 WebRTC MediaStream Identification in the Session Description Protocol
                    draft-ietf-mmusic-msid-13

Abstract

   This document specifies a Session Description Protocol (SDP) Grouping
   mechanism for RTP media streams that can be used to specify relations
   between media streams.

   This mechanism is used to signal the association between the SDP
   concept of "media description" and the WebRTC concept of
   "MediaStream" / "MediaStreamTrack" using SDP signaling.

   This document is a work item of the MMUSIC WG, whose discussion list
   is mmusic@ietf.org.

Requirements Language

   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].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

1.1. Structure Of This Document

   This document adds a new Session Description Protocol (SDP) [RFC4566]
   mechanism that can associate application layer identifiers with the
   binding between media streams, attaching identifiers to the media
   streams and attaching identifiers to the groupings they form. It is
   designed for use with WebRTC[I-D.ietf-rtcweb-overview] .

   Section 1.2 gives the background on why a new mechanism is needed.

   Section 2 gives the definition of the new mechanism.

   Section 3 gives the necessary semantic information and procedures for
   using the msid attribute to signal the association of
   MediaStreamTracks to MediaStreams in support of the WebRTC API
   [W3C.WD-webrtc-20150210].

1.2. Why A New Mechanism Is Needed

   When media is carried by RTP [RFC3550], each RTP media stream is
   distinguished inside an RTP session by its SSRC; each RTP session is
   distinguished from all other RTP sessions by being on a different
   transport association (strictly speaking, 2 transport associations,
   one used for RTP and one used for RTCP, unless RTP/RTCP multiplexing
   [RFC5761] is used).

   SDP gives a description based on media descriptions. According to
   the model used in [I-D.ietf-rtcweb-jsep], each media description
   describes exactly one media source, and if mulitple media sources are
   carried in an RTP session, this is signalled using BUNDLE
   [I-D.ietf-mmusic-sdp-bundle-negotiation]; if BUNDLE is not used, each
   media source is carried in its own RTP session.

The SDP grouping framework [RFC5888] can be used to group media descriptions. However, for the use case of WebRTC, there is the need for an application to specify some application-level information about the association between the media description and the group. This is not possible using the SDP grouping framework.

1.3. The WEBRTC MediaStream

The W3C WebRTC API specification [W3C.WD-webrtc-20150210] specifies that communication between WebRTC entities is done via MediaStreams, which contain MediaStreamTracks. A MediaStreamTrack is generally carried using a single SSRC in an RTP session (forming an RTP media stream. The collision of terminology is unfortunate.) There might possibly be additional SSRCs, possibly within additional RTP sessions, in order to support functionality like forward error correction or simulcast. This complication is ignored below.

MediaStreamTracks are unidirectional; they carry media on one direction only.

In the RTP specification, media streams are identified using the SSRC field. Streams are grouped into RTP Sessions, and also carry a CNAME. Neither CNAME nor RTP session correspond to a MediaStream. Therefore, the association of an RTP media stream to MediaStreams need to be explicitly signaled.

WebRTC defines a mapping (documented in [I-D.ietf-rtcweb-jsep]) where one SDP media description is used to describe each MediaStreamTrack, and the BUNDLE mechanism [I-D.ietf-mmusic-sdp-bundle-negotiation] is used to group MediaStreamTracks into RTP sessions. Therefore, the need is to specify the ID of a MediaStreamTrack and its associated MediaStream for each media description, which can be accomplished with a media-level SDP attribute.

This usage is described in Section 3.

2. The Msid Mechanism

This document defines a new SDP [RFC4566] media-level "msid" attribute. This new attribute allows endpoints to associate RTP media streams that are described in different media descriptions with the same MediaStreams as defined in [W3C.WD-webrtc-20150210]., and to carry an identifier for each MediaStreamTrack in its "appdata" field.

The value of the "msid" attribute consists of an identifier and an optional "appdata" field.

The name of the attribute is "msid".

The value of the attribute is specified by the following ABNF
[RFC5234] grammar:

```
msid-value = msid-id [ SP msid-appdata ]
msid-id = 1*64token-char ; see RFC 4566
msid-appdata = 1*64token-char  ; see RFC 4566
```

An example msid value for a group with the identifier "examplefoo"
and application data "examplebar" might look like this:

```
msid:examplefoo examplebar
```

The identifier is a string of ASCII characters that are legal in a
"token", consisting of between 1 and 64 characters.

