Enabling WebRTC Services for the Enterprise

Produced by nojitter

Sponsored by ORACLE

By Eric Krapf, Program Co-Chairman, Enterprise Connect; Editor, No Jitter

Executive Summary

WebRTC holds great promise for flexible real-time communications services within the enterprise, and between users in the enterprise and those connecting via the public Web. But enabling these services effectively means more than just enterprise deployment of voice/video-enabled browsers running WebRTC. Organizations will also have to implement server-side technologies to ensure that applications and services leveraging WebRTC do so while maintaining enterprise-grade levels of:

- **Reliability**
- **Interoperability**
- **Scalability**
- **Security**

This white paper will describe the concepts, technologies, and solutions required in order to provide enterprise-grade WebRTC-based communications. It will provide you with recommendations for evaluating WebRTC infrastructure elements, as they emerge, to verify that they can truly provide these four critical enterprise characteristics.

WebRTC represents the most significant step in years toward the "webification" of real-time communications, because it allows for embedding of voice and video capabilities natively within Web browsers and mobile apps, with no need for additional downloads or plugins. Thus WebRTC promises nothing short of a Web that is fully capable of supporting real-time communications in every session. And that means we'll be a major step closer to a world where the Internet/World Wide Web is the public communications network.

But simply enabling the endpoint, aka the browser, isn't enough to realize the dream of a Web that supports real-time communications that's suitable for the enterprise. To meet this criterion, the network itself must enable real-time communications; and if WebRTC-based communication is to be anything other than a hobbyists' plaything—if it's going to be enterprise-class—then upgrades to the network must do more than simply address well-known issues like Quality of Service.

Specifically, real-time enterprise communications across the Web must be reliable, interoperable, scalable, and secure. Furthermore, the experience for the WebRTC end user must reach the minimum standards of existing communications networks—sessions can't drop as the user moves around or switches devices in mid-session; the network must adapt to the vagaries and challenges of browser performance and best-effort Internet service; and communications must flow seamlessly among users on any endpoint. And it all has to happen in a secure environment.

Providing Enterprise-Grade Performance

Reliability

If WebRTC is going to be useful to the enterprise, it must function within systems built for enterprise-caliber performance. That's where our four key features of reliability, interoperability, security and scalability come in.

The first of these criteria, ***Reliability***, has come to mean much more than what this term used to represent in the era of the Public Switched Telephone Network (PSTN). In a communications-enabled Web, reliability also has to be more than what we've come to expect from the first-generation data-centric Web. In the future, Web-centric enterprise communications will demand reliability that draws from the best of both the PSTN and Web traditions, and overcomes the most serious flaws of prior generations.

For the most part, the reliability of the old PSTN was pretty solid—the famous five-nines or 99.999% uptime. In today's IP-centric world, devices like Session Border Controllers are already enabling the redundancy and fault-tolerance that IP networks require if they're to match this benchmark. So at the network level, one minimum requirement is to carry forward the PSTN's level of reliability for the Web-based world.

But high network availability and uptime won't be enough in the communications-enabled Web of the future. Reliability will also mean ensuring the continuity of each individual session, no matter what may threaten to break that session. This new generation of reliability issues goes all the way out to the endpoint, and touches issues that were never relevant in the PSTN.

Take browsers themselves: They've become more stable over the years, and browser crashes are less frequent than they used to be, but they do still occur; and multitasking users may accidentally close out a browser tab that's supporting a WebRTC session. Or a content issue may cause a user to reload the page that they're using for WebRTC-based collaboration—not realizing that the reload will disconnect the communications component of the session.

So to deal with such scenarios, the designers of the WebRTC standards came up with a process called *rehydration*. A system that supports rehydration maintains session state on a server within the network, rather than on the endpoint. Thus, if a session is discontinued without the proper process, the server can return the state information to the browser so that the session can be re-established at exactly the point it was dropped.

Rehydration is an example of how reliability in the era of Web communications can even exceed that of the PSTN: If you accidentally hang up on a landline call, or if a cellular call drops, one party has to actively re-start the session. With WebRTC rehydration, the network re-starts the session for you automatically.

Rehydration is also the technique by which WebRTC-enabled systems will provide for device-to-device handoffs. Since it's very likely that our world will continue to grow ever more mobile, and users will use ever-more devices, individual WebRTC sessions will need to be kept running when, say, a user starts a session on her workstation, but wants to transition the same session to her mobile as she leaves the office. As long as the session state is being maintained on the relevant server element within the network, presenting the same session credentials from a different device will allow the user to transfer the session seamlessly.

