Groupe de travail Réseau
Request for Comments : 3095
Catégorie : En cours de normalisation
juillet 2001

C. Bormann, éditeur, TZI/Uni Bremen
C. Burmeister, T. Wiebke, Matsushita
M. Degermark, Univ. of Arizona
H. Fukushima, A. Miyazaki & R. Hakenberg, Matsushita
H. Hannu, A. Martensson & L-E. Jonsson, Ericsson
T. Koren, Cisco
K. Le, Z. Liu, H. Zheng, Nokia
K. Svanbro, Ericsson
T. Yoshimura, NTT DoCoMo

Traduction Claude Brière de L'Isle

Compression d'en-tête robuste (ROHC) : cadre et quatre profils : RTP, UDP, ESP, et non compressé

Statut de ce mémoire
Le présent document spécifie un protocole Internet en cours de normalisation pour la communauté de l'Internet, et appelle à des discussions et des suggestions pour son amélioration. Prière de se reporter à l'édition actuelle du STD 1 "Normes des protocoles officiels de l'Internet" pour connaître l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

Résumé
Le présent document spécifie un schéma de compression d'en-tête très robuste et efficace pour les en-têtes du protocole de transport en temps réel (RTP, *Real-Time Transport Protocol*) du protocole de datagramme d'utilisateur (UDP, *User Datagram Protocol*) du protocole Internet (IP, *Internet Protocol*) et du protocole d'encapsulation de charge utile de sécurité sur IP (ESP/IP, *Encapsulating Security Payload*).

Les schémas existants de compression d'en-tête ne fonctionnent pas bien lorsque ils sont utilisés sur des liaisons avec des taux d'erreur significatifs et de longs délais d'aller retour. Pour de nombreuses liaisons à bande passante limitée où la compression d'en-tête est essentielle, de telles caractéristiques sont courantes.

Ceci est fait dans un cadre conçu pour être extensible. Par exemple, un schéma pour la compression d'en-têtes TCP/IP sera simple à ajouter et est en cours de développement. Les en-têtes spécifiques d'IPv4 mobile ne sont pas soumis à un traitement particulier mais on s'attend à ce qu'ils soient suffisamment bien compressés par la méthode fournie pour la compression de séquences d'en-têtes d'extension et d'en-têtes de tunnelage. Pour la plus grande part, la même chose va s'appliquer au travail en cours sur IPv6 mobile, mais des travaux complémentaires pourraient être nécessaires pour traiter certains en-têtes d'extension, lorsque le projet IPv6 mobile aura été amené au stade de la normalisation.

Table des matières

1. Introduction

Durant les cinq dernières années, deux technologies de communication en particulier sont devenues d'usage courant pour le public : la téléphonie cellulaire et l'Internet. La téléphonie cellulaire a apporté à ses utilisateurs la possibilité révolutionnaire d'être toujours accessible avec une qualité de service raisonnable quel que soit l'endroit où ils se trouvent. Le principal service fourni par les terminaux dédiés est vocal. L'Internet, d'un autre côté, a depuis le début été conçu pour de multiples services et sa souplesse pour toutes sortes d'usages a été une de ses forces. Les terminaux Internet ont habituellement été multi-usages et ont été rattachés par des connexions fixes. La qualité subie par certains services (comme la téléphonie Internet) a parfois été faible.

Aujourd'hui, la téléphonie IP est plus au point, grâce à des solutions techniques améliorées. Il semble raisonnable de croire que dans les années à venir, IP va devenir une façon courante de porter la téléphonie. Certaines futures liaisons de téléphonie cellulaire pourraient aussi se fonder sur IP et la téléphonie IP.
Les téléphones cellulaires ont pu devenir d'utilisation plus générale, et peuvent avoir des piles IP qui prennent en charge non seulement l'audio et la vidéo, mais aussi la navigation sur la Toile, la messagerie électronique, les jeux en ligne, etc.

Un des scénarios qu'on envisage pourrait alors être celui de la Figure 1.1, où deux terminaux mobiles communiquent ensemble. Tous deux sont connectés à des stations de base sur des liaisons cellulaires, et les stations de base sont connectées l'une à l'autre par un réseau filaire (ou éventuellement sans fil). Au lieu de deux terminaux mobiles, il pourrait bien sûr y avoir un mobile et un fixe, mais le cas avec deux liaisons cellulaires est techniquement plus exigeant.

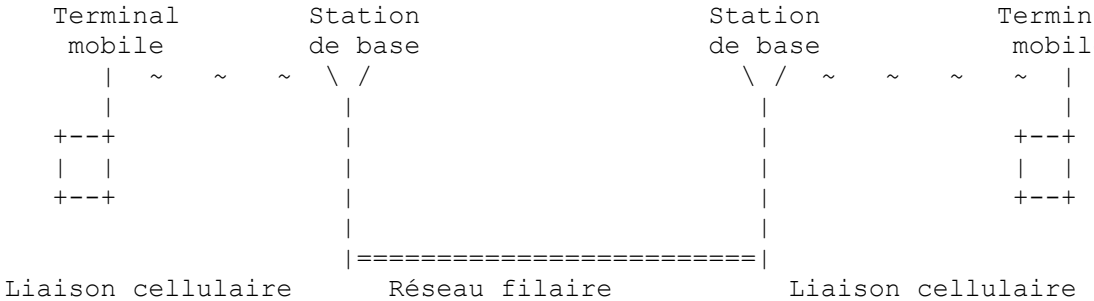


Figure 1.1 : Scénario pour téléphonie IP sur liaisons cellulaires

Il est évident que le réseau filaire peut être fondé sur IP. Avec les liaisons cellulaires, la situation est moins claire. IP pourrait se terminer dans le réseau fixe, et des solutions particulières mises en œuvre pour chaque service pris en charge sur la liaison cellulaire. Cependant, cela limiterait la souplesse des services pris en charge. Si elle était techniquement et économiquement faisable, une solution de pur IP tout le long du chemin de terminal à terminal aurait des avantages certains. Cependant, pour faire de cela une alternative viable, un certain nombre de problèmes doivent être réglés, en particulier ceux qui concernent l'efficacité de la bande passante.

Pour les systèmes de téléphonie cellulaire, il est d'une importance vitale d'utiliser les rares ressources radioélectriques d'une façon efficace. Un nombre suffisant d'utilisateurs par cellule est crucial, autrement les coûts de déploiement seront prohibitifs. La qualité du service vocal devrait aussi être aussi bonne que dans les systèmes cellulaires d'aujourd'hui. Il est vraisemblable que même avec la prise en charge de nouveaux services, une plus faible qualité du service vocal n'est acceptable que si les coûts sont significativement réduits.

Un problème avec IP sur liaisons cellulaires lorsque utilisé pour des conversations vocales interactives est celui de la grande redondance d'en-tête. Les données de parole pour la téléphonie IP seront très probablement portées par RTP [RFC1889]. Un paquet va alors, en plus du tramage de couche liaison, avoir un en-tête IP [RFC0791] (20 octets) un en-tête UDP [RFC0768] (8 octets) et un en-tête RTP (12 octets) pour un total de 40 octets. Avec IPv6 [RFC2460], l'en-tête IP fait 40 octets pour un total de 60 octets. La taille de la charge utile dépend du codage de la parole et des tailles de trame utilisées et peut être aussi basse que 15 à 20 octets.

D'après ces chiffres, la nécessité d'une réduction des tailles d'en-tête pour des raisons d'efficacité est évidente. Cependant, les liaisons cellulaires ont des caractéristiques qui font que la compression d'en-tête telle que définie dans les [RFC2507], [RFC2508] fonctionne moins que bien. La caractéristique la plus importante est le comportement enclin aux pertes des liaisons cellulaires, où un débit d'erreurs binaires (BER, *bit error rate*) aussi élevé que 1e-3 doit être accepté pour garder le niveau d'efficacité de l'utilisation des ressources radioélectriques. Dans des situations de fonctionnement sévères, le BER peut s'élever à 1e-2. L'autre caractéristique problématique est la longueur du temps d'aller-retour (RTT, *round-trip time*) de la liaison cellulaire, qui peut s'élever à 100-200 millisecondes. Un problème supplémentaire est que le BER résiduel n'est pas trivial, c'est-à-dire que les couches inférieures peuvent parfois délivrer des trames qui contiennent des erreurs non détectées. Un schéma de compression d'en-tête viable pour les liaisons cellulaires doit être capable de traiter les pertes sur la liaison entre les points de compression et de décompression tout autant que les pertes avant le point de compression.

La bande passante est la ressource la plus coûteuse dans les liaisons cellulaires. La puissance de traitement est très bon marché en comparaison. La simplicité de mise en œuvre ou de calcul d'un schéma de compression d'en-tête est donc de moindre importance que son taux de compression et sa robustesse.

2. Terminologie

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT" et "FACULTATIF" dans ce document sont à interpréter comme décrit dans la [RFC2119].

BER (*Bit Error Rate*) Taux d'erreurs binaires : Les liaisons radio cellulaires peuvent avoir un BER très élevé. Dans le présent document le BER est usuellement donné comme une probabilité, mais on doit aussi considérer la distribution des erreurs car les erreurs binaires ne sont pas indépendantes.

Liaisons cellulaires : Ce sont des liaisons sans fils entre les terminaux mobiles et les stations de base.

Efficacité de compression : Les performances d'un schéma de compression d'en-tête peuvent être décrites par trois paramètres : l'efficacité de compression, la robustesse et la transparence de compression. L'efficacité de compression est déterminée par combien de la taille d'en-tête peut être réduit par le schéma de compression.

Transparence de compression : Les performances d'un schéma de compression d'en-tête peuvent être décrites par trois paramètres : l'efficacité de compression, la robustesse et la transparence de compression. La transparence de compression est la mesure dans laquelle le schéma assure que les en-têtes décompressés sont sémantiquement identiques aux en-têtes d'origine. Si tous les en-têtes décompressés sont sémantiquement identiques aux en-têtes d'origine correspondants, la transparence est de 100 pour cent. La transparence de compression est élevée lorsque les dommages de propagation sont faibles.

Contexte : Le contexte du compresseur est l'état qu'il utilise pour compresser un en-tête. Le contexte du décompresseur est l'état qu'il utilise pour décompresser un en-tête. L'un ou l'autre ou les deux combinés sont usuellement appelés le "contexte", lorsque celui qui est visé apparaît clairement. Le contexte contient des informations pertinentes

provenant d'en-têtes précédents dans le flux de paquets, comme des champs statiques et d'éventuelles valeurs de référence pour la compression et la décompression. De plus, des informations supplémentaires pour décrire le flux de paquets font aussi partie du contexte, par exemple, des informations sur la façon dont change le champ Identifiant IP et l'augmentation typique des numéros de séquence ou des horodatages entre les paquets.

Dommage de contexte : Lorsque le contexte du décompresseur n'est pas cohérent avec le contexte du compresseur, la décompression peut ne pas réussir à reproduire l'en-tête original. Cette situation peut survenir lorsque le contexte du décompresseur n'a pas été initialisé correctement ou lorsque des paquets ont été perdus ou endommagés entre le compresseur et le décompresseur. Les paquets qui ne peuvent pas être décompressés du fait de contextes incohérents sont dits perdus à cause d'un dommage de contexte. Les paquets qui sont décompressés mais contiennent des erreurs dues à des contextes incohérents sont dits être endommagés à cause d'un dommage de contexte.

Mécanisme de réparation de contexte : Ce sont des mécanismes qui amènent les contextes à se synchroniser lorsque ils ne le sont pas. Cela est nécessaire pour éviter des pertes excessives dues à des dommages de contexte. Des exemples sont le mécanisme de réparation de contexte de CRTP, les mécanismes de NACK de mode O et R, et le rafraîchissement périodique du mode U.
Noter qu'il y a aussi des mécanismes qui empêchent (certaines) incohérences de contexte de se produire, par exemple la mise à jour fondée sur l'accusé de réception du contexte en mode R, les répétitions après changement en mode U et O, et les CRC qui protègent les informations de mise à jour de contexte.

CRC-DYNAMIC : Opposé de CRC-STATIC.

CRC-STATIC : Un CRC sur l'en-tête d'origine est le principal mécanisme utilisé par ROHC pour détecter une décompression incorrecte. Afin de diminuer la complexité de calcul, les champs de l'en-tête sont réarrangés conceptuellement lorsque le CRC est calculé, de sorte qu'il est d'abord calculé sur les octets qui sont statiques (appelés CRC-STATIC dans ce document) et ensuite sur les octets dont on s'attend à ce que leur valeur change entre les paquets (CRC-DYNAMIC). De cette manière, le résultat intermédiaire du calcul du CRC, après qu'il a couvert les champs CRC-STATIC, peut être réutilisé pour plusieurs paquets. Le calcul de CRC recommencé ne couvre que les octets de CRC-DYNAMIC. Voir au paragraphe 5.9.

Propagation de dommage : Livraison d'en-têtes décompressés incorrects, due à des erreurs dans de précédents en-têtes (c'est-à-dire, des pertes ou des dommages) ou des retours.

Propagation de perte : Perte d'en-têtes, due à des erreurs (c'est-à-dire, des pertes ou des dommages) dans des en-têtes ou retours précédents.

Détection d'erreur : Détection des erreurs. Si la détection d'erreur n'est pas parfaite, il y aura des erreurs résiduelles.

Propagation d'erreur : Propagation de dommage ou propagation de perte.

Profil de compression d'en-tête : C'est une spécification de la façon de compresser l'en-tête d'une certaine sorte de flux de paquets sur une certaine sorte de liaison. Les profils de compression fournissent les détails du cadre de compression d'en-tête introduit dans le présent document. Le concept de profil utilise des identifiants de profil pour séparer différents profils qui sont utilisés lors de l'établissement du schéma de compression. Toutes les variations et tous les paramètres du schéma de compression d'en-tête qui ne font pas partie de l'état de contexte sont traités par des identifiant de profil différents.

Paquet : Généralement, une unité de transmission et réception (unités de données de protocole). Précisément, lorsque on l'oppose à une "trame", le paquet compressé et ensuite décompressé par ROHC. Aussi appelé "paquet non compressé".

Flux de paquets : Séquence de paquets où les valeurs de champs et schémas de changement des valeurs de champs sont tels que les en-têtes puissent être compressés en utilisant le même contexte.

Liaisons pré-HC : Ce sont toutes les liaisons qu'un paquet a traversé avant le point de compression d'en-tête. Si on considère un chemin avec des liaisons cellulaires comme premier et dernier bonds, les liaisons pré-HC pour le compresseur à la dernière liaison sont la première liaison cellulaire plus les liaisons filaires incluses.

Erreur résiduelle : Erreur introduite durant la transmission et non détectée par les schémas de détection d'erreur de couche inférieure.

Robustesse ; Les performances d'un schéma de compression d'en-tête peuvent être décrites par trois paramètres : l'efficacité de compression, la robustesse et la transparence de compression. Un schéma robuste tolère des

pertes et des erreurs résiduelles sur la liaison sur laquelle a lieu la compression d'en-tête sans perdre de paquets supplémentaires ou sans introduire d'erreurs supplémentaires dans les en-têtes décompressés.

Délai d'aller-retour (RTT) : Le RTT *(round-trip time)* est le temps écoulé entre le moment où le compresseur envoie un paquet jusqu'à ce qu'il reçoive un retour en rapport avec ce paquet (lorsque un tel retour est envoyé).

Efficacité spectrale : Les ressources radioélectriques sont limitées et chères. Elles doivent donc être utilisées efficacement pour rendre le système économiquement viable. Dans les systèmes cellulaires, cela se fait en maximisant le nombre d'usagers desservis au sein de chaque cellule, tandis que la qualité des services fournis est gardée à un niveau acceptable. Une conséquence de l'utilisation efficace du spectre est un taux d'erreurs élevé (perte de trames et erreurs binaires résiduelles) même après le codage du canal avec correction d'erreur.

Chaîne : Séquence d'en-têtes dans laquelle la valeur de tous les champs compressés change selon un schéma qui est fixé par rapport à un numéro de séquence. Chaque en-tête dans une chaîne peut être compressé en le représentant par un en-tête ROHC qui ne porte essentiellement qu'un numéro de séquence codé. Les champs qui ne sont pas compressés (par exemple, un identifiant IP aléatoire, une somme de contrôle UDP) ne sont pas pertinents pour cette définition.

Pas d'horodatage : (TS_STRIDE) C'est l'augmentation attendue de la valeur de l'horodatage entre deux paquets RTP qui ont des numéros de séquence consécutifs.

2.1 Acronymes

Ce paragraphe fait la liste de la plupart des acronymes utilisés pour référence.

AH *(Authentication Header)* : En-tête d'authentification.
CID *(Context Identifier)* : Identifiant de contexte
CRC *(Cyclic Redundancy Check)* : Contrôle de redondance cyclique, mécanisme de détection d'erreur.
CRTP *(Compressed RTP)* : RTP compressé [RFC2508].
CSRC *(Contributing source)* : Source contributive. Liste facultative des CSRC dans un en-tête RTP.
CTCP *(Compressed TCP)* : TCP compressé. Aussi appelé compression d'en-tête VJ. [RFC1144]
ESP *(Encapsulating Security Payload)* : Encapsulation de charge utile de sécurité
FC *(Full Context)* : État de contexte plein (au décompresseur).
FO *(First Order)* : État de premier ordre (au compresseur).
GRE *(Generic Routing Encapsulation)* : Encapsulation générique d'acheminement [RFC2784], [RFC2890].
HC *(Header Compression)* : Compression d'en-tête
IPHC *(IP Header Compression)* : Compression d'en-tête IP [RFC2507]
IPX *(Flag in Extension 2)* : Fanion en extension 2
IR *(Initiation and Refresh)* : État d'initialisation et de rafraîchissement (au compresseur). Aussi paquet IR.
IR-DYN : Paquet IR-DYN.
LSB *(Least Significant Bits)* : Bits de moindre poids
MRRU *(Maximum Reconstructed Reception Unit)* : Unité maximum de réception reconstruite
MTU *(Maximum Transmission Unit)* : Unité maximum de transmission
MSB *(Most Significant Bits)* : Bits de poids fort
NBO *(Network Byte Order)* : Fanion qui indique si le IP-ID est dans l'ordre des octets du réseau
NC *(No Context)* : État sans contexte (au décompresseur).
O-mode : Mode optimiste bidirectionnel.
PPP *(Point-to-Point Protocol)* : Protocole point à point
R-mode : Mode bidirectionnel fiable.
RND : Fanion qui indique si le IP-ID se comporte au hasard.
ROHC *(RObust Header Compression)* : Compression d'en-tête robuste
RTCP *(Real-Time Control Protocol)* : Protocole de contrôle en temps réel ; voir RTP.
RTP *(Real-Time Protocol)* : Protocole de temps réel [RFC1889]
RTT *(Round Trip Time)* : Délai d'aller-retour (voir la section 2).
SC *(Static Context)* : État de contexte statique (au décompresseur).
SN *(Sequence Number)* : Numéro de séquence (compressé). Normalement un numéro de séquence RTP.
SO *(Second Order)* : État de second ordre (au compresseur).
SPI *(Security Parameters Index)* : Indice de paramètre de sécurité
SSRC *(Sending source)* : Source d'envoi. Champ dans un en-tête RTP.
TC *(Traffic Class)* : Classe de trafic. Octet dans un en-tête IPv6. Voir aussi TOS.
TOS (Type Of Service) : Type de service. Octet dans un en-tête IPv4. Voir aussi TC.

TS *(Timestamp)* : Horodatage (compressé) RTP.
U-mode : Mode unidirectionnel.
W-LSB *(Window based LSB encoding)* : Codage de LSB fondé sur la fenêtre. Voir le paragraphe 4.5.2.

3. Fondements

La présente section donne les fondements de la compression d'en-tête. Les idées fondamentales sont décrites avec les schémas de compression d'en-tête existants. Leurs inconvénients et leurs exigences sont ensuite exposés, avec les motivations des nouvelles solutions de compression d'en-tête.

3.1 Fondements de la compression d'en-tête

La principale raison de vouloir faire la compression d'en-tête est le fait qu'il y a une redondance significative entre les champs d'en-tête, aussi bien au sein du même en-tête de paquet mais surtout entre des paquets consécutifs qui appartiennent au même flux de paquets. En envoyant seulement initialement les informations d'en-tête statiques et en utilisant les dépendances et la prévisibilité pour les autres champs, la taille de l'en-tête peut être significativement réduite pour la plupart des paquets.

Les informations pertinentes provenant des paquets passés sont conservées dans un contexte. Les informations de contexte sont utilisées pour compresser (décompresser) les paquets suivants. Le compresseur et le décompresseur mettent à jour leurs contextes lors de certains événements. Les événements de dégradation peuvent conduire à des incohérences entre les contextes du compresseur et du décompresseur, qui à leur tour peuvent causer une décompression incorrecte. Un schéma de compression d'en-tête robuste a besoin de mécanismes pour éviter les incohérences de contexte et aussi pour rendre les contextes cohérents lorsque ils ne le sont pas.

3.2 Schémas existants de compression d'en-tête

Le schéma original de compression d'en-tête, CTCP [RFC1144], a été inventé par Van Jacobson. CTCP compresse les 40 octets d'en-tête IP+TCP à 4 octets. Le compresseur CTCP détecte les retransmissions de niveau transport et envoie un en-tête qui met à jour complètement le contexte lorsque elles se produisent. Ce mécanisme de réparation n'exige aucune signalisation explicite entre compresseur et décompresseur.

Un schéma général IP de compression d'en-tête, (IPHC, *IP Header Compression*) [RFC2507], améliore quelque peu CTCP et peut compresser des en-têtes arbitraires IP, TCP, et UDP. Lorsqu'il compresse des en-têtes non TCP, IPHC n'utilise pas de codage de delta et est robuste. Lorsque il compresse TCP, le mécanisme de réparation de CTCP est augmenté d'un schéma de non accusé de réception de niveau liaison qui accélère la réparation. IPHC ne compresse pas les en-têtes RTP.

CRTP [RFC2507], [RFC2508], de Casner et Jacobson, est un schéma de compression d'en-tête qui compresse 40 octets d'en-tête IPv4/UDP/RTP à un minimum de 2 octets lorsque la somme de contrôle UDP n'est pas activée. Si la somme de contrôle UDP est activée, l'en-tête CRTP minimum est de 4 octets. CRTP ne peut pas utiliser le même mécanisme de réparation que CTCP car UDP/RTP ne retransmet pas. À la place, CRTP utilise des messages de signalisation explicites du décompresseur au compresseur, appelés messages CONTEXT_STATE, pour indiquer que le contexte est hors synchronisation. Le temps d'aller-retour de la liaison va donc limiter la vitesse de ce mécanisme de réparation de contexte.

Sur les lignes à pertes avec de longs délais d'aller-retour, telles que la plupart des liaisons cellulaires, CRTP ne fonctionne pas bien. Chaque paquet perdu sur la liaison est cause de la perte de plusieurs paquets suivants parce que le contexte est hors de synchronisation durant au moins un délai d'aller-retour de liaison. Ce comportement est décrit dans [CRTPC]. Pour les conversations vocales de tels longs événements de perte vont dégrader la qualité vocale. De plus, de la bande passante est gâchée par les grands en-têtes envoyés par CRTP lors de la mise à jour du contexte. [CRTPC] a trouvé que CRTP n'a pas d'assez bonnes performances pour une liaison cellulaire à pertes. Il est clair que CRTP seul n'est pas un schéma de compression d'en-tête viable pour la téléphonie IP sur liaisons cellulaires.

Pour éviter de perdre des paquets à cause de la perte de synchronisation du contexte, les décompresseurs CRTP peuvent tenter de réparer le contexte en local en utilisant un mécanisme connu sous le nom de TWICE. Chaque paquet CRTP contient un compteur qui est incrémenté de un à chaque envoi de paquet par le compresseur CRTP. Si le compteur augmente de plus de un, au moins un paquet a été perdu sur la liaison. Le décompresseur tente alors de réparer le contexte en devinant comment le ou les paquets perdus l'auraient mis à jour. L'hypothèse est ensuite vérifiée en décompressant le paquet et en vérifiant la somme de contrôle UDP – si elle se vérifie, la réparation est réputée réussie et le paquet peut être

transmis ou livré. TWICE tient son nom de l'observation que lorsque le flux de paquets compressés est régulier, l'hypothèse correcte est d'appliquer deux fois la mise à jour au paquet en cours. [CRTPC] a trouvé que même avec TWICE, CRTP double le nombre de paquets perdus. TWICE améliore significativement les performances de CRTP. Cependant, l'utilisation de TWICE pose plusieurs problèmes :

1) Il devient obligatoire d'utiliser la somme de contrôle UDP :
- la taille minimale d'en-tête compressé augmente de 100 % à 4 octets.
- la plupart des codecs de parole développés pour les liaisons cellulaires tolèrent des erreurs dans les données codées. De tels codecs ne voudront pas activer la somme de contrôle UDP, car ils veulent que les paquets endommagés soient livrés.
- les erreurs dans la charge utile vont causer l'échec de la somme de contrôle UDP lorsque l'hypothèse est correcte (et pourraient la faire réussir lorsque l'hypothèse est fausse).

2) Une perte dans un flux RTP qui survient avant le point de compression rend les mises à jour dans les en-têtes CRTP moins régulières. Les versions les plus primaires de TWICE vont alors avoir des performances médiocres Des versions plus sophistiquées auraient besoin de plus de tentatives de réparation pour réussir.

3.3 Exigences pour un nouveau schéma de compression d'en-tête

Le problème majeur de CRTP est qu'il n'est pas suffisamment robuste contre les paquets endommagés entre le compresseur et le décompresseur. Un schéma viable de compression d'en-tête doit être moins fragile. Cette robustesse accrue doit être obtenue sans augmenter la taille de l'en-tête compressé ; un en-tête plus gros rendrait la téléphonie IP sur liaisons cellulaires économiquement inintéressante.

Une cause majeure des mauvaises performances de CRTP sur liaisons cellulaires est le long délai d'aller-retour de la liaison, durant lequel de nombreux paquets sont perdus lorsque le contexte est hors de synchronisation. Ce problème peut être attaqué directement en trouvant des moyens de réduire le délai d'aller-retour de la liaison. Des générations futures de technologies cellulaires pourront bien sûr réaliser de plus faibles délais d'aller-retour de liaison. Cependant, ceux-ci seront probablement toujours très élevés. Les avantages, en termes de plus faible perte et plus petite demande de bande passante si le contexte peut être réparé en local, seront présents même si le délai d'aller-retour de la liaison est diminué. Une façon fiable de détecter la réussite d'une réparation de contexte est alors nécessaire.

On pourrait avancer qu'une meilleure façon de résoudre le problème serait d'améliorer la liaison cellulaire de telle sorte que la perte de paquet ait une moindre probabilité de se produire. De telles modifications ont cependant un coût. Si les liaisons étaient rendues (presque) infaillibles, le système pourrait n'être pas capable de prendre en charge un nombre suffisamment grand d'utilisateurs par cellule et pourrait donc devenir économiquement infaisable.

On pourrait aussi avancer que les codecs vocaux devraient être capables de traiter la sorte de perte de paquet induite par CRTP, en particulier puisque les codecs vocaux doivent probablement être de toutes façons capables de traiter la perte de paquet si le flux RTP traverse l'Internet. Bien que ce dernier point soit vrai, la sorte de perte induite par CRTP est difficile à traiter. Il n'est usuellement pas possible de cacher complètement un événement de perte lorsque plus de 100 ms de son est complètement perdu. Si une telle perte survient fréquemment des deux côtés du chemin de bout en bout, la qualité vocale va en souffrir.

Une description détaillée des exigences spécifiées pour ROHC se trouve dans la [RFC3096].

3.4 Classification des champs d'en-tête

Comme mentionné précédemment, la compression d'en-tête est possible par le fait qu'il y a beaucoup de redondance entre les valeurs de champs d'en-tête au sein des paquets, mais particulièrement entre les paquets consécutifs. Pour utiliser ces propriétés pour la compression d'en-tête, il est important de comprendre les schémas de changements des divers champs d'en-tête.

Tous les champs d'en-tête ont été classés en détail dans l'Appendice A. Les champs sont d'abord classés à un haut niveau et ensuite, certains d'entre eux sont étudiés plus en détail. Finalement, l'appendice se conclut avec des recommandations sur la façon dont les divers champs devraient être traités par les algorithmes de compression d'en-tête. La principale conclusion qui peut être tirée est que la plupart des champs d'en-tête peuvent facilement être compressés car ils ne changent jamais ou rarement. Seuls cinq champs, avec une taille combinée d'environ 10 octets, ont besoin de mécanismes plus sophistiqués. Ces champs sont :

- Identification IPv4 (16 bits) - IP-ID
- Somme de contrôle UDP (16 bits)
- Marqueur RTP (1 bit) - M-bit

- Numéro de séquence RTP (16 bits) - SN
- Horodatage RTP (32 bits) - TS

L'analyse de l'Appendice A révèle que les valeurs des champs TS et IP-ID peuvent usuellement être prédites à partir du numéro de séquence RTP, qui s'incrémente de un pour chaque paquet émis par une source RTP. Le bit M est aussi normalement le même, mais a besoin occasionnellement d'être communiqué explicitement. La somme de contrôle UDP ne devrait pas être prédite et est envoyée telle qu'elle lorsque elle est activée.

La façon dont la compression RTP ROHC fonctionne est alors d'établir d'abord les fonctions à partir du SN vers les autres champs, et ensuite de communiquer fiablement le SN. Chaque fois que change une fonction du SN à un autre champ, c'est-à-dire, que la fonction existante donne un résultat qui est différent du champ dans l'en-tête à compresser, des informations supplémentaires sont envoyées pour mettre à jour les paramètres de cette fonction.

Les en-têtes spécifiques d'IP mobile (pour IPv4 ou IPv6) ne reçoivent aucun traitement particulier dans le présent document. Ils sont cependant compressibles, et on s'attend à ce que l'efficacité de compression pour les en-têtes IP mobile soit assez bonne à cause du traitement des listes d'extension d'en-tête et du tunnelage des en-têtes. Il serait relativement indolore d'introduire un nouveau profil ROHC avec un traitement particulier pour les en-têtes spécifiques de IPv6 mobile lorsque l'achèvement des travaux sur les protocoles IPv6 mobile (travaux en cours à l'IETF) le rendra nécessaire.

4. Cadre de la compression d'en-tête

4.1 Hypothèses de fonctionnement

Les liaisons cellulaires, qui sont une cible principale pour ROHC, ont un certain nombre de caractéristiques qui sont brièvement décrites ici. ROHC exige des fonctionnalités provenant des couches inférieures qui sont soulignées ici et décrites avec plus de précision dans le document sur les lignes directrices pour les couches inférieures [RFC3409].

Canaux
Les paquets dont les en-têtes sont compressés par ROHC s'écoulent sur des canaux. À la différence de nombreuses liaisons fixes, certaines liaisons radio cellulaires peuvent avoir plusieurs canaux qui connectent la même paire de nœuds. Chaque canal peut avoir des caractéristiques différentes en termes de taux d'erreur, de bande passante, etc.

Identifiants de contexte
Sur certains canaux est exigée la capacité à transporter plusieurs flux de paquets. Il peut aussi se faire qu'on ait des canaux dédiés à des flux de paquets individuels. Donc, ROHC utilise un espace d'identifiants de contexte distincts par canal et peut éliminer complètement des identifiants de contexte pour un des flux lorsque quelques flux partagent un canal.

Indication de type de paquet
L'indication du type de paquet est faite dans le schéma de compression d'en-tête lui-même. Sauf si la liaison a déjà un moyen d'indiquer les types de paquet qui peuvent être utilisés, comme PPP, cela assure de plus petits en-têtes compressés globalement. Il peut aussi être moins difficile d'allouer un seul type de paquet, plutôt que plusieurs, afin de faire fonctionner ROHC sur des liaisons telles que PPP.

Réarrangement
Le canal entre compresseur et décompresseur est obligé de maintenir l'ordre des paquets, c'est-à-dire que le décompresseur doit recevoir les paquets dans le même ordre que celui dans lequel le compresseur les a envoyé. (Cependant, le réarrangement avant le point de compression est pris en charge, c'est-à-dire qu'on ne suppose pas que le compresseur va seulement recevoir les paquets en séquence.)

Duplication
Le canal entre compresseur et décompresseur a l'obligation de ne pas dupliquer les paquets. (Cependant, la duplication avant le point de compression est prise en charge, c'est-à-dire qu'on ne suppose pas que le compresseur va recevoir seulement une copie de chaque paquet.)

Longueur de paquet
ROHC est conçu avec l'hypothèse que les couches inférieures indiquent la longueur d'un paquet compressé. Les paquets ROHC ne contiennent pas d'informations de longueur pour la charge utile.

Tramage
La couche de liaison doit fournir le tramage qui rend possible de distinguer les frontières de trame et les trames individuelles.

Détection/protection d'erreur
Le schéma de ROHC a été conçu pour venir à bout des erreurs résiduelles dans les en-têtes livrés au décompresseur. Les CRC et les vérifications de bonne santé sont utilisés pour empêcher ou réduire la propagation des dommages. Cependant, il est RECOMMANDÉ que les couches inférieures déploient la détection d'erreur pour les en-têtes ROHC et ne livrent pas d'en-têtes ROHC avec des taux élevés d'erreurs résiduelles.

Sans fixer une limite ferme au taux d'erreurs résiduelles acceptable pour ROHC, il est noté que pour un taux d'erreurs binaires résiduelles d'au plus 1E-5, le schéma de ROHC a été conçu pour ne pas augmenter le nombre d'en-têtes endommagés, c'est-à-dire que le nombre d'en-têtes endommagés dû à la propagation d'un dommage est conçu pour être inférieur au nombre d'en-têtes endommagés capturés par le schéma de détection d'erreur de ROHC.

Négociation
En plus des mécanismes de traitement de paquet décrits ci-dessus, la couche liaison DOIT fournir un moyen de négocier les paramètres de compression d'en-tête ; voir aussi le paragraphe 5.1.1. (Pour les liaisons unidirectionnelles, cette négociation peut être effectuée hors bande ou même à priori.)

4.2 Dynamisme

Le protocole ROHC réalise son gain de compression par l'établissement d'informations d'état aux deux extrémités de la liaison, c'est-à-dire, au compresseur et au décompresseur. Les différentes parties de l'état sont établies à des moments différents et à des fréquences différentes ; donc, on peut dire que certaines des informations d'état sont plus dynamiques que le reste.

Certaines informations d'état sont établies au moment où un canal est établi ; ROHC suppose l'existence d'un protocole de négociation hors bande (tel que PPP) ou un état de canal prédéfini (très utile pour les liaisons unidirectionnelles). Dans les deux cas, on parle d'un "état de canal négocié". ROHC ne suppose pas que cet état puisse changer de façon dynamique pendant la durée de vie du canal (et il ne prend pas explicitement en charge de tels changements, bien que certains puissent être inoffensifs du point de vue d'un protocole). Un exemple d'état de canal négocié est le plus fort numéro d'identifiant de contexte à utiliser par le compresseur (MAX_CID).

D'autres informations d'état sont associées aux flux individuels de paquets dans le canal ; cet état est dit faire partie du contexte. En utilisant des identifiants de contexte (CID), plusieurs flux de paquets avec des contextes différents peuvent partager un canal. L'état de canal négocié indique le plus fort identifiant de contexte à utiliser, ainsi que le choix d'une des deux façons d'indiquer le CID dans l'en-tête compressé.

Il appartient au compresseur de décider quels paquets associer à un contexte (ou, de façon équivalente, quels paquets constituent un seul flux) ; cependant, ROHC n'est efficace que lorsque tous les paquets d'un flux partagent certaines propriétés, comme d'avoir les mêmes valeurs pour des champs qui sont décrits comme "statiques" dans le présent document (par exemple, les adresses IP, les numéros d'accès, et les paramètres RTP tels que le type de charge utile). L'efficacité de ROHC RTP dépend aussi de ce que le compresseur voit la plupart des numéros de séquence RTP.

Les flux n'ont pas besoin de partager toutes les caractéristiques importantes pour la compression. ROHC a une notion de profils de compression : un profil de compression note un ensemble prédéfini de ces caractéristiques. Pour assurer l'extensibilité, l'état de canal négocié inclut l'ensemble des profils acceptables pour le décompresseur. L'état de contexte inclut le profil actuellement utilisé pour le contexte.

D'autres éléments de l'état du contexte peuvent inclure les valeurs actuelles de tous les champs d'en-tête (à partir desquels on peut déduire si un en-tête IPv4 est présent dans la chaîne d'en-tête, et si les sommes de contrôle UDP sont activées) ainsi qu'un contexte de compression supplémentaire qui ne fait pas partie d'un en-tête non compressé, par exemple, le pas d'horodatage (TS_STRIDE), les caractéristiques de l'identifiant IP (IP-ID) (s'incrémente t-il comme une valeur de 16 bits dans l'ordre des octets du réseau ? sont accroissement est il aléatoire ?) un certain nombre d'en-têtes de vieilles références, et les automates à état de compresseur/décompresseur (voir le paragraphe suivant).

Le présent document définit en fait quatre profils ROHC : un profil non compressé, le profil principal de compression RTP ROHC, et deux variantes de ce profil pour la compression de paquets avec des chaînes d'en-têtes qui se terminent respectivement en UDP et ESP, mais où la compression RTP n'est pas applicable. Le texte descriptif du reste de la section se réfère au profil principal de compression RTP ROHC.

4.3 États de compression et de décompression

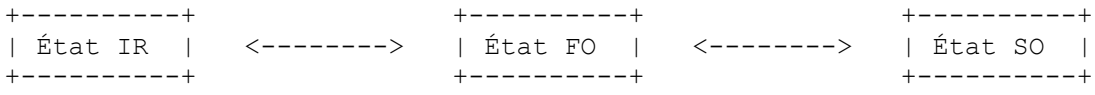
La compression d'en-tête avec ROHC peut être caractérisée comme une interaction entre deux automates à états, un au compresseur et un au décompresseur, chacun instancié une fois par contexte. Le compresseur et le décompresseur ont trois états chacun, qui sont de nombreuses façons en relation les uns avec les autres même si la signification des états est légèrement différente pour les deux parties. Les deux automates commencent dans le plus bas état de compression et transitent graduellement vers les états les plus élevés.

Les transitions n'ont pas besoin d'être synchronisées entre les deux automates. En fonctionnement normal, c'est seulement le compresseur qui revient temporairement aux états inférieurs. Le décompresseur ne va revenir en arrière que lorsque un dommage de contexte est détecté.

Les paragraphes suivants présentent une vue générale des automates à états et de leurs états correspondants, en commençant par le compresseur.

4.3.1 États de compresseur

Pour la compression ROHC, les trois états de compresseur sont Initialisation et rafraîchissement (IR), Premier ordre (FO, *First Order*) et Second ordre (SO). Le compresseur commence dans le plus bas état de compression (IR) et transite graduellement aux états de compression plus élevés. Le compresseur va toujours fonctionner dans l'état de compression le plus élevé possible, sous réserve que le compresseur soit suffisamment confiant que le décompresseur ait les informations nécessaires pour décompresser un en-tête compressé conformément à cet état.



Les décisions sur les transitions entre les divers états de compression sont prises par le compresseur sur la base :
- des variations dans les en-têtes de paquet
- de retours positifs provenant du décompresseur (Accusés de réception -- ACK)
- de retours négatifs provenant du décompresseur (Accusés de réception négatifs -- NACK)
- de fins de temporisation périodiques (en fonctionnement en mode unidirectionnel, c'est-à-dire, sur des canaux simplex ou lorsque le retour n'est pas activé)

La section 5 explique en détails comment sont effectuées les transitions pour chaque mode de fonctionnement.

4.3.1.1 État Initialisation et rafraîchissement (IR)

L'objet de l'état IR est d'initialiser les parties statiques du contexte chez le décompresseur ou de récupérer après défaillance. Dans cet état, le compresseur envoie des information d'en-tête complètes. Cela inclut tous les champs statiques et non statiques en forme non compressée plus quelques informations supplémentaires.

Le compresseur reste dans l'état IR jusqu'à ce qu'il soit bien sûr que le décompresseur a reçu correctement les informations statiques.

4.3.1.2 État Premier ordre (FO)

L'objet de l'état FO est de communiquer efficacement les irrégularités du flux de paquets. Lorsque il fonctionne dans cet état, le compresseur envoie rarement des informations sur ses champs dynamiques, et les informations envoyées sont usuellement compressées au moins partiellement. Seuls quelques champs statiques peuvent être mis à jour. La différence entre IR et FO devrait donc être claire.

Le compresseur entre dans cet état à partir de l'état IR, et à partir de l'état SO chaque fois que les en-têtes du flux de paquets ne se conforment pas au schéma qu'ils suivaient précédemment. Il reste dans l'état FO jusqu'à ce qu'il soit sûr que le décompresseur a acquis tous les paramètres du nouveau schéma. Des changements dans les champs qui sont toujours irréguliers sont communiqués dans tous les paquets et font donc partie de ce qui est un schéma uniforme.

Certains des paquets envoyés dans l'état FO, ou tous, portent des informations de mise à jour du contexte. Il est très important de détecter la corruption de tels paquets pour éviter des erreurs de mise à jour et des incohérences de contexte.

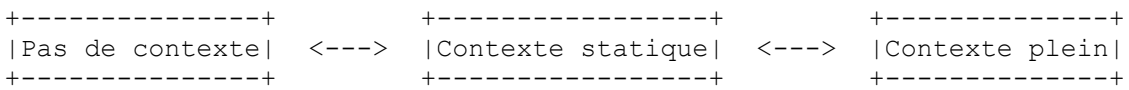
4.3.1.3 État Second ordre (SO)

C'est l'état où la compression est optimale. Le compresseur entre dans l'état SO lorsque l'en-tête à compresser est complètement prévisible étant donné le SN (numéro de séquence RTP) et que le compresseur est suffisamment sûr que le décompresseur a acquis tous les paramètres des fonctions du SN aux autres champs. La décompression correcte des paquets envoyés dans l'état SO ne dépend que de la décompression correcte du SN. Cependant, la réussite de la décompression exige aussi que les informations envoyées dans les paquets précédents dans l'état FO aient été bien reçues par le décompresseur.

Le compresseur quitte cet état et revient à l'état FO lorsque l'en-tête ne se conforme plus au schéma uniforme et qu'il ne peut plus être compressé indépendamment sur la base des informations de contexte précédentes.

4.3.2 États du décompresseur

Le décompresseur commence dans son plus bas état de compression, "Pas de contexte" et transite graduellement vers les états supérieurs. L'automate à états du décompresseur ne quitte normalement jamais l'état "Contexte plein" une fois qu'il est entré dans cet état.



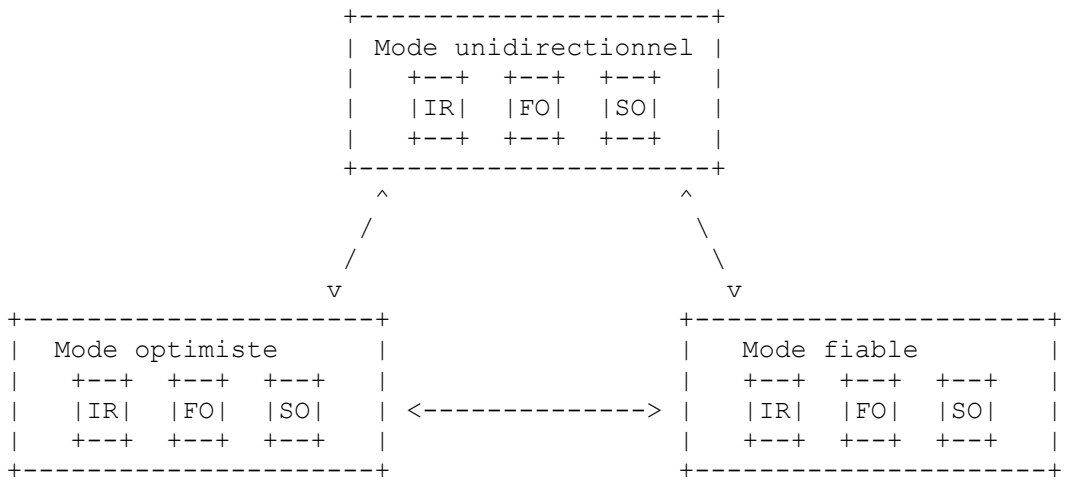
Au départ, lorsque il travaille dans l'état "Pas de contexte", le décompresseur n'a pas encore réussi à décompresser un paquet. Une fois qu'un paquet a été décompressé correctement (par exemple, à réception d'un paquet d'initialisation avec les informations statiques et dynamiques) le décompresseur peut passer directement à l'état "Contexte plein", et c'est seulement en cas de défaillances répétées qu'il va revenir à des états inférieurs. Cependant, lorsque cela arrive, il revient d'abord à l'état "Contexte statique". Là, la réception de tout paquet envoyé dans l'état FO est normalement suffisante pour activer à nouveau la transition à l'état "Contexte plein". C'est seulement lorsque la décompression de plusieurs paquets envoyés dans l'état FO échoue dans l'état "Contexte statique" que le décompresseur revient directement à l'état "Pas de contexte".

Le moment où les transitions d'état sont effectuées est expliqué en détails à la section 5.

4.4 Modes de fonctionnement

Le schéma ROHC a trois modes de fonctionnement, appelés Unidirectionnel, Bidirectionnel optimiste, et Bidirectionnel fiable.

Il est important de comprendre les différences entre les états et les modes, tels que décrits dans la section précédente. Ces abstractions sont orthogonales les unes aux autres. L'abstraction d'état est la même pour tous les modes de fonctionnement, alors que le mode contrôle la logique des transitions d'état et des actions à effectuer dans chaque état.



Le mode optimal de fonctionnement dépend des caractéristiques de l'environnement du protocole de compression, comme les capacités de retour, les probabilités et distributions d'erreurs, les effets des variations de taille d'en-tête, etc. Toutes les mises en œuvre de ROHC DOIVENT appliquer et prendre en charge les trois modes de fonctionnement. Ces trois modes

sont brièvement décrits dans les paragraphes qui suivent.

Les descriptions détaillées des trois modes de fonctionnement concernant la logique de compression et de décompression sont données à la section 5. Les mécanismes de transition de mode sont aussi décrits à la section 5.

