SIPREC
Internet-Draft
Intended status: Standards Track
Expires: February 8, 2015

R. Ravindranath
Cisco
Parthasarathi. Ravindran
Nokia Networks
Paul. Kyzivat
Huawei

August 7, 2014

Session Initiation Protocol (SIP) Recording Metadata
draft-ietf-siprec-metadata-16

Abstract

Session recording is a critical requirement in many communications environments such as call centers and financial trading. In some of these environments, all calls must be recorded for regulatory, compliance, and consumer protection reasons. Recording of a session is typically performed by sending a copy of a media stream to a recording device. This document describes the metadata model as viewed by Session Recording Server(SRS) and the Recording metadata format.

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/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 8, 2015.

Copyright Notice

Table of Contents

1. Introduction

Session recording is a critical requirement in many communications environments such as call centers and financial trading. In some of these environments, all calls must be recorded for regulatory, compliance, and consumer protection reasons. Recording of a session is typically performed by sending a copy of a media stream to a recording device. This document focuses on the Recording metadata which describes the communication session. The document describes a metadata model as viewed by Session Recording Server and the Recording metadata format, the requirements for which are described in [RFC6341] and the architecture for which is described in [RFC7245].

2. Terminology

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 [RFC2119]. This document only uses these key words when referencing normative statements in existing RFCs."

3. Definitions

Metadata Model: An abstract representation of metadata using a Unified Modelling Language(UML) class diagram.

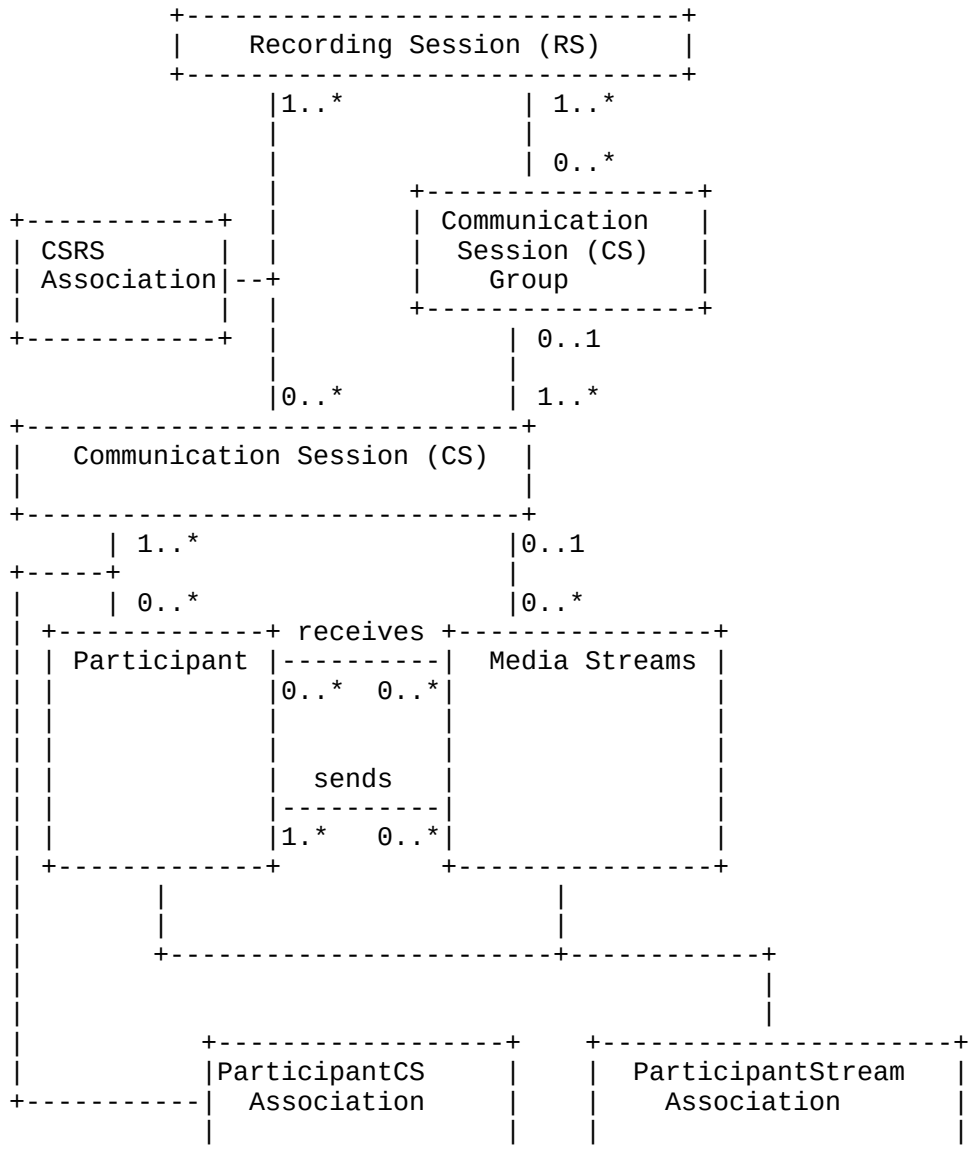
Metadata classes: Each block in the model represents a class. A class is a construct that is used as a blueprint to create instances(called objects) of itself. The description of each class also has representation of its attributes in a second compartment below the class name.

Attributes: Attributes represent the elements listed in each of the classes. The attributes of a class are listed in the second compartment below the class name. Each instance of class conveys values for these attributes which adds to the recording’s Metadata.

Linkages: Linkages represent the relationship between the classes in the model. Each represents a logical connection between classes(or objects) in class diagrams/ object diagrams. The linkages used in the Metadata model of this document are associations.

4. Metadata Model

Metadata is the information that describes recorded media and the CS to which they relate. The diagram below shows a model for Metadata as viewed by a Session Recording Server (SRS).



The Metadata model is a class diagram in Unified Modelling Language(UML). The model describes the structure of a metadata in general by showing the classes, their attributes, and the relationships among the classes. Each block in the model above represents a class. The linkages between the classes represents the relationships which can be associations or Composition. The metadata is conveyed from SRC to SRS.

The model allows the capture of a snapshot of a recording's Metadata at a given instant in time. Metadata changes to reflect changes in what is being recorded. For example, if a participant joins a conference, then the SRC sends the SRS a snapshot of metadata having that participant information (with attributes like name/AoR pair and associate-time.)

Some of the metadata is not required to be conveyed explicitly from the SRC to the SRS, if it can be obtained contextually by the SRS(e.g., from SIP or SDP signalling).

5. Recording Metadata Format

This section gives an overview of the Recording Metadata Format. Some data from the metadata model is assumed to be made available to the SRS through Session Description Protocol (SDP)[RFC4566], and therefore this data is not represented in the XML document format specified in this document. SDP attributes describe different media formats like audio, video. The other metadata attributes, such as participant details, are represented in a new Recording specific XML document of type 'application/rs-metadata+xml'. The SDP label attribute [RFC4574] provides an identifier by which a metadata XML document can refer to a specific media description in the SDP sent from the SRC to the SRS.

The XML document format can be used to represent either the complete metadata or a partial update to the metadata. The latter includes only elements that have changed compared to the previously reported metadata.

5.1. XML data format

Every recording metadata XML document MUST contain a <recording> element. The <recording> element acts as a container for all other elements in this XML document.

A recording object is a XML document. It MUST have the XML declaration and it SHOULD contain an encoding declaration in the XML declaration, e.g., "<?xml version='1.0' encoding='UTF-8'?>". If the charset parameter of the MIME content type declaration is present and it is different from the encoding declaration, the charset parameter takes precedence.

Every application conforming to this specification MUST accept the UTF-8 character encoding to ensure the minimal interoperability.

Syntax and semantic errors in an XML document should be reported to the originator using application specific mechanisms.

5.1.1. Namespace

The namespace URI for elements defined by this specification is a Uniform Resource Namespace (URN) [RFC2141], using the namespace identifier 'ietf' defined by [RFC2648] and extended by [RFC3688].

The URN is: urn:ietf:params:xml:ns:recording

5.1.2. recording

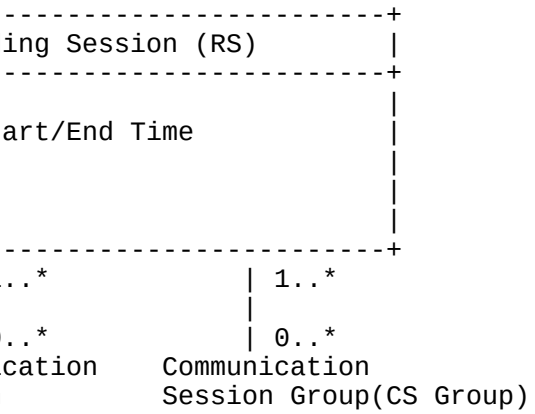
The <recording> element MUST contain an xmlns namespace attribute with value as urn:ietf:params:xml:ns:recording. One recording element MUST be present in every recording metadata XML document.

