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.

frozen mountain

LiveSwitch

Scalable Audio/Video Conferencing and Broadcasting

Introduction

LiveSwitch is a first-of-its-kind real-time communications (RTC) solution that dynamically scales real-time audio/video conferencing and broadcasting to include web, desktop, mobile, and telephony devices.

It employs a hybrid model that allows peer-to-peer (P2P), selective forwarding (SFU), and multipoint control (MCU) connections and text chat to co-exist in the same session. This makes it easy for application developers to build exactly what they need given the resources available and environmental restrictions of their client and server platforms.

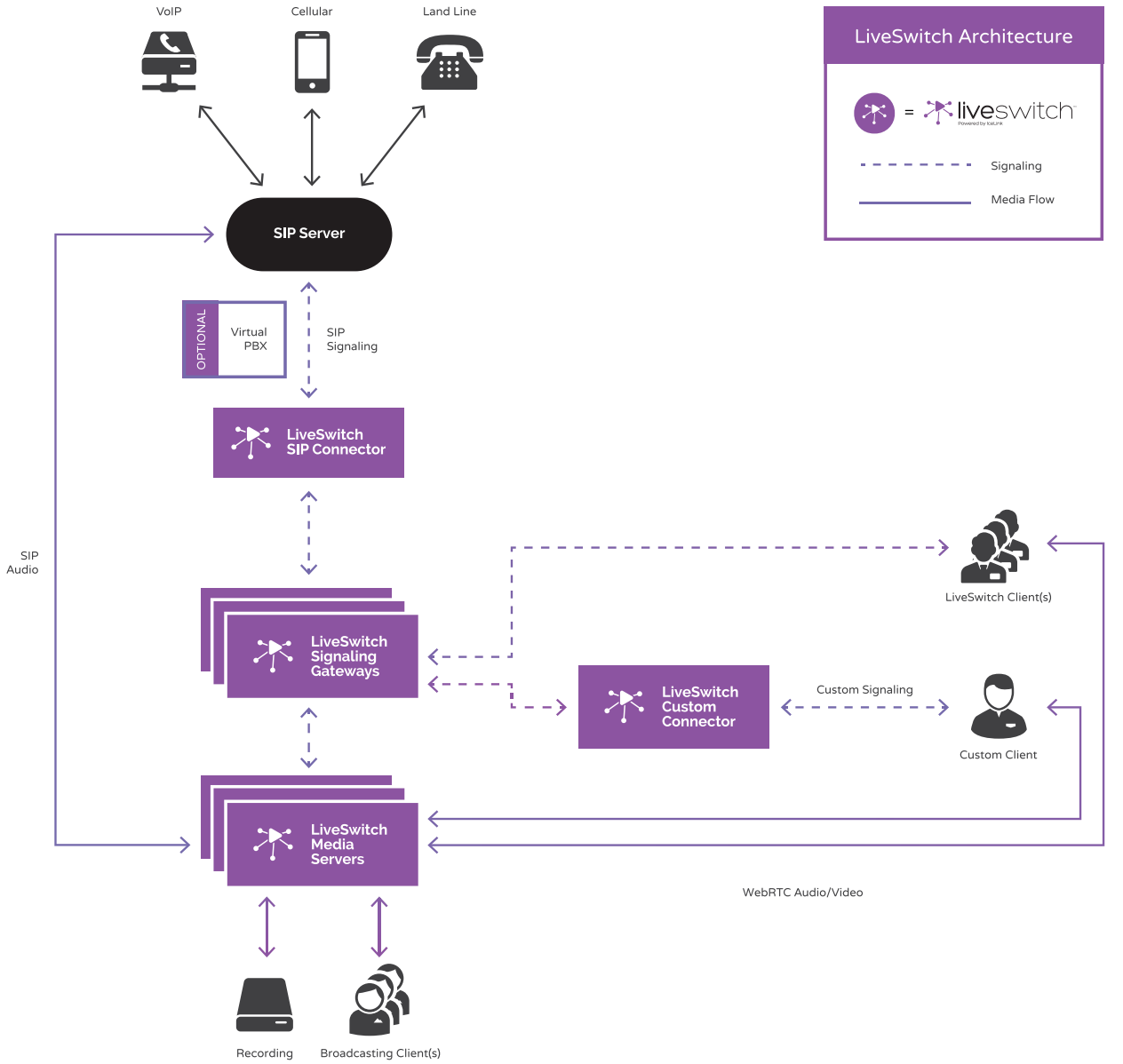
Full interoperability with SIP allows VoIP/PSTN subscribers to join conferences and live broadcasts by simply dialing a telephone number. Live recording of audio and video streams to the open standard Matroska free container format ensures that every bit of media can be archived for post-processing or long-term storage. Media server branching supports virtually unlimited scalability for broadcasting solutions that need to target high subscriber volumes with sub-second latency.

