nemertes RESEARCH
INDEPENDENCE INTEGRITY INSIGHT

issue paper

Q1 16

Achieving the Promise of WebRTC for Pervasive Communications

By Irwin Lazar
VP and Service Director, Nemertes Research

Compass Direction Points:

- **WebRTC has failed to live up to hype –** Just 4% of companies are actively implementing WebRTC.
- **Use cases exist; business cases don't –** IT leaders cite lack of ubiquitous browser support and video codecs, NAT traversal and network issues, and standard disagreements as limiting their adoption and evaluation efforts.
- **WebRTC alternatives are rapidly emerging –** Revisit the promise of browser based, plugin free, rich communications and evaluate new approaches to delivering business value.

Executive Summary

For almost five years WebRTC has been hyped as the next big thing in enterprise collaboration for its ability to enable voice, video and web conferencing on any browser, without the need for plugins or separate apps. To date, WebRTC has failed to achieve on its promise due to a variety of issues ranging from slow development of the WebRTC standards, lack of ubiquitous support in all browsers, lack of agreed-upon video codec, and network challenges. Now, new approaches including downloadable codecs and browser-based containers finally offer the ability for enterprises to realize the vision of ubiquitous conferencing, on any device, using a browser.

The Issue

Introduced in 2011 by Google as an open source project, WebRTC offered the promise to turn any browser into a voice, video, and/or web conferencing endpoint without the need for plugins or extensions.

Fast forward almost five years later and WebRTC's promise remains unfulfilled: Just 4% of companies participating in Nemertes' 2015-16 Unified Communications and Collaboration Benchmark are using it today, while 64% have no active WebRTC plans. Codec battles, standards disputes, and network configuration challenges have led the majority of end-user organizations to shun WebRTC, relegating it to a technology that developers can use to embed voice/video conferencing inside of dedicated apps, or one that only supports WebRTC-compliant browsers (e.g. Google Chrome and Mozilla Firefox.) Those wishing to extend UC to all browsers or integrate it with web-based apps must still rely on plug-ins that are complex to manage, need frequent upgrades, and may require paying a licensing fee for the use of video codecs. Now, new approaches are emerging that seek to deliver on WebRTC's promise, while overcoming its constraints.

What is WebRTC?

At its essence, WebRTC is a set of JavaScript Application Program Interfaces (APIs) that enables web browsers to natively support voice, video, and data sharing without the need for a plugin or extension. WebRTC provides for peer-to-peer connections between browsers, or it can allow for the extension of UC platforms to any browser through the use of a WebRTC gateway. WebRTC developers have leveraged as many existing protocols as possible – using secure HTTP (https) for

session establishment, RTP (real-time protocol) for media, and the Session Description Protocol (SDP) from the Session Interworking Protocol (SIP) standard for the exchange of session configuration information between endpoints (e.g. security configuration, voice/video codecs, etc.).

Realizing the Vision of Ubiquitous Browser-based Communications

WebRTC delivers three primary benefits:

1. The ability to extend enterprise UC platforms to any desktop or mobile device, inside or outside of the organization. This means that guest or remote workers, customers, and/or partners can participate in multimedia conferences without having to overcome IT security restrictions that often block the installation of software or browser plugins on their local devices.
2. Enabling click-to-call, click-to-videoconference, or click-to-share capabilities on customer-facing websites to offer an enhanced level of customer engagement, while also reducing costs associated with toll-free services.
3. The opportunity for software developers to embed voice, video, or screen sharing capabilities into any browser-based app, either point-to-point, or by interfacing with backend UC platforms.

Use cases for WebRTC are almost limitless. Examples provided by Nemertes research participants include:

- Enabling distance learning and employee training via video conferencing to customer locations. Using a WebRTC capable browser, a remote employee can join a video or web conference without conflicting with local restrictions against software download and installation.
- Web-based remote desktops for home workers, specifically contact center agents. Using a web-based desktop with embedded telephony eliminates the need to provision dedicated software that must be inventoried, secured typically via IPSec VPN, and frequently upgraded.
- Customer-facing services like 'click-to-videoconference' through a web site, enabling high-touch interaction with premier customers, or giving customers the ability to show customer service agents a problem they are having with a product.
- Using videoconferencing for HR purposes to interview prospective employees without having to rely on consumer services, or forcing the interviewee to download a software client that requires account provisioning.

