Towards a Flexible and Scalable Media Centric WebRTC
Introduction

Web Real Time Communications (WebRTC) transforms any device with a browser into a voice, video, and screen sharing endpoint, without the need of downloading, installing, and configuring special applications or browser plug-ins. With the almost ubiquitous presence of browsers, WebRTC promises to broaden the range of communicating devices to include game consoles and set-top boxes in addition to laptops, smartphones, tablets, etc.

WebRTC has been described as a transformative technology bridging the worlds of telephony and web. But what does this bridging really imply and have the offerings from vendors fully realized the possibilities that WebRTC promises?

In this paper we discuss the importance of WebRTC Service Enablers that help unlock the full potential of WebRTC and the architectures that these elements make possible in this new era of democratization.

New Elements in the WebRTC Network Architecture

While web-only communications mechanisms would require nothing more than web browsers and web servers, the usefulness of such web-only networks would be limited. Not only is there a need to interwork with existing SIP\(^1\) and IMS\(^2\) based networks, there is also a need to provide value added services. There are several products in the marketplace that attempt to offer this functionality. But most of the offerings are little more than basic gateways.

Such basic gateways perform simple protocol conversions where messages are translated from one domain to another. Such a simplistic gateway would convert call signaling from JSON\(^3\), XMPP\(^4\), or SIP over WebSockets, to SIP, and might also provide interoperability between media formats. These gateways would not be able to offer much value add in the network. Figure 1 shows such a gateway. For simple connectivity this type of WebRTC gateway may be sufficient, but for complex call models, media functions, reliable reconnects, or when the two domains are simply not synchronized, a simple gateway falls short of providing the unique services that WebRTC needs to realize its full potential.

---

1 Session Initiation Protocol
2 IP Multimedia Subsystem
3 JavaScript Object Notation
4 Extensible Messaging and Presence Protocol
Towards a Flexible and Scalable Media Centric WebRTC

Figure 1: A WebRTC Gateway Connecting WebRTC to IP Telephony

To bridge the gap between the World Wide Web and real time communications, functionality beyond a simple gateway is required. Since a gateway performs only a simple translation, a new network element is required that provides WebRTC session control, interoperability, security, and ensures the continuity of established sessions.

WebRTC bridges the Web world with the real time communications world – the two worlds that differ in availability and reliability expectations. The need to bridge these two worlds calls for a new class of WebRTC network element, a WebRTC Service Enabler. These enablers are specifically designed to extend SIP and IMS network services to the browser by integrating strong security and stateful session management and control into WebRTC sessions. With such an enabler, web applications could add robust real-time communications that leverage the availability, scalability, and interoperability found in carrier grade networks. In the same way, a communications provider could extend their service offerings through the enabler into the web domain.

Figure 2 shows the components that would help create a comprehensive solution for WebRTC acceptance. These components are:

- A Signaling Handler to normalize signaling between WebRTC clients and others. The handler normalizes nuances between various implementations of SIP protocols or JSON and converts between the two when necessary. It also enforces policy using an industry standard Diameter interface to a PCRF\(^5\) element. Since the Signaling Handler has a holistic view of the network and deployment requirements, it is in a position to influence the default behavior of WebRTC clients and enforce these requirements

- A Media Handler to provide media services, including transport protocol interworking if necessary. The handler mediates complex media conditions such as multiple media streams, media codec translation, network media devices, encryption, recording, etc.

\(^5\) Policy and Charging Rules Function
Figure 2: Comprehensive WebRTC Service Enabling Components

Such a decomposed architecture creates a platform for flexible and functional WebRTC deployments. The architecture:

- Promotes agility, elasticity, utilization, and high availability
- Optimizes media path and provides quality-of-service
- Enforces interoperability of disparate networks, transports, and protocols
- Is customizable with policy integration and enforcement
- Ensures user independence from devices and networks
- Leverages specialized equipment such as media servers for recording, conferencing, etc.
- Achieves optimum usage by load balancing of media over several media handlers

Media Centric Functions Facilitated by WebRTC Service Enabler

This architecture enables a variety of media functions by intelligently integrating the Signaling and Media Handlers. With the knowledge of the session requirements, the Signaling Handler may choose to let media flow directly between endpoints, or may instruct a Media Handler to anchor it. It may even change the course of the media in mid-session. This “Dynamic Media Anchoring” is a powerful way to bring a variety of services to a WebRTC environment, as described in the following sections.
Media Path Establishment and Optimization

The path that media takes through the network frequently determines the media’s perceived quality. The best path may not be the most direct path! Sometimes a direct path may not even exist. The Signaling Handler working with one or more Media Handlers determines the most optimum path media should take in the session.