In short, LiveSwitch was built to address all your RTC needs, whatever they might be.

LiveSwitch Architecture

A typical LiveSwitch architecture will include:

- One or more LiveSwitch gateways behind an HTTP/S load balancer.
- One or more LiveSwitch media servers, registering with the gateway cluster.
- One or more LiveSwitch clients, registering with the gateway cluster.
- Optionally, a LiveSwitch SIP connector, registering with the gateway cluster and a SIP registrar.
- Optionally, custom LiveSwitch connectors that link in other media streaming devices.



The LiveSwitch Gateway

The LiveSwitch gateway is the central signaling server to which all clients, media servers, and connectors register.

It is responsible for presence notifications and authorization of all requests using JSON Web Tokens (JWT). All communication with the gateway runs over HTTP, and can be secured using standard SSL/TLS (HTTPS). Any text-based communications, such as for LiveSwitch's text chat capabilities or system messages, route through the gateway.

The LiveSwitch gateway is responsible for mapping active sessions to media servers, ensuring that client media streams are directed to the correct place.

Any load balancing algorithm can be used, as the gateways themselves are stateless. LiveSwitch currently uses Redis for data storage and multi-server synchronization.

Multiple gateways can be deployed for high availability and load distribution.

The LiveSwitch Media Server

The LiveSwitch media server provides all the support required for forwarded (SFU) and mixed (MCU) streaming connections, including full recording capabilities. Peer connections run directly between clients.

The media server is responsible for sending notifications out to the clients when new upstreams become available or when the state of an existing connection changes, such as when it closes, when the video layout changes (mixed connections), or when quality is adjusted to meet changing network conditions.

Media servers can be deployed as needed to support demand, and shut down when not in use.

The LiveSwitch SIP Connector

The LiveSwitch SIP connector acts as a bridge between LiveSwitch and SIP signaling. It translates SIP messages into the format required by LiveSwitch, and vice versa.

It also maintains the map of inbound telephone numbers to channels so that LiveSwitch knows where to route inbound calls.

Each SIP connector can register with a single SIP registrar, such as a SIP trunk or a PBX platform like FreeSWITCH.

The LiveSwitch Clients

The LiveSwitch clients drive the application.

In conjunction with your application server, which generates the required authorization tokens, the clients determine which channels to join and what connections to open within the context of a channel, if any.

Large numbers of active participants, like a virtual classroom, might demand an MCU connection to shift the heavy lifting to the server, whereas a large broadcast would be better served with a forwarded connection again.

For example, an application may want to create a peer connection when there are only two participants, but opt for an SFU connection when there are three or more.

