Browser-to-Browser Authentication and Trust Relationships for WebRTC

Ibrahim Tariq Javed, Khalifa Toumi, Noel Crespi
Institut Mines-Telecom
Telecom SudParis, Evry, France
Email: {Ibrahim_Tariq.Javed, Khalifa.Toumi@telecom-sudparis.eu}, Noel.crespi@mines-telecom.fr

***Abstract*—WebRTC enables browsers to communicate in a Peer to Peer fashion without the use of any plug-ins. This technology is expected to lead a wave of disruptive yet innovative new communication services over the Web. However it also brings significant security and privacy concerns for the users. In WebRTC, authentication is decoupled from the website allowing users to validate each other directly using third party identity providers. User's privacy and security highly depends upon the mechanism used for end-to-end authentication. To achieve security and enhance user privacy it is also essential to define trust between various entities involved in WebRTC security architecture. Therefore, in this paper, we analyze the identity architecture in detail to provide a comparison of suitable authentication protocols. A clear definition of trust is presented by defining various trust relationships that exist in WebRTC identity architecture.**

Keywords*—*WebRTC; P2P; Identity Management; Authentication; Trust.

I. Introduction

Advancements in HTML standards and the introduction of Web Real Time Communications (WebRTC) has allowed browsers to send real time media in a Peer-to-Peer (P2P) manner [1]. WebRTC is independent of any type of browser, platform or device and enables users to communicate with any HTML compatible device. WebRTC is expected to transform the communication landscape over the Web by introducing real-time communication capabilities to any Web page with just a few lines of code.

However, the open source nature of this technology introduces new security and trust requirements presented in [2] and [3], respectively. Amongst the various security challenges introduced in [4], the most crucial one is to have reliable end-to-end authentication between communicating peers using trusted third party Independent Identity Providers (IdP) [5]. Several significant security issues and architectural challenges faced by the IdP based authentication mechanisms are presented in [6]–[8]. However, all of these existing studies analyze Single Sign On (SSO) solutions over Web, which allow users to sign once and have their identities automatically verified by each application. In contrast, WebRTC requires authentication protocols for end-to-end identity provisioning that enables users to directly receive and verify the identity of their communicating remote peers.

Another security issue that is critical to WebRTC is the clear definition of a trust. The rise of browser to browser communication has generated several questions including how to define trust in WebRTC, what are the different trust relationships and how to evaluate trust for each relationship. To the best of our knowledge, there is no study that provides a generic definition of the trust in WebRTC. Several surveys have been conducted on trust management in various emerging fields [9]–[14]. However, the concept of trust in WebRTC arena is yet to be explored.

The first contribution of this paper is to provide a review of suitable Web based IdP mechanisms for the purpose of identity provisioning. This study will help developers to choose the appropriate protocol for their applications as well as prompt researchers to propose new mechanisms adapted to the security and privacy requirements identified in this paper. A comparative study in terms of user privacy and security is conducted by mapping IdP based authentication mechanisms over WebRTC identity architecture. The second contribution of this paper is to provide a wider vision of trust in browser to browser communications by identifying various trust relationships, their objectives and the context and parameters influencing trust.

The rest of this paper is structured as follows: Section II gives a brief introduction of WebRTC identity architecture and Section III explains the process of authentication. Section IV lists several requirements for identity provision in WebRTC. Section V describes suitable Web authentication protocols that can be applied over the identity architecture of WebRTC whereas various trust relationship of WebRTC communications are presented in Section VII and the paper concludes with Section VIII.

II. WebRTC Identity Architecture

WebRTC is an open-source Web application that resides within the browser to exchange media in a P2P fashion. WebRTC identity architecture [15] aims to provide maximum amount of authentication with the minimum possible level of trust in Web Calling Site/Calling Server (CS). The multi domain call model of WebRTC is presented in Figure 1. Each CS is responsible for providing a JavaScript (JS) application that operates over the browser and initiates the PeerConnection component [16]. By calling appropriate JS APIs PeerConnection (PeerC) establishes direct media connection between browsers. CS is also responsible for implementing signaling where Session Description Protocol (SDP) is used to exchange reachable addresses and session parameters.

