Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 3, 2007

A. Niemi
M. Garcia-Martin
Nokia

March 2, 2007

Multi-party Instant Message (IM) Sessions Using the Message Session Relay Protocol (MSRP)
draft-niemi-simple-chat-06

Status of this Memo

Copyright Notice

Abstract

The Message Session Relay Protocol (MSRP) defines a mechanism for sending instant messages within a peer-to-peer session, negotiated using the Session Initiation Protocol (SIP) and the Session Description Protocol (SDP). This document defines the necessary tools for establishing multi-party instant messaging (IM) sessions, or chat rooms, with MSRP.

Table of Contents

1. Introduction

The Message Session Relay Protocol (MSRP) [I-D.ietf-simple-message-sessions] defines a mechanism for sending a series of instant messages within a session. The Session Initiation Protocol (SIP) [RFC3261] in combination with the Session Description Protocol (SDP) [RFC3264] allows for two peers to establish and manage such sessions.

In another application of SIP, a user agent can join in a multi-party session or conference that is hosted by a specialized user agent called a conference focus [RFC4353]. Such a conference can naturally involve an MSRP session as one of possibly many media components. It is the responsibility of an entity handling the media to relay instant messages received from one participant to the rest of the participants in the conference.

Several such systems already exist in the Internet. Participants in a chat room can be identified with a pseudonym or nickname, and decide whether their real identity is disclosed to other participants. Participants can also use a rich set of features, such as the ability to send private instant messages to one or more participants, and the ability to establish sub-conferences with one or more of the participants within the existing conference. They also allow combining instant messaging with other media components, such as voice, video, whiteboarding, screen sharing, and file transfer.

Such conferences are already available today with other technologies different than MSRP. For example, Internet Relay Chat (IRC) [RFC2810], Extensible Messaging and Presence Protocol [RFC3920] based chat rooms, and many other proprietary systems provide this kind of functionality. It makes sense to specify equivalent functionality for MSRP-based systems to both provide competitive features as well as enable interworking between the systems.

This document defines requirements, conventions, and extensions for providing private messages and nickname management in centralized conferences with MSRP. This document, however, does not specify functionality that can be used in conference with media different than MSRP. This memo uses the SIP Conferencing Framework [RFC4353] as a design basis. It also aims to be compatible with the Centralized Conferencing Framework [I-D.ietf-xcon-framework]. It is expected that future mechanisms will be developed for providing similar functionality in generic conferences, i.e., where the media is not only restricted to MSRP. The mechanisms described in this document provide a future compatible short-term solution for MSRP centralized conferences.

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 RFC 2119, BCP 14 [RFC2119], and indicate requirement levels for compliant implementations.

This memo deals with a particular case of tightly coupled SIP conferences where the media exchanged consist of session-based instant messaging. Unless otherwise noted, we use the terminology defined in the SIP Conferencing Framework [RFC4353] applied to the scope of this document. In addition to that terminology, we introduce some new terms:

Nickname: a descriptive name associated to a participant.

Nickname URI: A SIP URI that includes a nickname in the user part. See more information in Section 6.

Session-based Instant Messaging Conference: an instance of a tightly coupled conference, in which the media exchanged between the participants consist of (among others) MSRP based instant messages. Also known as a chat room.

Chat Room: a synonym for session-based instant messaging conference.

Chat Room URI: a URI that identifies a particular chat room in a conference server. Since a chat room is a specialized conference of instant messages, in the context of this document, a chat room URI is a synonym of a conference URI.

Conference Server: a (possibly decomposed) server that provides multipart text conference services. It is also the combination of a conference focus and an MSRP switch.

Sender: the conference participant that originally created an instant message and sent it to the chat room for delivery.

Recipient: the destination conference participant(s). This defaults to the full conference participant list, minus the IM Sender.

MSRP switch: a media level entity that receives MSRP messages and delivers them to the other conference participants. An MSRP switch has a similar role to a conference mixer with the exception that an MSRP switch does not actually "mix" together different

input media streams; it merely relays the messages between participants.

Private Instant Message: an instant message sent in a chat room whose intended recipient is something other than the default. The recipient of a private IM can either be one specific conference participant, or a subset of the full participant list. A private IM is usually rendered distinctly from the rest of the IMs, as to indicate that the message was a private communication.

3. Motivations and Requirements

Although conference frameworks describing many types of conferencing applications already exist, such as the Framework and Data Model for Centralized Conferencing [I-D.ietf-xcon-framework] and the SIP Conferencing Framework [RFC4353], the exact details of session-based instant messaging conferences are not well-defined at the moment.

To allow interoperable chat implementations, for both conference-aware, and conference-unaware user agents, certain conventions for MSRP conferences need to be defined. It also seems beneficial to provide a set of features that enhance the baseline multiparty MSRP in order to be able to create systems that have functionality on par with existing chat systems, as well as enable building interworking gateways to these existing chat systems.

We define the following requirements:

REQ-1: A basic requirement is the existence of a multiparty conference, where participants can join and leave the conference and get instant messages exchanged to the rest of the participants.

REQ-2: The conference must have the ability to host other media in addition to MSRP, as well as multiple streams of MSRP.

REQ-3: A conference participant must be able to determine the identities of the sender and recipient of the received IMs.

REQ-4: A conference participant must be able to determine the recipient of the received message. For instance, the recipient of the message might be the entire conference, a conference sidebar or a single participant of the conference (i.e., a private message).

