frozen mountain

Anton Venema
Chief Technology Officer

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

iRTC

Internet-Based Real Time Communications

Introduction

What's the first thought that comes into your mind when you think about real-time communications? Is it a phone call you had a few minutes ago? A text message sent to your mobile? Maybe something more modern, like a tweet or video broadcast? All of these fall under the umbrella of what we like to call Internet-based real time communications, or iRTC for short.

Real-time communications have been a part of our lives for a long time. From public telephone networks to radios all the way back to the telegraph, humanity has a history of seeking out new and better ways to use technology to improve communication. In the past decade, technology has arguably advanced more than the past century before it. Smartphones have stormed the market, mobile processors are advancing in line with Moore's law, LTE rollouts are delivering unprecedented Internet speeds across the world, and WiFi hotspots are becoming ubiquitous.

iRTC Includes Many Applications

The availability of high-speed Internet services just about everywhere is causing a fundamental shift in the way people want to communicate and consume media. Cable networks are finding it more difficult to distinguish between their Internet and TV services, especially when companies like Netflix and HBO are able to publish their content directly to consumers over the Internet. Even live broadcasts, TV's last stronghold, are being slowly replaced as platforms like YouTube allow content to be broadcast live to millions of users simultaneously over the Internet.

A similar phenomenon is happening in the telecom industry to mobile carriers. Just a few years after SMS/MMS text messaging was popularized, free messaging applications like iMessage and WhatsApp have rapidly outpaced its growth. Free calling services like Skype and Facetime that deliver video calling over the Internet are working hard to compete against traditional voice calls. People are demanding more, as well. A more technical customer base is creating a need for file sharing services, screencasting, live streaming, and more. Periscope and Twitch are great examples of services built using iRTC that couldn't have existed just a few years ago.

Running communications and media content delivery over the Internet allows industries to provide a better user experience and react faster to shifting market demand. It breaks down borders, eliminates the concept of “long-distance” communication, and makes room for new ideas on how people can communicate not just with each other, but with their “things” as well.

iRTC Covers Many Platforms

The ubiquity of the Internet and WiFi has greatly expanded the connectedness of the things in our lives. First it was desktops, and then came web browsers. Smartphones and responsive web applications came next, followed by a frenzy of Internet connectivity in everyday devices - from refrigerators to garage door openers to thermostats. Wearable devices like watches and eyewear are making significant gains in popularity now, and it’s not likely to slow down anytime soon.

What binds all these together is the Internet, and while each platform has its own unique characteristics, iRTC provides a way to bring them all together into a comprehensive architecture that delivers the content people want on the device of their choice.

It’s useful to break down iRTC into 4 categories, based on whether the communication is one-way (unidirectional) or two-way (bidirectional), and whether the communication is streaming (continuous delivery) or non-streaming (discontinuous delivery).

	Non-Streaming	Streaming
Two-Way	e.g. Text Chat, Collaborative Editing	e.g. Voice Call, Video Call
One-Way	e.g. Live Blogging, Push Notifications	e.g. Video Broadcast, GPS Data

iRTC Is Non-Streaming

An architecture is considered non-streaming if the application sends or receives real time messages, but not as a continuous, steady stream. Messages in a non-streaming architecture are typically:

- sent and received as text, possibly with inline media.
- reliable and ordered in delivery.
- delivered/relayed through a messaging server to receiving participants.
- bandwidth-light.

One-way non-streaming applications send and receive messages, but not from the same endpoint. They are typically designed around a small number of senders, often just one, and a large number of receivers. An example of this is a live blog, which delivers new content in real-time, but only as frequently as the author adds it. Push notifications are another great example. They are delivered in (near) real-time, but only as needed, triggered by a schedule or action in a third-party system.

Two-way non-streaming applications send and receive messages, and can do so from the same endpoint, so clients can be senders, receivers, or both. They are typically designed around a large number of client clusters, each of which has a small number of participants. All text chat systems are examples of this architecture in action, such as Slack, Internet Relay Chat (IRC), iMessage. Messages are delivered in real-time, but only as participants in the group type and send them.