In order to authenticate a user from IdP, PeerC downloads JS code "IdP proxy" from a specific location defined in the IdP domain. The Browser is responsible for segregating JS codes into sandboxes therefore restricting each script to interact with resources from the same origin. Thus IdP proxy is only able to communicate with its IdP in order to authentication user. In response, IdP generates user Identity Assertion (IA), which

 ISBN: 978-1-61208-505-0

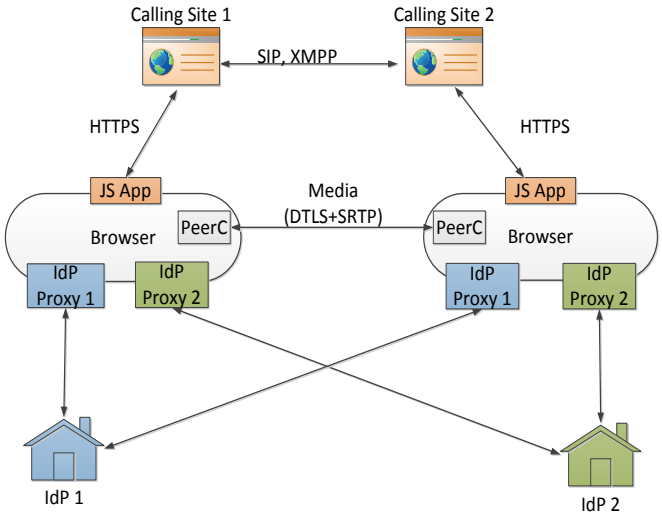

Fig. 1. WebRTC Communication Model

is included in the identity attribute of SDP descriptor message by the browser. The concept of IdP proxy allows the browser to support any type of IdP or authentication protocol as long as it is able to download and run the JS code from IdP.

The browser that establishes user identity by authenticating itself with the IdP is the Authenticating party (AP) whereas the browser which verifies the AP identity from the IdP is called the Relying Party (RP) [15]. In order for communicating parties to authenticate each other, both browsers will act as an AP and as an RP in the process of end-to-end authentication.

III. Authentication in WebRTC

There are two types of authentication that apply to WebRTC communications. First, it is the authentication of the CS/IdP in which the browser validates the ownership of origin by verifying the received digital certificate from the issuing certificate authority. The major drawback is that the browser will trust any certificate that is validated by the trusted issuing authority and has no means of verifying that the certificate truly belongs to the owner. Thus, for WebRTC efficient authentication mechanisms should be introduced to allow browsers to accurately verify that a digital certificate received is the correct certificate used by that website. The second type of authentication is between communicating peers. User identity information in the form of IA are exchanged between peers via the CS and are verified from the same IdP that generated them [17]. There are two major drawbacks of WebRTC identity provision process. Firstly, the IA are sent unencrypted, which allows CS to extract user identity information and track user activities based on identities across communications. Secondly the standard allows CS to force the selection of IdP which does not allow the user to select its own choice of IdP. In order to use the services of CS, user will have to authenticate to CS defined IdP which it may not trust.

The identity provision procedure presented in Figure 2 for end-to-end authentication in WebRTC is explained as follows:

A. *Identity Assertion Generation*

AP PeerC generates request for assertion by attaching fingerprint of DTLS-SRTP certificate. The request also contains the origin of CS which allows IdP to always be aware of the CS the user is using to communicate. IdP proxy is able to access user cookies which allows IdP to check whether the user is already logged in or not. If the user is not authenticated then the IdP proxy returns an error including the URL for entering user credentials. This error is handled by the JS Application or the CS as the IdP proxy is sandboxed and cannot directly demand the user to login. After successful authentication, IdP generates and returns IA. The IA includes user identity information and DTLS fingerprint. The received IA is attached to the SDP message by PeerC and is sent to the remote party via CS.

B. *Identity Assertion Verification*

The RP PeerC extracts the IA from the received SDP message. The domain name of IdP from IA is used to construct the URL in order to downland the IdP proxy. For identity verification, the user is not required to authenticate himself. Upon successful verification the verified IA is returned. PeerC verifies IdP by comparing name-space of received identifier in IA with domain name of IdP. In case of non-authoritative IdP where the name-space of identifier is not the same as domain name, the IdP should be explicitly configured as trusted in browser. Before establishing connection PeerC matches fingerprint in IA with DTLS certificate received over the media channel. This is to ensure that the party establishing peer connection is same as the one which provided the IA.

IV. Requirements for WebRTC Identity Provisioning

We derive a new set of trust requirements based on the weaknesses of identity architecture identified in the previous section. These requirements address the privacy, security and trust concerns raised during end-to-authentication.

