Message Session Relay Protocol (MSRP)
Architecture, Protocole et Services

EFORT
http://www.efort.com

RTP est un protocole du plan usager permettant le transport de flux tels que audio et vidéo dans le contexte d 'une session multimédia.
Pour échanger des médias tels que des messages ou des fichiers, il faut un protocole du plan usager qui soit adapté. Le protocole choisi dans le contexte IMS est MSRP (Message Session Relay Protocol).
MSRP est un protocole fonctionnant en mode connecté (transport TCP ou SCTP) pour échanger des messages (ou/et des fichiers) dans le contexte d 'une session. MRSP ne place aucune restriction sur la taille de message ou sur le type de contenu.
Il existe deux recommandations principales concernant MSRP : RFC 4975 est la spécification de base, alors que le RFC 4976 spécifie des extensions relai, qui facilitent la traversée de réseaux multiples (en particulier lorsque des firewalls sont présents sur le chemin du message).
MSRP ne peut pas établir lui même une session. Il s'agit d'un protocole plan usager uniquement. SIP est utilisé pour établir et libérer la session multimédia. MSRP ne peut être à ce jour utilisé qu 'avec SIP.

1 Session de messagerie MSRP

La figure 1 décrit l 'établissement et la libération d 'une session MSRP. Lorsque la session est établie, chaque usager envoie une requête MSRP SEND pour transmettre un message à un autre usager.

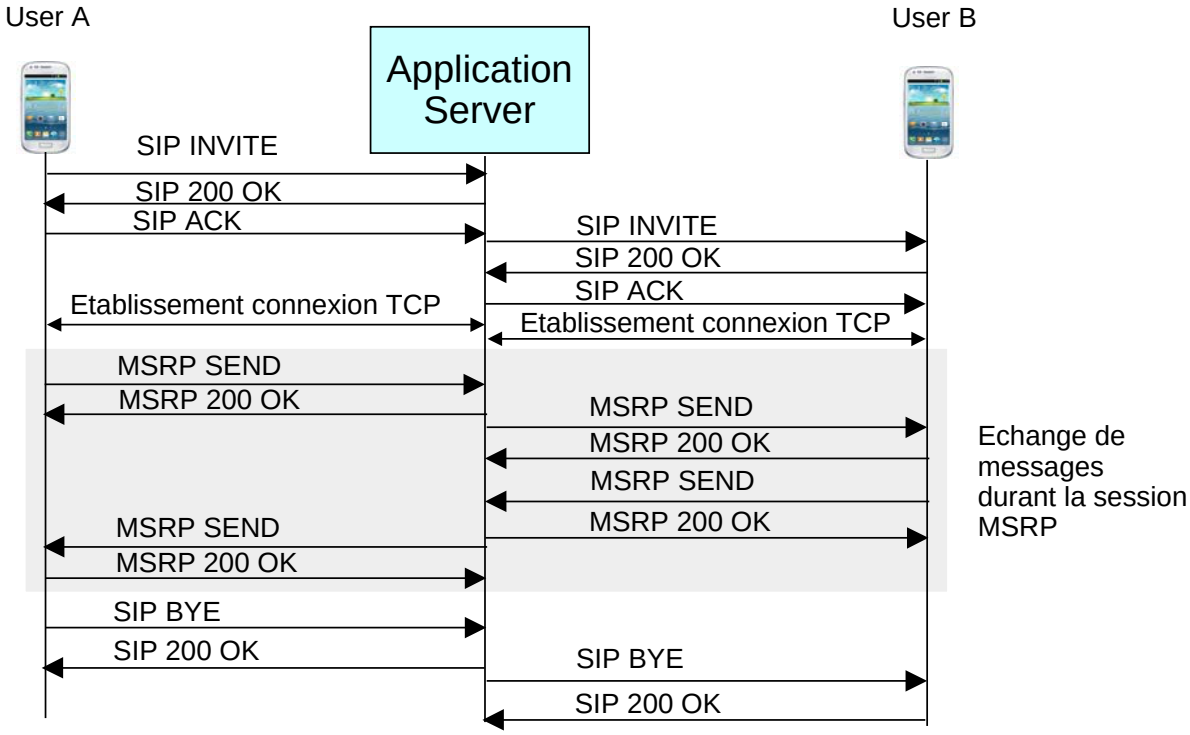

Figure 1 : Session de messagerie avec MSRP

La figure 1 décrit le trafic SIP et MSRP échangé avec l ’AS de messagerie. Cet AS est de type B2BUA (Back To Back User Agent).
Même si l ’AS est présent sur le plan de contrôle avec SIP, il n ’est pas forcément nécessaire que cet AS soit présent sur le plan usager avec MSRP. Toutefois la configuration montrée avec un AS impliqué aussi sur le plan usager est celle la plus commune. Cela permet entre autre à l ’AS de stocker tout l ’historique des messages échangés et d’assurer la taxation de cette session.
Pour des raisons de simplification les CSCF du plan de contrôle SIP/IMS ne sont pas présents

