Internet Engineering Task Force (IETF)
Request for Comments: 7573
Category: Standards Track
ISSN: 2070-1721

P. Saint-Andre
&yet
S. Loreto
Ericsson

June 2015

Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): One-to-One Text Chat Sessions

Abstract

This document defines a bidirectional protocol mapping for the exchange of instant messages in the context of a one-to-one chat session between a user of the Session Initiation Protocol (SIP) and a user of the Extensible Messaging and Presence Protocol (XMPP). Specifically for SIP text chat, this document specifies a mapping to the Message Session Relay Protocol (MSRP).

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7573.

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 one-to-one 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 one-to-one text chat sessions (sometimes called "session-mode" messaging); in particular, this document specifies mappings between XMPP messages of type "chat" and the Message Session Relay Protocol (MSRP) [RFC4975], which is commonly used in SIP-based systems for chat functionality (although note that MSRP is not conjoined to SIP, and can be used by non-SIP technologies). Mappings for single instant messages and groupchat are provided in [RFC7572] and [GROUPCHAT].

The approach taken here is to directly map syntax and semantics from one protocol to another. The mapping described herein depends on the protocols defined in the following specifications:

- XMPP chat sessions using message stanzas of type "chat" are specified in [RFC6121].

- MSRP chat sessions using the SIP INVITE and SEND request types are specified in [RFC4975].

In SIP-based systems that use MSRP, a chat session is formally negotiated (just as any other session type is negotiated when using SIP). By contrast, a one-to-one chat "session" in XMPP is an informal construct and is not formally negotiated: a user simply sends a message of type "chat" to a contact, the contact then replies to the message, and the sum total of such messages exchanged during a defined period of time is considered to be a chat session (ideally tied together using an XMPP <thread/> element as described in Section 5.1 of [RFC6121]). To overcome the disparity between these approaches, a gateway that wishes to map between SIP/MSRP and XMPP for one-to-one chat sessions needs to maintain some additional state, as described below.

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 who 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 chat-related specifications:

- "The Message Session Relay Protocol (MSRP)" [RFC4975]
- "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence" [RFC6121]
- "Indication of Message Composition for Instant Messaging" [RFC3994]
- "Chat State Notifications" [XEP-0085]

Note well that not all protocol-compliant messages are shown (such as SIP 100 TRYING messages), in order to focus the reader on the essential aspects of the protocol flows.

3. Terminology

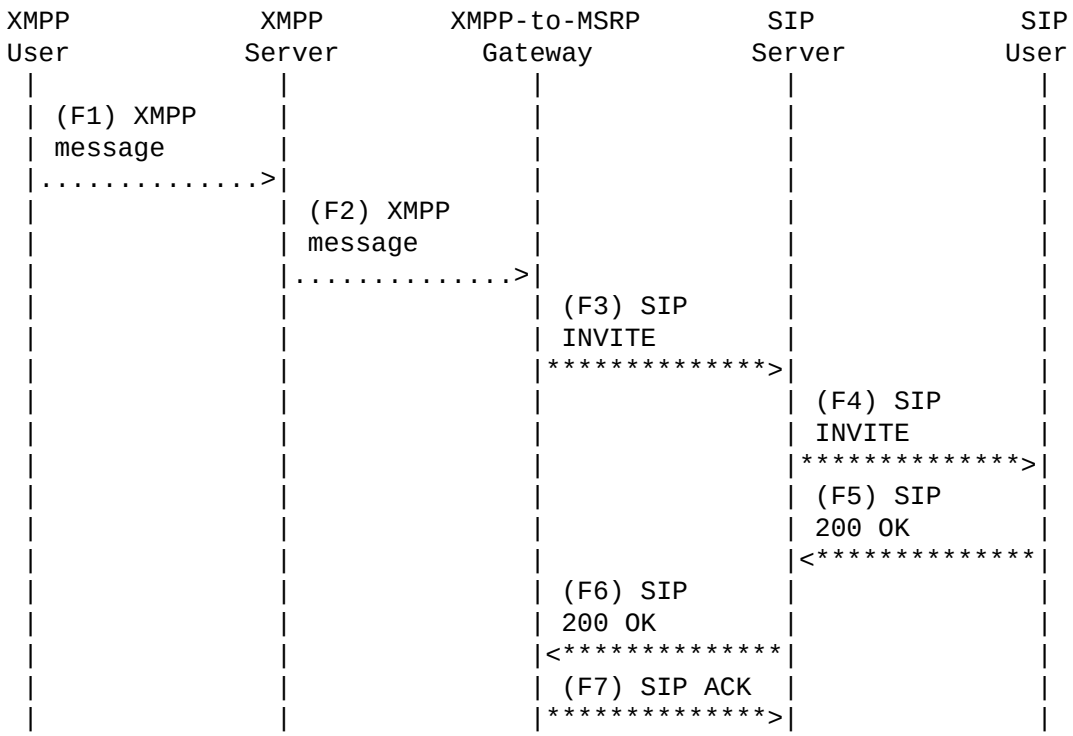
A number of terms used here are explained in [RFC3261], [RFC4975], [RFC6120], and [RFC6121].