REQ-5: It must be possible to send a message to a single participant, or a subset of the conference participants (i.e., a private instant message).

REQ-6: It must be possible to set up a sidebar session with one or more participants of the chat room.

REQ-7: A conference participant may have a nickname or pseudonym associated with their real identity.

REQ-8: It must be possible for a participant to change their nickname during the progress of the conference.

REQ-9: It must be possible that a participant is only known by their nickname and not their real identity to the rest of the conference.

REQ-10: It must be possible for the MSRP switch itself to send IMs to the conference (e.g., message of the day, welcome messages, server is shutting down, etc.)

REQ-11: It must be possible for participants to learn the capabilities support of the features described in this document (and perhaps others).

4. Overview of Operation

In order to set up a conference, one must first be created. Users wishing to host a conference themselves can of course do just that; their user agents simply morph from an ordinary user agent into a special purpose one called a conference focus. Another, commonly used setup is one where a dedicated node in the network functions as a conference focus.

Each chat room has an identity of its own: a SIP URI that participants use to join the conference, e.g., by sending an INVITE request. The conference focus processes the invitations, and as such, maintains SIP dialogs with each participant. In an instant messaging conference, or chat room, MSRP is one of the established media streams. Each conference participant establishes an MSRP session with an MSRP switch, which is a special purpose MSRP application. The MSRP switch is similar to a conference mixer in that it handles media sessions with each of the participants and bridges these streams together. However, unlike a conference mixer, the MSRP switch merely relays messages between participants but doesn't actually mix the streams in any way. The system is illustrated in Figure 1.

```
                                 +------+
                                 | MSRP |
                                 |Client|
               +------+          +--.---+          +------+
               | MSRP |             |              | MSRP |
               |Client|             |             _|Client|
               +------._            |           ,' +------+
                        `._         |         ,'
                           `.. +----------+ ,'
                              `|          |'
                               |   MSRP   |
                               |  Switch  |
                              ,|          |_
                         _,-'' +----------+ ``-._
               +------.-'            |           `--+------+
               | MSRP |              |              | MSRP |
               |Client|              |              |Client|
               +------+              |              +------+
                                 +---'--+
                                 | MSRP |
                                 |Client|
                                 +------+
```

Figure 1: Multiparty MSRP in a Centralized Conference

Typically conference participants also subscribe to the conference event package [RFC4575] to gather information about the conference roster in the form of conference state notifications. For example, participants can learn about other participants' identities.

All messages in the chat room use the 'Message/CPIM' wrapper content type [RFC3862], so that it is possible to distinguish between private and regular messages. When a participant wants to send an instant message to the conference, it constructs an MSRP SEND request and submits it to the MSRP switch including a regular payload (e.g., a Message/CPIM message that contains a text, html, an image, etc.). The Message/CPIM To header is set to the chat room URI. The switch then fans out the SEND request to all of the other participants using their existing MSRP sessions.

A participant can also send a private instant message addressed to one or more conference participants whose identities have been learnt, e.g., via a notification from the conference event package [RFC4575]. In this case the sender creates an MSRP SEND request with a Message/CPIM body whose To or Cc headers contain not the chat room URI but one or more nickname or participant URIs. The MSRP switch then fans out the SEND request to each of the participants listed in the To or Cc headers of the Message/CPIM body.

We extend the current MSRP negotiation that takes place in SDP [RFC4566] to allow participants to learn whether the chat room supports and is willing to accept (e.g., due to local policy restrictions) certain MSRP functions defined in this memo, such as nicknames or private messaging.

Naturally, when a participant wishes to leave a chat room, it sends a SIP BYE request to the conference focus and disconnects.

5. Creating, Joining, and Deleting a Chat Room

5.1. Creating a Chat Room

Since we consider a chat room a particular type of conference where one of the offered media happens to be MSRP, the methods defined by the SIP Conference Framework [RFC4353] for creating conferences are directly applicable to a chat room.

Once a chat room is created, it is identified by a SIP URI, like any other conference.

5.2. Joining a Chat Room

Participants usually join the conference by sending an INVITE request to the conference URI. As long as the conference policy allows, the INVITE request is accepted by the focus and the user is brought into the conference. Participants are aware that the peer is a focus due to the presence of the "isfocus" feature tag [RFC3840] in the Contact header field of the 200-class response to the INVITE request. Participants are also aware that the mixer is an MSRP switch due to the presence of an additional 'message' media type and either TCP/MSRP or TCP/TLS/MSRP as the protocol field in the SDP [RFC4566] media-line.

The conference focus of a chat room MUST include support for a Message/CPIM [RFC3862] top-level wrapper for the MSRP messages by setting the 'accept-types' MSRP media line attribute in the SDP offer or answer to include 'Message/CPIM'.

> Note that the 'Message/CPIM' wrapper is used to carry the sender information that, otherwise, it will not be available to the recipient. Additionally, 'Message/CPIM' wrapper carries the recipient information (e.g., To and Cc: headers).

The conference focus of a chat room MUST learn the chatroom capabilities of each participant that joins the chat room, and MUST inform the MSRP mixer of such support. This is to prevent that the

MSRP mixer distributes private messages to participants who do not support private messaging.

5.3. The SDP 'chatroom' attribute

There are a handful of use cases where a participant would like to learn the chatroom capabilities supported by the MSRP switch and the chat room. For example, a participant would like to learn if the MSRP switch supports private messaging, otherwise, the participant may send what he believes is a private instant message addressed to a few participants, but since the MSRP switch does not support the functions specified in this memo, the message gets eventually distributed to all the participants of the chat room.

