Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 25, 2015

P. Saint-Andre
&yet
S. Ibarra
AG Projects
S. Loreto
Ericsson

November 21, 2014

Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): Groupchat
draft-ietf-stox-groupchat-08

Abstract

This document defines a bidirectional protocol mapping for the exchange of instant messages in the context of a multiparty chat session among users of the Session Initiation Protocol (SIP) and users of the Extensible Messaging and Presence Protocol (XMPP). Specifically, this document defines a mapping between the SIP-based Message Session Relay Protocol (MSRP) and the XMPP Multi-User Chat (MUC) extension.

Table of Contents

1. Introduction

Both the Session Initiation Protocol (SIP) [RFC3261] and the Extensible Messaging and Presence Protocol (XMPP) [RFC6120] can be used for the purpose of multiparty text chat over the Internet. To

ensure interworking between these technologies, it is important to define bidirectional protocol mappings.

The architectural assumptions underlying such protocol mappings are provided in [RFC7247], including mapping of addresses and error conditions. This document specifies mappings for multiparty text chat sessions (often called "groupchat"); specifically, this document defines a mapping between the XMPP Multi-User Chat (MUC) extension [XEP-0045] and SIP-based multiparty chat using Message Session Relay Protocol [RFC4975] as specified in [I-D.ietf-simple-chat].

Both MUC and MSRP contain a large set of features, such as the ability to administer rooms, kick and ban users, reserve a nickname within a room, change room subject, enable room moderation, and destroy the room. This document covers only a basic subset of groupchat features: joining a room, establishing or changing (but not permanently registering) a room nickname, modifying presence information within the room, sending a message to all participants, sending a private message to a single participant, inviting another user to the room, and leaving the room. Future documents might define mappings for additional features beyond this set.

2. Intended Audience

The documents in this series are intended for use by software developers who have an existing system based on one of these technologies (e.g., SIP), and would like to enable communication from that existing system to systems based on the other technology (e.g., XMPP). We assume that readers are familiar with the core specifications for both SIP [RFC3261] and XMPP [RFC6120], with the base document for this series [RFC7247], and with the following groupchat-related specifications:

- Multi-party Chat Using MSRP [I-D.ietf-simple-chat]

- Multi-User Chat [XEP-0045]

3. Terminology

A number of technical terms used here are defined in [RFC3261], [RFC4975], [RFC6120], and [XEP-0045]. The term "JID" is short for "Jabber ID".

In flow diagrams, MSRP traffic is shown using arrows such as "&&&>", SIP traffic is shown using arrows such as "***>", XMPP traffic is shown using arrows such as "...>".

In protocol flows and examples, provisional SIP responses have been elided for the sake of brevity.

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

4. Architectural Assumptions

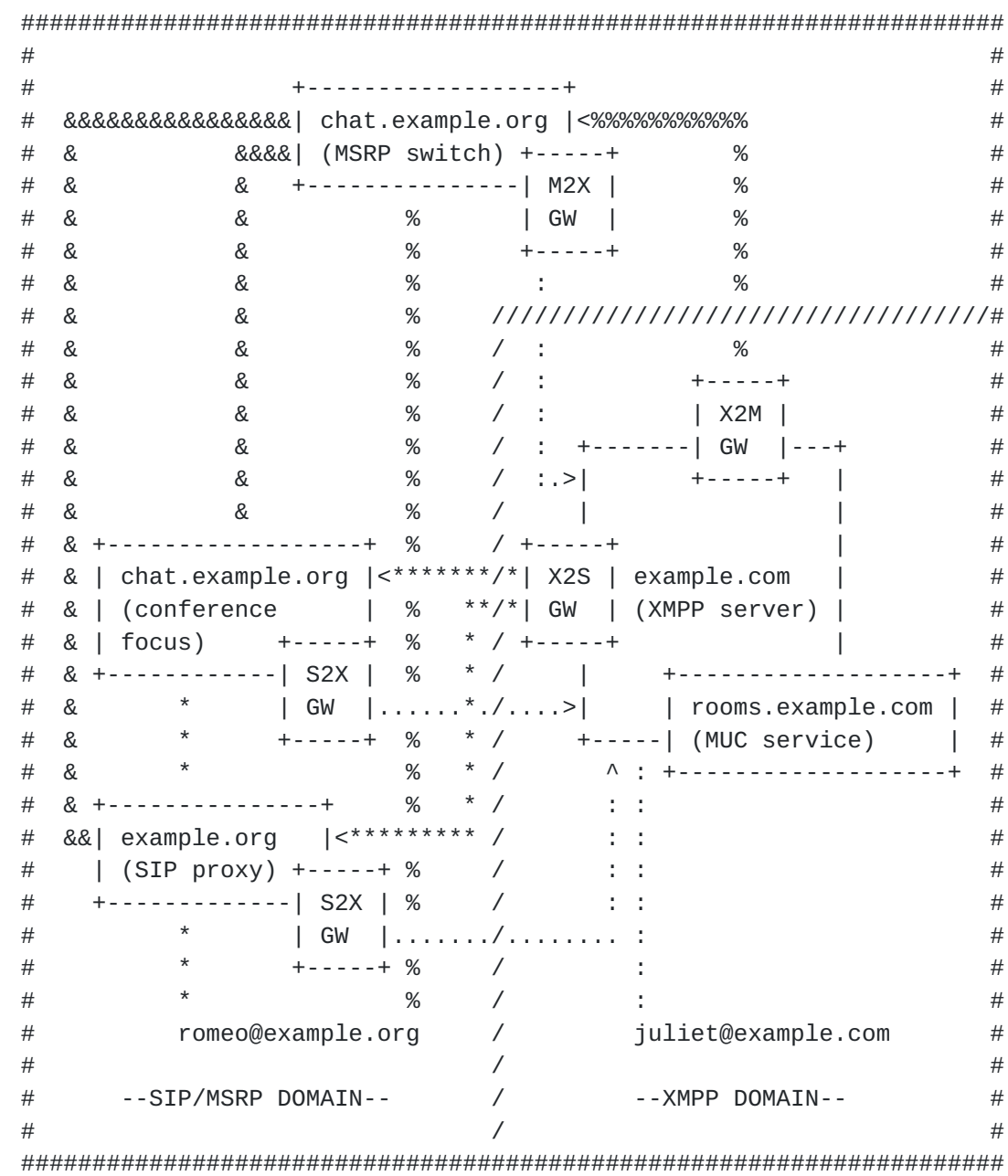
XMPP and MSRP differ in their assumptions regarding groupchat traffic. In XMPP, a message of type 'groupchat' is just another stanza, and is handled directly by an XMPP server or routed to an associated server component for multi-user chat. By contrast, in MSRP sessions (including groupchat sessions) are considered to be a type of media (similar to audio/video sessions): signaling to set up, manage, and tear down the session is handled by a "conference focus" (here we assume via SIP) but the session data itself is handled by a separate entity called an MSRP switch. How the conference focus and MSRP switch communicate is a matter of implementation and deployment.

An architectural diagram for a possible gateway deployment is shown below, where the entities have the following significance:

- romeo@example.org -- a SIP user.
- romeo@example.org;gr=c4pul3ts0rch4rd -- a particular endpoint associated with the SIP user.
- example.org -- a SIP-based conference focus with an associated MSRP switch (chat.example.org) and signaling gateway ("S2X GW").
- chat.example.org -- an MSRP switch with an associated gateway ("M2X GW") to XMPP.
- montague@chat.example.org -- a conference at an MSRP switch; not shown in diagram.
- juliet@example.com -- an XMPP user.
- juliet@example.com/myv3ry0wnb4lc0ny -- a particular endpoint associated with the XMPP user.
- example.com -- an XMPP server with an associated gateway ("X2S GW") to SIP and another gateway ("X2M GW") to MSRP.
- muc.example.com -- an XMPP MUC service associated with example.com.

- capulet@muc.example.com -- a chatroom at an XMPP MUC service; not shown in diagram.

These are logical entities, and several of them might be co-located in the same physical entity. For example, the SIP conference focus and MSRP switch and associated gateways, or the XMPP server and MUC service and associated gateways, might be part of the same deployed code. In addition, it is likely that an XMPP service would not have separate gateways for XMPP-to-SIP translation and XMPP-to-MSRP translation, but would instead have a single gateway.