Application data (msid-appdata) is carried on the same line as the
identifier, separated from the identifier by a space.

The identifier (msid-id) uniquely identifies a group within the scope
of an SDP description.

There may be multiple msid attributes in a single media description.
This represents the case where a single MediaStreamTrack is present
in multiple MediaStreams; the value of "msid-appdata" MUST be
identical for all occurences.

Multiple media descriptions with the same value for msid-id and msid-
appdata is not permitted.

Endpoints can update the associations between RTP media streams as
expressed by msid attributes at any time.

The msid attributes depend on the association of RTP media streams
with media descriptions, but does not depend on the association of
RTP media streams with RTP transports; therefore, its mux category is
NORMAL - the process of deciding on MSID attributes doesn't have to
take into consideration whether the media streams are bundled or not.

3. Procedures

This section describes the procedures for associating media
descriptions representing MediaStreamTracks within MediaStreams as
defined in [W3C.WD-webrtc-20150210].

In the Javascript API, each MediaStream and MediaStreamTrack has an
"id" attribute, which is a DOMString.

The value of the "msid-id" field in the msid consists of the "id"
attribute of a MediaStream, as defined in its WebIDL specification.

The value of the "msid-appdata" field in the msid consists of the
"id" attribute of a MediaStreamTrack, as defined in its WebIDL
specification.

When an SDP session description is updated, a specific "msid-id"
continues to refer to the same MediaStream, and a specific "msid-
appdata" to the same MediaStreamTrack. There is no memory apart from
the currently valid SDP descriptions; if an msid "identifier" value
disappears from the SDP and appears in a later negotiation, it will
be taken to refer to a new MediaStream.

The following is a high level description of the rules for handling
SDP updates. Detailed procedures are in Section 3.2.

o When a new msid "identifier" value occurs in a session
   description, the recipient can signal to its application that a
   new MediaStream has been added.

o When a session description is updated to have media descriptions
   with an msid "identifier" value, with one or more different
   "appdata" values, the recipient can signal to its application that
   new MediaStreamTracks have been added to the MediaStream. This is
   done for each different msid "identifier" value.

o When a session description is updated to no longer list any msid
   attribute on a specific media description, the recipient can
   signal to its application that the corresponding MediaStreamTrack
   has ended.

In addition to signaling that the track is closed when its msid
attribute disappears from the SDP, the track will also be signaled as
being closed when all associated SSRCs have disappeared by the rules
of [RFC3550] section 6.3.4 (BYE packet received) and 6.3.5 (timeout),
and when the corresponding media description is disabled by setting
the port number to zero. Changing the direction of the media
description (by setting "sendonly", "recvonly" or "inactive"
attributes) will not close the MediaStreamTrack.

The association between SSRCs and media descriptions is specified in
[I-D.ietf-rtcweb-jsep].

3.1. Handling of non-signalled tracks

Entities that do not use msid will not send msid. This means that
there will be some incoming RTP packets that the recipient has no
predefined MediaStream id value for.

Note that this handling is triggered by incoming RTP packets, not by
SDP negotiation.

When MSID is used, the only time this can happen is when, at a time
subsequent to the initial negotiation, a negotiation is performed
where the answerer adds a MediaStreamTrack to an already established
connection and starts sending data before the answer is received by
the offerer. For initial negotiation, packets won't flow until the
ICE candidates and fingerprints have been exchanged, so this is not
an issue.

The recipient of those packets will perform the following steps:

o When RTP packets are initially received, it will create an
   appropriate MediaStreamTrack based on the type of the media
   (carried in PayloadType), and use the MID RTP header extension
   [I-D.ietf-mmusic-sdp-bundle-negotiation] (if present) to associate
   the RTP packets with a specific media section. If the connection
   is not in the RTCSignalingState "stable", it will wait at this
   point.

o When the connection is in the RTCSignalingState "stable", it will
   look at the relevant media section to find the msid attribute.

o If there is an msid attribute, it will use that attribute to
   populate the "id" field of the MediaStreamTrack and associated
   MediaStreams, as described above.

o If there is no msid attribute, the identifier of the
   MediaStreamTrack will be set to a randomly generated string, and
   it will be signalled as being part of a MediaStream with the
   WebIDL "label" attribute set to "Non-WebRTC stream".

o After deciding on the "id" field to be applied to the
   MediaStreamTrack, the track will be signalled to the user.