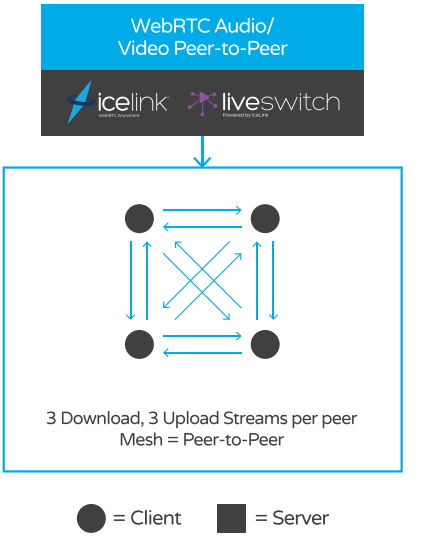
LiveSwitch Connections

There are three types of streaming media connections that can be established with LiveSwitch - peer, SFU, and MCU. In addition to this, simple text-based messages can be published through the gateway, supporting features like text chat or the transmission of diagnostic data.

LiveSwitch Gateway Connections

Gateway connections are real-time, non-streaming, signaling connections that every client needs. It is the first connection created in a LiveSwitch client application, as it registers with the cluster and sets up an initial set of channels for text chat and presence notifications.

Once a gateway connection has been established, text messages can be signaled back and forth between participants in a given channel, and streaming media connections can be created as desired.



LiveSwitch Peer Connections

Peer connections are, as you might guess, connections between peers. Peer connections are unique in two key ways:

1. **They do not talk to the media server.**
2. **They require the answering side to accept or reject the offer from the calling side.**

Because peer connections do not talk to the media server, they are a little more difficult to record. Recording can still be done client-side, but there are negative performance implications, and an out-of-band upload to a file server at a later point is required for secure storage.

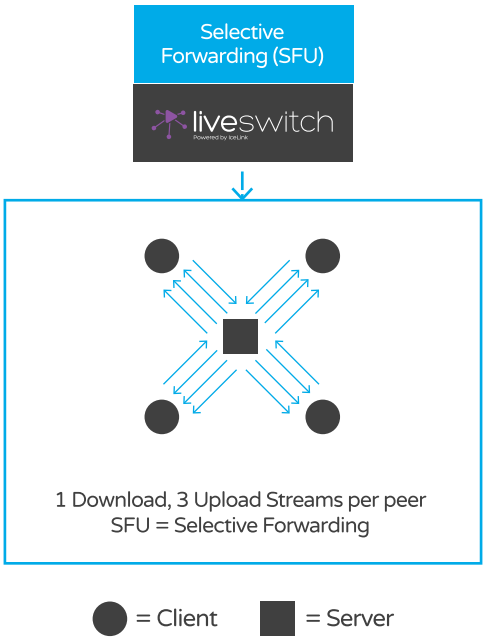
Since peer connections require the remote side to accept or reject the connection offer, there are a variety of ways that connection flows can be designed. Contact, or call-based systems, will typically prompt the user to accept or reject the offer. Conference, or room-based systems, however, will likely want to accept the offer right away without an interface prompt, with acceptance being implied by joining the active session.

LiveSwitch supports both these and any other connection flow that can be dreamed up.

LiveSwitch peer connections are an excellent choice for two-party (and perhaps three-party) calls, or for private communications within the context of a larger session, where recording is not critical.

LiveSwitch SFU Connections

SFU connections are unidirectional and take the form of either an upstream or downstream connection to the media server. An active upstream connection will deliver media from a client to a media server, while an active downstream connection will deliver media from a media server to a client.



Compared to peer connections, upstream bandwidth usage is reduced significantly when more than one peer is present.

By uploading to the server once instead of uploading to each peer individually, we have a fixed upstream cost per participant, which allows scalability that is unattainable with simple peer connections.

LiveSwitch SFU connections are an excellent choice for sessions of any size where the number of participants actively sending media to the server is relatively low.

LiveSwitch SFU connections are also quite lightweight on the server. The media stream does not need to be decoded, so media packets can simply be forwarded, reducing both latency and demand on server resources. When an SFU-based session is possible, it is often preferable because of the reduction in server costs.

This could be a small group session where everyone is actively sending and receiving or a large broadcast where a handful of participants are actively presenting with everyone else passively watching.

