Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 22, 2013

S. Nandakumar
C. Jennings
Cisco

February 18, 2013

Glareless addition of media to existing RTCWeb Sessions
draft-nandakumar-rtcweb-glare-handling-00.txt

Abstract

The RFC3264 Offer/Answer model specifies rule for the bilateral exchange of Session Description Protocol (SDP) [RFC4566] messages for setting up, updating and tearing down of multimedia streams. Rarely, there might be situations wherein either of the communicating parties, might end up being the offerer for updating an on-going session. This scenario is commonly known as "glare" condition and it needs to be handled nevertheless. This specification describes procedures for parties involved in an ongoing RTCWeb session to add new media in a glareless fashion.

There are various ways this problem might be solved - this draft sketches out one possible solution to the problem.

---

1. Introduction

In the most basic form,the RFC3264 **[RFC3264]** protocol operation begins by one of the participants sending an initial SDP offer describing its intent to start a multimedia communication session. The participant receiving the offer MAY generate an SDP answer accepting the offer or it MAY reject the offer. Once the session is setup, at any time, either agent MAY generate a new offer that updates the session. However, it MUST NOT generate a new offer if it has received an offer which it has not yet been answered or rejected. If an agent receives an offer after having sent one, but before receiving an answer to it,the situation is considered as "glare" condition.

This specification defines set of procedures for RTCWeb end-points to add new media to an ongoing session in a glareless fashion. The rest of this document is organized as follow. Section 2 provides motivation for dealing with glare condition. Section 3 explains the detailed call-flows with examples describing the solution proposed in this specification. Finally Section 4 concludes with a note on applicability.

---

2. Terminology

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

m-line: An **RFC4566** [RFC4566] media description identifier that starts with "m=" field and conveys following values:media type,transport port,transport protocol and media format descriptions.

m-block: An **RFC4566** [RFC4566] media description that starts with an m-line and is terminated by either the next m-line or by the end of the session description.

Offer: An **[RFC3264]** SDP message generated by the participant who wishes to initiate a multimedia communication session.

An Offer describes participants capabilities for engaging in a multimedia session.

Answer: An **[RFC3264]** SDP message generated by the participant in response to an Offer. An Answer describes participants capabilities in continuing with the multimedia session with in the constraints of the Offer.

---

3. Motivation

The following example serves as base case for "glare" condition that will be discussed throughout this document in the context of the proposed solution.

> Alice and Bob are in a two way audio-only RTCWeb Session.
>
> They decide to escalate to a video session.
>
> Each initiate the addition of their respective camera streams to their current session almost the same time. This is done by each side sending the Offer with appropriate media descriptions.
>
> Alice receives Offer from the Bob before receiving Answer for her Offer.
>
> Bob receives Alice's Offer when he is expecting an Answer for his Offer.
>
> Both Alice and Bob have outstanding Offers and they end up stuck in a "glare" situation.

End points stuck in the "glare" condition stay there forever unless the condition is resolved either by some higher layer protocol mechanisms or by some application logic. An example for the former is SIP **[RFC3261]** that provides means for ordering of messages in each direction to resolve "glare" condition. This documents proposes a case for the latter.

---

4. Solution

Here is the basic outline of the solution described herein.

1. Participants setup an initial RTCWeb session using normal Offer/Answer procedures. This can by any combination of audio, video or data-channel sessions.
2. Whenever participants decide to add new media, a m-block describing the capabilities of media stream is generated and exchanged with the peer.
3. Each sender "Opportunistically" acknowledges the m-block with an Answer that it is expecting for the m-blocks sent. This triggers the completion of Offer/Answer state machine at the senders thus not resulting in any outstanding Offers. In this specification, the Answer used for this purpose is called "ExpectedAnswer" since this is not the real answer from the Answerer.
4. When the m-blocks reach the Answerer's Javascript application, the application needs to re-order the m-blocks to be in the right order for the local SDP
5. Once re-ordered to match the order of m-blocks in the local SDP, the Offer is installed with the Peer Connection at the Answerer,as remote description, to generate an Answer
6. On obtaining the Answer, the Answerer's Javascript application extracts the m-block, sends it to the Offerer as response.
7. On receiving the response m-blocks from the Answerer, the Offerer needs to re-order the m-blocks to be in the right order for the Offerer's local SDP.
8. The Offerer's Javascript application then will generate an updated Offer with the m-blocks received and re-ordered, to be applied as remote description to generate an Answer. This is done to apply the real Answer from the Answerer and overwrite any number of "ExpectedAnswers" applied in the interim. The term "Overlapping Zone" is used to indicate time between initial m-block Offer and the final m-block Answer. During this time, the end-points at the either end MAY be executing one or more iterations of m-block exchanges".
9. There could be few limitations that might result in the failure while converting the m-block Answer to an Offer at the Offerer as described in the section **Section 5**
10. The Offerer's Javascript application will extract m-blocks from the generated Answer to match it with its local SDP to ensure that a successful m-block Offer/Answer exchange did happen. If not, the on-going session is terminated.
11.