The reverse case also exists. A participant, say Alice, whose user agent does not support the extensions defined by this document joins the chat room. The MSRP switch learns that Alice application does not support private messaging nor nicknames. If another participant, say Bob, sends a private message to Alice, the MSRP switch does not distribute it to Alice, because Alice is not able to differentiate it from a regular message sent to the whole roaster. Further more, if Alice replied to this message, she would do it to the whole roaster. Because of this, the MSRP mixer keeps also track of users who do not support the extensions defined in this document.

In another scenario, the policy of a chat room may indicate that certain functions are not allowed. For example, the policy may indicate that nicknames or private messages are not allowed.

In order to provide the user with a good chatroom experience, we define a new 'chatroom' SDP attribute. The 'chatroom' attribute is a media-level attribute that MAY be included in conjunction with and MSRP media stream (i.e., when an m= line in SDP indicates "TCP/MSRP" or "TCP/TLS/MSRP"). The 'chatroom' attribute indicates the intersection of support and chatroom local policy allowance for a number of functions specified in this document. Specifically, we provide the means for indicating support to use nicknames and private messaging.

The 'chatroom' SDP attribute has the following syntax:

```
        chatroom = chatroom-label ":" chat-token *(SP chat-token)
        chatroom-label = "chatroom"
        chat-token = (nicknames-token | private-msg-token | token)
        nicknames-token = "nicknames"
        private-msg-token = "private-messages"
```

A conference focus that includes the 'nicknames' token in the session

description is signalling that the MSRP switch supports and the chatroom allows to use the procedures specified in Section 6. A conference focus that includes the 'private-messages' in the SDP description is signalling that the MSRP switch supports and the chatroom allows to use the procedures specified in Section 7.2.

Example of the 'chatroom' attribute for an MSRP media stream that indicates the acceptance of nicknames and private messages:

```
        a=chatroom:nickname private-messages
```

5.4. Deleting a Chat Room

As with creating a conference, the methods defined by the SIP Conference Framework [RFC4353] for deleting a conference are directly applicable to a chat room.

Deleting a chat room is an action that heavily depends on the policy of the chat room. The policy can determine that the chat room is deleted when the creator leaves the conference, or with any out of band mechanism.

6. Nicknames

A common characteristic of existing chat room services is that participants have the ability to identify themselves with a nickname to the rest of the participants of the conference. This provides a layer of anonymity, whereby the conference server authenticates the participant, but still allows the participant to keep anonymity of his SIP URI towards the rest of the participants without downgrading his services. Specifically, anonymous participants are able to receive private instant messages from other participants without revealing their SIP URI.

One option to satisfy an aspect of nicknames consists of using the display name with a real identity as the URI, for example, in CPIM headers. A nickname in the display name offers a pseudonym that anyone can map to a real identity, thus not satisfying the anonymity requirements.

Another option consists of using a nicknaming service, that allows allocating nickname URIs to users. Using such a URI in a conference in effect anonymizes the user, but still allows the user to be reached outside the chat room using the same identity. However, defining such nicknaming service machinery is out of the scope of this specification.

Instead, we take the approach of defining a nickname as the combination of an optional quoted display name followed by a nickname URI, and allowing the use of such nicknames in To and Cc headers in CPIM. A nickname URI is a SIP URI formed from the chat room URI that embeds a nickname identifier. A nickname URI does not resolve to the user himself, but to the particular chat room where the user has joined.

In other words, a nickname is simply a username that is scoped for a particular chat room. Such nicknames are allocated on a first-come first-served policy, meaning they can also be "stolen". It is out of the scope of this specification to define nickname retention schemes, or nickaming services as discussed above.

> Note that for some hosted chat rooms, this feature of nicknames may be too much to tolerate. For such chat rooms, it may be more desirable to disallow nicknames altogether, and have chat room participants be identified with their own full SIP URI instead (or any other URI scheme they used to join the room).

Based on the above discussion, we define a nickname as follows:

```
Nickname = [Display-Name] (nickname-URI)
```

An example of a nickname is:

```
"Alice in wonderland" <sip:alice%20wonderland@chat.example.com>
```

The display name of a nickname is used only for displaying purposes. The nickname URI is used for routing. In particular, the conference server maintains a mapping table between nickname URIs, SIP URIs and MSRP sessions pertaining to a participant.

Nickname URIs are scoped to a chatroom. Therefore, a nickname identifier MUST be unique within a chatroom, and SHOULD be unique within a conference server or administrative domain. This way, two different users cannot have the same nickname in different rooms on the same chat server, unless there are valid reasons for allowing this. For example, some chat rooms might need to assign some well-known nickname to a secretary, which of course might be a different user in different rooms.

However, it is still possible that the same user is using different nicknames in different chat rooms hosted by the same conference server.

In order to maintain high compatibility with existing SIP User Agents, we define a convention for creating a nickname URI. The

convention consist on prepending an escaped nickname identifier and a possible escaped '@' sign to the existing username part of the chat room URI.

Let us take a look at an example. Assume the chat room URI allocated to a given chat room is 'sip:room34@example.com'. A user whose nickname identifier is set to 'nordicguy' is represented with the nickname URI: 'sip:nordicguy%40room34@example.com'.

