Interoperating WebRTC and IP Cameras

by **Satya K Vivek** | November 06, 2017

With millions of IP cameras already deployed in the field and the availability of new WebRTC (Web Real-Time Communication) technology, certainly there are many applications in video surveillance and broadcasting using WebRTC and IP-cameras.

WebRTC is a plugin-free peer-to-peer connection framework, which can be used to provide high-quality audio/ video/ data transfer between peers. It uses Real-time Transport Protocol to transfer audio and video. WebRTC enables direct, peer-to-peer, video, audio, and data communication between two web browsers. This allows for video calling, video chat, and peer-to-peer file sharing entirely in the web browser, with no plugins. WebRTC applications "just work" right out of the box. No Flash, no Silverlight, no Java applets, no ActiveX controls; just pure video, audio, and data communication on any webpage.

Because WebRTC is entirely peer-to-peer, you don't have to pay for any of the bandwidth across the wire and you get the highest performance and lowest latency possible.

Bridging media between IP-cameras and WebRTC

For integrating an IP camera with a WebRTC application we first need to achieve media interoperability. This means that the input stream from camera needs to be transformed to a form, compatible with the WebRTC codecs and formats supported by browsers. This means to translate whatever the IP camera speaks into

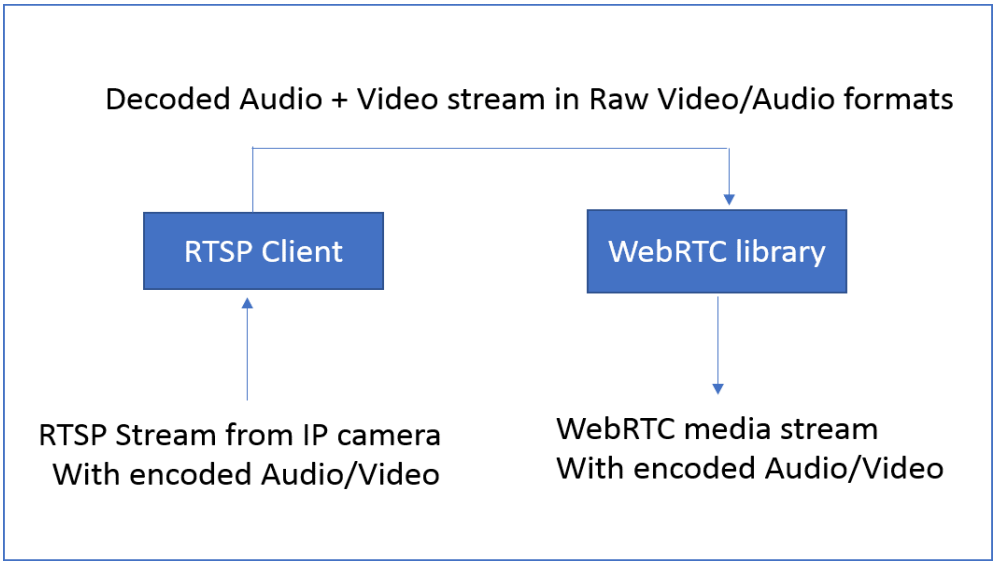
whatever the WebRTC browser supports. For this to happen, typically a piece of technology called a WebRTC Media Gateway is required.

Most IP cameras available in the market publish media using RTSP/H.264 or HTTP/MJPEG. To meet the high performance and low latency requirement of WebRTC, we need to use the media coming from IP camera in RTSP/H.264 protocol.

A media Gateway can be implemented using the RTSP client and WebRTC implementations in the host platform. However, there is a significant amount of customization needed to do this, as the default WebRTC implementation typically supports only the built-in camera of the system. For example, WebRTC implementation on Android supports only its on-board cameras. There are two ways the media can be bridged between RTSP and WebRTC.