- Embedding click-to-call into business apps, enabling task workers to easily chat with or call co-workers to address specific issues without the need to switch to a telephone or separate softphone client.

Why Has WebRTC Failed?

Despite its promise, WebRTC is not the ubiquitously deployed technology available in any desktop and mobile browser that early developers hoped to see. Instead WebRTC-based capabilities today are largely limited to those using Chrome or Firefox, via plugins into non-WebRTC-compliant browsers, or via the use of WebRTC components embedded into dedicated desktop and mobile applications like Cisco Spark, Facebook Messenger, and Unify Circuit. None of these approaches have allowed application developers to easily add voice/video/conferencing to any web-based application, or have enabled the easy extension of enterprise UC platforms beyond the firewall to partners, suppliers, and customers.

By the end of 2015 just 4% of participants, comprised primarily of large enterprises with more than 2,500 employees had actively deployed WebRTC within their organizations, a slight decline compared to 2014 (Source: Nemertes' 2015-16 Unified Communications and Collaboration Benchmark.)

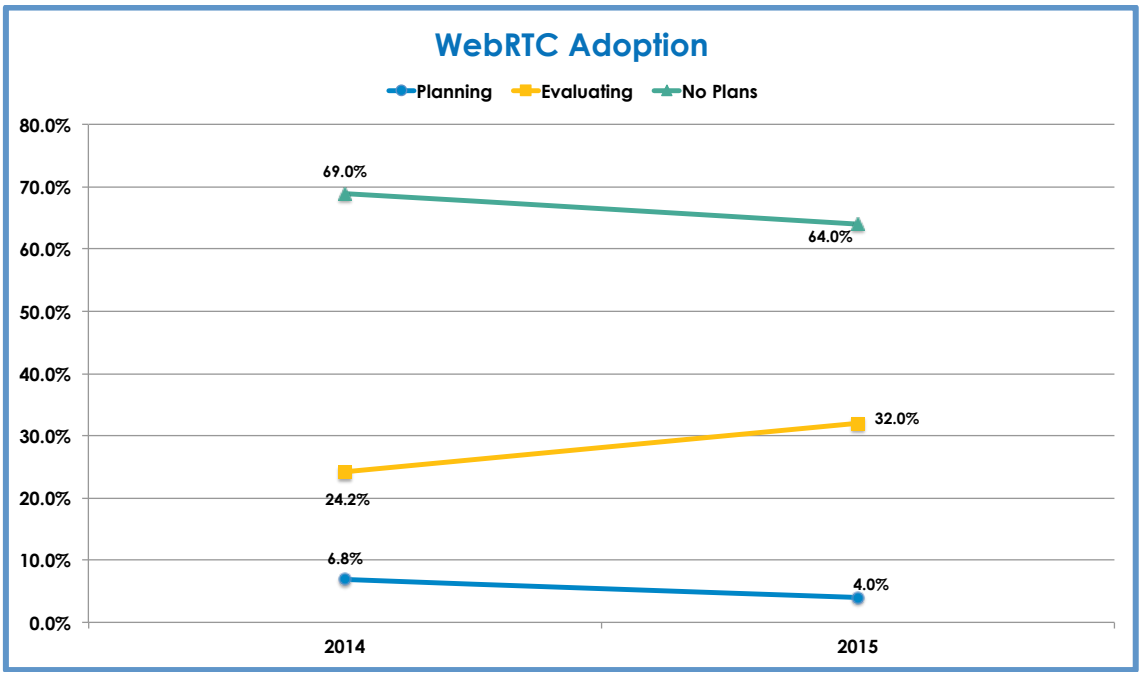

Figure 1: WebRTC Adoption

While the percentage of companies evaluating WebRTC did rise over that same period of time, few of those evaluating it in 2014 moved it into production deployments in 2015.

The vast majority of those with no plans to deploy cite the lack of business case as their primary reason for not using the technology. (Please see Figure 2.) Even if they are familiar with WebRTC, and its potential use cases, the lack of ubiquitous browser support, standard video codecs, and network complexity issues limit the ability to build a business cases that would drive adoption in WebRTC deployment. The director of UC and Collaboration for a global pharmaceutical company sums it up best. He said, "The WebRTC ship has sailed. Since it doesn't work on all browsers it's not a viable technology for us." The executive director for a global non-profit organization added, "It's a great idea, but it's just not available right now."