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

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. XMPP to MSRP

In XMPP, the "informal session" approach is to simply send someone a <message/> of type "chat" without starting any session negotiation ahead of time (as described in [RFC6121]). The XMPP "informal session" approach maps very well into a SIP MESSAGE request, as described in [RFC7572]. However, the XMPP informal session approach can also be mapped to MSRP if the XMPP-to-SIP gateway maintains additional state. The order of events is as follows.



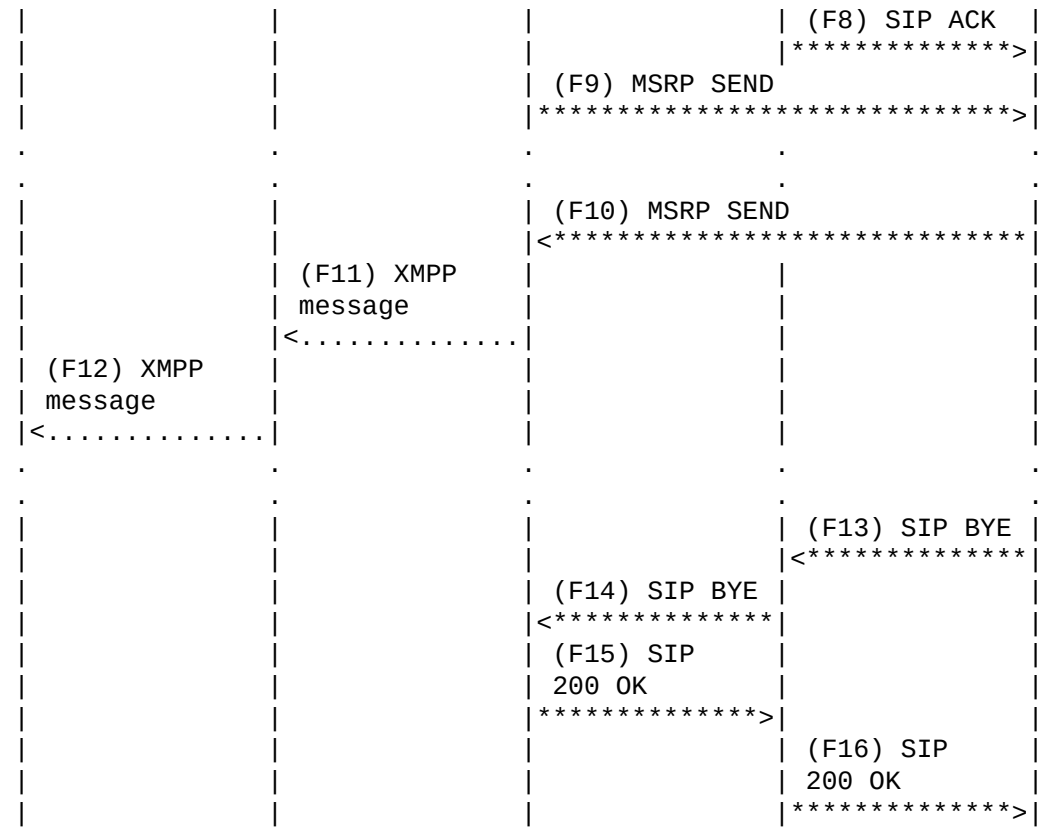


Figure 1: XMPP to MSRP Order of Events

The mapping of XMPP syntax to SIP syntax MUST be as specified in [RFC7572].

First, the XMPP user would generate an XMPP chat message.

Example 1: Juliet Sends XMPP Message (F1)

```
| <message from='juliet@example.com/yn0cl4bnw0yr3vym'
|          to='romeo@example.net'
|          id='a786hjs2'
|          type='chat'>
|   <thread>29377446-0CBB-4296-8958-590D79094C50</thread>
|   <body>Art thou not Romeo, and a Montague?</body>
| </message>
```

Upon receiving such a message stanza, the XMPP server needs to determine the identity of the domainpart in the 'to' address, which it does by following the procedures explained in Section 5 of [RFC7247]. If the domain is a SIP domain, the XMPP server will hand

off the message stanza to an XMPP-to-SIP gateway or connection manager that natively communicates with MSRP-aware SIP servers.

