Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 7, 2016

M. Westerlund
J. Mattsson
Ericsson

July 6, 2015

WebRTC Use Case and Framework for Privacy Enhanced RTP Conferencing (PERC)
draft-westerlund-perc-webrtc-use-case-00

Abstract

The work so far on Privacy Enhanced RTP Conferencing, which allows end-to-end security also in centralized switch RTP based conference, has not considered WebRTC in detail. This document looks at the use case of WebRTC based endpoints, it also considers implications of using external providers for both conference applications and centralized media distribution devices. From this a number of challenges have been determined, and requirements are derived from these. Finally the draft presents some straw man for possible solutions.

Table of Contents

1. Introduction

This document discusses the implications on PERC WG's work on providing end-to-end secure centralized RTP conferencing using WebRTC browsers as endpoints. The WebRTC environment contains a number of challenges that needs to be considered; these may affect how the final solution is designed. The authors have also have a strong interest in enabling usages where significant amount of sourcing of external resources are possible to perform. Not only the media distribution devices (MDD) and STUN/TURN resources, but also the core functionalities of the conference application, such as the find and connect to establish the conference. However, the control over the end-to-end security needs to be possible to maintain within a single organization. This organization needs to maintain control over both who is authorized to participate in a particular conference, as well as having control over the end-to-end keys used in that conference.

It needs to be stressed that the use case presented here is far from the only one where WebRTC endpoints could be used to establish a multiparty end-to-end secured conference. The authors have chose to focus on use case that combines WebRTC endpoints, contextual communication and outsourcing, a use case suitable for a number of enterprises, businesses, and services.

Section 2 goes through a possible use case and its high level motivation. Section 3 discusses the different trust domains as well as the entities that are considered in this use case. Section 4 discusses a number of challenges where several are unique to WebRTC compared to more native implementations of endpoints. Section 5 derives a number of requirements. Finally in Section 6 we present a straw man solution to some of the challenges we raised.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. The Use Case

This section discusses the representative use case in more detail as well as discussing relevant background information for this use case.

A use case (Enterprise Real-time contextual communication) is that an enterprise has the need for multi-party real-time conferencing with audio and video that is combined with enterprise's internal data that can be viewed and manipulated using a web application. The conferencing is intended to allow multiple employees or external consultants to discuss the data and manipulate its content as well as

present during the discussion, i.e. a form of contextual communication.

The already existing web application to view and manipulate the internal data is desirable to be able to re-use and the conference participants are already using a web browser for this purpose. Thus, basing the solution on WebRTC appears logical as that will enable integration of the Real-time communication (RTC) conferencing part with the existing web application.

The enterprise has no desire to maintain substantial RTC infrastructure to ensure well working conferencing, and prefers to source the needed services and components from external providers. However, the enterprise has strong interests in maintaining control of the security properties and ensure that its security goals are meet, it may even have legal requirements on its communication.

The enterprise already has existing methods for authenticating their employees, consultants and other external parties that have some access rights to the enterprise's data. It would be highly desirable to be able to re-use the existing user database and authorization verification.

3. Entities and Trust

This section discusses various entities and the trust in these roles.

3.1. Trust Domains

The entities belong to three different trust domains:

Trusted: Entities in the trusted domain are fully trusted to perform the role and actions put on them. They may have access to unprotected content and keying material used to protect content end-to-end.

Semi-trusted: Semi-trusted entities have no access to confidential material such as the content and the keying material used to protect content end-to-end. They are however trusted to perform basic operations for selective forwarding of content as well as session establishment.

Untrusted: Entities and functions in this domain are not trusted by the other entities participating in the conference or system. These entities are capable of preventing the conference from functioning, however.

3.2. Trusted Entities

The trusted entities that we consider in the use case are:

The Service Provider: The service provider is the organization that establishes the requirement for the end-to-end security in its RTC conference. It also has the control over entities that provide functionalities in the trusted domain for enabling the conference, such as the Key-Management Function and the Core Application. This represents the enterprise in the above use case description.

Conference Initiator: A human that schedules a conference. The conference contains a number of invited conference participants. These and only these participants shall be able to join this conference.

Conference Participant: A conference participant is a human or possibly another type of entity, like a conference room that has one or more identities that has been invited into the conference.

User Agent: The user agent is the WebRTC enabled endpoint that the conference participant uses to join the conference. In this document this is assumed to be a browser.

Core Application: The web application residing both in the client's server as well as the web based application in the User Agent. Should be able to delegate almost everything to the Conference Application. Non RTC content part of a contextual application is generally handled by this application.

