Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 16, 2014

E. Ivov
Jitsi

July 15, 2013

Economical Use of the Offer/Answer Model in Sessions with Multiple Media Sources

draft-ivov-mmusic-multiple-sources-00

Abstract

The Session Description Protocol (SDP) Offer/Answer model describes a mechanism that allows two entities to negotiate a multimedia session. The SDP syntax of the Offer/Answer model uses constructs known as media (m=) lines to describe each medium. In SDP itself these "m=" lines were designed to describe RTP sessions with any number of streams (SSRCs). Yet, Offer/Answer implementations in SIP applications have most often used them as an envelope for a maximum of two RTP streams (SSRCs) at a time: one in each direction. The most common reason for this has been the fact these applications could not meaningfully render multiple SSRCs simultaneously.

The above situation has led to difficulties once the need to represent multiple (SSRCs) in an interoperable manner became more common.

This document explores the use of "m=" lines for the negotiation of sessions with multiple media sources, as per their original design in SDP. It presents the advantages of such an approach as well as the challenges that it implies in terms of interoperability with already deployed legacy devices.

The model described here was first presented in the RTCWEB No Plan proposal. The reason to spin it off into this new document is mainly to separate the parts related to Offer/Answer and "m=" line semantics, from those that are specific to WebRTC.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

This Internet-Draft will expire on January 16, 2014.

Table of Contents

1. Introduction

Since its early days the Session Description Protocol (SDP) Offer/ Answer (O/A) model [RFC3264] has mainly been used for the negotiation of Real-time Transport Protocol (RTP) [RFC3550] sessions between endpoints that only use a single media source per medium. Examples of such sources include microphones, cameras, desktop streamers etc. The list can be extended to cases where multiple sources are mixed at the media level by an audio or video mixer and then appear as a single stream, i.e. RTP Synchronisation Source (SSRC) on the RTP and SDP [RFC4566] levels.

In such sessions each medium and its corresponding device are described by the SDP "m=" line construct and each media source is mapped to exactly one "m=" line. The exchanges that lead to the establishment of such sessions are relatively well covered by the specifications and most implementations.

Unfortunately, the situation becomes relatively confusing when it comes to transporting multiple media sources (SSRCs) per medium. Streaming any number of RTP streams is an inherent part of the protocol and describing such multi-stream RTP sessions is directly supported by SDP. Still, the Offer/Answer model [RFC3264] is relatively vague on the subject and relying on the multi-stream capabilities of an SDP "m=" line is likely to lead to unexpected results with most endpoints.

At the time of writing of this document, the MMUSIC working group is considering two approaches to addressing the issue. The approaches emerged in the RTCWEB working group and are often referred to as Plan A [PlanA] and Plan B [PlanB]. Both of them impose semantics and syntax that allow for detailed description and fine-grained control of multiple media sources entirely through SDP and Offer/Answer.

Both plans A and B present a number of problems most of which are due to the heavy reliance on SDP O/A. The problems are discussed in more detail in Section 6.

The goal of this document is to propose an alternative approach that simply uses "m=" lines in the way they were originally designed with SDP: descriptors of RTP sessions with any number of sources. Such an approach keeps the use of SDP Offer/Answer to the initialization of transport and media chains and delegates stream control to other, upper layer protocols.

The model described in this specification is intended for applications that require reliability, flexibility and scalability. It therefore tries to satisfy the following constraints

- the addition and removal of media sources (e.g. conference participants, multiple web cams or "slides") must be possible without the need of Offer/Answer exchanges;

- the addition or removal of simulcast or layered streams must be possible without the need for Offer/Answer exchanges beyond the initial declaration of such capabilities for either direction.

- call establishment must not require preliminary announcement or even knowledge of all potentially participating media sources;

- application specific signalling should be used to cover most semantics following call establishment, such as adding, removing or identifying SSRCs;

- straightforward interoperability with widely deployed legacy endpoints with rudimentary support for Offer/Answer. This includes devices that allow for one audio and potentially one video m= line and that expect to only ever be required to render a single RTP stream at a time for any of them. (Note that this does NOT include devices that expect to see multiple "m=video" lines for different SSRCs as they can hardly be viewed as "widely deployed legacy").

To achieve the above requirements this specification expects that endpoints will only use SDP Offer/Answer to establish transport channels and initialize an RTP stack and codec/processing chains. This also includes any renegotiation that requires the re-initialisation of these chains. For example, adding VP8 to a session that was setup with only H.264, would obviously still require an Offer/Answer exchange.