Many technology solutions for non-streaming architectures, including WebSync, make use of the publish-subscribe pattern. Clients connect into a highly scalable central server cluster and subscribe to one or more “channels”, which lets the server know which types of data they want to receive. The server can then publish messages to any of those channels, and the clients who have subscribed will receive a copy in real-time.

iRTC Is Streaming

An architecture is considered streaming if the application sends or receives real time data as a continuous, steady stream. Data in a streaming architecture is typically:

- sent and received as encoded/compressed binary.
- unreliable and with no guarantees on ordering (typically).
- delivered/relayed through a media server or directly between participants.
- bandwidth-heavy.

One-way streaming applications send and receive data, but not from the same endpoint. They are typically designed around a small number of senders, often just one, and a large number of receivers. The most common example of this is a live audio or video broadcast, but the streamed data can take any form; consider sensors streaming point data to a kiosk or GPS coordinates streamed to a map overlay for position

monitoring. In the healthcare industry, streaming diagnostic data from medical devices is a common use case.

For high-bandwidth use cases like video streaming, a geo-located content delivery network (CDN) can be used to distribute the load across multiple data centers and servers. Most CDN providers can handle static/cached content (YouTube, Netflix. etc.), but it's also possible to use the same technique in a live broadcast with a brief time delay.

Two-way streaming applications send and receive data, and can do so from the same endpoint, so clients can be senders, receivers, or both. They are typically designed around a large number of client clusters, each of which has a small number of participants. Applications that run voice and video calls/conferences are the most common examples of this architecture, like Facetime, GoToMeeting, and Voice-over-IP (VoIP) systems.

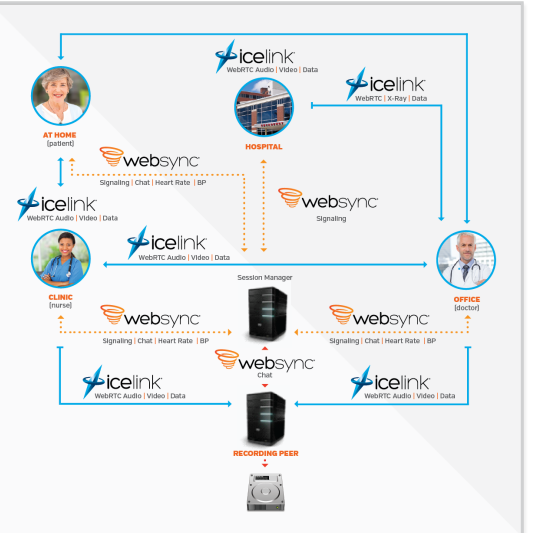
Technology solutions for streaming architectures, like IceLink, should be as efficient as possible to keep operational costs low, performance-focused to make the user experience great, and standards-based to maximize interoperability. If web browsers are a part of the topology, then WebRTC and ORTC compatibility is a must to avoid the need for plugins in Google Chrome, Mozilla Firefox, and Microsoft Edge.

Signaling

Every streaming application is supported by at least one non-streaming component to handle signaling. Signaling allows two endpoints (senders, receivers, or both) to communicate meta-information about the streaming connection between each other before actually setting it up. For example, before two endpoints can start a video call, one side has to call the other, and the called side has to respond. This call-and-response message flow (also known as offer-answer message flow) contains critical details about the streaming that will take place - the number and types of streams, how the media will be encoded, etc. - and is often formatted using the Session Description Protocol (SDP), a standard format used by many real-world systems, including VoIP and WebRTC.

iRTC for Healthcare

Telehealth applications that aim to connect patients with their doctor and/or a nurse are well-suited to a mesh streaming topology. A headless peer can be used to record and store the conversations and exchanged securely according to HIPAA requirements.



NAT Traversal and TURN

Once the initial signaling for a streaming connection has taken place, the two endpoints can begin the process of NAT (Network Address Translation) traversal. In most cases, unless the two endpoints are on the same local network, there will be one or more intermediary network devices (routers/gateways) between the two. These routers each have a NAT table which translates private/internal IP addresses and ports to public/external IP addresses and ports.