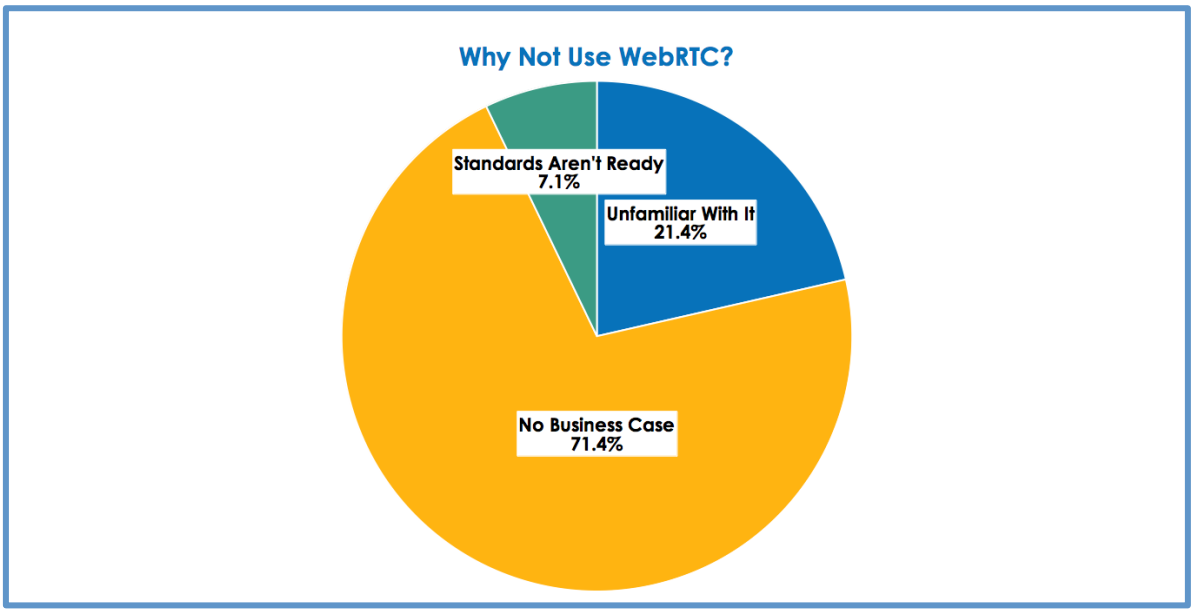

Figure 2: Why Not Use WebRTC?

Primary limitations include, but are not limited to:

- Lack of ubiquitous browser support
- Network address translation requirements
- Lack of an agreed-upon universal video codec
- Network impact
- Disagreements over standards development

Lack of Ubiquitous Browser Support

Today WebRTC 1.0 libraries are available within major browsers including Google Chrome and Mozilla Firefox, as well those with smaller market share such as Opera and Ericsson's mobile browser Bowser. Most notably missing from that list are Apple's Safari and Microsoft's Internet Explorer. Microsoft's latest browser, Edge,

does support a modified version of WebRTC called ORTC (Object API for RTC for WebRTC) that isn't supported by other browsers.

Network Address Translation Requirements

WebRTC's reuse of SIP/SDP components means that it, like SIP, requires special treatment in networks using private IP addressing and network address translation (NAT). A WebRTC-enabled browser attempting to establish a connection with another browser or a gateway to a UC platform, will embed its own IP address in the SDP portion of its session origination message, telling the far end where to send its response. This approach works fine if there exists an IP routed path between endpoints (or endpoint and gateway). However if the originating WebRTC-enabled browser is using a private IP address, the far end browser or gateway may not be able to respond because if it has no return path to the originating browser.