All other session control and signalling are to be left to upper layer protocol mechanisms.

2. Mechanism

The model presented in this specification relies on use of standard SDP and Offer/Answer for negotiating formats, establishing transport channels and exchanging, in a declarative way, media and transport parameters that are then used for the initialization of the corresponding stacks. It does not add new concepts and simply requires applications to abide by the original design of SDP and the "m=" line construct.

The following is an example presenting what this specification views as a typical offer sent by a multistream endpoint following this specification:

```
   v=0
   o=- 0 0 IN IP4 198.51.100.33
   s=
   t=0 0

   a=group:BUNDLE audio video               // declaring BUNDLE Support
   c=IN IP4 198.51.100.33
   a=ice-ufrag:Qq8o/jZwknkmXpIh             // initializing ICE
   a=ice-pwd:gTMACiJcZv1xdPrjfbTHL5qo
   a=ice-options:trickle
   a=fingerprint:sha-1                      // DTLS-SRTP keying
         a4:b1:97:ab:c7:12:9b:02:12:b8:47:45:df:d8:3a:97:54:08:3f:16

   m=audio 5000 RTP/SAVPF 96 0 8
   a=mid:audio
   a=rtcp-mux

   a=rtpmap:96 opus/48000/2                 // PT mappings
   a=rtpmap:0 PCMU/8000
   a=rtpmap:8 PCMA/8000

   a=extmap:1 urn:ietf:params:rtp-hdrext:csrc-audio-level  //5825 header
   a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level  //extensions

   [ICE Candidates]

   m=video 5002 RTP/SAVPF 97 98
   a=mid:video
   a=rtcp-mux


   a=rtpmap:97 VP8/90000     // PT mappings and resolutions capabilities
   a=imageattr:97 \
     send [x=[480:16:800],y=[320:16:640],par=[1.2-1.3],q=0.6] \
          [x=[176:8:208],y=[144:8:176],par=[1.2-1.3]] \
     recv *
   a=rtpmap:98 H264/90000
   a=imageattr:98 send [x=800,y=640,sar=1.1,q=0.6] [x=480,y=320] \
                  recv [x=330,y=250]

   a=extmap:3 urn:ietf:params:rtp-hdrext:fec-source-ssrc   //5825 header
   a=extmap:4 urn:ietf:params:rtp-hdrext:rtx-source-ssrc   //extensions

   a=max-send-ssrc:{*:1}                    // declaring maximum
   a=max-recv-ssrc:{*:4}                    // number of SSRCs

   [ICE Candidates]
```

The answer to the offer above would have substantially the same structure and content. For the sake of clarity:

```
v=0
o=- 0 0 IN IP4 203.0.113.12
s=
t=0 0

a=group:BUNDLE audio video                // declaring BUNDLE Support
c=IN IP4 203.0.113.12
a=ice-ufrag:Qq8o/jZwknkmXpIh              // initializing ICE
a=ice-pwd:gTMACiJcZv1xdPrjfbTHL5qo
a=ice-options:trickle
a=fingerprint:sha-1                       // DTLS-SRTP keying
      a4:b1:97:ab:c7:12:9b:02:12:b8:47:45:df:d8:3a:97:54:08:3f:16

m=audio 5000 RTP/SAVPF 96 0 8
a=mid:audio
a=rtcp-mux

a=rtpmap:96 opus/48000/2                  // PT mappings
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000

a=extmap:1 urn:ietf:params:rtp-hdrext:csrc-audio-level  //5825 header
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level  //extensions

[ICE Candidates]

m=video 5002 RTP/SAVPF 97 98
a=mid:video
a=rtcp-mux


a=rtpmap:97 VP8/90000     // PT mappings and resolutions capabilities
a=imageattr:97 \
  send [x=[480:16:800],y=[320:16:640],par=[1.2-1.3],q=0.6] \
       [x=[176:8:208],y=[144:8:176],par=[1.2-1.3]] \
  recv *
a=rtpmap:98 H264/90000
a=imageattr:98 send [x=800,y=640,sar=1.1,q=0.6] [x=480,y=320] \
               recv [x=330,y=250]

a=extmap:3 urn:ietf:params:rtp-hdrext:fec-source-ssrc   //5825 header
a=extmap:4 urn:ietf:params:rtp-hdrext:rtx-source-ssrc   //extensions

a=max-send-ssrc:{*:4}                     // declaring maximum
```

```
a=max-recv-ssrc:{*:4}                    // number of SSRCs

[ICE Candidates]
```

As already noted, the Offer/Answer exchange above remains essentially the same regardless of whether the following media session is going to contain one or multiple RTP streams (SSRCs) per medium in either direction.