A recording element MAY contain a <dataMode> element indicating whether the XML document is a complete document or a partial update. If no <dataMode> element is present then the default value is "complete".

6. Recording Metadata classes

This section describes each class of the metadata model, and the attributes of each class. This section also describes how different classes are linked and the XML element for each of them.

6.1. Recording Session



Each instance of a Recording Session class (namely the Recording Session Object) represents a SIP session created between an SRC and SRS for the purpose of recording a Communication Session.

6.1.1. Attributes

A Recording Session class has the following attributes:

- Start/End Time - Represents the Start/End time of a Recording Session object.

6.1.2. Linkages

Each instance of Recording Session has:

- Zero or more instances of Communication Session Group (CSG).
- Zero or more instances of Communication Session objects.

CSs and CSGs are optional to accommodate persistent recording, where there may sometimes be none.

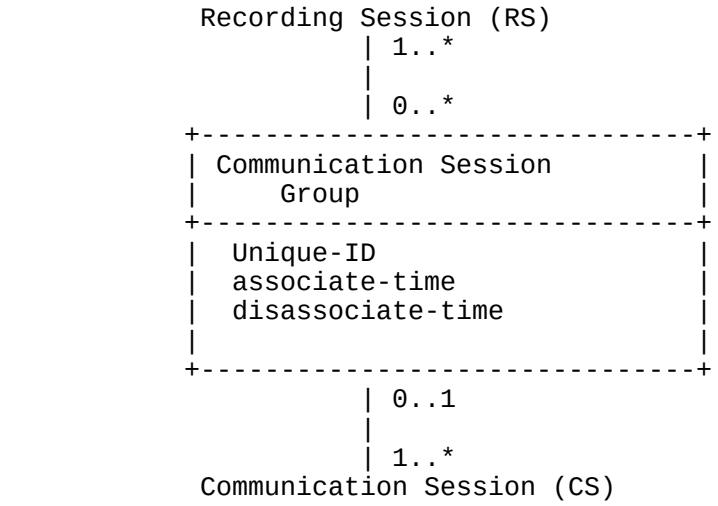
6.1.3. XML element

A Recording Session object is represented by a <recording> XML element. That in turn relies on the SIP/SDP session with which the XML document is associated to provide some of the attributes of the Recording Session element.

Start and End time value are derivable from Date header(if present in SIP message) in RS. In cases where Date header is not present,

Start/End time are derivable from the time at which SRS receives the notification of SIP message to setup RS / disconnect RS.

6.2. Communication Session Group



One instance of a Communication Session Group class (namely the Communication Session Group object) provides association or linking of Communication Sessions.

6.2.1. Attributes

A CS Group has the following attributes:

- Unique-ID - This Unique-ID is to group different CSs that are related. SRC (or SRS) is responsible for ensuring the uniqueness of Unique-ID in case multiple SRC interacts with the same SRS. The mechanism by which SRC groups the CS is outside the scope of SIPREC.
- Associate-time - Associate-time for CS-Group shall be calculated by SRC as the time when a grouping is formed. The rules that determine how a grouping of different Communication Session objects is done by SRC is outside the scope of SIPREC.
- Disassociate-time - Disassociate-time for CS-Group shall be calculated by SRC as the time when the grouping ends

6.2.2. Linkages

The linkages between Communication Session Group class and other classes are associations. A communication Session Group is

associated with RS and CS in the following manner:

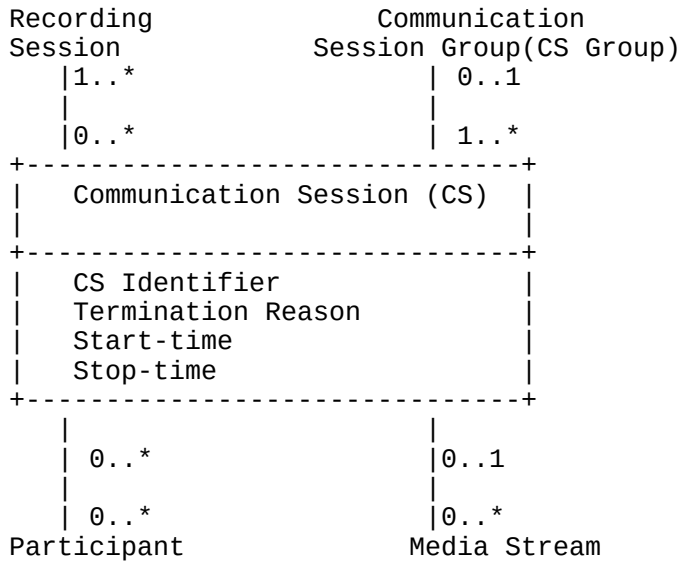
- o There are one or more Recording Session objects per Communication Session Group.
- o Each Communication Session Group object has to be associated with one or more RS [Here each RS can be setup by the potentially different SRCs]
- o There are one or more Communication Sessions per CS Group [e.g. Consult Transfer]

6.2.3. XML element

The <group> element is an optional element provides the information about the communication session group

Each communication session group (CSG)object is represented using one group element. Each <group> element has a unique 'group-id' attribute which uniquely identifies the CSG.

6.3. Communication Session



A Communication Session class and its object in the metadata model represents a Communication Session and its properties needed as seen by the SRC.

6.3.1. Attributes

A communication Session class has the following attributes:

- Termination Reason - This represents the reason why a CS was terminated. The communication session MAY contain a Call Termination Reason. This MAY be derived from SIP Reason header [RFC3326] of CS.
- CS Identifier - This attribute is used to uniquely identify a CS.
- Start-time - This optional attribute represents start time of CS as seen by SRC
- Stop-time - This optional attribute represents stop time of CS as seen by SRC

This document does not specify attributes relating to what should happen to a recording of a CS after it has been delivered to the SRS. (E.g., how long to retain the recording, what access controls to apply.) The SRS is assumed to behave in accordance with policy. The ability for the SRC to influence this policy is outside the scope of this document. However if there are implementations where SRC has enough information, this could be sent as Extension Data attached to the CS

6.3.2. Linkages

A Communication Session is linked to CS-Group, Participant, Media Stream and Recording Session classes using the association relationship. Association between CS and Participant allows:

- CS to have zero or more participants
- Participant is associated with zero or more CSs. This includes participants who are not directly part of any CS. An example of such a case is participants in a premixed media stream. The SRC may have knowledge of such Participants, yet not have any signaling relationship with them. This might arise if one participant in CS is a conference focus. To summarize, even if the SRC does not have direct signalling relationships with all participants in a CS, it should nevertheless create a Participant object for each participant that it knows about.
- The model also allows participants in CS that are not participants in the media. An example is the identity of a 3pcc controller that has initiated a CS to two or more participants of the CS. Another example is the identity of a conference focus. Of course a focus is probably in the media, but since it may only be there as a mixer, it may not report itself as a participant in any of the media streams.

Association between CS and Media Stream allows:

- o A CS to have zero or more Streams
- o A stream can be associated with at most one CS. Stream in persistent RS is not required to be associated with any CS before CS is created and hence the zero association is allowed.

Association between CS and RS allows:

- o Each instance of RS has Zero or more instances of Communication Session objects.
- o Each CS has to be associated with one more RS [Here each RS can be potentially setup by different SRCs]

6.3.3. XML element

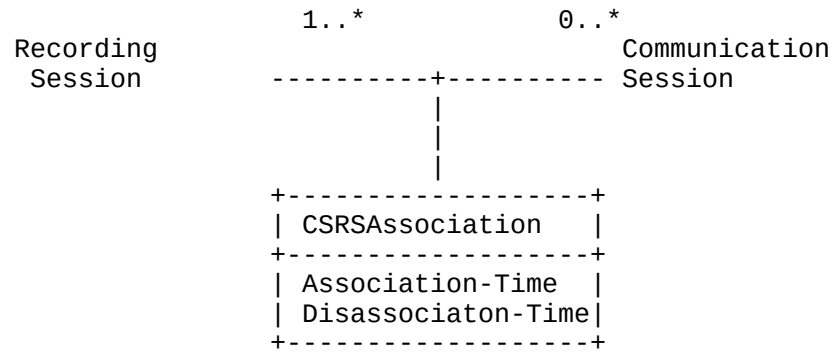
The <session> element provides the information about the Communication Session

Each communication session(CS) object is represented by one session element. Each session element has unique 'session_id' attribute which helps to uniquely identify the CS.

The XML <reason> element MAY be included in metadata to represent a CS Termination Reason. There MAY be multiple instances of the XML <reason> element inside a session element. The <reason> XML element has 'protocol' as an attribute, which indicates the protocol from which the reason string is derived. The default value for protocol attribute is "SIP". The <reason> element can be derived from a SIP Reason header in the CS.

A <group-ref> element MAY be present to indicate the group to which the enclosing session belongs.

6.4. CSRSAssociation



The CSRS Association class describes the association of a CS to an RS for a period of time. A single CS may be associated with different RSs (perhaps by different SRCs) and may be associated and dissociated several times.