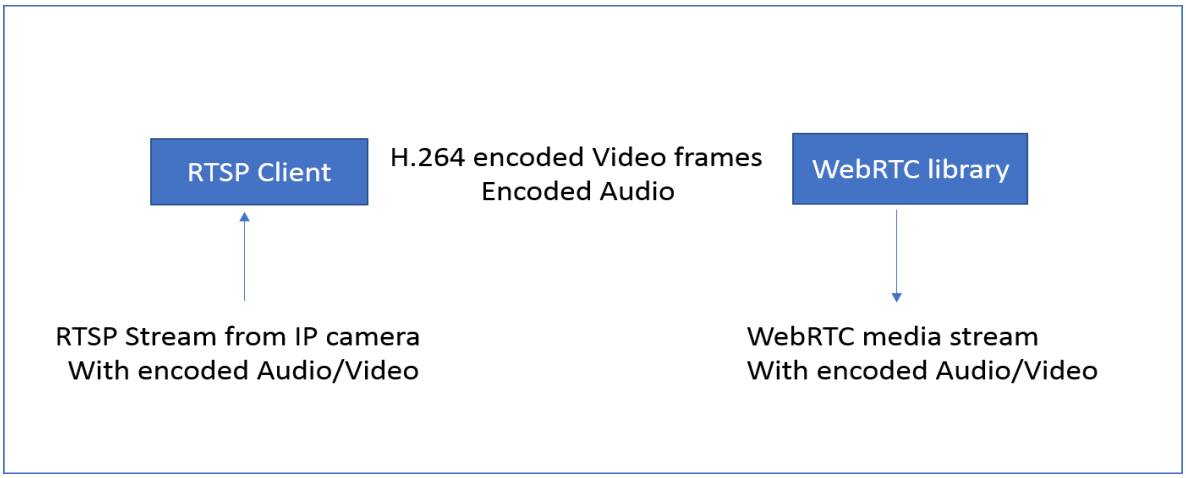
Option1 – Media Transcoding

One option is to get the video decoded into raw video formats (eg YUV_i420) and then re-encode the video using encoders, which are part of the WebRTC implementation. This requires the platform to decode the codecs in to raw audio and video formats and re-encode them to WebRTC codecs and media formats.



Option2- Media Forwarding

A more efficient way is to forward the H.264 encoded frames from RTSP directly to WebRTC. This avoids the processing overheads in transcoding the media stream. However, this needs a detailed understanding of the internals of the WebRTC implementation as well as the RTSP stream. One must also ensure that the media formats supported by the camera are compatible with WebRTC.



Implementing media forwarding involves video frame capture, video stream processing and playing the video on the WebRTC. Details on these topics are given in the sections below.

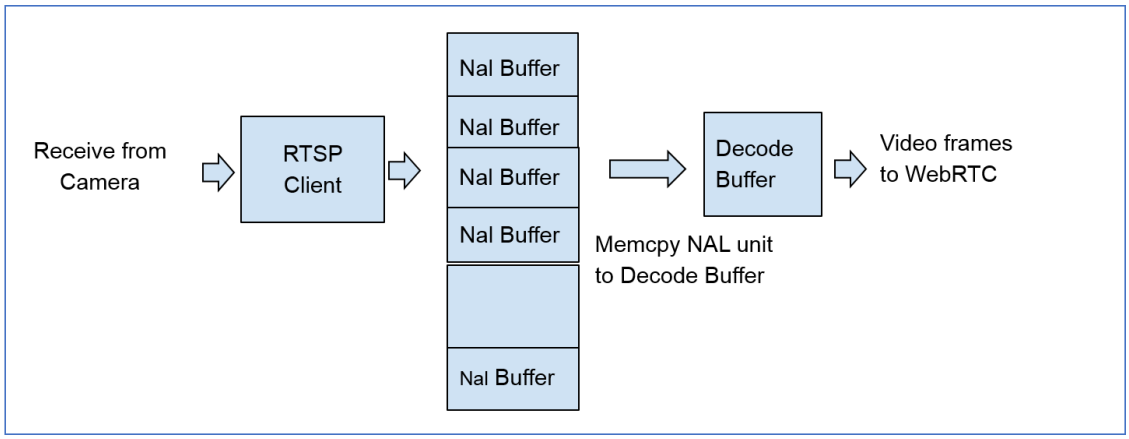
Video Frame Capture

In order to bridge the video from the IP camera to WebRTC, the gateway should capture the video stream. For this purpose an RTSP client application based on Live555 Streaming Media can be used.

IP camera uses H.264 codec for encoding the media stream. An H.264 bitstream contains a sequence of Network Abstraction Layer (NAL) units. RTSP client in Live555 decodes H.264 stream into NAL units. These NAL units are send over a socket to WebRTC end point.

Video Stream Processing and play out on WebRTC

WebRTC enabled browsers support media codecs like H.264, VP8 or VP9. Native WebRTC gets the raw frames from the device camera, then encodes it into browser- compatible formats and pass to the browser.



For the media gateway implementation, the WebRTC encoder output is overridden with the H.264 frames from RTSP. The encoder works on its input from camera. The output is packed in to RTP and sent over webRTC. We override the contents of the encoder to insert video frames from RTSP. This way, the need for decoding and re-encoding the media is avoided and the program runs more efficiently.

Conclusion

Creating a bridging application between IP-camera and WebRTC requires a great deal of understanding about various protocols like RTSP, RTP, H.264 and SDP. It can be a daunting task. Gadgeon, with years of experience in Audio/video communication can help to navigate the challenges and provide the most efficient solution.

About Gadgeon Smart Systems

Gadgeon is a Product Engineering Services company helping clients in realizing their ideas in the Internet of Things, Communication Service Provider, Healthcare, Industrial Automation, and Networking spaces. At Gadgeon, we have been delivering end to end solutions from requirements to deployment, from hardware to application.

With a team of 200+ experts with required breadth and depth of skills, domain knowledge, Program management and Process expertise we have been able to give significant ROI for both startups and established players in terms of quality and time to market.

WebRTC Online Broadcasting from IP-Cameras and Video Surveillance Systems Using *WebRTC* & Broadcasting Server.