



In-Browser Communication Strategies Fulfilling the Promise of WebRTC

By Ben Pinkerton & Mark Noble | March 2015

ABSTRACT: In-browser real-time communication offers the promise of simplified B2C and B2B communications. The VidyoWorks™ platform delivers the most robust and comprehensive in-browser video communications solutions today and will deliver even greater value as the WebRTC standard evolves to include Vidyo's scalable video coding technology through its collaboration with Google.

Table of Contents

Introduction	3
What is WebRTC?	3
What WebRTC is not	4
What is ORTC?	5
VP8 video codec	6
VP9 video codec	6
H.264 video codec.....	6
H.265 video codec.....	7
Audio Codecs	7
Browser Support.....	7
Mapping In-Browser Communication Approaches to Use Cases	9
Mobile Device Support.....	11
Conclusion.....	12
Resources.....	13
Vidyo	13

Introduction

In-browser communications can simplify the connection for the first time user by delivering the communication experience directly within a web portal and eliminating the need for a separate communications client. Today, there are two complementary approaches to achieve in-browser communication:

- WebRTC, an API definition that allows web developers to create clients embedded in webpages that access a device's audio and video components, as well as the codecs that are included with the browser, to deliver real-time audio and video without requiring the user to install any software when using a browser that supports the WebRTC standard
- Browser extension software that enables the use of audio and video devices and any codec in any web browser, providing flexibility to optimize the user experience and the efficiency of connections through an enterprise infrastructure

While both of these approaches have distinct advantages and disadvantages, they are not mutually exclusive and can be leveraged in tandem to serve the needs of different use cases.

There is also a clear path to the best of both worlds via a single approach in the future with the next generation of WebRTC.

Vidyo delivers the most robust and comprehensive solution suite for in-browser communications today and is driving the future of WebRTC in collaboration with Google. As a part of that relationship, Vidyo has licensed scalable video coding (SVC) technology for Google to include into the next generation WebRTC codec, known as scalable VP9, as well as software to be included in the Chrome™ browser.

This paper provides the information necessary to understand in-browser communications solutions available today and what is planned for the future in order to help an enterprise plan strategy around this important access technology.

What is WebRTC?

WebRTC is an open standard developed by W3C¹ to provide real-time communications from within a web browser. At the most basic level, WebRTC allows for the creation of an instant messaging, voice or videoconferencing client built into the web browser that can be embedded within a web page. By putting the basic building blocks for media handling, encoding / decoding, and transport into the web browser itself, real time communication

¹ <http://www.w3.org/TR/2015/WD-webrtc-20150210/>

clients can be created easily using standard web development techniques without requiring a download or installation of software.

The W3C specification is complemented by corresponding on-the-wire standards developed by the IETF, including tools for security and NAT traversal. An important novelty in WebRTC is that it allows one browser to natively send data directly to another browser, bypassing the web server, something that has not been possible in the past. This allows direct, “peer-to-peer” connections to be easily established between browsers in order to perform point-to-point calls.

With WebRTC much of the underlying complexity of building real-time applications is abstracted from the web developer. This means it is somewhat easy to set up basic point-to-point video calls using basic web development skills rather than building more complex native client applications. Because the core pieces of a video client are built into a web browser no client side software, other than the web browser, needs to be installed.

What WebRTC is not

As mentioned above, WebRTC is client side technology, and does not include any backend or server-side components. Therefore, if the application built using WebRTC requires anything beyond simple point-to-point calling, additional infrastructure will likely be required. There are several key functions that require some form of server infrastructure that any developer using WebRTC will need to address. Additionally, WebRTC still lacks some functions that are considered critical for many applications.

- **Call signaling** – WebRTC does not define any form of call signaling. Call signaling is necessary for handling call control and setup between endpoints and other application components. The good news is that virtually any type of call signaling can be plugged into WebRTC, however this requires another layer of expertise to select and then incorporate a signaling method. Using an established infrastructure that supports WebRTC and already provides for signaling eliminates this requirement.
- **Multipoint video** – While WebRTC can support limited multipoint without additional server infrastructure; this requires media to be sent using a mesh topology. A mesh requires n video streams to be transmitted from each endpoint in order to support n number of participants in a call. This topology significantly limits practical multipoint usage due to the processing and bandwidth burden it places upon the endpoint devices.
- **Recording** – In order to record audio and video some form of centralized infrastructure is required. The recording infrastructure must compose the multiple audio and video streams into a format that is viewable and provide content management, access control, and distribution. To attempt to perform client-side recording would add burden to the client machine performing the recording.

