RFC 6121 : Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 7 avril 2011

Date de publication du RFC : Mars 2011

http://www.bortzmeyer.org/6121.html

Le protocole XMPP, normalisé dans le RFC 6120[1], est souvent cité comme protocole de messagerie instantanée. Mais c'est en fait plus compliqué que cela : XMPP lui-même ne fournit qu'un service minimum de routage d'éléments XML, les vrais services devant être développés au dessus de XMPP. C'est ainsi que les services de messagerie instantanée et de présence sont normalisé dans ce RFC, le 6121. Il succède au premier RFC sur ces services, le RFC 3921.

Le protocole de base, décrit dans le RFC 6120, transmet des éléments XML, les **strophes** ("*stanzas*", j'ai choisi cette traduction parce qu'elle maintenait la métaphore musicale), en temps réel ou presque. Sur ce protocole de base, des tas de services sont possibles. La messagerie instantanée est le plus connu. Décrite dans le RFC 2779, elle a aujourd'hui éclipsé la plupart des autres usages de XMPP.

XMPP a été développé vers 1999 sous le nom de Jabber. Le nom de XMPP lui est venu lors de son adoption par l'IETF en 2002, les premiers RFC sont sortis en 2004 et ils viennent d'être mis à jour par la sortie des RFC 6120 (le protocole de base), RFC 6121 (la messagerie instantanée, qui remplace le RFC 3921) et RFC 7622 (le format des adresses). De son histoire, qui s'est déroulée partiellement en dehors de l'IETF, XMPP a gardé plusieurs extensions qui ne sont **pas** spécifiées dans un RFC, mais dans des documents connus sous le nom de **XEP** ("*XMPP Extension Protocols*") comme par exemple XEP-0160 <http://xmpp.org/extensions/xep-0160.html> pour le stockage des strophes si elles ne peuvent pas être délivrées immédiatement.

Qu'est-ce que la messagerie instantanée ? Vu de XMPP, ce sont une liste de connaissances (le **carnet** ou "*roster*") et la possibilité d'envoyer entre ces connaissances des messages relativement courts, de

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc6120.txt

manière quasi-instantanée. Contrairement au courrier électronique, dont la caractéristique principale est d'être asynchrone (on peut envoyer un message à quelqu'un qui est en vacances et qui le lira au retour), la messagerie instantanée (ou IM pour "*Instant Messaging*" ou "*chat*" ou clavardage si on veut parler comme l'Académie française) est **synchrone** : un message perd beaucoup de son intérêt s'il n'est pas lu tout de suite. Cela implique donc de gérer la **présence** : les correspondants ont besoin de savoir si le type en face est là ou pas. Comme cette information peut être considérée comme confidentielle, la présence n'est communiquée qu'à ceux qu'on a accepté comme correspondants ("*buddies*", ce qu'un utilisateur de Facebook traduirait sans doute par ami).

Donc, les quatre fonctions :

— Gérer le carnet,
— Échanger des messages avec ses correspondants,
— Informer de sa présence,
— Gérer la liste des gens qu'on informe de sa présence.

résument très bien le RFC 2779, qui fut le cahier des charges de XMPP.

La section 1.4 de notre RFC détaille les dites fonctions. Le transport des données, la connexion au serveur, l'authentification, etc, sont des choses très importantes mais elles sont absentes de ce RFC, puisqu'elles sont déjà couvertes dans le RFC 6120. Pour résumer ce dernier : un client XMPP se connecte à un serveur XMPP et s'authentifie. La session XMPP ainsi ouverte (et qui dure typiquement des heures, voire des jours) permet de transmettre des strophes (l'élément de base d'une conversation XMPP), ces strophes pouvant être des messages lisibles par un humain (« Ça va ? Tu viens bouffer ? ») comme des messages de service comme ceux informant de la présence d'un correspondant. (Il y a aussi des messages entre serveurs, pour qu'un utilisateur puisse parler à un client d'un autre service. Contrairement à Facebook ou MSN, XMPP est un protocole Internet, donc ouvert) Une session typique est :

— Récupération de son carnet, stocké sur le serveur,
— Envoi de l'information de présence (« Je suis là »),
— Échange de messages, mises à jour du carnet, réception des informations de présence des potes, etc, tout au long de la session.
— Fin de la session quand on va se coucher le soir.