The process above may involve a considerable amount of buffering
before the stable state is entered, If the implementation wishes to
limit this buffering, it MUST signal to the user that media has been
discarded.

It follows from the above that media stream tracks in the "default"
media stream cannot be closed by removing the msid attribute; the
application must instead signal these as closed when the SSRC
disappears according to the rules of RFC 3550 section 6.3.4 and 6.3.5
or by disabling the media description by setting its port to zero.

3.2. Detailed Offer/Answer Procedures

These procedures are given in terms of RFC 3264-recommended sections.
They describe the actions to be taken in terms of MediaStreams and
MediaStreamTracks; they do not include event signalling inside the
application, which is described in JSEP.

3.2.1. Generating the initial offer

For each media description in the offer, if there is an associated
outgoing MediaStreamTrack, the offerer adds one "a=msid" attribute to
the section for each MediaStream with which the MediaStreamTrack is
associated. The "identifier" field of the attribute is set to the
WebIDL "id" attribute of the MediaStream, and the "appdata" field is
set to the WebIDL "id" attribute of the MediaStreamTrack.

3.2.2. Answerer processing of the Offer

For each media description in the offer, and for each "a=msid"
attribute in the media description, the receiver of the offer will
perform the following steps:

o Extract the "appdata" field of the "a=msid" attribute

o Check if a MediaStreamTrack with the same WebIDL "id" attribute as
   the "appdata" field already exists, and is not in the "ended"
   state. If it is not found, create it.

o Extract the "identifier" field of the "a=msid" attribte.

o Check if a MediaStream with the same WebIDL "id" attribute already
   exists. If not, create it.

o Add the MediaStreamTrack to the MediaStream

o Signal to the user that a new MediaStreamTrack is available.

3.2.3. Generating the answer

The answer is generated in exactly the same manner as the offer.
"a=msid" values in the offer do not influence the answer.

3.2.4. Offerer processing of the answer

   The answer is processed in exactly the same manner as the offer.

3.2.5. Modifying the session

   On subsequent exchanges, precisely the same procedure as for the
   initial offer/answer is followed, but with one additional step in the
   parsing of the offer and answer:

   o For each MediaStreamTrack that has been created as a result of
      previous offer/answer exchanges, and is not in the "ended" state,
      check to see if there is still an "a=msid" attribute in the
      present SDP whose "appdata" field is the same as the WebIDL "id"
      attribute of the track.

   o If no such attribute is found, stop the MediaStreamTrack. This
      will set its state to "ended".

3.3. Example SDP description

   The following SDP description shows the representation of a WebRTC
   PeerConnection with two MediaStreams, each of which has one audio and
   one video track. Only the parts relevant to the MSID are shown.

   Line wrapping, empty lines and comments are added for clarity. They
   are not part of the SDP.

   # First MediaStream - id is 4701...
   m=audio 56500 UDP/TLS/RTP/SAVPF 96 0 8 97 98
   a=msid:47017fee-b6c1-4162-929c-a25110252400
          f83006c5-a0ff-4e0a-9ed9-d3e6747be7d9

   m=video 56502 UDP/TLS/RTP/SAVPF 100 101
   a=msid:47017fee-b6c1-4162-929c-a25110252400
          b47bdb4a-5db8-49b5-bcdc-e0c9a23172e0

   # Second MediaStream - id is 6131....
   m=audio 56503 UDP/TLS/RTP/SAVPF 96 0 8 97 98
   a=msid:61317484-2ed4-49d7-9eb7-1414322a7aae
          b94006c5-cade-4e0a-9ed9-d3e6747be7d9

   m=video 56504 UDP/TLS/RTP/SAVPF 100 101
   a=msid:61317484-2ed4-49d7-9eb7-1414322a7aae
          f30bdb4a-1497-49b5-3198-e0c9a23172e0

4. IANA Considerations

4.1. Attribute registration in existing registries

This document requests IANA to register the "msid" attribute in the
"att-field (media level only)" registry within the SDP parameters
registry, according to the procedures of [RFC4566]

The required information for "msid" is:

o Contact name, email: IETF, contacted via mmusic@ietf.org, or a
   successor address designated by IESG

o Attribute name: msid

o Long-form attribute name: Media stream group Identifier

o Subject to charset: The attribute value contains only ASCII
   characters, and is therefore not subject to the charset attribute.

o Purpose: The attribute can be used to signal the relationship
   between a WebRTC MediaStream and a set of media descriptions.

o Appropriate values: The details of appropriate values are given in
   RFC XXXX.

o MUX category: NORMAL

The MUX category is defined in [I-D.ietf-mmusic-sdp-mux-attributes].