6.4.1. Attributes

CSRS association class has the following attributes:

- Associate-time - associate-time is calculated by SRC as the time it sees a CS is associated to a RS
- Disassociate-time- Disassociate-time is calculated by SRC as the time it see a CS disassociate from a RS.

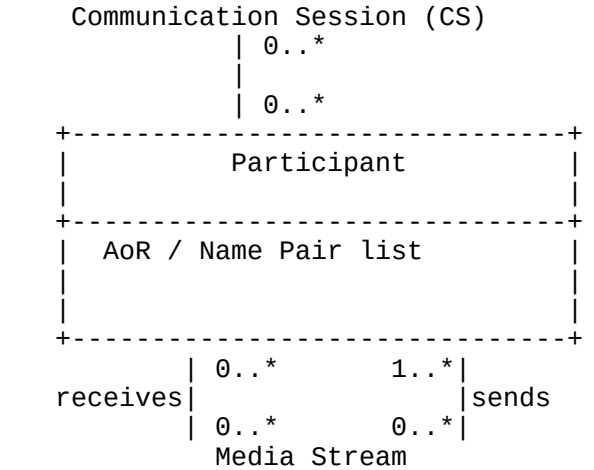
6.4.2. Linkages

CSRS association class is linked to CS and RS classes.

6.4.3. XML element

The <sessionrecordingassoc> XML element represents the CSRS association object. The 'session_id' attribute is used to uniquely identify this element and link with a specific session. The recording object is implicitly defined by the enclosing <recording> element.

6.5. Participant



A Participant class and its objects has information about a device that is part of a CS and/or contributes/consumes media stream(s) belonging to a CS.

6.5.1. Attributes

Participant has a single defined attribute:

- AoR / Name pair list - This attribute is a list of Name/AoR tuples. An AoR MAY be a SIP/SIPS/TEL URI. Name represents Participant name(SIP display name) or dialed number (DN) (when known). Multiple tuples are allowed for cases where a participant has more than one AoR. (For example a P-Asserted-identity header [RFC3325] can have both SIP and TEL URIs.)

This document does not specify other attributes relating to participant e.g. Participant Role, Participant type. An SRC which has information of these attributes can indicate the same as part of extension data to Participant from SRC to SRS.

6.5.2. Linkages

The participant class is linked to MS and CS class using association relationship. The association between participant and Media Stream allows:

- Participant to receive zero or more media streams
- Participant to send zero or more media streams. (Same participant provides multiple streams e.g. audio and video)
- Media stream to be received by zero or more participants. Its possible, though perhaps unlikely, that a stream is generated but sent only to the SRC and SRS, not to any participant. E.g. In conferencing where all participants are on hold and the SRC is collocated with the focus. Also a media stream may be received by multiple participants (e.g. Whisper calls, side conversations).
- Media stream to be sent by one or more participants (pre-mixed streams).

Example of a case where a participant receives zero or more streams - a supervisor may have side conversation with agent, while agent converses with customer.

6.5.3. XML element

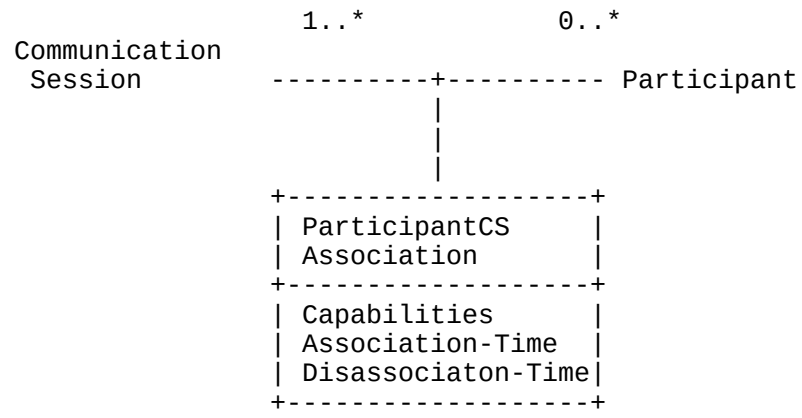
A <participant> element represents a Participant object.

Participant MUST have a NameID complex element which contains AoR as attribute and Name as element. AOR element is SIP/SIPS URI FQDN or

IP address which represents the user. name is an optional element to represent display name.

Each participant has a unique 'participant_id' attribute. This MUST be used for all references to a participant within a CSG, and MAY be used to reference the same participant more globally. (The decision to use a participant_id across multiple CSGs or recording sessions is at the discretion of the implementer.)

6.6. ParticipantCSAssociation



The Participant CS Association class describes the association of a Participant to an CS for a period of time. A participant may be associated and dissociated from a CS several times. (For example, connecting to a conference, then disconnecting, then connecting again.)

6.6.1. Attributes

ParticipantCS association class has the following attributes:

- Associate-time - associate-time is calculated by SRC as the time it sees a participant is associated to CS
- Disassociate-time- Disassociate-time is calculated by the SRC as the time it sees a participant disassociate from a CS. It is possible that a given participant can have multiple associate/ disassociate times within given communication session.
- Capabilities - An optional attribute describing the capabilities of a participant in a CS, as defined in [RFC3840]. Each participant may have zero or more capabilities. A participant may

use different capabilities depending on the role it plays at a particular instance. For example if a participant moves across different CSs (e.g., due to transfer) or is simultaneously present in different CSs its role may be different and hence the capability used.

6.6.2. Linkages

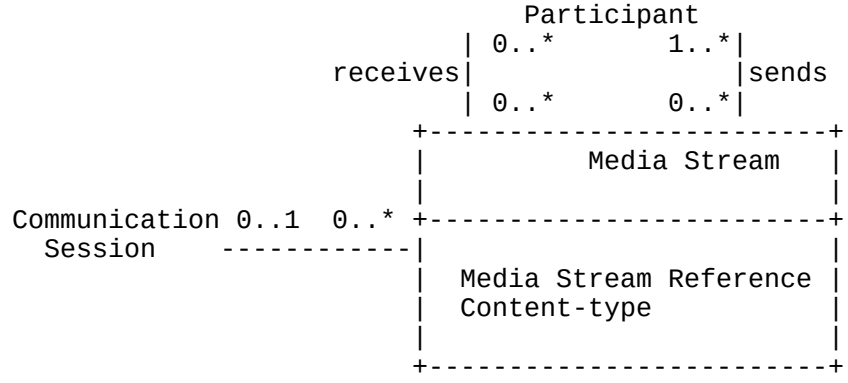
The participantCS association class is linked to participant and CS classes.

6.6.3. XML element

The <participantsessionassoc> XML element represents a participantCS association object. The 'participant_id' and 'session_id' attributes are used to uniquely identify this element.

NOTE: RFC 4235 encoding shall be used to represent capabilities attribute in XML.

6.7. Media Stream



A Media Stream class (and its objects) has the properties of media as seen by SRC and sent to SRS. Different snapshots of a media stream object may be sent whenever there is a change in media (e.g. direction change like pause/resume and/or codec change and/or participant change.).

6.7.1. Attributes

A Media Stream class has the the following attributes:

- Media Stream Reference - In implementations this references an m-line
- Content - The content of an MS element will be described in terms of value from the [RFC4796] registry.

The metadata model should include media streams that are not being delivered to the SRS. Examples include cases where SRC offered certain media types but SRS chooses to accept only a subset of them OR an SRC may not even offer a certain media type due it its restrictions to record

6.7.2. Linkages

A Media Stream is linked to participant and CS classes using the association relationship. The details of association with the Participant are described in the Participant class section. The details of association with CS is mentioned in the CS section.

6.7.3. XML element

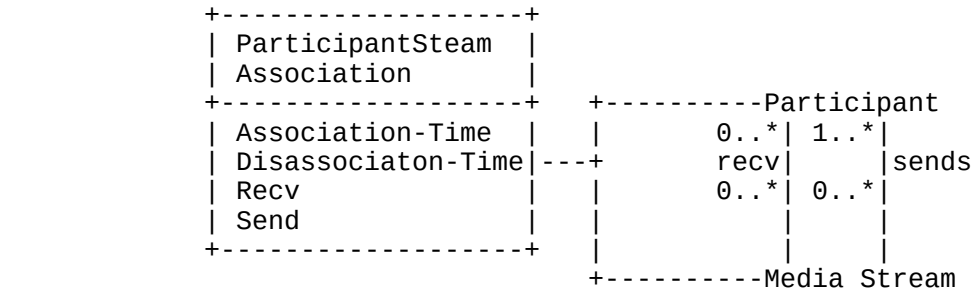
The <stream> element represents a Media Stream object. Stream element indicates SDP media lines associated with the session and participants.

The <label> element within the <stream> element references an SDP "a=label" attribute that identifies an m-line within the RS SDP. That m-line carries the media stream from the SRC to the SRS.

Each stream element has unique ’stream_id’ attribute which helps to uniquely identify stream and ’session_id’ attribute to associate the stream with specific session element.

If the SRC wishes to convey the Content-type to the SRS, it does so by including an ’a=content’ attribute with the m-line in the RS SDP.

