WebRTC Application

Project Description and Documentation

AGAM AGARWAL – CS12B1003
SACHIN JAISWAL – CS12B1033
RAKSHIT SINGLA – CS12B1029
AADITYA SAPKAL – ES12B1016

WebRTC Application

Project Description and Documentation

Introduction: Web Real-Time Communication (WebRTC) is a collection of standards, protocols, and JavaScript APIs, the combination of which enables peer-to-peer audio, video, and data sharing between browsers (peers). Instead of relying on third-party plug-ins or proprietary software, WebRTC turns real-time communication into a standard feature that any web application can leverage via a simple JavaScript API.

Delivering rich, high-quality, RTC applications such as audio and video teleconferencing and peer-to-peer data exchange requires a lot of new functionality in the browser: audio and video processing capabilities, new application APIs, and support for half a dozen new network protocols. Thankfully, the browser abstracts most of this complexity behind three primary APIs:

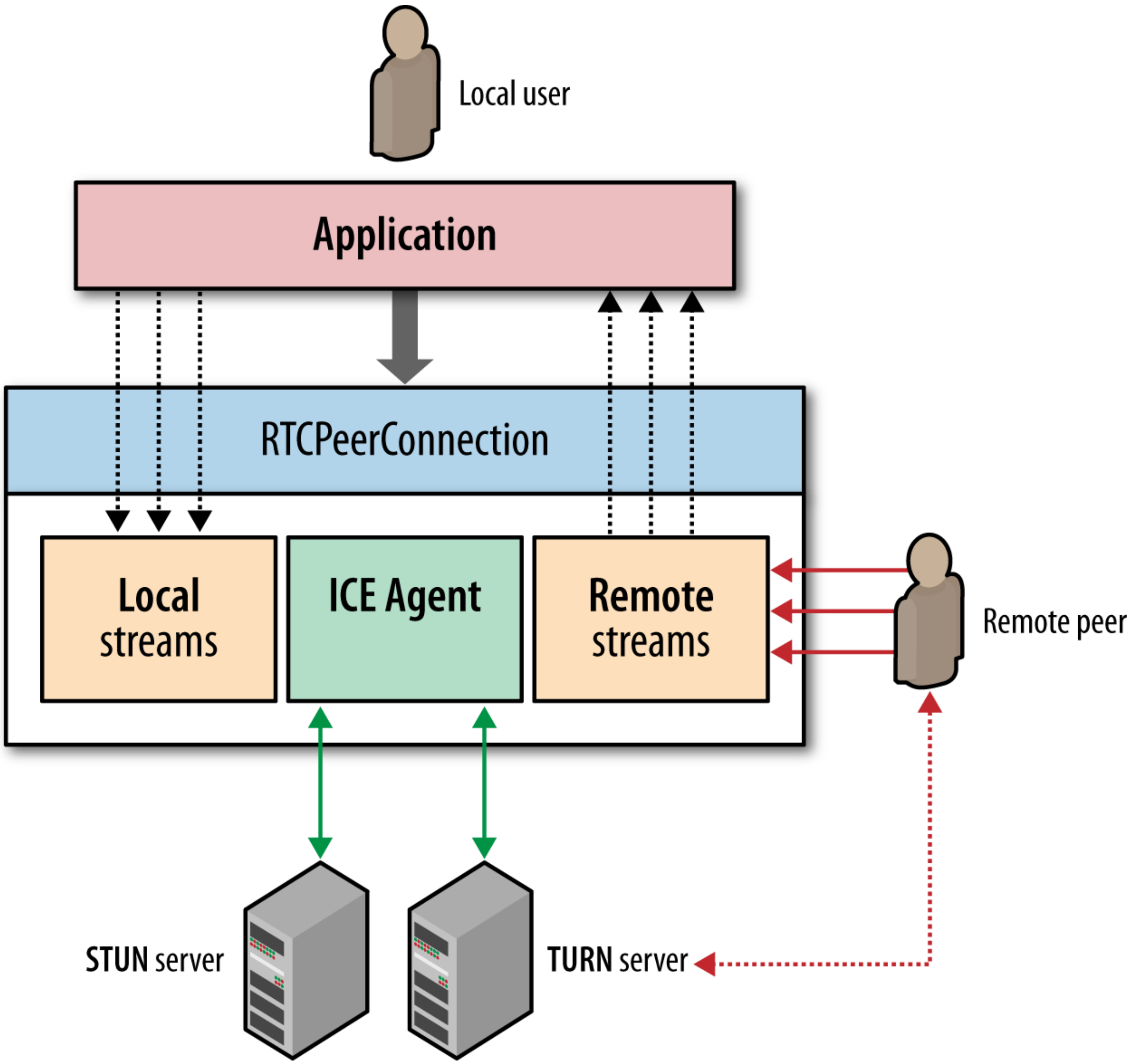
- MediaStream: acquisition of audio and video streams
- RTCPeerConnection: communication of audio and video data
- RTCDataChannel: communication of arbitrary application data

Network Transport: Real-time communication is time-sensitive. As a result, audio and video streaming applications are designed to tolerate intermittent packet loss: the audio and video codecs can fill in small data gaps, often with minimal impact on the output quality.

The requirement for timeliness over reliability is the primary reason why the UDP protocol is a preferred transport for delivery of real-time data. TCP delivers a reliable, ordered stream of data: if an intermediate packet is lost, then TCP buffers all the packets after it, waits for a retransmission, and then delivers the stream in order to the application. UDP offers no promises on reliability or order of the data, and delivers each packet to the application the moment it arrives.

ICE, STUN, and TURN are necessary to establish and maintain a peer-to-peer connection over UDP. DTLS is used to secure all data transfers between peers; encryption is a mandatory feature of WebRTC. Finally, SCTP and SRTP are the application protocols used to multiplex the different streams, provide congestion and flow control, and provide partially reliable delivery and other additional services on top of UDP.

- RTCPeerConnection manages the full ICE workflow for NAT traversal.
- RTCPeerConnection sends automatic (STUN) keepalives between peers.
- RTCPeerConnection keeps track of local streams.
- RTCPeerConnection keeps track of remote streams.
- RTCPeerConnection triggers automatic stream renegotiation as required.
- RTCPeerConnection provides necessary APIs to generate the connection offer, accept the answer, allows us to query the connection for its current state, and more.



RTCPeerConnection API