The XMPP-to-SIP gateway at the XMPP server would then initiate an MSRP session with Romeo on Juliet's behalf (since there is no reliable way for the gateway to determine whether Romeo's client supports MSRP, if that is not the case then MSRP session initiation might result in an error).

Example 2: Gateway Starts SIP Session on Behalf of Juliet (F3)

```
| INVITE sip:romeo@example.net SIP/2.0
| To: <sip:romeo@example.net>
| From: <sip:juliet@example.com>
| Contact: <sip:juliet@example.com>;gr=yn0cl4bnw0yr3vym
| Subject: Open chat with Juliet?
| Call-ID: 29377446-0CBB-4296-8958-590D79094C50
| CSeq: 1 INVITE
| Content-Type: application/sdp
|
| c=IN IP4 x2s.example.com
| m=message 7654 TCP/MSRP *
| a=accept-types:text/plain
| a=path:msrp://x2s.example.com:7654/jshA7weztas;tcp
```

Here we assume that Romeo's client supports MSRP and that Romeo accepts the MSRP session request.

Example 3: Romeo Accepts Session Request (F5)

```
| SIP/2.0 200 OK
| From: <sip:juliet@example.com>
| To: <sip:romeo@example.net>
| Contact: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| Call-ID: 29377446-0CBB-4296-8958-590D79094C50
| CSeq: 1 INVITE
| Content-Type: application/sdp
|
| c=IN IP4 s2x.example.net
| m=message 12763 TCP/MSRP *
| a=accept-types:text/plain
| a=path:msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
```

The XMPP-to-SIP gateway then acknowledges the session acceptance on behalf of Juliet.

Example 4: Gateway Sends ACK to Romeo (F7)

```
| ACK sip:juliet@example.com SIP/2.0
| To: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| From: <sip:juliet@example.com>
| Contact: <sip:juliet@example.com>;gr=yn0cl4bnw0yr3vym
| Call-ID: 29377446-0CBB-4296-8958-590D79094C50
| CSeq: 2 ACK
```

The XMPP-to-SIP gateway then transforms the original XMPP chat message into MSRP.

Example 5: Gateway Maps XMPP Message to MSRP (F9)

```
| MSRP a786hjs2 SEND
| From-Path: msrp://x2s.example.com:7654/jshA7weztas;tcp
| To-Path: msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
| Message-ID: 54C6F4F1-A39C-47D6-8718-FA65B3D0414A
| Byte-Range: 1-25/25
| Content-Type: text/plain
|
| Art thou not Romeo, and a Montague?
| -------a786hjs2$
```

Romeo can then send a reply using his MSRP client.

Example 6: Romeo Sends Reply (F10)

```
| MSRP di2fs53v SEND
| To-Path: msrp://x2s.example.com:7654/jshA7weztas;tcp
| From-Path: msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
| Message-ID: 6480C096-937A-46E7-BF9D-1353706B60AA
| Byte-Range: 1-25/25
| Failure-Report: no
| Content-Type: text/plain
|
| Neither, fair saint, if either thee dislike.
| -------di2fs53v$
```

The SIP-to-XMPP gateway would then transform that message into appropriate XMPP syntax for routing to the intended recipient.

Example 7: Gateway Maps MSRP Message to XMPP (F11)

```
| <message from='romeo@example.net/dr4hcr0st3lup4c'
|          to='juliet@example.com/yn0cl4bnw0yr3vym'
|          id='di2fs53v'
|          type='chat'>
|   <thread>29377446-0CBB-4296-8958-590D79094C50</thread>
|   <body>Neither, fair saint, if either thee dislike.</body>
| </message>
```

When the MSRP user wishes to end the chat session, the user's MSRP client sends a SIP BYE.

Example 8: Romeo Terminates Chat Session (F13)

```
| BYE juliet@example.com sip: SIP/2.0
| From: <sip:juliet@example.com>;tag=786
| To: <sip:romeo@example.net>;tag=087js
| Call-ID: 29377446-0CBB-4296-8958-590D79094C50
| CSeq: 3 BYE
| Content-Length: 0
```

The BYE is then acknowledged by the XMPP-to-SIP gateway.