6.8. ParticipantStream Association



A ParticipantStream association class describes the association of a Participant to a Media Stream for a period of time, as a sender or as a receiver, or both.

6.8.1. Attributes

A participantStream association class has the following attributes:

- Associate-Time: This attribute indicates the time a Participant started contributing to a Media Stream
- Disassociate-Time: This attribute indicates the time a Participant stopped contributing to a Media Stream
- Recv: This attribute indicates whether a Participant is receiving a media stream or not. This attribute has a value which points to a stream represented by its Unique_id. The presence of this attribute indicates that a participant is receiving a stream represented by the Unique_id. If due to changes in CS(like hold) the participants stops receiving a stream, a snapshot MUST be sent from SRC to SRS with no Recv element for that stream.
- Send: This attribute indicates whether a participant is contributing to a stream or not. This attribute has a value which points to stream represented by its unique_id. The presence of this attribute indicates that a participant is contributing to a stream represented by the Unique_id. If due to changes in CS if a participant stops contributing to a stream, a snapshot MUST be sent from SRC to SRS with no Send element for that stream.

6.8.2. Linkages

The participantStream association class is linked to participant and Stream classes.

6.8.3. XML element

The <participantstreamassoc> XML element represents the participant to stream association object. This XML element is used to represent a snapshot of a participant association with a stream. The send and recv XML elements MUST be used to indicate whether a participant is contributing to a stream or receiving a stream. There MAY be multiple instances of the send and recv XML elements inside a particpantstreamassoc element. If a metadata snapshot is sent with a participantstreamassoc that does not have any send and recv elements, it means that participant is neither contributing to any streams nor receiving any streams.

6.9. associate-time/disassociate-time

The XML <associate-time> and <disassociate-time> elements contain strings indicating the date and time of the status change of this tuple. The value of these elements MUST follow the IMPP datetime format [RFC3339]. Timestamps that contain 'T' or 'Z' MUST use the capitalized forms. At a time, any of the time tuple associate-time or disassociate-time MAY exist in the element namely group, session, participant and not both timestamp at the same time.

As a security measure, the timestamp element SHOULD be included in all tuples unless the exact time of the status change cannot be determined.

6.10. Unique ID format

A Unique id is generated in two steps:
- the UUID is created using [RFC4122]
- the UUID is encoded using base64 as defined in [RFC4648]

The above mentioned unique-id mechanism SHOULD be used for each metadata element. Multiple SRCs can refer to the same element/UUID (how each SRC learns the UUID here is out of scope of SIPREC)

6.11. Metadata version Indicator

This section defines a version indicator for metadata XML.

This version value allows the SRS to know the exact metadata XML schema used by the SRC. This document describes version 1. Implementations may not interoperate if the version implemented by the sender is not known by the receiver. No negotiation of versions is provided. There is no significance to the version number although documents which update or obsolete this document (possibly including drafts of such documents) should include a higher version number if

the metadata XML schema changes.

7. SIP Recording Metadata Example

7.1. Complete SIP Recording Metadata Example

The following example provides all the tuples involved in Recording Metadata XML body.

```
<?xml version="1.0" encoding="UTF-8"?>
       <recording xmlns='urn:ietf:params:xml:ns:recording:1'>
              <dataMode>complete</dataMode>
          <group group_id="7+OTCyoxTmqmqyA/1weDAg==">
            <associate-time>2010-12-16T23:41:07Z</associate-time>
             <!-- Standardized extension -->
            <call-center xmlns='urn:ietf:params:xml:ns:callcenter'>
              <supervisor>sip:alice@atlanta.com</supervisor>
            </call-center>
            <mydata xmlns='http://example.com/my'>
              <structure>FOO!</structure>
              <whatever>bar</whatever>
            </mydata>
          </group>
          <session session_id="hVpd7YQgRW2nD22h7q60JQ==">
            <group-ref>7+OTCyoxTmqmqyA/1weDAg==
            </group-ref>
             <!-- Standardized extension -->
             <uniqueid>FaXHlc+3WruaroDaNE87am==</uniqueid>
            <structure>FOO!</structure>
            <whatever>bar</whatever>
          </session>
          <sessionrecordingassoc session_id="hVpd7YQgRW2nD22h7q60JQ==">
          <associate-time>2010-12-16T23:41:07Z</associate-time>
          </sessionrecordingassoc>
          <participant
             participant_id="srfBElmCRp2QB23b7Mpk0w==">
             <nameID aor=sip:bob@biloxi.com>
                          <name xml:lang="it">Bob B</name>
                        </nameID>
             <!-- Standardized extension -->
            <structure>FOO!</structure>
            <whatever>bar</whatever>
          </participant>
          <participantsessionassoc
             participant_id="srfBElmCRp2QB23b7Mpk0w=="
             session_id="hVpd7YQgRW2nD22h7q60JQ==">
```

```
            <associate-time>2010-12-16T23:41:07Z</associate-time>
          </participantsessionassoc>
          <participantstreamassoc
             participant_id="srfBElmCRp2QB23b7Mpk0w==">
             <send>i1Pz3to5hGk8fuXl+PbwCw==</send>
             <send>UAAMm5GRQKSCMVvLyl4rFw==</send>
             <recv>8zc6e0lYTlWIINA6GR+3ag==</recv>
             <recv>EiXGlc+4TruqqoDaNE76ag==</recv>
          </participantstreamassoc>
          <participant
              participant_id="zSfPoSvdSDCmU3A3TRDxAw==">
              <nameID aor=sip:Paul@biloxy.com>
                          <name xml:lang="it">Paul</name>
                         </nameID>
              <!-- Standardized extension -->
            <structure>FOO!</structure>
            <whatever>bar</whatever>
          </participant>
          <participantsessionassoc
              participant_id="zSfPoSvdSDCmU3A3TRDxAw=="
              session_id="hVpd7YQgRW2nD22h7q60JQ==">
              <associate-time>2010-12-16T23:41:07Z</associate-time>
          </participantsessionassoc>
          <participantstreamassoc
              participant_id="zSfPoSvdSDCmU3A3TRDxAw==">
              <send>8zc6e0lYTlWIINA6GR+3ag==</send>
              <send>EiXGlc+4TruqqoDaNE76ag==</send>
              <recv>UAAMm5GRQKSCMVvLyl4rFw==</recv>
              <recv>i1Pz3to5hGk8fuXl+PbwCw==</recv>
          </participantstreamassoc>
          <stream stream_id="UAAMm5GRQKSCMVvLyl4rFw=="
              session session_id="hVpd7YQgRW2nD22h7q60JQ==">
              <label>96</label>
          </stream>
          <stream stream_id="i1Pz3to5hGk8fuXl+PbwCw=="
              session_id="hVpd7YQgRW2nD22h7q60JQ==">
              <label>97</label>
          </stream>
          <stream stream_id="8zc6e0lYTlWIINA6GR+3ag=="
              session_id="hVpd7YQgRW2nD22h7q60JQ==">
              <label>98</label>
          </stream>
          <stream stream_id="EiXGlc+4TruqqoDaNE76ag=="
              session_id="hVpd7YQgRW2nD22h7q60JQ==">
              <label>99</label>
          </stream>
      </recording>
```

SIP Recording Metadata Example XML body

7.2. Partial Update of Recording metadata XML body

The following example provides partial update in Recording Metadata XML body for the above example. The example has a snapshot that carries the disassociate-time for a participant from a session.

```
 <?xml version="1.0" encoding="UTF-8"?>
    <recording xmlns='urn:ietf:params:xml:ns:recording:1'>
      <dataMode>partial</dataMode>
      <participant
         participant_id="srfBElmCRp2QB23b7Mpk0w==">
         <name ID=sip:bob@biloxi.com>
                  <name xml:lang="it">Bob R</name>
                </nameID>
          <structure>FOO!</structure>
          <whatever>bar</whatever>
      </participant>
      <participantsessionassoc
            participant_id="srfBElmCRp2QB23b7Mpk0w=="
            session_id="hVpd7YQgRW2nD22h7q60JQ==">
          <disassociate-time>2010-12-16T23:41:07Z</disassociate-time>
       </participantsessionassoc>
    </recording>
```

