Groupe de travail Réseau
Request for Comments : 3550
STD 64
RFC rendue obsolète : 1889
Mise à jour par RFC 5506, RFC 5761, RFC 6051, RFC 6222, RFC 7022, RFC 7160, RFC 7164, RFC 8083, RFC 8108
Catégorie : Norme
Traduction Claude Brière de L'Isle

H. Schulzrinne, Columbia University
S. Casner, Packet Design
R. Frederick, Blue Coat Systems Inc.
V. Jacobson, Packet Design

juillet 2003
décembre 2009

RTP : Protocole de transport pour les applications en temps réel

Résumé
Le présent mémoire décrit le protocole de transport en temps réel (RTP, *real-time transport protocol*). RTP fournit des fonctions de transport réseau de bout en bout qui conviennent aux applications qui transmettent des données en temps réel, telles que des données audio, vidéo ou de simulation, sur des services réseau en diffusion groupée ou en envoi individuel. RTP ne s'adresse pas à la réservation de ressource et ne garantit pas la qualité de service pour les services en temps réel. Le transport des données est augmenté d'un protocole de contrôle (RTCP) pour permettre la surveillance de la livraison des données d'une manière échelonnée sur les grands réseaux de diffusion groupée, et pour fournir un contrôle et une fonction d'identification minimales. RTP et RTCP sont conçus pour être indépendants des couches transport et réseau sous-jacentes. Le protocole prend en charge les traducteurs et mixeurs de niveau RTP.

La plus grande partie du texte du présent mémoire est identique à celui de la RFC 1889 qu'il rend obsolète. Il n'y a pas de changement du format des paquets sur le réseau, et seulement des changements des règles et des algorithmes qui gouvernent l'utilisation du protocole. Le plus gros changement est une amélioration de l'algorithme de temporisation échelonnable pour le calcul du moment d'envoi des paquets RTCP afin de minimiser les transmissions dépassant le taux prévu lorsque de nombreux participants se joignent simultanément à une session.

Table des matières

1. Introduction

Le présent mémoire spécifie le protocole de transport en temps réel (RTP), qui fournit les services de livraison de bout en bout pour les données qui ont des caractéristiques de temps réel, comme l'audio et la vidéo interactive. Ces services comportent l'identification de type de charge utile, le numérotage des séquences, l'horodatage et la surveillance de la livraison. Les applications font normalement fonctionner RTP par dessus UDP pour utiliser ses services de multiplexage et de somme de contrôle ; les deux protocoles contribuent partiellement à la fonctionnalité de protocole de transport. Cependant, RTP peut être utilisé avec d'autres protocoles réseau ou transport sous-jacents convenables (voir la Section 11). RTP prend en charge le transfert de données vers des destinations multiples en utilisant la distribution par diffusion groupée si elle est fournie par le réseau sous-jacent.

Noter que RTP ne fournit pas par lui-même de mécanisme pour assurer la livraison à temps ou pour fournir d'autres garanties de qualité de service, mais s'appuie sur des services de couches inférieures pour le faire. Il ne garantit pas la livraison ni n'empêche la livraison dans le désordre, pas plus qu'il ne suppose que le réseau sous-jacent est fiable et livre les paquets dans l'ordre. Les numéros de séquence inclus dans RTP permettent au receveur de reconstruire la séquence des paquets de l'envoyeur, mais les numéros de séquence peuvent aussi être utilisés pour déterminer la localisation appropriée pour un paquet, par exemple dans le décodage de vidéo, sans nécessairement décoder les paquets en séquence.

Bien que RTP soit principalement conçu pour satisfaire les besoins des conférences multimédia multi-participants, il n'est pas limité à cette application particulière. Des applications de mémorisation de données en continu, de simulation interactive éclatée, d'insigne actif, et de contrôle et de mesures peuvent aussi trouver un intérêt à RTP.

Le présent document définit RTP, qui consiste en deux parties intimement liées :

o le protocole de transport en temps réel (RTP), pour porter des données qui ont des propriétés de temps réel.

- le protocole de contrôle RTP (RTCP), pour surveiller la qualité de service et porter des informations sur les participants à une session en cours. Ce dernier aspect de RTCP peut être suffisant pour les sessions à "contrôle lâche", c'est-à-dire, où il n'y a pas de contrôle d'adhésion explicite et d'établissement, mais il n'est pas nécessairement destiné à prendre en charge toutes les exigences de communication de contrôle d'une application. Cette fonctionnalité peut être pleinement ou partiellement englobée dans un protocole de contrôle de session séparé, qui sort du domaine d'application du présent document.

RTP représente un nouveau style de protocole qui suit les principes de trame de niveau d'application et de traitement de couche intégrée proposé par Clark et Tennenhouse [10]. C'est à dire que RTP est destiné à être malléable afin de fournir les informations demandées par une application particulière et sera souvent intégré dans le traitement de l'application plutôt que d'être mis en œuvre comme une couche séparée. RTP est un cadre de protocole qui est délibérément incomplet. Le présent document spécifie les fonctions dont on prévoit qu'elles seront communes aux applications pour lesquelles RTP devrait être approprié. À la différence des protocoles conventionnels dans lesquels des fonctions supplémentaires peuvent être fournies en rendant le protocole plus général ou en ajoutant un mécanisme d'option qui exigera une analyse, RTP est destiné à être taillé sur mesure à travers des modifications et/ou des ajouts aux en-têtes en tant que de besoin. Les exemples sont donnés aux paragraphes 5.3 et 6.4.3.

Donc, en plus du présent document, une spécification complète de RTP pour une application particulière exigera un ou plusieurs documents d'accompagnement (voir la Section 13) :

- Un document de spécification de profil, qui définisse un ensemble de codes de type de charge utile et leurs transpositions en formats de charge utile (par exemple, de codages du support). Un profil peut aussi définir des extensions ou des modifications à RTP qui soient spécifiques d'une classe d'applications particulières. Normalement une application va fonctionner seulement sur un profil. Un profil pour les données audio et vidéo se trouve dans la RFC 3551 [1].

- Les documents de spécification de format de charge utile qui définissent comment une charge utile particulière, comme des codages audio ou vidéo, sont à transporter dans RTP.

Un exposé sur les services et algorithmes de temps réel pour leur mise en œuvre aussi bien que comme discussion de fond sur certaines décision de conception de RTP se trouvent dans [11].

1.1 Terminologie

Dans le présent document, les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" sont à interpréter comme décrit dans le BCP 14, RFC 2119 [2] et indiquent les niveaux d'exigence pour les mises en œuvre conformes de RTP.

2. Scénarios d'utilisation de RTP

Les paragraphes qui suivent décrivent certains aspects de l'utilisation de RTP. Les exemples ont été choisis pour illustrer le fonctionnement de base des applications qui utilisent RTP, et non pour limiter ce à quoi RTP peut servir. Dans ces exemples, RTP est porté au dessus de IP et UDP, et suit les conventions établies par le profil pour les données audio et vidéo spécifié dans la RFC 3551.

2.1 Conférence audio simple en diffusion groupée

Un groupe de travail de l'IETF s'est réuni pour discuter du dernier document de protocole qui utilise les services de diffusion groupée sur IP de l'Internet pour les communications vocales. On obtient une adresse de groupe de diffusion groupée et une paire d'accès par un mécanisme d'allocation. Un accès est utilisé pour les données audio, et l'autre sert au paquets de contrôle (RTCP). Ces informations d'adresse et d'accès sont distribuées aux participants prévus. Si on désire la confidentialité, les paquets de données et de contrôle peuvent être chiffrés comme spécifié au paragraphe 9.1, auquel cas une clé de chiffrement doit aussi être générée et distribuée. Les détails exacts de ces mécanismes d'allocation et de distribution sortent du domaine d'application de RTP.

L'application de conférence audio utilisée par chaque participant à une conférence envoie des données audio par petits

tronçons d'une durée de, disons 20 ms. Chaque tronçon de données audio est précédé d'un en-tête RTP ; l'en-tête RTP et les données sont à leur tour contenus dans un paquet UDP. L'en-tête RTP indique quel type de codage audio (tel que MIC, MICDA ou LPC) est contenu dans chaque paquet de façon que l'envoyeur puisse changer le codage pendant une conférence, par exemple, pour s'accommoder d'un nouveau participant qui est connecté par une liaison à faible bande passante ou réagir à des indications d'encombrement du réseau.

L'Internet, comme tous les autres réseaux de paquets, perd et réordonne occasionnellement des paquets et les retarde de délais variables. Pour tenir compte de ces dégradations, l'en-tête RTP contient des informations horaires et un numéro de séquence qui permet au receveur de reconstruire l'horaire produit par la source, de sorte que dans cet exemple, des tronçons audio sont joués de façon contiguë au haut-parleur toutes les 20 ms. Cette reconstruction horaire est effectuée séparément pour chaque source de paquets RTP dans la conférence. Le numéro de séquence peut aussi être utilisé par le receveur pour estimer combien de paquets sont perdus.

Comme les membres du groupe de travail se joignent à la conférence et la quittent en cours, il est utile de savoir qui participe à un moment donné et dans quel état sont reçues les données audio. À cette fin, chaque instance de l'application audio dans la conférence envoie périodiquement en diffusion groupée un rapport de réception plus le nom de l'utilisateur sur l'accès RTCP (de contrôle). Le rapport de réception indique comment le haut-parleur actuel reçoit et il peut être utilisé pour contrôler le codage adaptatif. En plus du plan d'utilisateur, d'autres informations d'identification peuvent aussi être incluses sous réserve des limites de la bande passante de contrôle. Un site envoie le paquet RTCP BYE (paragraphe 6.6) lorsque il quitte la conférence.

2.2 Conférence audio et vidéo

Si les supports audio et vidéo sont tous deux utilisés dans une conférence, ils sont transmis dans des sessions RTP distinctes. C'est à dire que des paquets RTP et RTCP séparés sont transmis pour chaque support utilisant deux paires d'accès UDP et/ou adresses de diffusion groupée différentes. Il n'y a pas de couplage direct au niveau RTP entre les sessions audio et vidéo, sauf qu'un usager participant aux deux sessions devrait utiliser le même nom distinctif (canonique) dans les paquets RTCP pour les deux, de sorte que les sessions puissent être associées.

Un des motifs de cette séparation est de permettre à certains participants à la conférence de ne recevoir qu'un des supports si tel est leur choix. De plus amples explications sont données au paragraphe 5.2. En dépit de cette séparation, la présonorisation synchronisée des flux audio et vidéo d'une source peut être réalisée en utilisant les informations temporelles portées dans les paquets RTCP des deux sessions.

2.3 Mélangeurs et traducteurs

Jusqu'à présent, nous avons supposé que tous les sites veulent recevoir les données des supports dans le même format. Cependant, cela peut n'être pas toujours approprié. Considérons le cas où les participants dans une zone sont connectés au moyen d'une liaison à bas débit à la majorité des participants à la conférence qui eux jouissent d'un accès par réseau haut débit. Au lieu de forcer tout le monde à utiliser le codage audio à faible bande passante, de qualité réduite, un relais au niveau RTP appelé un mélangeur peut être placé près de la zone à faible bande passante. Ce mélangeur resynchronise les paquets audio entrants pour reconstruire l'espacement constant de 20 ms généré par l'envoyeur, mixe les flux audio reconstruits en un seul flux, traduit le codage audio dans la bande passante inférieure et transmet le flux de paquets de bande passante inférieure sur la liaison à basse vitesse. Ces paquets peuvent être en envoi individuel à un seul destinataire ou en diffusion groupée à une adresse différente pour plusieurs receveurs. L'en-tête RTP inclut un moyen pour que les mélangeurs identifient les sources qui contribuent à un paquet mixé de façon que l'indication correcte de la personne qui parle puisse être fournie aux receveurs.

Certains des participants prévus à la conférence audio peuvent être connectés par des liaisons à forte bande passante qui ne sont pas forcément joignables directement par une diffusion groupée IP. Par exemple, ils peuvent être derrière un pare-feu de niveau application qui ne laissera passer aucun paquet IP. Pour ces sites, le mixage peut n'être pas nécessaire, auquel cas un autre type de relais de niveau RTP appelé un traducteur peut être utilisé. Deux traducteurs sont installés, un de chaque côté du pare-feu, celui de l'extérieur servant d'entonnoir pour tous les paquets en diffusion groupée reçus à travers une connexion sécurisée vers le traducteur à l'intérieur du pare-feu. Le traducteur à l'intérieur du pare-feu les envoie à nouveau comme paquets en diffusion groupée à un groupe de diffusion groupée restreint au réseau interne du site.

Les mélangeurs et traducteurs peuvent être conçus pour une grande variété d'objets. Voici l'exemple d'un mélangeur vidéo qui échelonne les images d'individus en flux vidéo séparés et les compose en un seul flux vidéo pour simuler une scène de groupe. D'autres exemples de traduction sont la connexion d'un groupe d'hôtes ne parlant que IP/UDP à un groupe d'hôtes qui ne comprennent que le ST-II, ou la traduction du codage paquet par paquet de flux vidéo de sources individuelles sans

resynchronisation ni mixage. Les détails du fonctionnement des mélangeurs et traducteurs sont donnés à la Section 7.

2.4 Codages selon les couches

Les applications multimédia devraient être capables d'ajuster le débit de transmission de façon à correspondre à la capacité du receveur ou à s'adapter à l'encombrement du réseau. De nombreuses mises en œuvre placent la responsabilité de l'adaptation du débit à la source. Cela ne fonctionne pas bien avec la transmission en diffusion groupée à cause des exigences de bande passante contradictoires de récepteurs hétérogènes. Le résultat est souvent un scénario de plus petit dénominateur commun, où le plus petit tuyau dans la mosaïque du réseau dicte la qualité et la fidélité de la "diffusion" multimédia en direct.

Au lieu de cela, la responsabilité de l'adaptation du débit peut être donnée aux récepteurs en combinant un codage selon les couches avec un système de transmission en couches. Dans le contexte de RTP sur diffusion groupée IP, la source peut "éplucher" les couches successives d'un signal représenté comme une hiérarchie à travers plusieurs sessions RTP dont chacune est portée par son propre groupe de diffusion groupée. Les récepteurs peuvent alors s'adapter à l'hétérogénéité du réseau et contrôler leur bande passante de réception en ne se joignant qu'au sous-ensemble approprié des groupes de diffusion groupée.

Les détails de l'utilisation de RTP avec des codages différenciés selon les couches sont donnés aux paragraphes 6.3.9, 8.3 et à la Section 11.

3. Définitions

charge utile RTP : données transportées par RTP dans un paquet, par exemple des échantillons audio ou des données vidéo compressées. Le format et l'interprétation de la charge utile sortent du domaine d'application du présent document.

paquet RTP : un paquet de données consistant en un en-tête RTP fixé, une liste éventuellement vide des sources contributives (voir ci-dessous), et les données de charge utile. Certains protocoles sous-jacents peuvent exiger que soit définie une encapsulation du paquet RTP. Normalement, un paquet du protocole sous-jacent contient un seul paquet RTP, mais plusieurs paquets RTP PEUVENT être contenus si c'est permis par la méthode d'encapsulation (voir la Section 11).

paquet RTCP : un paquet de contrôle consistant en une partie d'en-tête fixe similaire à celle des paquets de données RTP, suivi par des éléments structurés qui varient selon le type de paquet RTCP. Les formats sont définis à la Section 6. Normalement, plusieurs paquets RTCP sont envoyés ensemble comme un paquet RTCP composé en un seul paquet du protocole sous-jacent ; ceci est activé par le champ Longueur dans l'en-tête fixe de chaque paquet RTCP.

accès : c'est "l'abstraction qu'utilisent les protocoles de transport pour faire la distinction entre plusieurs destinations au sein d'un ordinateur hôte donné. Les protocoles TCP/IP identifient les accès en utilisant de petits entiers positifs" [12]. Le sélecteur de transport (TSEL, *transport selector*) utilisé par la couche de transport OSI est équivalent à un accès. RTP s'appuie sur le protocole de couche inférieure pour fournir des mécanismes comme les accès pour multiplexer les paquets RTP et RTCP d'une session.

adresse de transport : combinaison d'adresse et d'accès réseau qui identifie un point d'extrémité de niveau transport, par exemple une adresse IP et un accès UDP. Les paquets sont transmis d'une adresse de transport de source à une adresse de transport de destination.

type de support RTP : c'est une collection de types de qui peuvent être portés au sein d'une seule session RTP. Le profil RTP alloue des types de support RTP aux types de RTP.

session multimédia : ensemble de sessions RTP concurrentes parmi un groupe commun de participants. Par exemple, une visioconférence (qui est une session multimédia) peut contenir une session RTP audio et une session RTP vidéo.

session RTP : association parmi un ensemble de participants qui communiquent avec RTP. Un participant peut être impliqué dans plusieurs sessions RTP en même temps. Dans une session multimédia, chaque support est normalement porté dans une session RTP séparée avec ses propres paquets RTCP sauf que le codage lui-même multiplexe plusieurs supports en un seul flux de données. Un participant distingue plusieurs sessions RTP par la réception de différentes sessions utilisant des paires différentes d'adresses de transport de destination, où une paire d'adresses de transport comprend une adresse réseau plus une paire d'accès pour RTP et RTCP. Tous les participants à une session RTP peuvent partager une paire

d'adresses de transport de destination commune, comme dans le cas de diffusion groupée IP, ou les paires peuvent être différentes pour chaque participant, comme dans le cas de paires d'adresse réseau et d'adresse d'accès en envoi individuel. Dans le cas de l'envoi individuel, un participant peut recevoir de tous les autres participants à la session en utilisant la même paire d'accès, ou bien il peut utiliser une paire d'accès distincte pour chacun.

La caractéristique distinctive d'une session RTP est que chacune entretient un espace complet et distinct d'identifiants de SSRC (définis plus loin). L'ensemble des participants inclus dans une session RTP comporte ceux qui peuvent recevoir un identifiant de SSRC transmis par un quelconque des participants à la session RTP comme SSRC ou comme CSRC (défini aussi plus loin) ou dans RTCP. Par exemple, considérons une conférence à trois mise en œuvre en utilisant UDP en envoi individuel où chaque participant reçoit des deux autres sur des paires d'accès distinctes. Si chaque participant envoie un retour RTCP sur les données reçues d'un des autres participants seulement à ce participant, la conférence est alors composée de trois sessions RTP point à point séparées. Si chaque participant fournit un retour RTCP sur sa réception d'un autre participant aux deux autres participants, la conférence est alors composée d'une session RTP multi-parties. Ce dernier cas simule le comportement qui surviendrait avec une communication IP en diffusion groupée entre les trois participants.

Le cadre RTP permet les variations définies ici, mais un protocole de contrôle particulier ou un concept d'application va généralement imposer des contraintes à ces variations.

source de synchronisation (SSRC) : La source d'un flux de paquets RTP, identifiée par un identifiant de SSRC de 32 bits porté dans l'en-tête RTP de façon à ne pas dépendre de l'adresse réseau. Tous les paquets provenant d'une source de synchronisation font partie du même espace de synchronisation et de numéro de séquence, de sorte qu'un receveur groupe les paquets par source de synchronisation pour la réexécution *(play-back)*. Dans les exemples de source de synchronisation on trouve l'envoyeur d'un flux de paquets dérivés d'une source de signaux comme un microphone ou une caméra, ou un mixeur RTP (voir ci-dessous). Une source de synchronisation peut changer son format de données, par exemple, le codage audio, au fil du temps. L'identifiant de SSRC est une valeur choisie au hasard destinée à être unique au monde au sein d'une session RTP particulière (voir la Section 8). Un participant n'a pas besoin d'utiliser le même identifiant de SSRC pour toutes les sessions RTP dans une session multimédia ; la liaison des identifiants de SSRC est fournie au moyen de RTCP (voir au paragraphe 6.5.1). Si un participant génère plusieurs flux dans une seule session RTP, par exemple à partir de différentes caméras vidéo, chacun DOIT être identifié par un SSRC différent.

source contributive (CSRC) : Une source d'un flux de paquets RTP qui a contribué au flux combiné produit par un mixeur RTP (voir ci-dessous). Le mixeur insère une liste des identifiants de SSRC des sources qui ont contribué à la génération d'un paquet particulier dans l'en-tête RTP de ce paquet. Cette liste est appelée liste de CSRC. Un exemple d'application serait une conférence audio où un mixeur indique tous les orateurs dont le discours est combiné pour produire le paquet sortant, permettant au receveur d'indiquer l'orateur actuel, même si tous les paquets audio contiennent le même identifiant de SSRC (celui du mixeur).

système terminal : Une application qui génère le contenu à envoyer dans les paquets RTP et/ou consomme le contenu des paquets RTP reçus. Un système terminal peut agir comme une ou plusieurs sources de synchronisation dans une session RTP particulière, mais normalement comme une seule.

mixeur : Un système intermédiaire qui reçoit des paquets RTP d'une ou plusieurs sources, change éventuellement le format des données, combine les paquets d'une certaine manière puis transmet un nouveau paquet RTP. Comme la synchronisation entre ces sources d'entrées multiples ne sera généralement pas assurée, le mixeur va faire les ajustements de synchronisation entre les flux et générer sa propre synchronisation du flux combiné. Et donc, tous les paquets de données originaires d'un mixeur seront identifiés comme ayant le mixeur comme source de synchronisation.

traducteur : Un système intermédiaire qui transmet les paquets RTP avec leur identifiant de source de synchronisation intact. Comme exemples de traducteurs on a des appareils qui convertissent les codages sans mixage, des duplicateurs de diffusion groupée en envoi individuel, et des filtres de niveau application dans les pare-feu.

Moniteur : Une application qui reçoit les paquets RTCP envoyés par les participants dans une session RTP, en particulier les rapports de réception, et estime la qualité de service actuelle pour la distribution de la surveillance, des diagnostics de fautes et des statistiques à long terme. La fonction de surveillance sera vraisemblablement construite dans la ou les applications participant à la session, mais peut aussi être une application distincte qui ne participe pas par ailleurs et n'envoie ni ne reçoit de paquets de données RTP (puisque elle est sur un accès séparé). Ceux-ci sont appelés des moniteurs tiers. On peut aussi accepter qu'un moniteur tiers reçoive les paquets de données RTP mais n'envoie pas de paquets RTCP ou ne soit pas autrement compté dans la session.

moyens non RTP : Les protocoles et mécanismes qui peuvent être nécessaires en plus de RTP pour fournir un service

utilisable. En particulier, pour les conférences multimédia, un protocole de contrôle peut distribuer des adresses de diffusion groupée et des clés de chiffrement, négocier l'algorithme de chiffrement à utiliser, et définir les transpositions dynamiques entre les valeurs de type de charge utile RTP et les formats de charge utile qu'elles représentent pour les formats qui n'ont pas une valeur prédéfinie de type de charge utile. Des exemples de tels protocoles sont le protocole d'initialisation de session (SIP) (RFC 3261 [13]), la Recommandation UIT-T H.323 [14] et les applications qui utilisent SDP (RFC 2327 [15]), telles que RTSP (RFC 2326 [16]). Pour les applications simples, la messagerie électronique ou une base de données de conférence peuvent aussi être utilisées. La spécification de tels protocoles et mécanismes sort du domaine d'application du présent document.

4. Ordre des octets, alignement, et format de l'heure

Tous les champs d'entiers sont portés dans l'ordre des octets du réseau, c'est-à-dire, l'octet de plus fort poids en premier. Cet ordre des octets est communément appelé gros-boutien. L'ordre de transmission est décrit en détail dans [3]. Sauf mention contraire, les constantes numériques sont en décimal (base 10).

Toutes les données d'en-tête sont alignées à leur longueur naturelle, c'est-à-dire que les champs de 16 bits sont alignés sur les décalages pairs, les champs de 32 bits sont alignés sur des décalages divisibles par quatre, etc. Les octets désignés comme bourrage ont la valeur zéro.

L'heure de l'horloge murale (date et heure absolues) est représentée en utilisant le format d'horodatage du protocole de l'heure du réseau (NTP), qui est en secondes par rapport à 0 h UTC du 1er janvier 1900 [4]. La pleine résolution de l'horodatage NTP est un nombre de 64 bits non signé à virgule fixe avec la partie entière dans les 32 premiers bits et la partie décimale dans les 32 derniers bits. Dans certains champs où une représentation plus compacte est appropriée, seuls les 32 bits du milieu sont utilisés, c'est-à-dire, les 16 bits de poids faible de la partie entière et les 16 bits de poids fort de la partie décimale. Les 16 bits de poids fort de la partie entière doivent être déterminés de façon indépendante.

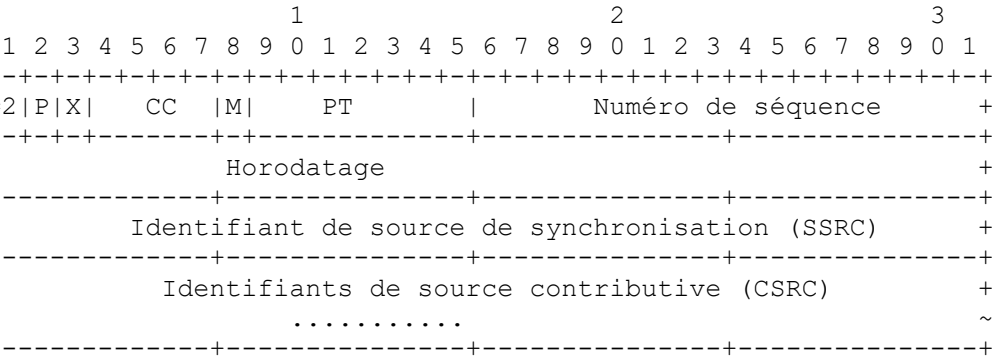
Une mise en œuvre n'est pas obligée d'utiliser le protocole de l'heure du réseau pour utiliser RTP. D'autres sources horaires, ou pas de source horaire du tout peuvent être utilisées (voir la description du champ horodatage de NTP au paragraphe 6.4.1). Cependant, utiliser NTP peut être utile pour synchroniser les flux transmis à partir d'hôtes séparés.

L'horodatage NTP reviendra à zéro quelque part en 2036, mais pour les besoins de RTP, on n'utilise que les différences entre les paires d'horodatages NTP. Pour autant que les paires d'horodatages puissent être supposées être dans les 68 ans l'une de l'autre, l'utilisation de l'arithmétique modulaire pour les soustractions et les comparaisons rend le retour à zéro sans importance.

5. Protocole de transfert de données RTP

5.1 Champs d'en-tête fixes RTP

L'en-tête RTP a le format suivant :



Les douze premiers octets sont présents dans tout paquet RTP, alors que la liste des identifiants de CSRC n'est présente que lorsqu'elle est insérée par un mixeur. Les champs ont la signification suivante.

version (V) : 2 bits

Ce champ identifie la version de RTP. La version définie par la présente spécification est deux (2). (La valeur 1 est utilisée par la version du premier projet de RTP et la valeur 0 est utilisée par le protocole initialement mis en œuvre dans l'outil audio "vat".)

bourrage (P) : 1 bit
Si le bit bourrage est mis (à 1), le paquet contient un ou plusieurs octets de bourrage supplémentaires en fin ; ils ne font pas partie de la charge utile. Le dernier octet du bourrage contient le compte du nombre d'octets de bourrage qui devraient être ignorés, y compris celui-là. Le bourrage peut être réclamé par certains algorithmes de chiffrement avec des tailles de bloc fixes ou pour porter plusieurs paquets RTP dans une unité de données de protocole de couche inférieure.

extension (X) : 1 bit
Si le bit extension est mis, l'en-tête fixe DOIT être suivi d'exactement une extension d'en-tête, avec un format défini au paragraphe 5.3.1.

compte de CSRC (CC) : 4 bits
Le compte de CSRC contient le nombre d'identifiants de CSRC qui suivent l'en-tête fixe.

marqueur (M) : 1 bit
L'interprétation du marqueur est définie par un profil. Il est destiné à permettre que des événements significatifs, tels que les limites de trame, soient marqués dans le flux de paquets. Un profil PEUT définir des bits de marqueur supplémentaires ou spécifier qu'il n'y a pas de bit marqueur en changeant le nombre de bits dans le champ de type de charge utile (voir au paragraphe 5.3).

Format de charge utile (PT) : 7 bits
Ce champ identifie le format de la charge utile RTP et détermine son interprétation par l'application. Un profil PEUT spécifier une transposition statique par défaut des codes de type de charge utile en formats de charge utile. Des types de codes de charge utile PEUVENT être définis de façon dynamique par des moyens non RTP (voir la Section 3). Un ensemble de transpositions par défaut pour l'audio et la vidéo est spécifié dans la RFC 3551 [1]. Une source RTP PEUT changer le type de charge utile durant une session, mais ce champ NE DEVRAIT PAS être utilisé pour multiplexer des flux de supports distincts (voir au paragraphe 5.2). Un receveur DOIT ignorer les paquets avec des types de charge utile qu'il ne comprend pas.

Numéro de séquence : 16 bits
Le numéro de séquence s'incrémente de un pour chaque paquet de données RTP envoyé, et peut être utilisé par le receveur pour détecter la perte de paquet et pour restaurer la séquence de paquets. La valeur initiale du numéro de séquence DEVRAIT être aléatoire (imprévisible) pour rendre plus difficiles les attaques de texte source connu contre le chiffrement, même si la source elle-même ne chiffre pas selon la méthode spécifiée au paragraphe 9.1, parce que les paquets peuvent s'écouler à travers un traducteur qui le fait. Les techniques de choix de nombres imprévisibles sont exposées dans [17].

Horodatage : 32 bits
L'horodatage reflète le moment d'échantillonnage du premier octet dans le paquet de données RTP. Le moment d'échantillonnage DOIT être déduit d'une horloge qui incrémente le temps de façon monotone et linéaire pour permettre les calculs de synchronisation et de gigue (voir au paragraphe 6.4.1). La résolution de l'horloge DOIT être suffisante pour la précision de synchronisation désirée et pour mesurer la gigue d'arrivée des paquets (un tic par trame vidéo n'est normalement pas suffisant). La fréquence d'horloge dépend du format des données portées comme charge utile et elle est spécifiée de façon statique dans le profil ou la spécification de format de charge utile qui définit le format, ou PEUT être spécifiée de façon dynamique pour les formats de charge utile définis par des moyens non RTP. Si les paquets RTP sont générés de façon périodique, on doit utiliser le moment d'échantillonnage nominal tel que déterminé à partir de l'horloge d'échantillonnage et non par lecture de l'horloge système. Par exemple, pour de l'audio à débit fixe, l'horloge d'horodatage va vraisemblablement s'incrémenter de un pour chaque période d'échantillonnage. Si une application audio lit des blocs couvrant 160 périodes d'échantillonnage à partir de l'appareil d'entrée, l'horodatage sera augmenté de 160 pour chaque bloc de cette sorte, sans considération du fait que le bloc est transmis dans un paquet ou abandonné parce que silencieux.

La valeur initiale de l'horodatage DEVRAIT être aléatoire, comme pour le numéro de séquence. Plusieurs paquets RTP consécutifs auront des horodatages égaux si ils sont (logiquement) générés en une seule fois, par exemple, appartenant à la même trame vidéo. Les paquets RTP consécutifs PEUVENT contenir des horodatages qui ne sont pas à croissance monotone, si les données ne sont pas transmises dans l'ordre dans lequel elles ont été échantillonnées, comme dans le cas de trames vidéo MPEG interpolées. (Les types de charge utile des paquets tels que transmis seront toujours monotones.)

Les horodatages RTP provenant de différents flux de support peuvent avancer à des vitesses différentes et ont généralement

des décalages indépendants et aléatoires. Donc, bien que ces horodatages soient suffisants pour reconstruire la succession dans le temps d'un seul flux, la comparaison directe des horodatages RTP des différents supports n'est pas efficace pour la synchronisation. Au lieu de cela, pour chaque support, l'horodatage RTP est rapporté au moment d'échantillonnage en l'appariant à un horodatage provenant de l'horloge de référence (horloge murale) qui représente l'heure à laquelle les données correspondant à l'horodatage RTP ont été échantillonnées. L'horloge de référence est partagée par tous les supports à synchroniser. Les paires d'horodatage ne sont pas transmises dans chaque paquet de données, mais à un débit inférieur dans les paquets SR de RTCP, comme décrit au paragraphe 6.4.

Le moment d'échantillonnage est choisi comme point de référence pour l'horodatage RTP parce qu'il est connu du point d'extrémité de transmission et a une définition commune pour tous les supports, indépendamment des délais de codage ou autres traitements. L'objectif est de permettre une présentation synchronisée de tous les supports échantillonnés au même moment.

Les applications qui transmettent les données mémorisées plutôt que les données échantillonnées en temps réel utilisent normalement une présentation virtuelle du temps dérivée de l'heure de l'horloge murale pour déterminer quand la prochaine trame ou autre unité de chaque support des données mémorisées devrait être présentée. Dans ce cas, l'horodatage RTP reflètera le moment de présentation pour chaque unité. C'est-à-dire, l'horodatage RTP pour chaque unité sera relaté à l'heure de l'horloge murale à laquelle l'unité devient actuelle sur le plan temporel de présentation virtuelle. La présentation réelle survient un petit peu après, comme déterminé par le receveur.

Un exemple qui décrit la narration audio en direct de vidéo préenregistrée illustre la signification du choix du moment d'échantillonnage comme point de référence. Dans ce scénario, la vidéo serait présentée localement pour que le narrateur la voit et serait simultanément transmise en utilisant RTP. Le "moment d'échantillonnage" d'une trame vidéo transmise en RTP serait établi en référençant son horodatage à l'heure de l'horloge murale à laquelle cette trame vidéo a été présentée au narrateur. Le moment d'échantillonnage pour les paquets audio RTP qui contiennent le discours du narrateur serait établi par référence à la même heure d'horloge murale que celle à laquelle le contenu audio a été échantillonné. L'audio et la vidéo peuvent même être transmis par des hôtes différents si les horloges de référence sur les deux hôtes sont synchronisées par des moyens tels que ceux de NTP. Un receveur peut alors synchroniser la présentation des paquets audio et vidéo en mettant en rapport leurs horodatages RTP en utilisant les paires d'horodatage dans les paquets SR RTCP.

SSRC : 32 bits
Le champ SSRC identifie la source de synchronisation. Cet identifiant DEVRAIT être choisi au hasard, avec l'intention que deux sources de synchronisations au sein de la même session RTP n'aient pas le même identifiant de SSRC. Un exemple d'algorithme pour générer un identifiant aléatoire est présenté à l'Appendice A.6. Bien que la probabilité que plusieurs sources choisissent le même identifiant soit faible, toutes les mises en œuvre RTP doivent être prêtes à détecter et résoudre les collisions. La Section 8 décrit la probabilité de collision ainsi qu'un mécanisme pour résoudre les collisions et détecter les boucles de transmission de niveau RTP sur la base de l'unicité de l'identifiant de SSRC. Si une source change son adresse de source de transport, elle doit aussi choisir un nouvel identifiant de SSRC pour éviter d'être interprétée comme une source en boucle (voir au paragraphe 8.2).

liste de CSRC : 0 à 15 éléments, de 32 bits chacun
La liste de CSRC identifie les sources contributives à la charge utile contenue dans ce paquet. Le nombre des identifiants est donné par le champ CC. Si il y a plus de 15 sources contributives, seules 15 peuvent être identifiées. Les identifiants de CSRC sont insérés par des mixeurs (voir au paragraphe 7.1) en utilisant les identifiants de SSRC des sources contributives. Par exemple, pour les paquets audio, on fait la liste des identifiants de CSRC de toutes les sources qui ont été mixées ensemble pour créer un paquet, ce qui permet de donner une indication correcte du locuteur au receveur.

5.2 Multiplexage des sessions RTP

Pour un traitement efficace du protocole, le nombre de points de multiplexage devrait être minimisé, comme décrit dans le principe de conception de traitement de couche intégré en [10]. Dans RTP, le multiplexage est fourni par l'adresse de transport de destination (adresse réseau et numéro d'accès) qui est différente pour chaque session RTP. Par exemple, dans une téléconférence composée de supports audio et vidéo codés séparément, chaque support DEVRAIT être porté dans une session RTP distincte avec sa propre adresse de transport de destination. Des flux audio et vidéo séparés NE DEVRAIENT PAS être portés dans une seule session RTP et démultiplexés sur la base des champs Type de charge utile ou SSRC. L'entrelaçage de paquets avec des types de support RTP différents mais utilisant le même SSRC introduirait plusieurs problèmes :

1. Si, par exemple, deux flux audio partagent la même session RTP et la même valeur de SSRC, et que l'un change de codage et donc acquière un numéro de séquence RTP différent, il n'y aurait aucun moyen général d'identifier lequel des

flux a changé de codage.

2. Un SSRC est défini comme identifiant un seul espace de temporisation et de numéro de séquence. L'entrelaçage de plusieurs types de charge utile exigerait des espaces de temporisation différents si les débits des horloges de support diffèrent et exigerait des espaces de numéros de séquence différents pour dire quel type de charge utile subit une perte de paquet.

3. Les rapports d'envoi et de réception RTCP (voir au paragraphe 6.4) ne peuvent décrire qu'un seul espace de temporisation et de numéro de séquence par SSRC et ne portent pas de champ Type de charge utile.

4. Un mixeur RTP ne serait pas capable de combiner des flux entrelacés de supports incompatibles en un seul flux.

5. Porter plusieurs supports dans une seule session RTP empêche : l'utilisation de chemins de réseau différents ou l'allocation de ressources réseau différentes si c'est approprié ; la réception d'un sous-ensemble du support si c'est souhaité, par exemple juste l'audio si la vidéo excèderait la bande passante disponible ; et des mises en œuvre de receveur qui utilisent des processus différents pour les différents supports, alors que l'utilisation de sessions RTP séparées permet des mises en œuvre à traitement aussi bien unique que multiple.

Utiliser un SSRC différent pour chaque support mais les envoyer dans la même session RTP éviterait les trois premiers problèmes mais pas les deux derniers.

D'un autre côté, le multiplexage de plusieurs sources se rapportant au même support dans une session RTP utilisant des valeurs de SSRC différentes est la norme pour les sessions en diffusion groupée. La liste des problèmes ci-dessus ne s'applique pas : un mixeur RTP peut, par exemple, combiner plusieurs sources audio, et le même traitement est applicable pour toutes. Il peut aussi être approprié de multiplexer les flux du même support en utilisant des valeurs de SSRC différentes dans d'autres scénarios où les deux derniers problèmes ne s'appliquent pas.

5.3 Modifications spécifiques du profil à l'en-tête RTP

L'en-tête de paquet de données RTP existant est supposé être complet pour l'ensemble des fonctions communes exigées à travers toutes les classes d'application que RTP peut accepter. Cependant, en respectant les principes de conception du verrouillage au niveau couche d'application, l'en-tête PEUT être construit sur mesure à travers les modifications ou ajouts définis dans une spécification de profil tout en permettant encore que fonctionnent des outils de surveillance et d'enregistrement indépendants du profil.

o Le bit marqueur et le champ type de charge utile portent des informations spécifiques du profil, mais ils sont alloués dans l'en-tête fixe dans la mesure où de nombreuses applications sont supposées avoir besoin d'eux et pourraient autrement devoir ajouter un autre mot de 32 bits pour les contenir. L'octet qui contient ces champs PEUT être redéfini par un profil pour répondre à des exigences différentes, par exemple avec plus ou moins de bits marqueurs. Si il y a des bits marqueurs, il DEVRAIT y en avoir un localisé dans le bit de plus fort poids de l'octet car les moniteurs indépendants du profil devraient pouvoir observer une corrélation entre les schémas de perte de paquet et le bit marqueur.

o Les informations supplémentaires qui sont requises pour un format de charge utile particulier, comme un codage vidéo, DEVRAIENT être portées dans la section charge utile du paquet. Cela peut être dans un en-tête qui est toujours présent au début de la section charge utile, ou peut être indiqué par une valeur réservée dans le gabarit des données.

o Si une classe d'applications particulière a besoin de fonctionnalités supplémentaires indépendantes du format de charge utile, le profil sous lequel ces applications fonctionnent DEVRAIT définir des champs fixes supplémentaires à suivre immédiatement après le champ SSRC de l'en-tête fixe existant. Ces applications seront capables d'accéder rapidement et directement aux champs additionnels alors que les moniteurs ou enregistreurs indépendants du profil pourront toujours traiter les paquets RTP en interprétant seulement les douze premiers octets.

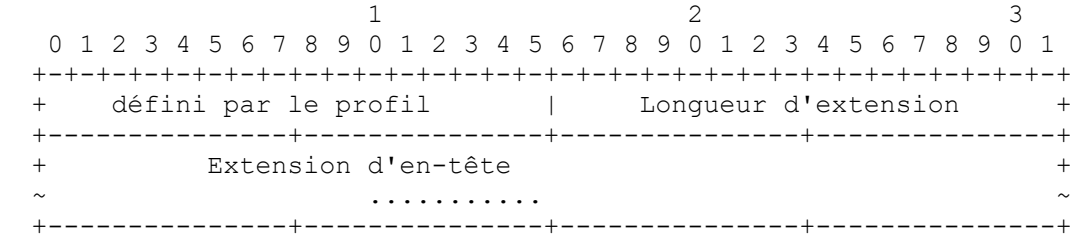
S'il se révélait qu'une fonctionnalité commune supplémentaire est nécessaire à travers tous les profils, une nouvelle version de RTP devrait être définie pour effectuer une modification permanente de l'en-tête fixe.

5.3.1 Extension d'en-tête RTP

Un mécanisme d'extension est fourni pour permettre aux mises en œuvre individuelles d'expérimenter de nouvelles fonctions indépendantes du format de charge utile qui exigent que des informations supplémentaires soient portées dans

l'en-tête de paquet de données RTP. Ce mécanisme est conçu de telle sorte que l'extension d'en-tête puisse être ignorée lors d'interopérations par les autres mises en œuvre qui n'ont pas reçu l'extension.

Noter que cette extension d'en-tête n'est destinée qu'à une utilisation limitée. La plupart des utilisations potentielles de ce mécanisme seraient mieux accomplies par d'autres moyens, en utilisant les méthodes décrites au paragraphe précédent. Par exemple, une extension du profil spécifique à l'en-tête fixe est moins coûteuse à traiter parce qu'elle n'est pas conditionnelle et n'est pas dans une localisation variable. Des informations supplémentaires nécessaires pour un format de charge utile particulier NE DEVRAIENT PAS utiliser cette extension d'en-tête, mais DEVRAIENT être portées dans la section charge utile du paquet.



Si le bit X dans l'en-tête RTP est à un, une extension d'en-tête de longueur variable DOIT être ajoutée à l'en-tête RTP, à la suite de la liste de CSRC si elle est présente. L'extension d'en-tête contient un champ de longueur de 16 bits qui compte le nombre de mots de 32 bits dans l'extension, à l'exclusion de l'extension d'en-tête de quatre octets (donc zéro est une longueur valide). Une seule extension peut être ajoutée à l'en-tête de données RTP. Pour permettre à plusieurs mises en œuvre interopérantes d'expérimenter chacune de façon indépendante des extensions d'en-tête différentes, ou pour permettre à une mise en œuvre particulière d'expérimenter plus d'un type d'extension d'en-tête, les 16 premiers bits de l'extension d'en-tête sont laissés ouverts pour distinguer les identifiants ou les paramètres. Le format de ces 16 bits est à définir par la spécification de profil avec lequel fonctionnent ces mises en œuvre. La présente spécification de RTP ne définit par elle-même aucune extension d'en-tête.

6. Protocole de contrôle RTP -- RTCP

Le protocole de contrôle RTP (RTCP) se fonde sur la transmission périodique de paquets de contrôle à tous les participants à la session, en utilisant le même mécanisme de distribution que celui des paquets de données. Le protocole sous-jacent DOIT fournir le multiplexage des données et des paquets de contrôle, par exemple en utilisant des numéros d'accès distincts pour UDP. RTCP effectue quatre fonctions :

1. La fonction principale est de fournir une rétroaction sur la qualité de la distribution des données. Ceci fait partie intégrante du rôle de RTP comme protocole de transport et se rapporte aux fonctions de contrôle de flux et d'encombrement des autres protocoles de transport (voir la Section 10 sur les exigences pour le contrôle de l'encombrement). La rétroaction peut être directement utile pour le contrôle des codages adaptatifs [18,19], mais les expériences de diffusion groupée IP ont montré qu'elle est aussi critique pour avoir un retour de la part des receveurs pour le diagnostic des fautes dans la distribution. L'envoi de rapports de retour de réception à tous les participants permet à celui qui observe les problèmes d'évaluer si ils sont locaux ou globaux. Avec un mécanisme de distribution comme la diffusion groupée IP, il est aussi possible à une entité comme un fournisseur de service réseau qui n'est pas autrement impliqué dans la session de recevoir des informations de retour et d'agir comme un surveillant tiers pour diagnostiquer les problèmes du réseau. Cette fonction de rétroaction est effectuée par les rapports d'envoi et de réception RTCP, décrits ci-dessous au paragraphe 6.4.

2. RTCP porte un identifiant persistent de niveau transport d'une source RTP appelée le nom canonique ou CNAME, paragraphe 6.5.1. Comme l'identifiant de SSRC peut changer si un conflit est découvert ou si un programme est redémarré, les receveurs demandent au CNAME de garder trace de chaque participant. Les receveurs peuvent aussi demander au CNAME d'associer plusieurs flux de données provenant d'un participant donné dans un ensemble de sessions RTP reliées, par exemple pour synchroniser l'audio et la vidéo. La synchronisation inter-supports exige aussi les horodatages NTP et RTP inclus dans les paquets RTCP par les envoyeurs de données.

3. Les deux premières fonctions exigent que tous les participants envoient des paquets RTCP, et donc le débit doit être contrôlé afin que RTP s'adapte à un grand nombre de participants. En ayant chaque participant qui envoie ses paquets de contrôle à tous les autres, chacun peut observer de façon indépendante le nombre de participants. Ce nombre est utilisé pour calculer le taux d'envoi des paquets, comme expliqué au paragraphe 6.2.

4. Une quatrième fonction, FACULTATIVE, est de convoyer des informations minimales de contrôle de session, par exemple l'affichage de l'identification de participant à l'interface d'utilisateur. Ceci est très vraisemblablement utile dans les sessions "à contrôle lâche" où les participants entrent et sortent sans contrôle d'adhésion ou négociation de paramètre. RTCP sert de canal pratique pour atteindre tous les participants, mais il n'en est pas nécessairement attendu qu'il prenne en charge toutes les exigences de contrôle de communication d'une application. Un protocole de contrôle de session de niveau supérieur, qui est en dehors du domaine d'application du présent document, peut être nécessaire.

Les fonctions 1 à 3 DEVRAIENT être utilisées dans tous les environnements, mais particulièrement dans un environnement de diffusion groupée IP. Les concepteurs d'applications RTP DEVRAIENT éviter des mécanismes qui ne peuvent fonctionner qu'en mode d'envoi individuel et ne s'adaptent pas aux grands nombres. La transmission de RTCP PEUT être contrôlée séparément pour les envoyeurs et les receveurs, comme décrit au paragraphe 6.2, pour des cas tels que ceux de liaisons unidirectionnelles où la rétroaction des receveurs n'est pas possible.

Note non normative : Dans l'approche d'acheminement de diffusion groupée appelée Diffusion groupée spécifique de source (SSM, *Source-Specific Multicast*), il y a seulement un envoyeur par "canal" (une paire adresse de source, adresse de groupe), et les receveurs (excepté le canal source) ne peuvent pas utiliser la diffusion groupée pour communiquer directement avec les autres canaux membres. Les recommandations présentes ne traitent de SSM qu'à travers l'option du paragraphe 6.2 de déconnexion complète des receveurs de RTCP. Des travaux à venir spécifieront l'adaptation de RTCP à SSM de façon que la rétroaction des receveurs puisse être conservée.

6.1 Format de paquet RTCP

La présente spécification définit plusieurs types de paquet RTCP à porter sur diverses informations de contrôle :

SR : Rapport d'envoi, pour les statistiques de transmission et réception provenant des participants qui sont des envoyeurs actifs.

RR : Rapports de réception, pour les statistiques de réception provenant des participants qui ne sont pas des envoyeurs actifs, et en combinaison avec les SR pour les envoyeurs actifs qui font rapport sur plus de 31 sources.

SDES : Élément de description de la source, y compris le CNAME.

BYE : Indique la fin de la participation.

APP : Fonctions spécifiques de l'application.

Chaque paquet RTCP commence par une partie fixe similaire à celle des paquets de données RTP, suivie par des éléments structurés qui PEUVENT être de longueur variable selon le type de paquet mais DOIVENT se terminer sur une limite de 32 bits. L'exigence d'alignement et d'un champ longueur dans la partie fixe de chaque paquet est incluse pour permettre la "mise en pile" des paquets RTCP. Plusieurs paquets RTCP peuvent être enchaînés sans qu'aucun séparateur n'intervienne pour former un paquet RTCP composé qui est envoyé dans un seul paquet au protocole de couche inférieure, par exemple UDP. Il n'y a pas de compte explicite des paquets RTCP individuels dans le paquet composé car les protocoles de couche inférieure sont supposés fournir la longueur totale pour déterminer la fin du paquet composé.

Chaque paquet RTCP individuel du paquet composé peut être traité indépendamment sans exigences sur l'ordre de combinaison des paquets. Cependant, afin d'effectuer les fonctions du protocole, les contraintes suivantes sont imposées :

- Les statistiques de réception (en SR ou RR) devraient être envoyées aussi souvent que les contraintes de bande passante le permettront afin de maximiser la résolution des statistiques, et donc chaque paquet RTCP composé transmis périodiquement DOIT comporter un paquet de rapport.

- Les nouveaux receveurs ont besoin de recevoir le CNAME d'une source aussitôt que possible pour identifier la source et commencer à associer les supports pour des besoins tels que lip-sync, aussi chaque paquet RTCP composé DOIT aussi inclure le CNAME de SDES sauf quand le paquet RTCP composé est partagé pour un chiffrement partiel comme décrit au paragraphe 9.1.

- Le nombre de types de paquets qui peuvent apparaître en premier dans le paquet composé doit être limité pour augmenter le nombre de bits constants dans le premier mot et la probabilité d'une validation réussie des paquets RTCP par rapport à des paquets de données RTP mal adressés ou autres paquets sans rapport.

Et donc, tous les paquets RTCP DOIVENT être envoyés dans un paquet composé d'au moins deux paquets individuels, avec le format suivant :

Préfixe de chiffrement : Si et seulement si le paquet composé est à chiffrer conformément à la méthode du paragraphe 9.1, il DOIT être préfixé d'une quantité aléatoire de 32 bits renouvelée pour chaque paquet composé transmis. Si du bourrage est nécessaire pour le chiffrement, il DOIT être ajouté au dernier paquet du paquet composé.

SR ou RR : Le premier paquet RTCP du paquet composé DOIT toujours être un paquet de rapport pour faciliter la validation d'en-tête, comme décrit à l'Appendice A.2. Ceci est vrai même si aucune donnée n'est envoyée ou reçue, auquel cas, un RR vide DOIT être envoyé, et même si le seul autre paquet RTCP dans le paquet composé est un BYE.

RR supplémentaires : Si le nombre de sources pour lesquelles les statistiques de réception sont rapportées excède 31, le nombre qui va tenir dans un paquet SR ou RR, des paquets RR supplémentaires DEVRAIENT suivre le paquet de rapport initial.

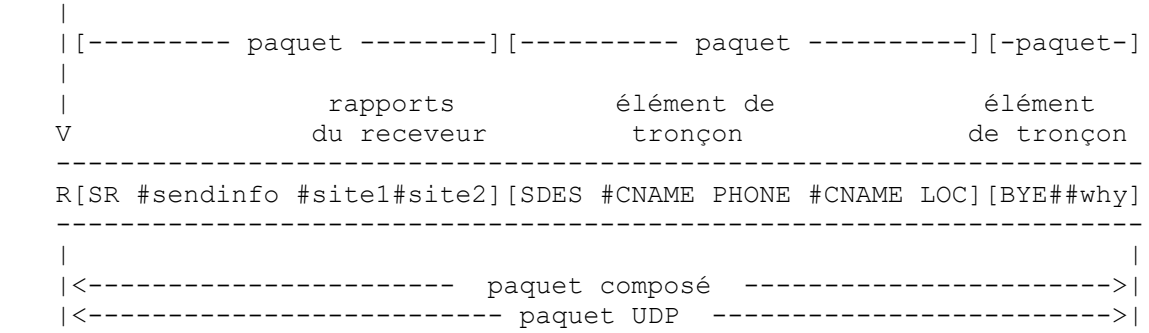
SDES : Un paquet SDES contenant un élément CNAME DOIT être inclus dans chaque paquet RTCP composé, excepté comme noté au paragraphe 9.1. D'autres éléments de description de source PEUVENT facultativement être inclus si c'est exigé par une application particulière, sous réserve des contraintes de bande passante (voir au paragraphe 6.3.9).

BYE ou APP : D'autres types de paquet RTCP, y compris ceux déjà définis, PEUVENT suivre dans un ordre quelconque, sauf que BYE DEVRAIT être le dernier paquet envoyé avec un SSRC/CSRC donné. Les types de paquet PEUVENT apparaître plus d'une fois.

Un participant RTP individuel DEVRAIT envoyer seulement un paquet RTCP composé par intervalle de rapport afin que la bande passante RTCP par participant soit estimée correctement (voir au paragraphe 6.2), excepté lorsque le paquet RTCP composé est partagé pour chiffrement partiel, comme décrit au paragraphe 9.1. Si il y a trop de sources pour faire tenir tous les paquets RR nécessaires dans un seul paquet RTCP composé sans excéder l'unité de transmission maximum (MTU, *maximum transmission unit*) du chemin de réseau, alors seul le sous-ensemble qui tient dans une MTU DEVRAIT être inclus dans chaque intervalle. Les sous-ensembles DEVRAIENT être sélectionnés par un round-robin à travers plusieurs intervalles de façon que toutes les sources soient rapportées.

Il est RECOMMANDÉ que les traducteurs et mixeurs combinent les paquets RTCP individuels à partir des multiples sources qu'ils transmettent en un paquet composé chaque fois que c'est faisable afin d'amortir la redondance de paquet (voir la Section 7). Un exemple de paquet RTCP composé qui pourrait être produit par un mixeur est indiqué à la Figure 1. Si la longueur totale d'un paquet composé devait dépasser la MTU du chemin de réseau, il DEVRAIT être segmenté en plusieurs paquets composés plus courts à transmettre dans des paquets séparés du protocole sous-jacent. Cela n'a pas d'impact sur l'estimation de bande passante RTCP parce que chaque paquet composé représente au moins un participant distinct. Noter que chaque paquet composé DOIT commencer par un paquet SR ou RR.

Une mise en œuvre DEVRAIT ignorer les paquets RTCP entrants avec des types qu'il ne connaît pas. Des types de paquet RTCP supplémentaires peuvent être enregistrés auprès de l'Autorité d'allocation des numéros de l'Internet (IANA) comme décrit à la Section 15.



: identifiant de SSRC/CSRC

Figure 1 : Exemple d'un paquet composé RTCP

6.2 Intervalle de transmission RTCP

RTP est conçu pour permettre à une application de s'adapter automatiquement à des tailles de session allant de quelques participants à plusieurs milliers. Par exemple, dans une conférence audio, le trafic de données est par nature auto limitant parce qu'à chaque instant, il n'y aura qu'une ou deux personnes à parler, aussi, avec la distribution en diffusion groupée, le débit de données sur une liaison donnée reste relativement constant indépendamment du nombre des participants. Cependant, le trafic de contrôle n'est pas auto limitant. Si les rapports de réception provenant de chaque participant étaient envoyés à un débit constant, le trafic de contrôle croîtrait de façon linéaire avec le nombre des participants. Donc, le débit doit être diminué de façon dynamique en calculant les intervalles entre les transmissions de paquets RTCP.

Pour chaque session, on suppose que le trafic de données est soumis à une limite agrégée appelée la "bande passante de session" à diviser entre les participants. Cette bande passante peut être réservée et la limite mise en application par le réseau. Si il n'y avait pas de réservation, il pourrait y avoir d'autres contraintes, en fonction de l'environnement, pour établir le maximum "raisonnable" à utiliser par la session, et qui serait la bande passante de session. La bande passante de session peut être choisie sur la base d'un coût ou de la connaissance à priori de la bande passante disponible du réseau pour la session. Elle est assez indépendante du codage du support, mais le choix du codage peut être limité par la bande passante de la session. Souvent, la bande passante de session est la somme des bandes passantes nominales des envoyeurs dont on s'attend à ce qu'ils soient actifs en même temps. Pour la téléconférence audio, ce nombre devrait normalement être la bande passante d'un seul envoyeur. Pour les codages mis en couches, chaque couche est une session RTP distincte avec son propre paramètre de bande passante de session.

Le paramètre bande passante de session est censé être fourni par une application de gestion de session lorsqu'elle invoque une application de support, mais les applications de support PEUVENT établir une valeur par défaut sur la base de la bande passante de données d'un seul envoyeur pour le codage choisi pour la session. L'application PEUT aussi appliquer des limites de bande passante sur la base de règles de portée de la diffusion groupé ou d'autres critères. Tous les participants DOIVENT utiliser la même valeur de bande passante de session afin que soit calculé le même intervalle RTCP .

Les calculs de bande passante pour le trafic de contrôle et de données incluent des protocoles de transport et de réseau de couche inférieure (par exemple, UDP et IP) car c'est ce que le système de réservation de ressource aura besoin de savoir. L'application peut être aussi supposée savoir lesquels de ces protocoles sont utilisés. Les en-têtes de niveau liaison ne sont pas inclus dans le calcul car le paquet sera encapsulé avec différents en-têtes de niveau liaison à mesure qu'il voyage.

Le trafic de contrôle devrait être limité à une petite fraction connue de la bande passante de session : assez petite pour que la fonction principale du protocole de transport pour porter les données ne soit pas entravée ; connue de telle sorte que le trafic de contrôle soit inclus dans la spécification de bande passante donnée pour un protocole de réservation de ressources, et que chaque participant puisse calculer sa part indépendamment. La bande passante du trafic de contrôle est en plus de la bande passante de session pour le trafic de données. Il est RECOMMANDÉ que la fraction de la bande passante de session ajoutée pour RTCP soit fixée à 5 %. Il est aussi RECOMMANDÉ que 1/4 de la bande passante RTCP soit dédiée aux participants qui envoient des données afin que dans les sessions qui ont un grand nombre de receveurs mais un petit nombre d'envoyeurs, les nouveaux participants qui la rejoignent reçoivent plus rapidement le CNAME pour les sites d'envoi. Lorsque la proportion d'envoyeurs est supérieure à 1/4 des participants, les envoyeurs reçoivent leur proportion de la bande passante RTCP complète. Bien que les valeurs de ces constantes et des autres dans le calcul d'intervalle ne soient pas critiques, tous les participants à la session DOIVENT utiliser les mêmes valeurs afin que le même intervalle soit calculé. Donc, ces constantes DEVRAIENT être fixées pour un profil particulier.

Un profil PEUT spécifier que la bande passante du trafic de contrôle peut être un paramètre séparé de la session plutôt qu'un strict pourcentage de la bande passante de session. Utiliser un paramètre distinct permet des applications qui s'adaptent à un débit de bande passante RTCP cohérent avec une bande passante de données "normale" qui est inférieure à la bande passante maximum spécifiée par le paramètre de bande passante de session.

Le profil PEUT spécifier de plus que la bande passante du trafic de contrôle peut être divisée en deux paramètres de session distincts entre les participants qui sont des envoyeurs actifs de données et ceux qui ne le sont pas ; appelons ces paramètres S et R. Suivant la recommandation que 1/4 de la bande passante RTCP soit dédiée aux envoyeurs de données, les valeurs RECOMMANDÉES par défaut pour ces deux paramètres seraient respectivement 1,25 % et 3,75 %. Lorsque la proportion d'envoyeurs est supérieure à S/(S+R) des participants, les envoyeurs reçoivent leur proportion de la somme de ces paramètres. Utiliser deux paramètres permet que les rapports de réception RTCP soient entièrement éliminés pour une session particulière en réglant à zéro la bande passante RTCP pour les non envoyeurs de données tout en conservant la bande passante RTCP pour les envoyeurs de données non zéro de sorte que les rapports d'envoyeurs puissent toujours être envoyés pour la synchronisation inter supports. L'élimination des rapports de réception RTCP N'EST PAS RECOMMANDÉE parce qu'ils sont nécessaires pour les fonctions énumérées au début de la Section 6, et en particulier le

retour sur la qualité de réception et le contrôle d'encombrement. Cependant, il peut être approprié de faire ainsi pour les systèmes qui fonctionnent sur des liaisons unidirectionnelles ou pour les sessions qui n'ont pas besoin de retour sur la qualité de réception ou sur la vitalité des receveurs et ont d'autres moyens d'éviter l'encombrement.

L'intervalle calculé entre les transmissions des paquets RTCP composés DEVRAIT aussi avoir une limite inférieure pour éviter d'avoir des salves de paquets qui excèdent la bande passante admise lorsque le nombre de participants est petit et que le trafic n'est pas lissé conformément à la loi des grands nombres. Il empêche aussi l'intervalle de rapport de devenir trop petit durant les coupures transitoires du genre partition de réseau de telle sorte que l'adaptation soit retardée lorsque la partition se répare. Au démarrage de l'application, un délai DEVRAIT être imposé avant l'envoi du premier paquet RTCP composé pour laisser aux paquets RTCP le temps d'être reçus de la part des autres participants afin que l'intervalle de rapport converge plus rapidement vers la valeur correcte. Ce délai PEUT être réglé à la moitié de l'intervalle minimum pour permettre une notification plus rapide de la présence du nouveau participant. La valeur RECOMMANDÉE pour un intervalle fixe minimum est de 5 secondes.

Une mise en œuvre PEUT régler l'intervalle RTCP minimum à une valeur plus petite inversement proportionnelle au paramètre de bande passante de session avec les limitations suivantes :

- Pour les sessions en diffusion groupée, seuls les envoyeurs de données actifs PEUVENT utiliser la valeur réduite minimum pour calculer l'intervalle pour la transmission des paquets RTCP composés.

- Pour les sessions en envoi individuel, la valeur réduite PEUT être aussi bien utilisée par les participants qui ne sont pas des envoyeurs de données actifs, et le délai avant l'envoi du paquet RTCP composé initial PEUT être zéro.

- Pour toutes les sessions, le minimum fixé DEVRAIT être utilisé lors du calcul de la temporisation de l'intervalle du participant (voir au paragraphe 6.3.5) de sorte que les mises en œuvre qui n'utilisent pas la valeur réduite pour la transmission des paquets RTCP ne soient pas prématurément périmées par les autres participants.

- La valeur RECOMMANDÉE en secondes pour le minimum réduit est de 360 divisé par la bande passante de session en kbit/s. Ce minimum est inférieur à 5 s pour les bandes passantes supérieures à 72 kbit/s.

L'algorithme décrit au paragraphe 6.3 et à l'Appendice A.7 a été conçu pour satisfaire les objectifs exposés dans cette section. Il calcule l'intervalle entre l'envoi des paquets RTCP composés pour diviser la bande passante de trafic de contrôle admise entre les participants. Cela permet à une application de fournir une réponse rapide aux petites sessions où, par exemple, l'identification de tous les participants est importante, tout en s'adaptant automatiquement aux grandes sessions. L'algorithme incorpore les caractéristiques suivantes :

- L'intervalle calculé entre les paquets RTCP est en proportion linéaire du nombre des membres du groupe. C'est ce facteur linéaire qui permet une somme constante de la quantité de trafic de contrôle entre tous les membres.

- L'intervalle entre les paquets RTCP varie de façon aléatoire sur la gamme [0,5 – 1,5] multipliée par l'intervalle calculé pour éviter la synchronisation involontaire de tous les participants [20]. Le premier paquet RTCP envoyé après avoir rejoint une session est aussi retardé d'une variation aléatoire de la moitié de l'intervalle RTCP minimum.

- Une estimation dynamique de la taille moyenne du paquet RTCP composé est calculée, incluant tous les paquets reçus et envoyés, pour s'adapter automatiquement aux changements de quantité des informations de contrôle portées.

- Comme l'intervalle calculé dépend du nombre des membres du groupe observé, il peut y avoir des effets de démarrage indésirables lorsque un nouvel utilisateur se joint à une session existante, ou lorsque de nombreux usagers se joignent simultanément à une nouvelle session. Ces nouveaux usagers vont initialement avoir des estimations incorrectes du nombre des membres du groupe, et donc leur intervalle de transmission RTCP sera trop court. Ce problème peut être significatif si de nombreux usagers se joignent simultanément à la session. Pour traiter cela, on utilise un algorithme appelé de "reconsidération de temporisation". Cet algorithme met en œuvre un mécanisme simple de sauvegarde qui cause la rétention de la transmission des paquets RTCP par les utilisateurs lorsque les tailles de groupe augmentent.

- Lorsque les usagers quittent une session, avec un BYE ou par l'expiration d'une temporisation, la taille du groupe diminue, et donc l'intervalle calculé devrait diminuer. Un algorithme de "reconsidération inverse" est utilisé pour permettre aux membres de réduire plus rapidement leurs intervalles en réponse à une diminution de taille du groupe.

- Les paquets BYE reçoivent un traitement différent des autres paquets RTCP. Lorsque un usager quitte un groupe, et souhaite envoyer un paquet BYE, il peut le faire avant son prochain paquet RTCP prévu. Cependant, la transmission des

BYE suit un algorithme de sauvegarde qui évite que des flots de paquets BYE soient envoyés si un grand nombre de membres quittent simultanément la session.

Cet algorithme peut être utilisé pour les sessions dans lesquelles tous les participants ont la permission d'envoyer. Dans ce cas, le paramètre de bande passante de session est le produit de la bande passante individuelle d'envoyeur multipliée par le nombre des participants, et la bande passante RTCP est de 5 % de cela.

Les détails du fonctionnement de cet algorithme sont donnés dans les paragraphes qui suivent. L'Appendice A.7 donne un exemple de mise en œuvre.

6.2.1 Maintenance du nombre des membres de la session

Le calcul de l'intervalle du paquet RTCP dépend d'une estimation du nombre de sites qui participent à la session. De nouveaux sites sont ajoutés au compte lorsqu'ils sont entendus, et une entrée pour chacun DEVRAIT être créée dans un tableau indexé par identifiant de SSRC ou CSRC (voir au paragraphe 8.2) pour garder leur trace. De nouvelles entrées PEUVENT être considérées comme non valides jusqu'à ce que plusieurs paquets portant le nouveau SSRC aient été reçus (voir l'Appendice A.1), ou jusqu'à ce qu'un paquet SDES RTCP contenant un CNAME pour ce SSRC ait été reçu. Les entrées PEUVENT être supprimées du tableau lorsque un paquet BYE RTCP est reçu avec l'identifiant de SSRC correspondant, sauf que des paquets de données en retard peuvent arriver après le BYE et causer la recréation de l'entrée. Au lieu de cela, l'entrée DEVRAIT être marquée comme ayant reçu un BYE puis supprimée après un délai approprié.

Un participant PEUT marquer un autre site inactif, ou le supprimer si il n'est pas encore valide, si aucun paquet RTP ou RTCP n'a été reçu pour un petit nombre d'intervalles de rapport RTCP (5 est RECOMMANDÉ). Cela procure une certaine robustesse contre la perte de paquets. Tous les sites doivent avoir la même valeur pour ce multiplicateur et doivent calculer en gros la même valeur pour l'intervalle de rapport RTCP afin que cette temporisation fonctionne correctement. Donc, ce multiplicateur DEVRAIT être fixé pour un profil particulier.

Pour les sessions avec un très grand nombre de participants, il peut être impraticable d'entretenir un tableau pour mémoriser l'identifiant de SSRC et les informations d'état pour chacun d'eux. Une mise en œuvre PEUT utiliser l'échantillonnage de SSRC, comme décrit dans [21], pour réduire les exigences de mémorisation. Une mise en œuvre PEUT utiliser tout autre algorithme ayant des performances similaires. Une exigence clé est que tout algorithme considéré NE DEVRAIT PAS sous-estimer de façon substantielle la taille de groupe, alors qu'il PEUT la surestimer.

6.3 Règles d'envoi et de réception du paquet RTCP

On précise ici les règles d'envoi d'un paquet RTCP, et ce qu'il convient de faire lorsque on en reçoit un. Une mise en œuvre qui permet le fonctionnement dans un environnement de diffusion groupée ou dans un environnement d'envoi individuel en multipoint DOIT satisfaire aux exigences du paragraphe 6.2. Une telle mise en œuvre PEUT utiliser l'algorithme défini dans la présente section pour satisfaire à ces exigences, ou PEUT utiliser certains autres algorithmes pour autant qu'ils fournissent des performances équivalentes ou meilleures. Une mise en œuvre qui est restreinte au fonctionnement en envoi individuel entre deux parties DEVRAIT quand même utiliser l'intervalle de transmission TTCP aléatoire pour éviter la synchronisation involontaire de plusieurs instances fonctionnant dans le même environnement, mais PEUT omettre les algorithmes de "reconsidération de temporisateur" et de "reconsidération inverse" dans les paragraphes 6.3.3, 6.3.6 et 6.3.7.

Pour exécuter ces règles, un participant à une session peut entretenir plusieurs éléments d'état :

tp : la dernière fois qu'un paquet RTCP a été transmis ;

tc : l'heure en cours ;

tn : l'heure de transmission prévue du prochain paquet RTCP ;

pmembers : le nombre estimé de membres de la session au moment où tn a été recalculé pour la dernière fois ;

members : l'estimation la plus courante du nombre des membres de la session ;

senders : l'estimation la plus courante du nombre des envoyeurs dans la session ;

rtcp_bw : la bande passante RTCP de la cible, c'est-à-dire, la bande passante totale qui sera utilisée pour les paquets RTCP

par tous les membres de cette session, en octets par seconde. Ce sera une fraction spécifiée du paramètre "bande passante de session" fourni à l'application au démarrage.

we_sent : fanion qui est vrai si l'application a envoyé des données depuis la transmission du deuxième rapport RTCP précédent.

avg_rtcp_size : taille moyenne du paquet RTCP composé, en octets, sur tous les paquets RTCP envoyés et reçus par ce participant. La taille inclut les en-têtes de protocole de transport et de réseau de couche inférieure (par exemple, UDP et IP) comme expliqué au paragraphe 6.2.

initial : fanion qui est vrai si l'application n'a pas encore envoyé de paquet RTCP.

Beaucoup de ces règles peuvent utiliser "l'intervalle calculé" entre les transmissions de paquet. Cet intervalle est décrit dans le paragraphe suivant.

6.3.1 Calcul de l'intervalle de transmission RTCP

Pour conserver la proportionnalité, l'intervalle moyen entre les paquets provenant d'un participant à une session devrait rester proportionnel à la taille du groupe. Cet intervalle est appelé l'intervalle calculé. Il est obtenu en combinant un certain nombre des éléments d'état décrits ci-dessus. L'intervalle calculé T est alors déterminé comme suit :

1. Si le nombre des envoyeurs est inférieur ou égal à 25 % du nombre des membres, l'intervalle dépendra de ce que le participant est un envoyeur ou non (sur la base de la valeur de we_sent). Si le participant est un envoyeur (we_sent est vrai), la constante C est réglée à la taille moyenne de paquet RTCP (avg_rtcp_size) divisée par 25 % de la bande passante RTCP (rtcp_bw), et la constante n est réglée au nombre des envoyeurs. Si we_sent n'est pas vrai, la constante C est réglée à la taille moyenne de paquet RTCP divisée par 75 % de la bande passante RTCP. La constante n est réglée au nombre de receveurs (membres - envoyeurs). Si le nombre d'envoyeurs est supérieur à 25 %, envoyeurs et receveurs sont traités ensemble. La constante C est réglée à la taille moyenne de paquet RTCP divisée par la bande passante RTCP totale et n est réglé au nombre total de membres. Comme déclaré au paragraphe 6.2, un profil RTP PEUT spécifier que la bande passante RTCP peut être explicitement définie par deux paramètres distincts (appelons les S et R) pour les participants qui sont envoyeurs et pour ceux qui ne le sont pas. Dans ce cas, la fraction des 25 % devient S/(S+R) et la fraction des 75 % devient R/(S+R). Noter que si R est zéro, le pourcentage d'envoyeurs n'est jamais supérieur à S/(S+R), et la mise en œuvre doit éviter une division par zéro.

2. Si le participant n'a pas encore envoyé un paquet RTCP (la variable initiale est vraie), la constante Tmin est réglée à 2,5 s, autrement, elle est réglée à 5 s.

3. L'intervalle déterministe Td calculé est réglé à max(Tmin, n*C).

4. L'intervalle T calculé est réglé à un nombre uniformément réparti entre 0,5 et 1,5 fois l'intervalle déterministe calculé.

5. La valeur résultante de T est divisée par e-3/2 = 1,21828 pour compenser le fait que l'algorithme de reconsidération de temporisateur converge vers une valeur de la bande passante RTCP inférieure à la moyenne prévue.

Cette procédure résulte en un intervalle aléatoire, mais qui, en moyenne, donne au moins 25 % de la bande passante RTCP aux envoyeurs et le reste aux receveurs. Si les envoyeurs constituent plus du quart des membres, cette procédure partage également la bande passante entre tous les participants, en moyenne.

6.3.2 Initialisation

En se joignant à la session, le participant initialise tp à 0, tc à 0, les envoyeurs à 0, pmembers à 1, members à 1, we_sent à faux, rtcp_bw à la fraction spécifiée de la bande passante de session, initial à vrai, et avg_rtcp_size à la taille probable du premier paquet RTCP que l'application va construire ultérieurement. L'intervalle calculé T est alors calculé, et le premier paquet est prévu pour l'heure tn = T. Cela signifie qu'un temporisateur de transmission est réglé pour arriver à expiration à l'heure T. Noter qu'une application PEUT utiliser toute approche désirée pour mettre en œuvre ce temporisateur.

Le participant ajoute son propre SSRC à son tableau des membres.

6.3.3 Réception d'un paquet RTP ou RTCP non BYE

Lorsque un paquet RTP ou RTCP est reçu d'un participant dont le SSRC n'est pas dans le tableau des membres, le SSRC est ajouté au tableau, et la valeur pour les membres est mise à jour une fois que le participant a été validé comme décrit au paragraphe 6.2.1. Le même traitement survient pour chaque CSRC dans un paquet RTP validé.

Lorsque un paquet RTP est reçu d'un participant dont le SSRC n'est pas dans le tableau des envoyeurs, le SSRC est ajouté au tableau, et la valeur pour les envoyeurs est mise à jour.

Pour chaque paquet RTCP composé reçu, la valeur de avg_rtcp_size est mise à jour :

avg_rtcp_size = (1/16) * packet_size + (15/16) * avg_rtcp_size

où packet_size est la taille du paquet RTCP qui vient d'être reçu.

6.3.4 Réception d'un paquet BYE RTCP

Sauf comme décrit au paragraphe 6.3.7 pour le cas où un paquet BYE RTCP doit être transmis, si le paquet reçu est un paquet BYE RTCP, le SSRC est vérifié par rapport au tableau des membres. S'il est présent, l'entrée est retirée du tableau, et la valeur pour les membres est mise à jour. Le SSRC est alors vérifié par rapport au tableau des envoyeurs. S'il est présent, l'entrée est retirée du tableau, et la valeur pour les envoyeurs est mise à jour.

De plus, pour rendre le taux de transmission des paquets RTCP plus réactif aux changements des membres du groupe, l'algorithme de "reconsidération inverse" suivant DEVRAIT être exécuté lorsque un paquet BYE est reçu qui réduit les membres à une valeur inférieure à pmembers :

o La valeur de tn est mise à jour conformément à la formule suivante :

tn = tc + (members/pmembers) * (tn - tc)

o La valeur de tp est mise à jour conformément à la formule suivante :

tp = tc - (members/pmembers) * (tc - tp).

o Le paquet RTCP suivant est reprogrammé pour une transmission à l'heure tn, qui est maintenant plus tôt.

o La valeur de pmembers est mise égale aux membres.

Cet algorithme n'empêche pas l'estimation de la taille du groupe de tomber incorrectement à zéro pour une brève période à cause de fins de temporisations prématurées lorsque la plupart des participants à une grosse session la quittent en même temps, mais que quelques uns restent. L'algorithme fait bien revenir l'estimation à la valeur correcte plus rapidement. Cette situation est assez inhabituelle et les conséquences en sont suffisamment bénignes pour que ce problème soit tenu pour négligeable.

6.3.5 Expiration de la temporisation d'un SSRC

À des intervalles occasionnels, le participant DOIT vérifier pour voir si un des autres participants arrive en fin de temporisation. Pour ce faire, le participant calcule l'intervalle calculé déterministe (sans le facteur aléatoire) Td pour un receveur, c'est-à-dire, avec we_sent faux. Tout autre membre de la session qui n'a pas envoyé un paquet RTP ou RTCP depuis le temps tc - MTd (M est le multiplicateur de temporisation, et prend la valeur 5 par défaut) est périmé. Cela signifie que son SSRC est retiré de la liste des membres, et members est mis à jour. Une vérification similaire est effectuée sur la liste des envoyeurs. Tout membre qui est sur la liste des envoyeurs et n'a pas envoyé de paquet RTP depuis le temps tc - 2T (pendant les deux derniers intervalles de rapport RTCP) est retiré de la liste des envoyeurs, et senders est mis à jour.

Si un membre est périmé, l'algorithme de reconsidération inverse décrit au paragraphe 6.3.4 DEVRAIT être effectué.

Le participant DOIT effectuer cette vérification au moins une fois par intervalle de transmission RTCP.

6.3.6 Expiration du temporisateur de transmission

Lorsque le temporisateur de transmission de paquet arrive à expiration, le participant effectue les opérations suivantes :

- L'intervalle de transmission T est calculé comme décrit au paragraphe 6.3.1, y compris le facteur aléatoire.

- Si tp + T est inférieur ou égal à tc, un paquet RTCP est transmis. tp est réglé à tc, puis une autre valeur est calculée pour T comme à l'étape précédente et tn est réglé à tc + T. Le temporisateur de transmission est réglé pour arriver à expiration à nouveau au moment tn. Si tp + T est supérieur à tc, tn est réglé à tp + T. Aucun paquet RTCP n'est transmis. Le temporisateur de transmission est réglé pour arriver à expiration au moment tn.

- pmembers est réglé à members.

Si un paquet RTCP est transmis, la valeur de initial est réglée à FAUX. De plus, la valeur de avg_rtcp_size est mise à jour :

avg_rtcp_size = (1/16) * packet_size + (15/16) * avg_rtcp_size

où packet_size est la taille du paquet RTCP qui vient d'être transmis.

6.3.7 Transmission d'un paquet BYE

Lorsque un participant souhaite quitter une session, un paquet BYE est transmis pour informer les autres participants de l'événement. Afin d'éviter un flot de paquets BYE lorsque de nombreux participants quittent le système, un participant DOIT exécuter l'algorithme suivant si le nombre de membres est supérieur à 50 lorsque le participant choisit de partir. Cet algorithme usurpe le rôle normal de la variable members pour compter à sa place le nombre de paquets BYE:

- Lorsque le participant décide de quitter le système, tp est remis à tc, l'heure actuelle, members et pmembers sont initialisés à 1, initial est mis à 1, we_sent est mis à faux, senders est mis à 0, et avg_rtcp_size est réglé à la taille du paquet composé BYE. L'intervalle calculé T est calculé. Le paquet BYE est alors programmé pour l'instant tn = tc + T.

- Chaque fois qu'un paquet BYE provenant d'un autre participant est reçu, members est incrémenté de 1 sans considération du fait que ce participant existe ou non dans le tableau des membres, et lorsque l'échantillonnage de SSRC est utilisé, sans considération de ce que le SSRC BYE devrait être ou non inclus dans l'échantillon. members N'EST PAS incrémenté lorsque d'autres paquets RTCP ou RTP sont reçus, mais seulement les paquets BYE. De même, avg_rtcp_size n'est mis à jour que pour les paquets BYE reçus. senders N'EST PAS mis à jour lorsque des paquets RTP arrivent ; il reste à 0.

- La transmission du paquet BYE suit alors les règles de transmission d'un paquet RTCP normal, comme ci-dessus.

Cela permet d'envoyer tout de suite les paquets BYE, et donc de contrôler leur utilisation totale de bande passante. Dans le pire des cas, cela pourrait être cause que les paquets de contrôle RTCP utilisent deux fois plus de bande passante que la normale (10 %) – 5 % pour les paquets RTCP non BYE et 5 % pour BYE.

Un participant qui ne veut pas attendre que le mécanisme ci-dessus permette la transmission d'un paquet BYE PEUT quitter le groupe sans envoyer de BYE du tout. Ce participant va finalement être mis en fin de temporisation par les autres membres du groupe.

Si l'estimation de la taille du groupe est inférieure à 50 lorsque le participant décide de le quitter, le participant PEUT envoyer immédiatement un paquet BYE. Autrement, le participant PEUT choisir d'exécuter l'algorithme de sauvegarde BYE ci-dessus.

Dans l'un et l'autre cas, un participant qui n'a jamais envoyé de paquet RTP ou RTCP NE DOIT PAS envoyer un paquet BYE lorsque il quitte le groupe.

6.3.8 Mise à jour de we_sent

La variable we_sent contient vrai si le participant a envoyé récemment un paquet RTP, faux autrement. Cette détermination se fait en utilisant le même mécanisme que pour gérer l'ensemble des autres participants énumérés dans le tableau des envoyeurs. Si le participant envoie un paquet RTP lorsque we_sent est faux, il s'ajoute lui-même au tableau des envoyeurs

et règle we_sent à vrai. L'algorithme de reconsidération inverse décrit au paragraphe 6.3.4 DEVRAIT être effectué pour éventuellement réduire le délai avant l'envoi d'un paquet SR. Chaque fois qu'un autre paquet RTP est envoyé, le temps de transmission de ce paquet est conservé dans le tableau. L'algorithme normal de temporisation d'envoi est alors appliqué au participant – si un paquet RTP n'a pas été transmis depuis tc - 2T, le participant se retire lui-même du tableau des envoyeurs, décrémente le compte des envoyeurs, et remet we_sent à faux.

6.3.9 Allocation de bande passante de description de source

Cette spécification définit plusieurs éléments de description de source (SDES) en plus de l'élément obligatoire CNAME, tels que NAME (nom de la personne) et EMAIL (adresse de messagerie). Elle fournit aussi un moyen pour définir de nouveaux types de paquet RTCP spécifiques de l'application. Les applications devraient faire attention en allouant de la bande passante de contrôle à ces informations supplémentaires parce que cela va ralentir le taux d'envoi des rapports de réception et des CNAME, et donc dégrader les performances du protocole. Il est RECOMMANDÉ que pas plus de 20 % de la bande passante RTCP allouée à un seul participant ne soit utilisée pour porter les informations supplémentaires. De plus, il n'est pas prévu que tous les éléments de SDES soient inclus dans chaque application. Ceux qui sont inclus DEVRAIENT se voir allouer une fraction de la bande passante conformément à leur utilité. Plutôt que d'estimer cette fraction de façon dynamique, il est recommandé que les pourcentages soit traduits de façon statique dans les comptes d'intervalles de rapports sur la base de la longueur normale d'un élément.

Par exemple, une application peut être conçue pour envoyer seulement CNAME, NAME et EMAIL et aucun des autres. NAME peut avoir reçu une bien plus forte priorité que EMAIL parce que le NAME sera affiché en continu dans l'interface d'utilisateur de l'application, alors que EMAIL ne sera affiché que sur demande. À chaque intervalle RTCP, un paquet RR et un paquet SDES avec l'élément CNAME sera envoyé. Pour une petite session fonctionnant à l'intervalle minimum, ce serait toutes les 5 secondes en moyenne. Chaque troisième intervalle (15 secondes), un élément supplémentaire serait inclus dans le paquet SDES. Sept fois sur huit, cela sera l'élément NAME, et toutes les huitièmes fois (2 minutes) ce sera l'élément EMAIL.

Lorsque plusieurs applications fonctionnent de concert en utilisant une liaison trans-application à travers un CNAME commun pour chaque participant, par exemple dans une conférence multimédia composée d'une session RTP pour chaque support, les informations de SDES additionnelles PEUVENT être envoyées dans seulement une session RTP. Les autres sessions vont porter seulement l'élément CNAME. En particulier, cette approche devrait être appliquée aux multiples sessions d'un schéma de codage en couches (voir au paragraphe 2.4).

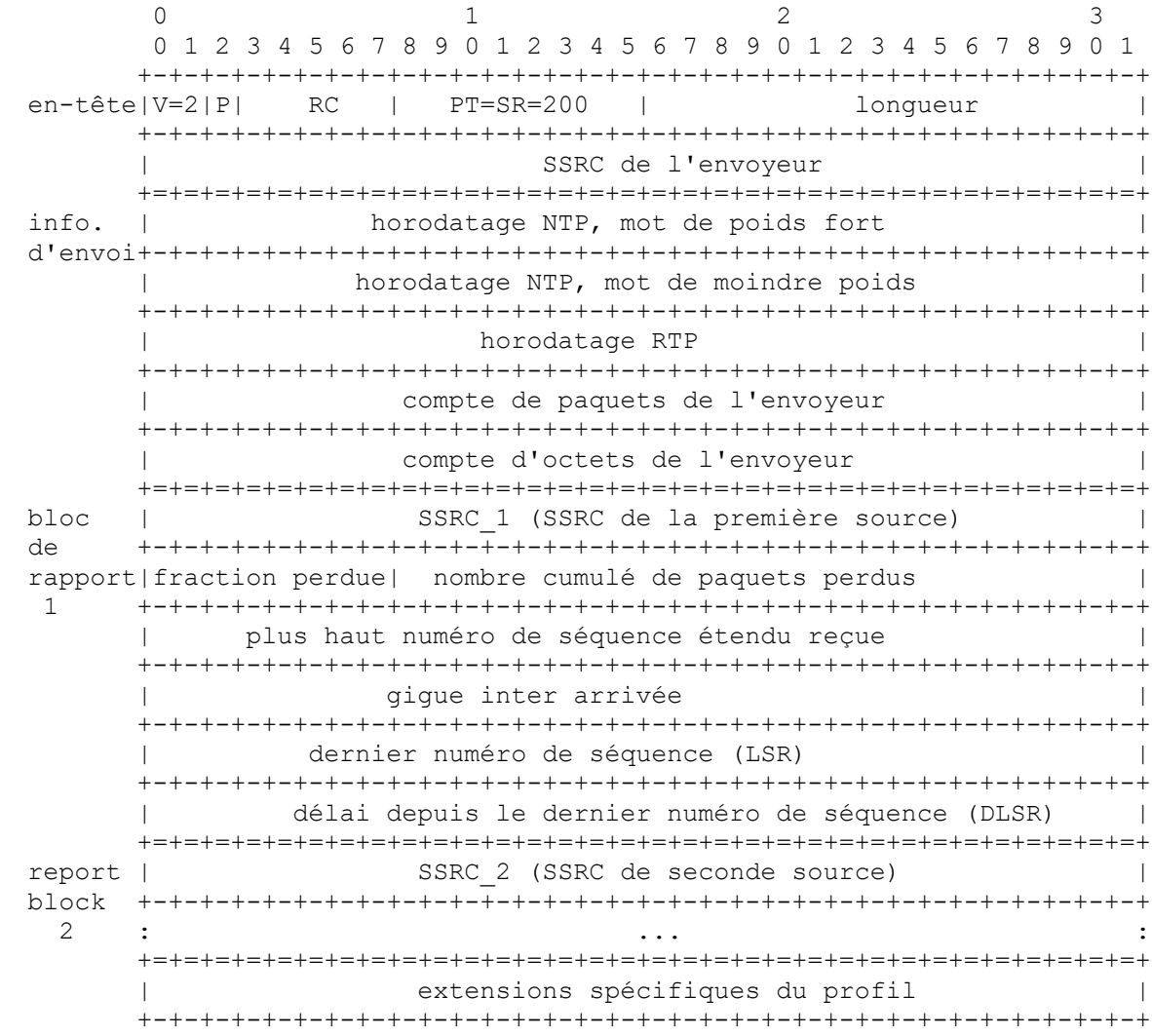
6.4 Rapports d'envoyeur et de receveur

Les receveurs RTP fournissent un retour de qualité de réception en utilisant les paquets de rapport RTCP qui peuvent prendre une forme parmi deux selon que le receveur est ou non aussi un envoyeur. La seule différence entre les formes de rapport d'envoyeur (SR) et de rapport de receveur (RR), à part le code de type de paquet, est que le rapport d'envoyeur comporte une section de 20 octets d'informations sur l'envoyeur qui est utilisée par les envoyeurs actifs. Le SR est produit si un site a envoyé des paquets de données durant l'intervalle écoulé depuis l'envoi du dernier rapport ou du précédent, autrement, c'est le RR qui est produit.

Les deux formes SR et RR comportent zéro, un ou plusieurs blocs de rapport de réception, un pour chacune des sources de synchronisation de laquelle ce receveur a reçu les paquets de données RTP depuis le dernier rapport. Les rapports ne sont pas produits pour les sources contributives énumérées dans la liste de CSRC. Chaque bloc de rapport de réception donne des statistiques sur les données reçues de la source particulière indiquée dans ce bloc. Comme un maximum de 31 blocs de rapport de réception va tenir dans un paquet SR ou RR, des paquets RR supplémentaires DEVRAIENT être empilés après le paquet initial SR ou RR comme nécessaire pour contenir les rapports de réception pour toutes les sources entendues durant l'intervalle écoulé depuis le dernier rapport. Si il y a trop de sources pour faire tenir tous les paquets RR nécessaires dans un paquet RTCP composé sans excéder la MTU du chemin de réseau, seul le sous-ensemble qui va tenir dans une MTU DEVRAIT alors être inclus dans chaque intervalle. Les sous-ensembles DEVRAIENT être choisis par round-robin à travers plusieurs intervalles afin que toutes les sources fassent l'objet de rapports.

Les paragraphes qui suivent définissent les formats des deux rapports, comment ils peuvent être étendus de façon spécifique d'un profil si une application requiert des informations de retour supplémentaires, et comment les rapports peuvent être utilisés. Les détails du rapport de réception par les traducteurs et mixeurs figurent à la Section 7.

6.4.1 SR : Paquet RTCP de rapport d'envoyeur



Le paquet de rapport de l'envoyeur comporte trois sections, éventuellement suivies par une quatrième section d'extensions spécifiques du profil si il en est une définie. La première section, l'en-tête, est longue de 8 octets. Les champs ont la signification suivante :

version (V) : 2 bits
Identifie la version de RTP, qui est la même dans les paquets RTCP que dans les paquets de données RTP. La version définie par la présente spécification est deux (2).

bourrage (P) : 1 bit
Si le bit bourrage est mis (à un), ce paquet RTCP individuel contient des octets de bourrage supplémentaires à la fin, qui ne font pas partie des informations de contrôle mais sont inclus dans le champ de longueur. Le dernier octet du bourrage est un compte du nombre d'octets de bourrage qui devraient être ignorés, y compris lui-même (il sera un multiple de quatre). Le bourrage peut être nécessaire pour certains algorithmes de chiffrement avec des tailles de bloc fixes. Dans un paquet RTCP composé, le bourrage n'est exigé que sur un paquet individuel parce que le paquet composé est chiffré comme un tout par la méthode du paragraphe 9.1. Et donc, le bourrage DOIT seulement être ajouté au dernier paquet individuel, et si le bourrage est ajouté à ce paquet, le bit de bourrage DOIT être mis sur ce seul paquet. Cette convention aide à la vérification de validité de l'en-tête décrite à l'Appendice A.2 et permet la détection de paquets provenant de certaines mises en œuvre anciennes qui établissent incorrectement le bit de bourrage sur le premier paquet individuel et ajoutent le bourrage au dernier paquet individuel.

compte de rapport de réception (RC) : 5 bits
C'est le nombre de blocs de rapport de réception contenus dans ce paquet. Une valeur de zéro est valide.

type de paquet (PT) : 8 bits

Il contient la constante 200 pour identifier que c'est un paquet SR RTCP.

longueur : 16 bits
La longueur de ce paquet RTCP en mots de 32 bits moins un, incluant l'en-tête et tout bourrage. (Le décalage de un fait de zéro une longueur valide et évite une éventuelle boucle infinie à l'examen d'un paquet RTCP composé, alors que le compte de mots de 32 bits évite une vérification de validité pour un multiple de 4.)

SSRC : 32 bits
L'identifiant de source de synchronisation pour l'origine de ce paquet SR.

La seconde section, les informations sur l'envoyeur, est longue de 20 octets et est présente dans chaque paquet de rapport de l'envoyeur. Elle récapitule la transmission des données provenant de cet envoyeur. Les champs ont la signification suivante :

horodatage NTP : 64 bits
Indique l'heure (voir à la Section 4) à laquelle ce rapport a été envoyé afin qu'elle puisse être utilisée en combinaison avec les horodatages retournés dans les rapports de réception provenant des autres receveurs pour mesurer le délai de propagation aller-retour jusqu'à ces receveurs. Les receveurs devraient s'attendre à ce que la précision de mesure de l'horodatage puisse être limitée à beaucoup moins que la résolution de l'horodatage NTP. L'incertitude de mesure de l'horodatage n'est pas indiquée car elle peut n'être pas connue. Sur un système qui n'a aucune notion de l'heure en cours mais a bien une horloge spécifique de système telle qu'un "temps de montée système", un envoyeur PEUT utiliser cette horloge comme référence pour calculer des horodatages NTP relatifs. Il est important de choisir une horloge d'utilisation courante afin que si des mises en œuvre distinctes sont utilisées pour produire les flux individuels d'une session multimédia, toutes les mises en œuvre utilisent la même horloge. Jusqu'à l'année 2036, les horodatages relatifs et absolus vont différer par le bit de poids fort de sorte que des comparaisons (invalides) vont montrer une grande différence ; à ce moment là, on peut espérer que les horodatages relatifs ne seront plus nécessaires. Un envoyeur qui n'a aucune notion de l'heure en cours ou du temps écoulé PEUT régler l'horodatage NTP à zéro.

horodatage RTP : 32 bits
Il correspond à la même heure que l'horodatage NTP (ci-dessus), mais dans les mêmes unités et avec le même décalage aléatoire que les horodatages RTP dans les paquets de données. Cette correspondance peut être utilisée pour la synchronisation intra- et inter-support pour les sources dont les horodatages NTP sont synchronisés, et peut être utilisée par des receveurs indépendants du support pour estimer la fréquence d'horloge RTP nominale. Noter que dans la plupart des cas, cet horodatage ne sera égal à l'horodatage RTP dans aucun paquet de données adjacent. Il DOIT plutôt être calculé à partir de l'horodatage NTP correspondant en utilisant les relations entre le compteur d'horodatage RTP et le temps réel tel qu'entretenu par la vérification périodique de l'heure courante à un moment d'échantillonnage.

compte de paquet de l'envoyeur : 32 bits
C'est le nombre total de paquets de données RTP transmis par l'envoyeur depuis le début de transmission jusque au moment où ce paquet SR a été généré. Le compte DEVRAIT être remis à zéro si l'envoyeur change son identifiant de SSRC.

compte d'octets de l'envoyeur : 32 bits
C'est le nombre total d'octets de charge utile (c'est-à-dire, non compris d'en-tête ou de bourrage) transmis dans les paquets de données RTP par l'envoyeur depuis le début de transmission jusque au moment où ce paquet SR a été généré. Le compte DEVRAIT être remis à zéro si l'envoyeur change son identifiant de SSRC. Ce champ peut être utilisé pour estimer le taux moyen de données de charge utile.

La troisième section contient zéro, un ou plusieurs blocs de rapport de réception selon le nombre des autres sources entendues par cet envoyeur depuis le dernier rapport. Chaque bloc de rapport de réception porte des statistiques sur la réception des paquets RTP provenant d'une seule source de synchronisation. Les receveurs NE DEVRAIENT PAS reporter les statistiques lorsque une source change son identifiant de SSRC à cause d'une collision. Ces statistiques sont :

SSRC_n (identifiant de source) : 32 bits
C'est l'identifiant de SSRC de la source à laquelle appartiennent les informations de ce bloc de rapport de réception.

fraction perdue : 8 bits
C'est la fraction des paquets de données RTP provenant de la source de SSRC_n qui est perdue depuis que le précédent paquet SR ou RR a été envoyé, exprimée par un nombre à virgule fixe avec la virgule binaire au bord gauche du champ. (Ceci est équivalent à prendre la partie entière après avoir multiplié la fraction de perte par 256.) Cette fraction se définit comme le nombre de pertes de paquets divisée par le nombre de paquets attendus, comme défini au paragraphe suivant.

Une mise en œuvre est donnée à l'Appendice A.3. Si la perte est négative du fait de duplications, la fraction de perte est réglée à zéro. Noter qu'un receveur ne peut pas dire que des paquets ont été perdus après le dernier reçu, et qu'il n'y aura pas de bloc de rapport de réception produit pour une source si tous les paquets provenant de cette source envoyés durant le dernier intervalle de rapport ont été perdus.

nombre cumulé de paquets perdus : 24 bits
C'est le nombre total de paquets de données RTP provenant de la source de SSRC_n qui ont été perdus depuis le début de la réception. Ce nombre est défini comme le nombre de paquets attendus moins le nombre de paquets réellement reçus, dans lequel le nombre de paquets reçus inclut tous ceux qui sont en retard ou dupliqués. Et donc, les paquets qui arrivent en retard ne sont pas comptés comme perdus, et la perte peut être négative si il y a des duplications. Le nombre de paquets attendus est défini comme étant le plus haut numéro de séquence étendu reçu, comme défini ensuite, moins le numéro de séquence initial reçu. Cela peut être calculé comme indiqué à l'Appendice A.3.

plus haut numéro de séquence étendu reçu : 32 bits
Les 16 bits de moindre poids contiennent le plus haut numéro de séquence reçu dans un paquet de données RTP provenant de la source de SSRC_n, et les 16 bits de poids fort étendent ce numéro de séquence avec le compte correspondant de cycles de numéro de séquence, qui peut être entretenu conformément à l'algorithme de l'Appendice A.1. Noter que différents receveurs au sein de la même session vont générer des extensions différentes au numéro de séquence si leurs instants de début diffèrent de façon significative.

gigue inter arrivée : 32 bits
C'est une estimation de la variance statistique du temps inter arrivée des paquets de données RTP, mesurée en unités d'horodatage et exprimée par un entier non signé. La gigue inter arrivées J est définie comme étant l'écart type (valeur absolue lissée) de la différence D de l'espacement de paquet chez le receveur comparée à l'envoyeur pour une paire de paquets. Comme montré dans l'équation ci-dessous, cela est équivalent à la différence du "temps de transit relatif" pour les deux paquets ; le temps de transit relatif est la différence entre l'horodatage RTP d'un paquet et l'horloge du receveur au moment de l'arrivée, mesurés dans les mêmes unités.

Si Si est l'horodatage RTP pour le paquet i, et si Ri est l'heure d'arrivée en unités d'horodatage RTP pour le paquet i, alors pour deux paquets i et j, D peut être exprimé par

$$D(i,j) = (Rj - Ri) - (Sj - Si) = (Rj - Sj) - (Ri - Si)$$

La gigue inter arrivée DEVRAIT être calculée en continu lorsque chaque paquet de données i est reçu de la source SSRC_n, en utilisant cette différence D pour ce paquet et le précédent paquet i-1 dans l'ordre d'arrivée (pas nécessairement en séquence), conformément à la formule

$$J(i) = J(i-1) + (|D(i-1,i)| - J(i-1))/16$$

Chaque fois qu'est produit un rapport de réception, la valeur courante de J est échantillonnée.

Le calcul de la gigue DOIT se conformer à la formule spécifiée ici afin de permettre aux moniteurs indépendants du profil de faire des interprétations valides des rapports provenant des différentes mises en œuvre. Cet algorithme est l'estimateur optimal de premier ordre et le paramètre gain 1/16 donne un bon rapport de réduction de bruit tout en conservant un taux de convergence raisonnable [22]. Un exemple de mise en œuvre est donné à l'Appendice A.8. Voir au paragraphe 6.4.4 une discussion sur les effets de la variation de la durée de paquet et du délai avant transmission.

dernier horodatage SR (LSR) : 32 bits
Les 32 bits médians parmi les 64 de l'horodatage NTP (comme expliqué à la Section 4) reçus au titre du plus récent paquet SDR de rapport d'envoyeur RTCP provenant de la source SSRC_n. Si aucun SR n'a été encore reçu, le champ est mis à zéro.

délai depuis le dernier SR (DLSR) : 32 bits
C'est le délai, exprimé en unités de 1/65536 s, entre la réception du dernier paquet SR de la source SSRC_n et l'envoi de ce bloc de rapport de réception. Si aucun paquet SR n'a encore été reçu de SSRC_n, le champ DLSR est mis à zéro.

Soit SSRC_r qui note le receveur produisant ce rapport de receveur. La source SSRC_n peut calculer le délai de propagation aller-retour au SSRC_r en enregistrant l'heure A à laquelle est reçu ce rapport de réception. Il calcule le temps total d'aller-retour A-LSR en utilisant le dernier champ d'horodatage SR (LSR), et en soustrayant ensuite ce champ pour donner le délai de propagation aller-retour de (A - LSR - DLSR). Cela est illustré à la Figure 2. Les temps sont donnés à la

fois en représentation hexadécimale des champs de 32 bits et en représentation en décimal à virgule flottante équivalente. Les deux points indiquent un champ de 32 bits divisé en partie entière de 16 bits et partie fraction de 16 bits.

Cela peut être utilisé comme une mesure approximative de la distance à la grappe receveuse, bien que certaines liaisons aient des délais très asymétriques.

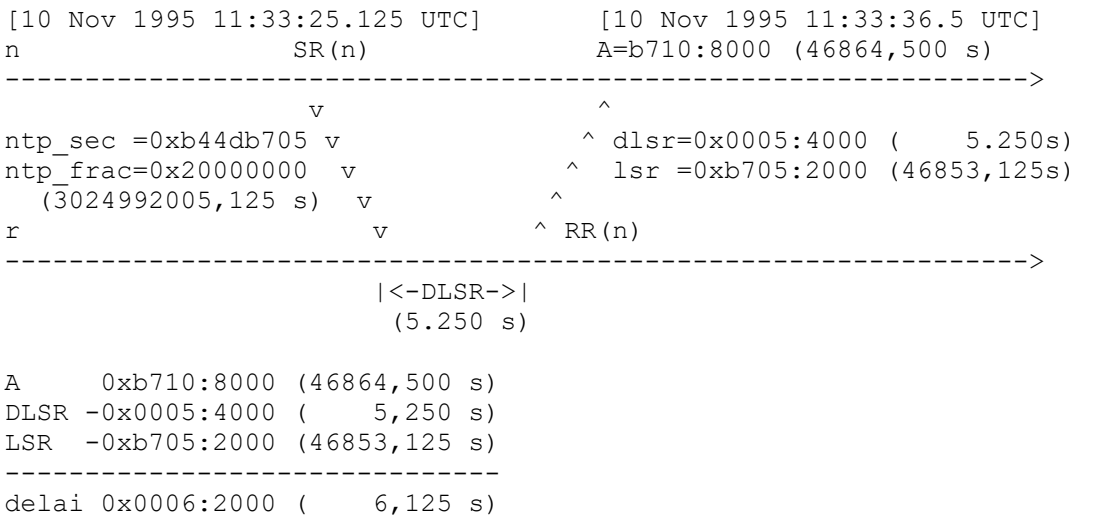
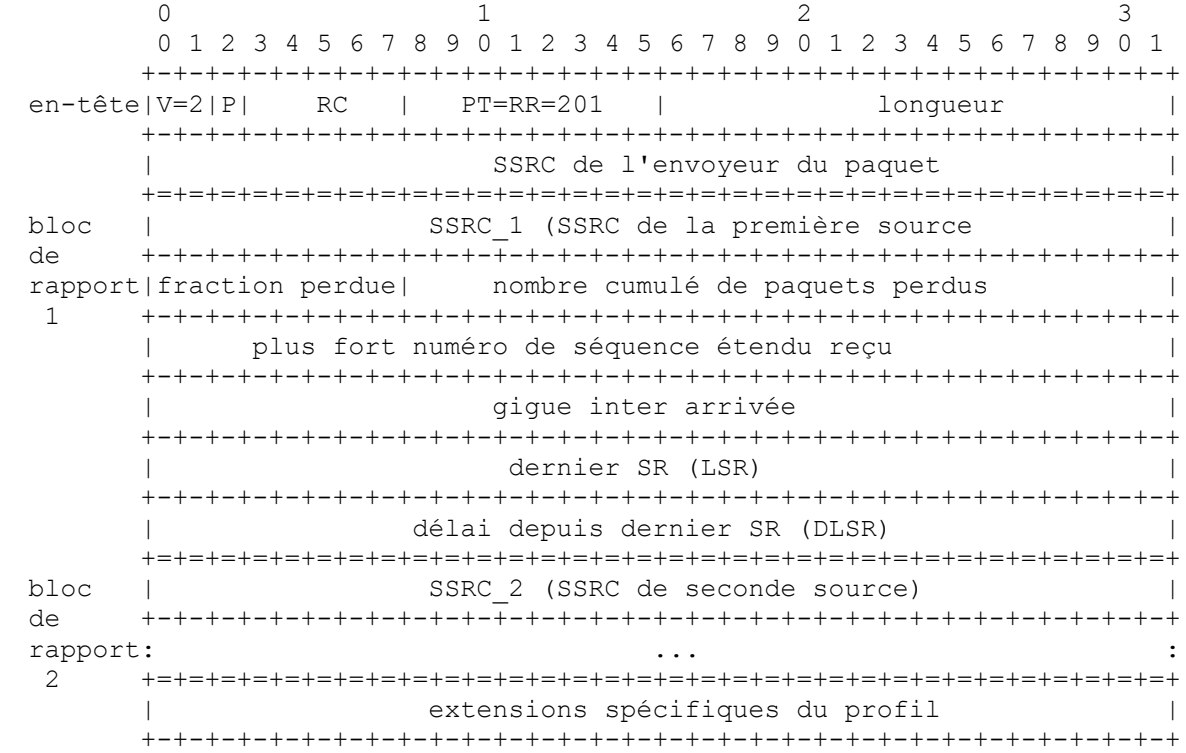


Figure 2 : Exemple de calcul de délais d'aller-retour

6.4.2 RR : Paquet RTCP de rapport de receveur



Le format du paquet de rapport de receveur (RR) est le même que celui du paquet SR sauf que le champ de type du paquet contient la constante 201 et que les cinq mots d'informations d'envoyeur sont omis (ce sont les horodatages NTP et RTP et les comptes de paquet et d'octet de l'envoyeur). Les champs restants ont la même signification que pour le paquet SR.

Un paquet RR vide (RC = 0) DOIT être mis en tête d'un paquet RTCP composé lorsque il n'y a pas de transmission ou réception de données à rapporter.

6.4.3 Extension des rapports d'envoyeur et de receveur

Un profil DEVRAIT définir les extensions spécifiques du profil aux rapports d'envoyeur et de receveur si il y a des informations supplémentaires qu'il est besoin de rapporter régulièrement sur l'envoyeur ou les receveurs. Cette méthode DEVRAIT être utilisée de préférence pour définir un autre type de paquet RTCP parce qu'il exige moins de redondance :

- o moins d'octets dans le paquet (pas d'en-tête RTCP ou de champ de SSRC) ;
- o analyse plus simple et plus rapide parce que les applications qui fonctionnent sous ce profil seront programmées pour toujours attendre les champs d'extension dans la localisation directement accessible après les rapports de réception.

L'extension est une quatrième section qui vient à la fin dans le paquet de rapport d'envoyeur ou receveur, après les blocs de rapport de réception, s'il en est. Si des informations d'envoyeur supplémentaires sont nécessaires, pour les rapports d'envoyeur elles seront alors incluses en premier dans la section extension, mais pour les rapports de receveur, elles ne seront pas présentes. Si des informations sur les receveurs sont à inclure, ces données DEVRAIT être structurées comme dans un dispositif de blocs parallèle au dispositif existant des blocs de rapport de réception ; c'est-à-dire que le nombre de blocs serait indiqué par le champ RC.

6.4.4 Analyse des rapports d'envoyeur et de receveur

Il est prévu que les retours de qualité de réception ne soient pas seulement utiles pour l'envoyeur mais aussi pour les autres receveurs et les surveillants tiers. L'envoyeur peut modifier ses émissions sur la base des retours ; les receveurs peuvent déterminer si des problèmes sont locaux, régionaux ou globaux ; les gestionnaires de réseau peuvent utiliser les surveillants indépendants du profil qui reçoivent seulement les paquets RTCP et non les paquets de données RTP correspondants pour évaluer les performances de leur réseau pour la distribution de la diffusion groupée.

Des comptes cumulatifs sont utilisés dans les blocs d'informations d'envoyeur et de rapport de receveur de telle sorte que les différences puissent être calculées entre deux rapports pour faire des mesures sur des courtes et des longues périodes, et pour fournir une résilience à la perte de rapports. La différence entre des deux derniers rapports reçus peut être utilisée pour estimer la qualité récente de la distribution. L'horodatage NTP est inclus de telle sorte que des taux puissent être calculés à partir de ces différences sur l'intervalle entre deux rapports. Comme cet horodatage est indépendant du débit d'horloge pour le codage des données, il est possible de mettre en œuvre des surveillants de qualité indépendants du profil et du codage.

Un exemple de calcul est le taux de perte de paquet sur l'intervalle entre deux rapports de réception. La différence des nombres cumulés de perte de paquets donne le nombre des pertes durant cet intervalle. La différence des derniers numéros de séquence étendus reçus donne le nombre de paquets attendus durant l'intervalle. Le ratio de ces deux valeurs est la fraction de perte de paquet sur l'intervalle. Ce ratio devrait être égal au champ Fraction perdue si les deux rapports sont consécutifs, mais autrement il peut ne pas l'être. Le taux de perte par seconde peut être obtenu en divisant la fraction perdue par la différence des horodatages NTP, exprimée en secondes. Le nombre de paquets reçus est le nombre de paquets attendus moins le nombre des perdus. Le nombre de paquets attendus peut aussi être utilisé pour juger de la validité statistique de toute estimation de pertes. Par exemple, la perte de 1 paquet sur 5 est moins significative que celle de 200 sur 1000.

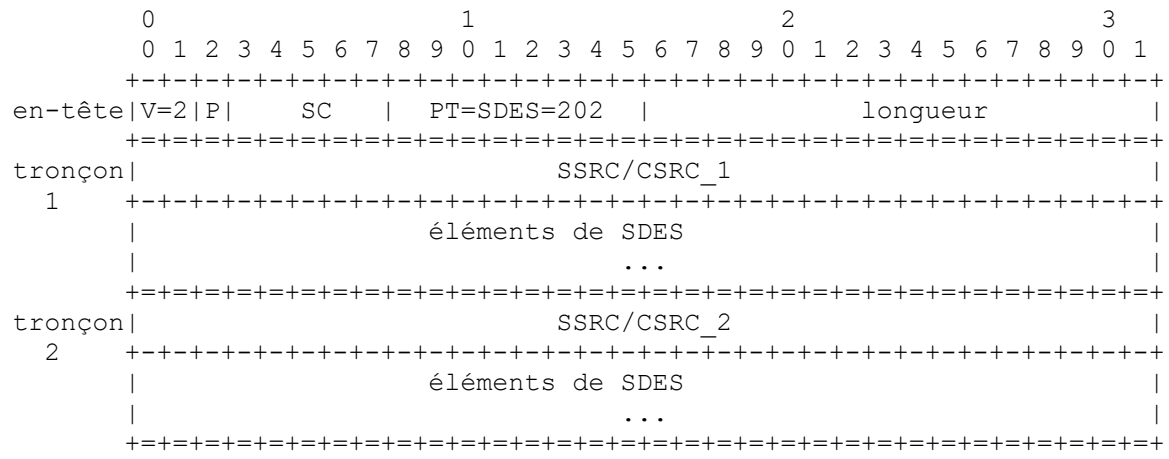
À partir des informations d'envoyeur, un surveillant tiers peut calculer le taux de données de charge utile moyen et le taux de paquet moyen sur un intervalle sans recevoir les données. Prendre le rapport des deux donne la taille moyenne de charge utile. Si on peut supposer que la perte de paquet est indépendante de la taille de paquet, le nombre de paquets reçus par un receveur particulier multiplié par la taille moyenne de charge utile (ou la taille de paquet correspondante) donne le débit apparent disponible pour ce receveur.

En plus des comptes cumulatifs qui permettent des mesures de perte de paquet à long terme en utilisant les différences entre les rapports, le champ Fraction perdue donne une mesure à court terme à partir d'un seul rapport. Cela devient plus important à mesure que la taille d'une session prend des proportions telles qu'il n'est plus possible de conserver pour tous les receveurs les informations d'état de réception, ou que l'intervalle entre les rapports devient assez long pour qu'un seul rapport ait été reçu d'un receveur particulier.

Le champ de gigue inter arrivée donne une seconde mesure à court terme de l'encombrement du réseau. La perte de paquet retrace l'encombrement persistent alors que la mesure de gigue retrace un encombrement transitoire. La mesure de la gigue peut indiquer l'encombrement avant qu'il ne conduise à la perte de paquet. Le champ de gigue inter arrivée est seulement un cliché au moment d'un rapport et n'est pas destiné à être pris au sens quantitatif. Il est plutôt destiné aux comparaisons à travers un certain nombre de rapports provenant d'un receveur sur une période, ou de plusieurs receveurs, par exemple, au sein d'un seul réseau, au même moment. Pour permettre une comparaison entre les receveurs, il est important que la gigue soit calculée selon la même formule par tous les receveurs.

Comme le calcul de la gigue se fonde sur l'horodatage RTP qui représente l'instant où les premières données du paquet sont échantillonnées, toute variation du délai entre ce moment d'échantillonnage et le moment où le paquet est transmis va affecter la gigue résultante qui sera calculée. Une telle variation du délai va survenir pour des paquets audio de durée variable. Elle va aussi survenir pour des codages vidéo parce que l'horodatage est le même pour tous les paquets d'une trame mais ces paquets ne sont pas tous transmis au même moment. La variation du délai jusqu'à la transmission réduit la précision du calcul de la gigue comme mesure du comportement du réseau par lui-même, mais son inclusion est appropriée en considérant que la mémoire tampon de réception doit s'en accommoder. Lorsque le calcul de la gigue est utilisé comme mesure de comparaison, la composante (constante) due à la variation de délai jusqu'à la transmission se retranche de sorte qu'un changement de la composante de gigue du réseau peut alors être observé sauf si il est relativement petit. Si le changement est petit, il est vraisemblable qu'il sera sans conséquence.

6.5 SDES : paquet RTCP de description de source



Le paquet SDES est une structure à trois niveaux composée d'un en-tête et de zéro, un ou plusieurs tronçons, chacun d'eux étant composé d'éléments qui décrivent la source identifiée dans ce tronçon. Les éléments sont décrits individuellement dans les paragraphes suivants.

version (V), bourrage (P, *padding*), longueur :
Comme décrit pour le paquet SR (voir au paragraphe 6.4.1).

type de paquet (PT) : 8 bits
Contient la constante 202 pour identifier cela comme un paquet SDES RTCP.

compte de source (SC, *source count*) : 5 bits
Le nombre de tronçons SSRC/CSRC contenus dans ce paquet SDES. Une valeur de zéro est valide mais inutile.

Chaque tronçon consiste en un identifiant de SSRC/CSRC suivi par une liste de zéro, un ou plusieurs éléments, qui portent les informations sur le SSRC/CSRC. Chaque tronçon commence sur une limite de 32 bits. Chaque élément comporte un champ de type de 8 bits, un compte d'octets de 8 bits qui décrit la longueur du texte (et qui n'inclut donc pas cet en-tête de deux octets), et le texte lui-même. Noter que le texte peut être inférieur à 255 octets, mais cela est cohérent avec la nécessité de limiter la consommation de bande passante RTCP.

Le texte est codé conformément au codage UTF-8 spécifié dans la RFC 2279 [5]. L'US-ASCII est un sous-ensemble de ce codage et exige un codage supplémentaire. La présence de codages multi-octets est indiquée en réglant le bit de poids fort d'un caractère à une valeur de un.

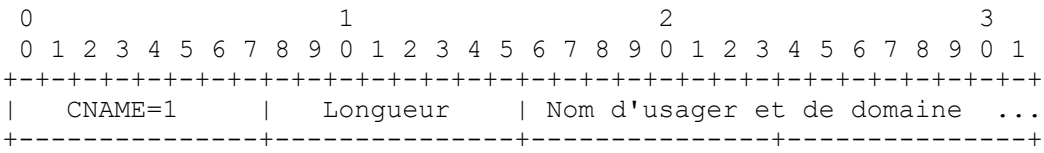
Les éléments sont contigus, c'est-à-dire que les éléments ne sont pas bourrés individuellement jusqu'à une limite de 32 bits. Le texte n'est pas terminé par des octets à zéro parce que certains codages multi-octets comportent des octets nuls. La liste des éléments dans chaque tronçon DOIT être terminée par un ou plusieurs octets nuls, dont le premier est interprété comme un type d'élément de zéro pour noter la fin de la liste. Aucun octet de longueur ne suit l'octet nul de type d'élément, mais des octets nuls supplémentaires DOIVENT être inclus si nécessaire pour bourrer jusqu'à la prochaine frontière de 32 bits. Noter que ce bourrage est distinct de celui indiqué par le bit P dans l'en-tête RTCP. Un tronçon avec zéro élément (quatre octets nuls) est valide mais inutile.

Les systèmes d'extrémité envoient un paquet SDES contenant leur propre identifiant de source (le même que le SSRC dans

l'en-tête fixe RTP). Un mixeur envoie un paquet SDES contenant un tronçon pour chaque source contributive d'où il reçoit des informations de SDES, ou plusieurs paquets SDES complets dans le format ci-dessus si il y a plus de 31 de ces sources (voir la section 7).

Les éléments de SDES actuellement définis sont décrits dans les paragraphes suivants. Seul l'élément CNAME est obligatoire. Certains éléments présentés ici peuvent n'être utiles que pour des profils particuliers, mais les types d'éléments sont tous alloués à partir d'un espace commun pour favoriser l'utilisation partagée et pour simplifier les applications indépendantes du profil. Des éléments supplémentaires peuvent être définis dans un profil en enregistrant les numéros de type auprès de l'IANA comme décrit à la Section 15.

6.5.1 CNAME : Élément SDES d'identifiant de point d'extrémité canonique



L'identifiant de CNAME a les propriétés suivantes :

- Comme l'identifiant de SSRC alloué de façon aléatoire peut changer si un conflit est découvert ou si un programme est redémarré, l'élément CNAME DOIT être inclus pour fournir le lien de l'identifiant de SSRC à un identifiant pour la source (d'envoyeur ou de receveur) qui reste constant.

- Comme l'identifiant de SSRC, l'identifiant de CNAME DEVRAIT aussi être unique parmi tous les participants au sein d'une session RTP.

- Pour fournir un lien à travers plusieurs outils de support utilisés par un participant dans un ensemble de sessions RTP en rapport, le CNAME DEVRAIT être fixe pour ce participant.

- Pour faciliter la surveillance par un tiers, le CNAME DEVRAIT convenir aussi bien pour qu'un programme ou qu'une personne localise la source.

Donc, le CNAME DEVRAIT être déduit par un algorithme et non entré manuellement, lorsque possible. Pour satisfaire ces exigences, le format suivant DEVRAIT être utilisé sauf si un profil spécifie une autre syntaxe ou sémantique. L'élément CNAME DEVRAIT avoir le format "usager@hôte" ou "hôte" si un nom d'utilisateur n'est pas disponible comme sur les systèmes à un seul utilisateur. Pour les deux formats, "hôte" est soit le nom de domaine pleinement qualifié de l'hôte d'où proviennent les données en temps réel, formaté conformément aux règles spécifiées dans les RFC 1034 [6], RFC 1035 [7] et au paragraphe 2.1 de la RFC 1123 [8] ; ou la représentation standard ASCII de l'adresse numérique de l'hôte sur l'interface utilisée pour la communication RTP. Par exemple, la représentation standard ASCII d'une adresse IP version 4 est en "décimal séparé par des points", et pour IP version 6, les adresses sont textuellement représentées comme des groupes de chiffres en hexadécimal séparés par deux points (avec les variations détaillées dans la RFC 3513 [23]). D'autres types d'adresses sont prévus pour avoir des représentations ASCII qui sont mutuellement uniques. Le nom de domaine pleinement qualifié est plus pratique pour un observateur humain et peut éviter d'avoir besoin d'envoyer en plus un élément NAME, mais il peut être difficile voire impossible de l'obtenir de façon fiable dans certains environnements de fonctionnement. Les applications qui pourraient fonctionner dans de tels environnements DEVRAIENT utiliser à la place la représentation ASCII de l'adresse.

Pour un système multi utilisateurs, des exemples sont "doe@sleepy.example.com", "doe@192.0.2.89" ou "doe@2201:056D::112E:144A:1E24". Sur un système sans nom d'utilisateur, des exemples seraient "sleepy.example.com", "192.0.2.89" ou "2201:056D::112E:144A:1E24".

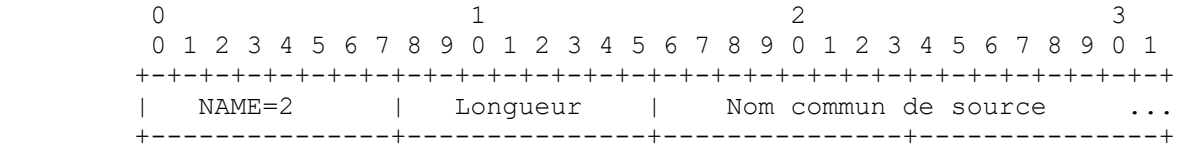
Le nom d'utilisateur DEVRAIT être sous une forme qu'un programme tel que "finger" ou "talk" puisse utiliser, c'est-à-dire, c'est normalement le nom de connexion plutôt qu'un nom de famille. Le nom de l'hôte n'est pas nécessairement identique à celui de l'adresse de messagerie électronique du participant.

Cette syntaxe ne va pas fournir des identifiants univoques pour chaque source si une application permet à un utilisateur de générer plusieurs sources à partir d'un hôte. Une telle application devrait s'appuyer sur le SSRC pour identifier la source, ou le profil pour cette application devrait spécifier une syntaxe supplémentaire pour l'identifiant de CNAME.

Si chaque application crée indépendamment son CNAME, les CNAME résultants peuvent n'être pas identiques comme il serait exigé pour fournir un lien à travers plusieurs outils de supports appartenant à un participant dans un ensemble de session RTP s'y rapportant. Si un lien inter-supports est exigé, il peut être nécessaire que le CNAME de chaque outil soit configuré en externe avec la même valeur par un outil de coordination.

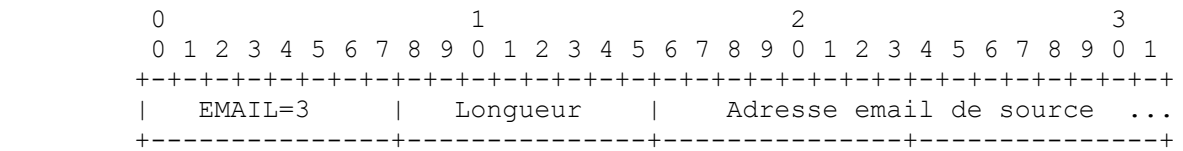
Les auteurs d'applications devraient être conscients que les allocations d'adresse de réseau privé telles que les allocations de Net-10 proposées dans la RFC 1918 [24] peuvent créer des adresses réseau qui ne sont pas uniques au monde. Cela conduirait à des CNAME non uniques si les hôtes avec des adresses privées et pas de connexité IP directe avec l'Internet public ont leurs paquets RTP transmis à l'Internet public à travers un traducteur de niveau RTP. (Voir aussi la RFC 1627 [25].) Pour traiter ce cas, les applications PEUVENT fournir des moyens de configurer un CNAME univoque, mais il appartient au traducteur de traduire les CNAME provenant d'adresses privées en adresses publiques si nécessaire pour empêcher la divulgation des adresses privées.

6.5.2 NAME : Élément SDES de nom d'utilisateur



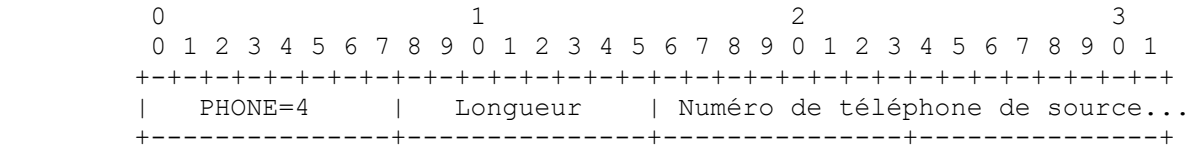
C'est le nom réel utilisé pour décrire la source, par exemple, "John Doe, Recycleur de bits". Il peut prendre toute forme au choix de l'utilisateur. Pour des applications telles que la téléconférence, cette forme de nom peut être la plus appropriée pour l'affichage dans la liste des participants, et risque donc d'être l'élément le plus fréquemment envoyé à part CNAME. Les profils PEUVENT établir de telles priorités. La valeur NAME est destinée à rester constante au moins pour la durée d'une session. On NE DEVRAIT PAS s'attendre à ce qu'il soit unique parmi tous les participants à la session.

6.5.3 EMAIL : Élément SDES d'adresse de messagerie électronique



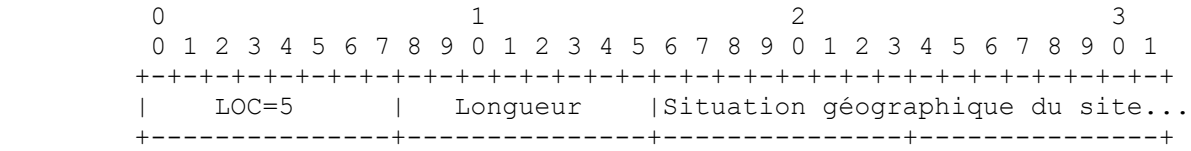
L'adresse de messagerie est formatée conformément à la RFC 2822 [9], par exemple, "John.Doe@exemple.com". La valeur EMAIL est supposée rester constante pour la durée d'une session.

6.5.4 PHONE : Élément SDES de numéro de téléphone



Le numéro de téléphone DEVRAIT être formaté avec le signe plus à la place du code d'accès international. Par exemple, "+1 908 555 1212" pour un numéro aux États Unis.

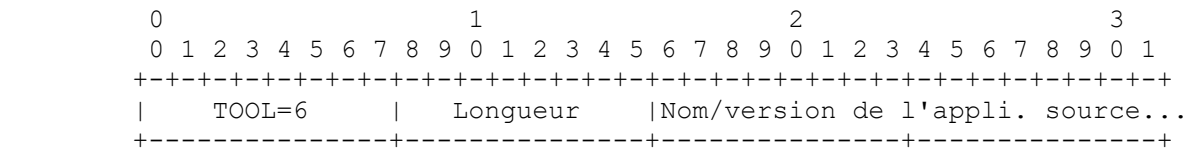
6.5.5 LOC : Élément SDES de localisation géographique de l'utilisateur



Selon l'application, différents degrés de détail sont appropriés pour cet élément. Pour les applications de conférence, une chaîne comme "Murray Hill, New Jersey" peut être suffisante, alors que pour un système d'insignes actifs, des chaînes

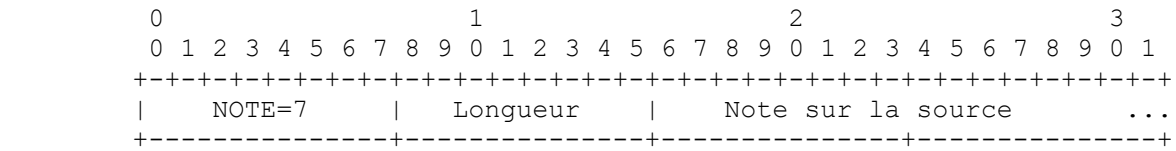
comme "Pièce 2A244, AT&T BL MH" peut être appropriée. Le degré de détail est laissé à la discrétion de la mise en œuvre et/ou utilisateur, mais format et contenu PEUVENT être prescrits par un profil. La valeur LOC est supposée rester constante pour la durée d'une session, excepté pour les hôtes mobiles.

6.5.6 TOOL : Élément SDES de nom d'application ou d'outil



C'est une chaîne qui donne le nom et éventuellement la version de l'application qui génère le flux, par exemple, "videotool 1.2". Cette information peut être utile aux fins de débogage et est similaire aux en-têtes Mailer ou Mail-System-Version de SMTP. La valeur TOOL est supposée rester constante pour la durée d'une session.

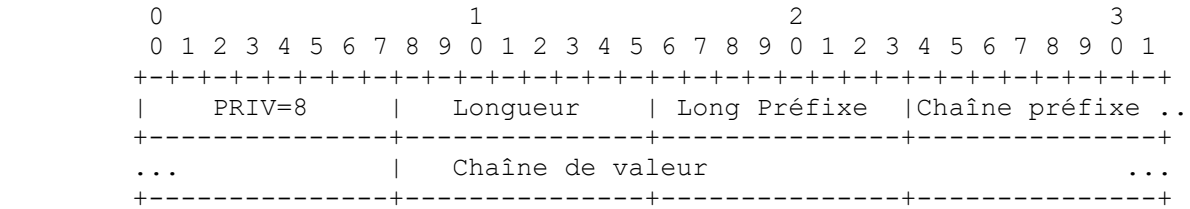
6.5.7 NOTE : Élément SDES de notice/état



La sémantique suivante est suggérée pour cet élément, mais elle, ou une autre sémantique, PEUT être explicitement définie par un profil. L'élément NOTE est destiné aux messages transitoires qui décrivent l'état actuel de la source, par exemple, "on ne peut pas parler par téléphone". Ou, durant un séminaire, cet élément peut être utilisé pour porter le titre de la conférence. Il ne devrait être utilisé que pour porter des informations exceptionnelles et NE DEVRAIT PAS être inclus systématiquement par tous les participants parce que cela ralentirait le débit d'envoi des rapports de réception et des CNAME, ce qui dégraderait les performances du protocole. En particulier, il NE DEVRAIT PAS être inclus comme élément dans un fichier de configuration d'utilisateur ni généré automatiquement dans une "citation du jour".

Comme l'élément NOTE peut être important à afficher alors qu'il est actif, le débit auquel d'autres éléments non CNAME tels que NAME sont transmis peut être réduit de telle sorte que l'élément NOTE puisse prendre leur part de la bande passante RTCP. Lorsque le message transitoire devient inactif, l'élément NOTE DEVRAIT continuer d'être transmis un petit nombre de fois au même rythme de répétition mais avec une chaîne de longueur zéro pour le signaler aux receveurs. Cependant, les receveurs DEVRAIENT aussi considérer l'élément NOTE comme inactif si il n'est pas reçu pour un faible multiple du rythme de répétition, ou peut-être 20 à 30 intervalles RTCP.

6.5.8 PRIV : Élément SDES d'extensions privées



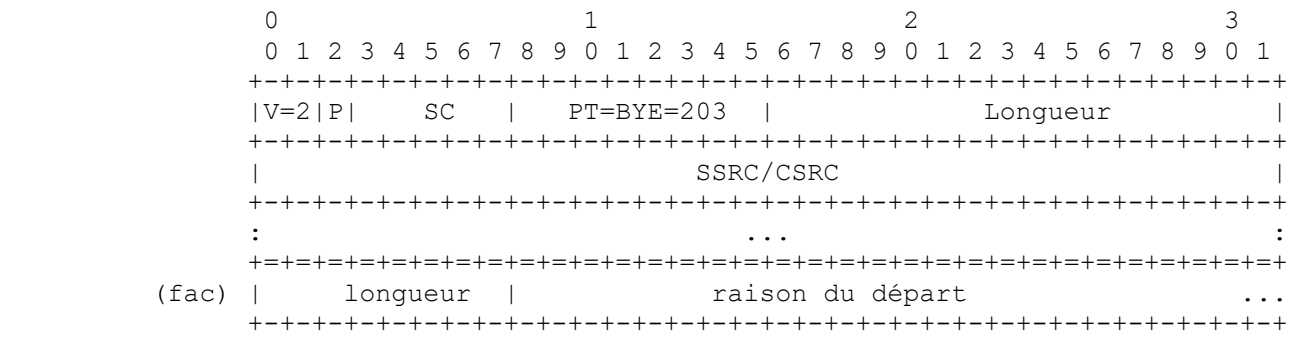
Cet élément est utilisé pour définir des extensions SDES expérimentales ou spécifiques de l'application. L'élément contient un préfixe consistant en une paire longueur-chaîne, suivie par la chaîne de valeur qui remplit le reste de l'élément et porte l'information désirée. Le champ longueur de préfixe est de 8 bits. La chaîne de préfixe est un nom choisi par la personne qui définit l'élément PRIV comme unique par rapport aux autres éléments PRIV que cette application pourrait recevoir. Le créateur de l'application pourrait choisir d'utiliser le nom de l'application plus une identification de sous-type supplémentaire si nécessaire. Autrement, il est RECOMMANDÉ que les autres choisissent un nom fondé sur l'entité qu'il représente, puis de coordonner l'utilisation du nom au sein de cette entité.

Noter que le préfixe consomme de l'espace au sein de la longueur totale de 255 octets de l'élément, de sorte que le préfixe devrait rester aussi court que possible. Cette facilité et la bande passante réduite de RTCP NE DEVRAIENT PAS être surchargées ; elle n'est pas destinée à satisfaire toutes les exigences de communication de contrôle de toutes les

applications.

Les préfixes SDES PRIV ne seront pas enregistrés par l'IANA. Si certaines formes de l'élément PRIV se révèlent être d'utilité générale, il DEVRAIT plutôt être alloué un type régulier d'élément SDES enregistré auprès de l'IANA afin qu'aucun préfixe ne soit exigé. Cela simplifie l'utilisation et accroît l'efficacité de la transmission.

6.6 Paquet RTCP BYE pour prendre congé



Le paquet BYE indique que une ou plusieurs sources ne sont plus actives.

version (V), bourrage (P), longueur :
Comme décrit pour le paquet SR (voir au paragraphe 6.4.1).

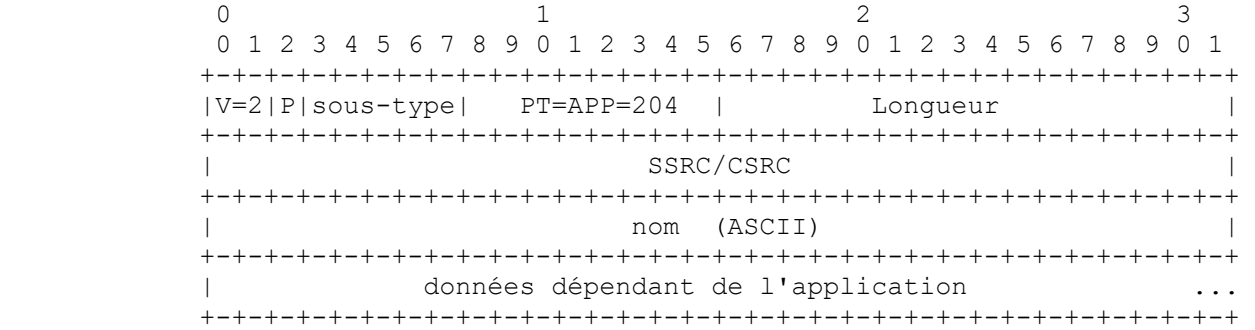
type de paquet (PT) : 8 bits
Contient la constante 203 pour identifier cela comme un paquet RTCP BYE.

compte de source (SC) : 5 bits
Le nombre de SSRC/identifiants de CSRC inclus dans ce paquet BYE. Une valeur de zéro est valide, mais inutile.

Les règles pour le moment où un paquet BYE devrait être envoyé sont spécifiées aux paragraphes 6.3.7 et 8.2.

Si un paquet BYE est reçu par un mixeur, celui-ci DEVRAIT transmettre le paquet BYE avec un ou des identifiants de SSRC/CSRC inchangés. Si le mixeur ferme, il DEVRAIT envoyer un paquet BYE faisant la liste de toutes les sources contributives qu'il traite, ainsi que son propre identifiant de SSRC. Facultativement, le paquet BYE PEUT inclure un compte d'octets de 8 bits suivi par ce nombre d'octets de texte indiquant la raison du départ, par exemple, "disfonctionnement de la caméra" ou "détection d'une boucle RTP". La chaîne a le même codage que celui décrit pour SDES. Si la chaîne remplit le paquet jusqu'à la prochaine frontière de 32 bits, elle n'est pas terminée par des octets nuls. Sinon, le paquet BYE DOIT être bourré avec des octets nuls jusqu'à la prochaine frontière de 32 bits. Ce bourrage est distinct de celui indiqué par le bit P dans l'en-tête RTCP.

6.7 APP : Paquet RTCP défini par l'application



Le paquet APP est destiné à une utilisation expérimentale lors du développement de nouvelles applications et dispositifs, sans qu'il soit besoin d'un enregistrement de valeur de type de paquet. Les paquets APP avec des noms non reconnus DEVRAIENT être ignorés. Après essais et si une utilisation plus large est justifiée, il est RECOMMANDÉ que chaque paquet APP soit redéfini sans les champs de sous-type et de nom et enregistrés auprès de l'IANA en utilisant un type de

paquet RTCP.

version (V), bourrage (P), longueur :
Comme décrit pour le paquet SR (voir au paragraphe 6.4.1).

sous-type : 5 bits
Peut être utilisé comme un sous-type pour permettre de définir un ensemble de paquets APP sous un nom unique, ou pour toutes données dépendantes de l'application.

type de paquet (PT) : 8 bits
Contient la constante 204 pour identifier cela comme un paquet APP de RTCP.

nom : 4 octets
C'est un nom choisi par la personne qui définit l'ensemble de paquets APP comme unique par rapport aux autres paquets APP que cette application pourrait recevoir. Le créateur de l'application pourrait choisir d'utiliser le nom de l'application, puis de coordonner l'allocation des valeurs de sous-type a d'autres qui voudraient définir un nouveau type de paquets pour l'application. Autrement, il est RECOMMANDÉ que ces autres choisissent un nom fondé sur l'entité qu'il représente, puis de coordonner l'utilisation du nom au sein de cette entité. Le nom est interprété comme une séquence de quatre caractères ASCII, les caractères majuscules et minuscules étant traités comme distincts.

données dépendantes de l'application : longueur variable
Les données dépendantes de l'application peuvent apparaître ou non dans un paquet APP. Elles sont interprétées par l'application et non par RTP. Leur longueur DOIT être un multiple de 32 bits.

7. Traducteurs et mixeurs RTP

En plus des systèmes d'extrémité, RTP prend en charge la notion de "traducteurs" et de "mixeurs", qui pourraient être considérés comme des "systèmes intermédiaires" au niveau de RTP. Bien que cette prise en charge ajoute de la complexité au protocole, la nécessite de ces fonctions a été clairement établie par des expériences avec des applications audio et vidéo en diffusion groupée sur l'Internet. Les exemples d'utilisations de traducteurs et de mixeurs donnés au paragraphe 2.3 proviennent de la présence de pare-feu et de connexions à faible bande passante, qui vont vraisemblablement durer.

7.1 Description générale

Un traducteur/mixeur RTP connecte deux "nuages" de niveau transport ou plus. Normalement, chaque nuage est défini par un réseau commun et un protocole de transport (par exemple, IP/UDP) plus une adresse de diffusion groupée et un accès de destination de niveau transport ou une paire d'adresses d'envoi individuel et d'accès. (Les traducteurs de protocole de niveau transport, tels que de IP version 4 en IP version 6, peuvent être présents au sein d'un nuage de façon invisible à RTP.) Un système peut servir de traducteur ou mixeur pour un certain nombre de sessions RTP, mais chacune est considérée comme une entité logique distincte.

Afin d'éviter de créer une boucle lors de l'installation d'un traducteur ou mixeur, les règles suivantes DOIVENT être observées :

- Chacun des nuages connectés par des traducteurs et mixeurs participant à une session RTP DOIT être distinct des autres par au moins un de ces paramètres (protocole, adresse, accès), ou DOIT être isolé des autres au niveau réseau.

- Une déduction de la première règle est qu'il NE DOIT PAS y avoir plusieurs traducteurs ou mixeurs connectés en parallèle à moins que par quelque arrangement ils partitionnent l'ensemble des sources à transmettre.

De même, tous les systèmes d'extrémité RTP qui peuvent communiquer à travers un ou plusieurs traducteurs ou mixeurs RTP partagent le même espace de SSRC, c'est-à-dire que les identifiants de SSRC DOIVENT être uniques parmi tous ces systèmes d'extrémité. Le paragraphe 8.2 décrit l'algorithme de résolution de collision par lequel les identifiants de SSRC sont conservés uniques et les boucles détectées.

Il peut y avoir de nombreuses variétés de traducteurs et mixeurs conçues pour différents objets et applications. Quelques exemples sont d'ajouter ou retirer le chiffrement, changer le codage des données ou les protocoles sous-jacents, ou

dupliquer entre une adresse de diffusion groupée et une ou plusieurs adresses d'envoi individuel. La distinction entre traducteurs et mixeurs est qu'un traducteur passe séparément à travers les flux de données provenant de différentes sources, alors qu'un mixeur les combine pour former un nouveau flux :

Traducteur :
Il transmet les paquets RTP avec leur identifiant de SSRC intact ; cela donne aux receveurs la possibilité d'identifier les sources individuelles même si les paquets provenant de toutes les sources passent à travers le même traducteur et portent l'adresse réseau de source du traducteur. Certains types de traducteurs vont passer les données inchangées, mais d'autres PEUVENT changer le codage des données et donc le type de charge utile et l'horodatage des données RTP. Si plusieurs paquets de données sont recodés en un seul, ou vice versa, un traducteur DOIT allouer de nouveaux numéros de séquence aux paquets sortants. Les pertes dans le flux entrant de paquets peuvent induire les trous correspondants dans les numéros de séquence sortants. Les receveurs ne peuvent pas détecter la présence d'un traducteur à moins qu'ils ne sachent par quelque autre moyen quel type de charge utile ou d'adresse de transport a été utilisée par la source originale.

Mixeur :
Il reçoit les flux de paquets de données RTP d'une ou plusieurs sources, change éventuellement le format des données, combine les flux d'une certaine manière puis transmet le flux combiné. Comme l'écoulement temporel entre plusieurs sources d'entrées n'est généralement pas synchronisé, le mixeur va faire des ajustements temporels entre les flux et générer sa propre synchronisation pour le flux combiné, de sorte qu'il est la source de synchronisation. Et donc, tous les paquets de données transmis par un mixeur DOIVENT être marqués avec le propre identifiant de SSRC du mixeur. Afin de préserver l'identité des sources originales qui contribuent au paquet mixé, le mixeur DEVRAIT insérer leurs identifiants de SSRC dans la liste des identifiants de CSRC suivis de l'en-tête RTP fixe du paquet. Un mixeur qui est aussi lui-même une source contributive pour des paquets DEVRAIT explicitement inclure son propre identifiant de SSRC dans la liste de CSRC pour ce paquet.

Pour certaines applications, il PEUT être acceptable qu'un mixeur n'identifie pas les sources dans la liste de CSRC. Cependant, cela introduit le danger que des boucles impliquant ces sources ne soient pas détectées.

L'avantage d'un mixeur sur un traducteur pour des applications comme l'audio est que la bande passante de sortie est limitée à celle d'une source même lorsque plusieurs sources sont actives du côté entrée. Cela peut être important pour les liaisons à faible bande passante. L'inconvénient est que les receveurs sur le côté sortie n'ont aucun contrôle sur les sources qui sont transmises ou assourdies, à moins que quelque mécanisme ne soit mis en œuvre pour le contrôle à distance du mixeur. La régénération des informations de synchronisation par des mixeurs signifie aussi que les receveurs ne peuvent pas faire de synchronisation inter supports des flux originaux. Un mixer multi-supports pourrait le faire.

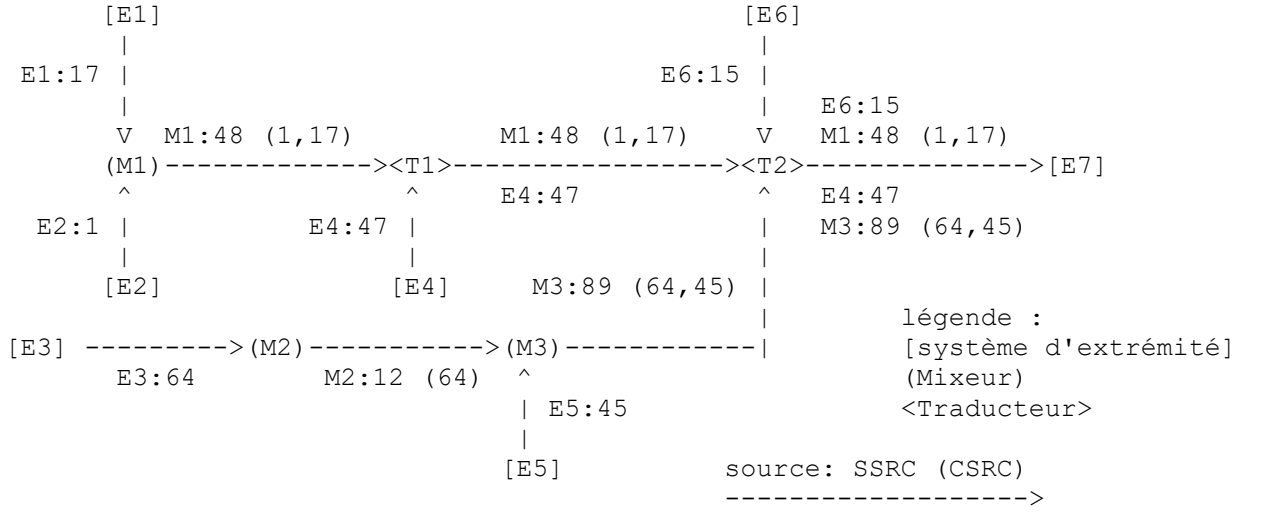


Figure 3 : Exemple de réseau RTP avec systèmes d'extrémité, mixeurs et traducteurs

Une collection de mixeurs et traducteurs est montrée à la Figure 3 pour illustrer leur effet sur les identifiants de SSRC et de CSRC. Sur la figure, les systèmes d'extrémité sont montrés comme des rectangles (nommés E), les traducteurs comme des triangles (nommés T) et les mixeurs comme des ovales (nommés M). La notation "M1: 48(1,17)" désigne un paquet originaire d'un mixeur M1, identifié par la valeur de SSRC de 48 de M1 (aléatoire) et deux identifiants de CSRC, 1 et 17, copiés des identifiants de SSRC des paquets provenant de E1 et E2.

7.2 Traitement de RTCP dans les traducteurs

En plus de la transmission des paquets de données, peut-être modifiés, traducteurs et mixeurs DOIVENT aussi traiter les paquets RTCP. Dans de nombreux cas, ils vont séparer les paquets RTCP composés reçus des systèmes terminaux pour agréger les informations de SDES et pour modifier les paquets SR ou RR. La retransmission de ces informations peut être déclanchée par l'arrivée du paquet ou par le temporisateur d'intervalle RTCP du traducteur ou mixeur lui-même.

Un traducteur qui ne modifie pas les paquets de données, par exemple qui ne fait que dupliquer d'une adresse de diffusion groupée à une adresse d'envoi individuel, PEUT simplement aussi bien transmettre les paquets RTCP non modifiés. Un traducteur qui transforme la charge utile d'une certaine façon DOIT faire les transformations correspondantes dans les informations SR et RR afin qu'elle reflètent encore les caractéristiques des données et la qualité de réception. Ces traducteurs NE DOIVENT PAS simplement transmettre les paquets RTCP. En général, un traducteur NE DEVRAIT PAS agréger de paquets SR et RR provenant de sources différentes en un seul paquet car cela réduirait la précision des mesures de délais de propagation fondées sur les champs LSR et DLSR.

Informations d'envoyeur de SR :
Un traducteur ne génère pas ses propres informations d'envoyeur, mais transmet les paquets SR reçus d'un nuage aux autres. Le SSRC est laissé intact mais les informations d'envoyeur DOIVENT être modifiées si c'est exigé par la traduction. Si un traducteur change le codage des données, il DOIT changer le champ "Compte d'octets de l'envoyeur". Si il combine aussi plusieurs paquets de données en un paquet de sortie, il DOIT changer le champ "Compte de paquets de l'envoyeur". Si il change la fréquence de l'horodatage, il DOIT changer le champ "Horodatage RTP" dans le paquet SR.

Blocs de rapport de réception SR/RR :
Un traducteur transmet les rapports de réception reçus d'un nuages aux autres. Noter qu'ils s'écoulent dans la direction opposée de celle des données. Le SSRC est laissé intact. Si un traducteur combine plusieurs paquets de données en un paquet de sortie, et donc change les numéros de séquence, il DOIT faire la manipulation inverse pour les champs de perte de paquet et le champ "Dernier numéro de séquence étendu". Cela peut être complexe. Dans le cas extrême, il peut n'y avoir pas de façon significative de traduire les rapports de réception, de sorte que le traducteur PEUT ne passer aucun rapport de réception du tout ou seulement un rapport de synthèse fondé sur sa propre réception. La règle générale est de faire ce qui a un sens pour une traduction particulière.

Un traducteur n'exige pas un identifiant de SSRC de lui-même, mais PEUT choisir d'en allouer un pour les besoins de l'envoi des rapports sur ce qu'il a reçu. Ceux-ci seraient envoyés à tous les nuages connectés, chacun correspondant à la traduction du flux de données tel qu'envoyé à ce nuage, car les rapports de réception sont normalement envoyés en diffusion groupée à tous les participants.

SDES :
Les traducteurs transmettent normalement sans changement les informations de SDES qu'ils reçoivent d'un nuage aux autres, mais PEUVENT, par exemple, décider de filtrer les informations de SDES non CNAME si la bande passante est limitée. Les CNAME DOIVENT être transmis pour permettre à la détection de collision d'identifiant de SSRC de fonctionner. Un traducteur qui génère ses propres paquets RR DOIT envoyer les informations CNAME de SDES sur lui-même aux mêmes nuages que ceux auxquels il envoie ces paquets RR.

BYE :
Les traducteurs transmettent les paquets BYE inchangés. Un traducteur qui est sur le point de cesser de transmettre les paquets DEVRAIT envoyer un paquet BYE à chaque nuage connecté contenant tous les identifiants de SSRC qui étaient précédemment transmis à ce nuage, y compris le propre identifiant de SSRC du traducteur si il envoyait des rapports en son nom propre.

APP : Les traducteurs transmettent les paquets APP inchangés.

7.3 Traitement de RTCP dans les mixeurs

Comme un mixeur génère de lui-même un nouveau flux de données, il ne passe pas du tout de paquet SR ou RR et génère à la place de nouvelles informations pour les deux côtés.

Informations d'envoyeur SR :
Un mixeur ne passe pas les informations d'envoyeur provenant des sources qu'il mixe parce que les caractéristiques des flux de source sont perdues dans le mixage. Comme source de synchronisation, le mixeur DEVRAIT générer ses propres paquets SR avec les informations d'envoyeur sur les flux de données mixés et les envoyer dans la même direction que le flux mixé.

Blocs de rapport de réception SR/RR :
Un mixeur génère ses propres rapports de réception pour les sources dans chaque nuage et ne les envoie que dans le même nuage. Il NE DOIT PAS envoyer ces rapports de réception aux autres nuages et NE DOIT PAS transmettre de rapports de réception provenant d'un nuage aux autres parce que là les sources ne seraient pas les SSRC (seulement les CSRC).

SDES :
Les mixeurs transmettent normalement sans changement les informations de SDES qu'ils reçoivent d'un nuage aux autres, mais PEUVENT, par exemple, décider de filtrer les informations de SDES non CNAME si la bande passante est limitée. Les CNAME DOIVENT être transmis pour permettre à la détection de collision d'identifiant de SSRC de fonctionner. (Un identifiant dans une liste de CSRC générée par un mixeur pourrait entrer en collision avec un identifiant de SSRC généré par un système d'extrémité.) Un mixeur DOIT envoyer des informations de CNAME SDES sur lui-même aux mêmes nuages que ceux auxquels il envoie les paquets SR ou RR.

Comme les mixeurs ne transmettent pas de paquets SR ou RR, ils vont normalement extraire les paquets SDES d'un paquet RTCP composé. Pour minimiser la redondance, les tronçons provenant des paquets de SDES PEUVENT être agrégés en un seul paquet de SDES qui est ensuite empilé sur un paquet SR ou RR généré par le mixeur. Un mixeur qui agrège les paquets de SDES utilisera plus de bande passante RTCP qu'une source individuelle parce que les paquets composés seront plus longs, mais c'est approprié car le mixeur représente plusieurs sources. De même, un mixeur qui passe des paquets SDES lorsque ils sont reçus va transmettre les paquets RTCP à un taux supérieur à celui d'une source simple, mais la encore ceci est correct car les paquets viennent de multiples sources. Le taux de paquet RTCP peut être différent de chaque côté du mixeur.

Un mixeur qui n'insère pas d'identifiants de CSRC PEUT aussi ne pas transmettre les CNAME de SDES. Dans ce cas, les espaces d'identifiant de SSRC dans les deux nuages sont indépendants. Comme mentionné plus haut, ce mode de fonctionnement crée le danger que des boucles ne soient pas détectées.

BYE :
Les mixeurs DOIVENT transmettre les paquets BYE. Un mixeur qui est sur le point de cesser de transmettre des paquets DEVRAIT envoyer un paquet BYE à chaque nuage connecté, contenant tous les identifiants de SSRC qui étaient précédemment transmis à ce nuage, y compris le propre identifiant de SSRC du mixeur si il envoyait des rapports de lui-même.

APP : Le traitement des paquets APP par les mixeurs est spécifique de l'application.

7.4 Mixeurs en cascade

Une session RTP peut impliquer une collection de mixeurs et traducteurs comme montré à la Figure 3. Si deux mixeurs sont placés en cascade, comme M2 et M3 sur la figure, les paquets reçus par un mixeur peuvent avoir déjà été mixés et peuvent comporter une liste de CSRC avec plusieurs identifiants. Le second mixeur DEVRAIT construire la liste de CSRC pour le paquet sortant en utilisant les identifiants de CSRC à partir des paquets d'entrée déjà mixés et les identifiants de SSRC provenant des paquets d'entrée non mixés. Cela est montré dans l'arc sortant du mixeur M3 marqué M3:89(64,45) sur la figure. Comme dans le cas de mixeurs qui ne sont pas en cascade, si la liste de CSRC résultante a plus de 15 identifiants, le reste ne peut pas être inclus.

8. Allocation et utilisation de l'identifiant de SSRC

L'identifiant de SSRC porté dans l'en-tête RTP et dans divers champs de paquets RTCP est un nombre aléatoire de 32 bits dont il est exigé qu'il soit unique au monde au sein d'une session RTP. Il est crucial que le nombre soit choisi avec soin afin que les participants sur le même réseau ou commençant au même moment ne risquent pas de choisir le même nombre.

Il n'est pas suffisant d'utiliser l'adresse du réseau local (comme une adresse IPv4) comme identifiant parce que l'adresse peut n'être pas unique. Comme les traducteurs et mixeurs RTP permettent l'interopération parmi de multiples réseaux avec différents espaces d'adresse, les schémas d'allocation des adresses au sein de deux espaces peuvent résulter en un taux de collision bien plus élevé que celui qui surviendrait avec une allocation aléatoire.

Plusieurs sources fonctionnant sur un seul hôte entreraient aussi en conflit.

Il n'est pas suffisant non plus d'obtenir un identifiant de SSRC simplement en invoquant random() sans initialiser

soigneusement l'état. Un exemple de la façon de générer un identifiant aléatoire est présenté à l'Appendice A.6.

8.1 Probabilité de collision

Comme les identifiants sont choisis de façon aléatoire, il est possible que deux sources ou plus choisissent le même nombre. Des collisions surviennent avec la plus forte probabilité lorsque toutes les sources débutent simultanément, par exemple lorsque elles sont déclanchées automatiquement par un événement de gestion de session. Si N est le nombre de sources et L la longueur de l'identifiant (ici, 32 bits) la probabilité que deux sources prennent indépendamment la même valeur peut être approximée pour un grand N [26] par 1 - exp(-N**2 / 2**(L+1)). Pour N = 1000, la probabilité est d'environ 10**-4.

La probabilité normale de collision est bien plus faible que le pire cas de ci-dessus. Lorsque une nouvelle source se joint à une session RTP dans laquelle toutes les autres sources ont déjà des identifiants univoques, la probabilité de collision est juste la fraction des nombres utilisés dans l'espace. À nouveau, si N est le nombre de sources et L la longueur de l'identifiant, la probabilité de collision est N / 2**L. Pour N = 1000, la probabilité est d'environ 2*10**-7.

La probabilité de collision est encore réduite par l'opportunité qu'une nouvelle source reçoive des paquets provenant d'autres participants avant l'envoi de son premier paquet (de données ou de contrôle). Si la nouvelle source garde trace des autres participants (par l'identifiant de SSRC) la nouvelle source peut, avant de transmettre son premier paquet, vérifier que son identifiant n'est en conflit avec aucun de ceux qui ont été reçus, ou alors de choisir à nouveau.

8.2 Résolution de collision et détection de boucle

Bien que la probabilité de collision d'identifiant de SSRC soit faible, toutes les mises en œuvre de RTP DOIVENT être prêtes à détecter les collisions et à prendre les mesures appropriées pour les résoudre. Si une source découvre à tout moment qu'une autre source utilise le même identifiant de SSRC qu'elle, elle DOIT envoyer un paquet RTCP BYE pour le vieil identifiant et en choisir un autre de façon aléatoire. (Comme expliqué ci-dessous, cette étape n'est franchie qu'une seule fois en cas de boucle.) Si un receveur découvre que deux autres sources sont en collision, il PEUT garder les paquets de l'une et éliminer les paquets de l'autre lorsque cela peut être détecté par des adresses de transport de source ou des CNAME différents. Les deux sources sont supposées résoudre la collision de telle sorte que la situation ne dure pas.

Comme les identifiants de SSRC aléatoires sont uniques au monde pour chaque session RTP, ils peuvent aussi être utilisés pour détecter les boucles qui pourraient être introduites par les mixeurs ou traducteurs. Un boucle cause une duplication des donnes et des informations de contrôle, non modifiées ou éventuellement mixées, comme dans les exemples suivants :

- Un traducteur peut transmettre de façon incorrecte un paquet au groupe de diffusion groupée même dont il a reçu le paquet, soit directement soit à travers une chaîne de traducteurs. Dans ce cas, le même paquet apparaît plusieurs fois, provenant de sources réseau différentes.

- Deux traducteurs réglés incorrectement en parallèle, c'est-à-dire, avec les mêmes groupes de diffusion groupée des deux côtés, transmettraient tous deux les paquets d'un groupe de diffusion groupée à l'autre. Les traducteurs unidirectionnels produiraient deux copies ; les traducteurs bidirectionnels formeraient une boucle.

- Un mixeur peut clore une boucle en envoyant des paquets à la même destination de transport sur laquelle il les reçoit, directement ou à travers un autre mixeur ou traducteur. Dans ce cas une source peut apparaître à la fois comme un SSRC sur un paquet de données et un CSRC dans un paquet de données mixé.

Une source peut découvrir que ses propres paquets sont en boucle, ou que les paquets provenant d'une autre source sont en boucle (boucle de tiers).

Les boucles et les collisions dans le choix aléatoire d'un identifiant de source résultent tous deux en ce que des paquets arrivent avec le même identifiant de SSRC mais une adresse différente de transport de source, qui peut être celle du système d'extrémité qui a généré le paquet ou celle d'un système intermédiaire.

Donc, si une source change son adresse de transport de source, elle PEUT aussi choisir un nouvel identifiant de SSRC pour éviter d'être interprétée comme une source en boucle. (Ce n'est pas DOIT parce que dans certaines applications de RTP on peut s'attendre à ce que les sources changent d'adresse durant une session.) Noter que si un traducteur redémarre et par conséquent change d'adresse de transport de source (par exemple, il change le numéro d'accès de source UDP) sur laquelle il transmet les paquets, tous ces paquets vont alors apparaître aux receveurs comme étant en boucle parce que les identifiants de SSRC sont appliqués par la source originale et ne vont pas changer. Ce problème peut être évité en gardant

l'adresse de transport de source fixe à travers les redémarrages, mais dans tous les cas il sera résolu après une fin de temporisation chez les receveurs.

Les boucles ou les collisions survenant sur le côté éloigné d'un traducteur ou mixeur ne peuvent pas être détectés en utilisant l'adresse de transport de source si toutes les copies des paquets passent à travers le traducteur ou mixeur, cependant, les collisions peuvent encore être détectées lorsque des tronçons provenant de deux paquets de SDES RTCP contiennent le même identifiant de SSRC mais des CNAME différents.

Pour détecter et résoudre ces conflits, une mise en œuvre de RTP DOIT inclure un algorithme similaire à celui décrit ci-dessous, bien que la mise en œuvre PUISSE choisir une politique différente pour empêcher les paquets d'entrer en collision avec les sources tierces. L'algorithme décrit ci-dessous ignore les paquets provenant d'une nouvelle source ou les boucles qui entrent en collision avec une source établie. Il résout les collisions avec le propre identifiant de SSRC du participant en envoyant un BYE RTCP pour l'ancien identifiant et en choisit un nouveau. Cependant, Lorsque la collision a été produite par une boucle des propres paquets du participant, l'algorithme va choisir un nouvel identifiant une seule fois et ensuite ignorera les paquets provenant de l'adresse de transport de la source en boucle. Ceci est exigé pour éviter une inondation de paquets BYE.

Cet algorithme exige de tenir un tableau indexé par identifiants de source et contenant les adresses de transport de source provenant du premier paquet RTP et du premier paquet RTCP reçus avec cet identifiant, ainsi que les autres états pour cette source. Deux adresses de transport de source sont donc requises, par exemple, les numéros d'accès UDP de source peuvent être différents sur les paquets RTP et les paquets RTCP. Cependant, on peut supposer que l'adresse réseau est la même dans les deux adresses de transport de source.

Chaque identifiant de SSRC ou de CSRC reçu dans un paquet RTP ou RTCP est recherché dans le tableau des identifiants de source afin de traiter ces informations de données ou de contrôle. L'adresse de transport de source provenant du paquet est comparée à l'adresse de transport de source correspondante dans le tableau pour détecter une boucle ou collision si elles ne correspondent pas. Pour les paquets de contrôle, chaque élément avec son propre identifiant de SSRC , par exemple un tronçon de SDES, exige une recherche distincte. (L'identifiant de SSRC dans un bloc de rapport de réception est une exception parce que il identifie une source entendue par le rapporteur, et cet identifiant de SSRC est sans relation avec l'adresse de transport de source du paquet RTCP envoyé par le rapporteur.) Si le SSRC ou CSRC n'est pas trouvé, une nouvelle entrée est créée. Ces entrées de tableau sont supprimées lorsque un paquet BYE RTCP est reçu avec l'identifiant de SSRC correspondant et qu'il est validé par une adresse de transport de source correspondante, ou après qu'aucun paquet n'est arrivé pendant un temps relativement long (voir au paragraphe 6.2.1).

Noter que si deux sources sur le même hôte émettent avec le même identifiant de source au moment où un receveur commence à fonctionner, il serait possible que le premier paquet RTP reçu provienne d'une des sources alors que le premier paquet RTCP reçu provient de l'autre. Cela serait cause que les mauvaises informations RTCP seraient associées aux données RTP, mais cette situation devrait être suffisamment rare et sans conséquences pour qu'on puisse l'oublier.
Afin de traquer les boucles des propres paquets de données du participant, la mise en œuvre DOIT aussi tenir une liste distincte des adresses de transport de source (pas les identifiants) qui se sont trouvées en conflit. Comme dans le tableau des identifiants de source, deux adresses de transport de source DOIVENT être conservées pour garder trace séparément des paquets RTP et RTCP en conflit. Noter que la liste des adresses en conflit devrait être courte, et habituellement vide. Chaque élément de cette liste mémorise les adresses de source plus l'heure à laquelle le plus récent paquet en conflit a été reçu. Un élément PEUT être retiré de la liste lorsque aucun paquet en conflit n'est arrivé de cette source pendant un temps de l'ordre de dix intervalles de rapport RTCP (voir au paragraphe 6.2).

Pour l'algorithme indiqué, on suppose que le propre identifiant de source du participant et l'état sont inclus dans le tableau des identifiant de source. L'algorithme pourrait être restructuré pour faire d'abord une comparaison distincte par rapport au propre identifiant de source du participant.

```
si (identifiant de SSRC ou de CSRC n'est pas trouvé dans le tableau des identifiants de source) {
    créer une nouvelle entrée mémorisant les données ou l'adresse de transport de source de contrôle, le SSRC ou CSRC
        et l'autre état ;
}
                                        /* L'identifiant est trouvé dans le tableau */

autrement si (l'entrée de tableau a été créée à réception d'un paquet de contrôle et que c'est le premier paquet de données
                ou vice versa) {
    mémoriser l'adresse de transport de source à partir de ce paquet;
}
autrement si (l'adresse de transport de source provenant du paquet ne correspond pas à celle sauvegardée dans l'entrée
```

```
                du tableau pour cet identifiant) {
                                        /* Une collision d'identifiant ou une boucle est indiquée */

        si (l'identifiant de source n'est pas celui du participant) {
                                        /* Étape FACULTATIVE de compteur d'erreurs */
            si (l'identifiant de source provient d'un tronçon de SDES RTCP contenant un élément CNAME qui diffère du
                CNAME dans l'entrée du tableau) {
                compter une collision avec un tiers ;
            } autrement {
                compter une boucle de tiers ;
            }
            interrompre le traitement du paquet de données ou de l'élément de contrôle ;
                                        /* PEUT choisir une politique différente pour garder une nouvelle source */
        }
                                        /* Une collision ou boucle des propres paquets du participant */

    autrement si (l'adresse de transport de source est trouvée dans la liste des données en conflit ou des adresses de transport
                de source de contrôle) {
                                        /* Étape FACULTATIVE de compteur d'erreurs */
            si (l'identifiant de source n'est pas celui d'un tronçon de SDES RTCP contenant un élément CNAME ou si
                CNAME est celui du participant) {
                compter l'occurrence de propre trafic en boucle ;
            }
            marquer l'heure en cours dans l'entrée de liste d'adresse en conflit ;
            interrompre le traitement du paquet de données ou de l'élément de contrôle ;
        }
                                        /* Nouvelle collision, changer l'identifiant de SSRC */

autrement {
            enregistrer l'occurrence d'une collision;
        créer une nouvelle entrée dans la liste des données en conflit ou des adresses de transport de source de contrôle et
        marquer l'heure actuelle ;
        envoyer un paquet BYE RTCP avec le vieil identifiant de SSRC ;
          choisir un nouvel identifiant de SSRC ;
        créer une nouvelle entrée dans le tableau d'identifiant de source avec le vieil SSRC plus l'adresse de transport de
        source provenant des données ou du paquet de contrôle en cours de traitement ;
    }
}
```

Dans cet algorithme, les paquets provenant d'une nouvelle adresse de source en conflit seront ignorés et les paquets provenant de l'adresse de source d'origine seront conservés. Si aucun paquet n'arrive de la source d'origine pendant une période étendue, l'entrée du tableau va avoir une fin de temporisation et la nouvelle source sera capable de prendre le dessus. Cela pourrait survenir si la source originale détecte la collision et passe à un nouvel identifiant de source, mais dans le cas usuel, un paquet BYE RTCP sera reçu de la source d'origine pour supprimer l'état sans avoir à attendre une fin de temporisation.

Si l'adresse de source d'origine a été reçue à travers un mixeur (c'est-à-dire, apprise comme CSRC) et qu'ultérieurement la même source est reçue directement, le receveur sera bien avisé de passer à la nouvelle adresse de source à moins que d'autres sources dans le mixage ne soient perdues. De plus, pour les applications telles que la téléphonie dans laquelle certaines sources comme des entités mobiles peuvent changer d'adresse durant le cours d'une session RTP, la mise en œuvre RTP DEVRAIT modifier l'algorithme de détection de collision de façon à accepter les paquets provenant de la nouvelle adresse de transport de source. Pour se garder du basculage entre les adresses si une authentique collision survenait, l'algorithme DEVRAIT inclure des moyens de détecter ce cas et éviter de commuter.

Lorsque un nouvel identifiant de SSRC est choisi du fait d'une collision, le candidat identifiant DEVRAIT d'abord être cherché dans un tableau d'identifiants de source pour voir si il a déjà été utilisé par une autre source. Si il l'est, un autre candidat DOIT être généré et le processus répété.

Une boucle des paquets de données sur une destination de diffusion groupée peut causer un débordement sévère dans le réseau. Tous les mixeurs et traducteurs DOIVENT mettre en œuvre un algorithme de détection de boucle comme celui ci-

dessus afin qu'ils puissent casser les boucles. Cela devrait limiter les excès de trafic à pas plus d'une copie dupliquée du trafic original, ce qui peut permettre à la session de continuer afin que la cause de la boucle soit trouvée et réparée. Cependant, dans les cas extrêmes où un mixeur ou traducteur ne casse pas correctement la boucle et qu'il en résulte de forts niveaux de trafic, il peut être nécessaire que les systèmes d'extrémité cessent entièrement de transmettre des paquets de données ou de contrôle. Cette décision peut dépendre de l'application. Une condition d'erreur DEVRAIT être indiquée comme approprié. La transmission PEUT être tentée à nouveau périodiquement après une durée longue aléatoire (de l'ordre de plusieurs minutes).

8.3 Utilisation avec des codages en couches

Pour les codages en couche transmis sur des sessions RTP séparées (voir au paragraphe 2.4) un seul espace d'identifiant de SSRC DEVRAIT être utilisé à travers les sessions de toutes les couches et la couche cœur (de base) DEVRAIT être utilisée pour l'allocation d'identifiant de SSRC et la résolution de collision. Lorsque une source découvre qu'elle a eu une collision, elle transmet un paquet BYE RTCP seulement sur la couche de base mais change l'identifiant de SSRC à la nouvelle valeur dans toutes les couches.

9. Sécurité

Les protocoles de couches inférieures peuvent éventuellement fournir tous les services de sécurité qui pourraient être souhaités pour les applications de RTP, incluant l'authentification, l'intégrité, et la confidentialité. Ces services ont été spécifiés pour IP dans [27]. Comme les applications audio et vidéo initiales qui utilisaient RTP avaient besoin d'un service de confidentialité avant que de tels services ne soient disponibles pour la couche IP, le service de confidentialité décrit dans le paragraphe qui suit avait été défini pour être utilisé avec RTP et RTCP. Cette description est incluse ici pour codifier les pratiques existantes. De nouvelles applications de RTP PEUVENT mettre en œuvre ce service de confidentialité spécifique de RTP pour la rétro compatibilité, et/ou elles PEUVENT mettre en œuvre d'autres services de sécurité. La redondance sur le protocole RTP pour ce service de confidentialité est faible, aussi la pénalisation sera minimale si ce service est rendu obsolète par d'autres services à l'avenir.

Autrement, d'autres services, d'autres mises en œuvre de services et d'autres algorithmes peuvent être définis pour RTP à l'avenir. En particulier, un profil RTP appelé protocole sécurisé de transport en temps réel (SRTP, *Secure Real-time Transport Protocol*) [28] est en cours de développement pour fournir la confidentialité de la charge utile RTP tout en laissant l'en-tête RTP en clair afin que les algorithmes de compression d'en-tête de niveau liaison puissent encore fonctionner. Il est prévu que SRTP soit le bon choix pour de nombreuses applications. SRTP se fonde sur la norme de chiffrement évolué (AES, *Advanced Encryption Standard*) et fournit une plus forte sécurité que le service décrit ici. Nous ne prétendons pas que les méthodes présentées ici sont appropriées pour un besoin de sécurité particulier. Un profil peut spécifier quels services et algorithmes devraient être offerts par les applications, et peut fournir des lignes directrices quant à leur utilisation appropriée.

La distribution des clés et des certificats sort du domaine d'application de ce document.

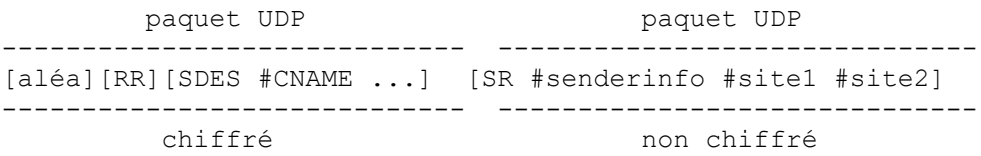
9.1 Confidentialité

Confidentialité signifie que seul le ou les receveurs prévus peuvent décoder les paquets reçus ; pour les autres, le paquet ne contient aucune information utile. La confidentialité du contenu est réalisée par le chiffrement.

Lorsque on désire chiffrer RTP ou RTCP conformément à la méthode spécifiée dans cette section, tous les octets qui vont être encapsulés pour la transmission dans un seul paquet de couche inférieure sont chiffrés comme une unité. Pour RTCP, un nombre aléatoire de 32 bits recalculé pour chaque unité DOIT être ajouté en tête de l'unité avant le chiffrement. Pour RTP, aucun préfixe n'est mis en tête ; à la place, les champs Numéro de séquence et Horodatage sont initialisés avec un décalage aléatoire. Ceci est considéré comme une valenr d'initialisation (IV) à cause des pauvres propriétés d'aléa. De plus, si le champ suivant, le SSRC, peut être manipulé par un ennemi, il y a une faiblesse supplémentaire de la méthode de chiffrement.

Pour RTCP, une mise en œuvre PEUT discriminer les paquets RTCP individuels dans un paquet RTCP composé en deux paquets RTCP composés séparés, un à chiffrer et un à envoyer en clair . Par exemple, les informations de SDES peuvent être chiffrées alors que les rapports de réception sont envoyés en clair pour s'accommoder des surveillants tiers qui ne sont pas au courant de la clé de chiffrement. Dans cet exemple, décrit à la Figure 4, les informations de SDES DOIVENT être ajoutées à un paquet RR sans rapport (et sans nombre aléatoire) pour satisfaire à l'exigence que tous les paquets RTCP

composés commencent par un paquet SR ou RR. L'élément CNAME de SDES est exigé soit dans le paquet chiffré soit dans le non chiffré, mais pas dans les deux. Les mêmes informations de SDES NE DEVRAIENT PAS être portées dans les deux paquets car cela peut compromettre le chiffrement.



: identifiant de SSRC

Figure 4 : Paquets RTCP chiffrés et non chiffrés

La présence du chiffrement et l'utilisation de la clé correcte sont confirmées par le receveur au moyen de la vérification de validité de l'en-tête et de la charge utile. Des exemples de telles vérifications de validité pour les en-têtes RTP et RTCP sont donnés aux Appendices A.1 et A.2.

Pour être cohérent avec les mises en œuvre existantes de la spécification initiale de RTP dans la RFC 1889, l'algorithme de chiffrement par défaut est l'algorithme de la norme de chiffrement des données (DES, *Data Encryption Standard*) en mode d'enchaînement de bloc de chiffrement (CBD, *cipher block chaining*) comme décrit au paragraphe 1.1 de la RFC 1423 [29], sauf que le bourrage à un multiple de 8 octets est indiqué comme décrit pour le bit P au paragraphe 5.1. Le vecteur d'initialisation est zéro parce que les valeurs aléatoires sont fournies dans l'en-tête RTP ou par le préfixe aléatoire pour les paquets composés RTCP. Pour des détails sur l'utilisation des vecteurs d'initialisation CBC, voir [30].

Les mises en œuvre qui prennent en charge la méthode de chiffrement spécifiée ici DEVRAIENT toujours accepter l'algorithme DES en mode CBC comme chiffrement par défaut pour cette méthode pour maximiser l'interopérabilité. Cette méthode a été choisie parce qu'il a été démontré qu'elle est d'utilisation facile et pratique dans les outils expérimentaux audio et vidéo qui fonctionnent sur l'Internet. Cependant, il a été trouvé depuis que DES est trop facile à casser.

Il est RECOMMANDÉ que des algorithmes de chiffrement plus forts tels que Triple-DES soient utilisés à la place de l'algorithme par défaut. De plus, le mode CBC sécurisé exige que le premier bloc de chaque paquet soit traité par l'opérateur OUX avec un vecteur d'initialisation aléatoire indépendant de la même taille que celle du bloc de chiffrement. Pour RTCP, ceci est (partiellement) réalisé par l'ajout en tête de chaque paquet d'un nombre aléatoire de 32 bits, choisi indépendamment pour chaque paquet. Pour RTP, l'horodatage et le numéro de séquence commencent à partir de valeurs aléatoires, mais les paquets suivants ne seront pas rendus aléatoires de façon indépendante. Il devrait être noté que l'aléa dans les deux cas (RTP et RTCP) est limité. Des applications de haute sécurité DEVRAIENT considérer d'autres moyens de protection, plus conventionnels. D'autres algorithmes de chiffrement PEUVENT être spécifiés de façon dynamique pour une session par des moyens non RTP. En particulier, le profil SRTP [28] fondé sur AES est en cours de développement pour prendre en compte les soucis connus de manipulations de libellé et de libellé CBC, et sera le choix correct à l'avenir.

Comme solution de remplacement au chiffrement au niveau IP ou au niveau RTP comme décrit ci-dessus, les profils PEUVENT définir des types de charge utile supplémentaires pour les codages chiffrés. Ces codages DOIVENT spécifier comment le bourrage et les autres aspects du chiffrement sont à traiter. Cette méthode permet de ne chiffrer que les données tout en laissant les en-têtes en clair pour les applications où cela est souhaité. Il peut être particulièrement utile pour les appareils matériels qui vont traiter à la fois le déchiffrement et le décodage. Il est aussi précieux pour les applications où la compression de niveau liaison des en-têtes RTP et de couche inférieure est désirée et où la confidentialité de la charge utile (mais pas des adresses) est suffisante car le chiffrement des en-têtes empêche la compression.

9.2 Authentification et intégrité du message

Les services d'authentification et d'intégrité du message ne sont pas définis au niveau de RTP car ces services ne seraient pas directement réalisables sans une infrastructure de gestion de clés. Il est prévu que des services d'authentification et d'intégrité seront fournis par les protocoles de couche inférieure.

10. Contrôle de l'encombrement

Tous les protocoles de transport utilisés sur l'Internet ont besoin de s'occuper d'une façon ou d'une autre du contrôle de l'encombrement [31]. RTP n'y fait pas exception, mais comme les données transportées sur RTP sont souvent inélastiques

(générées à un débit fixe ou contrôlé), les moyens de contrôler l'encombrement dans RTP peuvent être assez différents de ceux d'autres protocoles de transport tels que TCP. Dans un sens, l'inélasticité réduit le risque d'encombrement parce que le flux RTP ne va pas s'étendre pour consommer toute la bande passante disponible comme le peut un flux TCP. Cependant, l'inélasticité signifie aussi que le flux RTP ne peut pas réduire arbitrairement sa charge sur le réseau pour éliminer l'encombrement lorsque il survient.

Comme RTP peut être utilisé pour une grande variété d'applications dans de nombreux contextes différents, il n'y a pas un unique mécanisme de contrôle de l'encombrement qui puisse tout faire. Donc, le contrôle de l'encombrement DEVRAIT être défini dans chaque profil RTP comme approprié. Pour certains profils, il peut être suffisant d'inclure une déclaration d'applicabilité restreignant l'utilisation de ce profil aux environnements dans lesquels l'encombrement est évité par l'ingénierie. Pour les autres profils, des méthodes spécifiques, telles que l'adaptation du débit des données fondée sur les retours RTCP peuvent être requises.

11. RTP sur les protocoles de réseau et de transport

La présente section décrit les questions spécifique du transport des paquets RTP au sein de protocoles de réseau et de transport particuliers. Les règles suivantes s'appliquent sauf si elles sont supplantées par des définitions spécifiques de protocole en dehors de la présente spécification.

RTP s'appuie sur le ou les protocoles sous-jacents pour fournir le démultiplexage des données RTP et des flux de contrôle RTCP. Pour UDP et les protocoles similaires, RTP DEVRAIT utiliser un numéro pair d'accès de destination et le flux RTCP correspondant DEVRAIT utiliser le numéro d'accès de destination immédiatement supérieur (impair). Pour les applications qui prennent un seul numéro d'accès comme paramètre et déduisent la paire d'accès RTP et RTCP de ce numéro, si un numéro impair est fourni, alors l'application DEVRAIT remplacer ce numéro par le prochain nombre inférieur (pair) à utiliser comme base de la paire d'accès. Pour les applications dans lesquelles les numéros d'accès de destination RTP et RTCP sont spécifiés via des paramètres séparés explicites (utilisant un protocole de signalisation ou d'autres moyens) l'application PEUT négliger les restrictions sur les numéros d'accès pair/impair consécutifs bien que l'utilisation d'une paire pair/impair soit toujours conseillée. Les numéros d'accès RTP et RTCP NE DOIVENT PAS être les mêmes car RTP s'appuie sur les numéros d'accès pour démultiplexer les données RTP et les flux de contrôle RTCP.

Dans une session en envoi individuel, les deux participants ont besoin d'identifier une paire d'accès pour recevoir les paquets RTP et RTCP. Les deux participants PEUVENT utiliser la même paire d'accès. Un participant NE DOIT PAS supposer que l'accès de source du paquet RTP ou RTCP entrant peut être utilisé comme accès de destination pour les paquets RTP ou RTCP sortants. Lorsque des paquets de données RTP sont envoyés dans les deux directions, les paquets SR RTPC de chaque participant DOIVENT être envoyés à l'accès que l'autre participant a spécifié pour la réception de RTCP. Les paquets SR RTCP combinent des informations d'envoyeur pour les données sortantes et des informations de rapport de réception pour les données entrantes. Si un côté n'envoie pas activement des données (voir au paragraphe 6.4), un paquet RR RTCP est envoyé à la place.

Il est RECOMMANDÉ que les applications de codage en couches (voir au paragraphe 2.4) utilisent un ensemble de numéros d'accès contigus. Les numéros d'accès DOIVENT être distincts à cause d'une déficience largement répandue dans les systèmes d'exploitation existants qui empêche l'utilisation du même accès avec plusieurs adresses de diffusion groupée, et pour l'envoi individuel, il n'y a qu'une seule adresse permise. Et donc pour la couche n, l'accès des données est P + 2n, et l'accès de contrôle est P + 2n + 1. Lorsque la diffusion groupée IP est utilisée, les adresses DOIVENT aussi être distinctes parce que l'acheminement de diffusion groupée et les membres du groupe sont gérés selon une granularité d'adresse. Cependant, l'allocation d'adresses de diffusion groupée IP contiguës ne peut être supposée parce que certains groupes peuvent exiger des portées différentes et peuvent donc se voir allouer des gammes d'adresses différentes.

Le paragraphe précédent contredit la spécification de SPD, la RFC 2327 [15], qui dit qu'il est illégal aussi bien pour les adresses multiples que pour les accès multiples d'être spécifiés dans la même description de session parce que l'association des adresses et des accès pourrait être ambiguë. Il est prévu que cette restriction soit assouplie dans une révision de la RFC 2327 pour permettre à un nombre égal d'adresses et d'accès d'être spécifiés avec une correspondance univoque implicite.

Les paquets de données RTP ne contiennent pas de champ de longueur ou autre délimitation, et RTP s'appuie donc sur le ou les protocoles sous-jacents pour fournir une indication de longueur. La longueur maximum des paquets RTP est limitée seulement par les protocoles sous-jacents.

Si les paquets RTP doivent être portés dans un protocole sous-jacent qui fournit l'abstraction d'un flux continu d'octets

plutôt que des messages (paquets), une encapsulation des paquets RTP DOIT être définie pour fournir un mécanisme de tramage. Le tramage est aussi nécessaire si le protocole sous-jacent contient un bourrage et que donc l'étendue de la charge utile RTP ne peut être déterminée. Le mécanisme de tramage n'est pas défini ici.

Un profil PEUT spécifier une méthode de tramage à utiliser même lorsque RTP est porté dans des protocoles qui fournissent bien le tramage afin de permettre de porter plusieurs paquets RTP dans une unité de données de protocole de couche inférieure, telle qu'un paquet UDP. Porter plusieurs paquets RTP sur un seul paquet réseau ou transport réduit la redondance d'en-tête et peut simplifier la synchronisation entre différents flux.

12. Résumé des constantes de protocole

La présente section contient une liste résumé des constantes définies dans cette spécification.

Les constantes de type de charge utile RTP (PT, *payload type*) sont définies dans les profils plutôt que dans ce document. Cependant, l'octet de l'en-tête RTP qui contient le ou les bits marqueurs et type de charge utile DOIT éviter les valeurs réservées 200 et 201 (en décimal) pour distinguer les paquets RTP des types de paquet SR et RR RTCP pour la procédure de validation d'en-tête décrite à l'Appendice A.1. Pour la définition standard d'un bit marqueur et d'un champ à 7 bits de type de charge utile tels que figurant dans la présente spécification, cette restriction signifie que les types de charge utile 72 et 73 sont réservés.

12.1 Types de paquet RTCP

abréviation	nom	valeur
SR	rapport d'envoyeur	200
RR	rapport de receveur	201
SDES	description de source	202
BYE	au revoir	203
APP	défini par l'application	204

Ces valeurs de type ont été choisies dans la gamme 200 à 204 pour améliorer la vérification de validité des en-têtes de paquets RTCP comparés aux paquets RTP ou aux autres paquets sans rapport avec eux. Lorsque le champ Type de paquet RTCP est comparé à l'octet correspondant de l'en-tête RTP, cette gamme correspond au bit marqueur mis à 1 (ce qui n'est pas habituel dans les paquets de données) et au bit de poids fort du champ Type de charge utile standard mis à 1 (car le type de charge utile statique est normalement défini dans la moitié de moindre poids). Cette gamme a aussi été choisie comme étant numériquement à une certaine distance de 0 et 255 car les schémas tout à zéro et tout à un sont courants dans les schémas de données.

Comme tous les paquets RTCP composés DOIVENT commencer par SR ou RR, ces codes ont été choisis comme une paire pair/impair pour permettre à la vérification de validité RTCP de tester le nombre maximum de bits avec gabarit et valeur.

Des types supplémentaires de paquet RTCP peuvent être enregistrés auprès de l'IANA (voir la Section 15).

12.2 Types de SDES

abréviation	nom	valeur
END	fin de liste SDES	0
CNAME	nom canonique	1
NAME	nom d'utilisateur	2
EMAIL	adresse de messagerie électronique de l'utilisateur	3
PHONE	numéro de téléphone de l'utilisateur	4
LOC	localisation géographique de l'utilisateur	5
TOOL	nom de l'application ou de l'outil	6
NOTE	remarque sur la source	7
PRIV	extensions privées	8

Des types de SDES supplémentaires peuvent être enregistrés auprès de l'IANA (voir la Section 15).

13. Profils RTP et spécifications de format de charge utile

Une spécification complète de RTP pour une application particulière exigera un ou plusieurs documents d'accompagnement de deux types décrits ici : profils, et spécification de format de charge utile.

RTP peut être utilisé pour diverses applications qui ont des exigences quelque peu différentes. La souplesse pour s'adapter à ces exigences est fournie en permettant des choix multiples dans la spécification principale du protocole, puis en sélectionnant les choix appropriés ou en définissant des extensions pour un environnement et classe d'applications particuliers dans un document de profil distinct. Normalement, une application va fonctionner selon un seul profil dans une session RTP particulière, de sorte qu'il n'y a pas d'indication explicite au sein du protocole RTP lui-même sur le profil utilisé. Un profil pour des applications audio et vidéo se trouve dans le document d'accompagnement RFC 3551. Les profils sont normalement intitulés "Profil RTP pour ...".

Le second type de document d'accompagnement est une spécification de format de charge utile, qui définit comment un type particulier de données de charge utile, tel que de la vidéo codée en H.261, devrait être porté dans RTP. Ces documents sont normalement intitulés "Format de charge utile RTP pour codage audio/vidéo XYZ". Les formats de charge utile peuvent être utiles sous plusieurs profils et peuvent donc être définis indépendamment de tout profil particulier. Les documents de profil sont alors chargés d'allouer une transposition par défaut de ce format en une valeur de type de charge utile si nécessaire.

Dans la présente spécification, les éléments suivants ont été identifiés pour une éventuelle définition au sein d'un profil, mais cette liste n'a rien d'exhaustif :

en-tête de données RTP : l'octet qui dans l'en-tête de données RTP contient le bit marqueur et le champ Type de charge utile PEUT être redéfini par un profil pour convenir à des exigences différentes, par exemple avec plus ou moins de bits marqueurs (paragraphe 5.3).

types de charge utile : en supposant qu'un champ Type de charge utile soit inclus, le profil va habituellement définir un ensemble de formats de charge utile (par exemple, de codages du support) et une transposition statique par défaut de ces formats en valeurs de type de charge utile. Certains de ces formats de charge utile peuvent être définis par référence à des spécifications de format de charge utile distinctes. Pour chaque type de charge utile défini, le profil DOIT spécifier le débit d'horloge d'horodatage RTP à utiliser (paragraphe 5.1).

ajout d'en-tête de données RTP : des champs supplémentaires PEUVENT être ajoutés à l'en-tête fixe de données RTP si des fonctionnalités supplémentaires sont requises pour la classe d'applications du profil indépendamment du type de (paragraphe 5.3).

extensions d'en-tête de données RTP : le contenu des seize premiers bits de la structure d'extension d'en-tête de données RTP DOIT être défini si l'utilisation de ce mécanisme doit être permise dans ce profil pour des extensions spécifiques de la mise en œuvre (paragraphe 5.3.1).

types de paquet RTCP : de nouveaux types de paquet RTCP spécifiques de la classe d'application PEUVENT être définis et enregistrés auprès de l'IANA.

intervalle de rapport RTCP : un profil DEVRAIT spécifier que les valeurs suggérées au paragraphe 6.2 pour les constantes employées dans le calcul de l'intervalle de rapport RTCP seront utilisées. Ce sont la fraction de la bande passante de session RTCP, l'intervalle minimum de rapport, et le partage de la bande passante entre envoyeurs et receveurs. Un profil PEUT spécifier des valeurs de remplacement si elles ont été démontrées fonctionner de façon appropriée.

extension SR/RR : une section d'extension PEUT être définie pour les paquets SR et RR RTCP si il y a des informations supplémentaires qui devraient être rapportées régulièrement sur l'envoyeur ou les receveurs (paragraphe 6.4.3).

utilisation de SDES : le profil PEUT spécifier les priorités relatives pour les éléments de SDES RTCP à transmettre ou à exclure entièrement (paragraphe 6.3.9) ; une syntaxe ou sémantique de remplacement pour l'élément CNAME (paragraphe 6.5.1) ; le format de l'élément LOC (paragraphe 6.5.5) ; la sémantique et l'utilisation de l'élément NOTE (paragraphe 6.5.7) ; ou de nouveaux types d'éléments de SDES à enregistrer auprès de l'IANA.

sécurité : un profil PEUT spécifier quels services et algorithmes de sécurité devraient être offerts par les applications, et

PEUT donner des indications sur leur utilisation appropriée (Section 9).

transposition de chaîne à clé : un profil PEUT spécifier comment un mot ou phrase de passe fourni par l'utilisateur est transposé en une clé de chiffrement.

encombrement : un profil DEVRAIT spécifier le comportement de contrôle d'encombrement approprié pour ce profil.

protocole sous-jacent : l'utilisation d'un protocole de couche réseau ou transport sous-jacent particulier pour porter les paquets RTP PEUT être exigée.

transposition de transport : une transposition des adresses RTP et RTCP en adresses de niveau transport, par exemple, d'accès UDP, autre que la transposition standard définie à la Section 11, peut être spécifiée.

Encapsulation : une encapsulation des paquets RTP peut être définie pour permettre que plusieurs paquets de données RTP soient portés dans un paquet de couche inférieure ou de fournir le tramage sur des protocoles sous-jacents qui ne le font pas déjà (Section 11).

Il n'est pas prévu qu'un nouveau profil soit exigé pour chaque application. Au sein d'une classe d'application, il serait préférable d'étendre un profil existant plutôt que d'en faire un nouveau afin de faciliter l'interopération parmi les applications car chacune va normalement fonctionner avec un seul profil. Des extensions simples telles que la définition de valeurs supplémentaires de type de charge utile ou de types de paquet RTCP peuvent être réalisées par leur enregistrement auprès de l'IANA et la publication de leur description dans un addendum au profil ou dans une spécification de format de charge utile.

14. Considérations pour la sécurité

RTP souffre des mêmes problèmes de sécurité que les protocoles sous-jacents. Par exemple, un imposteur peut falsifier des adresses réseau de source ou de destination, ou changer l'en-tête ou la charge utile. Au sein de RTCP, les informations de CNAME et NAME peuvent être utilisées pour se faire passer pour un autre participant. De plus, RTP peut être envoyé via une diffusion groupée IP, ce qui ne donne aucun moyen direct à un envoyeur de connaître tous les receveurs des données envoyées et donc pas de mesure de la confidentialité. À tort ou à raison, les utilisateurs peuvent être plus sensibles aux questions de confidentialité avec une communication audio et vidéo qu'ils ne l'ont été avec des formes plus traditionnelles de communication réseau [33]. Donc, l'utilisation de mécanismes de sécurité avec RTP est importante. Ces mécanismes sont exposés à la Section 9.

Les traducteurs ou mixeurs de niveau RTP peuvent être utilisés pour permettre au trafic RTP d'atteindre des hôtes au-delà des pare-feu. Les principes et pratiques de sécurité de pare-feu appropriée vont au delà du domaine d'application de ce document et devraient être suivis dans la conception et l'installation de ces appareils et dans l'admission de l'utilisation des applications RTP derrière le pare-feu.

15. Considérations relatives à l'IANA

Des types supplémentaires de paquet RTCP et de types d'élément de SDES peuvent être enregistrés auprès de l'autorité d'allocation des numéros de l'Internet (IANA, *Internet Assigned Numbers Authority*). Comme ces espaces de numéros sont étroits, permettre un enregistrement libre de nouvelles valeurs ne serait pas prudent. Pour faciliter l'examen des demandes et promouvoir une utilisation partagée des nouveaux types parmi plusieurs applications, les demandes d'enregistrement de nouvelles valeurs doivent être documentées dans une RFC ou autre référence permanente et directement disponible telle que le produit d'un autre organisme coopératif de normalisation (par exemple, l'UIT-T). D'autres demandes peuvent aussi être acceptées, suivant l'avis d'un "expert désigné."

(Contacter l'IANA pour les informations de contact de l'expert actuel.)

Les spécifications de profil RTP DEVRAIENT enregistrer auprès de l'IANA un nom pour le profil sous la forme "RTP/xxx", où xxx est une courte abréviation du titre du profil. Ces noms sont destinés à être utilisés par les protocoles de contrôle de niveau supérieur, tels que le protocole de description de session (SDP, *Session Description Protocol*), RFC 2327 [15], pour se référer aux méthodes de transport.

16. Déclaration de droits de propriété intellectuelle

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autre droit qui pourrait être revendiqué au titre de la mise en œuvre ou l'utilisation de la technologie décrite dans le présent document ou sur la mesure dans laquelle toute licence sur de tels droits pourrait être ou n'être pas disponible ; pas plus qu'elle ne prétend avoir accompli aucun effort pour identifier de tels droits. Les informations sur les procédures de l'IETF au sujet des droits dans les documents en cours de normalisation et se rapportant aux normes figurent dans le BCP 11.

Des copies des dépôts d'IPR faites au secrétariat de l'IETF et toutes assurances de disponibilité de licences, ou le résultat de tentatives faites pour obtenir une licence ou permission générale d'utilisation de tels droits de propriété par ceux qui mettent en œuvre ou utilisent la présente spécification peuvent être obtenues sur le répertoire en ligne des IPR de l'IETF à http://www.ietf.org/ipr.

L'IETF invite toute partie intéressée à porter son attention sur tous copyrights, brevet ou applications de brevets, ou autres droits de propriété qui pourraient recouvrir la technologie qui pourrait être nécessaire pour mettre en œuvre la présente norme. Prière d'adresser les informations au directeur exécutif de l'IETF.

17. Remerciements

Ce mémoire se fonde sur des discussions au sein du groupe de travail Transport audio/vidéo de l'IETF présidé par Stephen Casner et Colin Perkins. Le protocole actuel tire son origine du protocole de la voix sur le réseau et du protocole de vidéo par paquet (Danny Cohen et Randy Cole) et du protocole mis en œuvre par l'application vat (Van Jacobson et Steve McCanne). Christian Huitema a fourni l'idée du générateur d'identifiant aléatoire. L'analyse et la simulation détaillée de l'algorithme de reconsidération du temporisateur a été faite par Jonathan Rosenberg. Les ajouts pour les codages en couche ont été spécifiés par Michael Speer et Steve McCanne.

Appendice A - Algorithmes

On donne des exemples de code C pour les aspects des algorithmes d'envoyeur et receveur RTP. Il peut y avoir d'autres méthodes de mise en œuvre qui sont plus rapides dans des environnements de fonctionnement particuliers ou qui présentent d'autres avantages. Ces notes de mise en œuvre sont seulement pour information et sont destinées à préciser la spécification de RTP.

Les définitions suivantes sont utilisées pour tous les exemples ; pour être clair et bref, les définitions de structure ne sont valides que pour les architectures gros boutiennes à 32 bits (octet de poids fort en premier). Les champs binaires sont supposés être mis en paquets serrés en ordre gros boutien des bits, sans bourrage additionnel. Des modifications seraient nécessaires pour construire une mise en œuvre portable.

```
/*
 * rtp.h  --  en-tête de fichier RTP
 */
#include <sys/types.h>

/*
 * Les définitions de type ci-dessous sont valides pour des architectures à 32 bits et peuvent devoir être ajustées
 *pour des architectures à 126 ou 64 bits.
 */
typedef unsigned char           u_int8;
typedef unsigned short          u_int16;
typedef unsigned int            u_int32;
typedef                         short int16;

/*
 * Version de protocole en cours.
 */
#define RTP_VERSION    2
```

```
#define RTP_SEQ_MOD (1<<16)
#define RTP_MAX_SDES 255        /* longueur de texte maximum pour SDES */

typedef enum {
        RTCP_SR   = 200,
        RTCP_RR   = 201,
        RTCP_SDES = 202,
        RTCP_BYE  = 203,
        RTCP_APP  = 204
} rtcp_type_t;

typedef enum {
        RTCP_SDES_END   = 0,
        RTCP_SDES_CNAME = 1,
        RTCP_SDES_NAME  = 2,
        RTCP_SDES_EMAIL = 3,
        RTCP_SDES_PHONE = 4,
        RTCP_SDES_LOC   = 5,
        RTCP_SDES_TOOL  = 6,
        RTCP_SDES_NOTE  = 7,
        RTCP_SDES_PRIV  = 8
} rtcp_sdes_type_t;

/*
* en-tête de données RTP
*/
typedef struct {
        unsigned int version:2;     /* version du protocole */
        unsigned int p:1;  /* fanion de bourrage */
        unsigned int x:1;  /* fanion d'extension d'en-tête */
        unsigned int cc:4;          /* compte de CSRC */
        unsigned int m:1;/* bit marqueur */
        unsigned int pt:7;/* type de charge utile */
        unsigned int seq:16;        /* numéro de séquence */
        u_int32 ts;         /* horodatage */
        u_int32 ssrc;       /* source de synchronisation */
        u_int32 csrc[1];   /* liste de CSRC facultative */
} rtp_hdr_t;

/*
* mot d'en-tête commun de RTCP
*/
typedef struct {
        unsigned int version:2;             /* version du protocole */
        unsigned int p:1;  /* fanion de bourrage */
        unsigned int count:5;       /* varie selon le type de paquet */
        unsigned int pt:8;/* type de paquet RTCP */
        u_int16 length;    /* longueur de paquet en mots, w/o ce mot */
} rtcp_common_t;

/*
* gabarit gros boutien pour la version, paire bit de bourrage et type de paquet
*/
#define RTCP_VALID_MASK (0xc000 | 0x2000 | 0xfe)
#define RTCP_VALID_VALUE ((RTP_VERSION << 14) | RTCP_SR)

/*
* bloc de rapport de réception
*/
typedef struct {
```

```
        u_int32 ssrc;                   /* source des données objet du rapport */
        unsigned int fraction:8;    /* fraction perdue depuis le dernier SR/RR */
        int lost:24;        /* cumul du nombre de paquets perdus (signé !) */
        u_int32 last_seq; /* dernier numéro de séquence étendu reçu */
        u_int32 jitter;     /* gigue inter arrivée */
        u_int32 lsr;        /* dernier paquet SR provenant de cette source */
        u_int32 dlsr;       /* délai depuis le dernier paquet SR */
} rtcp_rr_t;

/*
* élément de SDES
*/
typedef struct {
        u_int8 type;                    /* type d'élément (rtcp_sdes_type_t) */
        u_int8 length;      /* longueur de l'élément (en octets) */
        char data[1];       /* texte, non terminé par l'octet zéro */
} rtcp_sdes_item_t;

/*
* un paquet RTCP
*/
typedef struct {
        rtcp_common_t common;           /* en-tête commun */
        union {
                                        /* rapport d'envoyeur (SR) */
        struct {
        u_int32 ssrc;                   /* envoyeur qui génère ce rapport */
        u_int32 ntp_sec;  /* horodatage NTP */
        u_int32 ntp_frac;
        u_int32 rtp_ts;     /* horodatage RTP */
        u_int32 psent;      /* paquets envoyés */
        u_int32 osent;      /* octets envoyés */
        rtcp_rr_t rr[1];    /* liste de longueur variable */
} sr;

/* rapport de réception (RR) */
struct {
        u_int32 ssrc;                   /* receveur qui génère ce rapport */
        rtcp_rr_t rr[1];    /* liste de longueur variable */
} rr;

/* description de source (SDES) */
struct rtcp_sdes {
        u_int32 src;                    /* premier SSRC/CSRC */
        rtcp_sdes_item_t item[1]; /* liste des éléments de SDES */
} sdes;

/* BYE */
struct {
        u_int32 src[1];                 /* liste des sources */
                                        /* on ne peut exprimer le texte de cause en queue */
        } bye;
      } r;
} rtcp_t;

typedef struct rtcp_sdes rtcp_sdes_t;

/*
* informations d'état par source
*/
```

```
typedef struct {
        u_int16 max_seq;                /* plus fort numéro de séquence rencontré */
        u_int32 cycles;   /* compte glissant de cycles de numéros de séquence */
        u_int32 base_seq;           /* numéro de séquence de base */
        u_int32 bad_seq; /* dernier 'mauvais' numéro de séquence + 1 */
        u_int32 probation;          /* paquets en séquence depuis que la source est valide */
        u_int32 received; /* paquets reçus */
        u_int32 expected_prior;   /* paquets attendus au dernier intervalle */
        u_int32 received_prior;   /* paquets reçus au dernier intervalle */
        u_int32 transit;   /* temps de transit relatif pour le paquet précédent */
        u_int32 jitter;     /* gigue estimée */
        /* ... */
  } source;
```

A.1 Vérifications de validité des en-têtes de données RTP

Un receveur RTP devrait vérifier la validité de l'en-tête RTP sur les paquets entrants car ils pourraient être chiffrés ou pourraient provenir d'une application différente après avoir été mal adressés. De même, si le chiffrement est activé conformément à la méthode décrite à la Section 9, la vérification de validité de l'en-tête est nécessaire pour vérifier que les paquets entrants ont été correctement déchiffrés, bien qu'un échec de la vérification de validité de l'en-tête (par exemple, un type de charge utile inconnu) puisse ne pas indiquer nécessairement un échec du déchiffrement.

Seules des vérifications de validité faibles sont possibles sur un paquet de données RTP provenant d'une source qui n'avait pas été entendue auparavant :

- o le champ de version RTP doit être égal à 2 ;
- o le type de charge utile doit être connu, et en particulier il doit n'être pas égal à SR ou RR ;
- o si le bit P est mis, le dernier octet du paquet doit alors contenir un compte d'octets valide, en particulier, moins que la longueur totale de paquet moins la taille d'en-tête ;
- o le bit X doit être à zéro si le profil ne spécifie pas que le mécanisme d'extension d'en-tête peut être utilisé. Autrement, le champ Longueur d'extension doit être inférieur à la taille totale de paquet moins la longueur d'en-tête fixe et de bourrage ;
- o la longueur du paquet doit être cohérente avec CC et type de charge utile (si la charge utile a une longueur connue).

Les trois dernières vérifications sont un peu complexes et pas toujours possibles, ce qui ne laisse que les deux premières qui ne totalisent que peu de bits. Si l'identifiant de SSRC dans le paquet est un qui a été reçu auparavant, le paquet est alors probablement valide et vérifier si le numéro de séquence est dans la gamme attendue donne une confirmation de validation. Si l'identifiant de SSRC n'a jamais été vu auparavant, les paquets de données qui portent cet identifiant peuvent alors être considérés comme invalides jusqu'à ce qu'un petit nombre d'entre eux arrivent avec des numéros de séquence consécutifs. Ces paquets invalides PEUVENT être éliminés ou ils PEUVENT être mémorisés et délivrés une fois que la validation aura été réalisée si le délai résultant est acceptable.

Le sous-programme update_seq indiqué ci-dessous assure qu'une source est déclarée valide seulement après que MIN_SEQUENTIAL paquets ont été reçus en séquence. Il valide aussi le numéro de séquence seq d'un paquet nouvellement reçu et met à jour l'état de séquence pour la source du paquet dans la structure vers laquelle il pointe.

Lorsque une nouvelle source est entendue pour la première fois, c'est-à-dire, lorsque l'identifiant de SSRC n'est pas dans le tableau (voir au paragraphe 8.2) et que l'état par source est alloué pour lui, s->probation est mis au nombre de paquets séquentiels requis avant de déclarer qu'une source est valide (paramètre MIN_SEQUENTIAL) et les autres variables sont initialisée :

```
init_seq(s, seq);
s->max_seq = seq - 1;
s->probation = MIN_SEQUENTIAL;
```

Un s->probation différent de zéro marque la source comme pas encore valide de sorte que l'état peut être éliminé après une brève temporisation plutôt qu'une longue, comme exposé au paragraphe 6.2.1.

Après qu'une source est considérée valide, le numéro de séquence est considéré valide si il n'est pas supérieur à MAX_DROPOUT de plus de s->max_seq ni supérieur à MAX_MISORDER derrière. Si le nouveau numéro de séquence

est de plus de max_seq modulo la gamme de numéros de séquence RTP (16 bits) mais est plus petit que max_seq, il est revenu à zéro et le compte (glissant) de cycles de numéros de séquence est incrémenté. Une valeur de un est retournée pour indiquer un numéro de séquence valide.

Autrement, la valeur zéro est retournée pour indiquer que la validation a échoué, et le mauvais numéro de séquence plus 1 est mémorisé. Si le prochain paquet reçu porte le prochain plus fort numéro de séquence, il est considéré comme le début valide d'une nouvelle séquence de paquet vraisemblablement causée par un rejet étendu ou un redémarrage de source. Comme plusieurs cycles complets de numéros de séquence peuvent avoir été manqués, la statistique des pertes de paquet est remise à zéro.

On donne les valeurs types des paramètres, sur la base d'un temps maximum de désordre de 2 secondes à 50 paquets/s et un rejet maximum de 1 minute. Le paramètre de rejet MAX_DROPOUT devrait être une faible fraction de l'espace de 16 bits de numéros de séquence pour donner une probabilité raisonnable que les nouveaux numéros de séquence après redémarrage ne tombe pas dans la gamme acceptable pour les numéros de séquence d'avant le redémarrage.

```
void init_seq(source *s, u_int16 seq)
{
s->base_seq = seq;
s->max_seq = seq;
s->bad_seq = RTP_SEQ_MOD + 1;          /* ainsi seq == bad_seq est faux */
s->cycles = 0;
s->received = 0;
s->received_prior = 0;
s->expected_prior = 0;
/* autre initialisation */
}

int update_seq(source *s, u_int16 seq)
{
u_int16 udelta = seq - s->max_seq;
const int MAX_DROPOUT = 3000;
const int MAX_MISORDER = 100;
const int MIN_SEQUENTIAL = 2;

/*
 * la source n'est pas valide tant que MIN_SEQUENTIAL paquets avec les numéros de séquence en séquence
 * n'ont pas été reçus.
 */
si (s->probation) {
                                        /* le paquet est en séquence */
si (seq == s->max_seq + 1) {
        s->probation--;
        s->max_seq = seq;
        si (s->probation == 0) {
        init_seq(s, seq);
        s->received++;
        return 1;

        }
        } autrement {
        s->probation = MIN_SEQUENTIAL - 1;
        s->max_seq = seq;
        }
        return 0;
} autrement si (udelta < MAX_DROPOUT) {
                                        /* dans l'ordre, avec des trous possibles */
si (seq < s->max_seq) {
                                        /* le numéro de séquence est revenu à zéro - compter un autre cycle de 64 K. */
        s->cycles += RTP_SEQ_MOD;
        }
```

```
        s->max_seq = seq;
} autrement si (udelta <= RTP_SEQ_MOD - MAX_MISORDER) {
                                        /* le numéro de séquence a fait un très grand bond */
        si (seq == s->bad_seq) {
        /*
        * Deux paquets en séquence – on suppose que l'autre côté a redémarré sans sous dire de resynchroniser
        * (c'est-à-dire, prétend que c'est le premier paquet).
        */
        init_seq(s, seq);
}
autrement {
        s->bad_seq = (seq + 1) & (RTP_SEQ_MOD-1);
        return 0;
        }
} autrement {
                                        /* paquet dupliqué ou replacé */
}
s->received++;
return 1;
}
```

La vérification de validité peut être rendue plus forte en exigeant que deux paquets soient en séquence. L'inconvénient est qu'un plus grand nombre de paquets initiaux seront éliminés (ou retardés dans une file d'attente) et que de forts taux de perte de paquet pourrait empêcher la validation. Cependant, comme la validation d'en-tête RTCP est relativement forte, si un paquet RTCP est reçu d'une source avant les paquets de données, le compte pourrait être ajusté de telle sorte que seuls deux paquets soient exigés en séquence. Si la perte de données initiale pour quelques secondes peut être tolérée, une application PEUT choisir d'éliminer tous les paquets de données provenant d'une source jusqu'à ce qu'un paquet RTCP valide soit reçu de cette source.

Selon l'application et le codage, les algorithmes peuvent exploiter des connaissances supplémentaires sur le format de charge utile pour une validation plus poussée. Pour les types de charge utile où l'incrément d'horodatage est le même pour tous les paquets, les valeurs d'horodatage peuvent être prédites d'après le précédent paquet reçu de la même source en utilisant la différence de numéros de séquence (en supposant qu'il n'y a pas eu de changement du type de charge utile).

Une vérification forte "rapide" est possible car à une forte probabilité les quatre premiers octets de l'en-tête d'un paquet de données RTP nouvellement reçu seront juste les mêmes que ceux du précédent paquet provenant du même SSRC sauf que le numéro de séquence aura augmenté de un. De même, une antémémoire à une seule entrée peut être utilisée pour des recherches plus rapides de SSRC dans des applications où les données sont normalement reçues d'une source à la fois.

A.2 Vérification de validité des en-têtes RTCP

Les vérifications suivantes devraient être appliquées aux paquets RTCP.

- le champ Version RTP doit être égal à 2 ;
- le champ Type de charge utile du premier paquet RTCP dans un paquet composé doit être égal à SR ou RR ;
- le bit bourrage (P) devrait être à zéro pour le premier paquet d'un paquet RTCP composé parce que le bourrage ne devrait être appliqué, si nécessaire, qu'au dernier paquet ;
- l'addition des champs Longueur des paquets RTCP individuels doit donner la longueur totale du paquet RTCP composé tel que reçu. C'est une vérification très forte.

Le fragment de code ci-dessous effectue toutes ces vérifications. Le type de paquet n'est pas vérifié pour les paquets suivants car des types de paquet inconnus peuvent être présents et devraient être ignorés.

```
u_int32 len;                    /* longueur du paquet RTCP composé en mots */
rtcp_t *r;          /* en-tête RTCP */
rtcp_t *end;        /* fin du paquet RTCP composé */

si ((*(u_int16 *)r & RTCP_VALID_MASK) != RTCP_VALID_VALUE) {
                                        /* quelque chose ne va pas dans le format du paquet */
}
```

```
end = (rtcp_t *)((u_int32 *)r + len);
do r = (rtcp_t *)((u_int32 *)r + r->common.length + 1);
while (r < end && r->common.version == 2);

si (r != end) {
                                        /* quelque chose ne va pas dans le format du paquet */
}
```

A.3 Détermination du nombre de paquets attendus et perdus

Afin de calculer le taux de perte de paquets, le nombre de paquets RTP attendus et réellement reçus de chaque source doit être connu, en utilisant des informations d'état par source définies dans la structure de source référencée via le pointeur s dans le code ci-dessous. Le nombre de paquets reçus est simplement le compte de paquets tels qu'ils arrivent, y compris tout paquet en retard ou dupliqué. Le nombre de paquets attendus peut être calculé par le receveur comme la différence entre le plus fort numéro de séquence reçu (s->max_seq) et le premier numéro de séquence reçu (s->base_seq). Comme le numéro de séquence est seulement de 16 bits et va revenir à zéro, il est nécessaire d'étendre le plus fort numéro de séquence avec le compte (glissant) de retour à zéro du numéro de séquence (s->cycles). Le compte des paquets reçus et le compte de cycles sont tous deux tenus par le programme de vérification de validité de l'en-tête RTP de l'Appendice A.1.

```
extended_max = s->cycles + s->max_seq;
expected = extended_max - s->base_seq + 1;
```

Le nombre de paquets perdus est défini comme étant le nombre de paquets attendus moins le nombre de paquets réellement reçus :

```
perdus = attendus - s->reçus;
```

Comme ce nombre signé est porté sur 24 bits, il devrait être tendre vers 0x7fffff pour les pertes positives ou 0x800000 pour les pertes négatives plutôt que de revenir à zéro.

La fraction de paquets perdus durant le dernier intervalle de rapports (depuis l'envoi du précédent paquet SR ou RR) est calculé à partir des différences entre comptes de paquets attendus et reçus sur l'intervalle, où expected_prior et received_prior sont les valeurs sauvegardées lorsque le précédent rapport de réception avait été généré :

```
    expected_interval = expected - s->expected_prior;
s->expected_prior = expected;
received_interval = s->received - s->received_prior;
s->received_prior = s->received;
lost_interval = expected_interval - received_interval;
if (expected_interval == 0 || lost_interval <= 0) fraction = 0;
else fraction = (lost_interval << 8) / expected_interval;
```

La fraction résultante est un nombre de 8 bits à virgule fixe avec la virgule binaire au bord gauche.

A.4 Génération des paquets de SDES RTCP

Cette fonction construit un tronçon SDES dans la mémoire tampon b composée d'éléments argc fournis dans des arrangements de type, valeur et longueur. Elle retourne un pointeur sur la prochaine localisation disponible au sein de b.

```
char *rtp_write_sdes(char *b, u_int32 src, int argc, rtcp_sdes_type_t type[], char *value[], int length[])
    {
        rtcp_sdes_t *s = (rtcp_sdes_t *)b;
        rtcp_sdes_item_t *rsp;
        int i;
        int len;
        int pad;

                                        /* en-tête de SSRC */
```

```
s->src = src;
rsp = &s->item[0];

                                        /* éléments de SDES */
    for (i = 0; i < argc; i++) {
      rsp->type = type[i];
      len = length[i];
      if (len > RTP_MAX_SDES) {
                                        /* longueur invalide, peut vouloir entreprendre une autre action */
        len = RTP_MAX_SDES;
      }
      rsp->length = len;
      memcpy(rsp->data, value[i], len);
      rsp = (rtcp_sdes_item_t *)&rsp->data[len];
    }

                                /* se termine avec le marqueur de fin et bourre jusqu'à la prochaine limite de 4 octets */
    len = ((char *) rsp) - b;
    pad = 4 - (len & 0x3);
    b = (char *) rsp;
    while (pad--) *b++ = RTCP_SDES_END;

    return b;
  }
```

A.5 Analyse des paquets de SDES RTCP

Cette fonction analyse un paquet de SDES, invoquant les fonctions find_member() pour trouver un pointeur sur les informations sur un membre de la session étant donnés l'identifiant de SSRC et member_sdes() pour mémoriser les nouvelles informations de SSRC sur ce membre. Cette fonction s'attend à un pointeur sur l'en-tête du paquet RTCP.

```
  void rtp_read_sdes(rtcp_t *r)
  {
    int count = r->common.count;
    rtcp_sdes_t *sd = &r->r.sdes;
    rtcp_sdes_item_t *rsp, *rspn;
    rtcp_sdes_item_t *end = (rtcp_sdes_item_t *) ((u_int32 *)r + r->common.length + 1);
    source *s;

    while (--count >= 0) {
      rsp = &sd->item[0];
      if (rsp >= end) break;
      s = find_member(sd->src);

      for (; rsp->type; rsp = rspn ) {
        rspn = (rtcp_sdes_item_t *)((char*)rsp+rsp->length+2);
        if (rspn >= end) {
          rsp = rspn;
          break;
        }
        member_sdes(s, rsp->type, rsp->data, rsp->length);
      }
      sd = (rtcp_sdes_t *)
          ((u_int32 *)sd + (((char *)rsp - (char *)sd) >> 2)+1);
    }
    if (count >= 0) {
                                        /* format de paquet invalide */
    }
  }
```

A.6 Génération d'un identifiant aléatoire à 32 bits

Le sous programme suivant génère un identifiant aléatoire de 32 bits qui utilise le programme MD5 publié dans la RFC 1321 [32]. Le programme système peut n'être pas présent sur tous les systèmes d'exploitation, mais il devrait servir d'indication sur les sortes d'informations qui peuvent être utilisées. D'autres systèmes qui peuvent être appropriés sont

- o getdomainname(),
- o getwd(), ou
- o getrusage().

Les échantillons audio ou vidéo "en direct" sont aussi une bonne source de numéros aléatoires, mais il faut veiller à éviter d'utiliser un microphone éteint ou une caméra aveuglée comme source [17].