5. Security Considerations

An adversary with the ability to modify SDP descriptions has the
ability to switch around tracks between media streams. This is a
special case of the general security consideration that modification
of SDP descriptions needs to be confined to entities trusted by the
application.

If implementing buffering as mentioned in Section 3.1, the amount of
buffering should be limited to avoid memory exhaustion attacks.

No other attacks have been identified that depend on this mechanism.

6. Acknowledgements

This note is based on sketches from, among others, Justin Uberti and
Cullen Jennings.

Special thanks to Flemming Andreassen, Miguel Garcia, Martin Thomson, Ted Hardie, Adam Roach, Magnus Westerlund and Paul Kyzivat for their work in reviewing this draft, with many specific language suggestions.

7. References

7.1. Normative References

[I-D.ietf-mmusic-sdp-mux-attributes]
          Nandakumar, S., "A Framework for SDP Attributes when
          Multiplexing", draft-ietf-mmusic-sdp-mux-attributes-02
          (work in progress), July 2014.

[I-D.ietf-rtcweb-jsep]
          Uberti, J., Jennings, C., and E. Rescorla, "Javascript
          Session Establishment Protocol", draft-ietf-rtcweb-jsep-07
          (work in progress), July 2014.

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

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.
          Jacobson, "RTP: A Transport Protocol for Real-Time
          Applications", STD 64, RFC 3550, July 2003.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session
          Description Protocol", RFC 4566, July 2006.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax
          Specifications: ABNF", STD 68, RFC 5234, January 2008.

[W3C.WD-webrtc-20150210]
          Bergkvist, A., Burnett, D., Jennings, C., and A.
          Narayanan, "WebRTC 1.0: Real-time Communication Between
          Browsers", World Wide Web Consortium WD WD-
          webrtc-20150210, February 2015,
          <http://www.w3.org/TR/2015/WD-webrtc-20150210>.

7.2. Informative References

[I-D.ietf-mmusic-sdp-bundle-negotiation]
          Holmberg, C., Alvestrand, H., and C. Jennings,
          "Negotiating Media Multiplexing Using the Session
          Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-
          negotiation-07 (work in progress), April 2014.

   [I-D.ietf-rtcweb-overview]
              Alvestrand, H., "Overview: Real Time Protocols for
              Browser-based Applications", draft-ietf-rtcweb-overview-10
              (work in progress), June 2014.

   [RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and
              Control Packets on a Single Port", RFC 5761, April 2010.

   [RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description
              Protocol (SDP) Grouping Framework", RFC 5888, June 2010.

Appendix A. Design considerations, rejected alternatives

   One suggested mechanism has been to use CNAME instead of a new
   attribute. This was abandoned because CNAME identifies a
   synchronization context; one can imagine both wanting to have tracks
   from the same synchronization context in multiple MediaStreams and
   wanting to have tracks from multiple synchronization contexts within
   one MediaStream (but the latter is impossible, since a MediaStream is
   defined to impose synchronization on its members).

   Another suggestion has been to put the msid value within an attribute
   of RTCP SR (sender report) packets. This doesn't offer the ability
   to know that you have seen all the tracks currently configured for a
   media stream.

   A suggestion that survived for a number of drafts was to define
   "msid" as a generic mechanism, where the particular semantics of this
   usage of the mechanism would be defined by an "a=wms-semantic"
   attribute. This was removed in April 2015.

Appendix B. Change log

   This appendix should be deleted before publication as an RFC.

B.1. Changes from alvestrand-rtcweb-msid-00 to -01

   Added track identifier.

   Added inclusion-by-reference of draft-lennox-mmusic-source-selection
   for track muting.

   Some rewording.

B.2. Changes from alvestrand-rtcweb-msid-01 to -02

Split document into sections describing a generic grouping mechanism
and sections describing the application of this grouping mechanism to
the WebRTC MediaStream concept.

Removed the mechanism for muting tracks, since this is not central to
the MSID mechanism.

B.3. Changes from alvestrand-rtcweb-msid-02 to mmusic-msid-00

Changed the draft name according to the wishes of the MMUSIC group
chairs.

Added text indicting cases where it's appropriate to have the same
appdata for multiple SSRCs.

Minor textual updates.