The exchange also has the following important characteristics:

- Preserves interoperability with most kinds of legacy endpoints.
- Allows the negotiation of most parameters that concern the media/RTP stack (typically the browser).
- Only a single Offer/Answer exchange is required for session establishment and, in most cases, for the entire duration of a session.
- Leaves complete freedom to applications as to the way that they are going to signal any other information such as SSRC identification information or the addition or removal of RTP streams.

2.1. Discovery

It is important that an implementation using "m=" lines as an envelope for multiple RTP media streams, be able to reliably detect whether its peer is capable of receiving them. One way of achieving this would be the use of upper-layer protocols as explained in Section 3.

In cases where endpoints need to be able to detect this from the SDP Offer/Answer they could use the "max-send-ssrc" and "max-recv-ssrc" attributes defined in [MAX-SSRC]. It has to be noted however that this mechanism is still a work in progress and as such it would still need to be extended to provide ways of distinguishing between independent flows and complementary ones such as layered FEC and RTX.

Unless an endpoint detects the corresponding max-ssrc or upper level protocol indicators that a remote peer can actually handle multiple streams within a single "m=" line, it MUST assume that such support is unavailable.

2.2. Advantages

The advantages of using "m=" lines to represent multiple media sources in the way they were originally intended by SDP can be roughly summarized to the follwoing:

- It works. Existing implementations are already successfully using the approach.
- No Offer/Answer when adding/removing streams. Improves flexibility for applications allowing them to only signal information that they really need to.
- No added glare risk. Improves scalability and reliability.
- No need to pre-announce SSRCs. Improves scalability.
- Allows apps to choose fine-tuned signalling: Custom, XCON, RFC4575, WebRTC JavaScript, CLUE channels, or even Plan A and Plan B.

Combined, the above set of characteristics allow for a multi-stream management method that gives scalability, flexibility and reliability.

3. Additional Session Control and Signalling

- Adding and removing RTP streams to an existing session.
- Accepting and refusing some of them.
- Identifying SSRCs and obtaining additional metadata for them (e.g. the user corresponding to a specific SSRC).
- Requesting that additional SSRCs be added.
- Requesting that specific processing be applied on some SSRCs.

Support for any subset of the above semantics is highly dependent on the use cases and applications where they are employed. The position of this specification is that they should therefore be left to protocols that target more specific scenarios. There are numerous existing or emerging solutions, some of them developed by the IETF, that already cover this. This includes CLUE channels [CLUE], the SIP Event Package For Conference State [RFC4575] and its XMPP variant [COIN], as well as the protocols defined within the Centralised Conferencing IETF working group [XCON]. Additional mechanisms are very likely to emerge in the future as various applications address

specific use cases, scenarios and topologies. Examples for this could be WebRTC JavaScript applications using proprietary JSON descriptors, XMPP clients with new XML schemas and many others.

The most important part of this specification is hence to prevent certain assumptions or topologies from being imposed on applications. One example of this is the need to know and include in the Offer/Answer exchange, all the SSRCs that can show up in a session. This can be particularly problematic for scenarios that involve endpoints with varying constraints.

Large scale conference calls, potentially federated through RTP translator-like bridges, would be another problematic scenario. Being able to always pre-announce SSRCs in such situations could of course be made to work but it would come at a price. It would either require a very high number of Offer/Answer updates that propagate the information through the entire topology, or use of tricks such as pre-allocating a range of "fake" SSRCs, announcing them to participants and then overwriting the actual SSRCs with them. Depending on the scenario both options could prove inappropriate or inefficient while some applications may not even need such information. Others could be retrieving it through simplistic means such as access to a centralized resource (e.g. an URL pointing to a JSON description of the conference).

4. Demultiplexing and Identifying Streams (Use of Bundle)

For reasons of optimising traversal of Network Address Translation (NAT) gateways, it is likely for endpoints to use [BUNDLE]. This implies that all RTP streams would in many cases end up being received on the same port. A demuxing mechanism is therefore necessary in order for these packets to then be fed into the appropriate processing chain (i.e. matched to an "m=" line).

> Note: it is important to distinguish between the demultiplexing and the identification of incoming flows. Throughout this specification the former is used to refer to the process of choosing selecting a depacketizing/decoding/processing chain to feed incoming packets to. Such decisions depend solely on the format that is used to encode the content of incoming packets.