One media path concern arises because of the presence of NAT\(^6\) devices and firewalls. The WebRTC standards mandate that WebRTC endpoints use ICE\(^7\) and STUN\(^8\) protocols for this purpose. ICE and STUN protocols allow these clients to discover their external addresses and to test proposed media paths before selecting the most suitable path.

The WebRTC Service Enabler is particularly effective in determining media flow. Since signaling messages flow through the Signaling Handler, it can influence the path decision. The Media Handler includes a STUN server that helps clients discover their external addresses. Clients require their external addresses in order to generate a proper offer using the SDP\(^9\). If direct media flow between clients cannot be established using the STUN service, media may be anchored through the Media Handler, which can perform a relay function according to the TURN\(^{10}\) protocol.

---

\(^6\) Network Address Translation  
\(^7\) Interactive Connectivity Establishment  
\(^8\) Session Traversal Utilities for NAT  
\(^9\) Session Description Protocol  
\(^{10}\) Traversal Using Relays around NAT
Because a WebRTC Service Enabler features decomposed signaling and media handling functions, it offers flexibility to deploy several Media Handlers under the guidance of a single Signaling Handler. For instance, by analyzing the geographic locations of the communicating endpoints, the Signaling Handler can select the closest Media Handler instance to anchor media between client devices. This selection may help minimize the time media spends in an unmanaged network, thereby optimizing the media path and resulting in an increased quality of service.

**Figure 4** shows a distributed deployment of the Signaling and Media Handlers. The figure also shows a call between a WebRTC client on the unmanaged Internet and a SIP phone on a managed corporate network. Media from the WebRTC client hops onto the managed network at the closest Media Handler in order to preserve its quality.

In other calls the direct path between communicating endpoints may be best and in that case media anchoring may not even be required.
Figure 4: Selection of the Geographically Closest Media Handler

Media Encryption and Security

Security is a major part of any network, especially one that is designed to operate over the public internet. The WebRTC Service Enabler needs to ensure call signaling and media are secure. HTTPS\(^{11}\) ensures security for signaling between the user and the Signaling Handler. The verification of media addresses and setup of media channels is more complex and requires a two-step process. ICE and STUN procedures verify agreement to communicate between the media entities. DTLS\(^{12}\) handshakes are performed on every channel they establish and SRTP\(^{13}\) over DTLS is used for the media flow.

Since the SRTP over DTLS transport standard is not prevalent in traditional IP telephony environments, the Media Handler provides interworking between a WebRTC media channel and unencrypted SIP networks (as shown in Figure 5).

\(^{11}\) Hypertext Transfer Protocol over Secure Socket Layer  
\(^{12}\) Datagram Transport Layer Security  
\(^{13}\) Secure Real Time Protocol
Media Interoperability

There is a large variety of audio and video codecs available for media transmission. Each codec has distinguishing characteristics that make it the most suitable in a given use-case. Ideally, each endpoint is equipped with a variety of codecs and codec selection for a particular session takes into account the specific requirements for the session, the set of codecs supported by the communicating peer. This selection is generally enforced by a communications server.

The Signaling Handler controlling call signaling between WebRTC clients is able to enforce media policies and ensure interoperability. At times, due to policy enforcement and support, there may not be a common supported codec between the entities. In these cases, transcoding resources need to be inserted. The Media Handler is able to anchor the media and insert resources as needed to ensure compatibility.

Media Enhanced Services

The WebRTC standards specify the establishment of point-to-point media. If a session involves more than two parties, then the media from all of the participants in the session must be mixed by a media server and re-distributed. For optimal use of media server resources, ports on the server should be dynamically allocated and released even during the middle of a session. For example, a session may start as a simple person-to-person call with peer-to-peer media flow. One of the participants may invite another party, creating an ad-hoc conference. At this point, the media must be anchored for media mixing. After consultation, the added party may drop-off returning the call to a two-party session. Now, the media resources that were allocated are no longer required and should be released, thus ending the anchoring requirement.