Legend:
. = XMPP
% = MSRP
* = SIP
& = unstandardized communication paths
/ = separation of administrative domains

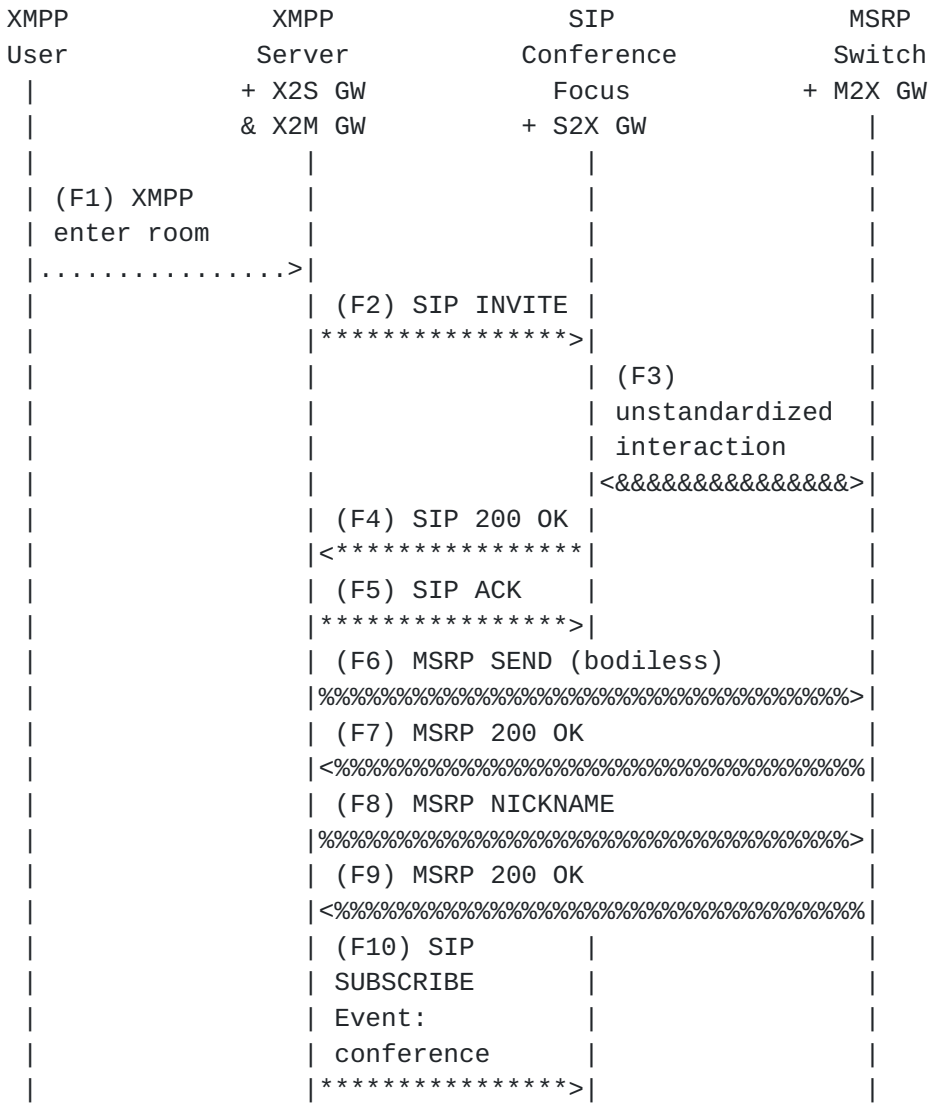
Figure 1: Logical Deployment Architecture

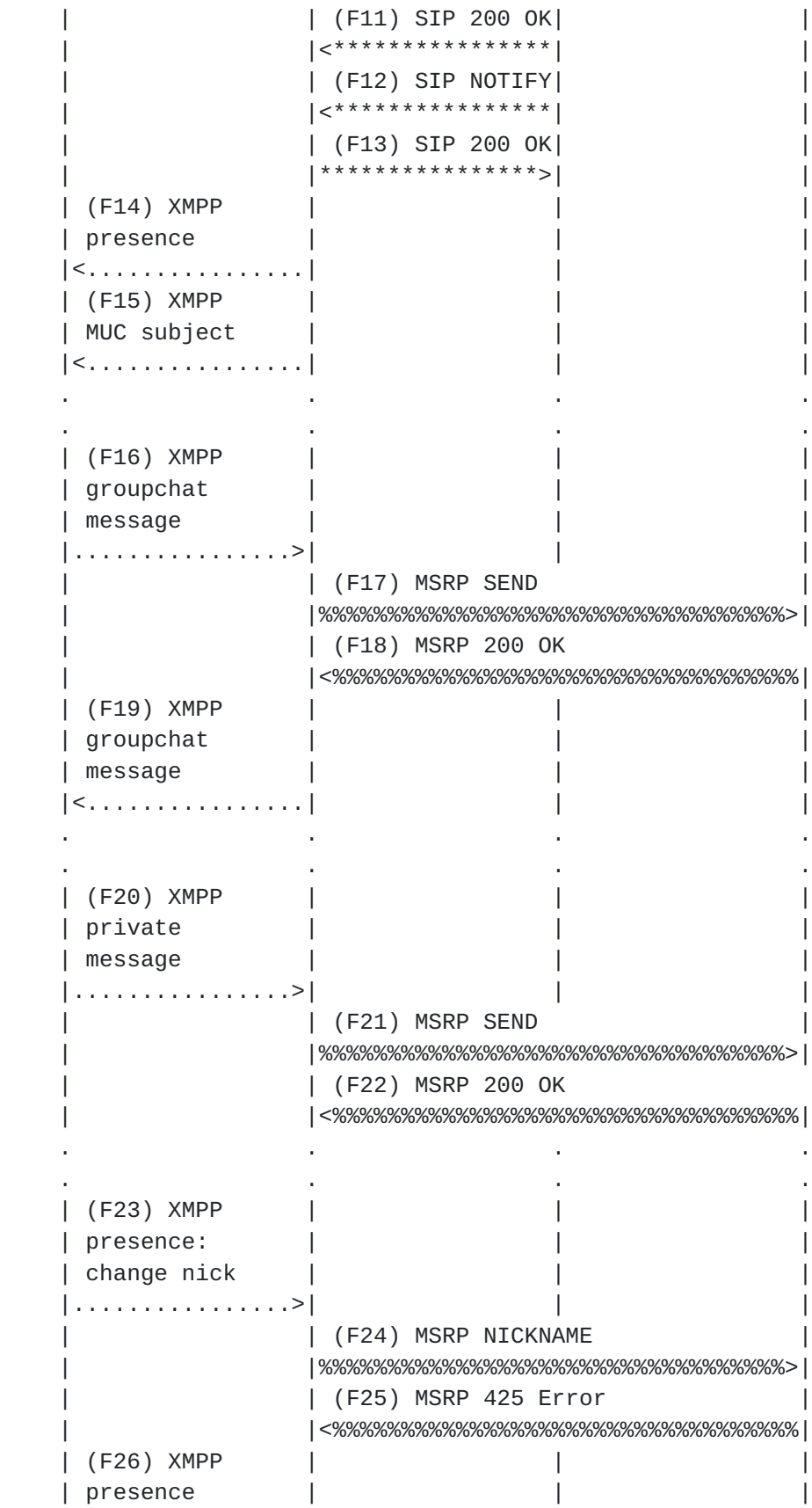
In SIP, there is no necessity for a SIP user such as romeo@example.org to make use of his SIP proxy in order to join a

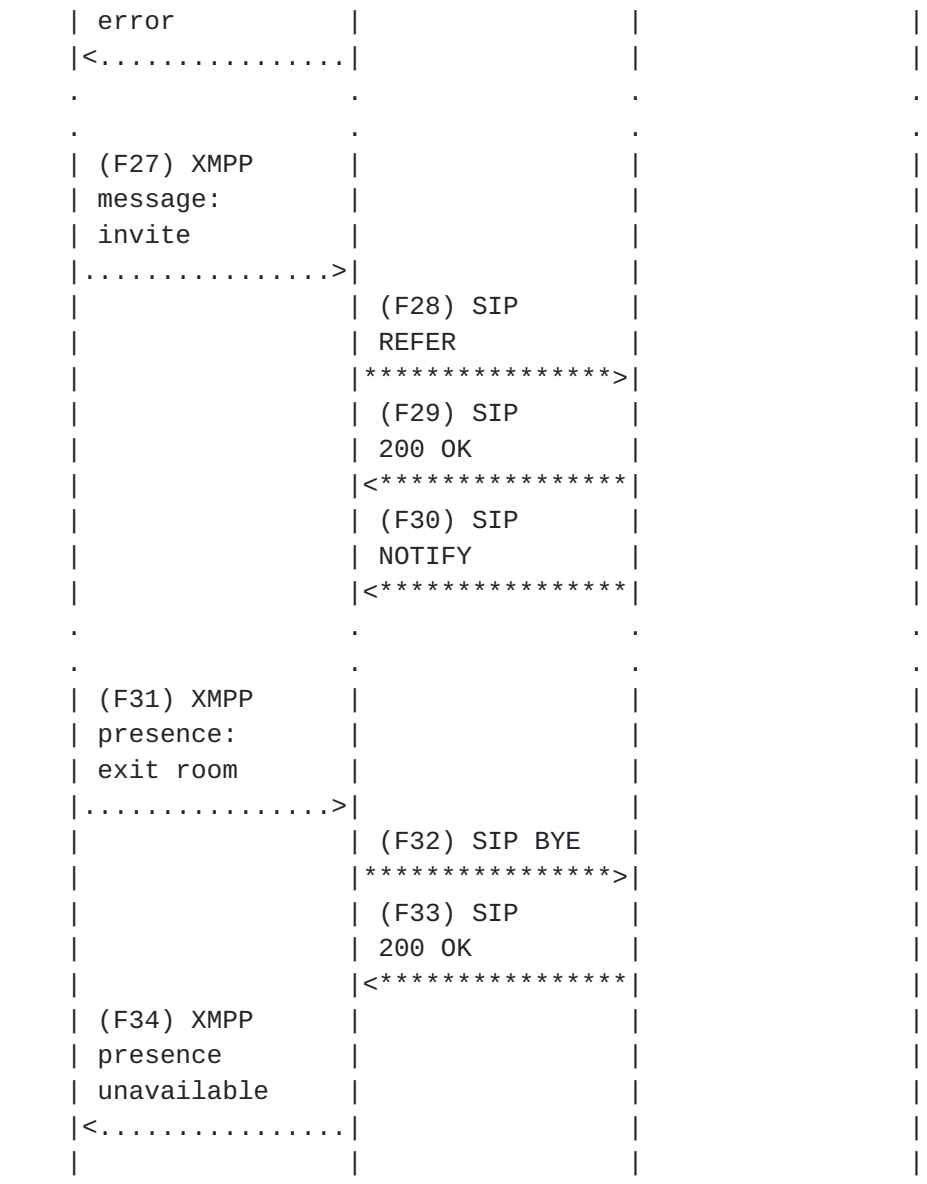
chatroom on the XMPP network; for example, he could try to directly find a SIP service at example.com or independently locate a SIP-to-XMPP gateway. Although as a simplifying assumption this document shows the more expected path of using one's "home" SIP proxy and shows gateways as associated with the sending domain, nothing in this document ought to be construed as discouraging other deployment architectures or communication paths (e.g., services hosting own inbound gateways).

