RFC 8831 : WebRTC Data Channels

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 19 janvier 2021

Date de publication du RFC : Janvier 2021

https://www.bortzmeyer.org/8831.html

Ce RFC décrit le canal de données (à l'exclusion de l'audio et de la vidéo) entre deux navigateurs Web se parlant avec WebRTC. Ce canal utilise le protocole de transport SCTP.

Le canal de données WebRTC est une modélisation d'un tuyau virtuel entre les deux parties qui communiquent, tuyau dans lequel circuleront les données. SCTP (RFC 9260[1]) n'est pas utilisé pour les données « multimédia » comme la vidéo ou l'audio, mais seulement en cas de transfert d'autres données. On peut ainsi utiliser WebRTC pour se transmettre un fichier. (Je ne connais pas à l'heure actuelle de service WebRTC permettant cela. Vous avez un exemple ?) Le multimédia, lui, passe via SRTP, les données nécessitant une plus grande fiabilité passent par SCTP lui-même transporté sur DTLS (RFC 9147) lui même sur UDP. Du fait de cette encapsulation, vous ne verrez pas SCTP avec un logiciel comme Wireshark. La raison pour laquelle UDP est utilisé ? Permettre le passage des données même à travers le pire routeur NAT (cf. RFC 6951).

La section 3 de notre RFC décrit certains scénarios d'usage. Notons que dans certains, la fiabilité du canal de données n'est pas indispensable :

— Un jeu en ligne, où on est informé des mouvements des autres joueurs,
— Des notifications pendant une visioconférence du genre « Mamadou a coupé son micro »,
— Des transferts de fichiers (pour l'anecdote, ce n'est pas le point où les systèmes précédant WebRTC, comme IRC ou XMPP, se débrouillaient le mieux…),
— Le bavardage textuel entre les participants à une visioconférence.

La section 4, elle, décrit les exigences du canal de données. Notamment :

— Possibilité d'avoir plusieurs canaux simultanés,
— Transfert fiable des données (ce que la vidéo, par exemple, n'exige pas),
— Contrôle de congestion (cf. RFC 8085),
— Authentification et confidentialité (RFC 8826 et RFC 8827),

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc9260.txt

— Ne pas laisser fuiter d'informations, comme l'adresse IP locale.

La section 5 décrit une solution à ces exigences, l'utilisation de SCTP sur DTLS. SCTP (RFC 9260) fournit (entre autres) le contrôle de congestion (par contre, on n'utilise pas ses capacités de "*multihoming*"). Son encapsulation dans DTLS est décrite dans le RFC 8261. DTLS lui fournit authentification et confidentialité. L'utilisation d'ICE (RFC 8445) lui permet de passer à travers les routeurs NAT et certains pare-feux.

Petit détail : le fait de mettre SCTP sur DTLS (et non pas le contraire comme dans le RFC 6083) fait que, si DTLS est mis en œuvre dans l'espace utilisateur du système d'exploitation, SCTP devra l'être aussi. (Voir par exemple `third_party/usrsctp/usrsctplib` dans le code de Chromium, qui vient de .) Autre conséquence de ce choix : SCTP ne recevra pas les messages ICMP associés, et il devra donc compter, pour la découverte de la MTU du chemin, sur le RFC 4821.

Quelques extensions de SCTP sont nécessaires (section 6) comme celles du RFC 6525, du RFC 5061 (enfin, une partie d'entre elles) et du RFC 3758.

La section 6 décrit les détails protocolaires de l'ouverture du canal de données, puis de son utilisation et enfin de sa fermeture.

Pour les programmeurs, cette bibliothèque WebRTC <`https://github.com/rawrtc/rawrtc`> met en œuvre, entre autres, le canal de données.