There are other features of a Web-based communications world that will add new, richer meaning to the notion of reliability. For example, the Opus codec, which is used for HD audio in WebRTC, is designed to deliver better service over variable Internet connections than what older codecs could provide. Similarly, when WebRTC-based communications accompany Website-based customer contact, the tighter integration will allow for more powerful "context" features—so first-call resolution will improve, and the experience of dealing with customer care will be a more reliable process.

Interoperability

WebRTC works in the browser, but as most people know if they've studied the technology at all, it still hasn't been implemented natively in *all* browsers. WebRTC is supported in the following desktop browsers and subsequent releases:

- Google Chrome 23
- Mozilla Firefox 22
- Opera 18

It is not supported in Microsoft Internet Explorer or Apple Safari, and those two companies have been tight-lipped about any potential plans for future support. In addition, as of this writing, the separate implementations of WebRTC do not fully interoperate (i.e., Chrome with Firefox, etc.).

This lack of interoperability has kept WebRTC from taking off virally in the consumer world, but it doesn't have to prevent WebRTC from being useful to the enterprise. In fact, when it comes to leveraging WebRTC functionality, enterprises may find that, for once, they have the jump on consumers. That's because enterprise networks are well suited to brokering diverse multimedia endpoints and allowing them to connect and share many features and functions.

A consumer-oriented website may be reluctant to base its Web communications on WebRTC at a time when not all users can take advantage of the technology in their preferred browsers. But enterprises already have deployed middleware and border elements that translate various endpoint protocols so that different multimedia services can talk to one another. Enterprises are familiar with using SBCs not only to interwork IP-based SIP trunks; they can even interwork different vendors' incompatible PBXs via the SBC. Adding WebRTC endpoints to this mix is straightforward—just add a similar software component that supports WebRTC (in its varying implementations) and now the enterprise can support as much WebRTC as the real world has enabled.

And make no mistake: While nobody knows exactly what the future holds, interoperability is likely to continue to be a challenge indefinitely for WebRTC, even if all the major browser vendors move to truly support the standard. With any standard, Vendor A's implementation may be fully compliant, as may Vendor B's—yet the two may well not fully interoperate. As different browsers and other application interfaces adopt WebRTC, it's more likely than not that their developers will be looking for ways to differentiate their product, and will discover such differentiation in features that don't quite mesh with other implementations of the same standard.

In the consumer-to-consumer world, this lack of full interoperability might be an annoyance, but the traditional trade-off for these customers is that they simply put up with some inconsistencies in return for totally free usage. When an enterprise gets involved—in either a B2B or B2C scenario—the need for a better experience will drive the implementation of network-side and border elements that broker a more seamless operation.

Scalability

When it comes to scalability, network-side upgrades will be key to allowing enterprises to provide ubiquitous communications to their employees as well as those outside the enterprise who wish to communicate with internal resources via WebRTC. But to provide scalable connectivity, the enterprise will need the ability to manage signaling and addressing across all users.

In its most basic form, WebRTC lets one browser talk to another browser via a Web server. In this scenario, you could go about finding the server you want to talk to via the Domain Name Service (DNS) that has driven Web addressing since the dawn of the World Wide Web. You could locate publicly-accessible resources via a Google or other search. But the whole point of enterprise networking—and one of its fundamental requirements--is that not every enterprise resource is, or should be, accessible by these public means. So locating and establishing contact with enterprise resources—in a manner that conforms to enterprise policies—requires an element that can broker such contact and session establishment requests, and can do so at scale.

The WebRTC standard doesn't specify signaling protocols or addressing that must be used. Instead, developers of WebRTC-based applications and browser implementations will choose whatever they believe works best for what they're trying to accomplish. This means that it's up to the enterprise to support signaling across the public Web and translate that signaling, where necessary, to conform to the enterprise's own standards—often this will be the Session Initiation Protocol (SIP).

In addition, this network element will be more useful if it provides a software development kit (SDK) that enables application developers to write WebRTC applications that will work more seamlessly with the enterprise platform. With such an element providing flexible, enterprise-wide addressing and signaling, as well as an SDK for application development, the enterprise will be able to scale its effective use of WebRTC to embrace both B2B and B2C scenarios in which users are accessing enterprise resources via the Web.

Security

The final component in an enterprise-grade Web communications system is security. WebRTC is inherently a highly secure technology, because packets are encrypted end to end. But while encryption is very effective in deterring many types of security breaches, most notably eavesdropping, it doesn't address all of the security-related concerns that enterprises are likely to have.