5. XMPP MUC to MSRP Multi-party Messaging Session

This section describes how to map an XMPP MUC session to an MSRP Multi-party Messaging session. The following diagram outlines the overall protocol flow of a sample session, which includes some optional exchanges (such as sending messages, changing nickname, and inviting another user).




(F11) SIP 200 OK
(F12) SIP NOTIFY
(F13) SIP 200 OK
(F14) XMPP presence
(F15) XMPP MUC subject
(F16) XMPP groupchat message
(F17) MSRP SEND
(F18) MSRP 200 OK
(F19) XMPP groupchat message
(F20) XMPP private message
(F21) MSRP SEND
(F22) MSRP 200 OK
(F23) XMPP presence: change nick
(F24) MSRP NICKNAME
(F25) MSRP 425 Error
(F26) XMPP presence



Detailed protocol flows and mappings are provided in the following sections.

5.1. Enter Room

As defined in the XMPP Multi-User Chat (MUC) specification [XEP-0045], when an XMPP user (say, "juliet@example.com") wants to join a groupchat room (say, "montague@chat.example.org"), she sends a directed <presence/> stanza [RFC6121] to that chat room. In her request she also specifies the nickname she wants to use within the room (say, "JuliC"); in XMPP this room nickname is the resourcepart of an occupant JID (thus "montague@chat.example.org/JuliC"). The joining client signals its ability to speak the multi-user chat

protocol by including in the initial presence stanza an empty <x/> element qualified by the 'http://jabber.org/protocol/muc' namespace.

Example 1: Juliet enters room (F1)

```
|  <presence from='juliet@example.com/myv3ry0wnb4lc0ny'
|            to='montague@chat.example.org/JuliC'>
|    <x xmlns='http='http://jabber.org/protocol/muc'/>
|  </presence>
```

Upon receiving such a presence stanza, the XMPP server needs to determine the identity of the domainpart in the 'to' address, which it does by following the procedures discussed in [RFC7247]. Here we assume that the XMPP server has determined the domain is serviced by a SIP server, that it contains or has available to it an XMPP-to-SIP gateway or connection manager (which enables it to speak natively to SIP servers), and that it hands off the presence stanza to the XMPP-to-SIP gateway.

Because a multi-user chat service accepts the presence stanza shown above as a request to enter a room, the XMPP-to-SIP gateway transforms it into a SIP INVITE request.

Example 2: SIP mapping of room join (F2)

```
|  INVITE sip:montague@chat.example.org SIP/2.0
|  To: <sip:montague@chat.example.org>
|  From: "Juliet" <sip:juliet@example.com>
|  Contact: <sip:juliet@example.com>;gr=myv3ry0wnb4lc0ny
|  Call-ID: BC466C1C-E01D-4FD1-B766-9AD174BAF2E7
|  Content-Type: application/sdp
|  Content-Length: ...
|
|  c=IN IP4 x2s.example.org
|  m=message 7654 TCP/MSRP *
|  a=accept-types:text/cpim
|  a=accept-wrapped-types:text/plain text/html
|  a=path:msrp://x2m.example.com:7654/jshA7weztas;tcp
|  a=chatroom:nickname private-messages
```

Here the Session Description Protocol offer specifies the XMPP-to-MSRP gateway on the XMPP side (in the SDP "path" attribute specified in [RFC4975]) as well as other particulars of the session.

There is no direct mapping for the MSRP URIs. In fact MSRP URIs identify a session of instant messages at a particular device; they are ephemeral and have no meaning outside the scope of that session. The authority component of the MSRP URI here MUST

contain the XMPP-to-MSRP gateway hostname or numeric IP address and an explicit port number.

As specified in [RFC7247], the mapping of XMPP syntax elements to SIP and [RFC4566] syntax elements is as shown in the following table.

Table 1: Message syntax mapping from XMPP to SIP/SDP

XMPP Element or Attribute	SIP Header or SDP Contents
from	From
to (without the /nick)	To

Here we assume that the SIP conference focus accepts the session establishment. The Contact header field of the SIP 200 OK response includes the 'isfocus' feature tag specified in [RFC4353] along with other relevant feature tags. The conference focus also includes an answer session description that acknowledges the choice of media, specifies the MSRP URI of the switch (in the "path" attribute specified in [RFC4975]), and contains the extensions specified in [I-D.ietf-simple-chat].

Example 3: Chat room accepts session establishment (F4)

```
|  SIP/2.0 200 OK
|  From: "Juliet" <sip:juliet@example.com>;tag=786
|  To: <sip:montague@chat.example.org>
|  Call-ID: BC466C1C-E01D-4FD1-B766-9AD174BAF2E7
|  Contact: <sip:montague@chat.example.org;transport=tcp>;isfocus
|  Content-Type: application/sdp
|  Content-Length: ...
|
|  v=0
|  c=IN IP4 example.org
|  s=-
|  m=message 12763 TCP/MSRP *
|  a=accept-types:message/cpim
|  a=accept-wrapped-types:text/plain text/html
|  a=path:msrp://chat.example.org:12763/kjhd37s2s20w2a;tcp
|  a=chatroom:nickname private-messages
```

Upon receiving such a response, the XMPP-to-SIP gateway sends a SIP ACK to the conference focus on behalf of the joining user.

Example 4: Gateway sends ACK to conference focus (F5)

```
|  ACK sip:montague@chat.example.org SIP/2.0
|  To: <sip:montague@chat.example.org>;tag=087js
|  From: "Juliet" <sip:juliet@example.com>;tag=786
|  Call-ID: BC466C1C-E01D-4FD1-B766-9AD174BAF2E7
```

In accordance with [RFC4975], the gateway sends a bodiless MSRP message (F6) to the switch immediately upon establising the connection, and the switch acknowledges the that message (F7).

5.2. Set Nickname

If the chat room server accepted the session, the XMPP-to-MSRP gateway sets up the nickname as received in the presence stanza (i.e., the resourcepart of the 'to' address, such "JuliC" in "montague@chat.example.org/JuliC"). This is done using the extension specified in [I-D.ietf-simple-chat].

Example 5: Gateway sets up nickname (F8)

```
|  MSRP a786hjs2 NICKNAME
|  To-Path: msrp://montague@chat.example.org:12763/kjhd37s2s20w2a;tcp
|  From-Path: msrp://x2m.example.com:7654/jshA7weztas;tcp
|  Use-Nickname: "JuliC"
|  -------a786hjs2
```

The MSRP switch analyzes the existing allocation of nicknames, accepts the nickname proposal, and answers with a 200 response.

Example 6: MSRP switch accepts nickname proposal (F9)

```
|  MSRP a786hjs2 200 OK
|  To-Path: msrp://x2m.example.com:7654/jshA7weztas;tcp
|  From-Path: msrp://montague@chat.example.org:12763/kjhd37s2s20w2a
|             ;tcp
|  -------a786hjs2
```

This section assumes that the nickname request is successful. The error flow resulting from a nickname conflict is described under Section 5.6.

5.3. Conference Subscription

As mentioned in [I-D.ietf-simple-chat], the joining user will typically also subscribe to a conference event package (see [RFC4575] and [RFC6502]) at the focus. Although such a subscription is not required by [I-D.ietf-simple-chat], in practice the temporary and

context-dependent presence subscriptions and room rosters involved in joining an XMPP MUC room are best mapped to the conference event package.

