RFC 6120 : Extensible Messaging and Presence Protocol (XMPP): Core

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 13 avril 2011

Date de publication du RFC : Mars 2011

http://www.bortzmeyer.org/6120.html

Jabber est le nom original, et XMPP le nom IETF d'un protocole d'échange d'informations encodées en XML, protocole surtout connu pour son utilisation dans la messagerie instantanée. Ce RFC 6120 [1] met à jour la définition originale de XMPP, qui était dans le RFC 3920.

La messagerie instantanée est une application très populaire surtout chez les moins de dix ans. Elle permet d'échanger des messages sans prendre le temps de réfléchir et en général sans craindre qu'ils soient archivés (contrairement au courrier électronique où il faut réfléchir car ce que l'on écrit restera). Elle sert à de nombreux usages, discussions entre "*geeks*", systèmes de surveillance avec notification des problèmes, etc. À l'exception du vénérable protocole IRC (décrit - insuffisamment - dans le RFC 2810 et suivants), les systèmes existants de messagerie instantanée sont tous basés sur des protocoles privés et fermés comme MSN. La normalisation qui fait le succès du courrier électronique n'a pas réussi ici.

D'où le vieux projet (RFC 2779) de développer un protocole moderne et ouvert, adapté à cette application. C'est ce qu'on fait les développeurs du protocole Jabber en 1999, projet adopté ultérieurement par l'IETF, rebaptisé XMPP en 2002 et normalisé désormais dans ce RFC. Notre RFC décrit un protocole généraliste, un RFC compagnon, le RFC 6121, normalise les aspects spécifiques à la messagerie instantanée. Il est donc objectivement incorrect de dire que XMPP est un protocole de messagerie instantanée (c'est en fait un protocole d'échange de données XML en temps réel) mais je suis sûr que mes lecteurs me pardonneront mes abus de langage. Ce RFC 6120 est très long (notons pour la petite histoire que dix RFC sont quand même encore plus longs, le détenteur du record étant le RFC 5661), avec plus de deux cents pages, mais il est surtout composé de longues listes de détails, les principes sont relativement simples. En dépit de la longueur de ce RFC, il ne spécifie pas tout XMPP de nombreuses extensions

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

existent, certaines très populaires (comme la possibilité pour un client de créer un compte sur un serveur directement via XMPP) et elles sont en général décrites, non pas dans un RFC, mais dans les XEP <http://xmpp.org/xmpp-protocols/xmpp-extensions/> (”*XMPP Extension Protocols*”) de la XMPP Standards Foundation.

Quels sont les principes de XMPP (section 1.3) ? Les données sont encodées en XML (avec quelques bémols notés en section 11 comme l'interdiction des commentaires XML, ou des encodages autres qu'UTF-8), dans des **strophes** (”*stanzas*”) qui sont des petits paquets d'information. Citons Wikipédia : « Les stances sont, dans les sujets graves et spirituels, ce que le couplet est dans les chansons et la strophe dans les odes ». Ces strophes sont transmises sur une connexion TCP (section 3) entre client et serveur, ou bien entre serveur et serveur (la section 2.5 contient une très intéressante présentation des concepts « réseau » de XMPP, notamment pour répondre à la question philosophique « XMPP est-il pair-à-pair ? »). Les entités qui communiquent (ce ne sont pas forcément des humains, Jabber ayant été conçu pour permettre également la communication entre programmes) sont identifiées par une adresse, qui a une forme syntaxique proche de celle du courrier électronique. Par exemple, mon adresse XMPP est bortzmeyer@gmail.com.

XMPP, comme le courrier électronique, fonctionne par le connexion de serveurs, chacun chargé d'un ou plusieurs domaines. Les clients se connectent à un serveur (par exemple talk.l.google.com. pour Google Talk, par exemple) et s'y authentifient, puis ils transmettent les messages aux serveurs, qui se chargent de les acheminer au serveur suivant (section 2.1 du RFC). Les serveurs sont trouvés en utilisant les enregistrements SRV (RFC 2782) du DNS (section 3.2) par exemple, pour un client de l'OARC <https://www.dns-oarc.net/> :

```
% dig SRV _xmpp-client._tcp.dns-oarc.net
...
;; ANSWER SECTION:
_xmpp-client._tcp.dns-oarc.net. 3223 IN SRV     0 0 5222 jabber.dns-oarc.net.
...
```

L'usage de SRV fait que le numéro de port utilisé n'est pas toujours le même mais 5222 (pour les communications de client à serveur) et 5269 (pour celles de serveur à serveur) sont les valeurs les plus courantes (section 14.7).