WebRTC standards address this issue by mandating support for STUN (Session Traversal Utilities for NAT), TURN (Traversal Using Relay Around NAT) and ICE (Interactive Connectivity Establishment) services. These approaches enable WebRTC browsers to learn their public IP address by communicating with a server outside of their own networks. Once they know how they appear to the outside world, they include their public IP addresses as the return addresses within the SDP header. Use of STUN/TURN/ICE requires availability of both publicly reachable servers, as well as the ability of a NAT device to store state information mapping internal private IP address to public IP address so that the NAT device knows how route the return packets to the correct private IP address. The NAT device also must be able to map media streams to session requests to ensure proper traversal.

Lack of an Agreed-Upon Video Codec

Perhaps the biggest issue over the last two years hindering WebRTC adoption has been the lack of an agreed-upon codec for encapsulating and decapsulating video. Google, in Chrome supports its own video codec (VP8) that it makes available for free use under an open source license. Nokia and MPEG-LA have sued Google in the past claiming that VP8 violates their patents.

Meanwhile most videoconferencing vendors use MPEG-LA's H.264 video codec. And some mobile devices, like Apple's iPhone, only provide hardware processing for H.264 (improving speed and reducing battery drain.) Use of H.264 (and its eventual successor, H.265) is complicated by the need for those using it to pay a royalty fee to MPEG-LA.

The competitive market further complicates the video codec battle. Apple and Microsoft compete with Google and thus have little incentive to support VP8, while Google has little incentive to add H.264 support to its browsers. To overcome this hinderance to broad WebRTC adoption, in November of 2014, the IETF's RTCweb working group mediated a compromise, requiring all browsers to support both VP8 and H.264. However, browser vendors have yet to comply, meaning those wishing to

use VP8 on browsers like Internet Explorer and Safari must rely on a plugin, eliminating the benefits of WebRTC's plugin free architecture. Absent universal support for H.264, VP8-based WebRTC clients will require transcoding to interconnect with the vast majority of room-based videoconferencing systems.

Network Impact

Even if the video standards battle is settled, and support for VP8 and H.264 becomes ubiquitous, these are both older codecs that are being supplanted by newer versions – VP9 and H.265, respectively. These advanced codecs offer a variety of benefits including reduced bandwidth consumption and support for higher video quality at the same bandwidth as older codecs. WebRTC standards do not yet mandate support for these newer standards, meaning that WebRTC-based video conferencing sessions will use more bandwidth than applications that are able to leverage these newer codecs. Furthermore, licensing fees associated with H.265 threaten to block widespread inclusion in browsers. Although several royalty-free alternatives are in the works such as Cisco's Project Thor and Daala, none of these are yet ready for wide-scale deployment or are supported by all browsers.

Disagreements over Standards

The final challenge to WebRTC deployment is the lack of agreement on its future. Led by Hookflash and Microsoft, a new alternative architecture has emerged: ORTC. The ORTC working group's approach within the W3C (World Wide Web Consortium) replaces SDP with a new JavaScript API that is designed to offer more programming flexibility. The ORTC effort is happening in parallel to WebRTC development efforts within the W3C WebRTC working group and as such is often referred to as WebRTC 2.0, or 1.1. Google is working with the ORTC working group, and it's likely that with the support of both Google and Microsoft, at some point we'll see ORTC as the next evolution of WebRTC, supported in Chrome, Edge, and hopefully Safari. But at this point, ORTC support is not yet available outside of Microsoft Edge, and ORTC does nothing to overcome the previously detailed issues around lack of browser ubiquity, NAT, video codecs, and network impact. The only advantage to ORTC is a simpler programming environment for application developers.

A New Approach

Because the use cases for WebRTC are still viable, a number of vendors have been hard at work developing alternative approaches that aim to realize WebRTC's goal of browser ubiquity while overcoming its drawbacks. Among these are CaféX's Meetings, Genband's Omni Client, and Temasys universal WebRTC plugin. An overview and assessment of each is provided below.

CaféX Meetings

CaféX has for the last several years delivered WebRTC-based solutions to enable browser-based voice/video calling and screen sharing, integrated into contact center platforms. With Meetings, CaféX aims to solve WebRTC problems as detailed in Table 1 by leveraging JavaScript capabilities available within all modern browsers to enable dynamic download of required codecs. In effect, Meetings provides an alternative to WebRTC that delivers on WebRTC's promise and is supportable by all browsers without requiring software downloads.