Key Management Function: The function within the service provider that generates, stores and distributes the keys giving access to the conference to User Agents that can provide identity assertions that it can be verified and matches the invited participants.

Authorization Module: A sub-function of the Key Management Function (KMF) that verifies identity assertions according to a particular authorization method.

Conference Session Database: A sub-system of the KMF that contains information about scheduled conferences and which identities that have been invited by the Conference Initiator.

3.3. Semi-trusted Entities

The semi-trusted entities that we consider in the use case are:

Conference Provider: Provides the conferencing service, potentially from within a separate administrative domain than the service provider. This service contains find and connect functions to establish the conference, media distribution devices to selectively forward protected media content to the conference participants as well as support functions for media path establishment such as STUN and TURN services.

Conference Application: The web application residing both in the conference provider's servers as well in the User Agent that performs find and connect and other functions to establish and maintain the conference.

Media Distribution Devices: A device that performs switching and forwarding of protected media to the various conference participants' user agents or to another cascading media distribution device. The switching is performed based on meta information about the content that has been provided by the user agent. It applies security policies that prevent forwarding of traffic not originating with a user agent that is an invited participant. This protection is through a hop-by-hop security mechanism that provides integrity and source authentication as well as confidentiality of the metadata.

STUN/TURN: Support functions that help establish the transport path used to send media between a user agent and the media distribution device.

4. Challenges

This section discusses a number of challenges in meeting the goals as discussed in [I-D.jones-perc-private-media-reqts].

4.1. Enable Delegation

As described above, the Service Provider delegates the communication service to the Conference Provider. The Conference Provider may in turn delegate functions like the media distribution device and STUN/TURN services by sourcing them from other providers. Further, the infrastructure (servers and network) that these functions are run on, can also be sourced from external providers. This puts even higher demands on control and the ability to verify other entities actions from the perspective of the Service Provider.

The main security goal of providing end-to-end confidentiality across a centralized conferencing infrastructure is the main enabler of delegation, as the required trust in large part of the infrastructure

are significantly reduced by freeing them from handle any content as plain-text. However, that is not sufficient as not only the content handling needs to be limited to only the entities that are required to handle it. Also the key-management and authorization parts of the solution need to consider how they can limit the trust. For example the find and connect service is a semi-trusted part as it needs to be capable of establishing the connectivity with the right entities. However, the key-management and authorization system is the one that verifies the participants and their right to participate in a particular conference, and first then provides that participant's endpoint with the necessary secrets.

The system design needs to minimize the privacy sensitive information a particular functions needs. Thus, enabling as much functionality as possible to be outside of the trusted domain. Important functions in the semi-trusted domain, when so necessary to ensure secure operation of the system, must be verified by trusted entities.

4.2. Dealing with JavaScript related Attacks

The application, such as JavaScript application, running in the browser is a potential attack vector. Using various attacks, including cross-site scripting, the application can be compromised and perform the actions an attacker dictates. Even if the application running in the browser is malicious it must not be able to compromise the security of the conference, only perform denial of service attacks such as preventing the user from joining the conference.

A compromised application must be prevented from getting access to content. This will most likely mean that when using the end-to-end confidentiality, corresponding measures to prevent forwarding (of plain text content), access to raw data through APIs etc. that the media confidentiality mode defined in [I-D.ietf-rtcweb-alpn] have to be applied.

The compromised application will get access to who the peer(s) are in the conference. This is unpreventable as the application is the responsible for establishing the communication legs that is creating the conference. An attacker will also be able to use a compromised entity to forward protected content to a destination of its choice.

4.3. Forcing End-to-End Security

The Service Provider needs a method to ensure that when the conference provider application launches an RTCPeerConnections, they are forced to use end-to-end security with the keys provided by the KMF the Service Provider designates, and not the normal hop-by-hop

security only. Thus, the service provider needs a way of applying policies on an web application context. Policies that are inherited by any child contexts and which can't be modified by the application in the user agent.

4.4. Restricting Usage of the e2e Keying Material

While the RTCPeerConnection must use end-to-end security with the key provided by the Service Provider, neither the core or conference applications must be able to extract the key or even use the e2e key material for anything other than encrypted key transport (EKT). The user-agent will be required to have a secure key-store for the duration the key-material is present at the user-agent and valid. When the validity of the key-material expires the key-material needs to be disposed of to reduce the risk of retrospective attacks.

4.5. Enabling Flexible Authentication

The authorization methods should be flexible and enable different types of authorization back-ends. It is desirable that the method for authorization does not need to be implemented as part of the user agent. Requiring user agent modifications makes deployment of new authorization method cumbersome and difficult and open up for down grade attacks due to need for backwards compatibility support.