In another example the chat room URI does not include a username part. For example, the chat room URI is 'sip:chat34.example.com'. In this context a user whose nickname is 'nordicguy' gets represented with a nickname URI of 'sip:nordicguy@chat34.example.com'.

An interesting property of this approach is that nickname URIs do not really resolve to the SIP UA or real identity of the user. Instead, they resolve to the conference server. Only the conference server and the owner of the nickname are able to map a nickname URI to the SIP URI of the user. Other participants can use the conference server as an intermediary for delivery of private messages addressed to any of the nickname URIs of the chat room.

As a consequence of the structure of the nickname URI, if a user has the same nickname identifier in two different chat rooms, the nickname URI will be different (because the chat room URIs are different). For example, the nickname URIs of 'nordicguy' in two different chat rooms would be 'sip:nordicguy%40conf12@example.com' and 'sip:nordicguy%40conf34@example.com'. Each one is used within its own chat room.

6.1. Provisioning Nicknames

Since nicknames are scoped within a chat room (and usually also within a chat server or administrative domain), we provide a mechanism for requesting and reserving a nickname for the user's disposal for the duration the user is logged into the chat room. The mechanism is based on the definition of the NICKNAME MSRP method (see below). Note that other mechanisms may exists (for example, a web page reservation system), although they are outside the scope of this document. Further more, the mechanism that we specify in this memo is able to reserve a nickname for the user's disposal for the time the user is logged into the chat room. Other mechanisms that provide persistent nicknames or nickname reservation across multiple chat rooms or conference servers are outside the scope of this memo.

A participant in a chat room MAY send a NICKNAME method to the MSRP switch to request the reservation of a nickname for the user's disposal for the duration of the session (i.e., while the participant

is joined to the chat room) at any time once the MSRP session has been established and authenticated. Typically users will reserve a nickname as soon as the join the chat room, prior to sending any messages.

We additionally define two new MSRP header fields "Set-Nickname" and "Proposed-Nickname". The "Set-Nickname" header field carries a nickname, while the "Proposed-Nickname" header field can carry one or more nicknames. Set-Nickname header field MUST only be included in a NICKNAME request. The Proposed-Nickname header fields MUST only be included a 423 responses to NICKNAME requests. URIs included in the Set-Nickname and Proposed-Nickname header fields MUST be formatted according to the conventions for nickname URIs.

The syntax of the NICKNAME method and the "Proposed-Nickname" header field is built upon the MSRP formal syntax [I-D.ietf-simple-message-sessions] and the SIP formal syntax [RFC3261]:

```
   ext-method =/ NICKNAMEm
   NICKNAMEm = %x4E.49.43.4B.4E.41.4D.45 ; NICKNAME in caps
   ext-header =/ Set-Nickname
                  ; ext-header is specified in RFC XXXX
                  ; name-addr is specified in RFC 3261
   Set-Nickname = "Set-Nickname" ":" name-addr
   ext-header =/ Proposed-Nickname
   Proposed-Nickname = "Proposed-Nickname" ":" name-addr
                        *(COMMA name-addr)
```

A conference participant who has established an MSRP session with an MSRP switch, where the MSRP switch has indicated the support and availability of nicknames with the 'nicknames' token in the 'chatroom' SDP attribute, MAY send a NICKNAME request to the MSRP switch. The MSRP NICKNAME request MUST contain a Set-Nickname header field that includes one nickname URI that the user would like to be known as. URIs included in the Set-Nickname header field MUST be formatted as nickname URIs.

An MSRP switch that receives a NICKNAME request containing a proposed nickname in the Set-Nickname header field SHOULD verify first whether the policy of the chat room allows the nickname functionality. If is not allowed, the MSRP switch MUST answer with a 501 response.

If the policy of the chat room allows conference participants to negotiate and use their nicknames, the MSRP switch then examines nickname contained in the Set-Nickname header field. If the URI included in the Set-Nickname header field is not formatted as a nickname URI (e.g., the chat room URI is not used), then the MSRP

switch discards that proposal and moves to the next one. For every valid nickname URI the MSRP switch finds if the proposed nickname URI is already in use or matches the local policy otherwise. If the proposal is not acceptable for any reason, the MSRP switch discards the proposal and moves to the next one. Note that the MSRP switch bases its decision on the nickname URI only, and it does not use the display name for this validation. If a proposed nickname URI is valid and not already used, the MSRP switch inserts the entry into its mapping table, associated to the user's SIP URI and MSRP session, and generates a 200 response to the NICKNAME request. The 200 response MUST include a Proposed-Nickname header field that contains the selected nickname.

If the MSRP NICKNAME request does not contain a Proposed-Nickname header field, or if it contains such header, but all the proposed nicknames are not acceptable (e.g., because they are already taken), the MSRP switch generates a 423 response. The 423 response SHOULD contain a Proposed-Nickname header field that contains one or more nickname URIs proposed by the MSRP switch.

The sender of an MSRP NICKNAME request can receive a 200 response that contains a Proposed-Nickname header field containing the nickname URI that the user has been granted for the duration of the session. If the response is a 423, then none of the proposals of the NICKNAME request were accepted. The 423 response includes a Proposed-Nickname header field that contains the MSRP switch proposals. The MSRP endpoint MAY send a new NICKNAME request that includes a new nickname proposal.

6.2. Modifying a Nickname

At any time during the session the MSRP endpoint may want to modify his nickname. Modification of the nickname is not different from the initial provision of a nickname, thus the NICKNAME method is used as described in Section 6.1.

