RFC 5388 : Information Model and XML Data Model for Traceroute Measurements

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 11 décembre 2008

Date de publication du RFC : Décembre 2008

http://www.bortzmeyer.org/5388.html

Le programme traceroute est un des piliers de l'Internet. Cet outil de déboguage a servi à des dizaines de milliers d'administrateurs réseaux, lors de difficiles sessions de recherche de panne ou de divers problèmes. Né de la nécessité, il n'a jamais eu de modèle de données formel, ni de norme de présentation des résultats. C'est désormais fait avec ce nouveau document.

Cette normalisation permet notamment de stocker les résultats d'un traceroute et de les comparer de manière automatique à un autre (il existe déjà de tels programmes de comparaison, par exemple sous forme d'un moniteur <http://search.cpan.org/~trockij/mon-0.99.2/mon.d/traceroute.monitor> pour le logiciel de surveillance réseau mon <http://mon.wiki.kernel.org/>, mais tous ne comparent que la sortie texte de traceroute et peuvent donc être perturbés par des changements purement de présentation). La section 1 détaille ce cahier des charges.

La section 3 rappelle le principe de fonctionnement de traceroute : envoyer des paquets UDP (certaines versions peuvent utiliser TCP ou ICMP ; l'annexe A contient une très intéressante description des différentes versions de traceroute) à une machine distante, en mettant délibérement des valeurs trop basses au TTL (ou "*Hop Count*") des paquets IP. Les routeurs qui, après la décrémentation de ce champ, trouveront zéro, signaleront leur existence en envoyant un paquet ICMP TIME_EXCEEDED (RFC 792[1]).

La section 4 liste ensuite les résultats qu'on obtient avec traceroute. Selon la version de ce dernier, on peut trouver :

— L'index du routeur dans le chemin,
— Son adresse IP,
— Le RTT,

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc792.txt

— Les labels MPLS (RFC 4950),
— etc.

Enfin, la section 5 définit le modèle de données utilisé pour créer le fichier XML de résultats. 5.1 contient les types de données. Certains sont tirés des schémas du W3C comme `Boolean` ou `DateTime`, d'autres tirés d'autres RFC (par exemple le RFC 4001 pour `InetAddress`, les adresses IP), d'autres enfin définis ici. La section 5.2 donne la liste de tous les élements XML utilisés comme `ProbeRoundTripTime` (section 5.2.3.8) qui contient le RTT en millisecondes. On peut donc écrire :

```
<ProbeRoundTripTime>
   <roundTripTime>13</roundTripTime>
</ProbeRoundTripTime>
```

Cet élement `ProbeRoundTripTime` peut, alternativement, contenir un élément `roundTripTimeNotAvailable`.

La section 6 explique le choix de XML, notamment parce que les fichiers XML sont lisibles par un humain et parce qu'il existe un grand nombre d'outils existants. Cette section explique aussi pourquoi des modèles de données proches comme la MIB du RFC 4560 (également discutée en annexe C) ou l'encodage d'IPFIX dans le RFC 7011 n'ont pas été utilisés.

Enfin, une longue section 7 contient le schéma lui-même, défini avec le langage W3C Schema. L'élément `ProbeRoundTripTime` cité plus haut est ainsi défini comme le choix entre un entier ou la valeur `roundTripTimeNotAvailable` :

```
<xs:element name="ProbeRoundTripTime"
                            type="_roundTripTime">
</xs:element>

<xs:complexType name="_roundTripTime">
   <xs:choice>

   <xs:element name="roundTripTime">
       <xs:simpleType>
         <xs:restriction base="xs:unsignedInt"/>
       </xs:simpleType>
     </xs:element>

     <xs:element name="roundTripTimeNotAvailable">
       <xs:complexType/>
     </xs:element>

   </xs:choice>
 </xs:complexType>
```

Je ne connais pas encore d'implémentation « officielle » de traceroute qui produise ces fichiers XML. Si vous en écrivez une, attention aux caractères spéciaux (speciaux pour XML comme & ou ¡) dans les noms de machines : rien ne vous garantit qu'une requête DNS inverse ne va pas vous renvoyer des noms avec de tels caractères. Comme toujours quand un programme produit du XML <`http://www.bortzmeyer.org/creer-xml-par-programme.html`>, il ne doit pas se contenter de mettre des <`balise`> un peu partout, il doit aussi veiller à ces caractères spéciaux.

Il existe à ma connaissance une mise en œuvre expérimentale de traceroute qui produit du XML à ce format (dérivée du traceroute de NANOG <`ftp://ftp.login.com/pub/software/traceroute/`>). Elle est assez sommaire mais elle marche. (en ligne sur `http://www.bortzmeyer.org/files/traceroute-nanog-xml.c`).

Il nécessite libxml2 <`http://xmlsoft.org/`>. Pour le compiler, sur GNU/Linux :

```
cc -DXML -lresolv  $(xml2-config --cflags) $(xml2-config --libs) \
             traceroute-nanog-xml.c -o traceroute-nanog-xml
```

Sur FreeBSD, même chose sans le `-lresolv`. Ensuite, on ne lance avec l'option `-X` pour produire du XML.

Une fois produits, les fichiers XML peuvent être vérifiés, par exemple avec xmllint <`http://xmlsoft.org/`> :

```
% xmllint --noout --schema traceroute.xsd traceroute.xml
traceroute.xml validates
```

Voici des exemples de résultats stockés dans ce format, un exemple pris dans le RFC (annexe D) (en ligne sur `http://www.bortzmeyer.org/files/traceroute-RFC-annexe-D.xml`) et un autre exemple (en ligne sur `http://www.bortzmeyer.org/files/traceroute-myprogram.xml`), produit par mon programme.