WebRTC Challenge	Meetings Solution
Lack of ubiquitous browser support	Uses inherent capabilities available in all leading browsers (including Chrome, Edge, Firefox, Internet Explorer, and Safari) via JavaScript to support voice/video/screen sharing without requiring additional software
Network address translation requirements	Can function in either P2P or client-server mode, using UDP or TCP, no embedding of IP address in header
Lack of an agreed-upon universal video codec	Dynamically downloadable codecs loaded from a Meetings server as sessions are established, supports connectivity to H.264-based room systems
Network impact	Eventual support for H.265 and VP9 will enable browser-based services to take advantage of emerging codecs. Ability for endpoints to form clusters reduces WAN bandwidth demand
Disagreements over standards	Not tied to WebRTC standards development, could eventually support whatever emerges out of ORTC and WebRTC working groups

Table 1: CaféX Meetings Assessment

Genband Omni Client

Genband Omni Client runs WebRTC within a containerized application on any operating system. While Omni Client enables WebRTC to run on any device, it does not do so via native capabilities available in local browsers. Instead, the Omni Client approach runs its own WebRTC-compliant browser within the container. This means that IT must provision the client, and support it, rather than relying on already deployed browsers. Table 2 provides an overview of Omni Client and its limitations.

WebRTC Challenge	Genband Solution
Lack of ubiquitous browser support	Omni Channel client technology runs inside of a dedicated app, on any endpoint, enabling support for WebRTC regardless of native device capabilities
Network address translation requirements	Supports ICE, STUN and TURN
Lack of an agreed-upon universal video codec	H.264 and VP8 natively, must traverse a gateway to connect to endpoints using other video codecs
Network impact	No support for emerging H.265 / VP9 video codecs

Disagreements over standards	Omni client will require upgrades as WebRTC standards evolve

Table 2: Genband Omni Client Assessment

Temasys WebRTC Plugin

Temasys delivers a universal plugin for non-WebRTC compliant browsers like Internet Explorer and Safari. Temasys' goal is to deliver a single plugin that any WebRTC solution can use, thus requiring only one plugin or extension for each browser. However, the Temasys solution is limited by the need for all WebRTC solutions to rely on it rather than requiring their own plugins. Table 4 provides an assessment of Temasys' ability to solve WebRTC challenges.

WebRTC Challenge	Temasys Solution
Lack of ubiquitous browser support	Plug-in for Safari and Internet Explorer. Available to any application wishing to use it, but many applications are likely to require their own plug-ins
Network address translation requirements	Limited by WebRTC constraints
Lack of an agreed-upon universal video codec	Supports VP8 and H.264, enabling connectivity to H.264-based room systems
Network impact	No support yet for emerging H.265 / VP9 video codecs
Disagreements over standards	Temasys plugin will require upgrades as WebRTC standards evolve

Table 3: Temasys WebRTC Plugin

Conclusion and Recommendations

WebRTC offers a great deal of promise to improve collaboration and customer engagement, but the lack of ubiquitous browser support, network address translation traversal challenges, lack of an agreed-upon universal video codec, lack of support for emerging next-generation video codecs, and ongoing disagreements over standards have limited its adoption among end-user organizations.

A number of vendors are delivering approaches to support WebRTC's use cases, while overcoming its limitations. CaféX Meetings delivers a replacement technology that enjoys universal browser support and an architecture that can reduce network bandwidth impact. Solutions from Genband and Temasys still require plugins or dedicated apps, but can extend WebRTC to non-compliant endpoints and (in the case of Temasys) provide a single plugin that any app can leverage. IT leaders should evaluate each of these solutions to determine which can best deliver on the promise of WebRTC to deliver rich communications to any browser, inside or outside

their organization, without the need for plugins or app downloads. Specifically, they should look for:

- Approaches that do not require maintaining plugins or dedicated apps
- That provide potential support for emerging VP9 and H.265 codecs
- That offer added benefits such as reduced network bandwidth utilization, or that eliminate the need for NAT
- That provide a path toward eventual support of ORTC
- That can be easily embedded into other web-based applications.

About Nemertes Research: Nemertes Research is a research-advisory and strategic-consulting firm that specializes in analyzing and quantifying the business value of emerging technologies. You can learn more about Nemertes Research at our Website, www.nemertes.com, or contact us directly at research@nemertes.com.