Example 9: Gateway Acknowledges Termination (F15)

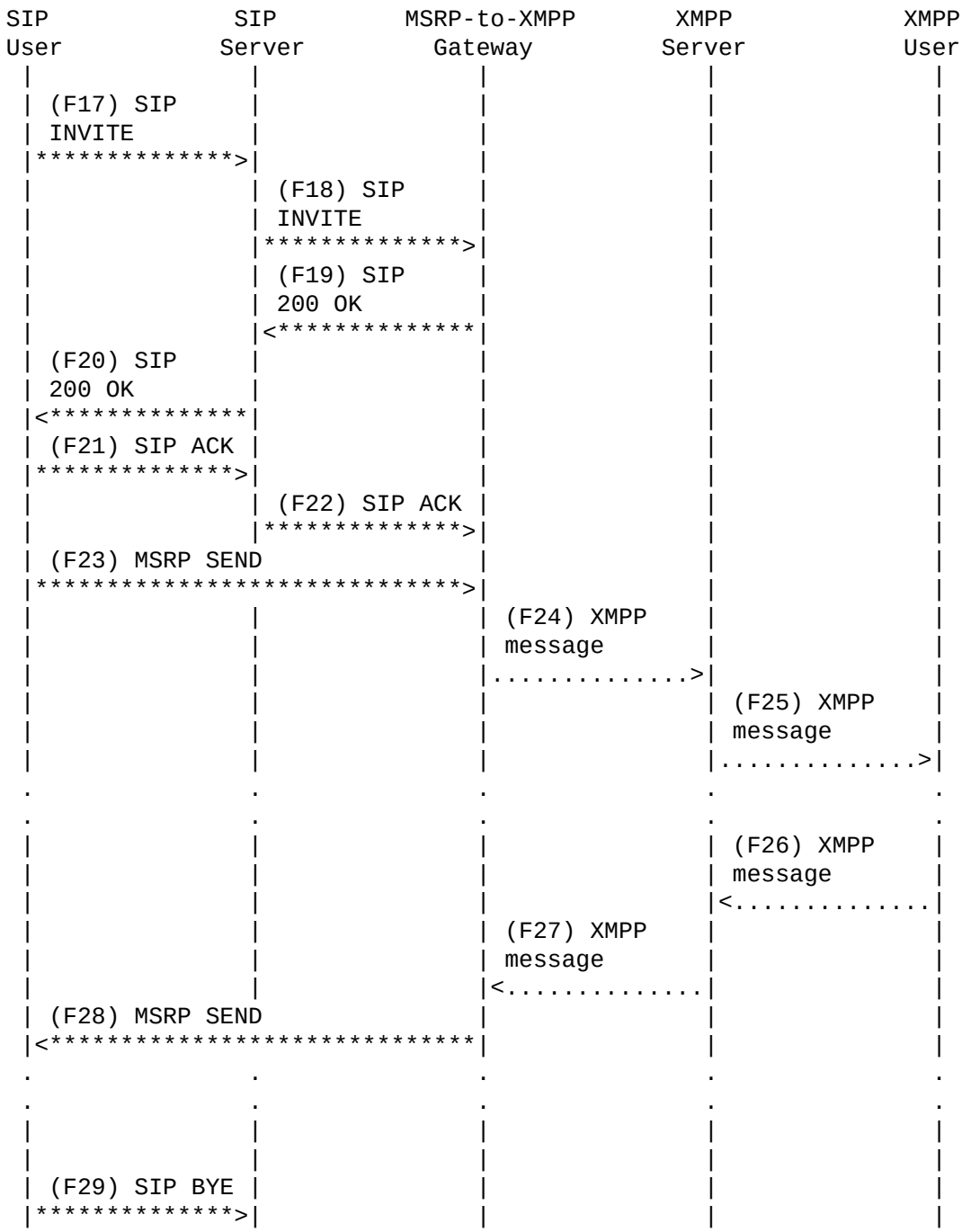
```
| SIP/2.0 200 OK
| From: <sip:juliet@example.com>;tag=786
| To: <sip:romeo@example.net>;tag=087js
| Call-ID: 29377446-0CBB-4296-8958-590D79094C50
| CSeq: 3 BYE
| Content-Length: 0
```

Because there is no formal session on the XMPP side, there is no corresponding communication from the gateway to the XMPP user. However, it is reasonable for the gateway to send a "gone" chat state notification [XEP-0085], as described under Section 6.1.