Partial update of SIP Recording Example XML body

8. XML Schema definition for Recording metadata

This section defines XML schema for Recording metadata document

```
<?xml version="1.0" encoding="UTF-8"?>
   <xs:schema targetNamespace="urn:ietf:params:xml:ns:recording:1"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:tns="urn:ietf:params:xml:ns:recording"
      elementFormDefault="qualified"
      attributeFormDefault="unqualified">
     <!-- This import brings in the XML language attribute xml:lang-->
     <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>
     <xs:element name="recording" type="tns:recording"/>
     <xs:complexType name="recording">
         <xs:sequence>
             <xs:element name="datamode" type="tns:dataMode"
```

```
                minOccurs="0"/>
          <xs:element name="group" type="tns:group"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="session" type="tns:session"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="participant" type="tns:participant"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="stream" type="tns:stream"
                minOccurs="0" maxOccurs="unbounded"/>
           <xs:element name="sessionrecordingassoc"
                type="tns:sessionrecordingassoc"
                minOccurs="0" maxOccurs="unbounded"/>
           <xs:element name="participantsessionassoc"
                type="tns:participantsessionassoc"
                minOccurs="0" maxOccurs="unbounded"/>
           <xs:element name="participantstreamassoc"
                type="tns:participantstreamassoc"
                minOccurs="0" maxOccurs="unbounded"/>
           <xs:any namespace='##other'
                    minOccurs='0'
                    maxOccurs='unbounded'
                    processContents='lax'/>
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="group">
        <xs:sequence>
          <xs:element name="associate-time" type="xs:dateTime"
             minOccurs="0"/>
          <xs:element name="disassociate-time" type="xs:dateTime"
             minOccurs="0"/>
         <xs:any namespace='##other'
                    minOccurs='0'
                    maxOccurs='unbounded'
                    processContents='lax'/>
        </xs:sequence>
        <xs:attribute name="group_id" type="xs:base64Binary"
            use="required"/>
      </xs:complexType>

      <xs:complexType name="session">
        <xs:sequence>
          <xs:element name="reason" type="tns:reason" minOccurs="0"
                   maxOccurs="unbounded"/>
          <xs:element name="group-ref" type="xs:base64Binary"
                minOccurs="0" maxOccurs="1"/>
          <xs:element name="start-time" type="xs:dateTime"
                minOccurs="0" maxOccurs="1"/>
          <xs:element name="stop-time" type="xs:dateTime"
```

```
                minOccurs="0" maxOccurs="1"/>
              <xs:any namespace='##other'
                      minOccurs='0'
                      maxOccurs='unbounded'
                      processContents='lax'/>
          </xs:sequence>
          <xs:attribute name="session_id" type="xs:base64Binary"
               use="required"/>
      </xs:complexType>
      <xs:complexType name="sessionrecordingassoc">
          <xs:sequence>
            <xs:element name="associate-time" type="xs:dateTime"
                   minOccurs="0"/>
            <xs:element name="disassociate-time" type="xs:dateTime"
                   minOccurs="0"/>
            <xs:any namespace='##other'
                      minOccurs='0'
                      maxOccurs='unbounded'
                      processContents='lax'/>
          </xs:sequence>
          <xs:attribute name="session_id" type="xs:base64Binary"
               use="required"/>
      </xs:complexType>
      <xs:complexType name="participant">
          <xs:sequence>
              <xs:element name="nameID" type="tns:nameID"
                      maxOccurs='unbounded'/>
              <xs:element name="param" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                   <xs:attribute name="pname" type="xs:string" use="required"/>
                   <xs:attribute name="pval" type="xs:string" use="required"/>
                 </xs:complexType>
              </xs:element>
           <xs:any namespace='##other'
                      minOccurs='0'
                      maxOccurs='unbounded'
                      processContents='lax'/>
          </xs:sequence>
          <xs:attribute name="participant_id" type="xs:base64Binary"
               use="required"/>
    </xs:complexType>
      <xs:complexType name="participantsessionassoc">
          <xs:sequence>
          <xs:element name="associate-time" type="xs:dateTime"
                   minOccurs="0"/>
          <xs:element name="disassociate-time" type="xs:dateTime"
                   minOccurs="0"/>
            <xs:any namespace='##other'
```

```
                    minOccurs='0'
                    maxOccurs='unbounded'
                    processContents='lax'/>
          </xs:sequence>
          <xs:attribute name="participant_id" type="xs:base64Binary"
               use="required"/>
          <xs:attribute name="session_id" type="xs:base64Binary"
               use="required"/>
      </xs:complexType>
      <xs:complexType name="participantstreamassoc">
          <xs:sequence>
              <xs:element name="send" type="xs:base64Binary"
                minOccurs="0" maxOccurs="unbounded"/>
              <xs:element name="recv" type="xs:base64Binary"
                minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="associate-time" type="xs:dateTime"
                   minOccurs="0"/>
             <xs:element name="disassociate-time" type="xs:dateTime"
                   minOccurs="0"/>
                <xs:any namespace='##other'
                      minOccurs='0'
                      maxOccurs='unbounded'
                      processContents='lax'/>
          </xs:sequence>
          <xs:attribute name="participant_id" type="xs:base64Binary"
               use="required"/>
      </xs:complexType>
      <xs:complexType name="stream">
          <xs:sequence>
            <xs:element name="label" type="xs:string"
                minOccurs="0" maxOccurs="1"/>
             <xs:any namespace='##other'
                      minOccurs='0'
                      maxOccurs='unbounded'
                      processContents='lax'/>
          </xs:sequence>
          <xs:attribute name="stream_id" type="xs:base64Binary"
               use="required"/>
          <xs:attribute name="session_id" type="xs:base64Binary"/>
      </xs:complexType>
        <xs:simpleType name="dataMode">
            <xs:restriction base="xs:string">
               <xs:enumeration value="complete"/>
                       <xs:enumeration value="partial"/>
            </xs:restriction>
      </xs:simpleType>
      <xs:complexType name="nameID">
       <xs:sequence>
```

```
        <xs:element name="name" type ="tns:name" minOccurs="0"
                          maxOccurs="1"/>
      </xs:sequence>
      <xs:attribute name="aor" type="xs:anyURI" use="required"/>
    </xs:complexType>
   <xs:complexType name="name">
   <xs:simpleContent>
          <xs:extension base="xs:string">
           <xs:attribute ref="xml:lang" use="optional"/>
          </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="reason">
      <xs:simpleContent>
       <xs:extension base="xs:string">
          <xs:attribute type="xs:short" name="cause" use="required"/>
          <xs:attribute type="xs:string" name="protocol" default="SIP"/>
       </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
</xs:schema>
```

9. Security Considerations

The metadata information sent from SRC to SRS MAY reveal sensitive information about different participants in a session. For this reason, it is RECOMMENDED that a SRC use a strong means for authentication and metadata information protection and that it apply comprehensive authorization rules when using the metadata format defined in this document. The below section discusses each of these aspects in more detail.

9.1. Connection Security

It is RECOMMENDED that a SRC authenticate the SRS using the normal SIP authentication mechanisms, such as Digest as defined in Section 22 of [RFC3261]. The mechanism used for conveying the metadata information MUST ensure integrity and SHOULD ensure confidentially of the information. In order to achieve these, an end-to-end SIP encryption mechanism, such as S/MIME described in [RFC3261], SHOULD be used.

If a strong end-to-end security means (such as above) is not available, it is RECOMMENDED that a SRC use mutual hop-by-hop Transport Layer Security (TLS) authentication and encryption mechanisms described in "SIPS URI Scheme" and "Interdomain Requests"

of [RFC3261].

This document describes an extensive set of metadata that may be recorded by the SRS. Most of the metadata could be considered private data. Some implementations may have SRC choose parts of Metadata that can be sent to SRS. In other cases, SRCs may send metadata that is not appropriate for the SRS to record. What of this metadata is actually recorded by the SRS must be carefully considered (a "retention policy") to balance privacy concerns with usability. Implementations MUST control what metadata is recorded, and MUST NOT save metadata sent by the SRC that does not conform to the retention policy of the SRS.

10. IANA Considerations

This specification registers a new XML namespace, and a new XML schema.

10.1. SIP recording metadata Schema Registration

URI: urn:ietf:params:xml:ns:recording

Registrant Contact: IETF SIPREC working group, Ram mohan R(rmohanr@cisco.com)

XML: the XML schema to be registered is contained in Section 8.

Its first line is <?xml version="1.0" encoding="UTF-8"?> and its last line is </xs:schema>

11. Acknowledgement

We wish to thank John Elwell, Henry Lum, Leon Portman, De Villers, Andrew Hutton(Siemens-Enterprise), Deepanshu Gautam(Huawei),Charles Eckel(Cisco), Muthu Arul Mozhi (Cisco), Michael Benenson(Cisco), Hadriel Kaplan (ACME), Brian Rosen, Scott Orton(Broadsoft), Ofir Roth (NICE), Mary Barnes(Polycom), Ken Rehor(Cisco) for their valuable comments and inputs.

We wish to thank Joe Hildebrand(Cisco), Peter Saint-Andre(Cisco), Matt Miller(Cisco) for the valuable XML related guidance and Martin Thompson for validating the XML schema and providing comments on the same.

12. References

12.1. Normative References

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

[RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997.

[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.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC3339] Klyne, G., Ed. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

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

[RFC4574] Levin, O. and G. Camarillo, "The Session Description Protocol (SDP) Label Attribute", RFC 4574, August 2006.

[RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796, February 2007.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, August 2004.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, July 2005.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

12.2. Informative References

[RFC6341] Rehor, K., Portman, L., Hutton, A., and R. Jain, "Use Cases and Requirements for SIP-Based Media Recording (SIPREC)", RFC 6341, August 2011.

[RFC7245] Hutton, A., Portman, L., Jain, R., and K. Rehor, "An Architecture for Media Recording Using the Session Initiation Protocol", RFC 7245, May 2014.