4.1. Requirements

Following represent the basic ideas and the requirements for the proposed solution to perform glareless addition of new media.

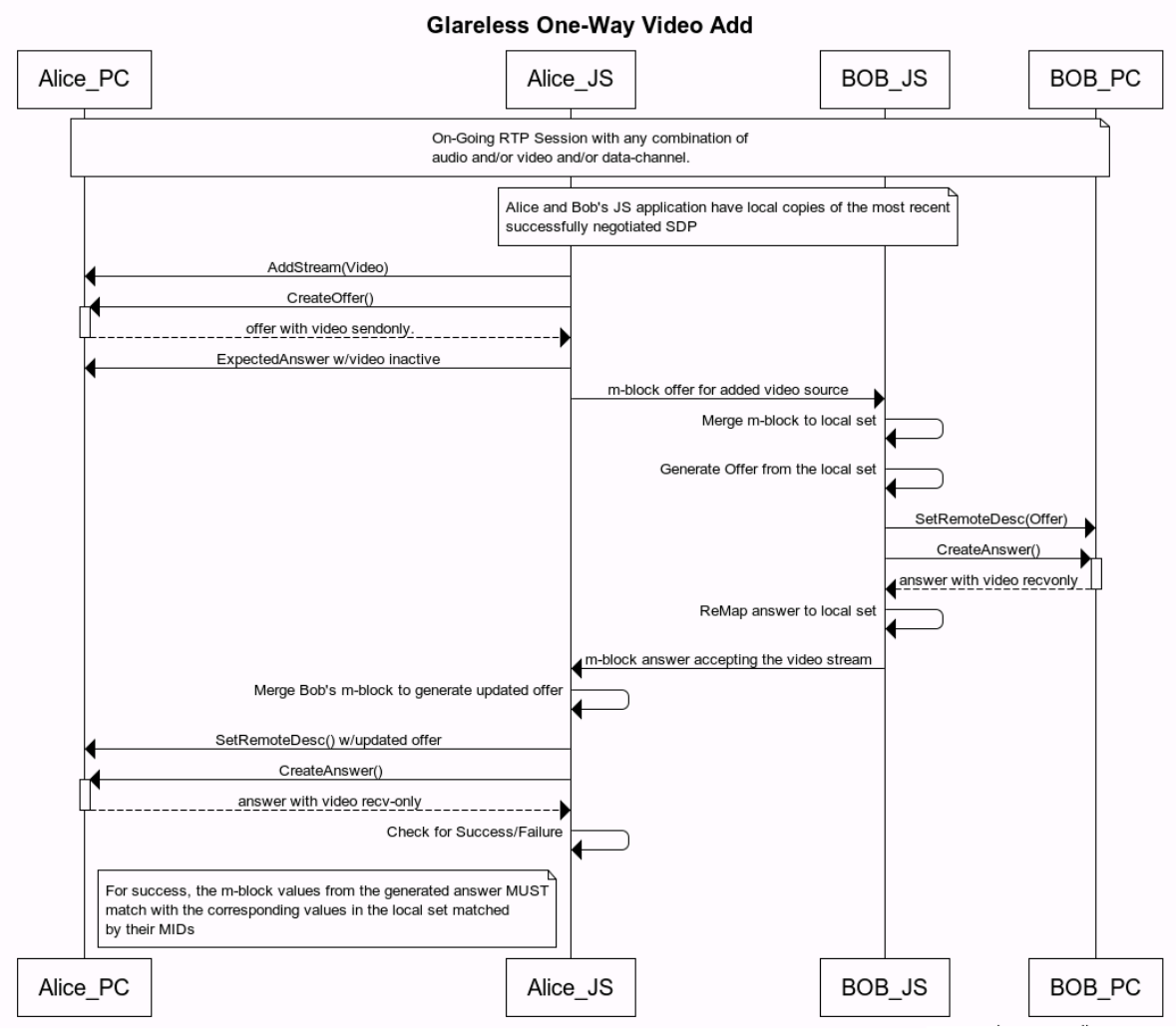
- There is an ongoing RTCWeb session between the parties, say Alice and Bob. This implies an RTP transport association has been successfully setup between the peers.
- The Javascript applications have their own copy of the most recent successfully negotiated SDP with the local ordering of m-blocks preserved.
- The Javascript application is responsible for maintaining the appropriate ordering of the m-blocks for that User Agent.
- Subsequent Offer/Answer exchanges after the initial exchange might not use complete SDP messages to indicate

updates to the session, say adding a new video stream. One approach proposed in this document involves exchange of "m-blocks" that describe the capabilities of the newly added media.