In addition, there is no explicit method defined in [RFC6121] for an XMPP user to formally terminate a chat session, so a gateway would need to listen for a "gone" chat state notification from the XMPP user or institute a timer that considers the XMPP informal chat session to be ended after some amount of time has elapsed ([XEP-0085] suggests generating a "gone" chat state if a user has not interacted with the chat session interface, system, or device for a relatively long period of time, e.g., 10 minutes).

5. MSRP to XMPP

When an MSRP client sends messages through a gateway to an XMPP client, the order of events is as follows.



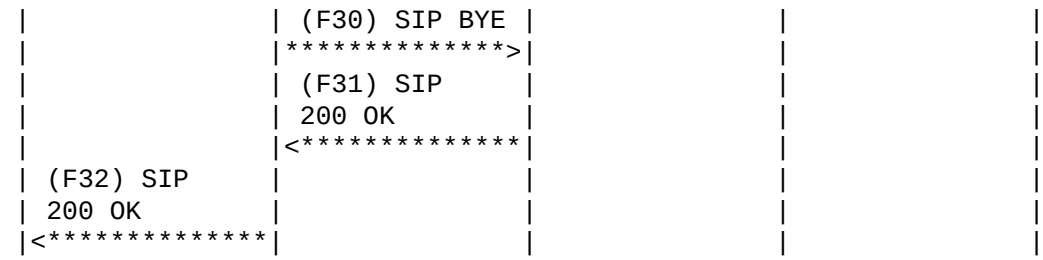


Figure 2: MSRP to XMPP Order of Events

The mapping of SIP syntax to XMPP syntax MUST be as specified in [RFC7572].

The protocol flow begins when Romeo starts a chat session with Juliet.

Example 10: Romeo Starts Chat Session (F17)

```
| INVITE sip:juliet@example.com SIP/2.0
| From: <sip:romeo@example.net>
| To: <sip:juliet@example.com>
| Contact: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| Subject: Open chat with Romeo?
| Call-ID: F6989A8C-DE8A-4E21-8E07-F0898304796F
| CSeq: 1 INVITE
| Content-Type: application/sdp
|
| c=IN IP4 s2x.example.net
| m=message 7313 TCP/MSRP *
| a=accept-types:text/plain
| a=path:msrp://s2x.example.net:7313/ansp71weztas;tcp
```

Upon receiving the INVITE, the SIP (MSRP) server needs to determine the identity of the domain portion of the Request-URI or To header, which it does by following the procedures explained in Section 5 of [RFC7247]. If the domain is an XMPP domain, the SIP server will hand off the INVITE to an associated MSRP-to-XMPP gateway or connection manager that natively communicates with XMPP servers.

Example 11: Gateway Accepts Session on Juliet's Behalf (F19)

```
| SIP/2.0 200 OK
| From: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| To: <sip:juliet@example.com>
| Contact: <sip:juliet@example.com>;gr=yn0cl4bnw0yr3vym
| Call-ID: F6989A8C-DE8A-4E21-8E07-F0898304796F
| CSeq: 1 INVITE
| Content-Type: application/sdp
|
| c=IN IP4 x2s.example.com
| m=message 8763 TCP/MSRP *
| a=accept-types:text/plain
| a=path:msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
```

Example 12: Romeo Sends ACK (F21)

```
| ACK sip:juliet@example.com SIP/2.0
| To: <sip:juliet@example.com>;gr=yn0cl4bnw0yr3vym
| From: <sip:romeo@example.net>
| Contact: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| Call-ID: F6989A8C-DE8A-4E21-8E07-F0898304796F
| CSeq: 2 ACK
```

Example 13: Romeo Sends Message (F23)

```
| MSRP ad49kswow SEND
| To-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
| From-Path: msrp://s2x.example.net:7313/ansp71weztas;tcp
| Message-ID: 676FDB92-7852-443A-8005-2A1B9FE44F4E
| Byte-Range: 1-32/32
| Failure-Report: no
| Content-Type: text/plain
|
| I take thee at thy word ...
| -------ad49kswow$
```

Example 14: MSRP-to-XMPP Gateway Maps MSRP Message to XMPP (F24)

```
| <message from='romeo@example.net'
|          to='juliet@example.com'
|          id='ad49kswow'
|          type='chat'>
|   <thread>F6989A8C-DE8A-4E21-8E07-F0898304796F</thread>
|   <body>I take thee at thy word ...</body>
| </message>
```

Example 15: Juliet Sends Reply (F26)

```
| <message from='juliet@example.com'
|          to='romeo@example.net'
|          id='ms53b7z9'
|          type='chat'>
|   <thread>29377446-0CBB-4296-8958-590D79094C50</thread>
|   <body>What man art thou ...?</body>
| </message>
```

