RFC 5572 : IPv6 Tunnel Broker with the Tunnel Setup Protocol (TSP)

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 19 février 2010

Date de publication du RFC : Février 2010

https://www.bortzmeyer.org/5572.html

Le protocole IPv6 étant très loin d'une connectivité complète dans l'Internet d'aujourd'hui (un très grand nombre d'opérateurs et de fournisseurs d'accès ne sont toujours pas capables de router de l'IPv6, malgré l'épuisement prochain des adresses IPv4 <https://www.bortzmeyer.org/epuisement-adresses-ipv4.html>), il faut souvent passer par des tunnels pour pouvoir faire de l'IPv6. Ces tunnels peuvent être configurés manuellement <https://www.bortzmeyer.org/ipv6-he.html> mais il peut être plus pratique de disposer d'un protocole qui permet aux serveurs de tunnels d'indiquer aux clients les paramètres de la connexion (comme avec PPP ou DHCP). Cela permet d'avoir des paramètres dynamiques (la section 3 liste les avantages de TSP). C'est un tel protocole que normalise ce RFC. (L'idée de base avait été présentée dans le RFC 3053 [1].)

TSP ("*Tunnel Setup Protocol*") permet donc à un **client** de demander un tunnel à un **serveur** et celui-ci, s'il est d'accord, va répondre avec les paramètres du tunnel, comme l'adresse IP.

Comme résumé dans la section 2 de notre RFC, TSP fonctionne au dessus de XML, lui-même au dessus de TCP ou UDP. Les paramètres qu'on peut négocier entre le client et le serveur sont, par exemple :

- l'authentification (certains serveurs peuvent accepter des accès anonymes mais pas tous),
- l'encapsulation, TSP permet de faire un tunnel IPv6-dans-IPv4, le cas le plus courant (RFC 4213), mais aussi IPv4-dans-IPv6 et même IPv6-dans-UDP-dans-IPv4 pour pouvoir contourner les NAT (et leur présence peut être détectée automatiquement, cf. section 2.1).
- les adresses IP utilisées,
- les serveurs DNS, aussi bien le résolveur à utiliser que les serveurs faisant autorité pour ip6.arpa, pour le préfixe délégué par le tunnel.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3053.txt

En toute rigueur, TSP n'implique pas que le serveur TSP soit également l'extrémité du tunnel. C'est le mode le plus simple (figure 2 dans le RFC) mais le serveur TSP peut être aussi un **courtier** ("*broker*"), négociant des paramètres pour le compte du serveur de tunnels (figure 1 dans le RFC et section 4.1 sur la terminologie). Le protocole de signalisation (pour se connecter au courtier) n'est pas forcément le même que le protocole du tunnel.

Le protocole complet est décrit dans la section 4. Le protocole est fort simple, le client TSP se connecte, s'authentifie, envoie sa demande et obtient une réponse. La demande est codée en XML, la réponse est précédée d'un code à trois chiffres résumant le succès ou l'échec (voir annexe B pour une liste de ces codes numériques).

Le client choisit de se connecter au dessus d'un des protocoles indiqués dans la section 4.4.1. Comme les autres exemples ultérieurs, l'exemple ici est pris sur le fichier de configuration du client TSP gw6 (qu'on obtient en s'inscrivant à leur service <http://www.go6.net/4105/freenet.asp>) pour Linux. Si on trouve dans `gw6c.conf` :

```
tunnel_mode=v6v4
```

cela indique que le client va se connecter au serveur en TCP (port 3653) avec le protocole IPv4, pour créer un tunnel IPv6. (La liste de toutes les méthodes figure dans le registre IANA <https://www.iana.org/assignments/tunnel-setup-protocol/tunnel-setup-protocol.xhtml>, décrit en section 7).

L'authentification figure en section 4.4.2. Elle utilise SASL. Elle se configure, par exemple, ainsi :

```
userid=mapetiteentreprise
passwd=monsupermotdepasse
```

Quant à la demande et à la réponse, elles sont décrites en section 4.4.3. Voici un exemple, vu dans le journal du client gw6 :

```
2009/07/20 08:43:04 I gw6c: Sent:
2009/07/20 08:43:04 I gw6c: Content-length: 217<tunnel action="create" type="v6anyv4" proxy="no">
     <client>  <address type="ipv4">208.75.84.80</address>  <keepalive interval="30">
             <address type="ipv6">::</address>  </keepalive> </client></tunnel>
2009/07/20 08:43:04 I gw6c: Received:
2009/07/20 08:43:04 I gw6c: 200 Success<tunnel action="info" type="v6v4" lifetime="604800">
       <server>    <address type="ipv4">64.86.88.116</address>
                   <address type="ipv6">2001:05c0:1000:000b:0000:0000:0000:0218</address>  </server>
       <client>    <address type="ipv4">208.75.84.80</address>
                   <address type="ipv6">2001:05c0:1000:000b:0000:0000:0000:0219</address>
                   <address type="dn">bortzmeyer.broker.freenet6.net</address>
                 <keepalive interval="30">
                 <address type="ipv6">2001:05c0:1000:000b:0000:0000:0000:0218</address>
                 </keepalive>  </client>
      </tunnel>
```

Une fois que la demande est acceptée, le tunnel est typiquement configuré automatiquement par le logiciel client (section 4.5), qui se charge d'exécuter les `ifconfig` appropriés. On voit alors son tunnel, ici en `2001:5c0:1000:b::219` :

```
sit1      Link encap:IPv6-in-IPv4
          inet6 addr: fe80::d04b:5450/64 Scope:Link
          inet6 addr: 2001:5c0:1000:b::219/128 Scope:Global
          UP POINTOPOINT RUNNING NOARP  MTU:1280  Metric:1
          RX packets:5069 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5015 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:4526230 (4.3 MiB)  TX bytes:487854 (476.4 KiB)
```

Les éléments XML échangés sont décrits dans la section 4.7 (et la DTD complète en annexe A). Par exemple, l'élément `<server>` (section 4.7.3) contient deux éléments, `<address>` et `<router>`, qui indiquent les caractéristiques IP de l'extrémité du tunnel.

Plusieurs exemples de requêtes TSP figurent dans la section 5. Avec tspc, on peut aussi les obtenir en utilisant l'option `-vvv` :

```
# /usr/sbin/tspc -vvvvv
Connecting to server with tcp
Using TSP protocol version 2.0.0
Establishing connection with tunnel broker...
Getting capabilities from server
Connection established
Authenticating bortzmeyer
Using authentification mecanism DIGEST-MD5
Authentication success
Asking for a tunnel
sent: Content-length: 252
<tunnel action="create" type="v6v4" proxy="no">
 <client>
  <address type="ipv4">192.134.7.249</address>
<keepalive interval="30"><address type="ipv6">::</address></keepalive <router>
   <prefix length="48"/>
  </router>
 </client>
</tunnel>
...
```

Et, sur le câble, voici à quoi ressemblent les paquets : `<http://www.pcapr.net/view/fropert/2010/4/4/4/tsp-freenet6.pcap.html>`.

Aujourd'hui, il existe deux implémentations libre du client, gw6c, déjà cité et tspc `<http://packages.debian.org/stable/tspc>`. Des exemples plus concrets de configuration peuvent être trouvés dans mon article sur les serveurs de tunnel `<https://www.bortzmeyer.org/tunnel-broker.html>`.