Additionally, making the recording easily accessible would require some form of content management and distribution.

- **Interoperability** – If the application requires integration with non-WebRTC based systems, some form of infrastructure will be required to provide interoperability. Infrastructure delivering interoperability will provide both media transcoding between disparate codecs and signaling interoperability. For example, a gateway type of infrastructure may be required to translate a WebRTC session into a H.323 session to allow browsers and traditional H.323 endpoints to communicate.
- **Integration to Systems and Applications** – Many communication processes utilizing WebRTC will benefit from integration into existing systems such as ERP, CRM, analytics, trouble ticketing, process automation and other applications. Using an established platform, which offers server API's for integration into these applications, can significantly reduce the web developer's workload and time to market.
- **NAT Traversal** – Most users connect to the Internet through some form of Network Address Translation (NAT) device. This can lead to a call where the media does not properly flow between endpoints. In order to establish end-to-end media connections across firewalls with Network Address Translation, the support of servers to provide services such as media termination, STUN, and TURN is required.

What is ORTC?

ORTC (Object Real-Time Communications) is an extension to WebRTC that allows for lower level access to the underlying technology within WebRTC. ORTC was largely driven by Microsoft and evolved from a competing standard to WebRTC. The competing standard did not gain acceptance by the standards community and thus ORTC was introduced to augment WebRTC with the capabilities that Microsoft deems important. There is a misconception that ORTC is an alternative to WebRTC, but this is not true. ORTC is an incremental step in WebRTC and can be thought of as WebRTC v.1.1. In fact, ORTC is developed within the W3C, in the ORTC Community Group.² ORTC adds additional capabilities for developers that want to have more granular control over sessions. For example, ORTC can provide access to adjust parameters of a specific media stream within a call rather than changing parameters for the entire call.

² See <https://www.w3.org/community/ortc/> and <http://ortc.org>.

VP8 video codec

The VP8 codec is an open-source video codec provided free by Google. On2 Technologies whom Google acquired in 2010 originally created the VP8 codec. After the acquisition of On2 Technologies, Google released VP8 and its specifications without charge as a means of driving video support. VP8 is used in the streaming WebM format as well as in WebRTC, where it is a required codec along with H.264 under the MTI (Mandatory To Implement) specification of the IETF. After a long period of deliberation for a Mandatory-to-Implement (MTI) codec between H.264 and VP8, the IETF decided to require both. In other words, in order for a web browser to be compliant to the WebRTC standard it must support the MTI codecs; VP8 and H.264.

VP9 video codec

VP9 is the successor video codec to VP8. There are two major benefits that VP9 delivers over VP8, bandwidth savings and scalability. The VP9 codec is anticipated to provide a 30% to 40% reduction in bandwidth versus VP8 for a given video quality. Alternatively, this increased efficiency can be used to improve the video quality that is possible over the same bandwidth.

The second major benefit of VP9 is with regard to scalable video coding. The VP8 codec supports only temporal scalability (frame rate). However, Vidyo is collaborating with Google so that Google can add temporal and spatial (resolution) scalability to the VP9 codec. The result will be a video codec that delivers two dimensions of scalability permitting an application to vary both frame rate and resolution. Scalability is extremely important for multipoint applications where video tile layout may change and for error resilience. With multipoint applications layouts frequently change and participants switch between small tiles and large tiles during a typical conversation. Another significant benefit of scalability is error resilience; something very important when transporting real-time video over unreliable networks such as Wi-Fi and 4G networks.

It is important to note that a codec with scalability does not provide multipoint and dynamic adjustment by itself. Infrastructure is still required in order to intelligently manage the video layers and adapt the video based upon available bandwidth, CPU, and screen resolution. Infrastructure that is scalable video coding aware does not need to transcode video to achieve multipoint and dynamic adaptation.

H.264 video codec

The H.264 video codec is one of the most widely used video codecs today. Having been standardized over a decade ago, it has been widely adopted. For this reason many organizations want to use H.264 to enable interoperability with existing video systems without the need to transcode video codecs. However, H.264 does have known royalty

obligations associated with it, while it is generally believed that VP8/VP9 is royalty-free. This was the source of contention in the IETF when attempting to define the MTI codecs in WebRTC. As mentioned above, a compromise was reached and both H.264 and VP8 are MTI codecs.