Example 16: Gateway Maps XMPP Message to MSRP (F28)

```
| MSRP ms53b7z9 SEND
| To-Path: msrp://s2x.example.net:7313/jshA7weztas;tcp
| From-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
| Message-ID: 17EBA17B-94C0-463B-AD84-DE405C4C9D41
| Byte-Range: 1-25/25
| Failure-Report: no
| Content-Type: text/plain
|
| What man art thou ...?
| -------ms53b7z9$
```

Example 17: Romeo Terminates Chat Session (F29)

```
| BYE juliet@example.com sip: SIP/2.0
| To: <sip:juliet@example.com>;gr=yn0cl4bnw0yr3vym
| From: <sip:romeo@example.net>
| Contact: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| Call-ID: F6989A8C-DE8A-4E21-8E07-F0898304796F
| CSeq: 3 BYE
| Content-Length: 0
```

Example 18: Gateway Acknowledges Termination of Session on Behalf of Juliet (F31)

```
| SIP/2.0 200 OK
| To: <sip:juliet@example.com>;gr=yn0cl4bnw0yr3vym
| From: <sip:romeo@example.net>
| Contact: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| Call-ID: F6989A8C-DE8A-4E21-8E07-F0898304796F
| CSeq: 3 BYE
```

6. Composing Events

Both XMPP and MSRP enable a client to receive notifications when a person's conversation partner is composing an instant message within the context of a chat session.

For XMPP, the Chat State Notifications specification [XEP-0085] defines five states: active, inactive, gone, composing, and paused. Some of these states are related to the act of message composition (composing, paused), whereas others are related to the sender's involvement with the chat session (active, inactive, gone). Note that the "gone" chat state is not to be confused with the <gone/> stanza error condition defined in [RFC6120].

For MSRP (and, in general, for so-called SIP for Instant Messaging and Presence Leveraging Extensions (SIMPLE) systems), the Indication of Message Composition for Instant Messaging specification [RFC3994] defines two states: idle and active. Here the idle state indicates that the sender is not actively composing a message, and the active state indicates that the sender is indeed actively composing a message (the sending client simply toggles between the two states).

Because XEP-0085 states can represent information that is not captured in RFC 3994, gateways can either (a) map only the composing-related states or (b) map all the XEP-0085 states.

The following mappings are suggested.

Table 3: Mapping of SIP/SIMPLE isComposing Events to XMPP Chat states

isComposing Event	Chat State
active	composing
idle	active

Table 4: Mapping of XMPP Chat States to SIP/SIMPLE isComposing Events

Chat State	isComposing Event
active	idle
inactive	idle
gone	none (Section 6.1)
composing	active
paused	idle

The XMPP Chat State Notifications specification [XEP-0085] allows the sending of "standalone notifications" outside the context of a message, theoretically even before any messages are exchanged; although a gateway could thus send an <active/> notification to the XMPP user when the SIP user accepts or initiates a chat session (i.e., after F6 in Section 4 or after F22 in Section 5), this usage might be unexpected by XMPP clients as a way to signal the beginning of an informal chat session.

6.1. Use of the Gone Chat State

Although there is no direct mapping for the "gone" chat state to an isComposing event, receipt of the "gone" state at an XMPP-to-MSRP gateway can serve as a trigger for terminating the formal chat session within MSRP, i.e., for sending a SIP BYE for the session from the XMPP-to-MSRP gateway to the SIP user. The following examples illustrate this indirect mapping (which would arise if, for example, the XMPP user were to send a "gone" chat state notification after step F12 in Figure 1 or step F28 in Figure 2; in either of these cases, the session would be terminated by the XMPP user instead of by the SIP user, as currently shown in Figures 1 and 2).

Example 19: Juliet Sends Gone Chat State

```
| <message from='juliet@example.com'
|          id='nx62f197'
|          to='romeo@example.net'
|          type='chat'>
|   <thread>29377446-0CBB-4296-8958-590D79094C50</thread>
|   <gone xmlns='http://jabber.org/protocol/chatstates'/>
| </message>
```

Example 20: XMPP-to-MSRP Gateway Maps Gone Chat State to SIP BYE

```
| BYE romeo@example.net sip: SIP/2.0
| From: <sip:juliet@example.com>;tag=786
| To: <sip:romeo@example.net>;tag=087js
| Call-ID: 29377446-0CBB-4296-8958-590D79094C50
| CSeq: 3 BYE
| Content-Length: 0
```

Similarly, receipt of a SIP BYE message at an MSRP-to-XMPP gateway can serve as a trigger for sending a "gone" chat state notification to the XMPP user. The following examples illustrate this indirect mapping (which would occur after step F14 in Figure 1 or step F30 in Figure 2).