Notons que les utilisateurs ne sont pas forcément des humains, ils peuvent être des "*bots*".

Terminées, les généralités, place au protocole. Ce RFC est très long (cent seize pages), mais la longueur vient surtout du nombre de types de messages possible, pas tellement de la complexité du protocole. Je ne vais pas détailler chaque fonction, seulement mettre en évidence quelques points.

La plus importante fonction est évidemment l'échange de messages, donc commençons par elle (alors que le RFC fait passer la gestion du carnet et celle de la présence d'abord). Elle figure en section 5. Le principe est que le client XMPP transmet au serveur des éléments `<message>`. Ceux-ci sont délivrés à un autre client connecté au même serveur, ou bien routés vers un autre serveur. XMPP fonctionne donc par "*push*", pas par "*pull*" : l'information doit être délivrée tout de suite (la section 8.5.2.1.1 précise ce que doit faire un serveur si un serveur distant n'est pas disponible de suite).

L'usage le plus courant est sous la forme d'une session de discussion où plusieurs messages vont se suivre. Dans ces cas, l'élément `<message>` reçoit un attribut `type` dont la valeur est `chat` (section 5.2.2). Voici un exemple de message (vous noterez vite que tous les examples XMPP sont tirés de Roméo et Juliette ; je n'ai pas traduit le texte, que j'ai laissé dans la langue originale, celle de Shakespeare, pas celle des amants de Vérone, qui s'échangeaient plutôt des mots doux en italien).

```
<message
    from='juliet@example.com/balcony'
    id='ktx72v49'
    to='romeo@example.net'
    type='chat'
    xml:lang='en'>
  <body>Art thou not Romeo, and a Montague?</body>
</message>
```

Tiens, justement, question langues, XMPP permet d'avoir plusieurs versions du même message, en des langues différentes (section 5.2.3). Voici un exemple en anglais et en tchèque :

```
<message
    from='juliet@example.com/balcony'
    id='z94nb37h'
    to='romeo@example.net'
    type='chat'
    xml:lang='en'>
  <body>Wherefore art thou, Romeo?</body>
  <body xml:lang='cs'>
     Pro&#269;e&#381; jsi ty, Romeo?
   </body>
</message>
```

La section 3 explique le mécanisme de gestion de la présence. Celle-ci n'est transmise qu'aux abonnés, pas au public. Pour s'abonner, on envoie une demande (ici de Roméo vers Juliette) :

```
<presence id='xk3h1v69'
                to='juliet@example.com'
                type='subscribe'/>
```

Cette demande est en général présentée à l'utilisateur pour qu'il l'approuve (section 3.1.3). Le serveur peut alors répondre :

```
<presence from='juliet@example.com'
                id='xk3h1v69'
                to='romeo@example.net'
                type='subscribed'/>
```

Une fois qu'on est approuvé, un peut recevoir des informations de présence (section 4). Celles-ci indiquent si l'entité (la personne ou le programme mais, en général, les programmes ne s'absentent pas) est disponible ou pas. Les clients XMPP l'affichent alors par exemple en vert si le contact est disponible et en rouge autrement, ou bien avec du texte comme "*online*" et "*offline*". Donc, lorsque je lance mon client XMPP, ou bien lorsque je change manuellement l'état (le menu en bas de la fenêtre des contacts, avec Pidgin), le client XMPP envoie au serveur :

```
<presence/>
```

et celui-ci le retransmet aux abonnés (section 4.2.2). Le client peut aussi marquer explicitement qu'il ne veut pas être dérangé :

```
<presence type='unavailable'/>
```

information à laquelle on peut ajouter une explication :

```
<presence type='unavailable' xml:lang='fr'>
         <status>Je suis en vacances.</status>
       </presence>
```