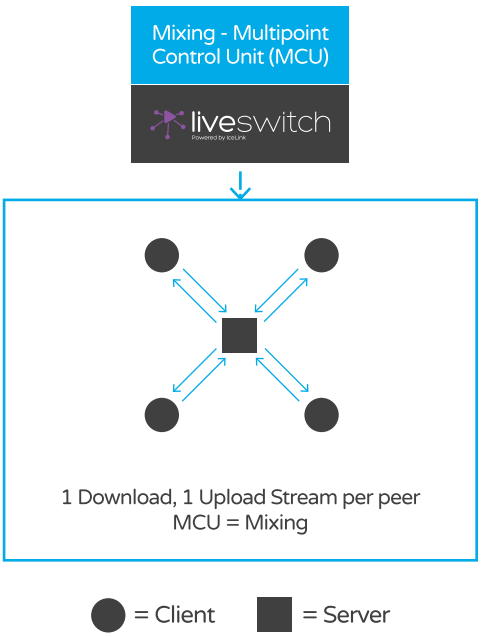
LiveSwitch MCU Connections

MCU connections are a bit like peer connections where one of the peers is the media server.

The media server represents all the other participants in the session, and as such mixes the audio and video from the other participants together as needed to deliver the media over a single connection.

Like SFU connections, upstream bandwidth usage is reduced significantly versus peer connections when more than one peer is present. By uploading to the server once instead of uploading to each peer individually, we have a fixed upstream cost per participant, which allows scalability that is unattainable with simple peer connections.

Compared to peer and SFU connections, LiveSwitch MCU connections reduce downstream bandwidth usage significantly.



By downloading from the server once instead of once per sending peer, we have a fixed downstream cost per participant, which allows virtually any device to participate in a session with even a high number of participants actively sending media.

LiveSwitch MCU connections have the greatest client-side scalability, but it comes at a cost on the server side.

In order to mix the audio and video together, each stream has to be "terminated" at the server, which means re-assembling fragmented frames and decoding them. Proper packet re-assembly requires a jitter buffer, which introduces latency, whereas the decoding process increases the demand on server resources. In addition to this, each participant requires its own individual audio mix to avoid echo, the uncompressed video frames have to be copied (to the video mix canvas) and the final image has to be encoded, all of which increase server load and impact scalability. LiveSwitch does all of this for you, but the hardware costs must be carefully considered.

Currently, all connections made using the LiveSwitch SIP connector are MCU-based.

Hybrid Sessions

LiveSwitch gives you ultimate flexibility in how you set up a live media session, whether your application delivers calling, conferencing, or other real-time meeting features. Any of the aforementioned streaming connections (peer, SFU, or MCU) can be created at any time.

For example, one participant may want to run all media streaming over SFU connections, and so join the session by opening a single upstream connection sending local media up to the server. Another participant on a low-powered device may want to run a single MCU connection.

LiveSwitch will automatically bridge the media from the SFU participant into the mixed video feed delivered to the MCU participant, and will automatically notify the SFU participant that a new downstream connection is available from the MCU connection that was opened. In this way, each application can determine, at runtime, what mode of operation makes the most sense.

Since the opening and closing of connections is driven entirely by client-side logic, it is possible for any given participant to make use of more than one type of connection.

Some sessions may desire to start with peer connections and upgrade to SFU or MCU connections once a certain number of participants has joined the session. Since the connection management is entirely client-side, it is possible to do this at runtime with minimal interruptions to the live feed.