Example 21: Romeo Terminates Chat Session

```
| BYE juliet@example.com sip: SIP/2.0
| To: <sip:juliet@example.com>;gr=yn0cl4bnw0yr3vym
| From: <sip:romeo@example.net>
| Contact: <sip:romeo@example.net>;gr=dr4hcr0st3lup4c
| Call-ID: F6989A8C-DE8A-4E21-8E07-F0898304796F
| CSeq: 3 BYE
| Content-Length: 0
```

Example 22: MSRP-to-XMPP Gateway Generates Gone Chat State

```
| <message from='romeo@example.net'
|          id='hs61v397'
|          to='juliet@example.com'
|          type='chat'>
|   <thread>F6989A8C-DE8A-4E21-8E07-F0898304796F</thread>
|   <gone xmlns='http://jabber.org/protocol/chatstates'/>
| </message>
```

To enable these uses, gateways that support chat state notifications MUST support the "gone" state (which is merely recommended, not required, by [XEP-0085]).

It is also reasonable for gateways to implement timers that automatically trigger a "gone" chat state if the XMPP user has not sent a message within the "session" for a given amount of time ([XEP-0085] suggests generating a "gone" chat state if a user has not interacted with the chat session interface, system, or device for a relatively long period of time, e.g., 10 minutes).

7. Delivery Reports

Both XMPP and MSRP enable a client to receive notifications when a message has been received by the intended recipient.

For XMPP, the Message Receipts specification [XEP-0184] defines a method and XML namespace for requesting and returning indications that a message has been received by a client controlled by the intended recipient.

For MSRP, a native reporting feature is included, in the form of REPORT chunks (see Sections 7.1.2 and 7.1.3 of [RFC4975]).

An XMPP Message Receipts element of <request xmlns='urn:xmpp:receipts'/> is to be mapped to an MSRP Success-Report header field with a value of "yes", and an XMPP Message Receipts

element of <received xmlns='urn:xmpp:receipts'/> is to be mapped to an MSRP REPORT request.

A Success-Report header field with a value of "yes" in an MSRP SEND request is to be mapped to an XMPP Message Receipts element of <request xmlns='urn:xmpp:receipts'/>, and an MSRP REPORT request is to be mapped to an XMPP message containing only a Message Receipts element of <received xmlns='urn:xmpp:receipts'/>.

Because the XMPP Message Receipts specification does not support failure reports, there is no mapping for the MSRP Failure-Report header field and gateways SHOULD set that header field to "no".

Examples follow.

First, the XMPP user sends a message containing a request for delivery notification.

Example 23: Juliet Sends XMPP Message with Receipt Request

```
| <message from='juliet@example.com'
|          id='bf9m36d5'
|          to='romeo@example.net'
|          type='chat'>
|   <thread>29377446-0CBB-4296-8958-590D79094C50</thread>
|   <body>What man art thou ...?</body>
|   <request xmlns='urn:xmpp:receipts'/>
| </message>
```

Example 24: Gateway Maps XMPP Message to MSRP

```
| MSRP bf9m36d5 SEND
| To-Path: msrp://s2x.example.net:7313/jshA7weztas;tcp
| From-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
| Message-ID: 6187CF9B-317A-41DA-BB6A-5E48A9C794EF
| Byte-Range: 1-25/25
| Success-Report: yes
| Failure-Report: no
| Content-Type: text/plain
|
| What man art thou ...?
| -------bf9m36d5$
```

Next, the recipient returns a report.

Example 25: Romeo Returns MSRP Receipt

```
| MSRP hx74g336 REPORT
| To-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
| From-Path: msrp://s2x.example.net:7313/jshA7weztas;tcp
| Message-ID: 6187CF9B-317A-41DA-BB6A-5E48A9C794EF
| Byte-Range: 1-106/106
| Status: 000 200 OK
| -------hx74g336$
```

Example 26: MSRP-to-XMPP Gateway Maps Receipt to XMPP

```
| <message from='romeo@example.net'
|          id='hx74g336'
|          to='juliet@example.com'>
|   <received xmlns='urn:xmpp:receipts' id='87652491'/>
| </message>
```

8. Message Size

Unlike page-mode messaging [RFC3428] (which specifies that the size of a MESSAGE request is not allowed to exceed 1300 bytes), session-mode messaging [RFC4975] can be used to send larger messages. MSRP includes a chunking mechanism such that larger messages can be broken up into multiple MSRP SEND requests. Because the MSRP gateway at an XMPP service acts as an MSRP endpoint, it is responsible for receiving chunked messages and reconstructing them into a single message for delivery toward the XMPP recipient. (Naturally, implementations need to be careful about accepting very large messages; see Section 14.5 of [RFC4975].)