Vidyo is a pioneer in the use of the H.264 Scalable Video Coding (SVC) codec in video communications. Vidyo participated in the standardization of H.264 SVC (Annex G of H.264) and delivered the first ever video communications platform based on H.264 SVC. Vidyo has several patents essential for implementing H.264 and is [a member of the MPEG-LA patent pool](#) for its contributions to the H.264 SVC standard.

H.265 video codec

The H.265 video codec is the successor video codec to H.264. The major benefit of H.265 over H.264 is one of bandwidth savings. The H.265 video codec was developed with the goal of delivering the same video quality as H.264 at half the bandwidth. While still very early in the adoption phase of the H.265 video codec, it does have a lot of potential. Like H.264, the H.265 video codec does require payment of royalties. However, the potential bandwidth savings cannot be ignored.

Vidyo has been instrumental in driving the H.265 HEVC video standard along with its scalable extension H.265 SHVC. In fact, Vidyo is a founding member of the newly created MPEG-LA H.265 patent pool due to the patents it owns that are essential for implementing the standard.

WebRTC was designed to be extensible to allow virtually any audio or video codec to be used. However, until the codec is part of the web browser it can not be used without requiring installation of software. As of the writing of this paper, no browsers support H.265.

Audio Codecs

WebRTC also supports multiple audio codecs. The primary audio codec being adopted is the Opus codec. Opus is an open source codec made available free that delivers excellent performance over a wide range of use cases from voice to music. In order to support backwards compatibility with older VoIP systems, G.711 is also supported.

Browser Support

The vision of WebRTC is to encourage wide scale voice and video communications across the web. However, for WebRTC to be successful in its vision, it must be supported in all major web browsers. To have pervasive communication, a person using Google Chrome™

should be able to communicate with another person using Internet Explorer®. Unfortunately, this is not yet the case with WebRTC. Not all web browsers have implemented support for WebRTC.

Google has championed the WebRTC standard and has the most advanced WebRTC support within the Google Chrome browser. Google has favored support for open-source codecs such as VP8 and VP9 it believes to be royalty-free.

Mozilla has also been fairly aggressive at supporting WebRTC. In fact, Google and Mozilla have done a significant amount of work testing interoperability between Chrome and Firefox™. Mozilla has favored the free codecs as well, currently supporting VP8. However, in 2013 Cisco provided an H.264 plug-in for Firefox and agreed to cover the cost of H.264 royalties, through the OpenH264 project. This enabled Firefox to be the first browser to support WebRTC with H.264.

Microsoft has been very active in the standards body for WebRTC, initially favoring an alternative standard to WebRTC. However, at the time of this writing, Internet Explorer still does not support WebRTC. Microsoft has announced they will support WebRTC via ORTC in a future version of Internet Explorer, but it is not shipping as this whitepaper is being written. In the meantime the only way to provide support for Internet Explorer is through the use of a browser add-on. Microsoft was a major supporter of using H.264 as the MTI codec instead of VP8. However in November of 2014, there was a compromise in the IETF making both VP8 and H.264 MTI codecs.

Apple has been silent regarding support for WebRTC. While they have been participating in the standards discussions, they have made no announcements regarding their intention to support or not to support WebRTC. Today, the only way to enable in-browser video communications in Safari® is through the use of a browser extension.

Vidyo provides the VidyoWeb™ browser extension as a native SVC endpoint capability that can be embedded within a web page in any of the major web browsers using simple JavaScript APIs. As a native SVC endpoint, it connects directly to the VidyoRouter™ server and delivers all of the error resilience and scalability benefits afforded by Vidyo's patented dynamically adaptive infrastructure. The first time a new user attempts to connect via the VidyoWeb browser extension they are prompted to install a plugin that adds the Vidyo SVC codec to the browser. Once installed, subsequent interactions are click-to-connect. Whether used as the primary means for guest engagement or as the means for gaining cross browser support for non-WebRTC browsers, the VidyoWeb™ browser extension plays a critical role in any in-browser communications strategy today.