La section 2 concerne la gestion du carnet ("*roster*"). Le serveur le stocke et le transmet à l'utilisateur, ce qui permet à celui-ci de disposer de son carnet depuis n'importe quelle machine (rappelons que XMPP est un protocole ouvert et que n'importe qui peut installer un serveur : celui-ci n'est donc pas forcément une grosse entreprise anonyme dans le nuage, il peut être géré par votre entreprise, votre association, par un copain ou par vous-même). Cela ne se fait évidemment qu'après authentification (la liste de mes copains ne regarde que moi...). À noter qu'un des changements depuis le RFC 3921 est que le carnet peut désormais être stocké sur un autre serveur que celui où on se connecte.

En gros, pour avoir le carnet, le client envoie la strophe :

```
<iq from='juliet@example.com/balcony'
          id='bv1bs71f'
          type='get'>
       <query xmlns='jabber:iq:roster'/>
</iq>
```

et le serveur lui transmet (ici, seulement trois correspondants, pour ne pas rendre cet article trop long) :

```
<iq id='bv1bs71f'
          to='juliet@example.com/balcony'
          type='result'>
        <query xmlns='jabber:iq:roster' ver='ver11'>
          <item jid='romeo@example.net'
                name='Romeo'
                subscription='both'>
            <group>Friends</group>
          </item>
          <item jid='mercutio@example.com'
                name='Mercutio'
                subscription='from'/>
          <item jid='benvolio@example.net'
                name='Benvolio'
                subscription='both'/>
        </query>
</iq>
```

Dans la liste d'élements `<item>` renvoyés, l'attribut `jid` contient l'adresse XMPP formelle (JID = "*Jabber ID*", voir le RFC 7622) et `name` un identificateur plus convivial. L'élément `<subscription>` indique si le correspondant est abonné à notre présence, nous à la sienne, ou bien les deux. Le schéma XML complet figure en annexe D, en utilisant le langage XSD.

La section 2 couvre également le cas des mises à jour du carnet (le client ajoute ces mises à jour dans son `<query>`). Notez que XMPP peut aussi utiiser des enregistrements au format vCard (RFC 6350) pour remplir le carnet (annexe C et XEP-0054 `<http://xmpp.org/extensions/xep-0054.html>`). Un serveur comme ejabberd gère cette possibilité.

La section 7 fournit un exemple plus détaillé d'une session complète, avec quatre utilisateurs. Lecture nécessaire si vous voulez comprendre tous les détails de XMPP.

Comme nous vivons dans un monde cruel (regardez ce qui est arrivé de tragique à Roméo et Juliette), XMPP a besoin d'une section sur la sécurité, la numéro 11. Elle traite des points comme l'importance de ne **pas** répondre aux demandes de présence si le demandeur n'est pas autorisé (soit explicitement, soit via une politique générale, par exemple l'autorisation de tous pour tous dans une même entreprise).

Qu'est-ce qui a changé depuis le RFC 3921 ? L'annexe E résume les principales différences. La plupart sont très pointues et ne se rencontrent pas tous les jours.

XMPP étant un protocole ancien et éprouvé, il est très largement déployé et on trouve de nombreuses implémentations :

— Des serveurs comme ejabberd ou jabberd, si on veut créer son propre service,
— Des clients comme Pidgin ou Empathy sur Unix (j'ai écrit un article spécifique pour les clients sur Android `<http://www.bortzmeyer.org/xmpp-android.html>`),
— Des serveurs gratuits comme jabber.org `<http://www.jabber.org/>` ou comme Google Talk (oui, c'est tout simplement du XMPP et un abonné de Google Talk peut interagir avec n'importe quel autre abonné d'un autre service XMPP).

Et enfin, si vous voulez m'écrire, mes adresses XMPP sont `bortzmeyer@dns-oarc.net` pour le travail et `bortzmeyer@gmail.com` (du Google Talk) pour le reste.