# RFC 7301 : Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 24 juillet 2014

Date de publication du RFC : Juillet 2014

https://www.bortzmeyer.org/7301.html

---

On le sait, désormais, tout l'Internet tourne sur le port 443 (et aussi sur 80 mais de moins en moins, en raison des révélations de Snowden). C'est en effet l'un des rares ports dont on peut être sûr qu'il est à peu près tout le temps ouvert, même sur les réseaux les plus affreux genre hôtel ou aéroport. En prime, les communications sur ce port sont chiffrées avec TLS (RFC 5246[1]). Si seul HTTP se servait de ce port, cela marcherait mais de plus en plus de protocoles applicatifs se bousculent pour l'utiliser. Comment leur permettre de partager le port 443 sans émeutes ? En ajoutant une extension à TLS qui permet d'indiquer le protocole applicatif qui va utiliser ce port, une fois la session TLS démarrée.

Cette extension avait surtout été conçue pour les besoins de HTTP/2 (autrefois nommée SPDY), version qui est loin d'être terminée. ALPN ("*Application-Layer Protocol Negotiation*"), objet de notre RFC, sert à indiquer la version de HTTP, et aussi désigner les services qui vont tourner au dessus de HTTP, comme WebRTC (RFC 8833). Mais elle pourra servir à d'autres protocoles. Comme, par exemple, faire coexister IMAP et HTTP sur le même port. Vous allez peut-être me dire « mais pourquoi ne pas faire cela dans une couche entre TLS et l'application, de manière à éviter de complexifier TLS ? » et le RFC vous répond que le but est de performance (éviter les allers-retours entre client et serveur) et de souplesse (choisir un certificat différent selon le protocole d'application).

À noter qu'il s'agit d'une extension **TLS** : si on veut faire coexister sur un même port des applications TLS comme HTTP et non-TLS comme SSH, il faut une autre solution (par exemple sslh <http://www.rutschle.net/tech/sslh.shtml>).

Comment fonctionne ALPN (section 3 de notre RFC) ? Le client envoie la liste des applications qu'il gère dans son message TLS `ClientHello` (section 7.4.1.2 du RFC 5246), le serveur en choisit une et

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5246.txt

la renvoie dans son ServerHello. Tout se fait donc dans la poignée de mains de TLS, sans imposer d'allers-retours supplémentaires. Si le client ne veut qu'une seule application, il envoie un message avec une liste ne comportant que cette application.

L'extension se nomme application_layer_protocol_negotiation et a le numéro 16 (désormais dans le registre IANA <https://www.iana.org/assignments/tls-extensiontype-values/tls-extensiontype-values.xhtml#tls-extensiontype-values-1>). Elle a un contenu (de type ProtocolNameList) qui contient la liste des protocoles applicatifs, en ordre de préférence décroissante. En termes TLS, cela s'écrit :

```
enum {
       application_layer_protocol_negotiation(16), (65535)
   } ExtensionType;

opaque ProtocolName<1..2^8-1>;

struct {
       ProtocolName protocol_name_list<2..2^16-1>
   } ProtocolNameList;
```

Les valeurs possibles pour les protocoles applicatifs sont dans un registre IANA <https://www.iana.org/assignments/tls-extensiontype-values/tls-extensiontype-values.xhtml#alpn-protocol-ids>. Des nouveaux protocoles pourront être ajoutés après un examen par un expert (RFC 5226, section 4.1). Pour l'instant, on y trouve le HTTP actuel, 1.1 (RFC 7230) et les variantes de SPDY. HTTP/2 sera placé dans ce registre après sa normalisation.

Cette extension sert aussi bien au client (liste des applications que le client veut utiliser) qu'au serveur (application - au singulier - choisie, après avoir ignoré celles que le serveur ne connait pas). Si le serveur ne met pas cette extension dans sa réponse, c'est ce qu'est un vieux serveur, qui ne la connait pas. S'il répond avec une alerte no_application_protocol, c'est que l'intersection de la liste du client avec celle configurée dans le serveur est hélas vide.

Si vous utilisez une application qui peut attirer l'attention de la NSA, attention : la négociation ALPN est en clair, avant que le chiffrement ne démarre (sections 4 et 5). C'était un des points les plus controversés lors des discussions à l'IETF mais cela a été décidé pour être cohérent avec le modèle d'extension de TLS (un concurrent d'ALPN, NPN <https://www.imperialviolet.org/2013/03/20/alpn.html>, n'avait pas ce défaut). Si vous tenez à cacher le protocole applicatif utilisé, vous devez faire une renégociation une fois le chiffrement commencé.

Des mises en œuvre de cette extension ? C'est apparemment fait pour NSS (voir #996250 <https://bugzilla.mozilla.org/show_bug.cgi?id=996250> et #959664 <https://bugzilla.mozilla.org/show_bug.cgi?id=959664>). Et Wireshark va pouvoir le décoder <http://www.wireshark.org/lists/wireshark-commits/201407/msg00383.html>. Pour les autres, voir le tableau synthétique <https://github.com/http2/http2-spec/wiki/ALPN-Status>.