Example 7: Gateway subscribes to the conference (F10)

```
|  SUBSCRIBE sip:montague@chat.example.org SIP/2.0
|  To: <sip:montague@chat.example.org>
|  From: "Juliet" <sip:juliet@example.com>
|  Contact: <sip:juliet@example.com>;gr=myv3ry0wnb4lc0ny
|  Call-ID: A260DEBD-4F1F-45D1-A2C0-3696636F6417
|  Event: conference
|  Expires: 600
|  Accept: application/conference-info+xml
|  Allow-Events: conference
|  Content-Length: 0
```

The focus will accept or reject the request based on local policy.

Example 8: Focus accepts subscription request (F11)

```
|  SIP/2.0 200 OK
|  From: <sip:montague@chat.example.org>
|  To: "Juliet" <sip:juliet@example.com>
|  Call-ID: A260DEBD-4F1F-45D1-A2C0-3696636F6417
|  Contact: <sip:montague@chat.example.org;transport=tcp>;isfocus
|  Expires: 600
|  Content-Length: 0
```

If the conference focus accepts the request to enter a room, the XMPP user expects to receive back presence information from all the existing occupants of the room. To make this happen, the XMPP-to-SIP gateway subscribes to the Conference Event package [RFC4575] at the focus.

5.4. Presence Broadcast

When the Conference Event package subscription is completed, the focus sends to the XMPP-to-SIP gateway a NOTIFY containing the presence information of all the existing occupants, represented using the format defined in [RFC4575].

Example 9: Conference focus sends presence information (F12)

```
|  NOTIFY sip:montague@chat.example.org SIP/2.0
|  To: "Juliet" <sip:juliet@example.com>;gr=myv3ry0wnb4lc0ny
|  From: <sip:montague@chat.example.org>;tag=a3343df32
|  Call-ID: 6F563BA1-8177-4CED-8710-78D4D9593B08
```

```
|  Event: conference
|  Subscription-State: active;expires=3600
|  Content-Type: application/conference-info+xml
|  Content-Length: ...
|
|  <conference-info version="0" state="full"
|      entity="sip:3402934234@chat.example.org">
|    <conference-description>
|      <subject>Today in Verona</subject>
|      <conf-uris>
|        <entry>
|          <uri>tel:+18882934234</uri>
|        </entry>
|      </conf-uris>
|    </conference-description>
|    <users>
|      <user entity="sip:montague@chat.example.org;gr=Romeo"
|            state="full">
|        <display-text>Romeo</display-text>
|        <roles>
|          <entry>participant</entry>
|        </roles>
|        <associated-aors>
|          <entry>
|            <uri>xmpp:romeo@example.org/c4pul3ts0rch4rd</uri>
|          </entry>
|        </associated-aors>
|        <endpoint entity="sip:montague@chat.example.org;gr=Romeo"
|                  state="full">
|          <status>connected</status>
|          <joining-info>
|            <when>2013-12-12T10:01:03.691128+01:00</when>
|          </joining-info>
|          <media id="211835820">
|            <type>message</type>
|          </media>
|        </endpoint>
|      </user>
|      <user entity="sip:montague@chat.example.org;gr=Ben"
|            state="full">
|        <display-text>Ben</display-text>
|        <roles>
|          <entry>participant</entry>
|        </roles>
|        <endpoint entity="sip:montague@chat.example.org;gr=Ben"
|                  state="full">
|          <status>connected</status>
|          <media id="211835821">
```

```
|            <type>message</type>
|          </media>
|        </endpoint>
|      </user>
|      <user entity="sip:montague@chat.example.org;gr=JuliC"
|            state="full">
|        <display-text>JuliC</display-text>
|        <roles>
|          <entry>participant</entry>
|        </roles>
|         <endpoint entity="sip:montague@chat.example.org;gr=JuliC"
|                   state="full">
|           <status>connected</status>
|           <media id="211835822">
|             <type>message</type>
|           </media>
|         </endpoint>
|      </user>
|    </users>
|  </conference-info>
```

The following table shows the syntax mapping from the RFC 4575 payload to the XMPP participant list. (Mappings for elements not mentioned are undefined.)

Table 2: Participant list mapping

RFC 4575 Element	XMPP Element or Attribute
conference-info entity	room JID
conference subject	room subject
user entity	occupant JID
user display-text / nickname	participant nickname
endpoint entity	occupant JID
user associated-aors	user full JID (if avail.)

Upon receiving such a response, the XMPP-to-SIP gateway sends a SIP 200 OK response to the conference focus (example not shown) and translate the participant list into a series of XMPP presence stanzas.

Example 10: XMPP mapping of chatroom presence (F14)

```
|  <presence from='montague@chat.example.org/Romeo'
|            to='juliet@example.com/myv3ry0wnb4lc0ny'>
|    <x xmlns='http://jabber.org/protocol/muc#user'>
|      <item affiliation='none' role='participant'/>
|    </x>
|  </presence>

|  <presence from='montague@chat.example.org/Ben'
|            to='juliet@example.com/myv3ry0wnb4lc0ny'>
|    <x xmlns='http://jabber.org/protocol/muc#user'>
|      <item affiliation='none' role='participant'/>
|    </x>
|  </presence>

|  <presence from='montague@chat.example.org/JuliC'
|            to='juliet@example.com/myv3ry0wnb4lc0ny'>
|    <x xmlns='http://jabber.org/protocol/muc#user'>
|      <item affiliation='none' role='participant'/>
|      <status code='110'/>
|    </x>
|  </presence>
```

If the NOTIFY included a subject, the gateway converts that into a separate XMPP message.

Example 11: XMPP mapping of chatroom subject (F15)

```
|  <message from='montague@chat.example.org/mayor'
|            to='juliet@example.com/myv3ry0wnb4lc0ny'
|            id='mbh2vd68'>
|    <subject>Today in Verona</subject>
|  </message>
```

The mapping of SIP and [RFC4575] payload syntax elements to XMPP syntax elements is as shown in the following table. (Mappings for elements not mentioned are undefined.)

Table 3: Message syntax mapping from SIP to XMPP

SIP Header or RFC4575 Contents	XMPP Element or Attribute
<user entity=...>	from
To + / <display-text>	to
roles	role
'none'	affiliation

5.5. Exchange Messages

Once the user has joined the chatroom, the user can exchange an unbounded number of messages, both public and private.

The mapping of XMPP syntax elements to MSRP syntax elements is as shown in the following table. (Mappings for elements not mentioned are undefined.)

Table 4: Message syntax mapping from XMPP Message to MSRP

XMPP Element or Attribute	CPIM Header
to	To
from	From
<body/>	body of the SEND request

5.5.1. Send a Message to All Occupants

When Juliet wants to sends a message to all other occupants in the room, she sends a message of type "groupchat" to the <room@service> itself (in our example, <montague@chat.example.org>).

The following examples show an exchange of a public message.

Example 12: Juliet sends message to all occupants (F16)

```
|  <message from='juliet@example.com/myv3ry0wnb4lc0ny'
|           to='montague@chat.example.org'
|           type='groupchat'
|           id='lzfed24s'>
|        <body>Who knows where Romeo is?</body>
|  </message>
```

Upon receiving such a message, the XMPP-to-MSRP gateway translates it into an MSRP SEND message.

Example 13: Gateway maps XMPP message to MSRP (F17)

```
|  MSRP a786hjs2 SEND
|  To-Path: msrp://chat.example.org:12763/kjhd37s2s20w2a;tcp
|  From-Path: msrp://x2m.example.com:7654/jshA7weztas;tcp
|  Message-ID: 87652491
|  Byte-Range: 1-*/*
|  Content-Type: message/cpim
|
|  To: <sip:montague@chat.example.org>
|  From: "Juliet" <sip:juliet@example.com>
|  DateTime: 2008-10-15T15:02:31-03:00
|  Content-Type: text/plain
|
|  Who knows where Romeo is?
|  -------a786hjs2$
```

Upon receiving the SEND request, if the request either contains a Failure-Report header field value of "yes" or does not contain a Failure-Report header at all, the MSRP switch immediately generates and sends a response.

Example 14: MSRP switch returns 200 OK (F18)

```
|  MSRP d93kswow 200 OK
|  To-Path: msrp://x2m.example.com:7654/jshA7weztas;tcp
|  From-Path: msrp://chat.example.org:12763/kjhd37s2s20w2a;tcp
|  -------d93kswow$
```

Since an XMPP MUC room could be moderated and an XMPP user cannot be sure whether her message has been accepted without receiving it back from the server, [XEP-0045] states that the sender needs to receive a reflected copy of the message it sent. So in this scenario the XMPP-to-MSRP gateway has to reflect the message back to the sender. This prodedure only applies to XMPP endpoints.

Example 15: Gateway reflects message to XMPP user (F19)

```
|  <message from='montague@chat.example.org/JuliC'
|           to='montague@chat.example.org'
|           type='groupchat'
|           id='ix51z73m'>
|        <body>Who knows where Romeo is?</body>
|  </message>
```

5.5.2. Send a Private Message

Since each occupant has a unique JID, Juliet can send a "private message" to a selected occupant through the service by sending a message to the user's occupant JID. The XMPP message type ought to be "chat" (and is not allowed to be "groupchat").

The following examples show an exchange of a private message.