Ensuring that communication can be directed through those devices requires the use of a standardized algorithm known as ICE (Interactive Connectivity Establishment). ICE defines a process by which connections between endpoints can be guaranteed, but it requires that a TURN (Traversal Using Relays around NAT) server be set up somewhere on the public Internet. The TURN server assists in the NAT traversal by helping the endpoints learn about the routers on their local networks, as well as blindly relaying data for one of the endpoints in the worst case scenario where a direct connection is not possible due to firewall restrictions.

Security

Once a direct or relayed (through a TURN server) streaming connection is established, the next step is to secure the connection. The preferred method for this is to use perfect forward secrecy (PFS) ciphers in a DTLS (Datagram Transport Layer Security) handshake to securely exchange key data. For audio and video, that key data can then be used to generate AES (Advanced Encryption Standard) keys that are in turn used by SRTP (Secure Real-time Transport Protocol) to encrypt and decrypt the media.

This acronym-rich stack of technologies translates to extremely secure connections that are impossible to break with current technology. Both WebRTC and ORTC mandate this particular stack, which is backwards-compatible and interoperable with VoIP systems.

Topologies

Within the world of streaming architectures are several different topologies that dictate who connects to who, what data is transferred, and where. Which topology is best for a given application depends largely on the expected use cases, as each one has its own unique set of benefits and drawbacks. We're going to look at the most common topologies here, but this is by no means an exhaustive list.

Mesh (P2P)

In a mesh topology, each participant in a session connects directly to the other participant(s). Mesh topologies can be further subdivided into full or partial, based on whether every participant connects to every other participant (full) or whether some participants remain intentionally disconnected from each other (partial).

A mesh topology has the advantages of being the lowest cost and easiest to set up. Aside from the minority of cases where a TURN server is required to relay media, bandwidth consumption for the media streaming is entirely client-side. Estimates on how many connections require a TURN server are subjective and depend a lot on how many clients are on mobile or corporate networks, but the range usually falls somewhere between 10 and 15 percent. The only other server required besides a TURN server is a signaling server, which uses little bandwidth and so can be run with minimal cost.

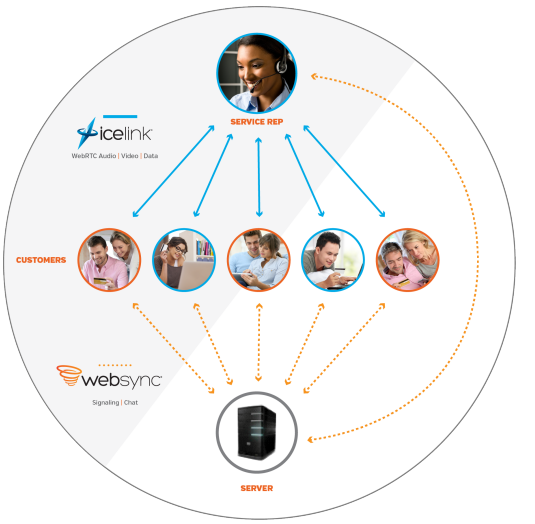
One of the major drawbacks of a mesh topology is that it doesn't scale well beyond a few participants. Getting more than 3 or 4 participants in a session can be challenging, especially on mobile devices. Upstream bandwidth, which is typically much lower than downstream bandwidth, can also become a significant constraint as each participant has to upload their media to each connected peer.

Since the connections are direct between peers, a mesh topology doesn't lend itself well to recording. Although media can be recorded locally on the client, it is challenging to make guarantees about the integrity of the recording, especially when industry regulations require a certain level of assurance in this regard. The same challenges arise when using a headless peer, where one participant in the conference, usually set up in advance on a server, records all media received from the other participants but doesn't send any media out. That said, these are both viable options when the requirements with respect to the integrity of the final recording are not overly strict.