(i) ***Identity Unlinkability:*** IdP needs to be able to provide user identity confidentiality against CS.
(ii) ***Identity Encryption:*** IdP needs to be able to provide encryption to user IA.
(iii) ***IdP Selection:*** User needs to select its own choice of IdP without being forced by CS.
(iv) ***CS Unlinkability:*** User needs to be able to hide the information about origin of CS from IdP.
(v) ***Identity Information Control:*** User needs to be able to select the identity information included in IA.
(vi) ***Certificate Verification:*** User needs to be able to verify that the digital certificate provided by CS for authentication is the correct certificate used by it.
(vii) ***User Anonymity by IdP:*** User needs to be able to acquire anonymous identity from IdP.

 ISBN: 978-1-61208-505-0

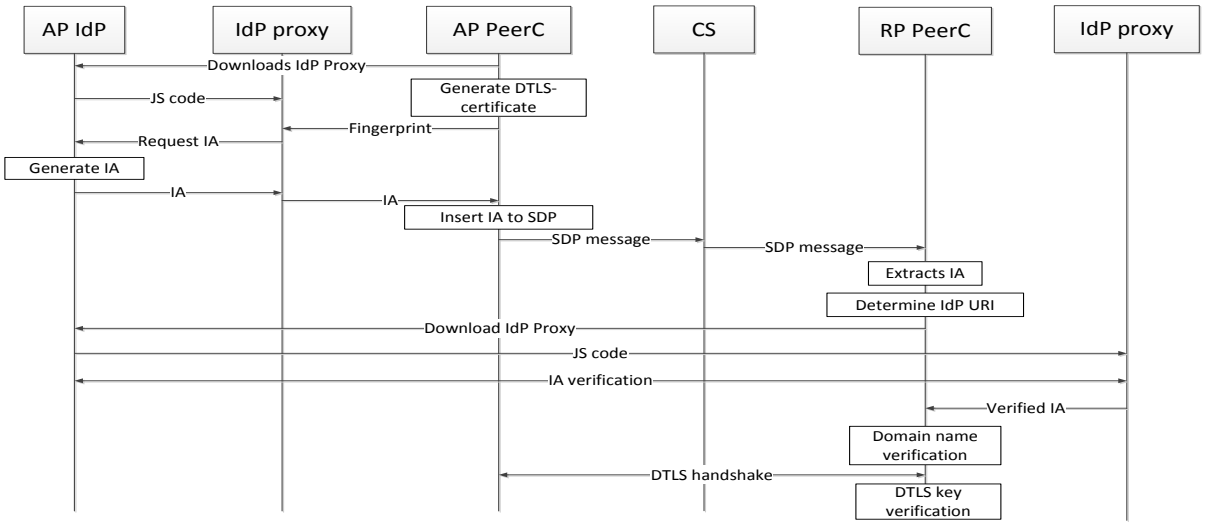

Fig. 2. End-to-End Authentication Flow Diagram

(viii) ***Privacy awareness:*** IdP needs to inform user about how privacy will be handled during P2P authentication.

In order to fulfill these requirements, new solutions/modifications to the architecture and procedures of WebRTC should be proposed.

V. Authentication Protocols for Identity Provisioning

WebRTC proposes the use of existing SSO [18] protocols which are designed for client server login. However using these protocols for WebRTC in order to have P2P authentication may require certain modifications. The use of IdP proxy makes WebRTC to be protocol independent. However the selection of a particular authentication protocol will profoundly affect the overall security and privacy of WebRTC communication. We tried to compare the two mechanisms, BrowserID and OAuth2.0 recommended by RTCWeb working group [19] by mapping them over the WebRTC identity architecture [15]. The third protocol applied is OpenID connect (OIDC) which constitutes a set of extensions on top of OAuth for the purpose of authentication.

A. BrowserID

BrowserID allows any website to receive assertion of email address ownership from the user [20]. The website is the RP whereas the browser is considered to be the client. In BrowserID specifications [21], client send Backed Identity Assertion (BIA) to the RP. The BIA is combination of User Certificate (UC) and IA. UC carries user email address and user public key, which is digitally signed by the IdP to certify the ownership of the email address and the public key of the user. Whereas IA contains the request to login into specific RP is signed by the user private key as shown in Figure 3. However when applied to WebRTC instead of the website browsers will authenticate each other. BrowserID can be mapped to WebRTC architecture as follows:

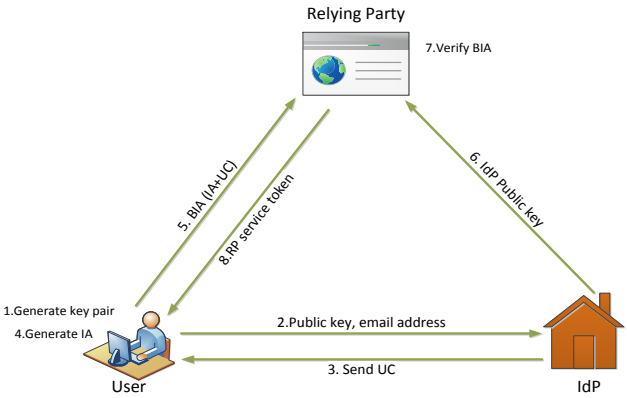

Fig. 3. BrowserID Authentication Overview

Identity Assertion Generation: A public private key pair is generated by the AP browser for asymmetric encryption. PeerC downloads the IdP proxy and requests the IdP to generate UC by including the user public key. After valid authentication of the user, IdP generates UC by signing the user identity (public email address) and public key. The PeerC generates the DTLS-SRTP key for establishing the media connection and sends the fingerprint to IdP proxy. The IA is generated by IdP proxy by signing the fingerprint with user private key. It should be noted that in BrowserID browser is not required to send the fingerprint to IdP as it generates the final assertion. Lastly the PeerC generates the final assertion BIA and includes it into the identity attribute of the SDP.

Identity Assertion Verification: The RP PeerC receives the SDP message and extracts IA and UC. The domain name is used to download IdP proxy to request the public key from the IdP. PeerC performs two checks, firstly it matches the received public key with the signature inside UC, secondly it verifies

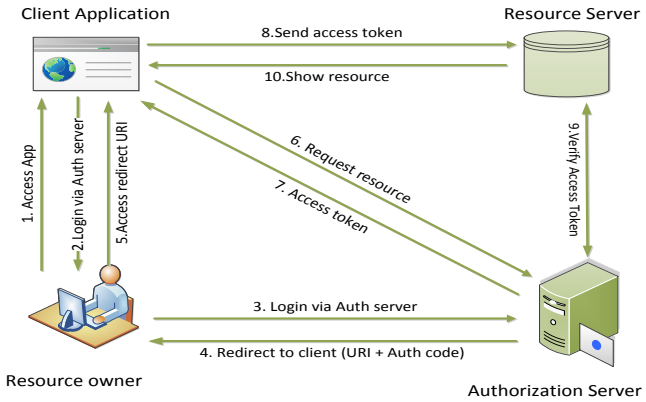

Fig. 4. OAuth Authorization Overview

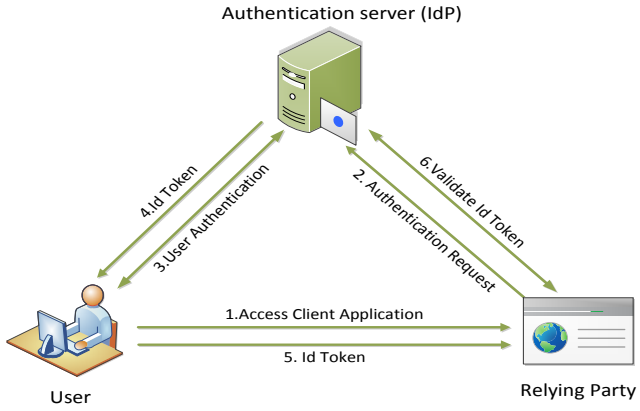

Fig. 5. OIDC Authentication Overview

user public key inside UC with the signature inside IA.

B. OAuth 2.0

OAuth 2.0 [22] being an authorization protocol allows client applications to access resources hosted on a Resource Server (RS) owned by Resource Owner (RO) as shown in Figure 4. The authorization to access resource is provided by the Authorization Server (AS) on behalf of the RO. However before accessing the resource client has to register with AS using the clientid [23]. The process of authorization is described in Figure 4.

To map OAuth protocol onto WebRTC architecture the client application can be considered as RP browser, the RO as the AP browser whereas the IdP acts as AS as well as RS.

Identity Assertion Generation: The AP PeerC using the IdP proxy authenticates the user to the IdP and registers an identity resource with IdP including the fingerprint of DTLS certificate. The IdP in return sends the IA which contains the authorization code. PeerC attaches the authorization code to the SDP and sends it to RP browser via the CS.