[RFC2648] Moats, R., "A URN Namespace for IETF Documents", RFC 2648, August 1999.

[RFC3326] Schulzrinne, H., Oran, D., and G. Camarillo, "The Reason Header Field for the Session Initiation Protocol (SIP)", RFC 3326, December 2002.

[RFC3325] Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002.

Authors' Addresses

Ram Mohan Ravindranath
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: rmohanr@cisco.com

Parthasarathi Ravindran
Nokia Networks
Bangalore, Karnataka
India

Email: partha@parthasarathi.co.in

Paul Kyzivat
Huawei
Hudson, MA
USA

Email: pkyzivat@alum.mit.edu

SIPREC
Internet-Draft
Intended status: Informational
Expires: February 28, 2015

M. Yan
P. Kyzivat
Huawei

August 27, 2014

Overview for MSRP Recording based on SIPREC
draft-yan-siprec-msrp-recording-02

Abstract

SIPREC is capable of recording interactive text media that is transmitted via RTP. However that format is not commonly used for message or chat scenarios. There is also a need for recording text media carried via MSRP. One case of note is exchange of text between hearing-impaired users and emergence service bureaus. Also, recording support is needed for MSRP used in chat conferences and multimedia conferences.

This document describes how to achieve MSRP channel recording within the mechanism of SIP Recording (SIPREC).

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

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/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 28, 2015.

Table of Contents

1. Introduction

SIPREC is capable of recording interactive text media that is transmitted via RTP, as defined by [RFC4103]. However that format is not commonly used for message or chat scenarios. There is also a need for recording text media carried via MSRP. One case of note is exchange of text between hearing-impaired users and emergence service bureaus. Also, recording support is needed for MSRP used in chat conferences (as defined by [I-D.ietf-simple-chat]) and multimedia conferences (as defined by [RFC4597]).

Instant message media is carried by a variety of protocols such as IRC, MSRP and XMPP/JINGLE. The SIP based MSRP protocol (as defined by [RFC4975] and [RFC4976]) supports the delivery of messages and files from one SIP UA to another. When a SIPREC SRC is recording a CS that contains an MSRP channel, it may want to record the messages passing over that channel. To gain access to the messages, the SRC may act as an MSRP client, relay, or switch. The SRC needs to replicate and deliver the messages over an MSRP channel within a Recording Session (RS) to an SRS. The replicated content could be in Message/CPIM format containing plain text, HTML, images, etc. In this document, file delivering sessions have not yet been considered. Other instant message protocols, like IRC or XMPP, are out of scope.

This document describes how MRSP sessions are established between an SRC and SRS, and used for conveying the replicated MSRP Media, and also specifies metadata that describes the recorded MSRP sessions. A Recording Session employing MSRP is established using the normal procedures for establishing INVITE initiated dialogs [RFC3261] and uses SDP [RFC4566] for describing the media to be used during the session as described by the SIPREC Architecture [RFC7245].

1.1. EDITOR NOTES

This version addresses comments received on the -01 version, both on the mailing list and at IETF90. The following is my list of things to address:

- Define a new MIME type that is used to wrap the CS MSRP messages that are being recorded. This allows the original message to be left as-is, so it is always clear what it was. While CPIM could be used for this, defining a new type will allow capturing other necessary metadata.

- Need to further consider the need to track message timing. Can the timing of messages received by the SRS on the RS MSRP stream be considered a sufficient proxy for the timing of messages in the CS, or should we explicitly pass timestamps of messages as received on the CS? (The issue was raised but not decided.)

- We need to clarify that there is no guarantee that messages received on the CS have been recorded.

- It was agreed that there is no need to record the MSRP URIs that are used to establish the CS MSRP session.

- It is important that we maintain a 1:1 consistency between MSRP MESSAGE-IDs used in recorded CS sessions and the MESSAGE-IDs used in the RS. But we should not violate MSRP by using the same MESSAGE-IDs. We came up with the idea of adding an SRC-specific prefix to the CS message ids to create unique ones for the RS. This should be done in a standard way so that the SRS can recover the original CS message ids, in order to support correlation across redundant SRCs.

- Will need to work out the details of what happens when a CS MSRP session is terminated with an incomplete message. It will be necessary to send the incomplete message to the SRS, but must it appear to be incomplete within the SRS MSRP session?

- There are a variety of reasons why the SRC may not want, or be able to, record individual messages in the CS session. (One example is because the message size is greater than the maximum indicated by the SRS. Another is because the mime type of the message is a type that the SRS did not indicate support for.) There should be a type of placeholder message that can be sent to the SRS to indicate a message has been dropped, why, and some key attributes about the message. The new SIPREC wrapper mime type could be designed to serve this purpose.

- REPORT messages on the CS can’t be sent directly on the RS. The new SIPREC wrapper mime type could also serve as a way to encapsulate those.

The primary change is to introduce a new wrapper MIME type ("application/msrp-recording") that is used in RS MSRP sessions for all CS MSRP messages that are to be recorded. This is used with SEND messages whether they have a CPIM wrapper or not. It also allows non-SEND messages from the CS to be sent intact in the RS for recording. And it provides a vehicle for carrying other data as needed.

Adding another layer of wrapper could substantially increase the total amount of data send on the RS session, relative to what is present on the CS. I've tried to mitigate that via the details of the design. For SEND messages, only the body of the SEND message is wrapped. And From and To headers in this wrapper can be omitted in cases where that information is redundant. I've assumed that messages other than SEND should in general be infrequent enough that extra overhead when sending them isn't worth a lot of concern.

This wrapper can carry a DateTime header. This provides a mechanism to address the timestamp issues. I've left it as optional to use.

I clarified the non-guarantee of recording in the architecture section.

I've provided a special header in the wrapper to carry the MESSAGE-ID from SEND messages in the CS. And SEND messages will get a separate MESSAGE-ID on the RS MSRP session when sent to the SRS. This provides the SRS with enough information to solve the correlation problem when a message is incomplete in one CS MSRP session and is resumed on another. (The problem of reassembly is left to the SRS.)

The wrapper format includes a mechanism for the SRC to report dropped messages to the SRS.

The wrapper format also includes a mechanism for encapsulating CS REPORT messages for sending to the SRS.

I realized that this level of wrapping provides an opportunity to multiplex unrelated CS MSRP sessions on a single RS MSRP session. To allow this I've provided a way to include the session-id from the metadata, that identifies the particular CS MSRP session, as a value in the wrapper of the message sent on the RS. But I also made that optional when it is redundant. This gives a choice: multiplex but make the messages bigger, or create a separate RS MSRP session for each CS MSRP session and keep the messages smaller. I've included this as a trial balloon for discussion. I'm undecided about it.

The formatting of all of this could be better. But for now I just wanted to get the basic concepts down for review. Once the approach is reasonably well worked out I'll try to improve the formatting.

There are many places here where I am uncertain what normative strength to apply to individual requirements. I've indicated this inline for many of those. Please comment on this.

2. Definitions

(TBD...)

3. MSRP Recording Architecture

For consistency with the SIPREC Architecture [RFC7245] and the SIPREC Protocol [I-D.ietf-siprec-protocol] MSRP recording needs to deliver duplicated MSRP message content from the SRC to the SRS, with suitable descriptive metadata. The SRC may be associated with SIP UA (endpoint) with an MSRP client, or with a SIP B2BUA that accesses the media via an MRSP Relay. An SRC may also be associated with a SIP conference focus and an MSRP switch.

Note: The decision to record or not is a policy decision on the part of both the SRC and the SRS. Support for this specification provides no guarantee that any particular MSRP session, or message within a session, will be recorded. However MSRP recording is subject to the notification requirements called out in Section 6.1.2 of [I-D.ietf-siprec-protocol].

3.1. MSRP Client acts as SRC

[RFC4975] and [RFC4976] describe how an MSRP client communicates to another MSRP client via a SIP session. A MSRP client that has access to the MSRP content to be recorded may act as SRC. The MSRP client may send the replicated media to the SRS along with corresponding metadata.

If the MSRP client/SRC is aware the MSRP session needs to be recorded, it can initiate the establishment of a SIP RS by sending an INVITE to SRS, or vice-versa. The MSRP client/SRC is responsible for notifying the other MSRP client involved in the CS that the MSRP session is being recorded. The MSRP client/SRC is responsible for complying with request from recording aware UAs or through some configured policies indicating that the CS should not be recorded.