If a NICKNAME method that attempts to modify the current nickname of the user for some reason fails, the current nickname stays in effect. The new nickname comes into effect and the old one is released only after a NICKNAME method is accepted and receives a 200-class repsonse.

6.3. Mapping Nicknames to Other Identities

The MSRP switch maintains a mapping table that correlates, for a given user, his nickname, SIP URI, and MSRP session ID. This correlation is valid for the duration of the session (unless mechanisms specified elsewhere exists to provide long-lasting

nicknames). Thus, at the dismissal of the session the MSRP switch should dispose the nickname and make it available to other participants.

Typically the conference focus acts as a notifier of the SIP conference event package [RFC4575]. The conference focus MAY notify subscribers of the nickname allocated to a given participant. We define an extension to the conference event package to include nicknames. The extension defines a new <nickname> element as a child element of the existing <user> element. The <nickname> element includes an 'entity' attribute that contains the nickname URI. A child <display-name> element contains the display name of the nickname.

> TO BE DONE: include a formal definition of the <nickname> extension to the conference event package.

7. Sending and Receiving Instant Messages

7.1. Regular Messages

This section describes the conventions used to send and receive instant messages that are addressed to all the participants in the chat room. These are sent over a regular MSRP SEND request that contains a Message/CPIM wrapper [RFC3862] that in turn contains the desired payload (e.g., text, image, video-clip, etc.).

When a chat room participant wishes to send an instant message to all the other participants in the chat room, he constructs an MSRP SEND request that MUST contain a top-level wrapper of type 'Message/CPIM' [RFC3862]. The actual instant message payload inside 'Message/CPIM' MAY be of any type negotiated in the SDP 'accepted-types' attribute according to the MSRP rules.

The sender SHOULD populate the From header of the Message/CPIM wrapper with a proper identity by which the user is recognized in the conference. Identities that can be used (among others) are:

- o A SIP URI [RFC3261] representing the participant's address-of-record
- o A tel URI [RFC3966] representing the participant's telephone number
- o An IM URI [RFC3860] representing the participant's instant messaging address

- o A nickname URI formatted according to the rules indicated in Section 6 and allocated for the user.

If the sender of the message wants to remain anonymous to the rest of the participants, and providing that the policy of the conference allows anonymous participation, the creator SHOULD populate the From header of the Message/CPIM body with an anonymous identity, e.g., using the "anonymous" SIP URI as described in RFC 3261 [RFC3261] Section 8.1.1.3. or using a nickname URI (see Section 6) that has been allocated to the user.

The sender MUST populate the To header field of the Message/CPIM body with the chat room URI.

An MSRP switch that receives a SEND request from a participant SHOULD first verify that the From header field of the Message/CPIM wrapper is correctly populated with a valid URI as indicated earlier. If the URI included in the From header field of the Message/CPIM wrapper is not valid (e.g, because it does not "belong" to the user), then the MSRP switch MUST generate a 403 response and MUST NOT forward the SEND request to any of the participants. Otherwise, the MSRP switch SHOULD generate a 200 response according to the MSRP rules for response generation.

Then the MSRP switch should inspect the To header field of the Message/CPIM wrapper. If the To header field of the Message/CPIM wrapper contains the chat room URI, the MSRP switch can generate a copy of the SEND request to each of the participants in the conference except the sender. The MSRP switch MUST NOT modify any of the bodies included in the received SEND request. Note that the MSRP switch does not need to wait for the reception of the complete MSRP chunk or MSRP message before it starts the distribution to the rest of the participants. Instead, once the MSRP switch has received the headers of the Message/CPIM body it SHOULD start the distribution process.

An MSRP endpoint that receives a SEND request from an MSRP switch containing a Message/CPIM wrapper SHOULD first inspect the To header field of the Message/CPIM body. If the To header field is set to the chat room URI, then it is a regular message that has been distributed to all the participants in the conference. Then the MSRP endpoint SHOULD inspect the From header field of the Message/CPIM body to identify the sender. The From header field will include a URI that identifies the sender. The endpoint might have also received further identity information through a subscription to the SIP conference event package [RFC4575].

7.2. Private Messages

This section describes the conventions used to send and receive private instant messages, i.e., instant messages that are addressed to one or more selected participants of the chat room rather to all of them. A private instant message is sent over a regular MSRP SEND request that contains a Message/CPIM wrapper [RFC3862] which contains the desired payload (e.g., text, image, video-clip, etc.), according to the procedures of RFC 3862 [RFC3862].

When a chat room participant wishes to send a private instant message to one or more participants in the chat room, he constructs an MSRP SEND request that MUST contain a top-level wrapper of type 'Message/CPIM' [RFC3862]. The actual instant message payload inside 'Message/CPIM' MAY be of any type negotiated in the SDP 'accepted-types' attribute according to the MSRP rules.

The sender SHOULD populate the From header of the Message/CPIM wrapper with a proper identity by which the user is recognized in the conference as indicated for regular instant messages. Then the sender MUST populate the To header field and MAY populate the Cc header field of the Message/CPIM with the identity of intended recipients. These identities include SIP, TEL, and IM URIs, and nickname URIs (see Section 6) typically learnt from the information received in notifications of the conference event package [RFC4575].