1.1 Schéma d’URI MSRP

A la différence de RTP, MSRP a son propre schéma d’URI.
SDP est utilisé pour décrire la session au niveau média avec MSRP. Alors que pour RTP sont échangés via SDP les numéros de port, les adresses IP et les codecs des participants, pour MSRP les mêmes principes s ’appliquent : des URIs MSRP sont échangés à la place de simples adresses IP. Exemple d ’URI MSRP :
msrp://5555::1:2:3:4:7900/kjhd37s2s20w2a;tcp
Le nom du schéma d ’URI (msrp dans ce cas) est toujours suivi par le caractère « : » . A la différences des URIs SIP, il n ’y a ni username ni caractère « @ ». Ce qui suit est soit une adresse IP, soit un hostname. 7900 est l ’identité de couche inférieure (numéro de port TCP). Ce qui suit est le caractère « / », suivi par une chaîne de caractère « kjhd37s2s20w2a » qui est appelée session-ID.
L’URI msrp : est utilisée pour le transport TCP. L’URI msrps : concerne un transport TLS sur TCP. L’URI smsrp : est pour le transport SCTP. Enfin l’URI smsrps : est associée à un transport TLS sur SCTP.

1.2 Etablissement de session MSRP et descriptions SDP associées

Considérons une session MSRP établie entre deux participants, à savoir Mary Taylor et John Cook telle que décrite à la figure 1.

1. Mary Taylor initie une session de messagerie via la requête SIP INVITE contenant la description SDP suivante :

m=message 7900 msrp/tcp *
a=accept-types:message/cpim text/plain text/html
a=path:msrp://[5555::1:2:3:4]:7900/kjhd37s2s20w2a;tcp
a=max-size:131072
a=curr: qos local none
a=curr: qos remote none
a=des: qos mandatory local sendrecv
a=des: qos mandatory remote sendrecv

La ligne m décrit e média. Une session de messagerie (session-based messaging) est définie par le mot clé « message » suivi par le numéro de port TCP choisi par le client pour émettre et recevoir des messages MSRP. Dans le futur, il devrait être possible de prendre en compte SCTP. C ’est pourquoi il faut préciser le transport. MSRP/TCP ou MSRP/SCTP. Comme la notion de codec n ’intervient pas pour des sessions de messagerie, la liste de codecs est remplacée par la caractère « * ».
La ligne a=path définit l ’URI MSRP du client comme décrit précédemment.

La ligne a= accept-types décrit les formats supportés par le client, e.g., text/plain, text/html message/cpim (RFC 3862), etc.
La ligne a = max-size décrit la taille maximum de message que le client est prêt à accepter indépendamment du fait que ce message sera fragmenté en chunks ou pas.
Les 4 lignes « a » associées à la QoS sont utilisées lorsque les préconditions sont mises en œuvre pour une QoS garantie pour la session de messagerie.

2. Mark Rich retourne une réponse SIP 200 OK contenant la description SDP suivante :

m=message 8400 msrp/tcp *
a=accept-types:message/cpim text/plain text/html
a=path:msrp://[5555::6:7:8:9]:8400/zt24wd44t5t34yb;tcp
a=max-size:131072
a=curr: qos local none
a=curr: qos remote none
a=des: qos mandatory local sendrecv
a=des: qos mandatory remote sendrecv

3. Mary Taylor acquitte la réponse SIP 200 OK par une requête SIP ACK.

4. Mary Taylor ouvre la connexion TCP avec Mark Rich

5. Mary Taylor envoie un message MSRP SEND à Mark Rich

6. Mark Rich acquitte le message MSRP SEND par une réponse MSRP 200 OK

1.3 Message MSRP et réponse associée

La requête SEND émise par Mary Taylor a le format suivant :

MSRP d93kswow SEND
To-Path: path:msrp://[5555::6:7:8:9]:8400/zt24wd44t5t34yb;tcp
From-Path: path:msrp://[5555::1:2:3:4]:7900/kjhd37s2s20w2a;tcp
Message-ID: 12339sdqwer
Byte-Range: 1-13/13
Content-Type: text/plain