Identity Assertion Verification: The RP PeerC receives the SDP message and extracts the authorization code from the identity attribute of SDP. The IdP proxy sends the authorization code and receives the access token from the IdP. Upon receiving the access token the IdP proxy retrieves the identity and fingerprint. The RP PeerC verifies the fingerprint with DTLS certificate received over the media channel.

C. OpenID Connect

OIDC adds an identity layer on top of the OAuth 2.0 protocol. It enables client to verify the identity of user based on the ID Token [24] that contains claims about the user authentication. The ID Token incorporates user (AP) identity information, the IdP identifier and the audience (RP) for which the token is intended for. The ID token is signed by the IdP and can optionally be encrypted. The authentication procedure for OIDC implicit flow is presented in Figure 5.The OIDC protocol can be applied to WebRTC identity architecture as follows:

Identity Assertion Generation: AP PeerC sends authentication request to IdP containing fingerprint and the audience (RP identity) to which the ID Token is intended for. The request may also indicate the type of identity information to be returned in the ID Token. The ID Token is generated and signed by the IdP which contains AP identity information, fingerprint, RP identity and the IdP identifier. PeerC includes the ID Token to the SDP and sends it to the RP.

Identity Assertion Verification: The RP PeerC receives the SDP message and downloads IdP proxy by using identifier domain name. It also extracts the ID Token and fingerprint from the identity attribute. The IdP proxy requests the IdP to validate the ID Token. IdP verifying the signature and returns the verified Identity to the RP. The RP PeerC then verifies the fingerprint with DTLS certificate received over the media channel.

D. Comparison in terms of User Privacy

In WebRTC, user privacy mainly deals with protection of user identity and associated profile information. Table 1 provides a quick comparison of authentication protocols in terms of user privacy properties [25] and features defined as follows:

(i) *Identity Verification*: User ability to verify the identity of remote party.
(ii) *Anonymity*: The inability of remote party and CS to learn user identity.
(iii) *Unlinkability from CS*: The inability of CS to track user activities based on user identities.
(iv) *Unlinkability from IdP*: The inability of IdP to track user activities across different CS.
(v) *Pseudonymity*: The ability of IdP to provide user with psedunonyms as anonymous identities.
(vi) *Identity Encryption*: The ability of IdP to encrypt user identity to achieve confidentiality from CP.
(vii) *Browser Centric*: The ability of browser to generate the identity assertion
(viii) *Information Control*: The ability of user to control the type of information IdP includes in the IA.

 ISBN: 978-1-61208-505-0

TABLE I: Comparison of Authentication Protocols for WebRTC

Authentication Protocols	Identity Verification	Anonymity	Unlinkability from CS	Unlinkability from IdP	Pseudonymity	Identity Encryption	Browser Centric	Information Control	Audience Verification	IdP Centric
BrowserID	✓			✓			✓			
OAuth2.0	✓	✓		✓	✓					✓
OIDC	✓	✓	✓	✓	✓	✓		✓	✓	✓

(ix) *Audience Verification*: The ability of IdP to disclose identity information exclusively to the person which it was intended for.

(x) *IdP Centric*: The ability of user to enforce rules and policies through a trustworthy IdP.

Browser centric approach of BrowserID makes it the simplest protocol that can be applied to WebRTC architecture for identity provisioning. When compared with OAuth and OIDC the considerable drawback of this protocol is the adoption of public email address as user identity. Firstly, it does not allow user to stay anonymous/unidentifiable during the communication. Secondly unlinkability from CS can never be achieved as public email address will always allow it track user activities. When having BrowserID for authentication users will have to trust their CS with their identity information.

However the fact that final IA is generated by the browser without the need of sending DTLS fingerprints to IdP makes BrowserID more reliable in case of distrusted IdP. In contrast to BrowserID protocol, OAuth and OIDC operate in an IdP centric format where IdP is responsible for generating and verifying the IAs. IdP centric nature will allow users to enforce policies and rules through their IdPs. Other than this Anonymity in both these protocols can easily be achieved by the user of pseudonyms.

In OAuth protocol, the redirection between browsers is impossible to achieve as browsers do not have the capability to accept HTTP connection from other browsers. Thus when using OAuth for P2P authentication AP browser is never aware of who is accessing its identity resource whereas IdP is unable to verify that RP has the authority to access AP identity. This brings about serious security concerns for WebRTC communication as any unauthorized party having access to authorization code will be able to obtain user identity information.

