readily available to handle the task.



In a real-time scenario things aren’t quite so easy. The sending side is often difficult to predict - the resources simply might not be there to encode, encrypt, and upload multiple streams. That said, it is a simple and effective solution that works well, provided the sending side can handle it.

Forwarding Limitations

There are two primary limitations of an SFU:

1. Clients bear the burden of receiving multiple streams.
2. Clients have to negotiate the same codec independently of each other to talk.

The first limitation becomes more obvious as the number of senders increases. A single-stream broadcast to thousands of clients can work really well, but a 20-person conference where everyone is sending and receiving is essentially impossible.

The second limitation is only really a limitation if you don't know the environment in which your application will be deployed. Proper planning in advance can mitigate the impact of this significantly, as can choosing well-known and broadly adopted codecs like PCMU, PCMA, Opus, VP8, and H.264. If the majority of the expected clients support a particular codec set, going with that and allowing the remaining minority to connect in as MCU clients is often a good strategy.

Wrap-Up

Selective forwarding is an excellent way to scale video communications in your application. Cost-effective, bandwidth-efficient, scalable, low-latency, and recording-ready, it is a very balanced approach for applications that need scaling beyond simple peer connections but want to avoid the cost and performance implications of mixing with an MCU.

If you are interested in learning more about how LiveSwitch can help you scale your video communications, please don't hesitate to contact us. Our product suite is custom-designed from the ground up to be flexible enough to work in every scenario for every customer, regardless of how unique or constrained your needs are, and yet powerful enough to serve massive customer bases. We look forward to hearing from you!