Security of Real-time Communication on the Web (WebRTC)

Lieven Desmet, KU Leuven

Martin Johns, SAP AG

Abstract

With the introduction of WebRTC, real-time communication on the Web, both web technology and real-time communication entered a new era. WebRTC is a joint effort of W3C and IETF, supported by a large industry consortium. The technology offers real-time peer-to-peer communication capabilities directly in the end-user's browser, accessible to websites via JavaScript APIs. Because of the numerous potential applications and the accessibility of the technology, WebRTC is expected to quickly become a game-changing technology in the communication landscape.

WebRTC brings an extremely powerful technology close to the end-user, and at the same time opens up the path for innovative adoptions within the communication landscape. Key of its success will be how this real-time communication of the web can be secured, both at the network as well as the application layer.

In order to make a WebRTC connection, the end-user typically visits a website of a service provider, called the *calling site*. By calling the appropriate JavaScript APIs in the end-user's browser, the calling page is able to set up the WebRTC connection with a remote party. WebRTC uses two distinct paths for communication: the *signaling path* follows the path from call initiator to call receiver via the existing HTTP(s) connections to the calling site(s), the *media path* is set up in a peer-to-peer fashion between call initiator and call receiver using network protocols such STUN for NAT traversal and DTLS-SRTP for real-time communication. For identity and authentication, both communicating parties employ an *Identity Provider (IdP)*.

The large flexibility in deployment is at the same time one of the most exciting as well as most challenging aspects of the technology. As will be illustrated in several articles in this issue, WebRTC can be applied in multiple deployment settings, ranging from a simple scenario with one user contacting the helpdesk from a calling site, to more complex scenarios with multiple calling sites, multiple identity providers, and media gateway services working together to set up video conferencing facilities. The specifications of WebRTC are deliberately open to allow multiple scenarios by explicitly decoupling the signaling plane and media plane, the calling pages and the identity providers, but the variety in trust models also means that no single security solution will fit all.

From a security point of view however, the combination of peer-to-peer real-time communication (e.g. across multiple firewalls and NAT boxes) as well as the availability of a rich set of JavaScript APIs (e.g. video capture and streaming) makes this an interesting target for a broad set of attacks. Attackers can for instance target the communication on the network, abuse the JavaScript APIs, hijack the website of the service provider, or create identity confusion - to name only a subset of potential resulting attack scenarios.

Furthermore, WebRTC is the first browser-based technology that breaks with the Web's strict client-server architecture by enabling direct server-less browser-to-browser communication. This has significant security consequences, as the Web's primary security policy (the Same-origin Policy) is based on assurance of server-identity, using the global SSL-PKI infrastructure. For the end-users however, no established method exists to ensure peer-to-peer authenticity and end-user identity, resulting in future challenges in the area. A fact that is well reflected in this issue's articles.

In this issue …

Security of Real-time Communication on the Web, the theme of this month's issue, focuses on the provisioned security characteristics and remaining security challenges of the emerging WebRTC technology. Although this issue only presents a small selection of security topics within the field of WebRTC, it adequately covers the major challenges within the technology: the overall security architecture, the provisioning of user identities in the browser environment and securing media gateway infrastructures.

Browser-to-Browser Security Assurances for WebRTC (Richard Barnes and Martin Thomson)

In "Browser-to-Browser Security Assurances for WebRTC", Richard Barnes and Martin Thomson assess the end-to-end security assurances based on the architectural model of WebRTC. This article discusses in a step-by-step fashion the relevant architectural characteristics of WebRTC, such as the signaling path via the calling site(s), the peer-to-peer media path, the signaling code running in the user's browser and the IDP-based identity provisioning in WebRTC.
In their assessment, the authors start from the observation that the calling site might not (always) be trusted. In this security setting, they identify a crucial role for the browser as trusted element in the end-to-end setup. Next to a set of security improvements for the browser environment, the authors also discuss the open problem of securing media that is shown in the browser.

User Identity for WebRTC Services: A Matter of Trust (Victoria Beltran, Emmanuel Bertin, Noël Crespi)

The most classic deployment model of WebRTC is the two-party communication via a shared communication provider (i.e. calling site), and the use of a separate identity provider for the end-user identification.
In "User Identity for WebRTC Services: A Matter of Trust", Victoria Beltran, Emmanuel Bertin and Noël Crespi discuss a variety of trust models, based on this classic deployment, and assess the impact on the user's privacy in each model. In particular, the article examines the use of existing identity provider solutions such as BrowserID, OAuth 2.0 and OpenID Connect in the context of WebRTC.