For these reasons, a mesh topology is best for simple applications that connect 2 to 4 participants, where low latency is important, and where recording isn't required or the integrity isn't critical.

iRTC for Customer Service

Customer service applications that connect service representatives to customers are well-suited to a mesh streaming topology. A simple recording agent on the representative-side application can be used to record the representative's media in pristine quality and the customer's media as it comes in off the network.



Multipoint Control (Mixing)

In a multipoint control topology, each participant in a session connects to a server which acts as a multiple control unit (MCU). The MCU receives media from each participant and decodes it, mixing the audio and video from the participants together into a single stream which is in turn encoded and sent to each participant. Media can be mixed individually for each participant, as in the case of audio (nobody wants to hear themselves), or mixed once for the entire session, as is often the case for video.
A multipoint control topology has the advantage of being the most scalable with minimal processing burden for clients. Whether there is a single participant or a hundred, each client creates a single bidirectional connection to the server. Clients only have to encode and upload one time and download and decode one time, so the connection has relatively constant upstream and downstream bandwidth requirements. Since images from the participants are sized down as needed to fit the mixed video stream canvas, there is also very little unnecessary data being transmitted.

The primary drawback of this topology is that the server requirements are significant. Decoding each media stream and then encoding for each mixed output means is extremely CPU-intensive. That said, the limit on how many participants can be active in a given session is only limited by the server hardware, which can be a good thing, especially when supporting resource-constrained client devices is critical. It's also worth noting that the process of decoding, mixing, and encoding introduces latency into the pipeline.

A TURN server is not required in a multipoint control topology, since the MCU can be deployed into a server environment where it listens directly on a public Internet address with no NAT to confound direct connections. A signaling server is still required, but it uses minimal bandwidth and can even be co-hosted with the MCU until additional servers are required to support the application load.

A multipoint control topology lends itself well to recording. The mixed output can be easily recorded after it is encoded, and the recording is guaranteed to be identical to what is received by the clients.

iRTC for Education

Classroom applications that connect students and teachers are well-suited to a multipoint control streaming topology. A central mixing server can record the virtual class, and simple application logic can be used to add and remove participants from the live feed so the teacher(s) can receive the focus and direct it to students as needed.

For these reasons, a multipoint control topology is best for large-scale applications that connect large numbers of participants, where a little added latency isn't a big concern, or where recording is required and integrity is critical.

Selective Forwarding

In a selective forwarding topology, each participant in a session connects to a server which acts as a selective forwarding unit (SFU). The SFU receives media from each participant and forwards the incoming packets to other participants. Like a mesh topology, the forwarding can be full or partial, which is where the "selective" part comes in. The server can selectively decide who should receive media packets based on application-level routing rules.

A selective forwarding topology has the advantages of being more scalable than a mesh topology and less expensive than a multipoint control topology. Each participant creates multiple connections to the server - an upstream connection for sending media (if contributing media) and any number of downstream connections for receiving media, generally one per peer. The media received on the upstream connections is not decoded or mixed, but simply relayed through to the corresponding downstream connections for each participant. By skipping the decoding, mixing, and encoding steps inherent to the multipoint control topology, resource consumption on the server is drastically lower, and the bulk of the added latency is avoided.

The drawback of this topology is that it's not quite as scalable on the client-side. While having just a single upstream connection makes it more upload-efficient than a mesh topology, having multiple downstream connections makes it less download-efficient than a multipoint control topology. Each client will eventually run out of resources once a certain number of participants is active in the session. That number varies from devices, but generally 10 to 15 participants is considered acceptable.

A TURN server is not required in a selective forwarding topology, since the SFU can be deployed into a server environment where it listens directly on a public Internet address with no NAT to confound direct connections. A signaling server is still required, but it uses minimal bandwidth and can even be co-hosted with the SFU until additional servers are required to support the application load.

A selective forwarding topology lends itself well to recording. The individual packet streams can be written to disk as they pass through the server, although video packets must be assembled first, and the recording is guaranteed to be identical to what is received by the clients.