Example 16: Juliet sends private message (F20)

```
|  <message from='juliet@example.com/myv3ry0wnb4lc0ny'
|           to='montague@chat.example.org/Romeo'
|           type='chat'
|           id='6sfln45q'>
|        <body>O Romeo, Romeo! wherefore art thou Romeo?</body>
|  </message>
```

Upon receiving such a message, the XMPP-to-MSRP gateway translates it into an MSRP SEND message.

Example 17: Gateway maps private message from XMPP to MSRP (F21)

```
|  MSRP a786hjs2 SEND
|  To-Path: msrp://chat.example.org:12763/kjhd37s2s20w2a;tcp
|  From-Path: msrp://x2m.example.com:7654/jshA7weztas;tcp
|  Message-ID: 87652491
|  Byte-Range: 1-*/*
|  Content-Type: message/cpim
|
|  To: <sip:montague@chat.example.org>;gr=Romeo
|  From: <sip:juliet@example.org>;gr=myv3ry0wnb4lc0ny
|  DateTime: 2008-10-15T15:02:31-03:00
|  Content-Type: text/plain
|
|  O Romeo, Romeo! wherefore art thou Romeo?
|  -------a786hjs2$
```

After acknowledging the message by sending an MSRP 200 OK (step F25, not shown), the MSRP switch is responsible for sending the message to the intended recipient. When doing so, it modifies the "From" header to the sender's address within the chatroom.

Example 18: Switch sends private message to SIP user

```
|  MSRP a786hjs2 SEND
|  To-Path: msrp://chat.example.org:12763/kjhd37s2s20w2a;tcp
|  From-Path: msrp://x2m.example.com:7654/jshA7weztas;tcp
|  Message-ID: 87652491
|  Byte-Range: 1-*/*
|  Content-Type: message/cpim
|
|  To: <sip:romeo@example.org>
|  From: <sip:montague@chat.example.org>;gr=JuliC
|  DateTime: 2008-10-15T15:02:31-03:00
|  Content-Type: text/plain
|
|  O Romeo, Romeo! wherefore art thou Romeo?
|  -------a786hjs2$
```

Note: If an XMPP-to-MSRP gateway has support for private messaging, it MUST advertise that fact by adding a "private-messages" value to the a=chatroom SDP attribute it sends to the conference focus, as specified in [I-D.ietf-simple-chat].

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

5.6. Change Nickname

The XMPP user might want to change her nickname. She can do so by sending an updated presence stanza to the room, containing a new nickname.

Example 19: Juliet changes her nickname (F22)

```
|  <presence from='juliet@example.com/myv3ry0wnb4lc0ny'
|            to='montague@chat.example.org/CapuletGirl'/>
```

So far we have assumed that the requested nickname did not conflict with any existing nicknames. The following text describes the handling of a nickname conflict.

The MSRP switch analyzes the existing allocation of nicknames, and detects that the nickname proposal is already provided to another participant. In this case the MSRP switch answers with a 425 response.

Example 20: MSRP switch does not accept nickname proposal (F24)

```
|  MSRP a786hjs2 425 Nickname usage failed
|  To-Path: msrp://x2m.example.com:7654/jshA7weztas;tcp
|  From-Path: msrp://chat.example.org:12763/kjhd37s2s20w2a;tcp
|  -------a786hjs2
```

Upon receiving such a response, the XMPP-to-MSRP gateway translates it into an XMPP presence stanza of type "error" specifying a <conflict/> error condition (which implies that the XMPP client will then need to choose another nickname and repeat the process of joining).

Example 21: Conflict error for nickname (F26)

```
|  <presence from='montague@chat.example.org/JuliC'
|            to='juliet@example.com/myv3ry0wnb4lc0ny'
|            type='error'>
|    <x xmlns='http='http://jabber.org/protocol/muc'/>
|    <error type='cancel' by='montague@chat.example.org'>
|      <conflict xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
|    </error>
|  </presence>
```

Alternatively, the gateway might generate a new nickname request on behalf of the XMPP user, thus shielding the XMPP client from handling the conflict error.

5.7. Invite Another User to a Room

In XMPP there are two methods for inviting another user to a room: direct invitations [XEP-0249] (sent directly from the user's real JID outside the room to the invitee's real JID) and mediated invitations (sent through the room from the user's occupant JID to the invitee's JID). In this document we cover mediated invitations only.

For example, if Juliet decides to invite Benvolio to the room, she sends a message stanza with an invite and Benvolio's JID (which could be his real JID or an occupant JID in another room).

Example 22: Juliet invites Benvolio to the room (F27)

```
|  <message from='juliet@example.com/myv3ry0wnb4lc0ny'
|           id='nzd143v8'
|           to='montague@chat.example.org'>
|    <x xmlns='http://jabber.org/protocol/muc#user'>
|      <invite to='benvolio@example.com'/>
|    </x>
|  </message>
```

The XMPP-to-SIP gateway then sends a SIP REFER request to the conference focus indicating who needs to be invited in the Refer-To header, as per [RFC4579] (sec 5.5)

Example 23: SIP mapping of invite (F28)

```
|  REFER sip:montague@chat.example.org SIP/2.0
|  To: <sip:montague@chat.example.org>
|  From: "Juliet" <sip:juliet@example.com>;tag=5534562
|  Call-ID: 7FFCD8F7-EEB6-4506-A566-80D3CFC4C6E6
|  Contact: <sip:juliet@example.com>;gr=myv3ry0wnb4lc0ny
|  Accept: message/sipfrag
|  Refer-To: <sip:benvolio@example.com>
|  Supported: replaces
|  Content-Length: 0
```

The conference focus then acknowledges the SIP REFER request with a 200 OK response (step F27, not shown).

The progress of the invitation will be tracked by the received NOTIFY requests as per [RFC3515].

Example 24: Progress notification for invitation (F30)

```
|  NOTIFY sip:juliet@example.com SIP/2.0
|  To: <sip:juliet@example.com>;tag=5534562
|  From: <sip:montague@chat.example.org>;tag=18747389
|  Call-ID: 7FFCD8F7-EEB6-4506-A566-80D3CFC4C6E6
|  Max-Forwards: 70
|  Event: refer
|  Subscription-State: active;expires=60
|  Contact: <sip:montague@chat.example.org;transport=tcp>;isfocus
|  Content-Type: message/sipfrag;version=2.0
|  Content-Length: ...
```

Note: Implementers might want to be aware that work is underway to modify the way in which REFER requests handle SIP notifications

([I-D.sparks-sipcore-refer-clarifications] and
[I-D.sparks-sipcore-refer-explicit-subscription]).

5.8. Exit Room

If Juliet decides to exit the chatroom, her client sends a directed presence stanza of type "unavailable" to the occupant JID she is currently using in the room (here <montague@chat.example.org/JuliC>).

Example 25: Juliet exits room (F31)

```
|  <presence from='juliet@example.com/myv3ry0wnb4lc0ny'
|            to='montague@chat.example.org/JuliC'
|            type='unavailable'/>
```

Upon receiving such a stanza, the XMPP-to-SIP gateway terminates the SIP session by sending a SIP BYE to the conference focus and the conference focus responds with a SIP 200 OK (steps F30 and F31, not shown).

Juliet can include a custom exit message in the presence stanza of type "unavailable", in which case it is broadcast to other participants using the methods described above.

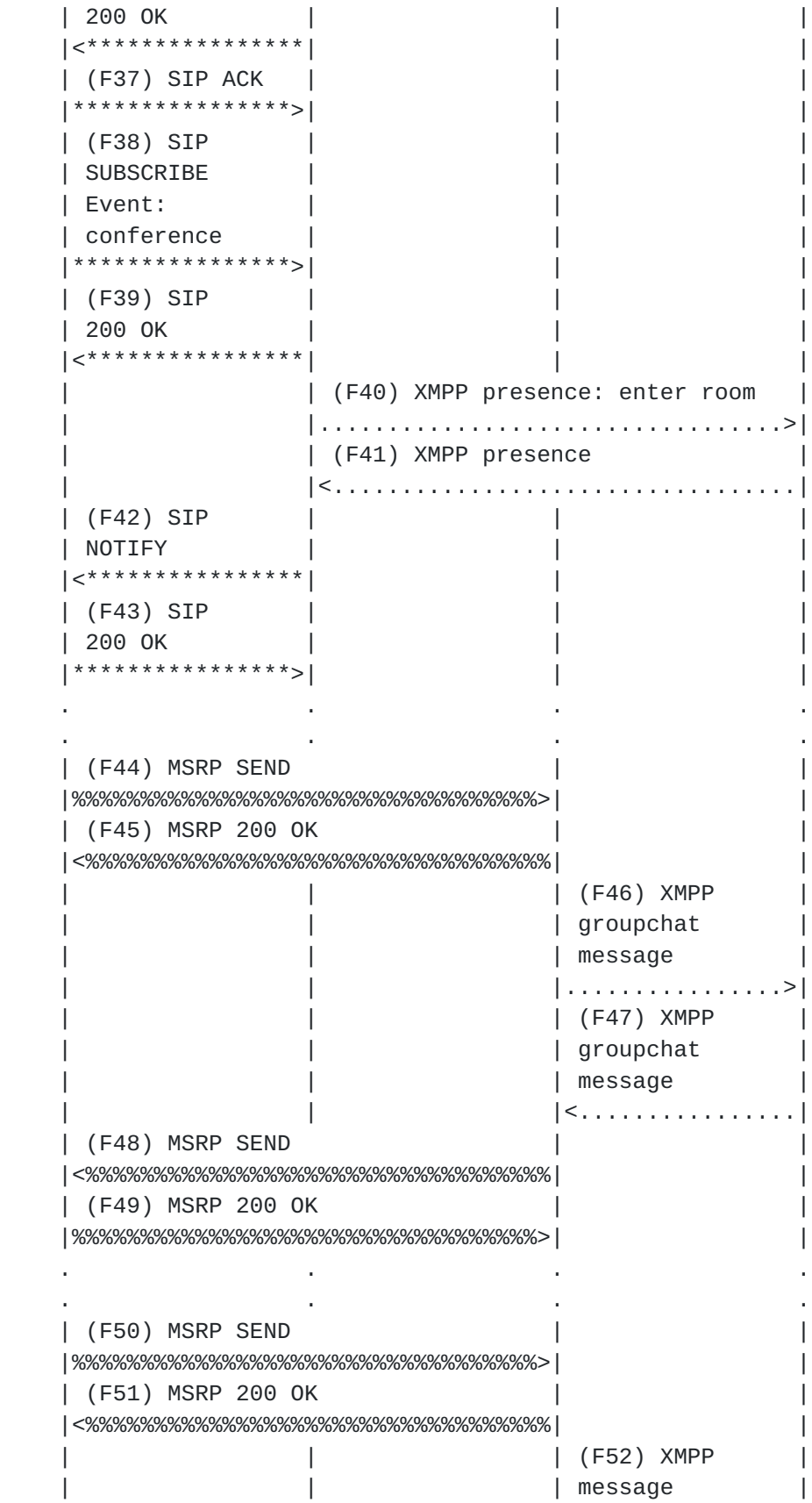
Example 26: Juliet exits the chatroom (F31)