As for regular messages, an MSRP switch that receives a SEND request from a participant SHOULD first verify that the From header field of the Message/CPIM wrapper is correctly populated with a valid URI as indicated earlier. If the URI included in the From header field of the Message/CPIM wrapper is not valid (e.g, because it does not "belong" to the user), then the MSRP switch MUST generate a 403 response and MUST NOT forward the SEND request to any of the participants. Otherwise, the MSRP switch SHOULD generate a 200 response according to the MSRP rules for response generation.

Then the MSRP switch MUST inspect the To header field of the Message/CPIM wrapper. If the To header field of the Message/CPIM wrapper does not contain the chatroom URI the MSRP switch inspects the URIs included in both the To and Cc headers. For each URI found there, the MSRP switch first searches for support for private messages from the intended recipient. If the recipient does not support private messages, the MSRP switch does not forward the message to that recipient. For each of the remaining intended recipients, the MSRP switch searches in its mapping table to find the MSRP session established towards the user's MSRP endpoint. Once a match is found the MSRP switch MUST create a SEND request on that MSRP session and MUST copy the contents (e.g., the whole Message/CPIM wrapper and its

bodies) to a SEND request and send it over that MSRP session.

There might be situations where one or more URIs included in the To or Cc headers of the Message/CPIM wrapper cannot resolve to existing MSRP sessions, e.g., due to a mistyped URI or because the recipient has abandoned the chat room. In this case it might be benefitial for the sender to become aware of which recipients the MSRP switch failed to resolve. To support this case we define a new MSRP response code 427. This response code is not used in MSRP responses, but as part of the REPORT status code. Note that the 427 status code in a REPORT request merely indicates a failure in resolving a URI to an active MSRP session, and it does not indicate whether the SEND request was successfully received by any of the recipients (it might be still possible that a URI resolves to an active MSRP session but the SEND request cannot be delivered due to congestion, failure of the TCP connection, or any failure at the recipient's MSRP endpoint).

If the MSRP switch cannot resolve any of the URIs included in the To or Cc headers, and the Failure-Report header field of the SEND request was either not present in the original request, or had a value of "yes", the MSRP switch MUST generate a REPORT request to the sender. The Status header field MUST be set to 427. The REPORT request MUST include a Message/CPIM wrapper, with the original From header field included in the SEND request, and the To and Cc header fields containing the subset of failed-to-resolve URIs included in the To and Cc header fields of original Message/CPIM wrapper, respectively.

An MSRP endpoint that receives a SEND request from an MSRP switch containing a Message/CPIM wrapper SHOULD first inspect the To header field of the Message/CPIM body. If the To header field is not set to the chat room URI, then it is a private message that has been distributed to only selected participants in the conference (addressed in the To and Cc headers of the Message/CPIM body). Then the MSRP endpoint SHOULD inspect the From header field of the Message/CPIM body to identify the sender. The From header field will include a URI that identifies the sender. The endpoint might have also received further identity information through a subscription to the SIP conference event package [RFC4575].

8. Sidebars

This document does not provide any protocol means to create, manipulate, or send messages to sidebars. In many cases, a sidebar is a logical subgroup of participants which exists only in each of the recipients endpoints. Sending a message to the sidebar is modelled as a private message addressed to each of the members of the

sidebar. As such, it is to possible to re-create the 'sidebar user experience' totally in the endpoints by correlating collections of private messages, thus, this document does not create any sidebar-specific mechanism.

9. Examples

9.1. Joining a chat room

Figure 6 presents a flow diagram where Alice joins a chat room by sending an INVITE request. This INVITE request contains a session description that includes the chatroom extensions defined in this document.

```
                    Alice               Conference focus
                      |                         |
                      |(1) (SIP) INVITE         |
                      |------------------------>|
                      |(2) (SIP) 200 OK         |
                      |<------------------------|
                      |(3) (SIP) ACK            |
                      |------------------------>|
                      |                         |
```

Figure 6: Flow diagram of a user joining a chat room

F1: Alice constructs an SDP description that includes an MSRP media stream. She also indicates her support for the chatroom extensions defined in this document. She sends the INVITE request to the chatroom server.

```
   INVITE sip:chatroom22@chat.example.com SIP/2.0
   Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
   Max-Forwards: 70
   From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
   To: Chatroom 22 <sip:chatroom22@chat.example.com>
   Call-ID: 3848276298220188511@atlanta.example.com
   CSeq: 1 INVITE
   Contact: <sip:alice@client.atlanta.example.com;transport=tcp>
   Content-Type: application/sdp
   Content-Length: [length]

   v=0
   o=alice 2890844526 2890844526 IN IP4 client.atlanta.example.com
   s=-
   c=IN IP4 atlanta.example.com
   m=message 7654 TCP/MSRP *
   a=accept-types:message/cpim text/plain text/html
   a=path:msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
   a=chatroom:nickname private-messages
```

Figure 7: INVITE request containing an SDP offer with chatroom extensions

F2: The chatroom server accepts the session establishment. It includes the 'isfocus' and other relevant feature tags in the Contact header field of the response. The chatroom server also builds an SDP answer that also that forces the reception of messages wrapped in message/cpim envelops. It also includes the the chatroom attribute with the allowed extensions.

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Chatroom 22 <sip:chatroom22@chat.example.com>;tag=8321234356
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Contact: <sip:chatroom22@chat.example.com;transport=tcp> \
          ;methods="INVITE,BYE,OPTIONS,ACK,CANCEL,SUBSCRIBE,NOTIFY" \
          ;automata;isfocus;message;event="conference"
Content-Type: application/sdp
Content-Length: [length]

v=0
o=chat 2890844527 2890844527 IN IP4 chat.example.com
s=-
c=IN IP4 chat.example.com
m=message 12763 TCP/MSRP *
a=accept-types:message/cpim
a=accept-wrapped-types:text/plain text/html *
a=path:msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
a=chatroom:nickname private-messages
```

Figure 8: 200 (OK) response including chatroom extensions

F3: The session established is acknowledged (details not shown).

9.2. Setting up a nickname

Figure 9 shows an example of Alice setting up a nickname. Her first proposal is not accepted because the proposed nickname is already in use. Her second proposal is accepted.

```
                  Alice                  MSRP mixer
                    |                         |
                    |(1) (MSRP) NICKNAME      |
                    |------------------------>|
                    |(2) (MSRP) 423           |
                    |<------------------------|
                    |(3) (MSRP) NICKNAME      |
                    |------------------------>|
                    |(4) (MSRP) 200           |
                    |<------------------------|
                    |                         |
```

Figure 9: Flow diagram of a user setting up her nickname

F1: Alice sends an MSRP NICKNAME request that contains her proposed

nicknames in the Set-Nickname header field.

```
MSRP d93kswow NICKNAME
To-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
From-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
Set-Nickname: "Alice" <sip:alice%40chatroom22@chat.example.com>
-------d93kswow$
```

Figure 10: MSRP NICKNAME request with an initial nickname proposal

F2: The MSRP mixer analyzes the existing allocation of nicknames and detects that the nickname identified by the URI sip:alice%40chatroom22@chat.example.com is already in used by another participant. The MSRP mixer answers with a 423 response that includes a Proposed-Nickname header field with a proposal of available nicknames.

```
MSRP d93kswow 423 Nickname already in use
To-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
From-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
Proposed-Nickname: "Alice 2" \
         <sip:alice22%40chatroom22@chat.example.com>
-------d93kswow$
```

Figure 11: MSRP 423 response

> NOTE: MSRP does not permit line folding. A "\" in the examples shows a line continuation due to limitations in line length of this document. Neither the backslash, nor the extra CRLF are included in the actual request or response.

F3: Alice receives the response, however, she does not like the proposed nickname by the MSRP mixer. She proposes two new nicknames in a new NICKNAME request.

```
MSRP 09swk2d NICKNAME
To-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
From-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
Set-Nickname: "Alice in Wonderland" \
     <sip:alice%20wonderland%40chatroom22@chat.example.com>, \
     "Alice in Wonderland" \
     <sip:alice%20wonder%40chatroom22@chat.example.com>
-------09swk2d$
```

Figure 12: MSRP NICKNAME request with a second nickname proposal

F4: The MRSP mixer accepts the nickname proposal and answers with a 200 response. The Set-Nickname header indicates the chosen nickname.

```
MSRP 09swk2d 200 OK
To-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
From-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
Set-Nickname: "Alice in Wonderland" \
     <sip:alice%20wonderland%40chatroom22@chat.example.com>
-------09swk2d$
```

Figure 13: MSRP NICKNAME request with an initial proposal

9.3. Sending a regular message to the chat room

Figure 14 depicts a flow diagram where Alice is sending a regular message addressed to the chat room. The MSRP mixer distributes the message to the rest of the participants.

```
    Alice               MSRP mixer                   Bob   Charlie
      |                      |                        |        |
      | (1) (MSRP) SEND      |                        |        |
      |--------------------->|  (3) (MSRP) SEND       |        |
      | (2) (MSRP) 200       |----------------------->|        |
      |<---------------------|  (4) (MSRP) SEND       |        |
      |                      |-------------------------------->|
      |                      |  (5) (MSRP) 200 OK     |        |
      |                      |<-----------------------|        |
      |                      |  (6) (MSRP) 200 OK     |        |
      |                      |<------------------------------ |
      |                      |                        |        |
      |                      |                        |        |
```

Figure 14: Sending a regular message to the chat room

F1: Alice builds a text message and wraps it in a CPIM message. She addresses the CPIM message to the chat room. She encloses the result in an MSRP SEND request and sends it to the MSRP mixer via the existing TCP connection.

```
MSRP 3490visdm SEND
To-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
From-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
Message-ID: 99s9s2
Byte-Range: 1-*/*
Content-Type: message/cpim

To: <sip:chatroom22@chat.example.com;transport=tcp>
From: "Alice in Wonderland" \
       <sip:alice%20wonderland%40chatroom22@chat.example.com>
DateTime: 2007-03-02T15:02:31-03:00
Content-Type: text/plain

Hello guys, how are you today?
-------3490visdm$
```

Figure 15: Instant message addressed to all participants in the chat room

F2: The MSRP mixer acknowledges the reception of the SEND request with a 200 (OK) response.

```
MSRP 3490visdm 200 OK
To-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
From-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
Message-ID: 99s9s2
Byte-Range: 1-*/*
-------3490visdm$
```

Figure 16: 200 (OK) response

F3: The MSRP mixer creates a new MSRP SEND request that contains the received message/cpim body and sends it to Bob.