4.4.1 Mode unidirectionnel – mode U

Quand ils sont dans le mode de fonctionnement unidirectionnel, les paquets sont envoyés dans une seule direction : du compresseur au décompresseur. Ce mode rend donc ROHC utilisable sur les liaisons où un chemin de retour du décompresseur au compresseur est indisponible ou indésirable.

En mode U, les transitions entre les états du compresseur ne sont effectuées qu'en fonction de temporisations périodiques et d'irrégularités dans les schémas de changement de champ d'en-tête dans le flux de paquets compressés. Du fait des rafraîchissements périodiques et du manque de retour pour l'initiation de récupération d'erreur, la compression dans le mode unidirectionnel sera moins efficace et aura une probabilité légèrement plus forte de propagation des pertes comparée à tous les modes bidirectionnels.

La compression avec ROHC DOIT débuter dans le mode unidirectionnel. La transition vers tout mode bidirectionnel peut être effectuée aussitôt qu'un paquet a atteint le décompresseur et qu'il a répondu par un paquet de retour indiquant qu'une transition de mode est désirée (voir la section 5).

4.4.2 Mode bidirectionnel optimiste – mode O

Le mode bidirectionnel optimiste est similaire au mode unidirectionnel. La différence est qu'un canal de retour est utilisé pour envoyer des demandes de récupération d'erreur et (facultativement) des accusés de réception de mises à jour significatifs du contexte du décompresseur au compresseur (non, cependant, pour les simples mises à jour de numéro de séquence). Les rafraîchissements périodiques ne sont pas utilisés dans le mode bidirectionnel optimiste.

Le mode O vise à maximiser l'efficacité de compression et un usage modéré du canal de retour. Il réduit le nombre d'en-têtes endommagés livrés aux couches supérieures du fait d'erreurs résiduelles ou d'invalidation du contexte. La fréquence des invalidations de contexte peut être supérieure à celle du mode R, en particulier lorsque surviennent de longues salves de pertes/erreurs. Se référer au paragraphe 4.7 pour les détails.

4.4.3 Mode bidirectionnel fiable – mode R

Le mode bidirectionnel fiable diffère sur de nombreux points des deux précédents. Les différences les plus importantes sont un usage plus intensif du canal de retour et une logique plus stricte aussi bien au compresseur qu'au décompresseur qui prévient les pertes de synchronisation de contexte entre compresseur et décompresseur sauf pour les taux d'erreurs binaires résiduelles très élevés. Un retour est envoyé pour accuser réception de toute mise à jour de contexte, y compris les mises à jour du champ Numéro de séquence. Cependant, tous les paquets ne mettent pas à jour le contexte en mode fiable.

Le mode R vise à maximiser la robustesse contre la propagation de perte et la propagation de dommage, c'est-à-dire, à minimiser la probabilité d'une invalidation de contexte, même dans des conditions de salves de perte/erreur d'en-tête. Il peut avoir une moindre probabilité d'invalidation de contexte que le mode O, mais un plus grand nombre d'en-têtes endommagés peut être livré lorsque le contexte est en fait invalidé. Se référer au paragraphe 4.7 pour les détails.

4.5 Méthodes de codage

Ce paragraphe décrit les méthodes de codage utilisées pour les champs d'en-tête. La façon dont les méthodes sont appliquées à chaque champ (par exemple, les valeurs des paramètres associés) est spécifiée au paragraphe 5.7.

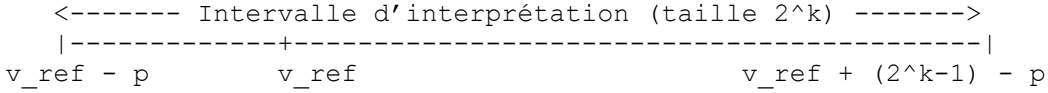
4.5.1 Codage des bits de moindre poids

Le codage des bits de moindre poids (LSB, *Least Significant Bit*) est utilisé pour les champs d'en-tête dont les valeurs sont habituellement sujettes à de petits changements. Avec le codage de LSB, les k bits de moindre poids de la valeur du champ sont transmis plutôt que la valeur originale du champ, où k est un entier positif. Après avoir reçu k bits, le décompresseur déduit la valeur originale en utilisant une valeur précédemment reçue comme référence (v_ref).

La correction du schéma est garantie si le compresseur et le décompresseur utilisent chacun les intervalles d'interprétation
1) dans lesquels réside la valeur originale, et

2) dans lesquels la valeur originale est la seule valeur qui a les exacts mêmes k bits de moindre poids que transmis.

L'intervalle d'interprétation peut se décrire comme une fonction f(v_ref, k).
Soit f(v_ref, k) = [v_ref - p, v_ref + (2^k - 1) - p]
où p est un entier.



La fonction f a les propriétés suivantes : pour toute valeur k, les k bits de moindre poids vont identifier de façon univoque une valeur dans f(v_ref, k).

Le paramètre p est introduit de sorte que l'intervalle d'interprétation puisse être déplacé par rapport à v_ref. Choisir une bonne valeur pour p donnera un codage plus efficace pour des champs qui ont certaines caractéristiques. Voici des exemples :

a) Pour les valeurs de champ dont on s'attend à ce qu'elles croissent toujours, p peut être réglé à -1. L'intervalle d'interprétation devient [v_ref + 1, v_ref + 2^k].

b) Pour les valeurs de champ qui restent les mêmes ou augmentent, p peut être réglé à 0. L'intervalle d'interprétation devient [v_ref, v_ref + 2^k - 1].

c) Pour les valeurs de champ dont on s'attend qu'elle ne dévient que légèrement d'une valeur constante, p peut être réglé à 2^(k-1) - 1. L'intervalle d'interprétation devient [v_ref - 2^(k-1) + 1, v_ref + 2^(k-1)].

d) Pour les valeurs de champ dont on s'attend qu'elles subissent de petits changements négatifs et de plus grands changements positifs, comme le TS RTP pour la vidéo, ou le SN RTP lorsque il y a un défaut de rangement, p peut être réglé à 2^(k-2) - 1. L'intervalle devient [v_ref - 2^(k-2) + 1, v_ref + 3 * 2^(k-2)], c'est-à-dire que les 3/4 de l'intervalle sont utilisés pour les changements positifs.

On donne ci-après une procédure simplifiée pour la compression et la décompression de LSB ; elle est modifiée pour les besoins de la robustesse et la protection contre la propagation des dommages dans le paragraphe suivant :

1) Le compresseur (décompresseur) utilise toujours v_ref_c (v_ref_d), la dernière valeur qui a été compressée (décompressée), comme v_ref ;

2) Lors de la compression d'une valeur v, le compresseur trouve la valeur minimum de k telle que v tombe dans l'intervalle f(v_ref_c, k). Appelons cette fonction k = g(v_ref_c, v). Lorsque seules quelques valeurs distinctes de k sont possibles, par exemple du fait de limitations imposées par les formats de paquet (voir le paragraphe 5.7) le compresseur va plutôt prendre le plus petit k qui mette v dans l'intervalle f(v_ref_c, k).

3) Lorsque il reçoit m LSB, le décompresseur utilise l'intervalle d'interprétation f(v_ref_d, m), appelé interval_d. Il prend comme valeur décompressée celle de l'interval_d dont les LSB correspondent aux m bits reçus.

Noter que les valeurs à coder ont une gamme finie ; par exemple, le SN RTP va de 0 à 0xFFFF. Lorsque la valeur de SN est proche de 0 ou de 0xFFFF, l'intervalle d'interprétation peut enjamber la limite de retour à zéro entre 0 et 0xFFFF.

Le schéma est compliqué par deux facteurs : la perte de paquets entre le compresseur et le décompresseur, et les erreurs de transmission non détectées par la couche inférieure. Dans le premier cas, le compresseur et le décompresseur vont perdre la synchronisation de v_ref, et donc aussi de l'intervalle d'interprétation. Si v est encore couvert par l'intersection (interval_c, interval_d) la décompression sera correcte. Autrement, une décompression incorrecte va en résulter. Le paragraphe suivant va traiter ce point plus en détails.

Dans le cas d'erreurs de transmission non détectées, les LSB corrompus vont donner une valeur décompressée incorrectement qui va ensuite être utilisée comme v_ref_d, qui à son tour va vraisemblablement conduire à la propagation des dommages. Ce problème est traité en utilisant une référence sûre, c'est-à-dire, une valeur de référence dont la correction est vérifiée par un CRC protecteur. Par conséquent, la procédure 1) ci-dessus est modifiée comme suit :
1) a) le compresseur utilise toujours pour v_ref_c la dernière valeur compressée et envoyée avec un CRC protecteur.
b) le décompresseur utilise toujours pour v_ref_d la dernière valeur correcte, vérifiée par un CRC réussi.

Noter que dans les modes U/O, 1) b) est modifié de telle sorte que si la décompression du SN échoue en utilisant la dernière référence de SN vérifiée, une autre tentative de décompression est faite en utilisant l'avant-dernière référence de SN

vérifiée. Cette procédure atténue la propagation de dommage lorsque un petit CRC échoue à détecter une valeur endommagée. Voir les détails au paragraphe 5.3.2.2.3.

4.5.2 Codage de LSB fondé sur la fenêtre (codage W-LSB)

Ce paragraphe décrit comment modifier l'algorithme simplifié du paragraphe 4.5.1 pour réaliser la robustesse.

Le compresseur peut n'être pas capable de déterminer la valeur exacte de v_ref_d qui sera utilisée par le décompresseur pour une certaine valeur v, car certains candidats pour v_ref_d peuvent avoir été perdus ou endommagés. Cependant, en utilisant les retours ou en faisant des hypothèses raisonnables, le compresseur peut limiter l'ensemble de candidats. Le compresseur calcule alors k de telle façon qu'il importe peu de savoir quel candidat v_ref_d utilisera le décompresseur dans l'ensemble des candidats, v étant couvert par l'interval_d résultant.

Comme le décompresseur utilise toujours comme référence la dernière valeur reçue lorsque le CRC a réussi, le compresseur entretient une fenêtre glissante qui contient les candidats pour v_ref_d. La fenêtre glissante est vide au départ. Les opérations suivantes sont effectuées sur la fenêtre glissante par le compresseur :

1) Après l'envoi d'une valeur v (compressée ou non compressée) protégée par un CRC, le compresseur ajoute v à la fenêtre glissante.
2) Pour chaque valeur v compressée, le compresseur choisit k = max(g(v_min, v), g(v_max, v)), où v_min et v_max sont les valeurs minimum et maximum dans la fenêtre glissante, et g est la fonction définie au paragraphe précédent.
3) Lorsque le compresseur est suffisamment sûr qu'une certaine valeur v et toutes les valeurs plus vieilles que v ne seront pas utilisées comme référence par le décompresseur, la fenêtre est avancée en retirant ces valeurs (y compris v). L'assurance peut être obtenue par divers moyens. En mode R, un ACK de la part du décompresseur implique que les valeurs plus vieilles que celle dont il est accusé réception peuvent être retirées de la fenêtre glissante. En mode U/O, il y a toujours un CRC pour vérifier la décompression correcte, et une fenêtre glissante avec une largeur maximum limitée est utilisée. La largeur de fenêtre est un paramètre d'optimisation qui dépend de la mise en œuvre.

Noter que le décompresseur suit la procédure décrite au paragraphe précédent, sauf qu'en mode R, il DOIT accuser réception de chaque en-tête reçu avec un CRC réussi (voir aussi au paragraphe 5.5).

4.5.3 Codage d'horodatage RTP adapté

L'horodatage (TS) RTP ne va normalement pas augmenter d'un nombre arbitraire d'un paquet à l'autre. L'augmentation va plutôt normalement être d'un entier multiple d'une certaine unité (TS_STRIDE). Par exemple, dans le cas d'un flux audio, le taux d'échantillonnage est normalement de 8 kHz et une trame vocale peut couvrir 20 ms. De plus, chaque trame vocale est souvent portée dans un seul paquet RTP. Dans ce cas, l'incrément RTP est toujours n * 160 (= 8000 * 0,02) pour un entier n. Noter que les périodes de silence n'ont pas d'impact sur cela, car l'horloge d'échantillon à la source continue normalement de courir sans changer le taux de trame ou les limites de trame.

Dans le cas de la vidéo, il y a normalement aussi un TS_STRIDE lorsque le niveau de trame vidéo est considéré. Le taux d'échantillon pour la plupart des codecs vidéo est 90 kHz. Si le taux de trames vidéo est fixé, disons, à 30 trames/seconde, le TS va augmenter de n * 3000 (= n * 90000 / 30) entre les trames vidéo. Noter qu'une trame vidéo est souvent divisée en plusieurs paquets RTP pour augmenter la robustesse contre la perte de paquets. Dans ce cas, plusieurs paquets RTP vont porter le même TS.

Lorsque on utilise le codage d'horodatage RTP adapté, le TS est diminué d'un facteur de TS_STRIDE avant compression. Cela économise les bits plancher(log2(TS_STRIDE)) pour chaque TS compressé. TS et TS_SCALED satisfont l'égalité suivante : TS = TS_SCALED * TS_STRIDE + TS_OFFSET

TS_STRIDE est explicitement, et TS_OFFSET implicitement, communiqué au décompresseur. L'algorithme suivant est utilisé :

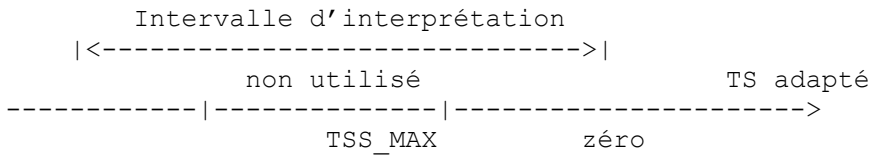
1. Initialisation : Le compresseur envoie au décompresseur la valeur de TS_STRIDE et la valeur absolue d'un ou plusieurs champs TS. Ces dernières sont utilisées par le décompresseur pour initialiser TS_OFFSET à (valeur absolue) modulo TS_STRIDE. Noter que TS_OFFSET est le même sans considération de la valeur absolue utilisée, tant que la valeur TS non adaptée ne revient pas à zéro ; voir le 4) ci-dessous.

2. Compression : Après l'initialisation, le compresseur ne compresse plus les valeurs de TS d'origine. Il compresse plutôt les valeurs diminuées : TS_SCALED = TS / TS_STRIDE. La méthode de compression pourrait être le codage W-LSB ou le codage fondé sur le temporisateur décrit au paragraphe suivant.

3. Décompression : Lorsque il reçoit la valeur compressée de TS_SCALED, le décompresseur déduit d'abord la valeur du TS_SCALED original. Le TS RTP original est alors calculé comme TS = TS_SCALED * TS_STRIDE + TS_OFFSET.

4. Décalage au retour à zéro : Le retour à zéro du TS de 32 bits non adapté va invalider la valeur courante de TS_OFFSET utilisée dans l'équation ci-dessus. Par exemple, supposons que TS_STRIDE = 160 = 0xA0 et que le TS courant = 0xFFFFFFF0. TS_OFFSET est alors de 0x50 = 80. Si le prochain TS RTP = 0x00000130 (c'est-à-dire, si l'incrément est 160 * 2 = 320) le nouveau TS_OFFSET devrait être 0x00000130 modulo 0xA0 = 0x90 = 144. Le compresseur n'est pas obligé de réinitialiser TS_OFFSET au moment du retour à zéro. Le décompresseur DOIT plutôt détecter le retour à zéro du TS non adapté (ce qui est trivial) et mettre à jour TS_OFFSET à TS_OFFSET = (TS non adapté revenu à zéro) modulo TS_STRIDE

5. Intervalle d'interprétation au moment du retour à zéro : Des règles spéciales sont nécessaires pour l'intervalle d'interprétation du TS adapté au moment du retour à zéro, car le TS adapté maximum, TSS_MAX, (0xFFFFFFFF / TS_STRIDE) peut ne pas avoir la forme 2^m - 1. Par exemple, lorsque TS_STRIDE est 160, le TS adapté est au plus de 26 843 545 qui a pour LSB 10011001. La limite de retour à zéro entre le TSS_MAX peut donc ne pas correspondre à une frontière naturelle entre les LSB.



Lorsque TSS_MAX fait partie de l'intervalle d'interprétation, un certain nombre de valeurs non utilisées y sont insérées après TSS_MAX afin que leurs LSB se suivent naturellement les uns les autres. Par exemple, pour TS_STRIDE = 160 et k = 4, les valeurs correspondant aux LSB de 1010 à 1111 sont insérées. Le nombre de valeurs insérées dépend de k et des LSB du TS maximum adapté. Le nombre de valeurs valides dans l'intervalle d'interprétation devrait être assez élevé pour maintenir la robustesse. Cela peut être assuré par la règle suivante :

> Soit a le nombre de LSB nécessaire si il n'y avait pas de retour à zéro, et soit b le nombre de LSB nécessaires pour ôter toute ambiguïté entre TSS_MAX et zéro lorsque les a LSB de TSS_MAX sont réglés à zéro. Le nombre de bits de LSB à envoyer lorsque TSS_MAX ou zéro fait partie de l'intervalle d'interprétation est b.

Cette méthode d'adaptation peut être appliquée à de nombreux codecs fondés sur la trame. Cependant, la valeur de TS_STRIDE pourrait changer durant une session, par exemple par suite de stratégies d'adaptation. Si cela arrive, le TS non adapté est compressé jusqu'à ce que la réinitialisation des nouveaux TS_STRIDE et TS_OFFSET soit achevée.

4.5.4 Compression d'horodatage RTP fondée sur la temporisation

L'horodatage RTP [RFC1889] est défini pour identifier le numéro du premier échantillon utilisé pour générer la charge utile. Lorsque 1) les paquets RTP portent des charges utiles correspondant à un intervalle d'échantillonnage fixe, 2) l'échantillonnage est fait à taux constant, et 3) les paquets sont générés en phase avec l'échantillonnage, alors la valeur de l'horodatage va s'approcher d'assez près d'une fonction linéaire du temps. C'est le cas pour les supports conversationnels, tels que la parole interactive. Le ratio linéaire est déterminé par le taux d'échantillon de source. Le schéma linéaire peut être compliqué par la mise en paquets (par exemple, dans le cas de vidéo où une trame vidéo correspond normalement à plusieurs paquets RTP) ou des réarrangements de trame (par exemple, des trames B sont envoyées déclassées par des codecs vidéo).

Avec un taux d'échantillonnage fixe de 8 kHz, 20 ms dans le domaine du temps est équivalent à un incrément de 160 dans le domaine TS non adapté, et à un incrément de 1 dans le domaine TS adapté avec TS_STRIDE = 160.

Par conséquent, le TS (adapté) des en-têtes qui arrivent au décompresseur sera une fonction linéaire de l'heure, avec des écarts dus à la gigue de retard (et à l'imprécision d'horloge) entre la source et le décompresseur. En fonctionnement normal, c'est-à-dire, pas de panne ou défaillance, la gigue de retard va être limitée pour satisfaire aux exigences du trafic de conversation en temps réel. Donc, en utilisant une horloge locale, le décompresseur peut obtenir une approximation du TS (adapté) dans l'en-tête à décompresser en considérant son heure d'arrivée. L'approximation peut alors être précisée avec les LSB du TS (adapté) porté dans l'en-tête. La valeur de k exigée pour s'assurer d'une décompression correcte est une fonction de la gigue entre la source et le décompresseur.

Si le compresseur connaît la gigue potentielle introduite entre le compresseur et le décompresseur, il peut déterminer k en utilisant une horloge locale pour estimer la gigue dans les heures d'arrivée de paquet, ou autrement il peut utiliser un k fixe et éliminer les paquets qui arrivent trop décalés.

Les avantages de ce schéma incluent :

a) que la taille du TS compressé soit constante et petite. En particulier, elle NE dépend PAS de la longueur des intervalles de silence. C'est à l'inverse des autres techniques de compression de TS, qui au début d'une salve de parole exigent l'envoi d'un nombre de bits qui dépend de la durée de l'intervalle de silence précédent.

b) qu'aucune synchronisation n'est requise entre l'horloge locale du compresseur et celle du décompresseur.

Noter que bien que ce schéma puisse être fait pour fonctionner en utilisant aussi bien des horodatages adaptés que non adaptés, en pratique il est toujours combiné avec le codage de TS adapté parce que moins exigeant sur la résolution d'horloge, par exemple, 20 ms au lieu de 1/8 ms. Donc, l'algorithme décrit ci-dessous suppose que le schéma de codage fondé sur l'horloge fonctionne sur le TS adapté. Le cas de TS non adapté serait similaire, avec des changements pour adapter les facteurs.

La tâche majeure du compresseur est de déterminer la valeur de k. Sa fenêtre glissante contient maintenant non seulement des valeurs de référence potentielles pour l'horodatage mais aussi leurs heures d'arrivée au compresseur.

1) Le compresseur entretient une fenêtre glissante {(T_j, a_j), pour chaque en-tête j qui peut être utilisée comme référence}, où T_j est le TS adapté pour l'en-tête j, et a_j est l'heure d'arrivée de l'en-tête j. La fenêtre glissante sert au même objet que la fenêtre glissante W-LSB du paragraphe 4.5.2.

2) Lorsque un nouvel en-tête n arrive avec T_n comme TS adapté, le compresseur note l'heure d'arrivée a_n. Il calcule ensuite Max_Jitter_BC = max {|(T_n - T_j) - ((a_n - a_j) / TIME_STRIDE)|, pour tous les en-têtes j dans la fenêtre glissante}, où TIME_STRIDE est l'équivalent d'intervalle de temps d'un TS_STRIDE, par exemple, 20 ms. Max_Jitter_BC est la gigue observée maximum avant le compresseur, en unités de TS_STRIDE, pour les en-têtes dans la fenêtre glissante.

3) k est calculé comme k = plafond(log2(2 * J + 1), où J = Max_Gigue_BC + Max_Gigue_CD + 2.

Max_Gigue_CD est la limite supérieure de la gigue attendue sur le canal de communication entre compresseur et décompresseur (CD-CC). Elle ne dépend que des caractéristiques de CD-CC.

La constante 2 tient compte de l'erreur de quantification introduite par les horloges au compresseur et décompresseur, qui peut être +/- 1.

Noter que le calcul de k suit l'algorithme de compression décrit au paragraphe 4.5.1, avec p = 2^(k-1) - 1.

4) La fenêtre glissante est soumise aux mêmes opérations de fenêtre qu'au paragraphe 4.5.2, 1) et 3), excepté que les valeurs ajoutées et retirées sont appariées avec leur heure d'arrivée.

Décompresseur :

1) Le décompresseur utilise comme en-tête de référence le dernier en-tête correctement décompressé (tel que vérifié par le CRC). Il maintient la paire (T_ref, a_ref), où T_ref est le TS adapté de l'en-tête de référence, et a_ref est l'heure d'arrivée de l'en-tête de référence.
2) Lorsque il reçoit un en-tête compressé n à l'instant a_n, l'approximation du TS original adapté est calculé par :
   T_approx = T_ref + (a_n - a_ref) / TIME_STRIDE.
3) L'approximation est alors précisée par les k bits de moindre poids portés dans l'en-tête n, suivant l'algorithme de décompression du paragraphe 4.5.1, avec p = 2^(k-1) - 1.

Note : L'algorithme ne suppose aucun schéma particulier dans les paquets qui arrivent au compresseur, c'est-à-dire, il tolère les réarrangement avant le compresseur et le comportement d'horodatage RTP non croissant.

Note : L'arithmétique d'entiers est utilisée dans toutes les équations ci-dessus. Si TIME_STRIDE n'est pas égal à un nombre entier de tics d'horloge, l'heure doit être normalisée de telle sorte que TIME_STRIDE soit un nombre entier de tics d'horloge. Par exemple, si un tic d'horloge est de 20 ms et si TIME_STRIDE est de 30 ms, (a_n - a_ref) dans 2) peut être multiplié par 3 et TIME_STRIDE peut avoir la valeur 2.

Note : La résolution d'horloge du compresseur ou du décompresseur peut être moins bonne que TIME_STRIDE, auquel cas la différence, c'est-à-dire, la résolution réelle - TIME_STRIDE, est traitée comme gigue additionnelle dans le calcul de k.

Note : La résolution d'horloge du décompresseur peut être communiquée au compresseur en utilisant l'option de retour

CLOCK.

Note : Le décompresseur peut observer la gigue et en faire rapport au compresseur en utilisant l'option de retour JITTER. Le compresseur peut utiliser ces informations pour préciser son estimation de Max_Jitter_CD.

4.5.5 Codage du décalage d'identifiant IP

Comme tous les paquets IPv4 ont un identifiant IP pour permettre la fragmentation, ROHC fournit une compression transparente de cet identifiant. Il n'y a pas de prise en charge explicite dans ROHC de l'en-tête de fragmentation IPv6, de sorte qu'il n'est jamais besoin de discuter des identifiants IP hors du contexte de IPv4.

Ce paragraphe suppose (initialement) que la pile IPv4 chez l'hôte de source alloue un identifiant IP conformément à la valeur d'un compteur de deux octets qui est augmenté de un après chaque allocation à un paquet sortant. Donc, le champ Identifiant IP d'un paquet IPv4 particulier s'écoule avec un incrément de 1 de paquet à paquet sauf quand la source a émis des paquets intermédiaires qui n'appartiennent pas à ce flux.

Pour de telles piles IPv4, le SN RTP va augmenter de 1 pour chaque paquet émis et l'identifiant IP va augmenter d'au moins autant. Donc, il est plus efficace de compresser le décalage, c'est-à-dire, (IP-ID - RTP SN) au lieu de l'identifiant IP lui-même.

Le reste du paragraphe 4.5.5 décrit comment compresser/décompresser la séquence de décalages en utilisant le codage/décodage W-LSB, avec p = 0 (voir au paragraphe 4.5.1). Toute l'arithmétique d'identifiant IP est faite en utilisant des quantités de 16 bits non signées, c'est-à-dire, modulo 2^16.

Compresseur :
Le compresseur utilise le codage W-LSB (paragraphe 4.5.2) pour compresser une séquence de décalages

Offset_i = ID_i - SN_i,

où ID_i et SN_i sont les valeurs de l'identifiant IP et du SN RTP de l'en-tête i. La fenêtre glissante contient de tels décalages et non les valeurs des champs d'en-tête, mais les règles pour ajouter et supprimer les décalages de la fenêtre suivent par ailleurs le paragraphe 4.5.2.

Décompresseur :
L'en-tête de référence est le dernier en-tête décompressé correctement (comme vérifié par le CRC).

Lorsque il reçoit un paquet compressé m, le décompresseur calcule Offset_ref = ID_ref - SN_ref, où ID_ref et SN_ref sont respectivement les valeurs d'identifiant IP et de SN RTP dans l'en-tête de référence.

Puis le décodage W-LSB est utilisé pour décompresser Offset_m, en utilisant les LSB reçus dans le paquet m et Offset_ref. Noter que m peut contenir zéro LSB pour Offset_m, auquel cas Offset_m = Offset_ref.

Finalement, l'identifiant IP pour le paquet m est régénéré par IP-ID pour m = SN décompressé du paquet m + Offset_m

Ordre des octets du réseau :
Certaines piles IPv4 n'utilisent pas un compteur pour générer les valeurs d'identifiant IP comme décrit, mais ne transmettent pas le contenu de ce compteur dans l'ordre des octets du réseau et envoient plutôt les deux octets inversés. Dans ce cas, le compresseur peut compresser le champ Identifiant IP après avoir interverti les octets. Par conséquent, le décompresseur intervertit aussi les octets de l'identifiant IP après décompression pour régénérer l'identifiant IP original. Cela exige que le compresseur et le décompresseur se synchronisent sur l'ordre des octets du champ Identifiant IP en utilisant le fanion NBO ou NBO2 (voir le paragraphe 5.7).

Identifiant IP aléatoire :
Certaines piles IPv4 génèrent les valeurs d'identifiant IP en utilisant un générateur de nombres pseudo-aléatoires. Bien que cela puisse apporter certains avantages pour la sécurité, cela rend sans objet de tenter de compresser le champ. Donc, le compresseur devrait détecter un tel comportement aléatoire du champ. Après la détection et la synchronisation avec le décompresseur en utilisant le fanion RND ou RND2, le champ est envoyé tel quel en totalité comme octets supplémentaires après l'en-tête compressé.

4.5.6 Valeurs de longueur variable auto décrites

Les valeurs de TS_STRIDE et quelques autres paramètres de compression peuvent varier largement. TS_STRIDE peut faire 160 pour la voix et 90 000 pour 1 f/s vidéo. Pour optimiser le transfert de telles valeurs, un nombre variable d'octets est utilisé pour les coder. Le nombre d'octets utilisé est déterminé par les premiers bits du premier octet :

Le premier bit est 0 : 1 octet.
7 bits transférés.
Jusqu'à 127 en décimal.
Octets codés en hexadécimal : 00 à 7F

Les premiers bits sont 10 : 2 octets.
14 bits transférés.
Jusqu'à 16 383 décimal.
Octets codés en hexadécimal : 80 00 à BF FF

Les premiers bits sont 110: 3 octets.
21 bits transférés.
Jusqu'à 2 097 151 en décimal.
Octets codés en hexadécimal : C0 00 00 à DF FF FF

Les premiers bits sont 111 : 4 octets.
29 bits transférés.
Jusqu'à 536 870 911 en décimal.
Octets codés en hexadécimal : E0 00 00 00 à FF FF FF FF

4.5.7 Valeurs codées sur plusieurs champs dans les en-têtes compressés

Lorsque un en-tête compressé a une extension, des parties d'une valeur codée peuvent être présentes dans plus d'un champ. Lorsque une valeur codée est étalée sur plusieurs champs de cette manière, les bits de poids fort de la valeur sont plus proches du début de l'en-tête. Si le nombre de bits disponibles dans les champs d'en-tête compressé excède le nombre de bits de la valeur, le champs de plus fort poids est bourré avec des zéros dans ses bits de poids fort

Par exemple, une valeur de TS non adaptée peut être transférée en utilisant un en-tête UOR-2 (voir au paragraphe 5.7) avec une extension de type 3. Le bit Tsc de l'extension est alors non établi (zéro) et le champ TS de longueur variable de l'extension est 4 octets, avec 29 bits disponible pour le TS (voir au paragraphe 4.5.6). Le champ TS UOR-2 va contenir les trois bits de plus fort poids du TS non adapté, et le champ TS de 4 octets dans l'extension va contenir les 29 bits restants.

4.6 Erreurs causées par des erreurs résiduelles

ROHC est conçu avec l'hypothèse que les paquets peuvent être endommagés entre le compresseur et le décompresseur, et que de tels paquets endommagés peuvent être livrés au décompresseur ("erreurs résiduelles").

Les erreurs résiduelles peuvent endommager le SN dans les en-têtes compressés. De tels dommages vont causer la génération d'un en-tête que les couches supérieures peuvent n'être pas capables de distinguer d'un en-tête correct. Lorsque l'en-tête compressé contient un CRC, le CRC va détecter le mauvais en-tête avec une probabilité qui dépend de la taille du CRC. Lorsque ROHC ne détecte pas le mauvais en-tête, celui-ci sera livré aux couches supérieures.

Le dommage n'est pas confiné au SN :

a) Le dommage aux bits d'indication de type de paquet peut être cause qu'un en-tête est interprété comme ayant un type de paquet différent.
b) Le dommage aux informations de CID peut être cause qu'un paquet est interprété selon un autre contexte et éventuellement aussi selon un autre profil. Des dommages aux CID seront plus dommageables lorsque une grande partie de l'espace de CID est utilisé, de sorte qu'il soit probable que le CID endommagé corresponde à un contexte actif.
c) Les informations de retour peuvent aussi être l'objet d'erreurs résiduelles, aussi bien lorsque le retour est porté en annexe que lorsque il est envoyé dans des paquets ROHC séparés. ROHC utilise des vérifications de bonne santé et ajoute des CRC aux informations de retour vitales pour permettre la détection de certains retours endommagés.

Noter qu'un dommage de contexte peut aussi résulter en la génération d'en-têtes incorrects ; le paragraphe 4.7 développe ce point.

4.7 Considérations sur la dégradation

Les dégradations aux en-têtes peuvent être classées selon les types suivants :
(1) la couche inférieure n'a pas été capable de décoder le paquet et ne l'a pas livré à ROHC,
(2) la couche inférieure a été capable de décoder le paquet, mais l'a éliminé à cause de la détection d'une erreur,
(3) ROHC a détecté une erreur dans l'en-tête généré et a éliminé le paquet, ou
(4) ROHC n'a pas détecté que l'en-tête régénéré soit endommagé et l'a livré aux couches supérieures.

Les dégradations causent la perte ou des dommages aux en-têtes individuels. Certains scénarios de dégradation causent aussi une invalidation du contexte, qui à son tour résulte en la propagation des pertes et des dommages. La propagation des dommages et les erreurs résiduelles indétectées contribuent toutes deux au nombre d'en-têtes endommagés livrés aux couches supérieures. La propagation des pertes et des dégradations résultant en la perte ou l'élimination de paquets isolés contribuent toutes deux à la perte de paquets vue par les couches supérieures.

Exemples de scénarios d'invalidation de contexte :

(a) Dégradation de type (4) sur le canal de transmission causant la mise à jour du contexte avec des informations incorrectes par le décompresseur.

(b)Salve de pertes/erreurs d'en-têtes de mise à jour de schéma : des dégradations de types (1) (2) et (3) sur des en-têtes de mise à jour de schéma consécutifs ; un en-tête de mise à jour de schéma est un en-tête qui porte de nouvelles informations de schéma, par exemple, au début d'une nouvelle salve de paroles ; cela cause la perte des informations de mise à jour de schéma par le décompresseur.

(c) Salve de perte/erreur d'en-têtes : les dégradations de type (1) (2) et (3) sur un certain nombre d'en-têtes consécutifs assez grand pour causer la perte de la synchronisation du SN par le décompresseur.

(d) Dégradation de type (4) sur le canal de retour qui simule un ACK valide et amène le compresseur à mettre à jour son contexte.

(e) Salve d'en-têtes considérés comme endommagés de type (3) par erreur qui déclanche la règle "k sur n" pour détecter une invalidation de contexte qui résulte en une séquence de NACK/mises à jour durant laquelle les en-têtes sont éliminés.

Le scénario (a) est atténué par le CRC porté dans tous les en-têtes de mise à jour de contexte. Les chances d'invalidation de contexte causée par (a) croissent avec la taille du CRC, Le CRC des en-têtes de mise à jour de contexte est toujours de 7 bits ou plus. En mode U/O, il est usuellement de 3 bits et parfois de 7 ou 8 bits.

Le scénario (b) est presque complètement éliminé lorsque le compresseur s'assure par des ACK qu'aucun en-tête de mise à jour de contexte n'est perdu, comme en mode R.

Le scénario (c) est presque complètement éliminé lorsque le compresseur s'assure par des ACK que le décompresseur va toujours détecter le retour à zéro des numéros de séquence, comme en mode R. Il est aussi atténué par les mécanismes de réparation de SN en mode U/O.

Le scénario (d) ne survient que lorsque le compresseur reçoit un en-tête endommagé qui simule un ACK de quelque en-tête présent dans la fenêtre W-LSB, disons un ACK d'en-tête 2, alors qu'en réalité l'en-tête 2 n'a jamais été reçu ou accepté par le décompresseur, c'est-à-dire que l'en-tête 2 a été l'objet d'une dégradation de type (1), (2) ou (3). L'en-tête endommagé doit simuler le type de paquet de retour, le type d'accusé de réception de retour, et les LSB de SN de quelque en-tête dans la fenêtre W-LSB.

Le scénario (e) se produit lorsque une salve d'erreurs résiduelles cause l'échec de la vérification du CRC dans k des derniers en-têtes portant des CRC. De grands k et n réduisent la probabilité du scénario (e), mais augmentent aussi le nombre d'en-têtes perdus ou endommagés par suite de toute invalidation de contexte.

ROHC détecte les en-têtes endommagés en utilisant les CRC sur les en-têtes d'origine. Les plus petits en-têtes dans le présent document incluent soit un CRC de 3 bits (en mode U/O) ou n'incluent pas de CRC (mode R). Pour les plus petits en-têtes, un dommage est donc détecté avec une probabilité d'environ 7/8 pour le mode U/O. Pour le mode R, le dommage au plus petits en-têtes n'est pas détecté.

Toutes choses égales par ailleurs (schéma de codage aux couches inférieures, etc.) le taux d'en-têtes endommagés par les erreurs résiduelles sera inférieur lorsque les en-têtes sont compressés par rapport à quand ils ne le sont pas, car moins de bits sont transmis. Par conséquent, pour un certain établissement de CRC ROHC le taux d'en-têtes incorrects livré aux applications sera aussi réduit.

L'analyse ci-dessus suggère que le mode U/O peut être plus enclin que le mode R à l'invalidation de contexte. D'un autre côté, le CRC présent dans tous les en-têtes de mode U/O masque continuellement les erreurs résiduelles venant des couches inférieures, réduit le nombre d'en-têtes endommagés livré aux couches supérieures lorsque le contexte est invalidé, et permet une détection rapide de l'invalidation de contexte.

Le mode R utilise toujours un CRC plus fort sur les en-têtes de mise à jour de contexte, mais aucun CRC dans les autres en-têtes. Une erreur résiduelle sur un en-tête qui ne porte pas de CRC va résulter en ce qu'un en-tête endommagé sera livré aux couches supérieures (4). Le nombre d'en-têtes endommagés livré aux couches supérieures dépend du ratio d'en-têtes avec CRC sur en-têtes sans CRC, qui est un paramètre du compresseur.

5. Le protocole

5.1 Structures de données

Le protocole ROHC se fonde sur un certain nombre de paramètres qui font partie de l'état de canal négocié et de l'état par contexte. Cette section décrit certaines de ces informations d'état d'une façon abstraite. Les mises en œuvre peuvent utiliser une structure et représentation différente de cet état. En particulier, les protocoles de négociation qui établissent l'état par canal ont besoin d'établir les informations qui constituent l'état de canal négocié, mais il n'est pas nécessaire de les échanger sous la forme décrite ici.

5.1.1 Paramètres par canal

MAX_CID : entier non négatif ; plus fort numéro d'identifiant de contexte à utiliser par le compresseur (noter que ce paramètre n'est pas couplé à LARGE_CIDS, mais il est effectivement contraint par lui).

LARGE_CIDS : Booléen ; si il est faux, la représentation courte de CID (0 octet ou 1 octet de préfixe, couvrant les CID de 0 à 15) est utilisée ; si il est vrai, la représentation de CID incorporé (1 ou 2 octets de CID incorporé couvrant les CID de 0 à 16383) est utilisée.

PROFILES : Ensemble d'entiers non négatifs, chaque entier indiquant un profil pris en charge par le décompresseur. Le compresseur NE DOIT PAS compresser en utilisant un profil qui n'est pas dans PROFILES.

FEEDBACK_FOR : Référence facultative pour un canal dans la direction inverse. Si il est fourni, ce paramètre indique à quel canal tout retour envoyé sur ce canal se réfère (voir le paragraphe 5.7.6.1).

MRRU : Unité maximum de réception reconstruite. C'est la taille en octets de la plus grande unité reconstruite que le décompresseur est supposé réassembler à partir des segments (voir le paragraphe 5.2.5). Noter que cette taille inclut le CRC. Si MRRU est négocié à 0, aucun segment d'en-tête n'est permis sur le canal.

5.1.2 Profils de paramètres par contexte

Les paramètres par contexte sont établis avec des en-têtes IP (voir au paragraphe 5.2.3). Un en-tête IR contient un identifiant de profil qui détermine comment le reste de l'en-tête sera interprété. Noter que le paramètre de profil détermine la syntaxe et la sémantique des identifiants de type de paquet et les types de paquet utilisés en conjonction avec un contexte spécifique. Le présent document décrit les profils 0x0000, 0x0001, 0x0002, et 0x0003 ; d'autres profils pourront être définis lorsque ROHC sera étendu à l'avenir.

Le profil 0x0000 est pour l'envoi de paquets IP non compressés. Voir le paragraphe 5.10.

Le profil 0x0001 est pour la compression RTP/UDP/IP, voir les paragraphes 5.3 à 5.9.

Le profil 0x0002 est pour la compression UDP/IP, c'est-à-dire que la compression des 12 premiers octets de la charge utile UDP n'est pas tentée. Voir le paragraphe 5.11.

Le profil 0x0003 est pour la compression ESP/IP, c'est-à-dire, la compression de la chaîne d'en-tête jusqu'à et y compris le

premier en-tête ESP, mais pas des sous en-têtes suivants. Voir le paragraphe 5.12.

Initialement, tous les contextes sont dans l'état Pas de contexte, c'est-à-dire, tous les paquets qui font référence à ce contexte excepté les paquets IR sont éliminés. Si elle est définie par un document "ROHC sur X", la négociation par canal peut être utilisée pour des informations d'état préétablies pour un contexte (par exemple, la négociation du profil 0x0000 pour le CID 15). De telles informations d'état peuvent aussi être marquées en lecture seule dans la négociation, ce qui causerait l'élimination par le décompresseur de tout paquet IR qui tenterait de la modifier.

5.1.3 Contextes et identifiants de contexte

Un contexte est associé à chaque flux compressé, et c'est l'état que le compresseur et le décompresseur maintiennent afin de correctement compresser ou décompresser les en-têtes du flux de paquets. Les contextes sont identifiés par un identifiant de contexte, un CID, qui est envoyé avec les en-têtes compressés et les informations de retour.

L'espace de CID est distinct pour chaque canal, c'est-à-dire que le CID 3 sur le canal A et le CID 3 sur le canal B ne se réfèrent pas au même contexte, même si les points d'extrémité de A et B sont sur le même nœud. En particulier, les CID pour toute paire de canaux de transmission et de canal inverse ne sont pas en relation (le canal de transmission et le canal inverse n'ont même pas besoin d'avoir des espaces de CID de la même taille).

Les informations de contexte sont conceptuellement conservées dans un tableau. Le tableau des contextes est indexé en utilisant le CID qui est envoyé avec des en-têtes compressés et des informations de retour. L'espace de CID peut être négocié comme étant petit, ce qui signifie que les CID peuvent prendre les valeurs de 0 à 15, ou grand, ce qui signifie que les CID prennent des valeurs entre 0 et 2^14 - 1 = 16383. La taille de l'espace de CID est négociée au moment de l'établissement d'un canal.

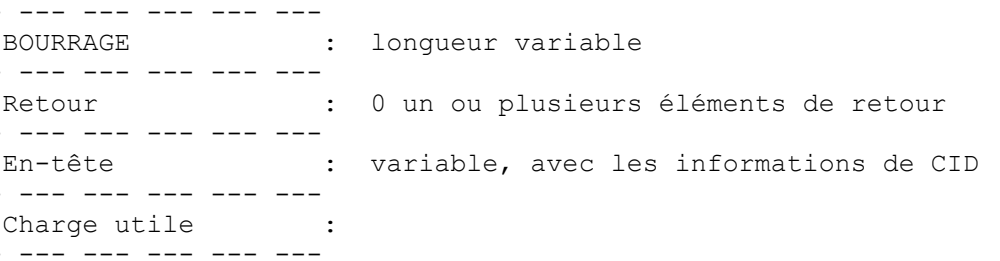
Un petit CID avec la valeur 0 est représenté en utilisant zéro bit. Un petit CID avec une valeur de 1 à 15 est représenté par un champ de quatre bits à la place d'un champ de type de paquet (Add-CID) plus quatre autres bits. Un grand CID est représenté en utilisant le schéma de codage du paragraphe 4.5.6, limité à deux octets.

5.2 Paquets et types de paquet ROHC

Le schéma d'indication de type de paquet pour ROHC a été conçu avec les contraintes suivantes :
a) il doit être possible d'utiliser seulement un nombre limité de tailles de paquet ;
b) il doit être possible d'envoyer des informations de retour dans des paquets ROHC séparés ainsi que portées sur des paquets transmis ;
c) il est souhaitable de permettre l'élimination du CID pour un flux de paquets lorsque quelques flux de paquets partagent un canal ;
d) il est prévu que certains paquets avec de gros en-têtes puissent être plus grands que la MTU de couches inférieures très contraintes.

Ces contraintes ont conduit à une conception qui inclut :
- un bourrage facultatif,
- un type de paquet de retour,
- un octet Add-CID facultatif qui fournit 4 bits de CID, et
- un mécanisme simple de segmentation et réassemblage.

Un paquet ROHC a le format général suivant (dans le diagramme, deux points ":" indiquent que la partie est facultative) :



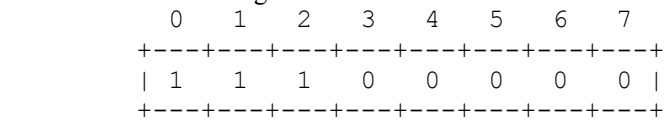
Bourrage est tout nombre (zéro, un ou plusieurs) d'octets de bourrage. Retour ou En-tête doit être présent.

Les éléments de retour commencent toujours par une indication de type de paquet. Les éléments de retour portent des informations internes de CID. Retour est décrit au paragraphe 5.2.2.

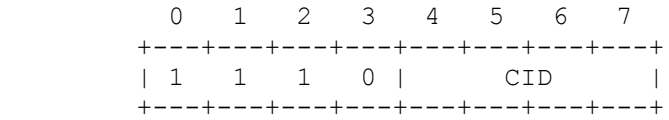
En-tête est soit un en-tête spécifique de profil, soit un en-tête IR ou IR-DYN (voir aux paragraphes 5.2.3 et 5.2.4). En-tête soit :
1) ne porte aucune information de CID (indiquant un CID de zéro), ou
2) inclut un octet Add-CID (voir ci-dessous), ou
3) contient des informations incorporées de CID longues d'un ou deux octets.

Les alternatives 1) et 2) ne s'appliquent qu'aux en-têtes compressés dans des canaux où l'espace de CID est petit. L'alternative 3) ne s'applique qu'aux en-têtes compressés dans des canaux où l'espace de CID est grand.