OIDC seems to be a better candidate than OAuth in terms of identity confidentially and unlinkability. The feature of encryption and audience in the ID Token does not allow any unauthorized party such as Man-in the middle or CS to obtain user identity information. The audience field in OIDC allows the AP to specify the identity of RP to which the information is intended for. This requires AP to be aware of RP identity before P2P authentication which may be communicated though the CS. Lastly OIDC gives user much more control over their identity information to be shared by indicating it in the authentication request.

VI. Trust Management Objectives

The choice of authentication protocol presented in Section V will further influence the extent to which entities are required to trust each other. For example in the case of BrowserID, user will have to put more trust in their CS as it will be able to track user activities based on its identities. In WebRTC, trust needs to be computed for each entity and displayed to user before connection is established. For now trust in WebRTC is based on valid authentication. However merely authenticating an entity never guarantees that the entity is trusted. Similarly unauthenticated entity does not imply that it may never be trusted for a particular task.

For the wide adaptation of WebRTC technology an efficient trust management framework is essential. To ensure trustworthiness in whole communication system a holistic trust management framework is required with the following objectives:

1) Trust Information Collection: Trust framework that is able to gather and combine information to evaluate trust. Appropriate information collection models are required based on the parameters influencing trust in each relationship.
2) Trust Evaluation: Trust models that are able to compute trust based on the context and parameters influencing trust. These models should be dynamic in nature in order to commute trust variations over time.
3) Privacy Preservation: Privacy enhanced trust models to measure the degree of confidence that a user can have in terms of preserving their privacy. User information should be preserved according to the expectations of each user.
4) Quality of Service: Trust management should ensure that communication and authentication services are offered at exactly the right place and time to the right person.
5) Human-Computer Interaction: Users should be able to interact with the browser in a secure and efficient manner in order to set their communication preferences.
6) Identity Trust: A scalable and efficient identity management system capable of authenticating each entity in a credible and verifiable manner.

VII. Trust Relationships

We define trust as a relation between two entities, a truster and a trustee. The entity that trusts the target entity is known as the truster whereas entity that is being trusted is the

 ISBN: 978-1-61208-505-0

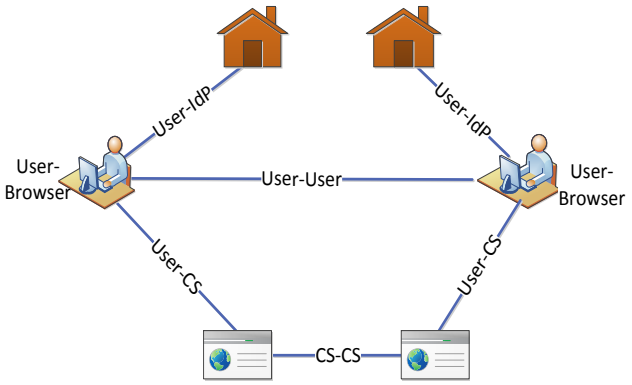

Fig. 6. Trust Relationships in WebRTC

trustee. Thus any trust relationship is described as truster trusting trustee at a given time for a particular context. Trust relationships can be expressed in terms of a trust vector wherein each element of the vector represents a parameter that contributes towards the trust value:

$$(Truster \xrightarrow{c} Trustee)_t = [K_{tr}^{te}, E_{tr}^{te}, R_{tr}^{te}, A_{tr}^{te}] \quad (1)$$

Where context 'c' is the information which characterizes the situation of the entities involved. The context of trust can be expressed as the combination of trust purpose and trustee aspects. Therefore a truster will always trust the ability of a trustee to perform a specific action with regards to certain trustee characteristic. For example, truster A trusts trustee B's security to access a resource that the truster controls. Time 't' is used to characterize the dynamic behavior of a trust relationship whereas the parameters used for evaluating trust are described as follows:

- Knowledge parameter K_{tr}^{te} is based on truster's awareness about the abilities of trustee in a specific context.
- Experience parameter E_{tr}^{te} is the cumulative effect of previous interactions between a trustee and a truster in a particular context and over a specified period of time.
- Reputation parameter R_{tr}^{te} is the sum of recommender's judgment about trustee in relation to the truster in a specific context. Each recommendation is weighted by the truster trust in recommender within that context.
- Authentication parameter A_{tr}^{te} defines the strength in the authentication process. The server identity will be verified form the certificate authority whereas the communication participant identity will be verified from the IdP. Therefore this parameter will be weighted with the amount of trust in IdP or the certificate authority.

We identify four trust relationship that exist in WebRTC: User-CS, User-User, User-IdP, CS-CS and User-Browser represented in Figure 6.