L'utilisation de ce programme ou d'un autre similaire est recommandée pour générer le germe initial du générateur de nombre aléatoire qui produit la période RTCP (comme indiqué à l'Appendice A.7) pour générer les valeurs initiales pour le numéro de séquence et l'horodatage, et pour générer les valeurs de SSRC. Comme ce programme va vraisemblablement être un gros consommateur de CPU, son utilisation directe pour générer des périodes RTCP est inappropriée parce que l'imprévisibilité n'est pas le problème. Noter que ce programme produit le même résultat sur des invocations répétées jusqu'à ce que la valeur de l'horloge système change, à moins que des valeurs différentes soient fournies pour l'argument de type.

```
/*
 * Générer une quantité de 32 bits aléatoire.
 */
#include <sys/types.h>      /* u_long */
#include <sys/time.h>       /* gettimeofday() */
#include <unistd.h>         /* get..() */
#include <stdio.h>          /* printf() */
#include <time.h>           /* clock() */
#include <sys/utsname.h> /* uname() */
#include "global.h"         /* d'après la RFC 1321 */
#include "md5.h"            /* d'après la RFC 1321 */

#define MD_CTX MD5_CTX
#define MDInit MD5Init
#define MDUpdate MD5Update
#define MDFinal MD5Final

static u_long md_32(char *string, int length)
{
        MD_CTX context;
          union {
             char         c[16];
          u_long     x[4];
        } digest;
          u_long r;
          int i;

        MDInit (&context);

        MDUpdate (&context, string, length);
MDFinal ((unsigned char *)&digest, &context);
r = 0;
for (i = 0; i < 3; i++) {
             r ^= digest.x[i];
        }
        return r;
}                                       /* md_32 */
```

```
/*
 * Retourner une quantité aléatoire non signée de 32 bits. Utiliser l'argument 'type' si vous avez besoin de générer
 * plusieurs valeurs différentes en proche succession.
 */
u_int32 random32(int type)
{
        struct {
            int         type;
struct  timeval tv;
clock_t cpu;
pid_t   pid;
u_long  hid;
uid_t   uid;
gid_t   gid;
struct  utsname name;
        } s;

gettimeofday(&s.tv, 0);
uname(&s.name);
s.type = type;
s.cpu  = clock();
s.pid  = getpid();
s.hid  = gethostid();
s.uid  = getuid();
s.gid  = getgid();
                                        /* aussi : heure de démarrage du système */

return md_32((char *)&s, sizeof(s));
}                                       /* random32 */
```

A.7 Calcul de l'intervalle de transmission RTCP

Les fonctions suivantes mettent en œuvre les règles de transmission et réception RTCP décrites au paragraphe 6.2. Ces règles sont codées dans plusieurs fonctions :

- rtcp_interval() calcule l'intervalle calculé déterministe, mesuré en secondes. Les paramètres sont définis au paragraphe 6.3.
- OnExpire() est invoqué lorsque le temporisateur de transmission RTCP arrive à expiration.
- OnReceive() est invoqué chaque fois qu'un paquet RTCP est reçu.

OnExpire() et OnReceive() ont tous deux l'événement e comme argument. C'est le prochain événement programmé pour ce participant, soit un rapport RTCP soit un paquet BYE. On suppose que les fonctions suivantes sont disponibles :

- Schedule(instant t, événement e) programme un événement e survenant à l'instant t. Lorsque l'instant t arrive, la fonction OnExpire est invoquée avec e comme argument.
- Reschedule(instant t, événement e) reprogramme un événement e précédemment programmé pour l'instant t.
- SendRTCPReport(événement e) envoie un rapport RTCP.
- SendBYEPacket(événement e) envoie un paquet BYE.
- TypeOfEvent(événement e) retourne EVENT_BYE si l'événement en cours de traitement est pour l'envoi d'un paquet BYE, autrement, il retourne EVENT_REPORT.
- PacketType(p) retourne PACKET_RTCP_REPORT si le paquet p est un rapport RTCP (pas un BYE), PACKET_BYE si c'est un paquet RTCP BYE, et PACKET_RTP si c'est un paquet de données RTP régulier.
- ReceivedPacketSize() et SentPacketSize() retourne la taille du paquet référencé en octets.
- NewMember(p) retourne un 1 si le participant qui a envoyé le paquet p n'est pas actuellement dans la liste des membres, 0 autrement. Noter que cette fonction n'est pas suffisante pour une mise en œuvre complète parce que chaque identifiant de CSRC dans un paquet RTP et chaque SSRC dans un paquet BYE devrait être traité.
- NewSender(p) retourne un 1 si le participant qui a envoyé le paquet p n'est pas actuellement dans la sous liste des envoyeurs de la liste des membres, 0 autrement.
- AddMember() et RemoveMember() pour ajouter et retirer des participants de la liste des membres.
- AddSender() et RemoveSender() pour ajouter et retirer des participants de la sous liste des envoyeurs de la liste des membres.

Ces fonctions devraient être étendues pour une mise en œuvre qui permet que des fractions de bande passante RTCP pour les envoyeurs et les non envoyeurs soient spécifiées comme paramètres explicites plutôt que comme valeurs fixes de 25 % et 75 %. La mise en œuvre étendue de rtcp_interval() devrait éviter la division par zéro si un des paramètres est zéro.

```
double rtcp_interval(int members,
                     int senders,
           double rtcp_bw,
           int we_sent,
           double avg_rtcp_size,
           int initial)
{
/*
 * Temps moyen minimum entre paquets RTCP à partir de ce site (en secondes). Ce temps empêche les rapports de
 * "piétiner" lorsque les sessions sont petites et que la loi des grands nombres n'aide pas à lisser le trafic. Il empêche aussi
 * l'intervalle de rapport de devenir ridiculement petit durant des coupures transitoires comme une partition de réseau.
 */
double const RTCP_MIN_TIME = 5.;
/*
 * Fraction de la bande passante RTCP à partager entre les envoyeurs actifs. (Cette fraction a été choisie de telle sorte que
 * dans une session normale avec un ou deux membres actifs, le temps de rapport calculé soit en gros égal au temps de
 * rapport minimum de façon à ne pas ralentir inutilement les rapports de receveur.) La fraction de receveur doit être 1 – la
 * fraction d'envoyeur.
 */
double const RTCP_SENDER_BW_FRACTION = 0.25;
double const RTCP_RCVR_BW_FRACTION = (1-RTCP_SENDER_BW_FRACTION);
/*
/* Pour compenser le "reconsidération de temporisateur" convergeant vers une valeur en dessous de la moyenne prévue.
*/
double const COMPENSATION = 2.71828 - 1.5;

double t;                                   /* intervalle */
double rtcp_min_time = RTCP_MIN_TIME;
int n;                                      /* nombre de membres pour le calcul */

/*
 * La toute première invocation au démarrage de l'application utilise la moitié du retard minimum pour des notification plus
* rapides tout en permettant un peu de temps avant de faire un rapport d'aléation et d'en apprendre sur les autres sources de
 * sorte que l'intervalle de rapport va converger plus rapidement vers l'intervalle correct.
*/
if (initial) {
    rtcp_min_time /= 2;
}
/*
 * Dédier une fraction de la bande passante RTCP aux envoyeurs sauf si le nombre des envoyeurs est assez grand pour que
 * leur part soit supérieure à cette fraction.
 */
n = members;
if (senders <= members * RTCP_SENDER_BW_FRACTION) {
    if (we_sent) {
        rtcp_bw *= RTCP_SENDER_BW_FRACTION;
          n = senders;
    } else {
        rtcp_bw *= RTCP_RCVR_BW_FRACTION;
          n -= senders;
     }
  }

/*
 * Le nombre effectif de sites multiplié par la taille moyenne de paquet est le nombre total des octets envoyés lorsque
```

```
 * chaque site envoie un rapport. Diviser cela par la bande passante efficace donne l'intervalle de temps sur lequel ces
 * paquets doivent être envoyés afin de satisfaire la bande passante cible, avec la mise en application d'un minimum. Dans
 * cet intervalle de temps on envoie un rapport de sorte que ce temps est aussi notre temps moyen entre les rapports.
 */
t = avg_rtcp_size * n / rtcp_bw;
if (t < rtcp_min_time) t = rtcp_min_time;

/*
 * Pour éviter que des salves de trafic ne se synchronisent involontairement avec d'autres sites, on prend alors notre
 * prochain intervalle de rapport réel comme un nombre aléatoire uniformément réparti entre 0,5*t et 1,5*t.
 */
t = t * (drand48() + 0.5);
t = t / COMPENSATION;
return t;
}

void OnExpire(event e,
            int         members,
            int         senders,
            double      rtcp_bw,
            int         we_sent,
            double      *avg_rtcp_size,
            int         *initial,
            time_tp     tc,
            time_tp     *tp,
            int         *pmembers)
{
/* Cette fonction est chargée de décider d'envoyer maintenant un rapport RTCP ou un paquet BYE, ou de reprogrammer la
 * transmission. Elle est aussi chargée de mettre à jour les variables d'état pmembers, initial, tp, et avg_rtcp_size. Cette
 * fonction devrait être invoquée à expiration du temporisateur d'événement utilisé par Schedule().
 */

double t;                                   /* Intervalle */
double tn;          /* Prochaine heure de transmission */

/* Dans le cas d'un BYE, on utilise "reconsidération de temporisateur" pour reprogrammer la transmission du BYE si
 * nécessaire */

if (TypeOfEvent(e) == EVENT_BYE) {
    t = rtcp_interval(members,
                      senders,
            rtcp_bw,
            we_sent,
            *avg_rtcp_size,
            *initial);
    tn = *tp + t;
            if (tn <= tc) {
        SendBYEPacket(e);
            exit(1);
    } else {
        Schedule(tn, e);
    }

} else if (TypeOfEvent(e) == EVENT_REPORT) {
    t = rtcp_interval(members,
                      senders,
            rtcp_bw,
            we_sent,
            *avg_rtcp_size,
            *initial);
```

```
    tn = *tp + t;
         if (tn <= tc) {
        SendRTCPReport(e);
         *avg_rtcp_size = (1./16.)*SentPacketSize(e) + (15./16.)*(*avg_rtcp_size);
         *tp = tc;

/* On doit redessiner l'intervalle. Ne pas réutiliser celui calculé ci-dessus, car il n'est plus réellement réparti de la même
 * façon, car nous avons pour condition qu'il soit assez petit pour causer l'envoi d'un paquet */

    t = rtcp_interval(members,
                      senders,
         rtcp_bw,
         we_sent,
         *avg_rtcp_size,
         *initial);

           Schedule(t+tc,e);
         *initial = 0;
        } else {
           Schedule(tn, e);
        }
         *pmembers = members;
    }
}

void OnReceive(packet p,
                  event e,
         int *members,
         int *pmembers,
         int *senders,
         double *avg_rtcp_size,
         double *tp,
         double tc,
         double tn)
```

```
            if (NewMember(p) == FALSE) {
                RemoveMember(p);
            *members -= 1;
              }
            if (*members < *pmembers) {
                tn = tc + (((double) *members)/(*pmembers))*(tn - tc);
            *tp = tc - (((double) *members)/(*pmembers))*(tc - *tp);
                                        /* Reprogramme le prochain rapport pour le temps tn */
                Reschedule(tn, e);
            *pmembers = *members;
              }
          } else if (TypeOfEvent(e) == EVENT_BYE) {
            *members += 1;
          }
      }
}
```

A.8 Estimation de la gigue inter arrivées

Les fragments de code ci-dessous mettent en œuvre l'algorithme donné au paragraphe 6.4.1 pour calculer une estimation de la variance statistique du temps inter arrivée des données de RTP à insérer dans le champ Gigue inter arrivée des rapports de réception. Les entrées sont "r->ts", l'horodatage provenant du paquet entrant, et "arrival", l'heure en cours dans les mêmes unités. Ici s pointe sur l'état de la source ; s->transit contient le temps de transit relatif pour le paquet précédent, et s->jitter contient la gigue estimée. Le champ gigue du rapport de réception est mesuré en unités d'horodatage et exprimé par un entier non signé, mais l'estimation de la gigue est conservée en virgule flottante. Lorsque chaque paquet de données arrive, l'estimation de la gigue est mise à jour :

```
int transit = arrival - r->ts;
int d = transit - s->transit;
s->transit = transit;
if (d < 0) d = -d;
s->jitter += (1./16.) * ((double)d - s->jitter);
```

Lorsque est généré un bloc de rapport de réception (sur lequel pointe rr) pour ce membre, l'estimation actuelle de gigue est retournée :

```
rr->jitter = (u_int32) s->jitter;
```

Autrement, l'estimation de la gigue pour être gardée sous forme d'un entier, mais échelonnée pour réduire l'erreur d'arrondi. Le calcul est le même sauf pour la dernière ligne :

```
s->jitter += d - ((s->jitter + 8) >> 4);
```

Dans ce cas, l'estimation est échantillonnée pour le rapport de réception par :

```
rr->jitter = s->jitter >> 4;
```

Appendice B - Changements par rapport à la RFC 1889

La plus grande partie de la présente RFC est identique à la RFC 1889. Il n'y a pas de changement dans le format de paquet sur le réseau, seulement des changements des règles et algorithmes qui gouvernent la façon d'utiliser le protocole. Le plus gros changement est une amélioration de l'algorithme d'échelonnement des temporisateurs pour calculer quand envoyer les paquets RTCP :

- L'algorithme pour calculer l'intervalle de transmission RTCP spécifié aux paragraphes 6.2 et 6.3 et illustré à l'Appendice A.7 est augmenté pour inclure la "reconsidération" pour minimiser la transmission qui excède le taux prévu lorsque de nombreux participants se joignent simultanément à une session, et la "reconsidération inverse" pour réduire

l'incidence et la durée de la temporisation de faux participant lorsque le nombre des participants chute rapidement. La reconsidération inverse est ainsi utilisée pour éventuellement raccourcir le délai avant d'envoyer un SR RTCP lors de la transition du mode de receveur passif à envoyeur actif.

- Le paragraphe 6.3.7 spécifie de nouvelles règles de contrôle du moment où un paquet BYE RTCP devrait être envoyé afin d'éviter une inondation de paquets lorsque de nombreux participants quittent simultanément une session.

- L'exigence de conserver l'état pour les participants inactifs pendant une période assez longue pour couvrir une partition de réseau ordinaire a été retirée du paragraphe 6.2.1. Dans une session où de nombreux participants se joignent pour un bref moment et oublient d'envoyer BYE, cette exigence causerait une surestimation significative du nombre de participants. L'algorithme de reconsidération ajouté à cette révision compense le grand nombre de nouveaux participants se joignant simultanément lorsque une partition se répare.

On devrait noter que ces améliorations n'ont d'effet significatif que lorsque le nombre des participants à une session est grand (des milliers) et que la plupart des participants s'y joignent ou la quittent au même moment. Cela rend difficiles les essais dans un réseau vivant. Cependant, l'algorithme a été soumis à une analyse et des simulations serrées pour vérifier ses performances. De plus, l'algorithme amélioré a été conçu pour interopérer ave l'algorithme de la RFC 1889 de telle sorte que le degré de réduction de l'excès de bande passante RTCP durant des adjonctions progressives soit proportionnel à la fraction de participants qui mettent en œuvre l'algorithme amélioré. L'interfonctionnement des deux algorithmes a été vérifié expérimentalement sur de vrais réseaux.

Les autres changements fonctionnels sont :

- Le paragraphe 6.2.1 spécifie que les mises en œuvre ne peuvent mémoriser qu'un échantillon des identifiants de SSRC des participants pour permettre de s'adapter aux très grandes sessions. Les algorithmes sont spécifiés dans la RFC 2762 [21].

- Au paragraphe 6.2 il est spécifié que les bandes passantes de l'envoyeur RTCP et du non envoyeur peuvent être réglées comme des paramètres séparés de la session plutôt qu'un strict pourcentage de la bande passante de session, et qu'elle peuvent être réglées à zéro. L'exigence que RTCP était obligatoire pour les sessions RTP utilisant la diffusion groupée IP a été assouplie. Cependant, il a aussi été précisé qu'il N'EST PAS RECOMMANDÉ de désactiver RTCP.

- Dans les paragraphes 6.2, 6.3.1 et à l'Appendice A.7, il est spécifié que la fraction de participants en dessous de laquelle les envoyeurs obtiennent une bande passante RTCP dédiée change du quart fixé à un ratio fondé sur les paramètres de bande passante de l'envoyeur RTCP et du non-envoyeur lorsque ceux–ci sont donnés. La condition qu'aucune bande passante ne soit dédiée aux envoyeurs lorsque il n'y a pas d'envoyeur a été retirée car il est prévu que ce soit un état transitoire. Cela empêche aussi les non-envoyeurs d'utiliser la bande passante d'envoyeur RTCP lorsque ce n'est pas prévu.

- Aussi au paragraphe 6.2 il est spécifié que l'intervalle RTCP minimum peut s'échelonner sur de plus petites valeurs pour les sessions à grosse largeur de bande, et que le délai initial RTCP peut être réglé à zéro pour les sessions en envoi individuel.

- L'arrivée à expiration de temporisation d'un participant se fonde sur l'inactivité pendant un certain nombre d'intervalles de rapport RTCP calculé en utilisant la fraction de bande passante RTCP de receveur même pour les envoyeurs actifs.

- Les paragraphes 7.2 et 7.3 spécifient que traducteurs et mixeurs devraient envoyer des paquets BYE pour les sources qu'ils ne transmettent plus.

- La règle change pour les codages en couche qui sont définis aux paragraphes 2.4, 6.3.9, 8.3 et à la Section 11. Dans cette dernière, il est noté que la règle d'allocation d'adresse et d'accès est en conflit avec la spécification de SDP, la RFC 2327 [15], mais il est prévu que cette restriction sera assouplie par une révision de la RFC 2327.

- La convention pour l'utilisation de paires d'accès pairs/impairs pour RTP et RTCP à la Section 11 a été précisée afin de se référer aux accès de destination. L'exigence d'utiliser une paire d'accès pair/impair a été retirée si les deux accès sont spécifiés explicitement. Pour les sessions RTP en envoi individuel, des paires d'accès distincts peuvent être utilisées pour les deux extrémités (Sections 3 et 11 et paragraphe 7.1).

- Une nouvelle Section 10 a été ajoutée pour expliquer l'exigence du contrôle d'encombrement dans les applications utilisant RTP.

- Au paragraphe 8.2, l'exigence qu'un nouvel identifiant de SSRC DOIVE être choisi chaque fois qu'est changée l'adresse de transport de source a été assouplie pour dire qu'un nouvel identifiant de SSRC PEUT être choisi. En conséquence, il a été précisé qu'une mise en œuvre PEUT choisir de garder des paquets provenant de la nouvelle adresse de source plutôt que de l'adresse de source existante lorsque une collision de SSRC survient entre deux autres participants, et DEVRAIT faire ainsi pour des applications telles que de téléphonie dans lesquelles certaines sources telles que des entités mobiles peuvent changer d'adresse durant le cours d'une session RTP.

- Une erreur de mise en page dans le pseudo-code de la RFC 1889 pour l'algorithme de détection et résolution de détection de collision du paragraphe 8.2 a été corrigée en retraduisant la syntaxe en pseudo langage C, et l'algorithme a été modifié pour retirer la restriction que RTP et RTCP doivent tous deux être envoyés du même numéro d'accès de source.

- La description du mécanisme de bourrage pour les paquets RTCP a été précisée et il est spécifié que le bourrage ne DOIT être appliqué que au dernier paquet d'un paquet RTCP composé.

- Au paragraphe A.1, l'initialisation de "base_seq" a été corrigée en "seq" plutôt que "seq – 1", et le texte a été corrigé pour dire que le mauvais numéro de séquence plus 1 est mémorisé. L'initialisation de "max_seq" et des autres variables pour l'algorithme a été séparée du texte pour rendre clair que cette initialisation doit être faite en plus de l'invocation de la fonction init_seq() (et quelques mots perdus dans la RFC 1889 lors du traitement du document entre la source à la forme publiée ont été restitués).

- La fixation du nombre de paquets perdus au paragraphe A.3 a été corrigée pour utiliser des limites à la fois positives et négatives.

- La spécification de horodatage NTP "relatif" dans le paragraphe SR RTCP définit maintenant ces horodatages comme étant fondés sur l'horloge commune spécifique du système, telle que le moment d'éveil du système, plutôt que sur le temps écoulé pour la session qui ne serait pas le même pour plusieurs applications commencées sur la même machine à des moments différents.

Changements non fonctionnels :

- Il est spécifié qu'un receveur DOIT ignorer les paquets avec des types de charge utile qu'il ne comprend pas.

- à la Figure 2, la virgule flottante de la valeur de l'horodatage NTP a été corrigée, quelques zéros en tête manquants ont été ajoutés dans un nombre hexadécimal, et le fuseau horaire UTC a été spécifié.

- L'incohérence du retour à zéro de l'horodatage NTP autour de l'an 2036 est expliquée.

- La politique d'enregistrement des types de paquet RTCP et des types de SDES a été précisée dans une nouvelle Section 15, Considérations relatives à l'IANA. La suggestion que les auteurs d'expériences enregistrent les numéros dont ils ont besoin puis désenregistrent ceux qui se révèlent inutiles a été retirée en faveur de l'utilisation de APP et PRIV. L'enregistrement des noms de profils a aussi été spécifié.

- La référence au jeu de caractères UTF-8 a été changée de la spécification préliminaire X/Open à la RFC 2279.

- La référence à la RFC 1597 a été mise à jour avec la RFC 1918 et la référence à la RFC 2543 en RFC 3261.

- Le dernier paragraphe de l'introduction de la RFC 1889, qui avertissait les développeurs de limiter les mises en œuvre dans l'Internet a été retiré parce qu'il n'est plus réputé pertinent.

- Une note non normative concernant l'utilisation de RTP avec la diffusion groupée spécifique de source (SSM, *Source-Specific Multicast*) a été ajoutée à la Section 6.

- La définition de "session RTP" à la Section 3 a été étendue pour rendre compte qu'une seule session peut utiliser plusieurs adresses de transport de destination (comme cela a toujours été le cas pour un traducteur ou mixeur) et expliquer que les caractéristiques distinctives d'une session RTP sont que chacune corresponde à un espace d'identifiants de SSRC distinct. Une nouvelle définition de "session multimédia" a été ajoutée pour réduire la confusion sur le mot "session".

- La signification de "moment d'échantillonnage" a été expliquée plus en détails au titre de la définition du champ horodatage de l'en-tête RTP au paragraphe 5.1.

- De petites précisions du texte ont été apportées à plusieurs endroits, certaines en réponse à des questions de lecteurs, en particulier :

- Dans la RFC 1889, les cinq premiers mots de la seconde phrase du paragraphe 2.2 avaient été perdus lors du traitement du document et sont maintenant restaurés.

- Une définition de "type de support RTP" était ajoutée à la Section 3 pour permettre à l'explication du multiplexage des sessions RTP au paragraphe 5.2 d'être plus claire concernant le multiplexage de plusieurs supports. Ce paragraphe explique maintenant aussi que le multiplexage de plusieurs sources du même support sur la base des identifiants de SSRC peut être appropriée et est la norme pour les sessions en diffusion groupée.

- La définition de "moyens non RTP" a été étendue pour inclure des exemples d'autres protocoles constituant des moyens non RTP.

- La description du paramètre Bande passante de session est étendue au paragraphe 6.2, incluant la précision que la bande passante de trafic de contrôle est en plus de l'ajout de la bande passante de session pour le trafic de données.

- L'effet d'une durée de paquet variable sur le calcul de la gigue a été expliqué au paragraphe 6.4.4.

- La méthode de terminaison et de bourrage d'une séquence d'éléments de SDES a été précisée au paragraphe 6.5.

- Des exemples d'adresse IPv6 ont été ajoutés dans la description du CNAME SDES au paragraphe 6.5.1, et "example.com" est utilisé à la place des autres exemples de nom de domaines.

- La Section Sécurité ajoute une référence formelle à IPSEC maintenant qu'il est disponible, et dit que la méthode de confidentialité définie dans cette spécification est principalement pour codifier les pratiques existantes. Il est RECOMMANDÉ que de plus forts algorithmes de chiffrement tels que Triple-DES soient utilisés à la place de l'algorithme par défaut, et noté que le profil SRTP fondé sur AES sera le choix correct à l'avenir. Un avertissement sur la faiblesse de l'en-tête RTP comme vecteur d'initialisation a été ajouté. Il a aussi été note que le chiffrement de la seule charge utile est nécessaire pour permettre la compression d'en-tête.

- La méthode pour le chiffrement partiel de RTCP a été précisée ; en particulier, le CNAME SDES est porté seulement dans une partie lorsque le paquet RTCP composé est partagé.

- Il est précisé que un seul paquet RTCP composé devrait être envoyé par intervalle de rapport et que si il y a trop de sources actives pour que les rapports tiennent dans la MTU, un sous-ensemble des sources devrait alors être choisi par round-robin sur plusieurs intervalles.

- Une note a été ajoutée à l'Appendice A.1 disant que les paquets peuvent être sauvegardés durant la validation d'en-tête RTP et délivrés en cas de succès.

- Le paragraphe 7.3 explique maintenant qu'un mixeur agrégeant des paquets de SDES utilise plus de bande passante RTCP du fait de paquets plus longs, et qu'un mixeur passant à travers RTCP envoie naturellement les paquets à un taux plus élevé que la seule source, mais que les deux comportements sont valides.

- La Section 13 précise qu'une application RTP peut utiliser plusieurs profils mais normalement un seul dans une session donnée.

- Les termes DOIT, DEVRAIT, PEUT, etc. sont utilisés comme défini dans la RFC 2119.

- La bibliographie a été divisée en références normatives et informatives.

Références

Références normatives

[1] H. Schulzrinne et S. Casner, "Profil RTP pour conférences audio et vidéo avec contrôle minimal", STD 65,

RFC 3551 , juillet 2003.
[2] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, RFC 2119 , mars 1997.
[3] J. Postel, éd., "Protocole Internet - Spécification du protocole du programme Internet DARPA", STD 5, RFC 791 , septembre 1981.
[4] D. Mills, "Spécification du protocole de l'heure du réseau (version 3) : Mise en œuvre et analyse", RFC 1305 , STD 12, mars 1992.
[5] F. Yergeau, "UTF-8, un format de transformation de la norme ISO 10646", RFC 2279, janvier 1998. (*Obsolète, voir* RFC3629) *(D.S.)*
[6] P. Mockapetris, USC/Information Sciences Institute, "Noms de domaines - Concepts et facilités", STD 13, RFC 1034 , novembre 1987.
[7] P. Mockapetris, USC/Information Sciences Institute, "Noms de domaines – Mise en œuvre et spécification", STD 13, RFC 1035 , novembre 1987.
[8] R. Braden, éditeur, "Exigences pour les hôtes Internet – Application et prise en charge", STD 3, RFC 1123 , octobre 1989.

[9] P. Resnick, "Format de message Internet", RFC 2822 , avril 2001. *(Remplacée par RFC5322)*

Références pour information

[10] Clark, D. et D. Tennenhouse, "Architectural Considerations for a New Generation of Protocols," in SIGCOMM Symposium on Communications Architectures et Protocols , (Philadelphia, Pennsylvania), pp. 200--208, IEEE Computer Communications Review, Vol. 20(4), septembre 1990.
[11] H. Schulzrinne, H., "Issues in designing a transport protocol for audio et video conferences and other multiparticipant real-time applications." projet Internet abandonné, octobre 1993.

[12] Comer, D., Internetworking with TCP/IP , vol. 1. Englewood Cliffs, New Jersey: Prentice Hall, 1991.

[13] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley et E. Schooler, "SIP : Protocole d'initialisation de session", RFC 3261 , juin 2002.

[14] Union internationale des télécommunications, Secteur de la normalisation, Recommandation UIT-T H.323, "Système et équipement de visiophonie pour les réseaux de zone locale qui fournissent une qualité de service non garantie", Genève, Confédération Helvétique, juillet 2003.

[15] M Handley et V. Jacobson, "SDP : Protocole de description de session", RFC 2327 , avril 1998.
[16] H. Schulzrinne, A. Rao et R. Lanphier, "Protocole de flux directs en temps réel (RTSP)", RFC 2326 , avril 1998.
[17] D. Eastlake, 3rd, S. Crocker et J. Schiller, "Recommandations d'aléa pour la sécurité", RFC 1750, décembre 1994. *(Information, remplacée par la* RFC 4086 *)*

[18] Bolot, J.-C., Turletti, T. et I. Wakeman, "Scalable Feedback Control for Multicast Video Distribution in the Internet", in SIGCOMM Symposium on Communications Architectures et Protocols, (London, England), pp. 58--67, ACM, août 1994.
[19] Busse, I., Deffner, B. et H. Schulzrinne, "Dynamic QoS Control of Multimedia Applications Based on RTP", Computer Communications , vol. 19, pp. 49--58, janvier 1996.
[20] Floyd, S. et V. Jacobson, "The Synchronization of Periodic Routing Messages", in SIGCOMM Symposium on Communications Architectures et Protocols (D. P. Sidhu, ed.), (San Francisco, California), pp. 33--44, ACM, septembre 1993. Aussi dans [34].

[21] J. Rosenberg et H. Schulzrinne, "Échantillonnage de la taille du groupe dans RTP", RFC 2762 , février 2000.

[22] Cadzow, J., Foundations of Digital Signal Processing et Data Analysis New York, New York: Macmillan, 1987.

[23] R. Hinden et S. Deering, "Architecture d'adressage du protocole Internet version 6 (IPv6)", RFC 3513 , avril 2003.
[24] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. de Groot et E. Lear, "Allocation d'adresse pour les internets privés", RFC 1918, février 1996.
[25] E. Lear, E. Fair, D. Crocker et T. Kessler, "10 réseaux considérés comme dangereux (certaines pratiques ne devraient pas être codifiées)", RFC 1627, juillet 1994. *(Rendue obsolète par la* RFC 1918*)*

[26] Feller, W., An Introduction to Probability Theory et its Applications, vol. 1. New York, New York : John Wiley et Sons, troisième édition, 1968.

[27] S. Kent et R. Atkinson, "Architecture de sécurité pour le protocole Internet", RFC 2401 , novembre 1998.
[28] M. Baugher, D. McGrew, M. Naslund, E. Carrara et K. Norrman, "Protocole de transport sécurisé en temps réel (SRTP)", RFC 3711 , mars 2004.

[29] D. Balenson, D., "Amélioration de la confidentialité pour la messagerie électronique Internet : Partie III --

Algorithmes, modes et identifiants", RFC 1423, février 1993. *(Historique)*
[30] Voydock, V. et S. Kent, "Security Mechanisms in High-Level Network Protocols", ACM Computing Surveys, vol. 15, pp. 135-171, juin 1983.
[31] Floyd, S., "Principes du contrôle d'encombrement", BCP 41, RFC 2914, septembre 2000.
[32] R. Rivest, "Algorithme de résumé de message MD5", RFC 1321 , avril 1992.
[33] Stubblebine, S., "Security Services for Multimedia Conferencing", dans 16th National Computer Security Conference, (Baltimore, Maryland), pp. 391--395, septembre 1993.
[34] Floyd, S. et V. Jacobson, "The Synchronization of Periodic Routing Messages", IEEE/ACM Transactions on Networking, vol. 2, pp. 122--136, avril 1994.

Adresse des auteurs

Henning Schulzrinne
Department of Computer Science
Columbia University
1214 Amsterdam Avenue
New York, NY 10027
United States
mél : schulzrinne@cs.columbia.edu

Stephen L. Casner
Packet Design
3400 Hillview Avenue, Building 3
Palo Alto, CA 94304
United States
mél : casner@acm.org

Ron Frederick
Blue Coat Systems Inc.
650 Almanor Avenue
Sunnyvale, CA 94085
United States
mél : ronf@bluecoat.com

Van Jacobson
Packet Design
3400 Hillview Avenue, Building 3
Palo Alto, CA 94304
United States
mél : van@packetdesign.com

Déclaration complète de droits de reproduction

Remerciement
Le financement de la fonction d'édition des RFC est actuellement fourni par l'Internet Society.