4.6. Binding Authorization to Endpoint

As the authorization will be used to retrieve the group key used to secure the RTP session end-to-end, it is important that the authorization is bound to the device and user agent where the user gave the authorization. Otherwise the conference provider would be able to move the authorization credentials to another endpoint, use that endpoint to retrieve the key and export it from that endpoint.

4.7. Securing Group Membership Changes

During an ongoing conference the set of participants participating in a conference will vary. In some usages it is important that joining participant can not use the received keying material to perform a retrospective attack and decrypt the content of the conference from a point prior to the participant joining. Nor should the participant after having left the conference be able continue to decrypt the content.

The known solution to this issue is to switch keys, both group key as well as the transport keys each endpoint uses to protect its streams. This puts certain requirements on key-management system. First the key-management system must track the current set of participants and

on changes initiate the change of the group key. This results in a second requirement that they key distribution method for the group key can handle asynchronous distribution events in the KMF to endpoint direction. Thirdly the transport key switching and distribution needs to handle non-synchronized switching by the different endpoints to the new group key.

A clear issue is how the KMF can ensure that a participant that is leaving is correctly accounted for and the key change happens in a timely fashion after the user left. First of all users may leave the conference abruptly due to severed communication or an endpoint that crashes. Secondly, the conference management application is only semi-trusted. The design will have to make choices on how to balance protocol complexities, resource consumption and achieved security properties.

An additional complexity with this mode of operation is that the conference participant likes to in a secure way know which other participants that currently are part of the conference. This information needs to be timely updated, and the current rooster needs to be authenticated to prevent attacks where participants are fooled to believe a particular participant has left, but is in fact still in the conference.

4.8. Key Revocation

The conference e2e group key is only required to reside on an endpoint for the duration its in use. That use is limited by the duration of the conference. When the conference ends there are no reasons to retain the key on the endpoint. Thus, when the conference ends it is desirable to have the key be revoked and deleted from the endpoint. This should be possible to initiate from the KMF when it learns that the conference has ended.

The user agent should upon the user closing the browsing context where the application runs deleting the keys to prevent their leakage.

5. Requirements

This section lists a number of derived requirements from the above challenges. The requirements are:

a. The Web application running in the User Agent MUST NOT be able to compromise the content confidentiality.

1. Including getting access to media content (raw or unencrypted) in the user agent through API or shared resources.

b. The conference provider’s application (server as well as in the user agent) MUST NOT be able to downgrade the intended security properties and policies established by the service provider and the core application.

c. The key material for the end-to-end protection MUST NOT be possible to extract from any web application.

1. The user agent MUST protect the key-material against extraction by user or other software running on the same device.

2. The key material MUST be bound to the usage its intended to prevent leakage.

3. Upon the termination of the conference or the browsing context containing the application the key material SHALL be deleted

d. Different Authorization methods MUST be supported.