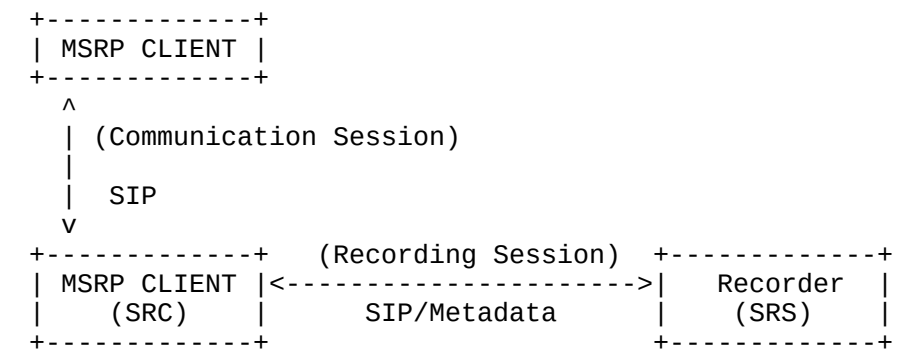


Figure 1: MSRP Client Acts as SRC

3.2. MSRP Relay acts as SRC

(TBD... RFC4976)

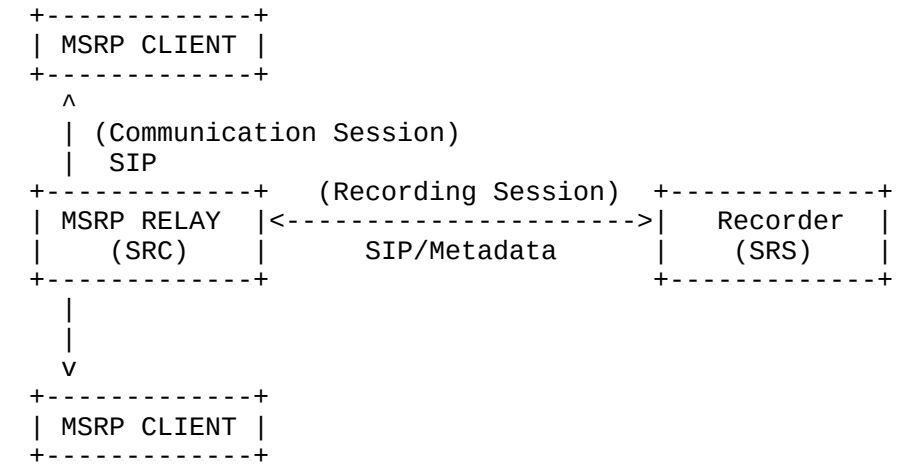


Figure 2: MSRP Relay Acts as SRC

3.3. MSRP Switch acts as SRC

(TBD... ietf-simple-chat)

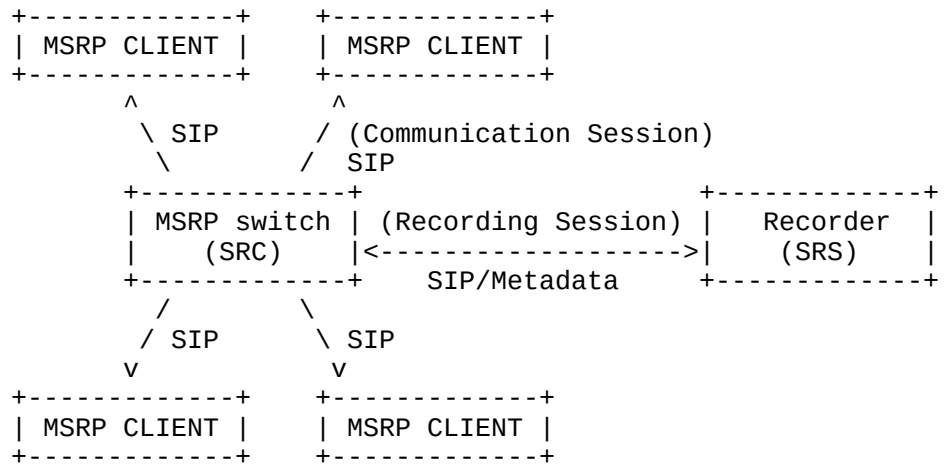


Figure 3: MSRP Switch Acts as SRC

4. MSRP Media Stream Mixing

[TODO: Revise this to cover multiplexing of unrelated media streams.]

Note: SIPREC metadata allows both the inclusion of multiple participants within a single <stream> element, and the mapping of multiple <stream> elements to a single MSRP m-line in the RS. These provide two ways to do very similar things.

Mapping multiple participants to a single <stream> is natural for a conference. It works well for MSRP chat sessions

By providing a way to specify the stream-id with an individual message on the RS, I've introduced a way to demux messages from multiple <stream>s that are mapped to the same MSRP m-line. This provides a way reduce the number of MSRP sessions in the RS. It also avoids confusion when an RS MSRP session is serially reused for distinct CS MSRP sessions.

I'm still considering whether it is good to have both of these mechanisms, or if one of them should be removed. Until I make a decision I haven't updated all the text that pertains to this.

Feedback on this will be appreciated.

As with RTP-based media, CS MSRP media streams from different participants may be mixed into a single RS media stream, or they may be conveyed as separate MSRP streams. In RTP, when media from

different participants is mixed, it is distinguished by CNAME and SSRC or CSRC. In MSRP, media from different participants is distinguished by wrapping the the message in a CPIM body, with the sender identified by the From header in the CPIM. If the SRC mixes MSRP media from multiple senders, then each message that isn't already in CPIM format SHOULD be embedded in a CPIM message, and the From and To headers of that CPIM wrapper SHOULD identify the sending and receiving participants for that message.

5. MSRP Session Usage by the SRC

[TODO: Revise this to cover multiplexing of unrelated media streams.]

When preparing to record a CS MSRP media stream, the SRC MUST choose a corresponding RS MSRP session. CS MSRP sessions that are being mixed share an RS MSRP session, while those that are not being mixed are assigned to unique RS MSRP sessions.

The RS MSRP session MAY be newly created, or a pre-existing RS MSRP session that is no longer in use MAY be repurposed. When an MSRP session is repurposed, the SRC communicates this change to the SRS via a change in the metadata. The SRC is responsible for ensuring that messages for the new session are not sent until the SRS has received the metadata describing this new session.

MSRP message flow on a RS MSRP session is always from the SRC to the SRS. The SRC generates SEND messages, and may receive REPORT messages. It does not receive SEND messages or send REPORT messages.

6. MSRP Session Usage by the SRS

[TODO: Revise this to cover multiplexing of unrelated media streams.]

The SRS MUST be able handle a case where an RS MSRP session if first used to record one CS MSRP session and then is repurposed to record a different CS MSRP session. The SRS is learns of this change via a change in the metadata.

MSRP message flow on a RS MSRP session is always from the SRC to the SRS. The SRS receives SEND messages, and sends REPORT messages. It does not generate SEND messages or receive REPORT messages.

7. File Transfer

A mechanism for doing file transfer via MSRP is specified in [RFC5547]. If this mechanism is used in the CS, then the SRC MAY use it in the RS to record those files. In turn, the SRS MAY choose to

accept some or all of those file transfer requests, or MAY reject them.

Both file push and file pull operations are defined. If the SRC chooses to record a file transfer, whether it is initiated in the CS via a push operation or a pull operation, within the RS the SRC MUST initiate the transfer with a push operation in an SDP Offer.

(SRS initiation of a file transfer is out of scope of this document.)

It is possible that the SRC may support file transfer while the SRS does not. If the SRC sends an SDP offer to the SRS containing an m-line initiating a file transfer, and the SRS sends an answer accepting the MSRP session, but fails to include a matching file-transfer-id, then the SRC MUST NOT send the content of CS MSRP file transfer session to the SRS.

8. Recording Chatrooms

An CS MSRP session might involve a chatroom. The SRC discovers this by observing use of the features defined in [I-D.ietf-simple-chat]

When the CS MSRP session involves a chatroom, the SRC SHOULD [MUST?] indicate this in the corresponding RS MSRP session. The key unique features of chatrooms are nicknames and private messages. If either of these features is indicated in an SDP 'chatroom' attribute in the CS, then this MAY also be indicated in the RS SDP.

Requests for nickanmes in the CS via the NICKNAME message are reported to the SRS using the mechanism described in Section 11.3.

When messages are sent by sources that have had a nickname assigned, the nickname is conveyed to the SRS using the mechanism described in Section 10.4.

Private messages used in a chatroom are identified in the CS via a CPIM wrapper with a To header that identifies the intended recipient(s) rather than the URI of the chatroom itself. This information is retained when the message is forwarded to the RS, while the chatroom URI is also conveyed using the "To" header of the "application/msrp-recording" wrapper, as described in section Section 11.

9. Metadata

The metadata defined in [I-D.ietf-siprec-metadata] can be used without change to describe MSRP streams.

10. MIME Type for MSRP Recording

The document defines a new MIME type "application/msrp-recording" as an extension to type "application/cpim". This type includes new headers for carrying details about the wrapped message. The new headers are all identified by a namespace prefix of "rs.".

> [Note: I found the details of how to make an application-specific extension to CPIM to be vague in RFC3862. I'm uncertain if extension headers must be referenced with a prefix, but that is my best guess. The details need more research.]

10.1. CPIM Extension Header - rs.Content

The value of the "rs.Content" header is a token identifying the sort of content contained in the body of this message. The following types of content are defined:

- send
- drop
- msrp

At most one "rs.Content" header may be present in a message. If no "rs.Content" header is is present, then "rs.Content: send" is implied.