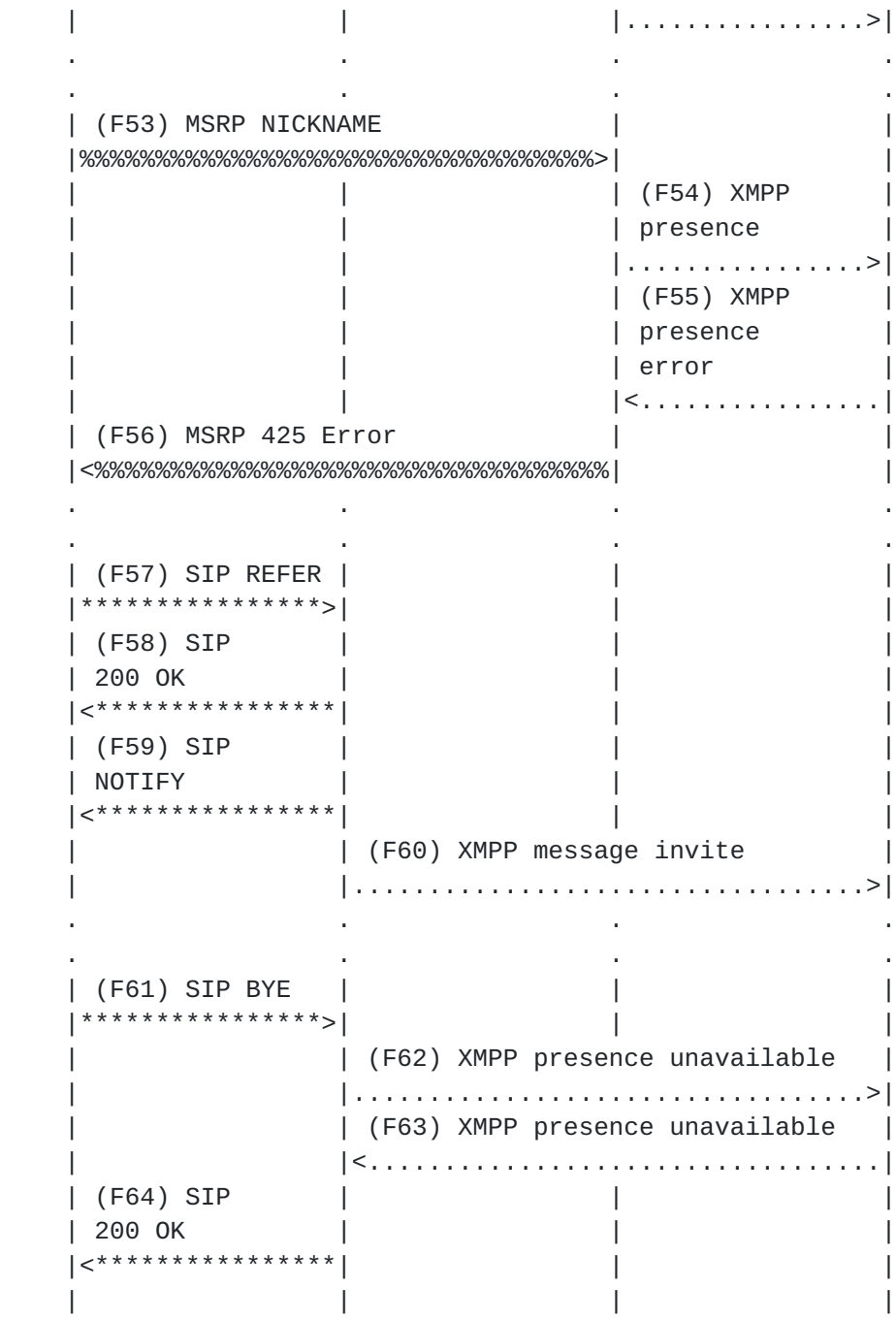
```
|  <presence from='juliet@example.com/myv3ry0wnb4lc0ny'
|            to='montague@chat.example.org/JuliC'
|            type='unavailable'>
|    <status>Time to go!</status>
|  </presence>
```

6. MSRP Multi-party Messaging Session to XMPP MUC

This section describes how to map a Multi-party Instant Message (IM) MSRP session to an XMPP Multi-User Chat (MUC) session. As before, the following diagram outlines the overall protocol flow of a sample session, which includes some optional exchanges (such as sending messages, changing nickname, and inviting another user).

```
SIP              SIP              MSRP             XMPP
User             Proxy            Switch           Server
 |             + S2X GW         + M2X GW       + MUC & GWs
 |                 |                |                |
 | (F33) SIP       |                |                |
 | INVITE          |                |                |
 |****************>|                |                |
 | (F36) SIP       |                |                |
```


200 OK
(F37) SIP ACK
(F38) SIP SUBSCRIBE Event: conference
(F39) SIP 200 OK
(F40) XMPP presence: enter room
(F41) XMPP presence
(F42) SIP NOTIFY
(F43) SIP 200 OK
(F44) MSRP SEND
(F45) MSRP 200 OK
(F46) XMPP groupchat message
(F47) XMPP groupchat message
(F48) MSRP SEND
(F49) MSRP 200 OK
(F50) MSRP SEND
(F51) MSRP 200 OK
(F52) XMPP message



If the XMPP presence stanza is received before the SIP SUBSCRIBE dialog is established for the "conference" event, then the server SHOULD cache the participant list until the subscription is established and delivered in a SIP NOTIFY request.

6.1. Enter Room

When the SIP user ("Romeo") wants to join a groupchat room ("capulet"), he first has to start the SIP session by sending out a SIP INVITE request containing an offered session description that includes an MSRP media line accompanied by a mandatory "path" and "chatroom" attributes. Here we assume that Romeo's user agent has been configured to be aware of an MSRP switch (chat.example.org) it can use. The MSRP media line is also accompanied by an "accept-types" attribute specifing support for a Message/CPIM top level wrapper for the MSRP message.

Example 27: SIP user starts session (F35)

```
|  INVITE sip:capulet@rooms.example.com SIP/2.0
|  From: "Romeo" <sip:romeo@example.org>
|  To: <sip:capulet@rooms.example.com>
|  Contact: <sip:romeo@example.org>;gr=c4pul3ts0rch4rd
|  Call-ID: 08CFDAA4-FAED-4E83-9317-253691908CD2
|  Content-Type: application/sdp
|  Content-Length: ...
|
|  c=IN IP4 s2x.example.org
|  m=message 7313 TCP/MSRP *
|  a=accept-types:message/cpim text/plain text/html
|  a=accept-wrapped-types:text/plain text/html
|  a=path:msrp://chat.example.org:7313/ansp71weztas;tcp
|  a=chatroom:nickname private-messages
```

Upon receiving the INVITE, the SIP proxy needs to determine the identity of the domain portion of the Request-URI or To header, which it does by following the procedures discussed in [RFC7247]. Here we assume that the SIP proxy has determined that the domain is serviced by an XMPP server, that it contains or has available to it a SIP-to-XMPP gateway or connection manager (which enables it to speak natively to XMPP servers), and that it hands off the message to the gateway.