Hi, I'm Mary!
-------d93kswow$

Pour envoyer une requête, l ’émetteur crée un identificateur de transaction et l ’utilise avec le nom de la méthode SEND MSRP. Cet identificateur de transaction doit être unique parmi toutes les requêtes en cours. Pour ce faire, il doit contenir au moins 64 bits aléatoires.
Puis, l ’émetteur renseigne les headers From-Path et To-Path. Il s ’agir des URIs MSRP des entités émettrice et réceptrice. Si plusieurs URIs sont présents dans le header To-Path, le premier correspond au premier nœud destinataire. Le dernier correspond au dernier nœud destinataire.
La requête a un corps. Elle doit indiquer le header Content-Type. Il s ’agit du dernier header du message. Le corps du message doit être séparé des headers du message par une ligne vide.
Une requête SEND ne contient pas forcément de corps, et dans ce cas, ne dispose par de header Content-Type. Un message SEND sans corps permet par exemple de maintenir les associations NATs actives.

En plus des headers From-Path, To-Path et Content-Type, il faut rajouter les headers Message-ID et Byte-Range.
Le Message-ID doit être unique et ne doit pas être utilisé par cette même instance dans le futur ni par aucune autre instance.
Si nécessaire le message peut être décomposé en segments (chunks). Il faut alors générer autant de requêtes SEND que de Chunks à envoyer. Le Message-ID fournit un identificateur unique qui fait référence au message initial à émettre indépendamment du nombre de chunks nécessaires pour l'émettre. Il existe alors une requête SEND par chunk du même message. Toutes ces requpetes SEND contiennent alors le même Message-ID.
Alors que l 'identificateur de transaction est unique par requête MSRP SEND, un même Message-ID se retrouve dans différentes requêtes SEND si elle contiennent des chunks d 'un même message que l 'émetteur souhaite envoyer et qui a été décomposé.
Le header Byte-Range contient une valeur de départ (premier numéro d 'octet du message initial présent dans cette requête SEND) suivie par le caractère "-", puis une valeur de fin (dernier numéro d 'octet du message initial présent dans cette requête SEND) suivie par le caractère "/", et finalement la longueur totale. Le premier octet du message a une position égale à 1 et non par 0.
Si un message a une longueur de 13 octets et n 'est pas segmenté, alors le header Byte-Range de la requête SEND contient tout le message a pour valeur 1-13/13.
La ligne de fin de requête SEND comme par 4 caractères "-", suivis par l'identificateur de transaction, suivi par le caractère "$".

La réponse 200 OK retournée par Mark Rich a le format suivant :

```
MSRP d93kswow 200 OK
    To-Path: path:msrp://[5555::6:7:8:9]:8400/zt24wd44t5t34yb;tcp
    From-Path: path:msrp://[5555::1:2:3:4]:7900/kjhd37s2s20w2a;tcp
    -------d93kswow$
```

Si la requête SEND contient un header Content-Type indiquant un type de média non supporté, et si la valeur de Failure-Report n'est pas "no", le récepteur doit générer une réponse avec un status code 415.

2 Segmentation de message avec MSRP

A la figure 2, le message MSRP de Mary Taylor est envoyé via deux segments, ou chunks (Cela a un sens si et seulement si le message est long, e.g., un message texte avec une pièce jointe volumineuse). Le nombre de chunks dépend de la taille du message sachant qu'il est suggéré que la taille maximum d 'un chunk soit de 2048 octets. Le logiciel MSRP de l 'usager A a la responsabilité de subdiviser le message sortant en chunks. Chaque requête MSRP SEND est acquittée par une réponse 200 OK. Comme pour SIP, 200 OK est une réponse positive pour MSRP. Le logiciel MSRP sur le terminal de l 'usager B a la responsabilité de réassembler les chunks en un message complet.
La fonction de segmentation est optionnelle, et aucune taille préférée de segment n 'est définie. Toutefois il existe deux bénéfices à utiliser le principe de segmentation :

- Lorsque des ASs de messagerie sont sur le chemin des messages MSRP, la segmentation peut améliorer la latence de bout en bout. Les ASs de messagerie peuvent commencer à relayer le premier segment aux participants de la session avant de recevoir le second segment.
- La segmentation permet un partage équitable de la connexion avec d 'autres applications. Si l 'on considère la figure 2, le terminal de l 'usager A dispose d 'une connectivité avec le réseau d 'un opérateur mobile. L 'application de messagerie dans cet exemple utilise cette connectivité pour communiquer avec un AS et au final avec

l ’usager B. D ’autres applications, par exemple d ’autres sessions de messagerie ou d ’autres applications IMS, peuvent chercher à utiliser la même connectivité simultanément. Une autre application peut être amenée à envoyer des paquets entre le premier segment et le second segment. La segmentation est particulièrement bénéfique dans le cas d ’applications sensibles au délai (e.g., Voix sur IP ou vidéotéléphonie sur IP) coexistant avec une application de messagerie qui émet des pièces jointes volumineuses.