The 'send' token indicates that the content of the message contains all or a fragment of the body of an MSRP SEND message.

The 'drop' token indicates that the content of a SEND message in the CS is not being sent to the RS for recording.

The 'msrp' token indicates that the body of the message contains a complete MSRP message from the CS. This form MAY be used to convey REPORT messages, NICKNAME messages, and transaction responses.

10.2. CPIM Extension Header - rs.Stream-ID

The value of the "rs.Stream-ID" header is the stream-id used in the SIPREC metadata to identify the stream that this message belongs to. This header MAY be omitted if the SIPREC metadata associates exactly one stream with this MSRP session. If present, the value MUST match the stream-id of exactly one of the streams associated with this MSRP session.

10.3. CPIM Extension Header - rs.Message-ID

The value of the "rs.Message-ID" header carries the value of the "Messsage-ID" from the MSRP SEND message. At most one "rs.Message-ID" header may be present in a message. It MUST be present when the "rs.Content" value is ’send’ or ’drop’, and MUST NOT be present in other cases.

10.4. CPIM Extension Header - rs.Nickname

The value of the "rs.Nickname" header carries the nickname of the sender of the MSRP SEND message. At most one "rs.Nickname" header may be present in a message. It MAY be present when the "rs.Content" value is ’send’ or ’drop’, and MUST NOT be present in other cases.

10.5. CPIM Extension Header - rs.Unsupported-Type

The value of the "rs.Unsupported-Type" header carries a content-type from a CS MSRP SEND message that is not supported by the SRS. It may be the outermost type, or the type of a component of a container type. Any number of "rs.Unsupported-Type" headers may be present in a message. It MAY be present when the "rs.Content" value is ’drop’, and MUST NOT be present in other cases.

10.6. CPIM Extension Header - rs.Size

The value of the "rs.Size" header carries the integer size of an CS MSRP SEND message. At most one "rs.Size" header may be present in a message. It MAY be present when the "rs.Content" value is ’drop’, and MUST NOT be present in other cases.

11. Representation of CS MSRP Messages in the RS

When CS MSRP messages are being recorded, the SRC encapsulates them in the wrapper type "application/msrp-recording". This wrapper type is used to encapsulate the basic MSRP SEND message content, and also to send CS MSRP control messages that should be recorded. It also provides the means for conveying per-message metadata.

The CPIM From and To headers of the wrapper are optional. They MUST be supplied when the proper value cannot be determined by other means:

- The From header may be omitted if the metadata for the stream indicates that there is only one possible sender, or if the message being encapsulated contains a CPIM From header with the proper value.

- The To header may be omitted if the metadata for the stream indicates that there is only one possible receiver, or if the message being encapsulated contains a CPIM To header with the proper value.

The CPIM DateTime header MAY be included. If included, it SHOULD indicate the time that the corresponding CS message was sent or received by the SRC.

11.1. Recording CS SEND Messages

When the SRC wishes to record a SEND message from the CS it rewraps the message, taking body from the CS SEND message, placing that into the body of a new "application/msrp-recording" message, and then sending that with a SEND message in the corresponding RS MSRP session.

The SRC MAY retain the fragmentation present in the CS, mapping one CS SEND message to one RS SEND message. Or it MAY merge CS message fragments and/or re-fragment CS SEND message fragments. If a received fragment ends with a continuation-flag of "#", then last fragment sent on the RS MUST also end with a continuation-flag of "#".

Each SEND message fragment MAY, but need not, contain a "rs.Content: send" header.

Each SEND message fragment MUST contain an "rs.Message-ID" header identifying the Message-ID from the corresponding CS MSRP SEND message. (The resulting RS MSRP SEND message will also contain a Message-ID in the RS. This is a distinct value.)

If the SRC knows that the sender of the message on the CS has an associated Nickname [I-D.ietf-simple-chat], then the SRC SHOULD insert an "rs.Nickname" header containing the nickname.

11.2. Dropping CS SEND Messages

[QUESTION: Do we need a way for the SRS to indicate a desire (or not) to receive indications of dropped messages?]

The SRC might decide not to record selected SEND messages from the CS MSRP session. When doing so is MAY send a ’drop’ message as a indicator that a message has been dropped. The following considerations apply when deciding whether to send a ’drop’ message:

- While the SRC is honoring a request within the CS to disable recording, it SHOULD [MUST?] NOT send 'drop' messages for CS SEND messages.

- If the total size from the Byte-Range of the initial fragment of a SEND message in the CS is acceptable for the CS, but exceeds the max-size for the RS session, then the SRC SHOULD send a 'drop' message, and SHOULD include an "rs.Size" header indicating the total size of the message.

- If a SEND message in the CS contains a continuation fragment, with a Byte-Range indicating that the total message will exceed the max-size for the RS session, then the SRC SHOULD send a 'drop' message, and SHOULD include an "rs.Size" header indicating the total size of the message.

- If a SEND message has a content type accepted by the 'accept-types' and 'accept-wrapped-types' attributes of the CS but is not accepted by the 'accept-types' or 'accept-wrapped-types' attributes of the RS, then the SRC SHOULD send a 'drop' message. The 'drop' message SHOULD contain an "rs.Unsupported-Type" header identifying the type that is not supported. (When a multipart body is present, the SRC MAY include multiple "rs.Unsupported-Type" headers identifying multiple types.) The SRC MAY choose to send a limited number of 'drop' messages for particular stream - either in total or per unacceptable type.

When a 'drop' message is sent:

- it MUST be terminated with a continuation-flag of "#";

- additional fragments with the same CS Message-ID MUST NOT be sent on the RS.

11.3. Recording NICKNAME Messages

The SRC SHOULD forward NICKNAME messages in the CS to the SRS.

[QUESTION: Do we need a way for the SRS to indicate a desire (or not) to receive CS Transaction Error messages?]

To forward a NICKNAME message from the CS to the RS, the SRC places the entire NICKNAME message into the body of of a new "application/msrp-recording" message, and then sends that with a SEND message in the corresponding RS MSRP session.

11.4. Recording CS REPORT Messages

The SRC SHOULD [MUST?] forward CS Failure Report messages on the RS.

The SRC MAY [SHOULD?] forward CS Success Report messages on the RS.

[QUESTION: Do we need a way for the SRS to indicate a desire (or not) to receive CS REPORT messages?]

To forward a REPORT message from the CS to the RS, the SRC places the entire REPORT message into the body of of a new "application/msrp-recording" message, and then sends that with a SEND message in the corresponding RS MSRP session.

11.5. Recording CS Transaction Responses

The SRC SHOULD [MUST?] forward CS transaction responses indicating errors to the SRS.

The SRC MAY, but SHOULD NOT forward CS transaction responses indicating success to the SRS. An exception is success responses to NICKNAME messages, which MAY [SHOULD?] be passed to the SRS.

[QUESTION: Do we need a way for the SRS to indicate a desire (or not) to receive CS transaction response messages?]

To forward a transaction response from the CS to the RS, the SRC places the entire transaction response message into the body of of a new "application/msrp-recording" message, and then sends that with a SEND message in the corresponding RS MSRP session.

12. Open Issues

13. IANA Considerations

[TODO: Register application/msrp-recording.]

14. Security Considerations

Not explicitly covered in this version.

15. References

15.1. Normative References

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

[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.

[RFC4975] Campbell, B., Mahy, R., and C. Jennings, "The Message Session Relay Protocol (MSRP)", RFC 4975, September 2007.

[RFC4976] Jennings, C., Mahy, R., and A. Roach, "Relay Extensions for the Message Sessions Relay Protocol (MSRP)", RFC 4976, September 2007.

[RFC5547] Garcia-Martin, M., Isomaki, M., Camarillo, G., Loreto, S., and P. Kyzivat, "A Session Description Protocol (SDP) Offer/Answer Mechanism to Enable File Transfer", RFC 5547, May 2009.

[RFC7245] Hutton, A., Portman, L., Jain, R., and K. Rehor, "An Architecture for Media Recording Using the Session Initiation Protocol", RFC 7245, May 2014.

[I-D.ietf-siprec-metadata] R, R., Ravindran, P., and P. Kyzivat, "Session Initiation Protocol (SIP) Recording Metadata", draft-ietf-siprec-metadata-16 (work in progress), August 2014.

[I-D.ietf-siprec-protocol] Portman, L., Lum, H., Eckel, C., Johnston, A., and A. Hutton, "Session Recording Protocol", draft-ietf-siprec-protocol-14 (work in progress), August 2014.

[I-D.ietf-simple-chat] Niemi, A., Garcia, M., and G. Sandbakken, "Multi-party Chat Using the Message Session Relay Protocol (MSRP)", draft-ietf-simple-chat-18 (work in progress), January 2013.

15.2. Informative References

[RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, June 2005.

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

[RFC4597] Even, R. and N. Ismail, "Conferencing Scenarios", RFC 4597, August 2006.

Authors' Addresses

Michael Yan
Huawei

Email: michael.yan@huawei.com

Paul H. Kyzivat
Huawei

Email: pkyzivat@alum.mit.edu