Implementations MAY wait until the nickname is set with an MSRP NICKNAME chunk before joining the XMPP MUC or MAY choose a temporary nickname (such as the SIP From header display name) and use it to join the room. Here we assume the latter.

Example 28: SIP-to-XMPP gateway acks session (F36)

```
|  SIP/2.0 200 OK
|  From: "Romeo" <sip:romeo@example.org>
|  To: <sip:capulet@rooms.example.com>
|  Contact: <sip:rooms.example.com;transport=tcp>;isfocus
|  Call-ID: 08CFDAA4-FAED-4E83-9317-253691908CD2
|  Content-Type: application/sdp
|
|  m=message 8763 TCP/MSRP *
|  a=accept-types:message/cpim text/plain text/html
|  a=accept-wrapped-types:text/plain text/html
|  a=path:msrp://chat.example.org:8763/lkjh37s2s20w2a;tcp
|  a=chatroom:nickname private-messages
```

The SIP/MSRP user agent subscribes to a conference event package at the destination groupchat service.

Example 29: Gateway subscribes to the conference (F40)

```
|  SUBSCRIBE sip:capulet@rooms.example.com SIP/2.0
|  To: <sip:capulet@rooms.example.com>
|  From: "Romeo" <sip:romeo@example.org>
|  Contact: <sip:romeo@example.org>;gr=c4pul3ts0rch4rd
|  Call-ID: A260DEBD-4F1F-45D1-A2C0-3696636F6417
|  Event: conference
|  Expires: 600
|  Accept: application/conference-info+xml
|  Allow-Events: conference
|  Content-Length: 0
```

After the conference subscription request is acknowledged, the SIP-to-XMPP gateway sends presence from Romeo to the MUC chatroom.

Example 30: Romeo enters XMPP chatroom (F41)

```
|  <presence from='romeo@example.org'
|            to='montague@chat.example.org/Romeo'>
|    <x xmlns='http='http://jabber.org/protocol/muc'/>
|  </presence>
```

6.2. Presence Broadcast

If the MUC service is able to add the SIP/MSRP user to the room, it sends presence from all the existing occupants' room JIDs to the new occupant's full JID, including extended presence information about roles in an <x/> element.

Example 31: XMPP service sends presence from existing occupants to new occupant (F43)

```
|  <presence from='capulet@rooms.example.com/Romeo'
|            to='romeo@example.org'>
|    <x xmlns='http://jabber.org/protocol/muc#user'>
|      <item affiliation='none' role='participant'/>
|      <status='110'/>
|    </x>
|  </presence>
|
|  <presence from='capulet@rooms.example.com/Ben'
|            to='romeo@example.org'>
|    <x xmlns='http://jabber.org/protocol/muc#user'>
|      <item affiliation='none' role='participant'/>
|    </x>
|  </presence>
|
|  <presence from='capulet@rooms.example.com/JuliC'
|            to='romeo@example.org'>
|    <x xmlns='http://jabber.org/protocol/muc#user'>
|      <item affiliation='none' role='participant'/>
|    </x>
|  </presence>
```

Upon receiving these presence stanzas, if the conference focus has already completed the subscription to the Conference Event package [RFC4575], the XMPP-to-SIP gateway translates them into a SIP NOTIFY request containing the participant list (represented in the conference-info format specified in [RFC4575]).

Example 32: SIP mapping of XMPP participant presence stanzas (F44)

```
|  NOTIFY sip:romeo@example.org SIP/2.0
|  To: <sip:romeo@example.org>;tag=43524545
|  From: <sip:capulet@rooms.example.com>;tag=a3343df32
|  Call-ID: 6F563BA1-8177-4CED-8710-78D4D9593B08
|  Event: conference
|  Subscription-State: active;expires=3600
|  Content-Type: application/conference-info+xml
|  Content-Length: ...
|
|  <conference-info version="0" state="full"
|      entity="sip:capulet@rooms.example.com">
|    <conference-description>
|      <subject>Today in Verona</subject>
|      <conf-uris>
|        <entry>
```

```
|          <uri>tel:+18882934234</uri>
|          <uri>sip:capulet@rooms.example.com</uri>
|        </entry>
|      </conf-uris>
|   </conference-description>
|   <users>
|     <user entity="sip:capulet@rooms.example.com;gr=JuliC"
|           state="full">
|       <display-text>JuliC</display-text>
|       <roles>
|         <entry>participant</entry>
|       </roles>
|       <endpoint entity="sip:capulet@rooms.example.com;gr=JuliC"
|                 state="full">
|         <status>connected</status>
|         <media id="211835821">
|           <type>message</type>
|         </media>
|       </endpoint>
|     </user>
|     <user entity="sip:capulet@rooms.example.com;gr=Ben"
|           state="full">
|       <display-text>Ben</display-text>
|       <roles>
|         <entry>participant</entry>
|       </roles>
|       <endpoint entity="sip:capulet@rooms.example.com;gr=Ben"
|                 state="full">
|         <status>connected</status>
|         <media id="211835822">
|           <type>message</type>
|         </media>
|       </endpoint>
|   </users>
|  </conference-info>
```

Because the "room roster" is communicated in XMPP by means of multiple presence stanzas (one for each participant) whereas the participant list is communicated in SIP by means of a single conference-info document, the SIP-to-XMPP gateway will need to keep track of the user's SIP URI and the mapping of that URI into an XMPP address; then, based on that mapping, it will need to determine when it has received a complete room roster from the MUC room, i.e., when it has received the in-room presence of the SIP user (which according to [XEP-0045] is the last presence stanza received in the initial batch sent after joining). Once that happens, the SIP-to-XMPP gateway can construct the conference-info document containing the complete participant list and send that to the SIP user.

6.3. Exchange Messages

Once the user has joined the chat room, the user can exchange an unbounded number of messages, both public and private.

The mapping of MSRP syntax elements to XMPP syntax elements is as shown in the following table. (Mappings for elements not mentioned are undefined.)

Table 5: Message syntax mapping from MSRP Message to XMPP

CPIM Header	XMPP Element or Attribute
To	to
From	from
body of the SEND request	<body/>

6.3.1. Send a Message to All Occupants

When Romeo wants to send a message to all other occupants in the room, he sends an MSRP SEND request to <room@service> (<capulet@rooms.example.com> in our example).

The following examples show an exchange of a public message.

Example 33: Romeo sends a message to the chat room (F46)

```
|  MSRP a786hjs2 SEND
|  To-Path: msrp://room.example.com:7313/ansp71weztas;tcp
|  From-Path: msrp://chat.example.org:8763/lkjh37s2s20w2a;tcp
|  Message-ID: 87652492
|  Byte-Range: 1-*/*
|  Content-Type: message/cpim
|
|  To: <sip:capulet@rooms.example.com>
|  From: "Romeo" <sip:romeo@example.org>;gr=c4pul3ts0rch4rd
|  DateTime: 2008-10-15T15:02:31-03:00
|  Content-Type: text/plain
|
|  Romeo is here!
|  -------a786hjs2$
```

Upon receiving the SEND request, if the request either contains a Failure-Report header field value of "yes" or does not contain a Failure-Report header at all, the SIP-to-XMPP gateway immediately

translates it into an XMPP message stanza and then generate and send an MSRP response.

Example 34: XMPP mapping of message (F48)

```
|  <message from='romeo@example.org/c4pul3ts0rch4rd'
|           to='capulet@rooms.example.com'
|           type='groupchat'
|           id='8gbx1g4p'>
|    <body>Romeo is here!</body>
|  </message>
```

Example 35: MSRP response to public message (F49)

```
|  MSRP d93kswow 200 OK
|  To-Path: msrp://rooms.example.com:8763/lkjh37s2s20w2a;tcp
|  From-Path: msrp://chat.example.org:7313/ansp71weztas;tcp
|  -------d93kswow$
```

Note well that the XMPP MUC room will reflect the sender's message back to all users, including the sender. In MSRP this reflected message is unnecessary.

6.3.2. Send a Private Message

Romeo can send a "private message" to a selected occupant via the chat room service by sending a message to the occupant's room nickname.

The following examples show an exchange of a private message.

Example 36: Romeo sends a private message (F51)

```
|  MSRP a786hjs2 SEND
|  To-Path: msrp://rooms.example.com:7313/ansp71weztas;tcp
|  From-Path: msrp://chat.example.org:8763/lkjh37s2s20w2a;tcp
|  Message-ID: 87652492
|  Byte-Range: 1-*/*
|  Content-Type: message/cpim
|
|  To: <sip:capulet@rooms.example.com>;gr=JuliC
|  From: "Romeo" <sip:romeo@example.org>;gr=c4pul3ts0rch4rd
|  DateTime: 2008-10-15T15:02:31-03:00
|  Content-Type: text/plain
|
|  I am here!!!
|  -------a786hjs2$
```

The MSRP switch is responsible for transforming the "From" address into an in-room address (not shown).

Once the MSRP switch sends that message to the gateway, the gateway is responsible for translating it into XMPP syntax.