	Google Chrome™	Mozilla Firefox™	Microsoft® Internet Explorer®	Apple® Safari®
WebRTC VP8	YES	YES	NO	NO
WebRTC H.264	NO	YES	NO	NO
WebRTC VP9	YES (experimental)	NO	NO	NO
VidyoWeb™ Browser Extension	YES	YES	YES	YES

Table 1: Comparison of in-browser communications support by browser vendor

Mapping In-Browser Communication Approaches to Use Cases

In today's context, both the native WebRTC and browser extension approaches deliver value but also have some limitations. By leveraging both approaches, enterprises can deliver the right solution based upon the needs of a given use case. When the next generation of WebRTC with support for scalable VP9 comes to market, the best of both worlds will be achievable via a single approach. The table below highlights the value of each approach and helps to align approach with use case.

VidyoWorks Delivers WebRTC & Better Solutions

Native WebRTC for Convenience, Vidyo Browser Extension for Performance 




	 VP8 WebRTC	 Vidyo Extension	 VP9 WebRTC
No installation (for supported browsers)	✓		✓
Availability	✓	✓	Future
No client cost	✓	✓	✓
API for integration into web apps	✓	✓	✓
Interoperability with legacy voice and video	✓	✓	✓
Reliable Performance over Internet/Wireless		✓	✓
Scalable Multi-party Conferencing		✓	✓
Available Across All Major Browsers		✓	

Table 2: Comparison of in-browser communications approaches using the Vidyo platform

Vidyo enables WebRTC developers to leverage their existing investment in their WebRTC clients by providing a JavaScript API for connection to Vidyo infrastructure. Vidyo brings native VP8 WebRTC clients into Vidyo conferences via an infrastructure software product called Vidyo Server for WebRTC. Delivered as a virtual machine for VMware™ environments, the Vidyo Server for WebRTC transcodes the VP8 media from the WebRTC client to H.264 SVC and connects to the VidyoRouter™ for selective forwarding to other participants in the conference, regardless of which access technology they are using. The

key advantage in using native VP8 WebRTC is the elimination of the need for any installation to get a first time user connected to a conference. The trade-off for this simplified initial connection is the processing cost of transcoding and its impact on scalability, the loss of Vidyo's patented network error resilience technology on the VP8 leg of the connection and the lack of ubiquity of WebRTC compliant browsers.

The VidyoWeb™ browser extension addresses the current limitations of WebRTC and provides the highest quality experience regardless of which web browser is used and scales most efficiently for large numbers of interactions. Since the VidyoWeb browser extension utilizes H.264 SVC, it is able to natively connect to the VidyoRouter™ server without transcoding and benefit fully from Vidyo's patented dynamic adaptation algorithms for superior performance over variable networks and end user devices. By eliminating the need for transcoding, the processing requirement per connection on the infrastructure side is reduced by an order of magnitude, enabling significantly greater density for scale. The VidyoWeb browser extension is also available across all of the major browsers in the market today. The VidyoWeb browser extension does, however, require the user to accept the installation of a browser plugin for the first time connection. This is a similar experience to installation of an Adobe® Flash® plugin, for example, and the user simply clicks to connect for any subsequent interaction.

Given the above, if first time installation is a non-starter for a given use case and the enterprise can dictate the browser to be used for connection, the native WebRTC approach via JavaScript API and Vidyo Server for WebRTC is the right solution. If the use case demands reliable high quality, flexibility across browsers or cost efficiency for large scale deployment, and can tolerate a first time plugin installation, then the VidyoWeb browser extension is the more efficient solution.

Vidyo has implemented its approach to guest access via web browser such that the user can be presented with the option to connect without installation (assuming they are using a WebRTC enabled browser) or provide a better quality experience via the VidyoWeb browser extension for those willing and able to install a plugin for initial connection.