The first such issue is authentication/authorization/accounting (AAA). These are as critical in WebRTC-based communications as they are in any other earlier-generation forms of communication, and the WebRTC standard itself does not provide for AAA.

AAA is not simply a matter of enterprises needing to make sure that only authorized users are using their systems or communicating with their servers. The enterprise must also consider the nature of that authorization: Just as with earlier generations of Web-based e-commerce, enterprises will necessarily want to open up WebRTC-based communications to the public, outside the firewall, and will want to do so in a way that's conducive to ease-of-use for the consumer, while still providing the necessary level of security. That means an enterprise may want to allow users to employ credentials from one or more social media sites (if it's a B2C scenario), or their enterprise-level credentials (for B2B). But enforcing such authentication is something that has to happen at a middleware level within the enterprise architecture that supports the WebRTC-enabled service.

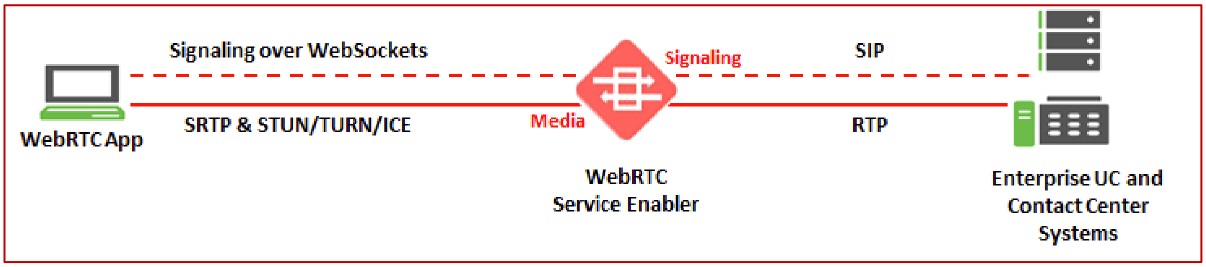
WebRTC also throws network architects, managers, and security professionals an additional curve: The data channel.

Because WebRTC has a channel that can pass data simultaneously with the exchange of real-time voice and video, opening up your enterprise to WebRTC poses significant security risks that must be addressed—basically, you're opening a bi-directional pipe from the inside of your network to the outside. Implementing strong AAA mechanisms will be crucial to ensuring that unauthorized people don't access enterprise data via the WebRTC data channel.

Finally, in addition to AAA, the enterprise has to protect its network from Denial of Service-type attacks that could bring down the whole enterprise. That's especially important in a world where e-commerce sites are both a likely early-adopter scenario for WebRTC, and a perennial target for hackers wanting to do damage to online businesses. Blocking DoS attacks is a proven necessity for existing network elements such as firewalls and Session Border Controllers, and this functionality will be required in next-generation WebRTC network elements as well.

The Solution: A Platform for Enterprise WebRTC Service Enablement
For all of these issues—reliability, scalability, interoperability, and security—a peer-to-peer WebRTC network simply can't provide adequate enterprise functionality. It requires a platform within the enterprise's own architecture that can provide the additional services, functionality, connectivity, and security that enterprises require.

At a high level, here's what that enterprise WebRTC Service Enabler looks like and where it sits in the network:



The "WebRTC App" can be a browser running WebRTC, or another type of application that uses WebRTC (such as a native mobile application) as its interface to the communications-enabled Web. Similarly, the WebRTC Service Enabler itself can be a distinct piece of software running on a physical or virtual server, or it can be a component of a larger network element. What is crucial is that the WebRTC Service Enabler is the point at which enterprise policies and technology choices are invoked in order to manage the relationship between the enterprise resources and those who wish to access those resources via real-time communications over the Web.

Conclusion

WebRTC offers a powerful paradigm for the next-generation communications-enabled World Wide Web: The notion of the real-time communications-enabled browser. Such an endpoint can open up a wealth of opportunities for enterprises to communicate with customers, business partners, remote employees, and others, in formats that can blend media types and integrate application functions in ways not easily implemented up to now.

But the enterprise will need to establish a platform from which to enable the many services that WebRTC potentially provides. Enterprise leaders should be thinking now about the next generation of Web-enabled communications, and how they will leverage this emerging trend in ways that both benefit and protect their business. A WebRTC service enablement platform and strategy is the place to start.

About Oracle

Oracle engineers hardware and software to work together in the cloud and in your data center. For more information about Oracle (NYSE: ORCL), visit www.oracle.com