A. User-IdP Trust Relationship

IdP provides user the functionality of storing and managing their identity information while allowing them to authenticate. The purpose of trust in User-IdP relationship is to trust an IdP's ability to provide authentication services while considering its ability to preserve privacy. We provide a set of IdP's attributes that should be considered while evaluating trust in IdP:

- *Privacy Protection*: An IdP's ability to provide identity confidentiality in terms of unlinkability from CS;
- *Anonymity*: An IdP's ability to provide anonymity by means of pseudonyms;
- *Encryption*: An IdP ability to provide encrypted IA;
- *Authentication Mechanism*: The type of authentication protocol being used by IdP; and
- *IdP Type*: If an IdP is authoritative or non-authoritative.

B. User-CS Trust Relationship

CS provides JS application that allows browser to communicate in a P2P fashion. CS is also responsible for implementing signaling for the exchange of session parameters, identities, call answer/offer request and user reachable addresses between communicating parties. The purpose of trust in a User-CS relationship reflects the CS ability to provide communication services whereas CS aspects that needs to be considered are security and reliability. For the evaluation of trust in CS the following should be considered:

- *Malware Detection*: Undesirable software installations by a CS;
- *Software Vulnerabilities*: Weaknesses detected in the JS code provided by CS;
- *Attack Detection*: The CS's ability to detect and prevent attacks;
- *Mixed Content*: The content loaded from an HTTP origin onto the HTTPS page of a CS; and
- *IdP Selection*: The IdP selection enforced by the CS.

C. User-User Trust Relationship

Before an exchange of real time media, each user needs to verify the identity of its communicating participant. The purpose of trust is user identification. Subjective aspects are considered such as user's honesty, accuracy and integrity while receiving the identity information. We present set of attributes that should be considered for the evaluation of trust in the received identity assertion:

- *Identity Proofing*: How strongly the identity information of user has been verified and vetted by the IdP;
- *Credential Verification*: How easily a user's credential can be spoofed or stolen;
- *Assertion Strength*: Proof that the identity was actually asserted by IdP for a given transaction;
- *Encryption*: If received IA is encrypted or not;
- *Audience Protection*: The received IA contains the audience identity for which the assertion is intended; and
- *Anonymity*: The use of an anonymous identity by the communicating participant.

 ISBN: 978-1-61208-505-0

D. CS-CS Trust Relationship

Several efforts are being made to achieve cross domain interoperability [26], [27], [28] where users from different domains are allowed to contact and communicate with each other. Each CS will be responsible for sharing availability status, identity information and the reachable addresses of users from different domains. Trust between CS needs to be computed and displayed to the subscribers of each CS before they decide to interact with the users of another domain. The purpose of trust in this relationship is to achieve interoperability whereas the aspects of a CS that need to be considered for trust assessment are its security and reliability.

E. User-Browser Trust Relationship

The browser is responsible for running JS codes in isolated sandboxes to connect with various entities on behalf of a user. The identity of each entity is verified by the browser before communication takes place. The overall security of WebRTC communication is highly dependent upon the selection of a trustworthy browser. Security in WebRTC can never be achieved if a browser is compromised. Therefore, it is essential for users to select trustworthy browsers that can be relied upon completely. However trust between browser and user is subjective and may not be computed.

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 645342, project reTHINK.

VIII. CONCLUSION

WebRTC technology is envisioned to lead innovative ways to share information and communicate over Web. The vast adaptation of this highly potential P2P communication technology requires an efficient identity and trust management framework. This paper aims (1) to analyze the end-to-end authentication procedure between browsers and (2) to give a clear definition of trust in WebRTC.

A study of WebRTC identity architecture and authentication mechanisms suitable for the purpose of end-to-end user identification is conducted. To address the security and privacy concerns identified in this paper we prompt the community to develop mechanisms particularly suitable for WebRTC communications. In order to have a wider vision of trust, we define various trust relationship, the context of trust and parameters influencing trust computation for browser to browser communication.

REFERENCES

[1] S. Loreto and S. P. Romano, "Real-time communications in the web: Issues, achievements, and ongoing standardization efforts," *IEEE Internet Computing*, vol. 16, no. 5, pp. 68–73, Sept 2012.

[2] E. Rescorla, "Security Considerations for WebRTC," Internet-Draft, Tech. Rep., February 2015.

[3] V. Beltran, E. Bertin, and S. Cazeaux, "Additional Use-cases and Requirements for WebRTC Identity Architecture," Internet-Draft, Tech. Rep., March 2015.