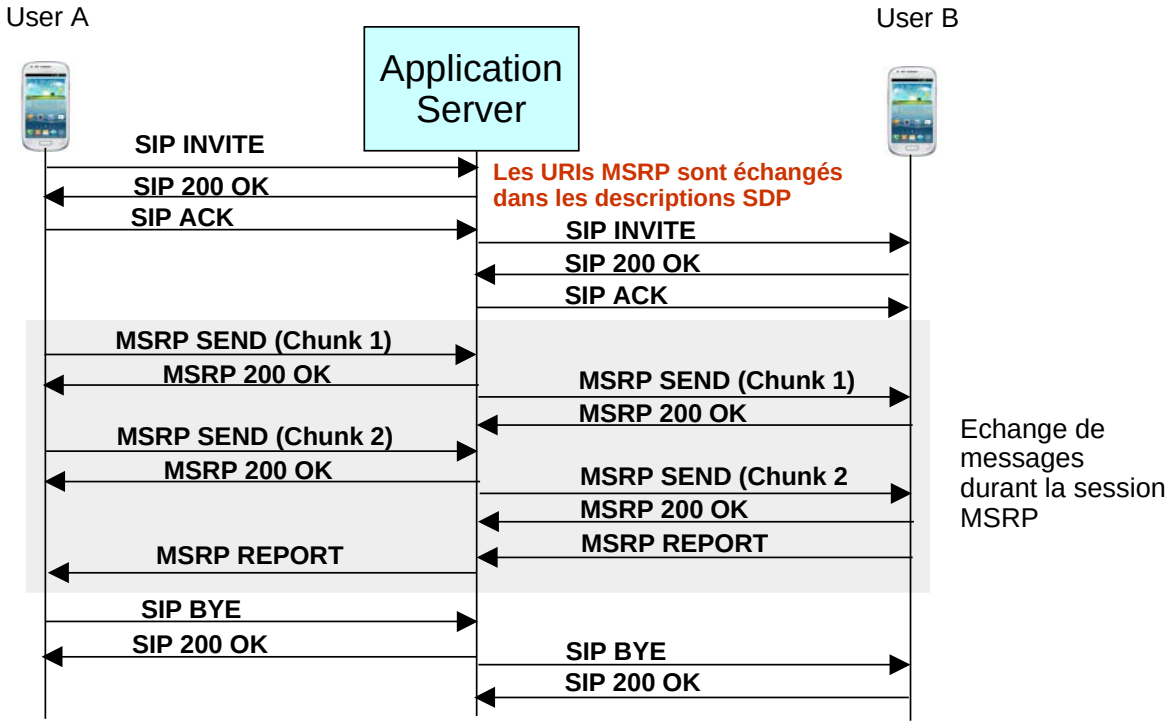

Figure 2 : Session MSRP avec fragmentation

Le premier segment MSRP a le format suivant.
MSRP d93kswow SEND
To-Path: path:msrp://[5555::6:7:8:9]:8400/zt24wd44t5t34yb;tcp
From-Path: path:msrp://[5555::1:2:3:4]:7900/kjhd37s2s20w2a;tcp
Message-ID: 21456sdqwer
Byte-Range: 1-137/148
Success-Report : yes
Content-Type: message/cpim

To: Mark Rich <Mark.Rich@orange.fr>
From: Mary Taylor <Mary.Taylor@orange.fr>
DateTime: 2013-01-15T15:02:31-03:00
Content-Type: text/plain

ABCD
-------d93kswow+

Le second segment a le format suivant :
MSRP op2nc9a SEND
To-Path: path:msrp://[5555::6:7:8:9]:8400/zt24wd44t5t34yb;tcp
From-Path: path:msrp://[5555::1:2:3:4]:7900/kjhd37s2s20w2a;tcp
Message-ID: 21456sdqwer
Byte-Range: 138-148/148