1. It’s preferable that authorization methods can be supported without user agent modifications.

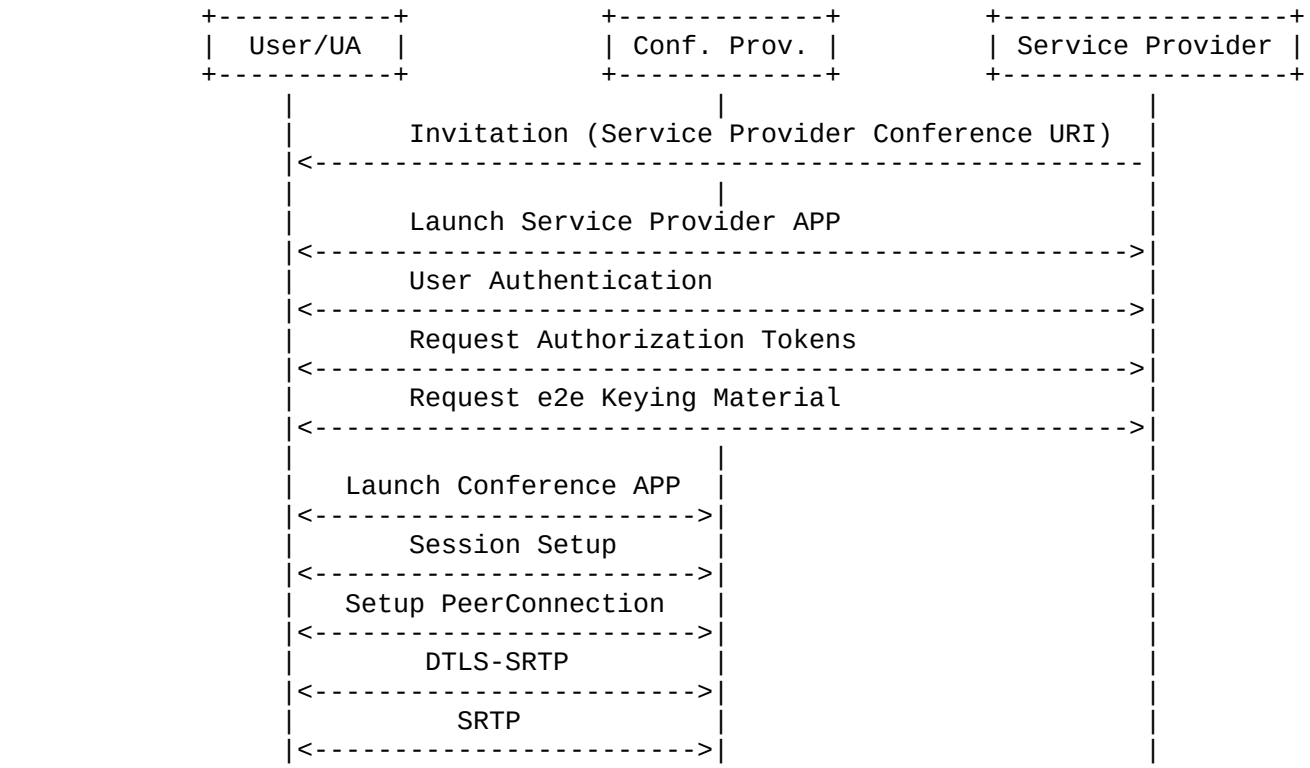
2. The authorization credentials MUST be bound to endpoint where the participant provided its credentials.

e. The design SHOULD support confidentiality where only the current set of participants has access to the media content.

6. Solution Straw-man

6.1. High Level Example Message Flows

The following figure shows a very high level illustration of an example message flow for Privacy Enhanced RTP Conferencing using WebRTC.



The Conference Initiator schedules a conference and invitations are sent out to the conference participants. This could for example be done via e-mail.

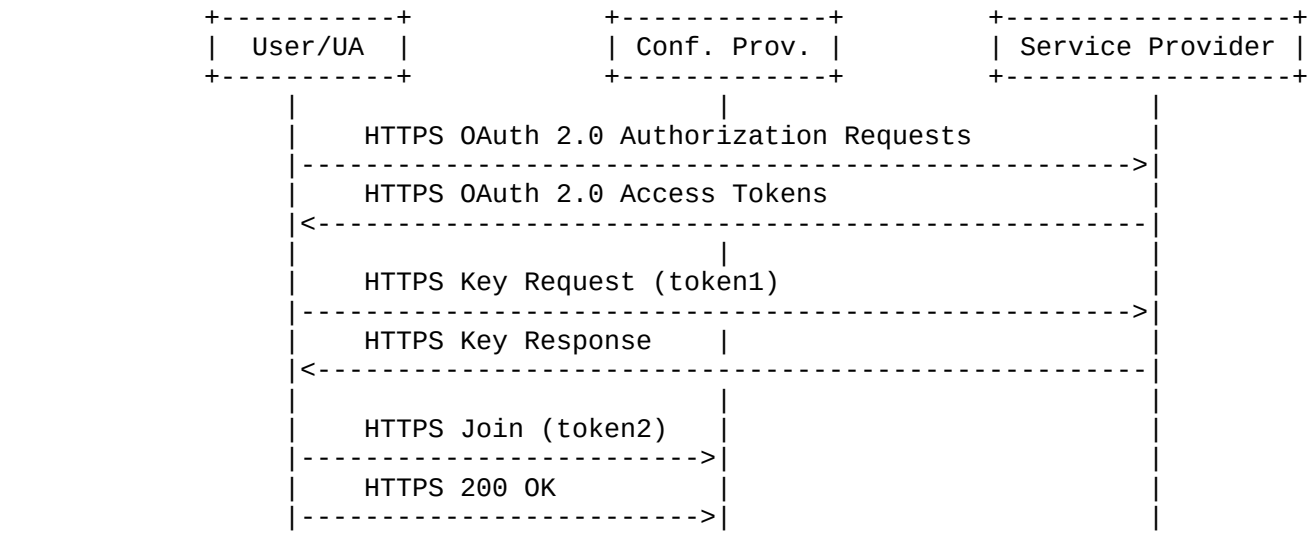
At a later stage, when the conference is about to start, the Conference Participant enters the conference URI in a browser it trusts to launch the core web application. The user then authenticates to the Service Provider Authorization Module (using a authentication method of choice), and downloads the end-to-end keying material from the Service Provider KMF.