For these reasons, a selective forwarding topology is best for applications that connect 10 to 15 participants, where low latency is important, or where recording is required and integrity is critical. This topology is generally considered the most balanced.

Hybrid Topology

Can't decide which topology is best? It is possible to structure your application to start with one topology and automatically switch topologies as participant counts increase and decrease. If recording is critical, for example, starting with a selective forwarding topology and then switching to a multipoint control topology around the 10-participant count could make the most sense. If cost is more important than integrity of recording, your application could start with a mesh topology and graduate as needed.

Comparison	Mesh	Multipoint Control	Selective Forwarding
Cost Effective?	Great	OK	Good
Bandwidth Efficient?	OK	Great	Good
Client Scalable?	OK	Great	Good
Server Scalable?	Great	OK	Good
Easy to Record?	OK	Great	Good
Low Latency?	Great	OK	Good

iRTC Is Extensible

Internet-based real time communications are extensible by virtue of the fact that they are Internet-based. As long as we stick with open and standardized on-the-wire protocols, integrating with third-party software is a matter of translating between domains.

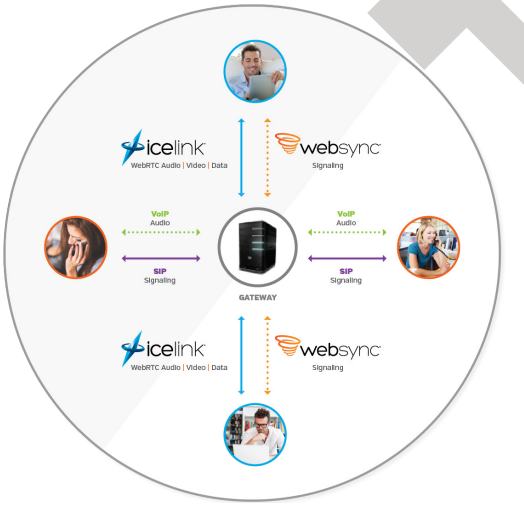
VoIP and PSTN

VoIP-based systems and the public switched telephone network (PSTN) are still a fundamental component in the usability of iRTC-based applications. While the traditional landline may be slipping away, mobile phones are still ubiquitous, and VoIP deployments in businesses are standard. Because of this, it's imperative for some applications that users are able to dial into an active session from a phone or have their phone ring when they are invited to join.

To do this, you need a gateway or switch that can talk the protocol used by VoIP phones everywhere - the Session Initiation Protocol, or SIP for short. Open source products like Asterisk and FreeSWITCH, which support WebRTC, can be very helpful for small-scale deployments. For large-scale deployments, solutions from a specialized provider like Sonus is more suitable.

iRTC for Telecom

Telecom applications that connect two people directly or multiple people via a conference bridge are well-suited to a hybrid streaming topology combined with a gateway for SIP-to-WebRTC translation. Mesh can keep two- or three-party calls inexpensive. Selective forwarding can ensure basic scalability and ensure compliance for lawful interception. Multipoint control can be used for high-volume conference calls and assist with SIP and PSTN integration.



Broadcasting

In a one-way iRTC streaming architecture for media, third-party solutions for changing encoding formats (known as transcoding, e.g. from VP8 to H.264) and switching communications protocols can be highly advantageous, especially at scale. Products like Wowza can, for example, take an RTP (Real-time Transport Protocol) audio/video stream encoded using Opus/VP8 from a WebRTC endpoint and transform it into an HLS (HTTP Live Streaming) stream using AAC/H.264. HLS introduces some latency to the pipeline, but can be efficiently redistributed across a content delivery network and is playable on virtually every device today, including mobile web browsers.

Wrap-Up

There is no doubt that iRTC is drastically reshaping the way we interact and communicate with each other. Entire industries are being forced to adapt or risk falling into obscurity as the Internet reaches into more and more areas of our lives. Thankfully, plugging into this is not nearly as big a challenge as it once was. Our collective understanding of how the Internet can be leveraged to reach people in new and exciting ways is greater than it has ever been.

If you are interested in learning more about how Frozen Mountain can help you with this, 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!