Les adresses (JID pour ”*Jabber Identifier*”) sont décrites dans un autre document, le RFC 7622. Outre le classique nom@domaine, elles peuvent comporter une **ressource**, indiquée après la barre oblique (par exemple bortzmeyer@gmail.com/AFNIC).

La représentation des données en XML est décrite dans la section 4. Un flux de données, un ”*stream*”, est un **élément** XML <stream>, qui comporte une ou plusieurs strophes, <stanza>. Il existe de nombreux types de strophes (section 8) sans compter les messages d'erreur, décrits en 4.9. Voici un exemple classique d'un échange (C étant le client et S le serveur), avec des strophes de type message :

```
C:   <message from='juliet@example.com'
             to='romeo@example.net'
             xml:lang='en'>
C:     <body>Art thou not Romeo, and a Montague?</body>
C:   </message>
S:   <message from='romeo@example.net'
             to='juliet@example.com'
             xml:lang='en'>
S:     <body>Neither, fair saint, if either thee dislike.</body>
S:   </message>
```

Des exemples plus détaillés figurent en section 9.

Contrairement à beaucoup d'autres protocoles de messagerie instantanée, conçus uniquement pour jouer, XMPP, étant prévu aussi pour le travail, dispose de fonctions de sécurité très riches. La section 5 précise l'utilisation de TLS et la 6 celle de SASL. La section 13 détaille en profondeur tous les problèmes de sécurité de XMPP.

La section 8 détaille les types de strophes disponibles. Trois types sont définis dans ce RFC, `<message>`, `<presence>` et `<iq>`. `<message>`, dont la sémantique est décrite en section 8.2.1, transporte un... message, tandis que `<iq>` (section 8.2.3) est une interrogation ("*Info / Query*"). Ces éléments XML ont un certain nombre d'**attributs** par exemple l'attribut `to`, sections 4.7.2 et 8.1.1, qui spécifie l'adresse du destinataire (`romeo@example.net` dans la pemière strophe de l'exemple ci-dessus). La syntaxe complète, exprimée en W3C Schema, figure dans l'annexe A.

Du fait de l'utilisation de XML, des attributs standard comme `xml:lang` sont possibles (section 8.1.5) et on peut ainsi marquer un message en tchèque :

```
<presence from='romeo@example.net/orchard' xml:lang='en'>
  <show>dnd</show>
  <status>Wooing Juliet</status>
  <status xml:lang='cs'>Dvo&#345;ím se Julii</status>
</presence
```

L'annexe D résume les changements depuis le RFC 3920, qui avait été fait en 2004. Le format des adresses a été sorti pour être dans un document à part, le RFC 7622, le vieux protocole de rappel ("*dial-back*") a été retiré des RFC, TLS est devenu obligatoire (section 13.8, il reste à voir si ce sera mis en œuvre : au moment de la sortie de notre RFC, très peu de clients XMPP Android `<http://www.bortzmeyer.org/xmpp-android.html>` avaient TLS), et le reste est un certain nombre de clarifications et de corrections : il ne s'agit donc pas d'une nouvelle version du protocole, juste d'une évolution inspirée par l'expérience.

XMPP est aujourd'hui mis en œuvre dans de nombreux logiciels. C'est ainsi que le service de messagerie instantanée Google Talk utilise XMPP. Le client XMPP (qui est aussi un client de nombreux autres protocoles) le plus populaire sur Unix est sans doute Pidgin (ex-Gaim). Côté serveurs, le plus pittoresque (et pas le moins utilisé) est sans doute ejabberd, écrit en Erlang. Mais il y en a d'autres comme Movim `<http://www.movim.eu/>`.

Les développeurs liront avec intérêt le livre "*Programming Jabber*" `<http://www.bortzmeyer.org/programming-jabber.html>` qui explique comment utiliser Jabber/XMPP dans ses propres programmes.