Figure 6 shows how a two-party call that begins with media flowing directly between the two endpoints dynamically converts to a three-party conference with media anchored by the Media Handler that directs the media stream to a media server for mixing and distribution.
Figure 6: Conferencing Services using the Media Handler

Media Policy Enforcement

The role of policy cannot be underestimated in a large scale deployment. Policy dictates who uses resources; who is authorized to establish sessions; the destinations and time of day, etc. 3GPP\textsuperscript{14} has defined the PCRF and the Rx interface the PCRF offers to an AF\textsuperscript{15}.

To take advantage of this architecture, the Signaling Handler acts as the AF and uses the Rx protocol to interact with a standard PCRF during call initiation and renegotiation to ensure that the caller has sufficient privileges for the desired action. In this way, the provider can control subscriber access to services; throttle the bandwidth; limit the number of streams that a user is allowed at a time and enforce other deployment specific rules.

Media Compliance Recording and Lawful Interception

As WebRTC moves to mainstream deployment and is used in industries subject to compliance and regulation, such as in finance or healthcare industries, service providers and enterprises may be required to implement session recording and real-time intercept. When the WebRTC Service Enabler anchors media, it can invoke recording or interception resources in the media path for both these purposes. Figure 7 shows how anchored media between endpoints can be forked to a recording device.

\textsuperscript{14} Third Generation Partnership Project

\textsuperscript{15} Application Function
Deployment Prerogatives Addressed by WebRTC Service Enabler

User Authentication

The WebRTC Service Enabler provides flexible mechanisms for verifying user identities, including third-party authentication services, such as those provided by Facebook™, LinkedIn™, and Google™. The Signaling Handler verifies user credentials with such providers using the OAuth¹⁶ protocol.

Service Agility, Availability, and Elasticity

A decomposed architecture significantly improves network flexibility scalability and resource utilization. The interface between Signaling and Media Handlers allows a single Signaling Handler to control several Media Handlers or vice-versa. Media capacity is increased by simply adding Media Handlers on the fly. A Signaling Handler may load-balance sessions among available Media Handlers for maximum utilization and availability.

Service agility and elasticity is further enhanced by virtualizing Signaling and Media Handlers. Handlers running as virtual machines on COTS¹⁷ hardware may dynamically be scaled or replicated as necessary to handle anticipated traffic without the need for costly server rollouts.

Session Continuity, Device and Network Handoff

Because WebRTC signaling and media software is downloaded into the browser on demand, communications sessions are vulnerable to interruption by a browser refresh, roaming between Wi-Fi access points or other external events. Users can accidentally kill an established communications

¹⁶ Open Standard for Authentication
¹⁷ Common off-the-shelf
Towards a Flexible and Scalable Media Centric WebRTC

Session rehydration can be extended from page refreshes in a single browser instance to browsers running on different devices and the same device roaming between networks. For example, the WebRTC Service Enabler enables a user to suspend a session on their mobile device when they come into the office and resume the session on their desktop device. Similarly an established session may be preserved if the user travels from Wi-Fi to 3G/4G network or vice-versa. The WebRTC Service Enabler recognizes the change in connectivity and reestablishes the session using the new transport. Figure 8 shows how a session is maintained and the media stream reconnected when a user crosses these boundaries.

Figure 8: Media Handover across Devices and Networks using WebRTC Service Enabler

Conclusion

WebRTC is an exciting new technology that brings real-time voice and video communications to any device capable of running a web browser. The technology is currently evolving and requires a strong network infrastructure to realize its full benefits.

A decomposed and distributed WebRTC Service Enabler provides network architects with mechanisms to design and deploy a variety of value added service offerings to their users and customers. Adopters benefit by not being locked into proprietary solutions. The WebRTC Service Enabler overcomes media challenges ensuring services can be quickly adopted. By taking these challenges off the table, the WebRTC Service Enabler helps democratize communications and facilitates mainstream WebRTC adoption.
For More Information

WebRTC Session Controller on Oracle.com:

White Paper on Delivering Enterprise-Class Communications with WebRTC: