RFC 6366 : Requirements for an Internet Audio Codec

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 7 septembre 2011

Date de publication du RFC : Août 2011

https://www.bortzmeyer.org/6366.html

Début 2010, l'IETF avait lancé un travail <https://www.bortzmeyer.org/ietf-codec-libre.html> pour la normalisation d'un codec libre pour l'Internet, c'est-à-dire qui ne serait pas plombé par des brevets ou autres appropriations intellectuelles et pourrait donc être utilisé par n'importe quel protocole Internet. Ce RFC est le premier produit par le groupe de travail, et contient le cahier des charges du futur codec (qui a finalement été publié dans le RFC 6716 [1]).

Comme toujours avec ce groupe de travail, cela n'a pas été sans mal. Par exemple, le RFC mentionne des codecs « à battre », c'est-à-dire où le nouveau codec doit être meilleur que ceux-ci, pour être pris en considération. Certains demandaient que les codecs à battre soient uniquement ceux qui sont déjà à peu près libres, d'autres voulaient qu'on compare le nouveau codec avec tous ses concurrents, même plombés. Vue la difficulté de la tâche technique, une telle comparaison aurait sérieusement menacé le projet. Finalement, les codecs libres ont été pris comme référence, mais avec mention des autres, en demandant que, à défaut de les battre, le nouveau codec ne soit pas trop pire qu'eux (section 5.2).

Le but est donc d'avoir un codec libre pour les applications Internet, comme la téléphonie sur IP, la téléconférence, l'exécution de musique en temps réel (concert où les musiciens ne sont pas au même endroit), etc. La section 3 liste, pour chaque application envisagée, ses caractéristiques propres. Car certaines applications sont bien plus exigeantes que les autres. Commençons par l'ordinaire téléphonie à deux. Un codec à large bande (16 kHz) est nécessaire, mais le cahier des charges ajoute la possibilité de fonctionner avec seulement 8 kHz, par exemple pour les vieux téléphones. En large bande, la consommation de capacité réseau attendue va de 12 à 32 kb/s. La latence imposée par le codec doit rester inférieure à 40 ms, mais 25 ms sont souhaitées. (Pour des détails sur les bandes et les latences, voir la section 5.1 du RFC.) Pas besoin d'être haute-fidélité, encore que le RFC note que le codec ne doit pas massacrer les musiques d'attente au point de les rendre désagréables.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc6716.txt

Plus exigeantes, la conférence (ou le bavardage en temps réel entre joueurs d'une même partie), où plusieur voix se mêlent. Comme ces applications ont en général davantage de capacité réseau, le codec doit cette fois fonctionner à plus de 24 kHz, pour un débit de 32 à 64 kb/s. Comme les participants à une conférence ont souvent les mains occupées par autre chose et ne tiennent pas le téléphone près de la bouche ou de l'oreille, les problèmes d'écho sont plus sévères qu'en conversation à deux. La latence doit donc être encore plus basse (moins de 30 ms, avec le souhait de descendre à 10), car elle influence la qualité de la suppression d'écho.

Mais, surtout, un système de conférence va mixer les voix de plusieurs participants. Le codec ne doit donc pas gêner le mixage.

Les applications dites de téléprésence sont encore plus demandeuses en terme de qualité, pour arriver à reproduire quelque chose qui ressemble à la présence physique.

Encore plus dur pour un codec, la téléopération (commande d'un robot à distance, par exemple). Si le retour d'information est par la voix (l'opérateur entend ce qui se passe près du robot), une très faible latence est nécessaire.

Et la diffusion de musique en train d'être jouée par plusieurs musiciens ? Des mesures faites avec des musiciens imposent une latence totale de moins de 25 ms (50 si on est prêt à accepter de sérieux problèmes de synchronisation dans l'orchestre). Compte tenu de la latence des autres composants (à commencer par le réseau), cela contraint la latence propre au codec à rester en dessous de 10 ms. Chaque milli-seconde gagnée dans le codec nous donne 200 km de distance en plus, compte-tenu de la vitesse de la lumière dans la silice de la fibre optique. On peut avoir un concert où les musiciens sont dans le même pays, mais pas s'ils sont dispersés aux quatres coins du monde. Et, pour cette application, il faut évidemment une grande qualité, donc un débit soutenu (facilement 128 kb/s).

Il y a bien sûr d'innombrables autres applications où un codec audio est utile mais ce tour d'horizon couvre bien les exigences essentielles.

Le futur codec est conçu avant tout pour être utilisé sur l'Internet. Il s'agit d'un environnement difficile pour un codec audio. La section 4 du RFC liste les contraintes qui découlent de cet environnement. D'abord, les pertes de paquets sont un phénomène normal sur l'Internet et le codec doit y réagir proprement (voir section 5.3). Notamment, lorsque les paquets recommencent à arriver après une ou plusieurs pertes :

— Il faut que le codec puisse lire les paquets, sans avoir besoin d'informations contenues dans les paquets précédents (qui sont perdus), comme demandé par le RFC 2736,
— Il faut que le décodeur puisse se resynchroniser rapidement.

Cela implique que la taille d'un paquet soit inférieure à la MTU (pour éviter la fragmentation : la perte d'un des fragments entraîne celle du paquet original tout entier), et qu'il n'y ait pas d'information implicite, car contenue dans un paquet précédent. Par exemple, si on change les paramètres d'encodage, les paquets suivants ne doivent pas considérer que le changement a été reçu : il pouvait être dans le paquet perdu. Les paramètres permettant le décodage doivent donc être dans chaque paquet (ou pouvoir être récupérés rapidement).

Cela n'interdit pas au décodeur de faire des prédictions sur les paquets suivants mais il doit pouvoir les corriger rapidement en cas de pertes. En gros, un codec résistant aux pertes de paquet doit être conçu comme un compromis entre l'efficacité (qui pousse à ne pas répéter les informations et à faire de chaque paquet un delta du précédent) et la robustesse (qui pousse à faire des paquets autonomes, pouvant être interprétés même si on a perdu les précédents).

Autre particularité de l'Internet, c'est un réseau qui fait « au mieux », et qui ne garantit pas la capacité réseau disponible. Le codec doit donc s'ajuster dynamiquement (augmentant et diminuant son débit), sans que cela ne s'entende après le décodage.

Les protocoles de la famille TCP/IP fournissent des mécanismes qui peuvent aider le codec à prendre ses décisions, comme l'ECN du RFC 3168, et il est donc recommandé des les utiliser.

Enfin, le nouveau codec arrivera dans un monde où il existe déjà des protocoles applicatifs qui ont besoin de lui, et il devra donc s'interfacer gentiment avec ces protocoles comme RTP (RFC 3550 et RFC 2736). Si le codec nécessite de négocier des paramètres entre les deux machines qui communiquent, cette négociation devra utiliser les protocoles de signalisation existants comme SDP (RFC 3261 et RFC 4566) dans le cas de SIP, et Jingle (XEP-O167 <http://xmpp.org/extensions/xep-0167.html>) dans le cas de XMPP.

La section 5 en vient aux exigences détaillées. Le but est que le rapport entre la qualité du son et le débit utilisé soit **meilleur** qu'avec les codecs existants comme Speex, iLBC (RFC 3951) et G.722.1 (norme UIT du même nom). Ces trois codecs sont a priori implémentables sans payer de "*royalties*" mais seul Speex est vraiment libre et c'est donc le seul que le nouveau codec **doit** battre. En outre, il serait souhaitable que le nouveau codec fasse ex-aequo avec AMR-NB en bande étroite (8 à 16 kb/s) et AMR-WB en bande large (12 à 32 kb/s). Cette qualité doit être mesurée pour plusieurs langues, y compris les langues à tons pour lesquelles une bonne qualité de son est impérative.

À noter que la seule exigence est de faire mieux que ces codecs : le RFC ne demande aucune interopérabilité avec eux (section 6.10).

Concernant la résistance aux pertes de paquets, le codev devra bien se comporter jusqu'à 5 % de pertes et la voix rester compréhensible jusqu'à 15 % de pertes. Comme la qualité ne dépend pas que du pourcentage global de pertes, mais aussi de leur répartition (continue ou au contraire en brusques trous noirs), la qualité devra être évaluée avec des traces réelles, et avec des répartitions de pertes telles qu'on les observe dans un accès ADSL typique.

Les codecs peuvent être gourmands en ressources machines, par exemple exiger un processeur très rapide. Or, il devra être mis en œuvre sur une grande variété de machines (de l'ordinateur généraliste au petit téléphone), qui ne disposeront parfois pas de FPU et dont les éventuels DSP auront des caractéristiques très différentes. Le RFC demande donc que le codec puisse être implémentable en virgule fixe. Du point de vue quantitatif, notre RFC exige une limite à la consommation de processeur. Pour un x86 récent, l'encodage et le décodage d'un signal en mono ne devrait prendre que 40 MHz en bande étroite (soit 2 % du temps d'un cœur à 2 Hz), et 200 MHz en très large bande. Ces chiffres peuvent paraître bas mais il faut se rappeler que le codec va aussi tourner sur des machines dont le processeur est bien plus lent que celui du PC de bureau typique.

Il est facile d'ammonceler dans un cahier des charges des exigences irréalistes, et ensuite de laisser les pauvres implémenteurs se débrouiller. Ce problème est fréquent dans les SDO où les participants à la normalisation sont des bureaucrates et des politiciens qui ne feront pas le travail de mise en œuvre eux-mêmes. L'IETF se méfie traditionnellement de ce risque et une des méthodes utilisées pour s'en préserver est l'**implémentation de référence**, une mise en œuvre du logiciel réalisée pour s'assurer que c'est possible, et qui est souvent publiée en même temps que le RFC. Notre RFC encadre cette implémentation de référence du futur codec en lui interdisant de tirer profit de fonctions spécifiques de tel ou tel matériel : elle doit être le plus portable possible. Par exemple, le RFC interdit de dépendre de l'arithmétique saturée, pour pouvoir tourner sur les processeurs qui n'ont pas cette fonction comme

certains ARM. Par contre, les « vraies » implémentations, elles, auront tout intérêt à exploiter à fond les particularités du matériel sur lesquelles elles tournent.

Outre la consommation de processeur, le nouveau codec ne doit pas être trop encombrant en mémoire (code et données), pour pouvoir être lancé sur des systèmes embarqués. Pour les données (l'état à garder en mémoire pour faire fonctionner le codec), le RFC met une limite à 2*R*C octets, où R est le taux d'échantillonage et C le nombre de canaux.

Et ce n'est qu'une petite partie des problèmes auxquels un codec sur l'Internet fait face. La section 6 charge la barque avec diverses considérations et des souhaits (les exigences précédentes étaient considérées comme impératives). Par exemple, s'il y a mixage (pour une téléconférence, par exemple), il serait bien de limiter le coût de l'opération. L'implémentation naïve du mixage est de décoder tous les flux audios, les mélanger et réencoder le résultat. Une approche où un décodage complet ne soit pas nécessaire est souhaitée. Autre souhait : un bon support de la stéréo, là encore en évitant l'approche naïve, qui consiste à encoder séparement les deux canaux. Comme ceux-ci ont probablement beaucoup de redondances, cette propriété devrait être utilisée pour limiter le débit utilisé. Il serait également très bien de gérer les erreurs dans les paquets (bits dont la valeur a changé). La plupart des applications Internet ne le font pas car le phénomène est rare. En général, un paquet arrive intact ou n'arrive pas. Mais des transformations non désirées se produisent parfois (avec des protocoles comme UDP Lite, il n'y a pas de somme de contrôle, cf. RFC 3828) et le RFC demande que la question soit étudiée, en insistant que, dans tous les cas, c'est la robustesse du codec aux pertes de paquets qui est prioritaire.

Plus rigolo, et pour les amateurs de polars, la section 6.9 demande que le codec n'élimine pas et ne dégrade pas le bruit de fond des appels : ce bruit peut servir à identifier l'origine d'un appel, et, dans le cas d'appels au 112, cela peut être crucial. Un bon exemple figure dans le génial film de Kurosawa, « Entre le ciel et l'enfer », où policiers et employés du tramway arrivent à reconnaître près de quelle ligne de tramway se trouve le criminel qui appelle, juste en écoutant le bruit de fond, et en remarquant un léger bruit, caractéristique d'un endroit où le câble est abîmé par un accident récent. (Les cinéphiles savent qu'il y a un exemple plus léger dans « La double vie de Véronique ».)

Restent les traditionnelles questions de sécurité : la section 7 rappelle l'évidence (le décodeur doit faire attention aux attaques par débordement de tampons) mais aussi l'importance d'avoir une consommation de ressources bornée : quels que soient les paquets entrants, le temps de processeur utilisé et la mémoire consommée ne doivent pas croître sans limites (autrement, une DoS serait possible en envoyant des paquets créés pour cela).

Plus subtil, le problème de la fuite d'informations lorsque la communication est normalement protégée par du chiffrement. Des attaques ont déjà été décrites où des informations importantes sur la conversation pouvaient être extraites d'un flux audio chiffré (« "*Spot me if you can : Uncovering spoken phrases in encrypted VoIP conversations*" <http://www.cs.jhu.edu/~cwright/oakland08.pdf> » et « "*Phonotactic Reconstruction of Encrypted VoIP Conversations : Hookt on fon-iks*" <http://www.cs.unc.edu/~fabian/papers/foniks-oak11.pdf> »). Le codec peut compliquer certaines de ces attaques, par exemple en ayant un mode où le débit est constant, quelle que soit la conversation.

Le codec choisi, Opus, a finalement été normalisé dans le RFC 6716 en septembre 2012.