```
   MSRP 490ej23 SEND
   To-Path: msrp://client.biloxi.example.com:4923/49dufdje2;tcp
   From-Path: msrp://chat.example.com:5678/jofofo3;tcp
   Message-ID: 304sse2
   Byte-Range: 1-*/*
   Content-Type: message/cpim

   To: <sip:chatroom22@chat.example.com;transport=tcp>
   From: "Alice in Wonderland" \
          <sip:alice%20wonderland%40chatroom22@chat.example.com>
   DateTime: 2007-03-02T15:02:31-03:00
   Content-Type: text/plain

   Hello guys, how are you today?
   -------490ej23$
```

Figure 17: Instant message sent to all participants

The rest of the message flows are analogous to the previous. They are not shown here.

9.4. Sending a private message to a participant

Figure 18 depicts a flow diagram where Alice is sending a private message addressed to Bob's nickname. The MSRP mixer distributes the message only to Bob.

```
       Alice               MSRP mixer                   Bob   Charlie
         |                       |                         |        |
         | (1) (MSRP) SEND       |                         |        |
         |---------------------->|  (3) (MSRP) SEND        |        |
         | (2) (MSRP) 200        |------------------------>|        |
         |<----------------------|  (4) (MSRP) SEND        |        |
         |                       |-------------------------------->|
         |                       |                         |        |
         |                       |                         |        |
```

Figure 18: Sending a private message to Bob

F1: Alice builds a text message and wraps it in a CPIM message. She addresses the CPIM message to the Bob's nickname, which she learnt from a notification in the conference event package. She encloses the result in an MSRP SEND request and sends it to the MSRP mixer via the existing TCP connection.

```
   MSRP 6959ssdf SEND
   To-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
   From-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
   Message-ID: okj3kw
   Byte-Range: 1-*/*
   Content-Type: message/cpim

   To: "Bob the great" \
       <sip:bob%20the%20great%40chatroom22@chat.example.com>
   From: "Alice in Wonderland" \
          <sip:alice%20wonderland%40chatroom22@chat.example.com>
   DateTime: 2007-03-02T15:02:31-03:00
   Content-Type: text/plain

   Hello Bob.
   -------6959ssdf$
```

Figure 19: Private instant message addressed to one participant

F2: The MSRP mixer acknowledges the reception of the SEND request with a 200 (OK) response.

```
   MSRP 6959ssdfm 200 OK
   To-Path: msrp://client.atlanta.example.com:7654/jshA7weztas;tcp
   From-Path: msrp://chat.example.com:12763/kjhd37s2s20w2a;tcp
   Message-ID: okj3kw
   Byte-Range: 1-*/*
   -------6959ssdfm$
```

Figure 20: 200 (OK) response

F3: The MSRP mixer creates a new MSRP SEND request that contains the received message/cpim body and sends it only to Bob. Bob can distinguish the sender in the From header of the CPIM message. He also identifies this as a private message due to the To CPIM header.

```
MSRP 9v9s2 SEND
To-Path: msrp://client.biloxi.example.com:4923/49dufdje2;tcp
From-Path: msrp://chat.example.com:5678/jofofo3;tcp
Message-ID: d9fghe982
Byte-Range: 1-*/*
Content-Type: message/cpim

To: "Bob the great" \
    <sip:bob%20the%20great%40chatroom22@chat.example.com>
From: "Alice in Wonderland" \
       <sip:alice%20wonderland%40chatroom22@chat.example.com>
DateTime: 2007-03-02T15:02:31-03:00
Content-Type: text/plain

Hello Bob.
-------9v9s2$
```

Figure 21: Private instant message sent to Bob

Flow F4 is not shown.

9.5. IANA Considerations

TBD.

9.6. Security Considerations

This document proposes extensions to the Message Session Relay Protocol [I-D.ietf-simple-message-sessions]. Therefore, the security considerations of such document apply to this document as well.

In general, messages sent to a multi-party session based messaging focus are not deem to expose any security threat. Nevertheless, if a participant wants to avoid eavesdropping from non authorized entities, it should send those messages a TLS [RFC4346] transport connection, as allowed by MSRP.

9.7. Acknowledgements

The authors want to thank Eva Leppanen, Adamu Haruna, Paul Kyzivat, and Adam Roach for providing comments.

10. References

10.1. 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.

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

[RFC3860] Peterson, J., "Common Profile for Instant Messaging (CPIM)", RFC 3860, August 2004.

[RFC3862] Klyne, G. and D. Atkins, "Common Presence and Instant Messaging (CPIM): Message Format", RFC 3862, August 2004.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[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.

[I-D.ietf-simple-message-sessions] Campbell, B., "The Message Session Relay Protocol", draft-ietf-simple-message-sessions-18 (work in progress), December 2006.

10.2. Informative References

[RFC2810] Kalt, C., "Internet Relay Chat: Architecture", RFC 2810, April 2000.

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

[RFC3920] Saint-Andre, P., Ed., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 3920, October 2004.

[RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers",

RFC 3966, December 2004.

[RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, February 2006.

[I-D.ietf-xcon-framework]
Barnes, M., "A Framework and Data Model for Centralized Conferencing", draft-ietf-xcon-framework-06 (work in progress), December 2006.

Authors' Addresses

Aki Niemi
Nokia
P.O. Box 407
NOKIA GROUP, FIN 00045
Finland

Phone: +358 50 389 1644
Email: aki.niemi@nokia.com

Miguel A. Garcia-Martin
Nokia
P.O. Box 407
NOKIA GROUP, FIN 00045
Finland

Phone: +358 50 480 4586
Email: miguel.an.garcia@nokia.com

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).