[4] R. L. Barnes and M. Thomson, "Browser-to-browser security assurances for webrtc," *IEEE Internet Computing*, vol. 18, no. 6, pp. 11–17, Nov 2014.

[5] A. Vapen, N. Carlsson, A. Mahanti, and N. Shahmehri, "A look at the third-party identity management landscape," *IEEE Internet Computing*, vol. 20, no. 2, pp. 18–25, Mar 2016.

[6] J. Torres, M. Nogueira, and G. Pujolle, "A survey on identity management for the future network," *IEEE Communications Surveys Tutorials*, vol. 15, no. 2, pp. 787–802, Second 2013.

[7] E. Maler and D. Reed, "The venn of identity: Options and issues in federated identity management," *IEEE Security and Privacy*, vol. 6, no. 2, pp. 16–23, 2008.

[8] E. Ghazizadeh, M. Zamani, J. l. Ab Manan, and A. Pashang, "A survey on security issues of federated identity in the cloud computing," in *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, Dec 2012, pp. 532–565.

[9] J. H. Cho, A. Swami, and I. R. Chen, "A survey on trust management for mobile ad hoc networks," *IEEE Communications Surveys Tutorials*, vol. 13, no. 4, pp. 562–583, Fourth 2011.

[10] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for internet of things," *Journal of Network and Computer Applications*, vol. 42, pp. 120 – 134, 2014.

[11] W. Sherchan, S. Nepal, and C. Paris, "A survey of trust in social networks," *ACM Comput. Surv.*, vol. 45, no. 4, pp. 47:1–47:33, Aug. 2013.

[12] A. Jsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618 – 644, 2007, emerging Issues in Collaborative Commerce.

[13] S. Marti and H. Garcia-Molina, "Taxonomy of trust: Categorizing p2p reputation systems," *Comput. Netw.*, vol. 50, no. 4, pp. 472–484, Mar. 2006.

[14] L. Margaret and M. Jeffrey, "Machine to machine trusted behaviors," in *The Eighth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM)*, Aug 2014.

[15] E. Rescorla, "WebRTC Security Architecture," IETF Internet Draft, Standards Track, June 2016.

[16] A. Bergkvist, D. C. Burnett, C. Jennings, A. Narayanan, and B. Aboba, "WebRTC 1.0: Real-time Communication Between Browsers," W3C Working Draft, Tech. Rep., May 2016.

[17] V. Beltran and E. Bertin, "Unified communications as a service and webrtc: An identity-centric perspective," *Computer Communications*, vol. 68, pp. 73 – 82, 2015.

[18] A. Pashalidis and C. J. Mitchell, "A taxonomy of single sign-on systems," in *Information security and privacy*. Springer, 2003, pp. 249–264.

[19] Web working group. [Online]. Available: http://tools.ietf.org/wg/rtcweb/charters

[20] D. Fett, R. Ksters, and G. Schmitz, "An expressive model for the web infrastructure: Definition and application to the browser id sso system," in *2014 IEEE Symposium on Security and Privacy*, May 2014, pp. 673–688.

[21] Browserid. [Online]. Available: https://github.com/mozilla/id-specs/blob/prod/browserid/index.md

[22] D. Hardt, "The OAuth 2.0 Authorization Framework," IETF RFC: 6749, Standards Track, Tech. Rep.

[23] B. Leiba, "Oauth web authorization protocol," *IEEE Internet Computing*, vol. 16, no. 1, pp. 74–77, Jan 2012.

[24] J. Bradley, B. de Medeir, and C. Mortimore, "Openid connect core 1.0," *The OpenID Foundation*.

[25] A. Pfitzmann and H. Tschofenig, "Terminology for Talking about Privacy by Data Minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management," Internet-Draft, Tech. Rep., July 2010.

[26] S. Becot, E. Bertin, J. M. Crom, V. Frey, and S. Tuffin, "Communication services in the web era: How can telco join the ott hangout?" in *18th conference on Innovations in Clouds, Internet and Networks (ICIN)*, Feb 2015, pp. 208–215.

[27] L. Li, W. Chou, Z. Qiu, and T. Cai, "Who is calling which page on the web?" *IEEE Internet Computing*, vol. 18, no. 6, pp. 26–33, Nov 2014.

[28] I. Javed and et al., "Global identity and reachability framework for interoperable p2p communication services," in *19th conference on Innovations in Clouds, Internet and Networks (ICIN 2016)*, March 2016.

 ISBN: 978-1-61208-505-0