Success-Report : yes
Content-Type: message/cpim

1234567890
-------op2nc9a$

MSRP dkei38sd REPORT
To-Path: path:msrp://[5555::1:2:3:4]:7900/kjhd37s2s20w2a;tcp
From-Path: path:msrp://[5555::6:7:8:9]:8400/zt24wd44t5t34yb;tcp
Message-ID: 21456sdqwer
Byte-Range: 1-148/148
Status: 000 200 OK
-------dkei38sd$

La requête REPORT qui indique le succès de la réception du message global a le format suivant :

MSRP dkei38sd REPORT
To-Path: path:msrp://[5555::1:2:3:4]:7900/kjhd37s2s20w2a;tcp
From-Path: path:msrp://[5555::6:7:8:9]:8400/zt24wd44t5t34yb;tcp
Message-ID: 21456sdqwer
Byte-Range: 1-148/148
Status: 000 200 OK
-------dkei38sd$

3 Headers Success-Report et Failure-Report

3.1 Header Success-Report

Lorsqu ’un client reçoit un message sous forme de un ou plusieurs chunks, qui contiennent un header Success-Report positionné à la valeur « yes », il doit retourner une (des) requête (s) REPORT indiquant que tous les octets du message origine ont été reçus.
Le récepteur peut attendre jusqu ’à recevoir le dernier chunk du message, et retourner une seule requête REPORT pour le message global.
Il est aussi possible de générer des requêtes REPORT, une par chunk reçu, ou encore de générer des requêtes REPORT, périodiquement indiquant le nombre d ’octets reçus jusque là.
Le client ne doit pas émettre de requête REPORT de succès si le header Success-Report n ’est pas présent ou si sa valeur est « no ».
Le header Success-Report peut être positionné à "no" dans les systèmes publics afin de réduire la charge, mais peuvent être positionnés à "yes" dans des systèmes d ’entreprise, tels que les systèmes financiers.

3.2 Header Failure-Report

Si un client reçoit une requête SEND avec un header Failure-Report positionné à « no », alors il ne doit pas émettre de requête REPORT en cas d ’échec.
Si la valeur de Failure-Report est « partial », il ne doit pas retourner de réponse à la requête SEND mais doit immédiatement retourner un REPORT d ’échec si une erreur s ’est produite.
Si un client reçoit une requête SEND avec un header Failure-Report positionné à « yes » ou sans Failure-Report, il doit retourner une réponse correspondante, et si une erreur s ’est produite, il doit retourner une requête REPORT.

Si le header Failure-Report d ’une requête SEND reçue est positionné à « no », aucune réponse n ’est retournée. Cela signifie qu ’il n ’y a aucune détection d ’échec, mis à part ce qui pourrait être détecté par TCP. Ce type de comportement a un sens pour certaines applications telles que des services d ’urgence qui diffusent des messages d ’information, ou encore des applications qui échangent de larges volumes de données et pour lesquelles la fiabilité apportée par TCP est suffisante. Failure-Report positionné à « no » est aussi utile pour des systèmes qui émettent des messages d ’information sans importance ou qui sont émis de façon récurrente.
Failure-Report positionné à « yes » est utile pour un grand nombre de systèmes qui souhaitent notifier l ’usager si le message n ’a pas été délivré correctement.
Failure-Report et Success-Report peut être positionnés indépendamment l ’un de l ’autre. Il est possible de ne pas recevoir de réponse et de requêtes REPORT d ’échec (Failure-Report positionné à « no ») mais recevoir des requêtes REPORT de succés (notamment lorsqu ’un message est segmenté en chunks et que l ’on souhaite recevoir un acquittement de la réception de tous les chunks du message; dans ce cas Success-Report est positionné à (« yes »).

4 Relai MSRP

Le RFC 4976 introduit la notion de relai MSRP et spécifie la procédure d ’authentification qui permet au client de s ’authentifier au relai MSRP à la couche MSRP.
Il existe deux raisons d ’être du relai :

- permettre un nombre réduit de connexions inter-relai pour transporter les messages d ’un très grand nombre de sessions MSRP.
- Pour des sessions de messagerie impliquant plusieurs domaines administratifs, permettre à chaque domaine d ’administration impliqué d ’appliquer un policy control.

Références

RFC 4975, B. Campbell, The Message Session Relay Protocol (MSRP), September 2007.
EFC 4976, C. Jennings, Relay Extensions for the Message Session Relay Protocol (MSRP), September 2007.