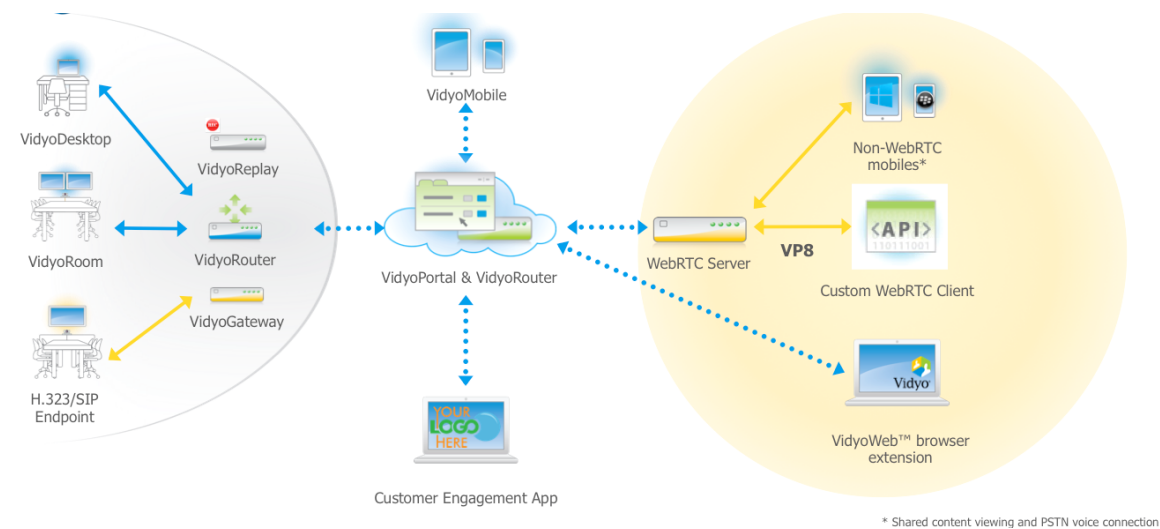


Fig 1: Native WebRTC and VidyoWeb browser extension implementation for the Vidyo infrastructure

When next generation WebRTC based upon scalable VP9 becomes available, the best of both of the above approaches will become available via a single WebRTC based approach for browsers that support the standard. This means that without any installation, the native WebRTC client will be able to connect directly to the VidyoRouter server eliminating the need for transcoding and making the solution more scalable, while also benefiting from Vidyo's error resiliency. The VidyoWeb browser extension will continue to be required for web browsers that don't support the WebRTC standard with scalable VP9.

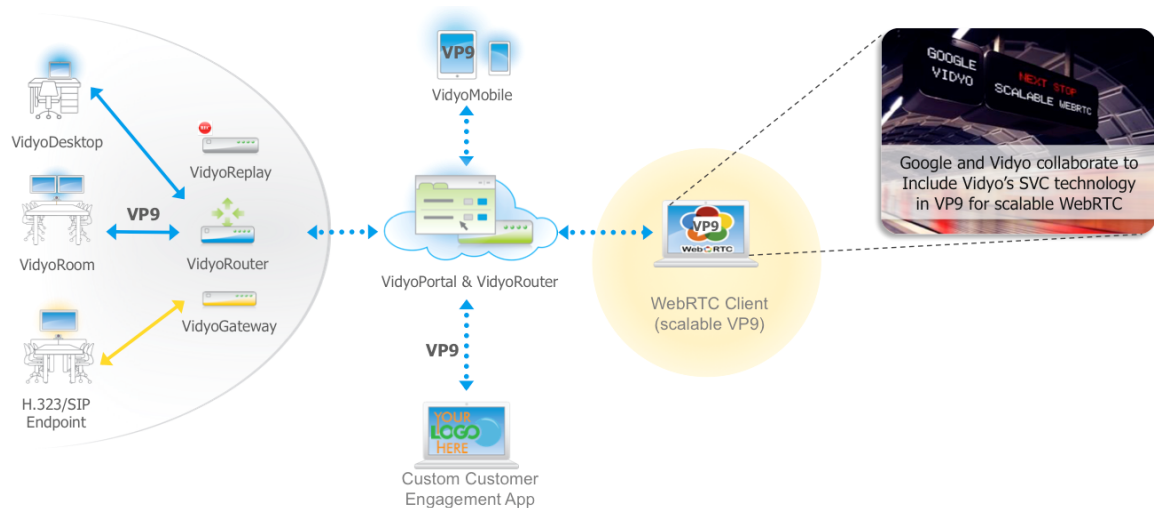


Fig 2: Future native WebRTC implementation on VidyoWorks™ platform using scalable VP9

Mobile Device Support

While the focus of this paper is in-browser communications, one important consideration is support for mobile devices. Increasingly, users are accessing online services using mobile devices such as phones and tablets. When building a communications strategy, mobile device support cannot be ignored. While newer Android™ based devices support WebRTC using Chrome for Android, there is no native WebRTC support for iOS based devices. On the surface this may seem to be problematic. However, the current practice for delivering mobile experiences is through apps that run natively on the mobile device.