Octet de bourrage



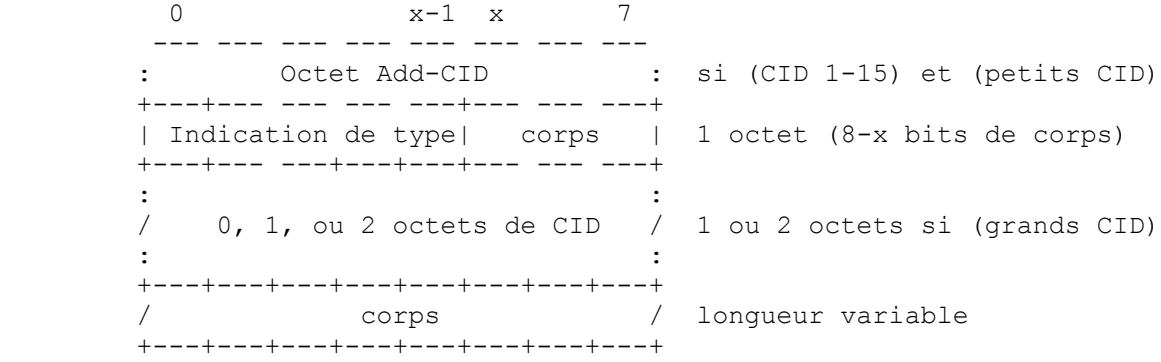
Octet Add-CID



CID : 0x1 à 0xF indique les CID de 1 à 15.

Note : L'octet Bourrage ressemble à un octet Add-CID pour le CID 0.

En-tête soit commence par une indication de type de paquet, soit a une indication de type de paquet qui suit immédiatement un octet Add-CID. Tous les types de paquet En-tête ont le format général suivant (dans le diagramme, les barres obliques "/" indiquent une longueur variable) :



Le grand CID, si il est présent, est codé conformément au paragraphe 4.5.6.

5.2.1 Retour d'informations ROHC

Retour porte des informations provenant du décompresseur pour le compresseur. Les principales sortes de retour sont prises en charge. En plus de la sorte de retour, d'autres informations peuvent être incluses dans des informations spécifiques du profil.

ACK : Accuse réception de la décompression réussie d'un paquet, ce qui signifie que le contexte est à jour avec une forte probabilité.

NACK : Indique que le contexte dynamique du décompresseur est hors synchronisation. Il est généré lorsque plusieurs paquets successifs ont échoué à être décompressés correctement.

STATIC-NACK : Indique que le contexte statique du décompresseur n'est pas valide ou n'a pas été établi.

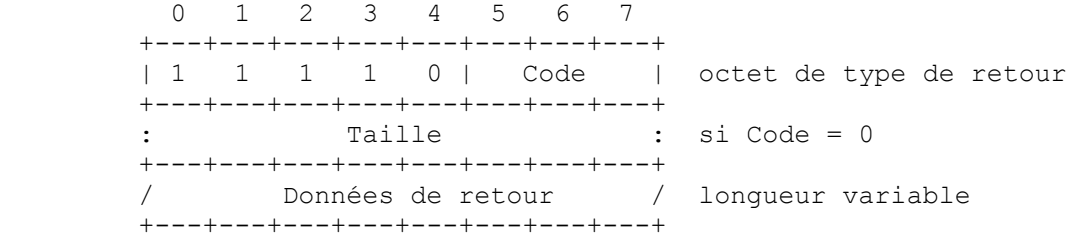
Il est prévu que le retour au compresseur puisse se faire de nombreuses façons, selon les propriétés de la couche inférieure particulière. Les détails exacts de la façon dont le retour est réalisé seront spécifiés dans un document "ROHC sur X", pour chaque couche inférieure X en question. Par exemple, le retour peut être réalisé en utilisant :
1) des mécanismes spécifiques de couche inférieure,
2) un canal dédié au seul retour, réalisé, par exemple par la couche inférieure qui fournit un moyen pour indiquer qu'un

paquet est un paquet de retour,
3) un canal dédié au seul retour, où le moment du retour donne des informations sur le paquet compressé qui a causé le retour,
4) l'entrelacement de paquets de retour parmi les paquets compressés normaux allant dans la même direction que le retour (les couches inférieures n'indiquent pas les retours),
5) le portage des informations de retour dans les paquets compressés qui vont dans la même direction que le retour (cette technique peut réduire la redondance par retour),
6) en l'entrelaçant et en le faisant porter sur le même canal, c'est-à-dire, à la fois 4) et 5).

Les alternatives 1 à 3 ne font peser aucune exigence particulière sur le schéma de type de paquet ROHC. Les alternatives 4 à 6 le font cependant. Le schéma de type de paquet ROHC a été conçu pour permettre les solutions 4 à 6 (qui peuvent être par exemple utilisées sur PPP) :
a) Le schéma ROHC fournit un type de paquet de retour. Le type de paquet est capable de porter des informations de retour de longueur variable.
b) Les informations de retour sur un certain canal sont passées, et interprétées par le compresseur associé au retour sur ce canal. Donc, les informations de retour doivent contenir les informations de CID si le compresseur associé peut utiliser plus d'un contexte. Le schéma de retour ROHC exige qu'un canal porte les retours à au plus un seul compresseur. Comment un compresseur est associé au retour sur un canal particulier devra être défini dans un document "ROHC sur X".
c) Le format des informations de retour ROHC est verrouillé sur l'octet, c'est-à-dire qu'il commence sur une limite d'octet, pour permettre d'utiliser le format sur un canal de retour dédié, 2).
d) Pour permettre le portage, 5), il est possible de déduire la longueur des informations de retour en examinant les quelques premiers octets du retour. Cela permet au décompresseur de passer les informations de retour portées au compresseur associé du même côté sans comprendre son format. Les informations de longueur découplent le décompresseur du compresseur dans ce sens que le décompresseur peut traiter l'en-tête compressé immédiatement sans attendre que le compresseur le renvoie après l'analyse des informations de retour.

5.2.2 Format de retour d'information ROHC

Le retour envoyé sur un canal ROHC consiste en un ou plusieurs éléments de retour enchaînés, où chaque élément de retour a le format suivant :



Code : 0 indique qu'un octet Taille est présent.
1-7 indique la taille du champ Données de retour en octets.

Taille : Octet facultatif qui indique la taille du champ Données de retour en octets.

Données de retour : Informations de retour spécifiques du profil. Inclut les informations de CID.

La taille totale du champ Données de retour est déterminable à réception par le décompresseur, en inspectant le champ Code et éventuellement le champ Taille. Ces informations de longueur explicites permettent le portage et aussi l'envoi de plus d'un élément Retour dans un paquet.

Lorsque le décompresseur a déterminé la taille du champ Données de retour, il retire l'octet Type de retour et le champ Taille (s'il est présent) et passe le reste au compresseur associé du même côté, avec une indication de la taille. Les données de retour reçues par le compresseur ont la structure suivante (le retour envoyé sur un canal dédié PEUT aussi utiliser ce format) :

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
:    Octet Add-CID              : pour les petits CID et (CID != 0)
+---+---+---+---+---+---+---+---+
:                               :
/  grand CID (codage de 4.5.6)  / 1-2 octets pour les grands CID
:                               :
+---+---+---+---+---+---+---+---+
/           Retour              /
+---+---+---+---+---+---+---+---+
```

Le grand CID, s'il est présent, est codé selon le paragraphe 4.5.6. Les informations de CID dans les données de retour indiquent le CID du flux de paquet pour lequel le retour est envoyé. Noter que le paramètre LARGE_CIDS qui contrôle si un grand CID est présent est tiré de l'état de canal du canal du compresseur receveur, et NON de celui du canal qui porte le retour.

Il est EXIGÉ que le champ Retour ait un des deux formats suivants :

FEEDBACK-1 :

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
|  Info. spécifiques du profil  |  1 octet
+---+---+---+---+---+---+---+---+
```

FEEDBACK-2 :

```
  0   1   2   3   4   5   6   7   +---+---+---+---+---+---+---+---+
|Acktype|                       |
+---+---+   Informations        /  au moins 2 octets
/     spécifiques du profil     |
+---+---+---+---+---+---+---+---+
```

Acktype : 0 = ACK
1 = NACK
2 = STATIC-NACK
3 est réservé (NE DOIT PAS être utilisé. Par ailleurs non analysable.)

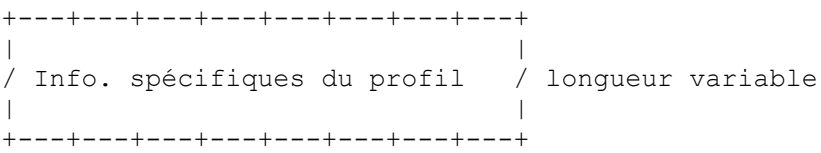
Le compresseur peut utiliser la logique suivante pour analyser le champ Retour :

1) Pour les grands CID, le retour va toujours commencer par un CID codé conformément au paragraphe 4.5.6. Si le premier bit est 0, le CID utilise un octet. Si le premier bit est 1, le CID utilise deux octets.
2) Pour les petits CID, si la taille est d'un octet, le retour est un FEEDBACK-1.
3) Pour les petits CID, si la taille est supérieurs à un octet, et si le retour commence par les deux bits 11, le retour commence par un octet Add-CID. Si la taille est 2, il est suivi par FEEDBACK-1. Si la taille est supérieure à 2, le Add-CID est suivi par FEEDBACK-2.
4) Autrement, il n'y a pas d'octet Add-CID, et le retour commence par un FEEDBACK-2.

5.2.3 Type de paquet IR ROHC

L'en-tête IR associe un CID à un profil, et initie aussi normalement le contexte. Il peut aussi normalement rafraîchir le (des parties du) contexte. Il a le format général suivant :

```
  0   1   2   3   4   5   6   7
 --- --- --- --- --- --- --- ---
:         Octet Add-CID         : pour petits CID et (CID != 0)
+---+---+---+---+---+---+---+---+
| 1   1   1   1   1   1   0 | x | octet de type IR
+---+---+---+---+---+---+---+---+
:                               :
/      0-2 octets de CID        / 1-2 octets si c'est un grand CID
:                               :
+---+---+---+---+---+---+---+---+
|            Profil             | 1 octet
+---+---+---+---+---+---+---+---+
|              CRC              | 1 octet
```



x : Informations spécifiques du profil. Interprété conformément au profil indiqué dans le champ Profil.

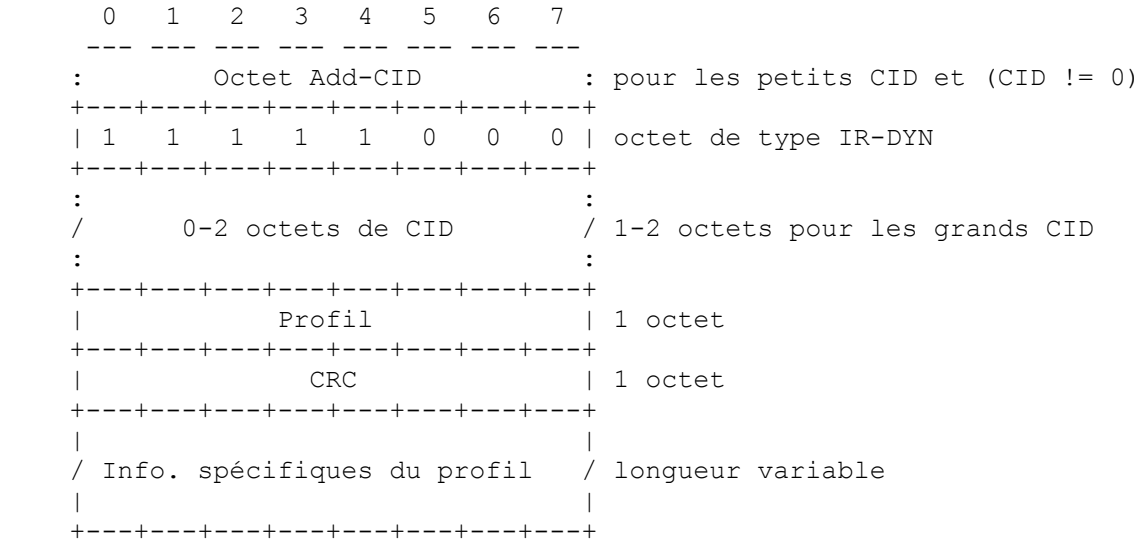
Profil : C'est le profil à associer au CID. Dans le paquet IR, l'identifiant de profil est abrégé aux 8 bits de moindre poids. Il choisit le profil de numéro le plus élevé dans les paramètres d'état de canal PROFILES qui correspondent aux 8 LSB donnés.

CRC : C'est un CRC de 8 bits calculé en utilisant le polynôme du paragraphe 5.9.1. Sa couverture dépend du profil, mais il DOIT couvrir au moins la partie initiale du paquet qui se termine par le champ Profil. Toute information qui initialise le contexte du décompresseur devrait être protégée par le CRC.

Informations spécifiques du profil : Le contenu de cette partie du paquet IR est défini par les profils individuels. Elles sont interprétées selon le profil indiqué dans le champ Profil.

5.2.4 Type de paquet ROHC IR-DYN

À la différence de l'en-tête IR, l'en-tête IR-DYN ne peut jamais initialiser un contexte non initialisé. Cependant, il peut redéfinir quel profil est associé à un contexte, voir par exemple aux paragraphes 5.11 (ROHC UDP) et 5.12 (ROHC ESP). Donc, le type doit être réservé au niveau du cadre. L'en-tête IR-DYN initialise ou rafraîchit normalement aussi des parties d'un contexte, normalement la partie dynamique. Il a le format général suivant :



Profil : C'est le profil à associer au CID. C'est abrégé de la même façon qu'avec les paquets IR.

CRC : C'est un CRC de 8 bits calculé en utilisant le polynôme du paragraphe 5.9.1. Sa couverture dépend du profil, mais il DOIT couvrir au moins la partie initiale du paquet qui se termine par le champ Profil. Toute information qui initialise le contexte du décompresseur devrait être protégée par le CRC.

Informations spécifiques du profil : Cette partie du paquet IR est définie par les profils individuels. Elles sont interprétées selon le profil indiqué dans le champ Profil.

5.2.5 Segmentation ROHC

Certaines couches de liaison peuvent fournir un service beaucoup plus efficace si l'ensemble des différentes tailles de paquet à transporter reste petit. Pour de telles couches liaison, ces tailles vont normalement être choisies pour transporter efficacement des paquets qui surviennent souvent, en s'adaptant éventuellement pour les paquets qui surviennent moins fréquemment sur la plus proche taille en ajoutant un bourrage. La couche liaison peut cependant être limitée à la taille des paquets qu'elle peut offrir dans ce mode efficace, ou il peut être souhaitable de ne demander qu'une plus grande taille limitée. Pour s'accommoder du paquet occasionnel qui est plus grand que cette plus grande taille négociée, ROHC définit

un protocole de segmentation simple.

5.2.5.1 Considérations sur l'utilisation de la segmentation

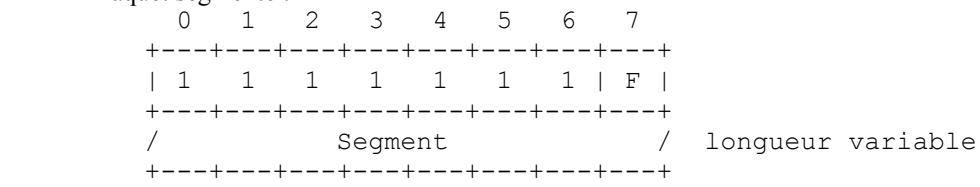
Le protocole de segmentation défini dans ROHC n'est pas particulièrement efficace. Il n'est pas destiné à remplacer les fonctions de segmentation de couche liaison ; celles-ci DEVRAIENT être utilisée chaque fois qu'elles sont disponibles et efficaces pour la tâche à entreprendre.

La segmentation ROHC ne devrait être utilisée que pour des paquets occasionnels avec des tailles supérieures à ce qu'il est efficace de s'accommoder, par exemple, dû à des en-têtes ROHC exceptionnellement grands. Le schéma de segmentation a été conçu pour réduire les variations de taille de paquet qui pourraient survenir à cause de valeurs excentriques dans la distribution des tailles d'en-tête. Dans d'autres cas, la segmentation devrait être faite aux couches inférieures. Le schéma de segmentation ne devrait être utilisé que pour les tailles de paquet qui sont supérieures à la taille maximum dans l'ensemble des tailles admises pour les couches inférieures.

En résumé, la segmentation ROHC devrait être utilisée de façon relativement peu fréquente dans le flux de paquets. Si cela ne peut pas être assuré, la segmentation devrait s'effectuer aux couches inférieures.

5.2.5.2 Protocole de segmentation

Paquet segmenté :

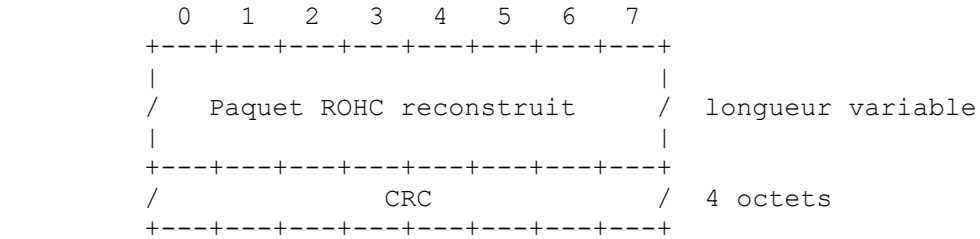


F : Bit final. S'il est établi , cela indique que c'est le dernier segment d'une unité reconstruite.

L'en-tête Segment peut être précédé d'octets de bourrage et/ou de retours. Il ne porte jamais un CID.

Tous les paquets d'en-tête Segment pour une unité reconstruite doivent être envoyés à la suite sur un canal, c'est-à-dire que tout paquet d'en-tête non Segment qui suit un en-tête Segment non final interrompt le réassemblage de l'unité actuellement en reconstruction et cause l'élimination par le décompresseur des segments non finaux reçus jusque là sur ce canal. Lorsque un en-tête segment final est reçu, le décompresseur réassemble le segment porté dans ce paquet et tout les segments non finaux qui le précédaient immédiatement en une seule unité reconstruite, dans l'ordre de leur réception. L'unité reconstruite a le format suivant :

Unité reconstruite :



Le CRC est utilisé par le décompresseur pour valider l'unité reconstruite. Il utilise l'algorithme FCS-32 avec le polynôme générateur suivant : x^0 + x^1 + x^2 + x^4 + x^5 + x^7 + x^8 + x^10 + x^11 + x^12 + x^16 + x^22 + x^23 + x^26 + x^32 [RFC1662]. Si l'unité reconstruite fait 4 octets ou moins, ou si le CRC échoue, ou si il est plus grand que le paramètre MRRU du canal (voir le paragraphe 5.1.1) l'unité reconstruite DOIT être éliminée par le décompresseur.

Si le CRC réussit, le paquet ROHC reconstruit est interprété comme un en-tête ROHC, facultativement suivi par une charge utile. Noter que cela signifie qu'il ne peut pas y avoir de bourrage et ni de retours dans l'unité reconstruite, et que le CID est déduit des octets initiaux de l'unité reconstruite.

(On devrait noter que le protocole de segmentation ROHC s'est inspiré du SEAL de Steve Deering et al., qui est ensuite devenu ATM AAL5. Les mêmes arguments s'appliquent aussi ici pour ne pas avoir de numéro de séquence dans les segments mais de fournir plutôt un CRC fort dans l'unité reconstruite. Noter que, comme résultat de ce protocole, il n'y a aucun moyen dans ROHC pour utiliser un segment qui a des erreurs binaires résiduelles.)

5.2.6 Traitement du décompresseur initial ROHC

Les types de paquet suivants sont réservés au niveau cadre dans le schéma ROHC :
1110 : Bourrage ou octet Add-CID
11110 : Retour
11111000 : Paquet IR-DYN
1111110 : Paquet IR
1111111 : Segment

D'autres types de paquet peuvent être utilisés à la discrétion des profils individuels.

Les étapes suivantes décrivent le traitement initial au décompresseur qui à réception d'un paquet ROHC peut déterminer son contenu.

1) Si le premier octet est un octet de bourrage (11100000), éliminer tous les octets de bourrage initial et passer à l'étape suivante.

2) Si le premier octet restant commence par 1110, c'est un octet Add-CID : mémoriser l'octet Add-CID ; retirer l'octet.

3) Si le premier octet restant commence par 11110, et si un octet Add-CID a été trouvé à l'étape 2), une erreur s'est produite ; l'en-tête DOIT être éliminé sans autre action.

4) Si le premier octet restant commence par 11110, et si un octet Add-CID n'a pas été trouvé à l'étape 2), c'est un retour : trouver la taille des données de retour, qu'on appelle s ; retirer l'octet de type Retour ; retirer l'octet Taille si Code est 0 ; envoyer les données de retour de longueur s au compresseur associé de même côté ; si le paquet est terminé, arrêter ; sinon, passer à 2).

5) Si le premier octet restant commence par 1111111, c'est un segment : tenter la reconstruction en utilisant le protocole de segmentation (5.2.5). Si un paquet reconstruit n'est pas produit, cela termine le traitement du paquet d'origine. Si un paquet reconstruit est produit, il est introduit à l'étape 1) ci-dessus. Bourrage, segments, et retours ne sont pas admis dans les paquets reconstruits, de sorte que lors de leur traitement, les étapes 1), 4), et 5) sont modifiées afin que le paquet soit éliminé sans autre action lorsque les conditions correspondent.

6) Maintenant, on sait que le reste sont des informations de transmission (sauf si l'en-tête est endommagé).

7) Si le trafic de transmission utilise des petits CID, il n'y a pas de grand CID dans le paquet. Si un Add-CID précédait immédiatement le type de paquet (étape 2), il a le CID du Add-CID ; autrement il a CID 0.

8) Si le trafic de transmission utilise des grands CID, le CID commence par le second octet restant. Si les premiers bits de cet octet ne sont pas 0 ou 10, le paquet DOIT être éliminé sans autre action. Si un octet Add-CID précédait immédiatement le type de paquet (étape 2) le paquet DOIT être éliminé sans autre action.

9) Utiliser le CID pour trouver le contexte.

10) Si le type de paquet est IR, le profil indiqué dans le paquet IR détermine comment il doit être traité. Si le CRC échoue à vérifier le paquet, il DOIT être éliminé. Si un profil est indiqué dans le contexte, la logique de ce profil détermine quel retour, s'il en est, doit être envoyé. Si aucun profil n'est noté dans le contexte, aucune autre action n'est entreprise.

11) Si le type de paquet est IR-DYN, le profil indiqué dans le paquet IR-DYN détermine comment il sera traité.
    a) Si le CRC échoue à vérifier le paquet, il DOIT être éliminé. Si un profil est indiqué dans le contexte, la logique de ce profil détermine quel retour, s'il en est, est à envoyer. Si aucun profil n'est noté dans le contexte, aucune autre action n'a lieu.
    b) Si le contexte n'a pas été initialisé par un paquet IR, le paquet DOIT être éliminé. La logique du profil indiqué dans l'en-tête IR-DYN (s'il est vérifié par le CRC) détermine quel retour, s'il en est, est à envoyer.

12) Autrement, le profil noté dans le contexte détermine comment le reste du paquet est à traiter. Si le contexte n'a pas été initialisé par un paquet IR, le paquet DOIT être éliminé sans autre action.

La procédure pour trouver la taille des données de retour est la suivante :
Examiner les trois bits qui suivent immédiatement le type de paquet Retour. Quand ces bits sont :
1-7, la taille des données de retour est donnée par les bits ;
0, un octet Taille, qui donne explicitement la taille des données de retour, est présent après l'octet Type de retour.

5.2.7 Formats ROHC de paquet RTP du compresseur au décompresseur

RTP ROHC utilise trois types de paquet pour identifier les en-têtes compressés, et deux pour l'initialisation/rafraîchissement. Le format d'un paquet compressé peut dépendre du mode. Donc, un schéma de dénomination de la forme

<modes format>-<numéro de type de paquet>-<propriétés>

est utilisé pour identifier de façon univoque le format lorsque nécessaire, par exemple, UOR-2, R-1. Pour les formats exacts des types de paquet, voir au paragraphe 5.7.

Type de paquet zéro : R-0, R-0-CRC, UO-0.

C'est le type minimal de paquet, qui est utilisé lorsque les paramètres de toutes les fonctions de SN sont connues par le décompresseur, et que l'en-tête à compresser adhère à ces fonctions. Donc, seul le SN RTP codé en W-LSB a besoin d'être communiqué.

Mode R : Seulement si un CRC est présent (type de paquet R-0-CRC) l'en-tête peut être utilisé comme référence pour la décompression qui va suivre.

Modes U et O : Un petit CRC est présent dans le paquet UO-0.

Type de paquet 1 : R-1, R-1-ID, R-1-TS, UO-1, UO-1-ID, UO-1-TS.

Ce type de paquet est utilisé lorsque le nombre de bits nécessaire pour le SN excède ceux disponibles dans le type de paquet zéro, ou lorsque les paramètres des fonctions de SN pour RTP TS ou IP-ID changent.

Mode R : Les paquets R-1-* ne sont pas utilisés comme références pour la décompression qui va suivre. Les valeurs pour les autres champs que RTP TS ou IP-ID peuvent être communiqués en utilisant une extension, mais ils ne mettent pas à jour le contexte.

Modes U et O : Seules les valeurs de RTP SN, RTP TS et IP-ID peuvent être utilisées comme références pour la future compression. Des valeurs non utilisées pour la mise à jour peuvent être fournies pour les autres champs en utilisant une extension (UO-1-ID).

Type de paquet 2 : UOR-2, UOR-2-ID, UOR-2-TS

Ce type de paquet peut être utilisé pour changer les paramètres de toute fonction SN, sauf celles pour la plupart des champs statiques. Les en-têtes des paquets transférés en utilisant le type de paquet 2 peuvent être utilisés comme références pour la décompression ultérieure.

Type de paquet : IR

Ce type de paquet communique la partie statique du contexte, c'est-à-dire, la valeur des fonctions SN constantes. Il peut aussi facultativement communiquer la partie dynamique du contexte, c'est-à-dire, les paramètres des fonctions SN non constantes.

Type de paquet : IR-DYN

Ce type de paquet communique la partie dynamique du contexte, c'est-à-dire, les paramètres des fonctions SN non constantes.

5.2.8 Paramètres nécessaire pour la transition de mode en RTP ROHC

Ces types de paquet IR (avec des informations dynamiques) IR-DYN, et UOR-2, sont communs à tous les modes. Ils peuvent porter un paramètre de mode qui peut prendre les valeurs U = Unidirectionnel, O = Bidirectionnel optimiste, et R = Bidirectionnel fiable.

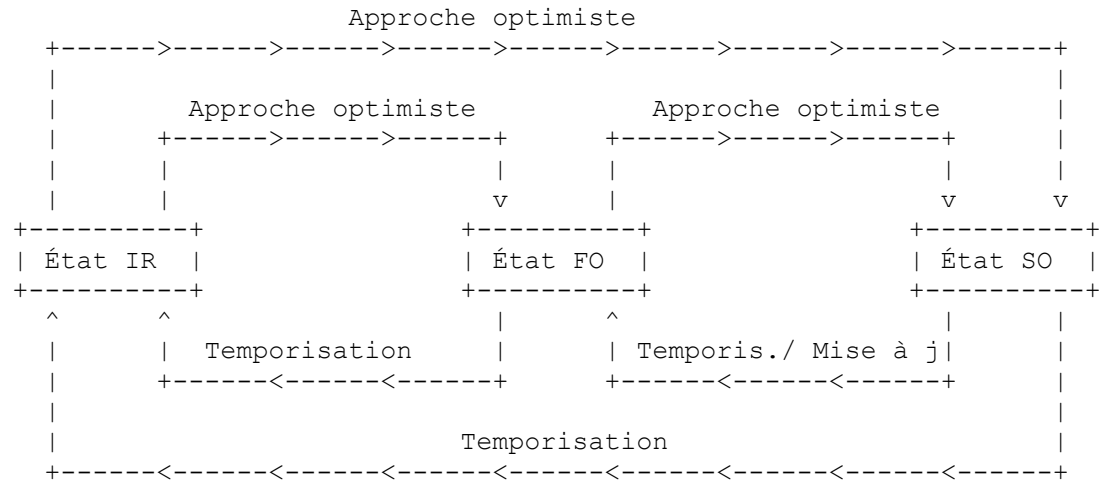
Les types de retour ACK, NACK, et STATIC-NACK portent des numéros de séquence, et les paquets de retour peuvent aussi porter un paramètre de mode qui indique le mode de compression désiré : U, O, ou R.

Comme abréviation, la notation PAQUET(mode) est utilisée pour indiquer quelle valeur de mode porte un paquet. Par exemple, un ACK avec un paramètre de mode R s'écrit ACK(R), et un UOR-2 avec un paramètre de mode O est écrit UOR-2(O).

5.3 Fonctionnement en mode unidirectionnel

5.3.1 États du compresseur et logique (mode U)

Ci-dessous figure l'automate à états pour le compresseur en mode unidirectionnel. Les détails des transitions entre états et logique de compression sont donnés à la suite de la figure.



5.3.1.1 Logique de transition d'état (mode U)

La logique de transition pour les états de compression en mode unidirectionnel se fonde sur trois principes : le principe d'approche optimiste, les temporisations, et le besoin de mise à jour.

5.3.1.1.1 Approche optimiste, transition vers le haut

La transition pour un état de compression supérieur en mode unidirectionnel est effectuée conformément au principe de l'approche optimiste. Cela signifie que le compresseur transite à un état de compression supérieur lorsque il a pleine confiance que le décompresseur a reçu assez d'informations pour correctement décompresser les paquets envoyés conformément à l'état de compression supérieur.

Lorsque le compresseur est dans l'état IR, il va y rester jusqu'à ce qu'il suppose que le décompresseur a correctement reçu les informations statiques de contexte. Pour la transition de l'état FO à l'état SO, le compresseur devrait être sûr que le décompresseur a tous les paramètres nécessaires pour décompresser conformément à un schéma fixé.

Le compresseur obtient normalement sa confiance quant à l'état du décompresseur en envoyant plusieurs paquets avec les mêmes informations conformément à l'état de compression inférieur. Si le décompresseur reçoit un de ces paquets, il va être synchronisé avec le compresseur. Le nombre de paquets consécutifs à envoyer pour créer la confiance n'est pas défini dans le présent document.

5.3.1.1.2. Temporisations, transition vers le bas

Lorsque l'approche optimiste est prise comme décrit ci-dessus, il y aura toujours une possibilité d'échec car le décompresseur peut n'avoir pas reçu des informations suffisantes pour une décompression correcte. Donc, le compresseur DOIT périodiquement transiter vers des états de compression inférieurs. La transition périodique à l'état IR DEVRAIT être effectuée moins souvent que la transition à l'état FO. Deux temporisations différentes DEVRAIENT donc être utilisées pour ces transitions. Pour un exemple de la façon de mettre en œuvre les rafraîchissements périodiques, voir la [RFC2507] paragraphes 3.3.1-3.3.2.

5.3.1.1.3 Besoin de mise à jour, transition vers le bas

En plus des transitions d'état vers le bas effectuées à cause des temporisations périodiques, le compresseur doit aussi revenir immédiatement à l'état FO lorsque l'en-tête à compresser ne se conforme pas au schéma établi.

5.3.1.2 Logique de compression et paquets utilisés (mode U)

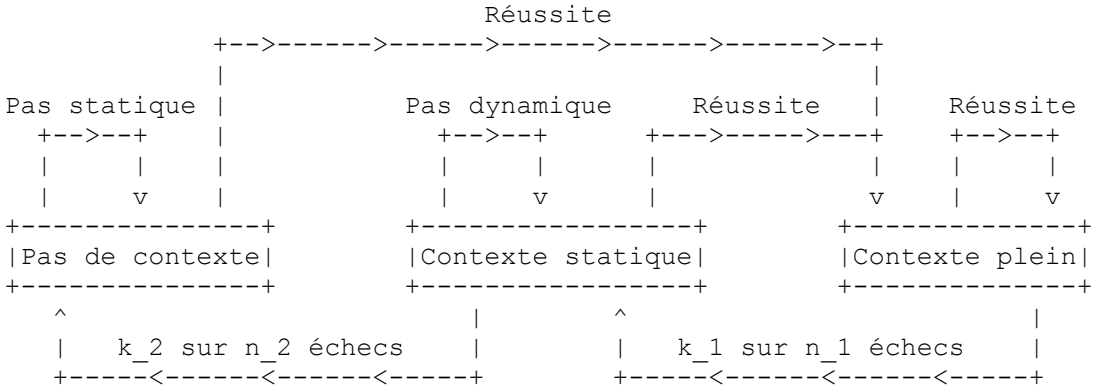
Le compresseur choisit le plus petit format possible qui peut communiquer les changements désirés, et a le nombre requis de bits pour les valeur codées de W-LSB.

5.3.1.3 Retour en mode Unidirectionnel

Le mode de fonctionnement unidirectionnel est conçu pour opérer sur des liaisons où un canal de retour n'est pas disponible. Cependant, si un canal de retour est disponible, le décompresseur PEUT envoyer un accusé de réception de décompression réussie avec le paramètre de mode réglé à U (envoi d'un ACK(U)). Lorsque le compresseur reçoit un tel message, il PEUT désactiver les rafraîchissements IR périodiques (ou augmenter l'intervalle entre eux).

5.3.2 États et logique du décompresseur (mode U)

Ci-dessous est représenté l'automate à états pour le décompresseur en mode unidirectionnel. Les détails des transitions entre les états et la logique de décompression sont donnés à la suite de la figure.



5.3.2.1 Logique de transition d'état (mode U)

Une décompression réussie va toujours passer le décompresseur à l'état Contexte plein. Un échec de décompression répété va forcer le décompresseur à repasser à un état inférieur. Le décompresseur ne tente pas du tout de décompresser les en-têtes dans les états Pas de contexte et Contexte statique sauf si des informations suffisantes sont incluses dans le paquet lui-même.

5.3.2.2 Logique de décompression (mode U)

La décompression en mode unidirectionnel est effectuée en trois étapes qui sont décrites dans les paragraphes suivants.

5.3.2.2.1 Décider si la décompression est permise

Dans l'état Contexte plein, la décompression peut être tentée sans considération de la sorte de paquet qui est reçu. Cependant, pour les autres états, la décompression n'est pas toujours permise. Dans l'état Pas de contexte, seuls les paquets IR, qui portent les champs d'informations statiques, peuvent être décompressés. De plus, dans l'état Contexte statique, seuls les paquets portant un CRC de 7 ou 8 bits peuvent être décompressés (c'est-à-dire, les paquets IR, IR-DYN, ou UOR-2). Si la décompression ne peut pas être effectuée, le paquet est éliminé, sauf si le mécanisme facultatif de décompression retardée est utilisé, voir au paragraphe 6.1.

5.3.2.2.2 Reconstruire et vérifier l'en-tête

Lors de la reconstruction de l'en-tête, le décompresseur prend les informations d'en-tête déjà mémorisées dans le contexte et le met à jour avec les informations reçues dans l'en-tête en cours. (Si l'en-tête reconstruit échoue à la vérification de CRC, ces mises à jour DOIVENT être défaites)

Le numéro de séquence est reconstruit en remplaçant les LSB du numéro de séquence dans le contexte par ceux reçus dans l'en-tête. La valeur résultante est alors vérifiée comme étant dans l'intervalle d'interprétation en comparant avec une valeur v_ref de référence précédemment reconstruite (voir au paragraphe 4.5.1). Si elle n'est pas dans cet intervalle, un ajustement est appliqué en ajoutant N x interval_size à la valeur reconstruite de sorte que le résultat soit amené dans l'intervalle d'interprétation. Noter que N peut être négatif.

Si les champs RTP Horodatage et Identification IP ne sont pas inclus dans l'en-tête reçu, ils sont supposés être calculés à partir du numéro de séquence. L'identifiant IP augmente usuellement du même delta que le numéro de séquence et l'horodatage du même delta fois une valeur fixe. Voir aux paragraphes 4.5.3 et 4.5.5 les détails de la façon dont ces champs sont codés dans les en-têtes compressés.

En mode unidirectionnel, tous les en-têtes compressés portent un CRC qui DOIT être utilisé pour vérifier la décompression.

5.3.2.2.3 Actions sur échec du CRC

Ce paragraphe est écrit de façon à être applicable à tous les modes.

Une discordance dans le CRC peut être causée par un ou plusieurs des événements suivants :

1. erreurs binaires résiduelles dans l'en-tête en cours,
2. contexte endommagé à cause d'erreurs binaires résiduelles dans les en-têtes précédents,
3. nombreux paquets consécutifs perdus entre le compresseur et le décompresseur (ceci peut être cause que les LSB du SN dans les paquets compressés soient mal interprétés, parce que le décompresseur n'a pas déplace l'intervalle d'interprétation par manque d'entrées -- par essence, une sorte de dommage de contexte).

(Les cas 2 et 3 ne s'appliquent pas aux paquets IR ; le cas 3 ne s'applique pas aux paquets IR-DYN.) Le CRC de 3 bits présent dans certains formats d'en-tête va éventuellement détecter fiablement un dommage de contexte, car la probabilité de dommage de contexte non détecté décroît de façon exponentielle avec chaque nouvel en-tête traité. Cependant, les erreurs binaires résiduelles dans l'en-tête en cours ne sont détectées qu'avec une bonne probabilité, et non fiablement.

Lorsque une discordance de CRC est causée par des erreurs binaires résiduelles dans l'en-tête en cours (cas 1 ci-dessus) le décompresseur devrait rester dans son état en cours pour éviter des pertes inutiles des paquets suivants. D'un autre côté, lorsque la discordance est causée par un contexte endommagé (cas 2) le décompresseur devrait tenter de réparer le contexte en local. Si la tentative de réparation locale échoue, il doit passer à un état inférieur pour éviter de livrer des en-têtes incorrects. Lorsque la discordance est causée par des pertes prolongées (cas 3) le décompresseur peut essayer des tentatives de décompression supplémentaires. Noter que le cas 3 ne survient pas en mode R.

Les actions suivantes DOIVENT avoir lieu lorsque une vérification de CRC échoue :

D'abord, tenter de déterminer si un retour à zéro de LSB SN (cas 3) est vraisemblable, et si il l'est, tenter une correction. Pour cela, l'algorithme du paragraphe 5.3.2.2.4 PEUT être utilisé. Si un autre algorithme est utilisé, il DOIT avoir au moins un taux de réparations correctes aussi élevé que celui du paragraphe 5.3.2.2.4. (Cette étape n'est pas applicable au mode R.)

Ensuite, si l'étape précédente n'a pas donné de correction, une réparation devrait être tentée dans l'hypothèse que le SN de référence a été incorrectement mis à jour. Pour cela, l'algorithme du paragraphe 5.3.2.2.5 PEUT être utilisé. Si un autre algorithme est utilisé, il DOIT avoir au moins un taux de réparations correctes aussi élevé que celui du paragraphe 5.3.2.2.5. (Cette étape n'est pas applicable au mode R.)

Si les deux étapes ci-dessus échouent, des tentatives de décompression supplémentaires NE DEVRAIENT PAS être faites. Il y a deux raisons possibles à l'échec du CRC : le cas 1 ou un dommage de contexte irrécupérable. Il est impossible de savoir avec certitude laquelle des deux est la cause réelle. Les règles suivantes sont à utiliser :

a. Lorsque un CRC n'échoue qu'occasionnellement, on suppose des erreurs résiduelles dans l'en-tête en cours et on élimine simplement le paquet. Des NACK NE DEVRAIENT PAS être envoyés à ce moment.
b. Dans l'état Contexte plein : Lorsque la vérification de CRC de k_1 sur les derniers n_1 paquets compressés ont échoué, un dommage de contexte DEVRAIT être supposé et un NACK DEVRAIT être envoyé en modes O et R. Le décompresseur passe à l'état Contexte statique et élimine tous les paquets jusqu'à ce que soit reçue une mise à jour (IR, IR-DYN, UOR-2) qui passe la vérification de CRC.
c. Dans l'état Contexte statique : Lorsque la vérification de CRC de k_2 sur les n_2 dernières mises à jour (IR, IR-DYN, UOR-2) a échoué, on DEVRAIT supposer un dommage de contexte statique et un STATIC-NACK est envoyé en modes O et R. Le décompresseur passe à l'état Pas de contexte.
d. Dans l'état Pas de contexte : Le décompresseur élimine tous les paquets jusqu'à ce qu'une mise à jour statique (IR) qui réussisse le CRC soit reçue. (En mode O et R les retours sont envoyés conformément aux paragraphes 5.4.2.2 et 5.5.2.2, respectivement.)

Noter que les valeurs appropriées pour k_1, n_1, k_2, et n_2, sont en relation avec le taux d'erreurs résiduelles de la liaison. Lorsque le taux d'erreurs résiduelles est proche de zéro, k_1 = n_1 = k_2 = n_2 = 1 peut être approprié.

5.3.2.2.4 Correction du retour à zéro de LSB de SN

Lorsque de nombreux paquets consécutifs sont perdus, il y a un risque de retour à zéro de LSB de numéro de séquence, c'est-à-dire que les LSB de SN soient mal interprétés à cause du non déplacement de l'intervalle d'interprétation par manque d'entrées. Le décompresseur pourrait être capable de détecter cette situation et éviter des dommages de contexte par l'utilisation d'une horloge locale. L'algorithme suivant PEUT être utilisé :

a. Le décompresseur note l'heure d'arrivée, a(i), de chaque paquet entrant i. L'heure d'arrivée des paquets où échoue la décompression est éliminée.
b. Lorsque la décompression échoue, le décompresseur calcule INTERVAL = a(i) - a(i - 1), c'est-à-dire, le temps écoulé entre l'arrivée du précédent paquet correctement décompressé et le paquet en cours.
c. Si un retour à zéro s'est produit, INTERVAL va correspondre à au moins 2^k temps inter paquet, où k est le nombre de bits de SN dans l'en-tête en cours. Sur la base d'une estimation du temps d'arrivée inter paquet, obtenue par exemple en utilisant une moyenne mobile des heures d'arrivée, TS_STRIDE, ou TS_TIME, le décompresseur juge si INTERVAL peut correspondre à 2^k temps inter paquet.
d. Si INTERVAL est jugé être au moins 2^k temps d'arrivée inter paquet, le décompresseur ajoute 2^k au SN de référence et tente de décompresser le paquet en utilisant le nouveau SN de référence.
e. Si cette décompression réussit, le décompresseur met à jour le contexte mais NE DEVRAIT PAS livrer le paquet aux couches supérieures. Le paquet suivant est aussi décompressé et met à jour le contexte si son CRC est vérifié, mais DEVRAIT être éliminé. Si la décompression du troisième paquet en utilisant le nouveau contexte réussit aussi, la réparation du contexte est réputée réussie et ce paquet décompressé et les suivants sont livrés aux couches supérieures.
f. Si une de ces trois tentatives de décompression de d. et e. échoue, le décompresseur élimine les paquets et agit

conformément aux règles a) à c) du paragraphe 5.3.2.2.3.

En utilisant ce mécanisme, le décompresseur peut être capable de réparer le contexte après une perte excessive, au prix de l'élimination de deux paquets.

5.3.2.2.5 Réparation de mises à jour incorrectes de SN

Le CRC peut échouer à détecter les erreurs résiduelles dans l'en-tête compressé à cause de sa longueur limitée, c'est-à-dire, il peut se trouver que le paquet incorrectement décompressé ait le même CRC qu'un paquet compressé original. L'en-tête décompressé incorrect va alors mettre à jour le contexte. Cela peut conduire à ce qu'un numéro de séquence de référence erroné soit utilisé dans le décodage W-LSB, car le SN de référence est mis à jour pour chaque en-tête décompressé réussi de certains types.

Dans cette situation, le décompresseur va détecter la décompression incorrecte du paquet suivant avec une forte probabilité, mais il ne sait pas la raison de l'échec. Le mécanisme suivant permet au décompresseur de juger si le contexte a été incorrectement mis à jour par un paquet antérieur et, si c'est le cas, de tenter une réparation.

a. Le décompresseur entretient deux numéros de séquence décompressés : le dernier (ref 0) et celui d'avant (ref -1).

b. À la réception d'un en-tête compressé, le SN (SN curr1) est décompressé en utilisant ref 0 comme référence. Les autres champs d'en-tête sont décompressés en utilisant ce SN curr1 décompressé. (Cela fait partie de la procédure normale de décompression avant tout échec de vérification de CRC.)

c. Si l'en-tête décompressé généré en b réussit la vérification de CRC, les références sont changés comme suit :
   ref -1 = ref 0
   ref 0 = SN curr1.

d. Si l'en-tête généré en b ne satisfait pas la vérification de CRC, et si le SN (SN curr2) généré en utilisant ref -1 comme référence est différent de SN curr1, une tentative supplémentaire de décompression est effectuée sur la base de SN curr2 comme SN décompressé.

e. Si l'en-tête décompressé généré en b ne satisfait pas à la vérification de CRC et si SN curr2 est le même que SN curr1, une tentative supplémentaire de décompression n'est pas utile et n'est pas tentée.

f. Si l'en-tête décompressé généré en d réussit à la vérification du CRC, ref -1 n'est pas changé tandis que ref 0 est réglé à SN curr2.

g. Si l'en-tête décompressé généré en d ne réussit pas à la vérification du CRC, le décompresseur agit conformément aux règles de a) à c) du paragraphe 5.3.2.2.3.

L'objet de cet algorithme est de réparer le contexte. Si l'en-tête généré en d. réussit à la vérification de CRC, les références sont mises à jour conformément à f., mais deux en-têtes de plus DOIVENT aussi réussir la décompression avant que la réparation soit réputée réussie. Des trois en-têtes réussis, les deux premiers DEVRAIENT être éliminés et seul le troisième livré aux couches supérieures. Si la décompression d'un des trois en-têtes échoue, le décompresseur DOIT éliminer cet en-tête et les en-têtes générés précédemment, et agir conformément aux règles a) à c) du paragraphe 5.3.2.2.3.

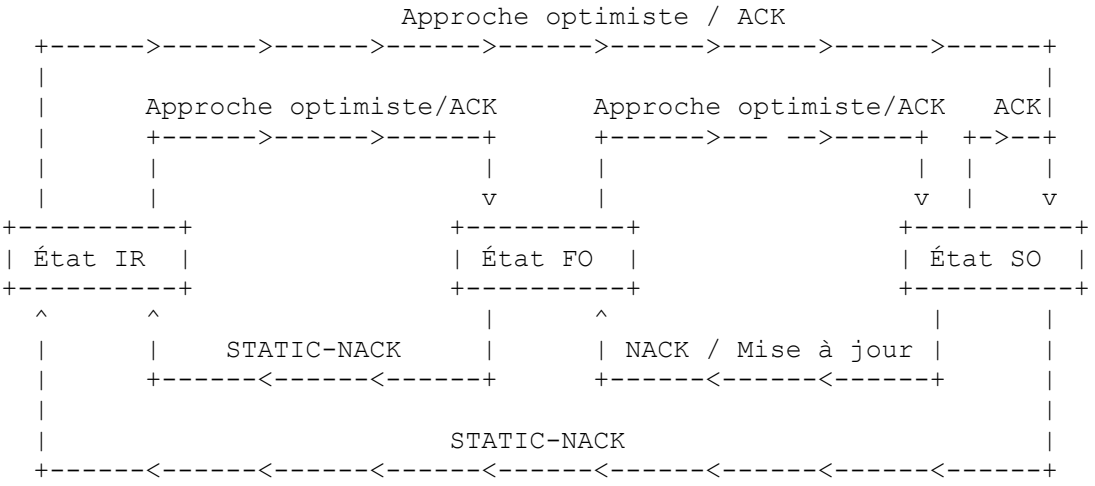
5.3.2.3 Retour en mode unidirectionnel

Pour améliorer les performances pour le mode unidirectionnel sur une liaison qui n'a pas de canal de retour, le décompresseur PEUT envoyer un accusé de réception lorsque la décompression réussit. En réglant le paramètre de mode dans le paquet ACK à U, on indique que le compresseur va rester en mode unidirectionnel. À réception d'un ACK(U), le compresseur devrait réduire la fréquence des paquets IR car les informations statiques ont été reçues correctement, mais il n'est pas exigé d'arrêter l'envoi de paquets IR. Si des paquets IR continuent d'arriver, le décompresseur PEUT répéter l'ACK(U), mais il NE DEVRAIT PAS répéter les ACK(U) continuellement.

5.4 Fonctionnement en mode bidirectionnel optimiste

5.4.1 États du compresseur et logique (O-mode)

La figure ci-dessus montre l'automate à états pour le compresseur en mode Bidirectionnel optimiste. Les détails de chaque état, transition d'état, et logique de compression, sont donnés à la suite de la figure.



5.4.1.1 Logique de transition d'état

La logique de transition pour les états de compression en mode bidirectionnel optimiste a beaucoup en commun avec celle du mode unidirectionnel. Le principe de l'approche optimiste et des transitions occasionnées par le besoin de mises à jour fonctionne de la même façon que décrit au paragraphe 5.3.1. Cependant, en mode optimiste, il n'y a pas de temporisations. À la place, le mode optimiste utilise un retour du décompresseur au compresseur pour les transitions dans la direction vers le bas et pour la transition FACULTATIVE améliorée vers l'avant.

5.4.1.1.1 Accusé de réception négatif (NACK), transition vers le bas

Les accusés de réception négatifs (NACK) aussi appelés demandes de contexte, évitent les mises à jour périodiques nécessaires en mode unidirectionnel. À réception d'un NACK, le compresseur revient à l'état FO et envoie des mises à jour (IR-DYN, UOR-2, ou éventuellement IR) au décompresseur.
Les NACK portent le SN du dernier paquet décompressé avec succès, et ces informations PEUVENT être utilisées par le compresseur pour déterminer quel champ a besoin d'être mis à jour.

De façon similaire, la réception d'un paquet STATIC-NACK fait revenir le compresseur à l'état IR.

5.4.1.1.2 Accusés de réception facultatifs, transition vers le haut

En plus des NACK, un retour positif (ACK) PEUT aussi être utilisé pour les paquets UOR-2 dans le mode bidirectionnel optimiste. À réception d'un ACK pour un paquet de mise à jour, le compresseur sait que le décompresseur a reçu le paquet dont il est accusé réception et que la transition à un état de compression supérieur peut être effectuée immédiatement. Cette fonction est facultative, de sorte qu'un compresseur NE DOIT PAS s'attendre à obtenir de tels ACK initialement.

Le compresseur PEUT utiliser l'algorithme suivant pour déterminer quand s'attendre à des ACK pour des paquets UOR-2. Soit un événement de mise à jour lorsque une séquence d'en-têtes UOR-2 est envoyée pour communiquer une irrégularité dans le flux de paquets. Lorsque des ACK ont été reçus pour k_3 sur les n_3 derniers événements de mise à jour, le compresseur va s'attendre à des ACK. Un compresseur qui s'attend à des ACK va répéter les mises à jour (éventuellement pas dans chaque paquet) jusqu'à ce qu'un ACK soit reçu.

5.4.1.2 Logique de compression et paquets utilisés

La logique de compression est la même pour le mode bidirectionnel optimiste que pour le mode unidirectionnel (voir au paragraphe 5.3.1.2).

5.4.2 États et logique au décompresseur (mode O)

Les états de décompression et la logique de transition d'état sont les mêmes que pour le cas unidirectionnel (voir au paragraphe 5.3.2). Ce qui diffère est la logique de décompression et de retours.

5.4.2.1 Logique de décompression, décompression d'horodatage fondé sur la temporisation

En mode bidirectionnel (ou si il y a d'autres moyens pour le compresseur d'obtenir la résolution d'horloge et la gigue de liaison du décompresseur) la décompression d'horodatage fondé sur la temporisation peut être utilisée pour améliorer l'efficacité de compression lorsque les valeurs d'horodatage RTP sont proportionnelles à l'heure de l'horloge. Les mécanismes utilisés sont ceux décrits en 4.5.4.

5.4.2.2 Logique des retours (mode O)

La logique des retours définit quels retours envoyer en fonction des différents événements en fonctionnement dans les divers états. Comme mentionné ci-dessus, il y a trois principales sortes de retours ; ACK, NACK et STATIC-NACK. De plus, la logique décrite ci-dessous se réfère aux différentes sortes de paquets qui peuvent être reçus par le décompresseur ; paquets d'initialisation et rafraîchissement (IR) paquets IR sans informations statiques (IR-DYN) et paquets de type 2 (UOR-2), ou de type 1 (UO-1) et de type 0 (UO-0). Un paquet de type 0 porte un en-tête de paquet compressé conformément à un schéma fixé, alors que les paquets de type 1, 2 et IR-DYN sont utilisés lorsque ce schéma ne fonctionne plus.

Ci-dessous, des règles sont définies établissant quel retour utiliser dans les différentes conditions. Si le retour facultatif est utilisé une fois, il est EXIGÉ du décompresseur qu'il continue d'envoyer des retours facultatifs pour la durée de vie du flux de paquets.

Actions d'état

NC : - Lorsque un paquet IR réussit la vérification de CRC, envoyer un ACK(O).
- À réception d'un paquet de type 0, 1, 2 ou IR-DYN, ou si un paquet IR a échoué à la vérification de CRC, envoyer un STATIC-NACK(O), sous réserve des considérations du début du paragraphe 5.7.6.

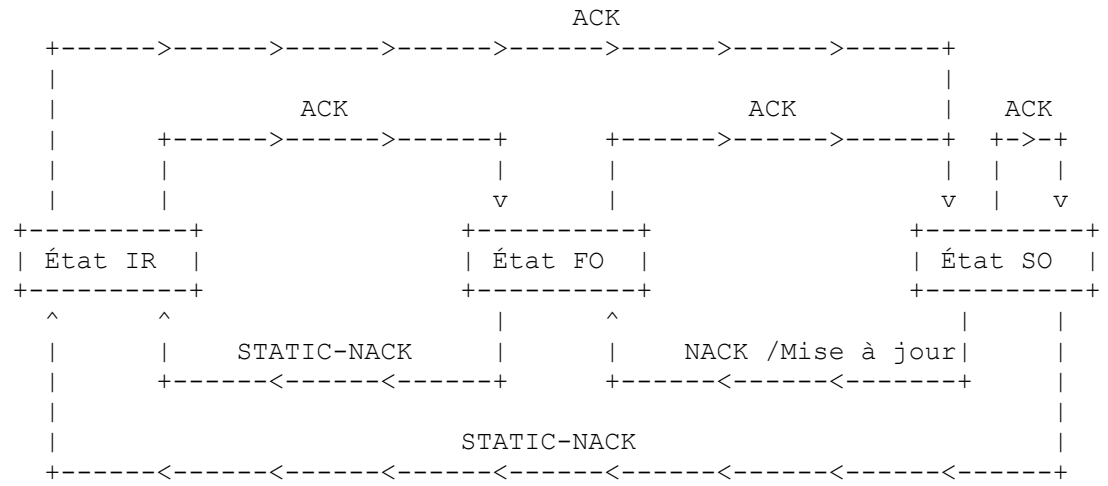
SC : - Lorsque un paquet IR est correctement décompressé, envoyer un ACK(O).
- Lorsque un paquet de type 2 ou IR-DYN est correctement décompressé, envoyer facultativement un ACK(O).
- Lorsque un paquet de type 0 ou 1 est reçu, le traiter comme un CRC discordant et utiliser la logique du paragraphe 5.3.2.2.3 pour décider si un NACK(O) devrait être envoyé.
- Lors de la décompression d'un paquet de type 2, un paquet IR-DYN ou un paquet IR a échoué, utiliser la logique du paragraphe 5.3.2.2.3 pour décider si un STATIC-NACK(O) devrait être envoyé.

FC : - Lorsque un paquet IR est correctement décompressé, envoyer un ACK(O).
- Lorsque un paquet de type 2 ou IR-DYN est correctement décompressé, envoyer facultativement un ACK(O).
- Lorsque un paquet de type 0 ou 1 est correctement décompressé, aucun retour n'est envoyé.
- Lorsque un paquet échoue à la vérification du CRC, utiliser la logique du 5.3.2.2.3 pour décider si un NACK(O) devrait être envoyé.

5.5 Fonctionnement en mode bidirectionnel fiable

5.5.1 États du compresseur et logique (mode R)

Ci-dessous est l'automate à états pour le compresseur en mode bidirectionnel fiable. Les détails de chaque état, les transitions d'état, et la logique de compression sont donnés à la suite de la figure.



5.5.1.1 Logique de transition d'état (mode R)

La logique de transition pour les état de compression en mode fiable se fonde sur trois principes : de référence sûre, du besoin de mise à jour, et d'accusés de réception négatifs.

5.5.1.1.1 Transition vers le haut

La transition vers le haut est déterminée par le principe de référence sûre. La procédure de transition est similaire à celle décrite au paragraphe 5.3.1.1.1, avec une importante différence : le compresseur fonde sa confiance sur les seuls accusés de réception reçus du décompresseur. Cela assure que la synchronisation entre les contextes de compression et de décompression ne sera jamais perdue à cause de pertes de paquets.

5.5.1.1.2 Transition sers le bas

Les transitions vers le bas sont déclanchées par le besoin de mises à jour ou par des accusés de réception négatifs (NACK et STATIC_NACK) comme décrit aux paragraphes 5.3.1.1.3 et 5.4.1.1.1, respectivement. Noter que les NACK devraient rarement survenir en mode R à cause de la référence sûre utilisée (voir le quatrième alinéa du paragraphe suivant).

5.5.1.2 Logique de compression et paquets utilisés (mode R)

Le compresseur commence dans l'état IR par envoyer des paquets IR. Il transite à l'état FO une fois qu'il a reçu un ACK valide pour un paquet IR envoyé (un ACK ne peut être valide que si il se réfère à un SN envoyé précédemment). Dans l'état FO, il envoie les plus petits paquets qui peuvent communiquer les changements, conformément aux règles de codage W-LSB ou autres. Ces paquets pourraient être du type R-1*, UOR-2, ou même IR-DYN.

Le compresseur va transiter à l'état SO après qu'il a déterminé la présence d'une chaîne (voir au paragraphe 2) tout en ayant confiance que le décompresseur a les paramètres de la chaîne. La confiance peut se fonder sur des ACK. Par exemple, dans un cas normal où le schéma de la chaîne a la forme d'un champ non SN = SN * pente + décalage, un ACK est suffisant si la pente a été précédemment établie par le décompresseur (c'est-à-dire, seul le nouveau décalage a besoin d'être synchronisé). Autrement, deux ACK sont nécessaires car le décompresseur a besoin de deux en-têtes pour apprendre à la fois la nouvelle pente et le nouveau décalage. Dans l'état SO, R-0* paquets seront envoyés.

Noter qu'une transition directe de l'état IR à l'état SO est possible.

Le principe de référence sûre est mis en application dans les deux logiques de compression et décompression. Le principe signifie que seul un paquet qui porte un CRC de 7 ou 8 bits peut mettre à jour le contexte de décompression et être utilisé comme référence pour la décompression ultérieure. Par conséquent, seules les valeurs de champs de paquets de mise à jour ont besoin d'être ajoutées à la fenêtre glissante de codage (voir le paragraphe 4.5) entretenue par le compresseur.

Pour le compresseur, les raisons d'envoyer des paquets de mise à jour incluent :

1) que la mise à jour peut conduire à une transition à une meilleure efficacité de compression (signifiant un état de compression supérieur ou de plus petits paquets dans le même état) ;
2) qu'il est désirable de réduire la taille des fenêtres glissantes. Les fenêtres ne sont réduites que lorsque un ACK est reçu.

La génération d'un CRC est peu fréquente car il n'est nécessaire que pour un paquet de mise à jour.

Un algorithme d'envoi des paquets de mise à jour pourrait être :

* Soit pRTT le nombre de paquets envoyés durant un délai d'aller-retour. Dans l'état SO, lorsque (64 - pRTT) en-têtes ont été envoyés depuis la dernière référence acquittée, le compresseur va envoyer m1 en-têtes R-0-CRC consécutifs, puis envoyer (pRTT - m1) en-têtes R-0. Après l'envoi de ces en-têtes, si le compresseur n'a pas reçu un ACK à au moins un des R0-CRC précédemment envoyés, il envoie continûment des en-têtes R-0-CRC jusqu'à ce qu'il reçoive un ACK correspondant. "m1" est un paramètre de mise en œuvre, qui peut être aussi grand que pRTT.
* Dans l'état FO, m2 UOR-2 en-têtes sont envoyés lorsque il y a un changement de schéma, après quoi le compresseur envoie (pRTT - m2) R-1-* en-têtes. "m2" est un paramètre de mise en œuvre, qui peut être aussi grand que pRTT. À ce moment, si le compresseur n'a pas reçu assez de ACK aux paquets UOR-2 envoyés précédemment afin de passer à l'état SO, il peut répéter le cycle avec le même m2, ou répéter le cycle avec un plus grand, ou envoyer en continu UOR-2 en-têtes (m2 = pRTT). L'opération s'arrête lorsque le compresseur a reçu assez de ACK pour faire la transition.

Un algorithme de traitement des ACK pourrait être :

* À réception d'un ACK, le compresseur déduit d'abord le numéro de séquence complet (voir au paragraphe 5.7.6.1). Puis il cherche dans la fenêtre glissante un paquet de mise à jour qui a le même SN. S'il en trouve, ce paquet est celui qui sera acquitté. Autrement, le ACK est invalide et DOIT être éliminé.
* Il est possible, bien que peu vraisemblable, que les erreurs résiduelles sur le canal inverse puissent être cause qu'un paquet mime un ACK de retour valide. Le compresseur peut utiliser une horloge locale pour réduire la probabilité de traiter un tel simulacre de ACK. Après avoir trouvé le paquet de mise à jour comme décrit ci-dessus, le compresseur peut vérifier le temps écoulé depuis l'envoi du paquet. Si ce temps est supérieur à RTT_U, ou plus court que RTT_L, le compresseur peut choisir d'éliminer le ACK. RTT_U et RTT_L correspondent aux limites, respectivement supérieure et inférieure du délai d'aller-retour. (Ces bornes devraient être choisies de façon à permettre une certaine variation du RTT.) Noter que le seul effet collatéral de l'élimination d'un bon ACK est une efficacité de compression légèrement réduite.

5.5.2 États du décompresseur et logique (mode R)

Les états de décompression et la logique de transition d'état sont les mêmes que dans le cas unidirectionnel (voir au paragraphe 5.3.2). Ce qui diffère est la logique de décompression et de retours.

5.5.2.1 Logique de décompression (mode R)

Les règles sur le moment où la décompression est permise sont les mêmes que pour le mode U. Bien que le schéma d'accusé de réception en mode R garantisse que les paquets non décompressibles ne sont jamais envoyés par le compresseur, des erreurs résiduelles peuvent causer la livraison de paquets inattendus pour lesquels la décompression ne devrait pas être tentée.

La décompression DOIT suivre le principe de référence sûre décrit au paragraphe 5.5.1.2.

La vérification du CRC est peu fréquente car seuls les paquets de mise à jour portent des CRC. Une discordance de CRC ne peut se produire qu'à cause 1) d'erreurs binaires résiduelles dans l'en-tête en cours, et/ou 2) d'un contexte endommagé dû à des erreurs binaires résiduelles dans les en-têtes ou retours précédents. Bien qu'il soit impossible de déterminer quelle est la cause réelle, le cas 1 est plus probable, car un en-tête précédent reconstruit selon un paquet endommagé a peu de chances de réussir la vérification de CRC à 7 ou 8 bits, et les paquets endommagés ont peu de chances de résulter en un retour qui endommage le contexte. Le décompresseur DEVRAIT agir conformément au paragraphe 5.3.2.2.3 lorsque les CRC échouent, sauf qu'aucune réparation locale n'est effectuée. Noter que tous les numéros de paramètre, k_1, n_1, k_2, et n_2, sont appliqués aux seuls paquets de mise à jour (c'est-à-dire, à l'exclusion de R-0, R-1*).

5.5.2.2 Logique de retour (mode R)

La logique des retours pour le mode bidirectionnel fiable est la suivante :

- Lorsque un paquet de mise à jour (c'est-à-dire, un paquet qui porte un CRC de 7 ou 8 bits) est correctement décompressé, envoyer un ACK(R), sous réserve du mécanisme d'ACK clairsemé décrit ci-dessous.
- Lorsque un dommage de contexte est détecté, envoyer un NACK(R) si on est dans l'état Contexte plein, ou un STATIC-NACK(R) si on est dans l'état Contexte statique.
- Dans l'état Pas de contexte, envoyer un STATIC-NACK(R) à réception d'un paquet non IR, sous réserve des considérations du début du paragraphe 5.7.6. Le décompresseur NE DEVRAIT PAS envoyer un STATIC-NACK(R) lors de la réception d'un paquet IR qui échoue à la vérification de CRC, car le compresseur va rester dans l'état IR et donc continuer d'envoyer des paquets IR jusqu'à ce qu'un ACK valide soit reçu (voir au paragraphe 5.5.1.2).
- Un retour n'est jamais envoyé pour les paquets qui ne mettent pas à jour le contexte (c'est-à-dire, les paquets qui ne portent pas de CRC)

Un mécanisme dit de "ACK clairsemé" peut être appliqué pour réduire la redondance de retour causée par un RTT élevé. Pour une séquence d'événements déclencheurs d'accusés de réception, un ensemble minimal d'ACK DOIT être envoyé :

1) pour une séquence de R-0-CRC paquets, le premier DOIT être acquitté :
2) pour une séquence de paquets UOR-2, IR, ou IR-DYN, les N premiers d'entre eux DOIVENT être acquittés, N étant le nombre de ACK nécessaire pour donner au compresseur la confiance que le décompresseur a acquis la nouvelle chaîne de paramètres (voir le second alinéa du paragraphe 5.5.1.2). Au cas où le décompresseur ne peut pas déterminer la valeur de N, la valeur par défaut 2 DEVRAIT être utilisée. Si les paquets reçus ensuite continuent le même schéma de changement des champs d'en-tête, l'ACK clairsemé peut être appliqué. Autrement, chaque nouveau schéma DOIT être traité comme une nouvelle séquence, c'est-à-dire, les N premiers paquets qui exhibent un nouveau schéma DOIVENT être acquittés.

Après l'envoi de ces ACK minimaux, le décompresseur PEUT choisir de n'accuser réception que des k paquets suivants par RTT ("ACK clairsemés") où k est un paramètre de mise en œuvre. Pour assurer la robustesse contre la perte des ACK, k DEVRAIT être au moins de 1.

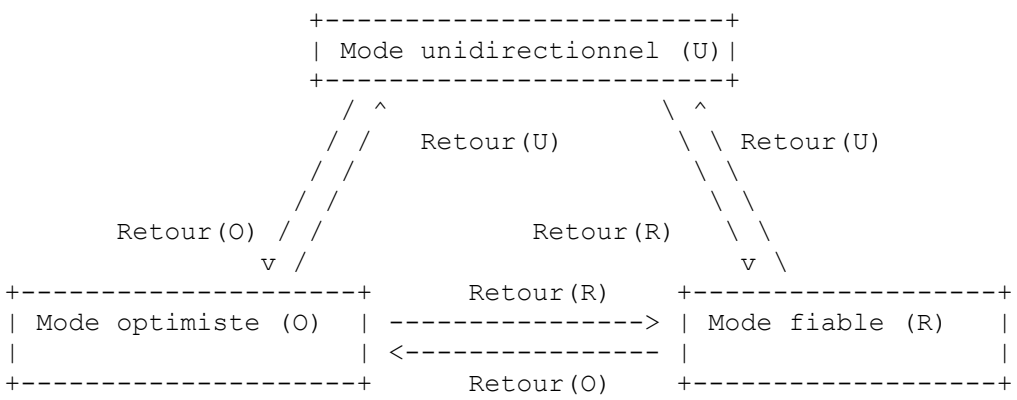
Pour éviter des ambiguïtés au compresseur, le décompresseur DOIT utiliser le format de retour dont la longueur de champ SN est égale ou supérieure à celle qui est dans le paquet compressé qui a déclanché le retour.

Le dommage du contexte est détecté conformément aux principes du paragraphe 5.3.2.2.3.

Lorsque le décompresseur est capable de compression fondée sur la temporisation de l'horodatage RTP (par exemple, il a accès à une horloge qui a une résolution suffisante, et la gigue introduite en interne dans le nœud receveur est suffisamment petite) il DEVRAIT signaler qu'il est prêt à faire la compression fondée sur la temporisation de l'horodatage RTP. Le compresseur va alors prendre une décision sur la base de sa connaissance du canal et des propriétés observées du flux de paquets.

5.6 Transitions de mode

La décision de passer d'un mode de compression à un autre est prise par le décompresseur et les transitions de mode possibles sont indiquées dans la figure ci-dessous. Les paragraphes qui suivent décrivent comment sont effectuées les transitions ainsi que les exceptions pour les fonctions de compression et décompression durant les transitions.



5.6.1 Compression et décompression durant les transitions de mode

Les paragraphes suivants supposent que pour chaque contexte le compresseur et le décompresseur entretiennent une variable dont la valeur est le mode de compression en cours pour ce contexte. La valeur de la variable contrôle, pour le contexte en question, quels types de paquet utiliser, quelles actions entreprendre, etc.

Comme garde-fou contre les erreurs résiduelles, tout retour envoyé durant une transition de mode DOIT être protégé par un CRC, c'est-à-dire que l'option CRC DOIT être utilisée. Une transition de mode NE DOIT PAS être initiée par un retour qui ne serait pas protégé par un CRC.

Les paragraphes qui suivent définissent exactement quand changer la valeur de la variable MODE. Lorsque ROHC transite entre deux modes de compression, il y a plusieurs cas où le comportement du compresseur ou décompresseur doit être contraint durant la phase de transition. Les restrictions sont définies par des paramètres d'exception qui spécifient quelles restrictions appliquer. Les descriptions des transitions dans les paragraphes qui suivent se réfèrent à ces paramètres d'exception et définissent quand ils sont établis et à quelles valeurs. Tous les paramètres en rapport avec le mode sont énumérés ci-dessous avec leurs valeurs possibles, des explications et les restrictions :

Paramètres pour le côté compresseur :

- C_MODE :
  Les valeurs possibles pour le paramètre C_MODE sont (U)nidirectionnel, (O)ptimiste et (R)eliable *(fiable)*. C_MODE DOIT être initialisé à U.

- C_TRANS :
  Les valeurs possibles pour le paramètre C_TRANS sont (P)ending *(en cours)* et (D)one *(fait)*. C_TRANS DOIT être initialisé à D. Lorsque C_TRANS est P, il est EXIGÉ :
  1) que le compresseur n'utilise que les formats de paquet commun à tous les modes,
  2) que les informations de mode soient incluses dans les paquets envoyés, au moins périodiquement,
  3) que le compresseur ne transite pas à l'état SO,
  4) que les nouvelles demandes de transition de mode soient ignorées.

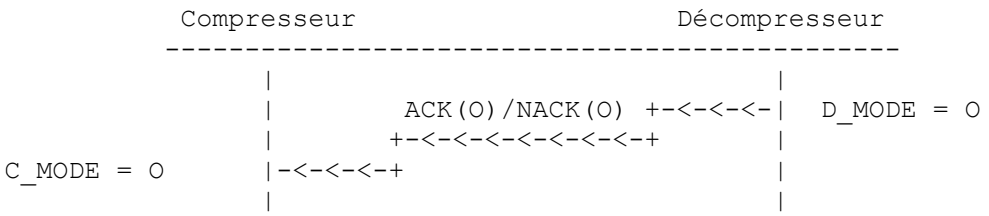
Paramètres pour le côté décompresseur :

- D_MODE :
  Les valeurs possibles pour le paramètre D_MODE sont (U)nidirectionnel, (O)ptimiste et (R)eliable *(fiable)*. _MODE DOIT être initialisé à U.

- D_TRANS :
  Les valeurs possibles pour le paramètre D_TRANS sont (I)nitialisé, (P)ending *(en cours)* et (D)one *(en cours)*. D_TRANS DOIT être initialisé à D. Une transition de mode ne peut être initialisée que lorsque D_TRANS est D. Lorsque D_TRANS est I, le décompresseur envoie un NACK ou ACK portant l'option CRC pour chaque paquet reçu.

5.6.2 Transition du mode Unidirectionnel au mode Optimiste

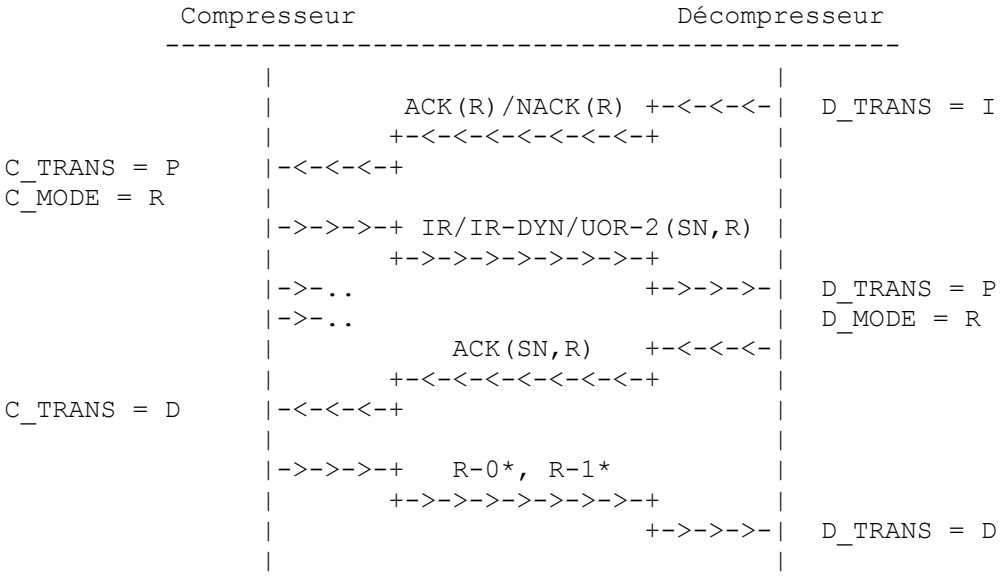
Lorsque il y a un canal de retour disponible, le décompresseur peut à tout moment décider d'initialiser une transition du mode unidirectionnel au mode bidirectionnel optimiste. Tout paquet de retour qui porte un CRC peut être utilisé avec le paramètre de mode réglé à O. Le décompresseur peut alors commencer à travailler directement en mode optimiste. Le compresseur passe du mode unidirectionnel au mode optimiste aussitôt qu'il reçoit un paquet de retour qui a le paramètre de mode réglé à O et qui réussit la vérification du CRC. La procédure de transition est décrite ci-dessous :



Si le paquet de retour est perdu, le compresseur va continuer de fonctionner en mode unidirectionnel, mais aussitôt qu'un paquet de retour atteint le compresseur, il va passer au mode optimiste.

5.6.3 Du mode optimiste au mode fiable

La transition du mode optimiste au mode fiable n'est permise qu'après qu'au moins un paquet a été correctement décompressé, ce qui signifie qu'au moins la partie statique du contexte est établie. Un paquet de retour ACK(R) ou NACK(R) portant un CRC est envoyé pour initialiser la transition de mode. Le compresseur NE DOIT PAS utiliser de paquet des types 0 ou 1 durant la transition. La procédure de transition est décrite ci-dessous :



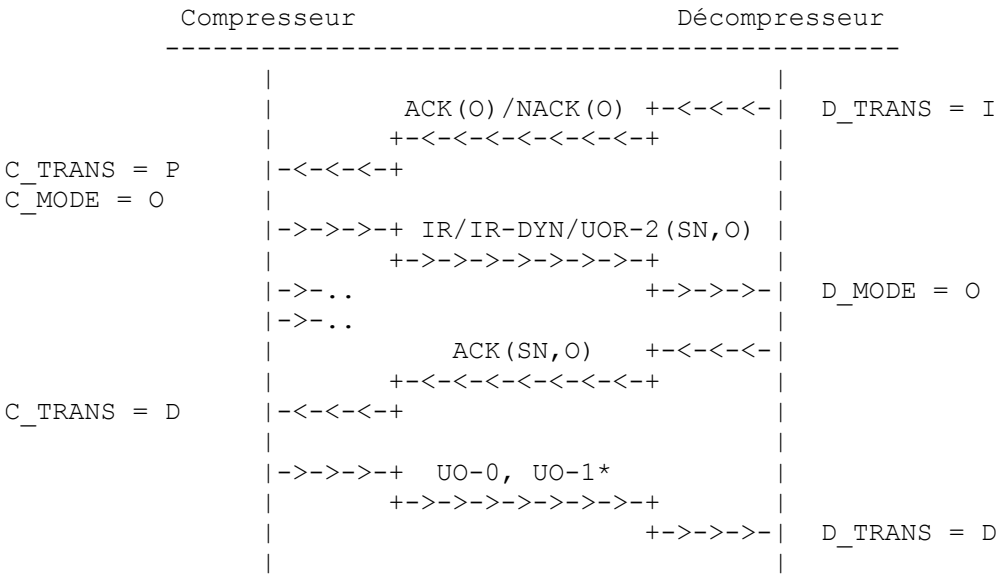
Tant que le décompresseur n'a pas reçu un paquet UOR-2, IR-DYN, ou IR avec le paramètre de transition de mode réglé à R, il doit rester en mode optimiste. Le compresseur ne doit pas envoyer de paquet des types 1 ou 0 tandis que C_TRANS est P, c'est-à-dire, pas jusqu'à ce qu'il ait reçu un ACK pour un paquet UOR-2, IR-DYN, ou IR envoyé avec le paramètre de transition de mode réglé à R. Lorsque le décompresseur reçoit un paquet des types 0 ou 1, après avoir accusé réception d'un paquet UOR-2, IR-DYN, ou IR, il règle D_TRANS à D.

5.6.4 Du mode unidirectionnel au mode fiable

La transition du mode unidirectionnel au mode fiable suit la même procédure de transition que définie au paragraphe 5.6.3 ci-dessus.

5.6.5 Du mode fiable au mode optimiste

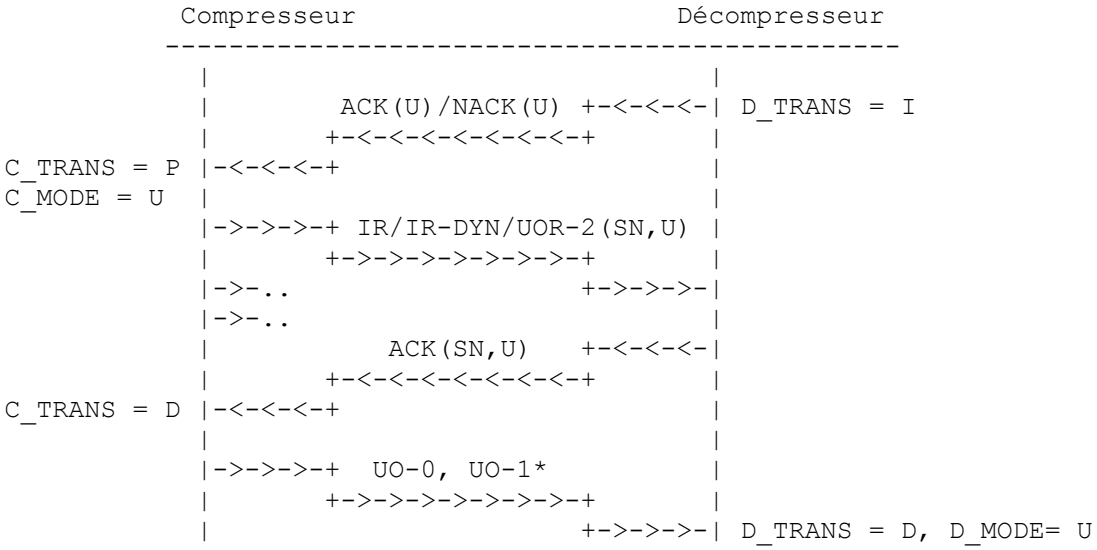
Le paquet de retour ACK(O) ou NACK(O) est utilisé pour initier la transition du mode fiable au mode optimiste et le compresseur DOIT toujours fonctionner dans l'état FO durant la transition. La procédure de transition est décrite ci-dessous :



Tant que le décompresseur n'a pas reçu un paquet UOR-2, IR-DYN, ou IR avec le paramètre de transition de mode réglé à O, il doit rester en mode fiable. Le compresseur ne doit pas envoyer de paquet des types 0 ou 1 tandis que C_TRANS est P, c'est-à-dire, tant qu'il n'a pas reçu un ACK pour un paquet UOR-2, IR-DYN, ou IR envoyé avec le paramètre de transition de mode réglé à O. Lorsque le décompresseur reçoit un paquet des types 0 ou 1, après avoir accusé réception du paquet UOR-2, IR-DYN, ou IR, il règle D_TRANS à D.

5.6.6 Transition au mode unidirectionnel

Le décompresseur peut forcer une transition de retour au mode unidirectionnel si il désire le faire. Sans considération du mode avec lequel cette transition commence, une prise de contact à trois phases DOIT être effectuée pour s'assurer d'une transition correcte du côté du compresseur. La procédure de transition est décrite ci-dessous :



Après avoir accusé réception du premier UOR-2(U), IR-DYN(U), ou IR(U), le décompresseur DOIT continuer d'envoyer des retours avec le paramètre Mode réglé à U jusqu'à ce qu'il reçoive un paquet des types 0 ou 1.

5.7 Formats de paquet

La notation suivante est utilisée dans ce paragraphe.

bits(X) = Nombre de bits pour le champ X présents dans l'en-tête compressé (y compris l'extension).

champ(X) = Valeur du champ X dans l'en-tête compressé.

contexte(X) = Valeur du champ X telle qu'établie dans le contexte.

valeur(X) = champ(X) si X est présent dans l'en-tête compressé; = contexte(X) autrement.

hdr(X) = Valeur du champ X dans l'en-tête non compressé ou décompressé.

Propriétés de mise à jour : Elles font la liste des champs dans le contexte qui sont directement mis à jour par le traitement de l'en-tête compressé. Noter qu'il peut y avoir des champs dépendants qui sont aussi implicitement mis à jour (par exemple, une mise à jour de contexte(SN) met souvent aussi à jour contexte(TS)). Voir aussi le paragraphe 5.2.7.

Les champs suivants surviennent dans plusieurs en-têtes et extensions :

SN : Numéro de séquence RTO compressé
Compressé avec W-LSB. Les intervalles d'interprétation, voir au paragraphe 4.5.1, sont définis comme suit :
p = 1 si bits(SN) ≤ 4
p = 2^(bits(SN)-5) – 1 si bits(SN) > 4

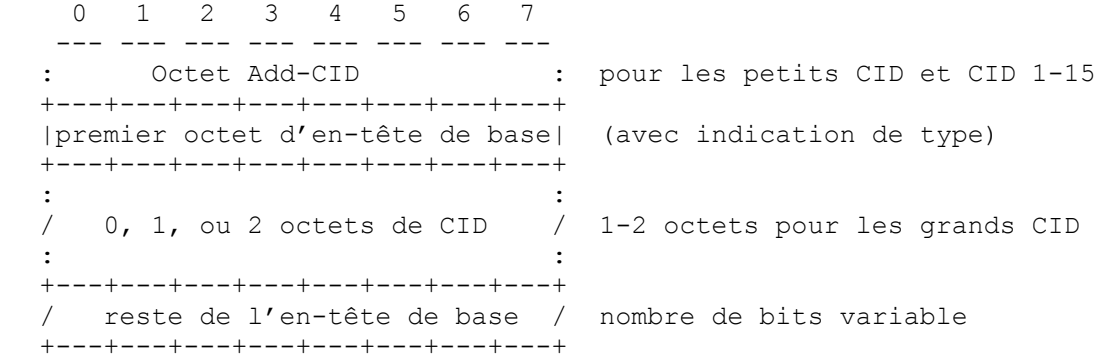
IP-ID : Champ IP-ID compressé.
Les champs IP-ID dans les en-têtes de base compressés portent l'identifiant IP compressé de l'en-tête IPv4 le plus interne dont le fanion RND correspondant n'est pas 1. Les règles ci-dessous supposent que l'IP-ID est pour l'en-tête IP le plus interne. Si il est pour un en-tête IP externe, les fanions RND2 et NBO2 sont utilisés à la place des RND et NBO.
Si valeur(RND) = 0, hdr(IP-ID) est compressé en utilisant le codage Décalage IP-ID (voir au paragraphe 4.5.5) en utilisant p = 0 et pente par défaut(décalage IP-ID) = 0.
Si valeur(RND) = 1, IP-ID est le hdr(IP-ID) non compressé. IP-ID est alors passé comme octet additionnel à la fin de l'en-tête compressé, après toutes les extensions.
Si valeur(NBO) = 0, les octets de hdr(IP-ID) sont échangés avant la compression et après la décompression. La valeur de NBO est ignorée lorsque valeur(RND) = 1.

TS : Valeur de l'horodatage RTP compressé.
Si valeur(TIME_STRIDE) > 0, on utilise la compression fondée sur la temporisation de l'horodatage RTP (voir au paragraphe 4.5.4).
Si valeur(Tsc) = 1, on utilise le codage d'horodatage RTP adapté avant la compression (voir au paragraphe 4.5.3), et pente par défaut(TS) = 1.
Si valeur(Tsc) = 0, la valeur de l'horodatage est compressée telle qu'elle, et pente par défaut(TS) = valeur(TS_STRIDE).
Les intervalles d'interprétation, voir au paragraphe 4.5.1, sont définis comme suit :
p = 2^(bits(TS)-2) - 1

CRC : CRC sur l'en-tête original, non compressé.
Pour les CRC de 3 bits, le polynôme du paragraphe 5.9.2 est utilisé.
Pour les CRC de 7 bits, le polynôme du paragraphe 5.9.2 est utilisé.
Pour les CRC de 8 bits, le polynôme du paragraphe 5.9.1 est utilisé.

M : Bit marqueur RTP.
Contexte(M) est initialement zéro et n'est jamais mis à jour. valeur(M) = 1 seulement quand champ(M) = 1.

Le format général pour un en-tête RTP compressé est comme suit :



```
:                               :
/     Extension (voir 5.7.5)    /  extension, si X = 1 dans l'en-tête de base
:                               :
 --- --- --- --- --- --- --- ---
:                               :
+  IP-ID d'en-tête IPv4 de base +  2 octets, si valeur(RND2) = 1
:                               :
 --- --- --- --- --- --- --- ---
/ Données AH pour liste externe /  variable (voir 5.8.4.2)
 --- --- --- --- --- --- --- ---
:                               :
+     Somme de contrôle GRE     +  2 octets, si fanion GRE C = 1 (voir 5.8.4.4)
:                               :
 --- --- --- --- --- --- --- ---
:                               :
+ IP-ID d'en-tête IPv4 interne  +  2 octets, si valeur(RND) = 1
:                               :
 --- --- --- --- --- --- --- ---
/ Données AH pour liste interne /  variable (voir 5.8.4.2)
 --- --- --- --- --- --- --- ---
:                               :
+     Somme de contrôle GRE     +  2 octets, si fanion GRE C = 1 (voir 5.8.4.4)
:                               :
 --- --- --- --- --- --- --- ---
:                               :
+      Somme de contrôle UDP    +  2 octets,
:                               :  si contexte(Somme de contrôle UDP) != 0
 --- --- --- --- --- --- --- ---
```

Noter que l'ordre des champs qui suivent l'extension facultative est le même que l'ordre entre les champs dans un en-tête non compressé.

Dans les paragraphes qui suivent, la position du grand CID dans les diagrammes est indiquée avec la notation suivante :

```
+===+===+===+===+===+===+===+===+
```

La présence/absence du champ Somme de contrôle UDP est contrôlée par la valeur de la somme de contrôle UDP dans le contexte. Si elle n'est pas zéro, la somme de contrôle est activée et envoyée avec chaque paquet. Si elle est de zéro, la somme de contrôle UDP est désactivée et n'est pas envoyée. Si hdr(Somme de contrôle UDP) n'est pas zéro lorsque contexte(Somme de contrôle UDP) est zéro, l'en-tête ne peut pas être compressé. Il doit être envoyé non compressé ou le contexte doit être réinitialisé en utilisant un paquet IR. Contexte(Somme de contrôle UDP) n'est mis à jour que par les en-têtes IR ou IR-DYN, jamais par les sommes de contrôle UDP envoyées dans les en-têtes de type 2, 1, ou 0.

Lorsque un en-tête IPv4 est présent dans le contexte statique, pour lequel le fanion RND correspondant n' pas été établi à 1, les paquets des types R-1 et UO-1 NE DOIVENT PAS être utilisés.

Lorsque aucun en-tête IPv4 n'est présent dans le contexte statique, ou si les fanions RND pour tous les en-têtes IPv4 dans le contexte ont été établis à 1, les paquets des types R-1-ID, R-1-TS, UO-1-ID, et UO-1-TS NE DOIVENT PAS être utilisés.

Dans l'état transitoire dans lequel un fanion RND est établi, les paquets des types R-1-ID, R-1-TS, UO-1-ID, et UO-1-TS NE DOIVENT PAS être utilisés. Cela implique que le ou les fanions RND de l'extension 3 peuvent devoir être inspectés avant que le format d'un en-tête de base portant une extension 3 puisse être déterminé.