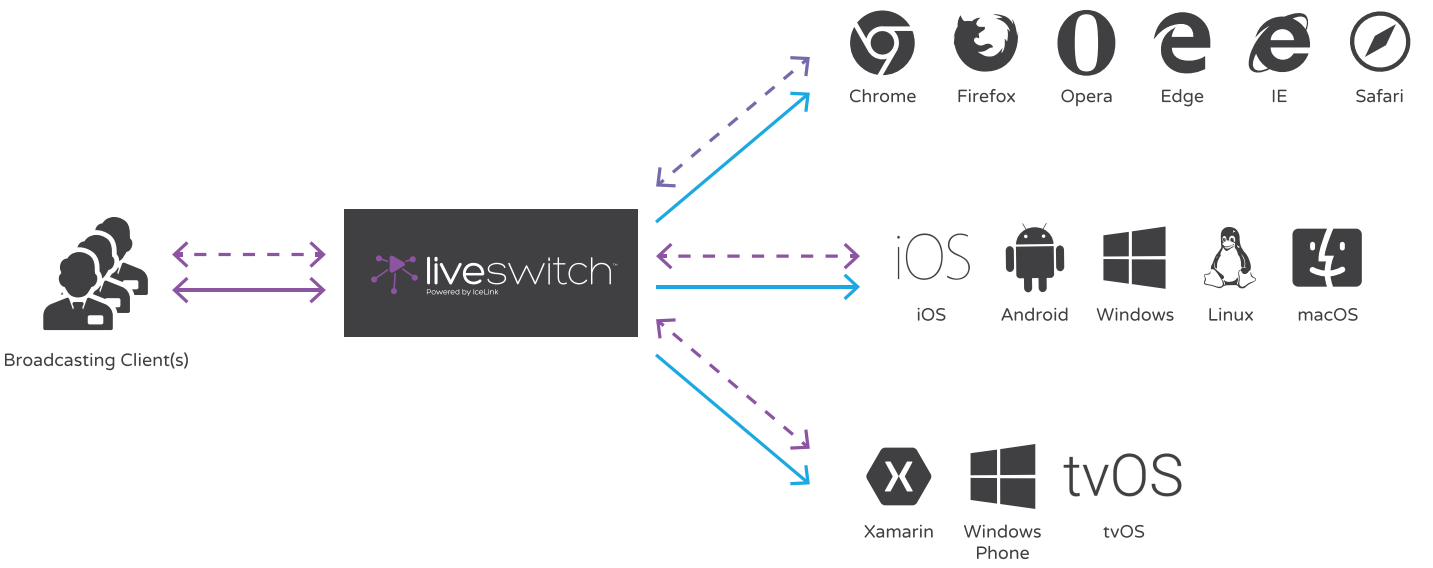
For example, a virtual classroom may have everyone participate through MCU connections for client-side scalability, but run screen-sharing as an SFU connection, or use peer connections for breakout sessions while the primary MCU connection remains active.

frozen mountain

Broadcasting

LiveSwitch's architecture lends itself extremely well to broadcasting scenarios. A single media server can deliver a live video stream to hundreds of viewers.

With media server branching coming to LiveSwitch soon, this number can easily be driven into the thousands and millions.



Broadcasting

2 - Way Conferencing

1 - Way Conferencing

Text Chat

A typical setup for a single-publisher scenario would be to have the broadcaster open an upstream SFU connection to send media to the server, with all viewers opening a single downstream SFU connection to receive it. Optionally, all participants can take part in a live text chat via the gateway connection as well.

Interactive broadcast is possible, too. Application logic can "promote" a viewer by allowing them to open an upstream connection to the media server alongside the original broadcaster. In this scenario, it is generally more desirable to have the upstream connections be MCU-based, especially with large viewership, to avoid requiring the downstream SFU-based viewers to open new connections each time this takes place.

Wrap-Up

Hopefully you have found this informative in understanding what LiveSwitch is and how it can be used to deploy a scalable audio/video conferencing or broadcasting solution.

We can't wait to see what you build!

Client SDKs are available for every major platform, including web, iOS, Android, .NET, Java, macOS, Universal Windows Platform, and Xamarin.

Download Your Free Trial of LiveSwitch Today.

Click Here

For more in-depth information on how LiveSwitch works, read the full documentation.

Click Here

info@frozenmountain.com

1-888-379-6686