If mobile device support is required, it is important to consider a platform that provides a mobile device SDK. This allows mobile app developers to easily integrate video communications directly into the app. While WebRTC is supported in Chrome on Android this is not the same as having WebRTC libraries to embed within an app. For example, to build a mobile app using only the open source WebRTC libraries there are a number of challenges to deal with. Differences in device screen sizes, camera resolutions, on-board echo cancellation, CPU power, etc. all need to be addressed by your application. Attempting to build an app without an SDK that is ready to embed can be much more complicated.

Vidyo provides mobile device APIs for both Android and iOS delivering high quality and error resilient video communications through a native SVC implementation, embedded into

any mobile application or using the off-the-shelf VidyoMobile™ client. Having error resilient communications is especially important for mobile devices as their access to the network is primarily via Wi-Fi and 4G wireless connections. Wireless connections tend to vary dramatically making quality communications quite challenging.

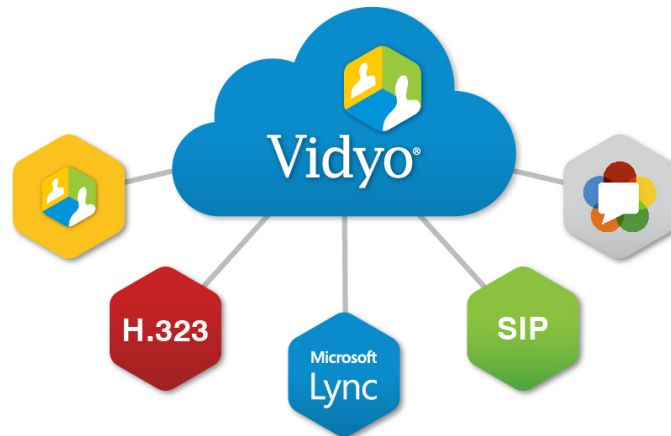


Figure 3: The VidyoWorks platform easily supports a variety of communication modes

Using the VidyoWorks™ platform, mobile devices can communicate with native WebRTC endpoints, Vidyo SVC based endpoints, third party H.323 and SIP based endpoints as well as Microsoft® Lync®, if the application demands integration with existing video communications systems and infrastructure. This high level of interoperability along with the flexibility of the VidyoWorks APIs makes it the perfect platform for building rich communications enabled applications.

Conclusion

As an access technology, WebRTC holds great promise for simplifying the future of in-browser communications. As we collectively look forward to the maturation of the standard and broad adoption across the major browser platforms, Vidyo delivers the platform, infrastructure, and most comprehensive suite of in-browser and mobile communications solutions, that solve real-world B2C and B2B use cases today with currently shipping, 3rd generation products that support both native WebRTC and browser extensions for richer, more scalable experiences across all major browser platforms. Vidyo customers and development partners will enhance their competitive advantage as the next generation of WebRTC, incorporating Vidyo's scalable video coding technology, comes to market. Consumers will benefit from the power of the VidyoWorks™ platform inside the WebRTC browser without the need for software installation.

Resources

Find more information about the VidyoWorks™ platform and the Vidyo products described in this paper by using the links listed below.

Vidyo

- Vidyo web site:
<http://www.vidyo.com>
- Vidyo Support Center:
<http://www.vidyo.com/services-support/technical-support/>
- Vidyo Server for WebRTC Datasheet:
http://www.vidyo.com/wp-content/uploads/DS-Vidyo_Server_for_WebRTC.pdf
- VidyoWorks™ web page:
<http://www.vidyo.com/products/extend/>

Vidyo, Inc. (Corporate Headquarters)

433 Hackensack Ave., Hackensack, NJ 07601, USA

Tel: 201.289.8597 Toll-free: 866.998.4396

Email: vidyoinfo@vidyo.com



Vidyo[®]

EMEA

emea@vidyo.com

+33 (0) 488 718 823

APAC

apac@vidyo.com

+852 3478 3870

INDIA

india@vidyo.com

+91 124 4111671

www.vidyo.com

© 2015 Vidyo, Inc. All rights reserved. Vidyo and other trademarks used herein are trademarks or registered trademarks of Vidyo, Inc. or their respective owners. All specifications subject to change without notice, system specifics may vary. Vidyo products are covered by one or more issued and/or pending US or foreign patents or patent applications. Visit www.vidyo.com/patent-notice for more information.

Rev: 2015-03-25