The above is not to be confused with the process of making rendering decision about a processed flow. Such decisions include showing a "current speaker" flow at a specific location, window or video tag, while choosing a different one for a second, "slides" flow. Another example would be the possibility to attach "Alice", "Bob" and "Carol" labels on top of the appropriate UI components. This specification leaves such rendering choices entirely to application-specific signalling as described in Section 3.

This specification uses demuxing based on RTP payload types. When creating offers and answers applications MUST therefore allocate RTP payload types only once per bundle group. In cases where rtcp-mux is in use this would mean a maximum of 96 payload types per bundle [RFC5761]. It has been pointed out that some legacy devices may have unpredictable behaviour with payload types that are outside the 96-127 range reserved by [RFC3551] for dynamic use. Some applications or implementations may therefore choose not to use values outside this range. Whatever the reason, offerers that find they need more than the available payload type numbers, will simply need to either use a second bundle group or not use BUNDLE at all (which in the case of a single audio and a single video "m=" line amounts to roughly the same thing). This would also imply building a dynamic table, mapping SSRCs to PTs and m= lines, in order to then also allow for RTCP demuxing.

While not desirable, the implications of such a decision would be relatively limited. Use of trickle ICE [TRICKLE-ICE] is going to lessen the impact on call establishment latency. Also, the fact that this would only occur in a limited number of cases makes it unlikely to have a significant effect on port consumption.

An additional requirement that has been expressed toward demuxing is the ability to assign incoming packets with the same payload type to different processing chains depending on their SSRCs. A possible example for this is a scenario where two video streams are being rendered on different video screens that each have their own decoding hardware.

While the above may appear as a demuxing and a decoding related problem it is really mostly a rendering policy specific to an application. As such it should be handled by app. specific signalling that could involve custom-formatted, per-SSRC information that accompanies SDP offers and answers.

5. Simulcasting, FEC, Layering and RTX (Open Issue)

Repair flows such as layering, FEC, RTX and to some extent simulcasting, present an interesting challenge, which is why they are considered an open issue by this specification.

On the one hand they are transport utilities that need to be understood, supported and used by the level of the media libraries in a way that is mostly transparent to applications. On the other, some applications may need to be made aware of them and given the option to control their use. This could be necessary in cases where their use needs to be signalled to endpoints through application-specific non-SDP mechanisms. Another example is the possibility for an application to choose to disable some or all repair flows because it has been made aware by application-specific signalling that they are temporarily not being used/rendered by the remote end (e.g. because it is only displaying a thumbnail or because a corresponding video tag is not currently visible).

One way of handling such flows would be to advertise them in the way suggested by [RFC5956] and to then control them through application specific signalling. This options has the merit of already existing but it also implies the pre-announcement and propagation of SSRCs and the bloated signalling that this incurs. Also, relying solely on Offer/Answer here would expose an offerer to the typical race condition of repair SSRCs arriving before the answer and the processing ambiguity that this would imply.

Another approach could be a combination of RTCP and RTP header extensions [RFC5285] in a way similar to the one employed by the Rapid Synchronisation of RTP Flows [RFC6051]. While such a mechanism is not currently defined by the IETF, specifying it could be relatively straightforward:

Every packet belonging to a repair flow could carry an RTP header extension [RFC5285] that points to the source stream (or source layer in case of layered mechanisms).

Again, these are just some possibilities. Different mechanisms may and probably will require different extensions or signalling ([SRCNAME] will likely be an option for some). In some cases, where layering information is provided by the codec, an extensions is not going to be necessary at all.

In cases where FEC or simulcast relations are not immediately needed by the recipient, this information could also be delayed until the reception of the first RTCP packet.

6. Problems with Plans A and B

As already mentioned both Plans A and B heavily rely on SDP and Offer /Answer for advanced stream control. They both require Offer/Answer exchanges in a number of situations where this could be avoided, particularly when adding or removing media sources to a session. This requirement applies equally to cases where a client adds the stream of a newly activated web cam, a simulcast flow or upon the arrival or departure of a conference participant.

Plan A handles such notifications with the addition or removal of independent m= lines [PlanA], while Plan B relies on the use of multiplexed m= lines but still depends on the Offer/Answer exchanges for the addition or removal of media stream identifiers [MSID].