Example 37: XMPP mapping of private message (F54)

```
|  <message from='capulet@rooms.example.com/Romeo'
|           to='capulet@rooms.example.com/JuliC'
|           type='chat'
|           id='rg2ca9k7'/>
|    <body>I am here!!!</body>
|  </message>
```

6.4. Change Nickname

If Romeo decides to change his nickname within the room, he sends a new MSRP NICKNAME request. In fact modification of the nickname in MSRP is not different from the initial reservation and usage of a nickname.

Example 38: MSRP user changes nickname (F55)

```
|  MSRP a786hjs2 NICKNAME
|  To-Path: msrp://chat.example.org:7313/ansp71weztas;tcp
|  From-Path: msrp://rooms.example.com:8763/lkjh37s2s20w2a;tcp
|  Use-Nickname: "montecchi"
|  -------a786hjs2
```

Upon receiving such a message, the MSRP-to-XMPP gateway translates it into an XMPP presence stanza.

Example 39: XMPP mapping of nickname change (F56)

```
|  <presence from='romeo@example.org'
|            to='capulet@rooms.example.com/montecchi'/>
```

The XMPP server will analyze the nickname allocation and determine if the requested nickname is available. In case the nickname is not available or not usable, the server will generate a presence stanza of type "error" specifying a <conflict/> error condition.

Example 40: XMPP conflict error for nickname (F57)

```
|  <presence from='capulet@rooms.example.com/Romeo'
|            to='romeo@example.org/c4pul3ts0rch4rd'
|            type='error'>
|    <x xmlns='http='http://jabber.org/protocol/muc'/>
|    <error type='cancel' by='capulet@rooms.example.com'>
|      <conflict xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
|    </error>
|  </presence>
```

Upon receiving this stanza, the XMPP-to-MSRP gateway will reply to the NICKNAME request with code 425

Example 41: Gateway translates XMPP nickname conflict to MSRP error code (F58)

```
|  MSRP a786hjs2 425 Nickname usage failed
|  To-Path: msrp://chat.example.org:7313/ansp71weztas;tcp
|  From-Path: msrp://rooms.example.com:8763/lkjh37s2s20w2a;tcp
|  -------a786hjs2
```

6.5. Invite Another User to a Room

If a SIP user wants to invite another user to join the conference he will send a REFER request indicating who needs to be invited in the Refer-To header, as per Section 5.5 of [RFC4579].

Example 42: SIP user invites another user (F59)

```
|  REFER sip:capulet@rooms.example.com SIP/2.0
|  To: <sip:capulet@rooms.example.com>
|  From: "Romeo" <sip:romeo@example.org>;tag=5534562
|  Call-ID: AA11FE6F-8E13-42F3-BF35-AB509FAADA39
|  Contact: <sip:romeo@example.org>;gr=c4pul3ts0rch4rd
|  Accept: message/sipfrag
|  Refer-To: <sip:benvolio@example.com>
|  Supported: replaces
|  Content-Length: 0
```

The SIP-to-XMPP gateway then acknowledges the SIP REFER request with a 200 OK response (step F58).

The gateway will then send a NOTIFY request as per [RFC3515] indicating that the invitation is in progress. Since there is no way to know the progress of the invitation until the user has joined, implementations are advised to terminate the REFER dialog

subscription upon receiving the first NOTIFY, with a status code of 100 Trying.

Example 43: Progress notification for invitation (F61)

```
|  NOTIFY sip:romeo@example.org SIP/2.0
|  To: <sip:romeo@example.org>;tag=5534562
|  From: <sip:capulet@rooms.example.com>;tag=18747389
|  Call-ID: AA11FE6F-8E13-42F3-BF35-AB509FAADA39
|  Event: refer
|  Subscription-State: terminated;reason=noresource
|  Contact: <sip:capulet@rooms.example.com;transport=tcp>;isfocus
|  Content-Type: message/sipfrag;version=2.0
|  Content-Length: ...
|
|  SIP/2.0 100 Trying
```

6.6. Exit Room

If Romeo decides to exit the chat room, his client sends a SIP BYE to the <montague@chat.example.org> chat room.

Example 44: Romeo terminates session (F61)

```
|  BYE sip:capulet@rooms.example.com SIP/2.0
|  Max-Forwards: 70
|  From: "Romeo" <sip:romeo@example.org>;tag=786
|  To: <sip:capulet@rooms.example.com>;tag=534
|  Call-ID: 08CFDAA4-FAED-4E83-9317-253691908CD2
|  Cseq: 1 BYE
|  Content-Length: 0
```

Upon receiving the SIP BYE, the SIP-to-XMPP gateway translates it into a presence stanza of type "unavailable" (F62) and sends it to the XMPP MUC room service. Then the SIP-to-XMPP gateway responds with a 200 OK to the MSRP user (F64).

Example 45: Romeo exits chatroom (F62)

```
|  <presence from='romeo@example.org'
|            to='capulet@rooms.example.com/Romeo'
|            type='unavailable'/>
```

7. Handling of Nicknames and Display Names

Fundamental rules for mapping addresses between XMPP and SIP are provided in [RFC7247]. However, chatrooms include a more specialized, unique identifier for each participant in a room, called

a nickname. Implementations are strongly encouraged to apply the rules for preparation and comparison of nicknames specified in [I-D.ietf-precis-nickname].

In addition to nicknames, some groupchat implementations also include display names (which might or might not be different from users' nicknames). A display name need not be unique within the context of a room but instead simply provides a user-friendly name for a participant.

In the SIP conference event package, the nickname is the value of the XCON 'nickname' attribute of the <user/> element [RFC6501] and the display name is the XML character data of the conference-info <display-text/> element [RFC4575]. In XMPP, the nickname is the value of the resourcepart of the occupant JID [XEP-0045] and the display name is the XML character data of the <nick/> element [XEP-0172].

In practice, the <display-text/> element is treated as canonical in SIP implementations, and the <nick/> element is rarely used in XMPP implementations. Therefore, for display purposes SIP implementations ought to use the <display-text/> element if the XCON 'nickname' attribute is not present, and XMPP implementations ought to use the resourcepart of the occupant JID if the <nick/> element is not present.

If there is a conflict between the SIP nickname and the XMPP nickname, the SIP-to-XMPP or XMPP-to-SIP gateway is responsible for adjusting the nickname to avoid the conflict and for informing the SIP or XMPP client of the unique nickname used to join the chatroom.

8. IANA Considerations

This document requests no actions of the IANA.

9. Security Considerations

The security considerations of [RFC3261], [RFC4975], [RFC6120], [RFC7247], [I-D.ietf-simple-chat], and [XEP-0045] apply.

10. References

10.1. Normative References

[I-D.ietf-precis-nickname]
Saint-Andre, P., "Preparation and Comparison of Nicknames", draft-ietf-precis-nickname-12 (work in progress), November 2014.

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

[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.

[RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", RFC 4579, August 2006.

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

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, March 2011.

[RFC6121] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 6121, March 2011.

[RFC7247] Saint-Andre, P., Houri, A., and J. Hildebrand, "Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): Architecture, Addresses, and Error Handling", RFC 7247, May 2014.

[XEP-0045]
Saint-Andre, P., "Multi-User Chat", XSF XEP 0045, July 2008.

10.2. Informative References

[I-D.sparks-sipcore-refer-clarifications]
Sparks, R. and A. Roach, "Clarifications for the use of REFER with RFC6665", draft-sparks-sipcore-refer-clarifications-05 (work in progress), October 2014.

[I-D.sparks-sipcore-refer-explicit-subscription]
Sparks, R., "Explicit Subscriptions for the REFER Method", draft-sparks-sipcore-refer-explicit-subscription-02 (work in progress), October 2014.

[RFC3515] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003.

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

[RFC6501] Novo, O., Camarillo, G., Morgan, D., and J. Urpalainen, "Conference Information Data Model for Centralized Conferencing (XCON)", RFC 6501, March 2012.

[RFC6502] Camarillo, G., Srinivasan, S., Even, R., and J. Urpalainen, "Conference Event Package Data Format Extension for Centralized Conferencing (XCON)", RFC 6502, March 2012.

[XEP-0172]
Saint-Andre, P. and V. Mercier, "User Nickname", XSF XEP 0172, March 2012.

[XEP-0249]
Saint-Andre, P., "Direct MUC Invitations", XSF XEP 0249, September 2011.

Appendix A. Acknowledgements

Special thanks to Fabio Forno for coauthoring an early version of this document, and to Ben Campbell for his detailed and insightful reviews.

Thanks also to Dave Crocker, Philipp Hancke, Olle Johansson, Paul Kyzivat, and Matt Ryan for their feedback.

The authors gratefully acknowledge the assistance of Markus Isomaki and Yana Stamcheva as the working group chairs and Gonzalo Camarillo and Alissa Cooper as the sponsoring Area Directors.

Some text in this document was borrowed from [RFC7247] and from [XEP-0045].

Peter Saint-Andre wishes to acknowledge Cisco Systems, Inc., for employing him during his work on earlier versions of this document.

Authors' Addresses

Peter Saint-Andre
&yet

Email: peter@andyet.com
URI: https://andyet.com/

Saul Ibarra Corretge
AG Projects
Dr. Leijdsstraat 92
Haarlem 2021RK
The Netherlands

Email: saul@ag-projects.com

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: Salvatore.Loreto@ericsson.com