The UA then launches the Conference Application, negotiates the session parameters and sets up the PeerConnection. The Conference provider validates that the participant is authorized by the Service Provider to join the conference. The hob-by-hop security is provided by DTLS-SRTP and SRTP (modified to handle end-to-end and hop-by-hop). The UA enforces the use of end-to-end security with the key provided by the Service Provider.

All communication except the invitation and the PeerConnection is to

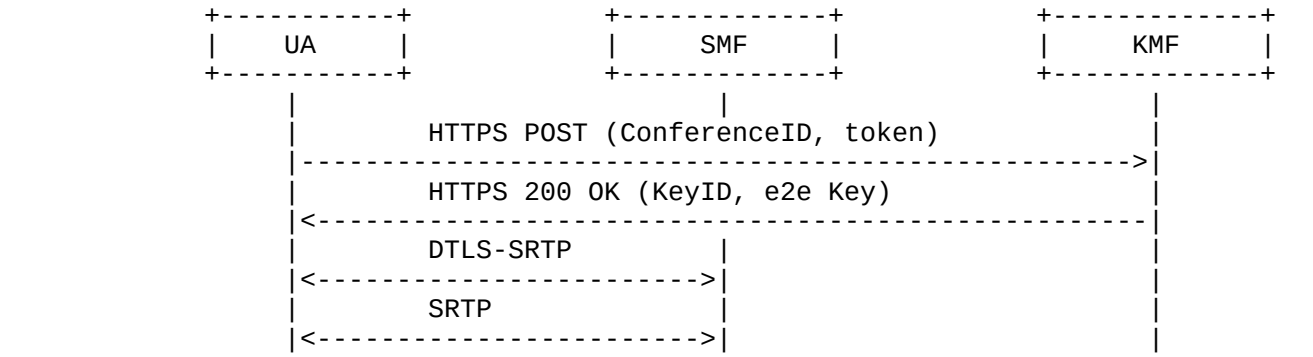
be done over HTTPS.

6.2. Authentication and Authorization



One way to make the end-to-end security solution flexible and enable integration with different types of Service Provider authorization back-ends is to use a general authorization framework such as OAuth 2.0. The User requests access tokens for all the protected resources from the Service Provider Authorization Module. The protected resources can be hosted by the Service Provider (e.g. the Key Management Function) as well as by the Conference provider. The use of OAuth 2.0 allows the same framework to be used in both cases.

6.3. Key Management



The core web application requests the end-to-end keying material from the Service Provider KMF. The successful HTTP response uses the HTTP Encryption-Key header [I-D.thomson-http-encryption] to distribute the end-to-end keying material to the UA. The new parameter "usage" and its value "EKT" instructs the UA that the keying material will be used with SRTP Encrypted Key Transport (EKT). The UA stores the keyid and the keying material for usage as the EKT Key. Key material received with the "usage=EKT" parameter SHALL NOT be extractable and SHALL only be used for EKT. The EKT processing MUST be handled by the UA.

An example successful HTTP response from the KMF is shown below:

```
HTTP/1.1 200 OK
Encryption-Key: keyid="pegh"; key="lupDujHomwIjlutebgharghmey";
usage="EKT"
Content-Length: 0
```

The hob-by-hop keying material is negotiated between the UA and the SMF using DTLS-SRTP.

7. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

8. Security Considerations

The whole document is about making WebRTC based cloud-based conferencing viable and trustworthy from a pervasive monitoring perspective.

9. Acknowledgements

The authors would like to thank Goeran AP Eriksson for challanging discussions and Russ White for valuable comments.

10. Informative References

[I-D.ietf-rtcweb-alpn]
Thomson, M., "Application Layer Protocol Negotiation for Web Real-Time Communications (WebRTC)",

draft-ietf-rtcweb-alpn-01 (work in progress), February 2015.

[I-D.jones-perc-private-media-reqts] Jones, P., Ismail, N., Benham, D., Buckles, N., Mattsson, J., Cheng, Y., and R. Barnes, "Requirements for Private Media in a Switched Conferencing Environment", draft-jones-perc-private-media-reqts-00 (work in progress), July 2015.

[I-D.thomson-http-encryption] Thomson, M., "Encrypted Content-Encoding for HTTP", draft-thomson-http-encryption-01 (work in progress), July 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Authors' Addresses

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

Phone: +46 10 714 82 87
Email: magnus.westerlund@ericsson.com

John Mattsson
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

Phone: +46 10 71 43 501
Email: john.mattsson@ericsson.com