Who is calling which page on the Web? (Li Li, Wu Chou, Tao Cai, Whihong Qiu)

In "Who is calling which page on the Web?", Li Li, Wu Chou, Tao Cai and Whihong Qiu investigate identity provisioning in more complex deployments, in which multiple calling sites and identity providers are involved.
To support a variety of identity providers, WebRTC describes the use of proxies to interact between the browser environment and the identity provider. In this article, the authors propose and compare several architectural approaches on top of the proxies to support identity provisioning, user lookup and authentication in a uniform way. Moreover, the article suggests the user of a Web-of-Trust identity model to overcome the limitations of hierarchical identity systems.

Experiments with AAA in WebRTC PaaS infrastructures: the case of Kurento (Luis López-Fernández, Micael Gallego, Boni García, David Fernández-López, Francisco Javier López)

To exploit the full potential of the WebRTC technology, also more complex usage scenarios can be realized by deploying additional WebRTC media servers. Media servers can for instance be responsible for transcoding media, recording and persistent archiving, or for setting up group communication.
In "Experiments with AAA in WebRTC PaaS infrastructures: the case of Kurento", Luis López-Fernández, Micael Gallego, Boni García, David Fernández-López and Francisco Javier López set up several experiments to offer these WebRTC media intermediaries as cloud services, and discuss how the WebRTC model can be expanded to accommodate these sophisticated infrastructures. Moreover, the authors evaluate various authorization schemes to grant users access to the media services, without exposing their WebRTC identity.

Outlook

In this issue, we have intentionally focused on the complex problem of identity provisioning in WebRTC based applications, as this is one of the major challenges in the broad adoption this high-potential technology. However, as mentioned before, this is only a subset of emerging security-related topics that require further investigation.

An important aspect in securing the client-side WebRTC environment, is the ability to enforce security policies in the JavaScript execution context. As for now, the JavaScript code that controls the whole WebRTC communication runs in the browser of the end-user (and executes on his behalf), but is at the same time under control of the calling page, and it will most probably include (and delegate control to) third-party libraries. Put together, this opens an interesting vector for various attacks (including injection attacks) while receiving and interpreting data via the signaling or media path, and running code from the calling site and third-party script JavaScript providers.

Another aspect is the overall complexity of the technology: the inherent complexity in setting up the peer-to-peer connections between browsers, and the combination of two rather distinct worlds - each with their own security model: end-to-end networking and JavaScript/Web. In particular, we expect the origin-based security model of the Web to conflict with the connection-based security model of real-time communication. For instance, questions arise as how to setup a connection across different JavaScript origins, or how to manage multiple connections and identities from within a single origin and JavaScript execution context.

Moreover, several technical solutions on the identity provisioning side as well as on the JavaScript permission side propose to enroll the end-user is making security-sensitive decisions (e.g. whether to allow the browser to set up a call with user@idp (once or permanent), whether to share his video stream or desktop, or whether to trust site A in calling external parties, ...). This will undoubtedly expose (some of) the complexity of the technology to the end-user, but even more critical, this also implies that the end-to-end security of WebRTC applications will strongly depend on the judgement call of a novice end-user, potentially “promoting” this end-user to the weakest link of WebRTC.

In conclusion, while being founded on a significant body of security considerations, WebRTC will keep security-minded professionals occupied for the foreseeable future nonetheless. Given the many facets and complexity of the underlying communication and application model, this is hardly surprising. The WebRTC case also clearly illustrates how well the web is maturing: the technological advances open up a completely new area of applications, but (even more important) security is no longer an afterthought - it is already considered early in the standardization process.

Biography

Dr. Lieven Desmet *is a Research Manager on Secure Software within the iMinds-DistriNet research group at the KU Leuven (Belgium). Lieven outlines and implement the research strategy on software security within the university, he coaches junior researchers in (web) application security and participates in dissemination and valorization activities.* Lieven received a Doctor degree in Engineering (Computer Science) from the University of Leuven. Lieven *is board member of the OWASP Chapter Belgium and program director of the yearly SecAppDev training courses on Secure Application Development.* Lieven has been investigating the security of WebRTC as part of the ongoing EU-FP7 project STREWS. Contact him at Lieven.Desmet@cs.kuleuven.be.

Dr. Martin Johns *is a Research Expert in the Security and Trust group within SAP AG, where he leads the Web application security team.* His research interests include software security, static analysis and exploring the Web ecosystem. Martin received a PhD in computer science from the University of Passau. He’s a board member of the Open Web Application Security Project’s German chapter. Within the EU-FP7 project STREWS, Martin is leading of the roadmapping activities for web security research in Europe. Contact him at martin.johns@sap.com.