- The Javascript application MUST be capable of generating complete SDP Offer or Answer from the session updates exchanged so as to supply the same to the User Agent or to negotiate the same with the peer, if needed.
- The end-points MUST support at least one way to identify the m-blocks. In this specification mid **[RFC5888]** is used for this purpose and it is open to other identification approaches such as msid **[I-D.alvestrand-mmusic-msid]**

4.2. Call Flow - Glareless One-Way Video Addition

Below is the high-level call flow that captures the procedure for adding one-way video to an ongoing RTCWeb session in a glareless fashion. Also to note, the call-flow captures the proposed solution from the Alice's perspective alone due to the implied symmetry.



4.3. Call Flow Details

The table below provides step-by-step analysis of various states as reflected at Alice's and Bob's end point when applied to example mentioned in the Motivation section **Section 3**

Only the relevant aspects of SDP media descriptions are captures for the sake of clarity.

Timeline	Alice's JS View	Bob's JS View
t+0	Local Set: m=audio (sendrecv)	Local Set: m=audio (sendrecv)
t+1	AddStream(Video)-sendonly	AddStream(Video)-sendonly
t+2	CreateOffer with m=audio,m=video[mid:1,sendonly]	CreateOffer with m=audio,m=video[mid:2,sendonly]
t+3	Generate ExpectedAnswer with m=audio,m=video[mid:1,inactive]	Generate ExpectedAnswer with m=audio,m=video[mic
t+4	Install ExpectedAnswer	Install ExpectedAnswer
t+5	Send video m-block [mid:1,sendonly]	Send video m-block [mid:2,sendonly]
t+6	Video m-block [mid:2,sendonly] Arrives	Video m-block [mid:1,sendonly] Arrives
t+7	Merge/Update Local Set	Merge/Update Local Set
t+8	LocalSet:m=audio,m=video[mid:1,sendonly],m=video[mid:2,sendonly]	Local Set:m=audio,m=video[mid:2,sendonly],m=video[mid
t+9	Install RemoteDescription	Install RemoteDescription
t+10	CreateAnswer m=audio,m=video[mid:1,sendonly] m=video[mid:2,recvonly]	CreateAnswer for m=audio,m=video[mid:1,recvonly] m=video[mid:2,sendonly]
t+11	Merge/Update Local Set	Merge/Update Local Set
t+12	Local Set:m=audio,m=video[mid:1,sendonly],m=video[mid:2 recvonly]	Local Set:m=audio,m=video[mid:2,sendonly],m=video[mid
t+13	Send video m-block [mid:2,recvonly]	Send video m-block [mid:1,recvonly]
t+14	Video m-block [mid:1,recvonly] Arrives	Video m-block [mid:2 recvonly] Arrives
t+15	Install updated Offer m=audio,m=video[mid:1 recvonly],m=video[mid:2,recvonly]	Generate Update Offer m=audio,m=video[mid:1 recvonly],m=video[mid:2,recvonly]
t+16	Create Answer with m=audio,m=video[mid:1 sendonly],m=video[mid:2 recvonly]	Create Answer with m=audio,m=video[mid:1 recvonly],m=video[mid:2,sendonly]
t+17	Compare Local Set and generated Answer	Compare Local Set and generated Answer
t+18	Local Set at t+12 matches with Answer m-blocks at t+16	Local Local Set at t+12 matches with Answer m-blocks

Timeline for Glareless Video Addition

---

5. Applicability Statement

As mentioned earlier, generating an Offer from the Answer created in response to m-block received from the Answerer might fail due to possible mismatches in the configurations between Offered m-block, applied Expected Answer and the actual Answer received from the Answerer.

On the other hand, this solution works best in all the cases where the first side can easily predict what the far side's answer will be. Given that the Offerer offering a capability that is supposed to fail on purpose is a rarity and also given the higher chances of Offer(s) being accepted in practice, we believe this solution should enable successful glareless media addition with high frequency.

---

6. Security Considerations

TBD

7. IANA Considerations

This document requires no actions from IANA.

8. References

8.1. Normative References

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

8.2. Informative References

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

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

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261, June 2002 (TXT).

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

[I-D.alvestrand-mmusic-msid] Alvestrand, H., "Cross Session Stream Identification in the Session Description Protocol," draft-alvestrand-mmusic-msid-02 (work in progress), December 2012 (TXT).

Authors' Addresses

Suhas Nandakumar
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
Email: snandaku@cisco.com

Cullen Jennings
Cisco
400 3rd Avenue SW, Suite 350
Calgary, AB T2P 4H2
Canada
Email: fluffy@iii.ca