By taking the Offer/Answer approach, both Plan A and Plan B take away from the application the opportunity to handle such events in a way that is most fitting for the use case, which, among other things, also goes against the working group's decision to not to define a specific signalling protocol. (It could be argued that it is therefore only natural how proponents of each plan, having different use cases in mind, are remarkably far from reaching consensus).

Reliance on preliminary announcement of SSRC identifiers is another issue. While this could be perceived as relatively straightforward in one-to-one sessions or even conference calls within controlled environments, it can be a problem in the following cases:

- interoperability with legacy endpoints

- use within non-controlled and potentially federated conference environments where new RTP streams may appear relatively often. In such cases the signalling required to describe all of them through Offer/Answer may represent substantial overhead while none or only a part of it (e.g. the description of a main, active speaker stream) may be required by the application.

By increasing the number of Offer/Answer exchanges Both Plan A and Plan B also increase the risk of encountering glare situations (i.e. cases where both parties attempt to modify a session at the same time). While glare is also possible with basic Offer/Answer and resolution of such situations must be implemented anyway, the need to frequently resort to such code may either negatively impact user experience (e.g. when "back off" resolution is used) or require substantial modifications in the Offer/Answer model and/or further venturing into the land of signalling protocols [ROACH-GLARELESS-ADD].

7. IANA Considerations

None.

8. Informative References

[BUNDLE] Holmberg, C., Alvestrand, H., and C. Jennings, "Multiplexing Negotiation Using Session Description Protocol (SDP) Port Numbers", reference.I-D.ietf-clue-framework (work in progress), June 2013, <reference.I-D.ietf-mmusic-sdp-bundle-negotiation>.

[CLUE] Duckworth, M., Pepperell, A., and S. Wenger, "Framework for Telepresence Multi-Streams", reference.I-D.ietf-clue-framework (work in progress), May 2013, <reference.I-D.ietf-clue-framework>.

[COIN] Ivov, E. and E. Marocco, "XEP-0298: Delivering Conference Information to Jingle Participants (Coin)", XSF XEP 0298, June 2011, <reference.I-D.ietf-coin-framework>.

[MAX-SSRC] Westerlund, M., Burman, B., and F. Jansson, "Multiple Synchronization sources (SSRC) in RTP Session Signaling ", reference.I-D.westerlund-avtcore-max-ssrc (work in progress), July 2012, <reference.I-D.westerlund-avtcore-max-ssrc>.

[MSID] Alvestrand, H., "Cross Session Stream Identification in the Session Description Protocol", reference.I-D.ietf-mmusic-msid (work in progress), February 2013, <reference.I-D.ietf-mmusic-msid>.

[PlanA] Roach, A. and M. Thomson, "Using SDP with Large Numbers of Media Flows", reference.I-D.roach-rtcweb-plan-a (work in progress), May 2013, <reference.I-D.roach-rtcweb-plan-a>.

[PlanB] Uberti, J., "Plan B: a proposal for signaling multiple media sources in WebRTC.", reference.I-D.uberti-rtcweb-plan (work in progress), May 2013, <reference.I-D.uberti-rtcweb-plan>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

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

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, July 2003.

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

[RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, August 2006.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

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

[RFC5956] Begen, A., "Forward Error Correction Grouping Semantics in the Session Description Protocol", RFC 5956, September 2010.

[RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, November 2010.

[ROACH-GLARELESS-ADD]
Roach, A., "An Approach for Adding RTCWEB Media Streams without Glare", reference.I-D.roach-rtcweb-glareless-add (work in progress), May 2013, <reference.I-D.roach-rtcweb-glareless-add>.

[SRCNAME] Westerlund, M., Burman, B., and P. Sandgren, "RTCP SDES Item SRCNAME to Label Individual Sources ", reference.I-D .westerlund-avtext-rtcp-sdes-srcname (work in progress), October 2012, <reference.I-D.westerlund-avtext-rtcp-sdes-srcname>.

[TRICKLE-ICE]
Ivov, E., Rescorla, E., and J. Uberti, "Trickle ICE: Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol ", reference.I-D .ivov-mmusic-trickle-ice (work in progress), March 2013, <reference.I-D.ivov-mmusic-trickle-ice>.

[XCON] , "Centralized Conferencing (XCON) Status Pages", , <http://tools.ietf.org/wg/xcon/>.

Appendix A. Acknowledgements

Many thanks to Jonathan Lennox for reviewing the document and providing valuable feedback.

Author's Address

Emil Ivov
Jitsi
Strasbourg 67000
France

Phone: +33-177-624-330
Email: emcho@jitsi.org