B.4. Changes from alvestrand-mmusic-msid-00 to -01

Increased the amount of explanatory text, much based on a review by
Miguel Garcia.

Removed references to BUNDLE, since that spec is under active
discussion.

Removed distinguished values of the MSID identifier.

B.5. Changes from alvestrand-mmusic-msid-01 to -02

Changed the order of the "msid-semantic: " attribute's value fields
and allowed multiple identifiers. This makes the attribute useful as
a marker for "I understand this semantic".

Changed the syntax for "identifier" and "appdata" to be "token".

Changed the registry for the "msid-semantic" attribute values to be a
new registry, based on advice given in Atlanta.

B.6. Changes from alvestrand-mmusic-msid-02 to ietf-mmusic-00

Updated terminology to refer to m-lines rather than RTP sessions when
discussing SDP formats and the ability of other linking mechanisms to
refer to SSRCs.

Changed the "default" mechanism to return independent streams after
considering the synchronization problem.

Removed the space from between "msid-semantic" and its value, to be consistent with RFC 5576.

B.7. Changes from mmusic-msid-00 to -01

Reworked msid mechanism to be a per-m-line attribute, to align with draft-roach-mmusic-unified-plan.

B.8. Changes from mmusic-msid-01 to -02

Corrected several missed cases where the word "ssrc" was not changed to "M-line".

Added pointer to unified-plan (which should be moved to point to -jsep)

Removed suggestion that ssrc-group attributes can be used with "msid-semantic", it is now only the msid-semantic registry.

B.9. Changes from mmusic-msid-02 to -03

Corrected even more cases where the word "ssrc" was not changed to "M-line".

Added the functionality of using an asterisk (*) in the msid-semantic line, in order to remove the need for listing all msids in the msid-semantic line whne only one msid-semantic is in use.

Removed some now-unnecessary text.

B.10. Changes from mmusic-msid-03 to -04

Changed title to reflect focus on WebRTC MediaStreams

Added a section on receiver-side media stream control, using the "msid-control" attribute.

B.11. Changes from -04 to -05

Removed the msid-control section after WG discussion.

Removed some text that seemed only to pertain to resolved issues.

B.12. Changes from -05 to -06

Addressed issues found in Fleming Andreassen's review

Referenced JSEP rather than unified-plan for the M-line mapping model

Relaxed MSID definition to allow "token-char" in values rather than
a-z 0-9 hyphen; tightened ABNF by adding length description to it.

Deleted discussion of abandoned alternatives, as part of preparing
for publication.

Added a "detailed procedures" section to the WMS semantics
description.

Added IANA registration of the "msid-semantic" attribute.

B.13. Changes from -06 to -07

Changed terminology from referring to "WebRTC device" to referring to
"entities that implement the WMS semantic".

Changed names for ABNF constructions based on a proposal by Paul
Kyzivat.

Included a section on generic offer/answer semantics.

B.14. Changes from -07 to -08

Removed Appendix B that described the (now obsolete) ssrc-specific
usage of MSID.

Adopted a restructuring of the IANA section based on a suggestion
from Martin Thomson.

A number of text and ABNF clarifications based on suggestions from
Ted Hardie, Paul Kyzivat and Adam Roach.

Changed the "non-signalled track handling" to create a single stream
with multiple tracks again, according to discussions at TPAC in
November 2014

B.15. Changes from -08 to -09

Removed "wms-semantic" and all mention of multiple semantics for
msid, as agreed at the Dallas IETF, March 2015.

Addressed a number of review comments from Fleming Andresen and
others.

Changed the term "m-line" to "media description", since that is the
term used in RFC 4566.

   Tried to make sure this document does not describe the API to the
   application.

B.16. Changes from -09 to -10

   Addressed review comments from Paul Kyzivat.

B.17. Changes from -10 to -11

   Defined the semantics of multiple MSIDs in a media section to be a
   MediaStreamTrack present in multiple MediaStreams.

   Made an explicit note that MediaStreamTracks are unidirectional.

   Disallowed the option of sending multiple media sections with the
   same msid (id and appdata identical).

B.18. Changes from -11 to -12

   Added mux-category to the IANA considerations section.

B.19. Changes from -12 to -13

   Modified registration description to delete dependency on -4566-bis

Author's Address

   Harald Alvestrand
   Google
   Kungsbron 2
   Stockholm 11122
   Sweden

   Email: harald@alvestrand.no