Although there is no hard limit on the size of an XMPP stanza, in practice, most XMPP services (at least on the public Internet) are configured with a maximum stanza size in order to help prevent denial-of-service attacks. As specified in Section 13.12 of [RFC6120], this maximum is not allowed to be less than 10,000 bytes.

The administrators of an XMPP service need to ensure that the associated MSRP gateway is configured with the same or smaller maximum MSRP message size as the maximum XMPP stanza size; this enables the gateway to return an appropriate value for the Session Description Protocol (SDP) "max-size" attribute (see Section 8.6 of [RFC4975]) and to properly handle incoming messages larger than the configured limits.

If an MSRP-to-XMPP gateway implementation receives an MSRP message that exceeds its configured limit as just described, it MUST return an MSRP 413 error (e.g., in response to the first SEND request whose Byte-Range header field indicates a byte total exceeding the limit).

9. Internationalization Considerations

Relevant discussion of internationalized text in messages can be found in [RFC7572].

10. Security Considerations

Detailed security considerations are given in the following documents:

- For instant messaging protocols in general, see [RFC2779]
- For MSRP chat, see [RFC4975]; for when SIP is used to negotiate MSRP sessions, see [RFC3261]
- For XMPP-based instant messaging, see [RFC6121] and also [RFC6120]
- For SIP-XMPP interworking in general, see [RFC7247]
- For end-to-end encryption of instant messages, see [RFC7572]

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC3994] Schulzrinne, H., "Indication of Message Composition for Instant Messaging", RFC 3994, DOI 10.17487/RFC3994, January 2005, <http://www.rfc-editor.org/info/rfc3994>.

[RFC4975] Campbell, B., Ed., Mahy, R., Ed., and C. Jennings, Ed., "The Message Session Relay Protocol (MSRP)", RFC 4975, DOI 10.17487/RFC4975, September 2007, <http://www.rfc-editor.org/info/rfc4975>.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <http://www.rfc-editor.org/info/rfc6120>.

[RFC6121] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 6121, DOI 10.17487/RFC6121, March 2011, <http://www.rfc-editor.org/info/rfc6121>.

[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, DOI 10.17487/RFC7247, May 2014, <http://www.rfc-editor.org/info/rfc7247>.

[RFC7572] Saint-Andre, P., Houri, A., and J. Hildebrand, "Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): Instant Messaging", RFC 7572, DOI 10.17487/RFC7572, June 2015, <http://www.rfc-editor.org/info/rfc7572>.

[XEP-0085] Saint-Andre, P. and D. Smith, "Chat State Notifications", XSF XEP 0085, September 2009.

[XEP-0184] Saint-Andre, P. and J. Hildebrand, "Message Delivery Receipts", XSF XEP 0184, March 2011.

11.2. Informative References

[GROUPCHAT] Saint-Andre, P., Corretge, S., and S. Loreto, "Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): Groupchat", Work in Progress, draft-ietf-stox-groupchat-11, March 2015.

[RFC2779] Day, M., Aggarwal, S., Mohr, G., and J. Vincent, "Instant Messaging / Presence Protocol Requirements", RFC 2779, DOI 10.17487/RFC2779, February 2000, <http://www.rfc-editor.org/info/rfc2779>.

[RFC3428] Campbell, B., Ed., Rosenberg, J., Schulzrinne, H., Huitema, C., and D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging", RFC 3428, DOI 10.17487/RFC3428, December 2002, <http://www.rfc-editor.org/info/rfc3428>.

Acknowledgements

Special thanks to Eddy Gavita and Nazin Hossain for coauthoring an early draft version of this document.

Thanks to Mary Barnes, Ben Campbell, Dave Crocker, Adrian Georgescu, Philipp Hancke, Saul Ibarra Corretge, Tory Patnoe, and Matt Ryan for their feedback.

Stephen Farrell, Brian Haberman, Joel Jaeggli, Barry Leiba, Kathleen Moriarty, and Pete Resnick provided helpful input during IESG review.

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.

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

Authors' Addresses

Peter Saint-Andre
&yet

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

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: Salvatore.Loreto@ericsson.com