5.7.1 Paquet de type 0 : UO-0, R-0, R-0-CRC

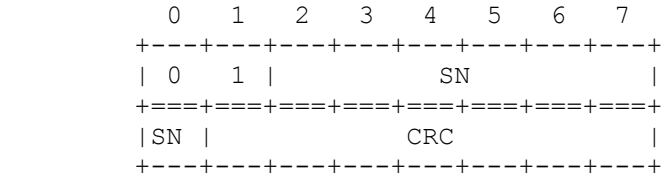
Un paquet de type 0 est indiqué par son premier bit à 0 :

R-0

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| 0   0 |          SN           |
+===+===+===+===+===+===+===+===+
```

Propriétés de mise à jour : Les paquets R-0 ne mettent à jour aucune partie du contexte.

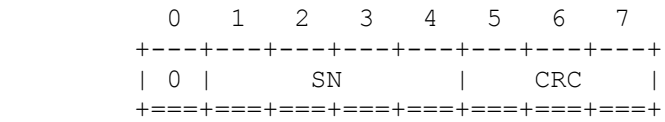
R-0-CRC



Note : Le champ SN enjambe le champ CID.

Propriétés de mise à jour : Les paquets R-0-CRC mettent à jour contexte(Numéro de séquence RTP).

UO-0

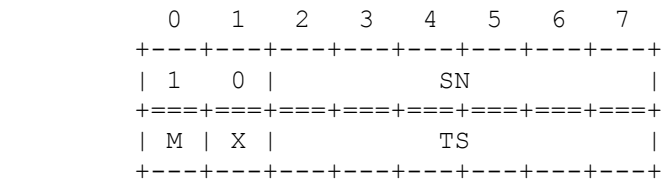


Propriétés de mise à jour : Les paquets UO-0 mettent à jour la valeur en cours de contexte(Numéro de séquence RTP).

5.7.2 Paquet de type 1 (mode R) : R-1, R-1-TS, R-1-ID

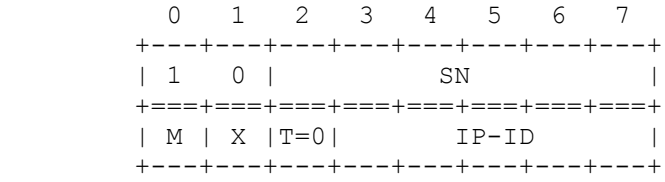
Un paquet de type 1 est indiqué par ses premiers bits qui font 10 :

R-1



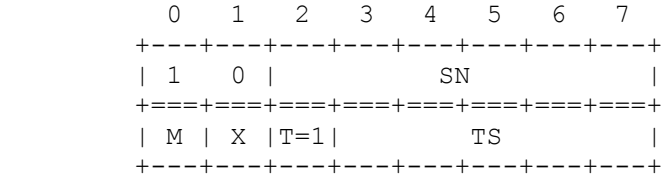
Note : R-1 ne peut pas être utilisé si le contexte contient au moins un en-tête IPv4 avec valeur(RND) = 0. Ceci le différencie sans ambiguïté de R-1-ID et R-1-TS.

R-1-ID



Note : R-1-ID ne peut pas être utilisé si il n'y a pas d'en-tête IPv4 dans le contexte ou si valeur(RND) et valeur(RND2) sont toutes deux à 1.

R-1-TS



Note : R-1-TS ne peut pas être utilisé si il n'y a pas d'en-tête IPv4 dans le contexte ou si valeur(RND) et valeur(RND2) sont toutes deux à 1.

X : X = 0 indique qu'aucune extension n'est présente ;
X = 1 indique qu'une extension est présente.

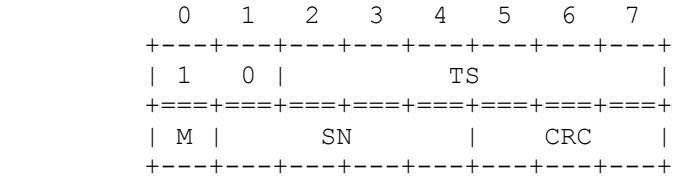
T : T = 0 indique le format R-1-ID ;

T = 1 indique le format R-1-TS.

Propriétés de mise à jour : Les en-têtes R-1* ne mettent à jour aucune partie du contexte.

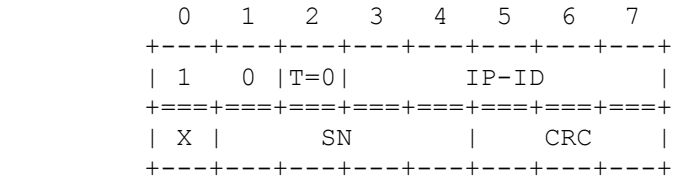
5.7.3 Paquet de type 1 (mode U/O)

UO-1, UO-1-ID, UO-1-TS UO-1



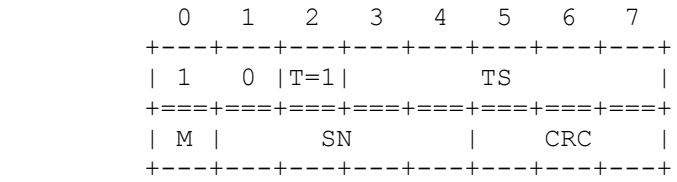
Note : UO-1 ne peut pas être utilisé si le contexte contient au moins un en-tête IPv4 avec valeur(RND) = 0. Cela empêche de le confondre avec UO- 1-ID et UO-1-TS.

UO-1-ID



Note : UO-1-ID ne peut pas être utilisé si il n'y a pas d'en-tête IPv4 dans le contexte ou si valeur(RND) et valeur(RND2) sont toutes deux à 1.

UO-1-TS



Note: UO-1-TS ne peut pas être utilisé si il n'y a pas d'en-tête IPv4 dans le contexte ou si valeur(RND) et valeur(RND2) sont toutes deux à 1.

X : X = 0 indique qu'aucune extension n'est présente ;
X = 1 indique qu'une extension est présente.

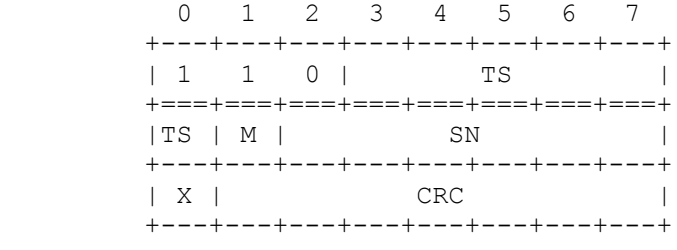
T : T = 0 indique le format UO-1-ID ;
T = 1 indique le format UO-1-TS.

Propriétés de mise à jour : Les paquets UO-1* mettent à jour contexte(Numéro de séquence RTP). Les paquets UO-1 et UO-1-TS mettent à jour contexte(Horodatage RTP). Les paquets UO-1-ID mettent à jour contexte(IP-ID). Les valeurs fournies dans les extensions, sauf celles dans les autres champs SN, TS, ou IP-ID, ne mettent pas à jour le contexte.

5.7.4 Paquet de type 2 : UOR-2

Le paquet de type 2 est indiqué par les premiers bits à 110 :

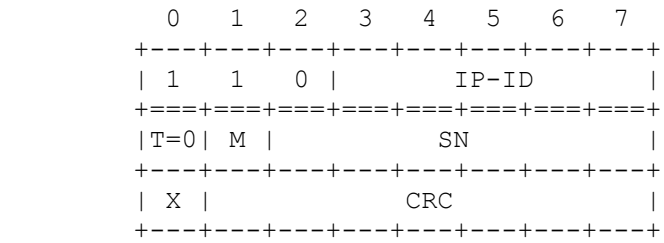
UOR-2



Note : UOR-2 ne peut pas être utilisé si le contexte contient au moins un en-tête IPv4 avec valeur(RND) = 0. Cela empêche de le confondre avec UOR-2-ID et UOR-2-TS.

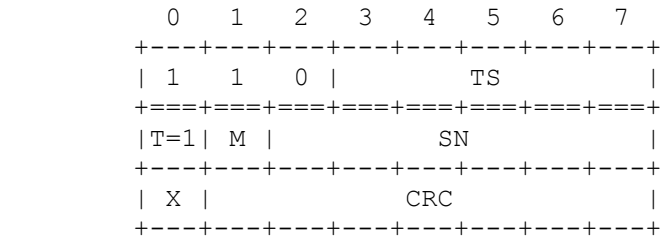
Note : Le champ TS enjambe le champ CID.

UOR-2-ID



Note : UOR-2-ID ne peut pas être utilisé si il n'y a pas d'en-tête IPv4 dans le contexte ou si valeur(RND) et valeur(RND2) sont toutes deux à 1.

UOR-2-TS



Note: UOR-2-TS ne peut pas être utilisé si il n'y a pas d'en-tête IPv4 dans le contexte ou si valeur(RND) et valeur(RND2) sont toutes deux à 1.

X : X = 0 indique qu'aucune extension n'est présente ;
X = 1 indique qu'une extension est présente.

T : T = 0 indique le format UOR-2-ID;
T = 1 indique le format UOR-2-TS.

Propriétés de mise à jour : Toutes les valeurs fournies dans les paquets UOR-2* mettent à jour le contexte, sauf mention contraire explicite.

5.7.5 Formats d'extension

Note : Le terme 'extension' tel qu'utilisé pour les informations supplémentaires contenues dans les en-têtes ROHC n'a aucune relation avec le terme 'en-tête d'extension' utilisé dans IP.

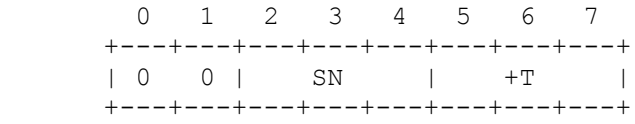
Les champs dans les extensions sont enchaînés avec le champ correspondant dans l'en-tête de base compressé, si il y en a un. Les bits dans une extension sont de moindre poids que les bits dans l'en-tête de base compressé (voir au paragraphe 4.5.7).

Le champ TS est adapté dans toutes les extensions, car il est dans l'en-tête de base, sauf facultativement lorsque on utilise l'extension 3 où le fanion Tsc peut indiquer que le champ TS n'est pas adapté. Valeur(TS_STRIDE) est utilisé comme facteur d'adaptation lors de l'adaptation du champ TS.

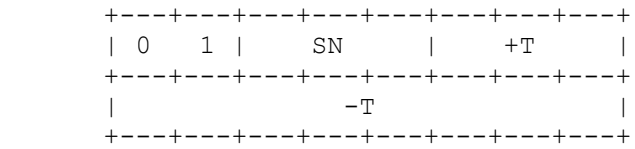
Dans les trois extensions suivantes, l'interprétation des champs dépend de si il y a un bit T dans l'en-tête de base compressé, et si il y en a un, de la valeur de ce champ. Lorsque il n'y a pas de bit T, +T et -T signifient tous deux TS. C'est le cas lorsque il n'y a pas d'en-tête IPv4 dans le contexte statique, et lorsque tous les en-têtes IPv4 dans le contexte statique ont leur fanion RND correspondant établi (c'est-à-dire, RND = 1).

Si il y a un bit T,
T = 1 indique que +T est TS, et -T est IP-ID ;
T = 0 indique que +T est IP-ID, et -T est TS.

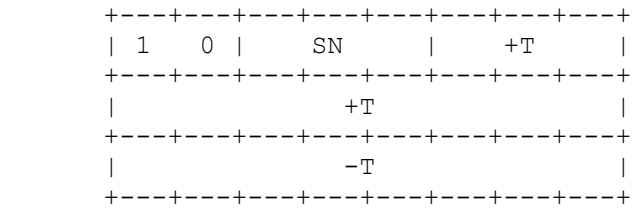
Extension 0 :



Extension 1 :

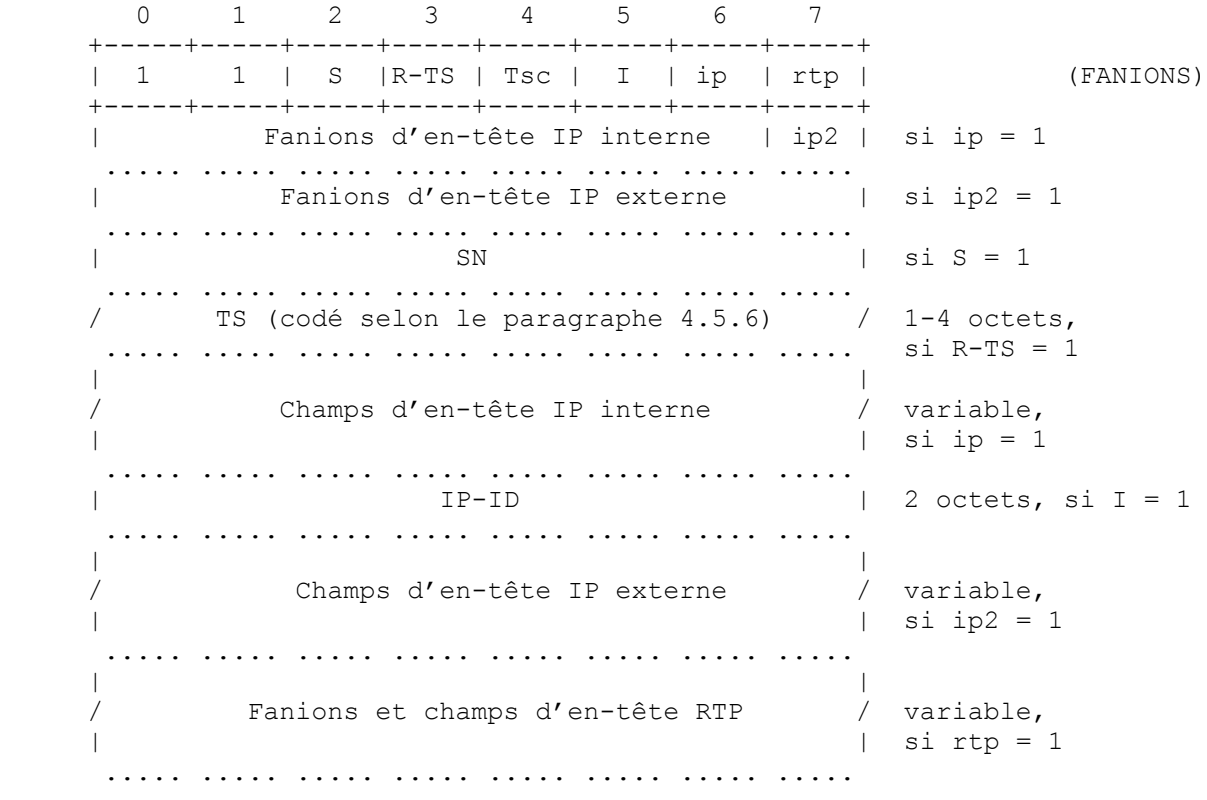


Extension 2 :



Extension 3 est une extension plus élaborée qui peut donner des valeurs pour des champs autres que SN, TS, et IP-ID. Trois octets de fanion facultatifs indiquent les changements, respectivement, aux en-têtes IP et aux en-tête RTP.

Extension 3 :



S, R-TS, I, ip, rtp, ip2 : Indiquent la présence des champs comme indiqué à droite de chaque champ ci-dessus.

Tsc : Tsc = 0 indique que TS n'est pas adapté ;
Tsc = 1 indique que TS est adapté selon le paragraphe 4.5.3, en utilisant valeur(TS_STRIDE).
Contexte(Tsc) est toujours 1. Si l'adaptation n'est pas désirée, le compresseur établira TS_STRIDE = 1.

SN : Voir le début du paragraphe 5.7.

TS : Nombre variable de bits de TS, codé selon le paragraphe 4.5.6. Voir le début du paragraphe 5.7.

IP-ID : Voir le début du paragraphe 5.7.

Fanions d'en-tête IP interne
Ils correspondent à l'en-tête IP interne si ils sont deux, et à l'en-tête IP seul autrement.

```
      0     1     2     3     4     5     6     7
    ..... ..... ..... ..... ..... ..... ..... .....
   | TOS | TTL | DF  | PR  | IPX | NBO | RND | ip2 |  si ip = 1
    ..... ..... ..... ..... ..... ..... ..... .....
```

TOS, TTL, PR, IPX : Indiquent la présence des champs comme montré à droite des champs en question ci-dessous.

DF : Bit Ne pas fragmenter de l'en-tête IP.

NBO : Indique si les octets de hdr(identifiant IP) de cet en-tête IP sont échangés avant compression et après décompression.

NBO = 1 indique que les octets n'ont pas besoin d'être échangés. NBO = 0 indique que les octets sont à échanger. Voir au paragraphe 4.5.5.

RND : Indique si hdr(identifiant IP) n'est pas à compresser mais plutôt à envoyer tel quel dans les en-têtes compressés.

IP2 : Indique la présence de champs d'en-tête IP interne. Sauf si le contexte statique contient deux en-têtes IP, IP2 est toujours zéro.

Champs d'en-tête IP interne

```
    ..... ..... ..... ..... ..... ..... ..... .....
   |         Type de service/Classe de trafic      |  si TOS = 1
    ..... ..... ..... ..... ..... ..... ..... .....
   |         Durée de vie/Limite de bonds           |  si TTL = 1
    ..... ..... ..... ..... ..... ..... ..... .....
   |         Protocole/Prochain en-tête             |  si PR = 1
    ..... ..... ..... ..... ..... ..... ..... .....
   /         En-têtes d'extension IP               /  variable,
    ..... ..... ..... ..... ..... ..... ..... .....   si IPX = 1
```

Type de service/Classe de trafic : Ce champ dans l'en-tête IP non compressé (valeur absolue).
Durée de vie/Limite de bonds : Ce champ dans l'en-tête IP non compressé.
Protocole/Prochain en-tête : Ce champ dans l'en-tête IP non compressé.
En-tête(s) d'extension IP : Selon le paragraphe 5.8.5.

Fanions d'en-tête IP externe
Les champs dans cette partie de l'en-tête Extension 3 se réfèrent à l'en-tête IP le plus externe :

```
        0     1     2     3     4     5     6     7
      ..... ..... ..... ..... ..... ..... ..... .....
     | TOS2| TTL2| DF2 | PR2 |IPX2 |NBO2 |RND2 |  I2 |  si ip2 = 1
      ..... ..... ..... ..... ..... ..... ..... .....
```

Ces fanions sont les mêmes que les fanions d'en-tête IP interne, mais se réfèrent à l'en-tête IP externe au lieu de l'en-tête IP interne. Les fanions suivant n'ont cependant pas de contrepartie dans les fanions d'en-tête IP interne :
I2 : Indique la présence du champ IP-ID.

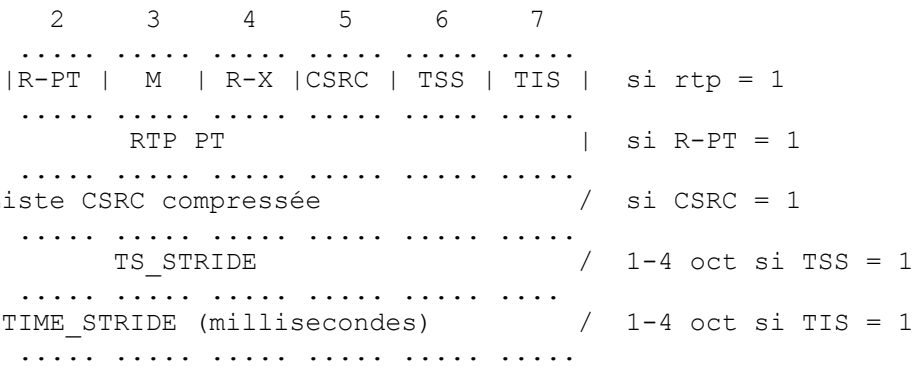
Champs d'en-tête IP externe

```
      ..... ..... ..... ..... ..... ..... ..... .....
     |      Type de service/Classe de trafic         |  si TOS2 = 1
      ..... ..... ..... ..... ..... ..... ..... .....
     |         Durée de vie/Limite de bonds          |  si TTL2 = 1
      ..... ..... ..... ..... ..... ..... ..... .....
     |         Protocole/Prochain en-tête            |  si PR2 = 1
      ..... ..... ..... ..... ..... ..... ..... .....
     /         En-têtes d'extension IP              /  variable,
      ..... ..... ..... ..... ..... ..... ..... .....    si IPX2 = 1
     |                 IP-ID                         |  2 octets,
      ..... ..... ..... ..... ..... ..... ..... .....   si I2 = 1
```

Les champs dans cette partie de Extension 3 sont comme pour les champs d'en-tête IP interne, mais ils se réfèrent à l'en-tête IP externe au lieu de l'en-tête IP interne. Les champs suivants n'ont cependant pas de contrepartie parmi les champs d'en-tête IP interne :

IP-ID : Champ Identifiant IP de l'en-tête IP externe, sauf si l'en-tête interne est un en-tête IPv6, auquel cas I2 est toujours zéro.

Fanions et champs d'en-tête RTP



Mode : Mode de compression. 0 = Réservé,
1 = Unidirectionnel,
2 = Bidirectionnel optimiste,
3 = Bidirectionnel fiable.

R-PT, CSRC, TSS, TIS : Indique la présence des champs comme indiqué à droite de chacun des champs ci-dessus.

R-P : Bit de bourrage RTP, valeur absolue (présumée zéro si absent).

R-X : Bit eXtension RTP, valeur absolue.

M : Voir le début du paragraphe 5.7.

RTP PT : Valeur absolue du champ Type de charge utile RTP.

Liste de CSRC compressée : Voir au paragraphe 5.8.1.

TS_STRIDE : Incrément/décrément prévu du champ Horodatage RTP lorsque il change. Codé selon le paragraphe 4.5.6.

TIME_STRIDE : Intervalle de temps prédit en millisecondes entre les changements dans l'horodatage RTP. Aussi indication que le compresseur désire effectuer une compression fondée sur la temporisation du champ Horodatage RTP : voir au paragraphe 4.5.4. Codé selon le paragraphe 4.5.6.

5.7.5.1 Fanions RND et types de paquet

Les valeurs des fanions RND et RND2 sont changées par l'envoi des en-têtes UOR-2 avec Extension 3, ou des en-têtes IR-DYN, où le ou les fanions ont leurs nouvelles valeurs. La procédure d'établissement des fanions est celle normale pour le mode en cours, c'est-à-dire qu'en mode U et O, les valeurs sont répétées plusieurs fois pour s'assurer que le décompresseur en reçoive au moins une. En mode R, les fanions sont envoyés jusqu'à ce qu'un accusé de réception soit reçu pour un paquet avec les nouvelles valeurs du fanion RND.

Le décompresseur met à jour les valeurs de ses fanions RND et RND2 chaque fois qu'il reçoit un UOR-2 avec Extension 3 portant les valeurs pour RND ou RND2, et le CRC UOR-2 vérifie la réussite de la décompression.

Lorsque un en-tête IPv4 pour lequel le fanion RND correspondant n'a pas été établi à 1 est présent dans le contexte statique, les paquets des types R-1 et UO-1 NE DOIVENT PAS être utilisés.

Lorsque aucun en-tête IPv4 n'est présent dans le contexte statique, ou lorsque les fanions RND pour tous les en-têtes IPv4 dans le contexte ont été établis à 1, les paquets des types R-1-ID, R-1-TS, UO-1-ID, et UO-1-TS NE DOIVENT PAS être utilisés.

Dans l'état transitoire dans lequel un fanion RND est établi, les paquets des types R-1-ID, R-1-TS, UO-1-ID, et UO-1-TS

NE DOIVENT PAS être utilisés. Cela implique que le ou les fanions RND de Extension 3 pourraient devoir être inspectés avant que puisse être déterminé le format exact d'un en-tête de base portant une Extension 3, c'est-à-dire, selon qu'un bit T est présent ou non.

5.7.5.2 Fanions/Champs dans le contexte

Certains fanions et champs dans Extension 3 ont besoin d'être maintenus dans le contexte du décompresseur. Leurs valeurs sont établies en utilisant le mécanisme approprié au mode de compression, sauf indication contraire dans le tableau ci-dessous et dans les paragraphes de référence.

Fanion/Champ	Valeur initiale	Commentaire
Mode	Unidirectionnel	Voir le paragraphe 5.6
NBO	1	Voir le paragraphe 4.5.5
RND	0	Voir les paragraphes 4.5.5, 5.7.5.1
NBO2	1	Comme NBO, mais pour en-tête externe
RND2	0	Comme RND, mais pour en-tête externe
TS_STRIDE	1	Voir le paragraphe 4.5.3
TIME_STRIDE	0	Voir le paragraphe 4.5.4
Tsc	1	Tsc est toujours 1 dans le contexte ; peut être 0 seulement quand une Extension 3 est présente. Voir la discussion du champ TS au début du paragraphe 5.7.

5.7.6 Paquets et formats de retour

Quand le délai d'aller-retour entre compresseur et décompresseur est grand, plusieurs paquets peuvent être en cours concurremment. Donc, plusieurs paquets peuvent être reçus par le décompresseur après qu'un retour a été envoyé et avant que le compresseur ait réagi au retour. De plus, la décompression peut échouer à cause d'erreurs résiduelles dans l'en-tête compressé.

Donc,

a) en mode O, le décompresseur DEVRAIT limiter le taux d'envoi des retours (s'il en est envoyé) sur décompression réussie ;
b) lorsque la décompression échoue, le retour ne DEVRAIT être envoyé que lorsque la décompression de plusieurs paquets consécutifs a échoué, et lorsque cela arrive, le taux de retours DEVRAIT être limité ;
c) lorsque sont reçus des paquets qui appartiennent à un flux de paquets rejeté, le taux de retours DEVRAIT être limité.

Un décompresseur PEUT limiter le taux de retours en n'envoyant de retour que pour un tous les k paquets provoquant le même retour (la même sorte de retour). La valeur appropriée de k dépend de la mise en œuvre ; k peut être choisi de telle sorte que le retour soit envoyé 1 à 3 fois par délai d'aller-retour de la liaison.

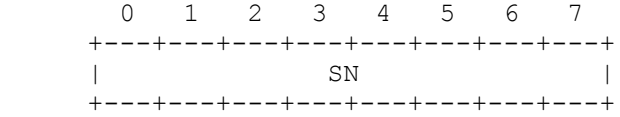
Voir au paragraphe 5.2.2 l'exposé concernant les façons de fournir des informations en retour au compresseur.

5.7.6.1 Formats des retours pour RTP ROHC

Ce paragraphe décrit le format des informations de retour dans RTP ROHC. Voir aussi 5.2.2.

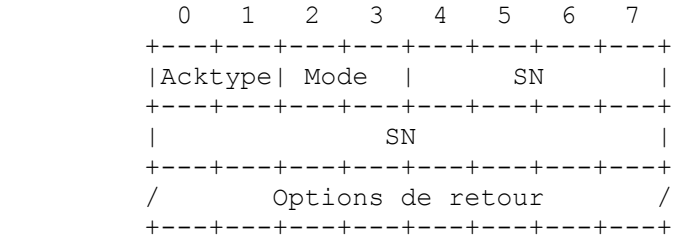
Plusieurs formats de retour portent un champ marqué SN. Le champ SN contient les LSB d'un numéro de séquence RTP. Le numéro de séquence à utiliser est le numéro de séquence de l'en-tête qui a causé l'envoi des informations de retour. Si ce numéro de séquence ne peut pas être déterminé, par exemple quand la décompression échoue, le numéro de séquence à utiliser est celui du dernier en-tête décompressé avec succès. Si aucun numéro de séquence n'est disponible, le retour DOIT porter une option SN-NOT-VALID. À réception, le compresseur confronte les LSB SN valides avec l'en-tête envoyé le plus récemment avec un SN dont les LSB correspondent. Le décompresseur doit s'assurer qu'il envoie assez de LSB de SN dans son retour pour que cette corrélation ne soit pas ambiguë; par exemple, si un champ de LSB de SN de 8 bits peut revenir à zéro dans un délai d'aller-retour, le format FEEDBACK-1 ne peut pas être utilisé.

FEEDBACK-1



Un FEEDBACK-1 est un ACK. Pour envoyer un NACK ou un STATIC-NACK, FEEDBACK-2 doit être utilisé. FEEDBACK-1 ne contient aucune information de mode ; FEEDBACK-2 doit être utilisé quand des informations de mode sont exigées.

FEEDBACK-2



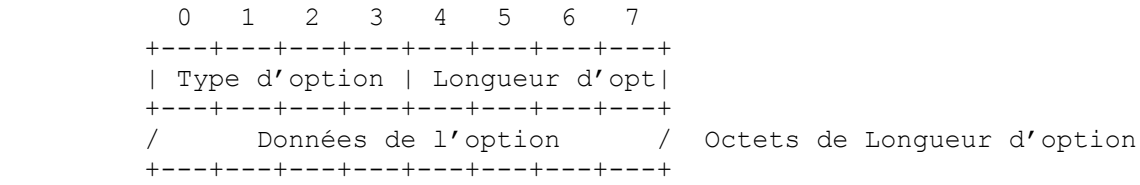
Acktype : 0 = ACK
1 = NACK
2 = STATIC-NACK
3 est réservé (NE DOIT PAS être utilisé pour l'analyse)

Mode : 0 est réservé
1 = Mode Unidirectionnel
2 = Mode Bidirectionnel optimiste
3 = Mode Bidirectionnel fiable

Options de retour : Nombre variable d'options de retour, voir au paragraphe 5.7.6.2. Les options peuvent apparaître dans n'importe quel ordre.

5.7.6.2 Options de retour RTP ROHC

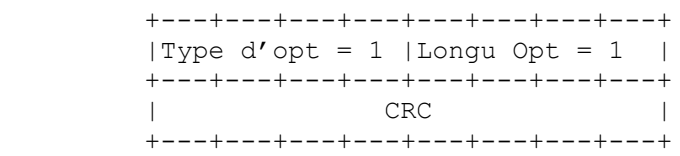
Une option de retour RTP ROHC a une longueur variable et le format général suivant :



Les paragraphe 5.7.6.3 à 5.7.6.9 décrivent les options de retour RTP ROHC actuellement définies.

5.7.6.3 Option CRC

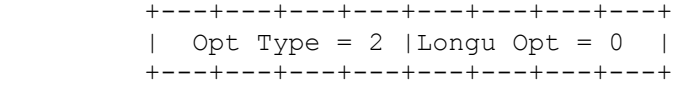
L'option CRC contient un CRC de 8 bits calculé sur la charge utile entière de retour, sans le type de paquet et l'octet de code, mais incluant tous champs de CID, en utilisant le polynôme du paragraphe 5.9.1. Si le CID est donné avec un octet Add-CID, l'octet Add-CID précède immédiatement le format FEEDBACK-1 ou FEEDBACK-2. Pour les besoins du calcul du CRC, les champs CRC de toutes les options de CRC sont à zéro.



En recevant des informations de retour avec une option CRC, le compresseur DOIT vérifier les informations en calculant le CRC et en comparant le résultat au CRC porté dans l'option CRC. Si les deux ne sont pas identiques, les informations de retour DOIVENT être ignorées.

5.7.6.4 Option REJECT

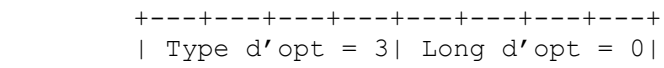
L'option REJECT informe le compresseur que le décompresseur n'a pas de ressources suffisantes pour traiter le flux.



Lorsque il reçoit un option REJECT, le compresseur arrête de compresser le flux de paquets, et devrait s'abstenir de tenter d'augmenter le nombre de flux de paquets compressés pendant un certain temps. Tout paquet FEEDBACK portant une option REJECT DOIT aussi porter une option CRC.

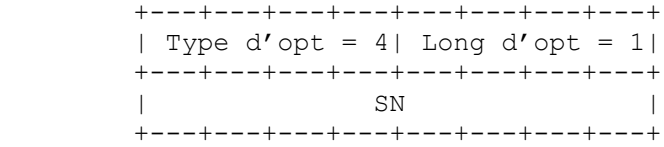
5.7.6.5 Option SN-NOT-VALID

L'option SN-NOT-VALID indique que le SN du retour n'est pas valide. Un compresseur NE DOIT PAS utiliser le SN du retour pour trouver l'en-tête envoyé correspondant lorsque cette option est présente.



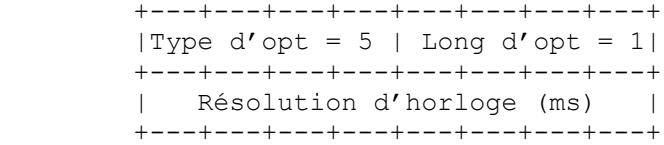
5.7.6.6 Option SN

L'option SN donne 8 bits supplémentaires de SN.



5.7.6.7 Option CLOCK

L'option CLOCK informe le compresseur de la résolution d'horloge du décompresseur. Cela est nécessaire pour permettre au compresseur d'estimer la gigue introduite par l'horloge du décompresseur lorsque il fait la compression fondée sur la temporisation de l'horodatage RTP.



La plus petite résolution d'horloge qui puisse être indiquée est 1 milliseconde. La valeur zéro a une signification particulière : elle indique que le décompresseur ne peut pas faire la compression fondée sur la temporisation de l'horodatage RTP. Tout paquet FEEDBACK portant une option CLOCK DEVRAIT aussi porter une option CRC.

5.7.6.8 Option JITTER

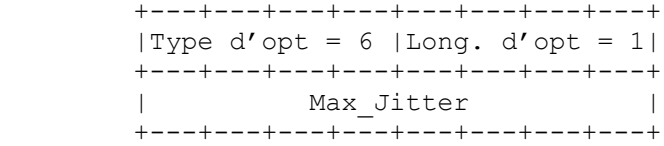
L'option JITTER permet au décompresseur de faire rapport de la gigue maximum qu'il a observé dernièrement, en utilisant la formule suivante qui est très semblable à la formule pour Max_Jitter_BC au paragraphe 4.5.4.

Soit une fenêtre d'observation i qui contient la meilleure approximation du décompresseur de la fenêtre glissante du compresseur (voir au paragraphe 4.5.4) quand l'en-tête i est reçu.

Max_Jitter_i = max {|(T_i - T_j) - ((a_i - a_j) / TIME_STRIDE)|, pour tout en-tête j dans la fenêtre d'observation i}

Max_Jitter = max { Max_Jitter_i, pour un grand nombre d'en-têtes i récents }

Ces informations peuvent être utilisées par le compresseur pour préciser la formule pour déterminer k lors d'une compression fondée sur la temporisation de l'horodatage RTP.



Le décompresseur PEUT ignorer les valeurs observées les plus anciennes de Max_Jitter_i. Donc, le Max_Jitter rapporté peut décroître. La robustesse sera réduite si le compresseur utilise une estimation trop faible de la gigue. Donc, un paquet FEEDBACK portant une option JITTER DEVRAIT aussi porter une option CRC. De plus, le compresseur PEUT ignorer les valeurs décroissantes de Max_Jitter.

5.7.6.9 Option LOSS

L'option LOSS permet au décompresseur de faire rapport du plus grand nombre observé de paquets perdus à la suite. Cette information PEUT être utilisée par le compresseur pour ajuster la taille de la fenêtre de référence utilisée en modes U et O.

```
+---+---+---+---+---+---+---+---+
|Type d'opt = 7 |Long. d'opt = 1|
+---+---+---+---+---+---+---+---+
|Plus long évmt de perte (pqts) |
+---+---+---+---+---+---+---+---+
```

Le décompresseur PEUT choisir d'ignorer les plus anciens événements de perte. Donc, la valeur rapportée peut décroître. Comme un réglage trop faible de la fenêtre de référence peut réduire la robustesse, un paquet FEEDBACK portant une option LOSS DEVRAIT aussi porter une option CRC. Le compresseur PEUT choisir d'ignorer les valeurs de perte décroissantes.

5.7.6.10 Types d'option inconnus

Si un type d'option inconnu du compresseur est rencontré, il doit continuer d'analyser le reste du paquet FEEDBACK, ce qui est possible car la longueur de l'option est explicite, mais DOIT autrement ignorer l'option inconnue.

5.7.6.11 Exemple de retour RTP

Le retour pour le CID 8 qui indique un ACK pour SN 17 et le mode bidirectionnel fiable peut avoir le format suivant.
En supposant de petits CID :

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| 1   1   1   1   0 | 0   1   1 |  Type de paquet de retour, Code = 3
+---+---+---+---+---+---+---+---+
| 1   1   1   0 | 1   0   0   0 |  Octet Add-CID avec CID = 8
+---+---+---+---+---+---+---+---+
| 0   0 | 1   1 |  SN MSB = 0   |  AckType = ACK, Mode = Fiable
+---+---+---+---+---+---+---+---+
|          SN LSB = 17          |
+---+---+---+---+---+---+---+---+
```

Les second, troisième, et quatrième octets sont passés au compresseur.

Le format FEEDBACK-1 peut aussi être utilisé. En supposant de grands CID :

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| 1   1   1   1   0 | 0   1   0 |  Type de paquet de retour, Code = 3
+---+---+---+---+---+---+---+---+
| 0   0   0   0   1   0   0   0 |  Grand CID avec valeur 8
+---+---+---+---+---+---+---+---+
|          LSB SN = 17          |
+---+---+---+---+---+---+---+---+Les second et troisième octets sont passés au compresseur.
```

En supposant des CID petits :

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| 1   1   1   1   0 | 0   1   0 |  Type de paquet de retour, Code = 2
+---+---+---+---+---+---+---+---+
| 1   1   1   0 | 1   0   0   0 |  Octet Add-CID avec CID = 8
+---+---+---+---+---+---+---+---+
|          LSB SN = 17          |
+---+---+---+---+---+---+---+---+
```

Les second et troisième octets sont passés au compresseur.

En supposant des CID petits et CID 0 à la place de CID 8 :

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| 1   1   1   1   0 | 0   0   1 |  Type de paquet de retour, Code = 1
+---+---+---+---+---+---+---+---+
|          LSB SN = 17          |
+---+---+---+---+---+---+---+---+
```

Le second octet est passé au compresseur.

5.7.7 Paquets RTP IR et IR-DYN

Les sous en-têtes qui sont compressibles sont composés d'une partie STATIQUE et d'une partie DYNAMIQUE. Ces

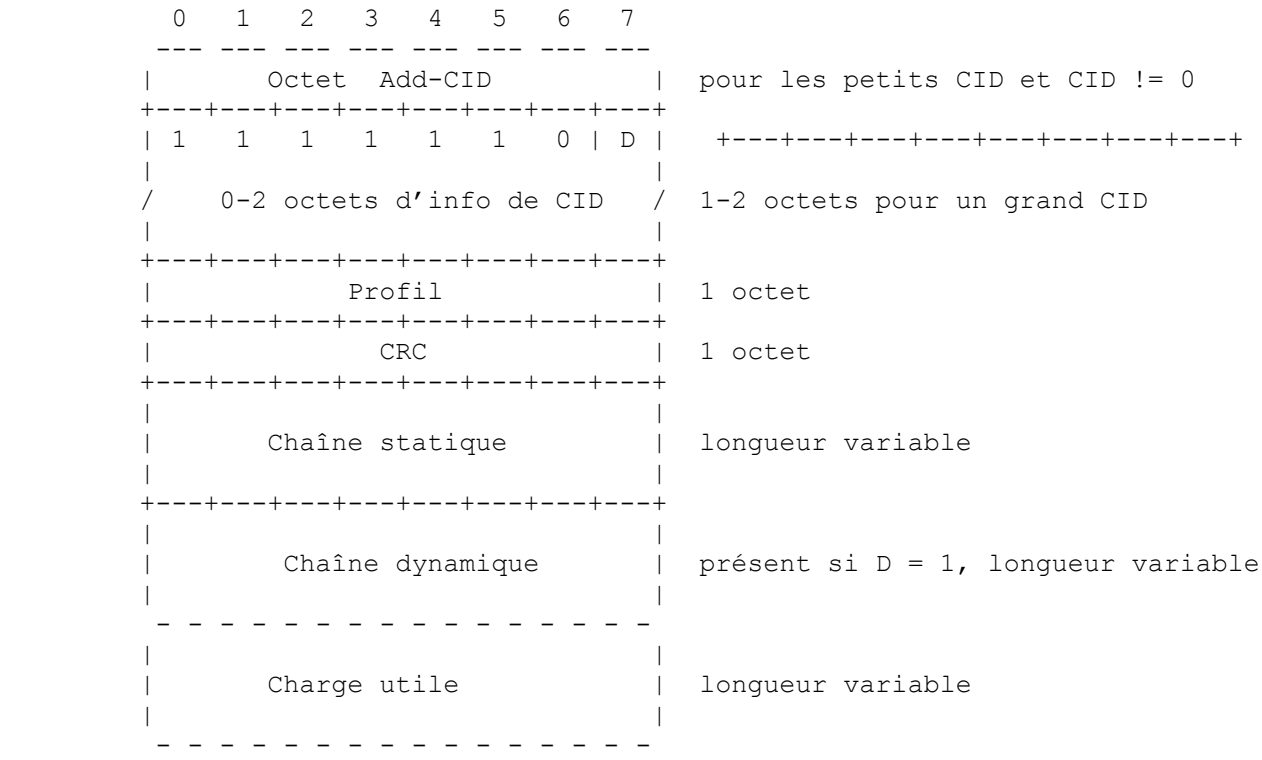
parties sont définies aux paragraphes 5.7.7.3 à 5.7.7.7.

La structure d'une chaîne de sous en-tête est déterminée par chaque en-tête qui a un champ Prochain en-tête, ou Protocole. Ce champ identifie le type de l'en-tête suivant. Chaque partie Statique ci-dessous qui est suivie par une autre partie Statique contient le champ Prochain en-tête/Protocole et permet l'analyse de la chaîne statique ; la chaîne dynamique, si elle est présente, est structurée de la même façon.

Les paquets IR et IR-DYN vont être cause qu'un paquet soit livré aux couches supérieures si et seulement si la charge utile n'est pas vide. Cela signifie qu'un paquet IP/UDP/RTP où la longueur UDP indique une charge utile UDP d'une taille de 12 octets ne peut pas être représentée par un paquet IR ou IR-DYN. De tels paquets peuvent plutôt être représentés en utilisant le profil NON COMPRESSÉ (paragraphe 5.10).

5.7.7.1 Structure de base du paquet IR

Ce type de paquet communique la partie statique du contexte, c'est-à-dire, les valeurs des fonctions de SN constantes. Il peut facultativement communiquer aussi la partie dynamique du contexte, c'est-à-dire, les paramètres des fonctions de SN non constantes. Il peut aussi communiquer facultativement la charge utile d'un paquet original, si il en est une.



D : D = 1 indique la présence de la chaîne dynamique.

Profil : Identifiant de profil, abrégé comme défini au paragraphe 5.2.3.

CRC : CRC de 8 bits, calculé selon le paragraphe 5.9.1.

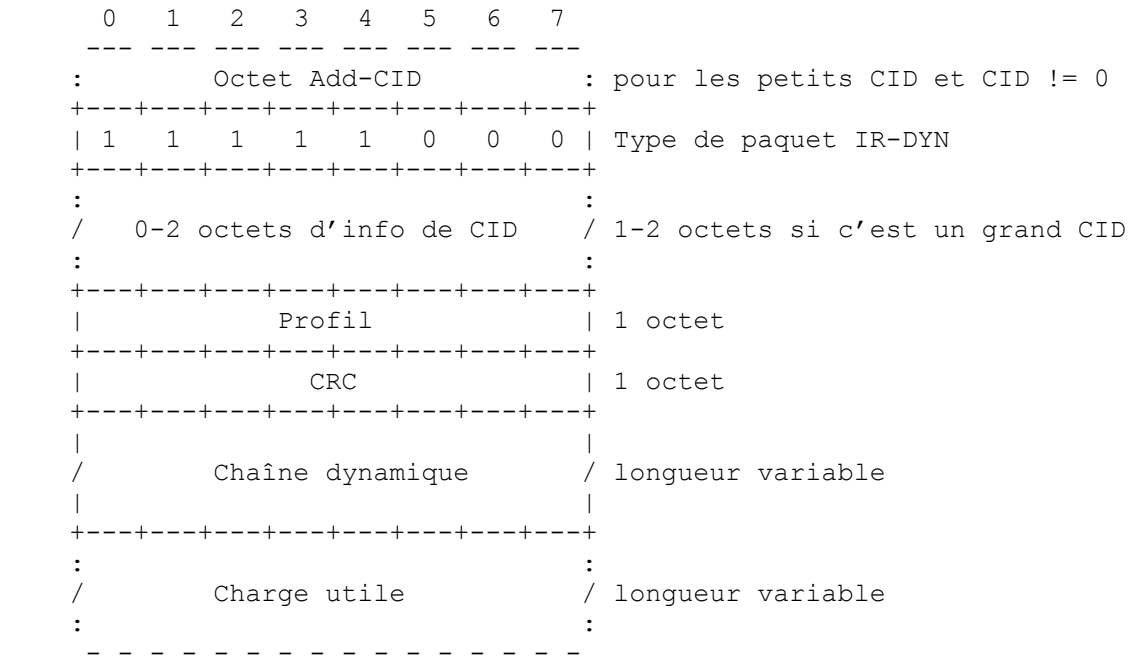
Chaîne statique : Chaîne des informations de sous en-tête statique.

Chaîne dynamique : Chaîne des informations de sous en-tête dynamique. Quelles informations dynamiques sont présentes est inféré par la chaîne statique.

Charge utile : C'est la charge utile du paquet original correspondant, s'il en est. La présence d'une charge utile est inférée par la Longueur du paquet.

5.7.7.2 Structure de base du paquet IR-DYN

Ce type de paquet communique la partie dynamique du contexte, c'est-à-dire, les paramètres des fonctions non constantes du SN.



Profil : Identifiant de profil, abrégé comme défini au paragraphe 5.2.3.

CRC : CRC de 8 bits, calculé selon le paragraphe 5.9.1.

Note : Comme le CRC ne vérifie que l'intégrité de l'en-tête lui-même, un accusé de réception de cet en-tête ne signifie pas que de précédents changements de la chaîne statique dans le contexte sont aussi acquittés. En particulier, on devrait faire attention lorsque des paquets IR qui mettent à jour un contexte existant sont suivis par des paquets IR-DYN.

Chaîne dynamique : Une chaîne d'informations de sous en-tête dynamiques. Quelles informations dynamiques sont présentes est inféré de la chaîne statique du contexte.

Charge utile : C'est la charge utile du paquet original correspondant, s'il en est. La présence d'une charge utile est inférée par la longueur du paquet.

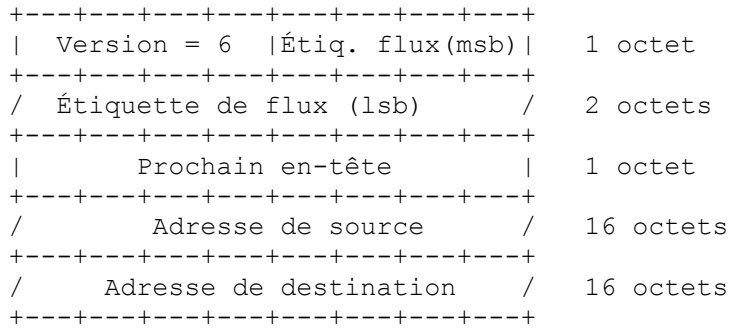
Note : Les chaînes statique et dynamique des paquets IR ou IR-DYN pour le profil 0x0001 (ROHC RTP) DOIVENT se terminer par les parties statique et dynamique d'un en-tête RTP. Sinon, le paquet DOIT être éliminé et le contexte NE DOIT PAS être mis à jour.

Note : Les chaînes statique ou dynamique des paquets IR ou IR-DYN pour le profil 0x0002 (ROHC UDP) DOIVENT se terminer pas les parties statique et dynamique d'un en-tête UDP. Sinon, le paquet DOIT être éliminé et le contexte NE DOIT PAS être mis à jour.

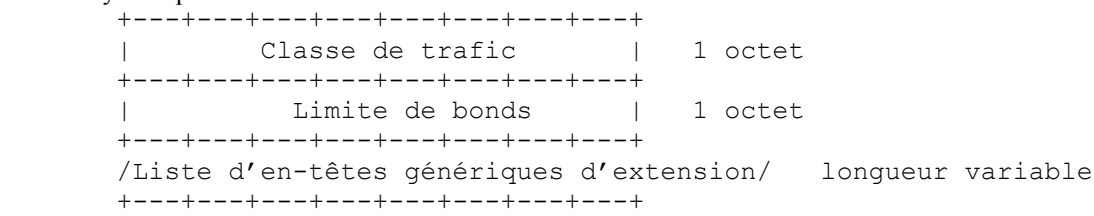
Note : Les chaînes statique ou dynamique des paquets IR ou IR-DYN pour le profil 0x0003 (ROHC ESP) DOIVENT se terminer pas les parties statique et dynamique d'un en-tête ESP. Sinon, le paquet DOIT être éliminé et le contexte NE DOIT PAS être mis à jour.

5.7.7.3 Initialisation de l'en-tête IPv6 [RFC2460]

Partie statique :



Partie dynamique :



Éliminé : Longueur de charge utile

Extras :

Liste d'en-têtes génériques d'extension	Codé selon le paragraphe 5.8.6.1, avec tous les éléments d'en-tête présents en forme non compressée.

CRC-DYNAMIC : Champ Longueur de charge utile (octets 5-6).

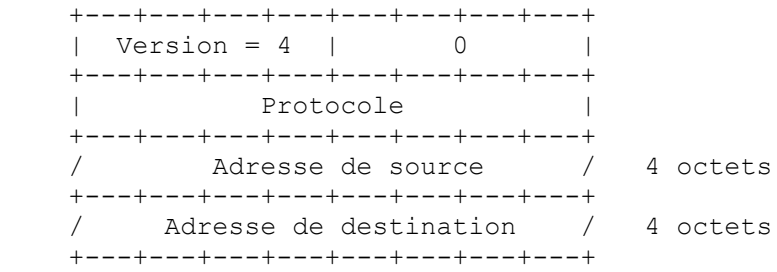
CRC-STATIC : Tous les autres champs (octets 1-4, 7-40).

La couverture du CRC pour les en-têtes d'extension est définie au paragraphe 5.8.7.

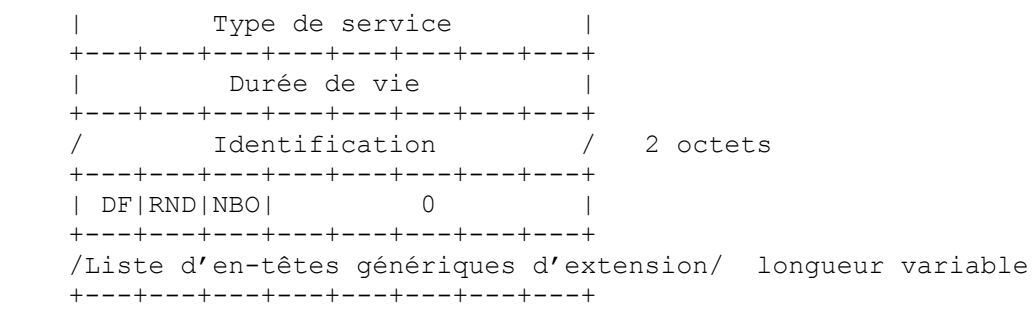
Note : Le champ Prochain en-tête indique le type de l'en-tête suivant dans la chaîne statique, plutôt que d'être une copie du champ Prochain en-tête de l'en-tête IPv6 original. Voir aussi le paragraphe 5.7.7.8.

5.7.7.4 Initialisation d'en-tête IPv4, [RFC0791] paragraphe 3.1

Partie statique :
Version, Protocole, Adresse de source, Adresse de destination.



Partie dynamique :
Type de service, Durée de vie, Identification, DF, RND, NBO, Liste d'en-têtes d'extension.



Éliminés :

IHL	(Longueur d'en-tête IP, doit être 5)
Longueur totale	(inféré du paquet décompressés)
Fanion MF	(Fanion Fragments à suivre, doit être 0)
Décalage de fragment	(doit être 0)
Somme de contrôle d'en-tête	(inféré du paquet décompressés)
Options, Bourrage	(ne doit pas être présent)

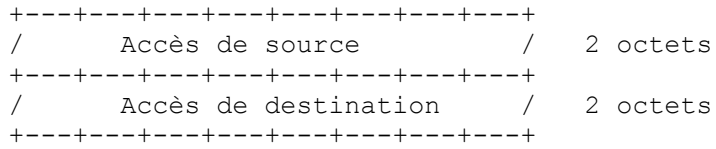
Extras :
RND, NBO Voir le paragraphe 5.7.
Liste d'en-têtes génériques d'extension : Codé selon le paragraphe 5.8.6.1, avec tous les éléments d'en-tête présents en forme non compressée.
CRC-DYNAMIC : Longueur totale, Identification, Somme de contrôle d'en-tête (octets 3-4, 5-6, 11-12).
CRC-STATIC : Tous les autres champs (octets 1-2, 7-10, 13-20)

La couverture du CRC pour les en-têtes d'extension est définie au paragraphe 5.8.7.

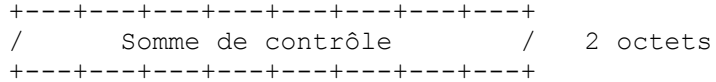
Note : Le champ Protocole indique le type de l'en-tête suivant dans la chaîne statique, plutôt que d'être une copie du champ Protocole de l'en-tête IPv4 original. Voir aussi le paragraphe 5.7.7.8.

5.7.7.5 Initialisation de l'en-tête UDP [RFC-768].

Partie statique :



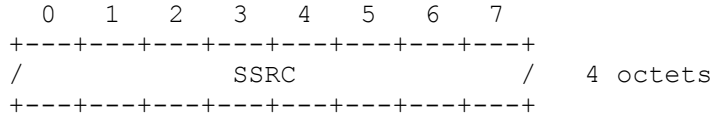
Partie dynamique :



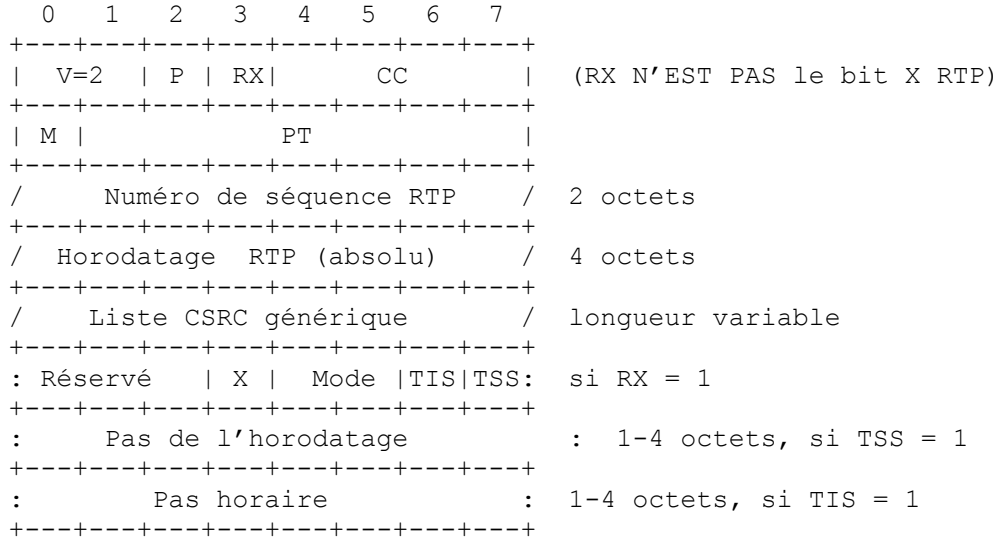
Éliminés :
Longueur
Le champ Longueur de l'en-tête UDP DOIT correspondre au champ Longueur des sous en-têtes précédents, c'est-à-dire, il ne doit y avoir aucun bourrage après la charge utile UDP qui est couverte par Longueur IP.
CRC-DYNAMIC : Champ Longueur, Somme de contrôle (octets 5-8).
CRC-STATIC : Tous les autres champs (octets 1-4).

5.7.7.6 Initialisation de l'en-tête RTP [RFC1889].

Partie statique : SSRC.



Partie dynamique : P, X, CC, PT, M, numéro de séquence, horodatage, pas d'horodatage, identifiants CSRC.



Éliminés : Rien.

Extras :
RX : Contrôle la présence d'extensions.

Mode : Mode de compression.
0 = Réservé,
1 = Unidirectionnel,
2 = Bidirectionnel optimiste,
3 = Bidirectionnel fiable.

X : Copie du bit X provenant de l'en-tête RTP (présumé 0 si RX = 0)

Réservé : Réglé à zéro à l'envoi, ignoré à réception.

Liste de CSRC génériques : Liste des CSRC codée selon le paragraphe 5.8.6.1, avec tous les éléments de CSRC présents.

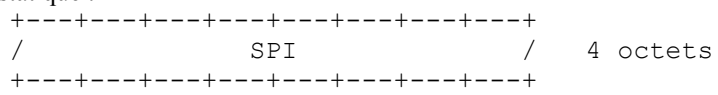
CRC-DYNAMIC : Octets contenant le bit M, le champ numéro de séquence, et l'horodatage (octets 2-8).

CRC-STATIC : Tous les autres champs (octets 1, 9-12, liste originale de CSRC).

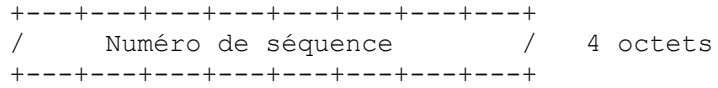
5.7.7.7 Initialisation d'en-tête ESP, [RFC2406] section 2

Ceci est pour le cas où l'algorithme de chiffrement NUL [RFC2410] N'EST PAS utilisé avec ESP, de sorte que les sous en-têtes après l'en-tête ESP sont chiffrés (voir le paragraphe 5.12). Voir en 5.8.4.3 la compression de l'en-tête ESP lorsque le chiffrement NUL est utilisé.

Partie statique :



Partie dynamique :



Éliminés : Les autres champs sont chiffrés, et ne peuvent être ni localisés ni compressés.

CRC-DYNAMIC : Numéro de séquence (octets 5-8)

CRC-STATIC : Tous les autres octets.

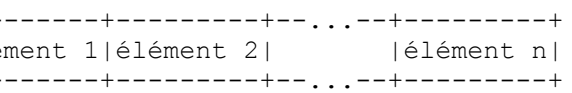
Note : Aucune donnée chiffrée n'est considérée faire partie de l'en-tête pour les besoins du calcul du CRC, c'est-à-dire que les octets après les huit octets ne sont pas considérés comme faisant partie de l'en-tête.

5.7.7.8 Initialisation d'autres en-têtes

Les en-têtes non explicitement énumérés dans les paragraphes précédents ne peuvent être compressés qu'en les incorporant dans une chaîne d'en-tête d'extension suivant un en-tête IPv4 ou IPv6, voir au paragraphe 5.8.

5.8 Compression de liste

Les informations d'en-tête provenant du flux de paquets à compresser peuvent être structurés comme une liste ordonnée, qui est essentiellement constante d'un paquet à l'autre. La structure générique d'une telle liste est la suivante.



Ce paragraphe décrit le schéma de compression pour de telles informations. Les principes de base de la compression fondée sur la liste sont les suivants :

1) Bien que la liste soit constante, aucune information sur la liste n'est envoyée dans les en-têtes compressés.
2) Les petits changements dans la liste sont représentés comme des ajouts (schéma d'insertion) ou des suppressions (schéma de retrait) ou les deux (schéma de retrait puis insertion).

3) La liste peut aussi être envoyée dans sa totalité (schéma générique).

Il y a deux sortes de listes : listes de CSRC dans les paquets RTP, et chaînes d'en-tête d'extension dans les paquets IP (IPv4 et IPv6).

Les en-têtes de base IPv6 et les en-têtes IPv4 ne peuvent pas faire partie d'une chaîne d'en-tête d'extension. Les en-têtes qui peuvent faire partie d'une chaîne d'en-tête d'extension incluent :
a) les en-têtes AH,
b) l'en-tête ESP nul,
c) l'en-tête d'encapsulation minimal, [RFC2004] ,paragraphe 3.1,
d) l'en-tête GRE, [RFC2784], [RFC2890]
e) les en-têtes d'extension IPv6.

Le schéma de compression d'élément fondée sur le tableau (5.8.1) qui réduit la taille de chaque élément, est décrit d'abord. Puis on définit quelle liste de référence utiliser dans les schémas d'insertion et de retrait (5.8.2). Les schémas de codage de liste sont décrits au paragraphe 5.8.3, et quelques cas particuliers au paragraphe 5.8.4. Enfin, les formats exacts sont décrits dans les paragraphes 5.8.5-5.8.6.

5.8.1 Compression d'élément fondée sur le tableau

Le schéma de compression d'élément fondé sur le tableau est un moyen de compresser des éléments individuels envoyés dans des listes compressées. Le compresseur alloue à chaque élément d'une liste un indice identifiant univoque. Le compresseur entretient conceptuellement un tableau avec tous les éléments, indexés. La paire (indice, élément) est envoyée dans une liste compressée jusqu'à ce que le compresseur ait acquis l'assurance que le décompresseur a observé la transposition entre l'élément et son indice. Une telle assurance est obtenue par la réception d'un accusé de réception de la part du décompresseur en mode R, et en mode U/O par l'envoi de L paires (indice, élément) (pas nécessairement consécutives). Après cela, l'indice seul est envoyé dans des listes compressées pour indiquer l'élément correspondant. Le compresseur peut réallouer un indice existant à un nouvel élément, et a alors besoin de rétablir la correspondance de la même manière que ci-dessus.

Le décompresseur entretient conceptuellement un tableau qui contient toutes les paires (indice, élément) qu'il connaît. Le tableau est mis à jour chaque fois qu'une paire (indice, élément) est reçue (et sa décompression est vérifiée par un CRC). Le décompresseur restitue l'élément à partir du tableau chaque fois qu'est reçu un indice sans élément accompagnant.

5.8.1.1 Tableau de traduction en mode R

Du côté du compresseur, une entrée dans le tableau de traduction a la structure suivante.

```
          +-------+-------+---------------+
Indice i  | Connu |élément| SN1, SN2, ... |
          +-------+-------+---------------+
```

Le fanion Connu indique si la transposition entre l'indice i et l'élément a été établie, c'est-à-dire, si Indice i seul peut être envoyé dans des listes compressées. Connu est initié à zéro. Il est aussi réglé à zéro chaque fois que Indice i est alloué à un nouvel élément. Connu est réglé à un lorsque la paire (indice, élément) correspondante est acquittée. Les accusés de réception sont fondés sur le numéro de séquence RTP, de sorte qu'une liste de numéros de séquence RTP de tous les paquets qui contiennent la paire (indice, élément) est incluse dans le tableau de traduction. Lorsque est acquitté un paquet avec un numéro de séquence dans la liste des numéros de séquence, le fanion Connu est établi, et la liste des numéros de séquence peut être éliminée.

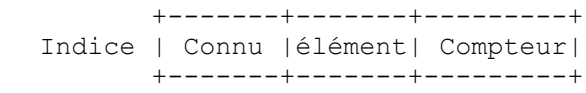
Chaque entrée dans le tableau de traduction du côté du décompresseur a la structure suivante :

```
          +-------+-------+
Indice i  | Connu |élément|
          +-------+-------+
```

Tout champ Connu est initialisé à zéro. Chaque fois que le décompresseur reçoit une paire (indice, élément) il insère l'élément dans le tableau à une position d'indice et règle le fanion Connu dans cette entrée à un. Si un indice sans un élément accompagnant est reçu pour lequel le fanion Connu est zéro, l'en-tête DOIT être éliminé et un NACK DEVRAIT être envoyé.

5.8.1.2 Tableau de traduction en mode U/O

Du côté compresseur, chaque entrée dans le tableau de traduction a la structure suivante :



Les champs Indice, Connu, et élément ont la même signification qu'au paragraphe 5.8.1.1.

Connu est établi lorsque la paire (Indice, élément) a été envoyée dans L listes compressées (pas nécessairement consécutives). Le champ Compteur garde trace du nombre de fois que la paire a été envoyée. Compteur est réglé à 0 pour chaque nouvelle entrée ajoutée au tableau, et chaque fois qu'un indice est alloué à un nouvel élément. Compteur est incrémenté de 1 chaque fois qu'une paire (indice, élément) est envoyée. Lorsque le compteur atteint L, le champ Connu est établi et après cela seul l'indice a besoin d'être envoyé dans les listes compressées.

Du côté du décompresseur, le tableau de traduction est le même que le tableau de traduction défini en mode R.

5.8.2 Détermination de la liste de référence

Dans les schémas de compression fondés sur la référence (c'est-à-dire, les schémas fondés sur l'ajout ou la suppression) la compression et la décompression d'une liste (curr_list) sont fondées sur une liste de référence (ref_list) qui est supposée être présente dans le contexte du compresseur et du décompresseur. La liste compressée est un codage des différences entre curr_list et ref_list. À réception d'une liste compressée, le décompresseur applique les différences à sa liste de référence afin d'obtenir la liste d'origine.

Pour identifier la liste de référence à utiliser, chaque liste compressée porte un identifiant (ref_id). La liste de référence est établie par des méthodes différentes en mode R et en mode U/O.

5.8.2.1 Liste de référence en mode R et en mode U/O

En mode R, le choix d'une liste de référence se fonde sur des accusés de réception, c'est-à-dire, le compresseur utilise comme ref_list la dernière liste qui a été acquittée par le décompresseur. La ref_list est mise à jour seulement à réception d'un accusé de réception. Les bits de moindre poids du numéro de séquence RTP du paquet acquitté sont utilisés comme ref_id.

En mode U/O, une séquence de listes identiques est considérée comme appartenant à la même génération et il leur est alloué le même identifiant de génération (gen_id). Les gen_id augmentent de 1 chaque fois que la liste change et ils sont portés dans des listes compressées et non compressées qui sont candidates à être utilisées comme listes de référence. Normalement, gen_id doit avoir été répété dans au moins L en-têtes avant que la liste puisse être utilisée comme ref_list. Cependant, certains accusés de réception peuvent être envoyés en mode O (et aussi en mode U) et chaque fois qu'un accusé de réception est reçu pour un en-tête, la liste de cet en-tête est considérée comme connue et n'a pas besoin d'être encore répétée. Les bits de moindre poids de Gen_id sont utilisés comme ref_id en mode U/O.

La logique de compresseur et de décompresseur pour la compression de liste fondée sur la référence est similaire à celle de SN et TS. La principale différence est que le décompresseur entretient une fenêtre glissante avec les candidates pour ref_list, et récupère ref_list de la fenêtre glissante en utilisant la ref_id de la liste compressée.

Logique du compresseur :

a) Dans l'état IR, le compresseur envoie des listes génériques (voir le paragraphe 5.8.5) contenant tous les éléments de la liste en cours afin d'établir ou de rafraîchir le contexte du décompresseur.
   En mode R, de telles listes génériques sont envoyées jusqu'à ce qu'un en-tête soit acquitté. La liste de cet en-tête peut être utilisée comme liste de référence pour compresser des listes ultérieures.
   En mode U/O, le compresseur envoie des identifiants de génération avec les listes génériques jusqu'à ce que :
   1) un identifiant de génération ait été répété L fois, ou
   2) qu'un accusé de réception pour un en-tête portant un identifiant de génération ait été reçu.
   La liste répétée (1) ou acquittée (2) peut être utilisée comme liste de référence pour compresser les listes suivantes et est gardée avec son identifiant de génération.
b) Lorsque on n'est pas dans l'état IR, le compresseur passe à l'état FO quand il observe une différence entre curr_list et la liste précédente. Il envoie des listes compressées fondées sur ref_list pour mettre à jour le contexte du décompresseur. (Cependant, voir d).)
   En mode R, le compresseur continue d'envoyer des listes compressées en utilisant la même référence jusqu'à ce qu'il reçoive un accusé de réception pour un paquet contenant la liste la plus récente. Le compresseur peut alors passer à l'état SO par rapport à la liste.

En mode U/O, le compresseur continue d'envoyer des listes compressées avec des identifiants de génération jusqu'à ce que :
1) un identifiant de génération ait été répété L fois, ou
2) qu'un accusé de réception pour un en-tête portant le dernier identifiant de génération ait été reçu.
La liste répétée ou acquittée est utilisée comme future liste de référence. Le compresseur peut passer à l'état SO par rapport à la liste.

c) En mode R, le compresseur entretient une fenêtre glissante contenant les listes qui ont été envoyées pour mettre à jour le contexte du décompresseur et n'ont pas encore été acquittées. La fenêtre glissante se rétrécit quant un accusé de réception arrive : toutes les listes envoyées avant l'accusé de réception sont retirées. Le compresseur peut utiliser l'indice pour représenter les éléments des listes dans la fenêtre glissante.
En mode U/O, le compresseur a besoin de mémoriser
1) les listes de référence et leur identifiant de génération, et
2) si l'identifiant de génération actuel est différent de la génération de référence, la liste actuelle et les numéros de séquence avec lesquels la liste actuelle a été envoyée. (2) est nécessaire pour déterminer si un accusé de réception concerne la dernière génération. Il n'est pas nécessaire en mode U.

d) En mode U/O, le compresseur peut choisir de ne pas envoyer d'identifiant de génération avec une liste compressée. De telles listes sans identifiant de génération ne reçoivent pas de nouvel identifiant de génération et ne doivent pas être utilisées comme futures listes de référence. Elles ne mettent pas à jour le contexte. Cette caractéristique est utile lorsque une nouvelle liste est répétée peu de fois et que la liste revient alors à son ancienne valeur.

Logique du décompresseur :

e) En mode R, le décompresseur accuse réception de toutes les listes reçues non compressées ou compressées qui établissent ou mettent à jour le contexte. (De tels en-têtes compressés contiennent un CRC.)
En mode O, le décompresseur PEUT accuser réception d'une liste avec un nouvel identifiant de génération, voir au paragraphe 5.4.2.2.
En mode U, le décompresseur PEUT accuser réception d'une liste envoyée dans un paquet IR, voir au paragraphe 5.3.2.3.

f) Le décompresseur entretient une fenêtre glissante qui contient les listes qui peuvent être utilisées comme listes de référence.
En mode R, la fenêtre glissante contient les listes qui ont été acquittées mais non encore utilisées comme listes de référence.
En mode U/O, la fenêtre glissante contient au plus une liste par génération. Elle contient toutes les générations vues par le décompresseur qui sont plus récentes que la dernière génération utilisée comme référence.

g) Lorsque le décompresseur reçoit une liste compressée, il récupère la ref_list appropriée dans la fenêtre glissante sur la base du ref_id, et décompresse la liste compressée pour obtenir curr_list.
En mode R, curr_list est insérée dans la fenêtre glissante si un accusé de réception est envoyé pour elle. La fenêtre glissante est rétrécie par le retrait de toutes les listes reçues avant ref_list.
En mode U/O, curr_list est insérée dans la fenêtre glissante avec son identifiant de génération si la liste compressée avait un identifiant de génération et si la fenêtre glissante ne contenait pas une liste avec cet identifiant de génération. Toutes les listes avec des générations plus anciennes que ref_id sont retirées de la fenêtre glissante.

5.8.3 Schémas de codage pour la liste compressée

On décrit ici quatre schémas de codage pour la liste compressée. Les formats exacts de la liste compressée de CSRC et de la liste d'en-têtes d'extension IP compressés à l'aide de ces schémas de codages sont décrits aux paragraphes 5.8.5-5.8.6.

Schéma générique
À la différence des schémas suivants, ce schéma ne s'appuie pas sur l'établissement d'une liste de référence. La liste entière est envoyée, en utilisant la compression fondée sur le tableau pour chaque élément individuel. Le schéma générique est toujours utilisé lors de l'établissement du contexte du décompresseur et peut aussi être utilisé à d'autre moments, comme le compresseur le juge utile.

Schéma d'insertion seulement
Lorsque la nouvelle liste peut être construite à partir de ref_list par l'ajout d'éléments, une liste des éléments ajoutés est envoyée (en utilisant la compression fondée sur le tableau) avec les positions dans ref_list où les nouveaux éléments seront insérés. Un gabarit binaire d'insertion indique les positions d'insertion dans ref_list.

À réception d'une liste compressée conformément au schéma Insertion seulement, curr_list est obtenue en examinant le gabarit binaire d'insertion de gauche à droite. Lorsque un '0' est observé, un élément est copié depuis la ref_list. Lorsque un '1' est observé, un élément est copié de la liste des éléments ajoutés. Si un '1' est observé lorsque la liste des éléments ajoutés est épuisée, une erreur est survenue et la décompression échoue : l'en-tête NE DOIT PAS être livré au couches supérieures ; il devrait être éliminé et NE DOIT PAS être l'objet d'un accusé de réception ni utilisé comme référence.

Pour construire le gabarit binaire d'insertion et la liste des éléments ajoutés, le compresseur PEUT utiliser l'algorithme suivant :

1) On commence par générer une liste binaire vide et une liste vide d'éléments insérés.
2) On examine le premier élément de curr_list et ref_list.
3) Si curr_list a un élément différent de ref_list,
   un bit établi (1) est ajouté à la liste binaire ; le premier élément dans curr_list (représenté en utilisant la compression d'élément fondée sur le tableau) est ajouté à la liste d'éléments insérés ; on passe à l'élément suivant de curr_list ;
   autrement,
   un bit zéro (0) est ajouté à la liste binaire ;
   on avance à l'élément suivant de curr_list ;
   on avance à l'élément suivant de ref_list.
4) Répéter 3) jusqu'à ce que curr_list soit épuisée.
5) Si la longueur de la liste binaire est inférieure à celle du gabarit binaire requis, ajouter des zéros supplémentaires.

Schéma Suppression seulement

Ce schéma peut être utilisé lorsque curr_list peut être obtenu en retirant des éléments de ref_list. Les positions des éléments qui sont dans ref_list, mais pas dans curr_list, sont envoyées comme gabarit binaire de retrait.

À réception de la liste compressée, le décompresseur obtient curr_list en examinant le gabarit binaire de retrait de gauche à droite. Lorsque un '0' est observé, l'élément suivant de ref_list est copié dans curr_list. Lorsque un '1' est observé, le prochain élément de ref_list est sauté sans être copié. Si un '0' est observé lorsque ref_list est épuisée, une erreur est survenue et la décompression échoue: L'en-tête NE DOIT PAS être livré aux couches supérieures ; il devrait être éliminé, et NE DOIT PAS faire l'objet d'un accusé de réception ni être utilisé comme référence.

Pour construire le gabarit binaire de retrait et la liste des éléments ajoutés, le compresseur PEUT utiliser l'algorithme suivant :

1) On commence par générer une liste binaire vide.
2) On examine le premier élément de curr_list et ref_list.
3) Si curr_list a un élément différent de ceux de ref_list,
   on ajoute un bit établi (1) dans la liste binaire ; on avance au prochain élément de ref_list ;
   autrement,
   un bit zéro (0) est ajouté à la liste binaire ;
   on avance à l'élément suivant de curr_list ;
   on avance à l'élément suivant de ref_list.
4) Répéter 3) jusqu'à épuisement de curr_list.
5) Si la longueur de la liste binaire est inférieure à la longueur du gabarit binaire requis, ajouter des uns supplémentaires.

Schéma Retirer puis insérer

Dans ce schéma, curr_list est obtenue en retirant d'abord des éléments de ref_list, puis en insérant des éléments dans la liste résultante. Un gabarit binaire de retrait, un gabarit binaire d'insertion, et une liste d'éléments ajoutés sont envoyés.

À réception de la liste compressée, le décompresseur traite le gabarit binaire de retrait comme dans le schéma Retrait seulement. La liste résultante est alors utilisée comme liste de référence lorsque le gabarit binaire d'insertion et la liste des éléments ajoutés sont traités comme dans le schéma Insertion seulement.

5.8.4 Traitements particuliers des en-têtes d'extension IP

Dans la compression de liste de CSRC, chaque CSRC reçoit un indice. À l'opposé, dans la compression de liste d'en-tête d'extension IP, un indice est normalement associé au type d'en-tête d'extension. Lorsque il y a plus d'un en-tête IP, il y a plus d'une liste d'en-têtes d'extension. Un indice par type par liste est alors utilisé.

L'association à un type signifie qu'un nouvel indice n'a pas toujours besoin d'être utilisé chaque fois que change un champ dans un en-tête d'extension IP. Cependant, lorsque change un champ dans un en-tête d'extension , la transposition entre l'indice et la nouvelle valeur de l'en-tête d'extension doit être établie, sauf dans les cas de traitement particulier définis dans les paragraphes suivants.

5.8.4.1 Champ Prochain en-tête

Le champ Prochain en-tête dans un en-tête IP ou en-tête d'extension change chaque fois que le type de l'en-tête qui suit immédiatement change, par exemple, lorsque un nouvel en-tête d'extension est inséré après lui, quand l'en-tête d'extension suivant immédiatement est retiré de la liste, ou quand l'ordre des en-têtes d'extension est changé. Donc, il peut n'être pas extraordinaire que, pour un certain en-tête, le champ de prochain en-tête change alors que les autres champs ne changent

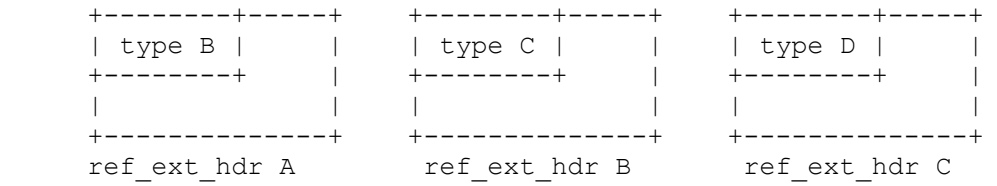
pas.

Donc, dans le cas où seul le champ Prochain en-tête change, l'en-tête d'extension est considéré comme inchangé et les règles de traitement particulier du changement dans le champ Prochain en-tête sont définies ci-dessous.

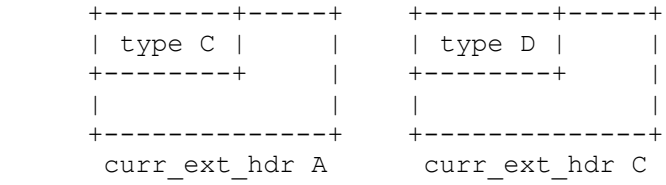
Tous les éléments d'en-tête d'extension non compressés communiqués indiquent leur propre type dans leur champ Prochain en-tête. Noter que les règles ci-dessous expliquent comment traiter les champs Prochain en-tête tout en montrant la liste de référence conceptuelle comme une re-création exacte de la liste originale d'en-tête d'extension non compressée.

a) Lorsque un en-tête d'extension suivant est retiré de la liste, la nouvelle valeur du prochain champ d'en-tête est obtenue de la liste de référence d'en-tête d'extension. Par exemple, en supposant que la liste d'en-tête de référence (ref_list) consiste en les en-têtes A, B et C (ref_ext_hdr A, B, C) et que la liste en cours d'en-tête d'extension (curr_list) ne comporte que les en-têtes d'extension A et C (curr_ext_hdr A, C). L'ordre et la valeur des prochains champs d'en-tête de ces en-têtes d'extension sont comme suit.

ref_list :



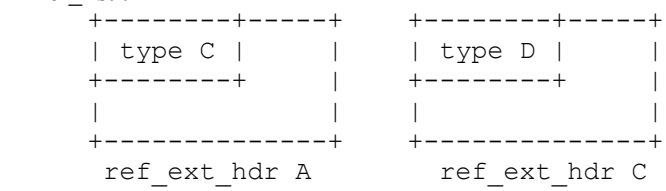
curr_list :



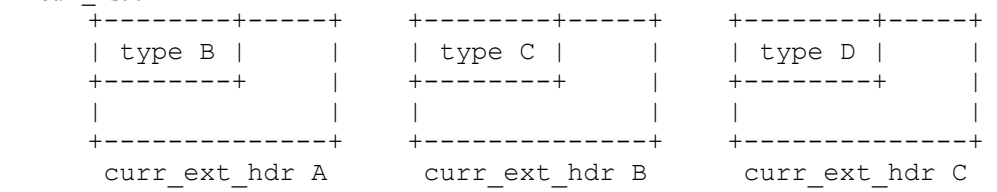
En comparant curr_ext_hdr A dans curr_list et ref_ext_hdr A dans ref_list, la valeur du champ Prochain en-tête est changée de "type B" en "type C" à cause du retrait de l'en-tête d'extension B. La nouvelle valeur du champ Prochain en-tête dans curr_ext_hdr A, c'est-à-dire, "type C", n'a pas besoin d'être envoyée au décompresseur. Elle est restituée à partir du champ Prochain en-tête du ref_ext_hdr B retiré.

b) Lorsque un nouvel en-tête d'extension est inséré après un en-tête d'extension existant, le champ prochain en-tête dans les éléments communiqués va porter son propre type, plutôt que le type de l'en-tête qui suit. Par exemple, supposons que la liste d'en-tête de référence (ref_list) comporte les en-têtes A et C (ref_ext_hdr A, C) et que la liste d'en-têtes en cours (curr_list) consiste en les en-têtes A, B et C (curr_ext_hdr A, B, C). L'ordre et la valeur des champs Prochain en-tête de ces en-tête d'extensions sont comme suit.

ref_list :



curr_list :



En comparant curr_list et ref_list, la valeur du champ Prochain en-tête dans l'en-tête d'extension A est changée de "type C" en "type B".

Le curr_ext_hdr B non compressé est porté dans la liste d'en-têtes compressés. Cependant, il porte "type B" au lieu de "type C" dans son champ Prochain en-tête. Lorsque le décompresseur insère un nouvel en-tête après curr_ext_hdr A, le

champ Prochain en-tête de A est tiré du prochain en-tête, et le champ Prochain en-tête du nouvel en-tête est tiré de ref_ext_hdr A.

c) Certains en-têtes dont la compression est définie dans le présent document ne contiennent pas de champ Prochain en-tête, ou n'ont pas leur champ Prochain en-tête dans la position standard (premier octet de l'en-tête). Les en-têtes GRE et ESP sont de tels en-têtes. Lorsque ils sont envoyés comme éléments non compressés dans les listes, ces en-têtes sont modifiés de telle sorte qu'ils aient un champ Prochain en-tête comme premier octet (voir les paragraphes 5.8.4.3 et 5.8.4.4). Ceci est nécessaire pour permettre au décompresseur de décoder l'élément.

5.8.4.2 En-tête d'authentification (AH)

Le champ Numéro de séquence dans AH [RFC2402] contient une valeur de compteur à accroissement monotone pour une association de sécurité. Donc, lorsque on comparé curr_list à ref_list, si le numéro de séquence dans AH change et si le champ SPI ne change pas, le AH n'est pas considéré comme changé.

Si le numéro de séquence dans AH augmente de façon linéaire lorsque le numéro de séquence RTP augmente, et si le compresseur est sûr que le décompresseur a obtenu le schéma, le numéro de séquence dans AH n'a pas besoin d'être envoyé. Le décompresseur applique une extrapolation linéaire pour reconstruire le numéro de séquence dans l'AH.

Autrement, un numéro de séquence compressé est inclus dans le champ compression IPX dans une Extension 3 d'un en-tête UOR-2.

Le champ Données d'authentification dans AH change d'un paquet à l'autre et est envoyé tel quel. Si le AH non compressé est envoyé, le champ Données d'authentification est envoyé à l'intérieur du AH non compressé ; autrement, il est envoyé après les en-têtes d'extension IP/UDP/RTP et IPv6 compressés et avant la charge utile. Voir le début du paragraphe 5.7.

Note : Le champ Longueur de charge utile de AH utilise une motion différente de longueur que les autres en-têtes d'extension IPv6.

5.8.4.3 En-tête d'encapsulation de charge utile de sécurité (ESP)

Quand l'en-tête d'encapsulation de charge utile de sécurité (ESP) [RFC2406] est présent et qu'un algorithme de chiffrement autre que NUL est utilisé, les en-têtes UDP et RTP sont tous deux chiffrés et ne peuvent pas être compressés. L'en-tête ESP termine donc la chaîne d'en-têtes compressibles. Le profil ESP ROHC défini au paragraphe 5.12 PEUT être utilisé pour le flux dans ce cas.

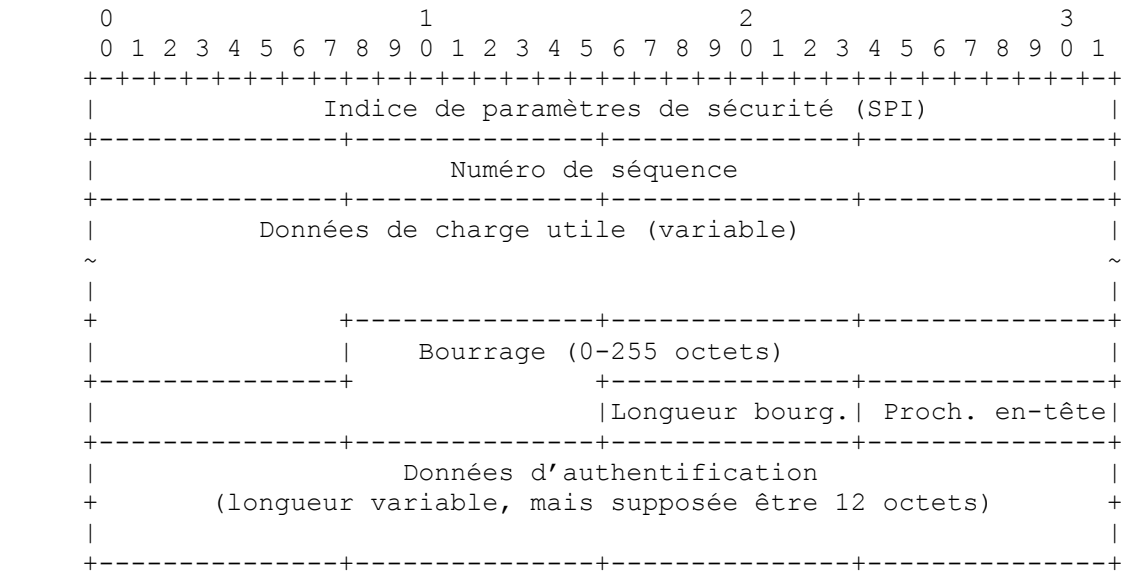
Un cas particulier est lorsque l'algorithme de chiffrement NUL est utilisé. C'est le cas lorsque l'en-tête ESP est utilisé pour la seule authentification, et non pour le chiffrement. La charge utile n'est pas chiffrée par l'algorithme de chiffrement NUL, de sorte que la compression du reste de la chaîne d'en-têtes est possible. Le reste de cette section décrit la compression de l'en-tête ESP lorsque l'algorithme de chiffrement NUL est utilisé avec ESP.

Il n'est pas possible de déterminer si le chiffrement NUL est utilisé en inspectant un en-tête dans le flux, ces informations ne sont présentes qu'aux points d'extrémité du chiffrement. Cependant, un compresseur peut tenter la compression dans l'hypothèse où l'algorithme de chiffrement NUL serait utilisé, et d'interrompre ultérieurement la compression si l'hypothèse se révèle fausse.

Le compresseur peut, par exemple, inspecter les champs Prochain en-tête et les champs d'en-tête supposés être statiques dans les en-têtes suivants afin de déterminer si le chiffrement NUL est utilisé. Si ceux-ci changent de façon imprévisible, un algorithme de chiffrement autre que NUL est probablement utilisé et la compression des en-têtes suivants DEVRAIT être interrompue. La compression du flux est alors soit interrompue, soit un profil qui compresse seulement jusqu'à l'en-tête ESP peut être utilisé (voir au paragraphe 5.12). Tout en tentant de compresser l'en-tête, le compresseur devrait utiliser le SPI de l'en-tête ESP ainsi que l'adresse IP de destination comme champs de définition pour déterminer quels paquets appartiennent au flux.

Dans l'en-tête ESP [RFC2406] section 2, les champs qui peuvent être compressés sont le SPI, le numéro de séquence, le prochain en-tête, et les octets de bourrage si il sont dans le format standard défini dans la [RFC2406]. (Comme toujours, le décompresseur réinsère ces champs sur la base des informations du contexte. Il faut prendre soin de réinsérer correctement toutes les informations car les données d'authentification doivent être vérifiées sur exactement les mêmes informations que celles sur lesquelles il a été calculé.)

En-tête ESP :



SPI : Statique. Si il change, il doit être rétabli.

Numéro de séquence : Non envoyé lorsque le décalage par rapport au numéro de séquence de l'en-tête compressé est constant. Quand le décalage n'est pas constant, le numéro de séquence peut être compressé par l'envoi des LSB. Voir 5.8.4.

Données de charge utile : C'est là qu'on va trouver les en-têtes suivants. Analyse selon le champ Prochain en-tête.

Bourrage : Les octets de bourrage sont supposés être comme défini dans la [RFC2406], c'est-à-dire, prendre les valeurs 1, 2, ..., k, où k = Longueur de bourrage. Si le bourrage dans le contexte statique a ce schéma, le bourrage dans les en-têtes compressés est supposé avoir aussi ce schéma et est retiré. Si le bourrage dans le contexte statique n'a pas ce schéma, le bourrage n'est pas retiré.

Longueur de bourrage : Dynamique. Toujours envoyé. 14ème octet à partir de la fin du paquet.

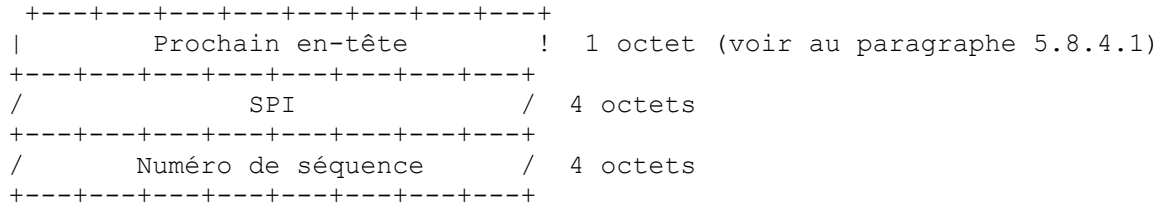
Prochain en-tête : Statique. 13ème octet à partir de la fin du paquet.

Données d'authentification : Elles peuvent avoir une longueur variable, mais quand la compression d'en-tête ESP en chiffrement NUL est tentée, on suppose une longueur de 12 octets.

Le numéro de séquence dans ESP a le même comportement que le champ Numéro de séquence dans AH. Quand il augmente de façon linéaire, il peut être compressé à zéro bit. Quand son augmentation n'est pas linéaire, un numéro de séquence compressé est inclus dans le champ compression IPX dans une Extension 3 d'un en-tête UOR-2.

Les informations qui font partie d'un élément non compressé d'une liste compressée sont le champ Prochain en-tête, suivi par le SPI et le numéro de séquence. Bourrage, Longueur de bourrage, Prochain en-tête, et Données d'authentification sont envoyés à la fin du paquet. Cela signifie que le prochain en-tête survient en deux endroits.

Élément non compressé de liste ESP :



Lors de l'envoi d'éléments de liste ESP non compressés, tous les champs ESP proches de la fin du paquet sont laissés inchangés (Bourrage, Longueur de bourrage, Prochain en-tête, Données d'authentification).

Un élément compressé consiste en un numéro de séquence compressé. Lorsque un élément est compressé, Bourrage (si il suit le schéma 1, 2, ..., k) et Prochain en-tête sont retirés près de la fin du paquet. Données d'authentification et Longueur de bourrage restent tels quels près de la fin du paquet.

5.8.4.4 En-tête GRE [RFC2784], [RFC2890]

L'en-tête GRE est un ensemble de fanions, suivi par un champ obligatoire Type de protocole et des parties facultatives comme indiqué par les fanions.

Le champ Numéro de séquence dans l'en-tête GRE contient une valeur de compteur pour un tunnel GRE. Donc, pour comparer curr_list avec ref_list, si le numéro de séquence dans GRE change, le GRE n'est pas considéré comme changé.

Si le numéro de séquence dans l'en-tête GRE augmente de façon linéaire comme augmente le numéro de séquence RTP et si le compresseur est sûr que le décompresseur a reçu le schéma, le numéro de séquence dans GRE n'a pas besoin d'être envoyé. Le décompresseur applique une extrapolation linéaire pour reconstruire le numéro de séquence dans l'en-tête GRE.

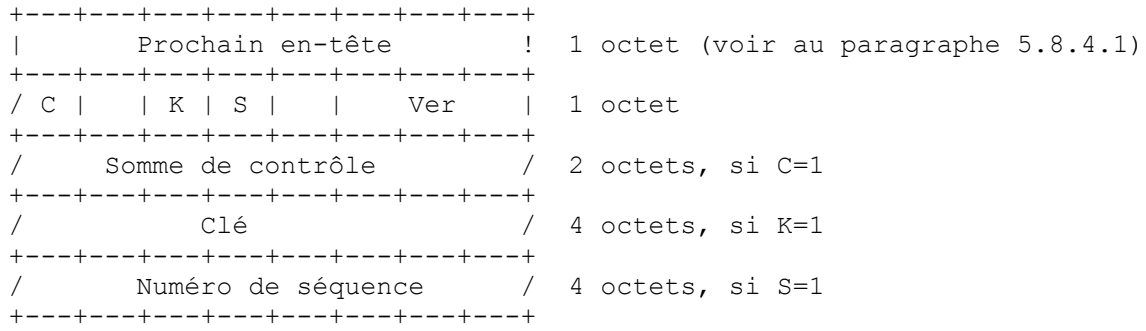
Autrement, un numéro de séquence compressé est inclus dans le champ compression IPX dans une Extension 3 d'un en-tête UOR-2.

Le champ Données de somme de contrôle dans GRE, s'il est présent, change d'un paquet à l'autre et est envoyé tel quel. Si l'en-tête GRE non compressé est envoyé, le champ Données de somme de contrôle est envoyé à l'intérieur de l'en-tête GRE non compressé ; autrement, s'il est présent, il est envoyé après les en-têtes d'extension IP/UDP/RTP et IPv6 compressés et avant la charge utile. Voir le début du paragraphe 5.7.

Afin de permettre une analyse simple des listes d'éléments, un en-tête GRE non compressé envoyé comme un élément dans une liste est modifié par rapport à l'en-tête GRE original de la manière suivante :

1) le champ Type de protocole de 16 bits qui code le type de l'en-tête suivant en utilisant les Ethertypes (voir la section Ethertypes dans la [RFC1700]) est retiré.
2) Un champ d'un octet Prochain en-tête est inséré comme premier octet de l'en-tête. La valeur du champ Prochain en-tête correspond à GRE (cette valeur est 47 selon la section des numéros alloués des protocoles de l'Internet [RFC1700]) quand l'élément non compressé est à insérer dans une liste, et au type de l'en-tête suivant quand l'élément non compressé est dans une liste générique. Noter que cela implique que seuls les en-têtes GRE avec des Ethertypes qui correspondent à un numéro de protocole IP peuvent être compressés.

Élément non compressé de liste GRE :

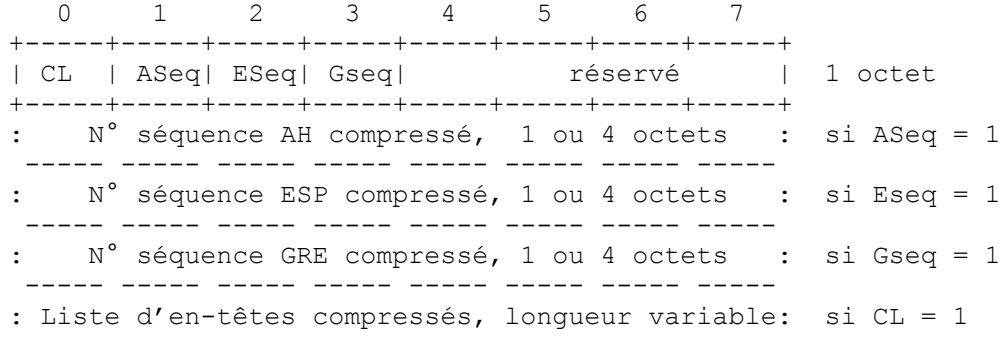


Les bits laissés en blanc dans le second octet sont réglés à zéro à l'émission et ignorés à réception.
Les champs Réservé0 et Réservé1 de l'en-tête GRE [RFC2890] doivent être tout de zéros ; autrement, le paquet ne peut pas être compressé par ce profil.

5.8.5 Format des listes compressées en Extension 3

5.8.5.1 Format du champ En-têtes d'extension IP

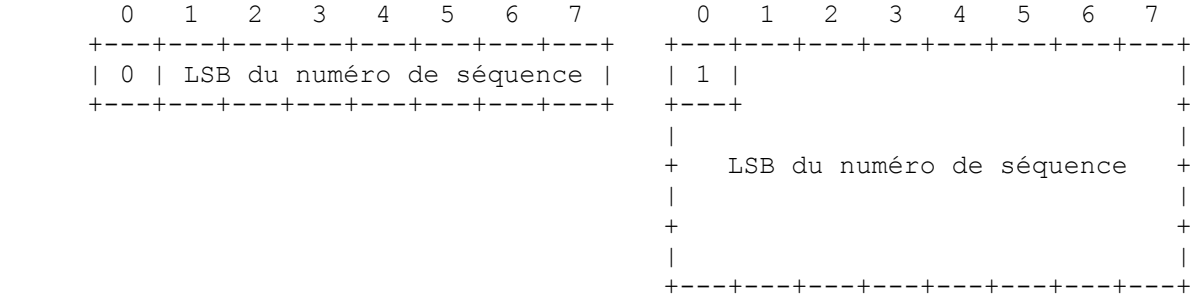
Dans Extension 3 (paragraphe 5.7.5), il y a un champ appelé En-têtes d'extension IP. Ce paragraphe décrit le format de ce champ.



Aseq : indique la présence de numéro de séquence AH compressé
Eseq : indique la présence de numéro de séquence ESP compressé
Gseq : indique la présence de numéro de séquence GRE compressé
CL : indique la présence d'une liste d'en-têtes compressés
réservé : réglé à zéro à l'émission, ignoré à réception

Lorsque Aseq, Eseq, ou Gseq est établi, l'élément d'en-tête correspondant (en-tête AH, ESP, ou GRE) est compressé. Lorsque il n'est pas établi, l'élément d'en-tête correspondant est envoyé non compressé ou n'est pas présent.

Le format des numéros de séquence AH, ESP et GRE compressés peut pour chacun être le suivant :



Le format du champ Liste d'en-têtes compressés est décrit au paragraphe 5.8.6.

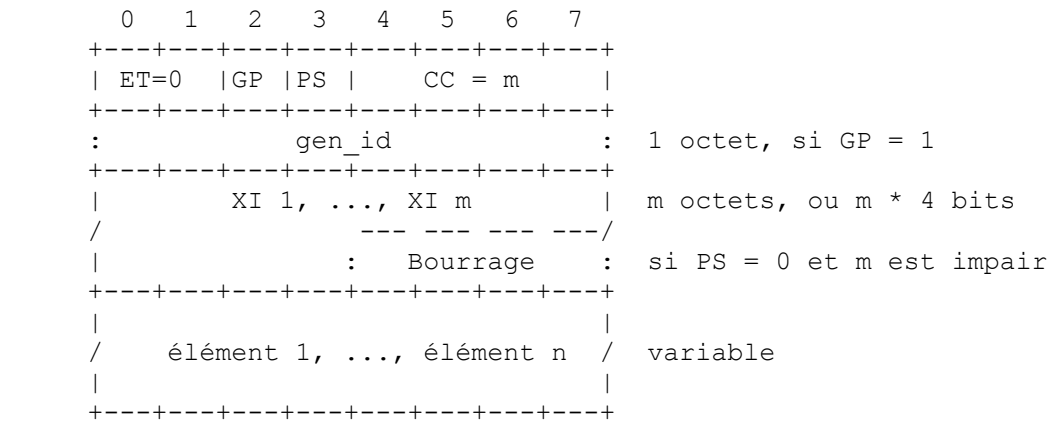
5.8.5.2 Format de Liste de CSRC compressée

Le champ Liste de CSRC compressés dans la partie en-tête RTP d'une Extension 3 (paragraphe 5.7.5) est comme au paragraphe 5.8.6.

5.8.6 Formats de liste compressée

Cette section décrit le format des listes compressées. Le format est le même pour les listes de CSRC et les listes d'en-têtes. Dans les listes de CSRC, les éléments sont des identifiants de CSRC ; dans les listes d'en-têtes, ce sont des en-têtes non compressés ou compressés, comme décrit en 5.8.4.2-4.

5.8.6.1 Type 0 de codage (schéma générique)



ET : *(Encoding Type)* Le type de codage est zéro.

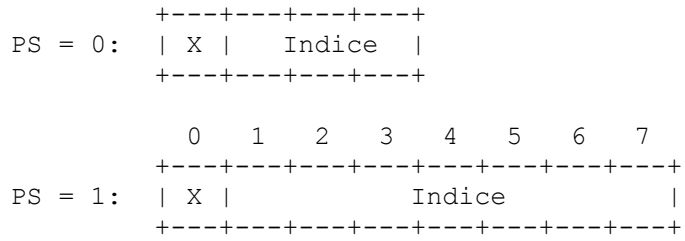
PS : Indique la taille des champs XI :
PS = 0 indique des champs XI de 4 bits ;
PS = 1 indique des champs XI de 8 bits.

GP : Indique la présence du champ gen_id.

CC : Compteur des CSRC à partir de l'en-tête RTP original.

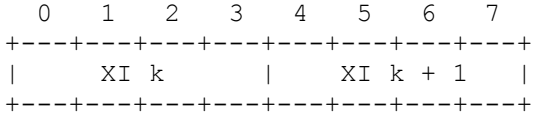
gen_id : Identifiant pour une séquence de listes identiques. Il est présent en mode U/O lorsque le compresseur décide qu'il peut utiliser cette liste comme future liste de référence.

XI 1, ..., XI m : m éléments XI. Le format d'un élément XI est le suivant :



X = 1 indique que l'élément correspondant à l'indice est envoyé dans la liste item 0, ..., item n.
X = 0 indique que l'élément correspondant à l'indice n'est pas envoyé.

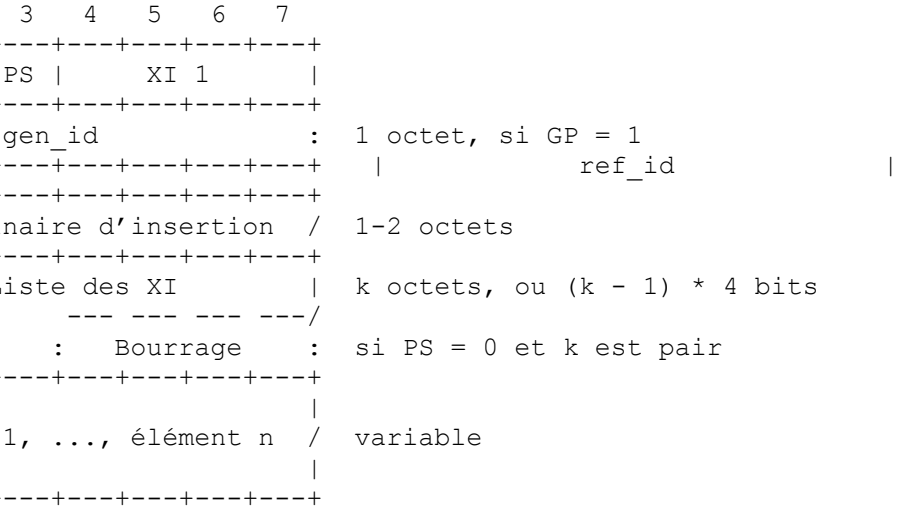
Lorsque des éléments XI de 4 bits sont utilisés et que m > 1, les éléments XI sont placés dans les octets de la manière suivante :



Bourrage : un champ de bourrage de 4 bits est présent lorsque PS = 0 et que m est impair. Le champ Bourrage est réglé à zéro à l'émission et ignoré à réception.

Élément 1, ..., élément n : Chaque élément correspond à un XI avec X = 1 dans XI 1, ..., XI m.

5.8.6.2 Type 1 de codage (schéma Insertion seule)



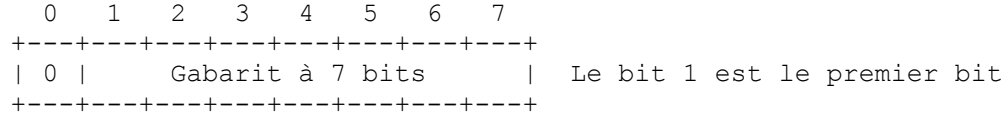
Sauf mention contraire explicite, les champs ont les mêmes valeurs et signification que pour le type 0 de codage.

ET : Le type de codage est un (1).

XI 1 : Lorsque PS = 0, le premier élément XI de 4 bits est placé ici. Lorsque PS = 1, le champ est réglé à zéro à l'émission et ignoré à réception.

ref_id : Identifiant de la liste de CSRC de référence utilisée lors de la compression de la liste. Ce sont les 8 bits de moindre poids du numéro de séquence RTP en mode R et gen_id (voir au paragraphe 5.8.2) en mode U/O.

Gabarit binaire d'insertion : C'est le gabarit binaire qui indique les positions où les nouveaux éléments sont à insérer. Voir le schéma Insertion seulement au paragraphe 5.8.3. Le gabarit binaire peut avoir l'un des deux formats suivants :



```
+---+---+---+---+---+---+---+---+
| 1 |                           |  Le bit 1 est le premier bit
+---+   Gabarit à 15 bits       +
|                               |  Le bit 7 est le dernier bit
+---+---+---+---+---+---+---+---+
```

Liste des XI : Ce sont les champs XI pour les éléments à insérer. Lorsque le gabarit binaire d'insertion a k uns, le nombre total de champs de XI est k. Lorsque PS = 1, tous les champs de XI sont dans la liste XI. Lorsque PS = 0, le premier champ de XI est dans le champ XI 1, et les k - 1 champs XI restants sont dans la liste des XI.

Bourrage : Présent lorsque PS = 0 et que k est pair.

élément 1, ..., élément n : Un élément pour chaque champ XI avec le bit X établi.

5.8.6.3 Type de codage 2 (schéma Retrait seulement)

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| ET=2  |GP |rés|     Compte    |
+---+---+---+---+---+---+---+---+
:            gen_id             :  1 octet, si GP = 1
+---+---+---+---+---+---+---+---+
|            ref_id             |
+---+---+---+---+---+---+---+---+
/  Gabarit binaire de retrait   /  1-2 octets
+---+---+---+---+---+---+---+---+
```

Sauf mention contraire explicite, les champs ont les mêmes valeurs et signification qu'au paragraphe 5.8.5.2.

ET : Le type de codage est 2.

rés : Réservé . Réglé à zéro à l'émission, ignoré à la réception.

Compte : Nombre d'éléments dans ref_list.

Gabarit binaire de retrait : Indique les éléments de ref_list à retirer afin d'obtenir la liste en cours. Voir en 5.8.3. Le gabarit binaire de retrait a le même format que le gabarit binaire d'insertion du paragraphe 5.8.6.2.

5.8.6.4 Type de codage 3 (schéma Retrait puis insertion)

Voir au paragraphe 5.8.3 la description du schéma Retrait puis insertion.

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| ET=3  |GP |PS |     XI 1      |
+---+---+---+---+---+---+---+---+
:            gen_id             :  1 octet, si GP = 1
+---+---+---+---+---+---+---+---+
|            ref_id             |
+---+---+---+---+---+---+---+---+
/    Gabarit binaire de retrait /  1-2 octets
+---+---+---+---+---+---+---+---+
/ Gabarit binaire d'insertion   /  1-2 octets
+---+---+---+---+---+---+---+---+
|           Liste XI            |  k octets, ou (k - 1) * 4 bits
/               --- --- --- ---/
|              :   Bourrage     :  si PS = 0 et k est pair
+---+---+---+---+---+---+---+---+
|                               |
/   élément 1, ..., élément n    /  variable
|                               |
+---+---+---+---+---+---+---+---+
```

Les champs dans cet en-tête ont les mêmes signification et formats qu'au paragraphe 5.8.5.2, sauf mention explicite contraire ci-dessous.

ET : Le type de codage est 3.

Gabarit binaire de retrait : Voir au paragraphe 5.8.6.3.

5.8.7 Couverture du CRC pour les en-têtes d'extension

Tous les champs d'en-tête d'extension sont CRC-STATIC, avec les exceptions suivantes qui sont CRC-DYNAMIC.
1) En-tête AH entier.
2) En-tête ESP entier.
3) Numéro de séquence en GRE, Somme de contrôle en GRE.

5.9 CRC de compression d'en-tête, couverture et polynômes

Cette section décrit comment calculer les CRC utilisés dans les en-têtes de paquet définis dans le présent document. (Noter qu'un autre type de CRC est défini pour les unités reconstruites au paragraphe 5.2.5.)

5.9.1 CRC de paquets IR et IR-DYN

Le CRC dans le paquet IR et IR-DYN est calculé sur le paquet IR ou IR-DYN entier, excluant la charge utile et incluant le CID ou tout octet Add-CID. Pour les besoins du calcul du CRC, le champ CRC dans l'en-tête est réglé à zéro.

Le contenu initial du registre CRC est à prérégler tout à un (1).

Le polynôme de CRC à utiliser pour le CRC à 8 bits est :

$$C(x) = 1 + x + x^2 + x^8$$

5.9.2 CRC dans les en-têtes compressés

Dans un en-tête compressé, le CRC est calculé sur tous les octets de l'en-tête d'origine entier, avant compression, de la manière suivante.

Les octets de l'en-tête sont classés soit comme CRC-STATIC, soit comme CRC-DYNAMIC, et le CRC est calculé sur :
1) les octets CRC-STATIC enchaînés de l'en-tête d'origine, placés dans le même ordre que celui dans lequel ils apparaissent dans l'en-tête d'origine, suivis par
2) les octets CRC-DYNAMIC enchaînés de l'en-tête d'origine, placés dans l'ordre dans lequel ils apparaissent dans l'en-tête d'origine.

L'intention est que l'état du calcul du CRC après 1) soit sauvegardé. Tant que les octets CRC-STATIC ne changent pas, le calcul du CRC aura alors seulement besoin de traiter les octets CRC-DYNAMIC.

Dans un en-tête normal RTP/UDP/IPv4, 25 octets sont CRC-STATIC et 15 sont CRC-DYNAMIC. Dans un en-tête RTP/UDP/IPv6 normal, 49 octets sont CRC-STATIC et 11 sont CRC-DYNAMIC. Cette technique va donc réduire la complexité du calcul du CRC d'environ 60 % pour RTP/UDP/IPv4 et d'environ 80 % pour RTP/UDP/IPv6.

Note : Chaque fois que les champs CRC-STATIC changent, le nouvel état de CRC sauvegardé après 1) est comparé à l'ancien état. Si les états sont identiques, le CRC ne peut pas saisir l'erreur qui consiste en ce que le décompresseur n'a pas mis à jour le contexte statique. En mode U/O, le compresseur DEVRAIT alors utiliser pendant un moment des types de paquet avec une autre longueur de CRC, pour laquelle il y a une différence dans l'état de CRC, pour assurer la détection d'erreur.

Le contenu initial du registre de CRC est préréglé tout à un.

Le polynôme à utiliser pour le CRC à trois bits est : $C(x) = 1 + x + x^3$

Le polynôme à utiliser pour le CRC à 7 bits est : $C(x) = 1 + x + x^2 + x^3 + x^6 + x^7$

Le CRC dans les en-têtes compressés est calculé sur l'en-tête d'origine entier, avant compression.

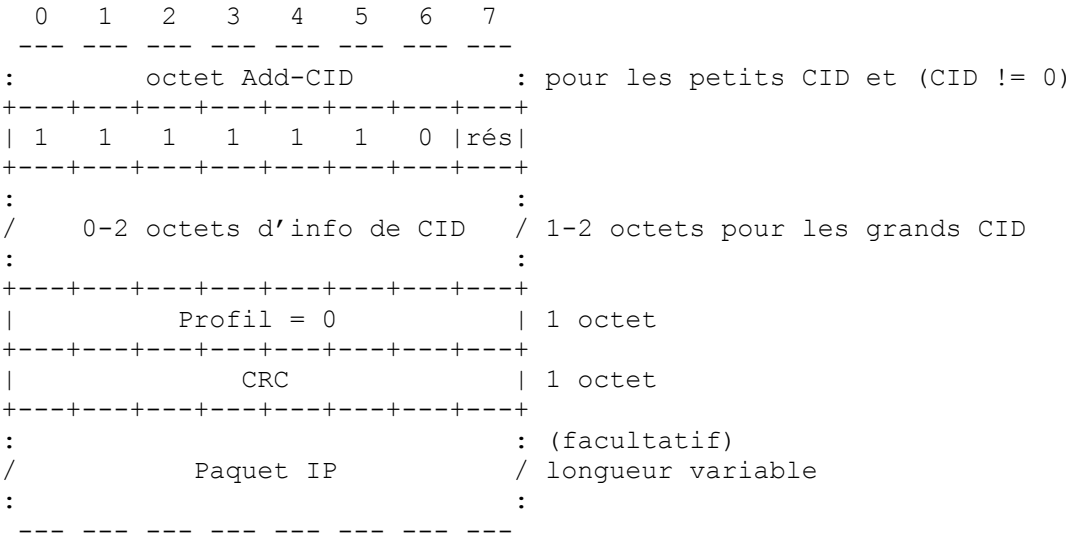
5.10 ROHC NON COMPRESSÉ – pas de compression (Profil 0x0000)

Dans ROHC, la compression n'a pas été définie pour toutes les sortes d'en-têtes IP. Le profil 0x0000 fournit un moyen d'envoyer des paquets IP sans les compresser. Cela peut être utilisé pour les fragments IP, les paquets RTCP, et en général pour tout paquet pour lequel la compression de l'en-tête n'a pas été définie, n'est pas possible due à des contraintes de ressources, ou n'est pas désirable pour quelque autre raison.

Après l'initialisation, la seule redondance pour l'envoi de paquets en utilisant le profil 0x0000 est la taille du CID. Lorsque les paquets non compressés sont fréquents, le profil 0x0000 devrait être associé à un CID de taille zéro ou un octet. Il n'est pas besoin d'associer le profil 0x0000 à plus d'un CID.

5.10.1 Paquet IR

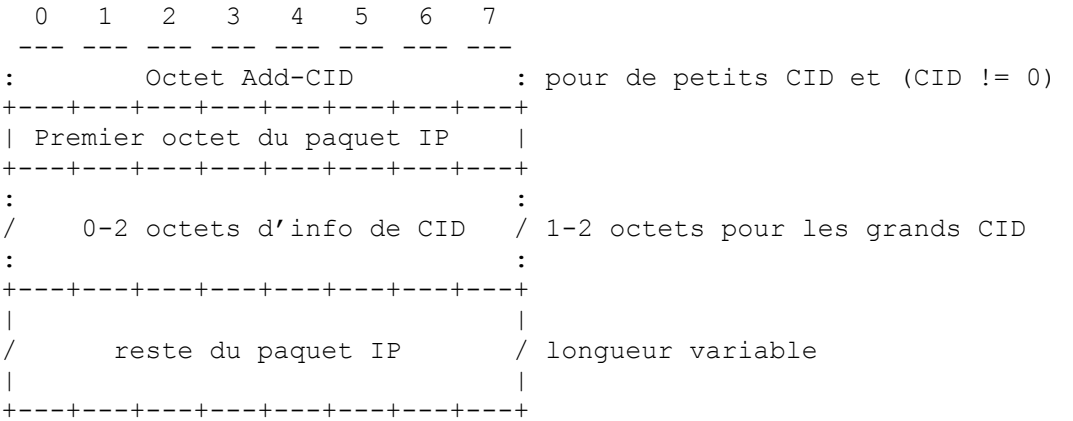
Le paquet d'initialisation (paquet IR) pour le profil 0x0000 a le format suivant :



rés : Toujours zéro.
Profil : 0.
CRC : CRC de 8 bits, calculé en utilisant le polynôme du paragraphe 5.9.1. Le CRC couvre le premier octet du paquet IR jusqu'à l'octet Profil du paquet IR, c'est-à-dire, il ne couvre pas le CRC lui-même ou le paquet IP.
Paquet IP : Un paquet IP non compressé peut être inclus dans le paquet IR. Le décompresseur détermine si le paquet IP est présent en considérant la longueur du paquet IR.

5.10.2 Paquet Normal

Un paquet Normal est un paquet IP normal plus des informations de CID. Lorsque le canal utilise de petits CID, et que le profil 0x0000 est associé à un CID > 0, un octet Add-CID est ajouté au paquet IP. Quand le canal utilise de grands CID, le CID est placé de telle sorte qu'il débute au second octet du paquet Normal.



Noter que le premier octet du paquet IP commence avec le schéma binaire 0100 (IPv4) ou 0110 (IPv6). Cela n'entre pas en conflit avec les types de paquet réservés. Donc, aucun bit en plus du CID n'est nécessaire. Le profil est raisonnablement à l'épreuve du futur car aucun problème n'apparaît jusqu'à IP version 14.

5.10.3 États et modes

Il y a deux modes dans le profil 0x0000 : le mode unidirectionnel et le mode bidirectionnel. En mode unidirectionnel, le compresseur répète périodiquement le paquet IR. En mode bidirectionnel, le compresseur ne répète jamais le paquet IR. Le compresseur et le décompresseur commencent toujours en mode unidirectionnel. Chaque fois qu'un retour est reçu, le compresseur passe en mode bidirectionnel.

Le compresseur peut être dans l'un ou l'autre des deux états : l'état IR ou l'état Normal. Il commence en état IR.

a) État IR : Seuls les paquets IR peuvent être envoyés. Après l'envoi d'un petit nombre de paquets IR (un seul lors d'un rafraîchissement) le compresseur passe au mode Normal.

b) État Normal : Seuls des paquets Normaux peuvent être envoyés. En mode unidirectionnel, le compresseur revient périodiquement à l'état IR. La longueur de la période dépend de la mise en œuvre, mais devrait être "très" longue. Le retard *(à accroissement)* exponentiel peut être utilisé.
c) Lorsque un retour est reçu dans tout état, le compresseur passe en mode bidirectionnel.

Le décompresseur peut être dans l'un des deux états : NO_CONTEXT ou FULL_CONTEXT. Il commence en NO_CONTEXT.

d) Lorsque un paquet IR est reçu dans l'état NO_CONTEXT, le décompresseur vérifie d'abord le paquet en utilisant le CRC. Si le paquet est bon, le décompresseur 1) passe à l'état FULL_CONTEXT, 2) livre le paquet IP aux couches supérieures s'il en est, 3) PEUT envoyer un ACK. Si le paquet n'est pas bon, il est éliminé sans autre action.

e) Lorsque un autre paquet est reçu dans l'état NO_CONTEXT, il est éliminé sans autre action.

f) Lorsque un paquet IR est reçu en état FULL_CONTEXT, le paquet est d'abord vérifié à l'aide du CRC. Si il est bon, le décompresseur 1) livre le paquet IP aux couches supérieures s'il en est, 2) PEUT envoyer un ACK. Si le paquet n'est pas bon, aucune action n'a lieu.

g) Lorsque un paquet Normal est reçu dans l'état FULL_CONTEXT, les informations de CID sont retirées et le paquet IP est livré aux couches supérieures.

5.10.4 Retour

La seule sorte de retour dans le profil 0x0000 est l'accusé de réception *(ACK)*. Le profil 0x0000 NE DOIT PAS être rejeté. Le profil 0x0000 DEVRAIT être associé au plus à un seul CID. Les ACK utilisent le format FEEDBACK-1 du paragraphe 5.2. La valeur de l'octet spécifique du profil dans l'accusé de réception FEEDBACK-1 est 0 (zéro).

5.11 UDP ROHC – compression UDP/IP non-RTP (profil 0x0002)

Les en-têtes UDP/IP n'ont pas un numéro de séquence qui a un aussi bon comportement que le numéro de séquence RTP. Pour UDP/IPv4, il y a un champ IP-ID qui peut recevoir un écho dans les informations de retour, mais lorsque aucun en-tête IPv4 n'est présent, une telle identification du retour devient problématique.

Donc, dans le profil UDP ROHC, le compresseur génère un numéro de séquence (SN) de 16 bits qui augmente de un pour chaque paquet reçu dans le flux de paquets. Ce numéro de séquence est donc d'un bon comportement relatif et peut servir de base à la plupart des mécanismes décrits pour RTP ROHC. Il est appelé ci-dessous SN ou SN UDP. Sauf mention contraire, les mécanismes de RTP ROHC sont utilisés aussi pour UDP ROHC, le SN UDP prenant le rôle du numéro de séquence RTP.

Le profil UDP ROHC utilise toujours p = -1 quand il interprète le SN, car il n'y aura pas de répétition ou de réarrangement du SN généré par le compresseur. L'intervalle d'interprétation commence donc toujours par (ref_SN + 1).

5.11.1 Initialisation

Le contexte statique pour les flux ROHC UDP peut être initialisé de l'une des deux façons suivantes :

1) En utilisant un paquet IR comme au paragraphe 5.7.7.1, où le profil est deux (2) et où la chaîne statique se termine par la partie statique d'un paquet UDP. Au compresseur, le SN UDP est initialisé à une valeur aléatoire lorsque le paquet IR est envoyé.

2) En réutilisant un contexte existant où la chaîne statique existante contient la partie statique d'un paquet UDP, par exemple, le contexte d'un flux compressé en utilisant RTP ROHC (profil 0x0001). Cela se fait avec un paquet IR-DYN (paragraphe 5.7.7.2) identifiant un profil 0x0002, où la chaîne dynamique correspond au préfixe de la chaîne statique existante qui se termine par l'en-tête UDP. Le SN UDP est initialisé au numéro de séquence RTP si le profil précédent était le profil 0x0001, et autrement à un nombre aléatoire.

Pour UDP ROHC, la partie dynamique d'un paquet UDP est différente de celle du paragraphe 5.7.7.5 : un champ de deux octets contenant le SN UDP est ajouté après le champ Somme de contrôle. Cela affecte le format des chaînes dynamiques dans les paquets IR et IR-DYN.

Note : 2) peut être utilisé pour les flux de paquets qui étaient initialement supposés être des flux RTP, de sorte que la compression avait été commencée avec le profil 0x0001, mais se sont ultérieurement trouvés n'être à l'évidence pas des flux RTP.

5.11.2 États et modes

UDP ROHC utilise les mêmes états et modes que RTP ROHC. Les transitions de mode et la logique d'état sont les mêmes sauf mention contraire explicite. Les mécanismes qui traitent des champs dans l'en-tête RTP (excepté le SN RTP) ne sont pas utilisés. Le SN UDP décompressé n'est jamais inclus dans un en-tête livré aux couches supérieures. Le SN UDP est utilisé à la place du SN RTP dans les retours.

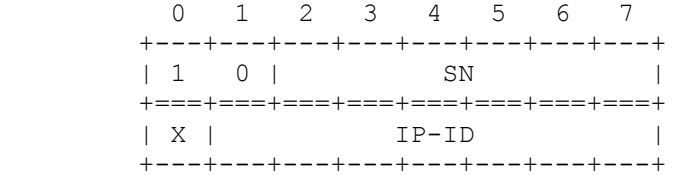
5.11.3 Types de paquet

Le format général d'un paquet UDP ROHC est le même que pour RTP ROHC (voir le début du paragraphe 5.7). Le bourrage et les CID sont les mêmes, comme l'est le type de paquet de retour (5.7.6.1) et le retour. Les paquets IR et IR-DYN (5.7.7) sont changés comme décrit en 5.11.1.

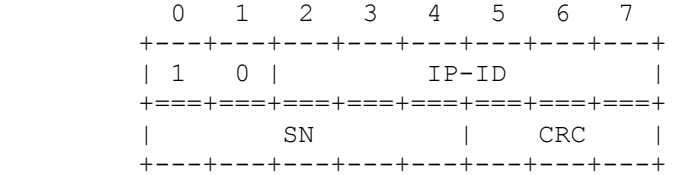
Le format général des paquets compressés est aussi le même, mais il y a des différences dans les formats et extensions spécifiques comme précisé ci-dessous.
Les différences sont causées par le retrait de toutes les informations spécifiques de RTP, sauf le SN RTP qui est remplacé par le SN UDP.

Sauf mention explicite contraire, les formats de paquet ci-dessous sont comme aux paragraphes 5.7.1-6.

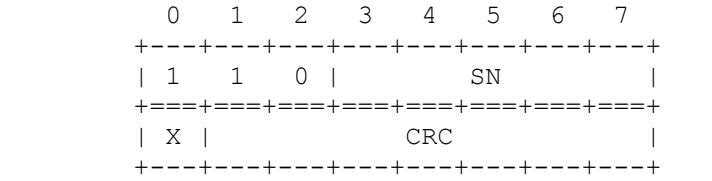
R-1
Le champ TS est remplacé par un champ IP-ID. Le fanion M a été incorporé à l'IP-ID. Le bit X a été déplacé. Les formats R-1-ID et R-1-TS ne sont pas utilisés.



UO-1
Le champ TS est remplacé par un champ IP-ID. Le fanion M a été incorporé au SN. Les formats UO-1-ID et UO-1-TS ne sont pas utilisés.



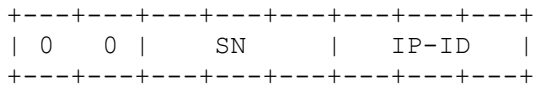
UOR-2 : Nouveau format :



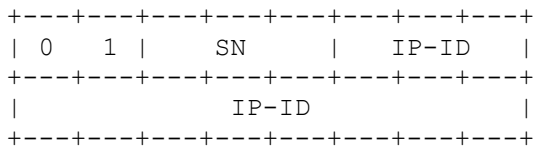
5.11.4. Extensions

Les extensions sont comme au paragraphe 5.7.5, avec les exceptions suivantes :

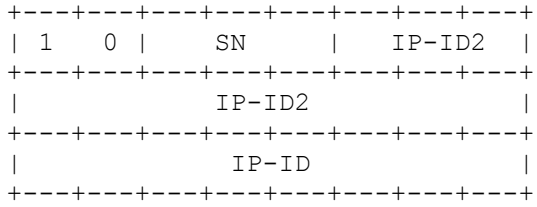
Extension 0 ::



Extension 1 :



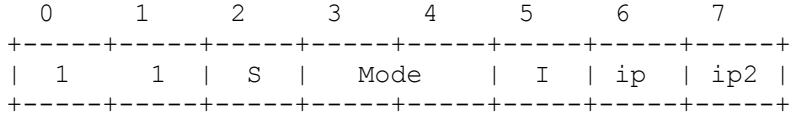
Extension 2 :



IP-ID2 : Pour champ IP-ID externe.

Extension 3 est la même que Extension 3 au paragraphe 5.7.5, avec les exceptions suivantes.

1) L'octet initial de fanion a le format suivant :



Mode : Remplace R-TS et Tsc du 5.7.5. Fournit des informations de mode comme l'était fait précédemment dans les fanions et champs d'en-tête RTP.

ip2 : Remplace le bit rtp du 5.7.5. Déplacé ici depuis l'octet des fanions d'en-tête IP interne.

2) Le bit qui était le fanion ip2 dans les fanions d'en-tête IP interne du 5.7.5 est réservé. Il est réglé à zéro à l'émission et ignoré en réception.

5.11.5 IP-ID

Traité comme dans RTP ROHC, mais le décalage est sur le SN UDP.

5.11.6 Retour

Le retour est comme dans RTP ROHC avec les exceptions suivantes :
1) Le SN UDP remplace le SN RTP dans le retour.
2) L'option CLOCK (5.7.6.6) n'est pas utilisée.
3) L'option JITTER (5.7.6.7) n'est pas utilisée.

5.12 ROHC ESP -- compression ESP/IP (Profil 0x0003)

Quand l'en-tête ESP est utilisé avec un algorithme de chiffrement autre que NUL, les sous en-têtes après l'en-tête ESP sont chiffrés et ne peuvent pas être compressés. Le profil 0x0003 est pour la compression de la chaîne d'en-têtes jusque et inclus l'en-tête ESP dans ce cas. Quand l'algorithme de chiffrement NUL est utilisé, d'autres profils peuvent être utilisés et pourraient donner de meilleurs taux de compression. Voir au paragraphe 5.8.4.3.

Ce profil est très similaire au profil UDP ROHC. Il utilise le numéro de séquence ESP comme base de la compression au lieu d'un nombre généré, mais est autrement très semblable à UDP ROHC. L'intervalle d'interprétation (valeur de p) pour le SN fondé sur ESP est comme avec RTP ROHC(profil 0x0001). À part cela, sauf mention explicite contraire ci-dessous, les mécanismes et les formats sont comme pour ROHC UDP.

5.12.1 Initialisation

Le contexte statique pour les flux ESP ROHC peut être initialisé d'une des deux façons suivantes :
1) en utilisant un paquet IR comme au paragraphe 5.7.7.1, où le profil est trois (3) et la chaîne statique se termine par la partie statique d'un en-tête ESP ;
2) en réutilisant un contexte existant, où la chaîne statique existante contient la partie statique d'un en-tête ESP. Cela se fait avec un paquet IR-DYN (paragraphe 5.7.7.2) identifiant le profil 0x0003, où la chaîne dynamique correspond au préfixe de la chaîne statique existante qui se termine avec l'en-tête ESP.

À la différence de UDP ROHC, aucun numéro de séquence supplémentaire n'est ajouté à la partie dynamique de l'en-tête ESP : le numéro de séquence ESP est le seul élément.

Note : 2) peut être utilisé pour des flux où la compression a été initiée avec l'hypothèse que le chiffrement NUL serait utilisé avec ESP. Lorsque il devient évident qu'un algorithme de chiffrement autre que NUL est utilisé, le compresseur peut envoyer un IR-DYN conformément à 2) pour passer au profil 0x0003 sans avoir à envoyer un paquet IR.

5.12.2 Types de paquet

Les types de paquet pour ESP ROHC sont les mêmes que pour ROHC UDP, sauf que le numéro de séquence ESP est utilisé à la place du numéro de séquence généré de ROHC UDP. L'en-tête ESP ne fait partie d'aucune liste compressée dans ESP ROHC.

6. Questions de mise en œuvre

Le présent document spécifie des mécanismes pour le protocole et laisse de nombreux détails sur l'utilisation de ces mécanismes à la charge des mises en œuvre. Cette section vise à donner des lignes directrices, des idées et des suggestions pour la mise en œuvre du schéma.

6.1 Décompression inverse

Ce paragraphe décrit un fonctionnement FACULTATIF de décompresseur pour réduire le nombre de paquets éliminés à cause d'un contexte invalide.

Une fois qu'un contexte devient invalide (par exemple, lorsque il survient plus de pertes consécutives de paquet qu'attendu) les paquets compressés suivants ne peuvent pas être immédiatement décompressés correctement. La décompression inverse vise à décompresser de tels paquets plus tard au lieu de les éliminer, en les mémorisant jusqu'à ce que le contexte ait été mis à jour et validé puis en tentant la décompression.

Soit la séquence de paquets mémorisé i, i + 1, ..., i + k, où i est le premier paquet et i + k est le dernier paquet avant que le contexte soit mis à jour. Le décompresseur va tenter de récupérer les paquets mémorisés en ordre inverse, c'est-à-dire, en commençant par i + k, et en les traitant à rebours jusqu'à i. Lorsque un paquet mémorisé a été reconstruit, il est vérifié qu'il est correct à l'aide de son CRC. Les paquets qui ne portent pas de CRC ne doivent pas être livrés aux couches supérieures. Les paquets dont le CRC est juste sont livrés aux couches supérieures dans leur ordre original, c'est-à-dire, i, i + 1, ..., i + k.

Noter que cette décompression inverse introduit des mises en mémoire tampon lorsque ils attendent que le contexte soit validé et cela introduit des délais supplémentaires. Donc, elle ne devrait être utilisée que quand un certain délai est acceptable. Par exemple, pour des paquets vidéo appartenant à la même trame vidéo, le délai des arrivées des paquets ne cause pas de délai de présentation. Les applications de direct sensibles au délai peuvent aussi être tolérante à de tels délais. Si le décompresseur ne peut pas déterminer si l'application peut tolérer le délai, il ne devrait pas effectuer la décompression inverse.

Ce qui suit illustre la procédure de décompression plus en détails :

1. Le décompresseur mémorise les paquets compressés qui ne peuvent pas être décompressés correctement à cause d'un contexte invalide.

2. Lorsque le décompresseur a reçu un paquet de mise à jour du contexte et que le contexte a été validé il procède à la récupération du dernier paquet mémorisé. Après la décompression, le décompresseur vérifie que l'en-tête reconstruit est correct en utilisant le CRC.

3. Si le CRC indique la réussite de la décompression, le décompresseur mémorise le paquet complet et tente de décompresser le paquet précédant. De cette façon, les paquets mémorisés sont récupérés en ordre inverse jusqu'à ce qu'il ne reste aucun paquet compressé. Pour chaque paquet, le décompresseur vérifie que les en-têtes décompressés sont corrects en utilisant le CRC de compression d'en-tête.

4. Si le CRC indique un paquet incorrectement décompressé, la tentative de décompression inverse DOIT se terminer et tous les paquets compressés restants DOIVENT être éliminés.

5. Finalement, le décompresseur transmet tous les paquets décompressés correctement aux couches supérieures dans leur ordre d'origine.

6.2 RTCP

RTCP est le protocole de contrôle du protocole de transport en temps réel (RTP) [RFC1889]. RTCP se fonde sur la transmission périodique de paquets de contrôle à tous les participants à une session, en utilisant le même mécanisme de distribution que pour les paquets de données. Sa principale fonction est de fournir des retours des receveurs de données sur la qualité de la distribution des données. Les informations de retour peuvent être utilisées pour des questions qui se rapportent aux fonctions de contrôle de l'encombrement, et directement utiles pour le contrôle des codages adaptatifs.

Dans une session RTP, il y aura deux types de flux de paquets : un avec l'en-tête RTP et les données d'application, et un avec les informations de contrôle RTCP. La différence entre les flux au niveau du transport est dans le numéro d'accès UDP : le numéro d'accès RTP est toujours pair, le numéro d'accès RTCP est ce nombre plus un et donc toujours impair ([RFC1889], section 10). Une mise en œuvre de compresseur d'en-tête ROHC dispose de plusieurs moyens pour traiter le flux RTCP :

1. Une entité compresseur/décompresseur portant les deux types de flux sur le même canal, utilisant les CID pour les distinguer. Pour envoyer un seul flux RTP avec ses paquets RTCP sur un canal, il est des plus efficace de régler LARGE_CIDS à faux, d'envoyer les paquets RTP avec les CID 0 impliqués et d'utiliser le mécanisme Add-CID pour envoyer les paquets RTCP.
2. Deux entités compresseur/décompresseur, une pour RTP et une autre pour RTCP, portant les deux types de flux sur des canaux distincts. Cela signifie qu'ils ne vont pas partager le même espace de nombres de CID.

Les en-têtes RTCP peuvent simplement être envoyés non compressés en utilisant le profil 0x0000. Plus efficacement, on peut utiliser la compression UDP ROHC (profil 0x0002).

6.3 Paramètres et signaux de mise en œuvre

Une mise en œuvre de ROHC peut avoir deux sortes de paramètres : des paramètres de configuration qui sont obligatoires et doivent être négociés entre les homologues compresseur et décompresseur, et les paramètres de mise en œuvre qui sont facultatifs et, quand ils sont utilisés, stipulent comment va fonctionner une mise en œuvre de ROHC.

Les paramètres de configuration sont obligatoires et doivent être négociés entre le compresseur et le décompresseur, de sorte qu'ils aient les mêmes valeurs au compresseur et au décompresseur, voir au paragraphe 5.1.1.

Les paramètres de mise en œuvre rendent possible à une entité externe de stipuler comment la mise en œuvre d'un compresseur ou décompresseur ROHC devrait fonctionner. Les paramètres de mise en œuvre ont une signification locale, sont d'utilisation facultative et ne sont donc pas nécessaires pour la négociation entre compresseur et décompresseur. Noter que cela n'empêche pas la signalisation ou la négociation de paramètres de mise en œuvre en utilisant des fonctions de couche inférieure afin de régler la façon de fonctionner d'une mise en œuvre ROHC. Certains paramètres de mise en œuvre ne sont valides qu'au compresseur ou qu'au décompresseur. Les paramètres de mise en œuvre peuvent de plus être divisés en paramètres qui permettent à une entité externe de décrire la façon dont la mise en œuvre devrait fonctionner, et en paramètres qui permettent à une entité externe de déclancher un événement spécifique, c'est-à-dire, des signaux.

6.3.1 Paramètres de mise en œuvre ROHC au compresseur

CONTEXT_REINITIALIZATION – signal
Ce paramètre déclanche une réinitialisation du contexte entier au décompresseur, des deux parties, statique et dynamique. Le compresseur DOIT, lorsque CONTEXT_REINITIALIZATION est déclanché, sauvegarder l'état IR et complètement réinitialiser le contexte par l'envoi de paquets IR avec les deux parties de chaîne statique et dynamique couvrant les en-têtes non compressés entiers jusqu'à ce qu'il soit raisonnablement sûr que les contextes du décompresseur sont réinitialisés. La réinitialisation du contexte DOIT être faite pour tous les contextes au compresseur. Ce paramètre peut, par exemple, être utilisé pour faire la relocalisation du contexte lorsque, par exemple, un transfert intercellulaire résulte en un changement de point de compression dans le réseau d'accès radio.

NO_OF_PACKET_SIZES_ALLOWED *(nombre de tailles de paquet admises)* – valeur : entier positif
Ce paramètre peut être réglé par une entité externe pour spécifier le nombre de tailles de paquet que peut utiliser une mise en œuvre ROHC. Cependant, le paramètre ne peut être utilisé que si PACKET_SIZES n'est pas utilisé par une entité externe. Avec ce paramètre établi, la mise en œuvre ROHC au compresseur NE DOIT PAS utiliser plus de tailles de paquet différentes que ce que la valeur de ce paramètre stipule. La mise en œuvre ROHC doit elle-même être capable de déterminer quelles tailles de paquet seront utilisées et de les décrire à une entité externe en utilisant PACKET_SIZES_USED. On devrait noter qu'une taille de paquet pourrait être utilisée pour plusieurs formats d'en-tête, et que le nombre de tailles de paquet peut être réduit en employant le bourrage et la segmentation.

NO_OF_PACKET_SIZES_USED *(nombre de tailles de paquet utilisées)* – valeur : entier positif
Ce paramètre est réglé par la mise en œuvre ROHC pour indiquer combien de tailles de paquet elle va réellement utiliser. Il peut être réglé à une grande valeur pour indiquer qu'aucune tentative particulière n'est faite pour minimiser ce nombre.

PACKET_SIZES_ALLOWED *(tailles de paquet admises)* – valeur : liste d'entiers positifs (octets)
Ce paramètre, s'il est établi, gouverne les tailles de paquet en octets qui peuvent être utilisées par la mise en œuvre ROHC. Donc, les tailles de paquet qui ne sont pas dans l'ensemble de valeurs de ce paramètre NE DOIVENT PAS être utilisées. Ainsi, une entité externe peut obliger une mise en œuvre ROHC à produire des tailles de paquet qui conviennent mieux aux couches inférieures préconfigurées. Si ce paramètre est utilisé pour stipuler quelles tailles de paquet peut utiliser une mise en œuvre ROHC, les règles suivantes s'appliquent :
- Un paquet assez grand pour contenir l'en-tête IR entier (chaînes statique et dynamique) DOIT faire partie de l'ensemble de tailles, sauf si MRRU est réglé à une valeur assez grande pour permettre la segmentation.
- La taille de paquet qui va vraisemblablement être la plus fréquemment utilisée dans l'état SO DEVRAIT faire partie de l'ensemble.
- La taille de paquet qui va vraisemblablement être la plus fréquemment utilisée dans l'état FO DEVRAIT faire partie de l'ensemble.

PACKET_SIZES_USED *(tailles de paquet utilisées)* – valeurs : ensemble d'entiers positifs (octets)
Ce paramètre décrit quelles tailles de paquet utilise une mise en œuvre ROHC si NO_OF_PACKET_SIZES_ALLOWED ou PACKET_SIZES_ALLOWED est utilisé par une entité externe pour stipuler combien de tailles de paquet devrait utiliser une mise en œuvre ROHC. Les informations sur les tailles de paquet utilisées (en octets) dans ce paramètre peuvent alors être utilisés pour configurer les couches inférieures.

PAYLOAD_SIZES *(tailles de charge utile)* – valeurs : ensemble de valeurs d'entiers positifs (octets)
Ce paramètre est réglé par une entité externe qui veut faire usage du paramètre PACKET_SIZES_USED pour indiquer quelles tailles de charge utile peuvent être espérées.

Lorsque une mise en œuvre ROHC a un ensemble limité de tailles de paquet admises, et que le format d'en-tête le plus préférable a une taille qui ne fait pas partie de l'ensemble, elle a les options suivantes :
- Choisir le plus proche format d'en-tête dans l'ensemble admis. C'est probablement le choix le plus efficace.
- Utiliser le format d'en-tête le plus préférable comme si il n'y avait pas de restriction de taille, et ajouter des octets de bourrage pour compléter un paquet à la prochaine taille supérieure de l'ensemble des tailles admises.
- Utiliser la segmentation pour fragmenter le paquet en morceaux qui vont constituer des paquets des tailles permises

(éventuellement après l'ajout d'un bourrage au dernier segment).

On devrait noter que même si les deux derniers paramètres introduisent la possibilité de restreindre le nombre de tailles de paquet utilisées, de telles restrictions auront un impact négatif sur les performances de compression.

6.3.2 Paramètres de mise en œuvre de ROHC au décompresseur

MODE – valeurs : [mode U, mode O, mode R]
Ce paramètre déclanche une transition de mode en utilisant le mécanisme décrit à la Section 5 lorsque le paramètre change de valeur, c'est-à-dire, au mode U (mode Unidirectionnel), mode O (mode Bidirectionnel optimiste) ou au mode R (mode Bidirectionnel fiable). La transition de mode est faite du mode en cours au nouveau mode comme signalé par le paramètre de mise en œuvre. Par exemple, si le mode en cours est le mode Bidirectionnel optimiste, MODE devrait avoir la valeur mode O. Si le MODE est changé en mode R, une transition de mode DOIT être faite du mode Bidirectionnel optimiste au mode Bidirectionnel fiable. MODE ne devrait pas seulement servir de déclencheur pour les transitions de mode, mais aussi rendre visible en quel mode fonctionne ROHC.

CLOCK_RESOLUTION *(résolution d'horloge)* – valeur : entier non négatif
Ce paramètre indique la résolution d'horloge système en unités de millisecondes. Une valeur de zéro (0) signifie qu'il n'y a pas d'horloge disponible. Si il n'est pas à zéro, ce paramètre permet au décompresseur d'utiliser la compression d'horodatage (TS) fondée sur le temporisateur (paragraphe 4.5.4) et la détection de retour à zéro du numéro de séquence (SN) (paragraphe 5.3.2.2.4). Dans ce cas, sa valeur spécifique est aussi significative de la correction des algorithmes.

REVERSE_DECOMPRESSION_DEPTH *(profondeur de décompression inverse)* – valeur : entier non négatif
Ce paramètre détermine si la décompression inverse telle que décrite au paragraphe 6.1 devrait ou non être utilisée, et si elle est utilisée, dans quelle mesure. La valeur indique le nombre maximum de paquets qui peuvent être mis en mémoire tampon, et donc éventuellement subir la décompression inverse par le décompresseur. Une valeur de zéro (0) signifie que la décompression inverse NE DOIT PAS être utilisée.

6.4 Traitement des limitations de ressource au décompresseur

Sur une liaison point à point, les deux nœuds peuvent se mettre d'accord sur le nombre de sessions compressées qu'elles sont prêtes à prendre en charge sur cette liaison. Il peut, cependant, n'être pas possible au décompresseur de prédire précisément quand il va se trouver à bout de ressources. ROHC permet que le nombre négocié de contextes soit supérieur à celui qui pourrait être traité dans le plus mauvais cas. Alors, lorsque les ressources de contexte sont consommées, une tentative d'établir un nouveau contexte peut être rejetée par le décompresseur, en utilisant l'option REJECT de la charge utile de retour.

À réception d'une option REJECT, le compresseur DEVRAIT attendre un moment avant de tenter de compresser des flux supplémentaires destinés au nœud qui a rejeté.

6.5 Structures de mise en œuvre

Cette section donne du matériel explicatif sur les structures de données qu'une mise en œuvre ROHC va avoir à entretenir sous une forme ou une autre. Elle n'est pas destinée à constituer une contrainte pour les mises en œuvre.

6.5.1 Contexte de compresseur

Le contexte de compresseur consiste en une partie statique et une partie dynamique. Le contenu de la partie statique est le même que la chaîne statique définie au paragraphe 5.7.7. La partie dynamique consiste en multiples éléments qui peuvent être répartis en quatre catégories.

a) Fenêtre glissante (SW, *Sliding Window*)
b) Tableau de traduction (TT)
c) Fanion
d) Champ

Ces éléments peuvent être communs à tous les modes ou spécifiques du mode. Le tableau qui suit résume tous ces éléments.

	Commun à tous les modes	**Spécifique du mode R**	**Spécifique du mode U/O**
SW	GSW	R_CSW	UO_CSW
		R_IESW	UO_IESW
TT		R_CTT	UO_CTT
		R_IETT	UO_IETT
Fanions	Somme de contrôle UDP	ACKED	
	TSS, TIS		
	RND, RND2		
	NBO, NBO2		
Champs	Profil	CSRC_REF_ID	
	C_MODE		_ID
	C_STATE		CSRC_GEN_COUNT
	C_TRANS		IPEH_REF_ID
	TS_STRIDE (si TSS = 1)		IPEH_GEN_ID
	TS_OFFSET (si TSS = 1)		IPEH_GEN_COUNT
	TIME_STRIDE (si TIS = 1)		
	CURR_TIME (si TIS = 1)		
	MAX_JITTER_CD (si TIS = 1)		
	LONGEST_LOSS_EVENT(O)		
	CLOCK_RESOLUTION(O)		
	MAX_JITTER(O)		

1) GSW : Fenêtre glissante générique W_LSB
   Chaque élément dans GSW consiste en tous les champs dynamiques dans la chaîne dynamique (définie au paragraphe 5.7.7) plus les champs spécifiés en a) mais excluant les champs spécifiés en b).
   a) Heure d'arrivée du paquet (si TIS = 1) Horodatage RTP adapté (si TSS = 1) (facultatif)Offset_i (si RND = 0) (facultatif)
   b) Somme de contrôle UDP, Pad d'horodatage, liste de CSRC, en-têtes d'extension IPv6

2) R_CSW : Fenêtre glissante de CSRC en mode R
   R_IESW : Fenêtre glissante d'en-têtes d'extension IPv6 en mode R
   UO_CSW : Fenêtre glissante de CSRC en mode U/O
   UO_IESW : Fenêtre glissante d'en-têtes d'extension IPv6 en mode U/O
   Chaque élément dans R_CSW, R_IESW, UO_CSW et UO_IESW est défini au paragraphe 6.5.3.

3) R_CTT : Tableau de traduction de CSRC en mode R
   R_IETT : Tableau de traduction d'en-têtes d'extension IPv6 en mode U/O
   UO_CTT : Tableau de traduction de CSRC en mode U/O
   UO_IETT : Tableau de traduction d'en-têtes d'extension IPv6 en mode U/O
   Chaque élément dans R_CTT et R_IETT est défini au paragraphe 5.8.1.1.
   Chaque élément dans UO_CTT et UO_IETT est défini au paragraphe 5.8.1.2.

4) ACKED : Indique si le décompresseur a déjà accusé réception ou non.

5) CURR_TIME : Valeur de l'heure en cours (utilisée pour la relocalisation de contexte lorsque la compression d'horodatage fondée sur la temporisation est utilisée)

6) Tous les autres fanions sont définis ailleurs dans le document ROHC.

6.5.2 Contexte de décompresseur

Le contexte de décompresseur consiste en une partie statique et une partie dynamique. Le contenu de la partie statique est le même que la chaîne statique définie au paragraphe 5.7.7. La partie dynamique consiste en plusieurs éléments, dont l'un est l'en-tête de référence non statique qui inclut tous les champs non statique. Ces champs non statiques sont les champs dans la chaîne dynamique définie au paragraphe 5.7.7, excluant Somme de contrôle UDP et Pas d'horodatage. Tous les éléments restants peuvent être rangés en quatre types :

a) Fenêtre glissante (SW, *Sliding Window*)
b) Tableau de traduction (TT)
d) Fanion
e) Champ

Ces éléments peuvent être spécifiques du mode ou communs à tous les modes. Le tableau suivant résumé tous ces

éléments.

	Commun à tous les modes	Spécifique du mode R	Spécifique du mode U/O
SW	R_CSW	O_CSW	
		_IESW	O_IESW
TT		R_CTT	UO_CTT
		R_IETT	UO_IETT
Fanion	Somme de contrôle UDP		ACKED
	TSS, TIS		
	RND, RND2		
	NBO, NBO2		
Champs	Profil		CSRC_GEN_ID
	D_MODE		IPEH_GEN_ID
	D_STATE		PRE_SN_V_REF
	D_TRANS		
	TS_STRIDE (si TSS = 1)		
	TS_OFFSET (si TSS = 1)		
	TIME_STRIDE (si TIS = 1)		
	PKT_ARR_TIME (si TIS = 1)		
	LONGEST_LOSS_EVENT(O)		
	CLOCK_RESOLUTION(O)		
	MAX_JITTER(O)		

1) ACKED : Indique si un accusé de réception a déjà été envoyé ou non.
2) PKT_ARR_TIME : Heure d'arrivée du paquet décompressé le plus récent et vérifié avec son CRC.
   PRE_SN_V_REF : Le numéro de séquence du paquet vérifié avant le paquet vérifié le plus récent.
   CSRC_GEN_ID : gen_id de CSRC du paquet le plus récent reçu.
   IPEH_GEN_ID : gen_id d'en-tête d'extension IPv6 du paquet reçu le plus récent.
3) Les éléments restants sont comme défini dans le contexte de compresseur.

6.5.3 Compression de liste : fenêtre glissante en mode R et en mode U/O

En compression de liste en mode R (voir au paragraphe 5.8.2.1) chaque entrée dans la fenêtre glissante, aussi bien du côté du compresseur que du côté du décompresseur, a la structure suivante :

```
+------------------------+--------+----------------+
| Numéro de séquence RTP | icount | liste d'indice |
+------------------------+--------+----------------+
```

La liste d'indice du tableau contient une liste d'indices. Chacun de ces indices correspond à l'élément dans la liste originale portée dans le paquet identifié par le numéro de séquence RTP. La transposition entre l'indice et l'élément est identifiée dans le tableau de traduction. Le champ icount porte le nombre d'indices de la liste d'indices suivante.

En compression de liste en mode U/O, chaque entrée dans la fenêtre glissante des deux côtés compresseur et décompresseur a la structure suivante :

```
+--------+--------+-----------------+
| Gen_id | icount | liste d'indices |
+--------+--------+-----------------+
```

Les champs icount et liste d'indices sont les mêmes que défini en mode R. Au lieu d'utiliser le numéro de séquence RTP pour identifier chaque entrée, le Gen_id est inclus dans la fenêtre glissante en mode U/O.

7. Considérations pour la sécurité

Comme le chiffrement élimine la redondance que les schémas de compression d'en-têtes essayent d'exploiter, il y a quelques incitations à renoncer au chiffrement des en-têtes afin d'activer le fonctionnement sur les liaisons à faible bande passante. Cependant, pour les cas où le chiffrement des données (et pas des en-têtes) est suffisant, RTP spécifie une méthode de chiffrement de remplacement dans laquelle seule la charge utile RTP est chiffrée et les en-têtes sont laissés en clair. Cela permettrait encore à la compression d'en-tête d'être appliquée.

La compression ROHC est transparente à l'égard des champs Numéro de séquence RTP et Horodatage RTP, de sorte que

les valeurs de ces champs peuvent être utilisées comme base de schémas de chiffrement de charge utile (par exemple, pour le calcul d'un vecteur d'initialisation).

Un compresseur d'en-têtes malveillant ou qui fonctionne mal pourrait être cause que le décompresseur d'en-tête reconstitue des paquets qui ne correspondent pas aux paquets originaux mais ait encore des en-têtes IP, UDP et RTP valides et éventuellement aussi des sommes de contrôle UDP valides. Une telle corruption peut être détectée avec des mécanismes d'authentification et d'intégrité de bout en bout qui ne seront pas affectés par la compression. De plus, ce schéma de compression d'en-tête utilise une somme de contrôle interne pour la vérification des en-têtes reconstruits. Cela réduit la probabilité de produire des en-têtes décompressés qui ne correspondent pas aux originaux sans que cela soit remarqué.

Les attaques de déni de service sont possibles si un intrus peut introduire (par exemple) des paquets STATIC, DYNAMIC ou FEEDBACK bogués sur la liaison et causer par là une réduction de l'efficacité de compression. Cependant, un intrus qui a la possibilité d'injecter des paquets arbitraires à la couche liaison de cette manière soulève des problèmes de sécurité supplémentaires qui surpassent ceux qui se rapportent à l'utilisation de la compression d'en-tête.

8. Considérations relatives à l'IANA

L'identifiant de profil ROHC est un entier non négatif. Dans de nombreux protocoles de négociation, il sera représenté par une valeur de 16 bits. De la façon dont l'identifiant de profil est abrégé dans les paquets ROHC, les 8 bits de moindre poids de l'identifiant de profil ont une signification particulière: Deux identifiants de profil avec huit LSB identiques ne devraient être alloués que si celui qui a le numéro le plus élevé est destiné à subroger celui de numéro inférieur. Pour souligner cette relation, les identifiants de profil devraient être donnés en hexadécimal (comme dans 0x1234, qui par exemple subrogerait 0x0A34).

Suivant les politiques exposées dans la [RFC2434], la politique de l'IANA pour l'allocation de nouvelles valeurs pour les identifiants de profil devra être Spécification exigée : les valeurs et leur signification doivent être documentées dans une RFC ou dans quelque autre référence permanente et directement disponible, dans un détail suffisant pour que l'interopérabilité entre des mises en œuvre indépendantes soit possible. Dans les 8 LSB, la gamme de 0 à 127 est réservée pour les spécifications de l'IETF en cours de normalisation ; la gamme de 128 à 254 est disponible pour d'autres spécifications qui satisfont à cette exigence (telles que les RFC pour information). La valeur de LSB 255 est réservée pour une future extensibilité de la présente spécification.

Les identifiants de profil suivants sont déjà alloués :

Identifiant de profil	**Document**	**Usage**
0x0000	RFC3095	ROHC non compressé
0x0001	RFC3095	ROHC RTP
0x0002	RFC3095	ROHC UDP
0x0003	RFC3095	ROHC ESP

9. Remerciements

Les schémas de compression d'en-têtes précédents décrits dans [CJHC], [RFC2507], et [RFC2508] ont été d'importantes sources d'idées et de renseignements.

L'éditeur tient à étendre ses chaleureux remerciements à Mikael Degermark, qui a en fait effectué une part du travail d'édition, et à Peter Eriksson, qui a fait une relecture complète du document, améliorant significativement sa cohérence rédactionnelle. Bien sûr, tous les problèmes rédactionnels restants ont été introduits par l'éditeur.

Merci à Andreas Jonsson (Lulea University), qui a soutenu ce travail par son étude des schémas de changement de champs d'en-tête.

Finalement, ce travail n'aurait pu réussir sans les avis continuels en navigant dans les projets de l'IETF, nourris de commentaires à la fois rédactionnels et techniques, des directeurs de la zone Transport de l'IETF, Allison Mankin et Scott Bradner.

10. Considérations sur les revendications de droits de propriété intellectuelle

Une revendication de droits de propriété intellectuelle a été notifiée à l'IETF à l'égard de tout ou partie de la spécification

contenue dans le présent document. Pour plus d'informations, consulter la liste en ligne de revendications de droits.

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autres droits qui pourraient être revendiqués 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'ISOC au sujet des droits dans les documents de l'ISOC figurent dans les BCP 78 et BCP 79.

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 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, licences ou applications de licence, ou autres droits de propriété qui pourraient couvrir les technologies qui peuvent être nécessaires pour mettre en œuvre la présente norme. Prière d'adresser les informations à l'IETF à ietf-ipr@ietf.org.

11. Références

11.1 Références normatives

[RFC0768] J. Postel, "Protocole de datagramme d'utilisateur (UDP)", (STD 6), 28 août 1980.

[RFC0791] J. Postel, éd., "Protocole Internet - Spécification du protocole du programme Internet", STD 5, septembre 1981.

[RFC1662] W. Simpson, éditeur, "PPP en trames de style HDLC", STD 51, juillet 1994.

[RFC1700] J. Reynolds et J. Postel, "Numéros alloués", STD 2, octobre 1994. *(Historique, voir www.iana.org))*

[RFC1889] H. Schulzrinne, S. Casner, R. Frederick et V. Jacobson, "RTP : protocole de transport pour applications en temps réel", janvier 1996. *(Obsolète, voir* RFC3550 *STD64)*

[RFC2402] S. Kent et R. Atkinson, "En-tête d'authentification IP", novembre 1998. (*Obsolète, voir* RFC4302, 4305)

[RFC2004] C. Perkins, "Encapsulation minimale au sein de IP", octobre 1996. *(P.S.)*

[RFC2406] S. Kent et R. Atkinson, "Encapsulation de charge utile de sécurité IP (ESP)", novembre 1998. *(Obsolète, voir* RFC4303)

[RFC2410] R. Glenn, S. Kent, "L'algorithme de chiffrement NULL et son utilisation avec IPsec", novembre 1998. *(P.S.)*

[RFC2460] S. Deering et R. Hinden, "Spécification du protocole Internet, version 6 (IPv6) ", décembre 1998. *(MàJ par les* RFC5095, 6564 *; D.S)*

[RFC2784] D. Farinacci, T. Li, S. Hanks, D. Meyer et P. Traina, "Encapsulation d'acheminement générique (GRE)", mars 2000.

[RFC2890] G. Dommety, "Extensions de clé et de numéro de séquence à GRE", septembre 2000. *(P.S.)*

11.2. Références pour information

[CRTPC] Degermark, M., Hannu, H., Jonsson, L.E., Svanbro, K., "Evaluation of CRTP Performance over Cellular Radio Networks", IEEE Personal Communication Magazine, Volume 7, numéro 4, pp. 20-25, août 2000.

[RFC1144] V. Jacobson, "Compression des en-têtes TCP/IP pour les liaisons série à faible débit", février 1990.

[RFC2434] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", BCP 26, octobre, 1998. *(Obsolète, voir* RFC5226*)*

[RFC2507] M. Degermark, B. Nordgren, S. Pink, "Compression d'en-tête IP", février 1999. *(P.S.)*

[RFC2508] S. Casner, V. Jacobson, "Compression d'en-têtes IP/UDP/RTP pour liaisons séries à bas débit", février 1999. *(P.S.)*

[RFC3096] M. Degermark, éd., "Exigences pour la compression d'en-tête robuste IP/UDP/RTP", juillet 2001. *(Info.)*

[RFC3409] K. Svanbro, "Lignes directrices pour la compression robuste d'en-tête RTP/UDP/IP de couche inférieure", décembre 2002. *(Information)*

12. Adresse des auteurs

Carsten Bormann, éditeur
Universitaet Bremen TZI
Postfach 330440
D-28334 Bremen, Germany
téléphone : +49 421 218 7024
Fax : +49 421 218 7000
mél : cabo@tzi.org

Carsten Burmeister
Panasonic European Laboratories GmbH
Monzastr. 4c
63225 Langen, Germany
Phone: +49-6103-766-263
Fax: +49-6103-766-166
EMail: burmeister@panasonic.de

Mikael Degermark
The University of Arizona
Dept of Computer Science
P.O. Box 210077
Tucson, AZ 85721-0077, USA
Phone: +1 520 621-3498
EMail: micke@cs.arizona.edu

Hideaki Fukushima
Matsushita Electric Industrial Co.,
Ltd006, Kadoma, Kadoma City,
Osaka, Japan
Phone: +81-6-6900-9192
Fax: +81-6-6900-9193
EMail: fukusima@isl.mei.co.jp

Hans Hannu
Box 920
Ericsson Erisoft AB
SE-971 28 Lulea, Sweden
Phone: +46 920 20 21 84
Fax: +46 920 20 20 99
EMail: hans.hannu@ericsson.com

Lars-Erik Jonsson
Box 920
Ericsson Erisoft AB
SE-971 28 Lulea, Sweden
Phone: +46 920 20 21 07
Fax: +46 920 20 20 99
EMail: lars-erik.jonsson@ericsson.com

Rolf Hakenberg
Panasonic European Laboratories GmbH
Monzastr. 4c
63225 Langen, Germany
Phone: +49-6103-766-162
Fax: +49-6103-766-166
mél : hakenberg@panasonic.de

Tmima Koren
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134, USA
Phone: +1 408-527-6169
mél : tmima@cisco.com

Khiem Le
2-700
Mobile Networks Laboratory
Nokia Research Center
6000 Connection Drive
Irving, TX 75039, USA
mél : khiem.le@nokia.com

Zhigang Liu
2-700
Mobile Networks Laboratory
Nokia Research Center
6000 Connection Drive
Irving, TX 75039, USA
mél : zhigang.liu@nokia.com

Anton Martensson
Ericsson Radio Systems AB
Torshamnsgatan 23
SE-164 80 Stockholm, Sweden
Phone: +46 8 404 3881
Fax: +46 8 757 5550
mél : anton.martensson@era.ericsson.se

Akihiro Miyazaki
Matsushita Electric Industrial Co., Ltd
1006, Kadoma, Kadoma City, Osaka, Japan
Phone: +81-6-6900-9192
Fax: +81-6-6900-9193
mél : akihiro@isl.mei.co.jp

Krister Svanbro
Box 920
Ericsson Erisoft AB
SE-971 28 Lulea, Sweden
Phone: +46 920 20 20 77
Fax: +46 920 20 20 99
mél : krister.svanbro@ericsson.com

Thomas Wiebke
Panasonic European Laboratories
Monzastr. 4c
63225 Langen, Germany
Phone: +49-6103-766-161
Fax: +49-6103-766-166
mél : wiebke@panasonic.de

Takeshi Yoshimura
NTT DoCoMo, Inc.
3-5, Hikarinooka
Yokosuka, Kanagawa, 239-8536, Japan
Phone: +81-468-40-3515
Fax: +81-468-40-3788
mél l: yoshi@spg.yrp.nttdocomo.co.jp

Haihong Zheng
2-700
Mobile Networks Laboratory
Nokia Research Center
6000 Connection Drive
Irving, TX 75039, USA
mél : haihong.zheng@nokia.com

Appendice A Classification détaillée des champs d'en-tête

La compression d'en-tête est possible grâce au fait que la plupart des champs d'en-tête ne varient pas au hasard d'un paquet à l'autre. Beaucoup de champs exhibent un comportement statique ou changent d'une façon plus ou moins prévisible. Pour la conception d'un schéma de compression d'en-tête, il est d'une importance fondamentale de comprendre le comportement en détails des champs.

Dans cet appendice, tous les champs d'en-tête IP, UDP et RTP sont classés et analysés en deux étapes. D'abord, on a une classification générale en A.1 où les champs sont classés sur la base de connaissances et hypothèses stables. La classification générale ne prend pas en compte les caractéristiques des changements des champs changeants parce que celles-ci vont varier plus ou moins selon la mise en œuvre et selon l'application utilisée. Une analyse moins stable mais plus détaillée des caractéristiques des changement est ensuite faite en A.2. Finalement, A.3 résume cet appendice avec des conclusions sur la façon dont les divers champs d'en-tête devraient être traités par le schéma de compression d'en-tête pour optimiser la compression et les fonctionnalités.

A.1 Classification générale

Au niveau général, les champs d'en-tête se répartissent en cinq classes :

INFERRED : Ces champs contiennent des valeurs qui peuvent être déduites d'autres valeurs, par exemple la taille de la trame qui porte le paquet, et n'ont donc pas à être traitées du tout par le schéma de compression.

STATIC : Ces champs sont supposés être constants tout au long de la durée de vie du flux de paquets. Les informations statiques doivent d'une certaine façon être communiquées une fois.

STATIC-DEF : Les champs STATIC dont les valeurs définissent un flux de paquets. Ils sont en général traités comme STATIC.

STATIC-KNOWN : Ces champs STATIC sont supposés avoir des valeurs bien connues et n'ont donc pas besoin d'être communiqués du tout.

CHANGING : Ces champs sont supposés varier d'une certaine manière : au hasard, au sein d'un ensemble ou gamme de valeurs limitées, ou d'une autre manière.

Dans cette section, chacun des champs d'en-tête IP, UDP et RTP est affecté à une de ces classes. Pour tous les champs excepté ceux classés comme CHANGING, le motif du classement est aussi indiqué. Au paragraphe A.2, les champs CHANGING sont examinés plus en détails et classés sur la base de leur comportement de changement attendu.

A.1.1 Champs d'en-tête IPv6

Champ	Taille (bits)	Classe
Version	4	STATIC
Classe de trafic	8	CHANGING
Étiquette de flux	20	STATIC-DEF
Longueur de charge utile	16	INFERRED
Prochain en-tête	8	STATIC
Limite de bonds	8	CHANGING
Adresse de source	128	STATIC-DEF
Adresse de destination	128	STATIC-DEF

Version
Le champ Version déclare quelle version IP est utilisée. Les paquets avec des valeurs différentes dans ce champ doivent être traités par des piles IP différentes. Tous les paquets d'un flux de paquets doivent donc être de la même version IP. En conséquence, le champ est classé STATIC.

Étiquette de flux
Ce champ peut être utilisé pour identifier les paquets qui appartiennent à un flux de paquets spécifique. Si il n'est pas utilisé, la valeur devrait être réglée à zéro. Autrement, tous les paquets qui appartiennent au même flux doivent avoir la même valeur dans ce champ, qui est un de ceux qui définissent le flux. Le champ est donc classé comme STATIC-DEF.

Longueur de charge utile
Les informations sur la longueur du paquet (et, par conséquent, sur la longueur de la charge utile) sont supposées être fournies par la couche de liaison. Le champ est donc classé comme INFERRED.

Prochain en-tête
Ce champ va normalement avoir la même valeur dans tous les paquets d'un flux. Il code le type de l'en-tête suivant. Pendant la durée de vie du flux, le champ ne va changer sa valeur que lorsque des en-têtes d'extension sont parfois présents et parfois non. Le champ est donc classé comme STATIC.

Adresses de source et de destination
Ces champs font partie de la définition d'un flux et doivent donc être constants pour tous les paquets du flux. Les champs sont donc classés comme STATIC-DEF.

Taille totale des champs dans chaque classe :

Classe	**Taille (octets)**
INFERRED	2
STATIC	1,5
STATIC-DEF	34,5
CHANGING	2

A.1.2 Champs d'en-tête IPv4

Champ	**Taille (bits)**	**Classe**
Version	4	STATIC
Longueur d'en-tête	4	STATIC-KNOWN
Type de service	8	CHANGING
Longueur de paquet	16	INFERRED
Identification	16	CHANGING
Fanion Réservé	1	STATIC-KNOWN
Fanion Ne pas fragmenter	1	STATIC
Fanion Fragments à suivre	1	STATIC-KNOWN
Décalage de fragment	13	STATIC-KNOWN
Durée de vie	8	CHANGING
Protocole	8	STATIC
Somme de contrôle d'en-tête	16	INFERRED
source	32	STATIC-DEF
Adresse de destination	32	STATIC-DEF

Version
Le champ version établit quelle version IP est utilisée. Les paquets avec des valeurs différentes dans ce champ doivent être traités par des piles IP différentes. Tous les paquets d'un flux doivent donc être de la même version IP. En conséquence, le champ est classé comme STATIC.

Longueur d'en-tête
Tant qu'aucune option n'est présente dans l'en-tête IP, la longueur de l'en-tête est constante et bien connue. Si il y a des options, les champs seront STATIC, mais on suppose ici qu'il n'y a pas d'options. Le champ est donc classé comme STATIC-KNOWN.

Longueur de paquet
Informations sur la longueur de paquet dont il est attendu qu'elle soit fournie par la couche de liaison. le champ est donc classé comme INFERRED.

Fanions
Le fanion Réservé doit être réglé à zéro et est donc classé comme STATIC-KNOWN. Le fanion Ne pas fragmenter (DF) sera constant pour tous les paquets d'un flux et est donc classé comme STATIC. Finalement, le fanion Fragments à suivre (MF) est supposé être à zéro parce que la fragmentation N'EST PAS attendue, due à la petite taille de paquet espérée. Le fanion Fragments à suivre est donc classé comme STATIC-KNOWN.

Décalage de fragment

Dans l'hypothèse qu'il ne survient pas de fragmentation, le décalage de fragment est toujours à zéro. Le champ est donc classé comme STATIC-KNOWN.

Protocole
Ce champ va normalement avoir la même valeur dans tous les paquets d'un flux. Il code le type de l'en-tête suivant. C'est seulement lorsque des en-têtes d'extension sont parfois présents et parfois pas, que le champ va changer sa valeur durant la durée de vie d'un flux. Le champ est donc classé comme STATIC.

Somme de contrôle d'en-tête
La somme de contrôle d'en-tête protège les bonds individuels contre le traitement d'en-têtes corrompus. Quand presque toutes les informations d'en-tête IP sont compressées, il ne sert à rien d'avoir cette somme de contrôle supplémentaire ; elle peut plutôt être régénérée du côté du décompresseur. Le champ est donc classé comme INFERRED.

Adresses de source et de destination
Ces champs font partie de la définition d'un flux et doivent donc être constants pour tous les paquets du flux. Les champs sont donc classés comme STATIC-DEF.

Taille totale des champs dans chaque classe :

Classe	**Taille (octets)**
INFERRED	4
STATIC	1 octet + 5 bits
STATIC-DEF	8
STATIC-KNOWN	2 octet + 3 bits
CHANGING	4

A.1.3 Champs d'en-tête UDP

Champ	**Taille (bits)**	**Classe**
Accès de source	16	STATIC-DEF
Accès de destination	16	STATIC-DEF
Longueur	16	INFERRED
Somme de contrôle	16	CHANGING

Accès de source et de destination
Ces champs font partie de la définition d'un flux et doivent donc être constants pour tous les paquets du flux. Les champs sont donc classés comme STATIC-DEF.

Longueur
Ce champ est redondant et est donc classé comme INFERRED.

Taille totale des champs dans chaque classe :

Classe	**Taille (octets)**
INFERRED	2
STATIC-DEF	4
CHANGING	2

A.1.4 Champs d'en-tête RTP

Champ	**Taille (bits)**	**Classe**
Version	2	STATIC-KNOWN
Bourrage	1	STATIC
Extension	1	STATIC
Compteur CSRC	4	CHANGING
Marqueur	1	CHANGING
Type de charge utile	7	CHANGING
Numéro de séquence	16	CHANGING
Horodatage	32	CHANGING
SSRC	32	STATIC-DEF

CSRC	0(-480)	CHANGING

Version
Il existe une seule version fonctionnelle de RTP, à savoir la version 2. Le champ est donc classé STATIC-KNOWN.

Bourrage
L'utilisation de ce champ dépend de l'application, mais quand le bourrage de la charge utile est utilisé, il sera vraisemblablement présent dans tous les paquets. Les champ est donc classé comme STATIC.

Extension
Si des extensions RTP sont utilisées par l'application, ces extensions seront vraisemblablement présentes dans tous les paquets (mais l'usage d'extensions est très peu fréquent). Cependant, pour des raisons de sûreté, ce champ est classé comme STATIC et non STATIC-KNOWN.

SSRC
Ce champ fait partie de la définition d'un flux et doit donc être constant pour tous les paquets du flux. Le champ est donc classé comme STATIC-DEF.

Taille totale des champs dans chaque classe :

Classe	**Taille (octets) |**
STATIC	2 bit
STATIC-DEF	4
STATIC-KNOWN	2 bits
CHANGING	7,5(-67,5)

A.1.5 Résumé pour IP/UDP/RTP

En résumant cela pour IP/UDP/RTP on obtient

Classe\ IP ver	**IPv6 (octets)**	**IPv4 (octets)**
INFERRED	4	6
STATIC	1 octet + 6 bits	1 octet + 7 bits
STATIC-DEF	42,5	16
STATIC-KNOWN	2 bits	2 octet + 5 bits
CHANGING	11,5(-71,5)	13,5(-73,5)
Total	60(-120)	40(-100)

A.2 Analyse des schémas de changement de champs d'en-tête

Pour concevoir des mécanismes convenables pour une compression efficace de tous les champs d'en-tête, leur schéma de changement doit être analysé. Pour cette raison, une classification étendue est faite sur la base de la classification générale du paragraphe A.1, en considérant les champs qui ont été étiquetés CHANGING dans cette classification. Des applications différentes vont utiliser les champs de façons différentes, ce qui peut affecter leur comportement. Pour les champs dont le comportement est variable, le comportement typique pour l'audio conversationnel et la vidéo sera exposé.

Les champs CHANGING sont répartis en cinq sous classes différentes :

STATIC : Ce sont des champs qui ont été classés comme CHANGING d'une façon générale, mais sont classés ici comme STATIC du fait de certaines hypothèses supplémentaires .

SEMISTATIC : Ces champs sont STATIC la plupart du temps. Cependant, occasionnellement, la valeur change mais revient à sa valeur originale après un nombre connu de paquets.

RARELY-CHANGING (RC) : Ce sont des champs qui occasionnellement changent leur valeur et gardent leur nouvelle valeur.

ALTERNATING : Ces champs alternent entre un petit nombre de différentes valeurs.

IRREGULAR : Ceux-là, finalement, sont les champs pour lesquels aucun schéma de changement ne peut être identifié.

Pour étendre les possibilités de classification sans accroître la complexité, la classification peut être faite selon les valeurs du champ et/ou selon les valeurs des deltas pour le champ.

Lorsque la classification est faite, d'autres détails sont aussi pris en compte en ce qui concerne de possibles connaissances supplémentaires sur les valeurs de champ et/ou les deltas du champ, selon la classification. Pour les champs classés comme STATIC ou SEMISTATIC, il se pourrait que la valeur du champ ne soit pas seulement STATIC mais aussi bien CONNUE a priori (deux états pour les champs SEMISTATIC). Pour les champs qui ont un comportement de changement non irrégulier, il pourrait être connu que les changements sont habituellement dans une gamme LIMITÉE comparée au changement maximal pour le champ. Pour les autres champs, les valeurs sont complètement INCONNUES.

Le Tableau A.1 classe tous les champs CHANGING sur la base de leur schéma de changement attendu, en particulier pour l'audio conversationnel et la vidéo.

Champ		**Valeur/Delta**	**Classe**	**Connaissance**
Identifiant IPv4 :	Séquentiel	Delta	STATIC	CONNU
	Saut de séquence	Delta	RC	LIMITÉ
	Aléatoire	Valeur	IRREGULAR	INCONNU
TOS IP / Classe de trafic		Valeur	RC	INCONNU
TTL IP/ Limite de bonds		Valeur	ALTERNATING	LIMITÉ
Somme de contrôle UDP	Désactivé	Valeur	STATIC	CONNU
	activé	Valeur	IRREGULAR	INCONNU
Compte CSRC :	Non mixte	Valeur	STATIC	CONNU
	Mixte	Valeur	RC	LIMITÉ
Marqueur RTP		Valeur	SEMISTATIC	CONNU/CONNU
Type de charge utile RTP		Valeur	RC	INCONNU
Numéro de séquence RTP		Delta	STATIC	CONNU
Horodatage RTP		Delta	RC	LIMITÉ
RTP CSRC List:	Non mixte	-	-	-
	Mixte	Valeur	RC	INCONNU

Tableau A.1 : Classification des champs d'en-tête CHANGING

Les paragraphes qui suivent exposent en détails les divers champs d'en-tête. Noter que le tableau A.1 et l'exposé qui suit ne prennent pas en considération les changements causés par la perte ou le réarrangement avant le point de compression.

A.2.1 Identification IPv4

Le champ Identification (IP ID) de l'en-tête IPv4 est là pour identifier quels fragments constituent un datagramme lors du réassemblage de datagrammes fragmentés. La spécification IPv4 ne spécifie pas exactement comment les valeurs sont allouées à ce champ, mais seulement que chaque paquet devrait avoir un identifiant IP qui soit unique pour la paire source-destination et protocole pour la durée pendant laquelle le datagramme (ou n'importe lequel de ses fragments) pourrait être en vie dans le réseau. Cela signifie que l'allocation de valeurs d'identifiants IP peut être faite de diverses façons, que nous avons séparées en trois classes.

Saut séquentiel
C'est la politique d'allocation la plus courante dans les piles IP d'aujourd'hui. Un seul compteur d'identifiants IP est utilisé pour tous les flux de paquets. Lorsque l'envoyeur gère plus d'un flux de paquets simultanément, l'identifiant IP peut s'accroître de plus d'un entre les paquets d'un flux. Les valeurs d'identifiant IP seront beaucoup plus prévisibles et exigeront moins de bits à transférer que des valeurs aléatoires, et l'incrément de paquet à paquet (déterminé par le nombre de flux de paquets sortants actifs et les fréquences d'envoi) sera normalement limité.

Aléatoire
Certaines piles IP allouent des valeurs d'identifiant IP en utilisant un générateur de nombres pseudo-aléatoires. Il n'y a donc pas de corrélation entre les valeurs d'identifiant des datagrammes successifs. Donc il n'y a pas de moyen de prédire la valeur de l'identifiant IP du prochain datagramme. Pour les besoins de la compression d'en-tête, cela signifie que le champ IP ID doit être envoyé non compressé avec chaque datagramme, résultant en deux octets d'en-tête supplémentaires. Les piles IP dans les terminaux cellulaires NE DEVRAIENT PAS utiliser cette politique d'allocation d'identifiants IP.

Séquentiel
Cette politique d'allocation tient un compteur distinct pour chaque flux de paquets sortant et donc, la valeur de l'identifiant IP va s'incrémenter de un pour chaque paquet du flux, sauf au retour à zéro. Donc, la valeur du delta du champ est constante et bien connue a priori. Lorsque RTP est utilisé par dessus UDP et IP, la valeur d'IP ID suit le numéro de séquence RTP. Cette politique d'allocation est la plus désirable pour les besoins de la compression d'en-tête. Cependant, son usage n'est pas aussi courant qu'il devrait peut-être l'être. La raison peut en être qu'elle ne peut être réalisée que lorsque UDP et IP sont mis en œuvre ensemble de sorte que UDP, qui sépare les flux de paquets par les champs d'identification Accès, peut faire qu'IP utilise des compteurs d'identifiants séparés pour chaque flux de paquet.

Afin d'éviter de violer la [RFC0791], les paquets qui partagent la même paire d'Adresse IP et Numéro de protocole IP ne peuvent pas utiliser les mêmes valeurs d'IP ID. Donc, les mises en œuvre de politiques séquentielles doivent rendre les espaces de numéro d'identifiant disjoints pour les flux de paquets du même protocole IP qui vont entre la même paire de nœuds. Cela peut être fait d'un certain nombre de façons, dont toutes introduisent des états occasionnels, et donc rendent la politique moins que parfaitement séquentielle. Pour les besoins de la compression d'en-tête des sauts moins fréquents sont préférés.

On devrait noter que l'identifiant est un mécanisme IPv4 et n'est donc pas un problème pour IPv6. Pour IPv4, l'identifiant pourrait être traité de trois façons différentes. D'abord, on a la solution inefficace mais fiable où les champs d'identifiant sont envoyés tel quels dans tous les paquets, augmentant les en-têtes compressés de deux octets. C'est la meilleure façon de traiter le champ ID si l'envoyeur utilise l'allocation aléatoire du champ ID. Ensuite, il peut y avoir des solutions avec des mécanismes plus flexibles exigeant moins de bits pour le traitement de l'identifiant pour autant que le saut séquentiel d'allocation soit utilisé. De telles solutions vont probablement exiger encore plus de bits si l'allocation aléatoire est utilisée par l'envoyeur. Des connaissances sur la politique d'allocation de l'envoyeur pourraient donc être utiles lors du choix entre les deux solutions ci-dessus. Finalement, même pour IPv4, la compression d'en-tête pourrait être conçue sans aucune information supplémentaire pour le champ ID inclus dans les en-têtes compressés. Pour utiliser de tels schémas, il faut savoir quelle politique d'allocation est utilisée pour le champ ID par l'envoyeur. Cela peut n'être pas possible à savoir, ce qui implique que l'applicabilité de telles solutions est très incertaine. Cependant, les concepteurs de piles IPv4 pour les terminaux cellulaires DEVRAIENT utiliser une politique d'allocation proche du séquentiel.

A.2.2 Classe de trafic / Type de service IP

Le champ Classe de trafic (IPv6) ou Type de service (IPv4) est supposé être constant durant la vie d'un flux de paquets, ou changer relativement rarement.

A.2.3 Limite de bond / Durée de vie IP

Le champ Limite de bonds (IPv6) ou Durée de vie (IPv4) est supposé être constant durant la vie d'un flux de paquets ou alterner entre un nombre limité de valeurs liées à des changements de chemin.

A.2.4 Somme de contrôle UDP

La somme de contrôle UDP est facultative. Si elle est désactivée, sa valeur est zéro de façon constante et pourrait être compressée. Si elle est activée, sa valeur dépend de la charge utile, ce qui pour les besoins de la compression est équivalent à un changement aléatoire à chaque paquet.

A.2.5 Compteur CSRC RTP

C'est un compteur qui indique le nombre d'éléments de CSRC présents dans la liste de CSRC. Ce nombre est supposé être presque constant de paquet à paquet et changer d'une petite quantité. Tant qu'un mixeur RTP n'est pas utilisé, la valeur de ce champ est zéro.

A.2.6 Marqueur RTP

Pour l'audio, le bit marqueur ne devrait être établi que dans le premier paquet d'une salve de parole, tandis que pour la vidéo il devrait être établi dans le dernier paquet de chaque image. Cela signifie que dans les deux cas, le marqueur RTP est classé comme SEMISTATIC avec des valeurs bien connues pour les deux états.

A.2.7 Type de charge utile RTP

Les changements du type de charge utile RTP au sein d'un flux de paquets sont supposés être rares. Les applications pourraient s'adapter à l'encombrement en changeant de type de charge utile et/ou de tailles de trame, mais cela n'est pas supposé arriver fréquemment.

A.2.8 Numéro de séquence RTP

Le numéro de séquence RTP sera incrémenté de un pour chaque paquet envoyé.

A.2.9 Horodatage RTP

Dans le cas audio :
Tant qu'il n'y a pas de pauses dans le flux audio, l'horodatage RTP sera incrémenté d'un delta constant, correspondant au nombre d'échantillons dans la trame de parole. Il va donc le plus souvent suivre le numéro de séquence RTP. Lorsque il y a eu une période de silence et que commence une nouvelle salve de paroles, l'horodatage va grimper en proportion de la longueur de la période de silence. Cependant, l'incrément va probablement être dans une gamme relativement limitée.

Dans le cas de la vidéo :
Entre deux paquets consécutifs, l'horodatage va être soit inchangé, soit augmenter d'un multiple d'une valeur fixe correspondant à la fréquence d'horloge de l'image. L'horodatage peut aussi diminuer d'un multiple de la valeur fixée si des images B sont utilisées. L'intervalle de delta, exprimé comme un multiple de la fréquence d'horloge de l'image, est dans la plupart des cas très limité.

A.2.10 Sources contributives (CSRC) RTP

Les participants à une session, qui sont identifiés par les champs de CSRC, sont supposés être presque toujours les mêmes d'un paquet à l'autre avec relativement peu d'ajouts et de retraits. Tant que des mixeurs RTP ne sont pas utilisés, aucun champ de CSRC n'est présent .

A.3 Stratégies de compression d'en-tête

Ce paragraphe développe ce qui a été fait dans les paragraphes précédents. Sur la base des classifications, on donne des recommandations sur la façon de traiter les divers champs dans le processus de compression d'en-tête. Sept différentes actions sont possibles ; elles sont énumérées avec les champs auxquels chaque action s'applique.

A.3.1 Ne pas envoyer du tout

Les champs qui ont des valeurs bien connues à priori n'ont pas à être envoyés du tout. Ce sont :
- Longueur de charge utile IPv6
- Longueur d'en-tête IPv4
- Fanion Réservé IPv4
- Fanion Dernier fragment IPv4
- Décalage de fragment IPv4
- Somme de contrôle UDP (si elle est désactivée)
- Version RTP

A.3.2 Transmission initiale seulement

Les champs qui sont constants tout au long de la durée de vie du flux de paquets doivent n'être transmis et correctement livrés au décompresseur qu'une seule fois. Ce sont :
- Version IP
- Adresse de source IP
- Adresse de destination IP
- Étiquette de flux IPv6
- Fanion Peut fragmenter IPv4
- Accès de source UDP
- Accès de destination UDP
- Fanion Bourrage RTP

- Fanion Extension RTP
- SSRC RTP

A.3.3 Transmission initiale, mais prêt à mettre à jour

Les champs qui ne changent qu'occasionnellement doivent être transmis initialement mais il doit aussi y avoir un moyen pour mettre à jour ces champs avec de nouvelles valeurs si ils changent. Ces champs sont :
- Prochain en-tête IPv6
- Classe de trafic IPv6
- Limite de bonds IPv6
- Protocole IPv4
- Type de service (TOS) IPv4
- Durée de vie IPv4 (TTL)
- Compteur CSRC RTP
- Type de charge utile RTP
- Liste de CSRC RTP

Comme les valeurs des champs Protocole IPv4 et Prochain en-tête IPv6 sont en effet reliés au type de l'en-tête suivant, ils méritent un traitement particulier lorsque des sous en-têtes sont insérés ou retirés.

A.3.4 Être prêt à mettre à jour ou envoyer en l'état fréquemment

Pour les champs qui sont normalement soit constants soit ont des valeurs déductibles de quelque autre champ, mais qui divergent fréquemment de ce comportement, il doit y avoir un moyen efficace de mettre à jour la valeur du champ ou de l'envoyer tel quel dans certains paquets. Ces champs sont :
- Identification IPv4 (si elle n'est pas allouée séquentiellement)
- Marqueur RTP
- Horodatage RTP

A.3.5 Robustesse continue garantie

Pour les champs qui se comportent comme un compteur avec un delta fixé pour TOUS les paquets, la seule exigence sur le codage de transmission est que les pertes de paquet entre le compresseur et le décompresseur doivent être tolérables. Si plusieurs de ces champs existent, tous peuvent être communiqués ensemble. De tels champs peuvent aussi être utilisés pour interpréter les valeurs pour les champs énumérés au paragraphe précédent. Les champs qui ont ce comportement de compteur sont :
- Identification IPv4 (si elle est allouée séquentiellement)
- Numéro de séquence RTP

A.3.6 Transmission en l'état dans tous les paquets

Les champs qui ont des valeurs complètement aléatoires pour chaque paquet doivent être incluses telles quelles dans tous les en-têtes compressés. Ces champs sont :
- Identification IPv4 (si elle est allouée au hasard)
- Somme de contrôle UDP (si elle est activée)

A.3.7 Établir le delta et être prêt à mettre à jour

Finalement, il y a un champ qui augmente normalement d'un delta fixe et est corrélé à un autre champ. Pour ce champ, il y aurait du sens à ce que ce delta fasse partie de l'état du contexte. Le delta doit alors être initié et mis à jour de la même façon que les champs énumérés en A.3.3. Le champ auquel cela s'applique est :
- Horodatage RTP

Déclaration complète de droits de reproduction

Remerciement

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